Документ подписан квалифицированной электронной подписью
Сертификат: 03489EA30058B4708C45B74E4D79B70F27
Владелец: "АНО ВО "РОССИЙСКИЙ НОВЫЙ УНИВЕРСИТЕТ"
Действителен: с 28.05.2026 по 28.08.2027

**АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ НОВЫЙ УНИВЕРСИТЕТ»**
**(АНО ВО «РосНОУ»)**

# ОПЦ.07 ТЕХНИЧЕСКИЕ СРЕДСТВА ИНФОРМАТИЗАЦИИ

специальность 10.02.05 Обеспечение информационной безопасности автоматизированных систем

**Методические указания к выполнению практических занятий**

Москва 2026

Методические указания к выполнению практических занятий для обучающихся 2 курса всех форм обучения образовательных организаций среднего профессионального образования по ОП.07 Технические средства информатизации специальности 10.02.05 Обеспечение информационной безопасности автоматизированных систем, разработаны в соответствии с:

Федеральным государственным образовательным стандартом

– по специальности среднего профессионального образования 10.02.05 Обеспечение информационной безопасности автомати-зированных систем утвержденным МИНОБРНАУКИ от 09.12.2016 № 1553.

# ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Методические указания к практическим занятиям по ОП.07 Технические средства информатизации для обучающихся всех форм обучения разработаны в соответствии с требованиями Федерального государственного стандарта (ФГОС) по специальности по специальности среднего профессионального образования (далее – СПО) 10.02.05 Обеспечение информационной безопасности автоматизированных систем и рабочей программы учебной дисциплины ОП.07 Технические средства информатизации.

Цель методических указаний: оказание помощи обучающимся в выполнении практических занятий по ОП.07 Технические средства информатизации.

Настоящие методические указания содержат практические занятия, которые позволят обучающимся закрепить теорию и направлены на формирование следующих профессиональных (ПК) и общих (ОК) компетенций:

| | |
|---|---|
| ПК 1.1 | Производить установку и настройку компонентов автоматизированных (информационных) систем в защищенном исполнении в соответствии с требованиями эксплуатационной документации |
| ПК 1.2 | Администрировать программные и программно-аппаратные компоненты автоматизированной (информационной) системы в защищенном исполнении |
| ПК 1.3 | Обеспечивать бесперебойную работу автоматизированных (информационных) систем в защищенном исполнении в соответствии с требованиями эксплуатационной документации |
| ПК 2.1 | Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации |
| ПК 2.2 | Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации |
| ПК 2.3 | Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации |
| ПК 3.1 | Осуществлять установку, монтаж, настройку и техническое обслуживание технических средств защиты информации в соответствии с требованиями эксплуатационной документации |
| ПК 3.2 | Осуществлять эксплуатацию технических средств защиты информации в соответствии с требованиями эксплуатационной документации |
| ПК 3.3 | Осуществлять измерение параметров побочных электромагнитных излучений и наводок, создаваемых техническими средствами обработки информации ограниченного доступа |
| ПК 3.4 | Осуществлять измерение параметров фоновых шумов, а также физических полей, создаваемых техническими средствами защиты информации |
| ОК 1. | Выбирать способы решения задач профессиональной деятельности, применительно к различным кон текстам |
| ОК 2. | Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности |
| ОК 3. | Планировать и реализовывать собственное профессиональное и личностное развитие |
| ОК 4. | Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами |

| | |
|---|---|
| ОК 5. | Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста |
| ОК 6. | Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей, в том числе с учетом гармонизации межнациональных и межрелигиозных отношений, применять стандарты антикоррупционного поведения |
| ОК 7. | Содействовать сохранению окружающей среды, ресурсосбережению, действовать в эффективно чрезвычайных ситуациях |
| ОК 8. | Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержание необходимого уровня физической подготовленности |
| ОК 9. | Использовать информационные технологии в профессиональной деятельности |
| ОК 10. | Пользоваться профессиональной документацией на государственном и иностранном языках |

В результате выполнения практических занятий по ОП.07 Технические средства информатизации обучающиеся должны уметь:

– пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации;

– правильно эксплуатировать и устранять типичные выявленные дефекты технических средств информатизации.

# ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ЧАСТЬ 1, 2

| Номер темы | Номер и наименование работы (занятия) | Ко-во аудиторных часов |
|---|---|---|
| 1 | 2 | 3 |
| 2.2 | Практическая работа № 1. Программирование ввода-вывода | 2 |
| 2.3 | Практическая работа № 2. Установка конфигурации системы при помощи улиты CMOSSetup. | 2 |
| 2.3 | Практическая работа № 3. Тестирование компонентов системной платы диагностическими программами | 2 |
| 2.4 | Практическая работа № 4. Идентификация и установка процессора | 2 |
| 2.4 | Практическая работа №5. Построение последовательности машинных операций для реализации простых вычислений | 2 |
| 2.4 | Практическая работа №6. Программирование арифметических и логических команд | 2 |
| 2.4 | Практическая работа №7. Программирование переходов, ввода-вывода | 2 |
| 3.1 | Практическая работа № 8. Форматирование магнитных дисков. Запись информации на оптические носители | 2 |
| 3.3 | Практическая работа № 9. Работа по подключению акустических систем и с программами обеспечения записи и воспроизведения звуковых файлов | 2 |
| 3.4 | Практическая работа № 10. Работа с настройкой сканеров и программами по сканированию. | 2 |

| | | |
|---|---|---|
| 3.5 | Практическая работа № 11. Настройка параметров работы принтеров. Замена картриджей | 2 |
| 3.6 | Практическая работа № 12. Подключение и работа с нестандартными периферийными устройствами ПК | 2 |
| 4.1 | Практическая работа № 13. Перевод чисел из одной системы счисления в другую | 2 |
| 4.1 | Практическая работа № 14. Выполнение арифметических операций над числами в прямом, обратном и дополнительных кодах | 2 |
| 4.2 | Практическая работа № 15. Логические элементы «2И», «2ИЛИ», «НЕ», «2И-НЕ», «2ИЛИ-НЕ», «Исключающие ИЛИ» | 2 |
| 4.2 | Практическая работа № 16. Мультиплексоры. Демультиплексоры | 2 |
| 4.2 | Практическая работа № 17. Шифраторы. Дешифраторы | 2 |
| 4.2 | Практическая работа № 18. Сумматоры | 2 |
| 4.2 | Практическая работа № 19. Триггеры | 2 |
| 4.2 | Практическая работа № 20. Счетчики | 2 |

# ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 1

## ПРОГРАММИРОВАНИЕ ВВОДА-ВЫВОДА

**Цель занятия:**

Углубление знаний по структуре и принципам работы процессора. Приобрести умение У1Пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации.

Освоение профессиональной компетенции ПК 2.1 Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации.

**Материально-техническое обеспечение:** методические указания, программное обеспечение.

**Краткие теоретические сведения:**

1. Назначение языка Ассемблера.
2. Формат команды Ассемблера.
3. Типы команд Ассемблера.
4. Директивы Ассемблера.
5. Правила записи программы на Ассемблере.
6. Команды ввода-вывода в Ассемблере.

Таблица 1.1 - Команды вывода на экран средствами DOS. Функции DOS

| Номер прерывания | Функция | Описание | При вызове | При возврате |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| Int 21h | 01h | Ввод символа с эхом. Вводит сим вол с клавиатуры и отображает его на экране. При отсутствии символа ждет его ввода | AH=01h | AL=код Введенного символа |

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| | 02h | Вывод символа на экран | AH=02h | DL=код выводимого символа |
| | 06h | Прямой ввод- вывод. Вводит с клавиатуры или выводит символ на экран | AH=06h DL=код Выводимого символа (при выводе) DL=FFh (при вводе) | AL=код введенного символа (при вводе) |
| | 09h | Вывод строки. Строка должна заканчиваться символом $ | AH=09h DS:DX =адрес выводимой строки | |
| | 25h | Установка вектора прерывания позволяет заполнить век тор прерывания ад ресом обработчика прерываний | AH=25h AL=номер Вектора прерывания DS:DX=адрес обработчика прерывания | |
| | 2Ah | Получение текущей даты | AH=2Ah | CX=год DH=месяц DL=день AL=день недели |
| | 2Ch | Получение текущего времени | Ah=2Ch | CH=часы CL=минуты DH=секунды |
| | 2Dh | Установка текущего времени. Позволяет изменить текущее время системных часов | AH=2Dh CH=часы CL=минуты DH=секунды | AL=00h – Успешное выполнение AL=FFh - недопустимое время, системное время не изменилось |

Функции DOS вывода на экран позволяют перенаправлять вывод в файл, но не позволяют вывести текст в любую позицию экрана и не позволяют изменить цвет текста. DOS предполагает, что для более тонкой работы с экраном программы должны использоваться видеофункции BIOS.

Для того чтобы вывести на экран строку текста нужно записать в сегмент данных эту строку:

msg1 db ‘Иванов Иван Иванович$’

Затем в сегменте кода в регистр AH записываем номер функции вывода на экран, в регистр DX записывается адрес строки, затем вызывается прерывание int 21h:

mov AH, 09h

mov DX, offset msg1 int 21h

Чтобы увидеть результат работы программы на экране рекомендуется использовать функцию

01h:

mov AH,01h int 21h

Вывод нескольких символов можно осуществить с помощью цикла. Количество повторений предварительно записывается в регистр CX (счетчик циклов):

mov CX, 223

В сегменте данных следует объявить массив, в который будут помещены символы, выводимые на экран:

symbolsdb 223 dup ('*'); заполняем массив из 224 символов *

Затем нужно заполнить массив символами (в сегменте кода):

; подготовка цикла

mov CX, 223; количество повторений цикла

mov SI, 0; индекс элемента в заполняемом массиве

mov AL, 32; код первого символа

; цикл заполнения массива

begin: movsymbols[SI], AL ;очередной код записывается в массив

inc AL; получаем код следующего символа

inc SI; переходим к следующему элементу массива loopbegin; повторяем цикл заданное число раз. После этого выводим массив на экран:

mov AH, 40h; функция вывода записывается в регистр AH

mov BX, 1; стандартный дескриптор экрана заносим в регистр BX

mov CX, 223; число выводимых байтов заносим в регистр CX

mov DX, offsetsymbols; адрес выводимого сообщения

int 21h; вызов DOS

Ввод с клавиатуры

Как и в случае вывода на экран, DOS предоставляет набор функций для чтения данных с клавиатуры, которые используют стандартное устройство ввода STDIN, так что можно использовать в качестве источника данных файл или стандартный вывод другой программы.

Функции посимвольного ввода без эха можно использовать для интерактивного управления программой.

**Методические указания:**

Работа выполняется каждым обучающимся индивидуально. В процессе выполнения обучающиеся разрабатывают программу и оформляют отчет по работе. Отчет оформляется в электронном варианте, в MicrosoftOfficeWord. Отчет должен содержать: номер работы, тему, цель работы, материально-техническое обеспечение и далее по порядку выполнения работы. К защите необходимо представить отчет в электронном виде (файл в Word). За работу обучающийся получает зачет и оценку.

**Порядок выполнения:**

**Задание 1.** Напишите программу, которая выведет на экран Вашу фамилию, имя и отчество.

Скомпилируйте программу. Проверьте работу программы.

**Задание 2.** Напишите программу, которая выводит на экран символы согласно варианту:

| Вариант | Задание |
|---|---|
| 1 | Прописные буквы латинского алфавита ABCDEFGHIJKL |
| 2 | Строчные буквы латинского алфавита ab c d e f g h i j |
| 3 | Цифры 0 1 2 3 4 5 67 8 9 |

| | |
|---|---|
| 4 | Первые десять прописных букв русского алфавита |
| 5 | Первые десять строчных букв русского алфавита |
| 6 | Символы ( ) * + , - . / |
| 7 | Символы ! " # $ % & ' () |

# ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 2

## УСТАНОВКА КОНФИГУРАЦИИ СИСТЕМЫ ПРИ ПОМОЩИ УЛИТЫ CMOSSETUP

**Цель занятия:**

Получение практических навыков по установке конфигурации системы при помощи утилиты CMOSSETUP.

Приобрести умение У1 Пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации.

Освоение профессиональной компетенции ПК 2.1 Осуществлять установку и настройку отдельных программных, программно-аппаратных средств защиты информации.

**Материально-техническое обеспечение:** методические указания с вариантами задания, программное обеспечение Excel.

**Краткие теоретические сведения:**

Для того чтобы изменить текущие настройки базовой системы ввода-вывода, нужно воспользоваться утилитой **CMOS Setup**, которая вызывается при загрузке системы нажатием на клавишу Delete, или F2, или Esc, или на другую клавишу либо на их сочетание, о чем пользователю сообщается в левом нижнем углу экрана (рис. 2.1).

Графический интерфейс программы позволяет, выбрав нужный пункт меню, внести в нем необходимые изменения. Для навигации здесь используются клавиши управления курсором («вверх», «вниз», «вправо», «влево»). Для выбора нужного пункта меню следует нажать клавишу Enter, а смена значений параметров производится с помощью клавиш «+» и «–» или PageUp и PageDown. Для того чтобы вернуть предыдущее значение настройки, нужно нажать клавишу F5, если же требуется установить значение по умолчанию, то следует воспользоваться клавишей F6, а при необходимости задать оптимальное значение, предопределенное производителем, — F7. Нажатием клавиши F10 можно осуществить быстрый выход из утилиты CMOS Setup с сохранением всех внесенных изменений, а нажатием Escape — без сохранения изменений. Кроме того, нажав F9, можно вы- звать окно системной информации, содержащее данные о процессоре, памяти, названии модели системной платы, версии прошивки BIOS и MAC- адресе интегрированного сетевого контроллера (если таковой существует), а с помощью F1 — окно помощи, где приводится описание горячих клавиш и ассоциированных с ними действий.

Стандартный графический интерфейс утилиты CMOS Setup приведен на рис. 2.2. Рассмотрим настройки, доступные в каждом из пунктов его меню.

Рис.2.1 Вид экрана при начальной загрузке

Рис.2.2 Графический интерфейс утилиты CMOS Setup

Выбрав первый пункт меню —**Standard CMOS Features**, мы попадаем в окно, содержащее настройки стандартных функций CMOS: системные часы, меню инициализации IDE- и FDD-устройств, а также устройств ручного ввода (клавиатуры и мыши) — рис. 3. Что же полезного мы можем сделать в этом окне? Во-первых, если выставлена неверная дата или врут часы, можно подкорректировать их показания, хотя это с успехом можно сделать и непосредственно в ОС. Так что если неточные показания часов

— единственное, что вас не устраивает в настройках вашей системы, то, может быть, и не стоит ради этого заглядывать в BIOS Setup.

Рис. 2.3. Меню Standard CMOS Features утилиты CMOS Setup

Что касается меню инициализации IDE-устройств, то здесь лучше все оставить без изменений, хотя если возникла необходимость отключить один из имеющихся в системе накопителей (например, жесткий диск или DVD-ROM), то это можно сделать путем установки для соответствующего интерфейса, к которому подключено выбранное устройство, значения None (IDE Channel x Master — None). После этого при инициализации устройств BIOS, приняв хранящиеся в CMOS настройки, будет считать, что такого устройства не существует. Вновь подключить это устройство можно установкой измененного ранее параметра в положение Auto.

Что касается настроек FDD-устройств, то что для них не существует

установки, позволяющей выполнить автоматическую инициализацию подключенного устройства, как это сделано для IDE, поэтому тип имеющегося флоппи-дисковода нужно задавать вручную (по умолчанию выбран 3,5-дюймовый FDD, работающий с дискетами 1,44 Мбайт, который является стандартом для всех современных ПК). Некоторые системы при отсутствии или отключении флоппи-дисковода при инициализации устройств во время загрузки BIOS выдают ошибку FDD — во избежание этого необходимо в настройках FDD установить значение None (Drive A — None; Drive B — None).

Пункт главного меню ^ **Advanced BIOS Features** утилиты CMOS Setup открывает доступ к расширенным настройкам функций BIOS, а также позволяет установить очередность устройств, в соответствии с которой будет осуществляться поиск загрузочной записи, или, проще говоря, очередность загрузки (рис. 2.4).

Рис. 2.4. Меню Advanced BIOS Features утилиты CMOS Setup

Меню Advanced BIOS Features обычно имеет пункт HardDiskBootPriority, посредством которого можно задать приоритет загрузки для установленных в системе жестких дисков. Это позволяет решить проблему, возникающую при установке в компьютере нескольких HDD, содержащих загрузочную запись, — при этом BIOS будет пытаться загрузить операционную систему с того из них, который занимает высшую строчку в списке HardDiskBootPriority. Реже подобные установки предусматриваются для съемных дисков и сетевых интерфейсов (RemovableBootPriority и NetworkBootPriority соответственно). Общая очередь загрузки обычно состоит из трех или четырех пунктов: FirstBootDevice, SecondBootDevice, ThirdBootDevice и иногда BootOtherDevice. Для каждой позиции следует выбрать тип загружаемого устройства: либо через специальное меню, вызываемое нажатием на клавишу Enter, либо перебором возможных вариантов с помощью стандартных клавиш управления.

В этом же меню обычно содержатся и другие настройки, относящиеся к процессу загрузки: QuickPowerOnSelfTest (иногда — QuickBoot), позволяющий BIOS пропустить некоторые тесты во время запуска системы (прежде всего это касается теста памяти), что позволяет значительно ускорить загрузку компьютера; BootNumLockStatus, устанавливающий положение (включена или выключена) клавиши NumLock; InitDisplayFirst, определяющий, какой видеоадаптер (интегрированный, подключенный по интерфейсу PCI, AGP или PCI Express) будет инициализирован первым — на подключенный к нему дисплей и будет выводиться информация о ходе загрузки BIOS; PasswordCheck — позволяет установить момент аутентификации

пользователя (при загрузке системы или же при входе в утилиту CMOS Setup); кроме того, здесь обычно можно найти пункты для включения логотипов (FullScreenLogoShow), выводимых на экран во время за- грузки BIOS и скрывающих служебную информацию о ходе инициализации устройств.

Помимо этого, меню Advanced BIOS Features содержит ряд настроек, касающихся работы процессорной подсистемы. Их количество зависит как от модели системной платы, так и от установленного в системе процессора. Рассмотрим наиболее типичные из них.

Настройка Limit CPUID Max. to 3 является наследием былых времен и уже потеряла свою актуальность (ее стоит включать только при установке какой-нибудь устаревшей ОС времен Windows NT 4) — при ее включении CPUID может принимать значения от 0 до 3, после чего становится недоступным ряд используемых CPU современных инструкций.

APIC Mode и MPS VersionControlFor OS — это две взаимосвязанные настройки, поэтому опишем сразу обе.APIC (AdvancedProgrammableInterruptController) — усовершенствованный программируемый контроллер прерываний, его включение является необходимым условием для работы многопроцессорной конфигурации, версия спецификации которой и определяется в пункте MPS VersionControlFor OS Пункт CPU InternalCache позволяет запретить центральному процессору использовать собственную кэш-память, но делать это, конечно, не стоит, поэтому эту настройку лучше оставить в значении Enabled.

В зависимости от того, какие функции реализованы в установленном в системе CPU и какие из них поддерживаются материнской платой, здесь также могут быть дополнительные настройки работы процессора. К примеру, для процессоров Intel это настройки, позволяющие включить следующие технологии:

- CPU Hyper-Threading — технология виртуальной многопроцессорности IntelHyper-Threading;
- CPU EnhancedHalt (C1M) — технология энергосбережения во время простоя процессора EnhancedHalt (C1M);
- CPU Thermal Monitor (TM) — технология термоконтроля процессора Intel Thermal Monitor (TM);
- CPU Thermal Monitor 2 (TM2) — технология термоконтроля процессора Intel Thermal Monitor 2 (TM2);
- CPU EIST Function — функция энергосбережения Enhanced Intel SpeedStep;
- VirtualizationTechnology — технология виртуализации;
- ExecuteDisableBit — функция процессора, в случае поддержки ее операционной системой обеспечивающая защиту от вирусных атак и вредоносного кода, направленных на переполнение буфера памяти.

В случае если материнская плата построена на чипсете с интегрированным графическим ядром, то меню Advanced BIOS Features имеет пункт, который, например, может называться On-ChipFrameBufferSize или

FrameBufferSize, позволяющий задать размер буфера в системной памяти, отведенный в распоряжение встроенной графической подсистемы.

Выбрав пункт меню CMOS Setup — ^ **AdvancedChipsetFeatures**, мы попадаем в меню, содержащее расширенные настройки чипсета и позволяющие оптимизировать производительность системы (рис. 2.5).

Набор доступных в CMOS Setup параметров у разных моделей системных плат даже одного и того же производителя может сильно различаться, что в значительной мере зависит как от возможностей набора микросхем, на котором построена материнская плата, так и от ее позиционирования.

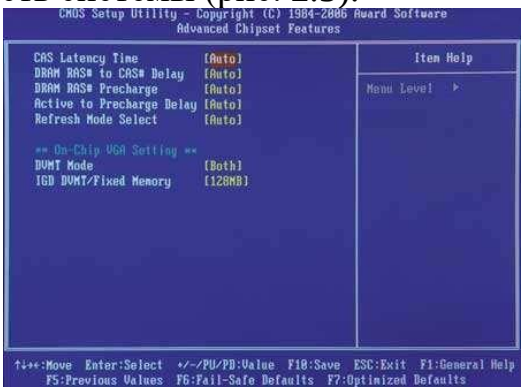



Рис. 2.5. Меню Advanced Chipset Features утилиты CMOS Setup

**Порядок выполнения:**

**Задание 1.** Запустите утилиту CMOS Setup. Изучите разделы утилиты CMOS Setup.

**Задание 2.** Поменяйте дату и время на компьютере.

**Задание 3.** Просмотрите и запишите информацию о жестких дисках.

**Задание 4.** Определите количество оперативной памяти.

**Задание 5.** Установите на следующий порядок загрузки (предварительно запомнив текущие настройки): CD-ROM, HDD-0, LAN. Верните настройки в первоначальное состояние.

**Задание 6.** Определите напряжение ядра процессора, также напряжение питания каждой линии на материнской плате. Также определите текущую температуру ядра процессора и скорость вращения вентилятора процессора.

**Задание 7.** Определите частоту системной шины, частоты шин PCI и AGP, частоту шины памяти.

# ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 3

## ТЕСТИРОВАНИЕ КОМПОНЕНТОВ СИСТЕМНОЙ ПЛАТЫ ДИАГНОСТИЧЕСКИМИ ПРОГРАММАМИ

**Цель работы:**

1. Освоение профессиональной компетенции ПК 1.4 Осуществлять проверку технического состояния, техническое обслуживание и текущий ремонт, устранять отказы и восстанавливать работоспособность автоматизированных (информационных) систем в защищенном исполнении.

2. Освоение У2 Правильно эксплуатировать и устранять типичные выявленные дефекты технических средств информатизации.

**Материально-техническое обеспечение:** методические указания, компьютер.

**Краткие теоретические сведения:**

Диагностика (тестирование) материнской платы

Проводится комплексная проверка материнской платы на предмет стабильной работы со всеми периферийными устройствами с помощью специальных программ диагностики. Процесс тестирования материнской платы может занять достаточно длительное время ввиду сложности устойства самой материнской платы, так и проблем совместимости с другими компонентами компьютера. Для теста материнской платы возможно привлечение других комплектующих для выяснения причин некорректной работы в данной конфигурации.

Материнская плата (системная плата) – самая главная и большая по размерам печатная плата в персональном компьютере. Именно на ней устанавливаются или крепятся все основные составные части компьютера: процессор, оперативная память, карты расширения и т.д. Материнская пла та – сложное устройство, состоящее из многих микросхем и других электронных компонентов, которые отвечают за нормальную работу установленных комплектующих их взаимодействия между собой. Так же на самой материнской плате обычно распаяны специальные контроллеры, отвечающие за соединение с различными периферийными устройствами (порты USB, PS/2 и т.п.), подключение компьютера с внешним миром (сетевые карты) или вывод аудио-видео информации (видеоадаптеры, аудио карты).

Что могут диагностические программы?

Утилиты для диагностики и анализа ПК выполняют полную инвентаризацию компьютера. После проведения обследования они выводят список с данными обо всех компонентах: процессоре, жестком диске, материнской плате, включая информацию об их производительности. Эти сведения окажутся весьма кстати, если, например, вы решили выполнить апгрейд своего компьютера или продать его (в объявлении необходимо будет указать технические характеристики ПК). Что же умеют делать диагностические программы? Вот краткий перечень их возможностей:

1. сбор сведений о комплектующих – «продвинутые» программы предоставляют самую подробную информацию, «отстающие» же вместо этого выводят на экран набор специфических обозначений, понятных только специалисту по «железу». Хотите нарастить объем оперативной памяти? С помощью программы диагностики вы сможете узнать, сколько слотов уже занято и какие модули памяти используются в системе.

2. обработка информации. Сбор сведений – это только первый шаг. Главная задача приложения заключается в том, чтобы представить данные в простом и наглядном виде. Так, например, аббревиатуру DDR2 SDRAM хорошая программа не оставит без пояснений – обозначение должно быть расшифровано и занесено в соответствующую категорию (в данном случае

– в раздел «Оперативная память»). При запросе подробной

информации по этому пункту пользователь увидит дополнительные сведения – например, о рабочей частоте оперативной памяти. Благодаря этим сведениям вы сможете точно узнать, какой тип модулей ОП следует покупать для своего компьютера.

3. тестирование производительности ПК – полученный результат можно сравнить с эталонными системами, чтобы принять правильное решение о том, стоит ли проводить модернизацию компьютера.

4. вывод подробных справочных сведений – диагностические утилиты, помимо информации о комплектующих, предоставляют и более подробные данные о том или ином компоненте системы или дают ссылки на веб-страницы производителя. Например, вам требуется проверить, установлена ли на вашем ПК самая последняя версия драйверов для звуковой платы. Для этого в программе диагностики перейдите в раздел аудиоустройств: здесь вы найдете информацию об актуальной версии драйверов. Идеальным дополнением к этому будет ссылка на веб-страницу производителя звуковой платы для загрузки соответствующего ПО.

5. обнаружение неисправности – если Windows отказывается загружаться, очень сложно определить причину поломки. В таких случаях поможет программа диагностики, записанная на «реанимационном» компакт-диске. Тогда даже при неработающей ОС можно будет определить, какой именно компонент вышел из строя.

*Как работают диагностические программы?*

Программа «распознает» комплектующие с помощью специальных диагностических команд, извлекая информацию об устройствах из их микропрограммного обеспечения (firmware), драйверов или системного реестра.

*Для чего нужна функция диагностики?*

Некоторые диагностические программы не только информируют об установленных в компьютере комплектующих, но и проверяют отдельные компоненты на наличие «слабых мест». Так, они умеют выявлять причины возникновения перегрузок ПК. Приведем пример: в жаркий летний денек компьютер отказывается работать. Это может быть вызвано перегревом процессора, который недостаточно хорошо охлаждается. Благодаря функции диагностики эту проблему можно решить еще до появления сбоев, установив более мощный кулер.

*Могут ли программы диагностики устранять сбои в работе ПК?*

Нет, программы для анализа и диагностики не могут проводить ремонт ПК. Дефекты аппаратных средств в принципе невозможно исправить при помощи программ. Однако с их помощью можно узнать, в чем именно заключается проблема. Для этого в диагностические утилиты включены специальные тесты, которые проверяют либо всю систему в целом, либо отдельные комплектующие. Например, если компьютер все время зависает, стоит провести тест модулей оперативной памяти. Если в результатах теста содержится много ошибок, это значит, что пришла пора менять ОП. Какой тип модуля памяти нужен и с каким максимальным объемом – все это

должна рассказать программа диагностики. Если она обнаружит, что жесткий диск вышел из строя – придется его заменить.

При возникновении незначительных проблем хорошие программы дают советы по их решению – например, сообщать, что появились обновления для «прошивок» устройств или новые версии драйверов. Правда, загружать из Интернета и устанавливать обновления вам придется самостоятельно.

**Порядок выполнения:**

1. Ответьте на теоретические вопросы:

1.1. Основные компоненты материнской платы.

1.2. Основные информационные шины.

1.3. Порты и контроллеры.

1.4. Понятие интерфейса. Виды интерфейсов.

1.5. Приведите примеры диагностических программ. Опишите их возможности.

2. Заполните таблицы:

Таблица 3.1

| | | | |
|---|---|---|---|
| Тип компьютера | | тип чипсета системной платы | |
| Тип ОС | | количество оперативной памяти | |
| имя компьютера | | тип видеоадаптера | |
| тип центрального процессора | | тип монитора | |
| тип системной платы | | тип и объем НЖМД | |
| имя пользователя | | перечислить устройства ввода\вывода | |

Таблица 3.2

| | |
|---|---|
| название материнской платы и фирмы изготовителя | |
| свойства системной шины | |
| свойства шины памяти | |
| название чипсета | |
| физическая информация о системной плате | |

Таблица 3.3

| | | | |
|---|---|---|---|
| название «северного моста» | | название «южного моста» | |
| версия | | версия | |
| фирма изготовитель | | фирма изготовитель | |

# ПРАКТИЧЕСКАЯ РАБОТА №4

## ИДЕНТИФИКАЦИЯ И УСТАНОВКА ПРОЦЕССОРА

**Цель:** приобрести практические навыки в идентификации и установке процессора.

**Сформировать следующие умения:**

**У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель работы: и**зучение характеристик процессора.

**Форма организации занятия**: работа малыми группами.

**Студент должен**

*Знать*:

- Основные характеристики процессора;
- Основные современные модели процессоров;
- Типы процессоров нового поколения;

*Уметь*:

- Идентифицировать и устанавливать процессоры.

**Методические указания**:

Основа вычислительной системы – микропроцессор (МП). МП характеризуется следующими параметрами:

Тактовая частота

- Степень интеграции микросхемы (сколько транзисторов содержится в чипе).
- Внутренняя разрядность данных (количество бит, которые МП может обрабатывать одновременно);
- Внешняя разрядность данных (количество одновременно передаваемых бит в процессе обмена данными с памятью и другими устройствами);
- Адресуемая память (зависит от числа адресных бит).

Типы МП

| Тип процессора | Поколение | Год выпуска | Разрядность шины данных | Разрядность шины адреса | Первичная кэш-память, Кбайт | | Тактовая частота шины, МГц | Тактовая частота процессора, МГц | Количество транзисторов, млн | Размер минимальной структуры, мкм |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Команды | Данные | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 8088 | 1 | 1979 | 8 | 20 | | | 4,77-8 | 4,77-8 | | |
| 8086 | 1 | 1978 | 16 | 20 | | | 4,77-8 | 4,77-8 | 0,029 | 3,0 |
| 80286 | 2 | 1982 | 16 | 24 | | | 6-20 | 6-20 | 0,13 | 1,5 |
| 80386DX | 3 | 1985 | 32 | 32 | | | 16-33 | 16-33 | 0,27 | 1,0 |
| 80386SX | 3 | 1988 | 16 | 32 | 8 | | 16-33 | 16-33 | 0,27 | 1,0 |
| 80486DX | 4 | 1989 | 32 | 32 | 8 | | 25-50 | 25-50 | 1,2 | 1,0-0,8 |
| 80486SX | 4 | 1989 | 32 | 32 | 8 | | 25-50 | 25-50 | 1,1 | 0,8 |
| 80486DX2 | 4 | 1992 | 32 | 32 | 8 | | 25-40 | 25-40 | | |
| 80486DX4 | 5 | 1994 | 32 | 32 | 8 | 8 | 25-40 | 75-120 | | |
| Pentium | 5 | 1993 | 64 | 32 | 8 | 8 | 60-66 | 60-200 | 3.1-3.3 | 0.8-0.35 |
| P-MMX | 5 | 1997 | 64 | 32 | 16 | 16 | 66 | 166-233 | 4.5 | 0.6-0.35 |
| Pentium Pro | 6 | 1995 | 64 | 32 | 8 | 8 | 66 | 150-200 | 5.5 | 0.35 |
| Pentium II | 6 | 1997 | 64 | 32 | 16 | 16 | 66 | 233-300 | 7.5 | 0.35-0.25 |
| Pentium II Celeron | 6 | 1998 | 64 | 32 | 16 | 16 | 66/100 | 266-533 | 7.5-19 | 0.25 |
| Pentium Xeon | 6 | 1998 | | | | | 100 | 400-1700 | | 0.18 |
| Pentium III | 6 | 1999 | 64 | 32 | 16 | 16 | 100 | 450-1200 | 9.5-44 | 0.25-0.13 |
| AMD Athlon | 7 | 1999 | 64 | 32 | 64 | 64 | 266 | 500-2200 | 22 | 0.25 |

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| Pentium 4 | 7 | 2000 | 64 | 32 | 12 | 8 | 400 | 1.4-3.4 ГГц | 42-125 | 0,18-0,09 |
| AMD Athlon 64 | 8 | 2003 | 64 | 64 | 64 | 64 | 400 | 2 ГГц | 54-106 | 0,13-0,09 |
| Pentium 4 Prescott | 8 | 2004 | 64 | 32 | 16 | 16 | 800 | 2.8-3.4 ГГц | 125 | 0.09 |

В сущности, МП – плоский квадратный слой кремния со схемами, выгравированными на его поверхности. Этот элемент укрепляется на основе - керамической или пластмассовой – образуя пакет с контактами, выполненными или по плоской нижней стороне, или по одному из краев. Пакет ЦП связан с системной платой через разъем формы гнездо (Soket) и слот (Slot).

