Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Воронежский государственный лесотехнический университет

имени Г.Ф. Морозова»

Кафедра Вычислительной техники и информационных систем

(название кафедры)

**Пояснительная записка**

**к курсовой работе**

(вид работы)

«Архитектура мобильных устройств, работающих под управлением ОС Windows Phone.»

(тема)

09.03.02 Информационные системы и технологии

(код и наименование направления подготовки)

По дисциплине: Архитектура ЭВМ и вычислительных систем

**Вариант 7**

Студент группы ИС2–191–ОБ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Величко В.А.

(номер группы) (подпись) (инициалы и фамилия)

Руководитель к.т.н. доцент \_\_\_\_\_\_\_\_\_\_\_\_\_ Зольников К.В.

(ученая степень, ученое звание) (подпись) (инициалы и фамилия)

Воронеж 2020 г.

**Оглавление**

[ВВЕДЕНИЕ 3](#_Toc59987032)

[1.Теоретическая часть 4](#_Toc59987033)

[1.1 Windows Mobile. 4](#_Toc59987034)

[1.2 Создание приложений для Windows Phone 5](#_Toc59987035)

[1.3 Архитектура Windows Phone. 5](#_Toc59987036)

[2.Практическая часть 9](#_Toc59987037)

[2.1. Задание 9](#_Toc59987038)

[ЗАКЛЮЧЕНИЕ 15](#_Toc59987039)

[СПИСОК ЛИТЕРАТУРЫ 16](#_Toc59987040)

# ВВЕДЕНИЕ

Задачи работы:

1. Изучить архитектуру мобильных устройств, работающих под управлением ОС Windows Phone.

2.Нарисовать блок-схему для решения поставленной задачи.

3. Написать программу на языке ассемблера.

Windows Phone — мобильная [операционная система](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), разработанная [Microsoft](https://ru.wikipedia.org/wiki/Microsoft), вышла 11 октября 2010 года. 21 октября начались поставки первых устройств на базе новой платформы. В [России](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D1%8F) телефоны с Windows Phone начали продаваться 16 сентября 2011 года, первым из которых стал [HTC 7 Mozart](https://ru.wikipedia.org/wiki/HTC_7_Mozart). 9 октября 2017 года исполнительный директор Microsoft Джо Бельфиор заявил о прекращении создания новых устройств и обновлений Windows 10 Mobile. В 2019 году компанией Microsoft официально заявлено, что поддержка Windows 10 Mobile будет прекращена 10 декабря 2019 года;

# 1.Теоретическая часть

## 1.1 Windows Mobile.

Windows Mobile для своего времени была хорошей и удобной мобильной операционной системой. С ее помощью Microsoft заняла доминирующее положение на рынке карманных компьютеров и первых смартфонов, отправила на свалку истории Palm OS и сделала Symbian нишевой платформой. Но смену парадигмы работы с мобильными компьютерами американская компания прозевала. Пока разработчики Microsoft пытались вдохнуть жизнь в Windows Mobile, Apple выдавала одну новую функцию в iOS за другой, а Google догонял и раздавал Android всем желающим производителям смартфонов.

По сути, в Windows Mobile было лишь два существенных недостатка: ориентированность графического интерфейса на управление пером и отсутствие централизованной системы установки и управления приложениями. Ядро Windows CE, на которой была основана эта 11 операционная система, остается достаточно производительным и функциональным и поныне. Под новым именем Windows Embedded Compact оно используется в различных устройствах OEM-производителей, которые поверх него устанавливают собственные оболочки и программное обеспечение.

Следующая версия мобильной операционной системы Microsoft Windows Phone получила большее распространение по нескольким причинам. Вопервых, компания исправила множество проблем, во-вторых, добавила необходимые для работы функции, в-третьих, обеспечила длительную поддержку и обновления прошивки даже для самых старых и маломощных устройств.

Windows Phone на сегодняшний день является одной из трех наиболее популярных ОС для смартфонов, в связи с чем возникает необходимость в приложениях различного рода для этой операционной системы. И хотя пока Windows Phone по популярности сильно уступает системам Android или iOS, так или иначе рождается потребность в приложениях под эту систему. Кроме того, есть сферы, где Windows Phone может составить конкуренцию другим системам, например, приложения для бизнеса.

