Государственное профессиональное образовательное учреждение
«Полысаевский индустриальный техникум»

Учебно-методический комплект
практических работ
по учебной дисциплине
«ОП.08 Архитектура электронно-вычислительных машин и вычислительных систем»

основной профессиональной образовательной программы (ОПОП)

09.02.05 Прикладная информатика

Полысаево, 2019

Учебно-методический комплект по учебной дисциплине «ОП.08 Архитектура электронно-вычислительных машин и вычислительных систем» разработан на основе Федерального образовательного стандарта по профессии, учебного плана, рабочей учебной программы по профессии **09.02.05 Прикладная информатика** в соответствии с методическими рекомендациями по планированию и проведению практических работ обучающихся ГПОУ ПИТ

Организация-разработчик: Государственное профессиональное образовательное учреждение «Полысаевский индустриальный техникум»

Разработчик: Денисова Мария Викторовна, преподаватель специальных дисциплин

Рассмотрено на заседании цикловой методической комиссии общепрофессиональных и профессиональных учебных циклов информационного профиля.

Протокол № 1 от 2 сентября 2019года

Председатель ЦМК_____Денисова М.В.

Рекомендовано
ЦМК ГПОУ ПИТ

Автор:
Денисова М.В. преподаватель специальных дисциплин

Денисова М.В.
Учебно-методический комплект практических работ обучающихся в процессе изучения учебной дисциплины «ОП.08 Архитектура электронно-вычислительных машин и вычислительных систем» [Текст]: учебно-методический комплект / сост. М.В. Денисова – Полысаево : ГПОУ ПИТ, 2019. – 84 с.

Методические рекомендации по выполнению практических работ обучающихся разработаны на основе Федерального образовательного стандарта по профессии, учебного плана, рабочей учебной программы по профессии 09.02.05 Прикладная информатика. Методические рекомендации содержат задания по дисциплине «Архитектура электронно-вычислительных машин и вычислительных систем» и технологию их выполнения для самостоятельной работы обучающихся.

Издание предназначено для обучающихся по профессии 09.02.05 Прикладная информатика и преподавателей специальных дисциплин.

Содержание

Пояснительная записка

Учебная дисциплина **«Архитектура электронно-вычислительных машин и вычислительных систем»** предназначена для реализации Федерального государственного образовательного стандарта по специальности «Прикладная информатика»

Содержание дисциплины **«Архитектура электронно-вычислительных машин и вычислительных систем»** направлено на развитие универсальных учебных действий, формирование личностных, метапредметных и предметных результатов в соответствии с требованиями ФГОС среднего общего образования, а также общих компетенций в соответствии с требованиями ФГОС среднего профессионального образования.

Практические работы являются неотъемлемым этапом изучения учебной дисциплины и проводятся с целью: формирования практических умений в соответствии с требованиями к уровню подготовки обучающихся, установленными рабочей программой учебной дисциплины; обобщения, систематизации, углубления, закрепления полученных теоретических знаний; готовности использовать теоретические знания на практике.

Практические занятия по учебной дисциплине **«Архитектура электронно-вычислительных машин и вычислительных систем»** способствуют формированию в дальнейшем при изучении профессиональных модулей, следующих общих компетенций:

ОК 1. Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.

ОК 2. Организовывать собственную деятельность, определять методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК 3. Решать проблемы, оценивать риски и принимать решения в нестандартных ситуациях.

ОК 4. Осуществлять поиск, анализ и оценку информации, необходимой для постановки и решения профессиональных задач, профессионального и личностного развития.

ОК 5. Использовать информационно-коммуникационные технологии для совершенствования профессиональной деятельности.

ОК 6. Работать в коллективе и команде, обеспечивать ее сплочение, эффективно общаться с коллегами, руководством, потребителями.

ОК 7. Ставить цели, мотивировать деятельность подчиненных, организовывать и контролировать.

ПК 1.1. Обрабатывать статический информационный контент.

ПК 1.2. Обрабатывать динамический информационный контент.

ПК 1.3. Осуществлять подготовку оборудования к работе.

ПК 1.4. Настраивать и работать с отраслевым оборудованием обработки информационного контента.

ПК 1.5. Контролировать работу компьютерных, периферийных устройств и телекоммуникационных систем, обеспечивать их правильную эксплуатацию.

ПК 4.1. Обеспечивать содержание проектных операций.

ПК 4.4. Определять ресурсы проектных операций.

В методических рекомендациях предлагаются к выполнению практические работы, предусмотренные учебной рабочей программой дисциплины **«Архитектура электронно-вычислительных машин и вычислительных систем»**. При разработке содержания практических работ учитывался уровень сложности освоения студентами соответствующей темы, общих и профессиональных компетенций, на формирование которых направлена дисциплина.

Методические рекомендации по учебной дисциплине **«Архитектура электронно-вычислительных машин и вычислительных систем»** имеют практическую направленность и значимость. Формируемые в процессе практических занятий умения могут быть использованы студентами в будущей профессиональной деятельности.

Практические занятия проводятся в учебном кабинете, обязательным этапом является самостоятельная деятельность студентов.

Оценки за выполнение практических работ выставляются по пятибалльной системе. Оценки за практические работы являются обязательными текущими оценками по учебной дисциплине и выставляются в журнале теоретического обучения.

УВАЖАЕМЫЙ СТУДЕНТ!

Методические рекомендации по дисциплине **«Архитектура электронно-вычислительных машин и вычислительных систем»** для выполнения практических работ созданы Вам в помощь для работы на занятиях, подготовки к практическим работам, правильного составления отчетов.

Приступая к выполнению практической работы, Вы должны внимательно прочитать цель и задачи занятия, ознакомиться с заданиями. Все задания к практической работе Вы должны выполнять в соответствии с инструкцией, анализировать полученные в ходе занятия результаты по приведенной методике.

Отчет о практической работе Вы должны выполнить по следующему алгоритму:

1. Тема работы.
2. Цель работы.
3. Письменные задания.

Наличие положительной оценки по практическим работам необходимо для допуска к экзамену по дисциплине **«Архитектура электронно-вычислительных машин и вычислительных систем»,** поэтому в случае отсутствия на занятии по любой причине или получения неудовлетворительной оценки за практическую работу, Вы должны найти время для ее выполнения или пересдачи.

Внимание! Если в процессе подготовки к практическим работам у Вас возникают вопросы, разрешить которые самостоятельно не удается, необходимо обратиться к преподавателю для получения разъяснений или указаний.

Условия и порядок выполнения работы:

1. Прочитать методические рекомендации по выполнению практической работы.
2. Изучить содержание заданий и начать выполнение.
3. Консультацию по выполнению работы получить у преподавателя или обучающегося, успешно выполнившего работу.
4. Работа оценивается в целом, по итогам выполнения работы выставляется оценка
5. Пропущенные практические работы отрабатываются в дополнительное время.

Желаем Вам успехов!!!

Перечень практических работ

Наименование практической работы	Количество часов
1. Оптимальная конфигурация устройств ЭВМ для решения конкретных задач;	2
2 Основные узлы персонального компьютера, разъемы для подключения внешних устройств;	2
3. Работа и особенности логических элементов ЭВМ	2
4. Схемная реализация элементарных логических операций;	2
5. Работа логических узлов ЭВМ.	2
6. Идентификация и установка процессора; Система команд	1
7. Построение памяти на RS-триггерах, JK-триггерах	2
8. Построение памяти на D-триггерах	2
9. Построение регистров хранения различной разрядности.	2
10. Изучение принципа работы АЛУ при выполнении арифметических действий над числами.	2
11. Составление микропрограммы по управлению арифметико-логическим устройством	2
12. Построение ЗУ заданной организации на БИС ЗУ различного типа	2
13. Представление команд процессора в машинном виде.	2
14. Задание режимов адресации 16-разрядного микропроцессора Intel-8086.	2
15. Изучение порядка взаимодействия УУ, АЛУ и ОЗУ при автоматическом выполнении команд.	2
16. Использование мультипрограммирования при различных режимах работы вычислительных систем.	2
17. Распределение памяти и методы сокращения времени адресного преобразования.	2
18. Изучение алгоритмов буферизации и кэширования данных.	3
ВСЕГО:	**40**

Практическая работа № 1

Тема: «Оптимальная конфигурация устройств ЭВМ для решения конкретных задач»

Цель работы: знакомство с основными техническими характеристиками устройств персонального компьютера; знакомство с номенклатурой и символикой; знакомство с принципами комплектации компьютера при покупке ПК; получение навыков в оценке стоимости комплекта устройств ПК.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

При сборке компьютера из отдельных комплектующих необходимо учитывать два основных момента. Первый из них касается круга задач, для решения которых будет использоваться компьютер. Условно компьютеры можно разделить на несколько групп, в зависимости от их функционального назначения: офисные, учебные, игровые, домашние, мультимедийных и т. д.

Назначение компьютера определяет тот набор устройств, из которых он должен состоять, а также их основные характеристики. Например, для офисного компьютера совершенно необходимым должно быть наличие принтера, а игровому не обойтись без мощного процессора, большого объема оперативной памяти, качественной видеокарты с достаточным объемом видеопамяти и хорошего монитора.

Второй момент касается совместимости отдельных устройств с материнской платой. Прежде всего, это относится к совместимости по интерфейсу подключения. Существует несколько различных процессорных интерфейсов, для каждого из которых выпускаются свои модели материнских плат. Для процессоров фирмы Intel, например, использовались интерфейсы Socket 478, Socket 775 LGA, а для процессоров фирмы AMD — Socket A, Socket 754, Socket 939, Socket S-AM2. Поэтому при выборе материнской платы всегда, в первую очередь, следует обращать внимание на ее процессорный интерфейс.

Стандартным интерфейсом для подключения видеокарт на данный момент является шина PCI-Express (PCIe или PCI-E), PCI-Express 16x и PCI-Express 2.0 – наиболее используемые интерфейс для подключения дискретных видеокарт. Основное различием между этими версиями в том, что в версии 2.0 была увеличена максимальная пропускная способность до 8 Гбит/с в каждом направлении, а также увеличивает возможности энергоподачи до 300 Вт, для этого на видеокарты устанавливается 2-4-х штырьковый разъем пи-

тания. PCI-Express реализован в различных версиях, отличающихся пропускной способностью: 1x, 2x, 4x, 8x, 16x и 32x. Видеоинтерфейс PCI-E 16x обеспечивает пропускную способность равную 4 Гб/с в каждом направлении. Также были реализации PCI-Exp 8x (в бюджетных SLI- или CrossFire-решениях) и PCI-E 4x (или PCI-Express Lite).

Современная оперативная память обычно имеет тип DDR, DDRII или DDRIII и соответствующие интерфейсы подключения к материнской плате. Иногда на одной материнской плате могут одновременно присутствовать оба этих типа разъемов.

Жесткие диски подключаются по интерфейсам Serial ATA,Serial ATA II и Serial ATA III(обозначаются SATA, SATA II и SATA III). Существуют также переносные жесткие диски, подключаемые по интерфейсу USB.

Также следует учитывать, что устройства, имеющие одинаковый интерфейс, могут отличаться по пропускной способности, которая измеряется в мегабайтах в секунду или мегабитах в секунду. Надо обращать внимание на то, какую пропускную способность имеет данное устройство, и какую пропускную способность обеспечивает выбранная материнская плата. Если они не совпадают, то либо само устройство, либо материнская плата будет работать не в оптимальном режиме, что будет влиять на быстродействие всей компьютерной системы в целом.

При комплектации компьютера необходимо также учитывать, что некоторые компоненты могут быть встроены непосредственно в материнскую плату (видеокарты, звуковые карты, сетевые карты) и приобретение дополнительных аналогичных устройств может быть оправдано только в том случае, если они имеют лучшие характеристики, чем интегрированное устройство. Наличие встроенной звуковой карты можно определить по названию кодека, обычно Realtek, а встроенной сетевой карты — по обозначению LAN, после которого обычно указывается пропускная способность в мегабитах в секунду.

Расчет мощности блока питания (перейти по ссылке):
http://myfirst-comp.com/index.php/blok-pitaniya/45-rasschityvaem-moshhnost-bloka-pitaniya-kompyutera.

Задание 1.

Выполнить описание типичных конфигураций компьютера (информацию найти в сети Интернет.

Задание 2.

Подобрать комплектующие для компьютера, предназначенного для решения определенного круга задач (игровой компьютер, офисный компьютер). Подсчитать стоимость данного компьютера. Для подбора различных вариантов решения указанной задачи использовать табличный процессор (электронные таблицы). Все компоненты должны стыковаться с материнской платой по интерфейсу подключения и пропускной способности.

Для подбора компонентов Вы можете воспользоваться сервисом «Конфигуратор системного блока» на сайте: http://www.ulmart.ru (http://www.ulmart.ru/configurator.php#configer) или на сайте: key.ru http://key.ru/shop/devices/.

1. Офисная/«домашняя» (lowend) конфигурация. Такой компьютер, в первую очередь, предназначен для работы.

Сюда можно отнести использование сети Интернет, работу с документами, офисными приложениями (Word, Excel и др.), математическими пакетами (Mathcad, Maple). Возможно также прослушивание музыки, просмотр фильмов. Относительно неплохо будут работать «лёгкие» (с невысокими системными требованиями) или старые компьютерные игры. Сумма для приобретения 25 000 руб.

2. Бюджетная игровая конфигурация. Помимо всех вышеперечисленных возможностей, системный блок этой конфигурации неплохо «потянет» не очень требовательные современные компьютерные игры, а также обеспечит достаточно комфортную работу с аудиозаписями и фотографиями. Сумма для приобретения 30 000 руб.

3. Игровая конфигурация среднего класса (middle-end). При умеренной стоимости системного блока, пользователь получает компьютер, который способен успешно справиться с большинством современных компьютерных игр и имеет приблизительный запас производительности на будущие ~2-3 года (при условии такой же скорости развития компьютерных технологий, как в нынешнее время). Сумма для приобретения 35 000 руб.

4. Игровая конфигурация высокого класса. Такой компьютер отлично справится с самыми требовательными играми (например, с современными 3D-шутерами), обеспечит отличную производительность при обработке звукозаписей, а также поддержку DirectX 11 и выше. Такая конфигурация имеет хороший запас производительности на ближайшие ~3-5 лет. Сумма для приобретения 50 000 руб.

5. Топовая игровая конфигурация (high-end). Достаточно дорогая и очень мощная конфигурация для экстремальных геймеров и энтузиастов технологий, не жалеющих никаких денег на самые современные и мощные комплектующие. Сумма для приобретения 60 000 руб.

6. Конфигурация для видеомонтажа. Отдельно стоит упомянуть достаточно специфическую конфигурацию, наиболее оптимально подходящую для работы с видеозаписями. Упор в таком компьютере делается на мощность центрального процессора и количество оперативной памяти, в то время как видеокарта играет незначительную роль. Поэтому такой ПК, несмотря на

мощный процессор, не подойдёт для современных компьютерных игр. Сумма для приобретения 45 000 руб.

Результаты записать в виде таблицы

Таблица 1

№ п/п	Изображение компонента	Наименование компонента	Цена в руб.
1		процессор AMD Athlon II X2 245, ADX245OCK23GM, 2.90ГГц, 2МБ, Socket AM3, OEM	2750
	Итого		

Контрольные вопросы:

1. Какие устройства обеспечивают минимальный состав ПК?
2. Дайте классификацию и назначение различных видов памяти.
4. Что входит в состав основных компонентов материнской платы ПК?
5. Каково назначение шин ПК?
6. Перечислите основные характеристики шин ПК.
7. В чем отличие шины и порта ПК?
8. Какие параметры характеризуют производительность процессора?
9. Перечислите основные характеристики микросхем памяти.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. – Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 2

Тема: «Основные узлы персонального компьютера, разъемы для подключения внешних устройств»

Цель: изучение работы основных логических узлов ЭВМ.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Узлами ЭВМ являются стандартизированные наборы логических элементов, из которых набираются схемы, входящие в состав микропроцессоров, блоков памяти, контроллеров, внешних устройств и пр.

Узлы ЭВМ разделяются на:

- **Комбинационные** (автоматы без памяти), выходные сигналы которых определяются только сигналом на входе. Примером является дешифратор.
- **Последовательностные** (автоматы с памятью) - это узлы, выходной сигнал которых зависит не только от комбинации входных сигналов, но и от предыдущего состояния узла. Например, счетчики.
- **Программируемые** узлы, функционируют в зависимости от того, какая программа в них записана. Например, программируемая логическая матрица.

Многоразрядный сумматор процессора состоит из полных одноразрядных сумматоров. На каждый разряд ставится одноразрядный сумматор, причем выход (перенос) сумматора младшего разряда подключен к входу сумматора старшего разряда. Например, схема вычисления суммы двух двоичных трехразрядных чисел выглядит следующим образом:

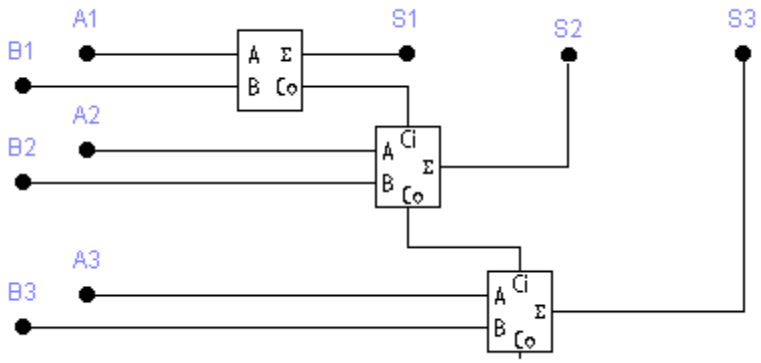


Рисунок 1. Схема трехразрядного сумматора

Для исследования сумматоров используем логический преобразователь. Подключив к нему полусумматор, последовательно нажимаем кнопки и получаем таблицу истинности, булево выражение.

Для анализа сумматора используется также генератор слова. К нему подключаются входы сумматора, а выходы подключаются к цифровому индикатору (Decoded Seven-Segment Display), расположенному в группе Indicators.

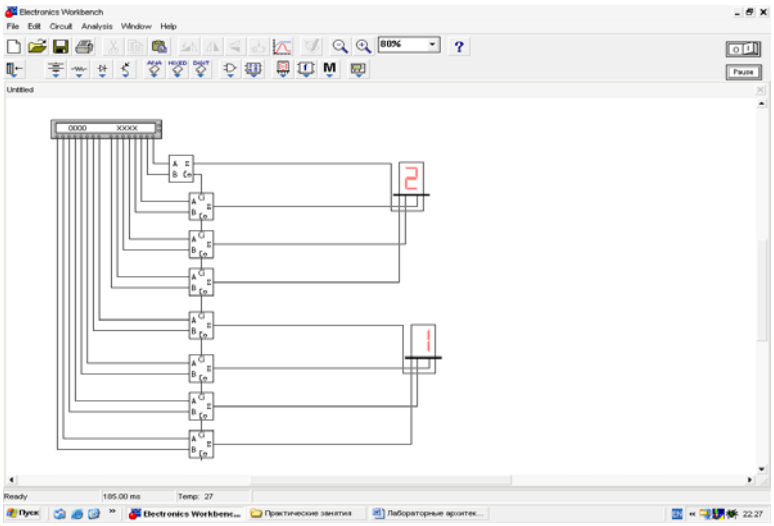


Рисунок 2. Схема восьмиразрядного сумматора

Регистры – это схемы хранения многоразрядных двоичных кодов. Основными типами регистров являются параллельные и сдвиговые. Параллельный регистр состоит из множества однобитовых элементов хранения информации, в которые можно параллельно записывать многоразрядный код.

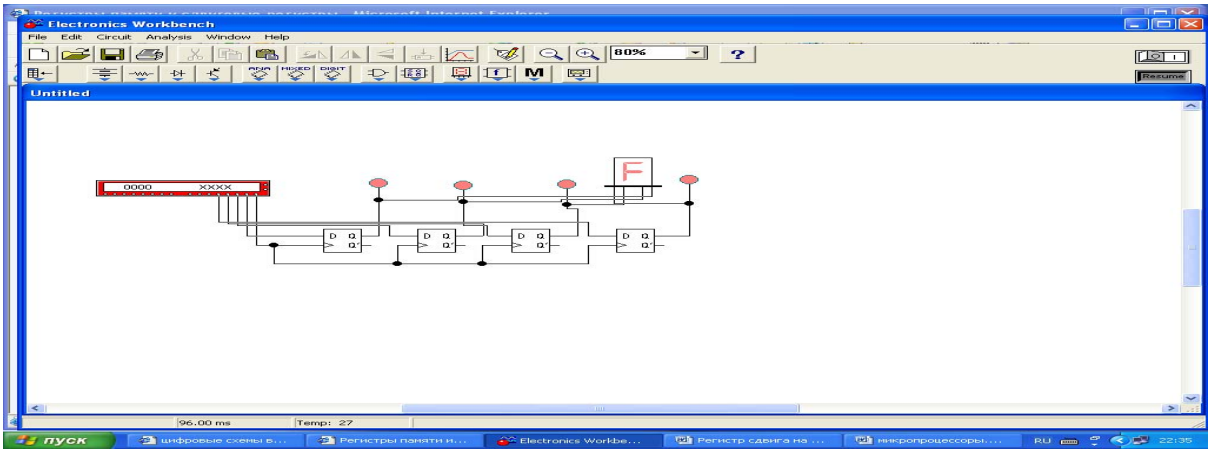


Рисунок 3. Параллельный регистр

Регистром сдвига называют цифровую схему, состоящую из последовательно включенных триггеров, содержимое которых можно сдвигать на один разряд влево или вправо подачей тактовых импульсов. Регистры сдвига широко применяются в цифровой вычислительной технике для преобразования последовательного кода в параллельный или параллельного в последовательный, а также при построении арифметическо-логических устройств. Составляется регистр сдвига из соединенных последовательно триггеров, в ко-

торые записываются разряды обрабатываемого кода. При наличии разрешающих сигналов импульс, приходящий на тактовый вход регистра, вызывает перемещение записанной информации на один разряд влево или вправо.

Информация в регистр сдвига может поступать последовательно и последовательно из него передаваться.

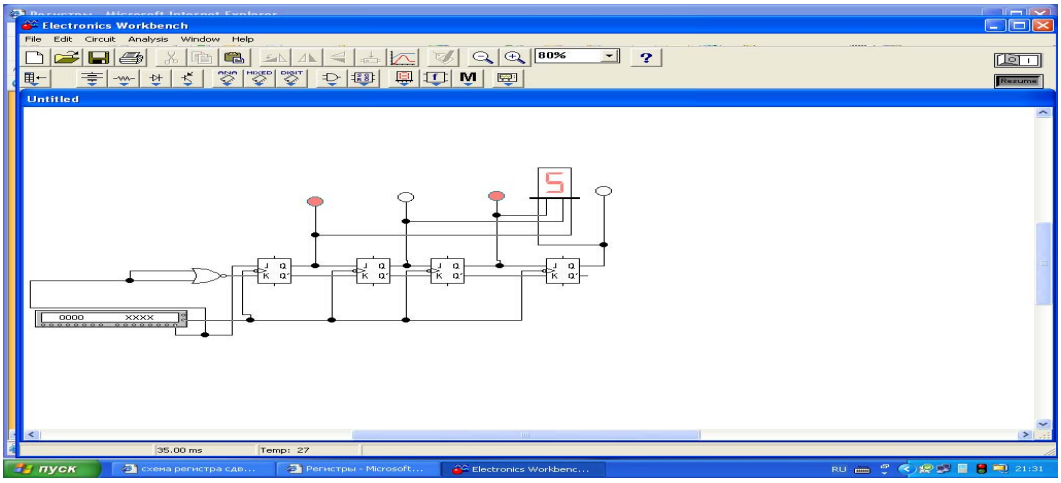


Рисунок 4. Регистр сдвига на JK-триггерах

Счетчик – это типовой узел цифровых устройств, предназначенный для подсчета количества входных сигналов. Он представляет собой регистр, двоичный код которого можно увеличивать на 1 при подаче входного сигнала.

