Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тамбовский государственный технический университет»

С. А. ВАСИЛЬЕВ, И. Л. КОРОБОВА

ОСНОВЫ ЦИФРОВОЙ СХЕМОТЕХНИКИ В ИНФОРМАЦИОННЫХ СИСТЕМАХ

Утверждено Учёным советом университета
в качестве учебного пособия для бакалавров, обучающихся
по направлению подготовки 09.03.01 «Информатика и вычислительная техника»,
и магистрантов, обучающихся по направлению подготовки
09.04.01 «Информатика и вычислительная техника» всех форм обучения

Учебное электронное издание

Тамбов
Издательский центр ФГБОУ ВО «ТГТУ»
2021

УДК 004.3’144
ББК 32.971.32-0
В19

Р е ц е н з е н т ы :

Кандидат технических наук,
доцент кафедры «Математическое моделирование и информационные технологии»
Института математики, физики и информационных технологий
ФГБОУ ВО «ТГУ имени Г. Р. Державина»
Д. С. Соловьев

Кандидат технических наук,
доцент кафедры «Информационные процессы и управление»
ФГБОУ ВО «ТГТУ»
И. А. Елизаров

Васильев, С. А.

В19 Основы цифровой схемотехники в информационных системах [Электронный ресурс] : учебное пособие / С. А. Васильев, И. Л. Коробова. – Тамбов : Издательский центр ФГБОУ ВО «ТГТУ», 2021. – 1 электрон. опт. диск (CD-ROM). – Системные требования : ПК не ниже класса Pentium II ; CD-ROM-дисковод ; 38,5 Mb ; RAM ; Windows 95/98/XP ; мышь. – Загл. с экрана.

ISBN 978-5-8265-2342-1

Рассматриваются основные сведения о цифровой схемотехнике. Представлена среда моделирования работы электрических схем Micro-CAP.

Предназначено для бакалавров, обучающихся по направлению подготовки 09.03.01 «Информатика и вычислительная техника», и магистрантов, обучающихся по направлению подготовки 09.04.01 «Информатика и вычислительная техника» всех форм обучения, а также может быть полезно при выполнении выпускной квалификационной работы.

УДК 004.3’144
ББК 32.971.32-0

ISBN 978-5-8265-2342-1

ВВЕДЕНИЕ

Современные реалии нашей жизни неразрывно связаны с использованием электронных модулей цифровой автоматики. Работа цифровых устройств, в отличие от аналоговых, отличается своей помехозащищенностью от внешней температуры, влажности, давления, напряжения питания и т.п. Кроме этого, цифровые устройства обладают однозначностью в процессе их функционирования, что позволяет строить системы управления с большой точностью.

Появление подобных устройств начинается с разработки принципиальных электрических схем, выполняющих задуманные разработчиками алгоритмы автоматизации. Но разработка и отладка электрических схем процесс творческий и требует больших временных затрат. И тут на помощь разработчику приходят специализированные в области схемотехники компьютерные CAD-программы. Одна из таких программ, завоевавшая большую популярность в сфере проектирования электрических схем, как аналоговых, так и цифровых, является программа – Micro-CAP (Microcomputer Circuit Analisys Programm) от фирмы Spectrum Software [2].

В данном учебном пособии представлены принципы функционирования основных элементов цифровой схемотехники, их условно-графические изображения и примеры использования типовых микросхем. Представлены исследования работы цифровых схем в среде программы Micro-CAP (V12).

1. НАЧАЛО ЦИФРОВОЙ СХЕМОТЕХНИКИ

Традиционное изучение элементов цифровых схем (пример микросхемы, рис. 1.1) предполагает их физическое наличие и возможность собирать из этих элементов электрические принципиальные схемы.

Микросхемы сами по себе не могут быть активизированы без внешнего источника стабилизированного питания (рис. 1.2).

Лабораторное подключение микросхем между собой требует специальных беспаячных макетниц (рис. 1.3) или монтажных плат с металлизированными контактами площадками, к которым необходимо припаивать микросхемы и проводники.

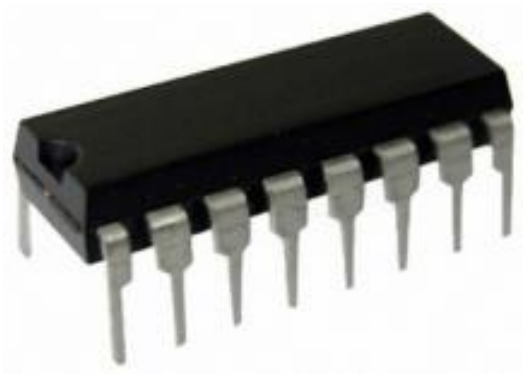


Рис. 1.1. Микросхема

Рис. 1.2. Лабораторный блок питания

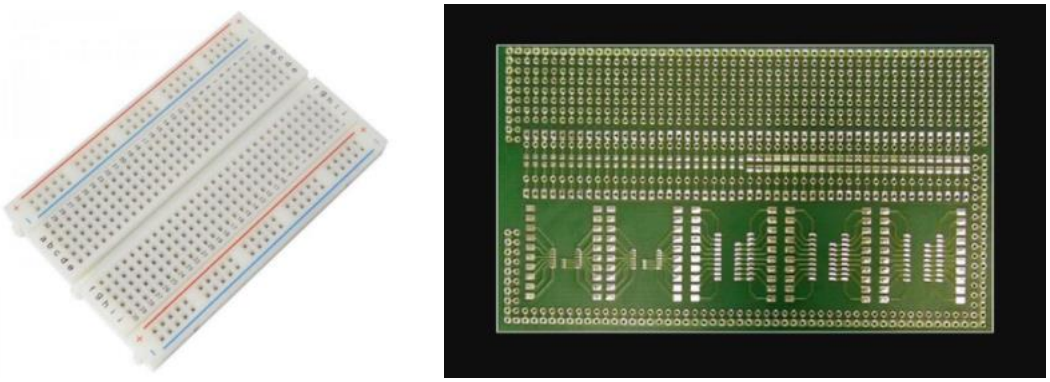


Рис. 1.3. Макетница беспаячная и паячная

Да и сам монтаж микросхем на плату требует особых навыков и соблюдения специальных технологий. Кроме этого, для исследования работы электрических схем требуется специализированный спектр измерительной аппаратуры. Как минимум, это мультиметр (тестер) для измерения конкретных значений напряжения (рис. 1.4) и осциллограф для визуализации форм сигналов и их расстановки во времени (рис. 1.5).

Подобное оборудование доступно студентам, как правило, исключительно в лабораторных условиях кафедры из-за аппаратной ёмкости и цены, что для домашнего исследования становится недоступно.

Но имеется эффективный способ заменить это оборудование обычной компьютерной программой – симулятор. Современные компьютерные технологии позволяют аналогичные исследования осуществлять в рамках программных симуляторов работы цифровых схем. Обычно, программные симуляторы входят составной частью в CAD-программы электрических схем. Например, Micro-Cap, OrCAD, PiCAD, PartSim и т.п.

Все программные симуляторы разделяются на два класса функционирования.

К первому классу относятся программы, выполняющие свои основные функции из ядра самой программы, расположенной на внешней памяти компьютера и под управлением ОС ЭВМ. И не обязательно такие программы должны работать под операционной системой типа Windows, MacOS или Linux. Распространенные носимые устройства под управлением ОС Android и iOS тоже могут исполнять некоторые симуляторы цифровых схем.

Рис. 1.4. Мультиметр (тестер)

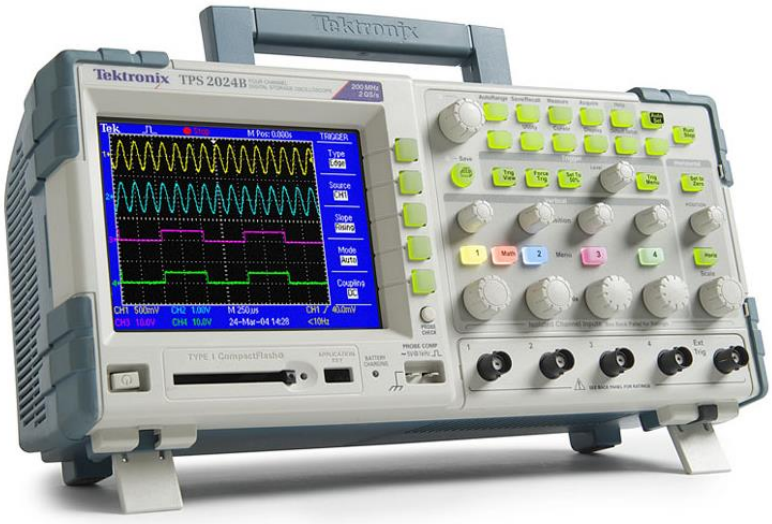


Рис. 1.5. Осциллограф

Последние удобны тем, что исследование работ в области схемотехники можно осуществлять на обычных смартфонах и мобильных планшетах. Наибольшую популярность для мобильных устройств завоевало приложение EveryCircuit (https://everycircuit.com/).

Ко второму классу программ-симуляторов относятся браузерные версии симуляторов. В таком варианте симулятор не хранится в памяти устройства, а размещен на удаленных серверах разработчиков. И для вызова симулятора достаточно набрать адрес симулятора в любом интернет-браузере. Например, http://falstad.com/circuit/circuitjs.html.

Следует отметить, что, как правило, все симуляторы с хорошим функционалом платные. И не стал исключением симулятор Micro-Cap. Но, симулятор Micro-Cap с 2019 года становится бесплатным, в связи с тем, что фирма-разработчик Spectrum Software прекращает дальнейшее развитие этого программного продукта. И если зайти на сайт разработчика http://www.spectrum-soft.com/download/download.shtm, то можно увидеть, что Micro-Cap 10, 11 и 12 теперь бесплатны и не требуют ключа. Следует отметить, что данный программный продукт разрабатывался фирмой Spectrum Software с 1982 года! И за 39 лет симулятор вышел на лидирующее место в списке программ-симуляторов, как аналоговых, так и цифровых схем. Данное учебное пособие будет опираться на Micro-Cap при исследовании цифровых схем.

Но если студент пожелает использовать другой симулятор для работы со схемами по дисциплине «Схемотехника», то это его право и никто не будет препятствовать этому. После того, как мы определились со средой моделирования наших электрических схем, пора и познакомиться, что такое цифровая схемотехника.

Из школьного курса физики Вы наверняка знаете, что электрические схемы предназначены для передачи через свои внутренние связи электрических сигналов для выполнения определённых функций. Различают аналоговые сигналы и дискретные (цифровые). Пример аналогового и цифрового сигнала можно увидеть на рис. 1.6.

У цифровых сигналов под условным значением «1» и «0» будем понимать некоторый диапазон напряжения U. Этот диапазон определяется своей технологией внутреннего строения микросхем (КМОП или ТТЛ). Например, для микросхем, где рабочее напряжение 5 В уровень «1» это диапазон 2…5 В, а «0» – 0…0,5 В. Но имеются микросхемы, где рабочее напряжение 3.3В, то в таких схемах уровень «1» это диапазон где-то 1,5…3,3 В, а «0» – 0…0,4 В. Бывают микросхемы и с напряжением питания 1,8, 1,5 и 1,2 В. Для них выделяется свой диапазон напряжений под значение логической единицы и нуля. В рамках нашей дисциплины мы будем оперировать понятиям высокий (1) или низкий (0) сигал.

Благодаря диапазонам напряжений, выделенных под значения логических уровней, цифровая техника менее восприимчива электрическим (электромагнитным) помехам, поступающим извне. Ведь результат любой помехи – это незапланированные дополнительные изменения напряжения в проводнике. А если этот скачек напряжения укладывается в диапазон логических уровней цифровых схем, то и сам цифровой сигнал не исказится (рис. 1.7). Обратите внимание, что форма сигнала немного изменилась, но логическая последовательность нулей и единиц осталась неизменной. Это и есть помехоустойчивость.

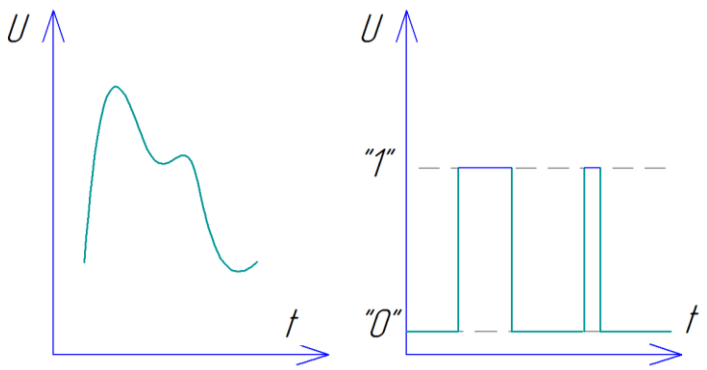


Рис. 1.6. Аналоговый (слева) и цифровой сигнал

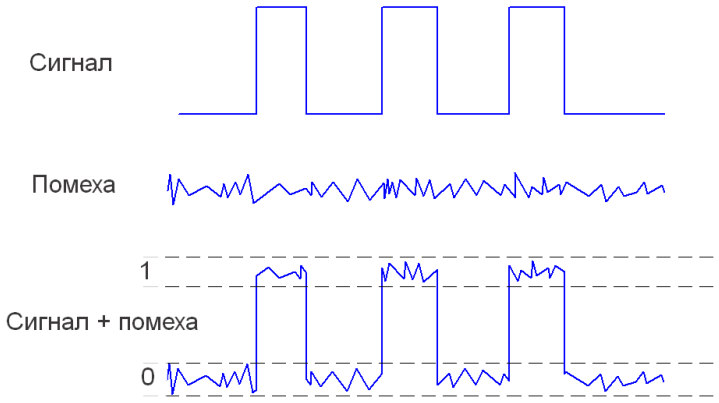


Рис. 1.7. Влияние помехи на цифровой сигнал

Но не следует безоговорочно отдавать предпочтение цифровым сигналам. Ведь цифровой сигнал – это всего два уровня 0 и 1, а аналоговый сигнал – это бесконечное количество уровней напряжений в любом диапазоне значений. И в один интервал времени аналоговым сигналом можно передать гораздо больше информации, чем цифровым. Но помеховосприимчивость аналоговых цепей сводит на нет это достоинство аналоговых сигналов.

Цифровые схемы принято формировать из логических микросхем. Логическая микросхема – это готовая сборка электрических схем (рис. 1.8). На рисунке справа представлена только схема первого верхнего элемента. Аналогичные схемы реализованы и для остальных трёх элементов 2И-НЕ.

Ушли в историю те времена, когда цифровые схемы представлялись в виде полупроводниковых элементов и их взаимных связей. Правда, на сегодняшний день этим вплотную занимаются разработчики самих микросхем. Нам же предоставляется функциональный цифровой блок, у которого имеются только входные и выходные проводники сигналов. Соединяя эти блоки между собой, мы можем формировать более сложные логические функции. Для документирования взаимосвязей компонентов электронных узлов используют графические схемы. Различают три типа схем:

- принципиальная схема;
- структурная схема;
- функциональная схема.

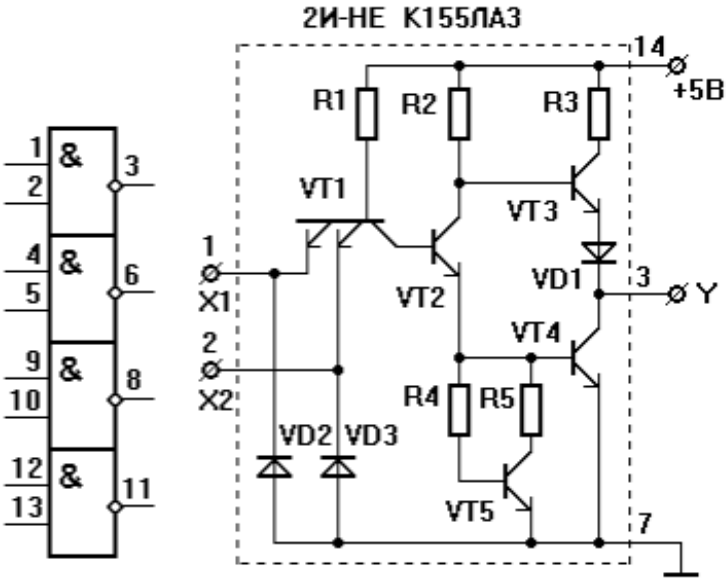


Рис. 1.8. Логическая микросхема 2И-НЕ и её схемное решение

Различаются они своим назначением и степенью детализации изображения компонентов электронного устройства.

Принципиальная схема – подробное описание всех электрических взаимосвязей компонентов устройства. Указываются абсолютно все электронные элементы схемы и их взаимосвязи. Расписываются название электронных компонентов, номера выводов микросхем, номинальные параметры, например, резисторов и конденсаторов. Принципиальные схемы изображаются по ГОСТ 2.702–2011 [1] и отклонятся от рекомендаций ГОСТа не рекомендуется.

Структурная схема – схема взаимодействия укрупненных модулей принципиальной схемы электронного изделия. Отображается общая структура электрической схемы устройства. Структурная схема даёт представление о работе электронного устройства и как взаимодействуют её части между собой. Представление структурных схем свободное, не нормированное. Главное, чтобы была понятна суть работы принципиальной схемы.

Функциональная схема предназначена для формирования знания о внутренней логике работы принципиальной схемы. Как и структурная схема функциональная не стандартизирована для изображения. Допускается слияние структурных модулей с фрагментами принципиальной схемы.

Каждый логический элемент микросхемы графически изображается в принципиальных электрических схемах в виде прямоугольника. С левой стороны всегда изображаются входные элементы микросхемы, а с правой – выходные. Цепи питания микросхем обычно не указываются на изображении, но если появляется необходимость, то для этого отводится верхняя часть и нижняя часть изображения. В верхней части указывается +Uпит., а в нижней части 0В (общий). Пример типового изображения логической микросхемы приведён на рис. 1.9.

Сверху изображения микросхем кроме питания могут располагаться выводы для управления работой внутренней схемы.

Для пояснения назначения входов и выходов микросхемы на её графическом изображении отводятся специальные поля, расположенные слева и справа.

Иногда специальное поле для выходов не отводится, например, как на рис. 1.11.

Функциональное назначение элемента цифровых микросхем указывают в верхней части основного поля УГО. На рисунке 1.10 это DC (дешифратор), а на рис. 1.11 – Т (Т-триггер).

Как уже было сказано ранее, цифровая техника работает с логическими сигналами. Принято, что логическая «1» это высокое напряжение, а «0» – низкое. В таком случае логика называется «положительная». Но иногда, в специальных схемах, например в работе системных шин и передаче данных на расстояние, для передачи логической «1» используют низкое напряжение, а для «0» используют высокое напряжение. Подобная логика называется «отрицательная».

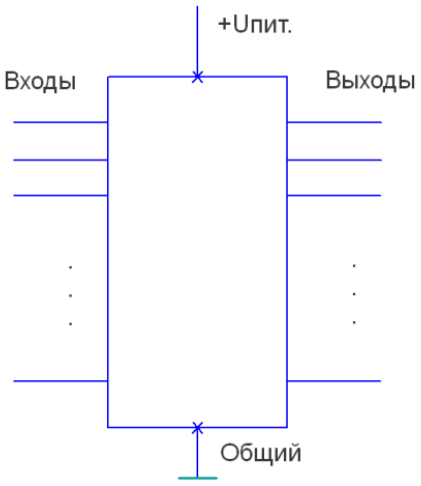


Рис. 1.9. Распределение входных и выходных сигналов на УГО микросхемы

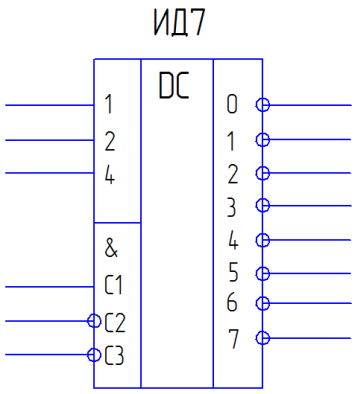


Рис. 1.10. УГО микросхемы–ИД7 (дешифратор)

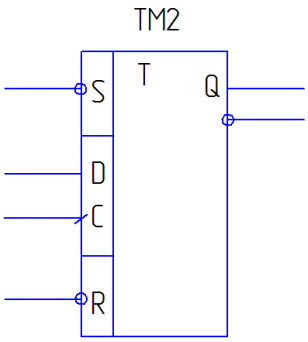


Рис. 1.11. УГО микросхемы–D-триггер

Теперь разберёмся, какие бывают сигналы в цифровых схемах.

Принято называть положительным цифровым сигналом, если активный уровень – высокое напряжение, а низкий уровень – низкое напряжение (рис. 1.12). Переход из низкого уровня в высокий называется фронтом сигнала или положительный фронт. А переход высокого уровня сигнала в низкий называется срезом сигнала или отрицательный фронт. Иногда эти фронты называются передний и задний, соответственно.

Кроме положительного сигнала в цифровой технике существует и отрицательный сигнал (рис. 1.13). У отрицательных сигналов активный уровень соответствует отсутствию сигнала (низкое напряжение), т.е. отсутствие сигнала соответствует логической «1», если пришёл сигнал, то это – логический «0».

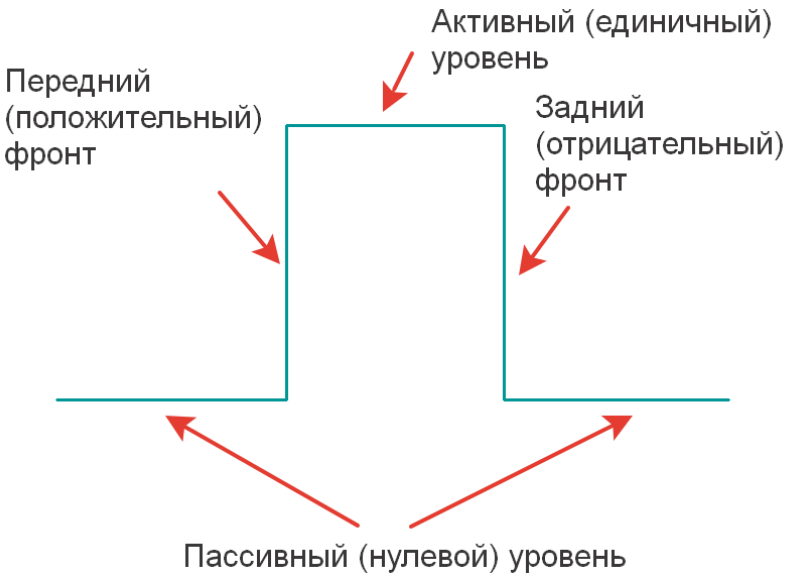


Рис. 1.12. Положительный сигнал

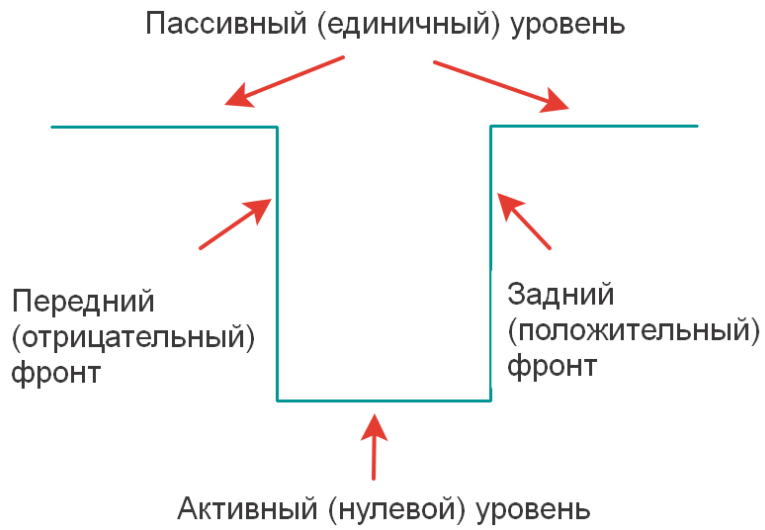


Рис. 1.13. Отрицательный сигнал

У микросхем различают прямой вход и инверсный. А также прямой выход и инверсный.

Прямой вход передаёт входной сигнал во внутреннюю схему микросхемы без изменения, а в инверсном входе сигнал инвертируется на противоположное значение. Например, если пришла логическая «1», то микросхема будет её воспринимать, как логический «0». И наоборот, если поступил сигнал со значением логический «0», то он будет инвертирован в логическую «1».

Прямой выход передаёт сигнал без изменения его логического состояния, а инверсный выход изменят значение логического сигнала на противоположное.

Для обозначения инверсного входа-выхода на УГО микросхемы принято изображать кружком, как на рис. 1.14.

Кроме прямых и инверсных входов, различают входы микросхем, которые влияют на её внутреннюю функцию по положительному или отрицательному фронту входного сигнала. На УГО для обозначения входа, реагирующего только на фронт сигнала, используют косую черту. Если её наклон влево, то такой вход реагируют на передний (положительный) фронт. Если наклон черты вправо – на отрицательный фронт (рис. 1.15).

Основное назначение микросхем – преобразование входных сигналов в выходные, согласно внутренней функции. Внутренняя функция задаётся в виде таблицы значений (таблицы истинности) или в виде временных диаграмм, где показываются изменения выходных сигналов во времени в зависимости от значений на входе (рис. 1.16).

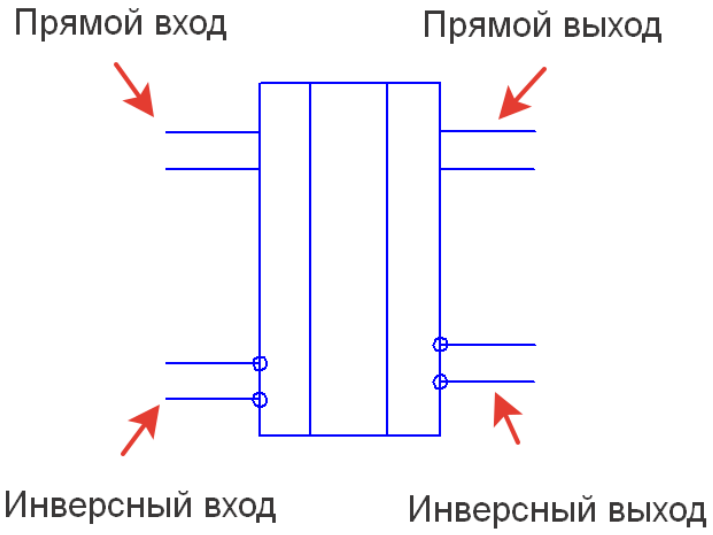


Рис. 1.14. Изображение прямого и инверсного входа-выхода

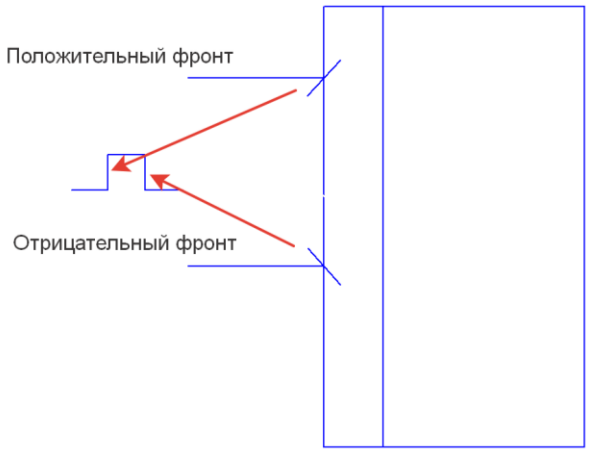
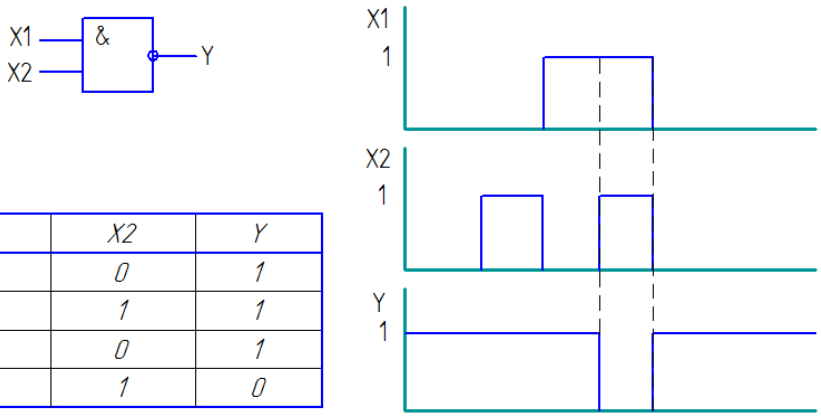


Рис. 1.15. Изображение входов микросхемы для переднего и заднего фронтов сигнала



X1	X2	Y
0	0	1
0	1	1
1	0	1
1	1	0

Рис. 1.16. Пример логического элемента 2И-НЕ
с двумя способами задания её внутренней функции

При разработке, анализе и описании цифровых схем используют различные представления моделей электронных компонентов. Практика показала, что достаточно трёх уровней представления цифровых устройств:

1. Логическая модель.
2. Временная модель.
3. Электрическая модель.

При логической модели отсутствуют такие понятия, как время прохождения сигнала, ёмкостные и индукционные составляющие, величины тока и напряжения и т.п. Здесь главенствуют Булева алгебра, законы и тождества логики. Обычно такие модели применяют при анализе низкочастотных цифровых схем.

Временная модель – это та же логическая модель, но с добавлением времени передачи сигнала от входа на выход. В программных симуляторах эта модель представлена, как минимальное (максимальное и среднее) время задержки сигнала, идущего с входа на выход, при переключении от 0 к 1 (1 к 0). Например, в программе Micro-CAP для элемента 2И-НЕ (U1) эти параметры представлены, как на рис. 1.17.

Здесь параметры TPHLMN, TPHLMX и TPHLY соответствуют задержке сигнала при переходе 1 к 0. TPHLMN – минимальное значение задержки сигнала. TPHLMX – максимальное значение задержки сигнала и TPHLY – среднее (рабочее) значение.

Здесь параметры TPLHMN, TPLHMX и TPLHY соответствуют задержке сигнала при переходе 0 к 1. TPLHMN – минимальное значение задержки сигнала. TPLHMX – максимальное значение задержки сигнала и TPLHY – среднее (рабочее) значение.

В Micro-CAP для перехода в логическую модель достаточно выбрать пункт, как на рис. 1.18. Обратите внимание, все параметры задержек приняли нулевые значения.

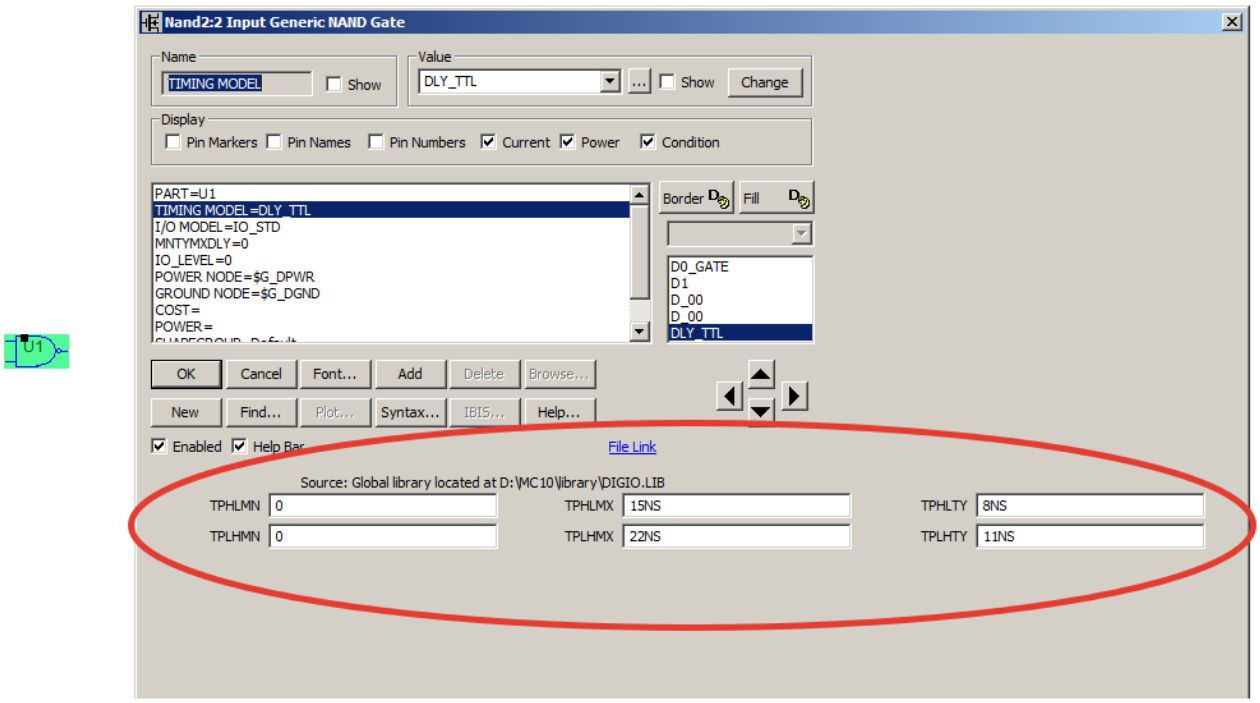


Рис. 1.17. Пример выбора временной модели логического элемента 2И-НЕ

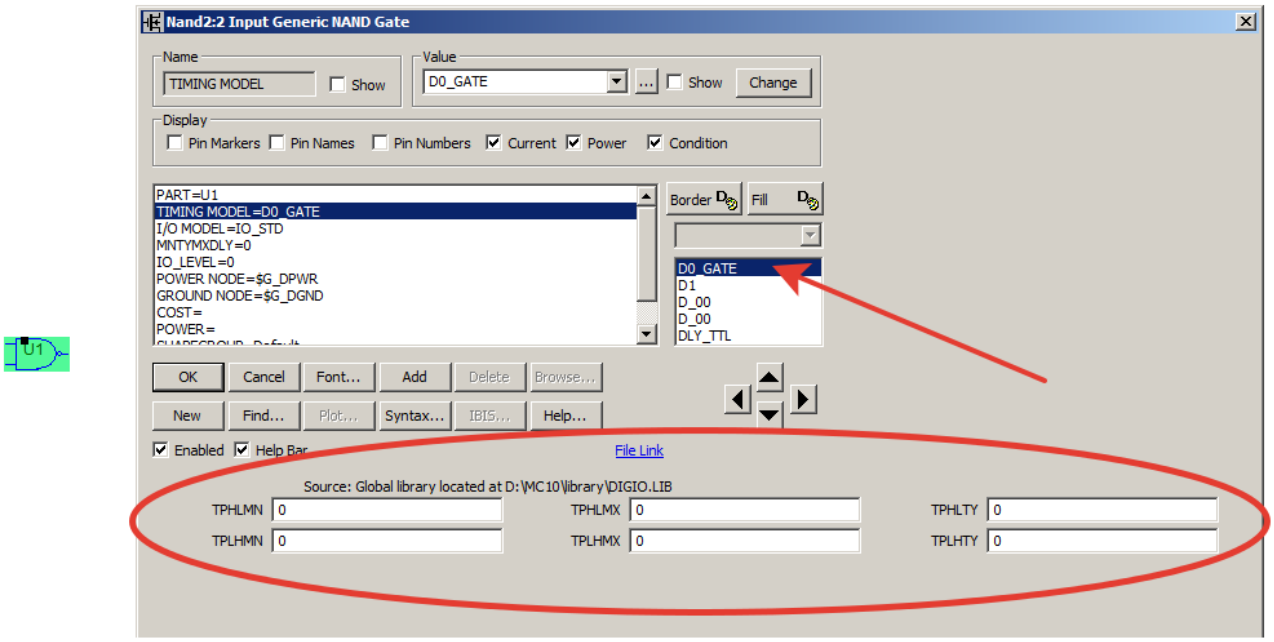


Рис. 1.18. Пример выбора логической модели цифрового элемента 2И-НЕ (U1)

Опыт проектирования цифровых схем показал, что третью модель представления цифровых элементов желательно использовать в 100% случаях проектирования. Благодаря таким моделям можно выявить все нюансы работы спроектированных цифровых схем при их практическом использовании, так как эта модель определяет эксплуатационные характеристики будущего цифрового изделия. Это только в двух первых моделях представления цифровых схем сигналы имеют прямоугольный вид, а на практике форма сигнала имеет некоторую кривизну. Это возникает из-за ёмкостных и индукционных свойств внутренней «начинки» микросхем. Да и сами линии передачи сигнала представляют в конечном счёте конденсатор из-за близкого расположения проводником. Такими нюансами и занимается электрическая модель. Электрическая модель обязательно применяется при моделировании схем, где про-

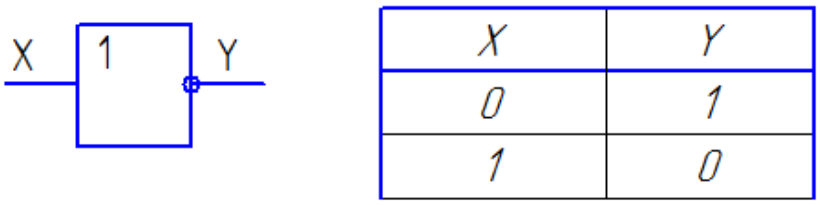
исходит стыковка модульных цифровых схем и особенно, если связь между модулями осуществляется на определённых расстояниях.

