МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

«БРАТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра управления в технических системах

УТВЕРЖДАЮ:
Проректор по учебной работе
_____ Е.И. Луковникова
«29» мая 2020 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

МИКРОКОНТРОЛЛЕРЫ И МИКРОПРОЦЕССОРЫ В СИСТЕМАХ УПРАВЛЕНИЯ

Б1.В.ДВ.08.02

НАПРАВЛЕНИЕ ПОДГОТОВКИ

27.03.04 Управление в технических системах

ПРОФИЛЬ ПОДГОТОВКИ

Управление и информатика в технических системах

Программа академического бакалавриата

Квалификация (степень) выпускника: бакалавр

Программа составлена в соответствии с федеральным государственным образовательным стандартом высшего образования по направлению подготовки 27.03.04 Управление в технических системах от 20.10.2015 г № 1171 и учебным планом ФГБОУ ВО «БрГУ» от 03.02.2020 г № 46 для очной формы обучения, заочно - ускоренной формы обучения для набора 2020 года

СОДЕРЖАНИЕ ПРОГРАММЫ **Стр.**

1. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ, СООТНЕСЕННЫХ С ПЛАНИРУЕМЫМИ РЕЗУЛЬТАТАМИ ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Вид деятельности выпускника

Дисциплина охватывает круг вопросов, относящихся к научно-исследовательскому виду профессиональной деятельности выпускника в соответствии с компетенциями и видами деятельности, указанными в учебном плане.

Цель дисциплины

Приобретение умений и навыков исследования проблем в своей предметной области, выбора методов и средств их решения, анализа результатов теоретических и экспериментальных исследований.

Задачи дисциплины

Формирование способностей анализа результатов исследований, выбора методов и средств решения проблем в своей предметной области.

Код компетенции	Содержание компетенций	Перечень планируемых результатов обучения по дисциплине
1	2	3
ПК-6	способность производить расчеты и проектирование отдельных блоков и устройств систем автоматизации и управления и выбирать стандартные средства автоматики, измерительной и вычислительной техники для проектирования систем автоматизации и управления в соответствии с техническим заданием	**знать:** – состав и средства микропроцессорных систем; – способы управления микропроцессорными системами; **уметь:** - выбирать методы и средства управления микропроцессорными системами; - выбирать методы управления микропроцессорными системами **владеть:** -навыками программирования и управления микропроцессорными системами

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Дисциплина Б1.В.ДВ.8.2 Микроконтроллеры и микропроцессоры в системах управления относится к дисциплинам по выбору.

Дисциплина Микроконтроллеры и микропроцессоры в системах управления базируется на знаниях, полученных по дисциплине Б1.Б.13 Вычислительные машины, системы и сети.

Микроконтроллеры и микропроцессоры в системах управления представляет основу для научно-исследовательской работы и подготовки к государственной итоговой аттестации. Такое системное междисциплинарное изучение направлено на достижение требуемого ФГОС уровня подготовки по квалификации бакалавр.

3. РАСПРЕДЕЛЕНИЕ ОБЪЕМА ДИСЦИПЛИНЫ

3.1. Распределение объема дисциплины по формам обучения

Форма обучения	*Курс*	*Семестр*	*Трудоемкость дисциплины в часах*						*Курсовая работа (проект), контрольная работа, реферат, РГР*	*Вид промежуточной аттестации*
			Всего часов (с экз.)	*Аудиторных часов*	*Лекции*	*Лабораторные работы*	*Практические занятия*	*Самостоятельная работа*		
1	2	3	4	5	6	7	8	9	10	11
Очная	3	6	144	54	36	18	-	90	-	экзамен
Заочная	3	-	144	16	6	10	-	128	-	экзамен
Заочная (ускоренное обучение)	3	-	144	12	6	6	-	132	-	экзамен
Очно-заочная	-	-	-	-	-	-	-	-	-	-

3.2. Распределение объема дисциплины по видам учебных занятий и трудоемкости

Вид учебных занятий	*Трудоемкость (час.)*	*в т.ч. в интерактивной, активной, инновационной формах,(час.)*	*Распределение по семестрам, час*
			6
1	2	3	4
I. Контактная работа обучающихся с преподавателем (всего)	54	12	54
Лекции (Лк)	36	-	36
Лабораторные работы (ЛР)	18	12	18
II.Самостоятельная работа обучающихся (СР)	63	-	63
Подготовка к практическим занятиям	40	-	40
Подготовка к экзамену в течении семестра	23	-	23
III. Промежуточная аттестация экзамен	27	-	27
Общая трудоемкость дисциплины час.	144	-	144
зач. ед.	4	-	4

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Распределение разделов дисциплины по видам учебных занятий - для очной формы обучения:

№ раздела и темы	*Наименование раздела и тема дисциплины*	*Трудоемкость, (час.)*	*Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость; (час.)*		
			учебные занятия		*самостоятельная работа обучающихся**
			лекции	*лабораторные занятия*	
1	2	3	4	6	7
1	**Введение. История развития микропроцессоров.**	**6**	**2**	**-**	**4**
1.1.	Первое поколение ЭВМ. Характерные черты.	1,5	0,5	-	1
1.2.	Второе поколение ЭВМ.	1,5	0,5	-	1
1.3.	Третье поколение ЭВМ.	1,5	0,5	-	1
1.4	Четвертое поколение ЭВМ.	1,5	0,5		1
2.	**Внутренняя структура микропроцессоров. Принципы фон Неймана.**	**6**	**2**	**-**	**4**
2.1.	Определение микропроцессора.	1,5	0,5	-	1
2.2.	История развития микропроцессоров.	1,5	0,5	-	1
2.3.	Архитектура фон Неймана.	1,5	0,5	-	1
2.4.	Принципы фон Неймана.	1,5	0,5	-	1
3	**Классификация микропроцессоров.**	**8**	**2**	**-**	**6**
3.1.	Классификация микропроцессоров по числу больших интегральных схем.	1,4	0,4	-	1
3.2.	Классификация микропроцессоров по назначению.	1,4	0,4	-	1
3.3.	Классификация микропроцессоров по виду обрабатываемых сигналов.	1,4	0,4	-	1
3.4.	Классификация микропроцессоров по характеру временной организации.	1,4	0,4	-	1
3.5.	Классификация микропроцессоров по организации структуры.	1,2	0,2	-	1
3.6.	Классификация микропроцессоров по количеству выполняемых программ.	1,2	0,2	-	1
4	**Внутренняя структура микропроцессоров. Арифметико-логический блок.**	**6**	**2**	**-**	**4**
4.1.	Арифметико-логический блок.	2,7	0,7	-	2
4.2.	Основные характеристики микропроцессора.	1,6	0,6	-	1
4.3.	Структура типового микропроцессора.	1,7	0,7	-	1
5	**Устройство управления.**	**10**	**2**	**4**	**4**
5.1	Принцип управления.	1,5	0,5	-	1
5.2.	Организация устройств управления.	1,5	0,5	-	1
5.3.	Устройство управления микропроцессора.	1,5	0,5	-	1
5.4.	Особенности программного и микропрограммного управления.	5,5	0,5	4	1
6	**Система команд микропроцессора.**	**9**	**2**	**3**	**4**
6.1.	Оптимальная система команд.	2,7	0,7	-	2
6.2.	Формат команды.	4,6	0,6	3	1

6.3.	Классификация команд.	1,7	0,7	-	1
7	**Режимы работы микропроцессора.**	**6**	**2**	**-**	**4**
7.1.	Режимы работы микропроцессора. Защищенный режим. Реальный режим.	3	1	-	2
7.2.	Режим системного управления. Переключение между режимами.	3	1	-	2
8	**Принципы формирования адресного пространства.**	**4**	**2**	**-**	**2**
8.1.	Среда выполнения задачи. Архитектурные решения микропроцессоров.	4	2	-	2
9	**Система адресации.**	**10**	**2**	**4**	**4**
9.1.	Режимы адресации.	6,7	0,7	4	2
9.2.	Способы адресации.	1,6	0,6	-	1
9.3.	Режимы адресации операндов. Возможности микропроцессоров по адресации.	1,7	0,7	-	1
10	**Память как функциональный узел.**	**6**	**2**	**-**	**4**
10.1.	Внутренняя память компьютера.	2,7	0,7	-	2
10.2.	Технология производства интегральных схем.	1,6	0,6	-	1
10.3.	Основные характеристики полупроводниковой памяти.	1,7	0,7	-	1
11	**Принципы организации памяти.**	**6**	**2**	**-**	**4**
11.1.	Организация физической памяти. ОЗУ, ПЗУ и FLASH – память.	3	1	-	2
11.2.	Логическая структура основной памяти.	3	1	-	2
12	**Виртуальная память.**	**8**	**2**	**-**	**6**
12.1.	Модели памяти.	2,4	0,4	-	2
12.2.	Понятие о сегментированной модели памяти.	1,4	0,4	-	1
12.3.	Понятие о страничной модели памяти.	1,4	0,4	-	1
12.4.	Сегментно-страничная модель памяти.	1,4	0,4	-	1
12.5.	Плоская модель памяти.	1,4	0,4	-	1
13	**Прерывания.**	**9**	**2**	**4**	**3**
13.1.	Понятие прерывания.	1	0,5	-	0,5
13.2.	Классификация прерываний.	1	0,5	-	0,5
13.3.	Внешние прерывания.	1,5	0,5	-	1
13.4.	Система прерываний.	5,5	0,5	4	1
14	**Поддержка многозадачности.**	**5**	**2**	**-**	**3**
14.1.	Многозадачность – определение, история развития.	1	0,5	-	0,5
14.2.	Режимы многозадачности.	1	0,5	-	0,5
14.3.	Невытесняющая многозадачность.	1,5	0,5	-	1
14.4.	Многозадачность в защищенном режиме.	1,5	0,5	-	1
15	**Программы-отладчики.**	**4**	**2**	**-**	**2**
15.1	Методы и средства отладки. Отладка на примере процессора 486.	4	2	-	2
16	**Принципы программирования микропроцессоров.**	**9**	**3**	**3**	**3**
16.1.	Программирование микропроцессора. Машинно-ориентированные языки.	5	1	3	1
16.2.	Языки высокого уровня.	4	2	-	2

	Специализированные языки.				
17	**Команды языка Ассемблер**	**5**	**3**	**-**	**2**
17.1.	Основные группы операций. Мнемокоды команд процессора Pentium.	5	3	-	2
		117	**36**	**18**	**63**

- для заочной формы обучения:

№ раздела и темы	***Наименование раздела и тема дисциплины***	***Трудоемкость, (час.)***	***Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость; (час.)***		
			учебные занятия		***самостоятельная работа обучающихся****
			лекции	***лабораторные занятия***	
1	**2**	**3**	**4**	**6**	**7**
1	**Введение. История развития микропроцессоров.**	**7,4**	**0,4**	**-**	**7**
1.1.	Первое поколение ЭВМ. Характерные черты.	2,1	0,1	-	2
1.2.	Второе поколение ЭВМ.	2,1	0,1	-	2
1.3.	Третье поколение ЭВМ.	2,1	0,1	-	2
1.4	Четвертое поколение ЭВМ.	1,1	0,1		1
2.	**Внутренняя структура микропроцессоров. Принципы фон Неймана.**	**7,4**	**0,4**	**-**	**7**
2.1.	Определение микропроцессора.	2,1	0,1	-	2
2.2.	История развития микропроцессоров.	2,1	0,1	-	2
2.3.	Архитектура фон Неймана.	2,1	0,1	-	2
2.4.	Принципы фон Неймана.	1,1	0,1	-	1
3	**Классификация микропроцессоров.**	**7,6**	**0,6**	**-**	**7**
3.1.	Классификация микропроцессоров по числу больших интегральных схем.	1,1	0,1	-	1
3.2.	Классификация микропроцессоров по назначению.	1,1	0,1	-	1
3.3.	Классификация микропроцессоров по виду обрабатываемых сигналов.	1,1	0,1	-	1
3.4.	Классификация микропроцессоров по характеру временной организации.	1,1	0,1	-	1
3.5.	Классификация микропроцессоров по организации структуры.	1,1	0,1	-	1
3.6.	Классификация микропроцессоров по количеству выполняемых программ.	2,1	0,1	-	2
4	**Внутренняя структура микропроцессоров. Арифметико-логический блок.**	**7,3**	**0,3**	**-**	**7**
4.1.	Арифметико-логический блок.	2,1	0,1	-	2
4.2.	Основные характеристики микропроцессора.	2,1	0,1	-	2
4.3.	Структура типового микропроцессора.	3,1	0,1	-	3
5	**Устройство управления.**	**9,4**	**0,4**	**2**	**7**
5.1	Принцип управления.	2,1	0,1	-	2
5.2.	Организация устройств управления.	2,1	0,1	-	2
5.3.	Устройство управления микропроцессора.	2,1	0,1	-	2
5.4.	Особенности программного и микро-	3,1	0,1	2	1

	программного управления.				
6	**Система команд микропроцессора.**	**9,3**	**0,3**	**2**	**7**
6.1.	Оптимальная система команд.	2,1	0,1	-	2
6.2.	Формат команды.	4,1	0,1	2	2
6.3.	Классификация команд.	3,1	0,1	-	3
7	**Режимы работы микропроцессора.**	**7,3**	**0,3**	**-**	**7**
7.1.	Режимы работы микропроцессора. Защищенный режим. Реальный режим.	3,2	0,2	-	3
7.2.	Режим системного управления. Переключение между режимами.	4,1	0,1	-	4
8	**Принципы формирования адресного пространства.**	**7,3**	**0,3**	**-**	**7**
8.1.	Среда выполнения задачи. Архитектурные решения микропроцессоров.	7,3	0,3	-	7
9	**Система адресации.**	**9,3**	**0,3**	**2**	**7**
9.1.	Режимы адресации.	4,1	0,1	2	2
9.2.	Способы адресации.	4,1	0,1	-	2
9.3.	Режимы адресации операндов. Возможности микропроцессоров по адресации.	3,1	0,1	-	3
10	**Память как функциональный узел.**	**7,3**	**0,3**	**-**	**7**
10.1.	Внутренняя память компьютера.	2,1	0,1	-	2
10.2.	Технология производства интегральных схем.	2,1	0,1	-	2
10.3.	Основные характеристики полупроводниковой памяти.	3,1	0,1	-	3
11	**Принципы организации памяти.**	**7,3**	**0,3**	**-**	**7**
11.1.	Организация физической памяти. ОЗУ, ПЗУ и FLASH – память.	3,1	0,1	-	3
11.2.	Логическая структура основной памяти.	4,2	0,2	-	4
12	**Виртуальная память.**	**7,5**	**0,5**	**-**	**7**
12.1.	Модели памяти.	1,1	0,1	-	1
12.2.	Понятие о сегментированной модели памяти.	1,1	0,1	-	1
12.3.	Понятие о страничной модели памяти.	1,1	0,1	-	1
12.4.	Сегментно-страничная модель памяти.	2,1	0,1	-	2
12.5.	Плоская модель памяти.	2,1	0,1	-	2
13	**Прерывания.**	**9,4**	**0,4**	**2**	**7**
13.1.	Понятие прерывания.	2,1	0,1	-	2
13.2.	Классификация прерываний.	2,1	0,1	-	2
13.3.	Внешние прерывания.	2,1	0,1	-	2
13.4.	Система прерываний.	3,1	0,1	2	1
14	**Поддержка многозадачности.**	**7,4**	**0,4**	**-**	**7**
14.1.	Многозадачность – определение, история развития.	2,1	0,1	-	2
14.2.	Режимы многозадачности.	2,1	0,1	-	2
14.3.	Невытесняющая многозадачность.	2,1	0,1	-	2
14.4.	Многозадачность в защищенном режиме.	1,1	0,1	-	1
15	**Программы-отладчики.**	**7,3**	**0,3**	**-**	**7**
15.1	Методы и средства отладки. Отладка на примере процессора 486.	7,3	0,3	-	7
16	**Принципы программирования**	**9,3**	**0,3**	**2**	**7**

	микропроцессоров.				
16.1.	Программирование микропроцессора. Машинно-ориентированные языки.	5,1	0,1	2	3
16.2.	Языки высокого уровня. Специализированные языки.	4,2	0,2	-	4
17	**Команды языка Ассемблер**	**7,2**	**0,2**	**-**	**7**
17.1.	Основные группы операций. Мнемокоды команд процессора Pentium.	7,2	0,2	-	7
		135	**6**	**10**	**119**

- для ускоренной формы обучения:

№ раздела и темы	*Наименование раздела и тема дисциплины*	*Трудоемкость, (час.)*	*Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость; (час.)*		
			учебные занятия		*самостоятельная работа обучающихся**
			лекции	*лабораторные занятия*	
1	2	3	4	6	7
1	**Введение. История развития микропроцессоров.**	**7,4**	**0,4**	**-**	**7**
1.1.	Первое поколение ЭВМ. Характерные черты.	2,1	0,1	-	2
1.2.	Второе поколение ЭВМ.	2,1	0,1	-	2
1.3.	Третье поколение ЭВМ.	2,1	0,1	-	2
1.4	Четвертое поколение ЭВМ.	1,1	0,1		1
2.	**Внутренняя структура микропроцессоров. Принципы фон Неймана.**	**7,4**	**0,4**	**-**	**7**
2.1.	Определение микропроцессора.	2,1	0,1	-	2
2.2.	История развития микропроцессоров.	2,1	0,1	-	2
2.3.	Архитектура фон Неймана.	2,1	0,1	-	2
2.4.	Принципы фон Неймана.	1,1	0,1	-	1
3	**Классификация микропроцессоров.**	**7,6**	**0,6**	**-**	**7**
3.1.	Классификация микропроцессоров по числу больших интегральных схем.	1,1	0,1	-	1
3.2.	Классификация микропроцессоров по назначению.	1,1	0,1	-	1
3.3.	Классификация микропроцессоров по виду обрабатываемых сигналов.	1,1	0,1	-	1
3.4.	Классификация микропроцессоров по характеру временной организации.	1,1	0,1	-	1
3.5.	Классификация микропроцессоров по организации структуры.	1,1	0,1	-	1
3.6.	Классификация микропроцессоров по количеству выполняемых программ.	2,1	0,1	-	2
4	**Внутренняя структура микропроцессоров. Арифметико-логический блок.**	**7,3**	**0,3**	**-**	**7**
4.1.	Арифметико-логический блок.	2,1	0,1	-	2
4.2.	Основные характеристики микропроцессора.	2,1	0,1	-	2
4.3.	Структура типового микропроцессора.	3,1	0,1	-	3
5	**Устройство управления.**	**9,4**	**0,4**	**2**	**7**
5.1	Принцип управления.	2,1	0,1	-	2