## 1.2 Создание приложений для Windows Phone

Для создания приложений для Windows Phone потребуется среда разработки Visual Studio Express for Windows. Также вместо экспресс-версии можно использовать другой более функциональный выпуск Visual Studio, например, Visual Studio Community, Professional или Ultimate. В плане функционала для разработки эти выпуски несильно отличаются. Кроме самой разработки также потребуются средства тестирования. Visual Studio по умолчанию предоставляет эмуляторы смартфонов, которые можно использовать для тестирования. Однако для подключения эмуляторов надо включить в BIOS режим Hyper-V, который имеет ряд аппаратных ограничений. Кроме того, режим Hyper-V доступен только для 64-битных версий Windows 8 Pro и Windows 8 Enterprise. И если ОС на рабочем компьютере является Windows 8 Basic или Windows RT, то Hyper-V будет невозможно использовать. Кроме того, можно использовать смартфон с OS Windows Phone. Хотя эмуляторы очень удобны: они предоставляют различный спектр устройств с различными настройками, но полную достоверную картину дает тестирование на реальных устройствах .

## 1.3 Архитектура Windows Phone.

В Windows Phone 8 используется новая архитектура Windows NT, которая используется в настольных операционных системах Microsoft. До недавнего времени разработчики приложений для Windows использовали две основные группы API: неуправляемый (native) через Win32 API и управляемый (managed) через .NET Framework. Вторая группа постепенно развивалась, получая различные новые и усовершенствованные библиотеки для создания пользовательского интерфейса (например, WPF, Windows Presentation 12 Foundation), работы с данными и сервисами, дополнительные инструменты для разработки, построения исходного кода и архитектуры приложений. Между тем, сама платформа Windows, получала не так много настоящих толчков к развитию базовой модели разработки. Пожалуй, последним существенным нововведением был COM (Component Object Model), появившийся еще в 90-е годы. Но все это время компьютеры не стояли на месте. Появлялись всевозможные новые устройства, экраны, чувствительные к прикосновениям, возникали новые форм-факторы, такие как планшеты, и т. д. Наконец, такой параметр, как энергопотребление, становился все более важным. Поэтому, создавая новую версию Windows, в Microsoft понимали, что необходимо разработать и новый API, который, будучи родным (native) для операционной системы, станет отвечать новым требованиям и веяниям времени. В результате появился Windows Runtime (WinRT). Windows Runtime — это новая модель разработки приложений, а также объектноориентированный языконезависимый программный интерфейс (API), написанный на неуправляемом коде и реализующий концепции асинхронного программирования. Все функции и методы, потенциально работающие более 50 мс, реализованы асинхронно. Синхронных аналогов для них нет. Это обеспечивает лучшие характеристики и бóльшую "отзывчивость" приложений. WinRT работает на основе новой оптимизированной версии COM, при этом благодаря системе метаданных и языковых проекций он может напрямую интегрироваться с управляемыми средами, такими как .NET Framework. Некоторые API, входящие в WinRT, могут быть использованы и в классических приложениях, но бóльшая часть из них доступна только для Windows Store-приложений. Windows Store-приложения могут создаваться на различных языках программирования: C#, JavaScript, Visual Basic и C++. Хочется отметить, что можно создавать невизуальные компоненты WinRT на одном из перечисленных языков (кроме JavaScript), например, на языке C#, и встраивать их в приложения, написанные на других языках программирования, например, на C++ или JavaScript. Все программные интерфейсы WinRT выглядят "родными" для каждого из поддерживаемых языков программирования. Поэтому для взаимодействия с WinRT не придется предпринимать никаких дополнительных усилий, как, например, для взаимодействия с COM из C#. Для разработчика управляемые типы .NET Framework и типы WinRT выглядят одинаково. Пользовательский интерфейс приложений, написанных на JavaScript, создается с помощью HTML 5, в остальных случаях применяется XAML (eXtensible Application Markup Language — расширяемый язык разметки приложений). XAML знаком многим разработчикам по WPF и Silverlight. На XAML разрабатывают приложения для Windows Phone. А теперь XAML стал и частью Windows Runtime, а значит, и одним из базовых компонентов операционной системы Windows. В общем случае Windows Store-приложения изолированы друг от друга. Это обеспечивает стабильность и безопасность как самих приложений, 13 так и системы в целом. Разработчики приложений должны декларативно объявить, какие потенциально небезопасные возможности они будут использовать. Например, если приложение захочет взаимодействовать с камерой, это должно быть объявлено заранее, а при первом обращении к камере система спросит у пользователя, разрешает ли он данное действие. Аналогично обстоит дело и, например, с определением местоположения. Пользователь может явно разрешить или запретить предоставление приложению такой информации . Архитектуру платформы Windows иллюстрирует рисунок 4:



Рисунок 4 – Платформа Windows.