Характеристики интерфейсов процессоров

| Наименование | Разъем (число контактов) | Описание |
|---|---|---|
| Soket 1 | 169 | Устанавливался на системных платах 486, напряжение 5 В, поддерживает 486 чипсет |
| Soket 2 | 238 | Минимальное обновление Soket 1, поддерживает те же схемные элементы |
| Soket 3 | 237 | Рабочее напряжение 5 В, но может работать также и при 3,3 В, переключатели размещены на системной плате |
| Soket 4 | 273 | Первый разъем, спроектированный для Pentium. рабочее напряжение 5 В |
| Soket 5 | 320 | Работает при 3,3 В поддерживает Pentium от 75 до 133 МГц |
| Soket 6 | 235 | Разработан для процессора 486. напряжение 3,3 В |
| Soket 7 | 321 | Разработан для Pentium-MMX, используется для всех Pentium с частотой шины 66 МГц |
| Soket 8 | 387 | Используется только для Pentium Pro, из-за дороговизны быстро вышел из употребления |
| Slot 1 | 242 | Поддерживает кэш-память L1 до 512 Кбайт, состоящая из двух по 256 Кбайт; оперирует на половинной частоте МП. Использовался для Pentium II, Pentium III и Celeron |
| Slot 2 | 330 | Аналогичен разъему Slot 1, но поддерживает до 2 Мбайт кэш-памяти, работающей на частоте МП. Использовался для Pentium II, Pentium III, Pentium Xeon |
| Slot A | 242 | Разработан для AMD Athlon, механически совместим с разъемом Slot 1, но поддерживает совершенно другие электрические цепи |
| Soket 370 | 370 | Заменяет Slot 1 для МП Celeron. Также используется для Pentium III |
| Soket A | 462 | Разработан для процессора AMD Athlon, который содержал на кристалле кэш-память L2 |
| Soket 423 | 423 | Введен для удовлетворения новых требований Pentium 4, который содержит совершенно новую системную шину (FSB). Включает теплорассеиватель |
| Soket 603 | 603 | Предназначен для Pentium 4. дополнительные контакты ориентированы на МП, которые будут содержать на кристалле кэш-память, а также для подключения других процессоров в мультипроцессорных системах |

| | | |
|---|---|---|
| Soket 478 | 478 | Разработан для поддержки технологий 0,13-мкм для Pentium 4 Northwood. |
| Socket AM2 940 | 940 | Выпущен AMD в 2006 году для Athlon 64 X2 на 5000+ и 4000+ |
| Socket LGA 1155 | 1155 | Предназначен для процессора Intel Core i7-3960X (Sandy Bridge-E). Процессор поддерживает четыре канала памяти и 40 линий PCI Express, что потребовало намного большего количества ножек для питания и передачи сигналов. |

**Задания на практическую работу**:

**Задание 1.**

Выберите процессор, подходящий для установки на целевой системной плате. Установите процессор на целевую системную плату

**Задание 2**.

Идентифицируйте процессор целевого компьютера. Назовите его основные характеристики. Дайте рекомендации по модернизации целевого компьютера.

**Вопросы для зачета**

1. Перечислите функции процессора
2. Прокомментируйте основные параметры процессора
3. Какие современные типы процессоров вы знаете?

**Контрольные вопросы:**

1. Перечислите известные вам марки процессоров.
2. Какие параметры системной платы необходимо учесть при подборе процессора.
3. Какие сокеты для установки процессора вам известны
4. Как правильно установить процессор Intel
5. Как правильно установить процессор AMD
6. Перечислить функции процессора
7. Какие современные типы процессоров вы знаете

# ПРАКТИЧЕСКАЯ РАБОТА №5

## ПОСТРОЕНИЕ ПОСЛЕДОВАТЕЛЬНОСТИ МАШИННЫХ ОПЕРАЦИЙ ДЛЯ РЕАЛИЗАЦИИ ПРОСТЫХ ВЫЧИСЛЕНИЙ

**Цель:** приобрести практические навыки в построении последовательности машинных операций для реализации простых вычислений.

**Сформировать следующие умения: У1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Используемое ПО:** программа-тренажер CompModel, инструкция к программе.

**Теоретическая часть:**

**Описание архитектуры учебной ЭВМ**

1. **Структура ЭВМ**

Моделируемая ЭВМ включает процессор, оперативную (ОЗУ) и сверхоперативная память, устройство ввода (УВв) и устройство вывода (УВыв). Процессор в свою очередь, состоит из центрального устройства управления (УУ), арифметического устройства (АУ) и системных регистров (CR, PC, и др.). Структурная схема ЭВМ показана на рис. 1.

В ячейках ОЗУ хранятся команды и данные. Емкость ОЗУ составляет 1000 ячеек. По сигналу MWr выполняется запись содержимого регистра данных (MDR) в ячейку памяти с адресом, указанным в регистре адреса (MAR) По сигналу MRd происходит считывание - содержимое ячейки памяти с адресом, содержащимся в MAR, передается в MDR.

Сверхоперативная память с прямой адресацией содержит десять регистров общего назначения R0-R9. Доступ к ним осуществляется (аналогично доступу к ОЗУ) через регистры RAR и RDR.

АЛУ осуществляет выполнение одной из арифметических операций, определяемой кодом операции (COP), над содержимым аккумулятора (Acc) и регистра операнда (DR). Результат операции всегда помещается в Acc. При завершении выполнения операции АУ вырабатывает сигналы признаков результата: Z (равен 1, если результат равен нулю); S (равен 1, если результат отрицателен); OV (равен 1, если при выполнении операции произошло переполнение разрядной сетки). В случаях, когда эти условия не выполняются, соответствующие сигналы имеют нулевое значение.

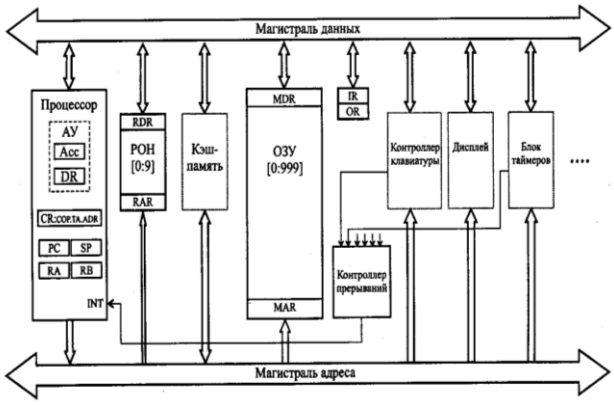
В модели ЭВМ предусмотрены внешние устройства двух типов. Во-первых регистры IR и OR, которые могут обмениваться с аккумулятором с помощью безадресных команд IN (Acc := IR) и OUT (OR := Acc). Во-вторых, это моделей внешних устройств, которые могут подключаться к системе и взаимодействовать с ней в соответствии с заложенными в моделях алгоритмами. Каждое внешнее устройство имеет ряд программно-доступных регистров, может иметь собственный *обозреватель* (окно видимых элементов).

УУ осуществляет выборку команд из ОЗУ в последовательности, определяемой естественным порядком выполнения команд (т. е. в порядке возрастания адресов команд в ОЗУ) или командами передачи управления; выборку из ОЗУ операндов, задаваемых адресами команды; инициирование выполнения операции, предписанной командой; останов или переход к выполнению с следующей команды.

В качестве сверхоперативной памяти в модель включены регистры общего назначения (РОН), и может подключаться модель кэш-памяти.

В состав УУ ЭВМ входят:

- PC - счетчик адреса команды, содержащий адрес текущей команды;
- CR - регистр команды, содержащий код команды;
- RB - регистр базового адреса, содержащий базовый адрес;
- SP - указатель стека, содержащий адрес верхушки стека;




• RA - регистр адреса, содержащий исполнительный адрес при косвенной адресации.

Регистры Acc, DR, IR, OR, CR и все ячейки ОЗУ, и РОН имеют длину 6 десятичных разрядов, регистры PC, SP, RA и RB - 3 разряда.

2. **Представление данных в модели**

Данные в ЭВМ представляются в формате, показанном на рис. 2. Это целые десятичные числа, изменяющиеся в диапазоне "-99 999 ... +99 999", содержащие знак и 5 десятичных цифр.

| 0 | 1 2 3 4 5 |
|---|---|
| знак | Десятичные цифры |

Рис. 2. Форматы десятичных данных

Старший разряд слова данных используется для кодирования знака: плюс (+) изображается как О, минус (-) - как 1. Если результат арифметической операции выходит за пределы указанного диапазона, то говорят, что произошло переполнение разрядной сетки. АЛУ в этом случае вырабатывает сигнал переполнения OV = 1. Результатом операции деления является целая часть частного. Деление на ноль вызывает переполнение

3. **Система команд**

При рассмотрении системы команд ЭВМ обычно анализируют три аспекта: форматы, способы адресации и систему операций.

**3.1. Форматы команд**

Большинство команд учебной ЭВМ являются одноадресными или безадресными, длиной в одно машинное слово (6 разрядов). Исключение оставляют двухсловные команды с непосредственной адресацией и команда MOV, являющаяся двухадресной.

В форматах команд выделяется три поля:

• два старших разряда [0:1] определяют код операции COP;

• разряд 2 может определять тип адресации (в одном случае (формат 5а) он определяет номер регистра);

• разряды [3:5] могут определять прямой или косвенный адрес памяти, номер регистра (В команде MOV номера двух регистров), адрес перехода или короткий непосредственный операнд. В двухсловных командах непосредственный операнд занимает поле [6: 11].

Полный список форматов команд показан на рис.3, где приняты следующие обозначения:

COP - код операции;

ADR - адрес операнда в памяти;

ADC - адрес перехода;
I - непосредственный операнд;
R, Rl, R2 - номер регистра;
ТА - тип адресации;
X-разряд не используется.

| Номер формата | 0 1 | 2 | 3 4 5 | | |
|---|---|---|---|---|---|
| 1 | COP | X | X X X | | |
| 2 | COP | TA | ADR | | |
| 3 | COP | TA | X X R | | |
| 3a | COP | TA | X R1 R2 | 6 | 6 11 |
| 4 | COP | X | X X X | 1 | I |
| 5 | COP | X | ADC | | |
| 5a | COP | R | ADC | | |

Рис. 3. Форматы команд учебной программы.

### 3.2. Способы адресации

В ЭВМ принято различать пять основных способов адресации: *прямая, косвенная, непосредственная, относительная, безадресная.*

Каждый способ имеет разновидности. В модели учебной ЭВМ реализованы семь способов в адресации, приведенные в табл1.

Таблица 1. Адресация в командах учебной ЭВМ.

| Код ТА | Тип адресации | Исполнительный адрес |
|---|---|---|
| 1 | 2 | 3 |
| 0 | Прямая (регистровая) | ADR(R) |
| 1 | Непосредственная | - |
| 1 | 2 | 3 |
| 2 | Косвенная | ОЗУ(ADR)[3:5] |
| 3 | Относительная | ADR+RB |
| 4 | Косвенно-регистровая | РОН(R)[3:5] |
| 5 | Индексная с постинкрементом | РОН(R)[3:5], R:= R + I |
| 6 | Индексная с преддекрементом | R:= R -1, РОН(R)[3 :5] |

### 3.3. Система операций

Система команд учебной ЭВМ включает команды следующих классов:

- *арифметико-логические и специальные*: сложение, вычитание, умножение, деление;
- *пересылки и загрузки:* чтение, запись, пересылка (из регистра в регистр), помещение в стек, извлечение из стека, загрузка указателя стека, загрузка базового регистра;
- *ввода/вывода:* ввод, вывод;
- *передачи управления:* безусловный и шесть условных переходов, вызов подпрограммы, возврат из подпрограммы, цикл, программное прерывание, возврат из прерывания.
- системные: пустая операция, разрешить прерывание, запретить прерывание, стон.

Список команд учебной ЭВМ приведен в табл. 4 и 6.

## 4. Состояния и режимы работы ЭВМ

Ядром УУ ЭВМ является управляющий автомат (УА), вырабатывающий сигналы управления, которые инициируют работу АЛУ, РОН, ОЗУ и УВВ, передачу информации между регистрами устройств ЭВМ и действия над содержимым регистров УУ.

ЭВМ может находиться в одном из двух состояний: Останов и Работа.

В состояние **Работа** ЭВМ переходит по действию команд Пуск или Шаг. Команда Пуск запускает выполнение программы, представляющую собой последовательность команд, записанных в ОЗУ, в автоматическом режиме до команды HLT или точки останова. Программа выполняется по командам, начиная с ячейки ОЗУ, на которую показывает PC, причем изменение состояний объектов модели отображается в окнах обозревателей.

В состояние **Останов** ЭВМ переходит по действию команды Стоп или автоматически в зависимости от установленного режима работы.

Команда **Шаг**, в зависимости от установленного режима работы, запускает выполнение одной команды или одной микрокоманды (если установлен **Режим микрокоманд**), после чего переходит в состояние **Останов**.

В состоянии **Останов** допускается просмотр и модификация объектов модели: регистров процессора и РОН, ячеек ОЗУ, устройств ввода/вывода. В процессе модификации ячеек ОЗУ и РОН можно вводить данные для программы в ячейки ОЗУ - программу в кодах. Кроме того, в режиме **Останов** можно менять параметры модели и режимы ее работы, вводить и/или редактировать программу в мнемокодах, ассемблировать мнемокоды, выполнять стандартные операции с файлами.

**5. Интерфейс ЭВМ**

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***.

**Окна основных обозревателей системы**

**Окно Процессор**

Окно Процессор обеспечивает доступ ко всем регистрам и флагам процессоров.

***Программно-доступные регистры и флаги:***

Acc — аккумулятор;

PC — счетчик адреса команды, содержащий адрес текущей команды;

SP — указатель стека, содержащий адрес верхушки стека;

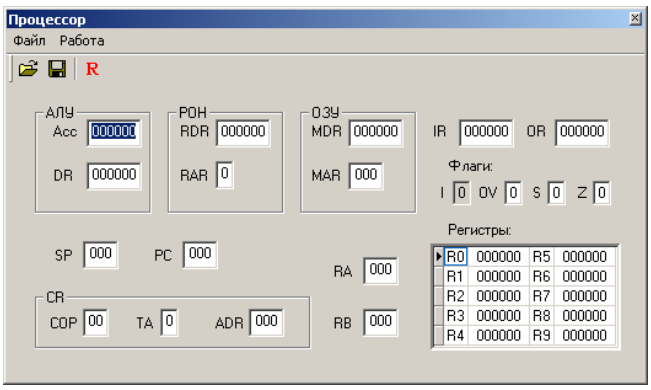



Рис. 8.4. Окно Процессор

RB — регистр базового адреса, содержащий базовый адрес;

RA — регистр адреса, содержащий исполнительный адрес при косвенной адресации;

IR — входной регистр;

OR — выходной регистр;

I— флаг разрешения прерываний.

***Системные регистры и флаги:*** DR — регистр данных АЛУ, содержащий второй операнд;

MDR — регистр данных ОЗУ; MAR — регистр адреса ОЗУ; RDR — регистр данных блока РОН; RAR — регистр адреса блока РОН; CR — регистр команд, содержащий поля: COP— код операции; TA — тип адресации; ADR— адрес или непосредственный операнд;

Z— флаг нулевого значения Acc;

S— флаг отрицательного значения Ace;

OV— флаг переполнения.

Регистры Acc, DR, IR, OR, CR и все ячейки ОЗУ, и РОН имеют длину 6 десятичных разрядов, регистры PC, SP, RA и RB — 3 разряда. В окне **Процессор** отражаются текущие значения регистров и флагов, причем в состоянии **Останов** все регистры, включая регистры блока РОН, и флаги (кроме флага 1) доступны для непосредственного редактирования.

Элементы управления окна **Процессор** включают меню и кнопки, вызывающие команды:

- **Сохранить;**
- **Загрузить;**
- **Reset;**
- **Reset R0-R9** (только команда меню **Работа).**

Команды **Сохранить, загрузить** позволяют сохранить текущее значение регистров и флагов процессора в файле и восстановить состояние процессора из файла. Команда **Reset** и кнопка R устанавливают все регистры (в т. ч. блок РОН) в начальное (нулевое) значение. Содержимое ячеек памяти при этом не меняется. Выполняемая лишь из меню **Работа** команда Reset R0-R9 очищает только регистры блока РОН.

**Окно *Память***

Окно **Память** (рис. 8.5) отражает текущее состояние ячеек ОЗУ. В этом окне допускается редактирование содержимого ячеек, кроме того, предусмотрена возможность выполнения (через меню или с помощью кнопок панели инструментов) пяти команд:

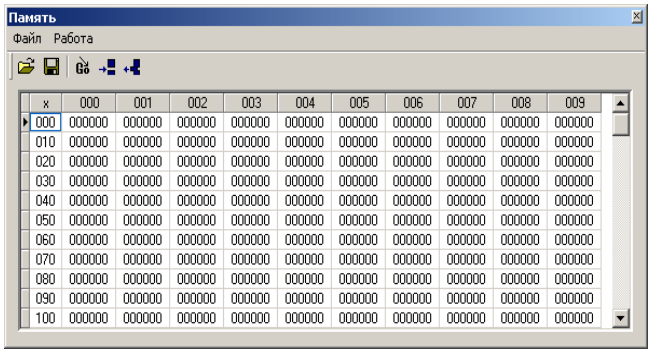


Рис. 8.5. Окно Память

**сохранить, загрузить, перейти к, вставить, Убрать.**

Команды **Сохранить, загрузить** во всех окнах, где они предусмотрены, работают одинаково — сохраняют в файле текущее состояние объекта (в данном случае памяти) и восстанавливают это состояние из выбранного файла, причем файл в каждом окне записывается по умолчанию с характерным для этого окна расширением.

Команда **Перейти к** открывает диалоговое окно, позволяющее перейти на заданную ячейку ОЗУ.

Команда **Убрать** открывает диалог, в котором указывается диапазон ячеек cm non. Содержимое ячеек в этом диапазоне теряется, а содержимое ячеек [(n +1): 999] перемещается в соседние ячейки с меньшими адресами. Освободившиеся ячейки с адресами 999, 998, ... заполняются нулями.

Команда **Вставить,** позволяющая задать номера ячеек, перемещает содержимое всех ячеек, начиная от*m* -й на*n* -*m* позиций в направлении больших адресов, ячейки заданного диапазона*[m:n]* заполняются нулями, а содержимое последних ячеек памяти теряется.

**Окно *Текст программы***

Окно **Текст программы** (рис. 8.6) содержит стандартное поле текстового редактора, в котором можно редактировать тексты, загружать в него текстовые файлы и сохранять подготовленный текст в виде файла.

Команды меню **Файл:**

**Новая** — открывает новый сеанс редактирования;

**Загрузить** — открывает стандартный диалог загрузки файла в окно редактора;

**Сохранить** — сохраняет файл под текущим именем;

**Сохранить как** — открывает стандартный диалог сохранения файла;

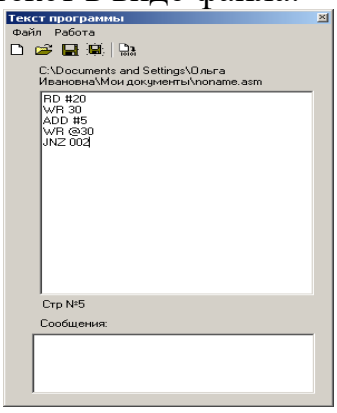


Рис. 8.6. Окно Текст программы

**Вставить** — позволяет вставить выбранный файл в позицию курсора.

Все перечисленные команды, кроме последней, дублированы кнопками на панели инструментов окна. На той же панели присутствует еще одна кнопка — **Компилировать,** которая запускает процедуру ассемблирования

текста в поле редактора.

Ту же процедуру можно запустить из меню **Работа.** Команда **Адрес вставки** позволяет задать адрес ячейки ОЗУ, начиная с которой программа будет размещаться в памяти. По умолчанию этот адрес принят равным 0.

Ниже области редактирования в строку состояния выводится позиция текущей строки редактора — номер строки, в которой находится курсор.

В случае обнаружения синтаксических ошибок в тексте программы диагностические сообщения процесса компиляции выводятся в окно сообщений, запись в память кодов (даже безошибочного начального фрагмента программы) не производится.

После исправления ошибок и повторной компиляции выдается сообщение об отсутствии ошибок, о расположении и размере области памяти, занятой под ассемблированную программу.

Окно ***Программа***

Окно **Программа** (рис. 8.7) отображает таблицу, имеющую 300 строк и4 столбца. Каждая строка таблицы соответствует дизассемблированной ячейке ОЗУ. Второй столбец содержит адрес ячейки ОЗУ, третий — дизассемблированный мнемокод, четвертый — машинный код команды. В первом столбце может помещаться указатель — на текущую команду (текущее значение PC) и точка останова — красная заливка ячейки.

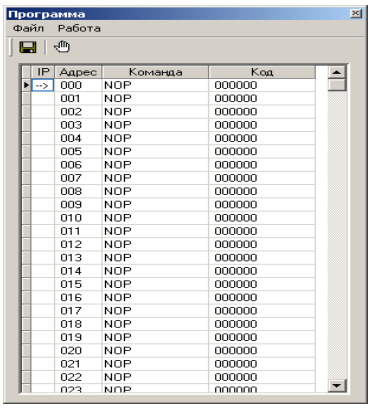


Рис. 8.7. Окно Программа

Окно **Программа** позволяет наблюдать процесс прохождения программы. В этом окне ничего нельзя редактировать. Органы управления окна позволяют сохранить содержимое окна в виде текстового файла, выбрать начальный адрес области ОЗУ, которая будет дизассемблироваться (размер области постоянный — 300 ячеек), а также установить/снять точку останова. Последнее можно проделать тремя способами: командой **Точка останова** из меню **Работа,** кнопкой на панели инструментов или двойным щелчком мыши в первой ячейке соответствующей строки. Характерно, что прочитать в это окно ничего нельзя. Сохраненный текстовыйasm-файл можно загрузить в окно **Текст программы,** ассемблировать его и тогда дизассемблированное значение заданной области памяти автоматически появится в окне **Программа.** Такую процедуру удобно использовать, если программа изначально пишется или редактируется непосредственно в памяти в машинных кодах. Начальный адрес области дизассемблирования задается в диалоге командой **Начальный адрес** меню **Работа.**

**Окно *Микрокомандный уровень***

Окно **Микрокомандный уровень** (рис. 8.8) используется только в режиме микрокоманд, который устанавливается командой **Режим микрокоманд** меню **Работа.** В это окно выводится мнемокод выполняемой команды, список микрокоманд, ее реализующих, и указатель на текущую

выполняемую микрокоманду.

Шаговый режим выполнения программы или запуск программы в автоматическом режиме с задержкой командного цикла позволяет наблюдать процесс выполнения программы на уровне микрокоманд.

Если открыть окно **Микрокомандный уровень,** не установив режим микрокоманд в меню **Работа,** то после начала выполнения программы в режиме **Шаг** (или в автоматическом режиме) в строке сообщений окна будет выдано сообщение "Режим микрокоманд неактивен".

Рис. 8.8. Окно Микрокомандный уровень

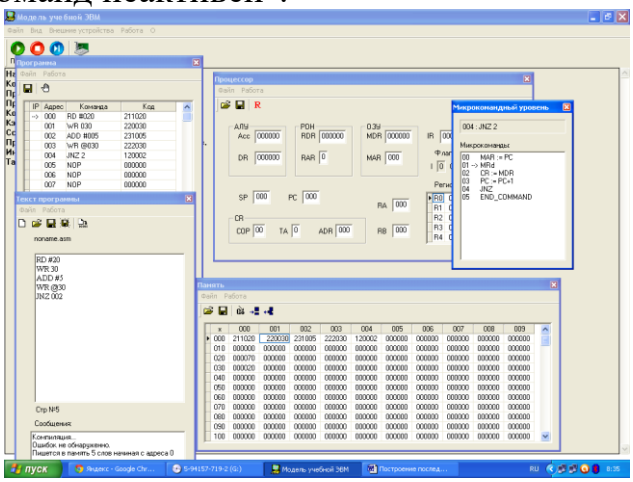


Рис. 8.9. Окна в режиме микрокомандного уровня

**Вспомогательные таблицы:**

В данном разделе представлены вспомогательные таблицы (табл. 8.4—8.8) для работы с моделью учебной ЭВМ.

Таблица 8.4 - Таблица команд учебной ЭВМ

| **Мл/Ст.** | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | NOP | JMP | | MOV | |
| 1 | IN | JZ | RD | RD | RDI |
| 2 | **OUT** | JNZ | WR | WR | |
| 3 | IRET | JS | ADD | ADD | ADI |
| 4 | WRRB | JNS | SUB | SUB | SBI |
| 5 | WRSP | JO | MUL | MUL | MULI |
| 6 | PUSH | JNO | DIV | DIV | DIVI |
| 7 | POP | JRNZ | | IN | |
| 8 | RET | INT | EI | OUT | |
| 9 | HLT | CALL | DI | | |

Таблица 8.5 - Типы адресации, их коды и обозначение

| **Обозначение** | Код | **Тип адресации** | **Пример команды** |
|---|---|---|---|
| | 0 | Прямая (регистровая) | ADD **23** (ADD **R3)** |
| # | 1 | Непосредственная | ADD **#33** |
| @ | 2 | Косвенная | ADD **@33** |
| **[ ]** | 3 | Относительная | ADD [33] |
| **@R** | 4 | Косвенно-регистровая | ADD @R3 |
| **@R+** | 5 | Индексная с постинкрементом | ADD @R3+ |
| **-@R** | 6 | Индексная с преддекрементом | ADD -@R3 |

В табл. 8.6 приняты следующие обозначения:

DD— данные, формируемые командой в качестве (второго) операнда: прямо или косвенно адресуемая ячейка памяти или трехразрядный непосредственный операнд;

R* — содержимое регистра или косвенно адресуемая через регистр ячейка памяти;

ADR* — два младших разряда ADR поля регистра CR;

V— адрес памяти, соответствующий вектору прерывания;

M(*) — ячейка памяти, прямо или косвенно адресуемая в команде; I— пятиразрядный непосредственный операнд со знаком.

Таблица 8.6 - Система команд учебной ЭВМ

| КОП | Мнемокод | Название | Действие |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 00 | NOP | Пустая операция | Нет |
| 01 | IN | Ввод | Acc <-IR |
| 02 | OUT | Вывод | OR <- Acc |
| 03 | I RET | Возврат из прерывания | FLAGS.PC <- M(SP); INC(SP) |
| 04 | WRRB | Загрузка RB | RB <- CR[ADR] |
| 05 | WRSP | Загрузка SP | SP <- CR[ADR] |
| 06 | PUSH | Поместить в стек | DEC(SP); M(SP) <- R |
| 07 | POP | Извлечь из стека | R -> M(SP); INC(SP) |
| 08 | RET | Возврат | PC -> M(SP); INC(SP) |
| 09 | HLT | Стоп | Конец командных циклов |
| 10 | JMP | Безусловный переход | PC <- CR[ADR] |
| 11 | JZ | Переход, если 0 | if Acc = 0 then PC <- CR[ADR] |
| 12 | JNZ | Переход, если не 0 | if Acc <> 0 then PC <- CR[ADR] |
| 13 | JS | Переход, если отрицательно | if Acc < 0 then PC <- CR[ADR] |
| 14 | JNS | Переход, если положительно | if Acc > 0 then PC <- CR[ADR] |
| 15 | JO | Переход, если переполнение | if \|Acc\| > 99999 then PC <- CR[ADR] |
| 16 | JNO | Переход, если нет переполнения | if \|Acc\| < =99999 then PC <- CR[ADR] |
| 17 | JRNZ | Цикл | DEC(R); if R > 0 then PC <- CR[ADR] |
| 18 | INT | Программное прерывание | DEC(SP); M(SP) <- FLAGS.PC; PC <- M(V) |
| 19 | CALL | Вызов подпрограммы | DEC(SP); M(SP) <- PC; PC <- CR(ADR) |
| 20 | Нет | | |
| 21 | RD | Чтение | Acc <- DD |
| 22 | WR | Запись | M(*) <- Acc |
| 23 | ADD | Сложение | Acc <- Acc + DD |
| 24 | SUB | Вычитание | Acc <- Acc – DD |
| 25 | MUL | Умножение | Acc <- Acc x DD |
| 26 | DIV | Деление | Acc <— Acc/DD |
| 27 | Нет | | |
| 28 | EI | Разрешить прерывание | IF<- 1 |

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| 29 | DI | Запретить прерывание | IF<-0 |
| 30 | MOV | Пересылка | Rl <-R2 |
| 31 | RD | Чтение | Acc <- R* |
| 32 | WR | Запись | R* <- Acc |
| 33 | ADD | Сложение | Acc<-Acc + R* |
| 34 | SUB | Вычитание | Acc <- Acc - R* |
| 35 | MUL | Умножение | Acc <- Acc x R* |
| 36 | DIV | Деление | Acc <- Acc/R* |
| 37 | IN | Ввод | Acc <- ВУ(CR[ADR*]) |
| 38 | OUT | Вывод | ВУ(CR[ADR*]) <- Acc |
| 39 | Нет | | |
| 40 | Нет | | |
| 41 | RDI | Чтение | Acc <-I |
| 42 | Нет | | |
| 43 | ADI | Сложение | Acc <— Acc + I |
| 44 | SBI | Вычитание | Acc <- Acc - I |
| 45 | MULI | Умножение | Acc <- Acc x I |
| 46 | DIVI | Деление | Acc <- Acc/I |

Ввод программы может осуществляться как в машинных кодах непосредственно в память модели, так и в мнемокодах в окно **Текст программы** с последующим ассемблированием.

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.

Практическая часть

В настоящей лабораторной работе будем программировать ЭВМ в машинных кодах.

**Пример.**

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 (табл. 8.7)-

Таблица 8.7 - Команды и коды

| **Последовательность** | **Значения** | | | | |
|---|---|---|---|---|---|
| **Команды** | RD #20 | WR 30 | ADD #5 | WR@30 | JNZ 002 |
| **Коды** | 21 1 020 | 22 0 030 | 23 1 005 | 22 2 030 | 12 0002 |

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) в табл. 8.8

Таблица 8.8 - Содержимое регистров

| **PC** | Ace | M(30) | M(20) | **PC** | **Ace** | M(30) | M(20) |
|---|---|---|---|---|---|---|---|
| **000** | **000000** | **000000** | **000000** | **004** | | | **000025** |
| **001** | **000020** | | | **002** | | | |
| **002** | | **000020** | | **003** | **000030** | | |
| **003** | **000025** | | | **004** | | | **000030** |

**Задание.**

1. Ознакомиться с архитектурой ЭВМ.

2. Записать в ОЗУ "программу", состоящую из пяти команд— варианты задания выбрать из табл. 9.3. Команды разместить в последовательных ячейках памяти.

2. При необходимости установить начальное значение в устройство ввода IR.

2. Определить те программно-доступные объекты ЭВМ, которые будут изменяться при выполнении этих команд.

2. Выполнить в режиме **Шаг** введенную последовательность **команд (в режиме микрокоманд!!!!)**, фиксируя изменения значений объектов, определенных в п. 4.

3. Оформить отчет согласно указанным требованиям к нему.

**Содержание отчета:**

1. Формулировка варианта задания.

1. Машинные коды команд, соответствующих варианту задания.

2. Последовательность микрокоманд для каждой команды.

3. Для каждой команды результаты выполнения последовательности команд в форме таблицы

| Код мик-роко-манды | АЛУ | | РОН | | ОЗУ | | SP | PC | RA | CR | | | RA | M(20) | M(30) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Acc | DR | RDR | RAR | MDR | MAR | | | | COP | TA | ADR | | | |

Таблица 8.9. Варианты задания 1

| № | **IR** | **Команда 1** | **Команда 2** | **Команда 3** | **Команда 4** | **Команда 5** |
|---|---|---|---|---|---|---|
| 1. | **000007** | **IN** | **MUL #2** | WR10 | **WR** 010 | JNS 001 |
| 2. | **X** | **RD #17** | SUB **#9** | **WR16** | **WR** 016 | JNS 001 |
| 3. | **100029** | **IN** | ADD **#16** | **WR8** | WR08 | **JS** 001 |
| 4. | **X** | **RD #2** | **MUL #6** | **WR 11** | **WR 011** | JNZ 00 |
| 5. | **000016** | **IN** | WR8 | **DIV #4** | **WR 08** | JMP 002 |
| 6. | **X** | **RD #4** | **WR 11** | **RD 011** | ADD #330 | **JS** 000 |
| 7. | **000000** | **IN** | **WR9** | **RD @9** | **SUB#1** | **JS** 001 |
| 8. | **X** | **RD 4** | SUB #8 | **WR8** | **WR @8** | JNZ 001 |
| 9. | **100005** | **IN** | **ADD #12** | **WR 10** | **WR @10** | **JS** 004 |
| 10. | **X** | **RD 4** | ADD #15 | **WR 13** | **WR @13** | JMP 001 |
| 11. | **000315** | **IN** | SUB #308 | **WR11** | **WR @11** | JMP 001 |
| 12. | **X** | **RD** #988 | ADD #19 | **WR9** | **WR @9** | JNZ 001 |
| 13. | **000017** | **IN** | WR11 | **ADD 11** | **WR @11** | **JMP 002** |
| 14. | **X** | **RD #5** | MUL #9 | **WR10** | **WR @10** | JNZ 001 |

**Контрольные вопросы:**

1. Что такое система команд ЭВМ;
2. Перечислите регистры процессора;
3. Как проходит выполнение цикла команды в регистрах процессора;
4. Какие классы команд представлены в модели;
5. Какие действия выполняют команды передачи управления
6. Какие способы адресации операндов применяются в командах ЭВМ

# ПРАКТИЧЕСКАЯ РАБОТА №6

## ПРОГРАММИРОВАНИЕ АРИФМЕТИЧЕСКИХ И ЛОГИЧЕСКИХ КОМАНД

**Цель:** приобрести практические навыки программирования арифметических и логических команд.

**Сформировать следующие умения:** У2. правильно эксплуатировать и устранять типичные выявленные дефекты технических средств информатизации;

**Теоретическая часть:**

Несмотря на то, что современные языки программирования высокого уровня обеспечивают не только удобное, но и эффективное системное программирование, в тех случаях, когда особенно важно получить оптимальный объектный код, необходимо использовать Ассемблер.

Исходный модуль программы на Ассемблере представляет собой последовательность строк, имеющих следующий формат:

[метка] команда [операнд(ы)] [комментарий]

После метки ставится двоеточие, команда и операнд разделяются по крайней мере одним пробелом; если операндов несколько, то они разделяются запятыми; перед комментарием ставится точка с запятой. В квадратные скобки заключены необязательные компоненты.

Ассемблер имеет ряд операторов, которые позволяют управлять процессом ассемблирования и формирования листинга. Эти операторы называются псевдокомандами или директивами. Они действуют только в процессе ассемблирования программы и не генерируют машинных кодов.