5.2.	Организация устройств управления.	2,1	0,1	-	2
5.3.	Устройство управления микропроцессора.	2,1	0,1	-	2
5.4.	Особенности программного и микропрограммного управления.	3,1	0,1	2	1
6	**Система команд микропроцессора.**	**9,3**	**0,3**	**1**	**7**
6.1.	Оптимальная система команд.	2,1	0,1	-	2
6.2.	Формат команды.	4,1	0,1	1	2
6.3.	Классификация команд.	3,1	0,1	-	3
7	**Режимы работы микропроцессора.**	**7,3**	**0,3**	**-**	**7**
7.1.	Режимы работы микропроцессора. Защищенный режим. Реальный режим.	3,2	0,2	-	3
7.2.	Режим системного управления. Переключение между режимами.	4,1	0,1	-	4
8	**Принципы формирования адресного пространства.**	**7,3**	**0,3**	**-**	**7**
8.1.	Среда выполнения задачи. Архитектурные решения микропроцессоров.	7,3	0,3	-	7
9	**Система адресации.**	**9,3**	**0,3**	**1**	**7**
9.1.	Режимы адресации.	4,1	0,1	1	2
9.2.	Способы адресации.	4,1	0,1	-	2
9.3.	Режимы адресации операндов. Возможности микропроцессоров по адресации.	3,1	0,1	-	3
10	**Память как функциональный узел.**	**7,3**	**0,3**	**-**	**7**
10.1.	Внутренняя память компьютера.	2,1	0,1	-	2
10.2.	Технология производства интегральных схем.	2,1	0,1	-	2
10.3.	Основные характеристики полупроводниковой памяти.	3,1	0,1	-	3
11	**Принципы организации памяти.**	**7,3**	**0,3**	**-**	**7**
11.1.	Организация физической памяти. ОЗУ, ПЗУ и FLASH – память.	3,1	0,1	-	3
11.2.	Логическая структура основной памяти.	4,2	0,2	-	4
12	**Виртуальная память.**	**7,5**	**0,5**	**-**	**7**
12.1.	Модели памяти.	1,1	0,1	-	1
12.2.	Понятие о сегментированной модели памяти.	1,1	0,1	-	1
12.3.	Понятие о страничной модели памяти.	1,1	0,1	-	1
12.4.	Сегментно-страничная модель памяти.	2,1	0,1	-	2
12.5.	Плоская модель памяти.	2,1	0,1	-	2
13	**Прерывания.**	**9,4**	**0,4**	**1**	**9**
13.1.	Понятие прерывания.	3,1	0,1	-	3
13.2.	Классификация прерываний.	2,1	0,1	-	2
13.3.	Внешние прерывания.	2,1	0,1	-	2
13.4.	Система прерываний.	2,1	0,1	1	1
14	**Поддержка многозадачности.**	**7,4**	**0,4**	**-**	**7**
14.1.	Многозадачность – определение, история развития.	2,1	0,1	-	2
14.2.	Режимы многозадачности.	2,1	0,1	-	2
14.3.	Невытесняющая многозадачность.	2,1	0,1	-	2
14.4.	Многозадачность в защищенном режиме.	1,1	0,1	-	1

15	**Программы-отладчики.**	**7,3**	**0,3**	**-**	**7**
15.1	Методы и средства отладки. Отладка на примере процессора 486.	7,3	0,3	-	7
16	**Принципы программирования микропроцессоров.**	**9,3**	**0,3**	**1**	**8**
16.1.	Программирование микропроцессора. Машинно-ориентированные языки.	5,1	0,1	1	4
16.2.	Языки высокого уровня. Специализированные языки.	4,2	0,2	-	4
17	**Команды языка Ассемблер**	**7,2**	**0,2**	**-**	**7**
17.1.	Основные группы операций. Мнемокоды команд процессора Pentium.	7,2	0,2	-	7
		135	**6**	**6**	**121**

4.2. Содержание дисциплины, структурированное по разделам и темам

Тема 1. Введение. История развития микропроцессоров

1.1. Первое поколение

Появление электронно-вакуумной лампы позволило ученым претворить в жизнь идею создания вычислительной машины. Она появилась в 1946 году в США для решения задач и получила название ЭНИАК (ENIAC - Electronic Numerical Integrator and Calculator, в переводе "электронный численный интегратор и калькулятор). От неё начался отсчет пути, по которому пошло развитие ЭВМ.

Элементная база: электронно-вакуумные лампы, резисторы, конденсаторы. Соединение элементов - навесной монтаж проводами.

Габариты: ЭВМ выполнена в виде громоздких шкафов и занимает специальный машинный зал.

Быстродействие: 10-20 тыс. оп/с.

Эксплуатация слишком сложна из-за частого выхода из строя. Существует опасность перегрева ЭВМ.

Программирование: трудоёмкий процесс в машинных кодах. При этом необходимо знать все команды машины, их двоичное представление, а также различные структуры ЭВМ.

1.2. Второе поколение

Второе поколение пришлось на период от конца 50-х до конца 60-х годов. Был изобретён транзистор, который пришёл на смену электронным лампам. Это позволило изменить элементную базу ЭВМ.

Элементная база: полупроводниковые элементы. Соединение элементов - печатные платы и навесной монтаж.

Габариты: ЭВМ выполнена в виде однотипных стоек, чуть выше человеческого роста. Для их размещения требуется специально оборудованный машинный зал.

Производительность: до 1 млн. оп/с.

Эксплуатация: упростилась. Появились вычислительные центры с большим штатом обслуживающего персонала.

Программирование: существенно изменилось, так как велось преимущественно на алгоритмических языках. Программисты уже не работали в зале, а отдавали свои программы на перфокартах или магнитных лентах специально обученным операторам.

1.3. Третье поколение

Этот период длился с конца 60-х до конца 70-х годов. Появление интегральных схем ознаменовало новый этап в развитии вычислительной техники.

Первой ЭВМ, выполненной на интегральных схемах, была IBM-360 фирмы IBM.

Элементная база: интегральные схемы, которые вставляются в специальные гнёзда на печатной плате.

Габариты: для их размещения также требуется машинный зал.

Производительность: сотни тысяч - миллионы оп/с.

Эксплуатация: более оперативно производится ремонт стандартных неисправностей, но из-за большой сложности системной организации требуется штат высококвалифицированных специалистов. Незаменимую роль играет системный программист. Появились дисплеи и графопостроители.

Программирование: во многих вычислительных центрах появились дисплейные залы, где каждый программист в определенное время мог подсоединиться к ЭВМ в режиме разделения времени. Как и прежде, основным оставался режим пакетной обработки задач.

1.4. Четвёртое поколение

Этот период оказался самым длинным - от конца 70-х годов по настоящее время. Новые технологии создания интегральных схем позволили разработать ЭВМ четвертого поколения на больших интегральных схемах (БИС), степень интеграции которых составляет десятки и сотни тысяч элементов на одном кристалле

Элементная база: микропроцессоры. Первый микропроцессор был создан фирмой Intel в 1971 году. На одном кристалле удалось сформировать минимальный по составу аппаратуры процессор, содержащий 2250 транзисторов.

Габариты: персональные компьютеры, размещающиеся на столе пользователя.

Производительность: десятки - сотни миллионов оп/с.

Эксплуатация: совместимость программного обеспечения снизу вверх и принцип открытой архитектуры, предусматривающий возможность дополнения имеющихся аппаратных средств без смены старых или их модификации без замены всего компьютера.

Тема 2. Внутренняя структура микропроцессоров. Принципы фон Неймана.

2.1. Определение микропроцессора.

Микропроцессор (МП) - это программно-управляемое электронное цифровое устройство, предназначенное для обработки цифровой информации и управления процессом этой обработки, выполненное на одной или нескольких интегральных схемах с высокой степенью интеграции электронных элементов.

2.2. История развития микропроцессора.

В 1970 году Маршиан Эдвард Хофф из фирмы Intel сконструировал интегральную схему, аналогичную по своим функциям центральному процессору большой ЭВМ - первый микропроцессор Intel-4004, который уже в 1971 году был выпущен в продажу.

15 ноября 1971 г. можно считать началом новой эры в электронике. В этот день компания приступила к поставкам первого в мире микропроцессора Intel 4004.

Это был настоящий прорыв, ибо МП Intel-4004 размером менее 3 см был производительнее гигантской машины ENIAC. Правда, работал он гораздо медленнее и мог обрабатывать одновременно только 4 бита информации (процессоры больших ЭВМ обрабатывали 16 или 32 бита одновременно), но и стоил первый МП в десятки тысяч раз дешевле.

Кристалл представлял собой 4-разрядный процессор с классической архитектурой ЭВМ гарвардского типа и изготавливался по передовой р-канальной МОП технологии с проектными нормами 10 мкм. Электрическая схема прибора насчитывала 2300 транзисторов. МП работал на тактовой частоте 750 кГц при длительности цикла команд 10,8 мкс. Чип i4004 имел адресный стек (счетчик команд и три регистра стека типа LIFO), блок РОНов (регистры сверхоперативной памяти или регистровый файл - РФ), 4-разрядное параллельное АЛУ, аккумулятор, регистр команд с дешифратором команд и схемой управления, а также схему связи с внешними устройствами. Все эти функциональные узлы объединялись между собой 4-разрядной ШД. Память команд достигала 4 Кбайт (для сравнения: объем ЗУ миниЭВМ в начале 70-х годов редко превышал 16 Кбайт), а РФ ЦП насчитывал 16 4-разрядных регистров, которые можно было использовать и как 8 8-разрядных. Такая организация РОНов сохранена и в последующих МП фирмы Intel. Три регистра стека обеспечивали три уровня вложения подпрограмм. МП i4004 монтировался в пластмассовый или металлокерамический корпус типа DIP (Dual In-line Package) всего с 16 выводами.

В систему его команд входило всего 46 инструкций.

Вместе с тем кристалл располагал весьма ограниченными средствами ввода/вывода, а в системе команд отсутствовали операции логической обработки данных (И, ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ), в связи с чем их приходилось реализовывать с помощью специальных подпрограмм. Модуль i4004 не имел возможности останова (команды HALT) и обработки прерываний.

Цикл команды процессора состоял из 8 тактов задающего генератора. Была мультиплексированная ША (шина адреса)/ШД (шина данных), адрес 12-разрядный передавался по 4-разряда.

1 апреля 1972 г. фирма Intel начала поставки первого в отрасли 8-разрядного прибора i8008. Кристалл изготавливался по р-канальной МОП-технологии с проектными нормами 10 мкм и содержал 3500 транзисторов. Процессор работал на частоте 500 кГц при длительности машинного цикла 20 мкс (10 периодов задающего генератора).

В отличие от своих предшественников МП имел архитектуру ЭВМ принстонского типа, а в качестве памяти допускал применение комбинации ПЗУ и ОЗУ.

По сравнению с i4004 число РОН уменьшилось с 16 до 8, причем два регистра использовались для хранения адреса при косвенной адресации памяти (ограничение технологии - блок РОН аналогично кристаллам 4004 и 4040 в МП 8008 был реализован в виде динамической памяти). Почти вдвое сократилась длительность машинного цикла (с 8 до 5 состояний). Для синхронизации работы с медленными устройствами был введен сигнал готовности READY.

Система команд насчитывала 65 инструкций. МП мог адресовать память объемом 16 Кбайт. Его производительность по сравнению с четырехразрядными МП возросла в 2,3 раза. В среднем для сопряжения процессора с памятью и устройствами ввода/вывода требовалось около 20 схем средней степени интеграции.

Возможности р-канальной технологии для создания сложных высокопроизводительных МП были почти исчерпаны, поэтому "направление главного удара" перенесли на n-канальную МОП технологию.

1 апреля 1974 МП Intel 8080 был представлен вниманию всех заинтересованных лиц. Благодаря использованию технологии n-МОП с проектными нормами 6 мкм, на кристалле удалось разместить 6 тыс. транзисторов. Тактовая частота процессора была доведена до 2 МГц, а длительность цикла команд составила уже 2 мкс. Объем памяти, адресуемой процессором, был увеличен до 64 Кбайт. За счет использования 40-выводного корпуса уда-

лось разделить ША и ШД, общее число микросхем, требовавшихся для построения системы в минимальной конфигурации сократилось до 6 (рис. 1).

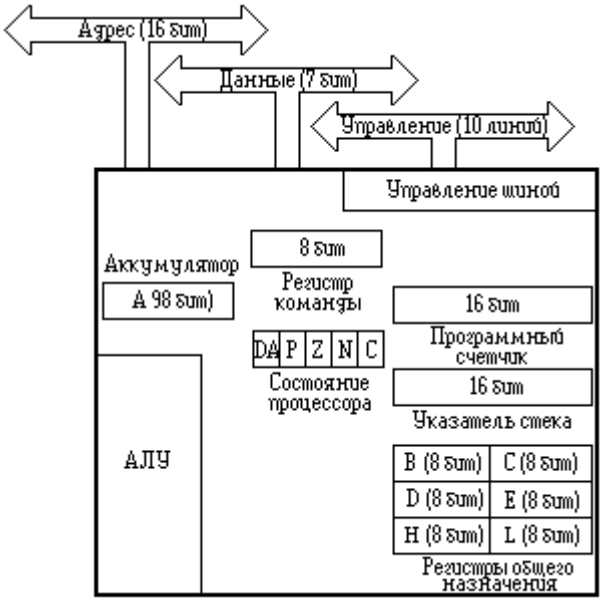

Рис. 1. Микропроцессор Intel 8080.

В РФ были введены указатель стека, активно используемый при обработке прерываний, а также два программнонедоступных регистра для внутренних пересылок. Блок РОНов был реализован на микросхемах статической памяти. Исключение аккумулятора из РФ и введение его в состав АЛУ упростило схему управления внутренней шиной.

Новое в архитектуре МП - использование многоуровневой системы прерываний по вектору. Такое техническое решение позволило довести общее число источников прерываний до 256 (до появления БИС контроллеров прерываний схема формирования векторов прерываний требовала применения до 10 дополнительных чипов средней интеграции). В i8080 появился механизм прямого доступа в память (ПДП) (как ранее в универсальных ЭВМ IBM System 360 и др.).

ПДП открыл зеленую улицу для применения в микроЭВМ таких сложных устройств, как накопители на магнитных дисках и лентах дисплеи на ЭЛТ, которые и превратили микроЭВМ в полноценную вычислительную систему.

Традицией компании, начиная с первого кристалла, стал выпуск не отдельного чипа ЦП, а семейства БИС, рассчитанных на совместное использование.

2.3. Архитектура фон Неймана.

Принципы фон Неймана – общие принципы, положенные в основу современных компьютеров.

Фон Нейман с соавторами выдвинули основные принципы логического устройства ЭВМ и предложили ее структуру, которая полностью воспроизводилась в течение первых двух поколений ЭВМ:

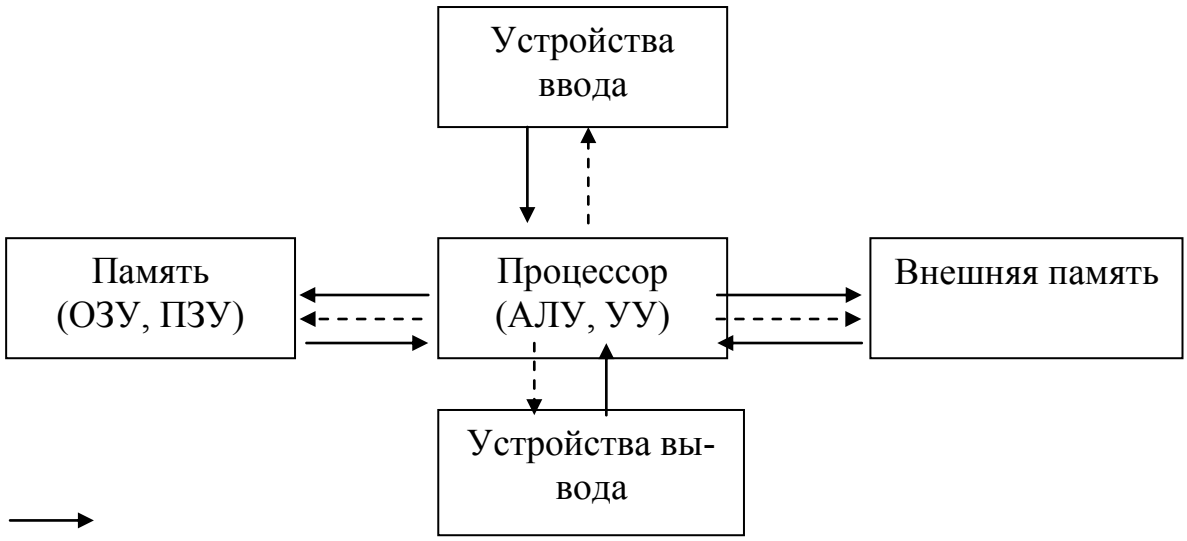

Рис. 2. Архитектура фон Неймана.

2.4. Принципы фон Неймана.

Кроме архитектуры ЭВМ Нейман предложил основополагающие принципы логического устройства ЭВМ.

1. Принцип программного управления. Он обеспечивает автоматизацию процессов вычислений на ЭВМ. Согласно этому принципу программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.

Как это выполняется? Введем 2 определения.

Регистр – специализированная дополнительная ячейка памяти в процессоре. Регистр выполняет функцию кратковременного хранения числа или команды.

Счетчик команд – регистр УУ, содержимое которого соответствует адресу очередной выполняемой команды, он служит для автоматической выборки программы из последовательных ячеек памяти. То есть, с его помощью осуществляется выборка программы из памяти. Этот регистр последовательно увеличивает хранимый в нем адрес очередной команды на длину команды.

А так как, команды программы расположены в памяти друг за другом, то тем самым осуществляется выборка цепочки команд из последовательно расположенных ячеек памяти.

Если же нужно после выполнения команды перейти не к следующей, а к какой – то другой, используются команды условного или безусловного переходов. Таким образом, процессор исполняет программу автоматически, без вмешательства человека.

2. Принцип однородности памяти. Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что храниться в данной ячейке памяти – число, текст или команда. Над командами можно выполнять такие же действия, как и над данными.

Иногда этот принцип называют «принцип хранимой команды». И это отсутствие принципиальной разницы между программой и данными дало возможность ЭВм самой формировать для себя программу в соответствии с результатом вычислений.

3. Принцип адресности. Структурно основная память состоит из пронумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка. Это позволяет обращаться к произвольной ячейке (адресу) без просмотра предыдущих.

Компьютеры, построенные на этих принципах, относят к типу фон - нейманских.

На сегодняшний день это подавляющие большинство компьютеров, в том числе и IBM PS – совместимые. Но есть и компьютерные системы с иной архитектурой – например системы для параллельных вычислений.

Тема 3. Классификация микропроцессоров.

Существует множество различных классификаций микропроцессоров. Остановимся на основных.

3.1. Классификация микропроцессоров по числу больших интегральных схем.

По числу больших интегральных схем (БИС) в микропроцессорном комплекте различают микропроцессоры однокристальные, многокристальные и многокристальные секционные.

Процессоры даже самых простых ЭВМ имеют сложную функциональную структуру, содержат большое количество электронных элементов и множество разветвленных связей. Изменять структуру процессора необходимо так, чтобы полная принципиальная схема или ее части имели количество элементов и связей, совместимое с возможностями БИС. При этом микропроцессоры приобретают внутреннюю магистральную архитектуру, т. е. в них к единой внутренней информационной магистрали подключаются все основные функциональные блоки (арифметико-логический, рабочих регистров, стека, прерываний, интерфейса, управления и синхронизации и др.).