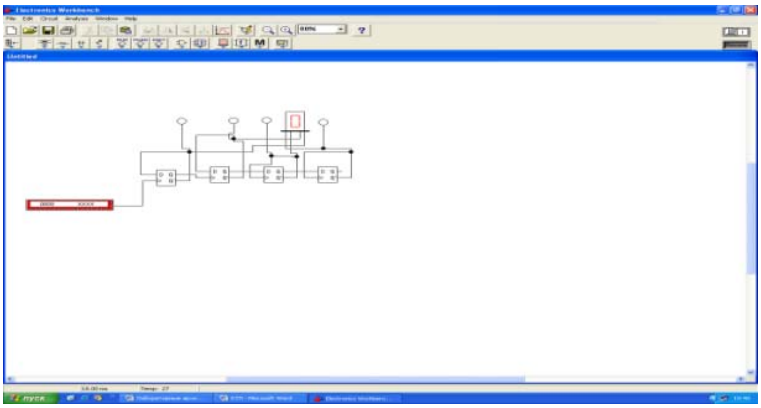


Рисунок 5. Счетчик последовательного типа

Мультиплексоры используются для коммутации в заданном порядке сигналов, поступающих с нескольких входных шин на одну выходную. Мультиплексоры применяются для выдачи на одни и те же выводы микропроцессора адреса и данных, что позволяет существенно сократить общее количество выводов микросхемы. В микропроцессорных системах управления мультиплексоры устанавливают на удаленных объектах для возможности передачи информации по одной линии связи от нескольких установленных на них датчиков.

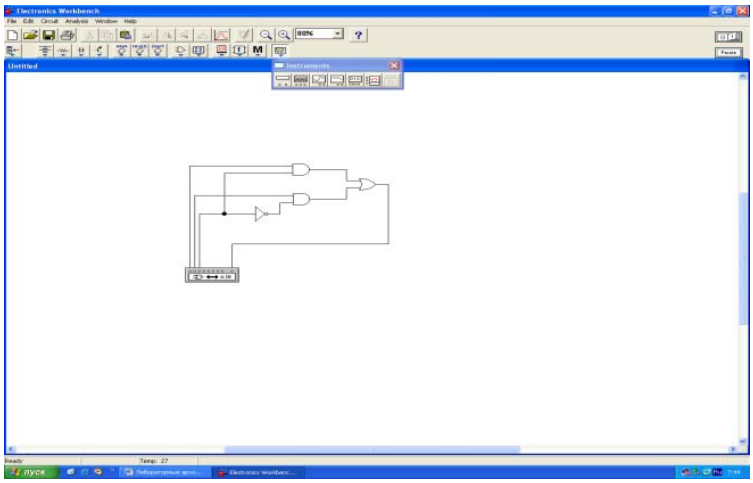


Рисунок 6. Схема двухканального мультиплексора

Шифраторы (кодеры) используются для преобразования десятичных чисел в двоичный или двоично-десятичный код.

Дешифратор (декодер) – устройство с несколькими входами и выходами, у которого определенным комбинациям входных сигналов соответствует активное состояние одного из выходов, т.е. дешифратор является обращенным по входам демультиплексором, у которого адресные входы стали информационными, а бывший информационный вход стал входом разрешения. Дешифраторы широко используются в информационно-измерительной технике и микропроцессорах управления в качестве коммутаторов-распределителей информационных сигналов и синхроимпульсов, для демультиплексирования данных и адресной логики в запоминающих устройствах, а также для преобразования двоично-десятичного кода в десятичный с целью управления индикаторными и печатающими устройствами.

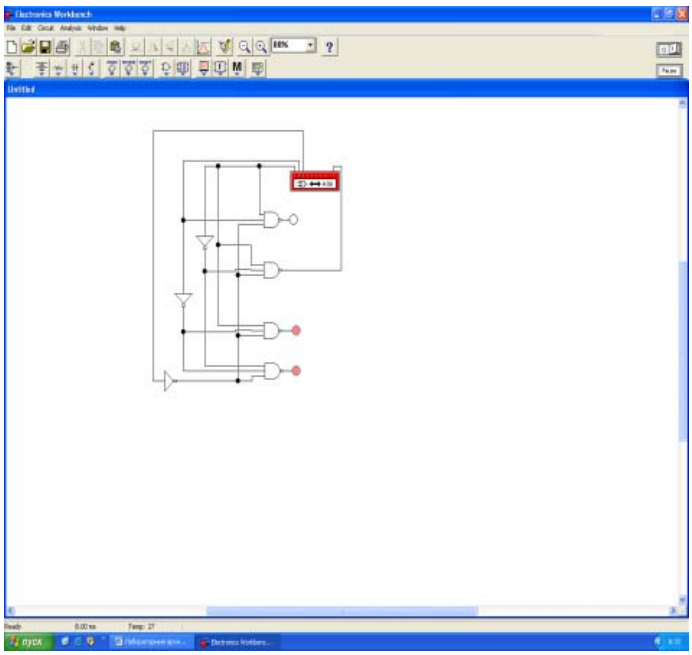


Рисунок 7. Схема дешифратора

Триггеры и сумматоры находятся в группе Digital.

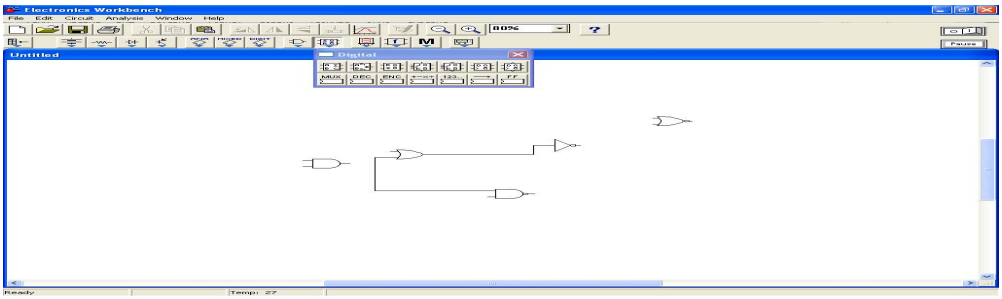


Рисунок 8. Триггеры, полусумматоры, одноразрядные сумматоры

Генератор слова позволяет изучать элементы с несколькими выходами. В генератор вносятся комбинации разных сигналов в шестнадцатеричном коде. Для ускорения ввода можно нажать кнопку Pattern и выбрать Up counter.

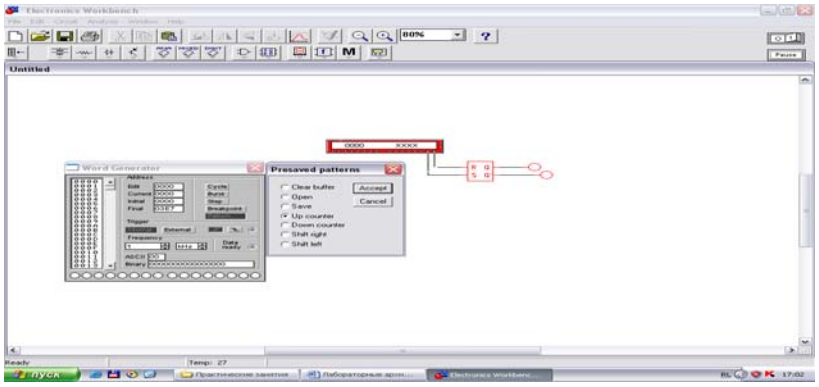


Рисунок 9. Генератор слова

При сборке схемы входы элементов и узлов подключаются к генератору слова, а к выходам подключаются световые индикаторы. Горящий индикатор означает 1.

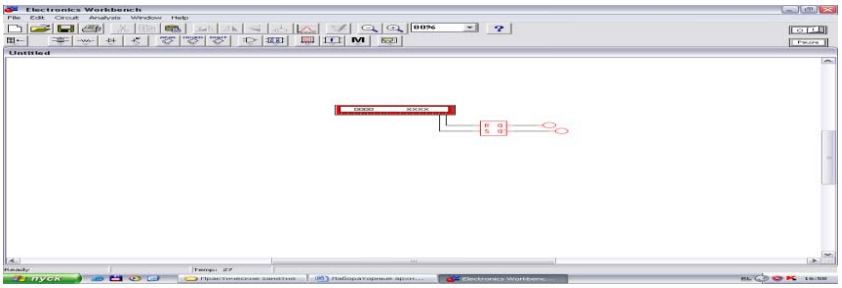


Рисунок 10. Схема для исследования триггера

Задания для практической работы:

Задание 1.

1. Соберите схему восьмиразрядного сумматора, выполните действия.

Таблица 1

Вариант	Задание	Вариант	Задание	Вариант	Задание
1	8+15	2	10+8	3	7+12
4	9+12	5	10+12	6	11+13
7	14+16	8	14+13	9	12+15
10	10+14	11	11+12	12	11+15
13	11+18	14	12+14	15	12+16
16	9+13	17	9+15	18	17+8
19	17+12	20	16+8	21	13+15
22	17+14	23	16+15	24	12+13
25	14+15	26	9+16	27	9+17
28	18+8	29	17+9	30	11+16
31	12+13	32	12+18	33	9+19
34	19+8	35	19+11		

Переведите результат в десятичную систему счисления.

Задание 2.

1. Соберите схему регистра сдвига и проведите ее испытание.
2. Соберите схему регистра-счетчика и проведите ее испытание.
3. Соберите схему регистра памяти и проведите ее испытание.
4. Соберите схему дешифратора и проведите ее испытание.
5. Соберите схему мультиплексора и проведите ее испытание.

Таблица 2

Вариант	Задание	Вариант	Задание	Вариант	Задание
1	2.1	2	2.3	3	2.5
4	2.2	5	2.4	6	2.1
7	2.3	8	2.5	9	2.2
10	2.4	11	2.1	12	2.3

13	2.5	14	2.2	15	2.4
16	2.1	17	2.3	18	2.5
19	2.2	20	2.4	21	2.1
22	2.3	23	2.5	24	2.2
25	2.4	26	2.1	27	2.3
28	2.5	29	2.2	30	2.4
31	2.1	32	2.3	33	2.5
34	2.2	35	2.4		

Контрольные вопросы:

1. Что из себя представляет дешифратор, при решении каких задач он используется?
2. Что из себя представляют счетчики, какого типа они бывают?
3. Что из себя представляет регистр, какие функции он выполняет?
4. Назовите типы регистров и их возможное применение.
5. Что из себя представляет мультиплексор, каково его назначение?

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 3

Тема: «Работа и особенности логических элементов ЭВМ»

Цель работы*:* изучение специального программного обеспечениея Electronic Work Banch; построение в программе Electronic Work Banch схемы триггера.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;

• лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

СИСТЕМА ELECTRONICS WORKBENCH.

Научно-техническое проектирование является основным в развитии науки и техники. Одним из направлений является компьютерное схемотехническое моделирование электронных устройств.

Использование интегрированных программных систем схемотехнического моделирования аналоговых и цифровых устройств (Micro-Cap V, DesignLab 8.0, Aplac 7.0, System View 1.9, Circuit Maker 6. радиоэлектронных 0, Electronics Workbench) позволяют решать **следующие задачи**:

- создание модели принципиальной электрической схемы устройства и ее редактирование;
- расчет режимов работы модели;
- расчет частотных характеристик и переходные процессы модели;
- провести оценку и анализ модели;
- наращивать библиотеку компонентов;
- представлять данные в форме, удобной для дальнейшей работы;
- разработка печатных плат;
- подготовку научно-технических документов и др.

Данный материал о системе Electronics Workbench 5.12 разработанный фирмой Interactive Image Technologies.

Особенностью системы является наличие контрольно-измерительных приборов, по внешнему виду и характеристикам приближенных к их промышленным аналогам. Система легко усваивается и достаточно удобна в работе.

Подробно изучит приемы работы с системой можно по специальным пособиям и руководствам. Мы ограничимся моделированием электрических схем из школьного курса физики и первоначальным знакомством с системой.

Знакомство с системой Electronics Workbench

Запустив интегрированный пакет Electronics Workbench, вы увидите диалоговое окно, и окно редактирования. Окно редактирования заполнено некоторыми компонентами. Диалоговое окно Electronics Workbench содержит поле меню, библиотеку компонентов и линейку контрольно-измерительных приборов расположенных в одном поле. Поле меню аналогичное с многими Windows-приложениями. Опции главного меню легко изучить самостоятельно.

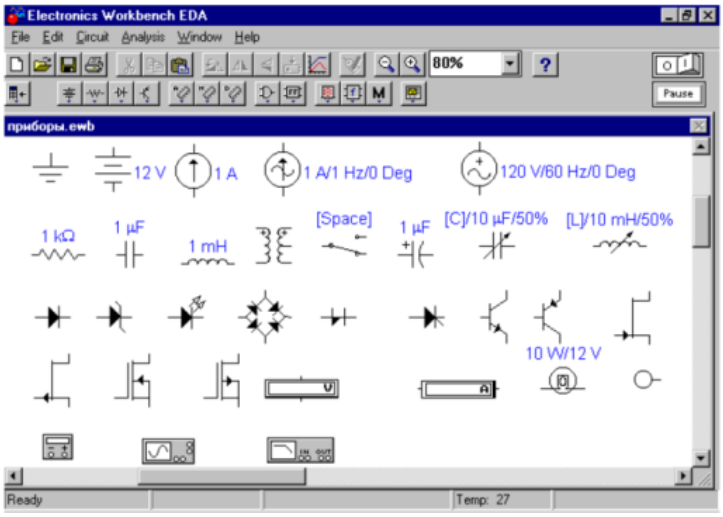


Рисунок 1. Панель инструментов

Несколько более подробно остановимся на некоторых компонентах и контрольно-измерительных приборах.

На рисунке 1. в окне редактирования, начиная слева сверху, двигаясь направо приведены обозначения следующих компонентов и контрольно-измерительных приборов: заземление, батарея, источник постоянного тока, источник переменного синусоидального тока (эффективное значения тока, частота, фаза), источник переменного синусоидального напряжения (эффективное значение тока, частота, фаза), резистор, конденсатор, катушка (индуктивность), трансформатор, переключатель, электролитический конденсатор, конденсатор переменной емкости, катушка переменной индуктивности, диод, стабилитрон, светодиод, диодный мост, диод Шок ли, n – p – n транзистор, p – n –p транзистор, далее 4 вида полевых транзисторов, вольтметр, амперметр, лампа накаливания (напряжение, мощность), светодиод (цвет свечения), мультиметр, осциллограф, измеритель амплитудночастотных и фазочастотных характеристик.

Алгоритм технологии подготовки и запуска электрических схем

1. Выбор необходимых компонентов электрической схемы и расположение их в окно редактирования Electronics Workbench. Для этого подводим указатель мыши к одной из пиктограмм библиотеки компонентов или линейке контрольно-измерительных приборов и щелкаем левой кнопкой мыши. Выпадает одна из выбранных групп компонентов. Для того, чтобы выбрать необходимый, подводим указатель мыши к компоненту, нажимаем левую кнопку мыши (не опускаем кнопку), перемещаем компонент на окно редактирования, опускаем кнопку.

2. Ввод и изменение параметров выбранных компонентов

Подводим указатель мыши к компоненту в окно редактирования и щелкаем два раза левой кнопкой мыши. Выпадает меню, состоящее из нескольких опций. Рассмотрим две из них:

Label – необходим для написания обозначения компонента;

Value – необходим для простановки значений компонента.

В контрольно-измерительных приборах, при необходимости, например в вольтметрах и амперметрах, при внесении параметров в опции Label, указываем для какого тока постоянного или переменного; в Mode выбираем DC – для постоянного тока, AC – для переменного.

3. Соединение компонентов электрической схемы

После размещения компонентов и простановки параметров производится соединение их выводов проводниками. При этом необходимо учитывать, что к выводу компонента можно подключить только один проводник. Для выполнения подключения указатель мыши подводим к выводу компонента и после появления жирной точки (указатель соединения) нажимаем левую кнопку мыши и появляющийся при этом проводник протягиваем к выводу другого компонента до появления на нем такой же жирной точки, после чего кнопку мыши отпускаем, соединение готово.

Если соединение нужно разорвать, указатель мыши подводим к одному из выводов компонента или к точке соединения и при появлении указателя соединения нажимаем левую кнопку, проводник отводим на свободное место рабочего поля, после чего кнопку отпускаем.

Если необходимо вывод компонента подключить к имеющемуся на схеме проводнику, то из вывода компонента проводник указателем мыши подводим к указанному проводнику и после появления точки соединения кнопку мыши отпускаем. Отметим, что прокладка соединительных проводов производится автоматически, причем препятствия – компоненты и проводники огибаются по ортогональным направлениям (по горизонтали или вертикали).

4. Подключение электрической схемы к питанию

В правом верхнем углу диалогового окна расположена пиктограмма 0 1

0 – отключено питание; 1 – включено питание.

После включения питания на контрольно-измерительных приборах регистрируются характеристики и значения собранной модели электрической схемы.

Любая информация в компьютере представляется в двоичном виде, поэтому рассмотрим запоминание и хранение элементарной порции информации - одного бита. Электронная схема, запоминающая один бит информации, называется **триггером**.

Триггеры – устройства, имеющие два устойчивых состояния. Под действием управляющих сигналов они переходят из одного состояния в другое и после снятия сигналов хранят это состояние до тех пор, пока не отключено напряжение питания. Таким образом, триггер является ячейкой памяти для одного двоичного разряда, т. е. бита информации. Для понимания процессов, происходящих в триггерах, приведем схему асинхронного однотактного RS – триггера на логических элементах И-НЕ.

В обычном состоянии на входы схемы подано постоянное напряжение 1.

При записи информации на один из входов подается напряжение 1. Посмотрим, как работает триггер. Пусть на вход 1 (Set - установить) подан сигнал «0», на вход 2 (Reset - переставить, сбросить) - «1». На выходе из элемента 1 (И-НЕ) независимо от другого входа элемента 1 появляется «1». На входы элемента 2 подаются «1», на выходе 2 появится «0».

Если на вход 1 подается сигнал «1», на вход 2 сигнал «0», то на выходе 1 сигнал «1», на выходе 2 - «0». Если на входы подать «0», то на выходах значение не изменится.

Назначение выводов триггеров следующее. Для всех триггеров выходы Q – прямой, Q’ – инверсный (обратный). Для RS – триггера R – установка триггера в 0, при сигнале 1 на этом входе Q=0, Q’=1; S – установка в 1, при сигнале 1 на этом входе Q=1, Q’=0; комбинация R=1, S=1 не изменяет состояние выходов, и относятся к **запрещенным**. Для JK триггера J, K – информационные входы, > - тактовый вход; вывод сверху – асинхронная предустановка триггера в единичное состояние (Q=1) вне зависимости от состояния сигналов на входах (функционально аналогичен входу S RS триггера); вывод внизу – асинхронная предустановка триггера в нулевое состояние (так называемая очистка триггера, после которой Q’=1); наличие кружочков на изображениях выводов обозначает, что активными являются сигналы низкого уровня, а для тактового входа – что переключение триггера производится не по переднему фронту тактового импульса, а по его срезу (так чаще всего называют задний фронт импульса). Для D – триггера вход D – информационный, состояние этого входа после подачи тактового импульса запоминается триггером, т. е. при D=1 имеем Q=1, при D=0 – Q=0.

Триггер имеет входы установки в 0 (R - вход, сигнал на инверсном выходе Q’=1) и 1 (S - вход, сигнал на прямом выходе Q=1). Установка триггера в 0 или 1 производится только при наличии сигнала синхронизации C=1. Возможные комбинации входных сигналов, имитирующие работу триггера в различных режимах, показаны на лицевой панели генератора слова.

Если схему триггера дополнить инвертором, то получим, в котором состояние выхода определяется сигналом на D - входе: при D=1 – Q=1, при D=0 –Q’=1. В качестве тактового сигнала используется выход синхросигнала генератора слова.

Если в D – триггере D – вход соединить с инверсным выходом Q’, то получится T – триггер с одним тактовым C – входом.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 4

Тема: «Схемная реализация элементарных логических операций»

Цель работы: освоение алгоритма построения таблиц истинности для логических функций; определение и анализ функции проводимости переключательных схем.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Построение таблиц истинности для логических функций

Алгебра логики – раздел математической логики, изучающий высказывания, рассматриваемые со стороны их логических значений (истинности или ложности) и логических операций над ними. Алгебра логики возникла в середине XIX века в трудах английского математика Джорджа Буля. Буль первым показал, что существует аналогия между алгебраическими и логическими действиями, так как и те, и другие предполагают лишь два варианта ответов – истина или ложь, нуль или единица.

На основе анализа логической связи между высказываниями делается логический вывод. Для получения логического вывода составляется **таблица истинности**, в которой записывают все возможные комбинации каждого простого высказывания.

Работа ЭВМ как автоматических устройств основана исключительно на математически строгих правилах выполнения команд, программ и интерпретации данных. Тем самым работа компьютеров допускает строгую однозначную проверку правильности своей работы в плане заложенных в них процедур и алгоритмов обработки информации. Это позволяет использовать математический аппарат для анализа и разработки логических устройств вычислительной техники.

Функцией логических переменных называют взаимосвязь логических переменных по законам логики. Значения входных переменных и выходных функций связаны некоторым преобразованием, которое реализует логическую функцию.

Логические операции:

1. Инверсия (логическое отрицание)

Операция, выражаемая словом «не», называется **логическим отрицанием** (инверсией) делает истинное выражение ложным и, наоборот, ложное – истинным. Обозначается « ».

Обозначение: НЕ, О*A*, $\bar{A}$, NOT *A*

Таблица истинности для логического выражения *A* имеет вид:

Таблица 1

A	$\bar{A}$
	1
	0

2. Конъюнкция (логическое умножение)

Операция, выражаемая связкой «**и**», называется **логическим умножением** (конъюнкцией) и обозначается « U» (может также обозначаться знаками «?» (точка) или &). Высказывание AUB истинно тогда и только тогда, когда оба высказывания *A* и *B* истинны.

Обозначение: **A и B, AUB, A?B, AANDB**

Таблица истинности для логических переменных A и B:

Таблица 2

A	***B***	***A/\B***
1	**1**	**1**
1	**0**	**0**
0	**1**	**0**
0	**0**	**0**

3. Дизъюнкция (логическое сложение)

Операция, выражаемая связкой «или» (в неисключающем смысле этого слова), называется логическим сложением (дизъюнкцией) и обозначается знаком U (или +). Высказывание *A* U *B* ложно тогда и только тогда, когда оба высказывания *A* и *B* ложны.

Обозначение: *A* ИЛИ *B*, *A*U*B*, *A*+*B*, *A* OR *B*

Таблица истинности для логических переменных A и B

Таблица 3

A	***B***	***A*U*B***
1	**1**	**1**
1	**0**	**1**
0	**1**	**1**
0	**0**	**0**

В алгебре логики любую логическую функцию можно выразить через основные логические операции, записать ее в виде логического выражения и упростить ее, применяя законы логики и свойства логических операций. По формуле логической функции легко рассчитать ее таблицу истинности. Необходимо только учитывать порядок выполнения логических операций (приоритет) и скобки.