Любые ассемблерные программы содержат по крайней мере один сегмент-сегмент кода. В некоторых программах используются сегмент для стековой памяти и сегмент данных для определения данных. Сегмент описывается директивой SEGMENT следующим образом:

имя SEGMENT [параметры]

имя ENDS

Сегмент кода содержит одну или несколько процедур, определяемых директивой PROC:

имя сегмента SEGMENT имя процедуры PROC

RET имя процедуры ENDP имя сегмента ENDS

Пример простой исходной программы на Ассемблере представлен ниже:

TITLE PROG1 ПРИМЕР РЕГИСТРОВЫХ ОПЕРАЦИЙ;

STACKSG SEGMENT PARA STACK 'Stack'

DB 12 DUP(?) STACKSG ENDS

CODESG SEGMENT PARA 'Code' BEGIN PROC FAR ASSUME SS: STACKSG, CS:CODESG, DS:NOTHING

PUSH DS; сохранение DS в стеке. (1)

SUB AX, AX; формирование нуля в AX. (2)

PUSH AX; сохранение AX в стеке. (3)

MOV AX,0123H; передача константы в регистр. (4)

ADD AX,0025H; сложение регистра с константой. (5) MOV BX, AX; передача из регистра в регистр. (6)

ADD BX, AX; сложение содержимого регистров. (7) MOV CX, BX; передача из регистра в регистр. (8)

SUB CX, AX; вычитание содержимого регистров (9) SUB AX, AX; формирование нуля в AX. (10)

NOP; нет операции (задержка). (11)

RET; возврат из процедуры. (12)

BEGIN ENDP; КОНЕЦ ПРОЦЕДУРЫ.

CODESG ENDS; КОНЕЦ СЕГМЕНТА.

END BEGIN; КОНЕЦ ПРОГРАММЫ.

В приведенном примере директива ASSUME сообщает Ассемблеру соответствие между сегментными регистрами и именами сегментов.

Исходный текст программы на Ассемблере создается с использованием любого доступного текстового редактора и сохраняется на магнитном диске под именем *.ASM или *. TXT. Компилятор Турбо Ассемблер является программой в виде загрузочного файла TASM.EXE, работающей в режиме командной строки. Он вызывается командой

TASM имя, где имя является именем исходного файла. Предполагается, что TASM и исходный файл сохраняются в одном каталоге. При вызове компилятора могут задаваться режимы его работы. Например, запись в командной строке:

TASM/ZI PROG1 PROG1 PROG1

приводит к созданию из исходного файла PROG1.ASM объектного модуля PROG1.OBJ и файла листинга PROG1.LST. Параметр /ZI приводит к внесению в файл листинга полных сведений о номерах строк и именах исходного модуля.

С целью создания исполнительной программы необходимо произвести компоновку объектных модулей с библиотечными модулями. Компоновщик является программой с именем TLINK.EXE. Он может быть вызван командой

TLINK имя объектного модуля

После компоновки создается загрузочный файл PROG1.EXE, а также

файл карты сборки PROG1.MAP.

Компоновка в режиме TLINK /t приводит к созданию исполняемого *.COM файла, а компоновка в режиме TLINK /v включает в файл исполняемой программы информацию, необходимую для ее инициализации и отладки.

**Практическая часть:**

1. Изучить методические указания.

2. Используя текстовый редактор, создайте исходный модуль программы HELLO.ASM:

```
Data SEGMENT
Greet DB 'Привет, Маша!',13,10,'$' Data ENDS
Stack SEGMENT Stack
DB 100H DUP(?); стек размером 256 байтов.
Stack ENDS
ASSUME CS:Code, DS:Data, SS:Stack
Code SEGMENT Start:
загрузка номера сегмента (1)
в регистр DS. (2)
загрузка в DX адреса симв. строки. (3)
задание функции вывода строки. (4)
вывод строки. (5)
завершение программы через (6)
систем ную функцию возврата (7)
в диспетчер MS DOS. (8)
MOV AX, DATA
MOV DS, AX
MOV DX, OFFSET Greet
MOV AH,9
INT 21H
MOV AL,0
MOV AH,4CH
INT 21H Code ENDS END Start; адрес начала программы.
```

3. Используя, компилятор Турбо Ассемблер, создайте файлы HELLO.OBJ и HELLO.LST. Выведите на экран текст HELLO.LST и проанализируйте его.

4. Используя, компоновщик TLINK, создайте файлы HELLO.EXE и HELLO.MAP.

5. Выведите на экран файл HELLO.MAP и ознакомьтесь с ним.

6. Убедитесь в работоспособности программы HELLO.

7. Проанализировать приведенную ниже программу HELLO.ASM и вставить в нее фрагмент, реализующий вывод на экран Вашей фамилии, имени, отчества с помощью символьной переменной STUDENT

8. Подготовить ответы на контрольные вопросы.

**Теоретическая часть:**

Если программа скомпонована в режиме /v, то после ее загрузки отладчиком, открывается окно Module. Символ стрелка показывает на подлежащую исполнению команду. Клавишей F2 можно расставлять и снимать ловушки в той строке, где расположен курсор. Окно Inspect можно открыть из локального меню окна Module (alt-F10). При этом отладчик запрашивает имя подлежащих контролю переменной или регистра. Контролировать состояния переменных можно также в окнах Variables и Watches, вызываемых из пункта View главного меню.

Окно переменных Variables позволяет наблюдать все переменные, доступные в месте останова программы. В локальном окне пункт Inspect дает доступ к полной информации о типе, значении и адресе хранения выделенной переменной. Отдельные переменные программист может задать для анализа в окне Watches. Для помещения переменной в это окно следует подвести курсор к идентификатору переменной и нажать Ctrl+W. Для анализа выражения его следует выделить (Ins+ стрелки), а затем нажать Ctrl+W. Используя локальное меню, можно исследовать внутреннюю структуру переменных. Положение и размер окон можно изменять, пользуясь курсорными клавишами при включенном режиме Scroll Lock (Размер регулируется при нажатой клавише Ctrl).

**Практическая часть:**

1. Изучить методические указания.

2. Подготовить ответы на контрольные вопросы.

3. Введите программу PROG4 с фрагментом из домашнего задания, используя текстовый редактор.

4. Оттранслируйте и скомпонуйте программу в режимах TASM/ZI, TLINK/V.

5. Загрузите отладчик и программу. Проведите ее трассировку. Целесообразно выполнять работу по частям, составляя и отлаживая законченные по смыслу фрагменты.

6. Наблюдайте результаты выполнения команд, используя окна Variables, Watches и/или Inspecting, открываемые из пункта View Главного меню. Сравнивайте полученные результаты с ожидаемыми. Для наблюдения состояния флажков используйте окно Registers.

7. Проанализировать приведенную ниже программу PROG4, написать комментарии ко всем командам сегмента кода, указав при этом тип используемых операндов и способ адресации.

| № вар. | операция | 1-ый операнд | 2-ой операнд |
|---|---|---|---|
| 1 | сложение | регистр | константа байт |
| 2 | сложение | регистр | переменная в памяти |
| 3 | умножение | регистр | регистр |
| 4 | сложение | переменная в памяти | константа байт |
| 5 | сложение | переменная в памяти | константа слово |
| 6 | умножение | регистр | переменная в памяти |
| 7 | вычитание | регистр | константа слово |
| 8 | сложение | переменная в памяти | константа байт |

| № вар. | операция | 1-ый операнд | 2-ой операнд |
|---|---|---|---|
| 9 | вычитание | регистр | константа байт |
| 10 | сложение | переменная в памяти | регистр |
| 11 | умножение | переменная в памяти | константа слово |
| 12 | вычитание | ЯП- косвенная адресация | регистр |
| 13 | вычитание | регистр | переменная в памяти |
| 14 | вычитание | переменная в памяти | константа слово |
| 15 | вычитание | переменная в памяти | константа байт |

8. Используя в качестве операндов переменные, определенные в сегменте данных, в соответствии с вариантом задания (Таб 4.1.) написать команды для выполнения указанных в таблице операций, вставить их в соответствующий фрагмент программы PROG4.

Таблица 4.1.

Пример программы

```
TITLE PROG4 АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ;
DATASG SEGMENT PARA
X0 DB 64H
X1 DB 40H
X2 DB 16H
X3 DW 4000H
X4 DW 2000H
X5 DW 1000H
Z1 DD 0123BC62H
Z2 DD 0012553AH REZ DD ? DATASG ENDS
STACKSG SEGMENT 'Stack'
DB 12 DUP(?) STACKSG ENDS
CODESG SEGMENT PARA 'Code'
BEGIN PROC FAR
ASSUME SS: STACKSG, CS: CODESG, DS: DATASG
PUSH DS SUB AX, AX корректное
PUSH AX MOV AX, DATASG завершение
MOV DS, AX программы. Инициализация сегментного регистра DS.
ПРИМЕРЫ СЛОЖЕНИЯ
MOV AL, X0
MOV BL, X1
ADD AL, BL
ADD AL, X2
ADD X0, BL
ADD BL,
10H ADD X0,25H
ПРИМЕРЫ ВЫЧИТАНИЯ
MOV AX, X3
MOV BX, X4
SUB AX, BX
```

SUB AX, X5
SUB X3, BX
SUB BX,1000H
SUB X3,256H;

П РИМЕР СЛОЖЕНИЯ ДВОЙНЫХ СЛОВ

LEA SI, Z1; адрес первого двойного слова.
LEA DI, Z2; адрес второго двойного слова
LEA BX, REZ.; адрес результата.
MOV AX, [SI]; загрузка младшего слова Z1.
ADD AX, [DI]; сложение младших слов Z1 и Z2.
MOV WORD PTR [BX], AX сохранение младшего слова результата.
INC SI формирование адреса старшего слова Z1
INC SI формирование адреса старшего слова Z2.
INC DI загрузка старшего слова Z1.
INC DI сложение старших слов Z1 и Z2
MOV AX, [SI] и возможного переноса.
ADC AX, [DI] сохранение старшего слова результата.;
MOV WORD PTR [BX+2], AX

.ПРИМЕР УМНОЖЕНИЯ MUL

MOV AL, X0
MUL X1
MOV AX, X3
MUL X5
MOV CX, 20H
MUL CX
ПРИМЕР УМНОЖЕНИЯ IMUL
M OV AX, X3
IMUL X3
MOV CX, -5
IMUL CX
RET
BEGIN END; КОНЕЦ ПРОЦЕДУРЫ;
CODESG ENDS; КОНЕЦ СЕГМЕНТА;
END BEGIN; КОНЕЦ ПРОГРАММЫ

**Контрольные вопросы:**

1. Какова цель сегментации памяти;
2. Что такое базовый адрес сегмента;
3. Какие значения может принимать базовый адрес сегмента;
4. Каков максимальный размер сегмента и почему;
5. Какие типы сегментов использует ассемблерная программа
6. Какими директивами описывается сегмент

# ПРАКТИЧЕСКАЯ РАБОТА №7

## ПРОГРАММИРОВАНИЕ ПЕРЕХОДОВ, ВВОДА-ВЫВОДА

**Цель:** приобрести практические навыки программирования переходов, ввода-вывода.

**Сформировать следующие умения: У1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель работы:** изучить функции ввода с клавиатуры и вывода на экран

**Средства:** Turbo Assembler, Turbo Debuger

**Краткие теоретические сведения:**

**Функции BIOS и DOS ввода с клавиатуры**

*Функция DOS 0Ah* — Считать строку символов из STDIN в буфер

| | |
|---|---|
| Ввод: | AH = 0Ah DS: DX = адрес буфера |
| Вывод: | Буфер содержит введенную строку |

Для вызова этой функции надо подготовить буфер, первый байт которого содержит максимальное число символов для ввода (1 – 254), а содержимое, если оно задано, может использоваться как подсказка для ввода. При наборе строки обрабатываются клавиши Esc, F3, F5, BS, Ctrl-C/Ctrl-Break и т.д., как при наборе команд DOS (то есть Esc начинает ввод сначала, F3 восстанавливает подсказку для ввода, F5 запоминает текущую строку как подсказку, Backspace стирает предыдущий символ). После нажатия клавиши Enter строка (включая последний символ CR (0Dh)) записывается в буфер, начиная с третьего байта. Во второй байт записывается длина реально введенной строки без учета последнего CR.

*Функция DOS 07h* — Считать символ из STDIN без эха, с ожиданием и без проверки на Ctrl-Break

| | |
|---|---|
| Ввод: | AH = 07h |
| Вывод: | AL = код символа |

*Функция DOS 06h* — Считать символ из STDIN без эха, без ожидания и без проверки на Ctrl-Break

| | |
|---|---|
| Ввод: | AH = 07h DL = 0FFh |
| Вывод: | ZF = 1, если не была нажата клавиша, и AL = 00 ZF = 0, если клавиша была нажата. В этом случае AL = код символа |

*Функция DOS 0Ch* — Очистить буфер и считать символ

| | |
|---|---|
| Ввод: | AH = 0Ch AL = Номер функции DOS (01, 06, 07, 08, 0Ah) |
| Вывод: | Зависит от вызванной функции |

Функция 0Ch очищает буфер клавиатуры, так что следующая функция чтения символа будет ждать ввода с клавиатуры, а не использовать нажатый ранее и еще не обработанный символ. Например, именно эта функция используется для считывания ответа на вопрос «Уверен ли пользователь в том, что он хочет отформатировать диск?».

**Функции BIOS и DOS вывода на экран**

*Функция DOS 02h* — Записать символ в STDOUT с проверкой на Ctrl-Break

| | |
|---|---|
| Ввод: | AH = 02h DL = ASCII-код символа |
| Вывод: | Никакого, согласно документации, но на самом деле: AL = код последнего записанного символа (равен DL, кроме случая, когда DL = 09h (табуляция), тогда в AL возвращается 20h). |

*Функция DOS 06h* — Записать символ в STDOUT без проверки на Ctrl-Break

| | |
|---|---|
| Ввод: | AH = 06h DL = ASCII-код символа (кроме FFh) |
| Вывод: | Никакого, согласно документации, но на самом деле: AL = код записанного символа (копия DL) |

Эта функция не обрабатывает управляющие символы (CR, LF, HT и BS выполняют свои функции при выводе на экран, но сохраняются при перенаправлении вывода в файл) и не проверяет нажатие Ctrl-Break.

*Функция DOS 09h* — Записать строку в STDOUT с проверкой на Ctrl-Break

| | |
|---|---|
| Ввод: | AH = 09h DS:DX = адрес строки, заканчивающейся символом $ (24h) |
| Вывод: | Никакого, согласно документации, но на самом деле: AL = 24h (код последнего символа) |

Действие этой функции полностью аналогично действию функции 02h, но выводится не один символ, а целая строка.

*Функция DOS 40h* — Записать в файл или устройство

| | |
|---|---|
| Ввод: | AH = 40h BX = 1 для STDOUT или 2 для STDERR DS:DX = адрес начала строки CX = длина строки |
| Вывод: | CF = 0, AX = число записанных байт |

Эта функция предназначена для записи в файл, но, если в регистр BX поместить число 1, функция 40h будет выводить данные на STDOUT, а если BX = 2 — на устройство STDERR. STDERR всегда выводит данные на экран и не перенаправляется в файлы. *INT 29h*: Быстрый вывод символа на экран

| | |
|---|---|
| Ввод: | AL = ASCII-код символа |

В большинстве случаев INT 29h просто немедленно вызывает функцию BIOS «вывод символа на экран в режиме телетайпа», так что никаких преимуществ, кроме экономии байт при написании как можно более коротких программ, она не имеет.

*Функция BIOS INT 10h, AH = 03* — Считать положение и размер курсора

| | |
|---|---|
| Ввод: | AH = 03 BH = номер страницы |
| Вывод: | DH, DL = строка и столбец текущей позиции курсора CH, CL = первая и последняя строки курсора |

Возвращает текущее состояние курсора на выбранной странице (каждая страница использует собственный независимый курсор).

*Функция BIOS INT 10h, AH = 0Eh* — Вывести символ в режиме телетайпа

| | |
|---|---|
| Ввод: | AH = 0Eh BH = номер страницы AL = ASCII-код символа |

Символы CR (0Dh), LF (0Ah), BEL (7) интерпретируются как управляющие символы. Если текст при записи выходит за пределы нижней строки, экран прокручивается вверх. В качестве атрибута используется атрибут символа, находившегося в этой позиции.

*Функция BIOS INT 10h, AH = 13h* — Вывести строку символов с заданными атрибутами

| | |
|---|---|
| Ввод: | AH = 13h AL = режим вывода:<br>бит 0 — переместить курсор в конец строки после вывода бит 1 — строка содержит не только символы, но также и атрибуты, так что каждый символ описывается двумя байтами: ASCII-код и атрибут биты 2 – 7 зарезервированы<br>CX = длина строки (только число символов) BL = атрибут, если строка содержит только символы DH,DL = строка и столбец, начиная с которых будет выводиться строки ES:BP = адрес начала строки в памяти |

Функция 13h выводит на экран строку символов, интерпретируя управляющие символы CR (0Dh), LF (0Ah), BS (08) и BEL (07). Если строка подготовлена в формате символ, атрибут — гораздо быстрее просто скопировать ее в видеопамять, о чем рассказано в следующем разделе.

**Задание:** вывести на середину экрана слово, введенное с клавиатуры.

**Порядок выполнения работы:**

1. Написать PASCAL-программу и программу на ассемблере (их изобразить так, чтобы все конструкции программ соответствовали друг другу).
2. Написать команды для ассемблирования, компоновки и запуска отладчика с отлаживаемой программой.
3. Заполнить таблицу с ошибками:

| № | Допущенная ошибка | Сообщение об ошибке |
|---|---|---|
| | | |

**Контрольные вопросы:**

1. Как задается функция для выполнения прерывания;
2. Куда вводятся символы при нажатии клавиши;
3. Что надо выполнить для вывода символа;
4. Что надо выполнить для ввода символа.

# ПРАКТИЧЕСКАЯ РАБОТА № 8

## ФОРМАТИРОВАНИЕ МАГНИТНЫХ ДИСКОВ. ЗАПИСЬ ИНФОРМАЦИИ НА ОПТИЧЕСКИЕ НОСИТЕЛИ

**Цель:** приобрести практические навыки в форматировании магнитных дисков. Записи информации на оптические носители; Получение навыков при работе со звуком.

**Сформировать следующие умения: У 2.** правильно эксплуатировать и устранять типичные выявленные дефекты технических средств информатизации;

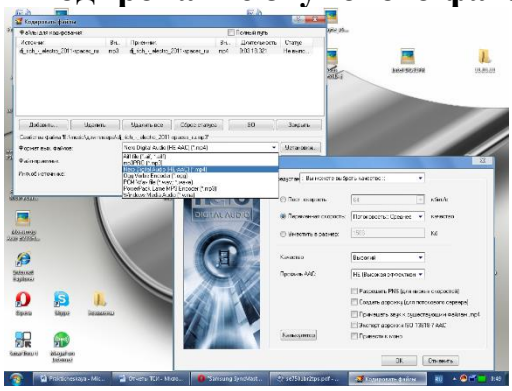
**Цель:** Изучение устройства жесткого и гибкого дисков, а также программ по обслуживанию жесткого диска.

**Ход работы:**

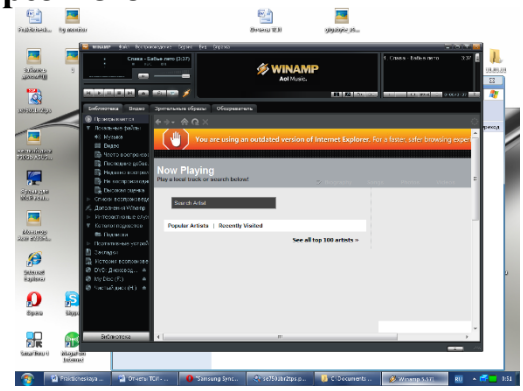
Технические характеристики жесткого диска **Asus Lamborghini External HDD 750GB Black;** Кодирование звукового файла через **Nero**

| | |
|---|---|
| Внутренний артикул | 158361 |
| Линейка | Lamborghini External HDD |
| Применение | внешний |
| Форм-фактор | 2.5" |
| Объем | 750 Гб |
| Подключение | USB |
| Внешняя скорость передачи данных | 600 Мб/с |

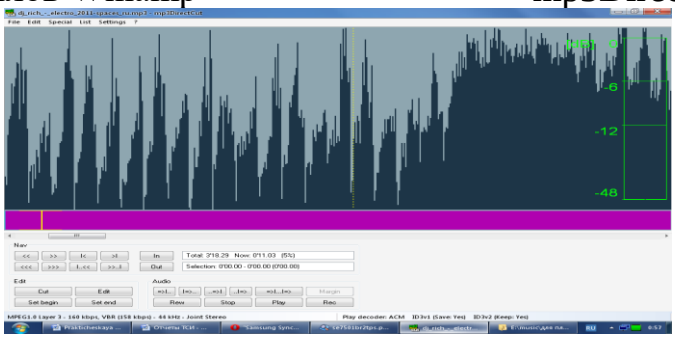
**Кодирование звукового файла через Nero**



Программа воспроизведения звуковых файлов Winamp



Редактор звуковых файлов mp3DirectCut



**Контрольные вопросы:**

1. Какие виды накопителей вы знаете?
2. Из каких основных элементов состоит дисковод для гибких магнитных дисков?
3. Назовите основные конструктивные элементы накопителя на жестких дисках?
4. Какова роль звука в пользовании компьютером?
5. Каким параметром определяется звук?
6. Какие основные функции выполняет звуковая система персонального компьютера?
7. Какие основные компоненты входят в состав звуковой системы персонального компьютера?

# ПРАКТИЧЕСКАЯ РАБОТА № 9

## РАБОТА ПО ПОДКЛЮЧЕНИЮ АКУСТИЧЕСКИХ СИСТЕМ И С ПРОГРАММАМИ ОБЕСПЕЧЕНИЯ ЗАПИСИ И ВОСПРОИЗВЕДЕНИЯ ЗВУКОВЫХ ФАЙЛОВ

**Цель:** приобрести практические навыки в правильной эксплуатации и устранении типичных выявленных дефектов технических средств информатизации.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель работы.**

1. Ознакомиться и получить навыки работы со специализированными программными и техническими средствами получения оцифрованной звуковой информации.

2. Ознакомиться и получить навыки работы со специализированными программными средствами перекодирования звуковой информации.

**Задание.**

1. Оборудовать рабочее место специализированными техническими и программными средствами для получения оцифрованной аудиоинформации и её перекодирования в другие форматы. Изучить справочные руководства по использованию программ записи и перекодирования звука.

2. Произвести запись звукового фрагмента длительностью 4–6 мин с максимальным качеством в формат WAV. Определить размер полученного файла звукозаписи.

3. Полученный файл звукозаписи перекодировать в формат MP3 с использованием различных значений ширины потока (32, 64, 96, 128, 160, 192, 224, 256, 320 кбит/с). Для каждого случая рассчитать размер целевого MP3-файла и сравнить его с фактически полученным значением. Сделать выводы по результатам сравнения. Указать причины несоответствия расчётных и фактических данных.

**Теоретическая часть:**

1. Цифровое представление аналоговой звуковой информации. Принципы сжатия звуковой информации (психоакустическая модель кодирования).

2. Краткая справка о форматах звуковых файлов, использованных в работе.

3. Описание программных и технических средств, используемых в работе.

4. Порядок подготовки рабочего места к работе.

5. Формула для расчёта размера MP3-файла.

**Практическая часть:**

1. Ход работы.

2. Схематичное изображение рабочего места с указанием использованных технических средств.

3. Протокол работы. Промежуточные выводы.

4. Графические иллюстрации полученных результатов:

– зависимости расчётного и фактического размеров MP3-файла от ширины потока в единой координатной системе;

– зависимость абсолютной ошибки расчёта от ширины потока;

– зависимость относительной ошибки расчёта от ширины потока;

– зависимость эффективности сжатия от ширины потока.

Описание к иллюстрациям и промежуточные выводы.

Протокол работы

Длительность звукозаписи ($\tau$): ___ мин ___ с (_____ с)

Размер wav-файла ($M_w$): _____ байт.

| №№ п./п. | Ширина потока ($b$), кбит/с | Расчётный размер ($M_i^p$), байт | Фактический размер ($M_i^\phi$), байт | Абсолютная ошибка $\Delta M_i=\lvert M_i^\phi - M_i^p \rvert$ | Относительная ошибка $\Delta M_i\%=(\Delta M_i/M_i^\phi)*100\%$ | Эффективность сжатия $\eta_i\%=(1-M_i^\phi/M_w)*100\%$ |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 32 | | | | | |
| 2 | 64 | | | | | |
| 3 | 96 | | | | | |
| 4 | 128 | | | | | |
| 5 | 160 | | | | | |
| 6 | 192 | | | | | |
| 7 | 224 | | | | | |
| 8 | 256 | | | | | |
| 9 | 320 | | | | | |

**Контрольные вопросы:**

1. Принципы сжатия звуковой информации;
2. Краткая справка о форматах звуковых файлов;
3. Цифровое представление аналоговой звуковой информации;
4. Формула для расчёта размера MP3-файла;

# ПРАКТИЧЕСКАЯ РАБОТА № 10

## РАБОТА С НАСТРОЙКОЙ СКАНЕРОВ И ПРОГРАММАМИ ПО СКАНИРОВАНИЮ

**Цель:** приобрести практические навыки в работе с настройкой сканеров и программами по сканированию.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель работы:** изучить устройство сканера, интерфейс связи с ПЭВМ, функции сканера, подключение и настройку сканера к работе.

Содержание работы:

6.1 Изучение устройств сканера.

6.1.1 Матрица сканера.

6.1.2 Корпус сканера.

6.1.3 Блок управления.

6.1.4 Источник света сканера.

6.1.5 Работа АЦП.

6.1.6 Процессор сканера.

6.1.7 Контроллер интерфейса.

6.1.8 Протяжный механизм сканера.

6.1.9 Блок питания сканера.

6.2 Задание.

6.3 Контрольные вопросы.

6.4 Содержание отчета.

6.5 Список использованных источников.

6.1. Изучение устройств сканера

6.1.1 Матрица сканера

Именно матрица является важнейшей частью любого сканера. Матрица трансформирует изменения цвета и яркости принимаемого светового потока в аналоговые электрические сигналы, которые будут понятны лишь единственному ее электронному другу – аналого-цифровому преобразователю (АЦП). Только он понимает матрицу, ведь никакие процессоры или контроллеры не разберут ее аналоговые сигналы без предварительного толкования преобразователем. Световой поток, падая на поверхность матрици, буквально "вышибает" электроны из ее чувствительных ячеек. И чем ярче свет, тем больше электронов окажется в накопителях матрицы, тем больше будет их сила, когда они непрерывным потоком ринутся к выходу. Однако сила тока электронов настолько несоизмеримо мала, что вряд ли их "услышит" даже самый чувствительный АЦП. Именно поэтому на выходе из матрицы их ждет усилитель, который сравним с огромным рупором, превращающим, образно говоря, даже комариный писк в вой громогласной сирены. Усиленный сигнал (пока еще аналоговый) "взвесит" преобразователь, и присвоит каждому электрону цифровое значение, согласно его силе тока. Дальше электроны будут представлять собой цифровую информацию, обработкой которой займутся другие специалисты. Работа над воссозданием изображения больше не требует помощи матрицы.

Большинство современных сканеров для дома и офиса базируются на матрицах двух типов: на CCD (Charge Coupled Device) или на CIS (Contact Image Sensor). Сей факт порождает в умах пользователей два вопроса: в чем разница и что лучше? Если разница заметна даже невооруженным взглядом – корпус CIS-сканера плоский, в сравнении с аналогичным CCD-аппаратом (его высота обычно составляет порядка 40-50 мм), то ответить на второй вопрос гораздо сложнее.

Для начала рассмотрим основные достоинства и недостатки этих двух классов сканеров.

Сканер – это устройство для перевода изображения на бумажном или ином носителе в электронный вид. Широко используются CCDиCIS– сканеры.

**Достоинства и недостатки CCD- и CIS-сканеров**

| | Достоинства | Недостатки |
|---|---|---|
| CCD | Высокая разрешающая способность (недорогие CCD-сканеры имеют на сегодняшний день разрешения до 2400 dpi);<br>Долгий срок службы лампы;<br>Высокое качество сканирования;<br>Большая глубина резкости;<br>Возможность работы со слайд-адаптерами и автоподатчиками документов | Сравнительно высокая стоимость (по отношению к CIS-сканерам);<br>Длительный прогрев лампы перед сканированием;<br>Необходимость в дополнительном источнике питания |
| CIS | Небольшие габариты;<br>Быстрый старт;<br>Невысокая стоимость;<br>Низкое энергопотребление (многие CIS-сканеры получают питание по USB);<br>Автономность | Ограниченное разрешение (до 1200 dpi);<br>Небольшая глубина резкости;<br>Чувствительность к боковой засветке;<br>Сравнительно низкое качество сканирования |

Рис. 6.1 – Достоинства и недостатки CCD- иCIS– сканеров

CCD-сканер обладает большей глубиной резкости, нежели его CIS-собрат. Достигается это за счет применения в его конструкции объектива и системы зеркал.

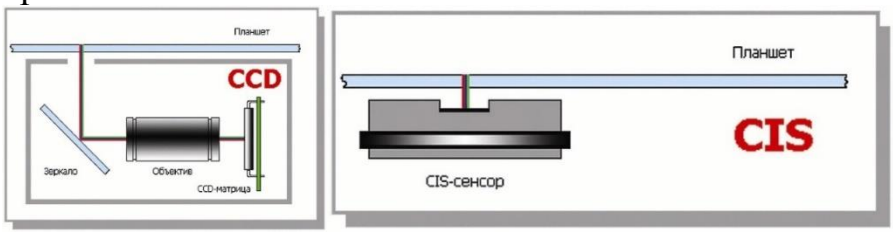



Рис. 6.2.– Объектив и системы зеркал сканера

Сканеры с CCD-матрицей распространены гораздо больше, чем CIS-аппараты. Объяснить это можно тем, что сканеры в большинстве случаев приобретаются не только для оцифровки листовых текстовых документов, но и для сканирования фотографий и цветных изображений. В этом плане, электронный вид изображения с наиболее точной и достоверной цветопередачей, лучше по светочувствительности CCD-сканер гораздо строже передает цветовые оттенки, света и полутона, нежели CIS-сканер. Погрешность разброса уровней цветовых оттенков, различаемых стандартными CCD-сканерами составляет порядка ±20%, тогда как у CIS-аппаратов эта погрешность составляет уже ±40%.

CIS-матрица состоит из светодиодной линейки, которая освещает поверхность сканируемого оригинала, самофокусирующихся микролинз и непосредственно самих сенсоров. Конструкция матрицы очень компактна, поэтому, сканер, в котором используется контактный сенсор, всегда будет намного тоньше CCD-сканера.

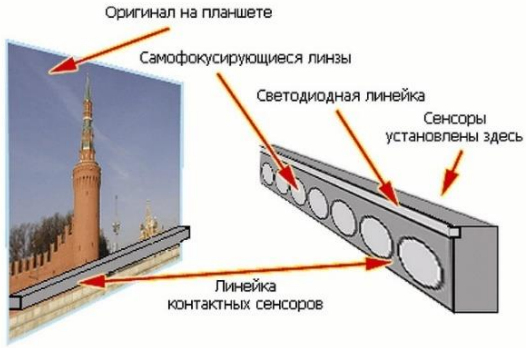



Рис. 6.3. Схематическое представление CIS-сенсора

К тому же, такие аппараты имеют низкое энергопотребление; они практически нечувствительны к механическим воздействиям. Однако CIS-сканеры несколько ограничены в применении: аппараты, как правило, не приспособлены к работе со слайд-модулями и автоподатчиками документов.

Из-за особенностей технологии CIS-матрица обладает сравнительно

небольшой глубиной резкости. Для сравнения, у CCD-сканеров глубина резкости составляет ±30 мм, у CIS – ±3 мм. Поэтому при сканировании толстой книги, получаем скан с размытой полосой посередине, т.е. в том месте, где оригинал не соприкасается со стеклом. У CCD-аппарата вся картина будет резкой, поскольку в его конструкции есть система зеркал и фокусирующая линза. В свою очередь, именно достаточно громоздкая оптическая система и не позволяет CCD-сканеру достичь столь же компактных размеров, как у CIS-собрата. Однако, с другой стороны, именно оптика обеспечивает очевидный выигрыш в качестве сканирования.

В плане разрешающей способности CIS-сканеры также слабее CCD - сканера. Уже сейчас некоторые модели CCD-сканеров для дома и офиса обладают оптическим разрешением порядка 3200 dpi, тогда как у CIS-аппаратов оптическое разрешение ограничено, пока что 1200 dpi. Сканеры с CIS-матрицей нашли свое применение там, где требуется оцифровывать не книги, а листовые оригиналы. Тот факт, что эти сканеры целиком получают питание по шине USB и не нуждаются в дополнительном источнике питания, пришелся как нельзя кстати владельцам портативных компьютеров. Оцифровать оригинал и перевести его в текстовый файл они могут где бы то ни было, не завязываясь с близостью электрических сетей, что позволяет закрыть глаза на ряд недостатков контактного сенсора.

Матрица CCD - сканера представляет собой "большую микросхему" со стеклянным окошком (рис. 6.4).

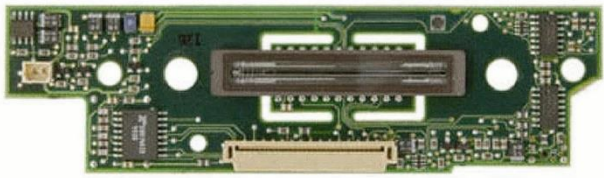


Рис. 6.4. CCD-матрица сканера

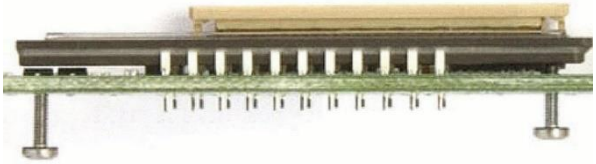


Рис. 6.5. Вид сбоку на CCD-матрицу

Именно сюда и фокусируется отраженный от оригинала свет. Матрица не прекращает работать все то время, пока лафет со сканирующей кареткой, приводимый шаговым электродвигателем, совершает путь от начала планшета, до его конца. Общая дистанция движения лафета по направлению "Y" называется частотой сэмплирования или механическим разрешением сканера. За один шаг матрица целиком захватывает горизонтальную линию планшета, которая называется линией растра. По истечении времени, достаточного для обработки одной такой линии, лафет сканирующего блока перемещается на небольшой шаг, и наступает очередь для сканирования следующей линии, и т.д.