Для разработки Windows Store-приложений вам потребуется x86 или x64 версия Windows 8. Разработка Windows Store-приложений на более ранних версиях Windows или на Windows RT для ARM-процессоров не поддерживается. Вместе с Visual Studio Express for Windows 8 в числе прочих продуктов, таких как Windows 8 SDK, будет установлен Blend for Visual Studio. Blend поставляется также и в составе старших редакций Visual Studio. Blend — это инструмент для создания (дизайна) пользовательского интерфейса приложений. Данный продукт существует уже достаточно давно, есть различные версии Blend, предназначенные, например, для создания приложений WPF (Windows Presentation Foundation), а также приложений для Windows Phone. Blend for Visual Studio позволяет создавать Windows Storeприложения. Разрабатывать Windows Store-приложения можно полностью в Visual Studio, ни разу не запуская Blend. Однако Blend предоставляет интерфейс, более удобный для формирования дизайна приложений, в то время как в Visual Studio удобней работать над исходным кодом. Blend имеет ряд полезных возможностей, таких как легкое создание анимации, работу с VSM (Visual State Manager) в графическом интерфейсе и т. д. Интерфейс 14 Blend привычней для дизайнеров, которых очень нелегко, если вообще возможно, заставить работать в Visual Studio. Несмотря на то, что Blend подразумевает работу в графическом редакторе, он поддерживает редактирование разметки XAML и кода на языке C# напрямую (или HTML и JavaScript в случае JavaScript-приложений). Таким образом, можно создать приложение полностью в Blend, ни разу не запуская Visual Studio. Visual Studio и Blend могут быть запущены параллельно, и разработка приложений может вестись одновременно с помощью обоих инструментов. Это происходит благодаря тому, что и в Blend, и в Visual Studio можно работать с одними и теми же типами проектов, над одними и теми же физическими файлами.

# 2.Практическая часть

# 2.1. Задание

С клавиатуры вводятся 7 символов, записать в память, начиная с 60 адреса все строчные латинские буквы из буфера клавиатуры.

Контроллер клавиатуры (Рис. 1) представляет собой модель внешнего устройства, принимающего ASCII-коды от клавиатуры ПЭВМ.

Символы помещаются последовательно в буфер символов, размер которого установлен равным 50 символам, и отображаются в окне обозревателя.

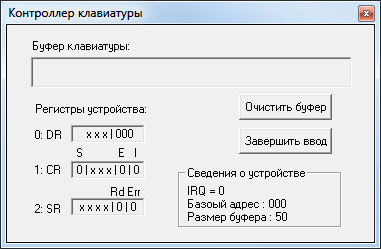


Рисунок 1 – Окно контроллера клавиатуры

В состав контроллера клавиатуры входят три программно-доступных регистра.

– DR (адрес 0) – регистр данных

– CR (адрес 1) – регистр управления, определяет режимы работы контроллера и содержит следующие флаги:

* Е – флаг разрешения приема кодов в буфер;
* I – флаг разрешения прерывания;
* S – флаг режима посимвольного ввода;

- SR (адрес 2) – регистр состояния, содержит два флага:

* Err – флаг ошибки;
* Rd – флаг готовности.

Регистр данных DR доступен только для чтения, через него считываются ASCII-коды из буфера, причем порядок чтения кодов из буфера соответствует порядку их записи в буфер — каждое чтение по адресу 0 автоматически перемещает указатель чтения буфера. В каждый момент времени DR содержит код символа по адресу указателя чтения буфера.

Флаги регистра управления CR устанавливаются и сбрасываются программно.

Флаг Е, будучи установленным, разрешает прием кодов в буфер. При Е = 0 контроллер игнорирует нажатие на клавиатуре, прием кодов в буфер не производится. На считывание кодов из буфера флаг Е влияния не оказывает.

Флаг I, будучи установленным, разрешает при определенных условиях формирование контроллером запроса на прерывание. При I = 0 запрос на прерывание не формируется.

Флаг S = 1 устанавливает так называемый режим посимвольного ввода, иначе контроллер работает в обычном режиме. Флаг S устанавливается и сбрасывается программно, кроме того, S сбрасывается при нажатии кнопки очистить буфер в окне контроллер клавиатуры.