Для обоснования классификации микропроцессоров по числу БИС надо распределить все аппаратные блоки процессора между основными тремя функциональными частями: операционной, управляющей и интерфейсной. Сложность операционной и управляющей частей процессора определяется их разрядностью, системой команд и требованиями к системе прерываний; сложность интерфейсной части разрядностью и возможностями подключения других устройств ЭВМ (памяти, внешних устройств, датчиков и исполнительных механизмов и др.). Интерфейс процессора содержит несколько десятков информационных шин данных (ШД), адресов (ША) и управления (ШУ).

Однокристальные микропроцессоры получаются при реализации всех аппаратных средств процессора в виде одной БИС или СБИС (сверхбольшой интегральной схемы). По мере увеличения степени интеграции элементов в кристалле и числа выводов корпуса параметры однокристальных микропроцессоров улучшаются. Однако возможности однокристальных микропроцессоров ограничены аппаратными ресурсами кристалла и корпуса. Для получения многокристального микропроцессора необходимо провести разбиение его логической структуры на функционально законченные части и реализовать их в виде БИС (СБИС). Функциональная законченность БИС многокристального микропроцессора означает, что его части выполняют заранее определенные функции и могут работать автономно.

На рис. 3,а показано функциональное разбиение структуры процессора при создании трехкристального микропроцессора (пунктирные линии), содержащего БИС операционного (ОП), БИС управляющего (УП) и БИС интерфейсного (ИП) процессоров.

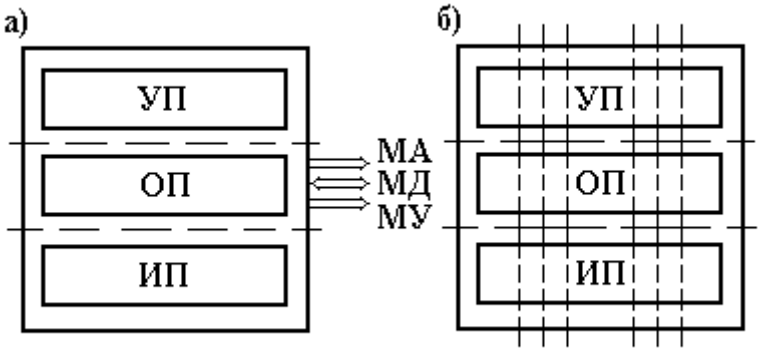

Рис. 3. Функциональная структура процессора (а) и ее разбиение для реализации процессора в виде комплекта секционных БИС (б).

Для создания высокопроизводительных многоразрядных микропроцессоров требуется столь много аппаратных средств, не реализуемых в доступных БИС, что может возникнуть необходимость еще и в функциональном разбиении структуры микропроцессора горизонтальными плоскостями. В результате рассмотренного функционального разделения структуры микропроцессора на функционально и конструктивно законченные части создаются условия реализации каждой из них в виде БИС. Все они образуют комплект секционных БИС МП.

3.2. Классификация микропроцессоров по назначению.

По назначению различают универсальные и специализированные микропроцессоры.

Универсальные микропроцессоры могут быть применены для решения широкого круга разнообразных задач. При этом их эффективная производительность слабо зависит от проблемной специфики решаемых задач. Специализация МП, т.е. его проблемная ориентация на ускоренное выполнение определенных функций позволяет резко увеличить эффективную производительность при решении только определенных задач.

Среди специализированных микропроцессоров можно выделить различные микроконтроллеры, ориентированные на выполнение сложных последовательностей логических операций, математические МП, предназначенные для повышения производительности при выполнении арифметических операций за счет, например, матричных методов их выполнения, МП для обработки данных в различных областях применений и т. д. С помощью специализированных МП можно эффективно решать новые сложные задачи параллельной обработки данных. Например, конволюция позволяет осуществить более сложную математическую обработку сигналов, чем широко используемые методы корреляции. Последние в основном сводятся к сравнению всего двух серий данных: входных, передаваемых формой сигнала, и фиксированных опорных и к определению их подобия. Конволюция дает возможность в реальном масштабе времени находить соответствие для сигналов изменяющейся формы путем сравнения их с различными эталонными сигналами, что, например, может позволить эффективно выделить полезный сигнал на фоне шума.

Разработанные однокристальные конвольверы используются в устройствах опознавания образов в тех случаях, когда возможности сбора данных превосходят способности системы обрабатывать эти данные.

3.3 Классификация микропроцессоров по виду обрабатываемых сигналов.

По виду обрабатываемых входных сигналов различают цифровые и аналоговые микропроцессоры. Сами микропроцессоры цифровые устройства, однако могут иметь встроенные аналого-цифровые и цифро-аналоговые преобразователи. Поэтому входные аналоговые сигналы передаются в МП через преобразователь в цифровой форме, обрабатываются и после обратного преобразования в аналоговую форму поступают на выход. С архитектурной точки зрения такие микропроцессоры представляют собой аналоговые функциональные преобразователи сигналов и называются аналоговыми микропроцессорами. Они выполняют функции любой аналоговой схемы (например, производят генерацию колебаний, модуляцию, смещение, фильтрацию, кодирование и декодирование сигналов в реальном масштабе времени и т.д., заменяя сложные схемы, состоящие из операционных усилителей, катушек индуктивности, конденсаторов и т.д.). При этом применение аналогового микропроцессора значительно повышает точность обработки аналоговых сигналов и их воспроизводимость, а также расширяет функциональные возможности за счет программной "настройки" цифровой части микропроцессора на различные алгоритмы обработки сигналов.

Обычно в составе однокристальных аналоговых МП имеется несколько каналов аналого-цифрового и цифро-аналогового преобразования. В аналоговом микропроцессоре разрядность обрабатываемых данных достигает 24 бит и более, большое значение уделяется увеличению скорости выполнения арифметических операций.

Отличительная черта аналоговых микропроцессоров способность к переработке большого объема числовых данных, т. е. к выполнению операций сложения и умножения с большой скоростью при необходимости даже за счет отказа от операций прерываний и переходов. Аналоговый сигнал, преобразованный в цифровую форму, обрабатывается в реальном масштабе времени и передается на выход обычно в аналоговой форме через цифро-аналоговый преобразователь. При этом согласно теореме Котельникова частота квантования аналогового сигнала должна вдвое превышать верхнюю частоту сигнала.

3.4. Классификация микропроцессоров по характеру временной организации.

По характеру временной организации работы микропроцессоры делят на синхронные и асинхронные.

Синхронные микропроцессоры - микропроцессоры, в которых начало и конец выполнения операций задаются устройством управления (время выполнения операций в этом случае не зависит от вида выполняемых команд и величин операндов).

Асинхронные микропроцессоры позволяют начало выполнения каждой следующей операции определить по сигналу фактического окончания выполнения предыдущей операции. Для более эффективного использования каждого устройства микропроцессорной системы в состав асинхронно работающих устройств вводят электронные цепи, обеспечивающие автономное функционирование устройств. Закончив работу над какой-либо операцией, устройство вырабатывает сигнал запроса, означающий его готовность к выполнению следующей операции. При этом роль естественного распределителя работ принимает на себя память, которая в соответствии с заранее установленным приоритетом выполняет запросы остальных устройств по обеспечению их командной информацией и данными.

3.5. Классификация микропроцессоров по организации структуры.

По организации структуры микропроцессорных систем различают микроЭВМ одно- и многомагистральные.

В одномагистральных микроЭВМ все устройства имеют одинаковый интерфейс и подключены к единой информационной магистрали, по которой передаются коды данных, адресов и управляющих сигналов.

В многомагистральных микроЭВМ устройства группами подключаются к своей информационной магистрали. Это позволяет осуществить одновременную передачу информационных сигналов по нескольким (или всем) магистралям. Такая организация систем усложняет их конструкцию, однако увеличивает производительность.

3.6. Классификация микропроцессоров по количеству выполняемых программ.

По количеству выполняемых программ различают одно- и многопрограммные микропроцессоры.

В однопрограммных микропроцессорах выполняется только одна программа. Переход к выполнению другой программы происходит после завершения текущей программы.

В много- или мультипрограммных микропроцессорах одновременно выполняется несколько (обычно несколько десятков) программ. Организация мультипрограммной работы микропроцессорных управляющих систем позволяет осуществить контроль за состоянием и управлением большим числом источников или приемников информации.

Тема 4. Внутренняя структура микропроцессора. Арифметико-логический блок.

4.1. Арифметико-логический блок.

Арифметико-логический блок выполняет арифметические и логические операции под воздействием устройства управления МП БИС. Он включает в себя 8-разрядное АЛУ, схему десятичной коррекции ДК, построенной на базе ПЗУ, 5-ти разрядный регистр признаков, аккумулятор А, буфер аккумулятора БФА и буферный регистр БФРг. Арифметико-логический блок позволяет осуществить арифметические операции сложения, вычитания, а также основные логические операции (И, ИЛИ, исключающее ИЛИ) и сдвиг. При проведении операции одно число всегда берется из буфера аккумулятора, а другое - из буферного регистра. По результату выполнения арифметико-логических операций АЛБ устанавливает в регистре признаков пять знаков.

Признак переноса (Carry - C) устанавливается в единицу, если при выполнении команд появляется единица переноса из старшего разряда.

Дополнительный признак переноса (Auxiliary carry - AC) устанавливается в единицу, если при выполнении команд возникает единица переноса из третьего разряда числа. Состояние разряда может быть проанализировано лишь командой десятичной коррекции числа.

Признак знака (Sign - S) в машинном слове можно представить числом от -128 до 127. В этом случае седьмой (старший) разряд числа - его знак. Единица в седьмом разряде при такой записи будет указывать на отрицательное число, а ноль- на положительное.

В разряд нулевого признака (Zero -Z) записывается единица, если при выполнении команды результат равен нулю.

В разряд признака четности (Parity - P) записывается единица, если при выполнении команды количество единиц в разрядах результата будет четным.

8-битное АЛУ может выполнять арифметические операции сложения, вычитания, умножения и деления; логические операции И, ИЛИ, исключающее ИЛИ, а также операции циклического сдвига, сброса, инвертирования и т.п. В АЛУ имеются программно недоступные регистры Т1 и Т2, предназначенные для временного хранения операндов, схема десятичной коррекции и схема формирования признаков.

Простейшая операция сложения используется в АЛУ для инкрементирования содержимого регистров, продвижения регистра-указателя данных и автоматического вычисления следующего адреса РПП. Простейшая операция вычитания используется в АЛУ для декрементирования регистров и сравнения переменных.

Простейшие операции автоматически образуют "тандемы" для выполнения в АЛУ таких операций, как, например, инкрементирование 16-битных регистровых пар. В АЛУ реализуется механизм каскадного выполнения простейших операций для реализации сложных команд. Так, например, при выполнении одной из команд условной передачи правления по результату сравнения в АЛУ трижды инкрементируется СК, дважды произво-

дится чтение из РПД, выполняется арифметическое сравнение двух переменных, формируется 16-битный адрес перехода и принимается решение о том, делать или не делать переход по программе. Все перечисленные операции выполняются в АЛУ всего лишь за 2 мкс.

Важной особенностью АЛУ является его способность оперировать не только байтами, но и битами. Отдельные программно-доступные биты могут быть установлены, сброшены, инвертированы, переданы, проверены и использованы в логических операциях. Эта способность АЛУ, оперировать битами, столь важна, что во многих описаниях Intel87C51FB говорится о наличии в нем "булевского процессора". Для управления объектами часто применяются алгоритмы, содержащие операции над входными и выходными булевскими переменными (истина/ложь), реализация которых средствами обычных микропроцессоров сопряжена с определенными трудностями.

Таким образом, АЛУ может оперировать четырьмя типами информационных объектов: булевскими (1 бит), цифровыми (4 бита), байтными (8 бит) и адресными (16 бит). В АЛУ выполняется 51 различная операция пересылки или преобразования этих данных. Так как используется 11 режимов адресации (7 для данных и 4 для адресов), то путем комбинирования "операция/ режим адресации" базовое число команд 111 расширяется до 255 из 256 возможных при однобайтном коде операции.

4.2. Основные характеристики микропроцессора.

Микропроцессор характеризуется:

1) тактовой частотой, определяющей максимальное время выполнения переключения элементов в ЭВМ;

2) разрядностью, т.е. максимальным числом одновременно обрабатываемых двоичных разрядов.

Разрядностть МП обозначается m/n/k/ и включает: m - разрядность внутренних регистров, определяет принадлежность к тому или иному классу процессоров; n - разрядность шины данных, определяет скорость передачи информации;

k - разрядность шины адреса, определяет размер адресного пространства. Например, МП i8088 характеризуется значениями m/n/k=16/8/20; 3) архитектурой. Понятие архитектуры микропроцессора включает в себя систему команд и способы адресации, возможность совмещения выполнения команд во времени, наличие дополнительных устройств в составе микропроцессора, принципы и режимы его работы. Выделяют понятия микроархитектуры и макроархитектуры.

Микроархитектура микропроцессора - это аппаратная организация и логическая структура микропроцессора, регистры, управляющие схемы, арифметико-логические устройства, запоминающие устройства и связывающие их информационные магистрали.

Макроархитектура - это система команд, типы обрабатываемых данных, режимы адресации и принципы работы микропроцессора.

В общем случае под архитектурой ЭВМ понимается абстрактное представление машины в терминах основных функциональных модулей, языка ЭВМ, структуры данных.

4.3. Структура типового микропроцессора

Архитектура типичной небольшой вычислительной системы на основе микроЭВМ показана на рис. 4.1. Такая микроЭВМ содержит все 5 основных блоков цифровой машины: устройство ввода информации, управляющее устройство (УУ), арифметико-логическое устройство (АЛУ) (входящие в состав микропроцессора), запоминающие устройства (ЗУ) и устройство вывода информации.

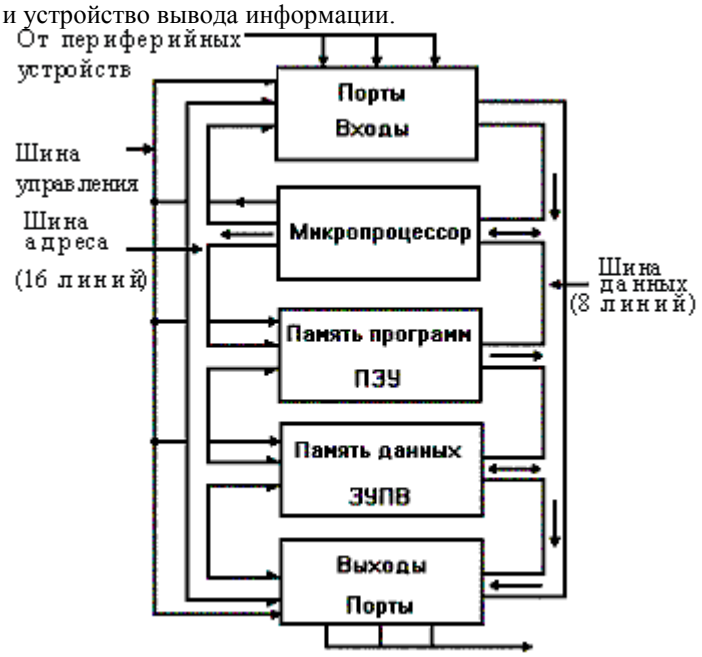

Рис. 4.1. Архитектура типового микропроцессора.

Микропроцессор координирует работу всех устройств цифровой системы с помощью шины управления (ШУ). Помимо ШУ имеется 16-разрядная адресная шина (ША), которая служит для выбора определенной ячейки памяти, порта ввода или порта вывода. По 8-разрядной информационной шине или шине данных (ШД) осуществляется двунаправленная пересылка данных к микропроцессору и от микропроцессора. Важно отметить, что МП

может посылать информацию в память микроЭВМ или к одному из портов вывода, а также получать информацию из памяти или от одного из портов ввода.
Постоянное запоминающее устройство (ПЗУ) в микроЭВМ содержит некоторую программу (на практике программу инициализации ЭВМ). Программы могут быть загружены в запоминающее устройство с произвольной выборкой (ЗУПВ) и из внешнего запоминающего устройства (ВЗУ). Это программы пользователя.

Тема 5 . Устройство управления.

5.1. Принцип управления.

Принцип микропрограммного управления был предложен М. Уилксом в 1951 г и с того времени практически не претерпел никаких изменений. Согласно ему любое электронное устройство (начиная от самого современного микропроцессора и заканчивая электронными часами) может быть представлено в виде пары: устройство управления — операционное устройство, где на долю управляющего устройства приходится полный контроль работы операционного устройства (сюда входит анализ текущего состояния устройства, контроль параметров, принятие решений и т.д.). На долю операционного устройства приходится вся "грязная работа": непосредственно осуществление полученных команд и преобразование контролируемых величин (в том числе и своего состояния) в удобный для управляющего устройства вид. Подводный камень кроется в описании поведения управляющего устройства, собственно именно это и представляет наибольшую сложность (и наибольшие затраты на разработку и реализацию), по сравнению с которой любое сколь угодно сложное контролируемое устройство бледнеет; и именно разработка простых и эффективных устройств управления пожалуй является основным направлением схемотехники.

5.2. Организация устройств управления.

Что ж, перейдем непосредственно к рассмотрению самих методов организации устройств управления. Самым старым, но как ни странно до сих пор актуальным, является организация управляющего устройства в виде конечного автомата, т.е. в виде устройства обладающего памятью, и результат работы которого зависит от его текущего состояния. УУ (т.е. управляющее устройство) получает сигналы о текущем состоянии — вектор X, и выдает управляющие сигналы — вектор У, переход из одного состояния в другое происходит по тактовым импульсам — Clock (т. е. схема является синхронной, что в общем-то необязательно). Операционный же автомат контролирует поведение чего-то, а может быть только себя с выдачей результатов работы, на основе полученных сигналов.
На текущий момент можно выделить три группы управляющих автоматов:

- автоматы с жесткой логикой, например Мили(Mealy) и Мура(Moor);
- автоматы с программируемой логикой (на сегодняшний день изжили себя как класс и могут рассматриваться лишь в историческом контексте): автоматы с принудительной адресацией, естественной адресацией и соответственно комбинированной;
- композиционные устройства управления — объединяют преимущества двух предыдущих типов устройств, однако они, на мой взгляд, так же уже не актуальны на сегодняшний день.

Пусть их применение не всегда целесообразно, однако можно с уверенностью сказать, что элемент управления выполненный в виде конечного автомата будет наиболее быстрым и наиболее экономичным (если, конечно не придется вводить для него новую микросхему) решением.

5.3. Устройство управления микропроцессора.