На виде сбоку (рисунке 6.5) можно заметить два обычных винта, с помощью которых производится точная юстировка матрицы, чтобы падающий на нее отраженный свет от зеркал ложился бы равномерно по всей ее поверхности.

В случае перекоса одного из элементов оптической системы воссозданное компьютером изображение окажется "полосатым".

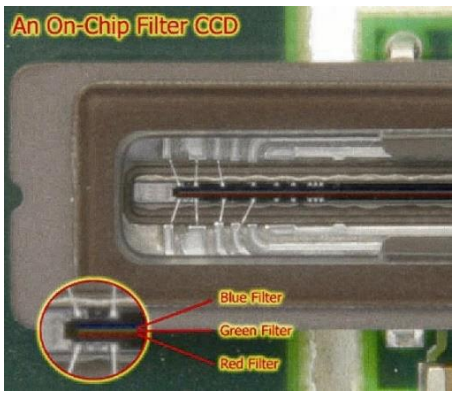



Рис. 6.6. Увеличенное изображние части CCD-матрицы

Рис. 6.7. Объектив сканера

На увеличенном рисунке CCD-матрицы видно, что CCD-матрица оснащена собственным RGB-фильтром. Именно он и представляет собой главный элемент системы разделения цветов. Обычно, многие обозреватели ограничиваются стандартной формулировкой: "стандартный планшетный сканер использует источник света, систему разделения цветов и прибор с зарядовой связью (CCD) для сбора оптической информации о сканируемом объекте". На самом деле, свет можно разделить на его цветовые составляющие, а затем сфокусировать на фильтрах матрицы. Столь же немаловажным элементом системы разделения цветов является объектив сканера.

6.1.2 Корпус сканера

Корпус сканера должен обладать достаточной жесткостью, чтобы исключить возможные перекосы конструкции. Безусловно, лучше всего, если основа сканера представляет собой металлическое шасси. Однако корпуса большинства выпускаемых сегодня сканеров для дома и офиса, в целях снижения стоимости, полностью сделаны из пластмассы.

Рис. 6.8. Расположение основных функциональных узлов сканера

В этом случае, необходимую прочность конструкции придают ребра жесткости.

Немаловажным элементом корпуса является транспортный фиксатор, наличие которого призвано уберечь сканирующую каретку от повреждений при транспортировке сканера. Необходимо помнить, что перед включением любого сканера, оснащенного таким фиксатором, нужно осуществить его разблокировку. В противном случае, можно повредить механизмы аппарата. Так как оптическая система сканера не терпит пыли, поэтому корпус аппарата должен быть герметичным, без каких-либо щелей (даже технологических).

Такое свойство аппарата как возможность отделения крышки планшета особенно полезно при сканировании таких оригиналов, как толстые книги или журналы.

Края планшета должны иметь пологий спуск – это облегчает задачу по быстрому извлечению оригинала со стекла. Кроме того, между стеклом и планшетом не должно быть никакого зазора, который препятствовал бы извлечению оригинала.

6.1.3 Блок управления

Все сканеры управляются с персонального компьютера, к которому они подключены, а необходимые настройки перед сканированием задаются в пользовательском окне управляющей программы. По этой причине, сканерам для дома и офиса совсем не обязательно иметь собственный блок управления. Однако многие производители устанавливают (обычно на лицевую панель) несколько кнопок "быстрого сканирования".

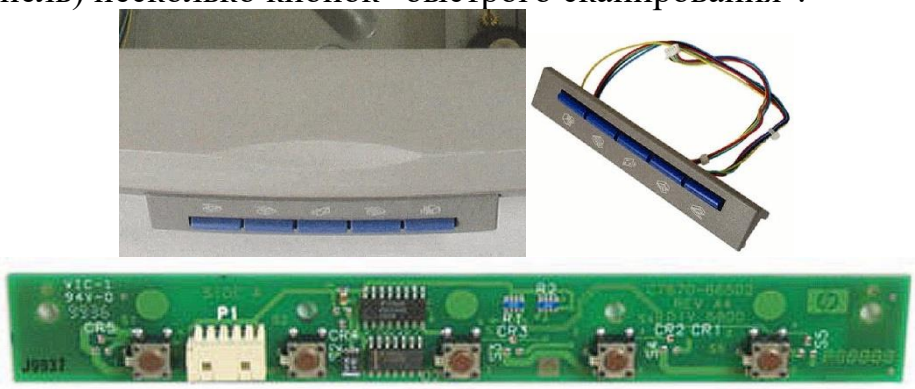



Рис. 6.9. Кнопки быстрого сканирования

На рисунке 6.9 видно, что каждой кнопке соответствует определенный значок. Типовые функции быстрого старта обычно подразумевают запуск стандартной операции сканирования, с выводом на принтер, с последующей отправкой по электронной почте, по факсу и т.п. Понятно, что для той или иной кнопки заданы конкретные параметры качества сканирования. Впрочем, нажатие на ту или иную кнопку сначала приводит к запуску на компьютере приложения (если таковых несколько), отвечающего за вызываемую операцию.

6.1.4 Источник света сканера

Абсолютно в каждом сканере используется свой осветитель. Так называется небольшой и мощный модуль, в задачу которого входит включение и выключение лампы сканера (или того, что эту лампу заменяет). В CIS-сканерах в качестве источников света применяют светодиодную линейку, за счет чего данный класс аппаратов потребляет так мало энергии.

В CCD-сканерах оригиналы стандартно освещает люминесцентная лампа с холодным катодом. Ее свет в тысячи раз ярче светодиодов. Но для того чтобы вызвать свечение газа внутри лампы нужно подать на ее вход очень высокое напряжение. Его вырабатывает отдельный блок, называемый инвертором.

Инвертор преобразует постоянный ток в переменный, а затем

повышает напряжение с 5 Вольт до нескольких киловольт.

Различают три главных вида ламп, использующихся в сканерах:

- ксеноновая газоразрядная лампа (Xenon Gas Discharge);
- флуоресцентная лампа с горячим катодом (Hot Cathode Fluorescent);
- флуоресцентная лампа с холодным катодом (Cold Cathode Fluorescent)

В сканерах для дома и офиса по ряду причин используются лишь лампы с холодным катодом.

Рис. 6.10. Высоковольтный модуль

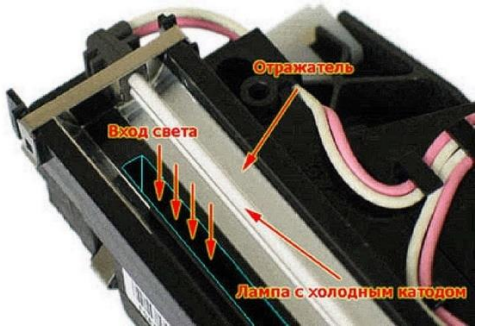



Рис. 6.11. Лампа с холодным катодом

Лампа сканера закреплена на пластмассовом шасси сканирующей каретки непосредственно над отражателем. Сам отражатель имеет форму рефлектора в форме увеличительного зеркала. Свет от него усиливается, чтобы ярко осветить объект на планшете. Отразившись от оригинала на стекле, свет проходит сквозь щель шасси и принимается первым, самым длинным зеркалом оптической системы.

Среди очевидных преимуществ лампы с холодным катодом можно отметить большой срок службы, который составляет 5 000 – 10 000 часов. По этой причине, кстати, в некоторых сканерах не используются отключение лампы после завершения операции сканирования. Кроме этого, лампы не требуют какого-то дополнительного охлаждения и очень дешевы при производстве. Из недостатков – очень медленное включение. Типовое время разогрева лампы от 30 секунд до нескольких минут.

Лампа оказывает важное воздействие на результат сканирования. Даже при небольшом уходе характеристик источника света изменяется и падающий на приемную матрицу отраженный от оригинала световой поток. Поэтому и нужно столь длительное время разогрева лампы перед сканированием. Некоторые драйверы позволяют уменьшить время разогрева, если качество оцифровки не так важно (например, при сканировании текстовой информации). Чтобы как-то скомпенсировать уход характеристик лампы (а это неизбежно происходит при длительной эксплуатации аппарата), сканеры автоматически выполняют процедуру самокалибровки по черно-белой мишени, располагающейся внутри корпуса.

6.1.5 Работа АЦП

Аналого-цифровой преобразователь, обеспечивает перевод аналоговых сигналов в цифровую форму.

Всегда лучше выбирать сканер, у которого разрядность больше.

Различия между результатами работы 36-ти и 42-х-битных сканеров практически незаметны (человеческий глаз способен различить примерно 24 бита цветовых оттенков, т.е. около 16,7 млн.). В нашем случае, разрядность преобразователя и глубина цвета – это одно и то же. Чем больше разрядность преобразователя, тем достовернее сканер может передать цвет каждой точки изображения. Соответственно, тем больше изображение будет походить на оригинал.

6.1.6 Процессор сканера

Современные сканеры оснащают специализированными процессорами. В число задач такого процессора входит согласование действий всех цепей и узлов, а также формирование данных об изображении для передачи персональному компьютеру. В некоторых моделях сканеров на процессор возлагаются также функции контроллера интерфейса.

Список программных инструкций для процессора хранится в микросхеме постоянной памяти. Данные в эту микросхему записываются производителем сканера на этапе производства. Содержимое микросхемы называется "микропрограммой" или "firmware". У некоторых профессиональных сканеров предусмотрена возможность ее обновления.

Помимо микросхемы постоянной памяти в сканерах используется и оперативная память, играющая роль буфера (ее типовые значения – 1 или 2 Мбайт). Сюда направляется сканируемая информация, которая практически сразу передается на ПК. После отправки содержимого из памяти персональному компьютеру, процессор обнуляет буфер для формирования новой посылки. Инструкции для процессора также заносятся в ячейки оперативной памяти, но уже самого процессора. Организация его памяти построена по принципу конвейера, т.е. после выполнения инструкции, стоящей в очереди первой, ее место занимает вторая, а место последней – новая инструкция.

6.1.7 Контроллер интерфейса

За обмен информацией и командами между сканером и компьютером отвечает контроллер интерфейса. Данная микросхема может отсутствовать в том случае, если процессор располагает интегрированным модулем контроллера. В настоящее время SOHO-сканеры ограничиваются интерфейсами USB, FireWire и SCSI.

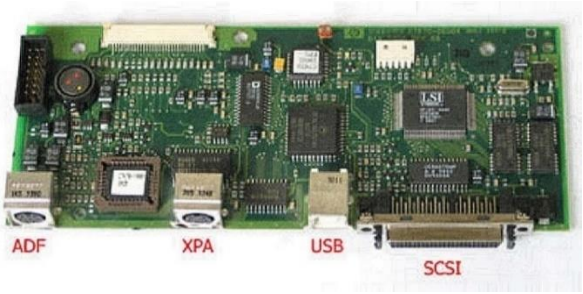



Рис. 6.12. SCSI- и USB-порты в интерфейсной плате

Совершенно очевидно, что в аппаратах с разными интерфейсами установлены такие же разные контроллеры. Между собой они не совместимы.

**SCSI (Small Computer Systems Interface).** Сканеры с интерфейсом SCSI были наиболее распространены несколько лет назад. Надо признать, что эра SCSI-сканеров подходит (или уже подошла) к концу. Основная причина – появление высокоскоростных интерфейсов USB и FireWire, не

требующих ни особой деликатности при подключении, ни дополнительных адаптеров. Среди достоинств SCSI-интерфейса можно выделить его высокую пропускную способность, а также возможность подключения до семи различных устройств на одну шину. Из основных недостатков SCSI – высокую стоимость организации интерфейса и необходимости задействования дополнительного контроллера.

**USB (Universal Serial Bus).** Интерфейс USB получил самое широкое распространение благодаря его интеграции во все современные системные платы в качестве основного разъема для периферийных устройств. Сегодня абсолютное большинство сканеров для дома выпускается именно с USB-интерфейсом. Кроме того, группа CIS-сканеров получает необходимое питание по USB-порту.

**FireWire (IEEE1394).** При выборе типа подключения, FireWire-интерфейс является более предпочтительным. FireWire представляет собой последовательный высокоскоростной интерфейс ввода/вывода, отличаясь от USB тем, что для обеспечения соединения он не требует управляющего контроллера. Организация его работы выполнена по схеме peer-to-peer. Собственно, за счет этого и достигается более низкая (в сравнении с USB) загрузка центрального процессора. В скором времени свет увидят периферийные устройства с новой модификацией этого интерфейса – FireWire 800 (IEEE1394b). Именно тогда он станет самым скоростным среди периферийных стандартов, которые когда-либо были разработаны.

6.1.8 Протяжный механизм сканера

Основной подвижный модуль сканера – его сканирующая каретка. В нее входят оптический блок, с системой линз и зеркал, светочувствительная матрица, лампа с холодным катодом (если это CCD-сканер) и плата инвертора. К сканирующей каретке жестко закреплен зубчатый протяжный ремень, который приводит в движение шаговый двигатель аппарата.

Рис. 6.13. Место крепления ремня к сканирующей каретке

Рис. 6.14. Элементы протяжного механизма

За плотный контакт ремня с шестеренками отвечает специальная натяжная пружина, которая надевается непосредственно на него. Лафет со сканирующей кареткой перемещается по направляющим салазкам, вдоль корпуса аппарата (рис. 6.14).

6.1.9 Блок питания сканера

Домашние или офисные сканеры потребляют не слишком много энергии от сети, поэтому в блоках питания SOHO-аппаратов не найти мощных элементов.

Внутренний блок питания рассматриваемого здесь аппарата выдает напряжения 24 Вольт / 0.69 А, 12 Вольт / 0.15 А и 5 Вольт / 1 А. Т.к. для источника света – лампы с холодным катодом, требуется высокое напряжение в несколько киловольт, за ее питание отвечает отдельный блок, в основе которого находится инвертор.

Рис. 6.15. Блок питания сканера

6.2 Задание

1. Изучить устройство и принцип работы сканера.
2. Подключить и настроить сканер.
3. Установить программное обеспечения необходимое для работы со сканером.
4. Сканирование двух листов по варианту.
5. Распознать изображение (текст, таблицы, рисунки) и сохранить их в MS Word.
6. Получить оптимальный (небольшой) файл для отправки по сети Internet. В зависимости от количества точек на дюйм (при сканировании).

6.3 Контрольные вопросы

1. Матрицы сканера двух типов. Достоинства и недостатки.
2. По каким причинам CCD-сканеры оцифровывают оригиналы гораздо качественнее, чем аппараты с контактным сенсором?
3. Почему важна разрядность преобразователя?
4. RGB-фильтр сканера.
5. Блок управления сканера.
6. Виды ламп, используемые в сканерах. Лампа с холодным катодом.
7. Чем отличается оптическое разрешение от механического?
8. Принцип работы АЦП сканера.
9. Интерфейсы сканера.
10. Как осуществляется взаимодействие электронных и механических частей сканера?

6.4 Содержание отчета

1. Название, цель, содержание работы.
2. Задание.
3. Результаты выполнения работы.
4. Письменные ответы на контрольные вопросы.
5. Выводы по работе.

6.5 Список использованных источников

1. Энциклопедия аппаратных средств IBMPC, М.Гук.
2. http://f-center.ru/
3. http://www.computermaster.ru/
4. http://my-comp.nm.ru

**Контрольные вопросы:**

1. Какой кнопкой производится первичное сканирование изображения;
2. С помощью чего можно выбрать необходимый объект на отсканированном изображении;
3. Как произвести окончательное сканирование картинки;
4. Как изменить настройки сканирования;
5. Для каких целей служит программа Центр решений HP

# ПРАКТИЧЕСКАЯ РАБОТА № 11

## НАСТРОЙКА ПАРАМЕТРОВ РАБОТЫ ПРИНТЕРОВ. ЗАМЕНА КАРТРИДЖЕЙ

**Цель:** приобрести практические навыки в настройке параметров работы принтеров. Замене картриджей.

**Сформировать следующие умения: У 2.** правильно эксплуатировать и устранять типичные выявленные дефекты технических средств информатизации;

**Структура отчёта по практической работе.**

1. Титульный лист.
2. Содержание.
3. Цель работы.
4. Задание.
5. Теоретическая часть.
6. Практическая часть.
7. Выводы.
8. Библиографический список.

**Цель работы.**

1. Ознакомиться и получить навыки работы со специализированными техническими средствами получения оцифрованной графической информации.
2. Ознакомиться и получить навыки работы со специализированными программными средствами распознавания текстовой информации на графическом изображении.

**Задание.**

1. Оборудовать рабочее место специализированными техническими и программными средствами для получения оцифрованной графической информации и преобразования её в текстовый формат. Изучить справочные руководства по использованию программы сканирования и распознавания текста.
2. Произвести сканирование с источника, содержащего текстовый фрагмент и графическое изображение с разрешающей способностью 72, 96, 120, 150, 200, 300 точек на дюйм. Определить размеры полученных графических файлов в формате BMP.

3. Выполнить распознавание текста для каждого из полученных графических файлов. Определить количество ошибок (неправильно распознанных символов, включая знаки препинания и пробелы) для каждого случая. Сделать выводы о факторах, влияющих на качество распознавания текстовой информации и способах уменьшения ошибок распознавания.

4. При помощи графического редактора сохранить файл, имеющий максимальное разрешение, в различных форматах (JPG (низкое качество), JPG (среднее качество), JPG (наилучшее качество), GIF, PNG). Выполнить качественное и количественное сравнение исходного BMP и файлов, использующих алгоритмы сжатия. Сделать выводы относительно достоинств и областей использования каждого из форматов.

**Теоретическая часть:**

1. Цифровое представление аналоговой Графической информации.
2. Графические форматы файлов.
3. Описание программных и технических средств, используемых в работе.
4. Порядок подготовки рабочего места к работе.

**Практическая часть:**

1. Ход работы.
2. Схематичное изображение рабочего места с указанием использованных технических средств.
3. Протокол работы №1. Распечатки распознанного текста для каждого разрешения.
4. График зависимости относительного количества ошибок от выбранного разрешения. Промежуточные выводы.
5. Протокол работы №2. Промежуточные выводы.

Протокол работы №1

Количество символов в оригинальном тексте ($N$): _____

| №№ п./п | Разрешение сканирования, dpi | Количество ошибок ($n$) | Относительное количество ошибок ($n/N$)*100% |
|---|---|---|---|
| 1 | 72 | | |
| 2 | 96 | | |
| 3 | 120 | | |
| 4 | 150 | | |
| 5 | 200 | | |
| 6 | 300 | | |

Протокол работы №2

Разрешение сканирования, dpi: _____

| №№ п./п | Формат (качество) | Размер файла, байт | Качественная оценка изображения в масштабе 100% (наблюдаемые явления, эффекты, особенности) |
|---|---|---|---|
| 1 | BMP | | |
| 2 | JPG (низкое качество) | | |
| 3 | JPG (среднее качество) | | |
| 4 | JPG (наилучшее качество) | | |
| 5 | GIF | | |
| 6 | PNG | | |

**Контрольные вопросы:**

1. Виды принтеров;
2. Краткая справка о механизме работы струйного принтера;
3. Режимы печати;
4. Описание программных и технических средств, используемых в работе;

# ПРАКТИЧЕСКАЯ РАБОТА № 12

## ПОДКЛЮЧЕНИЕ И РАБОТА С НЕСТАНДАРТНЫМИ ПЕРИФЕРИЙНЫМИ УСТРОЙСТВАМИ ПК

**Цель:** приобрести практические навыки в подключении и работой с нестандартными периферийными устройствами ПК.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Структура отчёта по практической работе**

1. Титульный лист.
2. Содержание.
3. Цель работы.
4. Задание.
5. Теоретическая часть.
6. Практическая часть.
7. Выводы.
8. Библиографический список.

**Цель работы:**

1. Ознакомиться и получить навыки работы с нестандартными периферийными устройствами. (Мультимедийный проектор)

2. Ознакомиться и получить навыки работы с программными средствами нестандартных периферийных устройств.

**Задание:**

1. Оборудовать рабочее место специализированными техническими и программными средствами. Изучить справочные руководства по установке устройства.

2. Произвести подключение и установку драйверов устройства. Произвести настройку работы устройства. Выполнить демонстрацию работы устройства.

**Теоретическая часть:**

1. Классификация устройства
2. Назначение устройства
3. Основные характеристики
4. Принцип работы

**Практическая часть:**

1. Выполнить подключение устройства
2. Установить программное обеспечение устройства
3. Изменить параметры работы устройства
4. Выполнить работу устройства

**Контрольные вопросы:**

1. Классификация устройства;
2. Изменить параметры работы устройства;
3. Принцип работы устройства;
4. Основные характеристики устройства;

# ПРАКТИЧЕСКАЯ РАБОТА № 13

## ПЕРЕВОД ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ

**Цель:** приобрести практические навыки в переводе чисел из одной системы счисления в другую.

**Сформировать следующие умения: У 2.** правильно эксплуатировать и устранять типичные выявленные дефекты технических средств информатизации;

**Цель работы:** научиться переводить числа из одной системы счисления в другую

*Система счисления (СС)* – система приемов и правил, которые позволяют устанавливать взаимно однозначное соответствие между любым числом и его представлением в виде совокупности конечного числа символов.

В зависимости от способа изображения чисел с помощью цифр системы счисления делятся на *позиционные и непозиционные.*

В *непозиционных* системах любое число определяется как некоторая функция от численных значений совокупности цифр, которые представляют это число. Цифры в непозиционных системах счисления отвечают некоторым фиксированным числам. Пример непозиционной системы – римская система исчисления:

1 – I
5 – V
10 – X
50 – L
100 – C
500 – D
1000 - M

В электронных цифровых устройствах применяются позиционные системы счисления. *Позиционной* системой счисления называется потому, что значение каждой входящей в число цифры зависит от ее положения в записи числа.

Любая позиционная СС с основанием $q$ может быть представлена в виде полинома:

$A(q) = rnqn + rn–1qn–1 + rn–2qn–2 + … + r1q1 + r0q0 + r–1q–1 + …,$

где $A$ – число в позиционной СС с основанием $q$; $ri$ – коэффициент; $n$ – степень и индекс.

Предположим, необходимо перевести число 458(10) в двоичную, восьмеричную и шестнадцатеричную:

**Перевод в двоичную систему счисления из десятичной**

Число 458 делится на 2 без остатка (на 2, потому что основание двоичной системы счисления 2), поэтому:

Число 229 не делится на 2 без остатка, ближайшее число, которое делится на 2 без остатка – 228, поэтому:

и так далее:

Теперь записываем полученные в остатках цифры снизу вверх, начиная с последней единицы (выделена красным):

111001010(2) = 458(10)

**Перевод в восьмеричную систему счисления из десятичной**

Число 458 не делится на 8 без остатка (на 8, потому что основание восьмеричной системы счисления 8), ближайшее число, которое делится на 8 без остатка - 456 поэтому:

Число 57 не делится на 8 без остатка, ближайшее число, которое делится на 8 без остатка - 56 поэтому:

Цифра 7, полученная в результате второго деления (выделена красным) меньше 8, поэтому последовательное деление на 8 на этом можно прекратить и выписать полученное число в восьмеричной системе счисления снизу вверх, начиная с последней семёрки:

712(8) = 458(10)

**Перевод в шестнадцатеричную систему счисления из десятичной**

Число 458 не делится на 16 без остатка (на 16, потому что основание шестнадцатеричной системы счисления 16), ближайшее число, которое делится на 16 без остатка - 448 поэтому:

Число 28 не делится на 16 без остатка, ближайшее число, которое делится на 16 без остатка - 16 поэтому:

Цифра 1, полученная в результате второго деления (выделена красным) меньше 16, поэтому последовательное деление на 16 на этом можно прекратить и выписать полученное число в шестнадцатеричной системе счисления снизу вверх, начиная с последней единицы, заменяя цифры от 10 до 15 буквами от A до F:

**10 – A**
**11 – B**
**12 – C**
**13 – D**
**14 – E**
**15 – F**

Число 458 в шестнадцатеричной системе счисления будет иметь вид:

1CA(16) = 458(10)

**Перевод дробной части из десятичной системы счисления в двоичную**

Предположим, необходимо перевести число 206,116 из десятичной системы счисления в двоичную. Целая часть переводится по правилам, описанным выше. 206(10) = 11001110(2)

Дробная часть постоянно умножается на 2 (поскольку это основание системы счисления, в которую нужно переводить):

0,116*2 = 0,232
0,232*2 = 0,464
0,464*2 = 0,928
0,928*2 = 1,856
0,856*2 = 1,712
0,712*2 = 1,424
0,424*2 = 0,848
0,848*2 = 1,696
0,696*2 = 1,392
0,392*2 = 0,784
и т.д.

Дробная часть записывается сверху вниз из цифр, выделенных красным.

206,116(10) = 11001110,0001110110(2)

Для восьмеричной и шестнадцатеричной систем счисления перевод дробной части будет аналогичен, только умножать дробную часть нужно на основание соответствующей системы счисления.

**Перевод из двоичной системы счисления в десятичную**

Предположим, необходимо перевести число 101101,101 из двоичной системы счисления в десятичную. Для целой части над каждым числом расставляем по порядку степени, начиная с конца:

150413120110

Для дробной части расставляем отрицательные степени:

150413120110,1-10-21-3

Умножаем каждую цифру данного числа на 2 (основание системы счисления, из которой мы переводим) в степени, расставленной над соответствующей цифрой (вертикальная черта разделяет целую и дробные части):

1*25+0*24+1+23+1*22+0*21+1*20 | 1*2-1+0*2-2+1*2-3=32+0+8+4+0+1 | 1/2-1+0+1/2-3= =32+8+4+1 | 0,5+0,125 = 45,625

Таким образом, 101101,101(2) = 45,625(10)

Для восьмеричной и шестнадцатеричной систем счисления перевод в десятичную систему счисления будет аналогичен, только умножать каждую цифру числа нужно на основание соответствующей системы счисления.

Таблица 1 - Таблица перевода из двоичной системы счисления в восьмеричную и шестнадцатеричную

| Двоичная | Восьмеричная | Шестнадцатеричная |
|---|---|---|
| 0000 | 0 | 0 |
| 0001 | 1 | 1 |
| 0010 | 2 | 2 |

| Двоичная | Восьмеричная | Шестнадцатеричная |
|---|---|---|
| 0011 | 3 | 3 |
| 0100 | 4 | 4 |
| 0101 | 5 | 5 |
| 0110 | 6 | 6 |
| 0111 | 7 | 7 |
| 1000 | | 8 |
| 1001 | | 9 |
| 1010 | | A |
| 1011 | | B |
| 1100 | | C |
| 1101 | | D |
| 1110 | | E |
| 1111 | | F |

Предположим, число 10101011(2) нужно перевести в восьмеричную и шестнадцатеричную системы счисления. Выделяем цифры данного числа по 3 (триады) **НАЧИНАЯ С КОНЦА** (для разделения чисел на триады воспользуемся тире):

010-101-011

Ноль впереди можно добавить, чтобы была полная триада, это не изменит числа, но добавление нуля в конце изменит исходное число.

Находим каждую триаду в столбце Двоичная таблицы 1 и берём соответствующее значение из столбца Восьмеричная (для разделения цифр используются тире):

2-5-3

Таким образом, 10101011(2) = 253(8)

Выделяем цифры данного числа по 4 (тетрады) **НАЧИНАЯ С КОНЦА** (для разделения чисел на тетрады воспользуемся тире):

1010-1011

Находим каждую тетраду в столбце Двоичная таблицы 1 и берём соответствующее значение из столбца Шестнадцатеричная (для разделения цифр используются тире):

A-B

Таким образом, 10101011(2) = AB(16)

Предположим, число 71(8) нужно перевести в двоичную систему счисления из восьмеричной.

Находим каждую цифру этого числа в столбце Восьмеричная таблицы 1 и берём соответствующее значение из столбца Двоичная, соблюдая правило, согласно которому каждой цифре восьмеричной системы счисления должно соответствовать 3 цифры двоичной системы счисления (для разделения цифр используются тире):

110-001

Таким образом, 71(8) = 110001(2)

Предположим, число D1F4(16) нужно перевести в двоичную систему

счисления из шестнадцатеричной.

Находим каждую цифру этого числа в столбце Шестнадцатеричная таблицы 1 и берём соответствующее значение из столбца Двоичная, соблюдая правило, согласно которому каждой цифре шестнадцатеричной системы счисления должно соответствовать 4 цифры двоичной системы счисления (для разделения цифр используются тире):

1101-0001-1111-0100

Таким образом, D1F4(16) = 1101000111110100(2)

Таблица 2 - Варианты заданий

| № варианта | Q | S | T | W | U | L | J |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 7 | 4 | 7 | 1 | A | 6 |
| 2 | 9 | 2 | 7 | 7 | F | 1 | 1 |
| 3 | 7 | 1 | 2 | 5 | 4 | D | 3 |
| 4 | 1 | 4 | 1 | 9 | 1 | E | 6 |
| 5 | 8 | 6 | 3 | 4 | A | B | 6 |
| 6 | 4 | 6 | 2 | 8 | C | 6 | 4 |
| 7 | 3 | 6 | 8 | 7 | 3 | A | 2 |
| 8 | 7 | 3 | 7 | 3 | A | F | 2 |
| 9 | 5 | 8 | 0 | 5 | E | 5 | 6 |
| 10 | 2 | 1 | 2 | 1 | E | E | 0 |
| 11 | 6 | 9 | 8 | 7 | F | 1 | 2 |
| 12 | 1 | 4 | 8 | 5 | A | 6 | 4 |
| 13 | 1 | 3 | 4 | 1 | D | 1 | 5 |
| 14 | 7 | 0 | 8 | 3 | C | F | 4 |
| 15 | 4 | 2 | 9 | 8 | B | A | 2 |
| 16 | 8 | 7 | 3 | 1 | A | C | 4 |
| 17 | 6 | 0 | 1 | 7 | F | D | 3 |
| 18 | 8 | 8 | 2 | 6 | F | E | 6 |
| 19 | 1 | 0 | 1 | 7 | 4 | D | 5 |
| 20 | 4 | 9 | 9 | 9 | E | C | 3 |
| 21 | 7 | 7 | 2 | 4 | B | 5 | 3 |
| 22 | 4 | 4 | 6 | 9 | D | A | 0 |
| 23 | 4 | 8 | 9 | 3 | F | 6 | 0 |
| 24 | 0 | 9 | 5 | 7 | D | A | 1 |
| 25 | 5 | 3 | 8 | 3 | C | E | 2 |
| 26 | 5 | 6 | 0 | 1 | E | 7 | 6 |
| 27 | 9 | 4 | 4 | 4 | E | F | 6 |
| 28 | 5 | 1 | 6 | 5 | C | 5 | 6 |
| 29 | 1 | 9 | 9 | 7 | B | F | 7 |
| 30 | 5 | 0 | 1 | 9 | E | B | 4 |

**Задание 1.** Перевести числа из десятичной системы счисления в двоичную, шестнадцатеричную и восьмеричную системы счисления согласно своему варианту в таблице 2 (номер варианта определяется по списку в журнале):

1. QSW (10).

2.W4, S (10) – с шестью знаками после запятой.

3. TWQ, Q (10) – с шестью знаками после запятой.

**Задание 2.** Перевести числа из десятичной системы счисления сначала в двоичную систему счисления, а из двоичной – в шестнадцатеричную и восьмеричную используя таблицу 1:

1. 4QW8(10)
2. ST72(10)
3. 74,2TS4(10) – с двенадцатью двоичными разрядами после запятой.

**Задание 3.** Перевести числа в десятичную систему:

1. 1J3(8) и UL9(16) – без промежуточного перевода в двоичную систему счисления.
2. J4, J13J (8) и LU,5UUL (16) – сначала в двоичную, а затем в десятичную с шестью знаками после запятой.
3. 23JJ,452(8) и ULU, LLUL (16) – сначала в двоичную, а затем в десятичную с шестью знаками после запятой.

**Задание 4.** Записать свою дату рождения в римской системе счисления в формате ДД.ММ.ГГГГ

Если задание выполняется двумя студентами, то указать фамилию первого студента, его дату рождения в римской системе счисления, на другой строке то же самое для второго студента. (Данные по римской системе счисления представлены в справочной информации к данной практической работе.)

**Задание 5.** Оформить отчет по лабораторной работе (форма отчета приведена в Приложении) и отправить его на проверку преподавателю электронной почтой: SorokinaSV.4@gmail.com

**Задание 6.** Перевести числа в десятичную систему:

1. 1J3(8) и UL9(16) – без промежуточного перевода в двоичную систему счисления.
2. J4, J13J(8) и LU,5UUL(16) – сначала в двоичную, а затем в десятичную с шестью знаками после запятой.
3. 23JJ,452(8) и ULU, LLUL (16) – сначала в двоичную, а затем в десятичную с шестью знаками после запятой.

**Задание 7.** Записать свою дату рождения в римской системе счисления в формате ДД.ММ.ГГГГ

Если задание выполняется двумя студентами, то указать фамилию первого студента, его дату рождения в римской системе счисления, на другой строке то же самое для второго студента. (Данные по римской системе счисления представлены в справочной информации к данной практической работе.)

**Задание 8.** Оформить отчет по лабораторной работе (форма отчета приведена в Приложении) и отправить его на проверку преподавателю электронной почтой: SorokinaSV.4@gmail.com

**Контрольные вопросы:**

1. Перевести числа в десятичную систему:

1J3(8) и UL9(16) – без промежуточного перевода в двоичную систему счисления.

2. Записать свою дату рождения в римской системе счисления в формате ДД.ММ.ГГГГ

3. Оформить отчет по лабораторной работе (форма отчета приведена в Приложении) и отправить его на проверку преподавателю электронной почтой: SorokinaSV.4@gmail.com

# ПРАКТИЧЕСКАЯ РАБОТА № 14

## ВЫПОЛНЕНИЕ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ НАД ЧИСЛАМИ В ПРЯМОМ, ОБРАТНОМ И ДОПОЛНИТЕЛЬНЫХ КОДАХ

**Цель:** приобрести практические навыки в выполнении арифметических операций над числами в прямом, обратном и дополнительных кодах.

**Сформировать следующие умения:**

**У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Тема:** Выполнение арифметических операций в естественной и нормальной форме

**Цель работы:** научиться производить операции сложения и вычитания в дополнительных и обратных кодах

**Краткие теоретические сведения:**

**Числа с фиксированной точкой.**

Запись числа с фиксированной точкой обычно имеет знаковый и цифровой разряды. Фиксированная точка означает, что на этапе конструирования ЭВМ было определено, сколько и какие разряды машинного слова отведены под изображение целой и дробной частей числа. Пример. Как частный случай числа с фиксированной точкой может быть рассмотрена запись целого числа (в этом случае все разряды, кроме знакового, используются для записи целой части).