Операции в логическом выражении выполняются слева направо с учетом скобок. **Приоритет выполнения логических операций**:

- инверсия,
- конъюнкция,
- дизъюнкция.

Задание 1.

Построить таблицу истинности для логической функции $F = A \wedge (B \vee \overline{B} \wedge \overline{C})$

1. Определить количество строк в таблице истинности, которое равно количеству возможных комбинаций значений логических переменных, входящих в логическое выражение: количество строк = 2n, где n – количество переменных.

Количество логических переменных – 3 (A, B, C) поэтому количество строк – 2n = 8.

Таблица 4

A	$\overline{B}$	C	$\overline{B}$	$\overline{C}$		$B \vee \overline{B} \wedge \overline{C}$	$A \wedge (B \vee \overline{B} \wedge \overline{C})$
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					

1	1	0					
$\overline{B} \wedge \overline{C}$ 1	1	1					

2. Определить количество столбцов:

- количество столбцов = количество переменных + количество операций.

Количество логических операций -5 (умножение – 2, сложение – 1, отрицание – 2), поэтому количество столбцов 3+5=8.

3. Построить таблицу истинности с указанным количеством строк и столбцов, обозначить столбцы и внести возможные наборы значений исходных логических переменных.

Таблица 5

A	B	C	$\overline{B}$	$\overline{B} \wedge \overline{C}$ $B \vee \overline{B} \wedge \overline{C}$	$A \wedge (B \vee \overline{B} \wedge \overline{C})$
0	0	0	1	1	0
0	0	1	1	0	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	0	1	1
1	1	1	0	1	1

4.Заполнить таблицу истинности по столбцам, выполняя базовые логические операции в необходимой последовательности и в соответствии с их таблицами истинности.

Задание 2.

Построить таблицы истинности для логических функций:

1) $F = A \vee \overline{B} \wedge (\overline{A \vee B})$.

2) $F = \overline{A} \wedge B \vee (\overline{A \wedge B})$.

3) $F = A \wedge B \wedge (C \vee \overline{A \wedge B})$.

Построение логических выражений для переключательных схем.

Переключательная схема — это схематическое изображение некоторого устройства, состоящего из переключателей и соединяющих их проводников, а также из входов и выходов, на которые подаётся и с которых снимается электрический сигнал.

В компьютерах и других автоматических устройствах широко применяются электрические схемы, содержащие сотни и тысячи переключательных элементов: реле, выключателей и т.п. При разработке схем используется аппарат алгебры логики.

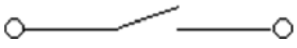
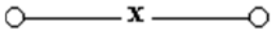
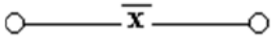
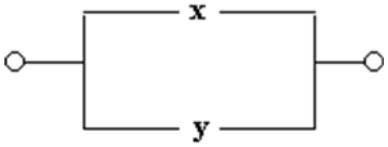
Каждый переключатель имеет только два состояния: замкнутое и разомкнутое. Переключателю **X** поставим в соответствие логическую переменную **x**, которая принимает значение 1 в том и только в том случае, когда переключатель **X** замкнут и схема проводит ток; если же переключатель разомкнут, то **x** равен нулю.

Будем считать, что два переключателя **X** и $\overline{X}$ связаны таким образом, что когда **X** замкнут, то $\overline{X}$ разомкнут, и наоборот. Следовательно, если переключателю X поставлена в соответствие логическая переменная **x**, то переключателю $\overline{X}$ должна соответствовать переменная $\bar{}$.

Всей переключательной схеме также можно поставить в соответствие логическую переменную, равную единице, если схема проводит ток, и равную нулю — если не проводит. Эта переменная является функцией от переменных, соответствующих всем переключателям схемы, и называется **функцией проводимости**.

Функции проводимости F некоторых переключательных схем:

Таблица 6

1)		Схема не содержит переключателей и проводит ток всегда, следовательно **F=1**;
2)		Схема содержит один постоянно разомкнутый контакт, следовательно **F=0**;
3)		Схема проводит ток, когда переключатель **x** замкнут, и не проводит, когда x разомкнут, следовательно, **F(x) = x**;
4)		Схема проводит ток, когда переключатель **x** разомкнут, и не проводит, когда x замкнут, следовательно, **(x) =** $\overline{x}$;
5)		Схема проводит ток, когда оба переключателя замкнуты, следовательно, **F(x,y) = x Uy**;
6)		Схема проводит ток, когда хотя бы один из переключателей замкнут, следовательно, **F(x,y)=x U y.**

Любая сложная схема может быть преобразована в отдельные группы и представлена в виде логических функций нескольких переменных.

Задание 3.

Определить и проанализировать функцию проводимости переключательной схемы.

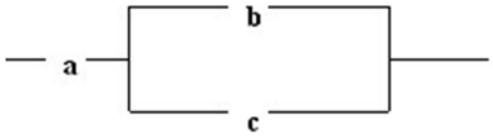


Рисунок 1. Переключательная схема

Функция проводимости имеет вид: **F (a, b, c) = a U (bUc).**

Анализируя таблицу истинности, можно сделать логический вывод, что для прохождения тока необходимо и достаточно, чтобы были замкнуты переключатели **a** и **b** или **a** и **c**, или все три **a**, **b**, **c**.

Задание 4.

Определить и проанализировать функции проводимости переключательных схем.

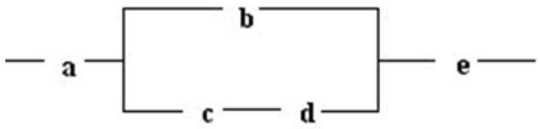


Рисунок. 2 Проводимость переключательной схемы а)

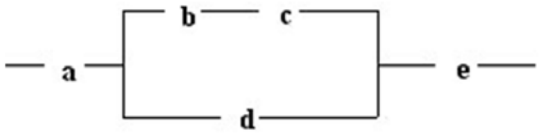


Рисунок. 3 Проводимость переключательной схемы в)

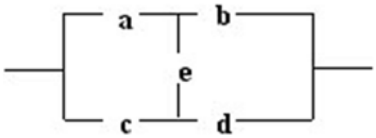


Рисунок 4. Проводимость переключательной схемы с)

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 5

Тема: «Работа логических узлов ЭВМ»

Цель работы: изучение работы основных логических узлов ЭВМ.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:
- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Узлами ЭВМ являются стандартизированные наборы логических элементов, из которых набираются схемы, входящие в состав микропроцессоров, блоков памяти, контроллеров, внешних устройств и пр.

Узлы ЭВМ разделяются на:

- **Комбинационные (автоматы без памяти),** выходные сигналы которых определяются только сигналом на входе. Примером является дешифратор.
- **Последовательностные (автоматы с памятью)** - это узлы, выходной сигнал которых зависит не только от комбинации входных сигналов, но и от предыдущего состояния узла. Н.р. счетчики.
- **Программируемые узлы**, функционируют в зависимости от того, какая программа в них записана. Н.р. программируемая логическая матрица.

Многоразрядный сумматор процессора состоит из полных одноразрядных сумматоров. На каждый разряд ставится одноразрядный сумматор, причем выход (перенос) сумматора младшего разряда подключен к входу сумматора старшего разряда. Например, схема вычисления суммы двух двоичных трехразрядных чисел выглядит следующим образом:

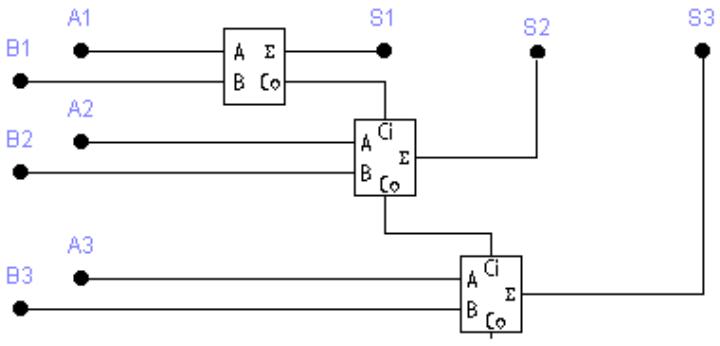


Рисунок 1. Схема трехразрядного сумматора

Для исследования сумматоров используем логический преобразователь. Подключив к нему полусумматор, последовательно нажимаем кнопки и получаем таблицу истинности, булево выражение.

Для анализа сумматора используется также генератор слова. К нему подключаются входы сумматора, а выходы подключаются к цифровому индикатору (Decoded Seven-Segment Display), расположенному в группе Indicators.

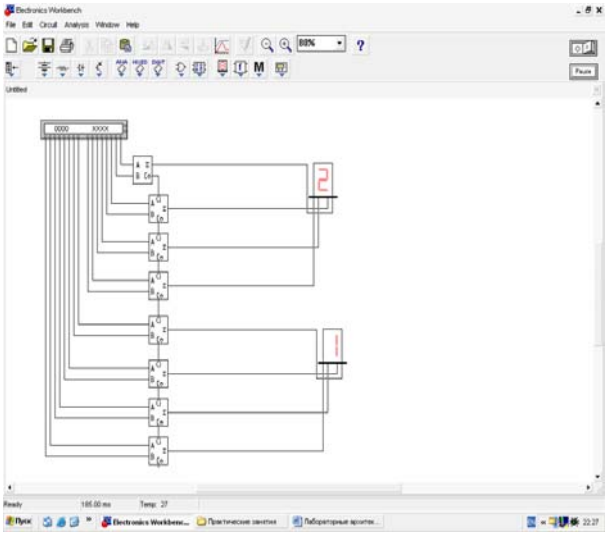


Рисунок 2. Схема восьмиразрядного сумматора

Регистры – это схемы хранения многоразрядных двоичных кодов. Основными типами регистров являются параллельные и сдвиговые. Параллельный регистр состоит из множества однобитовых элементов хранения информации, в которые можно параллельно записывать многоразрядный код.

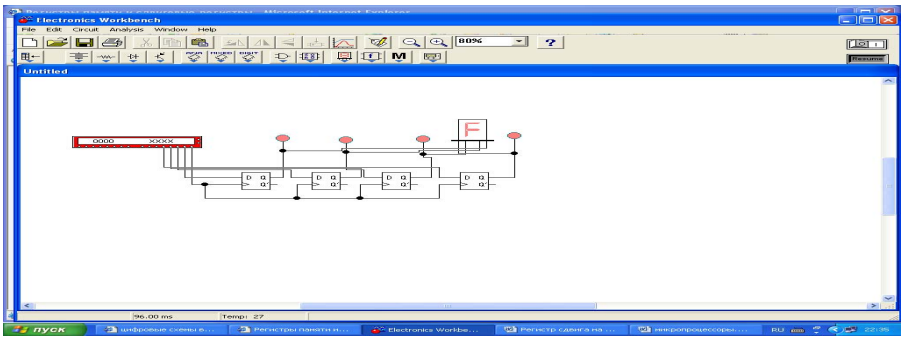


Рисунок 3. Параллельный регистр

Регистром сдвига называют цифровую схему, состоящую из последовательно включенных триггеров, содержимое которых можно сдвигать на один разряд влево или вправо подачей тактовых импульсов. Регистры сдвига широко применяются в цифровой вычислительной технике для преобразования последовательного кода в параллельный или параллельного в последовательный, а также при построении арифметическо-логических устройств. Составляется регистр сдвига из соединенных последовательно триггеров, в которые записываются разряды обрабатываемого кода.

При наличии разрешающих сигналов импульс, приходящий на тактовый вход регистра, вызывает перемещение записанной информации на один разряд влево или вправо.

Информация в регистр сдвига может поступать последовательно и последовательно из него передаваться.

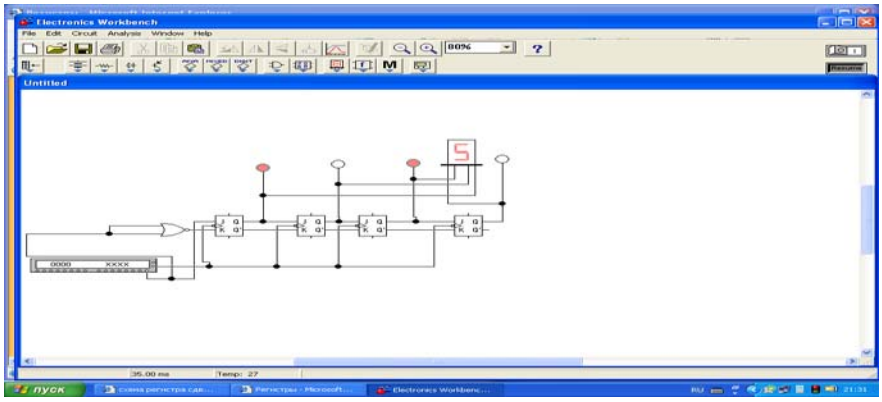


Рисунок 4. Регистр сдвига на JK-триггерах

Счетчик – это типовой узел цифровых устройств, предназначенный для подсчета количества входных сигналов. Он представляет собой регистр, двоичный код которого можно увеличивать на 1 при подаче входного сигнала.

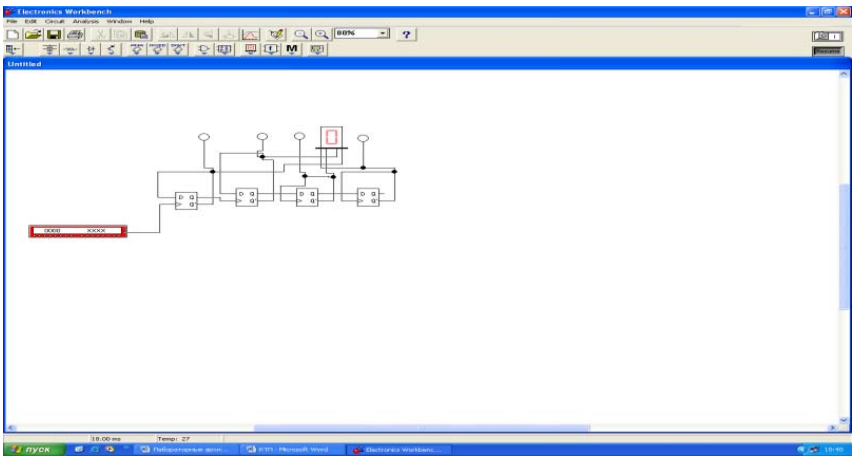


Рисунок 5. Счетчик последовательного типа

Мультиплексоры используются для коммутации в заданном порядке сигналов, поступающих с нескольких входных шин на одну выходную. мультиплексоры применяются для выдачи на одни и те же выводы микропроцессора адреса и данных, что позволяет существенно сократить общее количество выводов микросхемы.

В микропроцессорных системах управления мультиплексоры устанавливают на удаленных объектах для возможности передачи информации по одной линии связи от нескольких установленных на них датчиков.

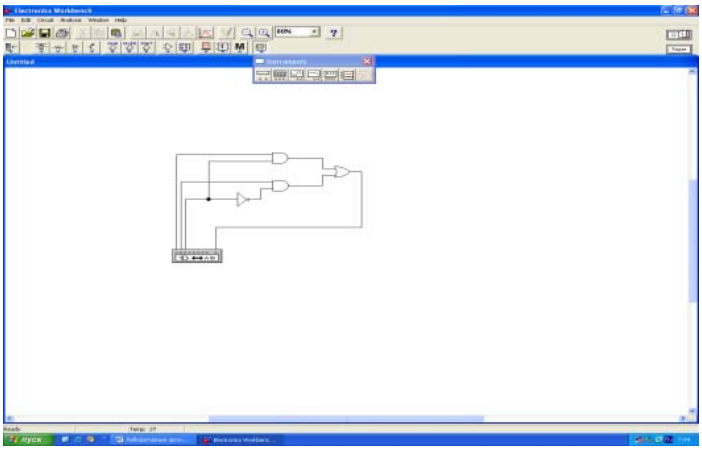


Рисунок 6. Схема двухканального мультиплексора

Шифраторы (кодеры) используются для преобразования десятичных чисел в двоичный или двоично-десятичный код.

Дешифратор (декодер) – устройство с несколькими входами и выходами, у которого определенным комбинациям входных сигналов соответствует активное состояние одного из выходов, т.е. дешифратор является обращенным по входам демультиплексором, у которого адресные входы стали информационными, а бывший информационный вход стал входом разрешения. Дешифраторы широко используются в информационно-измерительной технике и микропроцессорах управления в качестве коммутаторов-распределителей информационных сигналов и синхроимпульсов, для демультиплексирования данных и адресной логики в запоминающих устройствах, а также для преобразования двоично-десятичного кода в десятичный с целью управления индикаторными и печатающими устройствами.

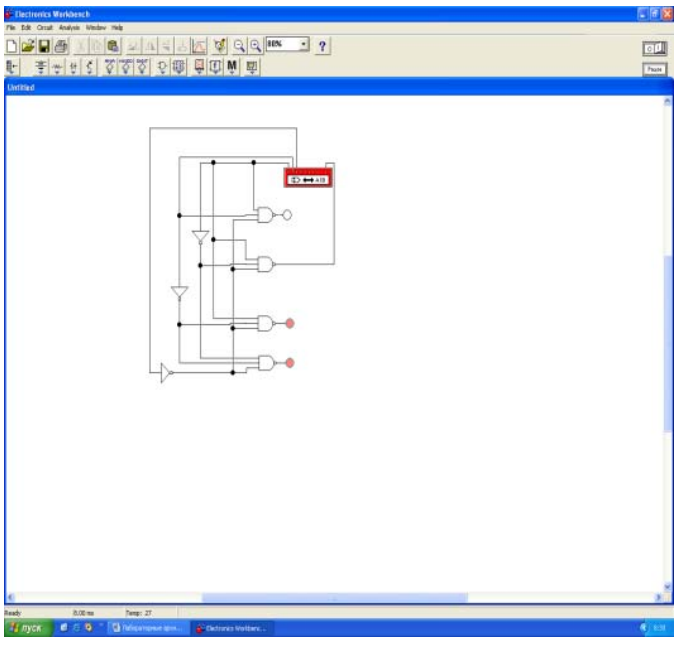


Рисунок 7. Схема дешифратора

Триггеры и сумматоры находятся в группе Digital.

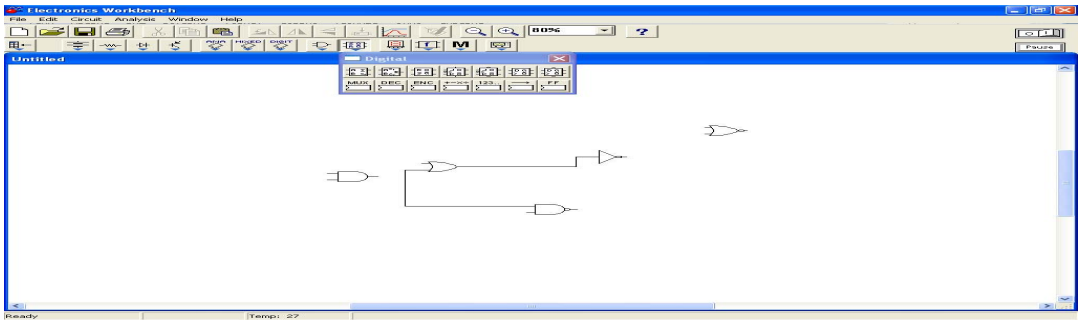


Рисунок 8. Триггеры, полусумматоры, одноразрядные сумматоры

Генератор слова позволяет изучать элементы с несколькими выходами. В генератор вносятся комбинации разных сигналов в шестнадцатеричном коде. Для ускорения ввода можно нажать кнопку Pattern и выбрать Up counter.

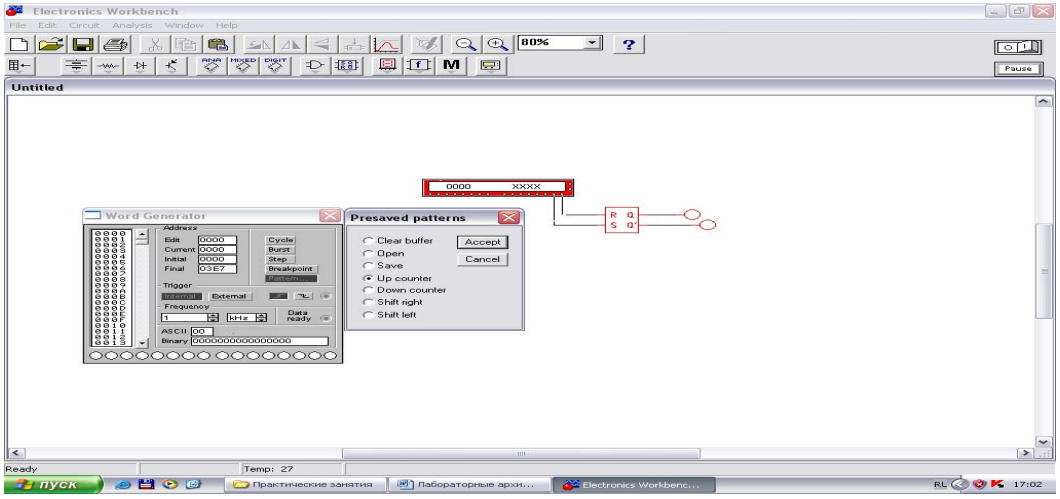


Рисунок 9. Генератор слова

При сборке схемы входы элементов и узлов подключаются к генератору слова, а к выходам подключаются световые индикаторы. Горящий индикатор означает 1.

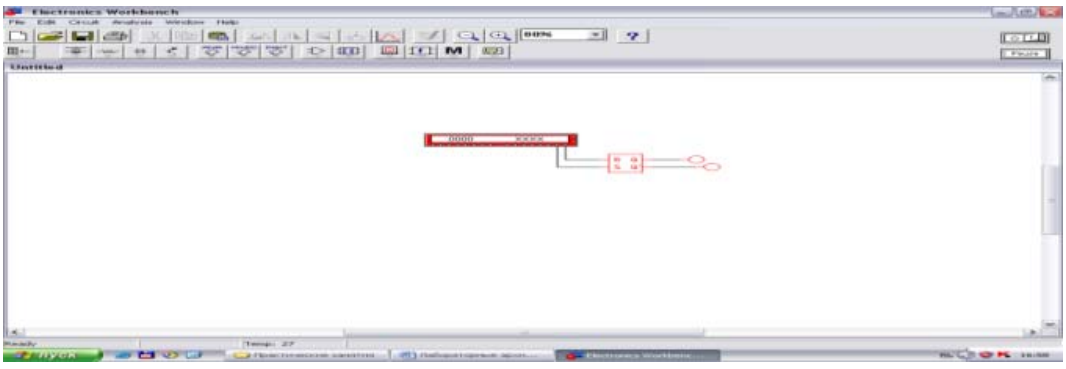


Рисунок 10. Схема для исследования триггера

Задания для практической работы:

Задание 1.

1. Соберите схему восьмиразрядного сумматора, выполните действия.

Таблица 1

Вариант	Задание	Вариант	Задание	Вариант	Задание
1	8+15	2	10+8	3	7+12
4	9+12	5	10+12	6	11+13
7	14+16	8	14+13	9	12+15
10	10+14	11	11+12	12	11+15
13	11+18	14	12+14	15	12+16
16	9+13	17	9+15	18	17+8
19	17+12	20	16+8	21	13+15
22	17+14	23	16+15	24	12+13
25	14+15	26	9+16	27	9+17
28	18+8	29	17+9	30	11+16
31	12+13	32	12+18	33	9+19
34	19+8	35	19+11		

Переведите результат в десятичную систему счисления.

Задание 2.

1. Соберите схему регистра сдвига и проведите ее испытание.
2. Соберите схему регистра-счетчика и проведите ее испытание.
3. Соберите схему регистра памяти и проведите ее испытание.
4. Соберите схему дешифратора и проведите ее испытание.
5. Соберите схему мультиплексора и проведите ее испытание.