Давайте рассмотрим простейший цифровой элемент – инвертор в трёх моделях представления.

Условное графическое обозначение инвертора и таблица истинности представлены на рис. 1.19.

Анализируя графики работы логического элемента в трёх моделях представления (рис. 1.20), можно заметить, что в первой модели (логической) выходной сигнал изменяется без задержек сразу, как изменится входной сигнал. Вторая модель (временная) более реалистично передаёт работу логического элемента. После изменения входного сигнала выходной изменяется через временной интервал, согласно параметров задержек TPHLY и TPLHTY. Но на самом деле, работа логического элемента соответствует третьей модели представления – электрической. На третьем графике видны кривые изменения выходного сигнала. Но в рамках изучения работы цифровых схем для упрощения используют первые две модели представления.

Если схема будет работать на низких частотах, то и вторая модель представления будет излишняя при анализе работы цифровой схемы. Под низкими частотами будем понимать частоты сигналов, длительность которых во много раз больше чем длительность переходов TPHLY и TPLHY. Последнее утверждение попробуем разобрать на примере, приведённом на рис. 1.21.



X	Y
0	1
1	0

Рис. 1.19. Элемент инвертор (НЕ)

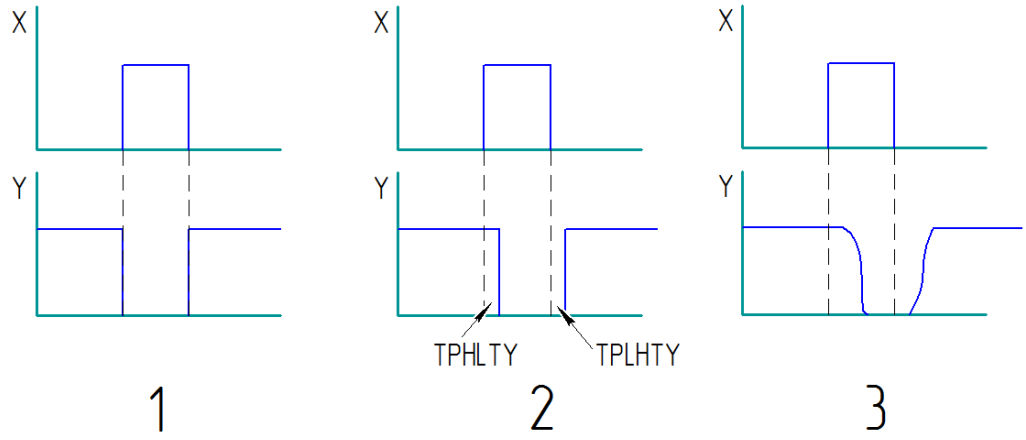


Рис. 1.20. Графики работы инвертора в трёх моделях представления

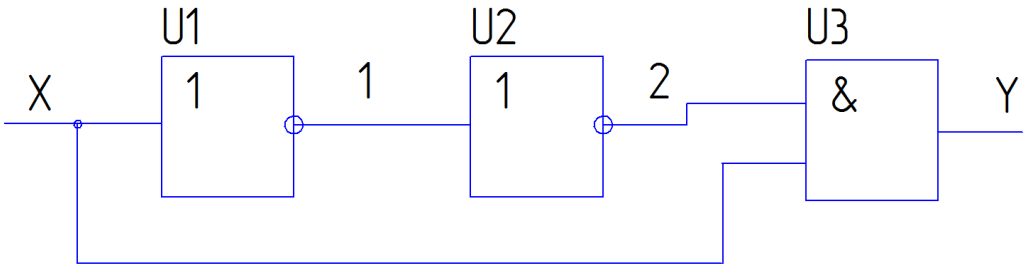


Рис. 1.21. Схема для сравнения логической модели с временной

Если рассматривать работу данной схемы при низкочастотных входных импульсах и использовать логические модели элементов U1-U3, то временные диаграммы прохождения сигналов соответствуют графикам А, приведённым на рис. 1.22. Красными стрелками указаны влияние фронтов сигналов на появление новых сформированных сигналов. Сигнал 1 появится по истечению времени срабатывания элемента U1. Сигнал 2 появится после того, как выполнит свою функцию элемент U2. Таким образом, сигналы 2 и X появятся на входе элемента U3 не одновременно. И функция «И» элемента U3 сформирует выходной сигнал Y, сдвинутый по времени, что соответствует повторению входного сигнала, но с запозданием и немного уменьшенной длительностью. Это соответствует логической модели представления цифровых элементов схемы. Последнее утверждение по поводу уменьшения длительности выходного сигнала (Y) попробуйте объяснить самостоятельно.

Но как только длительность входного сигнала становится приблизительно равной длительности срабатывания внутренних функций цифровых элементов (график В рис. 1.22), то на выходе Y сигнала мы вообще можем отсутствовать. Если посмотреть внимательно на график В, то можно заметить, что за время срабатывания логических элементов входные сигналы уже закончились из-за своей короткой длительности. Итоговые графики работы схемы с короткими входными сигналами не соответствует логической модели представления элементов U1 – U3. Таким образом, исследовать цифровые схемы с высокой частотой следования входных сигналов обязательно в режиме временных моделей. К этому вопросу мы подробно вернёмся при анализе рисков сбоя в комбинационных схемах.

Нужно помнить, что временная модель работы цифровых элементов не полностью передаёт реальную картину работы схемы. Не исключено, что исследование с третьим типом модели может выявить неучтённые сигналы в первых двух моделях.

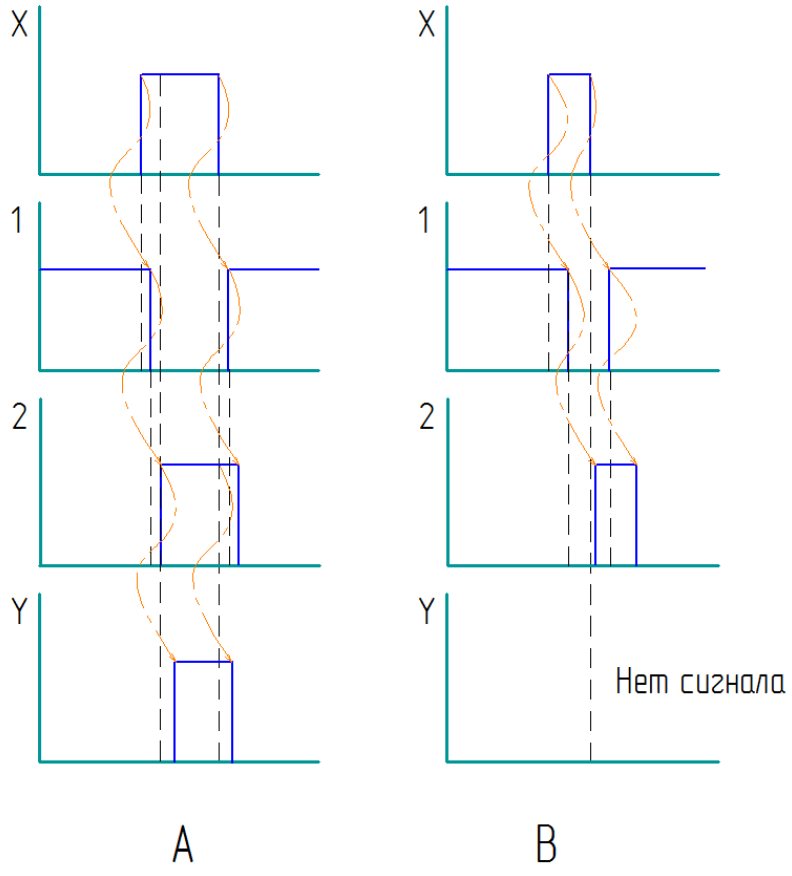


Рис. 1.22. Графики работы электрической схемы

2. ЭМУЛЯЦИЯ РАБОТЫ ЦИФРОВЫХ СХЕМ В MICRO-CAP

В рамках дисциплины «Схемотехника» будем проверять работоспособность проектируемых цифровых схем в среде программы Micro-CAP 12 для аналогового и цифрового моделирования электрических и электронных цепей. Программа разработана компанией Spectrum Software. С 04.07.2019 г. Компания Spectrum Software закрыта и Micro-CAP теперь бесплатен. Для загрузки программы можно воспользоваться ссылкой http://www.spectrum-soft.com/download/download.shtm

Рассмотрим основные этапы работы в среде Micro-CAP для симуляции работы цифровых схем.

При проектировании электрических схем в среде Micro-CAP первым этапом формируется электрическая схема в визуальном редакторе.

Для этого запускаем программу Micro-CAP. На экране компьютера появится интерфейсное окно, как на рис. 2.1.

Элементы цифровых схем выбираем в пункте **Component →Digital Primitives**, который располагается в верхней части меню Micro-CAP.

Например, для нашей проектируемой схемы требуется логический элемент 2И (функция «И» с двумя входами). На рисунке 2.2 показан путь для поиска данного элемента.

После выборки необходимого элемента на экране появится УГО этого элемента и окно с настройкой математической модели выбранного цифрового элемента. Если нас интересует исключительно логическая модель, то мы должны выбрать данную настройку согласно рис. 2.3.

Обратите внимание, что УГО логического элемента **2И** изображено вертикально (рис. 2.4). Для разворота изображения логического элемента в привычное положение, где входы справа, а выход слева, достаточно навести указатель на изображение элемента и нажать на правую кнопку мыши. Выбираем удобную ориентацию.

Должно получиться, как на рис. 2.5.

Номер элемента U1 можно легко переносить в любое удобное место. Достаточно зафиксировать его левой клавишей мышки и не отпуская кнопку можно переместить это название, например, как на рис. 2.6.

После того, как цифровые элементы будут размещены на рабочем поле редактора принципиальной схемы можно приступить к их соединению между собой. Для Micro-CAP мы будем использовать инструмент для наведения ортогональных связей (рис. 2.7). Быстрый вызов этого режима осуществляется одновременным нажатием кнопок Ctrl+W клавиатуры компьютера.

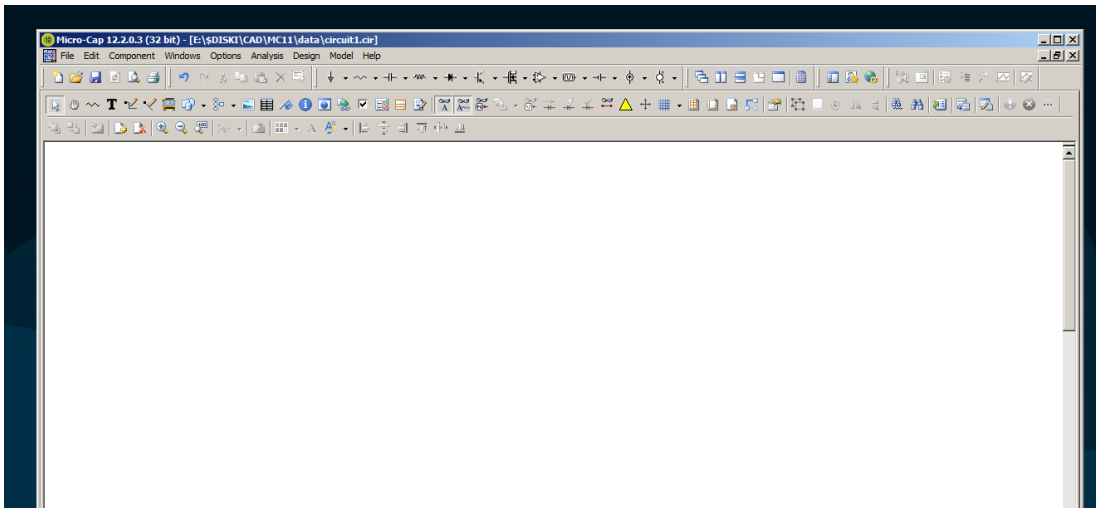


Рис. 2.1. Рабочее окно Micro-CAP 12

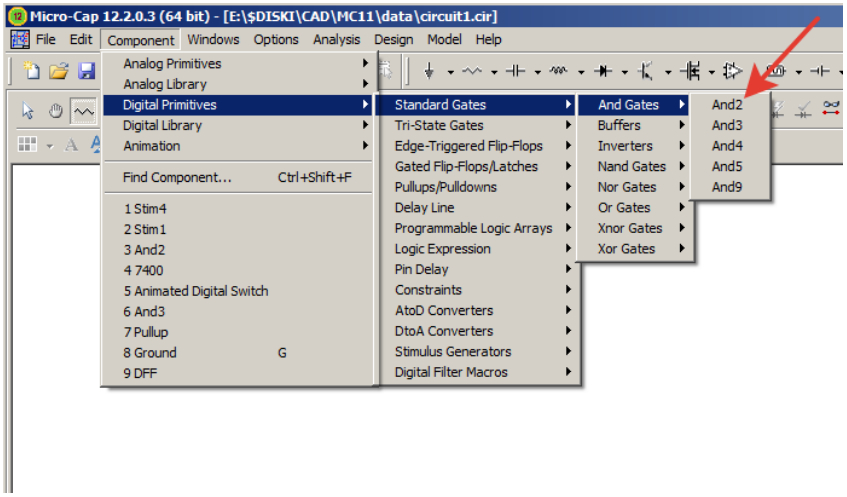


Рис. 2.2. Пример доступа к цифровым элементам в базе Micro-CAP

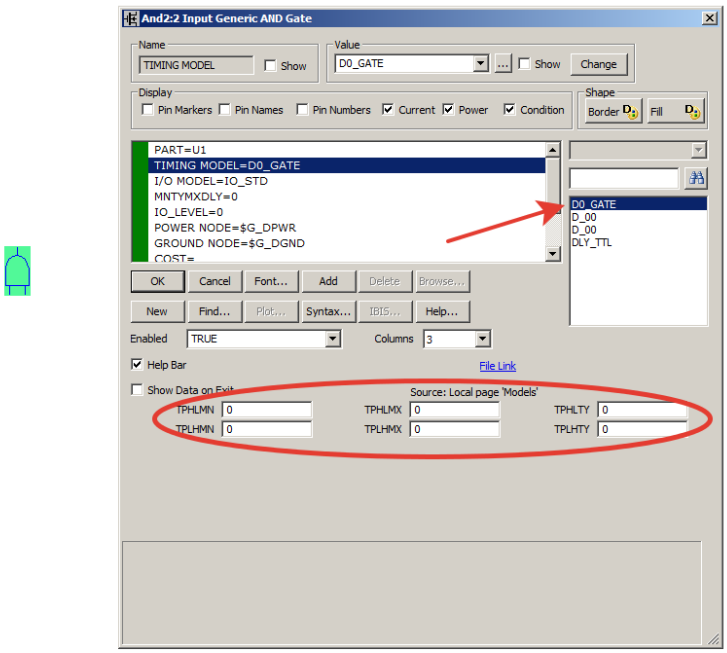


Рис. 2.3. Пример настройки логической модели цифрового элемента

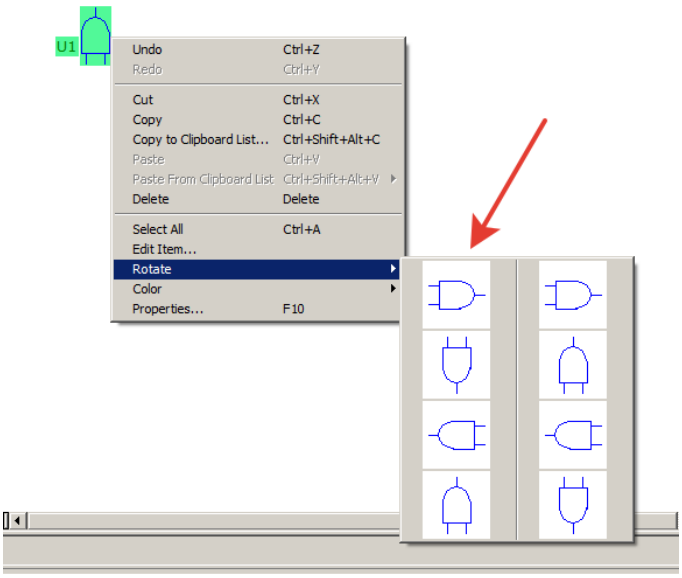


Рис. 2.4. Выбор ориентации УГО

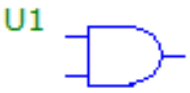


Рис. 2.5. Рабочая ориентация логического элемента

Рис. 2.6. Новое положение наименования U1 логического элемента

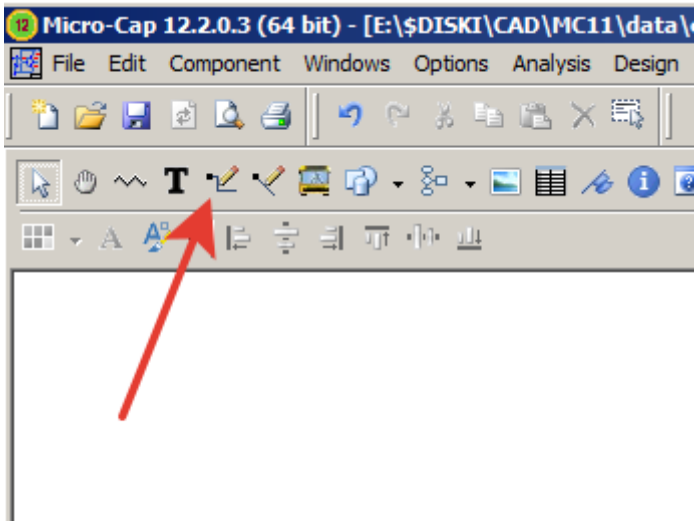


Рис. 2.7. Инструмент – ортогональные связи

Под ортогональностью будем понимать взаимно перпендикулярные связи.

Например, мы сформировали схему, как на рис. 2.8.

В этой схеме два входа и один выход. Для исследования работоспособности данной схемы мы должны определиться, какие точки данной схемы нас будут интересовать. В Micro-CAP существует инструмент, позволяющий именовать любые участки принципиальных схем (рис. 2.9).

Расставляем контрольные точки в схеме. В данном случае, это точки A, B, C и Y (рис. 2.10).

Рис. 2.8. Пример схемы

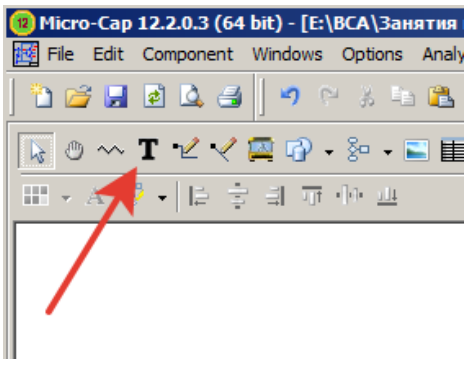


Рис. 2.9. Текстовый режим (Ctrl + T)

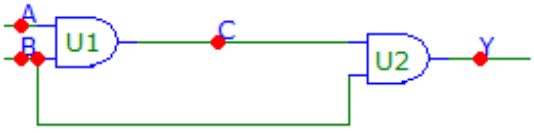


Рис. 2.10. Контрольные точки на схеме

Но для того, чтобы исследовать данную схему одних контрольных точек недостаточно. Нам необходимо подать на исследуемую схему контрольные логические сигналы для воссоздания всевозможных комбинаций логических сигналов на входе схемы. Так как в схеме два входа, то воспользуемся двухразрядным генератором логических сигналов **Stim2** (рис. 2.11).

Обратите внимание, что точка на УГО генератора обозначает выход старшего разряда (рис. 2.12).

В Micro-CAP существуют генераторы цифровых сигналов с 1, 2, 4, 8 и 16 выходами. Генератор программируется в схеме. Например, в данном случае, нам необходимо сформировать на выходах генератора **Stim2** (U3) все возможные коды двоичных значений в порядке их возрастания, т.е. 00, 01, 10 и 11. Для этого можно воспользоваться уже готовой программой моделирования выходных сигналов генератора, которая вызывается простым нажатием на кнопку **Count2** в окне (рис. 2.13).

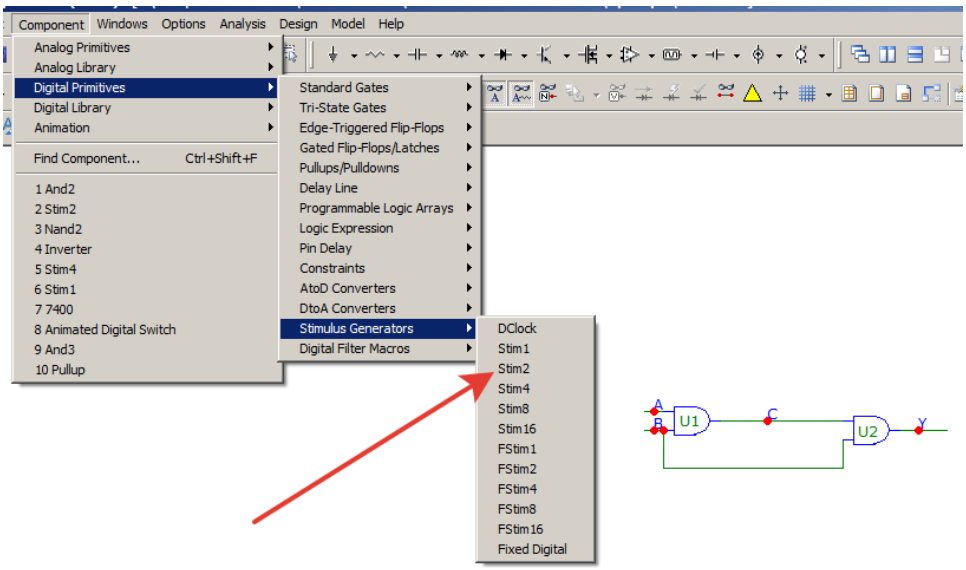


Рис. 2.11. Выбор генератора Stim2

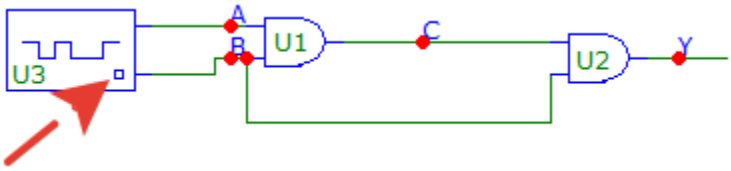


Рис. 2.12. Подключение двухразрядного генератора к исследуемой схеме

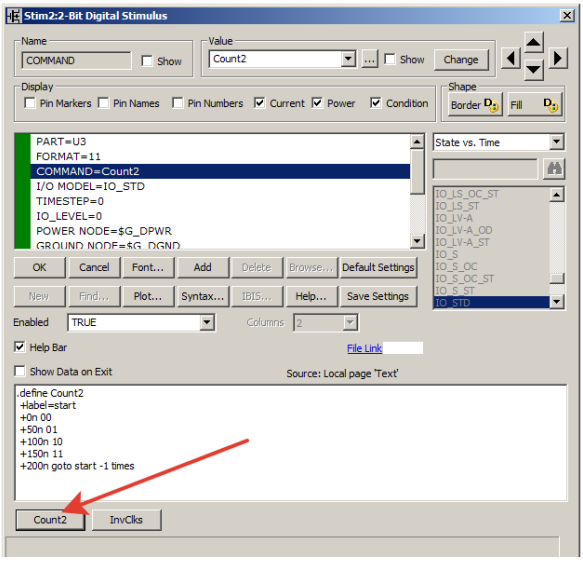


Рис. 2.13. Программирование генератора Stim2

Текст программы генератора **Stim2** в данном примере имеет вид:

```
.define Count2
+label=start
+0n 00
+50n 01
+100n 10
+150n 11
+200n goto start -1 times
```

Из текста программы генератора видно, что циклически воспроизводятся двоичные сигналы на выходе генератора с длительностью 50 наносекунд. Запись +0n 00 означает, что после запуска генератора сразу выставятся два значения 0 и 0 на выходах генератора **Stim2**. Через 50 наносекунд от начала работы генератора сигналы на выходе изменятся на значения 01. А после 100 наносекунд на выходе будет 10 и т.д. После 200 наносекунд весь цикл повторится. Для этого предусмотрена строка с переходом на метку «start» (+200n go to start –1 times). Напомню, что 1 наносекунда это 10^{-9} секунды. В Micro-CAP используются следующие величины длительностей секунды (табл. 2.1).

В Micro-CAP имеется возможность самому формировать последовательность цифровых сигналов **Stim**-генератора. Например, не циклические сигналы:

```
.define Count2
 +0u 00
 +1u 01
 +2u 10
 +3u 11
```

После того, как мы ввели схему и запрограммировали генератор входных сигналов, можно приступить к анализу работы схемы с формированием временных диаграмм в контрольных её точках (A, B, C и Y).

Для симуляции работы цифровых схем воспользуемся инструментом Micro-CAP – анализ переходных процессов. Активизация данного инструмента осуществляется по команде **Analysis → Transient**… (Alt + 1), как на рис. 2.14.

Тут следует пояснить назначение некоторых полей данного окна.

Поле **Maximum Run Time** (рис. 2.16) предназначено для указания длительности симуляции цифровой схемы. В данном случае стоит значение 1u, что соответствует 1 микросекунде ($1 \cdot 10^{-6}$ с).

Если вспомнить, что генератор **Stim2** в нашей схеме настроен на 200 наносекундные повторяющиеся комбинации выходных сигналов (00, 01, 10 и 11), то на интервале длительностью в 1u (1 микросекунда) будет 5 однотипных комбинаций сигналов генератора. Если нас не устраивает такой вариант визуализации исследования и нам необходимо на экране наблюдать только 4 комбинации выходных сигналов, то в окне **Maximum Run Time** следует записать значение 0.2u. Это и будет соответствовать длительности одного цикла работы генератора **Stim2**. Остальные поля рис. 2.16 оставляем без изменения.

Таблица 2.1

Величина	Обозначение в Micro-CAP
10^{12} Т (тера…)	T или t
10^{9} Г (гига…)	G или g
10^{6} М (мега…)	Meg или meg
10^{3} к (кило…)	K или K
10^{-3} м (милли…)	m или M
10^{-6} мк (микро…)	u или U
10^{-9} н (нано…)	n или N
10^{-12} п (пико…)	p или P

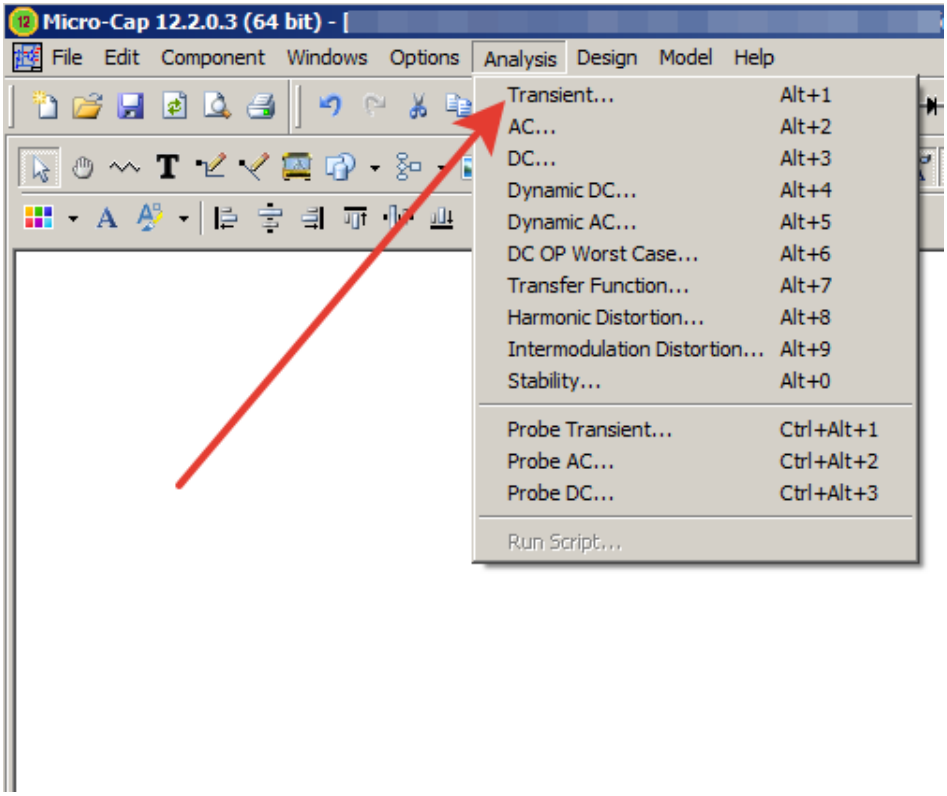


Рис. 2.14. Вызов симулятора цифровой схемы

Появится окно настройки симулятора (рис. 2.15).

В нижней части окна **Transient Analysis Limits** располагаются настройки графиков контрольных точек. Если требуется добавить ещё график какой-либо контрольной точки, то достаточно нажать на кнопку **Add** и в появившиеся поля нового графика вписать необходимые параметры. Заполнение полей для нового графика интуитивно понятно. Поле **Y Expression** предназначено для указания, какой параметр конкретной контрольной точки мы будем наблюдать на графике. В нашем случае, например, **d(A)** означает, что будет выводится дискретное значение сигнала в узле **A**.

Можно выводить десятичный код анализируемых графиков. Для этого необходимо записать, например, DEC(B,A). Если требуется видеть код в двоичной системе счисления, то необходимо записать BIN(B,A). Аналогично можно формировать восьмеричный (OCT) и шестнадцатеричный код (HEX).

Графики сигналов могут быть сгруппированы по отдельным группам. Для указания в какую группу поместить график сигналов имеется поле **p**. В нашем примере все графики контрольных сигналов будут выводиться в одном блоке. В поле **p** стоит значение 1. Если вписать, например, значение 2, то данный график будет выводиться в другое окно графических сигналов.

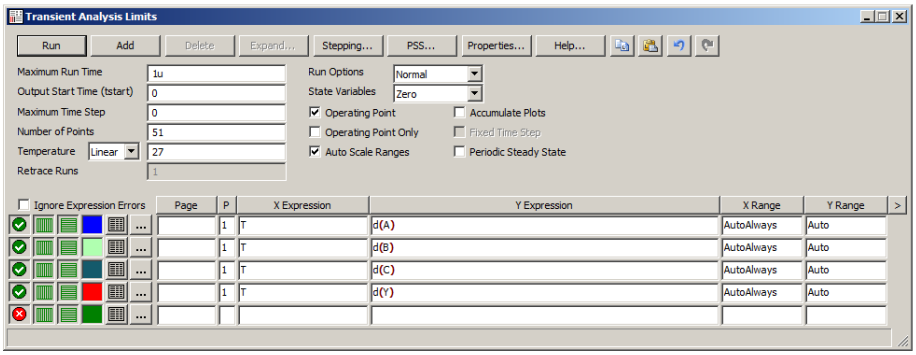


Рис. 2.15. Окно настройки временных диаграмм симулятора

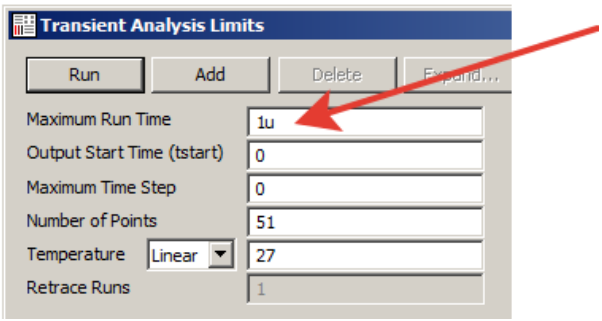


Рис. 2.16. Поле Maximum Run Time

Для построения графиков сигналов в контрольных точках необходимо нажать на кнопку Run.

Если в поле **Maximum Run Time** находится значение 1u, то графики сигналов исследуемых точек будут иметь вид, как на рис. 2.17. На рисунке 2.18 изображены графики симуляции работы схемы при диапазоне исследования 0.2u (200n).

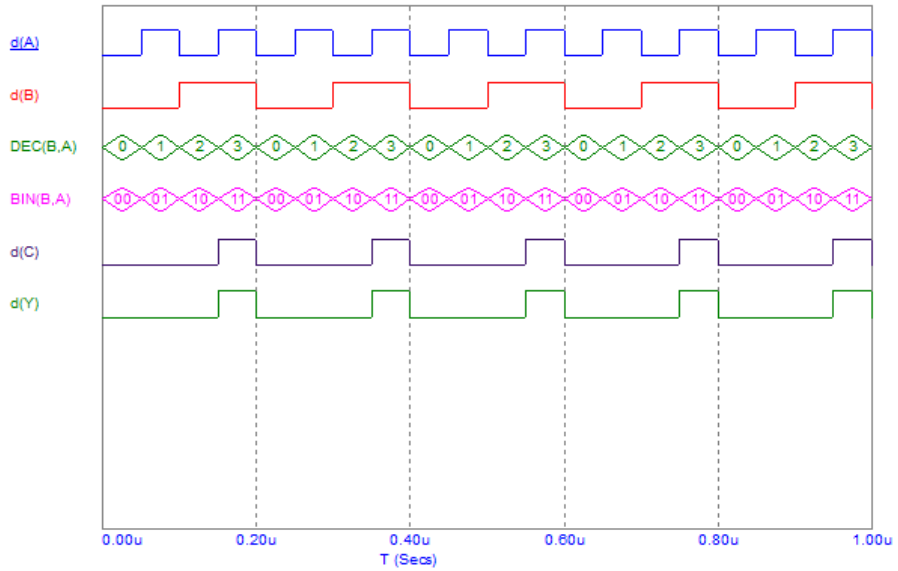


Рис. 2.17. Поле Maximum Run Time = 1u

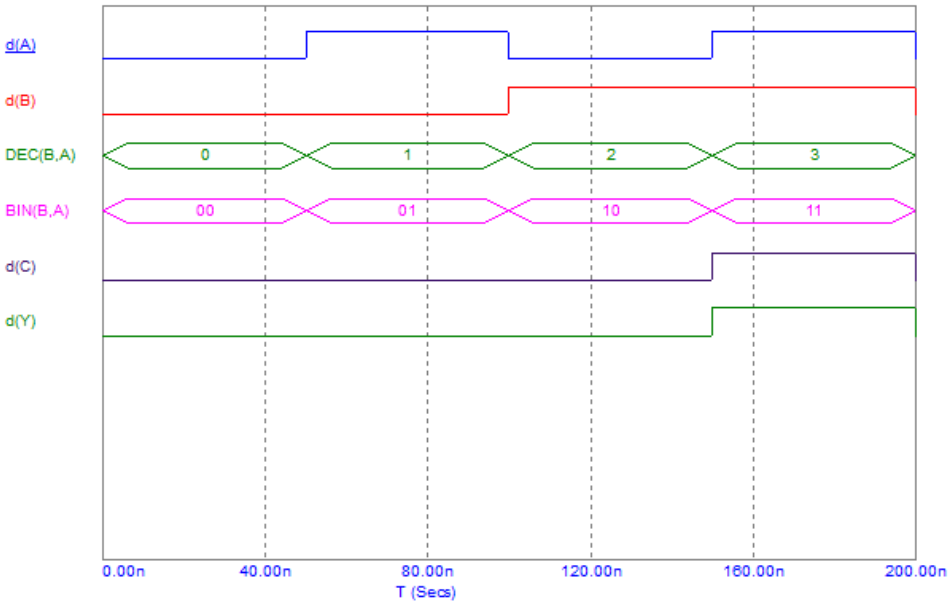


Рис. 2.18. Поле Maximum Run Time = 0.2u (200n)

3. ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ (НЕ, И, ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ)

В данной главе рассмотрим логические элементы, реализующие основные логические функции. Основными логическими элементами являются элементы **НЕ**, **И** и **ИЛИ**. Логический элемент **ИСКЛЮЧАЮЩЕЕ ИЛИ** организован из элементов **НЕ**, **И** и **ИЛИ.**

3.1. ЛОГИЧЕСКИЙ ЭЛЕМЕНТ «НЕ»

Простейшим логическим элементом считается инвертор (логический элемент **НЕ**). Основная функция инвертора – инвертирование, т.е. изменение уровня входного сигнала на противоположный. Элемент имеет один вход и один выход. УГО логического элемента **НЕ** бывает двух видов (рис. 3.1), что одно и то же. В Micro-CAP УГО инвертора представлено вторым вариантом.