Пример. Ячейка с записью целого числа.

К достоинствам использования чисел с фиксированной точкой относятся простота выполнения арифметических операций и высокая точность изображения чисел. К недостаткам - небольшой диапазон представления чисел.

**Числа с плавающей точкой.**

Для представления чисел с плавающей точкой (ЧПТ) используется полулогарифмическая форма записи числа:

$$N = \pm mq^{\pm p}$$

**где** ***q*** **- основание системы счисления;**
**p - порядок числа;**
***m*** **- мантисса числа *N*.**

Положение точки определяется значением порядка p. С изменением порядка точка перемещается (плавает) влево или вправо. Пример.

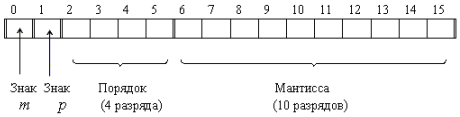
$125_{10}=12.5*10^1=1.25*10^2=0.125*10^3=0.0125*10^4=...$

Для установления однозначности при записи чисел принята **нормализованная форма** записи числа. Мантисса нормализованного числа может изменяться в диапазоне: $1/q \le | m | < 1$. Таким образом в нормализованных числах цифра после точки должна быть значащей.

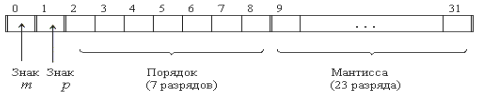
$$\underbrace{0.0832\cdot 10^3}_{\text{ненормализованное число}} = \underbrace{0832\cdot 10^2}_{\text{нормализованное число}}$$

Пример.

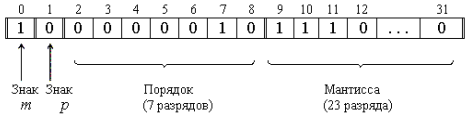
Для представления чисел в машинном слове выделяют группы разрядов для изображения мантиссы, порядка, знака числа и знака порядка: а) представление чисел в формате полуслова



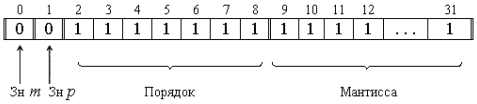

б) представление чисел в формате слова



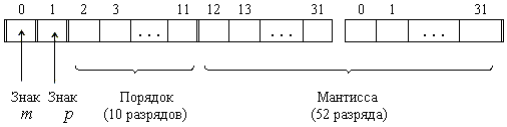

Наиболее типично представление ЧПТ в формате слова (32 разряда). Пример. Число $A=-3.5_{10}=-11.1_2=-0.111\cdot 10^{10}$




Максимальным числом представимым в формате слова будет $A=(0.1111...1\cdot 10^{1111111})_2 \approx (1\cdot 2^{127})_{10}$.




Таким образом числа с плавающей точкой позволяют увеличить диапазон обрабатываемых чисел, но при этом точность изображения чисел определяется только разрядами мантиссы и уменьшается по сравнению с числами с фиксированной точкой. При записи числа в формате слова диапазон представимых чисел будет от $-1\cdot 2^{127}$ до $1\cdot 2^{127}$ ($2^{127} \approx 10^{38}$), а точность определяться мантиссой, состоящей из 23 разрядов. Точность может быть повышена путем увеличения количества разрядов мантиссы. Это реализуется путем представления чисел с так называемой двойной точностью (используется формат двойного слова):




**Арифметические операции над числами с фиксированной точкой**

***Сложение (вычитание).*** Операция вычитания приводится к операции сложения путем преобразования чисел в обратный или дополнительный

код. Пусть числа A=>O и B=>O, тогда операция алгебраического сложения выполняется в соответствии с табл. 1.

Таблица 1 - Преобразования кодов при алгебраическом сложении

| Требуемая операция | Необходимое преобразование |
|---|---|
| A+B | A+B |
| A-B | A+(-B) |
| -A+B | (-A)+B |
| -A-B | (-A)+(-B) |

Скобки в представленных выражениях указывают на замену операции вычитания операцией сложения с обратным или дополнительным кодом соответствующего числа. При выполнении сложения цифр необходимо соблюдать следующие правила.

1. Слагаемые должны иметь одинаковое число разрядов. Для выравнивания разрядной сетки слагаемых можно дописывать незначащие нули слева к целой части числа и незначащие нули справа к дробной части числа.

2. Знаковые разряды чисел участвуют в сложении так же, как и значащие.

3. Необходимые преобразования кодов производятся с изменением знаков чисел. Приписанные незначащие нули изменяют свое значение при преобразованиях по общему правилу.

4. При образовании единицы переноса из старшего знакового разряда, в случае использования ОК, эта единица складывается с младшим числовым разрядом. При использовании ДК единица переноса теряется. Знак результата формируется автоматически, результат представляется в том коде, в котором представлены исходные слагаемые.

**Пример 1.** Сложить два числа $A_{10}=7$ $B_{10}=16$

$A_2$=+11=+0111;

$B_2$=+1000=+10000.

Исходные числа имеют различную разрядность, необходимо провести выравнивание разрядной сетки:

$[A_2]_П=[A_2]_{ОК}=[A_2]_{ДК}$=0: 00111;

$[B_2]_П=[B_2]_{ОК}=[B_2]_{ДК}$=0: 10000.

Сложение в обратном или дополнительном коде дает один и тот же результат

$$\begin{array}{r} 0\,\vdots\,00111 \\ +0\,\vdots\,10000 \\ \hline C_2=0\,\vdots\,10111 \end{array}$$

$C_{10}=+23.$

Обратим внимание, что при сложении цифр отсутствуют переносы в знаковый разряд и из знакового разряда, что свидетельствует о получении правильного результата.

**Пример 2** Сложить два числа $A_{10}$ = + 16 $B_{10}$ = ≈7 в ОК и ДК. В соответствии с табл. 1 должна быть реализована зависимость A+(-B), в *кото*рой второй член преобразуется с учетом знака

$[A_2]_п = 0 : 10000 = 0 : 10000$ $[A_2]_{ок} = 0 : 10000$ $[A_2]_{дк} = 0 : 10000;$
$[B_2]_п = 1 : 111 = 1 : 00111$ $[B_2]_{ок} = 1 : 11000$ $[B_2]_{дк} = 1 : 11001.$

| Сложение в ОК | Сложение в ДК |
|---|---|
| $[A_2]_{ок} = 0 : 10000$ | $[A_2]_{дк} = 0 : 10000$ |
| $+[B_2]_{ок} = 1 : 11000$ | $+[B_2]_{дк} = 1 : 11001$ |
| $0 : 01000$ | $0 : 01001$ |
| $+ \quad 1$ | |
| $0 : 01001$ | |
| $C_2 = 0 : 01001$ | $C_2 = 0 : 01001$ |
| $C_{10} = +9$ | $C_{10} = +9$ |

При сложении чисел в ОК и ДК были получены переносы в знаковый разряд и из знакового разряда. В случае ОК перенос из знакового разряда требует дополнительного прибавления единицы младшего разряда (см.п.4 правил). В случае ДК этот перенос игнорируется.

***Умножение.*** Умножение двоичных чисел наиболее просто. реализуется в прямом коде. Рассмотрим, каким образом оно приводится к операциям сложения и сдвигам.

**Пример 3.** Умножить два числа $A_{10}=7$ $B_{10}=5$.

Перемножим эти числа, представленные прямыми двоичными кодами так же, как это делается в десятичной системе.

Нетрудно видеть, что произведение получается путём сложения частных произведений, представляющих собой разряды множимого, сдвинутые влево в соответствии с позициями разрядов множителя. Частные произведения, полученные умножением на нуль игнорируются. Важной особенностью операции умножения n-разрядных сомножителей является увеличение разрядности произведения до n+n=2n. Знак произведения формируется путём сложения знаковых разрядов сомножителей. Возможные переносы из знакового разряда игнорируются.

***Деление.*** Операция деления, как и в десятичной арифметике, является обратной операции умножения. Покажем, что и эта операция приводится к последовательности операций сложения и сдвига.

**Пример 4.** Разделить два числа $A_{10}=45$ $B_{10}=5$

$[A_2]_п = 101101$
$[B_2]_п = 101$

| Делимое | Делитель |
|---|---|
| 101101 | \| 101 |
| -101 | 1001 – частное |
| 0101 | |
| - 101 | |
| 0 | |

$[C_2]_п = 1001$
$C_{10} = 9$

Деление произведено так же, как это делается обычно в десятичной системе. Сначала проверяется, можно ли вычесть значение делителя из старших разрядов делимого. Если возможно, то в разряде частного записывается единица и определяется частная разница. В противном случае в частное записывается нуль и разряды делителя сдвигаются вправо на один разряд по отношению к разрядам делимого. К полученной предыдущей разнице сносится очередная цифра делимого, и данный процесс повторяется, пока не будет получена необходимая точность. Если учесть, что все вычитания в

ЭВМ заменяются сложением в ОК или в ДК (см. табл.1), то действительно операция деления приводится к операциям сложения и сдвигам вправо разрядов делителя относительно разрядов делимого. Отметим, что делимое перед операцией деления должно быть приведено к 2*n*-разрядной сетке. Только в этом случае при делении на *n*-разрядный делитель получается *n*-разрядное частное.

Знак частного формируется также путем сложения знаковых разрядов делимого и делителя, как это делалось при умножении.

Арифметические операции над двоичными числами с плавающей точкой.

В современных ЭВМ числа с плавающей точкой хранятся в памяти машин, имея мантиссу и порядок (характеристику) в прямом коде и нормализованном виде. Все арифметические действия над этими числами выполняются так же, как это делается с ними, если они представлены в полулогарифмической форме (мантисса и десятичный порядок) в десятичной системе счисления. Порядки и мантиссы обрабатываются раздельно.

***Сложение (вычитание).*** Операция сложения (вычитания) производится в следующей последовательности.

1. Сравниваются порядки (характеристики) исходных чисел путем их вычитания p=p1-p2. При выполнении этой операции определяется, одинаковый ли порядок имеют исходные слагаемые.

2. Если разность порядков равна нулю, то это значит, что одноименные разряды мантисс имеют одинаковые веса (двоичный порядок). В противном случае должно проводиться выравнивание порядков.

3. Для выравнивания порядков число с меньшим порядком сдвигается вправо на разницу порядков *Ap.* Младшие выталкиваемые разряды при этом теряются.

4. После выравнивания порядков мантиссы чисел можно складывать (вычитать) в зависимости от требуемой операции. Операция вычитания заменяется операцией сложения в соответствии с данными табл. 2.3. Действия над слагаемыми производятся в ОК или ДК по общим правилам.

5. Порядок результата берется равным большему порядку.

6. Если мантисса результата не нормализована, то осуществляются нормализация и коррекция значений порядка.

**Пример 5.** Сложить два числа $A_{10}$=+1.375; $B_{10}$=-0.625.

$A_2$=+1.011=0: 1011*$10^1$; $B_2$=-0.101=-0:101*$10^0$.

В нормализованном виде эти числа будут иметь вид:

| | *Порядок* | *Мантиса* |
|---|---|---|
| $[A_2]_{ПК}$ = 0:1 | | ↑ 0:1011 |
| | *знак числа* | |
| $[B_2]_{ПК}$ = 0:0 | | ↓ 1:101 |

1. Вычитаем порядки $\Delta p = p_1 - p_2 = 1-0=1$. В машине эта операция требует операции сложения с преобразованием порядка чисел в дополнительный код:

$$\begin{array}{lll} p_1 = 0{:}1 & & [p_1]_{ДК} = 0{:}1 \\ p_2 = 0{:}0 & + & [p_2]_{ДК} = 0{:}0 \\ \hline \Delta p = 0{:}1 & & \end{array}$$

Определяем, что $\Delta p \neq 0$.

*2.* Порядок первого числа больше порядка второго числа на единицу. Требуется выравнивание порядков.

3. Для выравнивания порядков необходимо второе число сдвинуть вправо на один разряд.

$[B_2]$исх=0: 0 1: 101

после сдвига

$[B_2]$п=0: 11:0101

$[m_B]$дк= 1: 1011

4. Складываем мантиссы.

$$[m_A]_{дк} = 0{:}1011$$
$$+[m'_B]_{дк} = 1{:}1011$$
$$[m_C]_{дк} = 0{:}0110$$

Мантисса числа С - положительная.

5. Порядок числа С равен порядку числа с большим порядком, т.е. $p$ =+1.

$[C_2]$п=0: 1 0: 0110.

Видно, что мантисса результата не нормализована, так как старшая цифра мантиссы равна нулю.

6. Нормализуем результат путем сдвига мантиссы на один разряд влево и соответственно вычитаем из значения порядка единицу:

$$[C_2]П = 0{:}00{:}110$$
$$C_{10} = +0.75$$

***Умножение (деление).*** Операция умножения (деления) чисел с плавающей точкой также требует разных действий над порядками и мантиссами. Алгоритмы этих операций выполняются в следующей последовательности.

1. При умножении (делении) порядки складываются (вычитаются) так, как это делается над числами с фиксированной точкой.

2. При умножении (делении) мантиссы перемножаются (делятся).

3. Знаки произведения (частного) формируются путем сложения знаковых разрядов сомножителей (делимого и делителя). Возможные переносы из знакового разряда игнорируются.

**Задание:**

| | I | II | III | IV | V | VI |
|---|---|---|---|---|---|---|
| 1 | $149{,}375_{10}$ | $711{,}25_{10}$ | $360{,}25_{10}$ | $741{,}125_{10}$ | $597{,}25_{10}$ | $237{,}73_{10}$ |
| | $65{,}2_{16}$ | $-3BF{,}A_{16}$ | $-2FE{,}6_{16}$ | $24A{,}4_{16}$ | $24D{,}5_{16}$ | $BB{,}4_{16}$ |
| 2 | 101010000,10111 | 1100110110,0011 | 11001010,01 | 1111100100,11011 | 1101111111,1 | 1000101001,1 |
| | 11001100,01 | 11111110,01 | 1110001,001 | 101110111,011 | 1100111110,1011 | 1111101,1 |
| 3 | $X=69{,}4_{16}$ $Y=A{,}D_{16}$ | X=2B,A16; $Y=36{,}6_{16}$ | $X=7{,}4_{16}$; $Y=1D{,}4_{16}$ | $X=36{,}4_{16}$; $Y=A{,}A_{16}$ | $X=4B{,}2_{16}$; $Y=3C{,}3_{16}$ | $X=4A{,}3_{16}$ $Y=F{,}6_{16}$; |
| 4 | $X=326{,}8_{10}$; $Y=-15{,}2_{10}$ | $X=-220{,}15_{10}$; $Y=6{,}29_{10}$ | $X=150{,}46_{10}$; $Y=-7{,}523_{10}$ | $X=365{,}02_{10}$; $Y=-18{,}251_{10}$ | $X=-128{,}3_{10}$; $Y=6{,}415_{10}$ | $X=421{,}2_{10}$; $Y=-11{,}7_{10}$ |

**Порядок выполнения работы:**

Выполните арифметические операции в естественной и нормальной форме:

1. сложение

2. вычитание

3. умножение

4. деление

**Контрольные вопросы:**

1. Представление положительных и отрицательных чисел в прямом коде;

2. Представление положительных и отрицательных чисел в обратном коде;

3. Представление положительных и отрицательных чисел в дополнительном коде;

4. Представление положительных и отрицательных чисел в модифицированном обратном коде;

5. Представление положительных и отрицательных чисел в модифицированном дополнительном коде

6. Правила выполнения арифметических операций для чисел представленных в обратном коде

7. Правила выполнения арифметических операций для чисел представленных в дополнительном коде

# ПРАКТИЧЕСКАЯ РАБОТА № 15

## ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ «2И», «2ИЛИ», «НЕ», «2И-НЕ», «2ИЛИ-НЕ», «ИСКЛЮЧАЮЩИЕ ИЛИ»

**Цель: п**риобрести практические навыки в построении логических элементов «2И», «2ИЛИ», «НЕ», «2И-НЕ», «2ИЛИ-НЕ», «Исключающие ИЛИ».

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Порядок выполнения работы.**

1. Собрать схему симметричного транзистора мультивибратора по рис.5.1.

2. При помощи осциллографа наблюдать генерацию схемы при включении питания.

3. Замерить длительность периода колебаний и вычислить частоту при двух различных значениях емкостей **С**.

4. Зарисовать осциллограммы выходных сигналов при одинаковых и различных емкостях. С в коллекторно-базовых цепях транзисторов.

5. Собрать схему несимметричного мультивибратора на цифровых интегральных микросхемах по схеме рис. 5.2.

6. Наблюдать осциллографом генерируемые схемой сигнала при двух различных значениях емкости **С**.

Произвести замеры длительности периода генерации и вычислить частоту генерации.

7. Зарисовать осциллограммы генерируемых сигналов на выходе схемы.

8. Убедится, что замыкание тумблера **S** приводит к прекращению генерации.

9. Отключить емкость **С** и наблюдать высокочастотные колебания мультивибратора. Замерить период колебаний **Тм = 1/fм** и вычислить время задержки сигнала элементом микросхемы.

**Содержание отчета.**

Отчет должен содержать:

1. Принципиальные схемы опыта.
2. Тип и параметры используемых транзисторов и микросхем.
3. Тип и основные характеристики используемого осциллографа.
4. Результаты измерения длительности периода и вычисления частоты генерации.
5. Осциллограммы выходных сигналов.
6. Ответы на контрольные вопросы.

**Контрольные вопросы:**

1. Что такое скважность генерируемых сигналов;
2. чем ограничивается величина максимальной частоты генерации мультивибраторов в схеме по рис. 5.1. и 5.2.;
3. Как можно плавно регулировать частоту генерации мультивибратора по рис. 5.1.
4. Как будет вести себя схема по рис. 5.2. если отключить емкость **С**

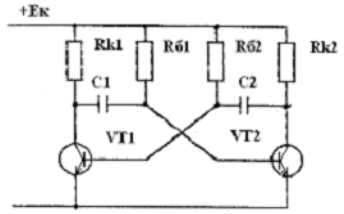



Рис.5.1.

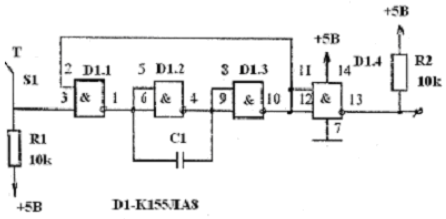



Рис.5.2.

# ПРАКТИЧЕСКАЯ РАБОТА № 16

## МУЛЬТИПЛЕКСОРЫ. ДЕМУЛЬТИПЛЕКСОРЫ

**Цель:** приобрести практические навыки в изучении мультиплексоров и демультиплексоров.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель:** ознакомиться с принципом работы мультиплексора и демультиплексора. Реализовать и исследовать функциональные модули на их основе, ознакомиться с основными характеристиками интегральных преобразователей кодов, получение навыков проектирования мультиплексоров и

демультиплексоров. Изучить связь между ними.

**Задание практической работы:**

1. Реализовать ФАЛ, заданной таблицей истинности, на мультиплексоре MUX 8-1.

2. Реализовать ФАЛ, заданной таблицей истинности, на демультиплексоре DMX 1-16

**Задание лабораторной работы:**

Согласно решению первого задания практической работы №5 построить логическую схему мультиплексора в системе multisim и проверить правильность выполненной работы. Аналогично построить демультиплексор 2-го задания и проверить правильность его работы.

**Результаты работы**

**Задание 1.**

| Входные переменные | |
|---|---|
| a b c d | 01 |
| 0 0 0 0 | 0 |
| 0 0 0 1 | 0 |
| 0 0 1 0 | 0 |
| 0 0 1 1 | 1 |
| 0 1 0 0 | 1 |
| 0 1 0 1 | 0 |
| 0 1 1 0 | 0 |
| 0 1 1 1 | 1 |
| 1 0 0 0 | 0 |
| 1 0 0 1 | 0 |
| 1 0 1 0 | 0 |
| 1 0 1 1 | 0 |
| 1 1 0 0 | 1 |
| 1 1 0 1 | 1 |
| 1 1 1 0 | 0 |
| 1 1 1 1 | 0 |

**Практическое решение задания №1**

| № набора | $A_3$ | $A_2$ | $A_1$ | $A_0$ | q | Примечание |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | $D_0=0$ |
| 1 | 0 | 0 | 0 | 1 | 0 | |
| 2 | 0 | 0 | 1 | 0 | 0 | $D_1=x$ |
| 3 | 0 | 0 | 1 | 1 | 1 | |
| 4 | 0 | 1 | 0 | 0 | 1 | $D_2=\dot{x}$ |
| 5 | 0 | 1 | 0 | 1 | 0 | |
| 6 | 0 | 1 | 1 | 0 | 0 | $D_3=$ |
| 7 | 0 | 1 | 1 | 1 | 1 | |
| 8 | 1 | 0 | 0 | 0 | 0 | $D_4=0$ |
| 9 | 1 | 0 | 0 | 1 | 0 | |
| 10 | 1 | 0 | 1 | 0 | 0 | $D_5=0$ |
| 11 | 1 | 0 | 1 | 1 | 0 | |
| 12 | 1 | 1 | 0 | 0 | 1 | $D_6=1$ |
| 13 | 1 | 1 | 0 | 1 | 1 | |
| 14 | 1 | 1 | 1 | 0 | 0 | $D_7=0$ |

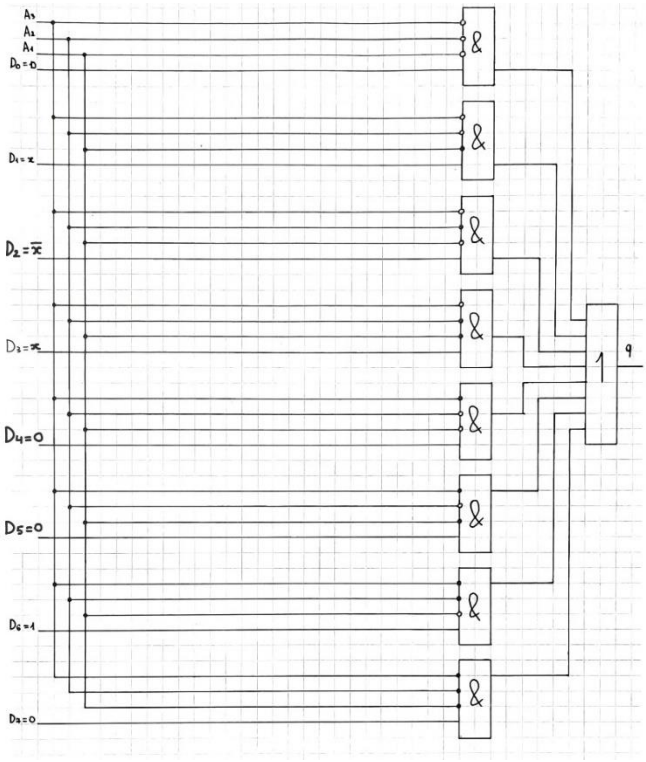
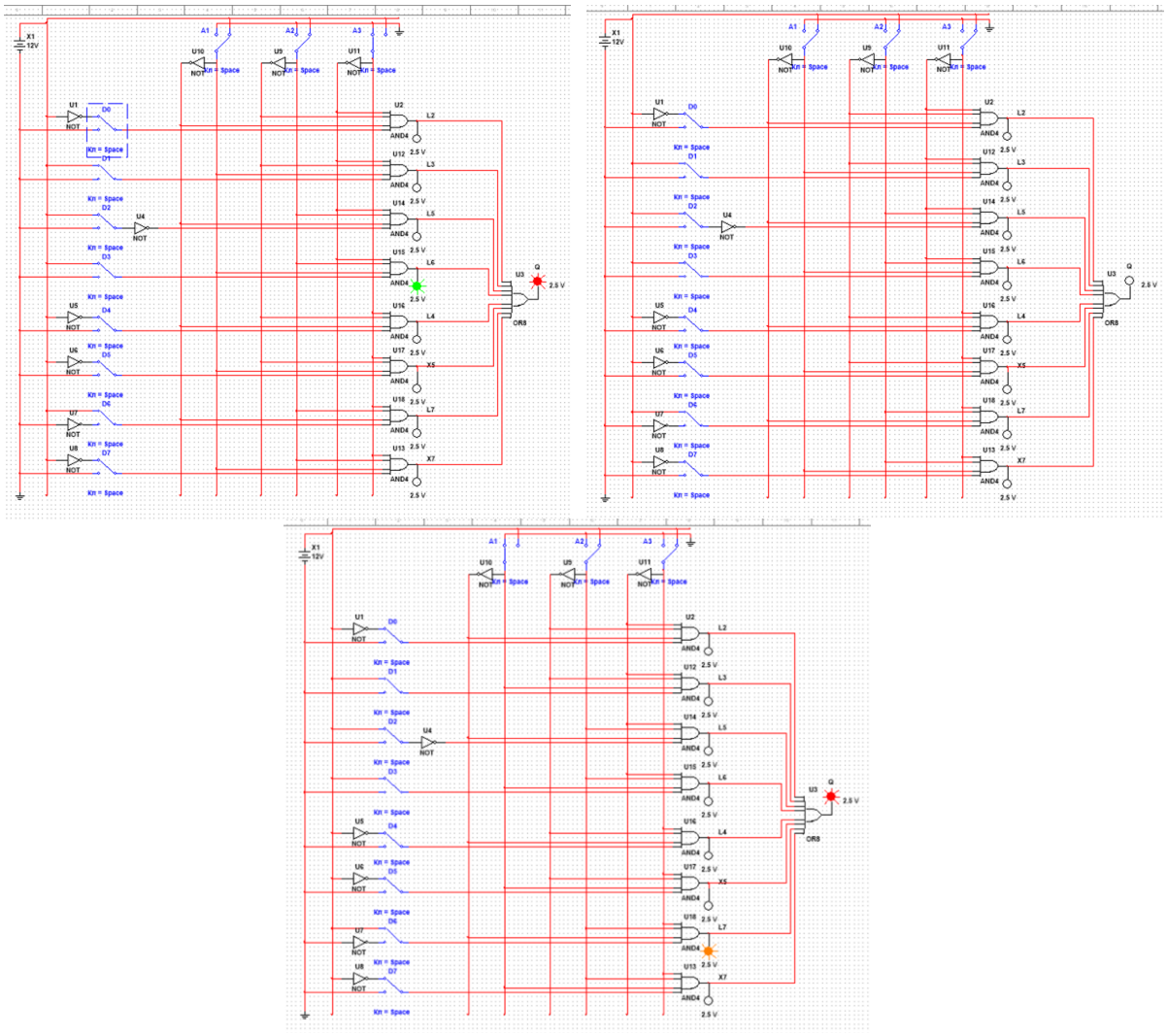



Рис. 2. Лабораторное решение задания №1



**Задание 2.**

| Входные переменные |
|---|
| a b c d |
| 0 0 0 0 |
| 0 0 0 1 |
| 0 0 1 0 |
| 0 0 1 1 |
| 0 1 0 0 |
| 0 1 0 1 |
| 0 1 1 0 |
| 0 1 1 1 |
| 1 0 0 0 |
| 1 0 0 1 |
| 1 0 1 0 |
| 1 0 1 1 |
| 1 1 0 0 |
| 1 1 0 1 |
| 1 1 1 0 |
| 1 1 1 1 |

**Практическое решение задания №2**

| № набора | $A_3$ | $A_2$ | $A_1$ | $A_0$ | q |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 |
| 4 | 0 | 0 | 1 | 1 | 1 |
| 5 | 0 | 1 | 0 | 0 | 1 |
| 6 | 0 | 1 | 0 | 1 | 0 |
| 7 | 0 | 1 | 1 | 0 | 0 |
| 8 | 0 | 1 | 1 | 1 | 1 |
| 9 | 1 | 0 | 0 | 0 | 0 |
| 10 | 1 | 0 | 0 | 1 | 0 |
| 11 | 1 | 0 | 1 | 0 | 0 |
| 12 | 1 | 0 | 1 | 1 | 0 |
| 13 | 1 | 1 | 0 | 0 | 1 |
| 14 | 1 | 1 | 0 | 1 | 1 |
| 15 | 1 | 1 | 1 | 0 | 0 |
| 16 | 1 | 1 | 1 | 1 | 0 |

На рисунке 4 изображено лабораторное решение задания №2

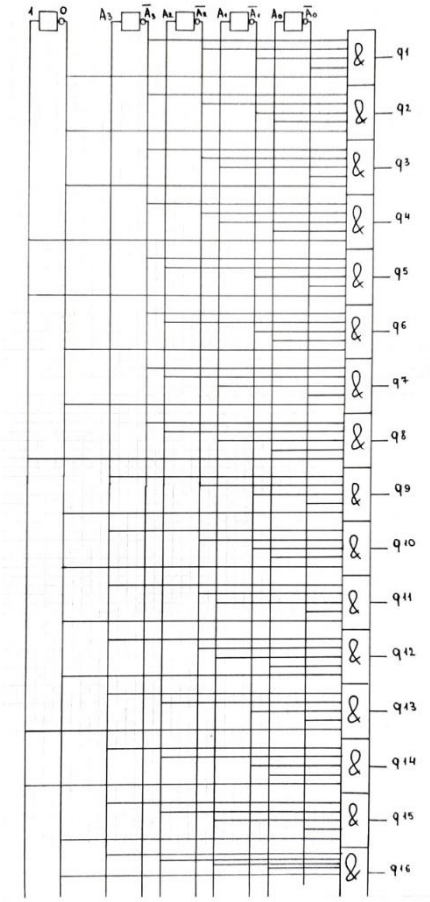
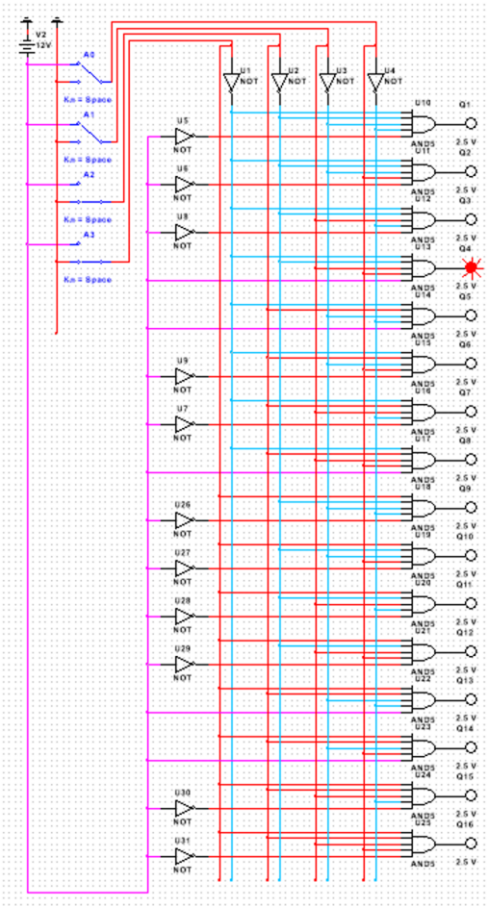
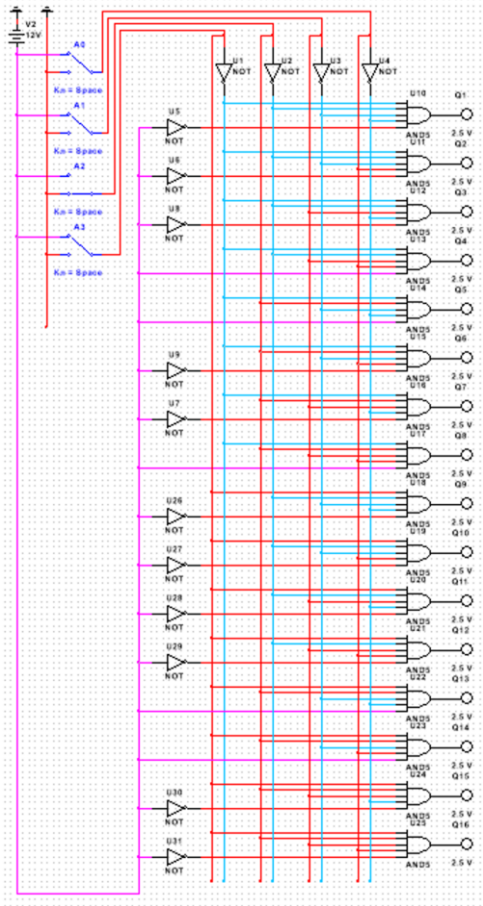
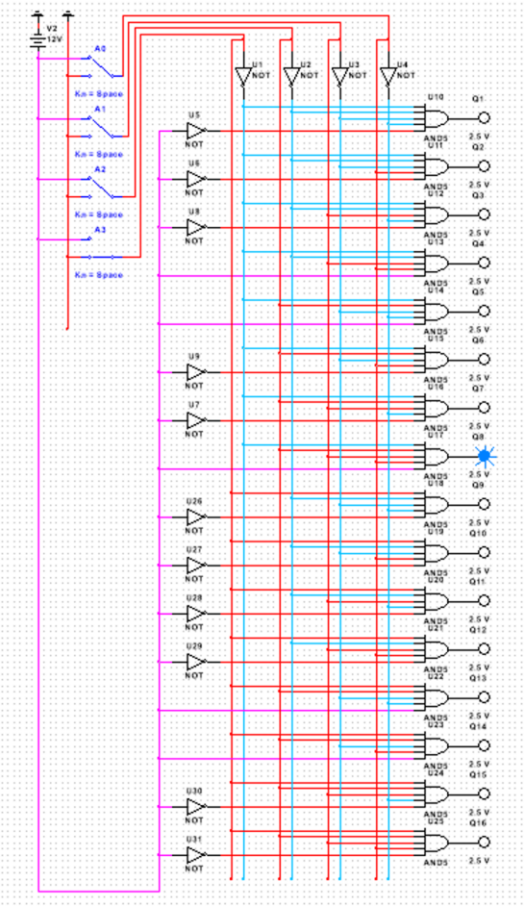


Рисунок 4 – Лабораторное решение задания №2







в

**Контрольные вопросы:**

1. Дайте определение мультиплексора и демультиплексора;
2. На какой базе строятся мультиплекс;
3. Назовите область применения мультиплексоров;
4. Объяснить работу мультиплексора по его логической структуре;
5. Объяснить работу демультиплексора по его логической структуре
6. Как работает демультиплексор
7. Как работает мультиплексор

# ПРАКТИЧЕСКАЯ РАБОТА № 17

## ШИФРАТОРЫ. ДЕШИФРАТОРЫ

**Цель:** приобрести практические навыки в изучении шифраторов и дешифраторов.

**Сформировать следующие умения: У1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель работы:**

1. Изучение принципов работы шифраторов, дешифраторов и мультиплексоров.
2. Реализация логических функций с помощью мультиплексоров.
3. Изучение способов применения дешифраторов.