Таблица 2

Вариант	Задание	Вариант	Задание	Вариант	Задание
1	2.1	2	2.3	3	2.5
4	2.2	5	2.4	6	2.1

7	2.3	8	2.5	9	2.2
10	2.4	11	2.1	12	2.3
13	2.5	14	2.2	15	2.4
16	2.1	17	2.3	18	2.5
19	2.2	20	2.4	21	2.1
22	2.3	23	2.5	24	2.2
25	2.4	26	2.1	27	2.3
28	2.5	29	2.2	30	2.4
31	2.1	32	2.3	33	2.5
34	2.2	35	2.4		

Контрольные вопросы:

1. Что из себя представляет дешифратор, при решении каких задач он используется?
2. Что из себя представляют счетчики, какого типа они бывают?
3. Что из себя представляет регистр, какие функции он выполняет?
4. Назовите типы регистров и их возможное применение.
5. Что из себя представляет мультиплексор, каково его назначение?

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 6

Тема: **«Идентификация и установка процессора»**

Цель работы: определение системной платы, определение компонентов системной платы, процессора, платы расширения.

Задачи:

1. изучить идентификацию системной платы, процессора, микросхем ROM и BIOS, шины и слотов расширения, RAM, видеоадаптера;

2. выполнить задания по теме;

3. оформить отчет по практической работе и представить преподавателю.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Перед началом работы

Включите системный блок и запишите основные параметры системы, которые выводит BIOS при загрузке:

- Версия BIOS;
- Объем ОЗУ;
- Обнаруженные устройства;
- Отключите компьютер от сети;
- Установите блок на стол и наденьте антистатический манжет;
- Проверьте, чтобы шнур питания был отключен от сети.

1. Идентификация системной платы

Шаг 1.

Снимите крышку корпуса.

Шаг 2.

Нарисуйте схему размещения разъемов и слотов расширения на системной плате, пометьте на ней ключи разъемов.

Определите назначение кабелей, подсоединенных к системной плате.

Запишите номера (идентификаторы) разъемов и слотов расширения и маркировки разъемов кабелей.

Отсоедините кабели и платы расширения.

! Эти номера и метки, возможно, трудно обнаружить и прочитать. Будьте внимательны.

Шаг 3.

Информация на веб-сайте производителя поможет определить информацию, связанную с типом чипсета, BIOS.

2. Идентификация процессора

CPU - основная часть компьютера. Вы должны определить, какой процессор установлен в системе, какие частоты процессоров системная плата

может поддерживать, какой тип сокета использован для установки процессора.

Шаг 1.

Какой тип сокета использован в системе?

_____

_____

Шаг 2.

Позволяет ли этот сокет модернизировать систему заменой процессора??

_____

_____

Шаг 3.

Какой тип процессора установлен?

_____

Шаг 4.

Совместим ли сокет с процессорами других производителей?

_____

_____

3.Идентификация микросхем ROM и BIOS

BIOS – BasicInput - Output System(базовая система ввода-вывода) – это микросхема ROM с записанными в нее командами, обеспечивающими самопроверку системы при включении и ее первоначальную загрузку.

Определите изготовителя и номер версии BIOS. Постарайтесь определить на веб-сайте производителя, имеются ли более новые версии этой системы команд.

Шаг 1.

Нанесите на схему место расположения микросхемы ROM BIOS.

Шаг 2.

Какой тип BIOS использован?

Шаг 3.

Можно ли произвести модернизацию BIOS?

_____

_____

_____

Шаг 4.

Что нужно сделать для модернизации BIOS?

_____

_____

_____

_____

4. Идентификация типа шины и слотов расширения

Слот расширения - длинный узкий разъем, расположенный на системной плате или дополнительной плате, устанавливаемой в некоторых моделях системной платы. На них могут размещаться такие устройства, как модем, звуковая плата, сетевая плата и др.

Шаг 1.

Определите местоположение слотов расширения, их количество, определите тип шины.

Какие устройства были установлены в слоты расширения?

_____

_____

4. Идентификация RAM

Random Access Memory(RAM) – память, используемая CPU для хранения данных во время исполнения программы. RAM не является постоянной, данные хранятся только во время включенного питания. RAM размещается на небольших платах расширения с разным числом краевых контактов.

Шаг 1.

Нанесите на схему местоположение слотов установки памяти.

Шаг 2.

Заполните таблицу.

Шаг 3.

Запишите информацию о линейках памяти.

6. Идентификация видеоадаптера.

Видеоадаптер – устройство, позволяющее компьютеру выводить изображения на экран монитора, задавать разрешение и цветопередачу монитора.

Контрольные вопросы:

1. Какой вид памяти используется CPU?
2. Что такое BIOS?
3. Назначение слота расширения?
4. Что такое видеоадаптер?
5. Какими параметрами характеризуются линейки памяти?

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –
Режим доступа: http://znanium.com/catalog/product/1017280
3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 7

Тема: «Построение памяти на RS – триггерах, JK – триггерах. Построение памяти на D – триггерах»

Цель работы: ознакомление с принципом работы триггеров и регистров, получение практических навыков в построении и контроле работоспособности триггеров и регистров, а также исследование логики работы триггеров и регистров в различных режимах методом моделирования с использованием программы Electronics Workbench.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:
- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Общие сведения об элементах памяти бортовых цифровых вычислительных устройств

Для построения цифровых устройств кроме логических элементов требуются элементы памяти, предназначенные для хранения двоичных кодов в течение требуемого времени.

В качестве статического элемента памяти используются бистабильные ячейки (БЯ), имеющие два устойчивых состояния. Бистабильные ячейки могут быть построены на двух логических элементах И-НЕ или ИЛИ-НЕ, соединенных перекрёстными связями.

В качестве элементов памяти используются так называемые триггеры.

Триггер - это цифровая электронная схема с двумя устойчивыми состояниями, которые устанавливаются при подаче соответствующей комбинации входных сигналов и сохраняются после снятия этих сигналов. Структурная схема триггера показана на рисунке 2. Триггер имеет несколько входов и два выхода - прямой и инверсный. Сигналы на выходах триггера всегда имеют различные значения. Если на прямом выходе сигнал равен 1, то на инверсном - 0 и наоборот. Состояние триггера определяется значением сигнала на пря-

мом выходе (Q). Если сигнал на прямом выходе равен 1, то триггер находится в состоянии 1.

Триггеры могут быть синхронными или асинхронными. Если изменения сигнала Q происходит только при наличии специального сигнала. C, являющегося сигналом синхронизации, то такой триггер называется синхронным триггером. Синхронизация триггера может происходить либо по уровню сигнала, либо по фронту сигнала (переднему или заднему).

Асинхронный триггер не имеет входа синхронизации, поэтому переключение триггера происходит только при поступлении на вход информационных входных сигналов X.

Логика переключения триггера из одного состояния в другое зависит от количества и назначения входов. Наиболее часто используются в цифровой технике следующие типы триггеров: RS-триггеры, JK-триггеры, D-триггеры и T-триггеры. Буквами R, S, J, K, D и T обозначаются информационные входы триггеров (X).

Асинхронные и синхронные триггеры разных типов

Асинхронные RS-триггеры

Асинхронный RS-триггер имеет два информационных входа - R и S. Вход S используется для установки триггера в состояние 1, а вход R - для установки в состояние 0.

Работа триггера описывается таблицей переходов, которая имеет вид таблицы 1.

Таблица 1

Входы		Состояния	
R	**S**	**Q(0)**	**Q(1)**
0	0	0	1
0	1	1	1
1	0	0	0
1	1	Не определено	

Из таблицы 1 может быть получено уравнение переходов триггера. После минимизации (например, с использованием карт Карно) уравнение переходов примет вид:

$$Q_{t+1}=S+\bar{R}\cdot Q_t$$

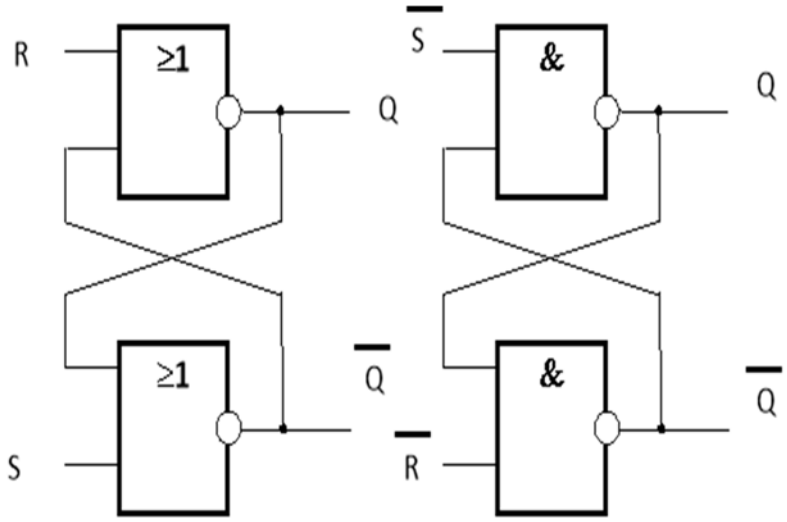


Рисунок 1. Уравнение перехода триггера

Из уравнения следует, что при S=1, R=0 всегда $Q_{t+1}=1$, при S=0, R=1 всегда $Q_{t+1}=0$, а при S=0, R=0 $Q_{t+1}=Q_t$. Комбинация сигналов S=1, R=1 является запрещенной, так состояние триггера не определено.

Для построения триггера на элементах И-НЕ уравнение необходимо преобразовать (двойным инвертированием) к другому виду:

$$Q_{t+1}=\overline{\overline{S}\cdot\overline{R}\cdot Q_t}$$

Для построения триггера на элементах ИЛИ-НЕ уравнение имеет вид:

$$\overline{Q_{t+1}}=\overline{S+R+Q_t}$$

Функциональные схемы асинхронных RS-триггеров, построенные на элементах ИЛИ-НЕ (слева) и И-НЕ (справа), и их условные графические обозначения (УГО).

Как видно из рисунка 1, асинхронный RS-триггер представляет собой бистабильную ячейку, построенную на элементах И-НЕ или ИЛИ-НЕ.

При построении RS-триггера на элементах И-НЕ действующими установочными сигналами являются инверсные значения информационных сигналов R и S.

Синхронные RS-триггеры

Синхронный триггер дополнительно имеет вход синхронизации C, на который поступает синхросигнал. Информационные сигналы R и S воздействуют на состояние триггера только при значении синхросигнала C=1.

Таблица переходов синхронного RS-триггера состоит из двух частей. Первая часть таблицы описывает переходы триггера при C=1 и совпадает с таблицей переходов асинхронного триггера (см. табл. 1), а вторая – при C=0.

При C=0 триггер не меняет своего состояния при любой комбинации сигналов на информационных входах R и S. В этом случае всегда $Q_{t+1}= Q_t$.

Уравнение синхронного RS-триггера имеет вид:

$$Q_{t+1}=S\cdot C+(\overline{R}+\overline{C})\cdot Q_t$$

$Q_{t+1}=S+R\cdot\overline{Q_t}$ Из уравнения следует, что при C=0 $Q_{t+1}= Q_t$, а при C=1т.е. работа описывается уравнением асинхронного триггера. На рисунке 1. приведены функциональные схемы синхронных RS-триггеров, реализованных на элементах И - НЕ для уравнения

$$\overline{Q_{t+1}}=\overline{S\cdot C}+\overline{(R\cdot C)\cdot Q_t}$$

и на элементах И-ИЛИ-НЕ для уравнения

$$\overline{Q_{t+1}}=\overline{S\cdot C+(R\cdot C)\cdot\overline{Q_t}}$$

На рисунке 4, кроме основных входов R и S, показаны дополнительные инверсные асинхронные входы R_1 и S_1.

Двухтактные RS-триггеры.

Триггеры в ЭВМ используются в различных узлах, между которыми осуществляется передача информации. Устойчивая работа цепочки триггеров возможна только в том случае, если запись новой информации в триггер осуществляется после считывания ранее записанной информации и передачи её в следующий по цепочке триггер. Это возможно при использовании двух серий синхроимпульсов, сдвинутых относительно друг друга на 180^o. Такой принцип управления и синхронизации применяется в двухтактных триггерах.

Простейшая схема двухтактного RS-триггера может быть построена на двух однотактных триггерах, причём синхроимпульсы на входы. С первого и второго триггеров должны подаваться в противофазе. Это делается с помощью инвертора (см. рис. 5).

При поступлении на вход первого однотактного триггера импульса C=1 информация на входах R и S устанавливает триггер в соответствующее новое состояние Q_{t+1}, а второй однотактный триггер хранит информацию о предыдущем состоянии Q_t, так как на его входе С сигнал равен нулю. По окончании действия синхроимпульса, т.е. при C=0, первый триггер переходит в режим хранения, а информация Q_{t+1}, записанная в первом триггере, передается во второй, так как на его входе С сигнал становится равным единице. В результате к началу следующего такта на выходе двухтактного RS-триггера появится сигнал, определяемый состоянием Q_{t+1} первого триггера. В таком триггере выходной сигнал формируется по заднему фронту синхроимпульса.

Двухтактный синхронный RS-триггер может быть использован для построения других типов триггеров, таких как D-, T- и JK-триггеров.

Для установки RS-триггера в 0 или 1 независимо от присутствия сигнала на входе С в схему вводят прямые или инверсные входы R и S асинхронной установки.

Асинхронный и синхронный D-триггеры.

В вычислительной технике широко применяется D-триггер, который реализует функцию временной задержки входного сигнала. D-триггер имеет

один информационный вход. Логика работы асинхронного D -триггера описывается таблицей переходов, которая имеет вид таблицы 2.

По таблице 2 может быть записано уравнение переходов D-триггера:

$\mathbf{Q_{t+1} = D_t,}$

где: t - текущий момент времени; t+1 – последующий момент времени.

Таблица 2

Вход	Состояния	
D	Q(0)	Q(1)
0	0	0
1	1	1

Как видно из уравнения, в асинхронном D-триггере состояние (выходной сигнал) $\mathbf{Q_{t+1}}$ повторяет значение входного сигнала $\mathbf{D_t}$. Поэтому асинхронный D-триггер по существу является не элементом памяти, а элементом задержки, и рассматривается только как основа для построения синхронного D-триггера.

Функциональная схема и УГО асинхронного D-триггера, построенного на основе асинхронного RS-триггера.

Для построения счётчиков, регистров и других цифровых схем используются синхронные D-триггеры как однотактные, так и двухтактные. Логика работы синхронного D-триггера описывается таблицей переходов, которая имеет вид таблицы 3.

Таблица 3

Входы		Состояния	
C	D	Q(0)	Q(1)
1	0	0	0
1	1	1	1
0	0	0	1
0	1	0	1

$$Q_{t+1} = C \cdot \overline{Q_t} + \overline{C} \cdot D$$

Уравнение переходов синхронного триггера, записанное по таблице 6.3, имеет следующий вид:

В соответствии с уравнением синхронный D-триггер при C=0 сохраняет свое состояние, а при C=1 работает как асинхронный.

Функциональная схема синхронного D-триггера на элементах ИЛИ-НЕ.

Функциональная схема двухтактного D-триггера, построенного на основе двухтактного RS- триггера.

Асинхронный и синхронный Т-триггеры.

Т-триггер имеет один информационный вход. Логика работы асинхронного Т-триггера может быть описана таблицей переходов, которая имеет вид таблицы 4.

Таблица 4

Вход	Состояния	
T	Q(0)	Q(1)
0	0	1
1	1	0

По таблице 4 может быть получено следующее уравнение асинхронного Т-триггера**:**

$$Qt+1 = \overline{T}Q_t + T\overline{Q_t}$$

Как видно из таблицы 4 и уравнения триггера, при Т=1 асинхронный Т-триггер меняет свое состояние на противоположное, а при Т=0 состояние триггера не изменяется.

Так как Т-триггер суммирует (или подсчитывает) по модулю два количество единиц, поступающих на его информационный вход, то Т-триггер называют также триггером со счетным входом.

Логика работы синхронного Т-триггера описывается таблицей переходов, которая имеет вид таблицы 5.

Таблица 5

Входы		Состояния	
C	T	Q(0)	Q(1)
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Из таблицы 5 видно, что при С=0 триггер не изменяет своего состояния, а при С=1 работает как асинхронный Т-триггер.

Функциональная схема Т-триггера может быть построена на основе синхронного RS-триггера (однотактного или двухтактного).

Схемы асинхронного Т-триггера и синхронного Т-триггера построены на основе синхронного двухтактного RS-триггера. Аналогичные схемы можно строить на основе однотактного RS-триггера. В двухтактных асинхронных Т-триггерах выходной сигнал формируется по заднему фронту входного сигнала Т, а в однотактных - по переднему фронту. В двухтактных синхронных Т-триггерах выходной сигнал формируется по заднему фронту сигнала С.

Схему асинхронного Т-триггера, в свою очередь, можно получить из D-триггера простой коммутацией входов и выходов.

JK-триггер

JK-триггер называется также универсальным триггером. Универсальность схемы JK-триггера состоит в том, что простой коммутацией входов и выходов можно получать схемы других типов триггеров.

JK-триггер имеет два информационных входа. Вход J используется для установки триггера в состояние 1, а вход К -для установки в состояние 0, т.е. входы J и К аналогичны входам R и S RS-триггера. Отличие заключается в том, что на входы J и К могут одновременно поступать сигналы 1. В этом случае JK- триггер изменяет свое состояние на противоположное.

Таблица переходов JK-триггера при C=1 имеет вид таблицы 6.

Таблица 6

Входы		Состояния	
J	K	Q(0)	Q(1)
0	0	0	1
0	1	0	0
1	0	1	1
1	1	1	0

Из таблицы 6 можно получить следующее уравнение JK-триггера:

$$Q_{t+1}=J\,\overline{Q_t} + K\,\overline{Q_t}$$

Следовательно, при J=1, K=0 всегда $Q_{t+1}=1$, а при J=0, K=1 всегда $Q_{t+1}=0$, т.е. JK-триггер работает как RS-триггер, если рассматривать входы J и K как входы S и R.

В свою очередь, при J=1, K=1_$Q_{t+1}=Q_t$, т.е. триггер переходит в противоположное состояние (работает как Т-триггер).

Функциональная схема двухтактного JK-триггера и УГО триггера. Примеры получения других типов триггеров на основе JK-триггера.

JK-триггер, кроме основных информационных входов и входа синхронизации, может иметь также дополнительные информационные входы, например, дополнительные инверсные асинхронные входы R и S, которые используются для установки триггера в 0 или 1 независимо от значения сигнала на входе синхронизации. Кроме того, триггер может иметь несколько входов J или K.

Регистры

Наиболее распространенным узлом цифровой техники и устройств автоматики являются **регистры**. Регистры строятся на базе синхронных одно- и двухступенчатых RS и D-триггеров. Регистры могут быть реализованы также на базе JK-триггеров.

Регистры с **параллельным** приемом и выдачей информации служат для хранения информации и называются регистрами **памяти** или **хранения**. Запись новой информации в регистр осуществляется после установки на входах

D0 ... Dm новой цифровой комбинации при поступлении синхроимпульса С. Количество разрядов записываемой цифровой информации определяется разрядностью регистра, которая, в свою очередь, определяется количеством триггеров, образующих этот регистр. Регистры памяти могут быть реализованы на D-триггерах, если информация поступает на входы регистра в виде однофазных сигналов и на RS-триггерах, если информация поступает в виде парафазных сигналов. В некоторых случаях регистры могут иметь вход для установки выходов в состояние «0». Этот асинхронный вход называют входом R «сброса» триггеров регистра.

Регистры с последовательным приемом или выдачей информации называются сдвиговыми регистрами или регистрами **сдвига**. Они могут выполнять функции хранения и преобразования информации (умножение и деление чисел двоичной системы счисления, преобразование параллельного кода в последовательный и наоборот и т.д.).

Порядок выполнения работы

Задание 1. Построить на элементах 2И-НЕ и 2ИЛИ-НЕ схемы асинхронных RS - триггеров и исследовать логику их работы в статическом режиме. Для этого собрать схемы с использованием пробников и переключателей.

Путем моделирования работы триггеров получить таблицы переходов и сравнить их с таблицей 1. Исследуемые схемы и таблицы занести в отчет.

Задание 2. Построить на элементах 2И-НЕ и 2-2И-2ИЛИ-НЕ схемы синхронных RS- триггеров и исследовать логику их работы в статическом режиме. В качестве элементов 2-2И-2ИЛИ-НЕ использована микросхема 7455, в которой располагается элемент 4-4И-2ИЛИ-НЕ. Исследуемые схемы и таблицы занести в отчет.

Задание 3. Исследовать в статическом режиме логику работы RS-триггера, который имеется в библиотеке программы. Для этого собрать схему. Получить таблицу переходов триггера и сравнить ее с таблицей 1. Исследуемую схему и таблицу занести в отчет.

Задание 4. Исследовать в статическом режиме логику работы двухтактного RS-триггера. Для этого собрать схему. Получить таблицу переходов триггера и сравнить ее с таблицей 1. Исследуемую схему и таблицу занести в отчет.

Задание 5. Исследовать в статическом режиме логику работы асинхронного D-триггера. Для этого собрать схему. Получить таблицу переходов триггера и сравнить ее с таблицей 3. Исследуемую схему и таблицу занести в отчет.

Задание 6. Исследовать в динамическом режиме логику работы асинхронного D-триггера. Для этого собрать схему. Для визуального наблюдения работы схемы установить частоту генератора 1 Гц. Зарисовать полученную осциллограмму. Исследуемую схему и таблицу занести в отчет.

Задание 7. Собрать и исследовать в статическом режиме схему синхронного D- триггера на элементе 2И-2И-2ИЛИ-НЕ, в качестве которого использовать микросхему 7451 с 2-мя элементами 2И-2И-2ИЛИ-НЕ.

Задание 8. Собрать и исследовать микросхему 7474, состоящую из 2-х синхронных D-триггеров. Результаты исследования занести в отчет.

Задание 9. Собрать схему и исследовать работу асинхронного T-триггера, построенного на базе синхронного D-триггера в статическом режиме. В качестве синхронного D-триггера использовать микросхему 7474 с дополнительными асинхронными входами установки и сброса (инверсные входы R и S). Результаты исследования занести в отчет.

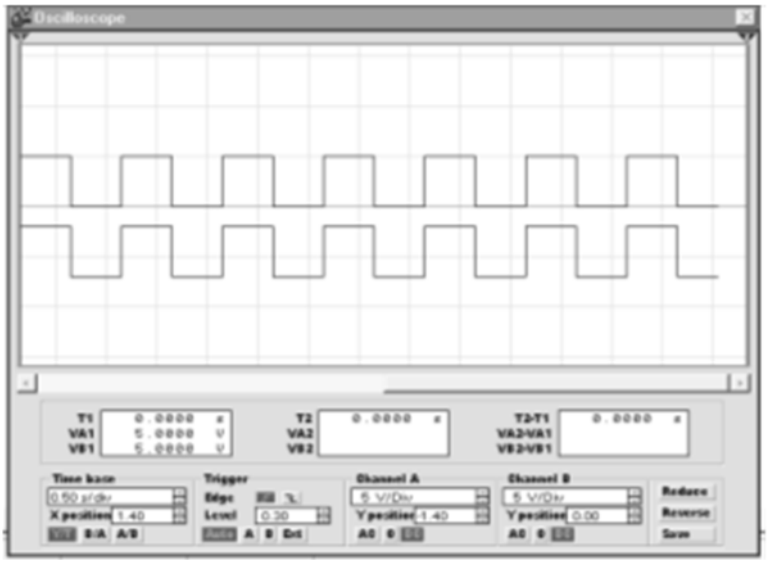


Рисунок 2. Схема асинхронного T-триггера

Задание 10. Исследовать работу синхронного JK-триггера в динамическом режиме. При подаче на входы J и K сигналов высокого уровня, а на вход синхронизации импульсов от генератора, триггер будет работать в режиме переключения с частотой в два раза ниже, чем частота генератора. Для визуальной индикации подключить осциллограф к выходам генератора и триггера.