Таблица истинности функции **НЕ** имеет вид (табл. 3.1).

Аналитический вид логической функции **НЕ** имеет вид $y = \bar{x}$.

Если подключить два инвертора последовательно, то можно реализовать функцию – повторитель. Напомню, что повторитель – это цифровая схема не изменяющая значение входного логического сигнала. Если поступил на вход повторителя логический сигнал 0, то на выходе будет 0. И наоборот, если поступил логический сигнал 1, то на выходе будет 1. За кажущуюся бессмысленность логической функции повторителя, этот элемент очень востребован в схемотехнике. Благодаря этому элементу можно согласовывать выход цифровой схемы с подключением большого количества входов другой цифровой схемы.

Дело в том, что выходной ток цифровой схемы порой не обеспечивает подключение нескольких входов одновременно. А повторитель позволяет «разгрузить» выходной каскад цифровой схемы от большого потребления входных токов (рис. 3.2).

Рис. 3.1. УГО логического элемента НЕ

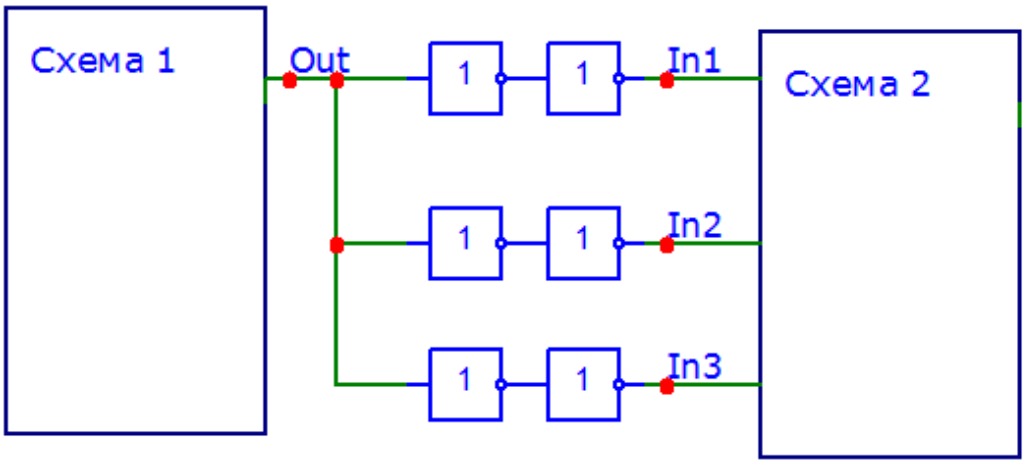


Рис. 3.2. Пример использования повторителей

Таблица 3.1

x	*y*
0	1
1	0

Кроме согласования, повторители используют для задержки сигнала. Но эту функции мы подробно рассмотрим позже, когда в анализе работы цифровых схем будем использовать временные модели логических элементов. Пока речь идёт исключительно о логических моделях, мгновенно выполняющих свои функции. Иногда в литературе можно встретить описание подобных моделей как идеальных.

На практике отечественная промышленность выпускает инвертор и повторитель как самостоятельные микросхемы.

Давайте соберём элементарную схему из инверторов, подключим генератор входных сигналов и проанализируем временные диаграммы контрольных точек (рис. 3.4 и 3.5).

Обратите внимание, что диаграмма Out1 зеркально отражает входной сигнал In. Это и есть инверсия входного сигнала. А повторитель, собранный из инверторов U2 и U3 на выходе Out2, выдаёт копию входного сигнала In.

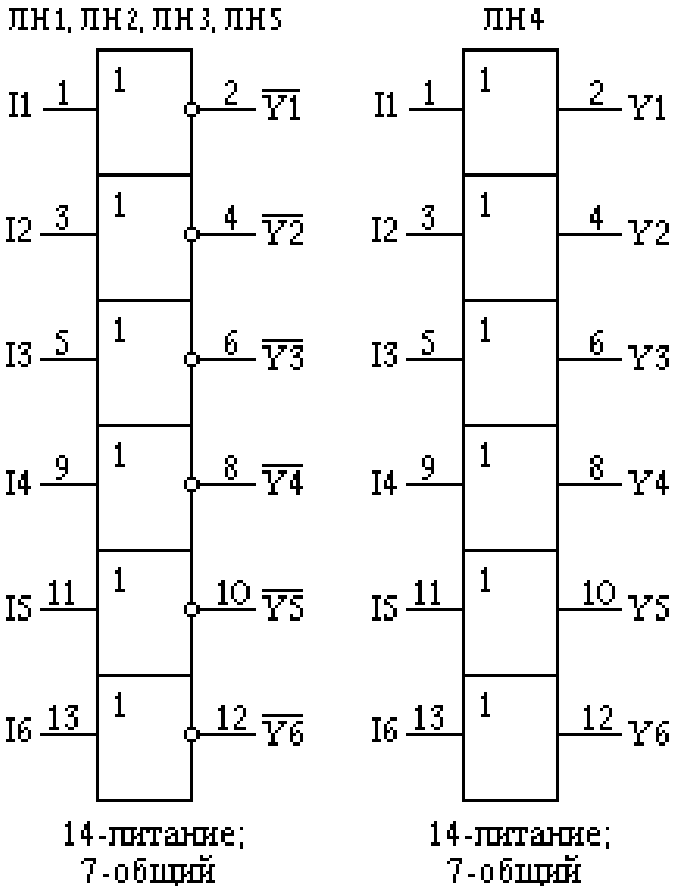


Рис. 3.3. Микросхемы элементов НЕ (слева) и повторителей (справа)

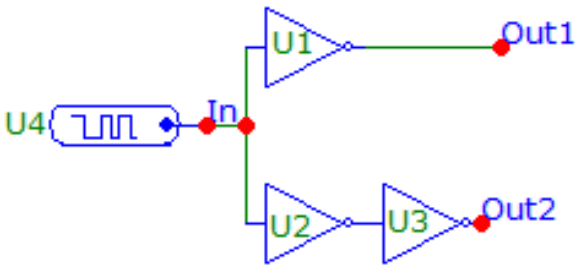


Рис. 3.4. Пример использования инверторов

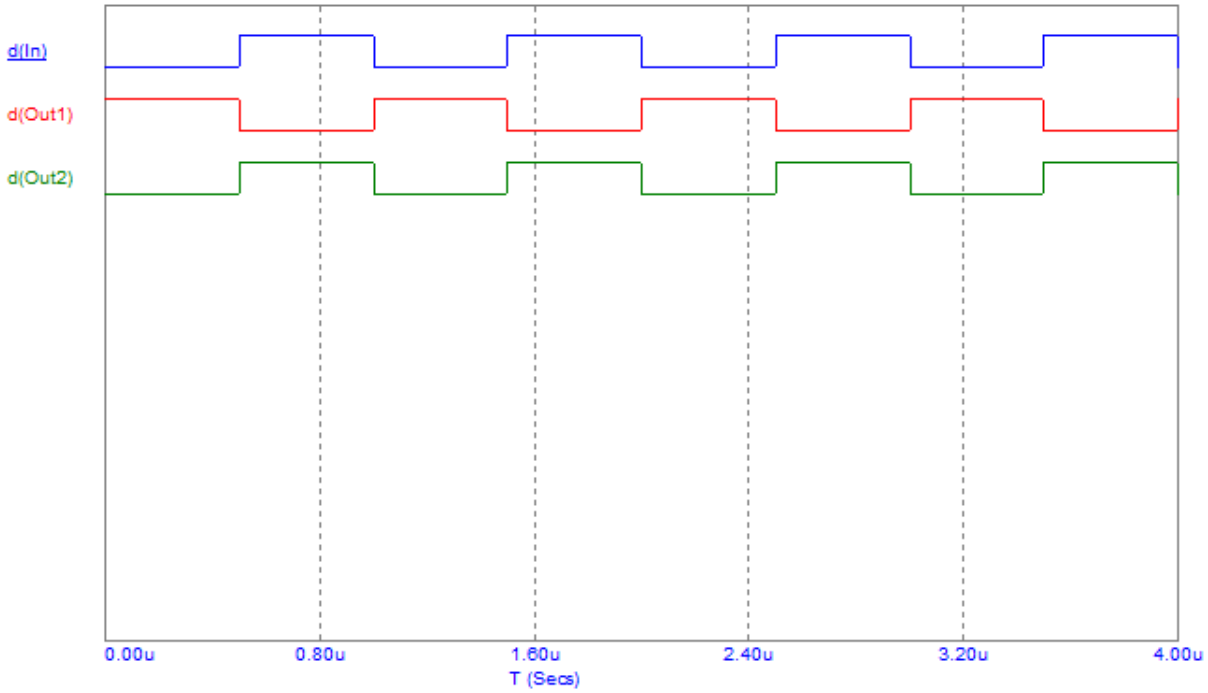


Рис. 3.5. Пример диаграммы сигналов контрольных точек

3.2. ЛОГИЧЕСКИЙ ЭЛЕМЕНТ «И» И «И-НЕ»

Цифровой элемент, выполняющий логическое умножение (конъюнкцию) входных сигналов, называется логический элемент **И**. УГО логического элемента **И** бывают двух видов (рис. 3.6).

Если инвертировать выходное значение логического элемента **И**, то получится функция **И-НЕ** (рис. 3.7). Бинарная операция **И-НЕ** над двумя переменными называется **штрих Шеффера**, введённая в рассмотрение Генри Шеффером в 1913 году.

Таблица истинности функции логического элемента **И** имеет вид (табл. 3.2).

Аналитический вид логической функции **И** имеет вид Вых = Вх1*Вх2.

Таблица истинности функции логического элемента **И-НЕ** имеет вид (табл. 3.3).

Рис. 3.6. УГО логического элемента И

Рис. 3.7. УГО логического элемента И-НЕ

Таблица 3.2

Вх1	Вх2	Вых
0	0	0
0	1	0
1	0	0
1	1	1

Таблица 3.3

Вх1	Вх2	Вых
0	0	1
0	1	1
1	0	1
1	1	0

Аналитический вид логической функции **И-НЕ** имеет вид Вых = $\overline{Вх1 * Вх2}$.

Обычно микросхемы с элементами 2**И** и 2**И-НЕ** выпускают в виде сборки, состоящей из 4 таких элементов (рис. 3.8).

Посмотрим, как элементы **И** и **И-НЕ** отрабатывают свои внутренние функции среде Micro-CAP. Соберём схему (рис. 3.9).

Настроим выходы генератора U3 (Stim2) на периодические двоичные коды длительностью 50n. Для этого воспользуемся стандартными настройками генератора. В окне настройки модели генератора нажмем на кнопку **Count2** и подтвердим выбор сигналов генератора нажатием на кнопку **OK**. Программа генератора имеет вид:

.

```
.define Count2
+label=start
+0n 00
+50n 01
+100n 10
+150n 11
+200n goto start -1 times
```

Входим в окно настройки симуляции **Analysis → Transient** и в окне **Maximum Run**.

Time указываем длительность симуляции 250n. Нажимаем кнопку **Run** и получаем временные диаграммы сигналов в контрольных точках (рис. 3.10).

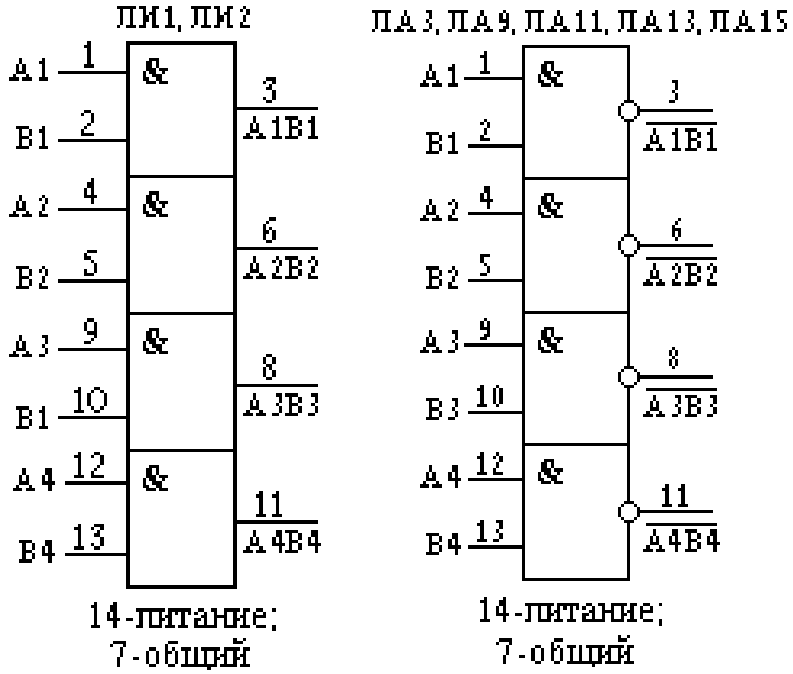


Рис. 3.8. Микросхемы элементов НЕ (слева) и повторителей (справа)

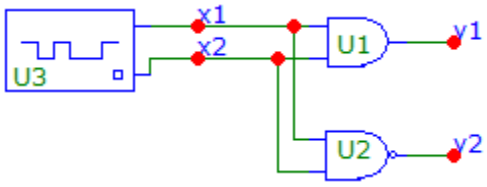


Рис. 3.9. Схема для исследования логических элементов И и И-НЕ

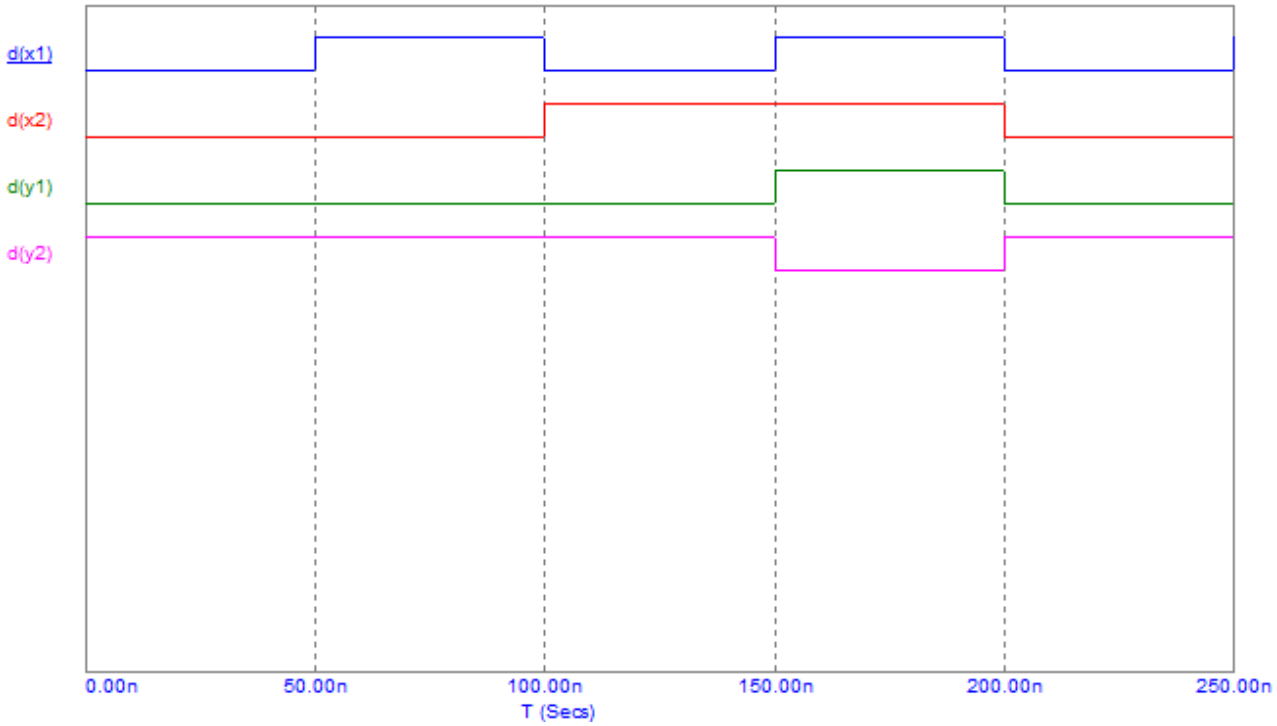


Рис. 3.10. Исследование логических элементов И и И-НЕ

Подобно водопроводному крану, который может пропускать или не пропускать через себя воду, работает и логический элемент **И**. Недаром это элемент относится к группе вентилей. Рассмотрим схему, где логические элементы **И** работают устройством, разрешающим или не разрешающим прохождение входного сигнала на выход (рис. 3.11).

Временные диаграммы вентиля **И**, работающего в качестве устройства разрешающего/запрещающего прохождения сигналов, приведены на рис. 3.12 (верхнее окно). Обратите внимание, что на выходе Out появляются входные сигналы лишь в случае, когда на втором входе логического элемента **И** находится логическая 1. Это классический пример вентиля для прохождения сигналов.

Если в качестве вентиля в схеме 3.11 установить логический элемент **И-НЕ**, то изменится выходной сигнал на противоположное значение по отношению к входному сигналу (рис. 3.12 (нижнее окно)).

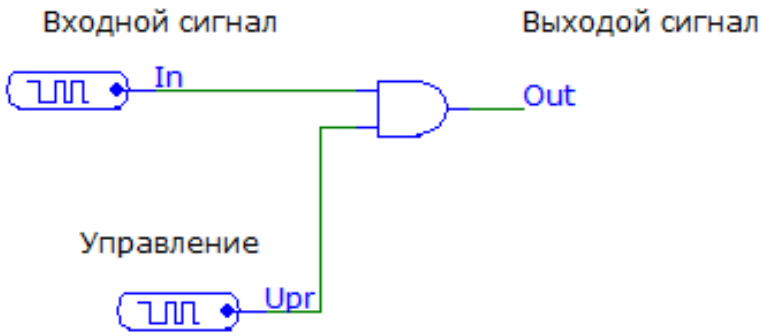


Рис. 3.11. Схема для исследования логического элемента И как вентиль

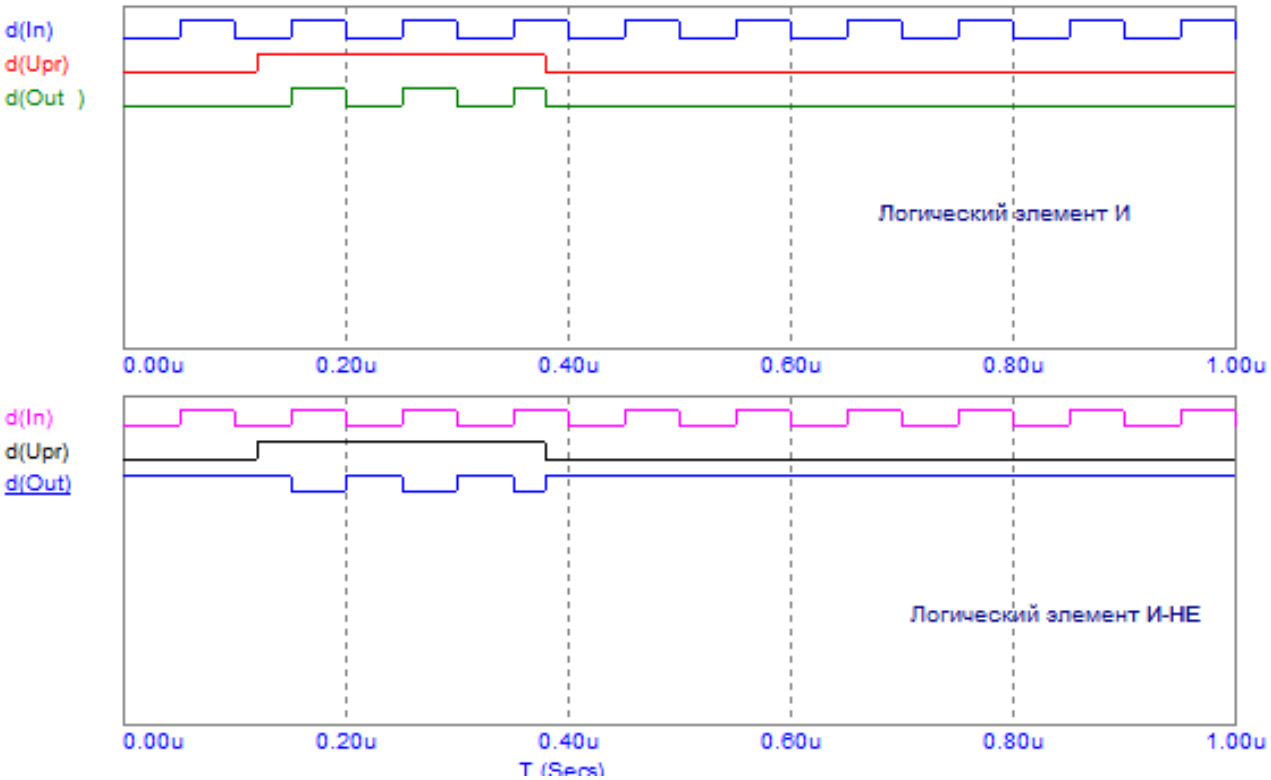


Рис. 3.12. Исследование вентиля И и И-НЕ

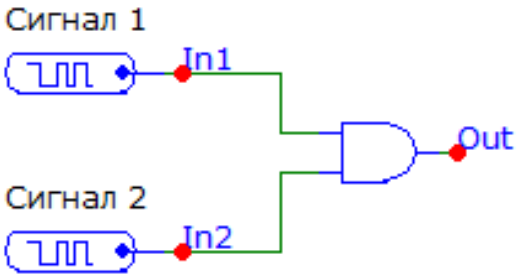


Рис. 3.13. Схема смешивания двух потоков данных

Иногда в цифровых схемах необходимо смешивать два сигнальных потока в один. Если данные в потоке определяются нулями, то для этой задачи используется логический элемент **И.** Схема для смешивания двух сигналов с использованием элемента **И** представлена на рис. 3.13.

На временной диаграмме контрольных точек видно, что отрицательные сигналы 1 и 2 через вход In1 элементом **И** смешались с сигналом 3 через вход In2 в поток на выходе Out – 1,3 и 2 (рис. 3.14).

Элемент **И** часто используют как элемент выделения совпадающих сигналов. А именно, когда на входах элемента **И** совпадают сигналы, то на выходе появляется сигнал совпадения – 1. Если совпадение отсутствует, то на выходе будет 0. Рассмотрим эту функцию на примере схемы, представленной на рис. 13.3. Здесь элемент **И** анализирует «Сигнал 1» и «Сигнал 2». Временная диаграмма смешивания представлена на рис. 3.15. Стрелкой указан сигнал совпадения.

Если заменить в схеме сравнения элемент **И** на элемент **И-НЕ**, то можем вырабатывать сигнал совпадения для отрицательной логики, т.е. в данном случае при совпадении положительных уровней сигнала на выходе появится логический 0. Временные диаграммы для элемента **И-НЕ** в схеме сравнения представлены на рис. 3.16.

Если объединить два входа у элемента **И-НЕ**, то получим повторитель, о котором ранее велась речь (рис. 3.17).

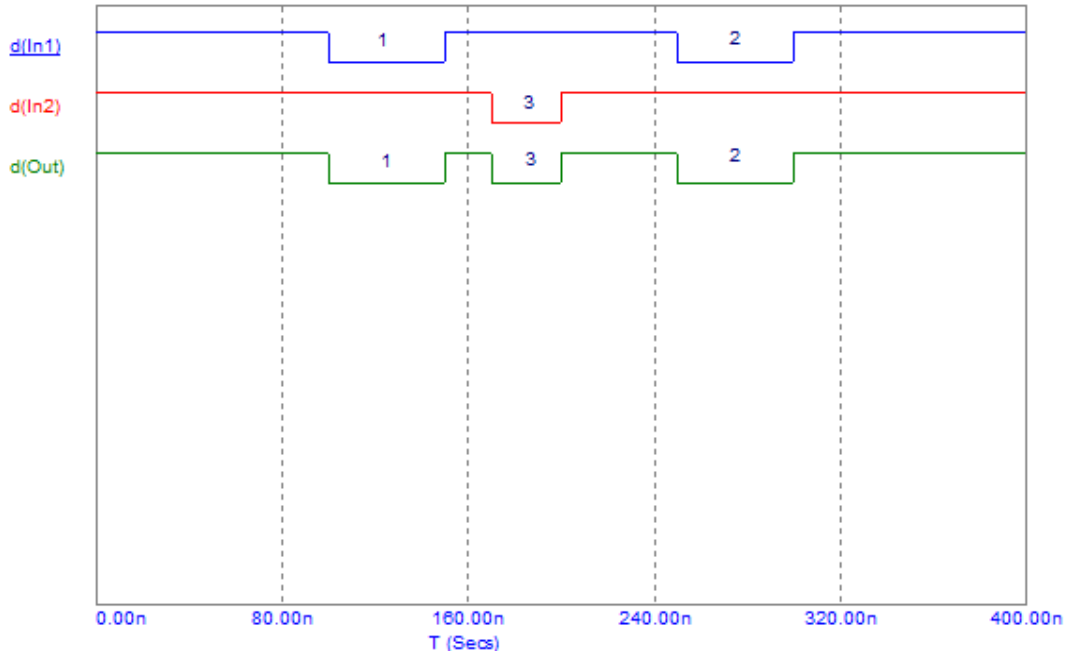


Рис. 3.14. Смешивание сигналов вентилем И

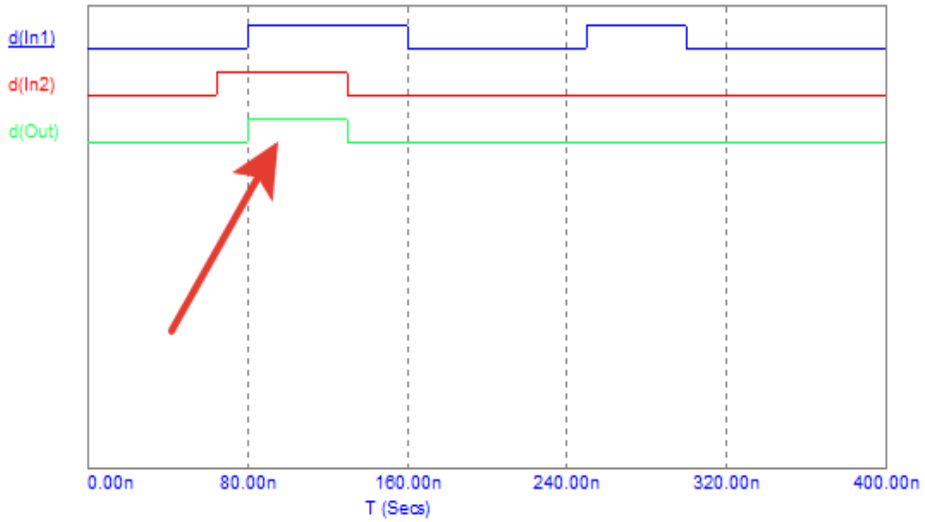


Рис. 3.15. Исследование вентиля И в схеме совпадения

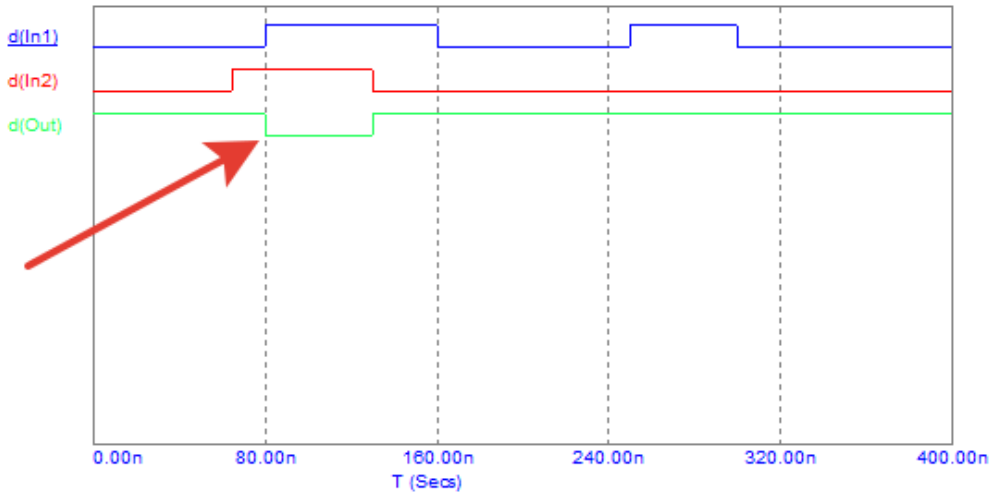


Рис. 3.16. Исследование вентиля И-НЕ в схеме совпадения

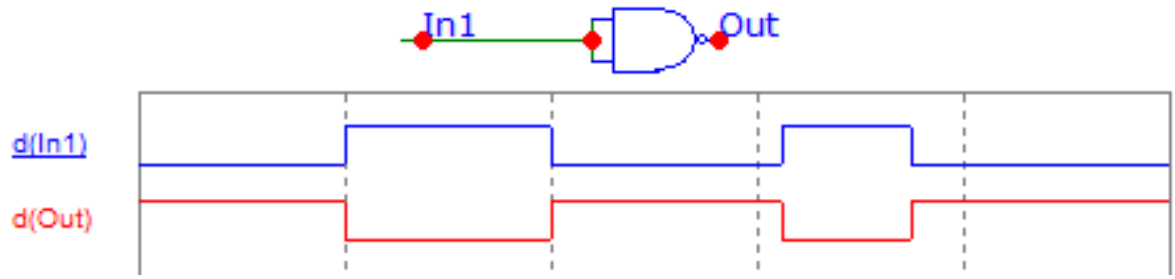


Рис. 3.17. Инвертор из элемента И-НЕ

3.3. ЛОГИЧЕСКИЙ ЭЛЕМЕНТ «ИЛИ» И «ИЛИ-НЕ»

Логическое сложение (дизъюнкция) входных сигналов выполняется цифровым элементом **ИЛИ**. УГО логического элемента **ИЛИ**, как и у элемента **И** бывают двух видов (рис. 3.18).

Если инвертировать выходное значение логического элемента **ИЛИ**, то получится функция **ИЛИ-НЕ** (рис. 3.19). Бинарная операция **ИЛИ-НЕ** над двумя переменными называется **стрелка Пирса**, введённая в рассмотрение Чарльзом Пирсом 1880 году. Стрелка Пирса иногда обозначается ↓.

Рис. 3.18. УГО логического элемента ИЛИ

Рис. 3.19. УГО логического элемента ИЛИ-НЕ

Таблица истинности функции логического элемента **ИЛИ** имеет вид (табл. 3.4).

Аналитический вид логической функции **ИЛИ** имеет вид Вых = Вх1 + Вх2.

Таблица истинности функции логического элемента **ИЛИ-НЕ** имеет вид (табл. 3.5).

Аналитический вид логической функции **ИЛИ-НЕ** имеет вид Вых $= \overline{Вх1 + Вх2}$ или Вых = Вх1↓Вх2.

Посмотрим, как элементы **ИЛИ** и **ИЛИ-НЕ** реагируют на входные сигналы в среде Micro-CAP. Соберём схему (рис. 3.20).

Здесь U1 – логический элемент **ИЛИ**, а U2 – **ИЛИ-НЕ**.

Программы для Stim1 (U3) и Stim1 (U4) приведены в табл. 3.6.

Выбираем интервал исследования 200n (200 наносекунд) и получаем следующие временные диаграммы (рис. 3.21).

Таблица 3.4

Вх1	Вх2	Вых
0	0	0
0	1	1
1	0	1
1	1	1

Таблица 3.5

Вх1	Вх2	Вых
0	0	1
0	1	0
1	0	0
1	1	0

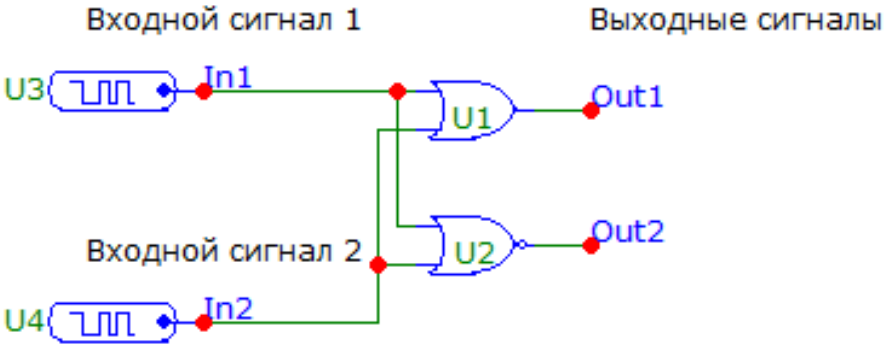


Рис. 3.20. Схема из элементов ИЛИ и ИЛИ-НЕ

Таблица 3.6

U3	U4
.define _S1 +0n 0 +50n 1 +100n 0	.define S2 +0n 0 +80n 1 +120n 0

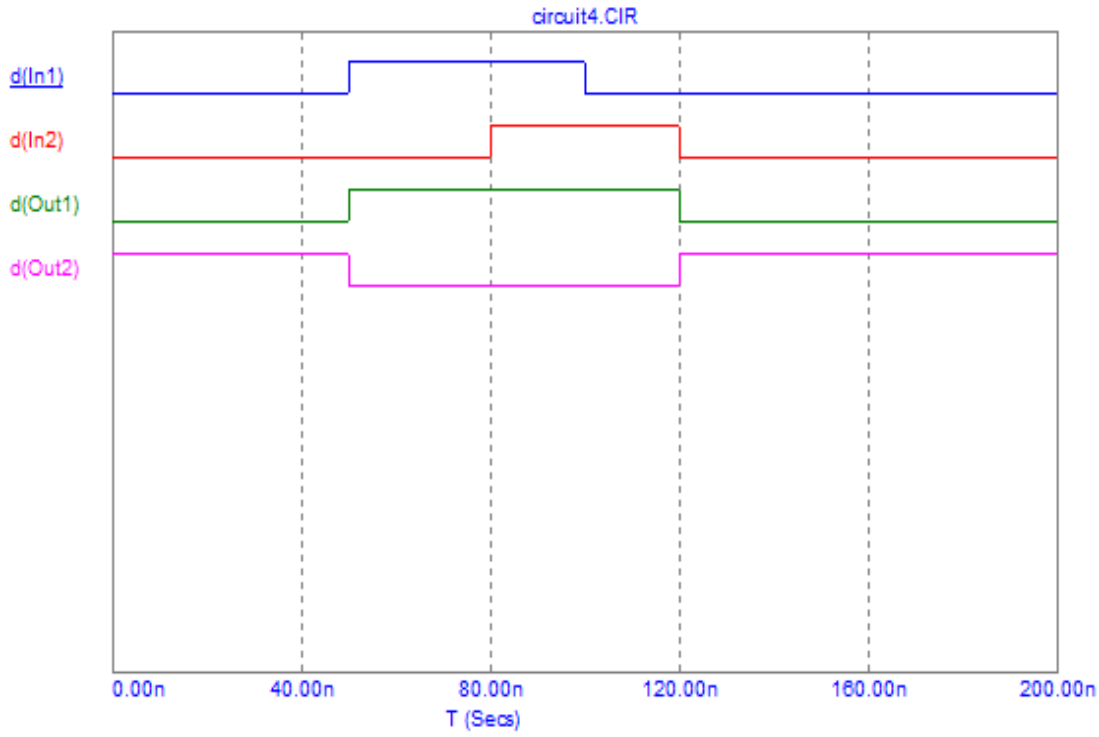


Рис. 3.21. Исследование временных диаграмм элементов ИЛИ и ИЛИ-НЕ

Здесь кривые d(Out1) и d(Out2) полностью передают логику работы элементов **ИЛИ** и **ИЛИ-НЕ**, согласно их таблиц истинности (табл. 3.4 и 3.5).