Поговорим о других распространенных устройствах управления — микропроцессорах и микроконтроллерах, первые используются в стационарных решениях и там где требуется большая вычислительная мощность, вторые же используются в портативных маломощных (как по питанию так и по вычислительной мощности) решениях; причем последние обладают дополнительным специфическим набором функций (в зависимости от специализации.
ЦП — собственно процессорный элемент; ША, ШД, ШУ — шины адреса, данных, управления соответственно; ОЗУ — текущая память (обычно используется для хранения результатов работы); ПЗУ — обычно содержит набор команд определяющих работу ЦП; В/В 1 … В/В N — устройства вода вывода. При чем для микроконтроллера, эта схема будет несколько отличаться: так например память находится на одном кристалле с ЦП (а так же все прочие непоказанные устройства — контроллер памяти, портов ввода/вывода, прерываний и т.д.) устройства подключенные к портам ввода/вывода не могут самостоятельно управлять шиной и т.п.
Собственно программирование поведения микропроцессора (в дальнейшем МП)/микроконтроллера (в дальнейшем МК) выполняется записью необходимой информации в ПЗУ со стартового адреса. Собственно в этом и заключается еще одно отличие МК и МП — поведение МК практически полностью определяется содержимым его ПЗУ (в большинстве своем они в состоянии выполнять команды только из адресного пространства ПЗУ); МП же могут изменять свое поведение загружая различные программы в ОЗУ, что делает их более гибкими. Однако это нельзя поставить в недостаток МК, т.к. они по определению являются компактными устройствами и не могут иметь большой объем памяти. МК и МП выпускаются в большом количестве и разнообразии для всевозможных задач, что делает возможным подобрать устройство наиболее подходящее по параметрам и соответственно облегчить жизнь разработчику.
К положительным сторонам МК и МП можно отнести сравнительную простоту разработки решений на их базе — наличие большого числа стандартных схем сопровождения, простоту программирования, наличия специали-

зированных программных продуктов для них, стандартность большинства решений (для проектирования большого числа проектов управляющая часть не понесет никаких изменений). К недостаткам можно отнести то, что даже при разработке минимальной схемы необходимо будет весь необходимый минимум микросхем — сателлит (различные контроллеры и т.д.).
Еще одна система управления появившаяся сравнительно недавно — управление с помощью ЭВМ.
Коды операции команд программы, воспринимаемые управляющей частью микропроцессора, расшифрованные и преобразованные в ней, дают информацию о том, какие операции надо выполнить, где в памяти расположены данные, куда надо направить результат и где расположена следующая за выполняемой команда.
Управляющее устройство имеет достаточно средств для того, чтобы после восприятия и интерпретации информации, получаемой в команде, обеспечить переключение (срабатывание) всех требуемых функциональных частей машины, а также для того, чтобы подвести к ним данные и воспринять полученные результаты. Именно срабатывание, т. е. изменение состояния двоичных логических элементов на противоположное, позволяет посредством коммутации вентилей выполнять элементарные логические и арифметические действия, а также передавать требуемые операнды в функциональные части микроЭВМ.
Устройство управления в строгой последовательности в рамках тактовых и цикловых временных интервалов работы микропроцессора (такт - минимальный рабочий интервал, в течение которого совершается одно элементарное действие; цикл - интервал времени, в течение которого выполняется одна машинная операция) осуществляет: выборку команды; интерпретацию ее с целью анализа формата, служебных признаков и вычисления адреса операнда (операндов); установление номенклатуры и временной последовательности всех функциональных управляющих сигналов; генерацию управляющих импульсов и передачу их на управляющие шины функциональных частей микроЭВМ и вентили между ними; анализ результата операции и изменение своего состояния так, чтобы определить месторасположение (адрес) следующей команды.

5.4. Особенности программного и микропрограммного управления.

В микропроцессорах используют два метода выработки совокупности функциональных управляющих сигналов: программный и микропрограммный.
Выполнение операций в машине сводится к элементарным преобразованиям информации (передача информации между узлами в блоках, сдвиг информации в узлах, логические поразрядные операции, проверка условий и т.д.) в логических элементах, узлах и блоках под воздействием функциональных управляющих сигналов блоков (устройств) управления. Элементарные преобразования, неразложимые на более простые, выполняются в течение одного такта сигналов синхронизации и называются микрооперациями.
В аппаратных (схемных) устройствах управления каждой операции соответствует свой набор логических схем, вырабатывающих определенные функциональные сигналы для выполнения микроопераций в определенные моменты времени. При этом способе построения устройства управления реализация микроопераций достигается за счет однажды соединенных между собой логических схем, поэтому ЭВМ с аппаратным устройством управления называют ЭВМ с жесткой логикой управления. Это понятие относится к фиксации системы команд в структуре связей ЭВМ и означает практическую невозможность каких-либо изменений в системе команд ЭВМ после ее изготовления.
При микропрограммной реализации устройства управления в состав последнего вводится ЗУ, каждый разряд выходного кода которого определяет появление определенного функционального сигнала управления. Поэтому каждой микрооперации ставится в соответствие свой информационный код - микрокоманда. Набор микрокоманд и последовательность их реализации обеспечивают выполнение любой сложной операции. Набор микроопераций называют микропрограммами. Способ управления операциями путем последовательного считывания и интерпретации микрокоманд из ЗУ (наиболее часто в виде микропрограммного ЗУ используют быстродействующие программируемые логические матрицы), а также использования кодов микрокоманд для генерации функциональных управляющих сигналов называют микропрограммным, а микроЭВМ с таким способом управления - микропрограммными или с хранимой (гибкой) логикой управления.
К микропрограммам предъявляют требования функциональной полноты и минимальности. Первое требование необходимо для обеспечения возможности разработки микропрограмм любых машинных операций, а второе связано с желанием уменьшить объем используемого оборудования. Учет фактора быстродействия ведет к расширению микропрограмм, поскольку усложнение последних позволяет сократить время выполнения команд программы.
Преобразование информации выполняется в универсальном арифметико-логическом блоке микропроцессора. Он обычно строится на основе комбинационных логических схем.
Для ускорения выполнения определенных операций вводятся дополнительно специальные операционные узлы (например, циклические сдвигатели). Кроме того, в состав микропроцессорного комплекта (МПК) БИС вводятся специализированные оперативные блоки арифметических расширителей.
Операционные возможности микропроцессора можно расширить за счет увеличения числа регистров. Если в регистровом буфере закрепление функций регистров отсутствует, то их можно использовать как для хранения данных, так и для хранения адресов. Подобные регистры микропроцессора называются регистрами общего назначения (РОН). По мере развития технологии реально осуществлено изготовление в микропроцессоре 16, 32 и более регистров.
В целом же, принцип микропрограммного управления (ПМУ) включает следующие позиции: 1) любая операция, реализуемая устройством, является последовательностью элементарных действий - микроопераций; 2) для управления порядком следования микроопераций используются логические условия; 3) процесс выполнения операций в устройстве описывается в форме алгоритма, представляемого в терминах микроопераций и логиче-

ских условий, называемого микропрограммой; 4) микропрограмма используется как форма представления функции устройства, на основе которой определяются структура и порядок функционирования устройства во времени.
ПМУ обеспечивает гибкость микропроцессорной системы и позволяет осуществлять проблемную ориентацию микро- и миниЭВМ.

Тема 6. Система команд микропроцессора.

6.1. Оптимальная система команд.

Проектирование системы команд оказывает влияние на структуру ЭВМ. Оптимальную систему команд иногда определяют как совокупность команд, которая удовлетворяет требованиям проблемно-ориентированных применений таким образом, что избыточность аппаратных и аппаратно-программных средств на реализацию редко используемых команд оказывается минимальной. В различных программах ЭВМ частота появления команд различна; например, по данным фирмы DEC в программах для ЭВМ семейства PDP-11 наиболее часто встречается команда передачи MOV(B), на ее долю приходится приблизительно 32% всех команд в типичных программах. Систему команд следует выбирать таким образом, чтобы затраты на редко используемые команды были минимальными.
При наличии статистических данных можно разработать (выбрать) ЭВМ с эффективной системой команд. Одним из подходов к достижению данной цели является разработка команд длиной в одно слово и кодирование их таким образом, чтобы разряды таких коротких команд использовать оптимально, что позволит сократить время реализации программы и ее длину.
Другим подходом к оптимизации системы команд является использование микроинструкций. В этом случае отдельные биты или группы бит команды используются для кодирования нескольких элементарных операций, которые выполняются в одном командном цикле. Эти элементарные операции не требуют обращения к памяти, а последовательность их реализации определяется аппаратной логикой.
Сокращение времени выполнения программ и емкости памяти достигается за счет увеличения сложности логики управления.

6.2. Формат команды.

Важной характеристикой команды является ее формат, определяющий структурные элементы команды, каждый из которых интерпретируется определенные образом при ее выполнении. Среди таких элементов (полей) команды выделяют следующие: код операции, определяющий выполняемое действие; адрес ячейки памяти, регистра процессора, внешнего устройства; режим адресации; операнд при использовании непосредственной адресации; код анализируемых признаков для команд условного перехода.

6.3. Классификация команд.

Классификация команд по основным признакам представлена на рис. 7.1. Важнейшим структурным элементом формата любой команды является код операции (КОП), определяющей действие, которое должно быть выполнено. Большое число КОП в процессоре очень важно, так как аппаратная реализация команд экономит память и время. Но при выборе ЭВМ необходимо концентрировать внимание на полноте операций с конкретными типами данных, а не только на числе команд, на доступных режимах адресации. Число бит, отводимое под КОП, является функцией полного набора реализуемых команд.

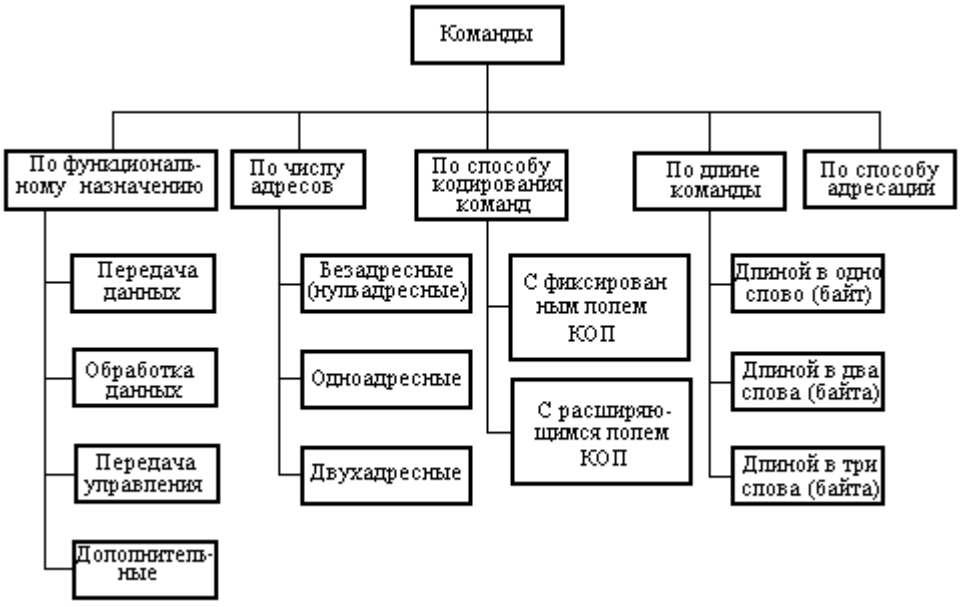

Рис. 6.1. Классификация команд.

При использовании фиксированного числа бит под КОП для кодирования всех m команд необходимо в поле КОП выделить двоичных разрядов. Однако, учитывая ограниченную длину слова мини- и микроЭВМ, различное функциональное назначение команд, источники и приемники результатов операций, а также то, что не все команды содержат адресную часть для обращения к памяти и периферийным устройствам, в малых ЭВМ для

кодирования команд широко используется принцип кодирования с переменным числом бит под поле КОП для различных групп команд.
В некоторых командах необходим только один операнд и они называются однооперандными (или одноадресными) командами в отличие от двухоперандных (или двухадресных), в которых требуются два операнда. При наличии двух операндов командой обычно изменяется только один из них. Так как информация берется только из одной ячейки, эту ячейку называются источником; ячейка, содержимое которой изменяется, называется приемником.
Ниже приведен формат двухадресной (двухоперандной) команды процессоров СМ.

Рис. 6.2. Формат команд процессоров СМ:
а) двухадресная команда;
б) одноадресная команда.

Тема 7. Режимы работы микропроцессора.

7.1. Режимы работы микропроцессора. Защищенный режим. Реальный режим.

Впервые о различных режимах работы процессоров стали говорить с появлением процессора 80286. Это был первый представитель данного семейства процессоров, в котором были реализованы многозадачность и защищенная архитектура. Чтобы обеспечить совместимость с предыдущими представителями этого семейства (8086/88, 80186/188) в процессоре 80286 было реализовано два режима функционирования: режим эмуляции 8086 (режим реального адреса) и защищенный режим, в котором используются все возможности процессора. В последующих поколениях процессоров этого семейства защищенный режим становится основным режимом работы.

В новых поколениях процессоров Intel появился еще один режим работы - режим системного управления. Впервые он был реализован в процессорах 80386SL и i486SL. Начиная с расширенных моделей Intel486, этот режим стал обязательным элементом архитектуры IA-32. С его помощью прозрачно даже для операционной системы на уровне BIOS реализуются функции энергосбережения.

Основным режимом работы микропроцессора является защищенный режим. Ключевыми особенностями защищенного режима являются: виртуальное адресное пространство, защита и многозадачность.
В защищенном режиме программа оперирует адресами, которые могут относиться к физически отсутствующим ячейкам памяти, поэтому такое адресное пространство называется виртуальным. Размер виртуального адресного пространства программы может превышать емкость физической памяти и достигать 64Тбайт. Для адресации виртуального адресного пространства используется сегментированная модель, в которой адрес состоит из двух элементов: селектора сегмента и смещения внутри сегмента. С каждым сегментом связана особая структура, хранящая информацию о нем, - дескриптор. Кроме "виртуализации" памяти на уровне сегментов существует возможность "виртуализации" памяти при помощи страниц - страничная трансляция. Страничная трансляция предоставляет удобные средства для реализации в операционной системе функций подкачки, а кроме того в процессорах P6+ обеспечивает 36-битную физическую адресацию памяти (64Гбайт).
Встроенные средства переключения задач обеспечивают многозадачность в защищенном режиме. Среда задачи состоит из содержимого регистров МП и всего кода с данными в пространстве памяти. Микропроцессор способен быстро переключаться из одной среды выполнения в другую, имитируя параллельную работу нескольких задач. Для некоторых задач может эмулироваться управление памятью как у процессора 8086. Такое состояние задачи называется режимом виртуального 8086 (Virtual 8086 Mode). О пребывании задачи в таком состоянии сигнализирует бит VM в регистре флагов. При этом задачи виртуального МП 8086 изолированы и защищены, как от друг друга, так и от обычных задач защищенного режима.
Защита задач обеспечивается следующими средствами: контроль пределов сегментов, контроль типов сегментов, контроль привилегий, привилегированные инструкции и защита на уровне страниц. Контроль пределов и типов сегментов обеспечивает целостность сегментов кода и данных. Программа не имеет права обращаться к виртуальной памяти, выходящей за предел того или иного сегмента. Программа не имеет права обращаться к сегменту данных как к коду и наоборот. Архитектура защиты микропроцессора обеспечивает 4 иерархических уровня привилегий, что позволяет ограничить задаче доступ к отдельным сегментам в зависимости от ее текущих привилегий. Кроме того, текущий уровень привилегий задачи влияет на возможность выполнения тех или иных специфических команд (привилегированных инструкций). Функции страничной трансляции, впервые появившиеся в МП Intel386, обеспечивают дополнительные механизмы защиты на уровне страниц.

В реальном режиме микропроцессор работает как очень быстрый 8086 с возможностью использования 32-битных расширений. Механизм адресации, размеры памяти и обработка прерываний (с их последовательными ограничениями) МП Intel386 в реальном режиме полностью совпадают с аналогичными функциями МП 8086. В отличие от 8086 микропроцессоры 286+ в определенных ситуациях генерируют исключения, например, при превышении предела сегмента, который для всех сегментов в реальном режиме - 0FFFFh.
Имеется две фиксированные области в памяти, которые резервируются в режиме реальной адресации:

- область инициализации системы
- область таблицы прерываний

Ячейки от 00000h до 003FFH резервируются для векторов прерываний. Каждое из 256 возможных прерываний имеет зарезервированный 4-байтовый адрес перехода. Ячейки от FFFFFFF0H до FFFFFFFFH резервируются для инициализации системы.

7.2. Режим системного управления (System Management Mode). Переключение между режимами

Режим системного управления предназначен для выполнения некоторых действий с возможностью их полной изоляции от прикладного программного обеспечения и даже операционной системы. Переход в этот режим возможен только аппаратно. Когда процессор находится в режиме SMM, он выставляет сигнал SMIACT#. Этот сигнал может служить для включения выделенной области физической памяти (System Management RAM), так что память SMRAM можно сделать доступной только для этого режима. При входе в режим SMM процессор сохраняет свой контекст в SMRAM (контекст сопроцессора не сохраняется) по адресу SMM Base и передает управление процедуре, называемой обработчиком System Management Interrupt, по адресу SMM Base+8000h (по умолчанию SMM Base содержит значение 30000h). Состояние процессора в этот момент точно определено: EFLAGS обнулен (кроме зарезервированных битов), сегментные регистры содержат селектор 0000, базы сегментов установлены в 00000000, пределы - 0FFFFFFFFh.
Следует отметить, что в режиме SMM не предусмотрена работа с прерываниями и особыми случаями: прерывания по IRQ и SMI# замаскированы, пошаговые ловушки и точки останова отключены, обработка прерывания по NMI откладывается до выхода из режима SMM. Если необходимо обеспечить работу с прерываниями или особыми случаями, то надо инициализировать IDT и разрешить прерывания, выставив флаг IF в регистре EFLAGS. Прерывания по NMI будут разблокированы автоматически после первой же команды IRET.
При возврате из SMM (по инструкции RSM) процессор восстанавливает свой контекст из SMRAM. Обработчик может программно внести изменения в образ контекста процессора, тогда процессор перейдет не в то состояние, в котором произошло SMI. Если SMI было получено во время выполнения инструкции HLT, то дальнейшие действия при выходе из SMM определяются значением поля "Auto HALT Restart": процессор может снова вернуться к инструкции останова или перейти к выполнению следующей команды. Если SMI произошло при выполнении инструкции ввода-вывода, то в зависимости от значения поля "I/O Instruction Restart" возможен рестарт инструкции ввода вывода.
Эти особенности режима системного управления позволяют использовать его для реализации системы управления энергосбережением компьютера или функций безопасности и контроля доступа.