**Приборы и элементы:**

Генератор слов (панель «Instruments/Word Generator») Логический анализатор (панель «Instruments/Logic Analyzer») Логические пробники (панель «Indicators/Red probe») Источник напряжения + 5 В (панель «Basic/Pull-Up Resistor») Земля (панель «Sources/Ground»)

Двухвходовые элементы И, И-НЕ, ИЛИ, ИЛИ-НЕ (панель «Logic Gates/2- Input AND, NAND, OR, NOR Gates»)

Двухпозиционные переключатели (панель «Basic/Switch») Дешифратор (панель «Digital/DEC/Generic 8-to-1 DEMUX») Мультиплексор (панель «Digital/MUX/ Generic 1-of 8 MUX»)

**Краткие теоретические сведения:**

**1. Комбинационные схемы**

*Комбинационной схемой* называется логическая схема, реализующая однозначное соответствие между значениями входных и выходных сигналов. Для реализации комбинационных схем используются логические элементы, выпускаемые в виде интегральных схем. В этот класс входят интегральные схемы дешифраторов, шифраторов, мультиплексоров, демультиплексоров и сумматоров.

2. **Шифраторы**

*Шифратор* – логическая комбинационная схема, которая имеет $2^n$ входов (где $n$ – число информационных выходов). Часто $n=3$, тогда $2^n=8$. Подаче на один из входов активного сигнала будет соответствовать двоичное число, которое можно сформировать из его $n$ выходов, эквивалентное номеру входа, на котором появился активный уровень.

*Примечание*: формирование двоичного числа на выходе шифратора означает следующее: каждый из выходов шифратора считается определенным разрядом искомого двоичного числа. Принцип работы шифратора противоположен принципу работы дешифратора, который подробно рассмотрен ниже.

Простейшим случаем применения шифратора может служить, например, схема отслеживание нажатия одной кнопки. Каждая кнопка представляет собой элементарный переключатель. Пусть таких кнопок всего 8, и они пронумерованы начиная с «0» и заканчивая «7». При нажатии определенной кнопки (например «3») формируется сигнал на входе шифратора (каждый вход – это вполне определенная кнопка), в итоге на выходе шифратора можно получить двоичный сигнал равный номеру нажатой кнопки. В нашем случае число выходов равно 3 ($2^3=8$), и на каждом из выходов получится следующая комбинация на 3-м = 0, на 2-м = 1, на 1-м = 1. Полученное двоичное число «011», которое в десятичном коде равно 3.

3. **Дешифраторы**

*Дешифратор* – логическая комбинационная схема, которая имеет $n$ информационных входов и $2^n$ выходов. Каждой комбинации логических уровней на входах будет соответствовать активный уровень на одном из $2^n$ выходов. Обычно $n$ равно 2, 3 или 4. В отечественной литературе входы

дешифратора принято обозначать (1, 2, 4, 8, …), в англоязычной (*A, B, C*, …). Работа дешифратора может быть проиллюстрирована в соответствии с таблицей истинности 1.

Таблица 1 – Таблица истинности дешифратора

| **Входы** | | | | **Выходы** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **8** | **4** | **2** | **1** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

На рис. 1 представлен дешифратор, имеющий таблицу истинности 1. Условное обозначения базовых элементов соответствует обозначению, принятому в отечественной литературе.

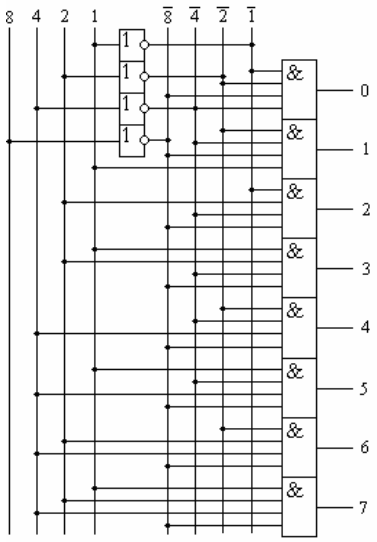



Рис. 1 – Дешифратор

В дальнейшем для детального рассмотрения дешифратора, мы будем пользоваться обозначением (микросхем и базовых элементов) принятых в EWB. На рис. 2 изображен дешифратор (блок микросхемы, использующийся для моделирования дешифратора в EWB) с $n = 3$.

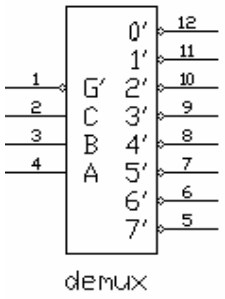



Рис. 2 – Условное обозначение

Активным уровнем сигнала данной микросхемы является уровень логического нуля. То есть в отличие от таблицы истинности 1, у данного дешифратора на выходах, по диагонали расположены нули, а остальные элементы равны единице. На входы *C, B, A* можно подать следующие комбинации логических уровней: «000», «001», «010», …, «111», всего 8 комбинаций. Схема имеет 8 выходов, на одном из которых формируется низкий потенциал, на остальных - высокий. Номер этого единственного выхода, на котором формируется активный (нулевой) уровень, соответствует числу *N*, определяемому состоянием входов *C, B, A* следующим образом:

$$N = C \cdot 2^2 + B \cdot 2^1 + A \cdot 2^0 \qquad (1)$$

Например, если на входы подана комбинация логических уровней «011», то из восьми выходов микросхемы (рис. 2) на выходе с номером

*N=3* (в двоичном исчислении 3 = 011) установится нулевой уровень сигнала (*Y3=0*), а все остальные выходы будут иметь уровень логической единицы. Этот принцип формирования выходного сигнала можно описать следующим образом:

$$Y_i = \begin{cases} 0, & если \quad i = k; \\ 1, & если \quad i \neq k; \\ k = 2^2 \cdot C + 2^1 \cdot B + 2^0 \cdot A. \end{cases} \qquad (2)$$

Помимо информационных входов *A,B,C* дешифраторы обычно имеют дополнительные входы управления *G*. Сигналы на этих входах, например, разрешают функционирование дешифратора или переводят его в пассивное со- стояние, при котором, независимо от сигналов на информационных входах, на всех выходах установится уровень логической единицы. Можно сказать, что существует некоторая функция разрешения, значение которой определяется состояниями управляющих входов.

Разрешающий вход дешифратора может быть прямым или инверсным. У дешифраторов с прямым разрешающим входом активным уровнем является уровень логической единицы, у дешифраторов с инверсным входом – уровень логического нуля. На рис. 3 представлен дешифратор с одним инверсным входом управления. Принцип формирования выходного сигнала в этом дешифраторе с учетом сигнала управления описывается следующим образом:

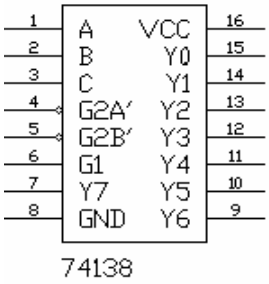



Рис. 3

$$Y_i = \begin{cases} \overline{1 \cdot \overline{G}}, & если \quad i = k; \\ 1, & если \quad i \neq k; \\ k = 2^2 \cdot C + 2^1 \cdot B + 2^0 \cdot A. \end{cases} \qquad (3)$$

У дешифратора с несколькими входами управления функция разрешения, как правило, представляет собой логическое произведение всех разрешающих сигналов управления. Например, для дешифратора серии 74138 с одним прямым входом управления *G1* и двумя инверсными *G2A* и *G2B* (Рис. 4) функции выхода $Y_i$ и разрешения *G* имеют вид:

$$Y_i = \begin{cases} \overline{1 \cdot \overline{G}}, & если \quad i = k; \\ 1, & если \quad i \neq k; \\ k = 2^2 \cdot C + 2^1 \cdot B + 2^0 \cdot A. \end{cases} \qquad (4)$$

$$G = G1 \cdot \overline{G2A} \cdot \overline{G2B} \qquad (5)$$

Обычно входы управления используются для каскадирования (увеличения разрядности) дешифраторов или при параллельной работе нескольких схем на общие выходные линии.

Дешифратор может быть использован как *демультиплексор* – логический коммутатор, подключающий входной сигнал к одному из выходов. В этом случае функцию информационного входа выполняет один из входов разрешения, а состояние входов *C, B* и *A* задает номер выхода, на который передается сигнал с входа разрешения.

4. Мультиплексоры

*Мультиплексор* – комбинационная логическая схема, представляющая собой управляемый переключатель, который подключает к выходу один из входов данных. Номер подключаемого входа равен числу (адресу), определяемому комбинацией логических уровней на адресных входах.

*Демультиплексорами* – называются устройства, которые позволяют подключать один вход к нескольким выходам.

В простейшем случае переключения (коммутацию) можно осуществить при помощи ключей, как это показано на рис. 4.

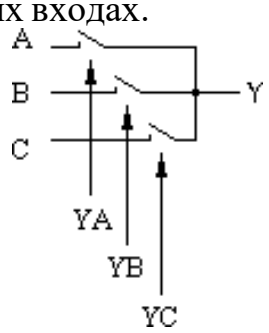


Рис. 4. Мультиплексор на ключах

В цифровых схемах управление ключами осуществляется при помощи логических сигналов. Сами ключи при этом, заменяются соответствующими логическим элементами.

Рассмотрим пример простейшей схемы мультиплексора. Для этого воспользуемся базовым логическим элементом «И» с таблицей истинности 2.

Таблица 2 – Таблица истинности элемента «И»

| **Входы** | | **Выход** |
|---|---|---|
| **X** | **Y** | **Out** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Теперь один из входов элемента будем рассматривать как информационный вход электронного ключа, а другой вход – как адресный. По таблице истинности отчетливо видно, что пока на адресный вход *Y* подан логический уровень «0» сигнал с входа *X* на выход *Out* не проходит. При подаче на адресный вход *Y* логической «1», сигнал, поступающий на вход *X,* поступает на выход *Out*. То есть логический элемент «И» можно использовать в качестве электронного ключа. При этом не важно, какой из входов элемента будет использоваться в качестве адресного входа, а какой – в качестве информационного. Остаётся только объединить выходы элементов «И» на один выход. Это делается при помощи элемента «ИЛИ». Условное обозначение такой схемы приведено на рис. 5.

Чаше всего для управления требуется много входов, поэтому в схему мультиплексора включают дешифратор. Это позволяет управлять переключением входов микросхемы на выход при помощи двоичных кодов. Пример такой схемы приведен на рис. 6.

Для дальнейшего рассмотрения мультиплексоров мы будем, как и прежде пользоваться условными обозначениями логических элементов и микросхем, принятых в EWB.

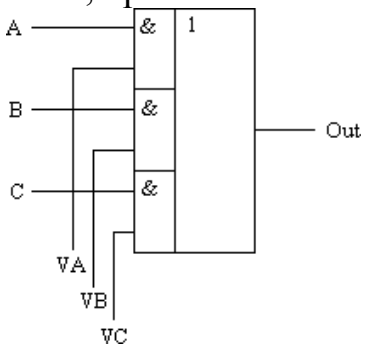



Рис. 5. Принципиальная схема мультиплексора

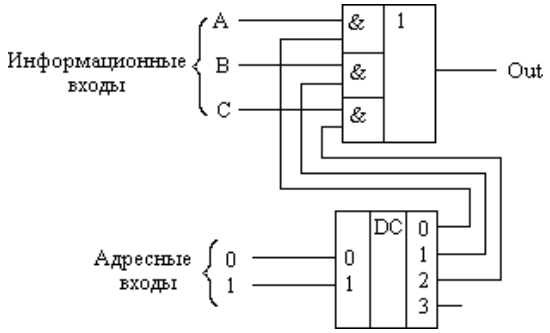



Рис. 6. Мультиплексор, управляемый двоичным кодом

Кроме информационных и адресных входов, схемы мультиплексоров содержат вход разрешения, при подаче на который активного уровня мультиплексор переходит в рабочее состояние. При подаче на вход разрешения пассивного уровня мультиплексор перейдет в нерабочее состояние, в котором сигнал на выходе сохраняет постоянное значение независимо от значений информационных и управляющих сигналов. Число информационных входов у мультиплексоров обычно равно 2, 4, 8 или 16.

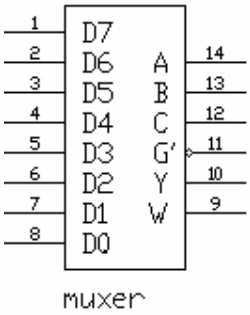



Рис. 7 – Мультиплексор 8x1

На рис. 7 представлен мультиплексор (блок микро- схемы, использующийся для моделирования дешифратора в EWB) 8x1 с инверсным входом разрешения *G*, прямым *Y* и инверсным *W*-выходами ($W = Y$)

5. Реализация логических функций

Функционирование мультиплексора, представленного на рис. 7, описывается характеристическим уравнением, связывающим сигнал на выходе (*Y*) с раз- решающим (*G*), входными информационными (*DO...D7*) и адресными (*A, B, C*) сигналами:

$$Y = \left( \begin{array}{l} \overline{C} \cdot \overline{B} \cdot \overline{A} \cdot D0 + \overline{C} \cdot \overline{B} \cdot A \cdot D1 + \overline{C} \cdot B \cdot \overline{A} \cdot D2 + \overline{C} \cdot B \cdot A \cdot D3 + \\ + C \cdot \overline{B} \cdot \overline{A} \cdot D4 + C \cdot \overline{B} \cdot A \cdot D5 + C \cdot B \cdot \overline{A} \cdot D6 + C \cdot B \cdot A \cdot D7 \end{array} \right) \cdot \overline{G} \qquad (6)$$

Как видно из уравнения, на мультиплексоре можно реализовать логические функции, для чего нужно определить, какие сигналы и логические константы следует подавать на входы мультиплексора.

Логическая функция *n* переменных определена для $2^n$ комбинаций значений переменных. Это позволяет реализовать функцию *n* переменных на мультиплексоре, имеющем *n* адресных и $2^n$ информационных входов. В этом случае каждой комбинации значений аргументов соответствует единственный информационный вход мультиплексора, на который подается значение функции.

Например, требуется реализовать функцию:

$$F1 = \bar{c} \cdot \bar{b} \cdot \bar{a} \vee \bar{c} \cdot \bar{b} \cdot a \vee c \cdot \bar{b} \cdot a \vee c \cdot b \cdot a \qquad (7)$$

Эта функция определена только для 8 комбинаций значений переменных, поэтому для её реализации можно использовать мультиплексор 8x1 с тремя адресными входами. Составим таблицу истинности функции:

Таблица 3 – Таблица истинности логической функции $F_1$

| **Вход мультиплексора** | **Логические переменные** | | | **Выход функции** |
|---|---|---|---|---|
| **N** | **c** | **b** | **a** | **$F_1$** |
| 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 0 |
| 5 | 1 | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |

Из таблицы 3 видно, что для реализации функции на мультиплексоре необходимо подать на информационный вход мультиплексора с номером $N$ сигнал, значение которого равно соответствующему значению функции $F_1$, т. е. на входы с номерами 1, 2, 4, 5 следует подать уровень логического нуля, а на остальные - уровень логической единицы.

Таким образом, при подаче комбинации логических уровней ($a$, $b$, $c$) на адресные входы мультиплексора, к его выходу подключится вход, значение сигнала на котором равно соответствующему значению функции. Схемная реализация приведена на рис. 8.

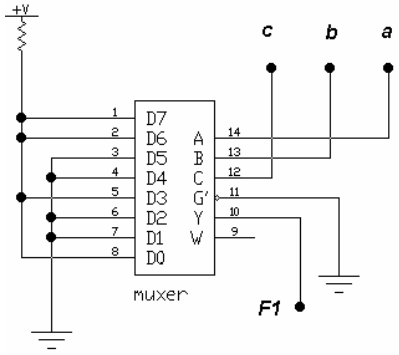



Рис. 8. Реализация логической функции

При реализации логических функций на информационные входы можно подавать не только константы, но и изменяющиеся входные сигналы.

Так, например, рассмотрим другой способ реализации функции $F_1$, определенной выражением (7). Для этого минимизируем выражение функции с помощью известных логических тождеств (см. практическую работу № 1) до вида:

$$F_1 = c \cdot b \cdot a \vee b \cdot c \vee b \cdot a \qquad (8)$$

Составим таблицу истинности функции (8) в зависимости от значений переменных $a$ и $b$ (см. таблицу 4).

Для составления таблицы в выражение (8) подставлялись комбинации $a$ и $b$ и, пользуясь логическим тождествами (см. практической работу № 1) получалось значение функции $F_1$. Заданную такой таблицей функцию реализуют, как и в предыдущем случае, подав на вход с номером $N$ сигнал,

значение которого соответствует значению функции $F_1$.

Таблица 4 – Таблица истинности упрощенной логической функции $F_1$

| **Вход мультиплексора** | **Логические переменные** | | | **Выход функции** |
|---|---|---|---|---|
| **N** | **b** | **a** | **$F_1$** | |
| 0 | 0 | 0 | *c* | |
| 1 | 0 | 1 | 0 | |
| 2 | 1 | 0 | *c* | |
| 3 | 1 | 1 | 1 | |

В данном случае сигналы *c* и *c* соответствующие переменной *c*, подаются на информационные входы, как указано в таблице истинности. При этом сокращается число управляющих входов.

Схемная реализация такого способа задания функции приведена на рис. 9.

Так как используются только два адресных входа, вход. С остается не подключенным. При этом состояние информационных входов *D4...D7* безразлично. Схема рис. 9, по существу, представляет собой мультиплексор 4x1 с двумя адресными и четырьмя информационными входами.

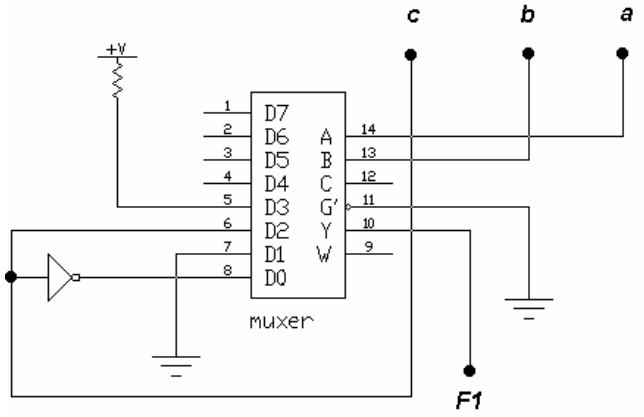



Рис. 9. Реализация упрощенной логической функции

Если функцию можно представить в виде произведения одночлена на многочлен, то её также можно реализовать при помощи мультиплексора. Как следует из уравнения мультиплексора, сигнал, соответствующий одночлену, нужно подать на вход разрешения. Например, требуется реализовать функцию $F_2$, описываемую следующим выражением:

$$F_2 = x \cdot (d \cdot c \cdot b \vee d \cdot b \cdot a \vee e \cdot c \cdot b \cdot a \vee c \cdot b \cdot a) \qquad (9)$$

При реализации данной функции на мультиплексоре сигнал, соответствующий переменной *x*, следует подать на его разрешающий вход. Рассмотрим, какие сигналы необходимо подать на управляющие входы мультиплексора. Выражение в скобках можно рассматривать как некоторую функцию *f* пяти переменных: *a, b, c, d, e*, из которых наиболее часто используются переменные *a, b* и *c*. Поэтому сигналы, соответствующие этим переменным, нужно подать на адресные входы мультиплексора.

Определим, какие сигналы следует подать на информационные входы, чтобы реализовать функцию *f*. Для этого составим таблицу истинности функции в зависимости от значений переменных *a, b* и *c* (таблица 5).

Таблица 5 – Таблица истинности логической функции $F_2$

| Вход мультиплексора | Логические переменные | | | Выход функции |
|---|---|---|---|---|
| **N** | **c** | **b** | **a** | ***f*** |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | *d* |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | *e* |
| 4 | 1 | 0 | 0 | *d* |
| 5 | 1 | 0 | 1 | *d* |
| 6 | 1 | 1 | 0 | 0 |
| 7 | 1 | 1 | 1 | 1 |

Из таблицы видно, что на информационные входы с номерами $N$ = 0, 2, 6 нужно подать уровень логического нуля. Сигнал, соответствующий переменной *d*, нужно подать на входы с номерами $N$ = 1, 4, 5, сигнал, соответствующий переменной *e*, – на вход с номером 3. Схемная реализация такого способа задания функции приведена на рис. 10.

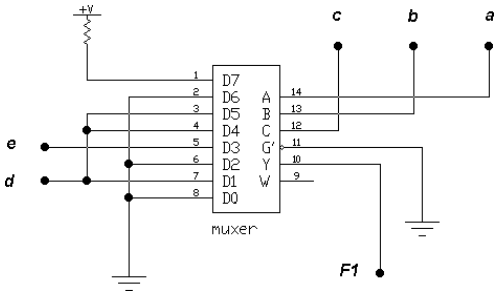


Рис. 10. Реализация логической функции $F_2$

**Порядок работы:**

**Задание 1.** Исследование работы шифратора Создайте схему изображенную на рис. 1. а.

Над генератором слов написаны цифры от 0 до 7 – они обозначают номера входов шифратора, на которые соответственно подаются сигналы управления. Сам шифратор составлен из трех элементов «ИЛИ». Выходы шифратора обозначаются как *A, B, C*. Где *A* – старший бит, *B* – средний бит, а *C* – младший бит двоичного числа, получаемого на выходе. (Это число показывает, на какой из входов подан логический сигнал).

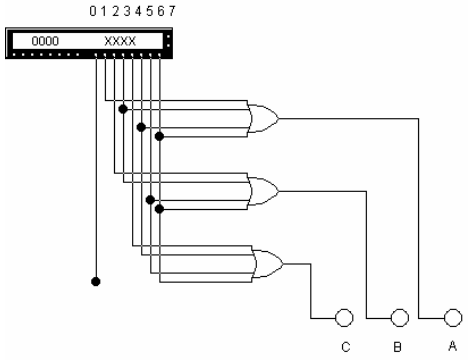


Рис. 1.а – Схема шифратора

Запрограммируйте генератор слов, так что бы на его выходах сформировалась двоичная последовательность, эмулирующая поочередную подачу на вход шифратора сигнала логической единицы. Пошагово изменяя значения входов дешифратора (кнопкой «Step» в генераторе слов), заполните таблицу истинности шифратора (таблица 1.а).

Таблица 1.а – Таблица истинности шифратора

| Входы шифратора | | | | | | | | | Выходы шифратора | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | Двоичное число | | Десятичное число |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **C** | **B** | **A** | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | | | | |

| Входы шифратора | | | | | | | | | Выходы шифратора | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | Двоичное число | | Десятичное число |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **C** | **B** | **A** | |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | | | | |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | | | | |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | | | | |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | | | | |

Переведите полученное двоичное число, составленное из разрядов *C B A* в десятичное. Сделайте вывод о работе шифратора.

**Задание 2**. Исследование работы дешифраторов *а) Исследование развернутой схемы дешифратора* Создайте схему изображенную на рис. 2. а.

Здесь представлен дешифратор 3*8 (3 входа, 8 выходов). Дешифратор составлен из элементов «И-НЕ». *C, B, A* – входы дешифратора, 0, 1, …, 7 – выходы дешифратора, *G* – вход разрешения.

Запрограммируйте генератор слов так, чтобы на его выходе сформировать все возможнее комбинации трехразрядного двоичного числа. Подавая на вход дешифратора различные комбинации двоичного числа *C, B, A* (кнопкой «Step» в генераторе слов) и разрешения *G* (ключом G), заполните таблицу истинности дешифратора (таблица 2.а).

Сделайте вывод о работе дешифратора.

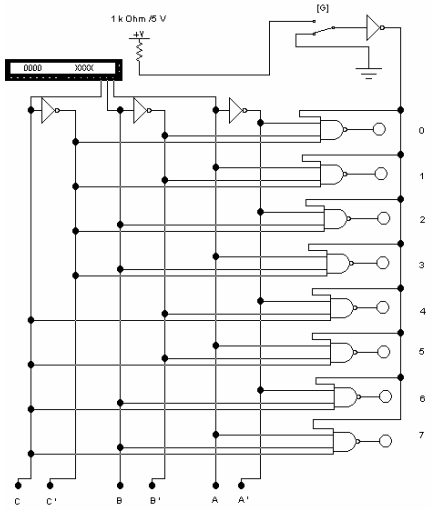



Рис. 2.а – Развернутая схема дешифратора

Таблица 2.а – Таблица истинности развернутой схемы дешифратора

| Входы дешифратора | | | | | Выходы дешифратора | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Число** | **C** | **B** | **A** | **G** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| 0 | 0 | 0 | 0 | 0 | | | | | | | | |
| 1 | 0 | 0 | 1 | 0 | | | | | | | | |
| 2 | 0 | 1 | 0 | 0 | | | | | | | | |
| 3 | 0 | 1 | 1 | 0 | | | | | | | | |
| 4 | 1 | 0 | 0 | 0 | | | | | | | | |
| 5 | 1 | 0 | 1 | 0 | | | | | | | | |
| 6 | 1 | 1 | 0 | 0 | | | | | | | | |
| 7 | 1 | 1 | 1 | 0 | | | | | | | | |
| 0 | 0 | 0 | 0 | 1 | | | | | | | | |
| 1 | 0 | 0 | 1 | 1 | | | | | | | | |
| 2 | 0 | 1 | 0 | 1 | | | | | | | | |
| 3 | 0 | 1 | 1 | 1 | | | | | | | | |
| 4 | 1 | 0 | 0 | 1 | | | | | | | | |
| 5 | 1 | 0 | 1 | 1 | | | | | | | | |
| 6 | 1 | 1 | 0 | 1 | | | | | | | | |
| 7 | 1 | 1 | 1 | 1 | | | | | | | | |

*б) Исследование работы схемы дешифратора 3*8 в основном режиме*

Создайте схему изображенную на рис. 2.б

Запрограммируйте генератор слов аналогично пункту *а*. Подавая на вход дешифратора различные комбинации двоичного числа *C, B, A* (кнопкой «Step» в генераторе слов) и разрешения *G* (ключом *G*), заполните таблицу истинности дешифратора (таблица 2.б, аналогично таблице 2.а).

Сделайте вывод о работе дешифратора. Сравните таблицы 2.а и 2.б.

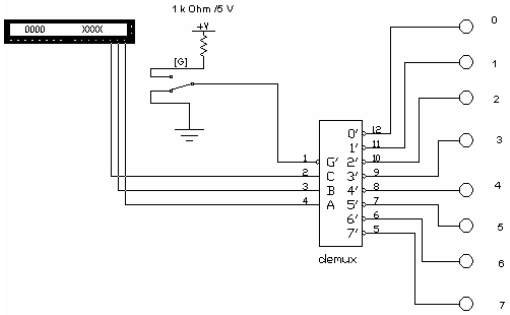



Рис. 2.б. Схема дешифратора

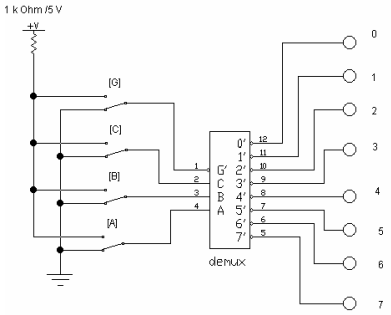



Рис. 2.в. Дешифратор в режиме 2*4

*в) Исследование работы схемы дешифратора 3*8 в режиме2*4*

Создайте схему изображенную на рис. 2.в.

В схеме рис. 2. подключите вход *C* к общему проводу (земле), задав *C=0*. Изменяя сигналы на входах *B* и *A,* пронаблюдайте уровни сигналов на выходах схемы с помощью пробников. Укажите выходы, на которых уровень сигнала не меняется.

Подключите вход. С к источнику питания (логической единицы) задав *C=1*. Аналогично изменяя сигналы на входах *B* и *A,* пронаблюдайте уровни сигналов на выходах схемы, с помощью пробников. Укажите выходы, на которых уровень сигнала не меняется.

Заземлите вход *B* (*B=0*), подавая на входы *A* и *C* все возможные комбинации логических уровней, сделайте вывод о работе схемы в этом случае.

Сформулируйте принцип, по которому можно использовать дешифраторы на меньшую разрешающую способность, чем на ту которую он рассчитан.

**Задание 3**. Применение дешифраторов

Применение дешифраторов в цифровой технике весьма различно. Наиболее часто они используются как формирующие элементы, например, в схемах различных микроконтроллеров для формирования сигнала выбора определенной микросхемы. В этом случае на входы дешифратора подаются сигналы с шины адреса микропроцессора. При этом каждому участку адресного пространства ставится в соответствие определенное назначение (например, для

ОЗУ выделяется первых 1024 байт, для ПЗУ - следующие 2048 байт и так далее). Дешифратор, в таком случае, помогает сформировать сигнал управления (выбора микросхемы), так как согласно его таблице истинности на каждом выходе активный уровень формируется лишь однажды, при

вполне определенной комбинации входных сигналов. Таким образом, не составляет особого труда сначала составить карту памяти (это подробная запись всего содержимого адресного пространства, с записью начального и конечного адреса каждого блока), а затем по ней выделить те разряды адреса, которые однозначно определяют обращение к тому или иному блоку адресного пространства. Именно эти разряды и будут являться входами для дешифраторов. Выход каждого дешифратора будет соединен с входом разрешения работы той микросхемы, которая в данном случае необходима.

В данном задании внимание в основном будет акцентировано на другом способе применения дешифраторов, который, в общем, является базовым и для всех других. Это использование дешифратора в совокупности с логическими элементами.

Помимо этого, в основном данном задании будет введено понятие временной диаграммы, которая также очень часто используется в технике для иллюстрации работы цифровых устройств.

*Временная диаграмма* в общем случае представляет собой график, по оси абсцисс которой откладывается время в тактах, а по оси ординат необходимая цифровая величина (вход или выход цифровой схемы), принимающее значения «0» или «1».

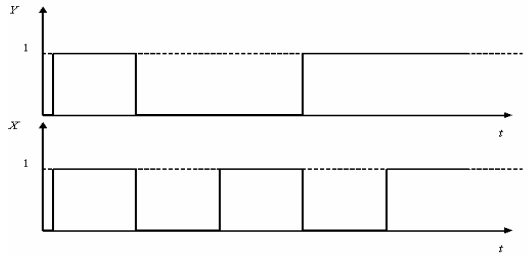



Рис. 3.а. Временная диаграмма

Необходимо отметить, что чаще всего осей ординат несколько, и они располагаются друг под другом. Собственно говоря, именно в этом и состоит основное преимущество применения временных диаграмм, работа цифрового устройства наглядно представляется во времени. Простейший пример временной диаграммы приведен на рис. 3.а.

По таблице истинности логической функции можно без труда построить временную диаграмму. В *EWB* кроме инструментов автоматического построения таблицы истинности присутствуют инструменты и для построения временных диаграмм. Это построение можно осуществить с помощью «Logic Analyzer». Общий вид окна свойств данного инструмента приведено на рис. 3.б (окно свойств вызывается двойным щелчком по изображению).

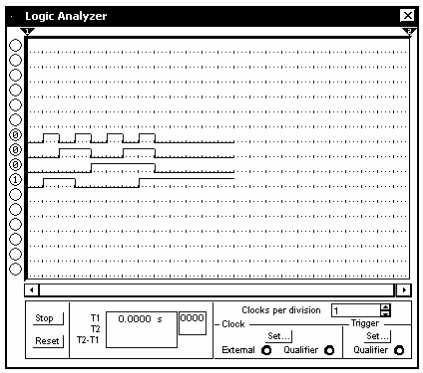



Рис. 3.б – Окно свойств Логического Анализатора

Построение временных диаграмм начинается автоматически при включении всей схемы. В меню «Clock per division» можно задавать масштаб просмотра по оси абсцисс (в данном случае времени). В области «Clock» кнопкой «Set» можно изменить установки внутреннего генератора времени (ось абсцисс), в частности задать частоту анализа («Internal clock rate»). В области «Trigger» можно изменить установки триггера (ось ординат).

Соберите схему изображенную на рис. 3.в.

Запрограммируйте генератор слов так, чтобы на его выходе сформировать все возможные комбинации трехразрядного двоичного числа.

Установите ключ *G* в разрешающее положение. Подавая на вход дешифратора различные комбинации двоичного числа *C, B, A* (кнопкой «Step» в генераторе слов), постройте временную диаграмму работы дешифратора с логическими элементами на выходе. Оси диаграммы выберите согласно рис. 3.г.

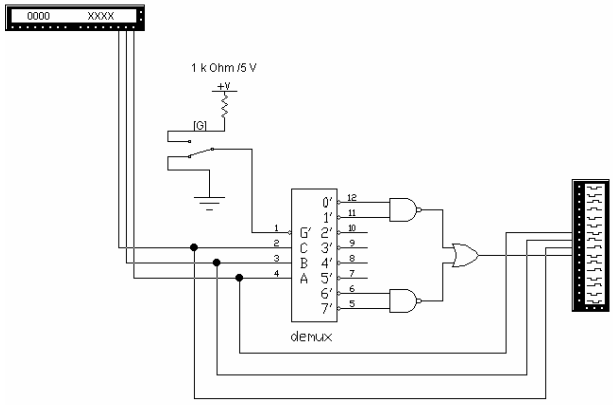



Рис. 3.в – Дешифратор с логическим элементами

Получите логическую функцию данного цифрового устройства. Для этого удобнее предварительно построить таблицу истинности, а по ней аналогично пунктам, изложенным выше построить саму логическую функцию.