Условия формирования запроса на прерывание определяются, с одной стороны, значением флага разрешения прерывания I, с другой – режимом работы контроллера. В режиме посимвольного ввода запрос на прерывание формируется после ввода каждого символа (разумеется, при I=1), в обычном режиме запрос будет сформирован по окончании набора строки.

Завершить набор строки можно, щелкнув по кнопке завершить ввод в окне Контроллер клавиатуры. При этом устанавливается флаг готовности Rd (от англ. ready) в регистре состояния SR. Флаг ошибки Err (от англ. error) в том же регистре устанавливается при попытке ввода в буфер 51-го символа. Ввод 51-го и всех последующих символов блокируется.

Сброс флага Rd осуществляется автоматически при чтении из регистра DR. флаг Err сбрасывается программно. Кроме того, оба эти флага сбрасываются при нажатии кнопки очистить буфер в окне контроллер клавиатуры; одновременно со сбросом флагов производится очистка буфера – весь буфер заполняется кодами 00h, и указатели записи и чтения устанавливаются на начало буфера.

Для программного управления контроллером предусмотрен ряд командных слов. Все команды выполняются при записи по адресу регистра управления CR кодов с 1 в третьем разряде.

Контроллер клавиатуры интерпретирует следующие командные слова:

- xxx101 – очистить буфер (действие команды эквивалентно нажатию кнопки Очистить буфер);

- ххх102 – сбросить флаг Err в регистре SR;

- ххх103 – установить флаг S в регистре CR;

- ххх104 – сбросить флаг S в регистре CR.

Задание

С клавиатуры вводятся 7 символов, записать в память, начиная с 60 адреса все строчные латинские буквы из буфера клавиатуры.

Для решения задачи требуется подключить клавиатуру. Данные символов ASCII считываются из регистра DR и записываются в Acc. Требуется записать все символы, которые являются строчными латинским буквами. Соответственно нужно записать в память начиная с 60 ячейки те символы, числовое представление которых лежит в пределах от 97 до 122 включительно.

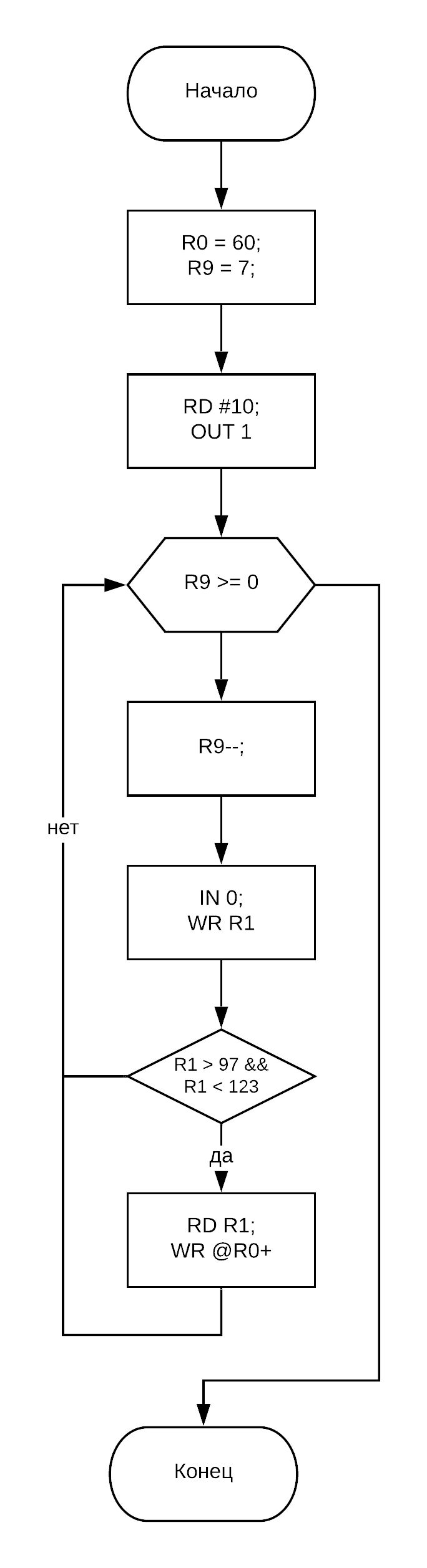


Рисунок 2- блок схема алгоритма

В таблице 1 показано размещение в памяти кодов и их описание.