Задание 11. Собрать схему и исследовать работу синхронного JK-триггера в статическом режиме. В качестве синхронного JK-триггера использовать микросхему 74112. Результаты исследования занести в отчет.

Задание 12.

1) Разработать и начертить схему электрическую функциональную четырехразрядного параллельного регистра на базе D-триггеров синхронизируемых фронтом для четных вариантов или на базе RS-триггеров, синхронизируемых фронтом для нечетных вариантов.

2) Разработать и начертить схему электрическую функциональную четырехразрядного регистра сдвига на базе на RS-триггеров, синхронизируемых фронтом, для четных вариантов или на базе D-триггеров, синхронизируемых фронтом, для нечетных вариантов.

3) Смоделировать параллельный регистр, в среде Electronics Workbench. Поочередно подать на входы D0 ... D3 код, соответствующий четырем младшим разрядам двоичного числа, равного номеру вашего варианта, и код на единицу меньший с помощью соответствующих ключей. Подать синхроимпульс С с помощью генератора слов Word Generation, включив его в ручном режиме **Step**, и убедиться в правильной работе параллельного регистра по состоянию логических пробников на его выходах.

4) Смоделировать регистр сдвига, разработанный в среде Electronics Workbench. Для имитации работы схемы подключить ее синхровход к генератору слов Word Generation, включив его в циклическом режиме **Sycle**. Подать на входы D0 ... D3 регистра код, соответствующий четырем младшим разрядам двоичного числа, равного номеру вашего варианта плюс три. Получить временные диаграммы входных и выходных сигналов сдвигающего регистра на экране логического анализатора Logic Analizer.

Контрольные вопросы:

1. Из каких логических элементов можно построить схему триггера?
2. Чем отличаются синхронные триггеры от асинхронных триггеров?
3. Можно ли построить схему D-триггера на основе RS- триггера?
4. Как построить схему Т-триггера, если использовать схему RS- триггера и логические элементы?
5. В каких случаях таблица переходов JK-триггера совпадает с таблицей переходов RS-триггера, в каких случаях отличается?
6. Почему JK-триггер называется универсальным триггером?
7. Почему Т-триггер называется триггером со счетным входом?
8. На какое время может быть задержана установка синхронного D-триггера по отношению к сигналу на его входе?
9. На какое время может быть задержана установка в 1 асинхронного D-триггера по отношению к сигналу на его входе?
10. Чем отличается двухтактный триггер от однотактного триггера?
11. Каково назначение регистров?
12. По каким признакам классифицируют регистры?
13. Чем определяется разрядность регистров?
14. Как работает параллельный регистр?
15. Каким образом осуществить операции умножения и деления в двоичной системе счисления в реверсивном регистре?

16. Как произвести с помощью регистра преобразование последовательного кода числа в параллельный код и обратно?

17. Как обозначаются регистры на схемах электрических функциональных и принципиальных?

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 8

Тема: «Построение регистров хранения различной разрядности»

Цель: освоение основных приемов создания различных видов регистров в программном комплексе «1С: Предприятие 8.0».

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Для анализа остатков и движений средств в системе «1С: Предприятие» используются регистры.

Регистр представляет собой многомерную систему хранения остатков или оборотов. Каждый регистр на этапе конфигурации описывается набором **измерений к ресурсам.**

Под **измерением** понимается набор значений, которые детализируют движения средств и в разрезе которых хранятся остатки. Под **ресурсом** понимается числовая величина, которая является количественным или суммовым значением, отражающим размер движения (остатка). Изменение остатков и оборотов по регистрам производится движениями регистров. Движения регистров записываются документами в момент проведения и имеют четко определенное место на оси времени, определяемое датой и временем документа. Каждый документ может порождать неограниченное количество движений по регистрам любых видов. Движения, записанные документом,

принадлежат ему и будут автоматически удаляться или изменяться при удалении или перепроведении документа.

Для работы в реальном времени система поддерживает точку актуальности итогов. Она может быть установлена принудительно, но ее могут изменять проводимые в потоке документы.

Проводимые в потоке документы получают мгновенный доступ к актуальным итогам по всем регистрам, например для контроля складских остатков. Хотя, разумеется, существует возможность проведения документа задним числом, с одной стороны, и получения итогов на любой момент — с другой.

Итоги по регистрам могут быть построены с любым набором разрезов исходя из измерений регистра. Так, в приведенном примере можно построить отчет о движении по товарам в разрезе складов или по складам в разрезе товаров.

Регистры сведений

Регистры сведений предназначены для хранения любой информации об объектах в разрезе заданных измерений, например регистр сведений Адресный классификатор. Если требуется хранить историю изменения информации, то регистр сведений делается периодическим, например «Курсы валют».

Регистры накопления

Регистры накопления накапливают числовую информацию в разрезе заданных измерений, например, регистры Выпуск продукции и услуг, Сведения о доходах, Остатки товаров компании. Движения регистров накопления всегда связаны с документами (регистраторами) и обычно создаются в момент проведения документа.

Регистры бухгалтерии

Регистры бухгалтерии хранят записи (проводки), основанные на определенном плане счетов, например регистр бухгалтерии хозрасчетный.

При редактировании регистра бухгалтерии разрабатывается структура регистра: создаются наборы измерений, ресурсов и реквизитов регистра. Отличительной особенностью регистра бухгалтерии является его связь с планом счетов и поддержка механизма двойной записи.

Каждая запись регистра содержит обязательные реквизиты «Счет Дт» и «Счет Кт» для регистров, поддерживающих корреспонденцию, и реквизит «Счет» - для не поддерживающих. **Измерений** у регистра бухгалтерии может быть несколько (например, Организация, подразделение). При определении структуры регистра создаются **ресурсы,** которые будут отражать числовую информацию (сумму проводки, количество).

У регистра бухгалтерии могут быть **реквизиты**. В них содержится дополнительная информация (например, «Комментарий»).

Планы видов расчета

Для описания алгоритмов, по которым выполняются те или иные вычисления, служит понятие «Виды расчетов». На этапе конфигурирования можно описать неограниченное количество видов расчетов. За понятием «вид расчета» не лежит реальных данных — это не более чем алгоритм вычисления, оперирующий данными регистров расчета, документов и справочников.

Алгоритм вида расчета описывается с помощью встроенного языка.

Для того чтобы при тех или иных расчетах можно было оперировать не только результатами расчетов по конкретным видам, но и результатами по нескольким видам расчетов, объединенных по определенному принципу, служит понятие планы видов расчета. В системе может быть определено неограниченное число планов видов расчета.

Планы видов расчета содержат виды расчета, объединенные по сходным признакам, к которым относятся одинаковые базовые виды расчета, одинаковые правила перерасчета, общие правила вытеснения по времени. В качестве примера можно привести планы видов расчета Основные начисления, Налоги.

Регистры расчета

Регистры расчета предназначены для хранения учетных записей сложных периодических расчетов, например регистр расчета Удержания. Каждый регистр расчета основан на каком-либо плане видов расчета.

В процессе конфигурирования настраивается неограниченное число регистров расчета, каждый из которых будет решать ту или иную задачу предметной области.

Непременными атрибутами каждой строки регистра расчетов являются: объект, для которого данный расчет проведен; вид расчета, по которому данный расчет проведен; дата начала и дата окончания действия данного расчета и результат расчета.

Для одного справочника могут быть созданы несколько регистров расчета, каждый из которых будет содержать данные определенной предметной области. Например, в том случае, если предприятие — акционерное общество закрытого типа, справочник сотрудников может выступать списком объектов расчета для журнала расчетов заработной платы и для журнала расчетов дивидендов акционеров.

Пример 1. Создание регистра бухгалтерии

Для отражения в бухгалтерском учете информации о хозяйственных операциях в системе 1С: Предприятие используются регистры бухгалтерии, описываемые на ветви дерева конфигурации «Регистры бухгалтерии».

При редактировании регистра бухгалтерии разрабатывается структура регистра: создаются наборы измерений, ресурсов и реквизитов регистра, если необходимо, создаются экранные и печатные формы просмотра движений регистра.

Отличительной особенностью регистра бухгалтерии является его связь с планом счетов и поддержка механизма двойной записи. Каждая запись регистра содержит обязательные реквизиты «Счет Дт» (счет дебета) и «Счет Кт» (счет кредита) для регистров, поддерживающих корреспонденцию, и реквизит «Счет» — для не поддерживающих.

В дереве конфигурации ветвь «Регистры бухгалтерии» и щелкнем по кнопке «Добавить». В первом окне Конструктора **(закладка «Основные»)** нужно заполнить поля ввода для имени (Наш Регистр Бухгалтерии) и синонима. Кроме этого, надо выбрать план счетов (Типовой) и включить опцию «Корреспонденция». Включение опции «Корреспонденция» означает, что каждая проводка имеет дебет и кредит, причем сумма проводки по дебету равна сумме проводки по кредиту.

Переходим в закладку «Данные» и добавляем ресурсы («Количество» и «Сумма»). Для ресурса Количество в палитре свойств указываем признак учета - количественный, признак учета субконто - суммовой.

В **закладке «Регистраторы»** помечаем «галочками» нужные документы (например, «Поступление товара» и «Расходная накладная».

Регистраторами являются те документы, которые будут влиять на содержание регистра (по которым будут формироваться проводки).

На закладке «Формы» создадим форму списка.

Закроем окно конструктора.

Настройка документов для формирования проводок

Щелкнем дважды по документу Поступление Товара. Откроется окно конструктора.

Перейдем в **закладку «Движения».** В ней регистр бухгалтерии уже выбран по умолчанию.

Щелкнем по кнопке «Конструктор движений». В нем счет дебета выберем 1330 (Товары), счет кредита — 3310 (Расчеты с Поставщиками).

В поле «Табличная часть» выберем «Товары». Щелкнем по кнопке «Заполнить выражения» (рис.1). Щелкнем по кнопке «ОК».

Аналогичную процедуру проведем с документом «Расходная Накладная» (счет дебета – 1210, счет кредита - 1330).

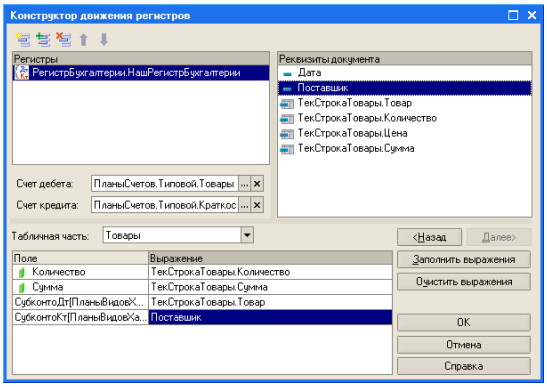


Рисунок 1. Конструктор движения регистров

Пример 2. Формирование регистров накопления

Создадим регистр накопления «Остатки», который будет хранить остатки товаров. Выделим в дереве конфигурации ветвь «Регистры накопления» и щелкнем по кнопке «Добавить».

В первом окне «Конструктора» (закладка **«Основные»)** нужно заполнить поля ввода для имени («Остатки») и синонима. Кроме этого, надо выбрать вид регистра («Остатки»).

Теперь **в закладке «Данные»** добавим одно измерение — Товар (тип данных — «Справочник Ссылка. Номенклатура») и один ресурс — «Количество».

На закладке «Регистраторы» пометим документы «Поступление Товара» **и** «Расходная накладная». Закроем окно редактирования.

Теперь создадим еще один регистр накопления «Продажи». Регистр будет хранить объем продаж за период в разрезе контрагентов и товаров, т.е. это будет регистр оборотов.

Для этого выделим в дереве конфигурации ветвь «Регистры накопления» и щелкнем по кнопке «Добавить».

В первом окне Конструктора (закладка «Основные») нужно заполнить поля ввода для имени («Продажи») и синонима. Кроме этого, надо выбрать вид регистра («Обороты»).

В этом регистре будут два измерения — Контрагент (тип данных — «Справочник Ссылка. Контрагенты») и Товар (тип данных — «Справочник Ссылка, Номенклатура»), и один ресурс — «Сумма».

Добавим их на закладке «Данные».

На закладке «Регистраторы» пометим документ «Расходная накладная».

Закроем окно редактирования.

Настройка процедуры проведения документов

Откроем окно редактирования документа «Поступление товара» (двойным щелчком).

Перейдем к закладке «Движения». Проверим, отмечены ли регистры, по которым этот документ может делать движения (регистр «Остатки»).

Далее создадим алгоритм проведения документа «Поступление Товара», при котором в регистре «Остатки» будет увеличиваться количество товаров. Для этого щелкнем по кнопке «Конструктор движений». Поскольку настройку движений этого документа мы уже делали при формировании проводок, то появится запрос «При запуске конструктора движений регистра, процедура «Обработка проведения» будет замещена. Продолжить?» на который надо ответить «Да».

Откроется окно конструктора движений регистров. В нем в левой верхней части располагаются регистры (Наш регистр «Бухгалтерия»). Добавим с помощью кнопки «Добавить» регистр накопления «Остатки». Выберем

тип движения регистра — «Приход». Табличную часть выберем — Товары. В правой части отображаются реквизиты документа и его табличных частей, в нижней части — измерения и ресурсы регистра. Щелкнем по кнопке «Заполнить выражения».

После нажатия на кнопку «ОК» конструктор движений сгенерирует текст программы на встроенном языке.

Закроем окно модуля объекта и окно редактирования документа.

Войдем в окно редактирования документа «Расходная накладная».

Перейдем к закладке «Движения». Проверим, отмечены ли регистры, по которым этот документ может делать движения (регистры «Остатки» и «Продажи»).

Документ «Расходная накладная» делает движения типа «Расход» по регистру «Остатки» и добавляет записи в регистр оборотов «Продажи». Для настройки этого щелкнем по кнопке «Конструктор движений». Появится запрос, на который надо ответить «Да».

Откроется окно конструктора движений регистров. В нем в левой верхней части добавим с помощью кнопки  «Добавить» регистр накопления «Остатки». Выберем тип движения регистра — «Расход». В табличной части выберем вкладку «Товары». Щелкнем по кнопке «Заполнить выражения».

Теперь добавим регистр накопления «Продажи». В табличной части выберем вкладку «Товары». Щелкнем дважды в правой части окна по реквизиту «Покупатель». Щелкнем по кнопке «Заполнить выражения».

После нажатия на кнопку «ОК» конструктор движений сгенерирует текст программы на встроенном языке.

Закроем окно модуля объекта и окно редактирования документа. Затем снова зайдем в окно редактирования документа.

Задание

1. Разработать структуру регистра сведений.
2. Упорядочить список измерений регистра сведений.
3. Разработать структуру регистров накопления (остатки, обороты).
4. Создать регистры бухгалтерии.

Контрольные вопросы:

1. Регистры сведений.
2. Свойства измерения (ресурса, реквизита) регистра сведений.
3. Регистры накопления.
4. Движения регистра накопления.
5. Итоги регистра накопления.
6. Регистры остатков и регистры оборотов.
7. Регистры бухгалтерии.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. — Москва : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. –
Режим доступа: http://znanium.com/catalog/product/1017280.
3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 9

Тема: «Изучение принципа АЛУ при выполнении арифметических действий над числами»

Цель работы: изучение принципа действия регистров и АЛУ; освоение методики вычислений и простейших математических операций.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:
- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Арифметико-логическое устройство (АЛУ) является основным функциональным узлом микропроцессора, предназначенным для обработки данных. АЛУ представляет собой комбинационную логическую схему, выполняющую логические и арифметические действия.

Для ввода, вывода и оперативного хранения информации, а также ее пошаговой загрузки по тактовому импульсу в АЛУ предназначен блок регистров: аккумулятор (А), буферные регистры (БР) или регистры общего назначения (РОН).

Совместная работа АЛУ и аккумулятора позволяет реализовать ряд арифметических и логических операций, в том числе сложение, вычитание, инверсию, сравнение, положительное или отрицательное приращение, сдвиг влево или вправо, логическое И, ИЛИ, исключающее ИЛИ и т.п. Из перечисленных элементарных операций набираются сложные задачи современной микропроцессорной техники.

Для изучения возможностей и имитации работы простого микропроцессора предназначена группа интегральных схем повышенной степени интеграции, входящих в состав лабораторного стенда.

Универсальная микросхема К155ИП3 представляет собой четырехразрядное АЛУ. Основу этой микросхемы составляют 30 простых логических элементов типа И-НЕ, И-ИЛИ-НЕ, исключающие ИЛИ и инверторов.

АЛУ может формировать 16 логических операций и выполнять 16 арифметических действий, включая суммирование и вычитание, увеличение

и уменьшение, удвоение и инверсию. Все операции производятся над четырехразрядными числами в двоичных кодах, причем одно из чисел подается на входы A3...A0, второе - на входы B3...B0, а результат вычислений появляется на выходах F3...F0.

Характер операций АЛУ зависит от уровня сигнала на входе режима *M*: при M=0 выполняются арифметические, а при M=1 - логические операции. Причем последние выполняются поразрядно. Вид выполняемых операций зависит от кода операции, подаваемого на управляющие входы S3...S0.

Ряд арифметических операций можно использовать либо для загрузки операндов (например, F=A или F=B для A=0 при S=0000 или S=1001 соответственно), либо для сложения в прямом коде F=A+B, или вычитания в дополнительном коде F=A-B, либо для более сложных действий.

При выполнении арифметических операций учитывается признак переноса с предыдущего разряда, подаваемый на вход $\overline{C0}$. При этом формируется признак переноса четвертого разряда $\overline{C4}$. Для удобства наращивания разрядности АЛУ при объединении нескольких микросхем вход и выход признаков переноса выполнены инверсными. При выполнении логических операций (логическое И, ИЛИ, исключающее ИЛИ и т.п.) с использованием прямых или инверсных кодов состояние входа переноса $\overline{C0}$ не влияет на полученные результаты.

Для расширения функциональных возможностей АЛУ предусмотрены выходы образования переноса G и распространения переноса P: первый переключается при достижении $1111_2=15_{10}$, а второй - при появлении переноса в любом из четырех разрядов.

Таблица 1

Код операции				Вид операции	
S3	S2	S1	S0	Арифметические (*M*=0)	Логические (*M*=1)
				$F=A+C_0$	__ F=A
				$F=A\acute{U}B+C_0$	_____ $F=A\acute{U}B$
				__ $F=A\acute{U}B+C_0$	__ $F=A\cdot B$
				$F=-1+C_0$	F=0
				$F=A+A\cdot\overline{B}+C_0$	_____ $F=A\cdot B$
				$F=A\cdot\overline{B}+(A\acute{U}B)+C_0$	__ F=B
				$F=A-B-1+C_0$	$F=A\ominus B$

				__ F=A·B-1+C_0	__ F= A·B
				F=A+A·B+C_0	__ F=AÚB
				F= A+B+C_0	_____ F= A $\oplus$ B
				__ F= A·B+(AÚB)+C_0	F=B
				F=A·B-1+C_0	F= A·B
				F= A+A+C_0	F=1
				F= A+(AÚB)+C_0	__ F= AÚB
				__ F= A+(AÚB)+C_0	F= AÚB
				F=A-1+C_0	F=A

Обозначения: Ú - логическое сложение; ·- **логическое умножение**;

$\oplus$ - сложение по модулю 2 (исключающее ИЛИ); + - арифметическое сложение; - - арифметическое вычитание; F=A+A - сдвиг влево на один разряд.

В схеме АЛУ предусмотрена также возможность сравнения операндов: если A=B, то на выходе K появляется уровень логической единицы.

Микросхема К155ИР11 представляет собой восьмиразрядный сдвигающий регистр, предназначенный для записи информации в параллельном или последовательном коде, ее хранения или сдвига влево и вправо. Основу этой микросхемы составляет 8 синхронизируемых триггеров в сочетании с логическими элементами И-НЕ, И-ИЛИ-НЕ и инверторами.

Схема имеет восемь входов D7...D0 параллельной записи, два входа последовательной записи DL - со сдвигом влево и DR - со сдвигом вправо, управляющие входы S0 и S1 для выбора режима работы, тактовый вход C и установочный вход $\overline{R}$ а также восемь выходов Q7...Q0.

Примечание: при переключении регистра сдвига в режим «хранение» необходимо кнопку «такт» держать в нажатом состоянии.

Если сигнал подавать на вход DL, то при S1=1 и S0=0 каждый тактовый импульс будет осуществлять ее последовательную запись с одновременным сдвигом влево. Как известно, такая операция эквивалентна умножению двоичного числа на два. При S1=0 и S0=1 по входу осуществляется последовательная запись со сдвигом вправо, что эквивалентно делению двоичного числа на два.

Естественно, что при отсутствии тактового импульса записанная информация будет храниться в регистре до прихода нового импульса, точнее, перепада 0/1. Вместе с тем, режим хранения обеспечивается и при установке S2=S1=0 как результат запрета на переключение триггеров.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 10

Тема: «Составление микропрограммы по управлению арифметико-логическим устройством

Цель: обучение составлению микропрограммы по управлению арифметико-логическим устройством.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Формирование микропрограммы по заданной ГСА рассмотрим на конкретном примере.

Составим микропрограмму для реализации ГСА

Объект управления характеризуется следующими параметрами:

– множество проверяемых условий

$X = \{ x_1, x_1, ..., x_{15} \}$;

– множество выполняемых микроопераций:

$Y = \{ y_1, y_2, ..., y_{120}\ y_k \}$, где(y_k — микрооперация, означающая последнюю микрокоманду микропрограммы);

–емкость памяти для записи микропрограмм

$V_{зу}$ = 2кбайт = 2•210байт;

–длина ячейки памяти

L = 16 бит;

–начальный адрес размещения составляемой микропрограммы в памяти равен

A_H = 530.

Исходя из характеристик управляемого объекта следует:

а) длина кода для кодирования микроопераций равна $к$ = 7, т.к. количество выполняемых в управляемом объекте микроопераций равно 120;

б) длина кода для кодирования условий равна p = 4, т. к. количество проверяемых условий в управляемом объекте равно15;

в) длина кода адреса равна p = 10, так как количество адресов в памяти, учитывая, что длина адресуемой ячейки равна 16 бит, т.е. 2 байтам, равно 1024.

Формат операционной микрокоманды (МКО) имеет длину 16 бит и включает:

1. поле типа микрокоманды (М), имеющее длину в один бит, занимает 0–ой разряд микрокоманды; в этом поле для данного типа микрокоманды записано значение «1»;

2. поле первой микрооперации (Y_1), которое занимает разряды с 1 по 7;

3. поле второй микрооперации (Y_2), которое занимает разряды с 8 по 14;

4. поле микрооперации $y_к$, которое используется только в последней микрокоманде.

–

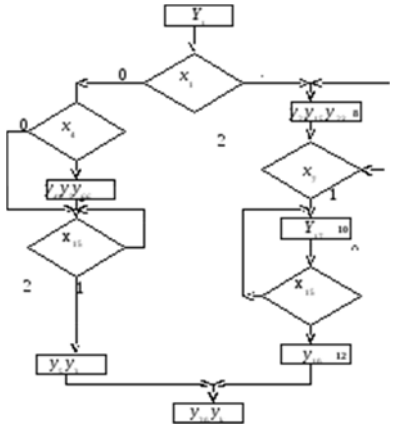


Рисунок 1. Граф - схема

Таким образом, формат микрокоманд для данного управляемого объекта имеет вид, показанный на рисунке 2.