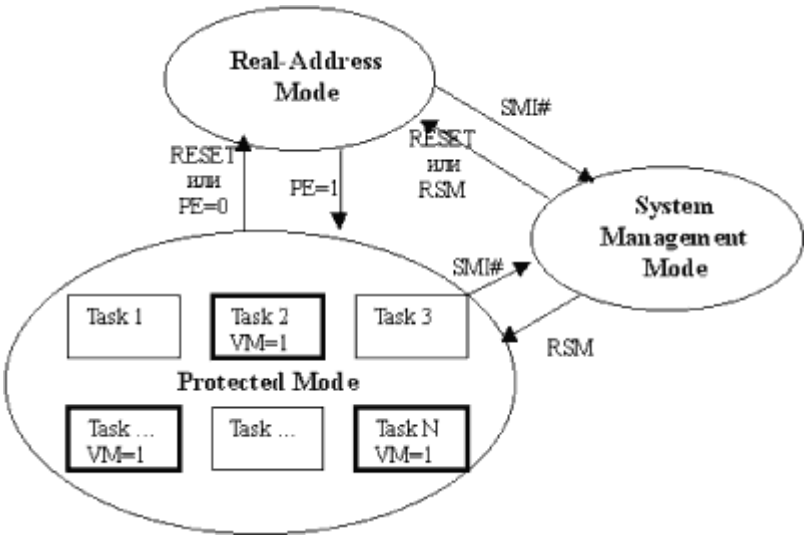

Рис. 7. Схема переключения между режимами.

После инициализации процессор находится в реальном режиме. Процессор может быть переведен в защищенный режим установкой бита 0 (Protect Enable) в регистре CR0:

```
MOV EAX,00000001h          или   MOV AX,0001h
MOV CR0,EAX                      LMSW AX
```

Второй вариант "достался в наследство" от 16-разрядной архитектуры 80286, для совместимости с которой ее регистр MSW (Machine Status Word) отображается на младшее слово регистра CR0.
Вернуться в режим реального адреса процессор может по сигналу RESET или (в отличие от 80286) сбросив бит PE:
MOV EAX,00000000h
MOV CR0,EAX
Для совместимости с 80286 инструкция LMSW бит PE не сбрасывает.
Режим системного управления изолирован от других режимов. Процессор переходит в этот режим только аппаратно: по низкому уровню на контакте SMI# или по команде с шины APIC (Pentium+). Никакой программный способ не предусмотрен для перехода в этот режим. Процессор возвращается из режима системного управления в тот режим, при работе в котором был получен сигнал SMI#. Возврат происходит по команде RSM. Эта команда работает только в режиме системного управления и в других режимах не распознается, генерирую исключение #6 (недействительный код операции).

Тема 8. Принципы формирования адресного пространства.

8.1. Среда выполнения задачи. Архитектурные решения микропроцессоров.

Любая задача, запущенная на микропроцессоре IA-32, оперирует определенным набором ресурсов для исполнения инструкций, сохранения данных и состояния задачи. Этот набор ресурсов называется средой выполнения. Функционирование некоторых элементов среды выполнения микропроцессора зависит от режима работы микропроцессора. К среде выполнения задачи относят:

- адресное пространство (размер адресного пространства и способы его адресации зависят от режима работы);
- регистры (8 регистров общего назначения, 6 сегментных регистров, регистр флагов и указатель команды);
- регистры сопроцессора (восемь 80-битовых регистров данных, регистр управления, регистр статуса, указатель команды, указатель операнда, слово тэгов);
- стек (для реализации вложенных подпрограмм и передачи параметров между ними).

Кроме перечисленных ресурсов, среда выполнения микропроцессора содержит так называемые системные ресурсы, предназначенные для поддержки операционных систем и системного программного обеспечения:

- пространство портов ввода-вывода (обеспечивает взаимодействие с периферийными устройствами);
- управляющие регистры (определяют режим работы процессора и состояние задачи);
- регистры управления памятью и дескрипторные таблицы (используются в защищенном режиме);
- регистры отладки;
- прочие регистры (MTRRs, MSRs и др.)

Анализируя адресные пространства программ и данных, определяют:

- МП с архитектурой фон Неймана (память программ и память данных находятся в едином пространстве и нет никаких признаков, указывающих на тип информации в ячейке памяти)

и

- МП с архитектурой Гарвардской лаборатории (память программ и память данных разделены, имеют свои адресные пространства и способы доступа к ним).

Мы рассмотрим более подробно основные типы архитектурных решений, выделяя связь со способами адресации памяти.
1. Регистровая архитектура определяется наличием достаточно большого регистрового файла внутри МП. Команды получают возможность обратиться к операндам, расположенным в одной из двух запоминающих сред: оперативной памяти или регистрах. Размер регистра обычно фиксирован и совпадает с размером слова, физически реализованного в оперативной памяти. К любому регистру можно обратиться непосредственно, поскольку регистры представлены в виде массива запоминающих элементов - регистрового файла. Типичным является выполнение арифметических операций только в регистре, при этом команда содержит два операнда (оба операнда в регистре или один операнд в регистре, а второй в оперативной памяти).
К данному типу архитектуры относится микропроцессор фирмы Zilog. Процессор Z80 - детище фирмы Zilog помимо расширенной системы команд, одного номинала питания и способности исполнять программы, написанные для i8080, имел архитектурные "изюминки".
В дополнение к основному набору РОН, в кристалле был реализован второй комплект аналогичных регистров. Это значительно упрощало работу при вызове подпрограмм или процедур обслуживания прерываний, поскольку программист мог использовать для них альтернативный набор регистров, избегая сохранения в стеке содержимого РОНов для основной программы с помощью операций PUSH. Кроме того, в систему команд был включен ряд специальных инструкций, ориентированных на обработку отдельных битов, а для поддержки регенерации динамической памяти в схему процессора введены соответствующие аппаратные средства. Z80 применялся в машинах Sinclair ZX, Sinclair Spectrum, Tandy TRS80.
Предельный вариант - архитектура с адресацией посредством аккумуляторов (меньший набор команд).
МП фирмы Motorola имел ряд существенных преимуществ. Прежде всего, кристалл MC6800 требовал для работы одного номинала питания, а система команд оказалась весьма прозрачной для программиста. Архитектура МП также имела ряд особенностей.

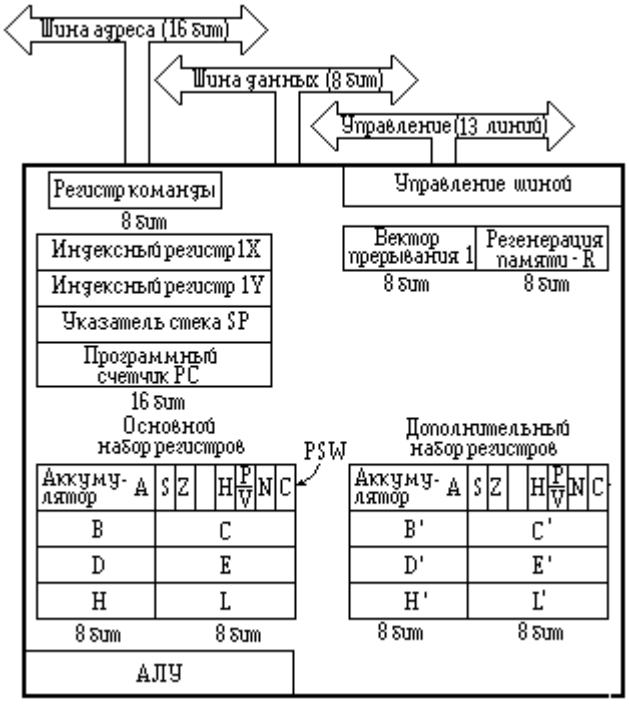

Рис. 8.1. Микропроцессор Z80 фирмы Zilog.

Микропроцессор МС 6800 содержал два аккумулятора, и результат операции АЛУ мог быть помещен в любой из них. Но самым ценным качеством структуры МС 6800 было автоматическое сохранение в стеке содержимого всех регистров процессора при обработке прерываний (Z80 требовалось для этого несколько команд PUSH). Процедура восстановления РОН из стека тоже выполнялась аппаратно.

2. Стековая архитектура дает возможность создать поле памяти с упорядоченной последовательностью записи и выборки информации.В общем случае команды неявно адресуются к элементу стека, расположенному на его вершине, или к двум верхним элементам стека.

3. Архитектура МП, ориентированная на оперативную память (типа "память-память"), обеспечивает высокую скорость работы и большую информационную емкость рабочих регистров и стека при их организации в оперативной памяти. Архитектура этого типа не предполагает явного определения аккумулятора, регистров общего назначения или стека; все операнды команд адресуются к области основной памяти.

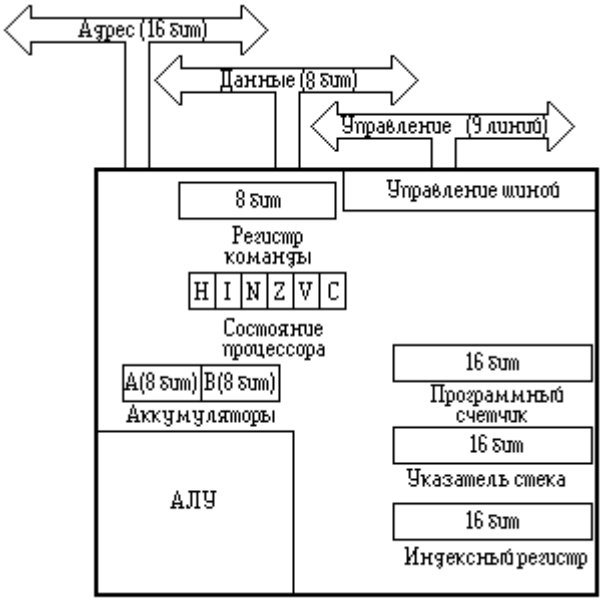

Рис8.2. Микропроцессор MC6800 фирмы Motorola.

Тема 9. Система адресации.

9.1. Режимы адресации.

Для взаимодействия с различными модулями в ЭВМ должны быть средства идентификации ячеек внешней памяти, ячеек внутренней памяти, регистров МП и регистров устройств ввода/вывода. Поэтому каждой из запоминающих ячеек присваивается адрес, т.е. однозначная комбинация бит. Количество бит определяет число идентифицируемых ячеек. Обычно ЭВМ имеет различные адресные пространства памяти и регистров МП, а

иногда - отдельные адресные пространства регистров устройств ввода/вывода и внутренней памяти. Кроме того, память хранит как данные, так и команды. Поэтому для ЭВМ разработано множество способов обращения к памяти, называемых режимами адресации.
Режим адресации памяти - это процедура или схема преобразования адресной информации об операнде в его исполнительный адрес.

9.2. Способы адресации.

Все способы адресации памяти можно разделить на: 1) прямой, когда исполнительный адрес берется непосредственно из команды или вычисляется с использованием значения, указанного в команде, и содержимого какого-либо регистра (прямая адресация, регистровая, базовая, индексная и т.д.); 2) косвенный, который предполагает, что в команде содержится значение косвенного адреса, т.е. адреса ячейки памяти, в которой находится окончательный исполнительный адрес (косвенная адресация).
В каждой микроЭВМ реализованы только некоторые режимы адресации, использование которых, как правило, определяется архитектурой МП.
Инструкция микропроцессора может содержать следующие поля:

префикс	КОП	Mod R/M	SIB	смещение	непосредственный операнд
0/1 байт	1/2 байта	0/1 байт	0/1 байт	0/1/2/4 байта	0/1/2/4 байта

Префикс - необязательная часть инструкции, позволяет изменить некоторые особенности ее выполнения. В команде может быть использовано сразу несколько префиксов разного типа. Типы префиксов:

- командные префиксы (префиксы повторения)
- префикс блокировки шины LOCK;
- префиксы размера (см. далее);
- префиксы замены сегмента.

Байт "Mod R/M" определяет режим адресации, а также иногда дополнительный код операции. Необходимость байта "Mod R/M" зависит от типа инструкции.
Байт SIB (Scale-Index-Base) определяет способ адресации при обращении к памяти в 32-битном режиме. Необходимость байта SIB зависит от режима адресации, задаваемого полем "Mod R/M".
Кроме того, инструкция может содержать непосредственный операнд и/или смещение операнда в сегменте данных.
На размер инструкции накладывается ограничение в 15 байт. Инструкция большего размера может получиться при некорректном использовании большого количества префиксов.

9.3. Режимы адресации операндов. Возможности микропроцессоров по адресации.

Если инструкция микропроцессора требует операнды, то они могут задаваться следующими способами:

- непосредственно в коде инструкции (только операнд-источник);
- в одном из регистров;
- через порт ввода-вывода;
- в памяти.

Для совместимости с 16-битными процессорами архитектура IA-32 использует одинаковые коды для инструкций, оперирующих как с 16-битными, так и 32-битными операндами. Новая архитектура предусматривает также новые возможности при указании адреса для операнда в памяти. Как процессор будет считать операнд или его адрес, зависит от эффективного размера операнда и эффективного размера адреса для данной команды. Эти значения определяются на основе режима работы, бита D дескриптора используемого сегмента и наличия в инструкции определенных префиксов.
Непосредственный режим адресации подразумевает включение операнда-источника в код инструкции. Операнд может быть 8-битным или 16-битным, если значение эффективного размера операнда - 16. Операнд может быть 8-битным или 32-битным, если значение эффективного размера операнда - 32. Обычно непосредственные операнды используются в арифметических инструкциях.
Регистровый режим адресации определят операнд-источник или операнд-приемник в одном из следующих регистров:

- регистры общего назначения (EAX/AX, EBX/BX, ECX/CX, EDX/DX, ESI/SI, ESP/SP, EBP/BP - 16-битный регистр, если эффективный размер операнда - 16, или 32-битный регистр в противном случае) или их младшие байты (AH, AL, BH, BL, CH, CL, DH, DL);
- сегментные регистры (CS, DS, SS, ES, FS, GS);
- регистр флагов (EFLAGS для 32-битных операндов или FLAGS для 16-битных);
- управляющие регистры (CR0/MSW, CR2, CR3, CR4) и регистры системных таблиц (GDTR, LDTR, IDTR, TR);
- регистры отладки (DR0-DR7);
- машинно-зависимые регистры (MSRs);
- регистры сопроцессора, MMX и XMM.

В некоторых случаях (например, в инструкциях DIV и MUL) могут использоваться пары 32-битных (или 16-битных) регистров (например, EDX:EAX), образуя 64-битный (32-битный) операнд.

Адресация через порт ввода-вывода подразумевает получение операнда или сохранение операнда через пространство портов ввода-вывода. Адрес порта ввода-вывода либо непосредственно включается в код инструкции, либо берется из регистра DX.
Очень распространенный способ адресации операнда - адресация через память. Таким образом может быть указан операнд-источник или операнд-приемник. Следует отметить, что процессор не позволяет одновременно задавать оба операнда через память (за исключением некоторых цепочечных команд).
Для получения операнда из памяти процессору необходимо знать селектор сегмента и смещение в сегменте. В некоторых командах селектор может быть указан непосредственно в коде инструкции. В других случаях процессор может явно или неявно использовать значение одного из сегментных регистров. Под неявным использованием сегментных регистров подразумевается то, что в зависимости от предназначения операнда процессор использует определенный сегментный регистр для обращения к памяти: CS - для выборки инструкций; SS - для работы со стеком или обращения к памяти через регистры ESP или EBP; ES - для получения адреса операнда-приемника в цепочечных командах; DS - при всех остальных обращениях к памяти. Явное использование сегментных регистров возможно, если в код инструкции включается префикс смены сегмента. Указание префикса смены сегмента допустимо не для всех команд: нельзя менять сегмент для команд работы со стеком (всегда используется SS), для цепочечных команд можно менять сегмент только операнда-источника (операнд-приемник всегда адресуется через ES).

Одной из важнейших архитектурных характеристик МП является перечень возможных способов обращения к памяти или видов адресации. Возможности МП по адресации существенны с двух точек зрения.
Во-первых, большой объем памяти требует большой длины адреса, так как n-разрядный адрес позволяет обращаться к памяти емкостью 2n слов. Типовые 8-разрядные слова МП дают возможность непосредственно обращаться только к 256 ячейкам памяти, что явно недостаточно. Если учесть, что обращение к памяти является наиболее часто встречающейся операцией, то очевидно, что эффективность использования МП во многом определяется способами адресации к памяти большого объема при малой разрядности МП.
Во-вторых, для удобства программирования желательно иметь простую систему формирования адресов данных при работе с массивами, таблицами и указателями. Рассмотрим способы решения этих проблем.
Если адресное поле в команде является ограниченным и недостаточным для непосредственного обращения к любой ячейке памяти, то память в таких случаях разбивают на страницы, где страницей считается 2n ячеек памяти.
Для согласования адресного поля команды малой разрядности с памятью большого объема (для решения “страничной” проблемы) в МП применяются различные виды адресации:
Прямая адресация к текущей странице. При такой адресации программный счетчик разбивается на два поля; старшие разряды указывают номер страницы, а младшие - адрес ячейки на странице. В адресном поле команды размещается адрес ячейки на странице, а адрес страницы должен быть установлен каким-то другим способом, например с помощью специальной команды.
Прямая адресация с использованием страничного регистра. В МП должен быть предусмотрен программно доступный страничный регистр, загружаемый специальной командой. Этот регистр добавляет к адресному полю команды несколько разрядов, необходимых для адресации ко всей памяти.
Прямая адресация с использованием двойных слов. Для увеличения длины адресного поля команды под адрес отводится дополнительное слово (а если нужно, то и два).
Адресация относительно программного счетчика. Адресное поле команды рассматривается как целое со знаком, которое складывается с содержимым программного счетчика для формирования исполнительного адреса. Такой способ относительной адресации создает плавающую страницу и упрощает перемещение программ в памяти.
Адресация относительно индексного регистра. Исполнительный адрес образуется суммированием содержимого индексного регистра и адресного поля команды, рассматриваемого как целое со знаком. Индексный регистр загружается специальными командами.
Косвенная адресация. При косвенной адресации в адресном поле команды указывается адрес на текущей странице, по которому хранится исполнительный адрес. В поле команды при этом требуется дополнительный разряд - признак косвенной адресации. Исполнительный адрес может храниться не в ячейке памяти, а в регистре общего назначения. В этом случае косвенная адресация называется регистровой.

Тема 10. Память как функциональный узел.
10.1 Внутренняя память компьютера.

Для хранения информации в микропроцессорных системах используются запоминающие устройства на основе полупроводниковых материалов, а также магнитные и оптические внешние носители. Внутренняя память компьютера представлена в виде отдельных интегральных микросхем (ИМС) собственно памяти и элементов, включенных в состав других ИМС, не выполняющих непосредственно функцию хранения программ и данных - это и внутренняя память центрального процессора, и видеопамять, и контроллеры различных устройств.
Для создания элементов запоминающих устройств, в основном, применяют СБИС со структурой МДП (металл-диэлектрик-полупроводник) на основе кремния (в связи с тем, что в качестве диэлектрика чаще всего используют его оксид $Si0_2$, то их обычно называют МОП (металл-оксид-полупроводник) структурами).