Если рассматривать элемент **ИЛИ** как пропускатель (вентиль) сигналов, то в отличие от элемента **И** управление пропусканием сигнала осуществляется отрицательным сигналом – 0 (рис. 3.22). Этот факт подтверждается временными диаграммами на рис. 3.23. Напомним, что для элемента **И** разрешение для пропускания сигнала через вентиль осуществляется положительным сигналом – 1. На приведённой схеме кроме вентиля из элемента **ИЛИ** (U1) приведён вентиль с инверсией **ИЛИ-НЕ** (U2), который изменяет сигнал на противоположный (см. d(Out2)) по отношению U1 (см. d(Out1)).

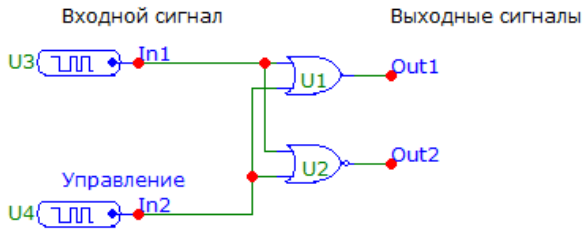


Рис. 3.22. Схема для исследования логического элемента ИЛИ и ИЛИ-НЕ, как вентиль

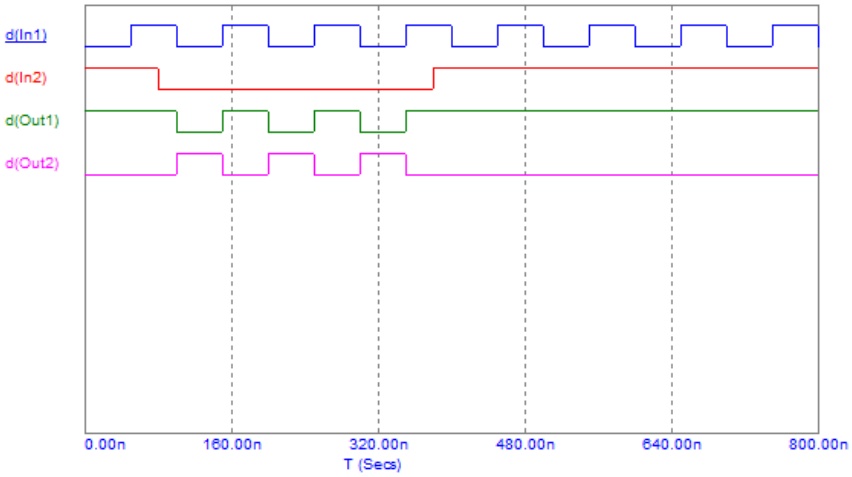


Рис. 3.23. Исследование вентиля ИЛИ и ИЛИ-НЕ

Элементом **ИЛИ** можно смешивать два сигнала. Но в отличие от элемента **И**, который смешивает отрицательные сигналы, элемент **ИЛИ** смешивает положительные сигналы. Рассмотрим схему с элементами **ИЛИ** и **ИЛИ-НЕ** для исследования режима смешивания входных сигналов (см. рис. 3.20). Генераторы U3 и U4 настроены, как в табл. 3.7.

Временные диаграммы исследования схемы смешивания сигналов представлены на рис. 3.24. Обратите внимание, как на результирующей кривой выходного сигнала Out1 совместились входные сигналы In1 и In2. Если использовать элемент **ИЛИ-НЕ**, то выходной сигнал Out2 полностью передаёт сигнал Out1, только в инверсном варианте.

Таблица 3.7

U3	U4
.define _S1	.define S2
+0ns 0	+0ns 0
+50ns 1	+120ns 1
+100n 0	+125ns 0
+150n 1	+130ns 1
+200n 0	+140ns 0

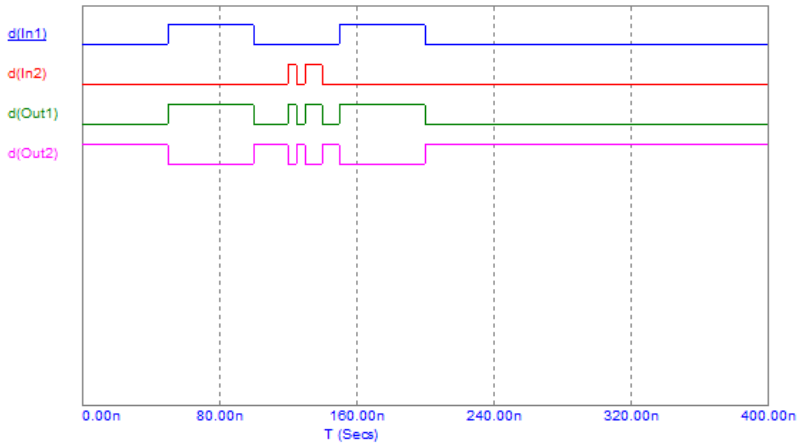


Рис. 3.24. Исследование элементов ИЛИ и ИЛИ-НЕ в качестве смесителя

Наконец, рассмотрим использование элемента **ИЛИ** в схеме выявления совпадения входных сигналов. В отличие от схемы с элементом **И**, логическое сложение выделяет отрицательные сигналы. При этом и сам сигнал совпадения будет отрицательным. Опять рассмотрим схему, как на рис. 3.20, но изменим настройки генераторов тестовых сигналов. Настройки приведены в табл. 3.8.

Настроив интервал исследования в симуляторе Micro-CAP в 300n, мы получаем следующие временные диаграммы исследуемой схемы (рис. 3.25).

Переход сигнала Out1 из единичного состояния в нулевое и наоборот – это и есть отрицательный сигнал совпадения нулевых значений сигналов In1 и In2. Инвертирование выходного сигнала элемента ИЛИ преобразует отрицательный сигнал совпадения в положительный сигнал (Out2).

Таблица 3.8

U3	U4
.define _S1 +0ns 1 +50n 0 +70n 1 +130n 0 +180n 1	.define S2 +0ns 1 +100ns 0 +195ns 1

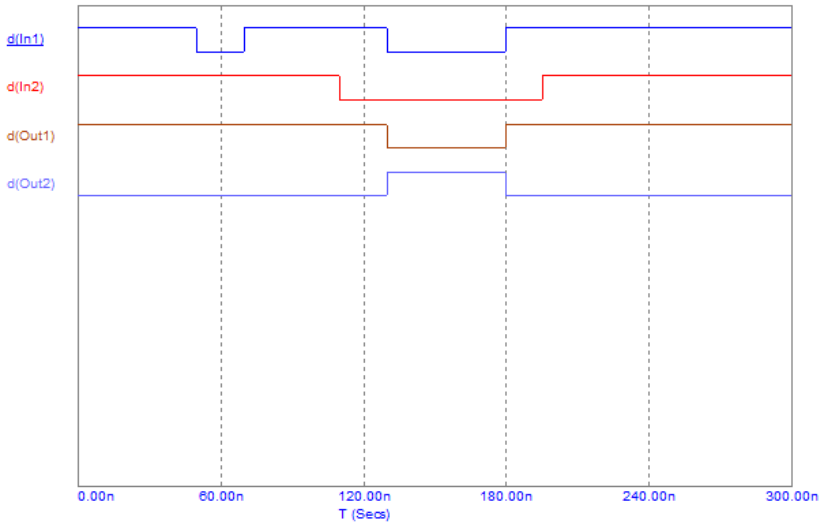


Рис. 3.25. Исследование ИЛИ и ИЛИ-НЕ в качестве схемы выделения одинаковых отрицательных сигналов

3.4. ЛОГИЧЕСКИЙ ЭЛЕМЕНТ «ИСКЛЮЧАЮЩЕЕ ИЛИ»

Логический элемент **Исключающее ИЛИ** относится к простым логическим элементам, но внутренняя функция данного элемента не совсем простая, как у **И**, **И-НЕ**, **ИЛИ** и **ИЛИ-НЕ**. Все входы элементов **Исключающее ИЛИ** равнозначные, но ни один из входов не может заблокировать другие входы, установив выходной сигнал в уровень единицы или нуля, как мы наблюдали, например, у вентиля **И**.

УГО логического элемента **Исключающее ИЛИ**, бывают двух видов (рис. 3.26). На рисунке слева отечественное изображение элемента, а справа – зарубежное.

Таблица истинности функции логического элемента **Исключающее ИЛИ** имеет вид (табл. 3.9).

Запишем логическую функцию элемента **Исключающее ИЛИ** по табл. 3.9 в СДНФ:

$$\text{Вых} = \overline{\text{Вх1}} \cdot \text{Вх2} + \text{Вх1} \cdot \overline{\text{Вх2}}$$

Принципиальная схема полученной функции имеет вид (рис. 3.27).

Видим из схемы, что элемент **Исключающее ИЛИ** состоит из примитивной логики, но из-за частого применения этой функции в схемотехнике, **Исключающее ИЛИ** применяют, как готовую микросхему. И на практике этот элемент никто не моделирует из простых логических элементов.

Если говорить о микросхеме **Исключающее ИЛИ**, то следует отметить, что в одной микросхеме размещается 4 подобных элемента (рис. 3.28).

Элемент **Исключающее ИЛИ** выполняет операцию так называемого суммирования по модулю 2. Поэтому эти элементы также называются сумматорами по модулю два и обозначается суммирование по модулю 2 знаком плюса, заключенного в кружок:

$$\text{Вых} = \text{Вх1} \oplus \text{Вх2}.$$

В схемотехнике элемент **Исключающее ИЛИ** применяют в основном для сравнения двух входных сигналов. В случае, если на входы пришли два одинаковых сигнала, например, нули или единицы, то на выходе формируется ноль. Если инвертировать выход, то совпадение сигналов будет сигнализировать на входе логическая единица.

Для исследования отработки данной функции элементом **Исключающее ИЛИ** соберём схему, как на рис. 3.29.

Подготовим исходные данные для исследования настройками генераторов U3 и U4, как в табл. 3.10.

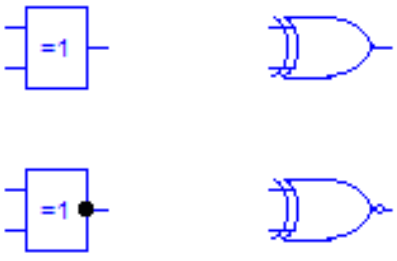


Рис. 3.26. УГО логического элемента Исключающее ИЛИ и Исключающее ИЛИ-НЕ

Таблица 3.9

Вх1	Вх2	Вых
0	0	0
0	1	1
1	0	1
1	1	0

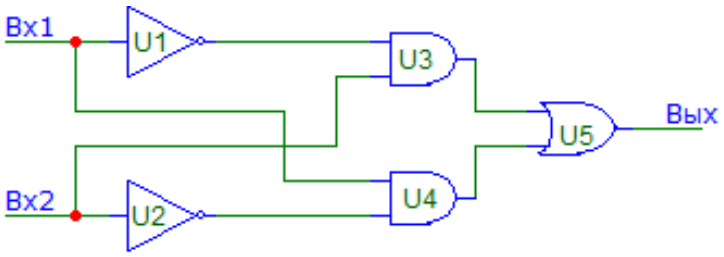


Рис. 3.27. Моделирование элемента Исключающее ИЛИ

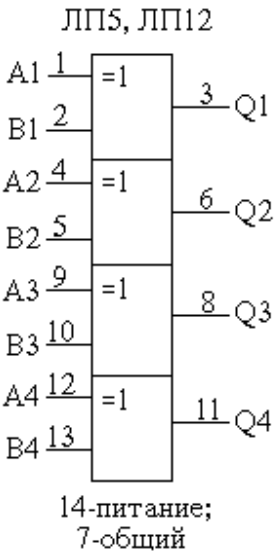


Рис. 3.28. Микросхема ЛП5 и ЛП12 элемента Исключающее ИЛИ

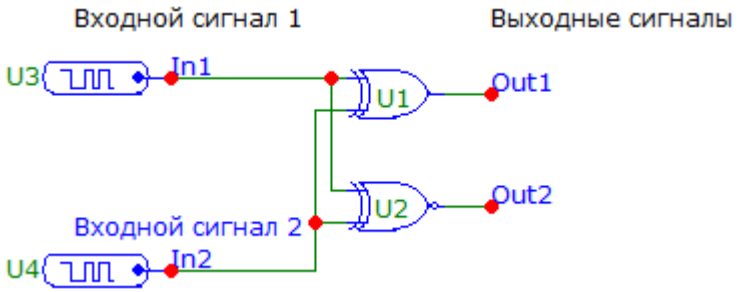


Рис. 3.29. Схема для исследования логических элементов Исключающее ИЛИ и Исключающее ИЛИ-НЕ, как компаратор

На временном интервале в 300n строим временные диаграммы сигналов в контрольных точках схемы (рис. 3.30).

Стрелками на рис. 3.30 указаны выходные сигналы Out1(**Исключающее ИЛИ**) и Out2 (**Исключающее ИЛИ-НЕ**) при совпадении входных сигналов In1 и In2.

Таблица 3.10

U3	U4
.define S1 +0n 0 +50n 1 +100n 0 +150n 1 +180n 0	.define S2 +0n 0 +130n 1 +195n 0

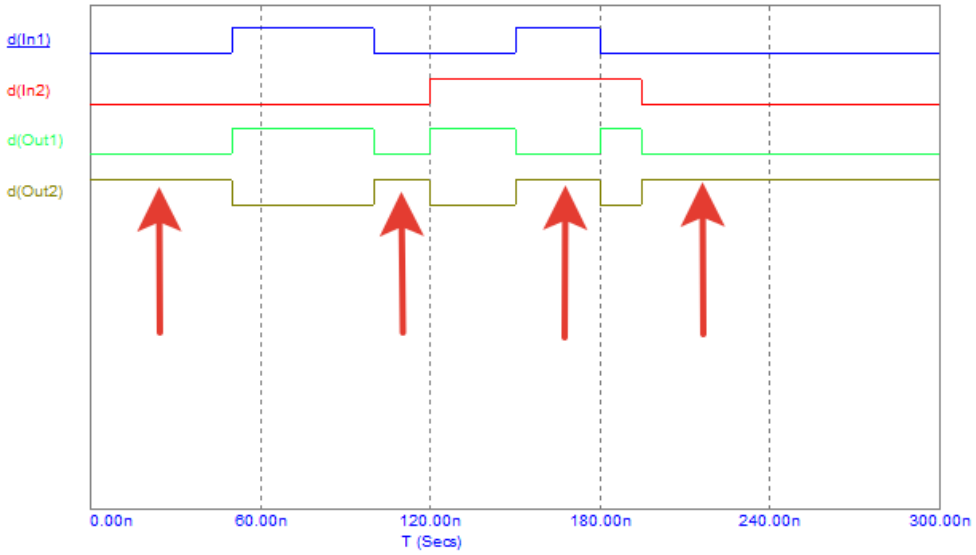


Рис. 3.30. Исследование элементов Исключающее ИЛИ и Исключающее ИЛИ-НЕ в качестве схемы сравнения входных сигналов

Кроме компарирования входных сигналов, элемент **Исключающее ИЛИ** часто используют в реализации управляющего инвертора. В данном случае один из входов элемента используется как управляющий вход, а на другой вход подаётся информационный сигнал. Если на управляющем входе будет логическая единица, то входные данные будут инвертироваться на выходе, если же ноль – не инвертироваться.

Рассмотрим схему, где элемент **Исключающее ИЛИ** выступает в роли управляемого инвертора (рис. 3.31).

Подготовим исходные данные для исследования настройками генераторов U3 и U4, как в табл. 3.11.

На интервале 10u получаем следующие временные диаграммы (рис. 3.32).

Входной сигнал In начинает инвертироваться на выходе Out с момента 5u. Это и есть – управляемый инвертор.

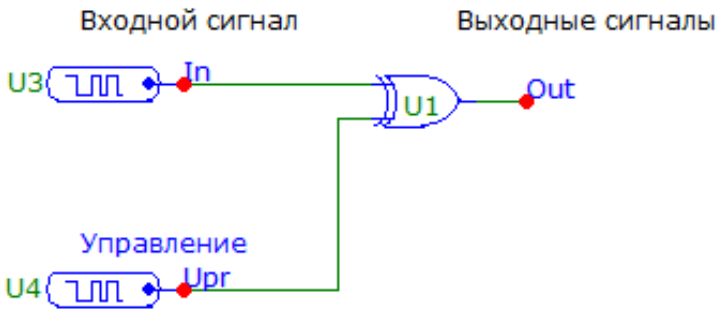


Рис. 3.31. Схема для исследования логических элементов Исключающее ИЛИ как управляющий инвертор

Таблица 3.11

U3	U4
.define _1MHzClk +0ns 0 +label=start +500n 1 +1u 0 +1.5u goto start -1 times	.define S1 +0u 0 +5u 1

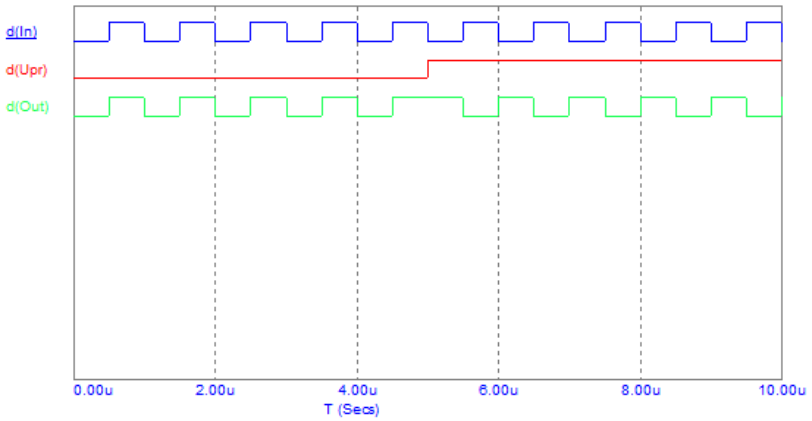


Рис. 3.32. Исследование элемента Исключающее ИЛИ в качестве схемы управляющего инвертора

Элемент **Исключающее ИЛИ** можно использовать и для смешивания сигналов. Допустим, имеются два сигнала одинаковой полярности (положительные или отрицательные) и исключается их одновременное появление. При любой полярности входных сигналов выходные сигналы элемента будут положительными. При положительных входных сигналах элемент **Исключающее ИЛИ** будет работать как элемент **2ИЛИ**, а при отрицательных он будет заменять элемент **2И-НЕ**.

Рассмотрим это свойство элемента **Исключающее ИЛИ** на примере схемы, представленной на рис. 3.33.

Проведём два эксперимента. В первом случае попробуем смешивать положительные сигналы (логические единицы), а во втором отрицательные (логические нули). Для этого настроим генераторы U3 и U4. Для положительных входных сигналов настройки приведены в табл. 3.12. На интервале в 300n мы наблюдаем корректное смешивание двух положительных сигналов (рис. 3.34).

Для отрицательных входных сигналов настройки приведены в табл. 3.13. На интервале в 300n мы наблюдаем корректное смешивание двух отрицательных сигналов (рис. 3.35).

Обратите внимание, что не зависимо, какие смешиваются сигналы, на выходе результирующие данные будут положительные (рис. 3.32 и 3.33).

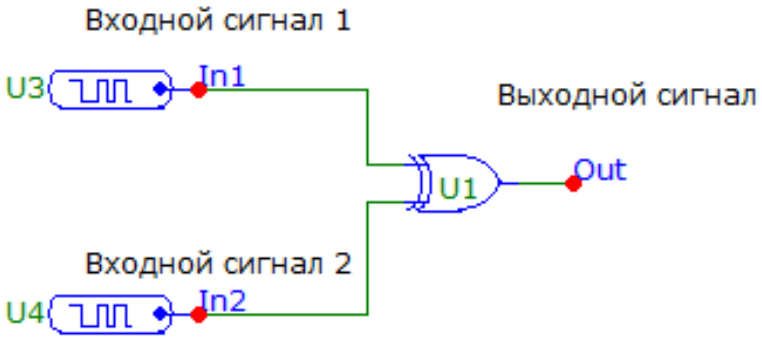


Рис. 3.33. Схема для исследования элемента Исключающее ИЛИ для смешивания двух сигналов

Таблица 3.12

U3	U4
.define S1 +0n 0 +50n 1 +100n 0	.define S2 +0n 0 +150n 1 +200n 0

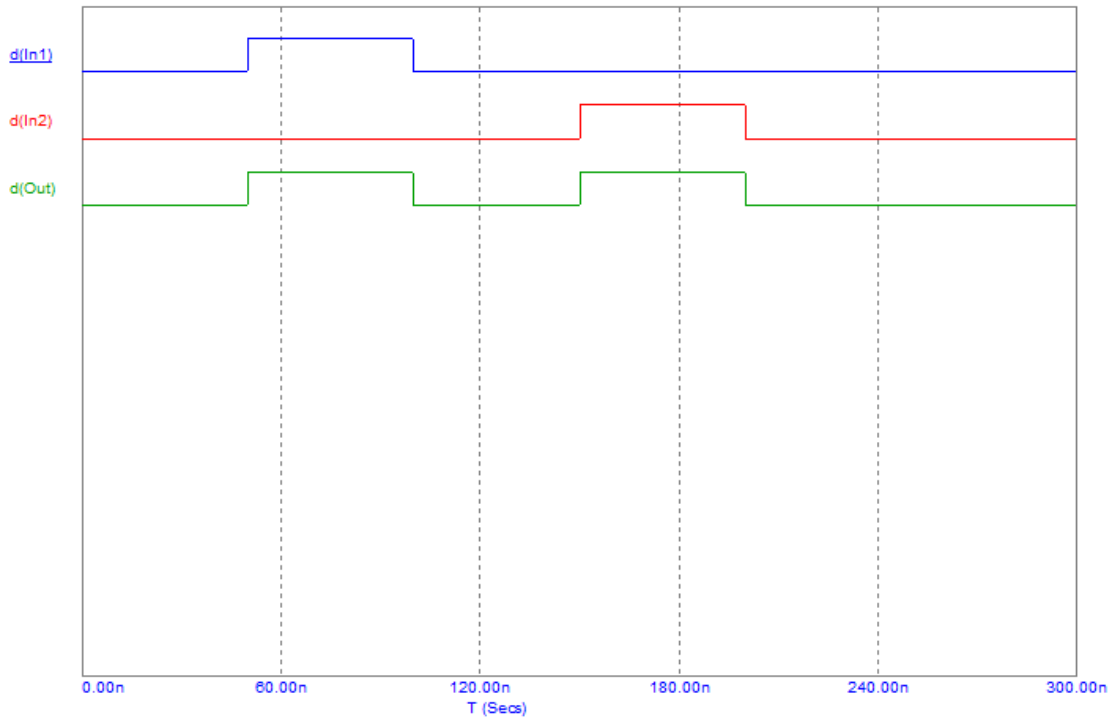


Рис. 3.34. Исключающее ИЛИ смешивает положительные сигналы

Таблица 3.13

U3	U4
.define S1 +0n 1 +50n 0 +100n 1	.define S2 +0n 1 +150n 0 +200n 1

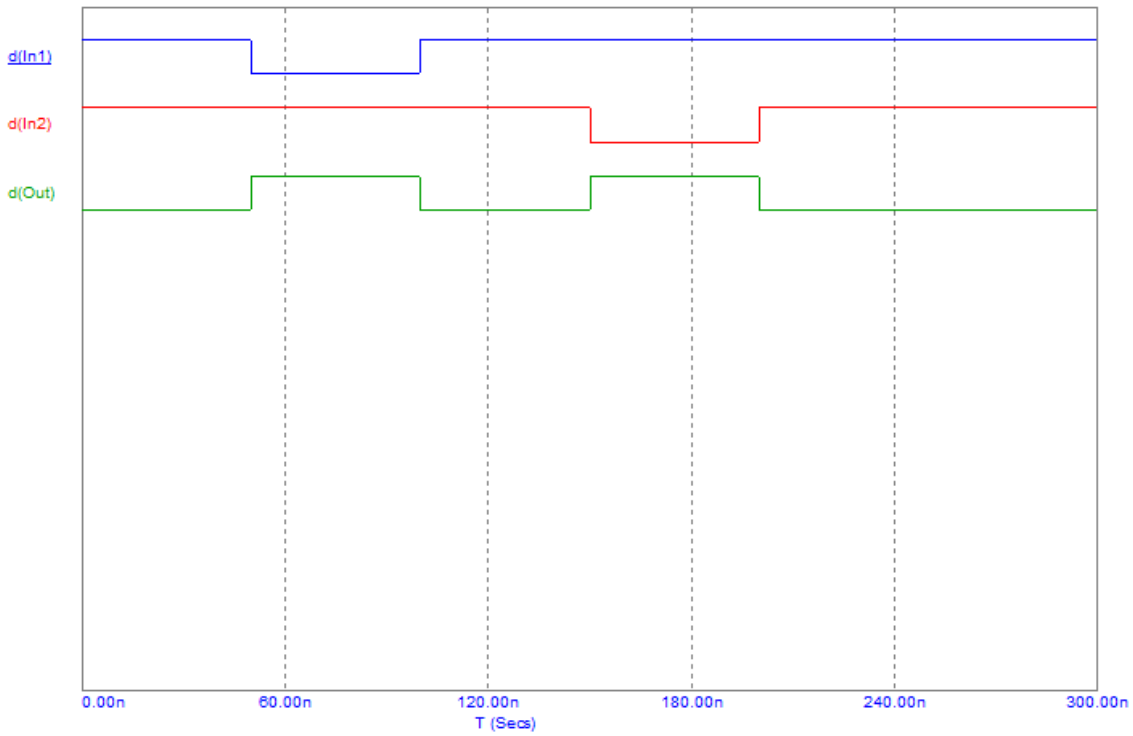


Рис. 3.35. Исключающее ИЛИ смешивает отрицательные сигналы

4. КОМБИНАЦИОННАЯ СХЕМОТЕХНИКА ЦИФРОВЫХ СХЕМ

Изучив базовые логические элементы цифровой схемотехники, мы можем приступить к созданию более сложных схем, состоящих из элементарной логики. И начнем с синтеза комбинационных схем. Под синтезом цифровых схем мы будем понимать построение схем с заданным условием функционирования.

Комбинационные схемы – это устройства без памяти. Выходные сигналы этого вида цифровых схем зависят только от текущей комбинации входных логических сигналов и не зависят от их предыдущих значений (рис. 4.1).

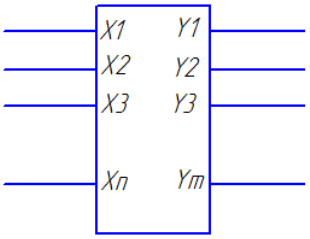


Рис. 4.1. Функциональная схема цифрового комбинационного устройства

Синтез комбинационных схем начинается с формирования её функциональных требований. Обычно на этом этапе формируется таблица истинности. После чего осуществляется анализ схемы. Анализ схемы – это особый этап синтеза, когда строится аналитическая запись логической функции по значению ячеек таблицы истинности. И далее, минимизируется аналитический вид логической функции, работоспособность которой должна быть адекватна исходным значениям таблицы истинности. По аналитической записи переключательной функции строится итоговая аппаратная схема из логических элементов.

Рассмотрим синтез комбинационной схемы, у которой имеется четыре входа и один выход.

Например, пусть будущая комбинационная схема должна функционировать согласно значений таблицы истинности, приведённой в табл. 4.1.

Для построения аналитической записи функции *Y* необходимо выбрать из таблицы строки (наборы), где значение функции равно 1. Таких наборов для данного примера будет 9 шт. В каждом выбранном наборе организуем логическое перемножение (конъюнкцию) входных данных по следующему правилу.

Таблица 4.1

X_1	X_2	X_3	X_4	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

В произведение помещается без изменения входная переменная, если её значение равно 1 и инверсия этой переменно, если 0. Полученные конъюнкции логически складываются, т.е. осуществляем дизъюнкцию конъюнкций входных данных. Так получаем дизъюнктивную нормальную форму (ДНФ) булевой функции, а точнее СДНФ. Дело в том, что если в конъюнкциях булевой функции присутствуют все входные переменные пусть в прямом виде или в инверсном виде, то такой вид функции называется совершенная ДНФ (СДНФ). В ДНФ возможно отсутствие в некоторых слагаемых некоторых входных переменных. Слагаемое в СДНФ (ДНФ) называется минтерм. Минтерм – это конъюнктивный терм, связывающий переменные, представленные в прямой или инверсной форме, знаком конъюнкция.

Для данного примера (табл. 4.1) логическая функция в СДНФ имеет вид:

$$Y = \overline{X}_1\overline{X}_2\overline{X}_3\overline{X}_4 + \overline{X}_1\overline{X}_2X_3\overline{X}_4 + \overline{X}_1X_2\overline{X}_3X_4 + \overline{X}_1X_2X_3X_4 + X_1\overline{X}_2X_3\overline{X}_4 + X_1X_2\overline{X}_3\overline{X}_4 + \\ + X_1X_2\overline{X}_3X_4 + X_1X_2X_3\overline{X}_4 + X_1X_2X_3X_4. \qquad (4.1)$$

Построим в среде Micro-CAP комбинационную схему по СДНФ (4.1) (рис. 4.2). Нам потребуются четы инвертора (U1-U4) для инвертирования входных сигналов X_1, X_2, X_3 и X_4. Для осуществления конъюнкции 9 логических элементов **4И** (And4) – U6-U14. Дизъюнкция девяти конъюнкций будет осуществляться девятивходовым элементом **9ИЛИ** (Or9) – U5.

Значения входных сигналов будем моделировать четырёхразрядным генератором Stim4 (U15) с внутренней программой, как в табл. 4.2.

Устанавливаем в симуляторе интервал исследования 1,6u и получаем временные диаграммы в контрольных точках схемы (рис. 4.3).

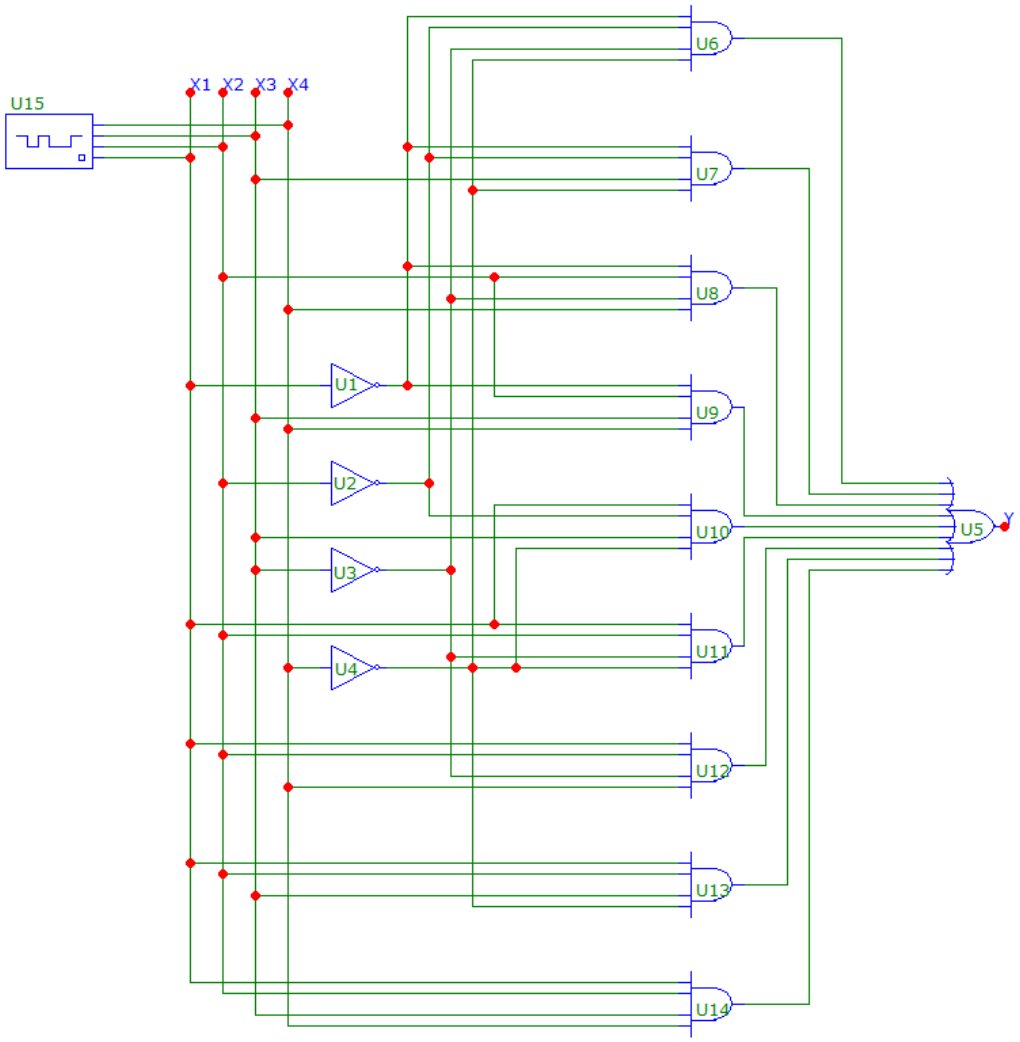


Рис. 4.2. Принципиальная комбинационная схема, построенная в СДНФ

Таблица 4.2

U15 (Stim4)
.define Count4
+0ns 0
+label=start
+100ns Incr By 1
+200ns goto start -1 times

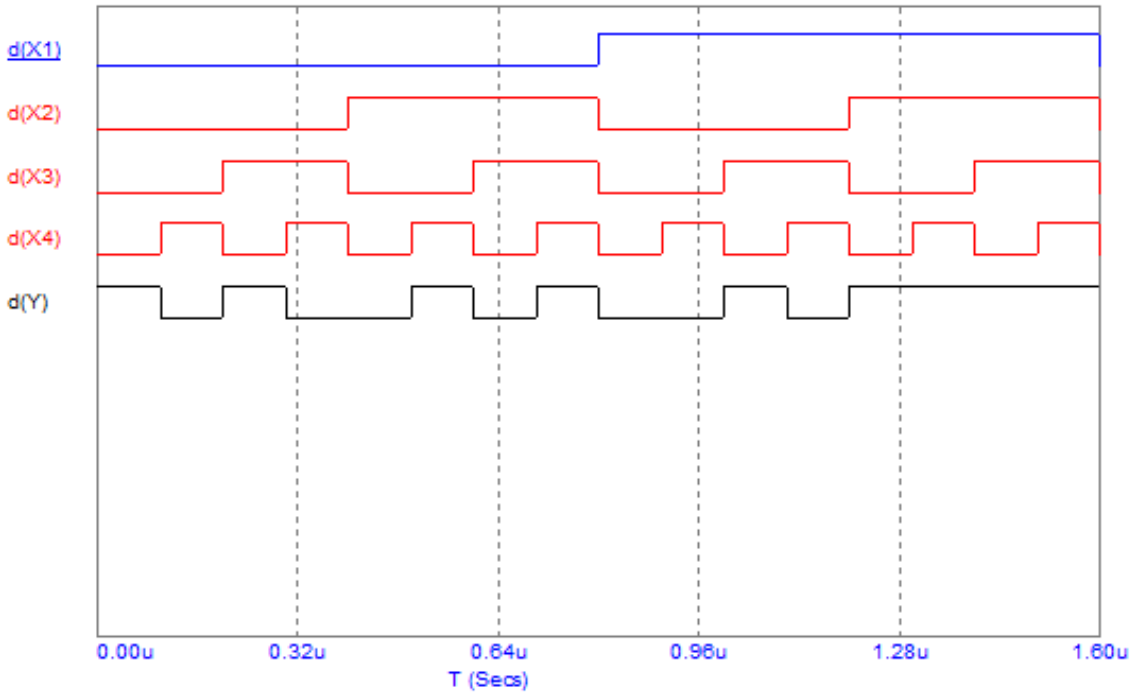


Рис. 4.3. Исследование схемы по СДНФ

Обратите внимание, что при $d(X_1) = 0$, $d(X_2) = 0$, $d(X_3) = 0$, $d(X_4) = 0$ и $d(Y) = 1$, что соответствует значению переключательной функции в таблице истинности (табл. 4.1). Аналогично схема выставляет на выходе логическое значение 1 при соответствующих значениях входных сигналов. Уровни кривой $d(Y)$ показывают, что схема собрана правильно.