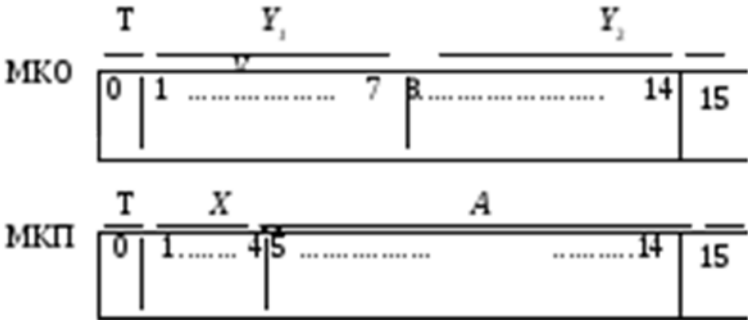


Рисунок 2. Форма микрокоманд

Таким образом, данный формат позволяет задавать при необходимости одной микрокомандой две микрооперации.

Формат микрокоманды перехода (МКП) имеет длину 16 бит и включает:

1. поле типа микрокоманды (Т), имеющее длину в один бит и занимающее 0–ой разряд микрокоманды; в этом поле для данного типа микрокоманды записано значение «0»;

2. поле проверяемого условия (*X*), которое занимает разряды с 1 по 4;

3. поле адреса (А), которое занимает разряды с 8 по 14;

4. поле модификатора дисциплины переход (М), имеющее длину в один бит и занимающее 15 разряд микрокоманды перехода; при М = 1 используется первая дисциплина перехода, при М = 0 – вторая.

При составлении микропрограммы необходимо реализовать с помощью микрокоманд все действия вершины, имеющиеся в ГСА, и обеспечить нужные ветвления процесса.

Микропрограмма, реализующая ГСА на рисунке 2., приведена в форме таблицы 1.

Таблица 1

Микропрограмма реализации граф – схемы алгоритма

№ п/ п	№ Вершины	Адрес расположения микрокоманды в ЗУ	Код микрокоманды	Примечание
	1	1000010010	1. 0000001. 0000000. 0	
	2	1000010011	0. 0001.1000011011. 1	3
	8	1000010100	1. 0010101. 0001111. 0	
	8’	1000010101	1. 0010111. 0000000. 0	
	9	1000010110	0. 0111.1000010100. 0	8
	10	1000010111	1. 0010001. 0000000. 0	
	11	1000011000	0. 1111.1000010111. 1	10
	12	1000011001	1. 0001010. 0000000. 0	
	7	1000011010	1. 0010100. 0000000. 1	
	3	1000011011	0. 0100.1000011110. 1	5
	4	1000011100	1. 0110000. 0000101. 0	
	4’	1000011101	1. 1000010. 0000000. 0	

	5	1000011110	0. 1111.1000011110. 0	5
	6	1000011111	1. 0000101. 0000011. 0	
	–	1000100000	0. 0000.1000011010. 1	7

В приведенной таблице:

1. в первой графе фиксируется номер строки;
2. во второй графе приводится номер вершины, реализуемой микрокомандой этой строки;
3. в третьей графе указан в двоичном коде адрес расположения данной микрокоманды в запоминающем устройстве;
4. в четвертой графе располагается код микрокоманд;
5. в пятой графе указаны номера вершин – ссылок, адрес которых должен быть отмечен в данной команде перехода.

В приведенной микропрограмме кодировка микроопераций и проверяемых условий осуществлена по их индексам в двоичном коде. Подчеркнутые коды адресов в микрокомандах перехода заполняются после записи последней строки формируемой микропрограммы, используя коды, указанные, а графе «Адрес» строк, соответствующих вершинам–ссылкам.

Микрокоманда **во второй** строке реализует первую вершину ГСА и поэтому записывается по адресу в ЗУ, соответствующему начальному адресу $A_н$, равному заданному начальному адресу 530 (в графе «Адрес расположения микропрограммы в ЗУ» в первой строчке записан двоичный эквивалент десятичного числа 530).

Данная микрокоманда реализует операторную вершину, поэтому в поле Т кода микрокоманды имеет место 1. В реализуемой вершине задается одна микрооперация y_1, поэтому в поле Y_1 записан двоичный семибитовый эквивалент ее индекса 1, в поле Y_2 записан двоичный эквивалент 0, а в поле $y_к$ записан 0, т. к. данная микропрограмма реализует не последнюю вершину ГСА.

Микрокоманда **в третьей строке** реализует вторую вершину ГСА. Она в любом случае выполняется вслед за микрокомандой, реализующей вершину №1, поэтому записывается по адресу в ЗУ, на единицу большему, чем адрес расположения в ЗУ микрокоманды, реализующей вершину №1. Данная микрокоманда реализует условную вершину исходного графа, поэтому в ней поле Т имеет значение 0. В поле X данной микрокоманды записан двоичный эквивалент индекса, проверяемого в реализуемой вершине условия x_1. При выполнении данного ветвления используется первая дисциплина перехода, поэтому в поле М данной микрокоманды установлена 1. При первой дисциплине перехода в поле А микрокоманды должен быть установлен адрес расположения в памяти микрокоманды, реализующей вершину 3 ГСА, расположенную по выходу «0» данной условной вершины. Поэтому в графе примечания записан номер этой вершины 3, а в следующем адресе находится микрокоманда, реализующая вершину 8, расположенную по выходу 1 данной

вершины 2. Поле адреса «А» в данной микрокоманде перехода и во всех других микрокомандах перехода первоначально не заполняется. Оно заполняется после того, как будут записаны в память все микрокоманды формируемой микропрограммы.

При реализации вершины 8 необходимо задать три микрооперации, что нельзя сделать с помощью одной операционной микрокоманды принятого формата. Поэтому данная вершина реализуется с помощью двух микрокоманд **(строки 4 и 5)**. Аналогичный случай имеет место при реализации вершины 4 (строки 12 и 13).

В **строке 10** представлена микрокоманда, реализующая последнюю вершину ГСА, поэтому в ее коде в поле y_k установлена единица. В следующем адресе ЗУ размещается микрокоманда, соответствующая вершине начала одной из еще не реализованной ветви ГСА (в данном случае это вершина 3).

После записи микрокоманды, реализующей вершину 6, необходимо расположить по следующему адресу в ЗУ микрокоманду, реализующую вершину 7. Однако вершина 7 уже представлена в микрокоманде (строка 10). Поэтому в следующем адресе **(строка 16)** записывается команда безусловного перехода к микрокоманде, реализующей вершину 7. Команда безусловного перехода реализована на базе микрокоманды перехода с первой дисциплиной перехода при задании для проверки кода несуществующего условия (в данном случае в качестве такого кода использован код 0000).

Любое управление может быть реализовано или на микропрограммном, или на аппаратном принципе.

При аппаратном принципе сложность устройства управления напрямую зависит от сложности алгоритма управления объектом. Кроме того, любое изменение алгоритма управления связано с необходимостью изменения аппаратной части синтезируемого устройства.

При микропрограммном принципе сложность устройства управления в основном влияет на объем микропрограммы, реализующей данный алгоритм. Изменения алгоритма управления в этом случае, как правило, связаны с изменением текста микропрограммы в памяти без изменения аппаратной части синтезируемого устройства управления. Однако микропрограммный принцип требует больших затрат времени, т. к. при реализации каждой вершины ГСА необходимо обращение к памяти для извлечения очередной микрокоманды.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 11

Тема: «Построение ЗУ заданной организацией на БИС ЗУ различного типа»

Цель работы: изучение построения ЗУ заданной организацией на БИС.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

В современных ЭВМ минимальной адресуемой единицей памяти является, как правило, 1 байт. В связи с этим обмен с памятью организуется блоками, кратными этой величине: байтами, словами, двойными словами, учетверенными словами, в зависимости от выполняемой процессором команды и разрядности внешней шины данных. Такой обмен проходит под управлением специальных сигналов, поступающих по системной шине. Преобразование информации из формата ее представления на шине данных в формат, учитывающий организацию конкретных схем памяти, осуществляется специальными интерфейсными схемами. Большие интегральные схемы (БИС), на которых строятся модули памяти, являются изделиями электронной промышленности и могут иметь различную организацию. Разработчики средств вычислительной техники должны учитывать имеющуюся у них номенклатуру БИС памяти, чтобы построить запоминающее устройство необходимой емкости и организации. Для этой цели может проводиться объединение нескольких БИС либо с целью увеличения количества слов в модуле памяти, либо для наращивания разрядности каждого слова, либо с той и другой целью одновременно. Рассмотрим варианты построения блока памяти необходимой организации при наличии заданных БИС памяти.

В реальных микросхемах шины данных записи и чтения (DI и DO) обычно представляют собой общую двунаправленную шину.

Сигналы на шине управления означают: MW - сигнал записи в память, MR - сигнал чтения из памяти.

Построить ОЗУ с организацией 1К*8 разрядов на БИС с организацией 1К*1 разряд (рис.1).

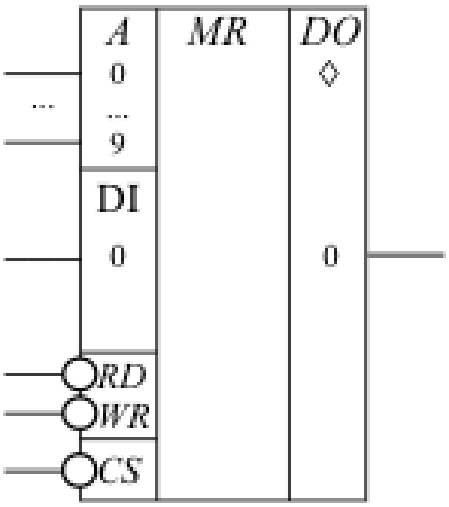


Рисунок 1. Условно-графическое обозначение БИС с организацией 1К*1 разряд

В данном случае требуется увеличить разрядность слова памяти. Так как все разряды одного слова должны записываться и считываться одновременно, то все БИС должны работать параллельно. Модуль памяти будет состоять из восьми БИС (рис. 1). Если разрабатываемый блок является частью модуля памяти, имеющего объем больше, чем 1К слов (например, 8К), то необходим специальный дешифратор, который будет дешифрировать старшие разряды адреса аналогично тому, как показано на рисунке 2 и включать в работу данный блок.

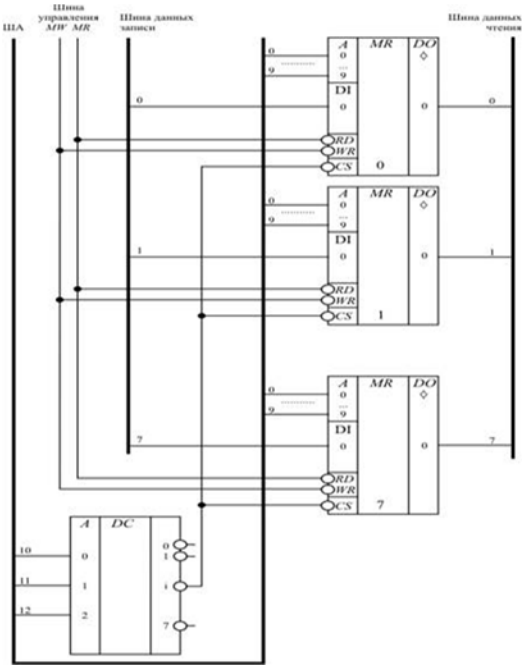


Рисунок 2. Запоминающее устройство объемом 1К*8 разрядов на БИС с организацией 1К*1 разряд

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –
Режим доступа: http://znanium.com/catalog/product/1017280.
3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 12

Тема: «Представление команд процессора в машинном виде»

Цель: знакомство с циклом работы процессора.

Формируемые компетенции: ОК 1 – 9; ПК 4.1, 4.4; ПК 3.2; ПК 1.2 - 1.6

Материально-техническое оснащение:
- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Для решения с помощью ЭВМ некоторой задачи должна быть разработана программа. Программа на языке ЭВМ представляет собой последовательность команд. Код каждой команды определяет выполняемую операцию, тип адресации и адрес. Выполнение программы, записанной в памяти ЭВМ, осуществляется последовательно по командам в порядке возрастания адресов команд или в порядке, определяемом командами передачи управления.

Для того чтобы получить результат выполнения программы, пользователь **должен**:
- ввести программу в память ЭВМ;
- определить, если это необходимо, содержимое ячеек ОЗУ и РОН, содержащих исходные данные, а также регистров IR и BR;
- установить в PC стартовый адрес программы;
- перевести модель в режим «Работа».

Каждое из этих действий выполняется посредством интерфейса модели. Ввод программы может осуществляться как в машинных кодах непосредственно в память модели, так и в мнемокодах в окно «Текст» программы с последующим ассемблированием.

Цель практической работы – знакомство с интерфейсом модели ЭВМ, методами ввода и отладки программы, действиями основных классов команд и способов адресации. Для этого необходимо ввести в память ЭВМ и выполнить в режиме Шаг некоторую последовательность команд (определенную

вариантом задания) и зафиксировать все изменения на уровне программно-доступных объектов ЭВМ, происходящие при выполнении этих команд.

Команды в память учебной ЭВМ вводятся в виде шестиразрядных десятичных.

В настоящей практической работе будем программировать ЭВМ в машинных кодах.

Пример 1.

Дана последовательность мнемокодов, которую необходимо преобразовать в машинные коды, занести в ОЗУ ЭВМ, выполнить в режиме Шаг. и зафиксировать изменение состояний программно-доступных объектов ЭВМ.

Таблица 1. Команды и коды

Введем полученные коды последовательно в ячейки ОЗУ, начиная с адреса 000. Выполняя команды в режиме Шаг, будем фиксировать изменения программно-доступных объектов (в данном случае это Асе, РС и ячейки ОЗУ 020 и 030).

Задание.

1. Ознакомиться с архитектурой ЭВМ.
2. Записать в ОЗУ «программу», состоящую из пяти. Команды разместить в последовательных ячейках памяти.
3. При необходимости установить начальное значение в устройство ввода IR.
4. Определить те программно-доступные объекты ЭВМ, которые будут изменяться при выполнении этих команд.
5. Выполнить в режиме «Шаг» введенную последовательность команд, фиксируя изменения значений объектов.
6. Если в программе образуется цикл, необходимо просмотреть не более двух повторений каждой команды, входящей в тело цикла.

Контрольные вопросы:

1. Из каких основных частей состоит ЭВМ, и какие из них представлены в модели?
2. Что такое система команд ЭВМ?
3. Какие классы команд представлены в модели?
4. Какие действия выполняют команды передачи управления?
5. Какие способы адресации использованы в модели ЭВМ? В чем отличие между ними?
6. Какие ограничения накладываются на способ представления данных в модели ЭВМ?
7. Какие режимы работы предусмотрены в модели и в чем отличие между ними?
8. Как записать программу в машинных кодах в память модели ЭВМ?

9. Как просмотреть содержимое регистров процессора и изменить содержимое некоторых регистров?

10. Как просмотреть и, при необходимости, отредактировать содержимое ячейки памяти?

11. Как запустить выполнение программы в режиме приостановки работы после выполнения каждой команды?

12. Какие способы адресации операндов применяются в командах ЭВМ?

13. Какие команды относятся к классу передачи управления?

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 13

Тема: «Задание режимов адресации 16 – разрядного микропроцессора Intel – 8086»

Цель работы: изучение режима адресации 16 – разрядного микропроцессора Intel – 8086.

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

В качестве примера мы выберем микропроцессор семейства 80X86 фирмы Intel. Этот микропроцессор, фактически, на многие годы стал эталоном архитектуры современных микропроцессоров, на нем построено множество ВС. Этот микропроцессор в модифицированном виде применяется и в настоящее время, поэтому материал кроме теоретического имеет и практическое значение.

Микропроцессор 8086 был представлен в 1978 и нашел применение как основа построения процессора микроЭВМ. Сегодня в мире имеются буквально миллионы систем основанных на этом микропроцессоре. С количеством **программного обеспечения**, написанного для 8086, не конкурирует никакая другая архитектура.

Однако в начале 80-х годов стало ясно, что замена для 8086 была необходима. Система на основе микропроцессора 8086 требовала множество дополнительных схем, чтобы выполнить даже несложную разработку. Фирма Intel почувствовала потребность интегрировать обычно используемые внешние устройства системы на тот же самый кремниевый кристалл, что и центральный процессор. В 1982 Intel удовлетворила эту потребность, представляя семейство микропроцессоров 80186. Первоначальный 80186 объединил расширенный центральный процессор 8086 с шестью обычно используемыми внешними устройствами системы. Параллельно Intel прикладывала усилия по разработке микропроцессора 80286. Этот микропроцессор начал путь к самой высокоэффективной архитектуре Intel, которая сегодня включает Intel386, Intel486 и Pentium микропроцессоры.

В 1987 Intel объявила второе поколение семейства микропроцессоров 80186: 80C186/C188. Семейство 80C186 совместимо с семейством 80186 и имеет расширенный набор элементов. Высокоэффективный процесс производства позволил вдвое повысить тактовую частоту и вчетверо снизить потребляемую мощность. Следующий важный шаг произошел в 1990 с введением в семейство 80C186EB. Центральный процессор 8086 был повторно разработан как статический модуль, внешние устройства семейства были также повторно разработаны как статические модули со стандартными сопряжениями.

В 1991 г. семейство 80C186 было снова расширено введением трех новых изделий: 80C186XL, 80C186EA и 80C186EC. Они отличаются сниженным энергопотреблением и повышенным быстродействием.

Функциональная схема микропроцессора 8086

Микросхема 8086 представляет собой однокристальный высокопроизводительный 16-разрядный микропроцессор с фиксированной системой команд. Микропроцессор предназначен для использования в качестве центрального процессорного устройства при построении средств вычислительной техники – от простейших одноплатных микроЭВМ до высокопроизводительных мультипроцессорных систем.

Микропроцессор обладает высоким быстродействием, обеспечивает возможность **прямой адресации памяти** объемом **до 1М байта**, **65536 устройств ввода** и **65536 устройств вывода**. Для вычисления адресов операндов, размещенных в памяти, используется **24 режима адресации**. Микропроцессор имеет **векторную структуру прерываний** и обеспечивает обработку до **256 запросов прерываний** трех типов: внешних, внутренних и программных.

Архитектурной особенностью микропроцессора 8086 является наличие аппаратно-программных средств, позволяющих упростить построение **мультипроцессорных систем** на его основе. Эти средства обеспечивают синхронизацию работы нескольких независимых (выполняющих собственные пото-

ки команд) процессоров, имеющих общие ресурсы, а также синхронизацию параллельной работы микропроцессора и сопроцессоров (специализированных процессоров, аппаратно реализующих команды сложных процедур). Микропроцессор 8086 характеризуется двумя режимами работы – **минимальным** и **максимальным**, которые отличаются способом формирования сигналов обмена и соответственно возможностями реализуемых систем.

Функциональная схема микропроцессора приведена на рисунке 1. Структура микропроцессора 8086 ориентирована на параллельное выполнение функций выборки и выполнения команд и состоит из **устройства сопряжения канала** (**УСК**), **устройства обработки** (**УО**) и устройства управления и синхронизации.

Устройство сопряжения канала предназначено для формирования физического адреса памяти, выборки команд из памяти и записи их в очередь команд, чтения операндов команд из памяти или регистров ввода/вывода, записи результатов выполнения команд в память или регистры ввода/вывода.

В УСК входят: **шесть** 8-разрядных **регистров очереди команд**; четыре 16-разрядных **сегментных регистра**; 16-разрядный **регистр адреса (указателя) команды**; 16-разрядный регистр обмена; 16-разрядный сумматор адреса.

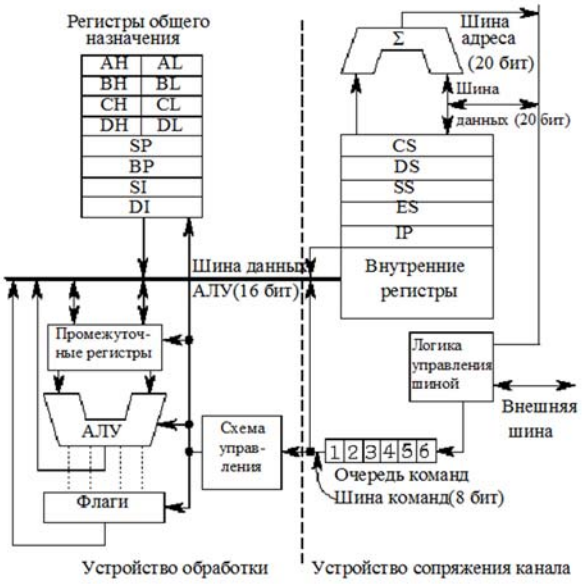


Рисунок 1. Функциональная схема микропроцессора

Устройство обработки предназначено для выполнения операций по обработке данных. Команды, выбранные из памяти и записанные в регистры очереди команд УСК, по запросам от УО поступают через 8-разрядную магистраль команд на микропрограммное устройство управления, которое декодирует команды и вырабатывает соответствующую последовательность микрокоманд, управляющую процессом выполнения текущей операции. УО не имеет непосредственной связи с внешней магистралью системы и обменивается данными через регистр обмена с УСК.

В устройство обработки входят: 16-разрядное **арифметико-логическое устройство**, восемь 16-разрядных **регистров общего назначения**, 16-разрядный **регистр признаков** состояния микропроцессора.

Команды всегда выбираются из памяти как слова, независимо от четности или нечетности адреса, по которому производится чтение команды.

Отличительной особенностью 8086 является возможность **аппаратной перестройки внутренней структуры** схемы управления и синхронизации.

Выбор режима функционирования этой схемы предоставляет разработчику системы возможность выбора подмножества выходных управляющих сигналов в соответствии со степенью сложности проектируемой микропроцессорной системы. Системная настройка обеспечивается специальным выводом выбора режима **MN/MX**.

Микропроцессор позволяет обрабатывать 256 типов прерываний с номерами от 0 до 255, которые делятся на внешние аппаратные, внутренние аппаратные и программные. Запросы на внешние прерывания формируются внешними по отношению к микропроцессору устройствами. Запросы на внутренние прерывания формируются при выполнении определенных команд или по некоторым условиям при выполнении команд. По любому прерыванию управление передается программе (процедуре) обслуживания прерывания посредством вектора прерывания, выбираемого из таблицы векторов прерывания, располагаемой в памяти.

Запросы на внешние прерывания воспринимаются и обрабатываются после выполнения текущей команды. **Внешние прерывания** поступают на микропроцессор по двум внешним выводам (INT и NMI) и делятся на **маскируемые** и **немаскируемые**.

Вопросы, связанные с использованием режима прерываний обсуждаются более подробно далее в шестой главе, которая посвящена принципам организации обмена данными между микропроцессором и внешними устройствами.

Интерфейсные сигналы микропроцессора, циклы обмена с шиной

Микропроцессор взаимодействует с системой посредством внешних интерфейсных сигналов. Эти сигналы могут быть разделены на три группы: **адрес/данные**, **управление** и **служебные**. Назначение некоторых выводов микропроцессора зависит от его режима работы. Режим работы определяется специальным сигналом **MN/MX**. Минимальный режим включается при подаче на вход MN/MX логической 1, максимальный – при подаче на вход MN/MX логического 0. Графическое представление микропроцессора показано на рисунке 2. На этом рисунке отрицание названия сигнала подразумевает, что активный уровень сигнала низкий. Обозначения выводов микропроцессора в минимальном режиме, если функции выводов в минимальном режиме отличаются от их функций в максимальном, даются в круглых скобках.