Для функционирования компьютерной системы необходимо наличие как оперативного запоминающего устройства (ОЗУ), так и постоянного запоминающего устройства (ПЗУ), обеспечивающего сохранение информации при выключении питания. ОЗУ может быть статическим и динамическим, а ПЗУ однократно или многократно программируемым.
Степень интеграции, быстродействие, электрические параметры ЗУ при записи и хранении информации, помехоустойчивость, долговременная стабильность, стабильность к внешним неблагоприятным факторам при функционировании и т.д. зависят от физических принципов работы приборов, применяемых материалов при производстве ИМС и параметров технологических процессов при их изготовлении.

10.2. Технология производства интегральных схем.

На развитие микропроцессорной техники решающее значение оказывает технология производства интегральных схем.
Полупроводниковые интегральные микросхемы подразделяются на биполярные ИМС и МОП схемы, причем первые - более быстродействующие, а вторые имеют большую степень интеграции, меньшую потребляемую мощность и меньшую стоимость. Цифровые микросхемы могут по идеологии, конструкторскому решению, технологии относится к разным семействам, но выполнять одинаковую функцию, т.е. быть инвертором, триггером или процессором. Наиболее популярными семействами можно назвать у биполярных ИМС: ТТЛ (транзисторно-транзисторная логика), ТТЛШ (с диодами Шоттки), ЭСЛ (эмиттерно-связанная логика); у МДП: n-МОП и КМОП.
Базовым материалом для изготовления ИМС является кремний. Несмотря на то, что он не обладает высокой подвижностью носителей заряда (mn=1500 $см^2$/Вс), а значит, приборы на его основе теоретически будут уступать по быстродействию приборам на основе арсенида галлия GaAs, однако система Si-SiO_2 существенно более технологична. С другой стороны, приборы на кремниевой основе (кремний-оксид кремния) обладают совершенной границей раздела Si-SiO_2, химической стойкостью, электрической прочностью и другими уникальными свойствами.
Технологический цикл производства ИМС включает: - эпитаксиальное наращивание слоя на подготовленную подложку;
- наращивание слоя SiO_2 на эпитаксиальный слой; - нанесение фоторезиста, маскирование и вытравливание окон в слое; - легирование примесью путем диффузии или имплантацией; - аналогично повторение операций для подготовки других легированных областей; - повторение операций для создания окон под контактные площадки;
- металлизацию всей поверхности алюминием или поликремнием; - повторение операций для создания межсоединений;
- удаление излишков алюминия или поликремния; - контроль функционирования; - помещение в корпус; - выходной контроль.
Наиболее критичным для увеличения степени интеграции является процесс литографии, т.е. процесс переноса геометрического рисунка шаблона на поверхность кремниевой пластины. С помощью этого рисунка формируют такие элементы схемы, как электроды затвора, контактные окна, металлические межкомпонентные соединения и т.п. На первой стадии изготовления ИМС после завершения испытаний схемы или моделирования с помощью ЭВМ формируют геометрический рисунок топологии схемы. С помощью электронно-лучевого устройства или засветки другим способом топологический рисунок схемы последовательно, уровень за уровнем можно переносить непосредственно на поверхность кремниевой пластины, но чаще на фоточувствительные стеклянные пластины, называемые фотошаблонами. Между переносом топологического рисунка с двух шаблонов могут быть проведены операции ионной имплантации, загонки, окисления и металлизации. После экспонирования пластины помещают в раствор, который проявляет изображение в фоточувствительном материале - фоторезисте.
Увеличивая частоту колебаний световой волны, можно уменьшить ширину линии рисунка, т. е. сократить размеры интегральных схем. Но возможности этой технологии ограничены, поскольку рентгеновские лучи трудно сфокусировать. Один из вариантов - использовать сам свет в качестве шаблона (так называемое позиционирование атомов фокусированным лазерным лучом). Этим способом, осветив двумя взаимно перпендикулярными лазерными пучками, можно изготовить решетку на кремниевой пластине из хромированных точек размером 80 нм. Сканируя лазером поверхность для создания произвольного рисунка интегральных наносхем, теоретически можно создавать схемы с шириной линии рисунка в 10 раз меньшей, чем сегодняшние. Второе ограничение при литографии накладывает органическая природа фоторезиста. Путь ее решения - применение неорганических материалов, например, оксидов ванадия.
Физические процессы, протекающие в изделиях микроэлектроники (и в микросхемах памяти тоже), технология изготовления и конструктивные особенности ИМС высокой степени интеграции могут влиять на архитектуру и методы проектирования ЭВМ и систем. Естественно, уменьшение геометрических размеров транзисторов приводит к увеличению электрических полей, особенно в районе стока. Это может привести к развитию лавинного пробоя и, как следствие, к изменению выходной ВАХ МОП транзистора: - включению паразитного биполярного транзистора (исток-подложка-сток); - неравномерному заряжению диэлектрика у стока; - деградации приповерхностной области полупроводника; - пробою диэлектрика.
Поэтому необходимо уменьшение напряжения питания СБИС до 3.6, 3.3, 3 В и т.п., при этом известно, что блок питания компьютера обеспечивает обычно напряжения +5В, +12В, -12В.

Однако инжекция и заряжение диэлектрика не всегда процесс отрицательный или паразитный. Уменьшение напряжения записи информационного заряда в репрограммируемых ЗУ ниже 12 В позволяет их программировать внутри микропроцессорной системы, а не специальным устройством (программатором). Тогда для разработчика открываются большие возможности для программирования не только адреса микросхем контроллера или адаптера в пространстве устройств ввода/вывода или номера прерывания, но и творить необходимое устройство самому (если иметь такую ИМС). Однако отметим, что кроме "хозяина" это может сделать и компьютерный вирус, который будет, естественно, разрушать, а не созидать что-либо.

10.3. Основные характеристики полупроводниковой памяти.

Полупроводниковая память имеет большое число характеристик и параметров, которые необходимо учитывать при проектировании систем:

1. Емкость памяти определяется числом бит хранимой информации. Емкость кристалла обычно выражается также в битах и составляет 1024 бита, 4 Кбит, 16 Кбит, 64 Кбит и т.п. Важной характеристикой кристалла является информационная организация кристалла памяти MxN, где M - число слов, N - разрядность слова. Например, кристалл емкостью 16 Кбит может иметь различную организацию: 16 Кх1, 4 Кх2 Кх8. При одинаковом времени обращения память с большей шириной выборки обладает большей информационной емкостью.
2. Временные характеристики памяти.

Время доступа - временной интервал, определяемый от момента, когда центральный процессор выставил на шину адреса адрес требуемой ячейки памяти и послал по шине управления приказ на чтение или запись данных, до момента осуществления связи адресуемой ячейки с шиной данных.

Время восстановления - это время, необходимое для приведения памяти в исходное состояние после того, как ЦП снял с ША - адрес, с ШУ - сигнал "чтение" или "запись" и с ШД - данные.

3. Удельная стоимость запоминающего устройства определяется отношением его стоимости к информационной емкости, т.е. определяется стоимостью бита хранимой информации.
4. Потребляемая энергия (или рассеиваемая мощность) приводится для двух режимов работы кристалла: режима пассивного хранения информации и активного режима, когда операции записи и считывания выполняются с номинальным быстродействием. Кристаллы динамической МОП-памяти в резервном режиме потребляют примерно в десять раз меньше энергии, чем в активном режиме. Наибольшее потребление энергии, не зависящее от режима работы, характерно для кристаллов биполярной памяти.
5. Плотность упаковки определяется площадью запоминающего элемента и зависит от числа транзисторов в схеме элемента и используемой технологии. Наибольшая плотность упаковки достигнута в кристаллах динамической МОП-памяти.
6. Допустимая температура окружающей среды обычно указывается отдельно для активной работы, для пассивного хранения информации и для нерабочего состояния с отключенным питанием. Указывается тип корпуса, если он стандартный, или чертеж корпуса с указанием всех размеров, маркировкой и нумерацией контактов, если корпус новый. Приводятся также условия эксплуатации: рабочее положение, механические воздействия, допустимая влажность и другие.

Тема 11. Принципы организации памяти.

11.1. Организация физической памяти. ОЗУ, ПЗУ и FLASH – память.

Физическая память, к которой микропроцессов имеет доступ по шине адреса, называется ***оперативной памятью*** (или оперативным запоминающим устройством - ОЗУ).

ОП организована как последовательность байтов. Каждому байту соответствует уникальный адрес (его номер), который называется ***физическим адресом***.

Диапазон значений адресов зависит от разрядности шины адреса микропроцессора.

Для i486 и Pentium он находится в диапазоне от 0 до 2**32-1 --- 4 Гбайт (32-разрядная шина адреса).

Механизм управления памятью полностью аппаратный, т.е. программа сама не может сформировать физический адрес памяти на адресной шине.

Оперативное запоминающее устройство предназначено для хранения информации (программ и данных), непосредственно участвующей в вычислительном процессе на текущем этапе функционирования ПК.

ОЗУ - энергозависимая память*:* при отключении напряжения питания информация, хранящаяся в ней, теряется. Основу ОЗУ составляют большие интегральные схемы, содержащие матрицы полупроводниковых запоминающих элементов (триггеров). Запоминающие элементы расположены на пересечении вертикальных и горизонтальных шин матрицы; запись и считывание информации осуществляются подачей электрических импульсов по тем шинам матрицы, которые соединены с элементами, принадлежащими выбранной ячейке памяти.

Конструктивно элементы оперативной памяти выполняются в виде отдельных микросхем типа DIP (Dual In-line Package - двухрядное расположение выводов) или в виде модулей памяти типа SIP (Single In-line Package - однорядное расположение выводов), или, что чаще, SIMM (Single In line Memory Module - модуль памяти с однорядным расположением выводов). Модули SIMM имеют емкость 256Кбайт, 1, 4, 8, 16 или 32 Мбайта, с контролем и без контроля четности хранимых битов; могут иметь 30- ("короткие") и 72-("длинные") контактные

разъемы, соответствующие разъемам на материнской плате компьютера. На материнскую плату можно установить несколько (четыре и более) модулей SIMM.
Постоянное запоминающее устройство также строится на основе установленных на материнской плате модулей (кассет) и используется для хранения неизменяемой информации: загрузочных программ операционной системы, программ тестирования устройств компьютера и некоторых драйверов базовой системы ввода-вывода (BIOS - Base Input-Output System) и др. Из ПЗУ можно только считывать информацию, запись информации в ПЗУ выполняется вне ЭВМ в лабораторных условиях. Модули и кассеты ПЗУ имеют емкость, как правило, не превышающую нескольких сот килобайт. ПЗУ - энергонезависимое запоминающее устройство.
Примечание**.** В последние годы в некоторых ПК стали использоваться полупостоянные. перепрограммируемые запоминающие устройства - FLASH-память. Модули или карты FLASH-памяти могут устанавливаться прямо в разъемы материнской платы и имеют следующие параметры: емкость от 32 Кбайт до 4 Мбайт, время доступа по считыванию 0.06 мкс, время записи одного байта примерно 10 мкс: FLASH-память - энергонезависимое запоминающее устройство.
Для перезаписи информации необходимо подать на специальный вход FLASH-памяти напряжение программирования (12В), что исключает возможность случайного стирания информации. Перепрограммирование FLASH-памяти может выполняться непосредственно с дискеты или с клавиатуры ПК при наличии; специального контроллера либо с внешнего программатора, подключаемого к ПК.
FLASH-память может быть полезной как для создания весьма быстродействующих компактных, альтернативных НЖМД запоминающих устройств - "твердотельных дисков", так и для замены ПЗУ, хранящего программы BIOS, позволяя "прямо с дискеты" обновлять и заменять эти программы на более новые версии при модернизации ПК.
Структурно основная память состоит из миллионов отдельных ячеек памяти емкостью 1 байт каждая. Общая емкость основной памяти современных ПК обычно лежит в пределах от 1 до 32 Мбайт. Емкость ОЗУ на один-два порядка превышает емкость ПЗУ: ПЗУ занимает 128 (реже 256) Кбайт, остальной объем - это ОЗУ.

11.2. Логическая структура основной памяти.

Каждая ячейка памяти имеет свой уникальный (отличный от всех других) адрес. Основная память имеет для ОЗУ и ПЗУ единое адресное пространство.
Адресное пространство определяет максимально возможное количество непосредственно адресуемых ячеек основной памяти.
Адресное пространство зависит от разрядности адресных шин, ибо максимальное количество разных адресов определяется разнообразием двоичных чисел, которые можно отобразить в n разрядах, т.е. адресное пространство равно 2^n, где *n* - разрядность адреса.
Для ПК характерно стандартное распределение непосредственно адресуемой памяти между ОЗУ, ПЗУ и функционально ориентированной информацией (рис. 11.1).
Основная память в соответствии с методами доступа и адресации делится на отдельные, иногда частично или полностью перекрывающие друг друга области, имеющие общепринятые названия. В частности, укрупненно логическая структура основной памяти ПК обшей емкостью, например, 16 Мбайт представлена на рис.11.2.

Стандартная память 640 Кбайт		Верхняя память 384 Кбайт	
64 Кбайта Область служебных программ и данных ОС	576 Кбайт Область программ и данных пользователя	256 Кбайт Область видеопамяти дисплея и служебных программ	128 Кбайт Область программ начальной загрузки ОС и др.
ОЗУ			ПЗУ

Рис. 11.1. Распределение 1-Мбайтной области ОП

Непосредственно адресуемая память		Расширенная память	
Стандартная (обычная) память (CMA)	Верхняя память (блоки UMA)	Высокая память (HMA)	
640 Кбайт	384 Кбайта	64 Кбайта	
640 Кбайт	1024 Кбайта	1088 Кбайта	16 Мбайт

Рис. 11.2. Логическая структура основной памяти

Прежде всего основная память компьютера делится на две логические области: непосредственно адресуемую память, занимающую первые 1024 Кбайта ячеек с адресами от 0 до 1024 Кбайт-1, расширенную память, доступ к ячейкам которой возможен при использовании специальных программ-драйверов.
Стандартной памятью (CMA - Conventional Memory Area) называется непосредственно адресуемая память в диапазоне от 0 до 640 Кбайт.
Непосредственно адресуемая память в диапазоне адресов от 640 до 1024 Кбайт называется верхней памятью (UMA - Upper Memory Area). Верхняя память зарезервирована для памяти дисплея (видеопамяти) и постоянного запоминающего устройства. Однако обычно в ней остаются свободные участки - "окна", которые могут быть использованы при помощи диспетчера памяти в качестве оперативной памяти общего назначения.
Расширенная память - это память с адресами 1024 Кбайта и выше.
Непосредственный доступ к этой памяти возможен только в защищенном режиме работы микропроцессора.
В реальном режиме имеются два способа доступа к этой памяти, но только при использовании драйверов:

по спецификации XMS (эту память называют тогда XMA - eXtended Memory Area);
по спецификации EMS (память называют EM -Expanded Memory*).*
Спецификация EMS (Expanded Memory Specification) является более ранней. Согласно этой спецификации доступ реализуется путем отображения по мере необходимости отдельных полей Expanded Memory в определенную область верхней памяти. При этом хранится не обрабатываемая информация, а лишь адреса, обеспечивающие доступ к этой информации. Память, организуемая по спецификации EMS, носит название отображаемой, поэтому и сочетание слов Expanded Memory (EM) часто переводят как отображаемая память. Для организации отображаемой памяти необходимо воспользоваться драйвером EMM386.EXE (Expanded Memory Manager) или пакетом управления памятью QEMM.
Расширенная память может быть использована главным образом для хранения дат и некоторых программ ОС. Часто расширенную память используют для организации виртуальных (электронных) дисков.
Исключение составляет небольшая 64-Кбайтная область памяти с адресами от 1024 до 1088 Кбайт (так называемая высокая память, иногда ее называют старшая: HMA - High Memory Area), которая может адресоваться и непосредственно при использовании драйвера HIMEM.SYS (High Memory Manager) в соответствии со спецификацией XMS. HMA обычно используется для хранения программ и данных операционной системы.

Тема 12. Виртуальная память.

12.1. Модели памяти.

Микропроцессор аппаратно поддерживает несколько моделей использования оперативной памяти:

- сегментированная модель
- страничная модель
- сегментно-страничная модель

12.2. Понятие о сегментированной модели памяти.

Память для программы делится на непрерывные области памяти, называемые ***сегментами***.
Сегменты - это логические элементы программы.
Сама программа может обращаться только к данным, которые находятся в этих сегментах.
Сегмент представляет собой независимый, поддерживаемый на аппаратном уровне блок памяти.
Сегментация - механизм адресации, обеспечивающий существование нескольких независимых адресных пространств как в пределах одной задачи, так и в системе в целом для защиты задач от взаимного влияния.
Замечание. Программист может либо самостоятельно разбивать программу на фрагменты (сегменты), либо автоматизировать этот процесс и возложить его на систему программирования.
Для микропроцессоров Intel принят особый подход к управлению памятью. Каждая программа в общем случае может состоять из любого количества сегментов, но непосредственный доступ она имеет только к 3 основным сегментам: кода, данных и стека и к дополнительным сегментам данных (всего 3).
Операционная система (а не сама программа) размещает сегменты программы в ОП по определенным физическим адресам, а значения этих адресов записывает в определенные места, в зависимости от режима работы микропроцессора:

- в реальном режиме адреса помещаются непосредственно в сегментные регистры (cs, ds, ss, es, gs, fs);
- в защищенном режиме - в специальную системную дескрипторную таблицу (Элементом дескрипторной таблицы является дескриптор сегмента. Каждый сегмент имеет дескриптор сегмента -8 байт. Существует три дескрипторные таблицы. Адрес каждой таблицы записывается в специальный системный регистр).

Для доступа к данным внутри сегмента обращение производится относительно начала сегмента линейно, т.е. начиная с 0 и заканчивая адресом, равным размеру сегмента. Этот адрес называется ***смещением (offset*)**.
Таким образом, для обращения к конкретному физическому адресу ОП необходимо определить адрес начала сегмента и смещение внутри сегмента.
Физический адрес принято записывать парой этих значений, разделенных двоеточием

segment : offset

Например, 0040:001Ch; 0000:041Ch; 0020:021Ch; 0041:000Ch.
Каждый сегмент описывается дескриптором сегмента.
ОС строит для каждого исполняемого процесса соответствующую таблицу дескрипторов сегментов и при размещении каждого из сегментов в ОП или внешней памяти в дескрипторе отмечает его текущее местоположение (бит присутствия).
Дескриптор содержит поле адреса, с которого сегмент начинается и поле длины сегмента. Благодаря этому можно осуществлять контроль

1) размещения сегментов без наложения друг на друга
2) обращается ли код исполняющейся задачи за пределы текущего сегмента.

В дескрипторе содержатся также данные о правах доступа к сегменту (запрет на модификацию, можно ли его предоставлять другой задаче) $\Rightarrow$ защита.
Достоинства:

1) общий объем виртуальной памяти превосходит объем физической памяти

2) возможность размещать в памяти как можно больше задач (до определенного предела) ⇒ увеличивает загрузку системы и более эффективно используются ресурсы системы

Недостатки:

1) увеличивается время на доступ к искомой ячейке памяти, т.к. должны вначале прочитать дескриптор сегмента, а потом уже, используя его данные, можно вычислить физический адрес (для уменьшения этих потерь используется кэширование - дескрипторы, с которыми работа идет в данный момент размещаются в сверхоперативной памяти - в специальных регистрах процессора);
2) фрагментация;
3) потери памяти на размещение дескрипторных таблиц
4) потери процессорного времени на обработку дескрипторных таблиц.