Кроме СДНФ можно представить работу табл. 4.1 в СКНФ. СКНФ – это совершенная конъюнктивная форма представления логической переключательной функции. Обычно этой формой пользуются нечасто, но когда в таблице истинности мало нулевых значений в значениях выходной функции, то СКНФ использовать предпочтительней.

Для формирования СКНФ используют следующее правило. После построения таблицы истинности переключательной функции ищутся аргументы (входные значения), на которые функция принимает значение 0. Выписываются простые дизъюнкции (логические сложения) для такого набора исходных данных. Если в этот набор попадает переменная со значением 1, то это аргумент входит в дизъюнкцию с отрицанием. Дизъюнкции подобных наборов объединяются с помощью конъюнкции (логическое умножение) (табл. 4.3).

Составим логическую функцию в СКНФ по табл. 4.3.

$$Y = (X_1 + X_2 + X_3 + \overline{X}_4)(X_1 + X_2 + \overline{X}_3 + \overline{X}_4)(X_1 + \overline{X}_2 + X_3 + X_4)\times$$
$$\times(X_1 + \overline{X}_2 + \overline{X}_3 + X_4)(\overline{X}_1 + X_2 + X_3 + X_4)(\overline{X}_1 + X_2 + X_3 + \overline{X}_4)(\overline{X}_1 + X_2 + \overline{X}_3 + \overline{X}_4). \quad (4.2)$$

По функции 4.2 составим схему в Micro-CAP (рис. 4.4).

Используя настройки генератора U15, как на табл. 4.2 строим временные диаграммы логических сигналов контрольных точек на интервале 1.6u (рис. 4.5).

Таблица 4.3

X_1	X_2	X_3	X_4	X
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

При сравнении два графических исследования, изображённых на рис. 4.5 и 4.3, имеют идентичные значения. Что доказывает равнозначность СДНФ и СКНФ для формирования логической функции, отрабатывающей значения таблицы истинности переключательной функции.

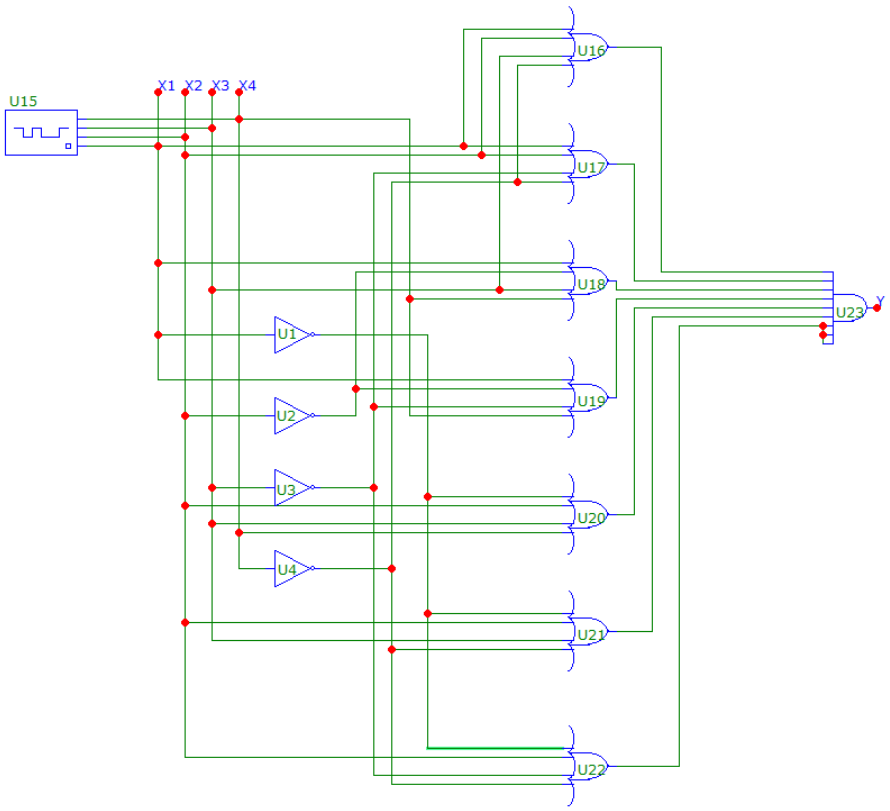


Рис. 4.4. Принципиальная комбинационная схема, построенная в СКНФ

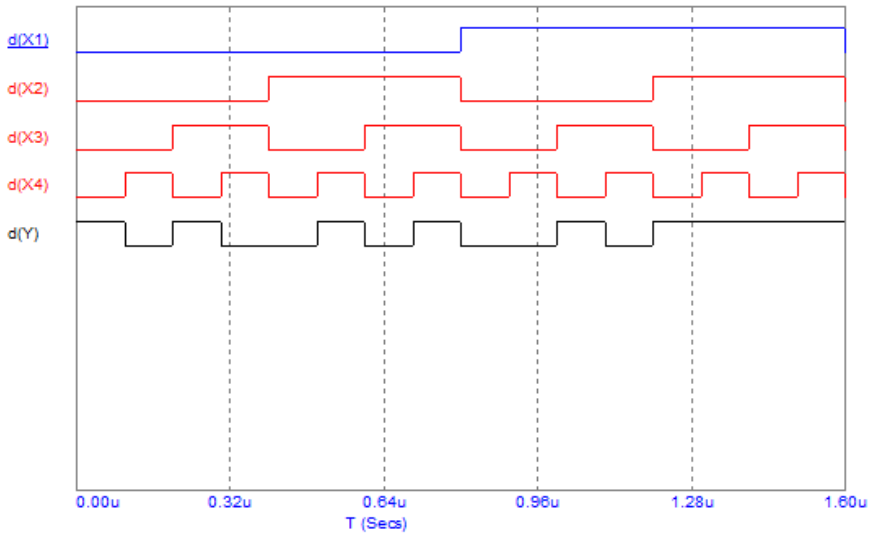


Рис. 4.5. Исследование схемы по СКНФ

На первый вид схемы, можно сказать, что задача выполнена. Но, как правило, такое построение схемы изобилует лишними элементами при соблюдении функционала. Схему необходимо оптимизировать. Уменьшение количества логических элементов в цифровой схеме удешевляет изделие, уменьшает потребление электроэнергии и увеличивает быстродействие схемы. Ведь для прохождения электрического сигнала через логический элемент требуется определённое время. А если цепь прохождения сигнала уменьшается за счёт уменьшения количества логических элементов, то и длительность прохождения сигнала по такой оптимизированной цепи уменьшается. Что поднимает быстродействие изделия в целом.

4.1. ЗАКОНЫ АЛГЕБРЫ ЛОГИКИ. МИНИМИЗАЦИЯ ЛОГИЧЕСКИХ ФУНКЦИЙ

Для оптимизации логических схем можно воспользоваться аксиомами (тождествами) и законами логических операций (табл. 4.4).

Таблица 4.4

Закон	Для **ИЛИ**	Для **И**
Аксиомы (тождества)	$0 + X = X$ $1 + X = 1$	$0 \cdot X = 0$ $1 \cdot X = X$
Идемпотентности	$X + X = X$	$X \cdot X = X$
Противоречия		$X \cdot \overline{X} = 0$
Исключения третьего	$X + \overline{X} = 1$	
Двойного отрицания	$\overline{\overline{X}} = X$	
Коммутативности	$X + Y = Y + X$	$X \cdot Y = Y \cdot X$
Ассоциативности	$X + Y + Z = X + (Y + Z)$	$X \cdot Y \cdot Z = X \cdot (Y \cdot Z)$
Дистрибутивности (распределительности)	$X + (Y \cdot Z) = (X + Y) \cdot (X + Z)$ $X \cdot (Y + Z) = (X \cdot Y) + (X \cdot Z)$ $X + Y \cdot Z = (X + Y)(X + Z)$	
Де Моргана (дуальности)	$\overline{X + Y} = \overline{X} \cdot \overline{Y}$ $\overline{X \cdot Y} = \overline{X} + \overline{Y}$	
Поглощения	$X + X \cdot Y = X$ $X \cdot (X + Y) = X$	

Склеивания	$X \cdot Y + X \cdot \overline{Y} = X$

Попробуем сделать оптимизацию, например, следующей записи логической формулы:

$$F = X_1X_2X_3\overline{X}_4 + X_1X_2X_3X_4. \quad (4.3)$$

Здесь мы можем вынести за скобки $X_1X_2X_3$ и получим запись в виде:

$$F = X_1X_2X_3\overline{(X}_4 + X_4). \quad (4.4)$$

Используя закон «Склеивания»

$$F = X_1X_2X_3. \quad (4.5)$$

Обратите внимание, что выражение (4.5) радикально отличается от исходного выражения (4.3). Как следствие такой оптимизации и сокращается схемная реализация логических выражений.

Используя основные законы логических операций для оптимизации формул (4.1) и (4.2) можно ощутимо сократить выражение логических функций, что даст ощутимое сокращение принципиальных схем переключательных функций.

Но оптимизация логических функций является творческим процессом и порой основывается на чисто эвристических подходах. Эффективным подспорьем в процессе оптимизации логических переключательных схем является использование, например, карт Карно.

4.1.1. ОПТИМИЗАЦИЯ ПЕРЕКЛЮЧАТЕЛЬНЫХ ФУНКЦИЙ. КАРТА КАРНО

Карты Карно были изобретены в 1952 г. Эдвардом В. Вейч'ем и усовершенствованы в 1953 г. Морисом Карно, физиком из «Бэлл Лабс».

Карта Карно – графический способ облегчения минимизации переключательных функций. Карно позволяет сгруппировать такие объединения (термы) из таблицы истинности переключательной функции, где легко применить операции попарного неполного склеивания и элементарного поглощения. Карты Карно можно рассматривать как определённую плоскую развертку *n*-мерного булева куба таблицы истинности переключательной функции.

Наибольшую эффективность карты Карно проявляют на булевых функциях с 4–5 переменными.

Рассмотрим пример карты Карно для нашего случая переключательной функции с табл. 4.1 истинности. В таблицу 4.5, где строки и столбцы проиндексированы кодами Грея, впишем значения нашей таблицы истинности. Обратите внимание на код Грея. Для данной кодировки соседний код отличается от текущего лишь изменением одного разряда.

Если мы будем минимизировать СДНФ, то расматриваем те клетки таблицы Карно, где расположены единицы (1). Для минимизации СКНФ нас будут интересовать нули (0). И интересовать нас будут не сами эти значения, а их объединения.

Правило формирования объединения для СДНФ:

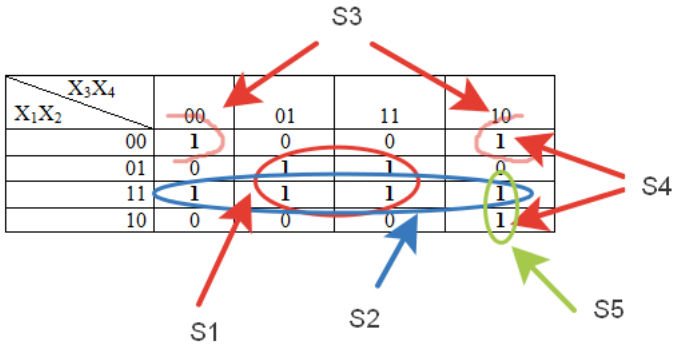
1. Объединяем смежные клетки, содержащие единицы в область так, чтобы одна область содержала 2^n (n целое число) клеток. В объединении возможно 1, 2, 4 или 8 единиц. И количество единиц в области должно быть максимальным. А количество объединений должно быть минимальным.
2. Контуры объединений могут пересекаться между собой.

Таблица 4.5

X_1X_2 \ X_3X_4	00	01	11	10
00	**1**	0	0	**1**
01	0	**1**	**1**	0

11	**1**	**1**	**1**	**1**
10	0	0	0	**1**

Таблица 4.6



3. В области не должно находиться клеток, содержащих нули.
4. Крайние столбцы являются соседними. Это распространяется и на крайние строки.
5. Отдельно стоящая единица – это контур. Его тоже необходимо учитывать.

Руководствуясь данными правилами выделим объединения в карте Карно (табл. 4.6). Получилось шесть объединений S1 – S5.

Выпишем каждое объединение и выполним оптимизацию.

$S1=\overline{X}_1X_2\overline{X}_3X_4+\overline{X}_1X_2X_3X_4+X_1X_2\overline{X}_3X_4+X_1X_2X_3X_4=\overline{X}_1X_2X_4+X_1X_2X_4=X_2X_4;$

$S2=X_1X_2\overline{X}_3\overline{X}_4+X_1X_2\overline{X}_3X_4+X_1X_2X_3X_4+X_1X_2X_3\overline{X}_4=X_1X_2\overline{X}_3+X_1X_2X_3=X_1X_2;$

$S3=\overline{X}_1\overline{X}_2\overline{X}_3\overline{X}_4+\overline{X}_1\overline{X}_2X_3\overline{X}_4=\overline{X}_1\overline{X}_2\overline{X}_4;$

$S4=X_1\overline{X}_2X_3\overline{X}_4+\overline{X}_1\overline{X}_2X_3\overline{X}_4=\overline{X}_2X_3\overline{X}_4;$

$S5=X_1\overline{X}_2\,X_3\overline{X}_4+X_1X_2X_3\overline{X}_4=X_1X_3\overline{X}_4.$

Сформируем переключательную функцию Y из дизъюнкции объединений S1 – S5.

$$Y=X_2X_4+X_1X_2+\overline{X}_1\overline{X}_2\overline{X}_4+\overline{X}_2X_3\overline{X}_4+X_1X_3\overline{X}_4. \quad (4.6)$$

Безусловно, что это не окончательный оптимизационный вид функции. Предлагаю самостоятельно доказать, что предложенную функцию можно ещё «ужать» до вида:

$$Y=\overline{X}_1\,\overline{X}_2X_4+X_2X_4+X_1X_2+X_1X_3\overline{X}_4. \quad (4.7)$$

По логической функции (4.6) строим схему из цифровых элементов (рис. 4.6).

Устанавливаем настройки генератора U10, как в табл. 4.2 и на интервале в 1.6u строим временные диаграммы контрольных точек (рис. 4.7).

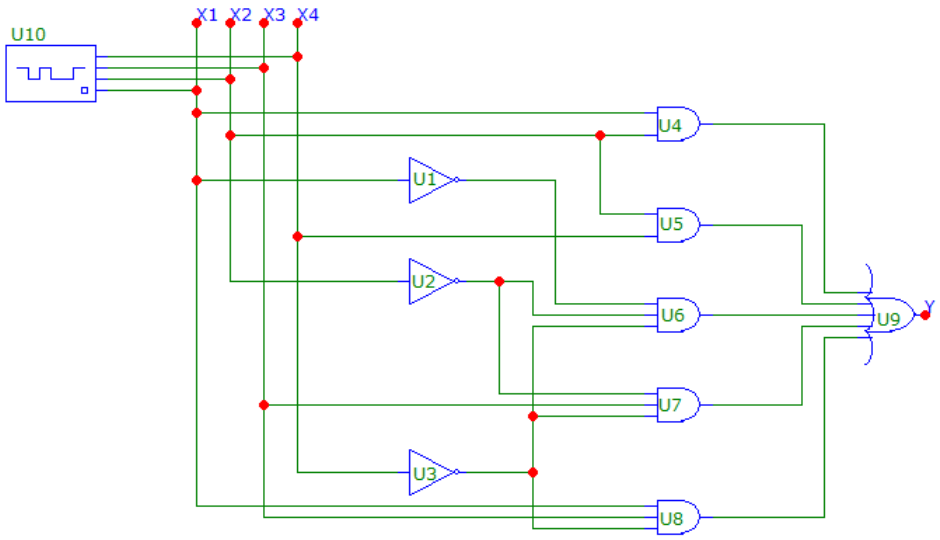


Рис. 4.6. Принципиальная оптимизированная схема

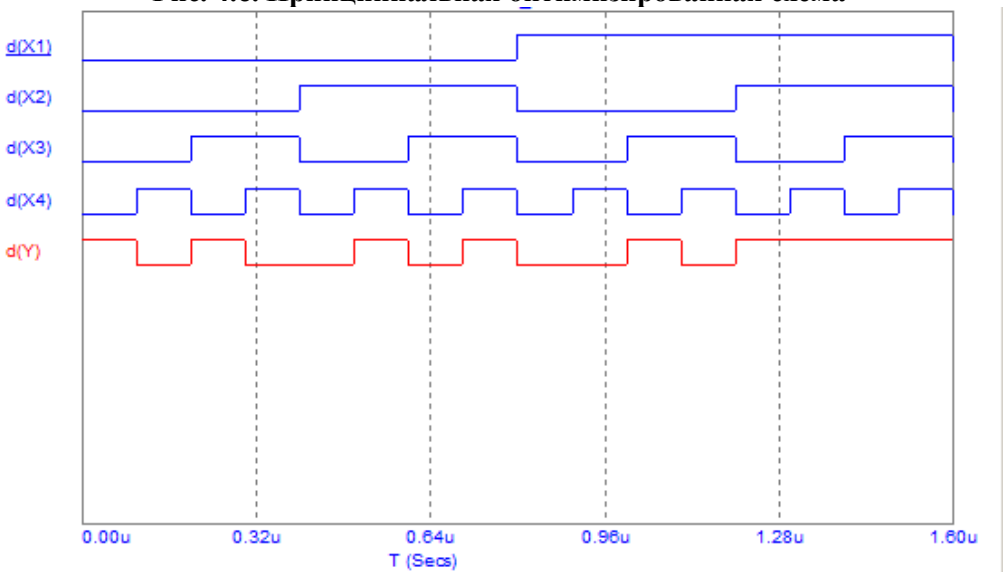


Рис. 4.7. Исследование схемы оптимизированной по таблице Карно СДНФ

Графики показали, что схема выполняет данные таблицы истинности 4.1.

Сравнивая две схемы, представленные на рис. 4.2 и 4.6, виден очевидный выигрыш по количеству логических элементов в последнем решении. Так в первом варианте неоптимизированной схемы необходимы: 4 инвертора (**НЕ**), 9 элементов **4И** и один 9-и входовой элемент **ИЛИ**. После оптимизации схема сократилась до 3 инверторов (**НЕ**), 5 элементов **2И** и одного 5-и входового элемента **ИЛИ**. Выигрыш по количественному составу элементной базы очевиден.

Если требуется рассмотреть карту Карно для логической функции, состоящей из пяти переменных, то разбивается список переменных на 2 и 3 или 3 и 2. И заполняется таблица, вид которой может быть представлен как табл. 4.7 или 4.8.

Дальнейшие действия с данными таблицами аналогичные, как для таблиц, заполненных по таблицы истинности переключательной функции четырёх переменных.

Таблица 4.7

X_1X_2 \ $X_3X_4X_5$	000	001	011	010	110	111	101	100
00								

01								
11								
10								

Таблица 4.8

$X_1X_2X_3$ \ X_4X_5	00	01	11	10
000				
001				
011				
010				
110				
111				
101				
100				

4.2. ДЕШИФРАТОРЫ И ШИФРАТОРЫ

В цифровой технике часто приходится преобразовывать двоичные коды в сигнал десятичного представления, т.е. дешифрировать входной код. Для этого существуют специализированные микросхемы, выполняющие функции дешифрирования входного сигнала. Функциональная схема дешифратора представлена на рис. 4.8.

Количество выходных сигналов дешифратора и входных сигналов шифратора равно количеству возможных состояний двоичного кода (входного кода у дешифратора и выходного кода у шифратора), т.е. 2^n, где n – разрядность двоичного кода. Микросхемы дешифраторов обозначаются на схемах буквами DC (от английского Decoder). В отечественных микросхемах дешифратор обозначается буквами ИД.

Электронная промышленность выпускает дешифраторы определённого набора разрядности входных и выходных сигналов. Различают дешифраторы с двумя разрядами на входе и с четырьмя выходами (2 – 4), три разряда на входе и восемь выходов (3 – 8), а также четыре входных разряда и шестнадцать выходов (4 – 16).

На рисунке 4.9 представлен пример подключения двоично-десятичного дешифратора ИД14 к генератору U1 (Stim2) для исследования основных функций дешифратора.

Обратите внимание, что A0 – младший входной разряд, а A1 – старший. Вход E – инверсный вход разрешения дешифрирования (вход стробирования). В данном случае на этот вход подан логический 0. А на практике, обычно, этот сигнал указывает, что все входные разряды установлены и можно их дешифрировать. Выходные сигналы у данной микросхемы, да и вообще у большинства дешифраторов инверсные. Это ввели не случайно. Если нам потребуется расширить дешифратор по входным разрядам и, как следствие, выходов, то обратная полярность на выходе поможет легко реализовать пирамидальные и каскадные структуры укрупненных дешифраторов. На рисунке 4.10 представлены временные диаграммы контрольных точек дешифратора 2 – 4. Только один выход бывает активный.

Давайте расширим дешифратор до 16 выходов при четырёх входах. И создавать такой дешифратор будем из дешифраторов 2 – 4. Для этого возьмем пять дешифраторов. Один дешифратор будет переключать остальные четыре по значению 3 и 4 входного разряда. Первые два входных разряда запараллелим поразрядно с соответствующими входами дешифраторов (рис. 4.11).

Настроим генератор тестовых сигналов (U3), как в табл. 4.9.

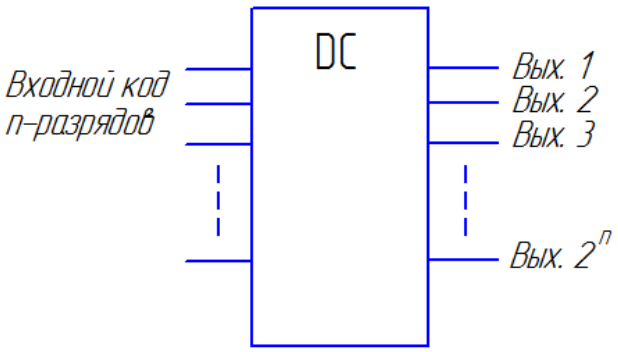


Рис. 4.8. Функциональная схема дешифратора

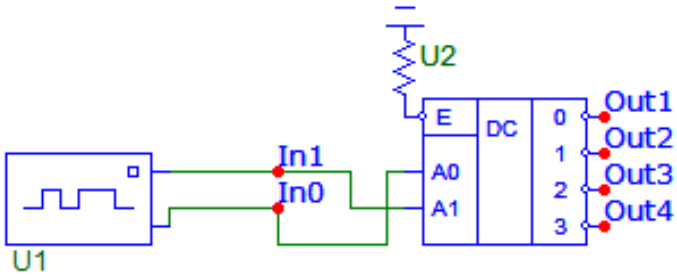


Рис. 4.9. Схема для исследования дешифратора 2-4 (ИД14)

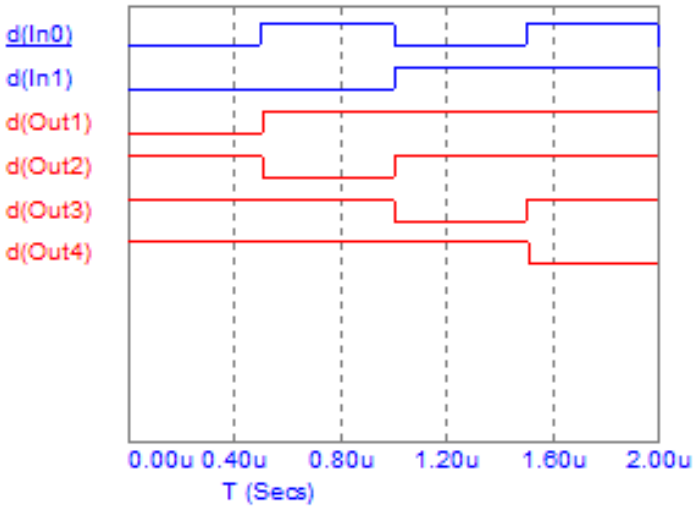


Рис. 4.10. Временные диаграммы работы дешифратора 2-4 (ИД14)

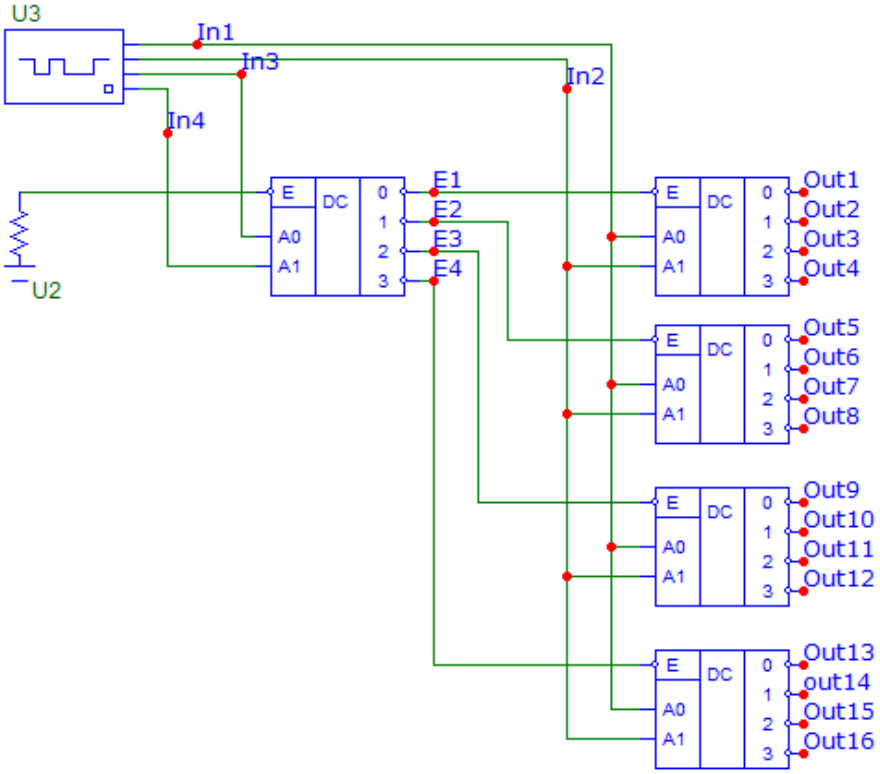


Рис. 4.11. Схема дешифратора 4-16

Таблица 4.9

U2 (Stim4)
.define Count4 +0ns 0 +label=start +10u Incr By 1 +20u goto start -1 times

Временные диаграммы контрольных точек на интервале 160u имеет вид (рис. 4.12). На графиках видно, как появляются одиночные сигналы на выходах дешифратора Out1 – Out16 согласно входного значения кода In1 – In4.

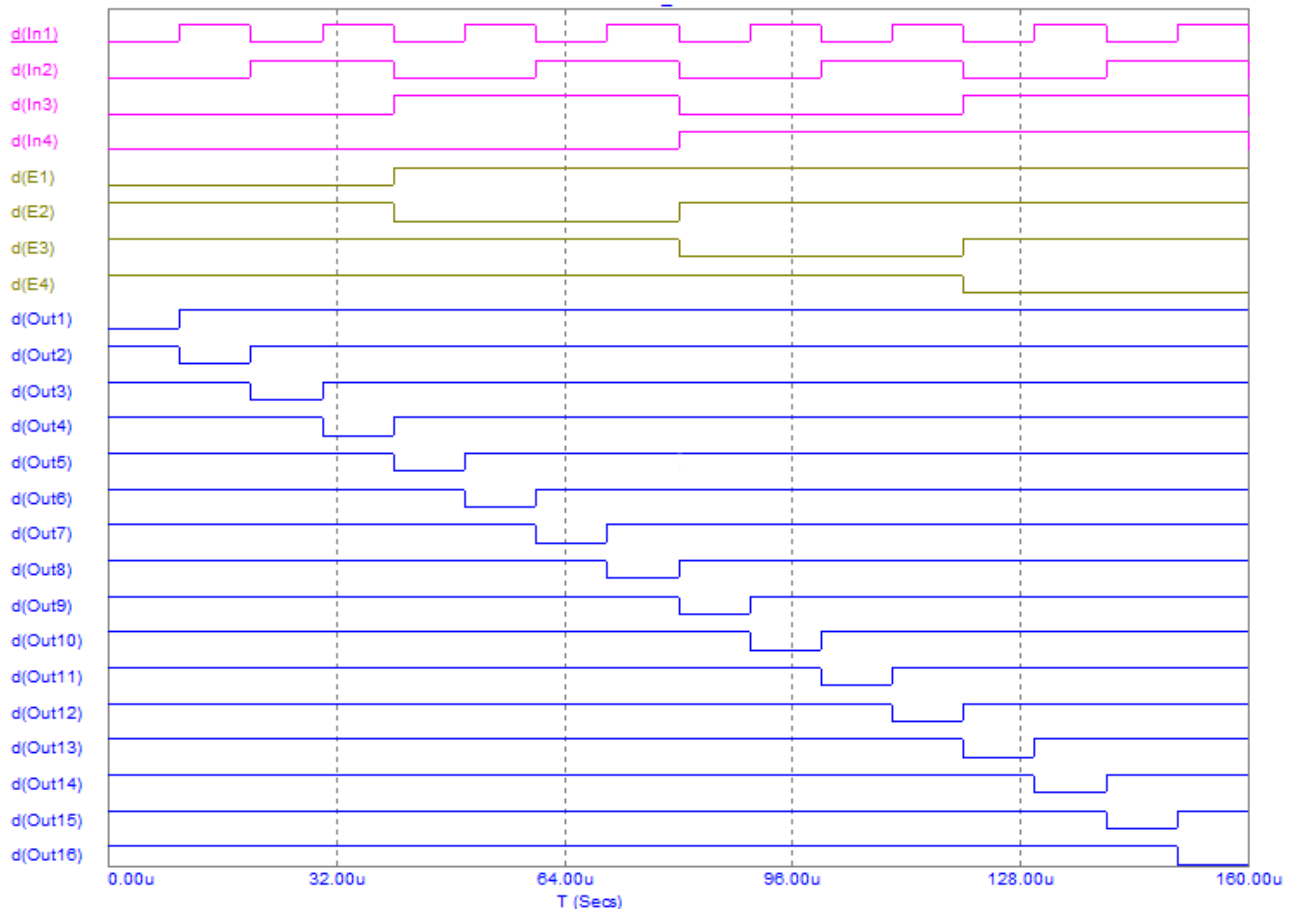


Рис. 4.12. Временные диаграммы работы дешифратора 4-16

На практике нет необходимости моделировать дешифраторы высокого порядка из малоразрядных дешифраторов. Промышленностью выпускаются уже готовые дешифраторы с организацией: 3 – 8, 4 – 10 и 4 – 16 (рис. 4.13). Но иногда не оказывается под рукой нужный дешифратор и поэтому необходимо знать, как можно моделировать укрупненные дешифраторы из дешифраторов с минимальным набором данных.

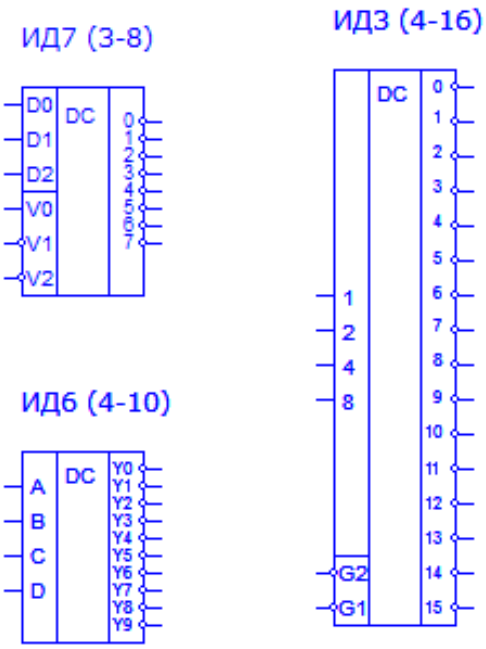


Рис. 4.13. Варианты дешифраторов

Дешифратор можно использовать совершенно в несвойственной для этой схемы задачи. Например, дешифратором можно заниматься перекоммутированием входного сигнала на несколько выходов. Одним словом, дешифратор может выполнять функции демультиплексора входных сигналов, который позволяет разделить входные сигналы, приходящие в разные моменты времени, на одну входную линию (мультиплексированные сигналы). При этом входы, например, 1, 2 и 4 (D0, D1 и D2) дешифратора используются в качестве адресных линий, определяющих, на какой выход переслать пришедший в данный момент входной сигнал. А один из управляющих входов V выступает в роли порта входного сигнала, который пересылается на заданный выход. Если у микросхемы имеется несколько стробирующих входов V, то оставшиеся входы V можно использовать в качестве разрешающих работу дешифратора (демультиплексора).

На рисунке 4.14 представлена схема использования дешифратора в качестве демультиплексора. Рассмотрим работу такого мультиплексора для передачи данных (In) на выход 2 (Out2) и 5 (Out5). Адрес активизации соответствующего выхода формируется на входах D0 – D2.

Пример работы такого демультиплексора можно увидеть на графиках сигналов в контрольных точках (рис. 4.15). Генератор U3 был настроен на активизацию двух адресов: 2 и 5. Генератор U4 выступает в качестве входного сигнала. Мы видим, как при поступлении сигнала 010 (2_{10}) на входы D0, D1 и D2 активизируется выход Out2. А при поступлении адресного кода 101 (5_{10}) активируется выход Out5 и на нём появляется входной сигнал In. Это и есть классическая схема работы демультиплексора.

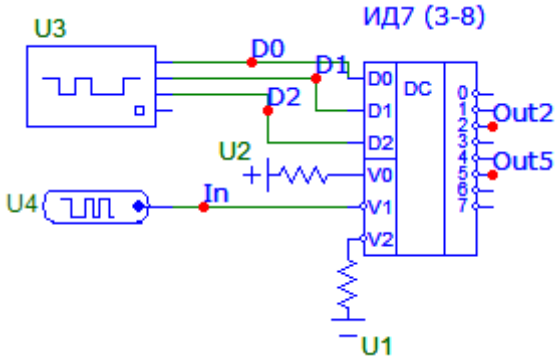


Рис. 4.14. Включение дешифратора как демультиплексора

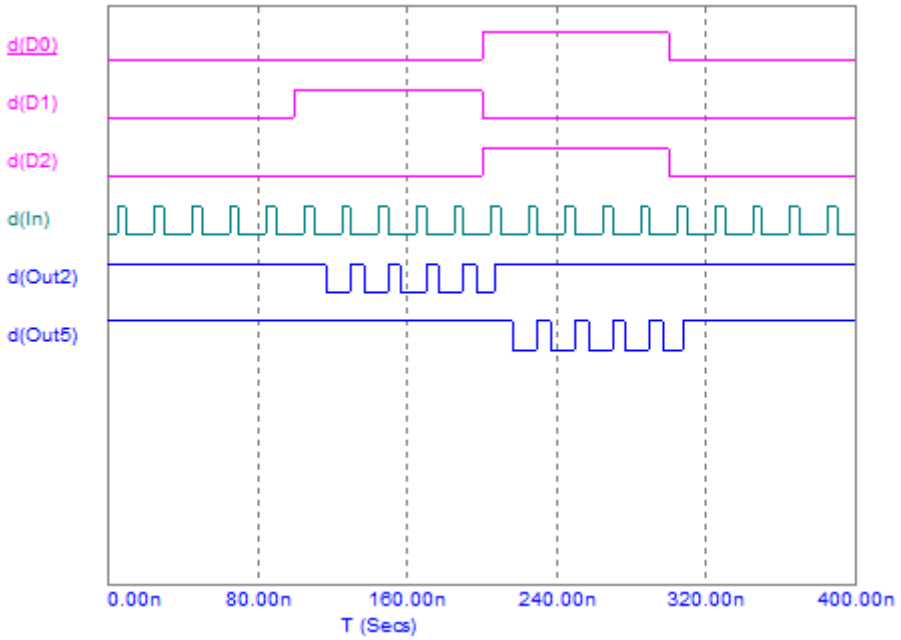


Рис. 4.15. Работа дешифратора-демультиплексора

Функция шифратора – обратная дешифратора – преобразование десятичного входного кода в его эквивалент двоичного кода на выходе (рис. 4.16).