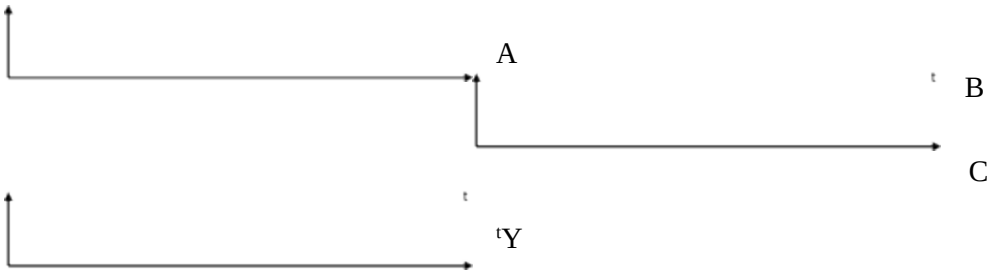



Рис. 3.г. Временная диаграмма

**Задание 4**. Исследование работы мультиплексора *а) Исследование развернутой схемы мультиплексора* Создайте схему изображенную на рис. 4.а.

Здесь, на рис. 4.а приняты следующие условные обозначения: *G* – сигнал разрешения, *d0, d1* – информационные входы, *A* – адресный вход.

Исследуйте поведение схемы мультиплексора, задавая различные сочетания логических уровней на входе схемы, заполнив при этом таблицу истинности, приведенную ниже (таблица 4.а).

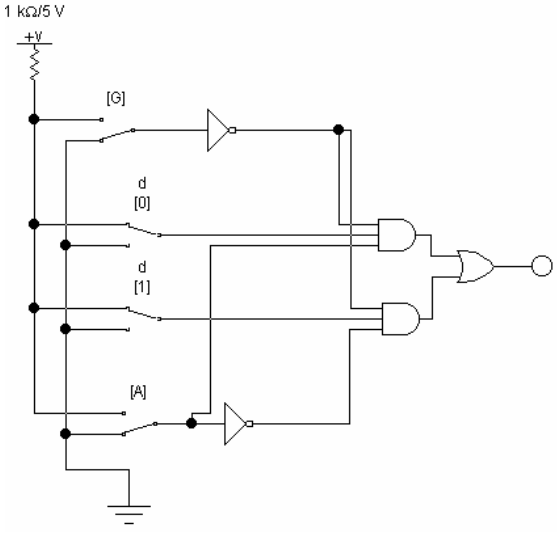



Рис. 4.а – Развернутая схема мультиплексора

Таблица 4.а – Развернутая схема мультиплексора

| Входы | | Адрес | Выход |
|---|---|---|---|
| **d0** | **d1** | **A** | **Q** |
| 0 | 0 | 0 | |
| 0 | 1 | | |
| 1 | 0 | | |
| 1 | 1 | | |
| 0 | 0 | 1 | |
| 0 | 1 | | |
| 1 | 0 | | |
| 1 | 1 | | |

*б) Исследование работы схемы мультиплексора 3*8 в основном режиме*

Соберите схему представленную на рис. 4.б.

Запрограммируйте генератор слов так, чтобы на адресные входы мультиплексора (*C, B, A*) подавались все возможные комбинации логических уровней.

Подавая на адресный вход мультиплексора различные комбинации адреса *C, B, A* (кнопкой «Step» в генераторе слов), заполните таблицу истинности мультиплексора (таблица 4.б).

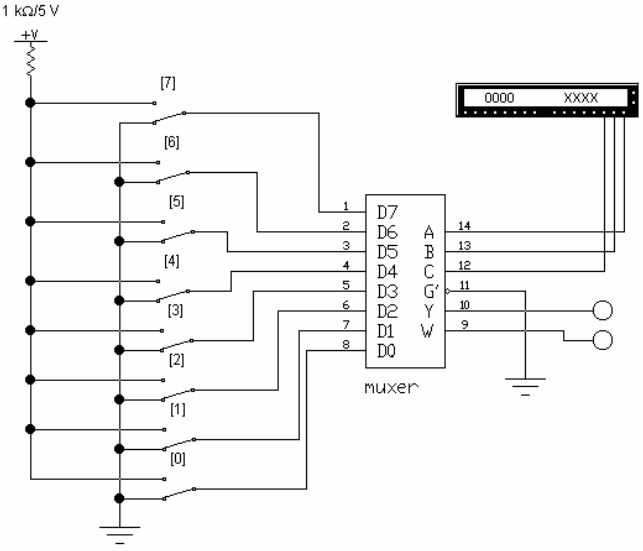



Рис. 4.б. Схема мультиплексора *3*8*

Таблица 4.б – Таблица истинности мультиплексора

| Входы | | | | | | | | | | | Выход | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **d0** | **d1** | **d2** | **d3** | **d4** | **d5** | **d6** | **d7** | **C** | **B** | **A** | **Y** | **W** |
| | | | | | | | | 0 | 0 | 0 | | |
| | | | | | | | | 0 | 0 | 1 | | |
| | | | | | | | | 0 | 1 | 0 | | |
| | | | | | | | | 0 | 1 | 1 | | |
| | | | | | | | | 1 | 0 | 0 | | |
| | | | | | | | | 1 | 0 | 1 | | |
| | | | | | | | | 1 | 1 | 0 | | |
| | | | | | | | | 1 | 1 | 1 | | |

При этом комбинации соответствующих сигналов на входе *D0, D1, …,* D*7* выбираются в соответствии с вариантом, заданным таблицей 4.в.

По таблице 4.б проследите правильность работы мультиплексора.

**Задание 5**. Реализация логической функции с помощью мультиплексора.

По заданной таблице истинности (таблица 5.а) реализуйте логическую

функцию с помощью мультиплексора. Выбор значений выхода логической функции производится согласно вашему варианту. Реализация логической функции может быть осуществлена с помощью любого стандартного мультиплексора.

Таблица 4.в – Варианты для входов данных таблицы 4.б

| **Входы** | | | | | | | | **Вариант** |
|---|---|---|---|---|---|---|---|---|
| **d0** | **d1** | **d2** | **d3** | **d4** | **d5** | **d6** | **d7** | |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 2 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 3 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 4 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 5 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 6 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 7 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 8 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 9 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 10 |

Таблица 5.а – Варианты таблицы истинности

| **Значения логических переменных (для всех вариантов)** | | | **Варианты задания** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| **A** | **B** | **D** | **Значения логической функции (для каждого варианта)** | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |

Создайте схему с вашим вариантом реализации логической функции в EWB, занесите полученную схему в отчет.

С помощью Логического Анализатора постройте временные диаграммы работы вашей логической функции. По ним проверьте правильность функционирования схемы.

**Контрольные вопросы:**

1. В чем состоят достоинства и недостатки различных типов дешифраторов;
2. Сравнить быстродействие и аппаратурные затраты различных методов конструирования дешифраторов;
3. Как синтезируется преобразователь кодов;
4. Синтез шифратора;

# ПРАКТИЧЕСКАЯ РАБОТА № 18

## СУММАТОРЫ

**Цель:** приобрести практические навыки в изучении сумматоров.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации.

**Цель работы:**

1. Исследование сумматоров и полусумматоров.
2. Изучение структуры и исследование работы суммирующих и вычитающих счетчиков, счетчиков с измененным коэффициентом пересчета.
3. Изучение регистров.

**Приборы и элементы:**

Генератор слов (панель «Instruments/Word Generator») Логические пробники (панель «Indicators/Red probe») Источник напряжения + 5 В (панель «Basic/Pull-Up Resistor») Земля (панель «Sources/Ground»)

Двухпозиционные переключатели (панель «Basic/Switch»)

Двухвходовые элементы И, И-НЕ, ИЛИ, ИЛИ-НЕ (панель «Logic Gates/2- Input AND, NAND, OR, NOR Gates»)

Сумматор (панель «Digital/Half-Adder») D-триггер (панель «Digital/»)

Декодер (панель «Indicators/Decoded 7 segment display)

**Краткие теоретические сведения:**

**Сумматоры и полусумматоры**

Широкое применение в цифровой технике находят элементы, выполняющие различные арифметические действия. Операция суммирования – базовая арифметическая операция в двоичной алгебре. Поэтому для дальнейшего изучения цифровой техники необходимо исследовать способы получения сумматоров.

**1. Сумматоры по модулю два**

Построение двоичных сумматоров обычно начинается с сумматора по модулю 2. Ниже представлена таблица истинности этого сумматора.

Таблица 1 – Таблица истинности сумматора по модулю два

| **Входы** | | **Выход** |
|---|---|---|
| **X** | **Y** | **Out** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Из таблицы 1 видно, что логическая функция, выражающая принцип работы сумматора по модулю два, имеет вид:

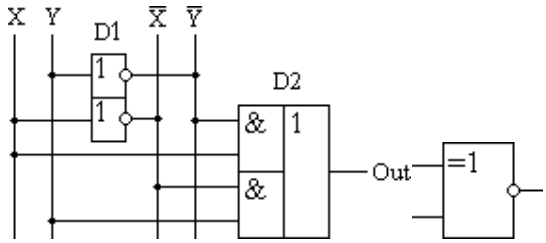
$$F = X \oplus Y = X\bar{Y} \vee \bar{X}Y \quad (1)$$

и представляет собой описанную ранее (формула 7, Практическая работа №1) функцию «исключающего ИЛИ».

На рис. 1 представлена схемная реализация сумматора по модулю два, составленная по таблице истинности 1. На рис. 2 приведено условное обозначение этой же схемы в виде одного элемента – «Исключающего ИЛИ».

Однако представленная схема имеет недостаток, который можно увидеть в таблице 1 – не учет переноса при сложении *X =1* и *Y =1*.

Сумматор по модулю 2 выполняет суммирование без учёта переноса. В обычном двоичном сумматоре требуется учитывать перенос, поэтому требуются схемы, позволяющие формировать перенос в следующий двоичный разряд.




1. Расширенная схема 2) «Исключающее ИЛИ»

Рис. 1. Сумматор по модулю два

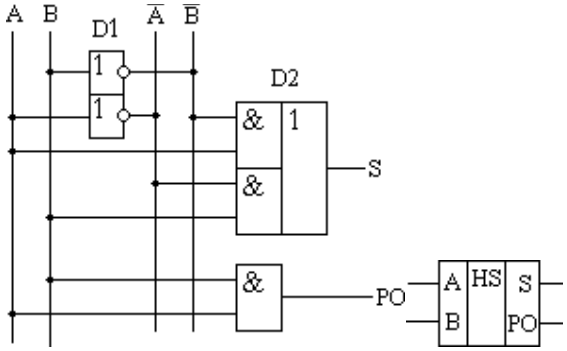
2. **Полусумматоры**

Таблица истинности полусумматора приведена ниже.

Таблица 2 – Таблица истинности полусумматора

| **Входы** | | **Выходы** | |
|---|---|---|---|
| **A** | **B** | **S** | **PO** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

На рис. 2 представлена соответствующая схемная реализация полусумматора на базе логических элементов (рис. 2.1) и в виде одного устройства (рис. 2.2).




1. Расширенная схема 2) Условное обозначение

Рис. 2 – Схема полусумматора

Схема полусумматора формирует перенос в следующий разряд (*PO*), но не может учитывать перенос из предыдущего разряда, поэтому она и называется полусумматором. Для реализации же полного суммирования

(пусть пока и одноразрядного) необходимо помимо формирования переноса в следующий разряд учитывать еще и перенос из предыдущего разряда (это нужно для формирования многоразрядных сумматоров).

2. Изменение коэффициента пересчета

Счетчики характеризуются числом состояний в течение одного периода (цикла). Для схем на рис. 5 и рис. 7 цикл содержит $N=2^3=8$ состояний (от «000» до «111»). Часто число состояний называют *коэффициентом пересчета* $K_{сч}$, который равен отношению числа импульсов $N_c$ на входе к числу импульсов $N_{Qст}$ на выходе старшего разряда за период:

$$Ксч = Nc / NQст. \quad (3)$$

Если на вход счетчика подавать периодическую последовательность импульсов с частотой $f_c$, то частота $f_Q$ на выходе старшего, разряда счетчика будет меньше в $K_{сч}$ раз:

$$Ксч = F_C/F_Q \quad (4)$$

Поэтому счетчики также называют делителями частоты, а величину $K_{сч}$ – коэффициентом деления. Для увеличения величины $K_{сч}$ приходится увеличивать число триггеров в цепочке. Каждый дополнительный триггер удваивает число состояний счетчика и число $K_{сч}$. Для уменьшения коэффициента $K_{сч}$ можно в качестве выхода счетчика рассматривать выходы триггеров промежуточных каскадов.

Например, для счетчика на трех триггерах $K_{сч}=8$, если взять выход 1 и 2-го триггера, то $K_{сч}=4$. При этом $K_{сч}$ является целой степенью числа 2: *2, 4, 8, 16* и т. д.

Можно реализовать счетчик, для которого $K_{сч}$ – любое целое число. Например, для счетчика на трех триггерах можно сделать $K_{сч}$ от «2» до «7», но при этом один или два триггера могут оказаться лишними. При использовании всех трех триггеров можно получить $K_{сч}=5...7$:

$$2^2<Ксч<2^3 \quad (5)$$

Счетчик с $K_{сч}=5$ должен иметь 5 состояний, которые в простейшем случае образуют последовательность: {*0, 1, 2, 3, 4*}. Циклическое повторение этой последовательности означает, что коэффициент деления счетчика равен «5».

Для построения суммирующего счетчика с $K_{сч}=5$ надо, чтобы после формирования последнего числа из последовательности *{O, 1, 2, 3, 4}* счетчик переходил не к числу «5», а к числу «0». В двоичном коде это означает, что от числа «100» нужно перейти к числу «000», а не «101». Изменение естественного порядка счета возможно при введении дополнительных связей между триггерами счетчика. Можно воспользоваться следующим способом: как только счетчик попадает в нерабочее состояние (в данном случае «101»), этот факт должен быть опознан и повлечь последующую выработку сигнала, который перевел бы счетчик в состояние «000». Рассмотрим этот способ более детально.

Факт попадания счетчика в нерабочее состояние описывается логическим уравнением:

$$F = (101) \vee (110) \vee (111) = Q_3 \cdot \overline{Q_2} \cdot Q_1 \vee Q_3 \cdot Q_2 \cdot \overline{Q_1} \vee Q_3 \cdot Q_2 \cdot Q_1 = Q_3 \cdot Q_1 \vee Q_3 \cdot Q_2 \quad (6)$$

Состояния «110» и «111» также являются нерабочими и поэтому учтены при составлении уравнения. Если на выходе эквивалентной логической схемы *F=0,* значит, счетчик находится в одном из рабочих состояний: «0» или «1» или «2» или «3» или «4». Как только он попадает в одно из нерабочих состояний «5» или «6» или «7», формируется сигнал *F=1*. Появление сигнала *F=1* должно переводить счетчик в начальное состояние «000», следовательно, этот сигнал нужно использовать для воздействия на установочные входы триггеров счетчика, которые осуществляли бы сброс в состояние $Q_1 = Q_2 = Q_3 = 0$ . При реализации счетчика на триггерах с входами установки логическим нулем для сброса триггеров требуется подать на входы сброса сигнал *R'=0,* следовательно, логическую функцию *F* необходимо инвертировать. Для обнаружения факта попадания в нерабочее состояние используем схему, реализующую функцию *F* и выполненную на элементах И-НЕ. Для этого преобразуем выражение для функции:

$$F = \overline{Q_3 \cdot Q_1 \vee Q_3 \cdot Q_2} = \overline{Q_3 \cdot \left(\overline{\overline{Q_1} \cdot \overline{Q_2}}\right)} \quad (7)$$

Соответствующая схемная реализация приведена на рис. 9.

Счетчик будет работать следующим образом: при счете от «0» до «4» все происходит как в обычном суммирующем счетчике с $K_{сч}=8$. Установочные сигналы равны «1» и естественному порядку счета не препятствуют. Счет происходит по положительному фронту импульса на счетном входе *C*.

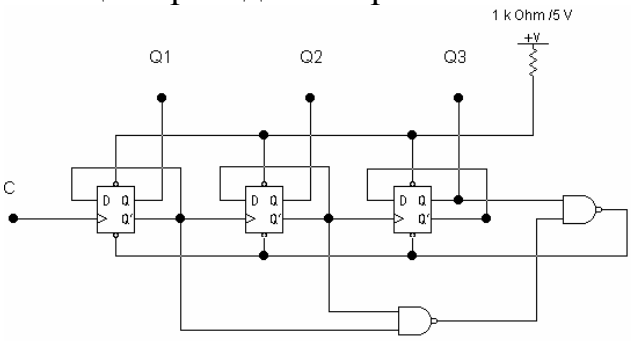



Рис. 9 – Счетчик с измененным коэффициентом пересчета

В тот момент, когда счетчик находится в состоянии «4» («100»), следующий тактовый импульс сначала переводит счетчик в состояние «5» («101»), что немедленно (задолго до прихода следующего тактового импульса) приводит к формированию сигнала сброса, который поступает на установочный вход *R*' триггеров. В результате счетчик сбрасывается в «0» и ждет прихода следующего тактового импульса на счетный вход. Один цикл счета закончился, счетчик готов к началу следующего цикла.

Применяя такие схемы с обратной связью для сброса счетчика, нужно иметь в виду, что операция сброса занимает конечное время, поэтому непосредственно перед сбросом счетчика в «0» на выходе первого триггера появляются кратковременные импульсы, или «иголки». Это не имеет значения при подключении счетчика напрямую к индикатору, но при использовании

этого выхода счетчика в качестве источника тактовых импульсов могут возникнуть определенные проблемы. Схема, в которой это явление устранено, приведена на рис. 3.

Важным отличием является то, что схема обнаруживает не факт попадания в нерабочее состояние «101», а факт попадания в состояние «100» и в следующем такте вырабатывает сигнал сброса.

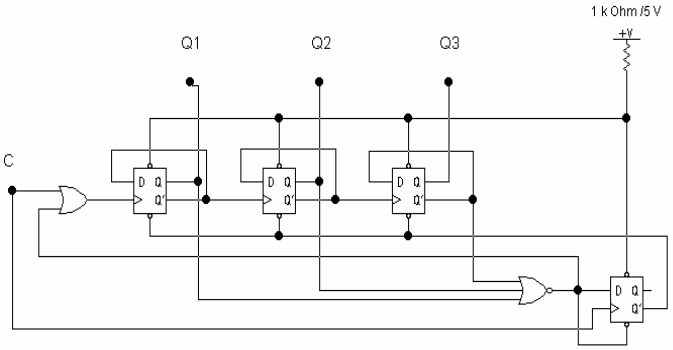


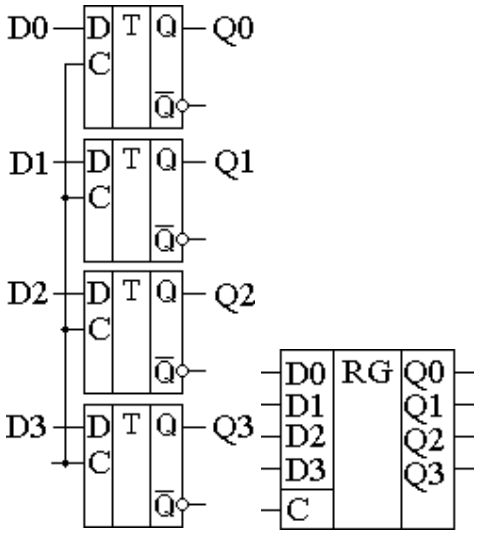

Рис. 3 – Счетчик с измененным коэффициентом пересчета

Регистры

*Регистром* называется последовательное или параллельное соединение триггеров. Регистры обычно строятся на основе *D-триггеров*. При этом для построения регистров могут использоваться как универсальные *D-триггеры*, так и триггеры-защелки.

**1. Параллельный регистр**

*Параллельный регистр* служит для запоминания многоразрядного двоичного слова. При использовании для построения параллельного регистра *триггеров-защелок* регистр называется *регистр-защелка*. Количество триггеров, входящее в состав параллельного регистра, определяет его разрядность. При записи информации в параллельный регистр все биты (двоичные разряды) записываются одновременно. Схема четырёхразрядного параллельного регистра приведена на рисунке 4.1, а его обозначение – на рисунке 4.2.

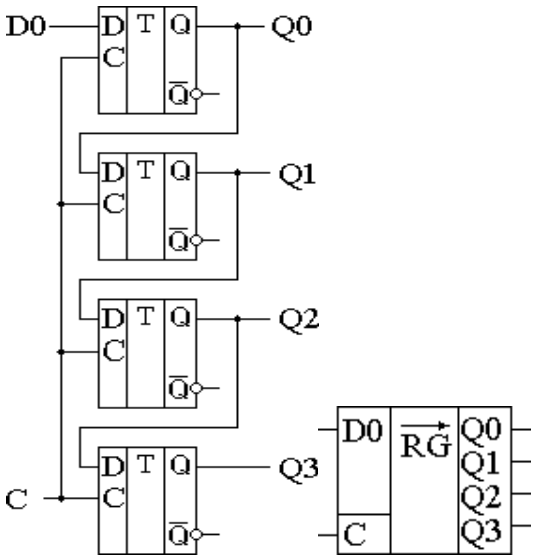



1. Расширенная схема 2) Условное обозначение

Рис. 4 – Схема параллельного регистра

## 2. Последовательный регистр

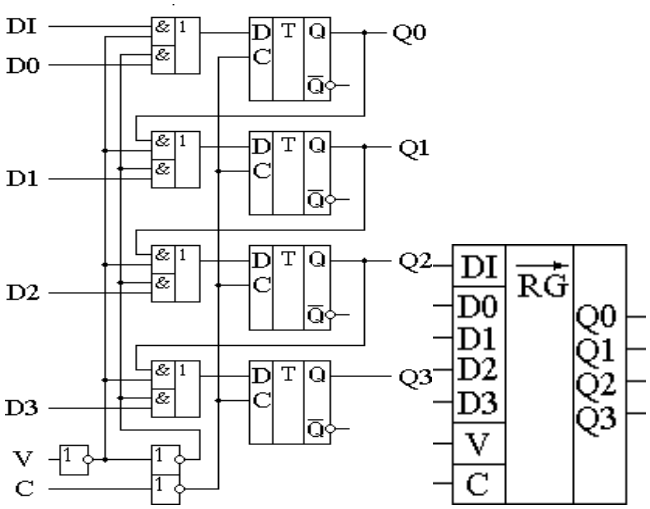
*Последовательный регистр* (регистр сдвига) обычно служит для преобразования последовательного кода в параллельный и наоборот. Схема регистра, осуществляющего преобразование последовательного кода в параллельный, приведена на рисунке 5.1, а его изображение на принципиальных схемах – на рисунке 5.2.




1. Расширенная схема 2) Условное обозначение

Рис. 5 – Схема последовательного регистра

Регистры сдвига выполняют обычно как универсальные *последовательно- параллельные микросхемы*. Переключение регистра из параллельного режима в последовательный и наоборот осуществляется при помощи мультиплексора. Схема такого регистра приведена на рисунке 6.1, а его изображение на принципиальных схемах – на рисунке 6.2.




1. Расширенная схема 2) Условное обозначение

Рис. 6 – Схема универсального регистра

**Порядок работы:**

**Задание 1.** Изучение полусумматоров и сумматоров

*а) Изучение работы полусумматора*

Соберите схему, приведенную на рис. 1.а.

Задавая различные комбинации логических уровней, на входах полусумматора *A* и *B,* заполните таблицу истинности 1.а.

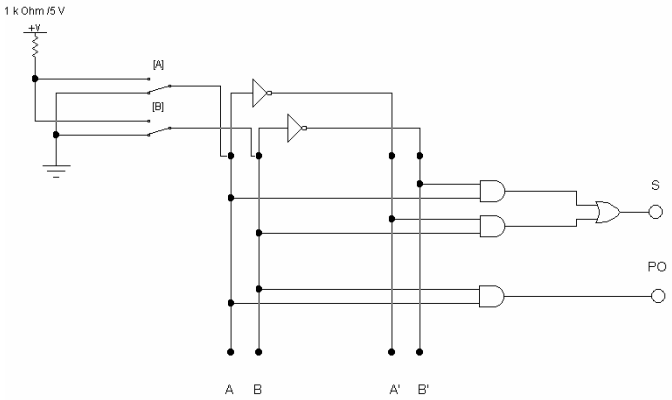



Рис. 1.а – Схема двоичного полусумматора

Таблица 1.а. – Таблица истинности полусумматора

| **Входы** | | **Выходы** | |
|---|---|---|---|
| **A** | **B** | **S** | **PO** |
| 0 | 0 | **0** | **0** |
| 0 | 1 | **1** | **0** |
| 1 | 0 | **1** | **0** |
| 1 | 1 | **0** | **1** |

Сравните полученную таблицу с таблицей 1 краткой теории, сделайте выводы.

*б) Изучение работы одноразрядного сумматора*

Соберите схему, приведенную на рис. 1.б.

Задавая различные комбинации логических уровней на входах сумматора *A* и *B,* а также уровень сигнала переноса из предыдущего разряда *PI* заполните таблицу истинности 1.б.

Сравните полученную таблицу с таблицей 2 краткой теории, сделайте выводы.

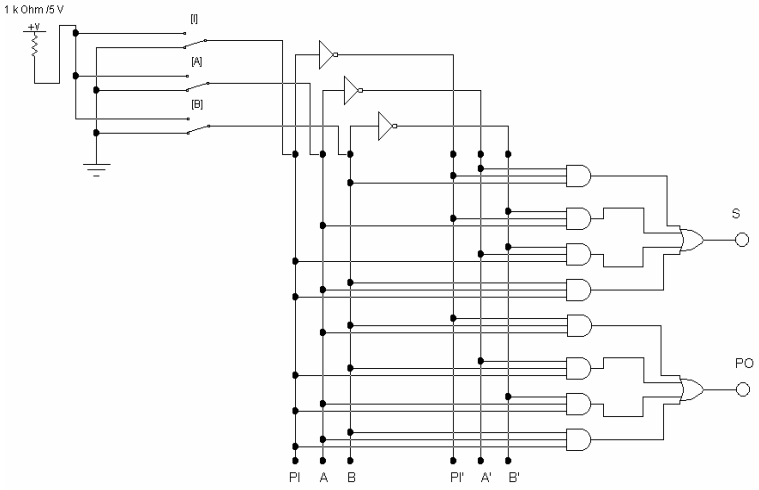



Рис. 1.б. Схема одноразрядного двоичного сумматора

Таблица 1.б. – Таблица истинности одноразрядного сумматора

| **Входы** | | | **Выходы** | |
|---|---|---|---|---|
| **PI** | **A** | **B** | **S** | **PO** |
| 0 | 0 | 0 | **0** | **0** |
| 0 | 0 | 1 | **1** | **0** |
| 0 | 1 | 0 | **1** | **0** |
| 0 | 1 | 1 | **0** | **1** |
| 1 | 0 | 0 | **1** | **0** |
| 1 | 0 | 1 | **0** | **1** |
| 1 | 1 | 0 | **0** | **1** |
| 1 | 1 | 1 | **1** | **1** |

*в) Изучение работы двоичного трехразрядного сумматора*

Соберите схему, аналогичную рис. 1.в.

Здесь в качестве сумматоров используются специальные блоки EWB «Half-Adder» имеющие аналогично условному обозначению рис. 3.2 два входа *A* и *B,* выход сума «S» (в нашем случае на рис. 1.в – «Σ») и сигнал переноса в следующий разряд «PO» (в нашем случае на рис. 1.в – «$C_0$»). В отличие от сумматора, приведенного на рис. 3.2 у данного сумматора нет учета переноса, как из предыдущего разряда, так и в следующий. Для учета переноса из предыдущих разрядов в схеме рис. 1. используются дополнительные сумматоры и логические элементы.

Для проверки правильности функционирования данной схемы необходимо предварительно запрограммировать генератор слов так, чтобы на его используемых выходах формировались все возможные комбинации суммируемых двоичных числе «a2 a1 a0» и «b2 b1 b0».

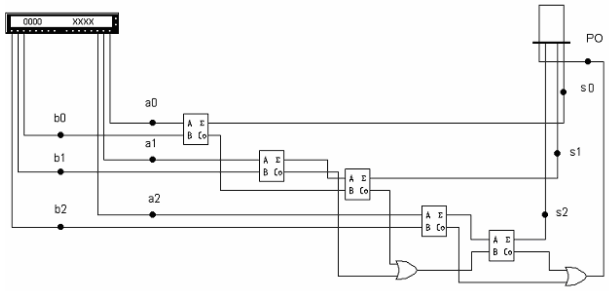



Рис. 1.в – Схема трехразрядного двоичного сумматора

Ниже приведена таблица, поясняющая принцип программирования генератора слов (таблица 1.в).

Таблица 1.в – Программирование генератора слов

| **Шестнадцатеричный код** | **Двоичный код** | |
|---|---|---|
| 0000 | 0000 | 0000 |
| 0001 | 0000 | 0001 |
| 0002 | 0000 | 0010 |
| 0003 | 0000 | 0011 |
| 0004 | 0000 | 0100 |
| 0005 | 0000 | 0101 |
| 0006 | 0000 | 0110 |
| 0007 | 0000 | 0111 |
| 2000 | 0010 | 0000 |
| 2001 | 0010 | 0001 |
| 2002 | 0010 | 0010 |
| 2003 | 0010 | 0011 |
| 2004 | 0010 | 0100 |
| 2005 | 0010 | 0101 |
| 2006 | 0010 | 0110 |
| 2007 | 0010 | 0111 |
| 4000 | 0100 | 0000 |
| … | … | |
| 4007 | 0100 | 0111 |
| 6000 | 0110 | 0000 |
| … | … | |
| 6007 | 0110 | 0111 |
| 8000 | 1000 | 0000 |
| … | … | |
| 8007 | 1000 | 0111 |
| A000 | 1010 | 0000 |
| … | … | |
| A007 | 1010 | 0111 |
| C000 | 1100 | 0000 |
| … | … | |
| C007 | 1100 | 0111 |
| E000 | 1110 | 0000 |
| … | … | |
| E007 | 1110 | 0111 |

Таким образом, из таблицы 1.в видно, что при подключении к трем младшим разрядам генератора слов одного двоичного числа «a2 a1 a0», а к трем старшим разрядам другого двоичного числа «b2 b1 b0» можно просуммировать все возможные комбинации двух трех разрядных чисел.

*Примечание:* генератор слов удобнее программировать в шестнадцатеричном коде (задание 5.б практической работы №1); семисегментный дисплей («Decoded Seven-Segment Display») сразу декодирует двоичное число (переводит двоичное число в шестнадцатеричное) «s2 s1 s0» и отображает его в удобном для анализа виде.

Запрограммировав генератор слов, необходимо изучить работу трехразрядного двоичного сумматора, заполнив таблицу 1.г.