Сегментированная модель памяти поддерживается и в реальном, и в защищенном режимах работы микропроцессора.

12.3. Понятие о страничной модели памяти.

Это надстройка над сегментной моделью.
ОП делится на блоки фиксированного размера 4 Кб (должно быть число, кратное степени двойки, чтобы операции сложения можно было бы заменить на операции конкатенации).
Каждый такой блок называется ***страницей***.
Их число 1.048.576 ⇒ 4 Гб адресуемой памяти.

Основное применение этой модели связано с организацией **виртуальной памяти**.

Для того, чтобы использовать для работы программ пространство памяти большее, чем объем физической памяти используется механизм виртуальной памяти.
Суть его заключается в том, что у микропроцессора существует возможность по обмену страницами памяти с жестким диском. В случае, если программа требует памяти больше, чем объем физической памяти, редко используемые страницы памяти записываются на жесткий диск в специальный файл виртуальной памяти (файл обмена, или страничный файл, или файл подкачки, чаще swap-файлом, подчеркивая, что страницы этого файла замещают друг друга в ОП).
Замечание. В некоторых ОС выгруженные страницы располагаются не в файле, а в специальном разделе диска, например, в ОС UNIX есть специальный раздел, но могут использоваться и файлы, если не достаточно объема раздела.
В настоящее время файл подкачки может динамически изменять свой размер в зависимости от потребностей системы.
Для i486 и Pentium размер возможной виртуальной памяти может достигать 4 Тб (терабайт).
Обратим внимание на то, что программа также разбивается на фрагменты - страницы. Все фрагменты программы одинаковой длины, кроме последней страницы.
Говорят, что память разбивается на физические страницы, а программа - на виртуальные страницы.
Трансляция (отображение) виртуального адресного пространства задачи на физическую память осуществляется с помощью таблицы страниц.
Для каждой текущей задачи создается **таблица страниц**.
Диспетчер памяти для каждой страницы формирует соответствующий дескриптор. Дескриптор содержит так называемый бит присутствия. Если он = 1, то данная страница сейчас размещена в ОП. Если он = 0, то страница расположена во внешней памяти.
Защита страничной памяти основана на контроле уровня доступа к каждой странице.
Каждая страница снабжается кодом уровня доступа (только чтение; чтение и запись; только выполнение). При работе со страницей сравнивается значение кода разрешенного уровня доступа с фактически требуемым. При несовпадении работа программы прерывается.
Страничная модель памяти поддерживается только в защищенном режиме работы микропроцессора.
Основное достоинство страничного способа распределения памяти -
минимально возможная фрагментация (эффективное распределение памяти).
Недостатки:

1) потери памяти на размещение таблиц страниц
2) потери процессорного времени на обработку таблиц страниц (диспетчер памяти).
3) Программы разбиваются на страницы случайно, без учета логических взаимосвязей, имеющихся в коде ⇒ межстраничные переходы осуществляются чаще, чем межсегментные + трудности в организации разделения программных модулей между выполняющими процессами

Чтобы избежать недостатка №3, был предложен сегментно-страничный способ распределения памяти.

12.4. Сегментно-страничная модель памяти.

Программа разбивается на сегменты, которые разбиваются на страницы.
Адрес, по-прежнему, состоит из двух частей - сегмент + смещение.
Но смещение относительно начала сегмента может состоять из двух полей: виртуальной страницы и индекса.

Для доступа к памяти необходимо:
1) вычислить адрес дескриптора сегмента и причитать его;
2) вычислить адрес элемента таблицы страниц этого сегмента и извлечь из памяти необходимый элемент;
3) к номеру (адресу) физической страницы приписать номер (адрес) ячейки в странице.

$\Rightarrow$ Задержка в доступе к памяти (в три раза больше, чем при прямой адресации).
Чтобы избежать этого вводится кэширование (кэш строится по ассоциативному принципу).

12.5. Плоская модель памяти.

Если считать, что задача состоит из одного сегмента, который, в свою очередь, разбит на страницы, то фактически мы получаем только один страничный механизм работы с виртуальной памятью.
Это подход называется плоской памятью.
Достоинства:

- При использовании плоской модели памяти упрощается создание и ОС, и систем программирования.
- уменьшаются расходы памяти на поддержку системных информационных структур

В абсолютном большинстве современных 32-разрядных ОС (для микропроцессоров Intel) используется плоская модель памяти.

Тема 13. Прерывания.

13.1. Понятие прерывания.

Прерывание означает временное прекращение основного процесса вычислений для выполнения некоторых запланированных или незапланированных действий, вызываемых работой аппаратуры или программы.
Т.е. это процесс, временно переключающий микропроцессор на выполнение другой программы с последующим возвратом к прерванной программе.
Нажимая клавишу на клавиатуре, мы инициируем немедленный вызов программы, которая распознает клавишу, заносит ее код в буфер клавиатуры, из которого он считывается другой программой. Т.е. на некоторое время микропроцессор прерывает выполнение текущей программы и переключается на программу обработки прерывания, так наз. обработчик прерывания. После того, как обработчик прерывания завершит свою работу, прерванная программа продолжит выполнение с точки, где было приостановлено ее выполнение.
Адрес программы-обработчика прерывания вычисляется по таблице векторов прерываний.
Механизм прерываний поддерживается на аппаратном уровне.

13.2. Классификация прерываний.

В зависимости от источника, прерывания делятся на:

· аппаратные - возникают как реакция микропроцессора на физический сигнал от некоторого устройства (клавиатура, системные часы, клавиатура, жесткий диск и т.д.), по времени возникновения эти прерывания асинхронны, т.е. происходят в случайные моменты времени;

· программные - вызываются искусственно с помощью соответствующей команды из программы (int), предназначены для выполнения некоторых действий операционной системы, являются синхронными;

· исключения - являются реакцией микропроцессора на нестандартную ситуацию, возникшую внутри микропроцессора во время выполнения некоторой команды программы (деление на ноль, прерывание по флагу TF (трассировка)).

Общая классификация прерываний

· **внешние** - вызываются внешними по отношению к микропроцессору событиями
(по существу - это группа аппаратных прерываний) Вложенных прерываний нет!

· **внутренние** - возникают внутри микропроцессора во время вычислительного процесса (по существу - это исключительные ситуации и программные прерывания).

13.3. Внешние прерывания.

Внешние прерывания возникают по сигналу какого-нибудь внешнего устройства.
Внешние прерывания подразделяются на **немаскируемые и маскируемые.**
В связи с тем, что существуют два специальных внешних сигнала среди входных сигналов процессора, при помощи которых можно прервать выполнение текущей программы и тем самым переключить работу центрального процессора. Это сигналы NMI (no mask interrupt, немаскируемое прерывание) и INTR (interrupt request, запрос на прерывание).

Маскируемые прерывания генерируются контроллером прерываний по заявке определенных периферийных устройств. Контроллер прерываний (выполнен в виде специальной микросхемы i8259A) поддерживает восемь уровней (линий) приоритета; к каждому уровню “привязано” одно периферийное устройство. Именно маскируемые прерывания часто называют аппаратными прерываниями.

Немаскируемые прерывания (говорят, что оно одно, т.к. подается на вывод микропроцессора NMI) инициируют источники, требующие безотлагательного вмешательства со стороны микропроцессора.

В реальном и защищенном режиме работы микропроцессора обработка прерываний осуществляется принципиально разными методами.

13.4. Система прерываний.

Система прерываний - это совокупность программных и аппаратных средств, реализующих механизм прерываний.

К *аппаратным средствам* системы прерываний относятся:
· выводы микропроцессора - на них формируются сигналы, извещающие микропроцессор либо о том, что некоторое внешнее устройство «просит уделить ему внимание» (INTR), либо о том, что требуется безотлагательная обработка некоторого события или катастрофическая ошибка (NMI)
· INTR - вывод для входного сигнала запроса на прерывание,
· NMI - вывод для входного сигнала немаскируемого прерывания
· INTA - вывод для выходного сигнала подтверждения получения сигнала прерывания микропроцессором (этот сигнал поступает на одноименный вход микросхемы конроллера 8259A;
· программируемый контроллер прерываний 8259A (предназначен для фиксирования сигналов прерываний от восьми различных внешних устройств; он выполнен в виде микросхемы; обычно используют две последовательно соединенные микросхемы, поэтому кол-во возможных источников внешних прерываний до 15 плюс одно немаскируемое прер.; именно он формирует номер вектора прерывания и выдает его шину данных);
· внешние устройства (таймер, клавиатура, магнитные диски и т.п.)
К *программным средствам* системы прерываний Реального режима относятся:
· **таблица векторов прерываний**.
Занимает первый килобайт ОП (адреса 00000h-003FFh).
Она содержит адреса (векторы - «векторы», т.к. два значения для указания адреса) обработчиков прерываний и состоит из 256 (0..255) элементов по 4 байта каждый:
- 2 байта - новое значение для регистра IP
- 2 байта - новое значение для регистра CS.
Расположение таблицы векторов прерываний в процессорах i80286 и старше определяется значением регистра IDTR.
Таблица векторов прерываний инициализируется при запуске системы, но в принципе может быть изменена и перемещена.
Каждый вектор имеет свой номер и называется номером прерывания.
· **два флага в регистре флагов flags/eflags**:
· IF (Interrupt Flag) - флаг прерывания. Предназначен для маскирования (запрещения) аппаратных прерываний. Если IF=1, микропроцессор обрабатывает внешние прерывания, если = 0, то игнорирует;
· TF(Trace Flag) - флаг трассировки. Если он=1, то микропроцессор переходит в режим покомандной работы. В этом режиме в микропроцессоре генерируется внутреннее прерывание с номером 1;
· **машинные команды микропроцессора: int, into** (прерывание по переполнению)**, iret, cli, sti**

Тема 14. Поддержка многозадачности.

14.1. Многозадачность – определение, история развития.

Многозадачность (multitasking) - это способность операционной системы выполнять несколько программ одновременно. В основе этого принципа лежит использование операционной системой аппаратного таймера для выделения отрезков времени (time slices) для каждого из одновременно выполняемых процессов. Если эти отрезки времени достаточно малы, и машина не перегружена слишком большим числом программ, то пользователю кажется, что все эти программы выполняются параллельно.
В обязанности процессора входит обеспечение одновременного выполнения нескольких задач. Для реализации многозадачности применяются системные регистры процессора.

Для каждой из задач процессор выделяет очень короткий промежуток (квант) времени. Процессор переключается с выполнения одной задачи на другую, это происходит настолько быстро, что создается иллюзия одновременности выполняемых операций.

Для того чтобы программы MS DOS могли работать внутри защищенной многозадачной среды, используется режим работы процессора виртуальной адресации i86 PM 86 (Protected Mode 86). В результате включения механизма PM 86 реализуется не четырех-, а менее эффективная двухуровневая система защиты. Память разбивается на страницы. В каждой странице формируются окна емкостью 1 Мбайт. В подобном окне может храниться отдельная программа DOS или другой операционной системы. Совокупность хранимых подобным образом программ называется виртуальными машинами, которые работают в многозадачном режиме.
Идея многозадачности не нова. Многозадачность реализуется на больших компьютерах типа мэйнфрэйм (mainframe), к которым подключены десятки, а иногда и сотни, терминалов. У каждого пользователя, сидящего за экраном такого терминала, создается впечатление, что он имеет эксклюзивный доступ ко всей машине. Кро-

ме того, операционные системы мэйнфрэймов часто дают возможность пользователям перевести задачу в фоновый режим, где они выполняются в то время, как пользователь может работать с другой программой.
Для того, чтобы многозадачность стала реальностью на персональных компьютерах, потребовалось достаточно много времени. Но, кажется, сейчас мы приближаемся к эпохе использования многозадачности на ПК (PC). Как мы увидим вскоре, некоторые расширенные 16-разрядные версии Windows поддерживают многозадачность, а имеющиеся теперь в нашем распоряжении Windows NT и Windows 95 - 32-разрядные версии Windows, поддерживают кроме многозадачности еще и многопоточность (multithreading).Многопоточность - это возможность программы самой быть многозадачной. Программа может быть разделена на отдельные потоки выполнения (threads), которые, как кажется, выполняются параллельно. На первый взгляд эта концепция может показаться едва ли полезной, но оказывается, что программы могут использовать многопоточность для выполнения протяженных во времени операций в фоновом режиме, не вынуждая пользователя надолго отрываться от машины.

14.2. Режимы многозадачности.

На заре использования персональных компьютеров некоторые отстаивали идею многозадачности для будущего, но многие ломали головы над вопросом: какая польза от многозадачности на однопользовательской машине? В действительности оказалось, что многозадачность - это именно то, что необходимо пользователям, даже не подозревавшим об этом.
Многозадачность в DOS
Микропроцессор Intel 8088, использовавшийся в первых ПК, не был специально разработан для реализации многозадачности. Частично проблема (как было показано в предыдущей главе) заключалась в недостатках управления памятью. В то время, как множество программ начинает и заканчивает свое выполнение, многозадачная операционная система должна осуществлять перемещение блоков памяти для объединения свободного пространства. На процессоре 8088 это было невозможно реализовать в стиле, прозрачном для приложений.
Сама DOS не могла здесь чем-либо существенно помочь. Будучи разработанной таким образом, чтобы быть маленькой и не мешать приложениям, DOS поддерживала, кроме загрузки программ и обеспечения им доступа к файловой системе, еще не так много средств.Тем не менее, творческие программисты, работавшие с DOS на заре ее появления, нашли путь преодоления этих препятствий, преимущественно при использовании резидентных (terminate-and-stay-resident, TSR) программ. Некоторые TSR-программы, такие как спулер печати, использовали прерывание аппаратного таймера для выполнения процесса в фоновом режиме. Другие, подобно всплывающим (popup) утилитам, таким как SideKick, могли выполнять одну из задач переключения - приостановку выполнения приложения на время работы утилиты. DOS также была усовершенствована для обеспечения поддержки резидентных программ.Некоторые производители программного обеспечения пытались создать многозадачные оболочки или оболочки, использующие переключение между задачами, как надстройки над DOS (например, Quarterdeck's DeskView), но только одна из этих оболочек получила широкое распространение на рынке. Это, конечно, Windows.

14.3. Невытесняющая многозадачность.

Когда Microsoft выпустила на рынок Windows 1.0 в 1985 году, это было еще в большой степени искусственным решением, придуманным для преодоления ограничений MS DOS. В то время Windows работала в реальном режиме (real mode), но даже тогда она была способна перемещать блоки физической памяти (одно из необходимых условий многозадачности) и делала это, хотя и не очень прозрачно для приложений, но все-таки вполне удовлетворительно.
В графической оконной среде многозадачность приобретает гораздо больший смысл, чем в однопользовательской операционной системе, использующей командную строку. Например, в классической операционной системе UNIX, работающей с командной строкой, существует возможность запускать программы из командной строки так, чтобы они выполнялись в фоновом режиме. Однако, любой вывод на экран из программы должен быть переадресован в файл, иначе этот вывод смешается с текущим содержимым экрана.
Оконная оболочка позволяет нескольким программам выполняться совместно, разделяя один экран. Переключение вперед и назад становится тривиальным, существует возможность быстро передавать данные из одной программы в другую, например, разместить картинку, созданную в программе рисования, в текстовом файле, образованном с помощью текстового процессора. Передача данных поддерживалась в различных версиях Windows: сначала с использованием папки обмена (clipboard), позднее - посредством механизма динамического обмена данными (Dynamic Data Exchange, DDE), сейчас - через внедрение и связывание объектов (Object Linking and Embedding, OLE).И все же, реализованная в ранних версиях Windows многозадачность не была традиционной вытесняющей, основанной на выделении отрезков времени, как в многопользовательских операционных системах. Такие операционные системы используют системный таймер для периодического прерывания выполнения одной задачи и запуска другой. 16-разрядные версии Windows поддерживали так называемую невытесняющую многозадачность (non-preemptive multitasking). Такой тип многозадачности был возможен благодаря основанной на сообщениях архитектуре Windows. В общем случае, Windows-программа находилась в памяти и не выполнялась до тех пор, пока не получала сообщение. Эти сообщения часто являлись прямым или косвенным результатом ввода информации пользователем с клавиатуры или мыши. После обработки сообщения программа возвращала управление обратно Windows.
16-разрядные версии Windows не имели возможности произвольно переключать управление с одной Windows-программы на другую, основываясь на квантах времени таймера. Переключение между задачами происходило в момент, когда программа завершала обработку сообщения и возвращала управление Windows. Такую невытесняющую многозадачность называют также кооперативной многозадачностью (cooperative multitasking) потому, что она требует некоторого согласования между приложениями. Одна Windows-программа могла парализовать

работу всей системы, если ей требовалось много времени для обработки сообщения.Хотя невытесняющая многозадачность была основным типом многозадачности в 16-разрядных версиях Windows, некоторые элементы вытесняющей (примитивной, preemptive) многозадачности в них тоже присутствовали.
Windows использовала вытесняющую многозадачность для выполнения DOS-программ, а также позволяла библиотекам динамической компоновки (DLL) получать прерывания аппаратного таймера для задач мультимедиа.
16-разрядные версии Windows имели некоторые особенности, которые помогали программистам если не разрешить, то, по крайней мере, справиться с ограничениями, связанными с невытесняющей многозадачностью. Наиболее известной является отображение курсора мыши в виде песочных часов. Конечно, это не решение проблемы, а только лишь возможность дать знать пользователю, что программа занята выполнением протяженной во времени работы, и что система какое-то время будет недоступна. Другим частичным решением является использование системного таймера Windows, что позволяет выполнять какие-либо действия периодически. Таймер часто используется в приложениях типа часов и приложениях, работающих с анимацией.
Многозадачная ОС должна обеспечивать управление работой вычислительной системой в тех случаях, когда эта работа состоит из нескольких, одновременно идущих видов деятельности.
В многозадачной системе каждый вид деятельности, который может осуществляться одновременно с другими, называется процессом или задачей. Каждый процесс выполняет программу, состоящую из команд и исходных данных. Одна и та же программа может выполняться несколькими процессами. Процесс - это совокупность инструкций, выполняемых процессором, данных и информации о выполняемой задаче, такой как размещенная память, открытые файлы и статус процесса.

14.4. Многозадачность в защищенном режиме.