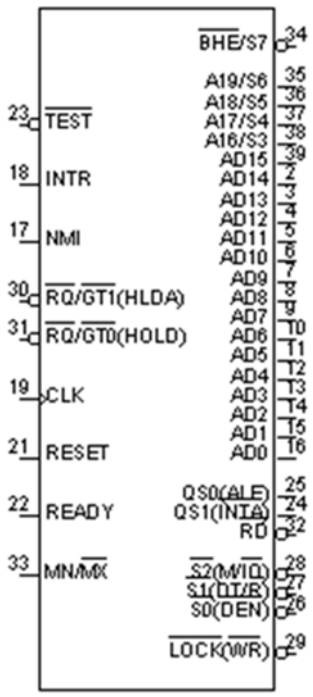


Рисунок 2. Графическое представление микропроцессора 8086

Теперь опишем основные сигналы микропроцессора.

Сначала мы рассмотрим сигналы адреса/данных. На рисунке 2. эти сигналы имеют мнемонику **AD** или **A**. Микропроцессор 8086 имеет совмещенную шину адреса и данных, обычно называемую мультиплексированной шиной адреса/данных. Временное мультиплексирование данных и адресов позволяет сократить требуемое количество выводов корпуса микропроцессора. Шина может быть разделена с помощью внешних регистров-защелок на отдельные шины адреса и данных. Выводы шины адреса (**A19 … A16**) указывают четыре старших адресных бита. Активный уровень этих сигналов – высокий. Сигналы шины адреса/данных (**AD0…AD15**) образуют мультиплексированную шину адреса данных микропроцессора.

Сигнал **BHE**(разрешение старшего байта шины) используется, чтобы разрешить передачу данных по старшей половине шины данных (**D8…D15**). Сигнал **BHE** должен иметь низкий уровень для разрешения передачи старшего байта данных. Сигнал **BHE** не нуждается в промежуточном запоминании в регистре-защелке. Передача слова или байта по шине данных определяется значениями сигналов **A0** и **BHE**.

Сигналы управления отличаются в минимальном и максимальном режимах. В минимальном режиме микропроцессор вырабатывает все необходимые системе сигналы управления.

В максимальном режиме необходим внешний системный контроллер. Микропроцессор вырабатывает сигналы состояния, которые поступают на этот контроллер, а он вырабатывает все необходимые управляющие сигналы.

Далее описаны сигналы для микропроцессора в минимальном режиме.

Сигнал **ALE** (разрешение фиксации адреса) обеспечивается процессором, чтобы зафиксировать адрес на мультиплексированной шине адреса/данных. Активный уровень сигнала **ALE** – высокий, правильное значение адреса гарантируется по срезу данного сигнала.

Строб записи (**WR**) указывает, что данные с шины данных должны быть написаны в память или устройство ввода/вывода. Этот сигнал имеет активный низкий уровень.

Строб чтения (**RD**) – сигнал с активным низким уровнем, который указывает, что процессор выполняет цикл чтения памяти или устройства ввода/вывода.

Передача/прием данных (**DT/R**) управляет направлением потока данных через внешний приемопередатчик шины данных. Когда сигнал имеет низкий уровень, данные передаются к процессору. При высоком уровне процессор передает данные на шину данных.

Разрешение данных (**DEN**) активизирует приемопередатчики шины данных. **DEN** активен всегда, когда происходит передача данных. Активный уровень – низкий. **DEN** не активен всякий раз, когда **DT/R** изменяет состояние.

Память/УВВ (**M/IO**) – определяет, куда предаются данные. Когда уровень сигнала низкий, данные передаются к устройству ввода/вывода. Когда высокий – данные передаются в память.

Запрос на прерывание (**INTR**) сигнал требования прерывания внешним устройством. Это входной сигнал, его активный уровень – высокий. В ответ микропроцессор обеспечивает сигнал подтверждения прерывания (**INTA**) с активным низким уровнем. Это сигнал маскируемого прерывания.

Немаскируемое прерывание (**NMI**) вызывает прерывание с вектором 2. Немаскируемое прерывание нельзя запретить программно.

HOLD (активен при высоком уровне сигнала) указывает, что другое устройство управления передачей данных по шине требует внешнюю шину. Процессор генерирует сигнал **HLDA** в ответ на запрос. Одновременно с формированием сигнала **HLDA**, процессор переводит шины в высокоимпедансное состояние. После снятия сигнала **HOLD** процессор переводит сигнал **HLDA** в неактивное состояние. Когда процессор должен выполнить другой цикл шины, он будет снова управлять локальной шиной и шинами управления.

В **максимальном режиме** сигнал **LOCK** указывает, что другие устройства управления передачей данных по шине системы не могут получить прав управления. Сигнал **LOCK** имеет активный низкий уровень.

Сигнал **LOCK** формируется по специальной команде префикса блокировки (**LOCK**) и активизируется в начале первого цикла передачи данных, связанного с командой следующей непосредственно после префикса блокировки и остается активным до завершения этой команды. Если сигнал **LOCK** активен, выборки команд из памяти в очередь команд не происходит.

Выводы **QS0** и **QS1** несут информацию о состоянии очереди команд процессора. Таблица 1 показывает возможные состояния очереди команд.

Сигналы состояния цикла шины (**S0**…**S2**) кодируют тип машинного цикла, выполняемого микропроцессором, как показано в таблице 2.

В максимальном режиме **HOLD-HLDA** протокол трансформируется в протокол управления доступом к шине Запрос/Разрешение (**RQ/GT**). Это позволяет другим сопроцессорам включаться в общую систему с микропроцессором 8086.

Таблица 1

Операции с очередью команд

QS1	QS0	Операции с очередью команд
0	0	Нет операций
0	1	Первый байт кода операции выбирается из очереди
1	1	Выборка следующего байта из очереди
1	0	Очередь команд пуста

Таблица 2

Типы машинных циклов

S2	S1	S0	Тип цикла
0	0	0	Подтверждение прерывания
0	0	1	Чтение УВВ
0	1	0	Запись УВВ
0	1	1	Остановка
1	0	0	Выборка команды
1	0	1	Чтение данных из памяти
1	1	0	Запись данных в память
1	1	1	Пассивный (нет операций)

CLK, **RESET**, **READY** - сервисные сигналы. Активный уровень сигнала **RESET** заставляет процессор немедленно закончить текущее действие, очистить внутреннюю логику, и перейти в неактивное состояние. Процессор начинает выбирать команды через несколько циклов тактовых сигналов после того, как **RESET** возвращается в неактивное состояние. Входной сигнал **CLK** должен быть подключен к генератору синхронизации. Сигнал **READY** сообщает процессору, что память или устройство ввода/вывода закончило передачу или прием данных.

Соединение **READY** с уровнем логической 1 будет всегда устанавливать состояние готовности для процессора.

Для пользователя действия, выполняемые микропроцессором, представляют собой последовательность циклов канала по обмену информацией с памятью или периферийными устройствами. Каждый цикл канала микропроцессора состоит, как минимум, из четырех машинных тактов T1 - T4. Ма-

шинный такт начинается по спаду импульса синхронизации **CLK** и продолжается один период синхронизации.

Любой цикл шины можно условно разделить **на две фазы**:

- фаза передачи адреса/статуса;

- фаза передачи данных.

Фаза передачи адреса начинается перед началом такта Т1 и продолжается в течение этого такта. Фаза передачи данных начинается в такте Т2 и заканчивается в такте Т4. В такте Т1 на канал адреса/данных всегда выдается адресная информация. В этом же такте вырабатывается сигнал **ALE**, который позволяет идентифицировать начало цикла канала и используется как стробирующий импульс для занесения адресной информации во внешний регистр адреса.

В такте Т2 производится переключение направления работы канала адреса/данных. Передача данных по каналу происходит в тактах Т3 и Т4. Длительность цикла канала может быть удлинена использованием управляющего сигнала **READY**. Этот сигнал позволяет разработчику синхронизировать скорость работы внешней памяти со скоростью работы микропроцессора введением в цикл канала между тактами Т3 и Т4 дополнительных тактов ожидания. В течение тактов ожидания данные на канале остаются неизменными. Между тактом Т4 текущего цикла и тактом Т1 следующего цикла канала процессор может вводить дополнительные (холостые) такты, предназначенные для выполнения внутренних действий. Моменты введения этих тактов и их число зависят от состояния очереди команд и выполняемой команды в УО.

На рисунке 3. представлена типовая временная диаграмма выполнения циклов чтения и записи.

Цикл чтения начинается с выработки сигнала **ALE**. Этот сигнал используется для занесения адресной информации во внешний регистр адреса. В такте **T2** канал **A/D** переключается в высокоумное состояние, вырабатывается сигнал **RD**, который используется для чтения адресуемого устройства. Для управления шинными формирователями, обеспечивающими развязку канала адреса/данных микропроцессора от системного канала данных, используются сигналы **DT/R** и **DEN**.

Цикл записи (как и цикл чтения) начинается с выдачи сигнала **ALE** и адреса на шину адреса/данных. В такте Т2 непосредственно за выдачей адреса на шину **A/D** выдаются данные для записи в адресуемое устройство. Эта информация остается истинной на канале данных до окончания такта Т4. Сигнал **WR** вырабатывается в начале такта Т2 и остается в этом состоянии до начала такта Т4.

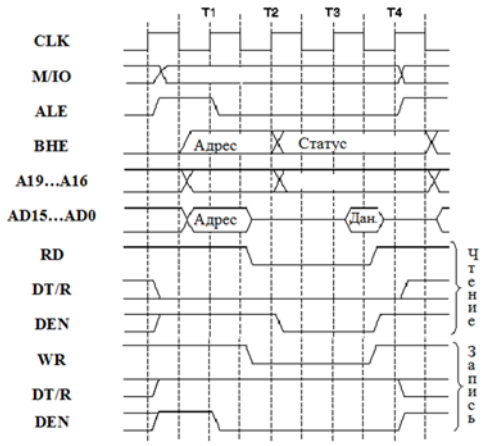


Рисунок 3. Временная диаграмма циклов чтения и записи

Программная модель микропроцессора

Программно-доступными функциональными частями микропроцессора являются **регистры общего назначения**, **сегментные регистры**, регистры **указатели** адреса (**индексные регистры**) и **регистр признаков**. Они показаны на рисунке 4.

Общие регистры используются для хранения операндов и результатов выполнения команд и делятся на две группы: регистры данных (регистры общего назначения), индексные регистры и указатели.

В группу регистров данных входят: регистр аккумулятора AX; регистр указателя базы данных BX, регистр счетчика циклов CX, регистр данных DX.

В группу индексных регистров и регистров указателей входят регистр указателя стека SP, регистр указателя базы стека BP, регистр индекса источника SI, регистр индекса приемника DI.

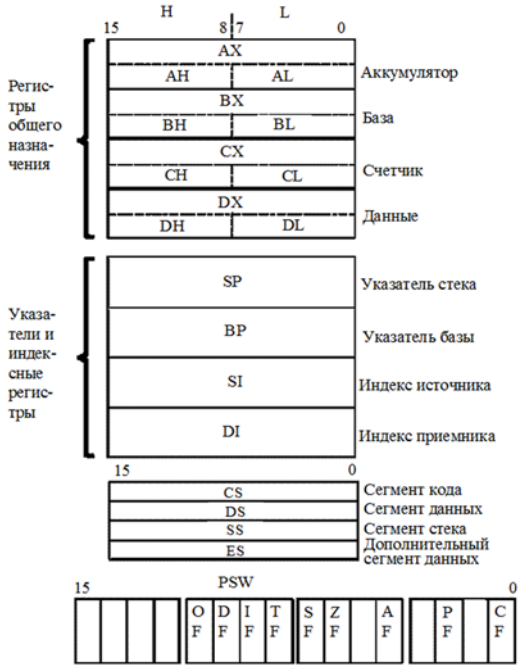


Рисунок 4. Программно доступные регистры микропроцессора

Старшие и младшие восемь разрядов группы регистров общего назначения могут быть адресованы раздельно. Они образуют набор 8-разрядных ре-

гистров общего назначения (AH, AL, BH, BL, CH, CL, DH, DL), причем регистрам AH, BH, CH, DH соответствуют старшие восемь разрядов, а регистрам AL, BL, CL, DL - младшие восемь разрядов группы регистров. Некоторые команды по умолчанию используют регистры для выполнения определенных функций. Они перечислены в таблице 3.

Таблица 3

Использование общих регистров

Регистр	Операция
AX	Умножение и деление слов, ввод/вывод слов
AL	Умножение и деление байтов, ввод/вывод байтов, десятичная арифметика, перекодирование
AH	Умножение и деление байтов
BX	Перекодирование
CX	Строковые операции, циклы
CL	Сдвиги переменных
DX	Умножение и деление слов, косвенная адресация портов ввода/вывода
SP	Стековые операции
SI	Операции со строками
DI	Операции со строками

Сегментные регистры используются для организации сегментной адресации памяти и предназначены для хранения базовых адресов текущих сегментов памяти. В 8086 имеется четыре 16-разрядных сегментных регистра: кода **CS**, данных **DS**, стека **SS**, дополнительного сегмента данных **ES**.

Разряды регистра признаков содержат признаки состояния микропроцессора, которые разделены на две группы: признаки результата и признаки управления.

В группу признаков результата входят:

- признак переполнения **OF**, указывающий на переполнение в случае выполнения операций над целыми числами;
- признак знака **SF**, указывающий на знак результата;
- признак нуля **ZF**, указывающий на равенство нулю результата;
- признак вспомогательного переноса **AF**, указывающий на перенос из третьего разряда или на заем в третий разряд результата при выполнении арифметических операций;
- признак четности **PF**, указывающий на четное число единиц в младшем байте результата;
- признак переноса **CF**, указывающий на перенос из старшего разряда или на заем в старший разряд результата.

В группу признаков управления входят:

• признак направления **DF**, указывающий направление обработки строк данных;

• признак разрешения прерывания **IF**, разрешающий или запрещающий маскируемые прерывания;

• признак пошагового режима **TF**, управляющий пошаговыми прерываниями.

• Микропроцессор обеспечивает формирование 20-разрядного адреса для адресации ячейки внешней памяти.

• Память организована как линейная последовательность ячеек памяти объемом в 1М байт с адресами от ОООООН до FFFFFH. Структурными единицами памяти являются: ячейка, слово, двойное слово и сегмент.

Ячейка памяти - минимальная адресуемая единица памяти, используемая для запоминания 8-разрядных данных (байта данных).

Слово памяти - две последовательные ячейки памяти, которые используются для запоминания 16-разрядных данных (слова данных), причем младшие восемь разрядов всегда хранятся в ячейке памяти с меньшим адресом, а старшие - с большим. При адресации 16-разрядных данных указывается адрес первой ячейки слова памяти. Слово памяти может располагаться в памяти как по четному, так и нечетному адресу. Чтение (запись) данных из слова памяти по четному адресу осуществляется за одно обращение к памяти, а по нечетному - за два обращения.

Двойное слово памяти - четыре последовательные ячейки памяти или два последовательных слова памяти, которые используются для запоминания 32-разрядных данных. При адресации 32-разрядных данных указывается адрес первой ячейки двойного слова памяти, Двойное слово памяти также может иметь четный или нечетный адрес.

Для достижения максимальной производительности слова и двойные слова данных должны размещаться в памяти по четным адресам.

Программы, написанные для микропроцессора 8086, рассматривают 1М байт памяти как группу сегментов, определяемых конкретным применением.

Сегмент памяти - участок памяти, емкость которого может изменяться от 01.01.016 байт; начинается с адреса, кратного 10Н. Каждому сегменту соответствует непрерывная и отдельно адресуемая область памяти. Примеры сегментации показаны на рисунке 5.

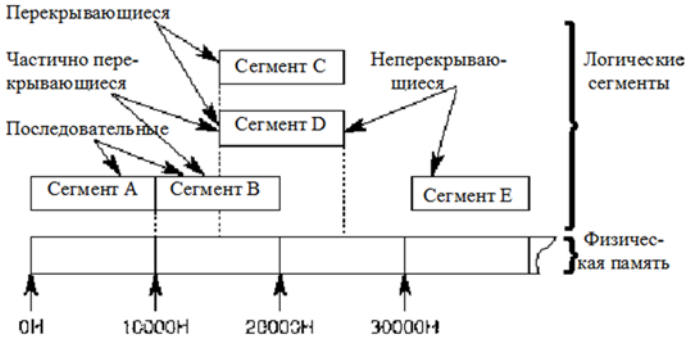


Рисунок 5. Сегментация памяти микропроцессора

Сегменты могут следовать друг за другом непрерывно, с интервалом или могут перекрываться. Максимальное количество следующих непрерывно друг за другом сегментов емкостью 65536 байт равно 16.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 14

Тема: «Изучение порядка взаимодействия УУ, АЛУ и ОЗУ при автоматическом выполнении команд»

Цель работы: изучение порядка взаимодействия УУ, АЛУ и ОЗУ при автоматическом выполнении команд.

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Устройство управления (УУ) управляет всем ходом вычислительного и логического процесса в компьютере, т.е. выполняет функции «регулировщика движения» информации. УУ читает команду, расшифровывает ее и подключает необходимые цепи для ее выполнения. Считывание следующей команды происходит автоматически.

Фактически УУ выполняет следующий цикл действий:

1. формирование адреса очередной команды;
2. чтение команды из памяти и ее расшифровка;
3. выполнение команды.

В современных компьютерах функции УУ и АЛУ выполняет одно устройство, называемое центральным процессором.

Назначение процессора – это автоматическое выполнение программы. Другими словами, он является основным компонентом любого компьютера.

Устройство процессора

Ключевыми компонентами процессора являются **арифметико-логическое устройство** (АЛУ), **регистры** и **устройство управления**. АЛУ выполнят основные математические и логические операции. Все вычисления производятся в двоичной системе счисления. От устройства управления зависит согласованность работы частей самого процессора и его связь с другими (внешними для него) устройствами. В регистрах временно хранятся текущая команда, исходные, промежуточные и конечные данные (результат вычислений АЛУ). Разрядность всех регистров одинакова.

Кэш данных и команд хранит часто используемые данные и команды. Обращение в кэш происходит намного быстрее, чем в оперативную память, поэтому, чем он больше, тем лучше.

Работает процессор под управлением программы, находящейся в оперативной памяти.

Для того, чтобы персональный компьютер мог работать, необходимо, чтобы в его оперативной памяти находилась программа и данные, и между ними происходил обмен. При работе программы часто бывает, необходим ввод информации от пользователя или вывод ее на экран. Такой обмен называется вводом-выводом. Для его осуществления имеются два промежуточных звена:

1. Для каждого внешнего устройства ПК имеется электронная схема, которая им управляет. Его называют контроллером или адаптером.

2. Все контроллеры или адаптеры взаимодействуют с микропроцессором и оперативной памятью через системную магистраль передачи данных, которую называют шиной. Системная шина является каналом соединения микропроцессора, оперативной памяти и интегральных устройств. Физически шина находится на материнской плате.

Для обмена данными с памятью и устройствами ввода-вывода служат разные компоненты шины: взаимодействие микропроцессора с периферийными устройствами идет через шину данных, а адресация памяти происходит при помощи шины адреса. Иногда физически они находятся в одном канале связи.

Сокращенно оперативную память компьютера называют ОЗУ (оперативное запоминающее устройство) или RAM (Random Access Memory — память с произвольным доступом).

Назначение ОЗУ

Хранение данных и команд для дальнейшей их передачи процессору для обработки. Информация может поступать из оперативной памяти не сразу на обработку процессору, а в более быструю, чем ОЗУ, кэш-память процессора.

Хранение результатов вычислений, произведенных процессором.

Считывание (или запись) содержимого ячеек.

Особенности работы ОЗУ.

Оперативная память может сохранять данные лишь при включенном компьютере. Поэтому при его выключении обрабатываемые данные следует сохранять на жестком диске или другом носителе информации. При запуске программ информация поступает в ОЗУ, например, с жесткого диска компьютера. Пока идет работа с программой она присутствуют в оперативной памяти (обычно). Как только работа с ней закончена, данные перезаписываются на жесткий диск. Другими словами, потоки информации в оперативной памяти очень динамичны.

ОЗУ представляет собой запоминающее устройство с произвольным доступом. Это означает, что прочитать/записать данные можно из любой ячейки ОЗУ в любой момент времени. Для сравнения, например, магнитная лента, является запоминающим устройством с последовательным доступом.

Логическое устройство оперативной памяти.

Оперативная память состоит их ячеек, каждая из которых имеет свой собственный адрес. Все ячейки содержат одинаковое число бит. Соседние ячейки имеют последовательные адреса. Адреса памяти также как и данные выражаются в двоичных числах.

Обычно одна ячейка содержит 1 байт информации (8 бит, то же самое, что 8 разрядов) и является минимальной единицей информации, к которой возможно обращение. Однако многие команды работают с так называемыми словами. Слово представляет собой область памяти, состоящую из 4 или 8 байт (возможны другие варианты).

Типы оперативной памяти

Принято выделять два вида оперативной памяти: статическую (SRAM) и динамическую (DRAM). SRAM используется в качестве кэш-памяти процессора, а DRAM - непосредственно в роли оперативной памяти компьютера.

SRAM состоит из триггеров. Триггеры могут находиться лишь в двух состояниях: «включен» или «выключен» (хранение бита). Триггер не хранит

заряд, поэтому переключение между состояниями происходит очень быстро. Однако триггеры требуют более сложную технологию производства, и занимает много места (на микроуровне), в результате SRAM получается достаточно большим устройством.

В **DRAM** нет триггеров, а бит сохраняется за счет использования одного транзистора и одного конденсатора. Получается дешевле и компактней. Однако конденсаторы хранят заряд, а процесс зарядки-разрядки более длительный, чем переключение триггера. Как следствие, DRAM работает медленнее. Второй минус – это самопроизвольная разрядка конденсаторов. Для поддержания заряда его регенерируют через определенные промежутки времени, на что тратится дополнительное время.

Магнитные диски компьютера служат для длительного хранения информации (она не стирается при выключении ЭВМ). При этом в процессе работы данные могут удаляться, а другие записываться.

Жесткий диск позволяет хранить большие объемы информации. Емкость жестких дисков современных компьютеров может составлять терабайты.

Первый жесткий диск был создан фирмой IBM в 1973 году. Он позволял хранить до 16 Мбайт информации. Поскольку этот диск имел 30 цилиндров, разбитых на 30 секторов, то он обозначался как 30/30. По аналогии с автоматическими винтовками, имеющими калибр 30/30, этот диск получил прозвище «винчестер».

Жесткий диск представляет собой герметичную железную коробку, внутри которой находится один или несколько магнитных дисков вместе с блоком головок чтения/записи и электродвигателем. При включении компьютера электродвигатель раскручивает магнитный диск до высокой скорости (несколько тысяч оборотов в минуту) и диск продолжает вращаться все время, пока компьютер включен. Над диском «парят» специальные магнитные головки, которые записывают и считывают информацию так же, как и на гибких дисках.

Головки парят над диском вследствие его высокой скорости вращения. Если бы головки касались диска, то из-за силы трения диск быстро вышел бы из строя.

Любой магнитный диск имеет логическую структуру, которая включает в себя следующие элементы:

- загрузочный сектор;

- таблицы размещения файлов;

- область данных.

Загрузочный сектор (Boot Record) занимает сектор с номером 0. В нем содержится небольшая программа IPL2 (Initial Program Loading 2), с помо-

щью которой компьютер определяет возможность загрузить операционную систему с данного диска.