Микросхемы шифраторов обозначаются на схемах буквами CD (от английского Coder). В отечественных сериях микросхем шифратор имеет название ИВ. Рассмотрим несколько шифраторов с разной организацией. На рисунке 4.17 изображены шифраторы ИВ1 и ИВ3.

Таблица истинности работы шифратора ИВ1 представлена в табл. 4.8.

Обратите внимание, что входы и выходы, как и управляющий сигнал поступают в инверсном виде, т.е. низким уровнем. Инверсный вход EI предназначен для разрешения шифрации входных данных. Инверсный выход CS (GS) значением 0 указывает, что поступил входной код на дешифрацию, при этом на входе EI должно быть разрешение (0).

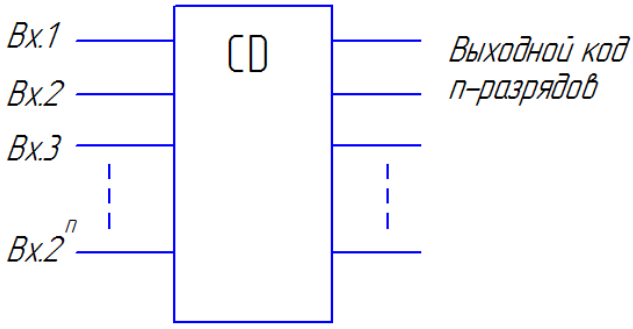


Рис. 4.16. Функциональная схема шифратора

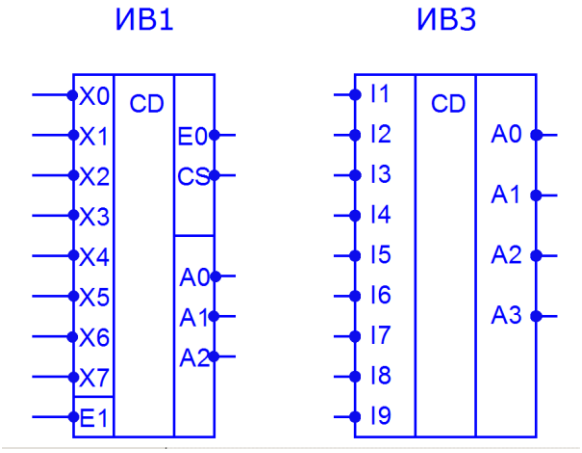


Рис. 4.17. Шифраторы ИВ1 (7-3) и ИВ3 (9-4)

Таблица 4.8.

E1	X7	X6	X5	X4	X3	X2	X1	X0	GS	A2	A1	A0	E0
1	x	x	x	x	x	x	x	x	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	**0**	0	**1**	**1**	**1**	1
0	1	1	1	1	1	1	**0**	x	0	**1**	**1**	**0**	1
0	1	1	1	1	1	**0**	x	x	0	**1**	**0**	**1**	1
0	1	1	1	1	**0**	x	x	x	0	**1**	**0**	**0**	1
0	1	1	1	**0**	x	x	x	x	0	**0**	**1**	**1**	1
0	1	1	**0**	x	x	x	x	x	0	**0**	**1**	**0**	1
0	0	**0**	x	x	x	x	x	x	0	**0**	**0**	**1**	1
0	**0**	x	x	x	x	x	x	x	0	**0**	**0**	**0**	1

Инверсный выход E0 становится активным (0), если разрешено шефрирование (EI = 0) и отсутствует входной код. Это удобно использовать для управления другими шифраторами, когда строится их каскадное включение для расширения входного кода.

Если на вход появятся сразу несколько сигналов, то будет отрабатываться только старший номер. Остальные сигналы будут игнорироваться. Такое свойство шифратора называется приоритетным.

Соберём схему для исследования работы шифратора ИВ1 (западный аналог 74LS148). Для примера, подадим на вход I1 и I5 отрицательные сигналы в разное время и убедимся, что на выходе шифратора будут сформированы соответствующие коды в двоичной системе счисления. Для разрешения шифрирования подадим на вход EI логический ноль (U3). В Micro-CAP существуют специальные элементы, моделирующие постоянный отрицательный сигнал (Pullup) и положительный (Pulldown) (рис. 4.18).

На не занятые в схеме входы шифратора ИВ1 подаем положительный постоянный уровень (U4). Настраиваем генераторы U1 и U2, как в табл. 4.9.

Готовую схему можно увидеть на рис. 4.19.

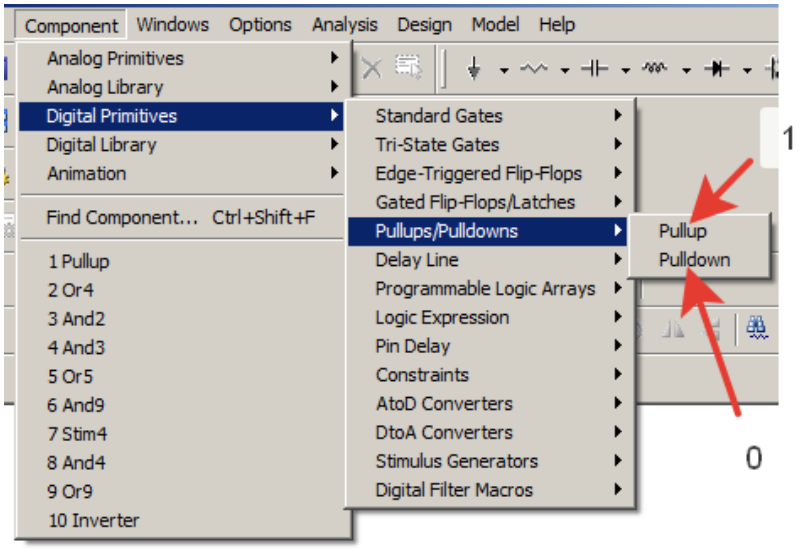


Рис. 4.18. Выбор констант логических уровней в Micro-CAP

Таблица 4.9

U1	U2
.define _I1 +0u 1 +10u 0 +20u 1	.define _I5 +0u 1 +25u 0 +35u 1

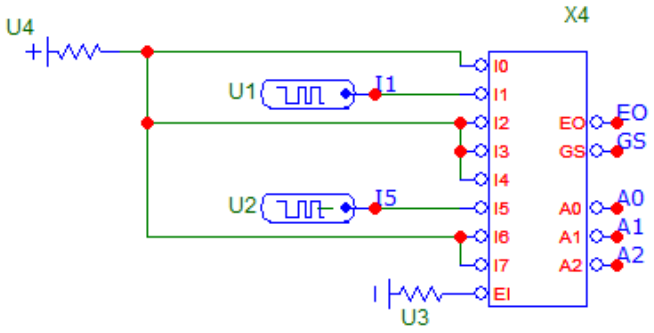


Рис. 4.19. Схема для исследования шифратора ИВ1

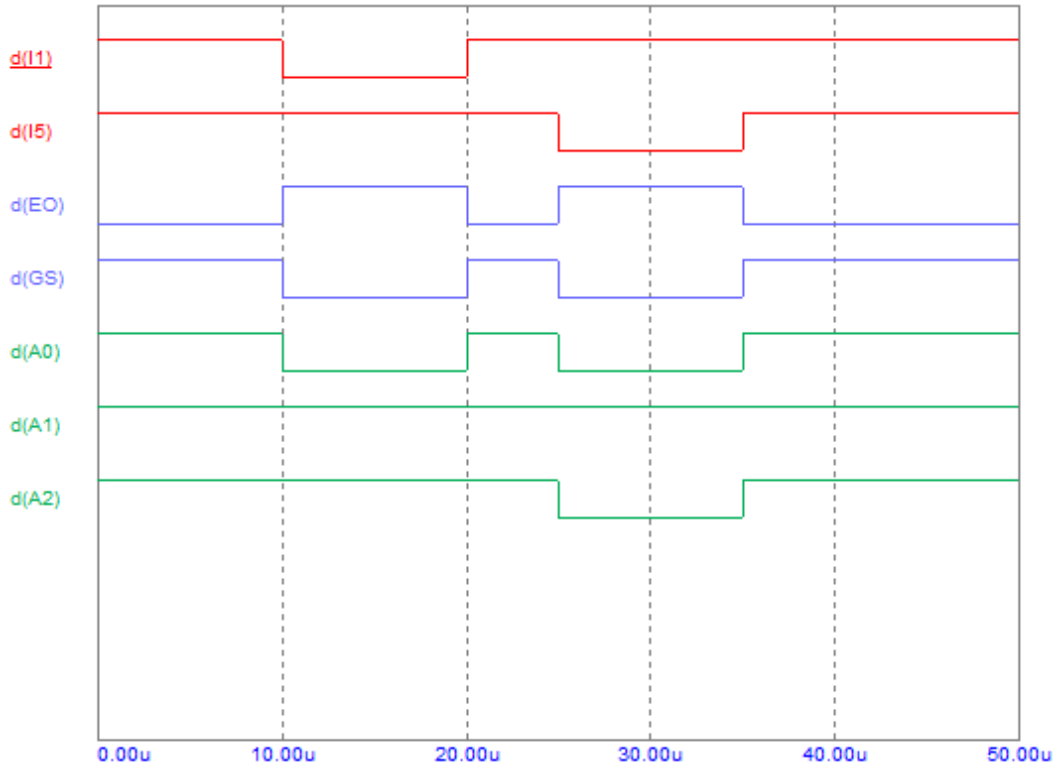


Рис. 4.20. Исследования шифратора ИВ1

Во временном интервале 50u строим временные диаграммы интересующих нас контрольных точек (рис. 4.20).

4.3. МУЛЬТИПЛЕКСОРЫ И ДЕМУЛЬТИПЛЕКСОРЫ

Мультиплексор – это цифровое устройство, у которого имеется несколько входов данных, один выход и несколько входов управления. В зависимости от состояния входов управления мультиплексор коммутирует входной сигнал на выход.

Абстрактно мультиплексор можно представить, как на рис. 4.21. Где D0-D3 – входы мультиплексора, Y – выход, а A0-A1 – адрес активного входа. Количество входов мультиплексора определяет канальность этого устройства. Так, если у мультиплексора 4 входа данных, то такой мультиплексор называется четырёх канальный. У мультиплексора с 16 входами название – 16-тиканальный.

В отечественных сериях микросхем мультиплексоры обозначаются аббревиатурой КП, а на схемах MS. На рисунке 4.22 представлены два вида мультиплексоров КП1 (74150) и КП5 (74HC152).

Из ОГУ мультиплексоров интуитивно понятны назначение входов и выходов. У КП1 вход С предназначен для разрешения коммутирования входного сигнала на выход. У мультиплексора КП5 такого разрешения нет.

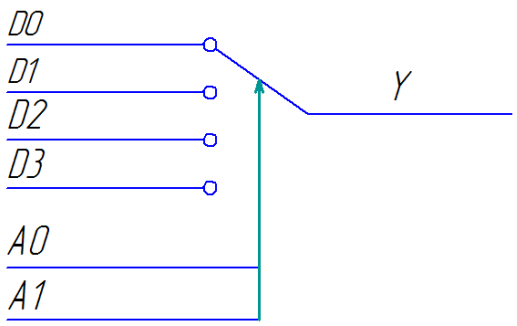


Рис. 4.21. Упрощенное представление мультиплексора

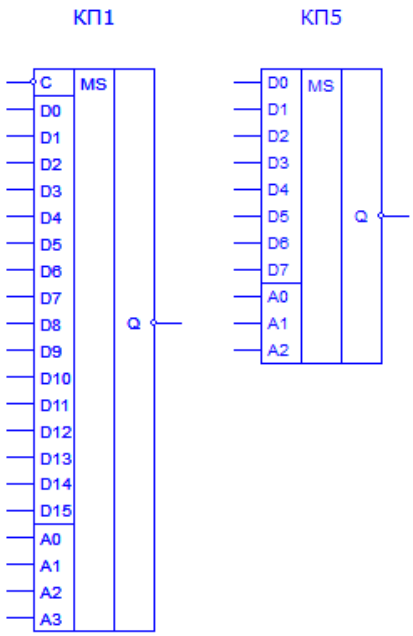


Рис. 4.22. Мультиплексоры КП1 и КП5

Если задаться целью восстановить внутреннюю функцию мультиплексора, например, КП5, то она выглядит так:

$$Q = \overline{A0} \cdot \overline{A1} \cdot \overline{A2} \cdot D0 + A0 \cdot \overline{A1} \cdot \overline{A2} \cdot D1 + \overline{A0} \cdot A1 \cdot \overline{A2} \cdot D2 + A0 \cdot A1 \cdot \overline{A2} \cdot D3 +$$
$$+\overline{A0} \cdot \overline{A1} \cdot A2 \cdot D4 + A0 \cdot \overline{A1} \cdot A2 \cdot D5 + \overline{A0} \cdot A1 \cdot A2 \cdot D6 + A0 \cdot A1 \cdot A2 \cdot D7$$

Восстановить принципиальную схему по такой функции очень просто (рис. 4.23). Поэтому мультиплексоры легко моделировать из элементов простой логики. Но, конечно же, лучше этим не заниматься, а брать готовые микросхемы мультиплексоров, которые занимают гораздо меньше места, нежели мультиплексоры, собранные из простой логики.

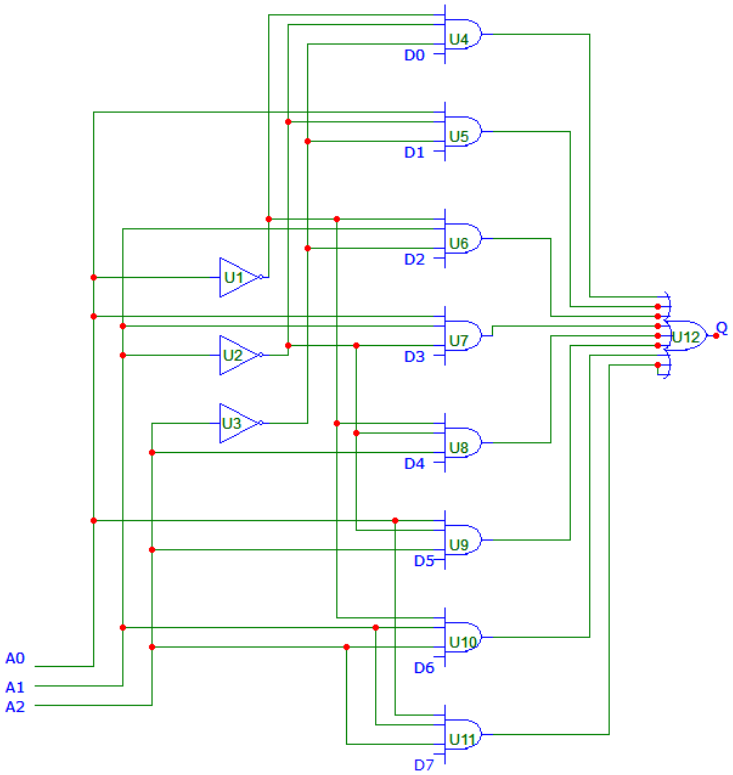


Рис. 4.23. Моделирование мультиплексора КП5

Рассмотрим работу мультиплексора, например, КП5 (рис. 4.24). Допустим, имеются два сигнала, моделируемых генераторами U1 и U2. И пусть первый сигнал D1 поступает на вход D1 мультиплексора, а сигнал D4 на одноименный вход D4. Для активизации канала передачи данных будем использовать генератор U3. Настроим его так, что с 1 микросекунды (1u) на выходе генератора будет сформирован код 0001 (1_{10}) и этот код продержится до 5u, после чего на выходе выставится 0000 (0_{10}). С 6u до 10u на выходе формируется код 0100 (4_{10}).

На диаграмме d(Q) видно, что входные сигналы прошли на выход мультиплексора согласно адресации входов.

На временном интервале в 12u строим временные диаграммы контрольных точек схемы (рис. 4.25).

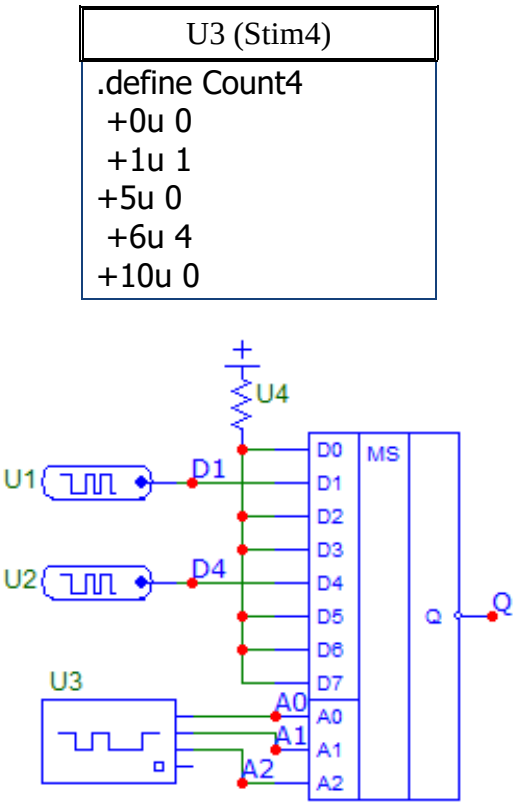


Рис. 4.24. Схема для исследования работы мультиплексора КП5

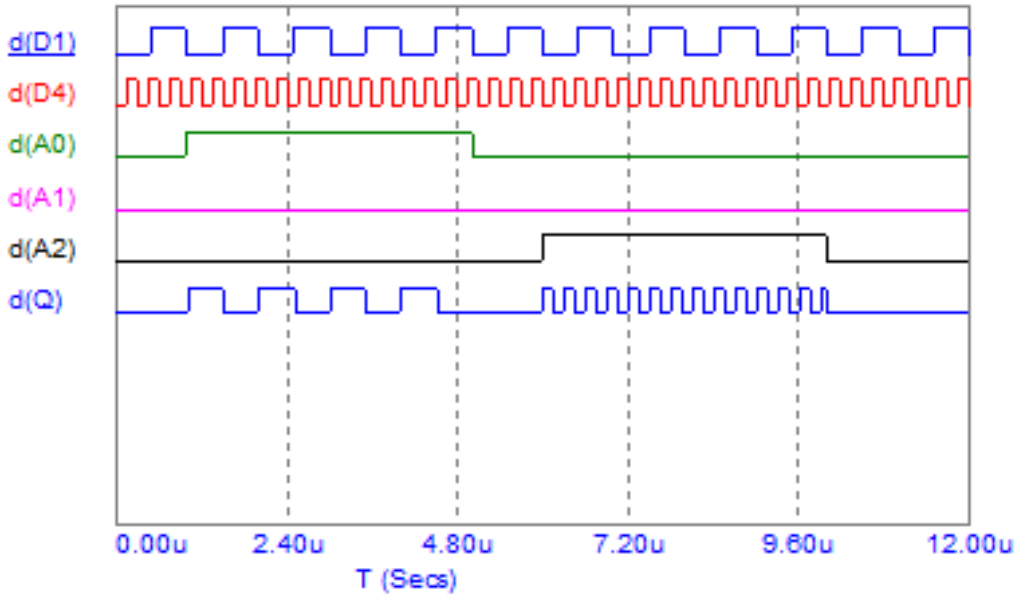


Рис. 4.25. Работа мультиплексора КП5 с выборкой двух адресов D1 и D4

4.4. КОМПАРАТОРЫ КОДОВ

В схемотехнике, когда требуется сравнивать между собой два кода применяют специальные микросхемы – компараторы. У этих микросхем, кроме входов, куда заводятся сравниваемые коды, имеются выходы, где формируются сигналы результата сравнения. В УГО компаратор обозначается знаком «= =». Рассмотрим отечественный 4-х разрядный компаратор СП1 (рис. 4.26).

Помимо восьми входов для сравниваемых кодов (два 4-х разрядных кода, обозначаемых A0...A3 и B0...B3), компаратор СП1 имеет три управляющих входа для наращивания разрядности (A > B, A < B, A = B) и три выхода результирующих сигналов (A > B, A < B, A = B). Таблица истинности компаратора СП1 представлена в табл. 4.10.

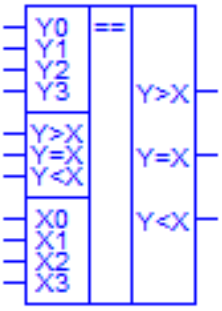


Рис. 4.26. УГО компаратора СП1

Таблица 4.10

Входы сравниваемых кодов				Входы наращивания			Выходы		
A3,B3	**A2,B2**	**A1,B1**	**A0,B0**	**A > B**	**A<B**	**A = B**	**A > B**	**A<B**	**A = B**
A3 > B3	X	X	X	X	X	X	1	0	0
A3<B3	X	X	X	X	X	X	0	1	0
A3 = B3	A2 > B2	X	X	X	X	X	1	0	0
A3 = B3	A2<B2	X	X	X	X	X	0	1	0
A3 = B3	A2 = B2	A1 > B1	X	X	X	X	1	0	0
A3 = B3	A2 = B2	A1<B1	X	X	X	X	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 > B0	X	X	X	1	0	0
A3 = B3	A2 = B2	A1 = B1	A0<B0	X	X	X	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	1	0	0	1	0	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	0	1	0	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	X	X	1	0	0	1
A3 = B3	A2 = B2	A1 = B1	A0 = B0	1	1	0	0	0	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	0	0	0	1	1	0

Для каскадного наращивания компараторов применяют следующее схемное решение:

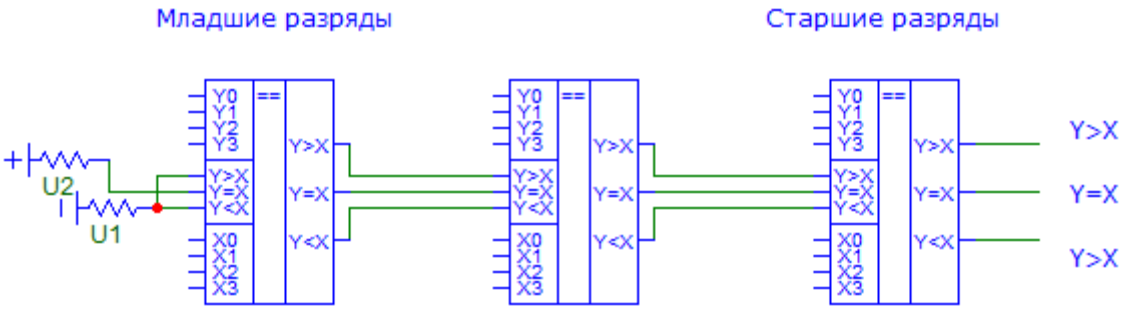


Рис. 4.27. Каскадное расширение компараторов СП1

Давайте попробуем смоделировать внутреннюю схему четырёхразрядного компаратора и тем самым разберёмся, как он работает на уровне простой логики.

Для начала рассмотрим работу простого одноразрядного компаратора (рис. 4.28).

Таблица истинности такого компаратора представлена в табл. 4.11.

По данным таблицы составим логические функции выходов компаратора.

$$Out(A > B) = A \cdot \overline{B};$$
$$Out(A < B) = \overline{A} \cdot B;$$
$$Out(A = B) = \overline{A} \cdot \overline{B} + A \cdot B = A \oplus B = \overline{A \cdot \overline{B} + \overline{A} \cdot B}.$$

Теперь можем перевести с языка формул в цифровую схему (рис. 4.29).

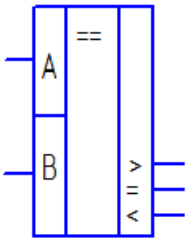


Рис. 4.28. Одноразрядный компаратор

Таблица 4.11

A	B	A > B	A = B	A < B
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

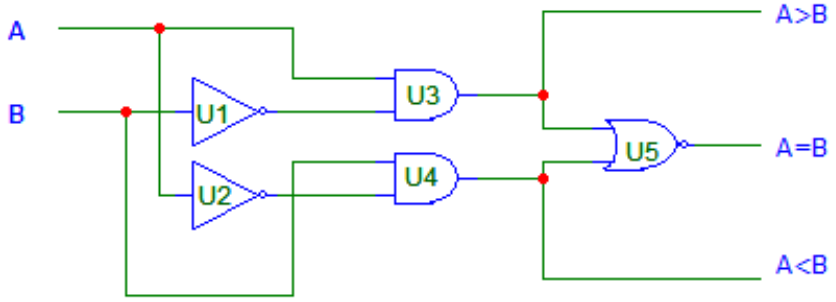


Рис. 4.29. Схема сравнения двух одноразрядных сигналов A и B

После того, как мы разобрались с работой одноразрядного компаратора, можем приступить к построению схемы компаратора четырёхразрядных сигналов. Но в данном учебном пособии ограничимся построением компаратора трёхразрядных сигналов. А четырёхразрядный компаратор предлагаем построить самостоятельно (готовую схему показать преподавателю). Итак, на рис. 4.30 представлена схема трёхразрядного компаратора. В этой схеме используются готовые схемы одноразрядных компараторов (рис. 4.28) в трёх местах – по каждому разряду. В схеме A0 и B0 это младшие разряды, а A2 и B2 – старшие. Следует заметить, что одноразрядный компаратор – это несуществующая микросхема. Мы сами вели в рассмотрение УГО одноразрядного компаратора для упрощения изложения дальнейшего материала дисциплины.

Одно из основных применений компараторов кодов состоит в селектировании входных кодов. Селектирование входного кода это регистрация факта – поступил определённый код или нет. Для решения подобной задачи нет необходимости определять, какой из поступивших кодов больше или меньше. Поэтому промышленность для таких целей выпускает компараторы с одним выходом, где появляется активный сигнал только при поступлении на вход одинаковых кодов. Пример такого компаратора SN74ALS521 (отечественный аналог – КР559СК1) для сравнения двух 16-ти разрядных слов (рис. 4.31).

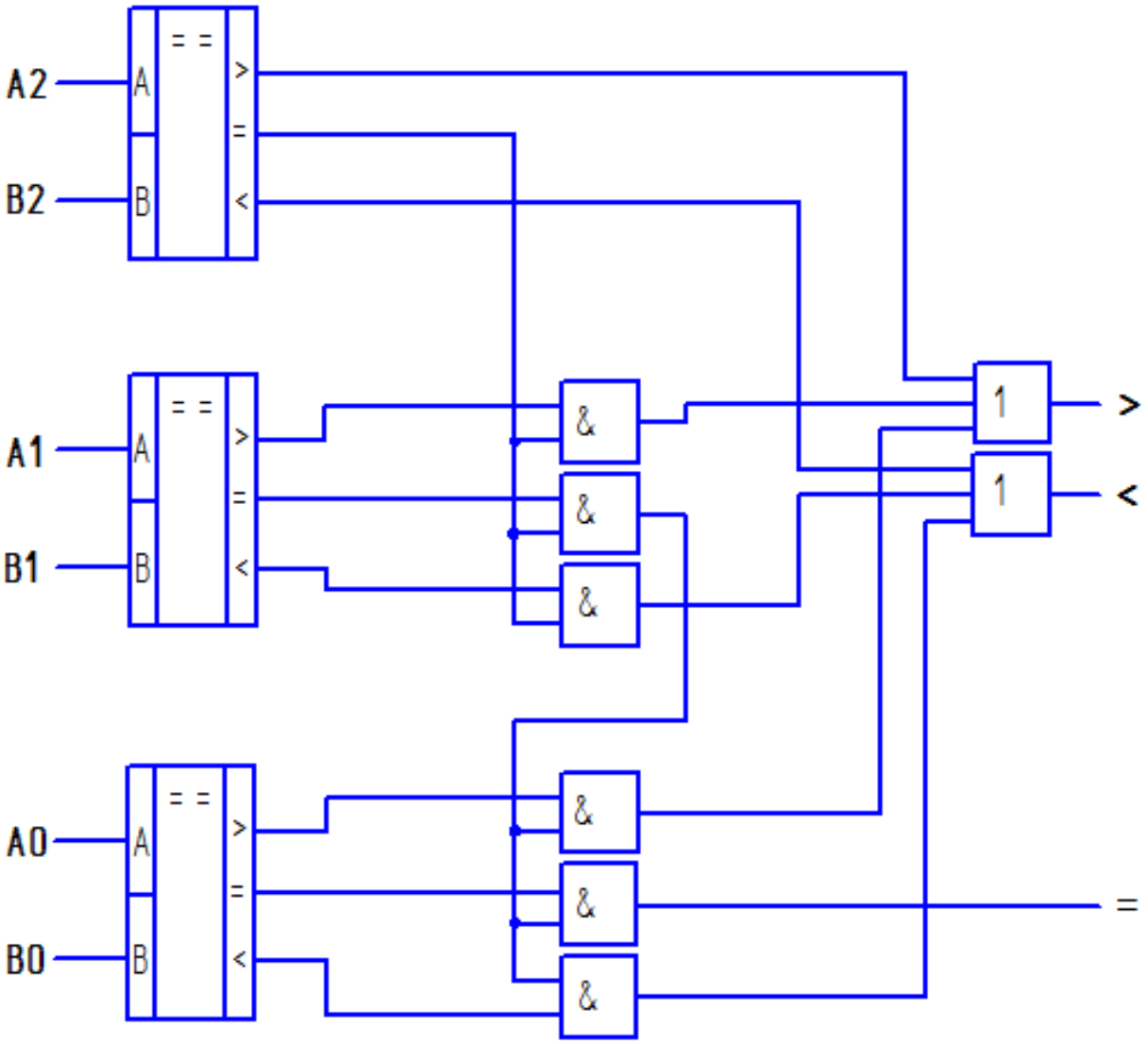


Рис. 4.30. Схема сравнения двух трёхразрядных сигналов A и B

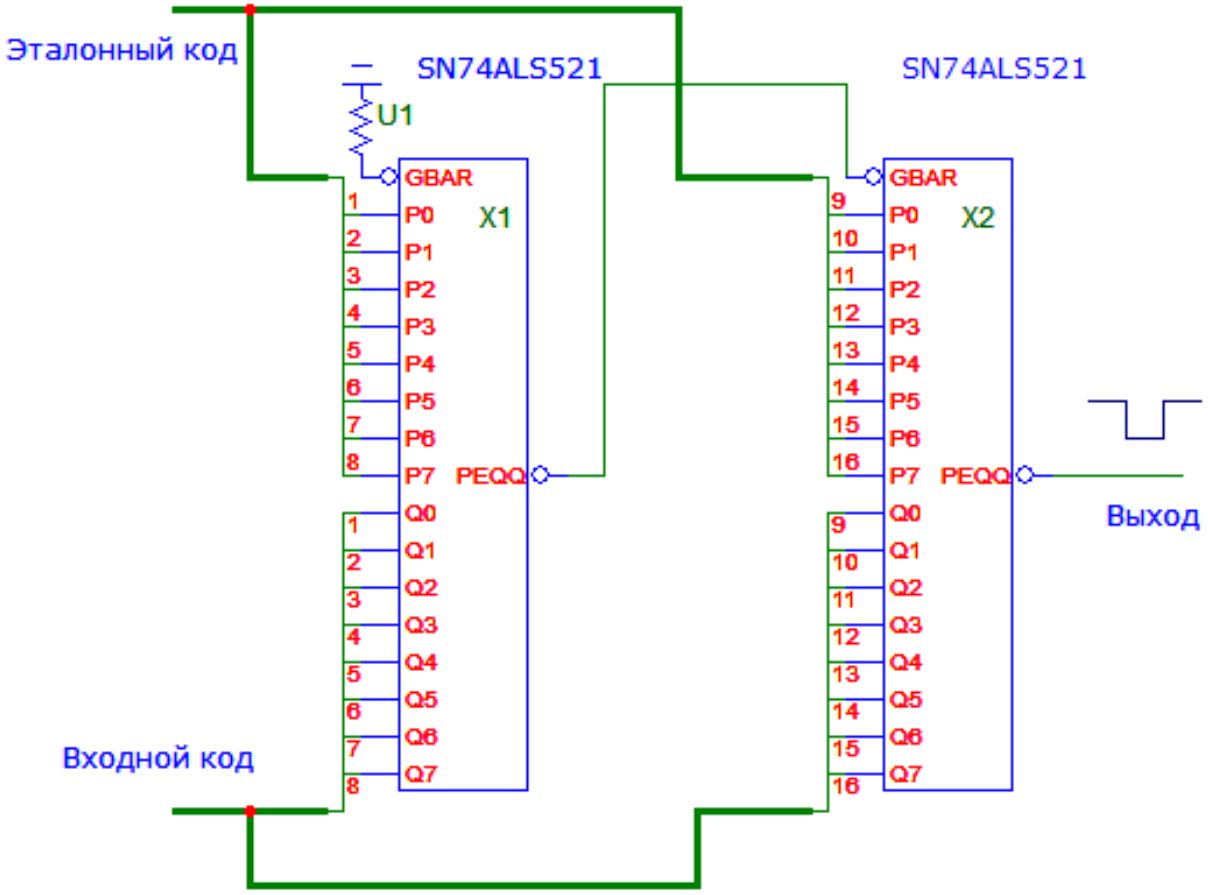


Рис. 4.31. Селектирование 16-разрядных кодов

4.5. СУММАТОРЫ

Микросхемы сумматоров предназначены для арифметического суммирования двух входных двоичных кодов. Например, если один входной код – 5 (0101), а второй – 2 (0010), то суммарный код на выходе будет 7 (0111). Для суммирования чисел, разрядность суммы которых превосходит разрядность исходных величин, в сумматорах предусмотрен специальный дополнительный разряд (старший) – выход переноса.

УГО сумматора обозначается аббревиатурой SM. Отечественные микросхемы сумматоров обозначают – ИМ.

По количеству суммируемых разрядов, сумматоры бывают: 2-х разрядные и 4-х разрядные (рис. 4.32).

Кроме входных разрядов суммирования, сумматоры имеют вход расширения (вход переноса) C. Это дополнительный вход необходим для объединения нескольких сумматоров с целью увеличения разрядности суммирования. Если на этот вход приходит единица, то выходная сумма увеличивается на единицу, если же приходит нуль, то выходная сумма не увеличивается. При использовании единственной микросхемы суммирования на её вход расширения C необходимо подать «0».

Сумматор может вычислять и разность входных кодов (рис. 4.33). В качестве сумматора была взята из Digital Library (Micro-CAP) микросхема 74AC283. Для этого вычитаемое число необходимо поразрядно проинвертировать и подать на входы сигнала B, а на вход переноса CIN подать единичный сигнал, что соответствует добавлению 1 к общей сумме. А это можно истолковать, как добавление 1 к младшему разряду обратному коду числа B. Такое видоизменение прямого кода называется дополнительный код обратного кода.

Рассмотрим каскадное расширение сумматоров, например, до 8 разрядов. На рисунке 4.34 видно, как просто взаимодействуют два 4-х разрядных сумматоров (74AC283) для образования одного 8-и разрядного. Обратите внимание, что к результирующему 8-и разрядному коду

A+B добавился дополнительный 9-й разряд (P). Это разряд, где может появиться логическая 1 в случае переполнения 8-и разрядного кода. Аналогично можно построить сумматор на большее количество разрядов.