|  |  |  |
| --- | --- | --- |
| **Адрес** | **Примечание** | |
| **Мнемокод** |  |
| 000 | RD #60 | Начальная ячейка памяти |
| 001 | WR R0 |
| 002 | RD #7 | Запись количества элементов массива |
| 003 | WR R9 |
| 004 | RD #10 | Разрешение на посимвольный ввод с клавиатуры |
| 005 | OUT 1 |
| 006 | LOOP: RD R9 | Запуск цикла |
| 007 | SUB #1 | Изменение счетчика. Если счетчик меньше 0, программа завершается |
| 008 | WR R9 |
| 009 | JS END |
| 010 | IN 0 | Считывание символа из буфера клавиатуры. Если символ отсутствует, то программа завершает свою работу |
| 011 | JZ END |
| 012 | WR R1 |
| 013 | SUB #97 | Проверка того, что символ является прописной латинской буквой |
| 014 | JS LOOP |
| 015 | RD R1 |
| 016 | SUB #123 |
| 017 | JNS LOOP |
| 018 | RD R1 | Если символ удовлетворяет условию, то запись его в память |
| 019 | WR @R0+ |
| 020 | JMP LOOP | Повторение итерации цикла |
| 021 | END: HLT | Конец программы |

Таблица 1 размещение в памяти кодов и их описание.

По условию задачи вводим 7 символов из буфера клавиатуры. Если количество символов превышает 7, то все последующие игнорируются. Значения записываются в память начиная с 60 ячейки.

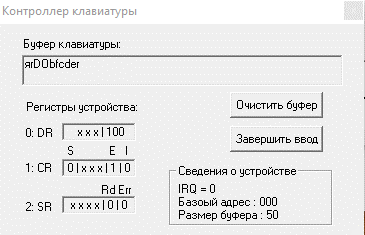


Рисунок 3. – Контроллер клавиатуры после ввода исходных данных

На рисунке 4 представлена память после выполнения программы.

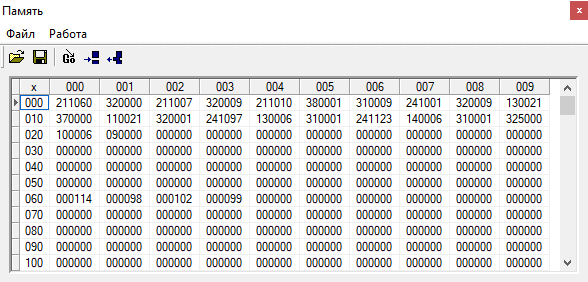


Рисунок 4. – Память ОЗУ после выполнения программы.

В результате работы программы были записаны 4 символа – «r» (код 114), «b» (код 98), «f» (код 102), «c» (код 99). Все последующие символы были прогнорированны, так как выходят за рамки условий задачи – их индекс превышает 7.

# ЗАКЛЮЧЕНИЕ

В данной курсовой работе была рассмотрена архитектура мобильных устройств, работающих под управлением ОС Windows Phone.

Была создана программа, считывающая символы в виде букв кириллицы алфавита с внешнего устройства как контроллер клавиатуры. Программа считывает с буфера клавиатуры 7 символов по одному символу за операцию цикла и, если символ является буквой кириллицы маленькой или большой, записывает в ячейки памяти ОЗУ последовательно. В память записываются ASCII коды каждой веденой буквы.

# СПИСОК ЛИТЕРАТУРЫ

1. 1.Папков В.И. Система памяти ЭВМ (Функциональный подход). Учеб. пособие. СПб.: Изд.центр СПб ГМТУ. 2002. 238 с.
2. 2.http://metanit.com/sharp/windowsphone Руководство по разработке для Windows Phone 8.1
3. http://www.oszone.net Компьютерный официальный портал.
4. http://blog.websofter.ru/ Блог о программировании.
5. Столлингс В. Структурная организация и архитектура компьютерных систем. 5-е издание. - М.: Издательский дом "Вильямс", 2002. - 896 с
6. Касперский К. Техника оптимизации программ. Эффективное использование памяти. - СПб.: БХВ-Петербург, 2003. - 464 с.
7. Цилькер Б. Я., Орлов С.А. Организация ЭВМ и систем. СПб.: Питер, 2006. - 668 с.