Таблица 1.г – Суммирование в трехразрядном двоичном сумматоре

| **Входы, в различных кодах** | | | | | | | | **Выходы, в различных кодах** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **2ый** | | | | | | **10ый** | **16ый** | **2ый** | | | | **10ый** |
| **a2** | **a1** | **a0** | **b2** | **b1** | **b0** | ***b2b1b0*** | ***s2s1s0*** | **PO** | **s2** | **s1** | **s0** | ***s2s1s0*** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** |
| 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| 0 | 0 | 1 | 0 | 0 | 0 | **0** | **1** | **1** | **0** | **0** | **1** | **1** |
| 0 | 1 | 0 | 0 | 0 | 0 | **0** | **2** | **2** | **0** | **1** | **0** | **2** |
| 0 | 1 | 1 | 0 | 0 | 0 | **0** | **3** | **3** | **0** | **1** | **1** | **3** |
| 1 | 0 | 0 | 0 | 0 | 0 | **0** | **4** | **4** | **1** | **0** | **0** | **4** |
| 1 | 0 | 1 | 0 | 0 | 0 | **0** | **5** | **5** | **1** | **0** | **1** | **5** |
| 1 | 1 | 0 | 0 | 0 | 0 | **0** | **6** | **6** | **1** | **1** | **0** | **6** |
| 1 | 1 | 1 | 0 | 0 | 0 | **0** | **7** | **7** | **1** | **1** | **1** | **7** |
| 0 | 0 | 0 | 0 | 0 | 1 | **1** | **2000** | **1** | **0** | **0** | **1** | **1** |
| 0 | 0 | 1 | 0 | 0 | 1 | **1** | **2001** | **2** | **0** | **1** | **0** | **2** |
| 0 | 1 | 0 | 0 | 0 | 1 | **1** | **2002** | **3** | **0** | **1** | **1** | **3** |
| 0 | 1 | 1 | 0 | 0 | 1 | **1** | **2003** | **4** | **1** | **0** | **0** | **4** |
| 1 | 0 | 0 | 0 | 0 | 1 | **1** | **2004** | **5** | **1** | **0** | **1** | **5** |
| 1 | 0 | 1 | 0 | 0 | 1 | **1** | **2005** | **6** | **1** | **1** | **0** | **6** |
| 1 | 1 | 0 | 0 | 0 | 1 | **1** | **2006** | **7** | **1** | **1** | **1** | **7** |
| 1 | 1 | 1 | 0 | 0 | 1 | **1** | **2007** | **8** | **0** | **0** | **0** | **0** |
| 0 | 0 | 0 | 0 | 1 | 0 | **2** | **4000** | **2** | **0** | **1** | **0** | **2** |
| 0 | 0 | 1 | 0 | 1 | 0 | **2** | **4001** | **3** | **0** | **1** | **1** | **3** |
| 0 | 1 | 0 | 0 | 1 | 0 | **2** | **4002** | **4** | **1** | **0** | **0** | **4** |
| 0 | 1 | 1 | 0 | 1 | 0 | **2** | **4003** | **5** | **1** | **0** | **1** | **5** |
| 1 | 0 | 0 | 0 | 1 | 0 | **2** | **4004** | **6** | **1** | **1** | **0** | **6** |
| 1 | 0 | 1 | 0 | 1 | 0 | **2** | **4005** | **7** | **1** | **1** | **1** | **7** |
| 1 | 1 | 0 | 0 | 1 | 0 | **2** | **4006** | **8** | **0** | **0** | **0** | **0** |
| 1 | 1 | 1 | 0 | 1 | 0 | **2** | **4007** | **9** | **0** | **0** | **1** | **1** |
| 0 | 0 | 0 | 0 | 1 | 1 | **3** | **6000** | **3** | **0** | **1** | **1** | **3** |
| 0 | 0 | 1 | 0 | 1 | 1 | **3** | **6001** | **4** | **1** | **0** | **0** | **4** |
| 0 | 1 | 0 | 0 | 1 | 1 | **3** | **6002** | **5** | **1** | **0** | **1** | **5** |
| 0 | 1 | 1 | 0 | 1 | 1 | **3** | **6003** | **6** | **1** | **1** | **0** | **6** |
| 1 | 0 | 0 | 0 | 1 | 1 | **3** | **6004** | **7** | **1** | **1** | **1** | **7** |
| 1 | 0 | 1 | 0 | 1 | 1 | **3** | **6005** | **8** | **0** | **0** | **0** | **0** |
| 1 | 1 | 0 | 0 | 1 | 1 | **3** | **6006** | **9** | **0** | **0** | **1** | **1** |
| 1 | 1 | 1 | 0 | 1 | 1 | **3** | **6007** | **A** | **0** | **1** | **0** | **2** |

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | **4** | **8000** | **4** | **1** | **0** | **0** | **4** |
| 0 | 0 | 1 | 1 | 0 | 0 | **4** | **8001** | **5** | **1** | **0** | **1** | **5** |
| 0 | 1 | 0 | 1 | 0 | 0 | **4** | **8002** | **6** | **1** | **1** | **0** | **6** |
| 0 | 1 | 1 | 1 | 0 | 0 | **4** | **8003** | **7** | **1** | **1** | **1** | **7** |
| 1 | 0 | 0 | 1 | 0 | 0 | **4** | **8004** | **8** | **0** | **0** | **0** | **0** |
| 1 | 0 | 1 | 1 | 0 | 0 | **4** | **8005** | **9** | **0** | **0** | **1** | **1** |
| 1 | 1 | 0 | 1 | 0 | 0 | **4** | **8006** | **A** | **0** | **1** | **0** | **2** |
| 1 | 1 | 1 | 1 | 0 | 0 | **4** | **8007** | **B** | **0** | **1** | **1** | **3** |
| 0 | 0 | 0 | 1 | 0 | 1 | **5** | **A000** | **5** | **1** | **0** | **1** | **5** |
| 0 | 0 | 1 | 1 | 0 | 1 | **5** | **A001** | **6** | **1** | **1** | **0** | **6** |
| 0 | 1 | 0 | 1 | 0 | 1 | **5** | **A002** | **7** | **1** | **1** | **1** | **7** |
| 0 | 1 | 1 | 1 | 0 | 1 | **5** | **A003** | **8** | **0** | **0** | **0** | **0** |
| 1 | 0 | 0 | 1 | 0 | 1 | **5** | **A004** | **9** | **0** | **0** | **1** | **1** |
| 1 | 0 | 1 | 1 | 0 | 1 | **5** | **A005** | **A** | **0** | **1** | **0** | **2** |
| 1 | 1 | 0 | 1 | 0 | 1 | **5** | **A006** | **B** | **0** | **1** | **1** | **3** |
| 1 | 1 | 1 | 1 | 0 | 1 | **5** | **A007** | **C** | **1** | **0** | **0** | **4** |
| 0 | 0 | 0 | 1 | 1 | 0 | **6** | **C000** | **6** | **1** | **1** | **0** | **6** |
| 0 | 0 | 1 | 1 | 1 | 0 | **6** | **C001** | **7** | **1** | **1** | **1** | **7** |
| 0 | 1 | 0 | 1 | 1 | 0 | **6** | **C002** | **8** | **0** | **0** | **0** | **0** |
| 0 | 1 | 1 | 1 | 1 | 0 | **6** | **C003** | **9** | **0** | **0** | **1** | **1** |
| 1 | 0 | 0 | 1 | 1 | 0 | **6** | **C004** | **A** | **0** | **1** | **0** | **2** |
| 1 | 0 | 1 | 1 | 1 | 0 | **6** | **C005** | **B** | **0** | **1** | **1** | **3** |
| 1 | 1 | 0 | 1 | 1 | 0 | **6** | **C006** | **C** | **1** | **0** | **0** | **4** |
| 1 | 1 | 1 | 1 | 1 | 0 | **6** | **C007** | **D** | **1** | **0** | **1** | **5** |
| 0 | 0 | 0 | 1 | 1 | 1 | **7** | **E000** | **7** | **1** | **1** | **1** | **7** |
| 0 | 0 | 1 | 1 | 1 | 1 | **7** | **E001** | **8** | **0** | **0** | **0** | **0** |
| 0 | 1 | 0 | 1 | 1 | 1 | **7** | **E002** | **9** | **0** | **0** | **1** | **1** |
| 0 | 1 | 1 | 1 | 1 | 1 | **7** | **E003** | **A** | **0** | **1** | **0** | **2** |
| 1 | 0 | 0 | 1 | 1 | 1 | **7** | **E004** | **B** | **0** | **1** | **1** | **3** |
| 1 | 0 | 1 | 1 | 1 | 1 | **7** | **E005** | **C** | **1** | **0** | **0** | **4** |
| 1 | 1 | 0 | 1 | 1 | 1 | **7** | **E006** | **D** | **1** | **0** | **1** | **5** |
| 1 | 1 | 1 | 1 | 1 | 1 | **7** | **E007** | **E** | **1** | **1** | **0** | **6** |

**Задание 2.** Исследование работы суммирующего и вычитающего счетчиков

*а) Исследование первой реализации счетчика*

Соберите схему изображенную на рис. 2.а.

Включите схему. Подавая на вход схемы тактовые импульсы при по мощи ключа. С и наблюдая состояние выходов счетчика при помощи семи- сегментного дисплея декодера и логических пробников, составьте временные диаграммы работы суммирующего (вычитающего) счетчика. Сделайте выводы о работе схемы.

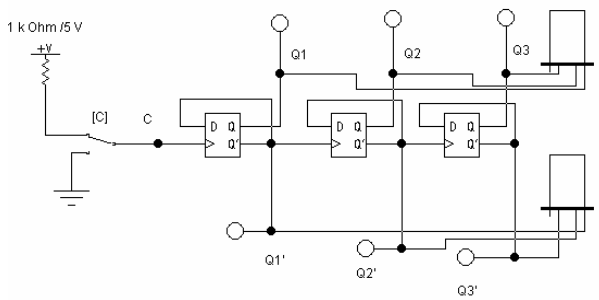



Рис. 2.а. Первая реализация счетчика

*Примечание:* в данном случае семисегментный дисплей предназначен для автоматического перевода поступающего на его вход двоичного числа в шестнадцатеричное и отображение последнего.

*б) Исследование второй реализации счетчика*

Создайте вторую реализацию суммирующего и вычитающего счетчика (см. теорию).

Занесите полученную схему.

Составьте временные диаграммы данной реализации.

**Задание 3**. Исследование счетчика с измененным коэффициентом пересчета

*а) Первый вариант.*

Соберите схему, изображенную на рис. 3.а.

Включите схему. Подавая на вход схемы тактовые импульсы при помощи ключа. С и наблюдая состояние выходов счетчика при помощи логических пробников, составьте временные диаграммы работы счетчика и определите коэффициент пересчета.

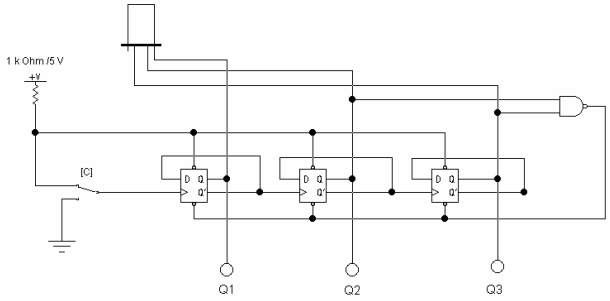



Рис. 3.а – Счетчик с измененным коэффициентом пересчета

*б) Второй вариант.*

Соберите схему изображенную на рис. 3.б.

Включите схему. Подавая на вход схемы тактовые импульсы, при помощи ключа *С* и наблюдая состояние выходов счетчика при помощи логических пробников, составьте временные диаграммы работы счетчика и определите коэффициент пересчета.

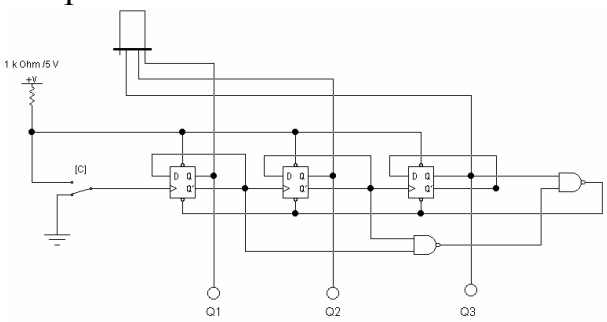



Рис. 3.б – Счетчик с измененным коэффициентом пересчета

**Задание 4**. Исследование регистров *а) Исследование параллельного регистра* Создайте схему, изображенную на рис. 4.а.

Включите схему. Исследуйте работу полученного двоичного четырех-

разрядного регистра, заполнив таблицу истинности, приведенную ниже (таблица 4.а).

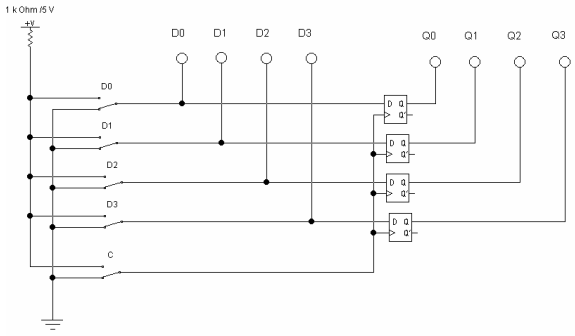



Рис. 4.а – Схема четырехразрядного параллельного регистра

Таблица 4.а – Параллельный регистр

| Входы | | | | Выходы | | | |
|---|---|---|---|---|---|---|---|
| **D0** | **D1** | **D2** | **D3** | **Q0** | **Q1** | **Q2** | **Q3** |
| 0 | 0 | 0 | 0 | | | | |
| 0 | 0 | 0 | 1 | | | | |
| 0 | 0 | 1 | 0 | | | | |
| 0 | 0 | 1 | 1 | | | | |
| 0 | 1 | 0 | 0 | | | | |
| 0 | 1 | 0 | 1 | | | | |
| 0 | 1 | 1 | 0 | | | | |
| 0 | 1 | 1 | 1 | | | | |
| 1 | 0 | 0 | 0 | | | | |
| 1 | 0 | 0 | 1 | | | | |
| 1 | 0 | 1 | 0 | | | | |
| 1 | 0 | 1 | 1 | | | | |
| 1 | 1 | 0 | 0 | | | | |
| 1 | 1 | 0 | 1 | | | | |
| 1 | 1 | 1 | 0 | | | | |
| 1 | 1 | 1 | 1 | | | | |

*Примечание:* запись числа в регистр происходит по переднему фронту разрезающего импульса и сохраняется до появления следующего импульса.

*б) Исследование параллельного регистра*

Создайте схему, изображенную на рис. 4.б.

Включите схему. Согласно вашему варианту из таблицы 4.б необходимо выбрать двоичное число, которое требуется получить на выходе регистра сдвига (последовательного регистра).

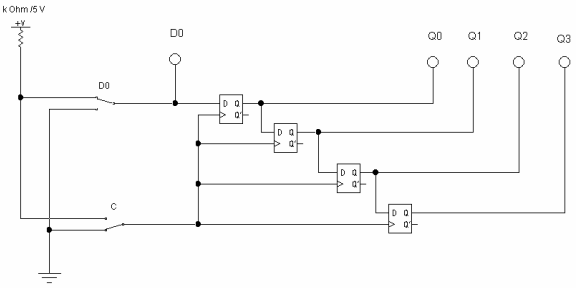



Рис. 4.б – Схема четырехразрядного последовательного регистра

Задавая необходимые комбинации на входе *D0* и занося информацию в регистр с помощью сигнала разрешения *C,* добейтесь получения на выходе *Q0 Q1 Q2 Q3* требуемого числа.

Постройте временную диаграмму получения в регистре сдвига требуемого числа.

Таблица 4.б – Варианты задания числа для последовательного регистра

| **Вариант** | **Число** | | | |
|---|---|---|---|---|
| | **Q0** | **Q1** | **Q2** | **Q3** |
| **1** | **2** | **3** | **4** | **5** |
| 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 0 | 1 |
| 4 | 0 | 1 | 1 | 0 |
| 5 | 0 | 1 | 1 | 1 |
| 6 | 1 | 0 | 0 | 0 |

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 7 | 1 | 0 | 0 | 1 |
| 8 | 1 | 0 | 1 | 0 |
| 9 | 1 | 0 | 1 | 1 |
| 10 | 1 | 1 | 0 | 0 |

**Контрольные вопросы:**

1. Чем отличается полусумматор от одноразрядного сумматора;
2. Сколько входов имеет полный сумматор;
3. Сколько выходов имеет полный сумматор;
4. Запишите функцию алгебры логики, которая реализует арифметическое суммирование одноразрядных двоичных чисел;

# ПРАКТИЧЕСКАЯ РАБОТА № 19

## ТРИГГЕРЫ

**Цель:** приобрести практические в изучении триггеров.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации

**Цель работы:** ознакомится на практике с работой некоторых логических схем и триггеров в интегральном исполнении серии **К 561**.

**Используемое оборудование и аппаратура:**

**1. Источник питания постоянного тока стабилизированный (5 – 12) В**.

2. Цифровые микросхемы **К561ЛЕ5, ЛА7, ЛП2, ТМ2, ТВ1,** подлежащие исследованию, резисторы **10 – 20 кОм**, тумблеры и кнопки.

3. Электронно-лучевой осциллограф или вольтметр постоянного тока с большим входным сопротивлением и пределом измерения, равным напряжению источника питания.

4. Паяльник.

5. Монтажные провода, монтажная плата.

**Теоретические сведения:**

В лабораторной работе исследуются цифровые логические микросхемы технологии КМОП серии **К561: 2ИЛИ-НЕ (ЛЕ5), 2И-НЕ (ЛА7),** исключающие **ИЛИ (ЛП2), Д** – триггер **(ТМ2)** и **ІК** – триггер **(ТВ1).**

Напряжение питания микросхем этой серии может находится в пределах **(13 – 15) В**. Входные и выходные сигналы могут иметь только два дискретных значения: Лог.0 (не более 0,2) В и Лог.1 (практически **Un**). Предельная частота срабатывания микросхем от **1** до **4 МГц** соответственно при питании напряжением **3** и **15В**. В лабораторной работе сигнал Лог1. на входе микросхемы задается подключением входа к **+U**, а сигнал Лог.0 – подключением к вертикали. Во избежание выхода из строя микросхем входы незадействованных элементов микросхемы должны быть подключены к одному

из полюсов источника питания.

При работе с микросхемами **КМОП** технологии следует соблюдать следующие правила. При монтаже микросхемы сначала подключать выводы питания, а потом – логические входы и выходы. Не разрешается подавать сигнал на вход микросхемы, не имеющий напряжения питания. Нельзя оставлять неподключенными входы микросхем. При работе с микросхемами принимать меры для защиты их от статического электричества.

Состояние логических микросхем полностью определяется состоянием сигналов на их входах. Исследование работы микросхем такого типа заключается в задании всех возможных комбинаций состояний логических сигналов на входах микросхемы и определение состояния входа, соответствующего каждой комбинации. Для исследуемых двухвходовых логических схем задаются **4** возможных комбинации состояния входных сигналов:

**0 0**
**0 1**
**1 0**
**1 1**

Триггер микросхемы **К561ТМ2** содержит два установочных входа **R, S**, информационный вход **D** и таковой вход **C**, а также два парафазных выхода **Q, Q.** Установочные **R, S** входы предназначены для установки триггера в одно из 2-ух возможных состояний путем подачи потенциала уровня Лог.1 на один из этих 2-ух входов. Сигнал на тактирующем входе **C** вызывает установку триггера своим положительным фронтом: триггер устанавливается в одно состояние при наличии Лог.1 на входе **D** и в другое состояние при наличии Лог 0 на входе **D** в момент перепада напряжения на тактирующем входе **C**. При этом установочные **R, S** входы являются приоритетными.

**I** – **K** триггер микросхемы **К651ТВ1** содержит 2 установочных **R, S** входа, **2** информационных входа **I** и **K**, а также тактирующий вход **C**. Активностью обладают положительный фронт тактирующего сигнала: Он устанавливает триггер в одно состояние при наличии Лог.1 на входе **К**. При наличии Лог.1 на обоих входах **I** и **K** триггер меняет свое состояние за каждым положительным перепадом тактирующего сигнала.

**Порядок выполнения работы:**

1. Собрать схему по рис. 6.1. Комбинация состояния тумблеров **S1, S2,** поочередно установить все 4 возможные комбинации входных сигналов элемента **2ИЛИ-НЕ** и при вольтметре постоянного тока или электронно-лучевого осциллографа определить состояние выхода элемента, соответствующее каждой из этих комбинаций. Составить таблицу истинности логического элемента **2ИЛИ-НЕ**.

2. Сменить микросхему **К561ЛЕ5** на **К561ЛА7** и проделать аналогичные операции. Составить таблицу истинности логического элемента **2И-НЕ.**

3. Установить на монтажную плату микросхему **К561ЛП2** и проделать аналогичные операции. Составить таблицу истинности логического

элемента ИСКЛЮЧАЮЩЕЕ ИЛИ.

4. Собрать схему исследования **D –** триггера по рис. 6.4. Проверить реакцию триггера на состояние установочных входов **R, S** и на тактовый сигнал **C** при различных состояниях информационного входа **D**. Составить таблицу переходов **D –** триггера. Таковой сигнал задавать нажатием кнопки **S2**.

5. Собрать схему исследования **I – K** – триггера по рис. 6.5. Проверить реакцию триггера на состояние установочных входов **R, S** и на тактовый сигнал **C** при различных состояниях информационных сигналов **I – K**. Составить таблицу переходов **I – K** – триггера. Таковой сигнал задавать нажатием кнопки **S3**.

**Содержание отчета.**

Отчет должен содержать:

1. Принципиальные схемы опытов.
2. Перечень исследуемых микросхем, их основные характеристики.
3. Тип и основные характеристики используемых контрольно-измерительных приборов.
4. Результаты замера величины напряжения питания, уровень Лог.0 и Лог.1 на выходах микросхем.
5. Таблицы истинности исследуемых логических схем и таблицы переходов исследуемых триггеров.
6. Выводы, сравнение **D** и **I – K** – триггеров, ответы на контрольные вопросы.

**Контрольные вопросы:**

1. Из каких логических элементов можно построить схему триггера;
2. Можно ли построить схему D-триггера на основе RS- триггера;
3. Чем отличается двухтактный триггер от однотактного триггера;
4. Каково назначение регистров;
5. Чем определяется разрядность регистров
6. Как построить схему Т-триггера, если использовать схему RS- триггера и логические элементы
7. Можно ли построить схему D-триггера на основе RS- триггера
8. Как обозначаются регистры на схемах электрических функциональных и принципиальных

# ПРАКТИЧЕСКАЯ РАБОТА № 20

## СЧЕТЧИКИ

**Цель:** приобрести практические навыки в изучении счетчиков.

**Сформировать следующие умения: У 1.** пользоваться основными видами современной вычислительной техники, периферийных и мобильных устройств и других технических средств информатизации.

**Ход работы:**

1. Приборы учета горячей и холодной воды;
2. Приборы учета тепловой энергии;
3. Приборы учета электроэнергии;
4. Приборы учета газа.

1. **Приборы учета горячей и холодной воды:**

Есть несколько видов счетчиков воды, но все их можно разделить на две основные группы: энергонезависимые (используют механические процессы) и энергозависимые (нуждаются в наличии электричества).

Вообще существует четыре вида счетчиков:

- Тахометрические (турбинные или крыльчатые). Относятся к механическим устройствам (энергонезависимые). Основная рабочая деталь — небольшая турбина или крыльчатка, которая находится в воде. Количество воды считается по количеству оборотов, которые крыльчатка делает. Обороты оси крыльчатки заставляют двигаться механический счетчик. Работает с любой водой (жесткой, с повышенным содержанием железа и т.д.), но точность показаний не очень высокая, есть вероятность того, что через несколько лет крыльчатка зарастет и станет вращаться медленнее. Но может так случиться, что крутиться она будет быстрее чем надо. В общем, вариант недорогой и такого типа счетчики в основном и ставят в квартирах.

- Индукционные или электромагнитные. Для работы этого прибора требуется электричество. Действие прибора основано на способности воды проводить электричество. Прибор дорог, срок службы большой — более 10 лет, при отсутствии посторонних включений в воде и ржавчины/накипи в трубах выдает точные данные. Так как прозрачная вода и идеальное состояние трубопровода — это не про наши сети, ставят его нечасто.

- Ультразвуковые. Тоже требуют наличия электропитания. Принцип работы — сравнение скорости распространения ультразвука «по» и «против» течения. Точность высокая, но тоже только на чистых трубах. Цена еще выше, хоть и срок службы в районе 15 лет.

- Вихревые. Они анализируют скорость образования вихревой дорожки за препятствием особой формы. Поведение водяного вихря анализируется электронным устройством (тоже требуется электропитание). Работают нормально только с чистой водой без примесей, срок службы — порядка 8 лет.

Из всего сказанного выше получается, что в нашим реальным условиям — старым трубам и не самой чистой воде — подходят только тахометрические счетчики. Их в основном и устанавливают.

***Виды тахометрических счетчиков***

Механические счетчики хороши тем, что для их работы не требуется никаких дополнительных условий. Есть поток воды — они работают, нет его — они стоят. Вот только длительный простой они не очень любят — из-за наличия солей в воде крыльчатка или турбина могут перестать вращаться. Это значит, что вам необходимо будет прибор учета менять.

Классификация по температуре воды

Приборы учета расхода воды (это официальное название) есть для холодной и горячей воды. Отличаются они диапазоном рабочих температур:

- для холодной максимум составляет +30°С;
- для горячей +130°С;
- универсальные — рабочий диапазон температур — до +90°С.

Даже если счетчики воды одинакового типа и производитель у них один, разница в цене ощутима. Но она оправдана — для ГВС используются термостойкие материалы, а они стоят намного дороже. Чтобы внешне их можно было легко различать, корпуса окрашивают в разный цвет — счетчики на холодную воду обычно синего цвета, на горячую — красного, универсальные имеют оранжевую окраску.

2. Приборы учета тепловой энергии:

Счетчики тепла предназначены для организации коммерческого учета тепловой энергии, теплоносителя, горячей воды в жилых домах и промышленных предприятий, а также для организации систем дистанционного сбора информации о потреблении энергоресурсов. По способу применения счетчики тепла делятся на *квартирные* и *домовые (промышленные)*. По принципу действия - *механические* и *ультразвуковые*

***Квартирные счетчики тепла***

Квартирные счетчики тепла представляют собой приборы относительно небольших диаметров каналов, не более 15-20 мм, с диапазоном измерения теплоносителя менее 0,6-2,5 м3/ч. Возможны методы измерения расхода электромагнитный, вихревой, турбинный.

При выборе метрологических характеристик расходомера для квартирных теплосчетчиков обычно стремятся применять расходомер, обладающий наиболее высокой точностью. Однако это не всегда оправдано. При измерении тепловой энергии, потребляемой на обогрев помещения, необходимо знать, кроме объемного расхода теплоносителя, его плотность, разность температуры на входе и выходе теплотрассы, рабочее давление в трубопроводе.

Учитывая сезонность тепловой нагрузки, квартирные счетчики тепла должны иметь двух и более предельный диапазон измерения, или один, но очень широкий (1/1000). Последнее гораздо хуже, т.к. приводит к

необходимости повышать точность измерения расхода, что нежелательно, т.к. квартирные счетчики тепла в этом случае удорожают.

***Промышленные счетчики тепла***

Промышленные счетчики тепла представляют собой приборы больших диаметров каналов, начиная с 25 мм до 300 мм. диапазоном измерения теплоносителя менее 0,6-2,5 м3/ч. Возможны методы измерения расхода электромагнитный, вихревой, турбинный.

***Механические счетчики тепла***

Механические счетчики тепла (крыльчатые, турбинные, винтовые) наиболее простые приборы. Их принцип действия основан на преобразовании поступательного движения потока жидкости во вращательное движение измерительной части. Механические счетчики тепла состоят из тепловычислителя и механических роторных или крыльчатых водосчетчиков. Это наиболее дешевые счетчики. Однако к их стоимости надо обязательно добавлять стоимость специальных фильтров, которые устанавливаются перед каждым механическим счетчиком тепла. В результате, цена таких комплектов на 10-15% ниже счетчиков других типов, но только для условных диаметров трубопровода не более 32 мм.

К недостаткам механических счетчиков тепла относится невозможность их использования при повышенной жесткости воды, присутствии в ней мелких частиц окалины, ржавчины и накипи, которые забивают фильтры и механические расходомеры. Кроме того, механические расходомеры создают наибольшие потери давления воды по сравнению с расходомерами других типов.

***Утразвуковые счетчики тепла***

Существует множество модификаций ультразвуковых счетчиков тепла, но основной принцип работы любого из них заключается примерно в следующем: на трубе друг напротив друга устанавливаются излучатель и приемник ультразвукового сигнала. Излучатель посылает сигнал сквозь поток жидкости, а приемник через некоторое время получает его. Время задержки сигнала между моментами его излучения и приема прямо пропорционально скорости потока жидкости в трубе: оно измеряется и по его величине вычисляется расход жидкости в трубопроводе. В зависимости от взаимного положения излучателя(ей) и приемника(ов) существует порядка 10 модификаций ультразвуковых расходомеров.

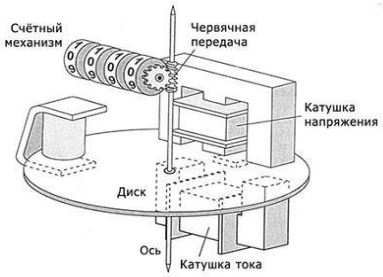
Кроме стандартных функций по измерению расхода, объема теплоносителя, его температуры и давления, вычисления потребленного или произведенного тепла, ультразвуковые счетчики также могут иметь функцию регулирования подачи теплоносителя по двум независимым каналам.

3. Приборы учета электроэнергии:

Счётчик электрической энергии (электрический счётчик) — прибор для измерения расхода электроэнергии переменного или постоянного тока

(обычно в кВт·ч или А·ч).

***Принцип работы***



**Устройство индукционного счётчика**

Для учёта активной и реактивной электроэнергии переменного тока служат индукционные одно- и трёхфазные приборы, для учёта расхода электроэнергии постоянного тока (электрический транспорт, электрифицированная железная дорога) — электродинамические счётчики. Число оборотов подвижной части прибора, пропорциональное количеству электроэнергии, регистрируется счётным механизмом.

В электрическом счётчике индукционной системы подвижная часть (алюминиевый диск) вращается во время потребления электроэнергии, расход которой определяется по показаниям счётного механизма. Диск вращается за счёт вихревых токов, наводимых в нём магнитным полем катушки счётчика, — магнитное поле вихревых токов взаимодействует с магнитным полем катушки счётчика.

В электрическом счетчике электронного типа переменный ток и напряжение воздействуют на твердотельные (электронные) элементы для создания на выходе импульсов, число которых пропорционально измеряемой активной энергии.

**Виды и типы**

Счетчики электроэнергии можно классифицировать по типу измеряемых величин, типу подключения и по типу конструкции.

По типу подключения все счетчики разделяют на приборы прямого включения в силовую цепь и приборы трансформаторного включения, подключаемые к силовой цепи через специальные измерительные трансформаторы.

По измеряемым величинам электросчетчики разделяют на однофазные (измерение переменного тока 220 В, 50 Гц) и трехфазные (380 В, 50 Гц). Все современные электронные трехфазные счетчики поддерживают однофазный учёт.

Также существуют трехфазные счетчики для измерения тока напряжением в 100 В, которые применяются только с трансформаторами тока в высоковольтных (напряжением выше 660 В) цепях.

По *конструкции*:

**Индукционным** (электромеханическим электросчетчиком)

называется электросчетчик, в котором магнитное поле неподвижных токопроводящих катушек влияет на подвижный элемент из проводящего материала. Подвижный элемент представляет собой диск, по которому протекают токи, индуцированные магнитным полем катушек. Количество оборотов диска в этом случае прямо пропорционально потребленной электроэнергии.

Индукционный счётчик

Индукционные (механические) счётчики электроэнергии постоянно вытесняются с рынка электронными счетчиками из-за отдельных недостатков: отсутствие дистанционного автоматического снятия показаний, однотарифность, погрешности учёта, плохая защита от краж электроэнергии, а также низкой функциональности, неудобства в установке и эксплуатации по сравнению с современными электронными приборами. Индукционные счетчики хорошо подходят для квартир с низким энергопотреблением.

**Электронным** (статическим электросчетчиком) называется электросчетчик, в котором переменный ток и напряжение воздействуют на твердотельные (электронные) элементы для создания на выходе импульсов, число которых пропорционально измеряемой активной энергии. Счетный механизм представляет собой электромеханическое (имеет преимущество в областях с холодным климатом, при условии установки прибора на улице) или электронное устройство, содержащее как запоминающее устройство, так и дисплей. Электронные счетчики хорошо подходят для квартир с высоким энергопотреблением и для предприятий.

Электронный счётчик

Основными достоинствами электронных электросчетчиков является возможность учёта электроэнергии по дифференцированным тарифам. Электронные электросчетчики имеют больший межповерочный период (4-16 лет).

**Гибридные** счётчики электроэнергии — редко используемый промежуточный вариант с цифровым интерфейсом, измерительной частью индукционного или электронного типа, механическим вычислительным устройством.

3. Приборы учета газа:

Прибор контроля расходования голубого топлива называется счетчиком газа. Он включается в питающую дом или квартиру газовую магистраль (трубу) снаружи помещения, либо внутри его. Используя разные методы определения скорости проходящего через него газообразного вещества, он определяет объем потребляемого газа. Эту величину, пересчитанную в кубические метры, мы и видим на табло прибора.

В электронных расходомерах применяются жидкокристаллические табло, в механических – механический счетчик, который имеет красные деления, показывающие дробные части кубометра.

Газовый расходомер состоит из трех основных частей: герметичного корпуса, измерительного преобразователя, счетного устройства. Измерительный механизм является основным элементом, а принцип преобразования потока в механическое или электрическое воздействие на счетное устройство определяет вид прибора. Механические преобразователи используют разные свойства газов для перемещения рычага, воздействующего на счетный прибор. Электронные датчики преобразуют свойства потока в электрический импульс, который увеличивает показания на табло.

***Принцип работы***

Измерительный преобразователь использует физические свойства потока газообразного вещества. По принципу его работы различают несколько видов счетчиков:

Мембранный или диафрагменный – поочередно заполняет две измерительные камеры, разделенные мембраной. Каждая камера имеет фиксированный объем. Клапаны на входе и выходе асимметрично синхронизированы и поочередно впускают, и выпускают газ. Движения мембраны преобразуются рычагом в поворот привода счетного устройства, которое показывает объем прошедшего через прибор топлива.

***Типы счетчиков газа***

Основной характеристикой, которая разделяет газовые расходомеры на два типа, является их пропускная способность или объем газа, который может пройти через счетный прибор за 1 час. По этому показателю они делятся на:

***Бытовые*** – это самый распространенный тип с пропускной способностью до 12 куб.м/ ч. Для них применяется мембранный, диафрагменный и ротационный принцип измерения.

***Промышленные*** – все расходомеры с пропускной способностью выше 12 куб.м в час. До величины в 200 куб.м/ч в основном применяются ротационные преобразователи, свыше этого показателя – турбинные, вихревые.

- Мембранного типа. Приборы просты и надёжны. Это немецкие счётчики, предназначенные для бытового и промышленного пользования. Их отличает выгодная стоимость, высокая точность показаний. К недостаткам следует отнести чувствительность к постоянным и переменным нагрузкам, габариты. Ротационного типа. Счётчики нагрузок не боятся, отличаются долговечностью, большой пропускной способностью, точностью показаний. Но, имея малые размеры и лёгкий вес, приборы имеют высокую цену из-за дорогостоящих элементов.
- Турбинного типа. Приборы названы по принципу работы, поток топлива вращает турбину, фиксируя, таким образом, потребление. Современные модели созданы для онлайн передачи информации на сервер службы. Стоит недешево, пока не получили широкого распространения.
- Вихревого типа. Главной особенностью счётчика является зависимость от электричества. Это же считается недостатком устройства, плюсы его в низкой чувствительности к загрязнениям и значительный диапазон

измерения потока.

• Электрические счётчики отличаются простотой настройки и эксплуатации, они не шумят, компактный прибор не требует много места.

Счётчики делятся на:

• механические модели выгодны тем, что немного стоят, просты и надежны, их минусы шумность и подверженность температурам;

•

электронные не шумят, показывают более точные данные, не боятся температур и перепадов, бесшумны, компактны, но дороги. В отдельных случаях все их плюсы не окупаются.

**Контрольные вопросы:**

1. По каким признакам можно классифицировать счетчики;
2. Как учитываются наборы, соответствующие запрещенным состояниям счетчиков, при минимизации функций возбуждения триггеров;
3. Как можно установить счетчик в исходное состояние;
4. Какой порядок счета называется естественным, а какой искусственным;
5. Назовите основные временные характеристики счетчиков

Укажите преимущества и недостатки кольцевых счетчиков

## ПЕРЕЧЕНЬ РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

**Перечень рекомендуемых учебных изданий, Интернет-ресурсов, дополнительной литературы**

1. Гагарина, Л. Г. Технические средства информатизации: учеб. пособие / Л.Г. Гагарина. — Москва: ИД «ФОРУМ»: ИНФРА-М, 2018. — 255 с. — (Среднее профессиональное образование). [Электронный ресурс; Режим доступа http://znanium.com]
2. Зверева, В. П. Технические средства информатизации: Учебник. - Москва: КУРС: ИНФРА-М, 2020. - 256 с. - (Среднее профессиональное образование). [Электронный ресурс; Режим доступа http://znanium.com

# СОДЕРЖАНИЕ

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**……... 3
**ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**…….. 4
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №1**…… 5
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №2**…… 8
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №3**…… 12
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №4**…… 15
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №5**…… 18
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №6**…… 30
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №7**…… 36
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №8**…… 38
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №9**…… 40
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №10**…….. 41
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №11**…….. 51
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №12**…….. 53
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №13**…….. 54
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №14**…….. 60
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №15**…….. 66
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №16**…….. 67
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №17**…….. 72
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №18**…….. 87
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №19**…….. 99
**ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №20**…….. 101
**ПЕРЕЧЕНЬ РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ**……. 108