Особенностью винчестера является наличие помимо загрузочного сектора еще одной области - главного загрузочного сектора (Master Boot Record). Дело в том, что единый жесткий диск может быть разбит на несколько логических дисков. Для главного загрузочного сектора на жестком диске всегда выделяется физический сектор 1. Этот сектор содержит программу IPL1 (Initial Program Loading 1), которая при своем выполнении определяет загрузочный диск.

Таблица размещения файлов используется для хранения сведений о размещении файлов на диске. Для магнитных дисков обычно используются две копии таблиц, которые следует одна за другой, и содержимое их полностью совпадает. Это делается на тот случай, если на диске произошли какие либо сбои, то диск всегда можно «отремонтировать», используя вторую копию таблицы. Если будут испорчены обе копии, то вся информация на диске будет потеряна.

Область данных (Data Area) занимает основную часть дискового пространства и служит непосредственно для хранения данных. Обычно жесткий диск делят на несколько разделов. Это бывает удобно для хранения файлов и является необходимым условием при установке нескольких операционных систем на один физический жесткий диск компьютера.

Итак, **раздел диска** – это часть жесткого диска, используемая под определенные задачи: файловую систему того или иного типа, область подкачки и т.п. Изменение содержимого и файловой системы одного раздела никак не сказывается на других.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 15

Тема: «Использование мультипрограммирования при различных режимах работы вычислительных систем»

Цель работы: применение мультипрограммирования при различных режимах работы вычислительных систем».

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;

• лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Основные характеристики мультипрограммного режима работы ЭВМ.

Мультипрограммным режимом работы (многозадачностью) называется такой способ организации работы системы, при котором в ее памяти одновременно содержатся программы и данные для выполнения нескольких процессов обработки информации (задач).

Мультипрограммирование позволяет повысить производительность работы ЭВМ за счет более эффективного использования ее ресурсов.

Базовыми понятиями мультипрограммного режима функционирования ЭВМ являются процесс и ресурс.

В строгом понимании **процесс** – это система действий, реализующая определенную функцию в вычислительной системе и оформленная так, что управляющая программа вычислительной системы может перераспределять ресурсы этой системы в целях обеспечения мультипрограммирования. То есть процесс – это некоторая деятельность, связанная с исполнением программы на процессоре.

Процесс может находиться в следующих состояниях:

а) порождение – подготавливаются условия для первого исполнения на процессоре;

б) активное состояние – исполнение программы на центральном процессоре;

в) готовность (Ready – программа не исполняется, но для исполнения предоставлены все необходимые в текущий момент ресурсы, кроме центрального процессора;

г) исполнение программы на каком-либо другом устройстве компьютера;

д) ожидание (Wait*)* – программа не исполняется по причине занятости какого-либо ресурса;

е) окончание – нормальное или аварийное завершение исполнения программы, после которого процессор и другие ресурсы ей не предоставляются.

Время между порождением и окончанием процесса называется **интервалом существования процесса.**

Понятие ресурса строго не определено. Будем считать, что всякий потребляемый объект (независимо от формы его существования), обладающий некоторой практической ценностью для потребителя, является **ресурсом**.

Ресурсы различаются по запасу выделяемых единиц ресурса и бывают в этом смысле **исчерпаемым и неисчерпаемыми.**

К исчерпаемым ресурсам относится центральный процессор. В качестве неисчерпаемого ресурса можно представить, например, память, выделяемую программе, если рассматривать ее как совокупность всех имеющихся в компьютере запоминающих устройств. В то же время, запоминающее устройство, состоящее только из оперативной памяти, представляет собой исчерпаемый ресурс.

Исчерпаемость ресурса, как правило, приводит к конфликтам среди потребителей этого ресурса. Для регулирования конфликтов ресурсы должны распределяться между потребителями по каким-то правилам, в наибольшей степени их удовлетворяющим.

Основные черты мультипрограммного режима:

- в оперативной памяти находятся несколько пользовательских программ в состояниях активности, ожидания или готовности;
- время работы процессора разделяется между программами, находящимися в памяти в состоянии готовности;
- параллельно с работой процессора происходит подготовка и обмен с несколькими устройствами ввода-вывода.

Мультипрограммирование предназначено для повышения пропускной способности вычислительной системы путем более равномерной и плотной загрузки всего ее оборудования, в первую очередь процессора. При этом скорость работы самого процессора и номинальная производительность ЭВМ не зависят от мультипрограммирования.

Мультипрограммный режим имеет в ЭВМ аппаратную и программную поддержку:

1. аппаратная:

- контроллеры устройств ввода-вывода, которые могут работать параллельно с процессором;
- система прерывания;
- система защиты программ и данных и т. п.;

2. программная:

- мультизадачная операционная система (ОС);
- системные программы, управляющие работой устройств ввода-вывода и специализированных вычислительных средств компьютера.

Управляющая программа (ОС), реализуя мультипрограммный режим, должна распределять (в том числе динамически) ресурсы системы (время процессора, оперативную и внешнюю память, устройства ввода-вывода и т. д.) между параллельно выполняемыми программами, чтобы обеспечить увеличение пропускной способности компьютера с учетом ограничений на ресурсы и требований по срочности выполнения отдельных программ.

Производительность мультипрограммной ЭВМ оценивается количеством задач, выполненных в единицу времени *(***пропускная способность) и временем выполнения каждой программы Ti.**

При анализе работы ЭВМ важно определить степень использования ее ресурсов.

Для этого широко применяются следующие показатели:

K_q – коэффициент загрузки устройства; $k_q = \frac{T_q}{T}$, где T_q – время занятости устройства q за общее время T работы ЭВМ; $L_q = \frac{\sum_{i=1}^{n} L_{q_i} \times \Delta t_i}{T}$ – средняя длина очереди запросов к устройству q, где L_{q_i} – длина очереди к устройству q на интервале времени Δt_i и $\sum_{i=1}^{n} \Delta t_i = T$.

Пусть работа некоторого устройства q характеризуется диаграммой.

Tt - когда рассмотренные выше показатели работы этого устройства будут следующими:

$k_q = \frac{7}{10}$; $L_q = \frac{0 \bullet 2 + 1 \bullet 1 + 0 \bullet 4 + 1 \bullet 1 + 1 \bullet 2 + 1 \bullet 1}{10} = \frac{5}{10}$.

Помимо средней длины очереди важна также и динамика изменения текущей длины очереди.

По значениям k_q, L_q и динамике изменения L_q можно определить наиболее дефицитный ресурс в системе, ее «узкое место».

Устранить «узкие места» можно или увеличением производительности соответствующего ресурса, или выбором такой смеси задач, которая обеспечивала бы более равномерное использование всех ресурсов (например, одни задачи более активно используют процессор (счетные задачи), другие – жесткий диск (работа с базами данных), третьи – устройства ввода-вывода).

Работа мультипрограммной ЭВМ в большой степени зависит от **коэффициента мультипрограммирования** (Км) – количества программ, которое может одновременно обрабатываться в мультипрограммном режиме.

Пример выполнения программ в мультипрограммном режиме при Км=2. Предполагается, что выполнение каждой программы включает следующую последовательность действий: счет1 – ввод – счет2 –вывод. Счет выполняется на процессоре (CPU), для ввода и вывода данных используются отдельные внешние устройства (IN и OUT).

Время на выполнение соответствующих блоков программ задано в таблице 1.

Таблица 1

Программа	*CPU*1	*IN*	*CPU*2	*OUT*
1	2	1	4	2
2	2	2	1	3
3	4	3	3	1
4	2	2	2	2

На графике помечены номера программ, которые в данный момент занимают тот или иной ресурс.

Если построить аналогичные графики для ЭВМ, работающей с различными коэффициентами мультипрограммирования, то получим следующие сравнительные характеристики работы ЭВМ для рассматриваемого пакета программ (табл. 2).

Таблица 2

Характеристика	Км = 1	Км = 2	Км = 3
Время выполнения программы Т1	9	10	10
– " – Т2	8	13	13
– " – Т3	11	11	19
– " – Т4	8	11	12
Время выполнения всех программ (Т)	36	24	22
Пропускная способность (П)	0,11	0,17	0,18
k_{CPU}	0,56	0,83	0,91
k_{IN}	0,22	0,33	0,36
k_{OUT}	0,22	0,33	0,36

Под временем выполнения программы понимается время, прошедшее от начала выполнения программы или ее постановки в очередь к процессору, до ее завершения, а время выполнения всех программ определяется моментом завершения выполнения последней программы пакета.

При увеличении коэффициента мультипрограммирования изменение значений показателей эффективности зависит от того, в каком состоянии находится система: перегрузки или недогрузки. Если какие-либо ресурсы ЭВМ используются достаточно интенсивно, то добавление новой программы, активно использующей эти ресурсы, будет малоэффективным. Очевидно, что зависимость пропускной способности (П), времени выполнения каждой про-

граммы (Ti) и времени выполнения всего пакета программ (T) от коэффициента мультипрограммирования будет иметь вид.

На изменение эффективности работы мультипрограммной ЭВМ может повлиять назначение различных приоритетов выполняемым программам. Перераспределение приоритетов может привести как к увеличению, так и к снижению пропускной способности ЭВМ. Конкретный результат зависит от характеристик выполняемых программ. В частности, если в составе мультипрограммной смеси имеется единственная программа, надолго занимающая процессор, то увеличение ее приоритета понизит, а уменьшение – повысит пропускную способность ЭВМ. Это объясняется тем, что выполнение программ, обладающих меньшим приоритетом, чем рассматриваемая, фактически блокируется из-за недоступности процессора. Аналогичная ситуация может сложиться и в отношении других совместно используемых ресурсов. Особое значение при этом имеют те из них, которые являются наиболее дефицитными, то есть имеют наибольший коэффициент загрузки и наибольшую среднюю длину очереди.

Как правило, наиболее высокий приоритет назначается тем программам, которые в состоянии быстро освободить наиболее дефицитный ресурс. Такого рода проблемы решаются в рамках теории расписаний. При этом поиск решения зачастую сводится к полному перебору вариантов. Ввиду сложности полной теоретической оценки всех возможных вариантов, на практике широко используются различные эвристические алгоритмы, дающие не оптимальные, а рациональные решения.

В мультипрограммной ЭВМ ресурсы могут распределяться как на **статической, так и на динамической основе.** В первом случае ресурсы распределяются до момента порождения процесса и являются для него постоянными. Освобождение ресурсов, занятых каким-либо процессом, происходит только в момент окончания этого процесса.

При динамическом распределении ресурсы выделяются процессу по мере его развития.

Распределение на статической основе способствует наиболее быстрому развитию процессов в системе с момента их порождения. Распределение же ресурсов на динамической основе позволяет обеспечить эффективное использование ресурсов с точки зрения минимизации их простоев.

Схема статического распределения используется в том случае, когда необходимо гарантировать выполнение процесса с момента его порождения. В качестве недостатка этого подхода следует отметить возможность длительных задержек заявок на порождение процесса с момента поступления таких заявок в систему, так как необходимо ожидать освобождения всех требуемых заявке ресурсов и только при наличии их полного состава порождать процесс. Часто распределение ресурсов с использованием исключительно статического принципа приводит фактически к однопрограммному режиму работы.

При динамическом распределении стремление уменьшить простои ресурсов приводит к увеличению сложности системы распределения ресурсов и, как следствие, к увеличению системных затрат на управление процессами. Поэтому необходим компромисс между сложностью алгоритмов планирования распределения ресурсов и эффективностью выполнения пакета задач.

Ресурсы разделяются на физические и виртуальные:

- под **физическим** понимают ресурс, который реально существует и при распределении его между пользователями обладает всеми присущими ему физическими характеристиками;

- виртуальный ресурс – это некоторая модель, которая строится на базе физического ресурса, имеет расширенные функциональные возможности по отношению к физическому ресурсу, на базе которого он создан, или обладает некоторыми дополнительными свойствами, которых физический ресурс не имеет.

Например, расширенные функциональные возможности имеет виртуальная память, представляющаяся как запоминающее устройство, имеющее больший объем, чем физическая. Дополнительные свойства имеет виртуальный процессор, одновременно обрабатывающий несколько задач.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации : учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. — 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветков, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 16

Тема: «Распределение памяти и методы сокращения времени адресного преобразования»

Цель работы: изучение распределения памяти и методов сокращения времени адресного преобразования.

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Методы распределения памяти

Необходимым условием для того, чтобы программа могла выполняться, является ее нахождение в оперативной памяти. Только в этом случае процессор может извлекать команды из памяти и интерпретировать их, выполняя заданные действия.

Все алгоритмы распределения памяти разделены на два класса: алгоритмы, в которых используется перемещение сегментов процессов между оперативной памятью и диском, и алгоритмы, в которых внешняя память не привлекается.

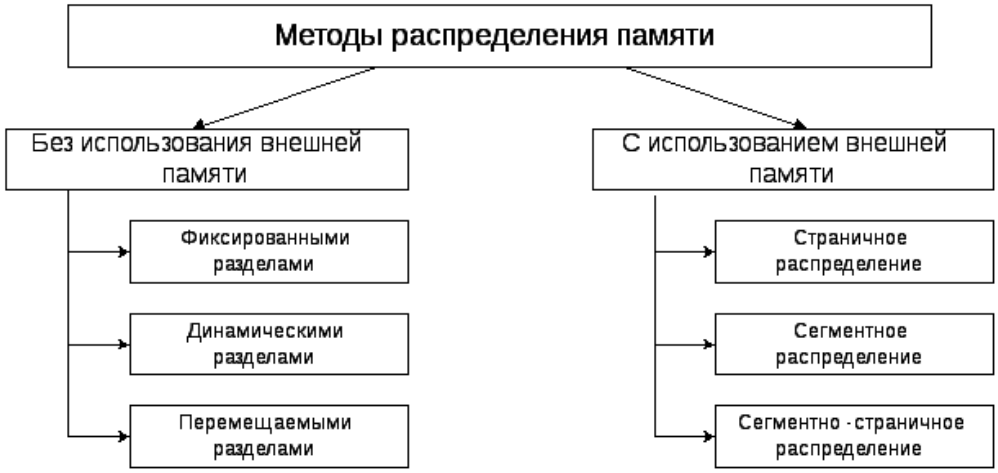


Схема 1. Классификация методов распределения памяти

Распределение памяти перемещаемыми разделами

Особенность – используется процедура дефрагментации, т.е. перемещение всех занятых участков в сторону старших или младших адресов, так, чтобы вся свободная память образовала единую свободную область.

Функции операционной системы, предназначенные для реализации данного метода управления памятью:

1. Ведение таблиц свободных и занятых областей, в которых указываются начальные адреса и размеры участков памяти.

2. При создании нового процесса – анализ требований к памяти, просмотр таблицы свободных областей и выбор раздела, размер которого достаточен для размещения кодов и данных нового процесса.

3. Загрузка программы в выделенный ей раздел и корректировка таблиц свободных и занятых областей.

4. После завершения процесса корректировка таблиц свободных и занятых областей.

В дополнение к перечисленным функциям, в данном случае она должна еще время от времени копировать содержимое разделов из одного места памяти в другое, корректируя таблицы свободных и занятых областей. Эта

процедура называется сжатием. Сжатие может выполняться либо при каждом завершении процесса, либо только тогда, когда для вновь создаваемого процесса нет свободного раздела достаточного размера. В первом случае требуется меньше вычислительной работы при корректировке таблиц свободных и занятых областей, а во втором – реже выполняется процедура сжатия.

Достоинство: эффективное использование памяти.

Недостаток: снижение производительности системы в целом, поскольку процедура сжатия может требовать значительного времени. Методы распределения памяти. Страничное распределение.

Методы распределения памяти

Необходимым условием для того, чтобы программа могла выполняться, является ее нахождение в оперативной памяти. Только в этом случае процессор может извлекать команды из памяти и интерпретировать их, выполняя заданные действия.

Все алгоритмы распределения памяти разделены на два класса: алгоритмы, в которых используется перемещение сегментов процессов между оперативной памятью и диском, и алгоритмы, в которых внешняя память не привлекается.

Страничное распределение

Виртуальное адресное пространство каждого процесса делится на части одинакового, фиксированного для данной системы размера, называемые **виртуальными страницами** (virtual pages). В общем случае размер виртуального адресного пространства процесса не кратен размеру страницы, поэтому последняя страница каждого процесса дополняется фиктивной областью.

Вся оперативная память машины также делится на части такого же размера, называемые **физическими страницами** (или блоками, или кадрами).

Размер страницы выбирается равным степени двойки: 512, 1024, 4096 байт и т.д. Это позволяет упростить механизм преобразования адресов.

При создании процесса ОС загружает в оперативную память несколько его виртуальных страниц (начальные страницы кодового сегмента и сегмента данных). Копия всего виртуального адресного пространства процесса находится на диске. Смежные виртуальные страницы не обязательно располагаются в смежных физических страницах.

Для каждого процесса операционная система создает **таблицу страниц** – информационную структуру, содержащую записи обо всех виртуальных страницах процесса. Запись таблицы, называемая **дескриптором страницы**, включает следующую информацию:

- номер физической страницы, в которую загружена данная виртуальная страница;
- признак присутствия, устанавливаемый в единицу, если виртуальная страница находится в оперативной памяти;
- признак модификации страницы, который устанавливается в единицу всякий раз, когда производится запись по адресу, относящемуся к данной странице;
- признак обращения к странице, называемый также битом доступа, который устанавливается в единицу при каждом обращении по адресу, относящемуся данной странице.

При каждом обращении к памяти выполняется поиск номера виртуальной страницы, содержащей требуемый адрес, затем по этому номеру определяется нужный элемент таблицы страниц, и из него извлекается описывающая страницу информация. Далее анализируется признак присутствия, и, если данная виртуальная страница находится в оперативной памяти, то выполняется преобразование виртуального адреса в физический, то есть виртуальный адрес заменяется указанным в записи таблицы физическим адресом. Если же нужная виртуальная страница в данный момент выгружена на диск, то происходит так называемое страничное прерывание. Выполняющийся процесс переводится в состояние ожидания, и активизируется другой процесс из очереди процессов, находящихся в состоянии готовности. Параллельно программа обработки страничного прерывания находит на диске требуемую виртуальную страницу (для этого операционная система должна помнить положение вытесненной страницы в страничном файле диска) и пытается загрузить ее в оперативную память. Если в памяти имеется свободная физическая страница, то загрузка выполняется немедленно, если же свободных страниц нет, то на основании принятой в данной системе стратегии замещения страниц решается вопрос о том, какую страницу следует выгрузить из оперативной памяти.

После того как выбрана страница, которая должна покинуть оперативную память, обнуляется ее бит присутствия и анализируется ее признак модификации.

Если выталкиваемая страница за время последнего пребывания в оперативной памяти была модифицирована, то ее новая версия должна быть переписана на диск. Если нет, то принимается во внимание, что на диске уже имеется предыдущая копия этой виртуальной страницы, и никакой записи на диск не производится. Физическая страница объявляется свободной.

Достоинство: возможность работы с виртуальной памятью большого объема позволяет успешно справляться с фрагментацией физической памяти.

Недостаток: относительная сложность реализации.

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации: учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.

Практическая работа № 17

Тема: «Изучение алгоритмов буферизации и кэширования данных»

Цель работы: изучение и применение алгоритмов буферизации и кэширования данных.

Материально-техническое оснащение:

- компьютер с мультимедиа проектором;
- лицензионное программное обеспечение, ознакомительные версии программного обеспечения.

Теоретические сведения

Под буфером обычно понимается некоторая область памяти для запоминания информации при обмене данных между двумя устройствами, двумя процессами или процессом и устройством. Обмен информацией между двумя процессами относится к области кооперации процессов, и мы подробно рассмотрели его организацию в соответствующей лекции. Здесь нас будет интересовать использование буферов в том случае, когда одним из участников обмена является внешнее устройство. Существует три причины, приводящие к использованию буферов в **базовой подсистеме ввода-вывода**.

Первая причина буферизации – это разные скорости приема и передачи информации, которыми обладают участники обмена. Рассмотрим, например, случай передачи потока данных от клавиатуры к модему. Скорость, с которой поставляет информацию клавиатура, определяется скоростью набора текста человеком и обычно существенно меньше скорости передачи данных модемом. Для того чтобы не занимать модем на все время набора текста, делая его недоступным для других процессов и устройств, целесообразно накапливать введенную информацию в буфере или нескольких буферах достаточного размера и отсылать ее через модем после заполнения буферов.

Вторая причина буферизации – это разные объемы данных, которые могут быть приняты или получены участниками обмена единовременно. Возьмем другой пример. Пусть информация поставляется модемом и записывается на жесткий диск. Помимо обладания разными скоростями совершения операций, модем и жесткий диск представляют собой устройства разного типа. Модем является **символьным устройством** и выдает данные байт за байтом, в то время как диск является **блочным устройством** и для проведения операции записи для него требуется накопить необходимый блок данных в буфере. Здесь также можно применять более одного буфера. После заполнения первого буфера модем начинает заполнять второй, одновременно с записью первого на жесткий диск. Поскольку скорость работы жесткого диска в тысячи раз больше, чем скорость работы модема, к моменту заполнения второго буфера операция записи первого будет завершена, и модем снова сможет заполнять первый буфер одновременно с записью второго на диск.

Третья причина буферизации связана с необходимостью копирования информации из приложений, осуществляющих ввод-вывод, в буфер ядра операционной системы и обратно.

Допустим, что некоторый пользовательский процесс пожелал вывести информацию из своего адресного пространства на внешнее устройство. Для этого он должен выполнить системный вызов с обобщенным названием write, передав в качестве параметров адрес области памяти, где расположены данные, и их объем. Если внешнее устройство временно занято, то возможна ситуация, когда к моменту его освобождения содержимое нужной области окажется испорченным (например, при использовании асинхронной формы системного вызова). Чтобы избежать возникновения подобных ситуаций, проще всего в начале работы системного вызова скопировать необходимые данные в буфер ядра операционной системы, постоянно находящийся в оперативной памяти, и выводить их на устройство из этого буфера.

Под словом **кэш** (cache – «наличные»), этимологию которого мы не будем здесь рассматривать, обычно понимают область быстрой памяти, содержащую копию данных, расположенных где-либо в более медленной памяти, предназначенную для ускорения работы вычислительной системы. Мы с вами сталкивались с этим понятием при рассмотрении иерархии памяти.

В **базовой подсистеме ввода-вывода** не следует смешивать два понятия, **буферизацию и кэширование**, хотя зачастую для выполнения этих функций отводится одна и та же область памяти. Буфер часто содержит единственный набор данных, существующий в системе, в то время как кэш по определению содержит копию данных, существующих где-нибудь еще. Например, буфер, используемый базовой подсистемой для копирования данных из пользовательского пространства процесса при выводе на диск, может в свою очередь применяться как кэш для этих данных, если операции модификации и повторного чтения данного блока выполняются достаточно часто.

Функции **буферизации и кэширования** не обязательно должны быть локализованы в **базовой подсистеме ввода-вывода**. Они могут быть частично реализованы в драйверах и даже в **контроллерах устройств**, скрытно по отношению к **базовой подсистеме.**

Литература и информационные источники:

1. Гребнюк, Е.И. Технические средства информатизации. – Москва : Академия, 2014.

2. Елесина, С.И. ЭВМ и периферийные устройства. Устройства ввода-вывода информации : учебник / С.И. Елесина, Е.Р. Муратов, М.Б. Никифоров. – Москва : КУРС, 2018. – 208 с. - ISBN 978-5-906923-55-4. –

Режим доступа: http://znanium.com/catalog/product/1017280.

3. Цветкова, М.С. Информатика и ИКТ. – Москва : Академия, 2014.