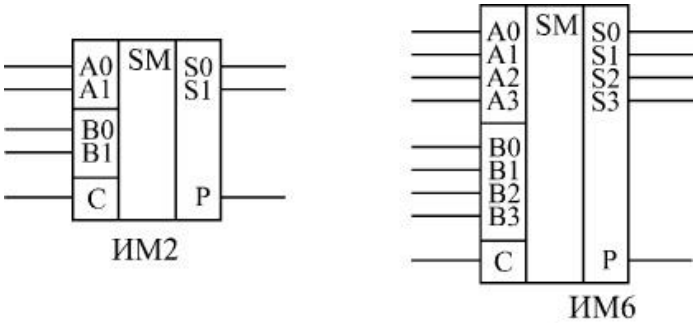


Рис. 4.32. 2-х разрядный (ИМ2) и 4-х разрядный (ИМ6) сумматоры

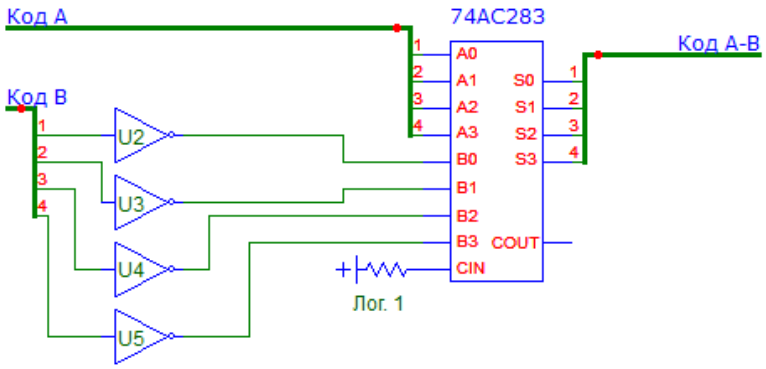


Рис. 4.33. Включение 4-х разрядного сумматора для вычитания A-B

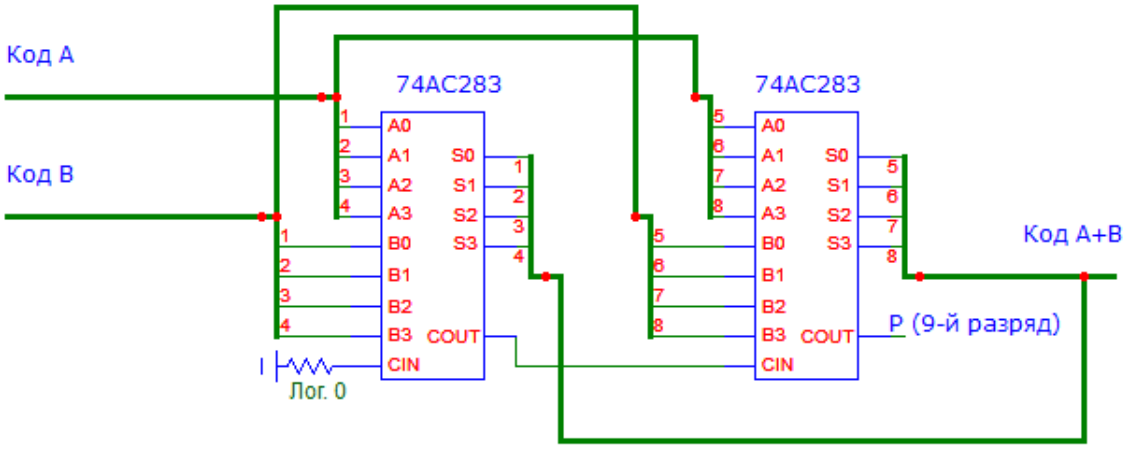


Рис. 4.34. Каскадирование сумматоров для увеличения разрядности

5. ТРИГГЕРЫ

Триггеры являются простейшими представителями цифровых микросхем, имеющих внутреннюю память. В отличие от простой логики и комбинационных схем, где выходной сигнал однозначно определяется входными сигналами, в устройствах, имеющих память, выходной сигнал зависит не только от наличия входных сигналов, но и от последовательности их появления. Благодаря такой логике работы устройств с памятью, можно строить более интеллектуальные схемы с более сложной логикой работы. Микросхемы с внутренней памятью в зависимости от их организации называются последовательными или параллельными, в отличие от комбинационных микросхем.

Триггер предназначен для хранения одного бита информации. На триггерах можно строить более сложные структуры хранения данных – регистры. Регистр – хранилище многобитовой информации записанной в двоичном коде.

Триггеры и регистры сохраняют свою память только до тех пор, пока на них подаётся напряжение питания, поэтому такая память называется оперативная. После выключения питания и его последующего включения триггеры и регистры могут перейти в случайное состояние по своим выходам. Но это произойдет лишь в том случае, если не предусмотрены схемные решения для корректной инициализации устройств памяти. Это необходимо учитывать при проектировании схем, где используются устройства типа триггеры или регистры.

В основе любого триггера лежит схема из двух логических элементов, у которых выходы одних элементов соединены со входами других логических элементов. Такая обратная связь выходов со входами называется положительной обратной связью. В результате подобного включения схема может находиться в одном из двух устойчивых состояний, причём находиться сколь угодно долго, пока на неё подано напряжение питания.

Различают RS-триггер, D-триггер, T-триггер и JK-триггер. Рассмотрим каждый триггер в отдельности.

5.1. RS-ТРИГГЕР

Пример триггерной ячейки на двух логических элементах **2И-НЕ** представлен на рис. 5.1. Это классический RS-триггер. В схеме имеются два инверсных входа: **–R** – сброс (от английского Reset), и **–S** – установка (от английского Set), а также два выхода: прямой выход **Q** и инверсный выход **–Q** ($\overline{Q}$). Название триггера состоит из начальных букв его основных функций **S**et и **R**eset (установка и сброс).

Таблица истинности триггера на логике **2И-НЕ** представлена в табл. 5.1.

В этой схеме значения сигналов для изменения состояния RS-триггера являются отрицательные сигналы – лог.0. На рисунке 5.2 представлены графики исследования работы триггера в среде Micro-CAP. На графиках мы видим, что до 50 ns, когда на входах –S и –R находятся положительные сигналы – лог.1, то на выходах Q и $\overline{Q}$ находится неопределённость по уровням. И как только на входе –R появляется активный сигнал (лог.0), то выход Q переходит в устойчивое состояние – лог. 0. После этого, многократное изменение состояния входа – R не влияет на состояние выходов триггера. На 130 ns поступает сигнал (лог.0) на вход S и выход Q взводится в единичное состояние (лог. 1). Сигнал –S уходит со входа схемы на 150 ns, но выход RS-триггера Q не меняет состояния. Триггер запомнил своё состояние. И только вновь появившийся сигнал на 160 ns сбрасывает выход триггера в состояние лог. 0. Так работает RS-триггер.

Внимание: – для правильной работы схемы 5.1. отрицательные сигналы (лог.0) должны поступать на её входы не одновременно.

RS-триггер можно поострить и из элементов **2ИЛИ-НЕ** (рис. 5.3).

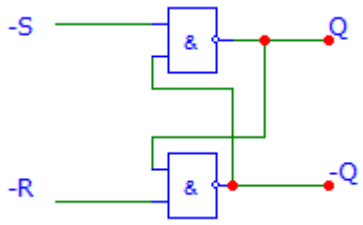


Рис. 5.1. Асинхронный RS-триггер на логике 2И-НЕ

Таблица 5.1

$\overline{R}$	$\overline{S}$	Q	$\overline{Q}$	Пояснения
0	0	–	–	Запрещённая комбинация !
0	1	0	1	Установка «0»
1	0	1	0	Установка «1»
1	1	Q	$\overline{Q}$	Хранение

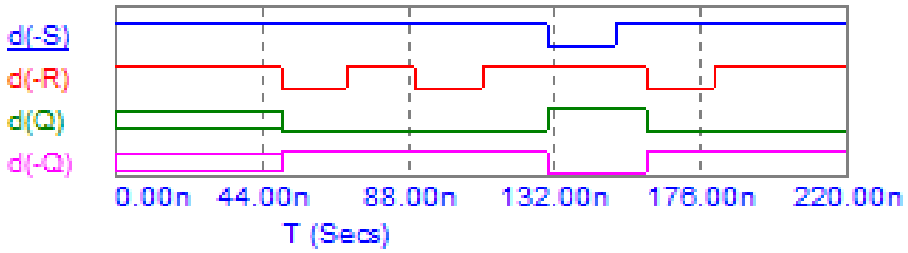


Рис. 5.2. Работа RS-триггера, собранного из двух элементов 2И-НЕ

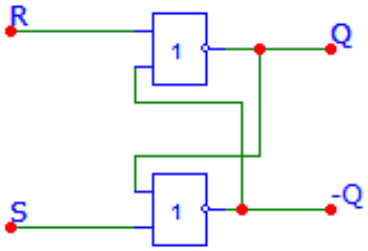


Рис. 5.3. Асинхронный RS-триггер на логике 2 ИЛИ-НЕ

В этой схеме установка значений выходов триггера осуществляется логической 1.

Таблица истинности работы данного триггера представлена в табл. 5.2.

Работа триггера, собранного на логике **2ИЛИ-НЕ**, представлена на рис. 5.4.

Внимание: для правильной работы схемы 5.3 не допустимы появления положительных сигналов (лог. 1) одновременно на входе S и R.

Оба рассмотренных триггера изменяют своё состояние немедленно, как только изменятся входные сигналы. Такой режим работы триггера называется **асинхронный**. Порой такой режим работы не всегда удобен. Ведь для установки входных сигналов по входу S и R иногда требуется разное время. А это может выставить на выходах не те значения, которые ожидались. Для исключения подобного недостатка схему управления триггером немного изменяют (рис. 5.5). После такого изменения триггер будет запоминать входные сигналы только в момент разрешения. Разрешение на запись определяется сигналом синхронизации C. На схеме U1-U2 логические элементы **2И** работают в качестве управляемых вентилей (ключей). Сигнал C разрешает или запрещает проходить сигналам S и R на входы асинхронного RS-триггера. Такая доработка схемы превращает асинхронный триггер в синхронный.

Таблица 5.2

S	R	Q	$\overline{Q}$	Пояснения
0	0	Q	$\overline{Q}$	Хранение
0	1	0	1	Установка «0»
1	0	1	0	Установка «1»
1	1	–	–	Запрещённая комбинация !

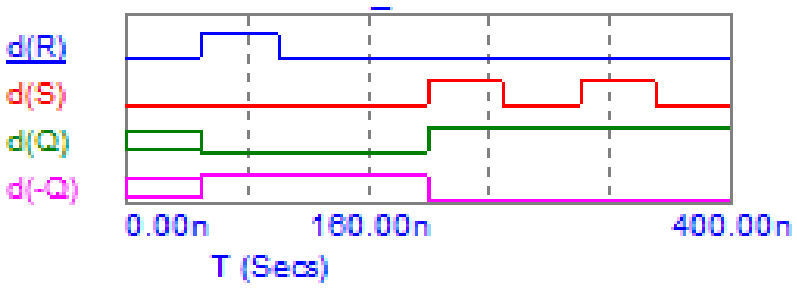


Рис. 5.4. Работа RS-триггера, собранного из двух элементов 2ИЛИ-НЕ

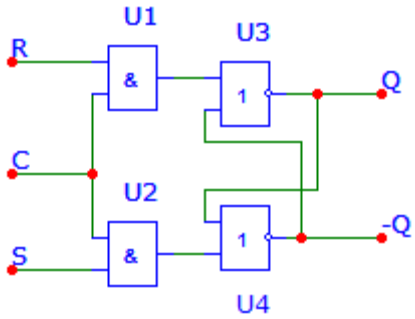


Рис. 5.5. Синхронный RS-триггер на логике 2ИЛИ-НЕ

Если посмотреть на графики работы контрольных точек (рис. 5.6), то можно заметить, что изменяется состояние триггера d(Q) при появлении лог. 1 на входе С. Это и есть синхронизация RS-триггера.

УГО RS-триггера выглядит как на рис. 5.7.

Отечественная промышленность выпускает микросхемы, где представлены RS-триггеры в чистом виде. Это микросхема, например, 561ТР2 (рис. 5.8).

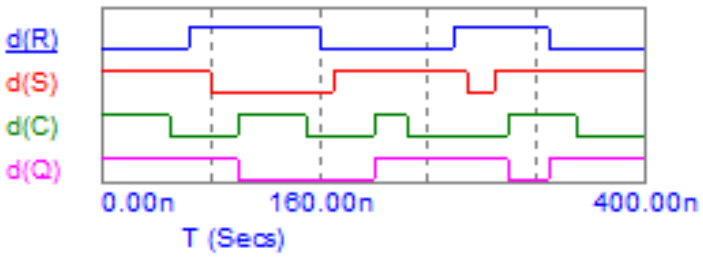


Рис. 5.6. Работа синхронного RS-триггера

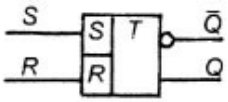


Рис. 5.7. УГО RS-триггера

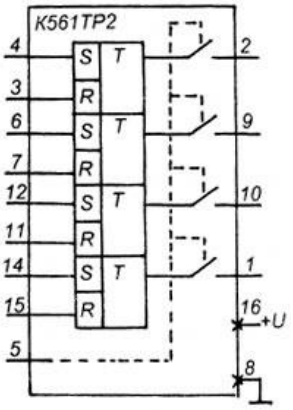


Рис. 5.8. Микросхема 561ТР2

5.2. D-ТРИГГЕР

Большим неудобством при работе с RS-триггером является подача на разные входы сигналов для установки и сброса цифровой схемы памяти. Если сигнал установки S подать через инвертор на вход R, то можно создать триггер, у которого один установочный вход – D. А если рассматривать синхронный RS-триггер, собранный на логике 2И-НЕ, то получим следующую схему синхронного D-триггера (рис. 5.9).

Работает D-триггер следующим образом. Если установить логический сигнал 1 на входе D, то после подачи положительного сигнала на вход синхронизации C D-триггер взведется выходом Q в единичное состояние. Если на входе D был бы 0, то выход Q сбросится в нулевое значение. На рисунке 5.10 временными диаграммами показана работа D-триггера.

Давайте разберём графики на рис. 5.10. По очертанию синхросигнала можно выделить три участка. Участок 1 показывает, что было на предыдущем состоянии, до появления сигнала C. Участок 2 называется «прозрачность». Здесь выход Q сразу изменяется при изменении значения входа D, так как на входе C лог. 1. А после того, как уровень сигнала C изменится до лог. 0 (участок 3), то данные на выходе Q «защёлкиваются» и не меняются даже при изменении сигнала на входе D. Из графика видно, что D-триггер хранит данные, когда на входе C будет нулевой уровень. Поэтому, в технической литературе можно встретить название D-триггера, как триггер-защёлка (latch). Объединяя несколько D-триггеров создают регистры-защёлки. О них пойдёт речь позже.

На схемах D-триггер (защёлка) изображается, как на рис. 5.11.

Основным недостатком D-триггера (защёлки) является наличие режима «прозрачность». В этот период времени, когда высокий уровень на входе C, триггер постоянно записывает в себя изменения по входу D. Для борьбы с этим порой приходится на вход C подавать короткий импульс для записи. Недаром, D-триггеры, построенные по вышеизложенной схеме, называются статические. В идеале, длительность импульса на входе C должен быть равен 0 с, а лучше, если триггер срабатывал бы на только фронт импульса.

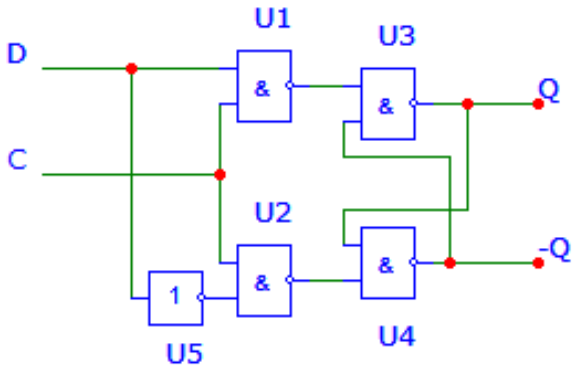


Рис. 5.9. Синхронный D-триггер на логике 2И-НЕ

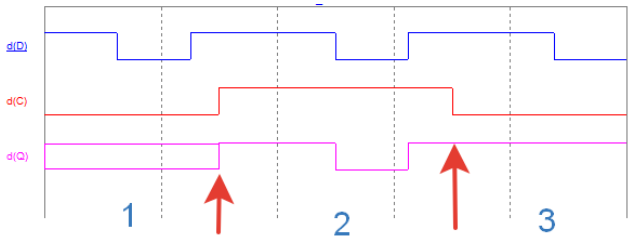


Рис. 5.10. Пример работы синхронного D-триггера

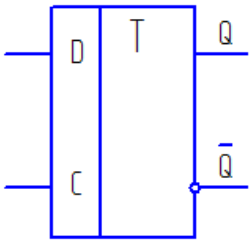


Рис. 5.11. УГО синхронного статического D-триггера

Для записи данных в D-триггер по фронту синхронизации изменяют схему до динамического D-триггера, который собирается из двух статических D-триггеров (рис. 5.12). Графики потенциалов контрольных точек исследования динамического D-триггера представлены на рис. 5.13.

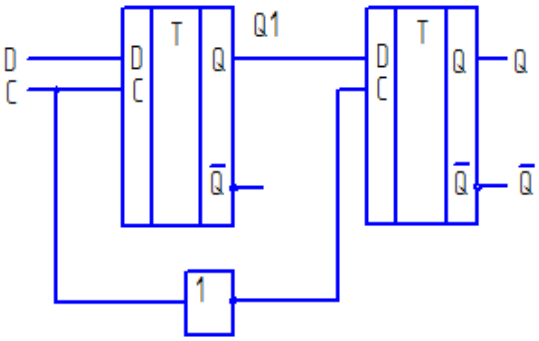


Рис. 5.12. Схема динамического D-триггера

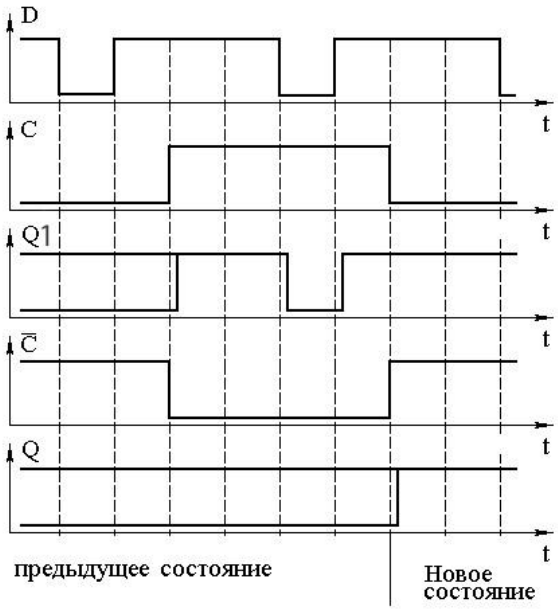


Рис. 5.13. Работа динамического D-триггера

Исследуя графики работы D-триггера, представленного на рис. 5.12, можно заметить, что сигнал на выходе (Q) всей схемы D-триггера в целом не зависит от сигнала на входе D. Если первый D-триггер пропускает сигнал данных со своего входа на выход, то второй статический D-триггер в это время находится в режиме хранения и поддерживает на выходе предыдущее значение сигнала, что соответствует неизменному выходу D-триггера в целом.

В результате проведённого анализа временных диаграмм мы определили, что сигнал запоминается только в момент изменения сигнала на синхронизирующем входе C с единичного потенциала на нулевой. Такое изменение сигнала на входе C называется задний фронт сигнала.

На схемах УГО динамического D-триггера, реагирующего на задний (обратный, отрицательный) фронт синхросигнала может иметь два изображения, как на рис. 5.14. Существуют D-триггеры, записывающие в себя информацию по переднему фронту (рис. 5.15). Из-за того, что в динамическом D-триггере присутствуют два статических D-триггера, то на их УГО записываются две буквы Т – ТТ. Допускается и запись одной буквы Т.

Промышленность выпускает готовые микросхемы, в которых реализованы динамические D-триггеры, например, 561ТМ2 (рис. 5.16). В этой микросхеме присутствуют два D-триггера. В данном триггере имеются дополнительные входы S и R, назначение которых аналогичны, как в RS-триггере. В некоторых микросхемах можно встретить инверсные входы S и R, например, в серии 155.

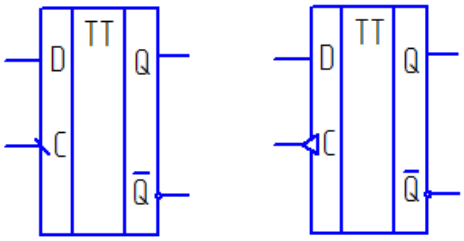


Рис. 5.14. УГО динамического D-триггера, реагирующего на задний фронт сигнала

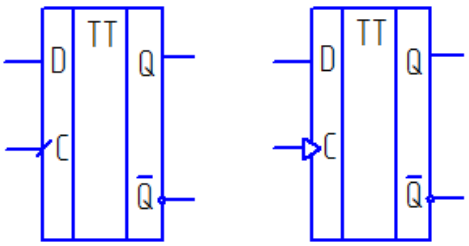


Рис. 5.15. УГО динамического D-триггера, реагирующего на передний фронт сигнала

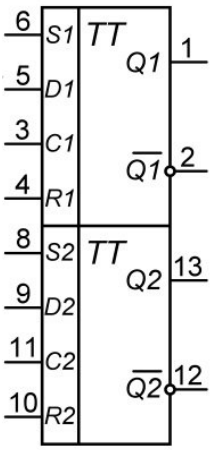


Рис. 5.16. Микросхема D-триггера 561ТМ2

5.3. Т-ТРИГГЕР

Если соединить выход $\overline{Q}$ динамического (двухступенчатого) D-триггера со входом D, то получится Т-триггер, умеющий подсчитывать входные импульсы. Также называют Т-триггер счётным триггером (рис. 5.17).

Т-триггер имеет только один вход (Т). После поступления на вход Т-импульса, состояние триггера меняется на противоположное. Счётным он называется потому, что Т-триггер как бы подсчитывает количество импульсов, поступивших на его вход. Подсчитывает Т-триггер всего один входной сигнал. При поступлении второго импульса Т-триггер снова сбрасывается в исходное состояние. Исследование в среде Micro-CAP Т-триггера, собранного из динамического D-триггера, представлено на рис. 5.18.

В рассмотренном примере при появлении нового фронта на входе d(T) выход d(Q) триггера меняется на противоположное состояние.

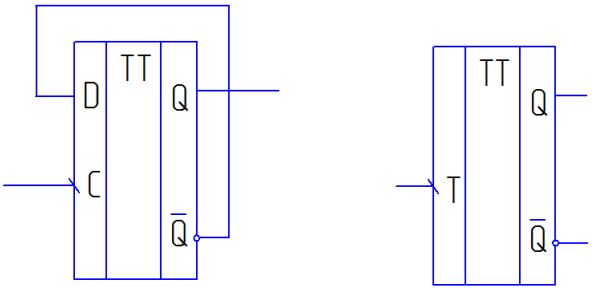


Рис. 5.17. Трансформация D-триггера(справа) в Т-триггер (слева)

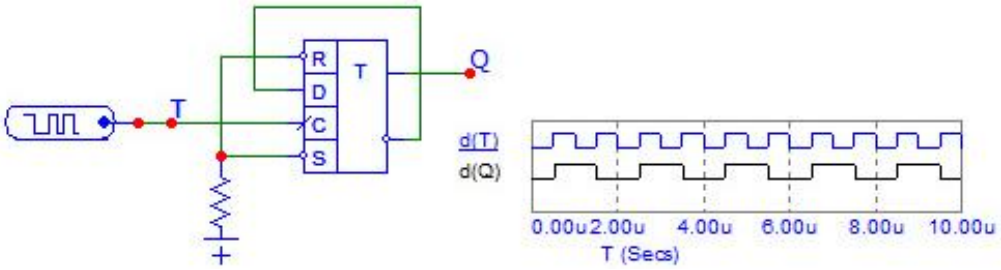


Рис. 5.18. Исследование Т-триггера

5.4. JK-ТРИГГЕР

JK-триггер считается самым сложным, но и самым универсальным из семейства триггеров. У данного триггера основными входами считаются J (от англ. Jump – прыжок), K (от англ. Kill – убить) и C, а выходы Q и $\overline{Q}$. Вход J играет роль подобно входа S, а K-вход соответствует входу R. По структуре и принципу работы JK-триггер напоминает RS-триггер, за исключением того, что в JK–триггере отсутствуют запрещенные комбинации входных сигналов. Если в каждом такте тактирующего сигнала C входная комбинация J=K=1, то JK-триггер меняет своё состояние на противоположное, т.е. становится Т-триггером. JK–триггер состоит из двух синхронных RS-триггеров с дополнительной логикой, которая и исключает запрещенную комбинацию (рис. 5.19). Таблица истинности JK-триггера представлена на рис. 5.3.

Алгоритм функционирования JK-триггера можно представить формулой:

$$Q(t+1) = \overline{Q}(t)\cdot J + Q(t)\cdot \overline{K}.$$

На рисунке 5.20 представлены варианты подключения JK-триггера для моделирования D-триггера и Т-триггера. Но это все одноступенчатые триггеры. Для улучшения стабильности работы триггеров нужно применять двухступенчатые триггеры, что способствует выделению тактирующего фронта для изменения состояния. Так как каждая ступень триггера тактируется поочередно, что исключают паразитные генерации в схеме.

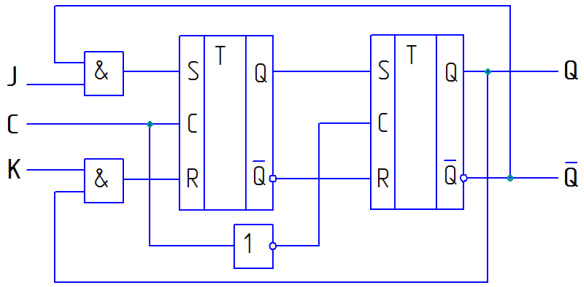
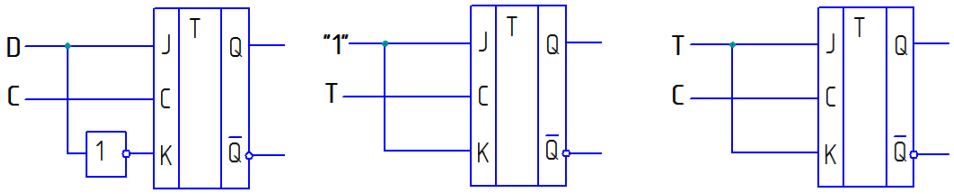


Рис. 5.19. Структура JK-триггера

Таблица 5.3

C	K	J	Q(t)	Q(t + 1)	Пояснения
0	x	x	0	0	Режим хранения информации
0	x	x	1	1	
1	0	0	0	0	Режим хранения информации
1	0	0	1	1	
1	0	1	0	1	Режим установки единицы J = 1
1	0	1	1	1	
1	1	0	0	0	Режим записи нуля K = 1
1	1	0	1	0	
1	1	1	0	1	Счётный режим триггера K = J = 1
1	1	1	1	0	



а) *б*) *в*)

Рис. 5.20. Использование JK-триггера в качестве:
а – D-триггер; *б* – асинхронный T-триггер; *в* – синхронный T-триггер

Промышленность много выпускает различных микросхем, где реализованы JK-триггеры. Например, универсальный JK-триггер К155ТВ1 (КМ155ТВ1). Это двухступенчатый триггер со сложной входной логикой. J и K представлены тремя отдельными входами, объединёнными по схеме логического **И**. Отрицательные входы R и S предназначены для предварительной установки выходов триггера, как у RS-триггера. УГО подобной микросхемы имеет вид на рис. 5.21.

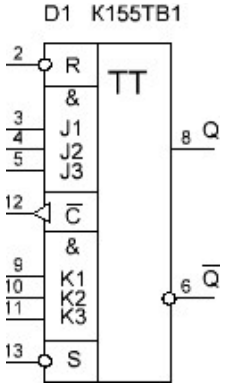


Рис. 5.21. Микросхема К155ТВ1

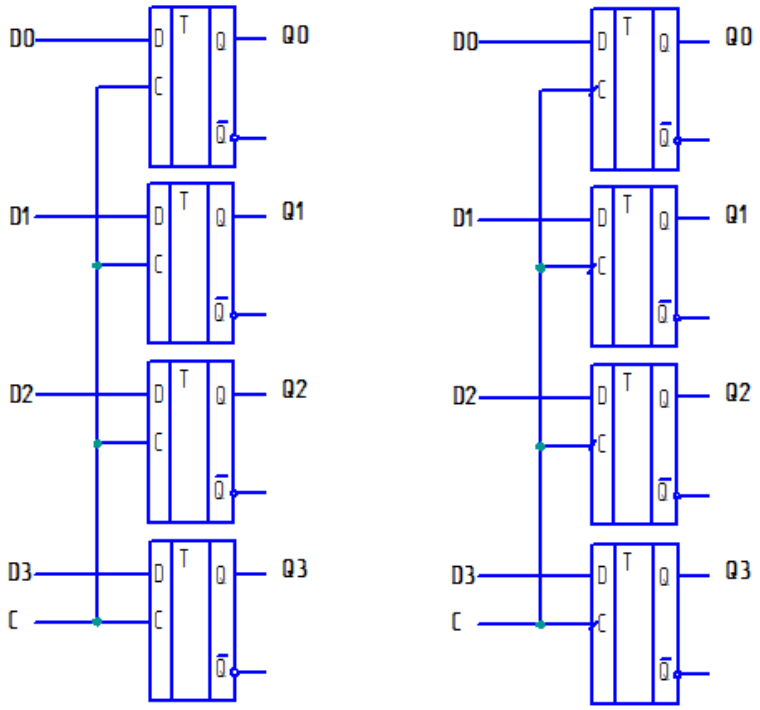
6. РЕГИСТРЫ

Регистр – последовательное или параллельное включение триггеров. Как правило, в качестве триггеров в регистрах используют статические, так и динамические D-триггеры.

6.1. ПАРАЛЛЕЛЬНЫЕ РЕГИСТРЫ

Параллельные регистры предназначены для запоминания многоразрядных двоичных слов. Для каждого разряда в регистре предоставлен один триггер. Различают **стробируемые регистры**, срабатывающие по **уровню входного сигнала** (вход C), и **тактируемые регистры**, запоминающие данные по **фронту сигнала** управления (вход C). На рисунке 6.1 представлены примеры 4-х разрядных параллельных регистров, выполненных на статических триггерах (*а*) (стробируемый регистр) и на динамических (*б*) (тактируемый регистр). Регистры, выполненные по варианту (*а*), ещё называются регистры-защёлки. Здесь D0-D3 разряды входных данных, а Q0-Q3 – разряды запомненных в регистре данных.

На представленном (рис. 6.2) примере УГО параллельного регистра инверсные выходы не показаны, так как в микросхемах регистров инверсные выходы не выведены наружу и к ним нет доступа. УГО реальных микросхем параллельных регистров будут немного отличаться от изображения на рис. 6.2 из-за наличия дополнительной входной логики и сигналов управления.



а) *б*)

Рис. 6.1. Параллельные регистры, выполненные на D-триггерах

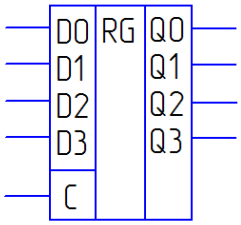


Рис. 6.2. УГО параллельного регистра

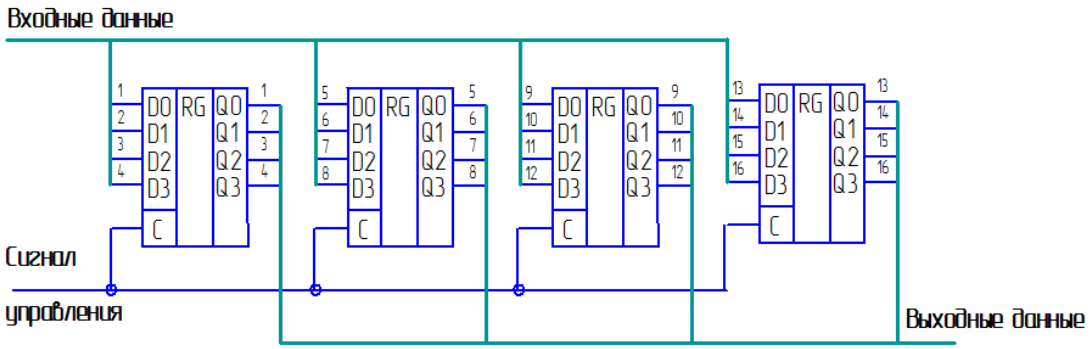


Рис. 6.3. Расширение параллельного регистра до 16-и разрядов

Промышленностью выпускаются четырёхразрядные и восьмиразрядные микросхемы параллельных регистров. Для расширения разрядности регистров, например, до 16 разрядов, объединяют 4-е параллельных регистров, например, как на рис. 6.3.

Здесь номера разрядов входных данных и выходных соответствуют 1-16.

6.2. ПОСЛЕДОВАТЕЛЬНЫЕ (СДВИГОВЫЕ) РЕГИСТРЫ

Регистры сдвига или сдвиговые регистры представляют собой последовательно соединенную цепочку триггеров. Основной режим их работы – это сдвиг разрядов кода, записанного в эти триггеры, т.е. по тактовому сигналу содержимое каждого предыдущего триггера переписывается в следующий по порядку в цепочке триггер. Код, хранящийся в регистре, с каждым тактом сдвигается на один разряд в сторону старших разрядов или в сторону младших разрядов, что и дало название регистрам данного типа.

Последовательный регистр (регистр сдвига или сдвиговый регистр) 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).

На рисунке 6.4 изображение 8-и разрядного последовательного регистра, собранного из D-триггеров. На первом такте значение входа (старший разряд D7) перепишется в Q0. На втором такте в Q1, а разряд D6 в D0 и т.д. За восемь тактов восемь в D-триггеров запишутся все восемь разрядов записываемого слова. Стрелкой на рисунке показано продвижение данных в регистре при тактировании схемы.

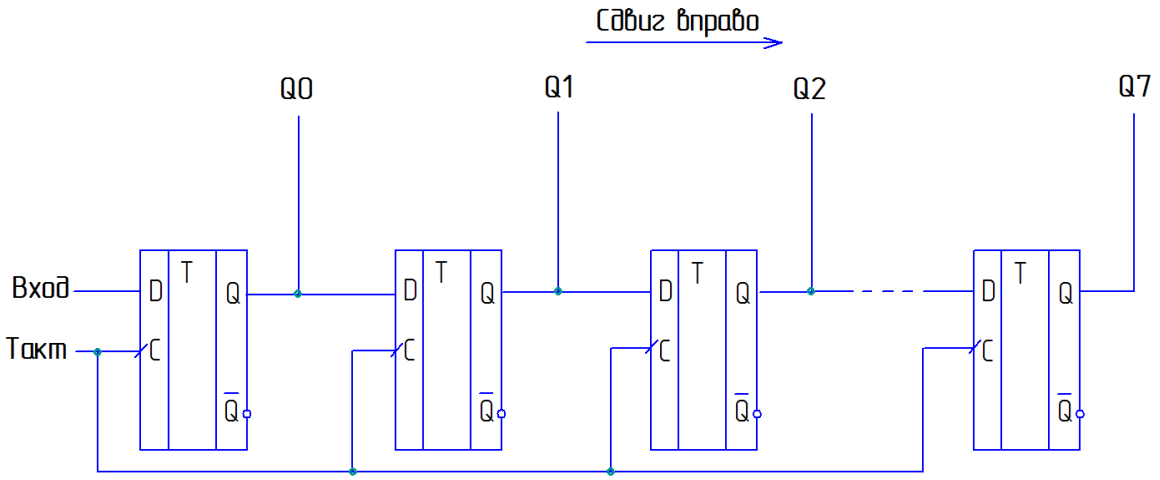


Рис. 6.4. Последовательный регистр, выполненный на D-триггерах

В стандартные серии цифровых микросхем входит несколько типов сдвиговых регистров, отличающихся возможными режимами работы, режимами записи, чтения и сдвига, а также типом выходных каскадов (2С или 3С). Последнее нужно пояснить. 2С – это классический выход микросхем с двумя возможными состояниями: 0 или 1. 3С – это 2С + третье состояние, при котором на выходе появляется высокое сопротивление (импеданс), что способствует отключению выхода от последующего входа (входов) другой микросхемы. Такое состояние называется высоким импедансом. Большинство регистров сдвига имеет восемь разрядов. На рисунке 6.5 показаны примеры выпускаемых микросхем сдвиговых регистров.

Регистр ИР8 является прообразом регистра, изображённого на рис. 6.4. У этого регистра два входа D объединены функцией 2**И**. Отрицательный вход R необходим для принудительного сброса выходов регистра.

Регистр ИР9 выполняет функцию, обратную регистру ИР8. Если ИР8 преобразует входную последовательную информацию в выходную параллельную, то регистр ИР9 преобразует входную параллельную информацию в выходную последовательную. Однако суть сдвига не меняется, просто в ИР9 все внутренние триггеры имеют выведенные параллельные входы, и только один, последний триггер имеет выход (причём как прямой, так и инверсный). Запись входного кода в регистр производится по нулевому сигналу на входе -WR. Сдвиг осуществляется по положительному фронту на одном из двух тактовых входов С1 и С2, объединённых по функции 2**ИЛИ**. Имеется также вход расширения DR, сигнал с которого в режиме сдвига перезаписывается в младший разряд сдвигового регистра.

Рассмотрим работы сдвигового регистра в среде Micro-CAP на примере регистра ИР8 (рис. 6.6).

Для работоспособности регистра на вход – R подали лог.1. Входы данных (A и B) объединим и на него подадим сигналы с генератора U2. Тактировать схему будем генератором U1. Данные для эксперимента – двоичных код: 10010100. Так как для сдвигового регистра первым на вход данных должен поступить старший разряд кода (!), то данные поступали на вход в той же последовательности, как бинарные числа записаны в коде. Для этого была создана программа для генератора U2. Программы генераторов U1 и U2 представлены в табл. 6.1.

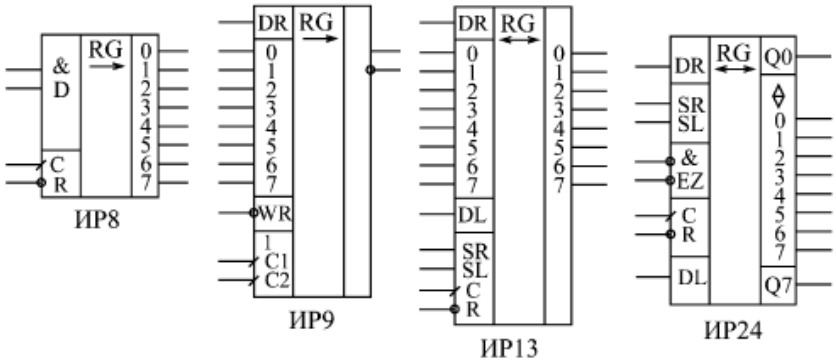


Рис. 6.5. Микросхемы последовательных регистров

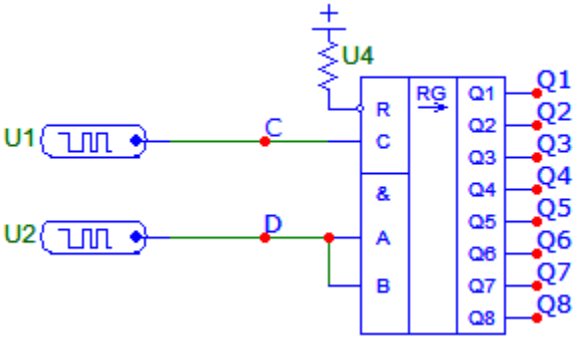


Рис. 6.6. Схема включения сдвигового регистра ИР8

Таблица 6.1

U1	U2
.define _C +0ns 0 +label=start +500n 1 +600n 0 +1500n goto start -1 times	.define _D +0ns 0 +460n 1 +940n 0 +3470n 1 +3940n 0 +5400n 1 +5900n 0

В симуляторе Micro-CAP строим временные диаграммы контрольных точек схемы (рис. 6.7).

За восемь импульсов синхросигнала входа C в регистр запишутся все восемь разрядов последовательного входного кода. На восьмом такте сигнала C видно, как на выходах сдвигового регистра Q1 – Q8 появились значения входного кода. Логические единицы на выходе на графиках уточнены красными стрелками.

Обратите внимание, что после 8-го такта синхросигнала данные в регистре начнут «уходить» вправо и на освободившиеся разряды регистра запишутся лог. 0 Освобождение выходных разрядов начнется с младших.

Сдвиговые регистры легко расширяются до большей разрядности. На рисунке 6.8 показан пример схемы на 16 разрядов из двух 8-и разрядных сдвиговых регистров ИР8.

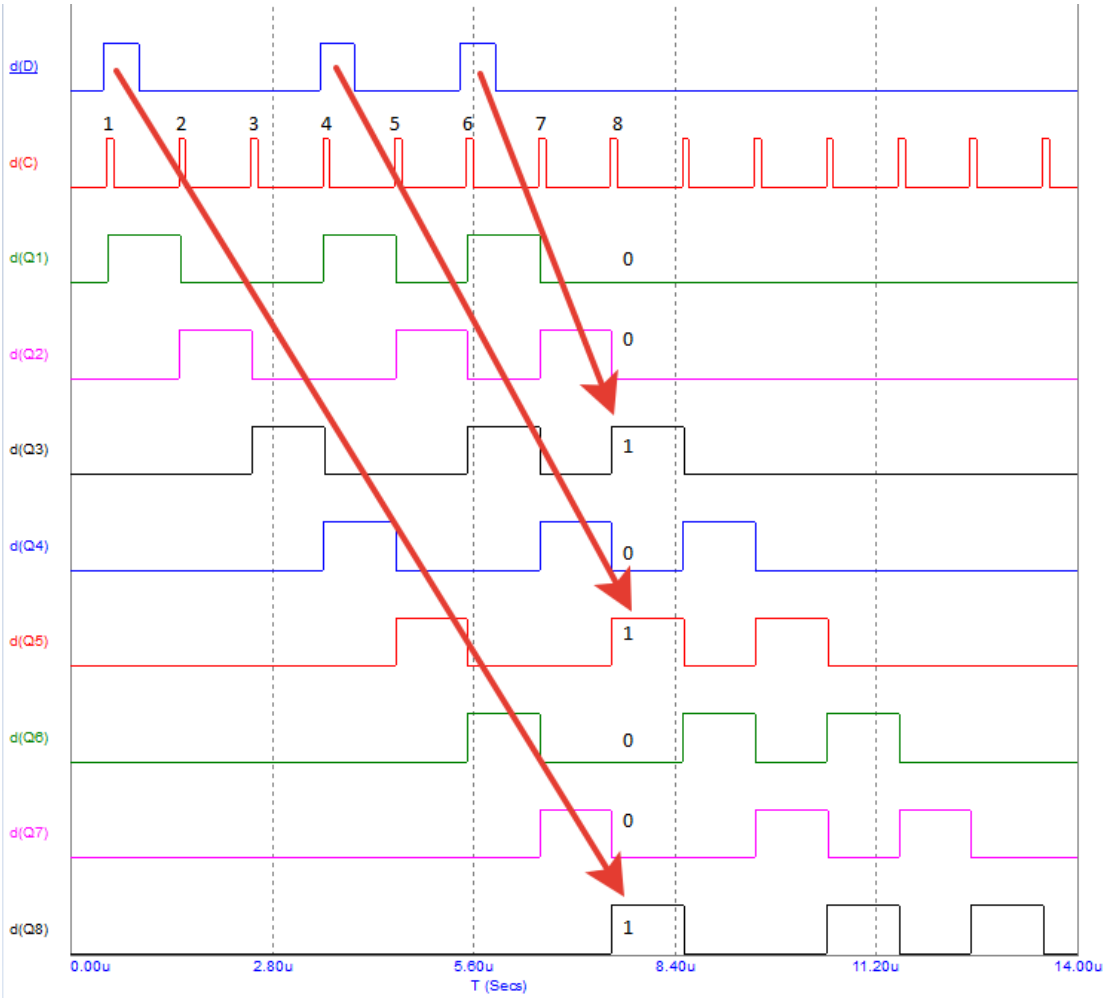


Рис. 6.7. Исследование работы сдвигового регистра ИР8

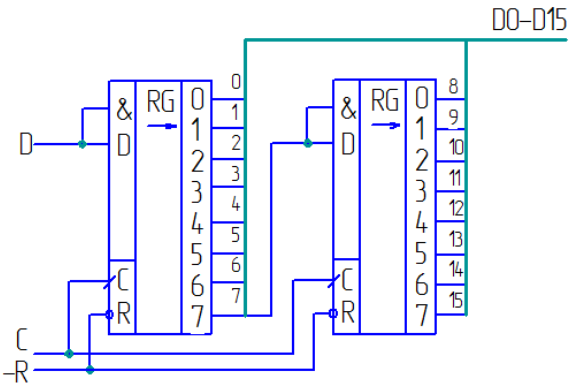


Рис. 6.8. Расширение последовательных регистров

Главное применение последовательных регистров состоит в преобразовании параллельного кода в последовательный, и наоборот. Такое преобразование используется, например, при передаче информации на большие расстояния (в информационных сетях), при записи информации на внешние носители, а также во многих других случаях.

Для примера на рис. 6.9 показана схема передачи цифровой информации в последовательном коде по двум линиям: информационной и синхронизующей. Такая передача позволяет сократить количество соединительных проводов, а также упростить защиту передаваемых данных от действия внешних электромагнитных помех. Следует заметить, что для более корректной связи нужен общий 0В. На схеме это не указано.

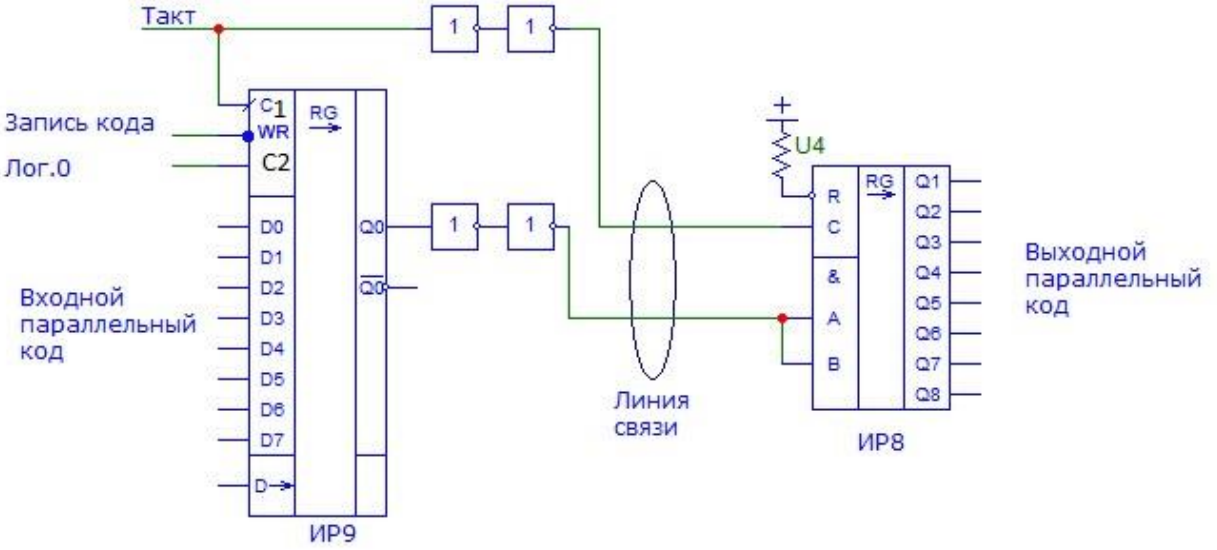


Рис. 6.9. Пример использования сдвиговых регистров для дистанционной связи

7. АСИНХРОННЫЕ И СИНХРОННЫЕ СЧЁТЧИКИ

Счётчиком называется последовательное устройство, собранное из триггеров и предназначенное для счёта входных сигналов и фиксации результата на своих выходах.

Основные функции счётчиков:

- запись входной информации в параллельном формате;
- хранение информации;
- выдача хранимой информации в параллельном формате;
- увеличение хранимой информации на единицу (инкремент) при поступлении входного сигнала;
- уменьшение хранимой информации на единицу (декремент) при поступлении входного сигнала;

Главным параметром счётчика считается модуль счёта. Это максимальное количество поступивших на вход счётчика импульсов, после которого значение счётчика сбрасывается в исходное значение, т.е. модуль счёта – это максимальное количество импульсов, которое может быть подсчитано счётчиком.

По модулю счёта счётчики классифицируются:

- двоичные;
- двоично-десятичные;
- с постоянным модулем счёта;
- с переменным модулем счёта;
- суммирующие;
- вычитающие;
- реверсивные.

По организации внутренних связей счётчики бывают:

- с последовательным переносом;
- с параллельным переносом;
- со смешанным переносом;
- кольцевые.

Рассмотрим некоторые из классифицированных счётчиков. Из библиотеки Micro-CAP Digital Library выберем JK-триггер 74107 (Digital Library–>74xx42–>105–>74107). Выбор данного триггера обусловлен простотой его управляющих сигналов и отрицательным тактирующим сигналом. Кроме этого, нас будет интересовать и реальная временная модель триггеров.

Из JK-триггеров смоделируем Т-триггеры, объединив входы J и K логическим уровнем 1, и соберём схему (рис. 7.1). Прямые выходы Q триггеров соединим с синхронизирующими входами CLK последующих триггеров. Получим суммирующий входные импульсы асинхронный двоичный счётчик.

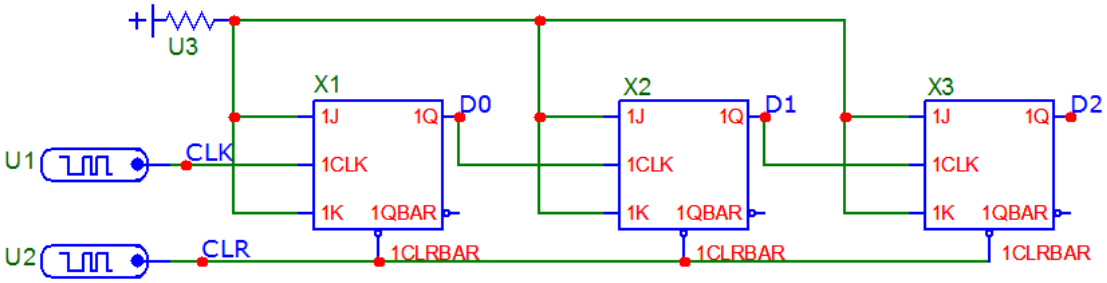


Рис. 7.1. Асинхронный суммирующий двоичный счётчик с цепями последовательного переноса

На входы сброса CLRBAR состояния триггеров подадим отрицательный импульс в начале цикла работы счётчика, а на вход синхронизации CLK первого триггера (X1) подадим сигналы CLK с генератора U1. Состояние выходов Q0-Q2 триггеров X1-X3 можно увидеть на графиках 7.2.

Обратите внимание, что на восьмом импульсе состояние триггеров переходит в состояние 000. Таким образом, предложенный трёхразрядный счётчик считает входные сигналы с модулем счёта – 8. Если к схеме добавить ещё один триггер, то модуль счёта изменится до 16. Увеличивая количество триггеров можно увеличивать модуль счёта.

Если немного изменить схему и взять, например, триггеры 74H106, у которых имеются входы для принудительной установки лог. 1 на выходах Q, то можем получить вычитающий двоичный счётчик (рис. 7.3). У вычитающего счётчика сигналы на инверсные синхронизирующие входы CLK триггеров берутся не с прямых выходов Q, а с инверсных $\overline{Q}$.

Генератор U1, переходом сигнала из 1 в 0, выставит на всех выходах триггеров лог. 1. Это соответствует состоянию 111_2 (7_{10}). После первого входного импульса CLK от генератора U2 на выходах данной схемы будет код 110_2, что соответствует десятичному представлению – 6. После второго импульса выходы изменятся на $101_2 = 5_{10}$ и т.д. На седьмом входном импульсе состояния триггеров сбросятся в значение 000_2. Таким образом, данная схема счётчика умеет вычитать по единице из своего предыдущего значения при каждом появлении заднего фронта сигнала на счётном входе CLK. Временные диаграммы работы вычитающего счётчика с тактовой частотой CLK в 1МГц представлены на рис. 7.4.

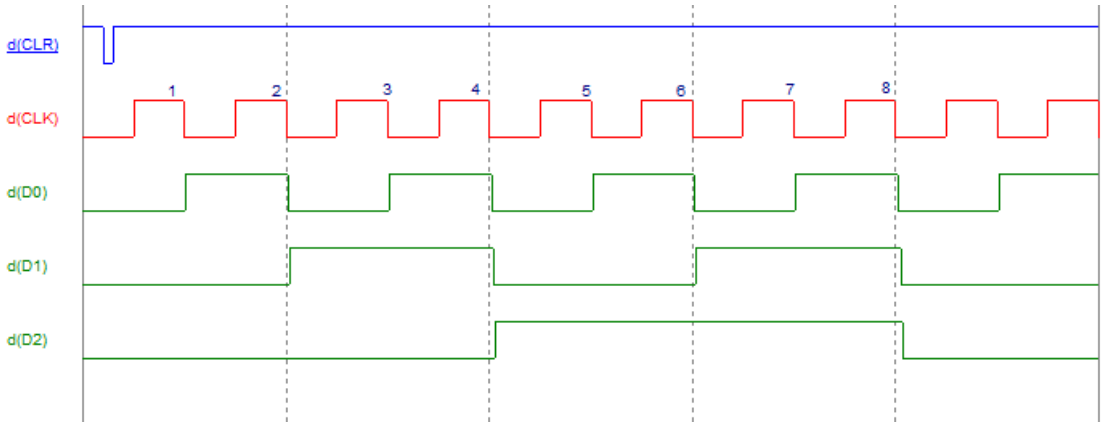


Рис. 7.2. Исследование суммирующего асинхронного счётчика

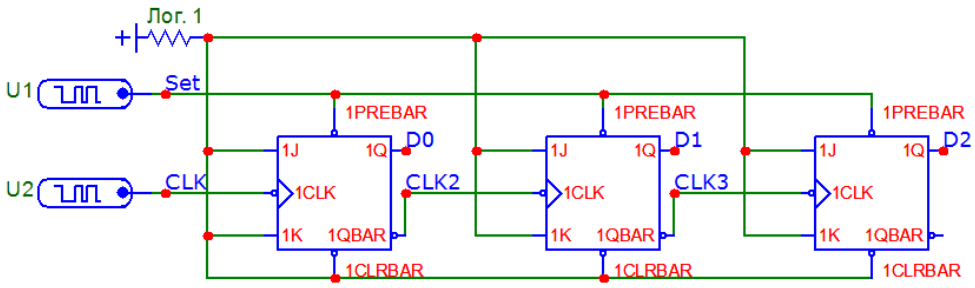


Рис. 7.3. Асинхронный вычитающий двоичный счётчик с цепями последовательного переноса

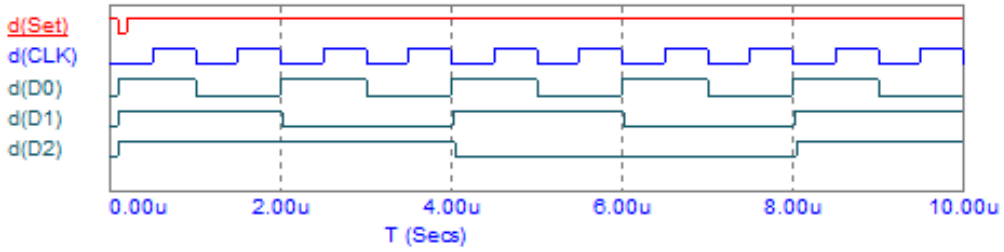


Рис. 7.4. Исследование вычитающего счётчика на частоте 1 МГц CLK

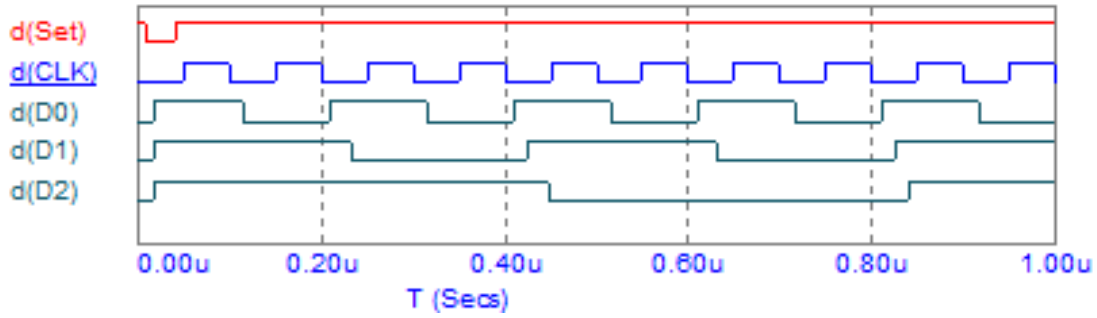


Рис. 7.5. Исследование вычитающего счётчика на частоте 10 МГц CLK

Для эксперимента увеличим входную частоту тактовых импульсов до 10 МГц. Полученные временные диаграммы выходов счётчиков представлены на рис. 7.5.

Из графиков видно, что с увеличением частоты CLK становится более выражено смещение во времени изменения выходов асинхронных счётчиков в старших разрядах. Само смещение графиков по длительности не изменилось, просто оно стало заметнее на фоне уменьшенной длительности входного импульса синхронизирующего сигнала. Это касается, как вычитающих счётчиков, так и суммирующих, построенных по выше приведённым схемам. Это связано с тем, что счётчики асинхронные. Каждый последующий триггер сработает после того, как сработает предыдущий. Поэтому, с увеличением разрядности асинхронных счётчиков увеличивается и запаздывание переключения разрядов подсчёта.

Устранить подобную проблему в счётчиках можно с переходом в синхронный режим. А для этого синхронизирующий сигнал CLK должен поступать одновременно на все триггеры счётчика. И только тот триггер должен изменить своё состояние, у которого имеется разрешение. А разрешение определяется шагом подсчёта. Подобная схема представлена на рис. 7.6.

Первый триггер X1 будет постоянно изменять своё состояние с приходом очередного синхросигнала CLK. Триггер X2 будет изменять своё состояние, только когда на выходе X1 будет лог. 1. А это каждый четный импульс CLK. Триггер X3 взведет свой выход Q2 в единичное состояние, когда на выходах D0 и D1 будут лог. 1. Это условие отслеживает логический элемент **И** (X5). Четвёртый JK-триггер X4 через логический элемент **И** (X6) становится T-триггером и изменяет своё состояние при соответствующей фазе синхросигнала. Временные диаграммы работы синхронного счётчика представлены на рис. 7.7. Если сравнить временные диаграммы с рис. 7.5 и 7.7, то можно отчетливо заметить, что задержки переключения внутренних триггеров синхронного счётчика на частоте 10МГц отсутствуют. Одновременное небольшое смещение изменений состояний выходов асинхронного счётчика обусловлено длительностью переключения самого триггера и логических элементов схемы.

Готовые счётчики, как асинхронные, так и синхронные, выпускаются в виде интегральных микросхем.

В составе стандартных серий цифровых микросхем асинхронных счётчиков можно рассмотреть, например, 4-разрядный двоично-десятичный счётчик ИЕ2 и 4-разрядный двоичный счётчик ИЕ5 (рис. 7.8).

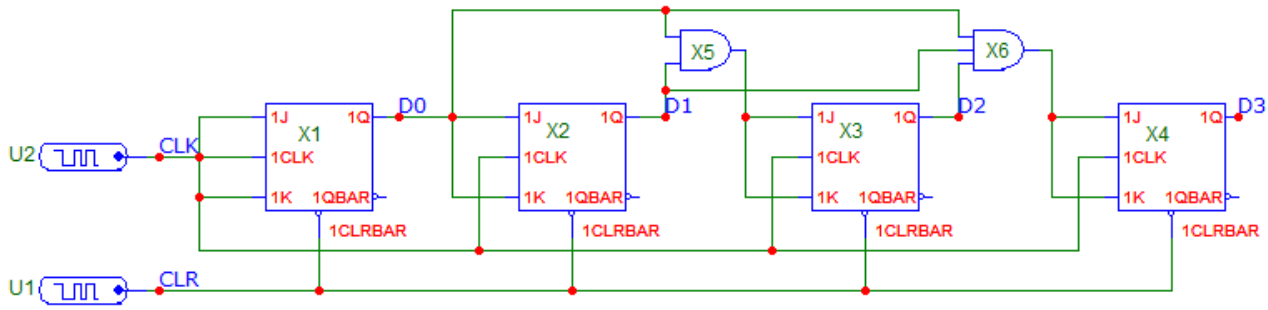


Рис. 7.6. Синхронный двоичный счётчик

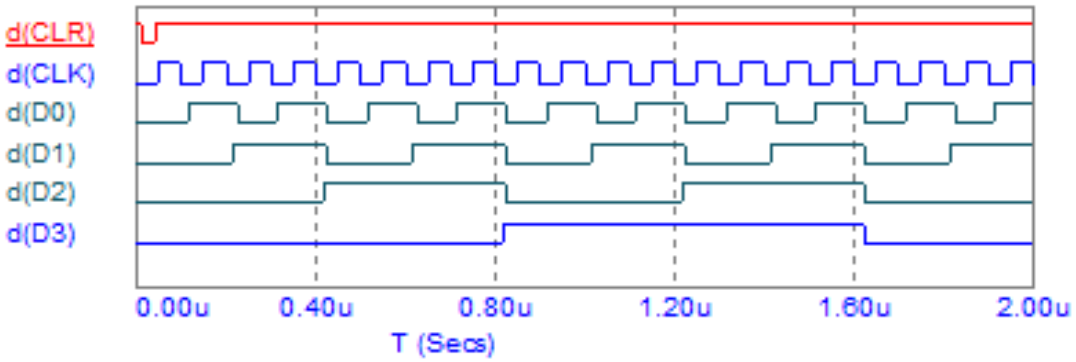


Рис. 7.7. Временная диаграмма синхронного счётчика на частоте 2 МГц CLK

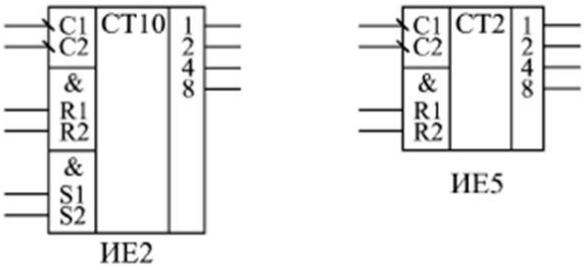


Рис. 7.8. Асинхронные счётчики стандартных серий

У каждого счётчика имеются входы сброса в 0000, объединённые логической функцией **И**. Только у ИЕ2 имеется вход для установки значения 1001_2 (9_{10}). Все асинхронные счётчики работают по отрицательному (заднему) фронту входного сигнала С. У всех двух счётчиков выделены две независимые части, что увеличивает возможности их применения. При объединении этих двух частей получается счётчик максимальной разрядности.

Счётчик ИЕ2 имеет две части: один триггер (вход С1, выход 1) и три триггера (вход С2 и выходы 2, 4, 8). Таким образом, он состоит из одноразрядного счётчика и трёхразрядного счётчика. Одиночный триггер работает в обычном счётном режиме, изменяя своё состояние по каждому отрицательному фронту сигнала С1, т.е. делит частоту входного сигнала на 2. Три оставшихся триггера включены таким образом, чтобы считать до 5. После достижения кода 4 (т.е. 100) на выходах 2, 4 и 8 этот трёхразрядный счётчик по следующему отрицательному фронту сигнала С2 сбрасывается в нуль. В результате при объединении выхода 1 микросхемы со входом С2 мы получаем 4-разрядный двоично-десятичный счётчик, делящий частоту входного сигнала С1 на 10 и сбрасывающийся в нуль после достижения на выходах 1, 2, 4, 8 кода 9 (т.е. 1001) по отрицательному фронту сигнала С1.

Счётчик ИЕ5, точно так же как и ИЕ2, имеет две части: один триггер (одноразрядный счётчик) с входом С1 и выходом 1 и три триггера (трёхразрядный счётчик) со входом С2 и выходами 2, 4, 8. Оба счётчика – двоичные, т.е. первый считает до двух, а второй – до 8. При объединении входа С2 с выходом 1 получается 4-разрядный двоичный счётчик, считающий до 16. Счёт производится по отрицательному фронту входных сигналов С1 и С2. Предусмотрена возможность сброса счётчика в нуль по сигналам R1 и R2, объединённым по функции И.

Счётчики часто используют при построении всевозможных делителей частот. При этом иногда предъявляют требование не только к полученной часты на выходе делителя, но и к форме итогового сигнала. Форма сигнала определяется её скважностью. Скважность сигнала – это отношение периода сигнала к его длительности. Если скважность сигнала равна 2 (длительность импульсов равна длительности паузы между ними), то говорят, что получен меандр сигнала. Получить меандр из любого сигнала довольно просто: надо использовать дополнительный делитель частоты на 2, правда, при этом частота выходного сигнала уменьшится ещё вдвое.

Приведём пример делителя частоты на десять на счётчике ИЕ2 (рис. 7.9). У счётчика ИЕ2 одноразрядный внутренний счётчик включим после трёхразрядного внутреннего счётчика. Трёхразрядный счётчик делит частоту входного сигнала на 5, но выходные импульсы имеют скважность, не равную двум (она равна 5). Одноразрядный счётчик делит частоту ещё вдвое и одновременно формирует меандр. Так как в примере используется счётчик с логической моделью представления, то задержки переключения разрядов счётчика относительно друг друга на рисунке не показаны. На практике задержки присутствуют.

На рисунке 7.10 представлены УГО двух синхронных счётчиков, выпускаемых отечественной промышленностью. Оба счётчика реверсивные с возможностью параллельного занесения первоначального значения во внутренние регистры счёта по отрицательному сигналу С. Счётчик ИЕ6 – двоично-десятичный. Он считает от 0 до 9. При этом имеются выходы переноса ≥9 и ≤0, что требуется при расширении счётчика. На этих выходах появляется сигнал длительностью счётного сигнала при достижении граничного состояния счётчика. Счётчик ИЕ7 – двоичный. Его диапазон счёта 0 – 15. А счётчика ИЕ6 десятичный и диапазон счёта 0-9.

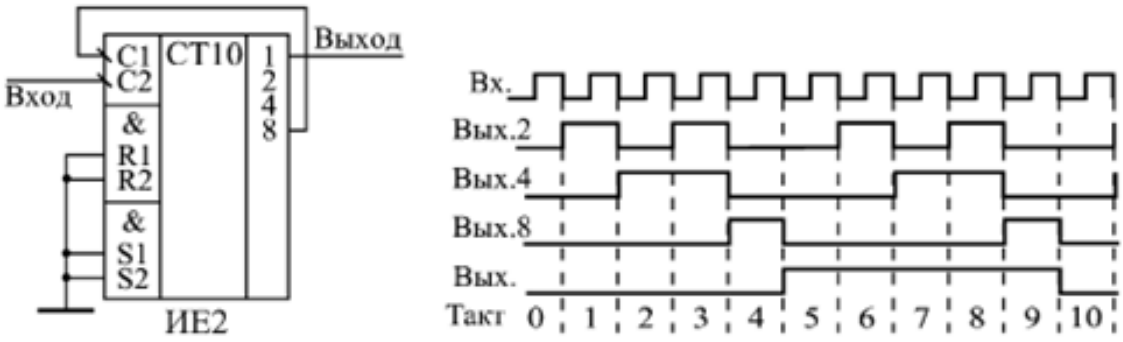


Рис. 7.9. Делитель частоты на 10

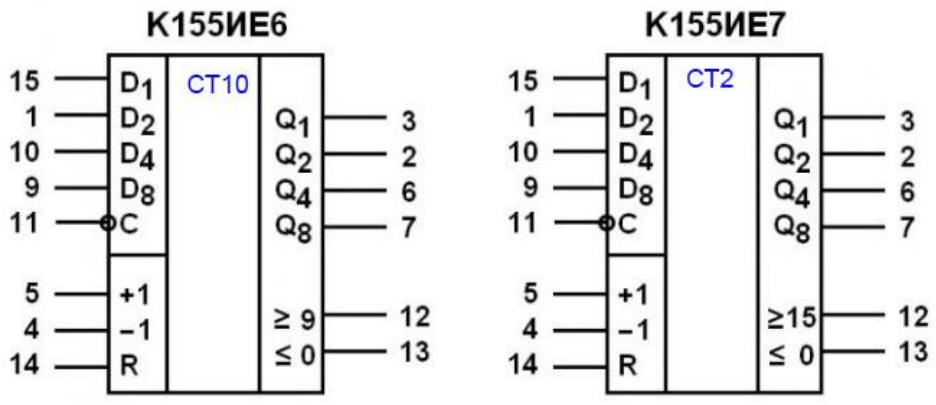


Рис. 7.10. Синхронные реверсивные счётчики

ЗАКЛЮЧЕНИЕ

Современные компьютерные возможности позволяют моделировать и исследовать сложные (многокомпонентные) электронные устройства без затрат на приобретение микросхем, предварительного макетирования и применения дорогостоящих специализированных лабораторий. Современные программы типа Micro-CAP способны ставить любые по сложности эксперименты с разрабатываемой схемой. И если вопрос ставится по изучению дисциплины «Схемотехника», то исследование схем можно осуществлять и в домашних условиях, используя для этой цели свой персональный компьютер.

Данное учебное пособие предназначено не только для облегчения изучения основных тем дисциплины «Схемотехника», но и инструмента исследования цифровых схем – программы Micro-CAP.

Сегодняшние реалии развития схемотехники указывают на то, что уходит в прошлое разработка специализированных электрических схем локальной автоматизации, а все чаще используют универсальные схемы – контроллеры и программируемые логические реле [3]. В этом случае разработка схемы сводится к написанию программы или составлению функциональной схемы в среде разработки программного обеспечения контроллеров. Следует заметить, что функциональные схемы в программируемых логических реле – это те же схемы, только виртуально реализованные в среде контроллера.

Задачи проектирования цифровых схем можно успешно решать только при глубоком понимании работы цифровых устройств [4].

Автор надеется, что данное учебное пособие поможет в освоении дисциплины «Схемотехника».

СПИСОК ЛИТЕРАТУРЫ

1. **ГОСТ** 2.702–2011. Единая система конструкторской документации (ЕСКД). Правила выполнения электрических схем.

2. **http:**//www.spectrum-soft.com

3. **Васильев, С. А.** Промышленные логические программируемые реле (web-формат) [Электронный ресурс, мультимедиа] : учебное пособие / С. А. Васильев. – Тамбов : Изд-во ФГБОУ ВО «ТГТУ», 2017. – URL : https://www.tstu.ru/book/elib3/mm/2017/vasilev/vasilev.zip

4. **Касьянов, А. Н.** Micro-Cap в схемотехнике : учебное пособие / А. Н. Касьянов. – Тамбов : Изд-во ТГТУ, 2004. (pdf-файл).

ОГЛАВЛЕНИЕ

Учебное электронное издание

ВАСИЛЬЕВ Сергей Александрович
КОРОБОВА Ирина Львовна

ОСНОВЫ ЦИФРОВОЙ СХЕМОТЕХНИКИ В ИНФОРМАЦИОННЫХ СИСТЕМАХ

Учебное пособие

Редактирование Е. С. Мордасовой
Инженер по компьютерному макетированию Т. Ю. Зотова

ISBN 978-5-8265-2342-1

Подписано к использованию 21.04.2021.
Тираж 50 шт. Заказ № 58

Издательский центр ФГБОУ ВО «ТГТУ»
392000, г. Тамбов, ул. Советская, д. 106, к. 14
Тел./факс (4752) 63-81-08.
E-mail: izdatelstvo@tstu.ru