Полностью механизмы поддержки многозадачности процессор реализует в специальном режиме работы, называемом защищенным, так как именно в этом режиме возможно выполнение нескольких процессов с полной изоляцией их друг от друга. Каждый из них будет "считать", что выполняется только он один и все ресурсы процессора принадлежат ему; он не нарушит работу ОС и других параллельно выполняемых программ.
В отличие от реального режима, в защищенном режиме микропроцессор предоставляет аппаратную поддержку многозадачности. Эта поддержка включает в себя возможность сохранения задачи и передачи управления другой задаче
Каждая задача описывается селектором дескриптора сегмента *TSS* - Task State Segment или сегмент состояния задачи. Сегмент TSS для каждой задачи должен быть описан в таблице GDT (загрузка дескриптора из LDT вызывает ошибку). Сегмент состояния задачи представляет собой 0x67 байт данных, в которых сохраняется состояние задачи (т.е. - содержание регистров процессора и некоторые другие данные) при прекращении ее выполнения.
Пример дескриптора сегмента TSS в таблице GDT**67 00 00 00 00 89 40 00** Первые 0x67 байт линейного адресного пространства будут использованы для хранения состояния задачи, селектор которой указывает на этот дескриптор Бит 4 пятого байта дескриптора определяет дескриптор как системный. В этом случае биты 0-3 определяют тип дескриптора. Для TSS это:0001 - свободный 16 битный сегмент состояния задачи 0011 - занятый 16 битный TSS1001 - свободный 32 битный TSS1011 - занятый 32 битный TSS A также:0101 - шлюз задачи
Селектор выполняющейся в текущий момент времени задачи содержится в регистре TR. В ОС обычно этот регистр загружается командой LTR при запуске первой задачи, выполняющейся в многозадачном режиме
Переключение задач производится с помощью дальнего JMP или CALL, если селектор указывает на TSS или шлюз задачи (а также при прерывании, если сегмент обработчика - TSS). При переключении на задачу, ее тип в дескрипторе изменяется на занятый, т.е. задачи не обладают повторной входимостью. Попытка переключения на занятую задачу вызывает ошибку защиты.

Тема 15. Программы-отладчики.

15.1. Методы и средства отладки. Отладка на примере процессора i486.

Отладка – это этап разработки программы, ставящий целью обнаружение, поиск и устранение ошибки.
Методы и средства отладки.
1 "Ручной метод" - программист вставляет в исходный текст программы фрагменты, позволяющие визуально контролировать ход выполнения программы.Обычно это контрольный вывод промежуточных и окончательных результатов. Некоторые трансляторы с языков высокого уровня позволяют помечать отдельные операторы как комментарии особого рода. Такому транслятору можно указать режим трансляции при котором эти комментарии будут восприниматься как обычные операторы. После отладки программы достаточно перетранслировать программу обычным образом и "лишние" операторы будут исключены.В случае простейших систем контрольной информацией является подача звуковых или световых сигналов, отражающих ход процесса (звуковые сигналы при тестировании и загрузке ОС в современных компьютерах).Недостатками этого метода являются необходимость набора дополнительных частей программы и искажения, вносимые добавлением и удалением фрагментов.
Преимуществом - полный контроль программиста за процедурой поиска ошибки.
2 "Сервис трансляторов" - контроль наиболее характерных ошибок разработчик транслятора может возложить на библиотеку стандартных операций языка (контроль правильности индекса массива при вычислении действительного адреса, проверка допустимости аргумента функции или значения переменной ограниченного типа...).Современные трансляторы позволяют создать исполняемый код, выполняющий программу в отладочном режиме пооператорно с выводом значений указанных программистом переменных после каждого ша-

га.Недостаток метода - дополнительные затраты процессорного времени на постоянный контроль значений (обычно этот контроль можно отключить, но надо иметь дополнительные знания по управлению транслятором).
Преимущество - достаточный автоматизм поиска грубых ошибок, что важно для начинающего пользователя (типичный пример язык "ПАСКАЛЬ" с полным отсутствием понятия "по умолчанию").
3 "Отладчики" - в отличие от вышеперечисленных методов позволяют, в общем случае, отладить программу несвязанную с конкретным языком программирования (программу в кодах).Эти отладочные системы можно поделить на две группы:

- "Интерпретаторы" - программы отладчиков, моделирующие работу отлаживаемой программы в среде конкретной вычислительной системы. При этом программа анализируется байт за байтом так, как это делал бы реальный процессор, а при обнаружении операции ввода/вывода формируется соответствующий запрос к оператору.

Недостатки интерпретирующей системы - невозможность отладить программу, работающую в реальном масштабе времени, кроме этого сама система занимает много места в памяти машины.Преимущества - отладка может вестись на более мощном инструментальном компьютере с другой системой команд процессора и структурой внешних устройств. Возможна отладка самомодифицирующихся программ.

- Системы работающие "в живую" - отладчики выступающие как дополнения к отлаживаемой программе. Эти системы вносят небольшие изменения в программу, позволяющие им в нужный, с точки зрения оператора, момент получать управление и осуществлять оговоренные оператором дополнительные действия. Для передачи управления осуществляется замена части кода отлаживаемой программы на команду обращения к подпрограмме или на программное прерывание, после получения управления отладчиком, исходный код восстанавливается. Современные микропроцессоры могут имеют встроенные средства автоматического обращения к программе отладчика, обычно переключение выполняется после выполнения каждой команды.Недостатки системы - могут возникнуть проблемы при отладке модифицирующейся программы если неправильно выбрана точка прерывания. Нет возможности проконтролировать работу программы, расположенной в ПЗУ. Преимущество - отладчик может быть очень простым и быть размещен даже при малом объеме свободной памяти.

4 "Внутрисхемные эмуляторы" - аппаратно-программные сверхбыстродействующие устройства, подключаемые к отлаживаемой системе вместо одной или нескольких микросхем полностью заменяя их и предоставляя дополнительные возможности. Недостаток системы - большая стоимость. Преимущества - полное отсутствие искажений в отлаживаемой системе и максимальный сервис при отладке.

Процессор i486 имеет расширенные встроенные средства отладки, среди которых:

- Пошаговое выполнение;
- Прерывание контрольной точки (командное);
- Ловушка переключения задач;
- Система отладочных регистров.

Пошаговое выполнение
Обеспечивает останов после выполнения каждой команды программы. Вводится загрузкой значения TF=1 в регистре флагов EFLAGS (бит8). Исключение 1 пошагового режима имеет место как ловушка после выполнения команды, если до выполнения команды был установлен флаг TF (сама команда, установившая флаг прерывания не вызывает). Процессор очищает флаг перед вызовом обработчика исключений. Флаг не очищается при изменении привилегированности внутри задачи, поэтому за этим должна следить ОС. Флаг очищается при выполнении команды INT, следовательно отладчики должны эмулировать эти команды, если необходимо сохранить пошаговый режим для программы прерывания.
Прерывание контрольной точки.
Ловушка контрольной точки вызывается выполнением команды INT 3. Контрольная точка подготавливается отладчиком, который заменяет первый байт кода операции на значение CCh. При выполнении исключения, адрес возврата указывает на байт, следующий за командой INT 3 и должен быть скорректирован при восстановлении исходного значения кода. В процессоре i486 используется как вспомогательное средство.
Ловушка переключения задач.
Отладочное исключение происходит после переключения задачи при установленном бите T в TSS новой задачи. Исключение (как и в случае пошагового выполнения номер 1) происходит перед выполнением первой команды задачи.
ВНИМАНИЕ! Если отладчик оформлен в виде исключения с установленным битом ловушки, то произойдет зацикливание процессора.
Система отладочных регистров.
Для управления отладкой используется шесть регистров. Доступ к ним осуществляется при помощи команды MOV. Обращение возможно только из нулевого кольца привилегий.
Всего имеется восемь регистров, их назначение:

- DR0..DR3 Каждый из этих регистров содержит линейный адрес одной из четырех контрольных точек. Если подкачка страниц разрешена, то их значения транслируются в физические адреса по общему алгоритму;
- DR4..DR5 Регистра зарезервированы и в процессоре i486 не используются;
- DR6 Отладочный регистр состояния. Он сообщает об условиях, выявленных во время генерирования отладочного исключения (номер 1). Биты регистра устанавливаются аппаратно, а сбрасываются программно;
- DR7 Регистр задает вид доступа к памяти, связанный с каждой контрольной точкой.

Тема 16. Принципы программирования микропроцессоров.
16.1. Программирование микропроцессора. Машинно-ориентированные языки.

Принципиальным достоинством МП является программируемость. Это означает, что, подавая на вход МП команды, можно обеспечить нужную последовательность операций, т.е. реализацию определенного алгоритма. Алгоритм решаемой задачи может быть сколь угодно сложным, необходимо лишь, чтобы этот алгоритм был разбит на шаги в соответствии с системой команд МП. Поэтому система команд важна не только с точки зрения, что МП может делать, но и как выполняется алгоритм. Наличие или отсутствие какой-либо команды или группы команд может существенно повлиять на выбор МП для конкретного применения.
Возвращаясь к языково-программным классификационным характеристикам, нельзя не упомянуть о языках программирования.
Все языки программирования условно можно разделить на три уровня:
- машинный код;
- автокод (язык ассемблера);
- языки высокого уровня (процедурные языки - BASIC, FORTRAN, PASCAL, C, MODULA-2, ADA; и языки искусственного интеллекта - LISP, PROLOG, SMALLTALK, OCCAM).

Более понятные для ЭВМ - это так называемые машинно-ориентированные языки (машинный код и язык ассемблера). Более понятные для человека именуют языками высокого уровня.
Программное обеспечение на машинно-ориентированном языке экономично в эксплуатации, однако сравнительно высокая трудоемкость и длительность разработки программного обеспечения обуславливают преимущественное применение их для создания и развития программного обеспечения драйверов и операционных систем с целью наилучшего использования аппаратных особенностей каждой конкретной ЭВМ.

16.2. Языки высокого уровня. Специализированные языки.

Алгоритмические языки (языки программирования высокого уровня общего назначения) являются машинно-независимыми, позволяют создавать компактные обозримые программы при относительно небольших затратах времени и труда программистов. Разработка программ значительно упрощается при использовании языков высокого уровня в качестве языков программирования. Однако при этом снижается эффективность программ по быстродействию и затратам памяти в сравнении с применением языка ассемблера. Но этот недостаток с лихвой перекрывается четкостью и легкостью написания программы.
Языки высокого уровня в свою очередь подразделяются на языки процедурного (или императивного) и эвристического (декларативного) стиля программирования (языки искусственного интеллекта). Наиболее популярные языки программирования ПЭВМ высокого уровня приведены в таблице 2.

Таблица 16.

Язык	Год разработки	Разработчик	Основное применение
FORTRAN	1954	Дж. Бэкус (США)	Математические расчеты, научные исследования
BASIC	1965	Дж. Кенеми (США)	Обучение, тестовые программы
PASCAL	1971	Н.Вирт (Швейцария)	Обучение, широкое применение
C	1972	Д.М.Ричи (США)	Системное программирование
MODULA-2	1981	Н.Вирт (Швейцария)	Разработка больших программных комплексов
LISP	1960	Дж. Маккарти (США)	Системы искусственного интеллекта
PROLOG	1971	А.Колмедауэр (Франция)	Принятие решений, логический вывод
SMALLTALK	Середина 1970-х	А.КейАнглия)	Системы диалога со средствами машинной графики
OCCAM	Начало 1980-х	Фирма INMOS(Англия)	Системы с параллельными процессами

Кроме того, в настоящее время появились языки так называемого 4-го поколения - это языки СУБД, электронных таблиц, интегрированных систем и т.д., которые предназначены для решения узкого круга задач прикладного характера (например, обработка баз данных), но зато еще больше, по сравнению с языками общего назначения, снижают затраты времени и труда на создание выходного продукта.

Опыт применения ПЭВМ для построения прикладных систем обработки данных показывает, что самым эффективным инструментом создания контроллера являются не универсальные языки высокого уровня, а узкоспециализированные языки - как правило языки высокого манипулирования с особенностями микропроцессора. Система микропрограммирования является набором компактных программных продуктов для разработки программ для микропроцессоров. СМ реализована для работы на ряде компьютеров, от небольших 16-разрядных персональных машин до 32-разрядных суперминикомпьютеров.

В нем имеется ряд примеров использования как стандартных, так и имеющих особенности средств СМ. Отметим, что независимые средства ассемблера СМ очень просты и эффективны.

СМ ассемблеры - это мощные МАКРО-ассемблеры со средствами перемещения программ, с универсальными характеристиками и применением. Хотя ассемблеры созданы на базе одного и того же основного пакета, они обладают высокой степенью совместимости с ассемблерами разработчиков микропроцессоров. Основные предметы - это способы использования ассемблера, поддержка модульного программирования и связь с языками высокого уровня.
Все ассемблеры двухпроходные, выполняются как одна программа. Во время выполнения не создается временных файлов.
Все ассемблеры, так же как и XLINK, используют для внутренних вычислений 32-разрядные структуры, что позволяет виртуально генерировать код любого размера (т.е. не существует предела в 64 кБайт, что могло бы затруднить использование процессоров типа 68000).
Для обеспечения совместимости внутри пакета было применено несколько компромиссных решений, с учетом совместимости с ассемблерами разработчиков микропроцессоров. Особенно это касается макро-конструкций, которые сильно различаются у различных разработчиков. Во многих отношениях, однако, СМ превосходит оригинальные ассемблеры.
Они должны быть совместимы по:
- машинным командам (именам и синтаксису)
- директивам определения констант (именам и синтаксису)
- директивам распределения памяти (именам и синтаксису)
- разделителям
- меткам
- основным операторам (+,-,*,/)
- ORG и EQU

Могут быть несовместимы по:
- директивам перемещения
- расширениям операторов
- средствам условной трансляции
- опциям и командам управления ассемблером
- макросредствам.

Заметим, что средства, перечисленные в разделе "не совместимы", часто отличаются от оригинальных ассемблеров разработчиков только синтаксисом.
Наиболее популярными на сегодняшний день у программистов являются С-51 и Assembler 8051, так как оба они позволяют получать исходный код.

Тема 17. Команды языка Ассемблер.
17.1. Основные группы операций. Мнемокоды команд процессора Pentium.

Микропроцессоры выполняют набор команд, которые реализуют следующие основные группы операций:
- операции пересылки,
- арифметические операции,
- логические операции,
- операции сдвига,
- операции сравнения и тестирования,
- битовые операции,
- операции управления программой;
- операции управления процессором.

При описании команд обычно используются их мнемонические обозначения (мнемокоды), которые служат для задания команды при программировании на языке Ассемблера. Для различных версий Ассемблера мнемокоды некоторых команд могут отличаться. Например, для команды вызова подпрограммы используется мнемокод ***CALL*** или ***JSR*** (“Jump to SubRoutine”). Однако мнемокоды большинства команд для основных типов микропроцессоров совпадают или отличаются незначительно, так как они являются сокращениями соответствующих английских слов, определяющих выполняемую операцию. Рассмотрим мнемокоды команд, принятые для процессоров Pentium.

Команды пересылки. Основной командой этой группы является команда ***MOV***, которая обеспечивает пересылку данных между двумя регистрами или между регистром и ячейкой памяти. В некоторых микропроцессорах реализуется пересылка между двумя ячейками памяти, а также групповая пересылка содержимого нескольких регистров в память или их загрузка из памяти. Например, микропроцессоры семейства 68xxx компании Motorola выполняют команду ***MOVE***, обеспечивающую пересылку из одной ячейки памяти в другую, и команду ***MOVEM***, которая производит запись в память или загрузку из памяти содержимого заданного набора регистров (до 16 регистров). Команда ***XCHG*** производит взаимный обмен содержимым двух регистров процессора или регистра и ячейки памяти.

Команды ввода ***IN*** и вывода ***OUT*** реализуют пересылку данных из регистра процессора во внешнее устройство или прием данных из внешнего устройства в регистр. В этих командах задается номер интерфейсного устройства (порта ввода-вывода), через которое производится передача данных. Отметим, что многие микропроцессоры не имеют специальных команд для обращения к внешним устройствам. В этом случае ввод и вывод данных в системе выполняется с помощью команды ***MOV***, в которой задается адрес требуемого интерфейсного устройства. Таким образом внешнее устройство адресуется как ячейка памяти, а в адресном пространстве выделяется определенный раздел, в котором располагаются адреса подключенных к системе интерфейсных устройств (портов).

Команды арифметических операций. Основными в этой группе являются команды сложения, вычитания, умножения и деления, которые имеют ряд вариантов. Команды сложения ***ADD*** и вычитания ***SUB*** выполняют соответствующие операции с содержимым двух регистров, регистра и ячейки памяти или с использованием непосредственного операнда. Команды ***ADC, SBB*** производят сложение и вычитание с учетом значения признака **C**, устанавливаемого при формировании переноса в процессе выполнения предыдущей операции. С помощью этих команд реализуется последовательное сложение операндов, число разрядов которых превышает разрядность процессора. Команда ***NEG*** изменяет знак операнда, переводя его в дополнительный код.

Операции умножения и деления могут выполняться над числами со знаком (команды ***IMUL, IDIV***) или без знака (команды ***MUL, DIV***). Один из операндов всегда размещается в регистре, второй может находиться в регистре, ячейке памяти или быть непосредственным операндом. Результат операции располагается в регистре. При умножении (команды ***MUL, IMUL***) получается результат удвоенной разрядности, для размещения которого используется два регистра. При делении (команды ***DIV, IDIV***) в качестве делимого используется операнд удвоенной разрядности, размещаемый в двух регистрах, а в качестве результата в два регистра записывается частное и остаток.

Команды логических операций. Практически все микропроцессоры производят логические операции И, ИЛИ, Исключающее ИЛИ, которые выполняются над одноименными разрядами операндов с помощью команд ***AND, OR, XOR***. Операции выполняются над содержимым двух регистров, регистра и ячейки памяти или с использованием непосредственного операнда. Команда ***NOT*** инвертирует значение каждого разряда операнда.

Команды сдвига. Микропроцессоры осуществляют арифметические, логические и циклические сдвиги адресуемых операндов на один или несколько разрядов. Сдвигаемый операнд может находиться в регистре или ячейке памяти, а число разрядов сдвига задается с помощью непосредственного операнда, содержащегося в команде, или определяется содержимым заданного регистра. В реализации сдвига обычно участвует признак переноса **C** в регистре состояний (**SR** или **EFLAGS**), в котором располагается последний разряд операнда, выдвигаемый из регистра или ячейки памяти.

Команды сравнения и тестирования. Сравнение операндов обычно производится с помощью команды ***CMP***, которая производит вычитание операндов с установкой значений признаков **N, Z, V, C** в регистре состояния в соответствии с полученным результатом. При этом результат вычитания не сохраняется, и значения операндов не изменяются. Последующий анализ полученных значений признаков позволяет определить относительное значение (>, <, =) операндов со знаком или без знака. Использование различных способов адресации позволяет производит сравнение содержимого двух регистров, регистра и ячейки памяти, непосредственно заданного операнда с содержимым регистра или ячейки памяти.

Некоторые микропроцессоры выполняют команду тестирования ***TST***, которая является однооперандным вариантом команды сравнения. При выполнении этой команды устанавливаются признаки **N, Z** в соответствии со знаком и значением (равно или не равно нулю) адресуемого операнда.

Команды битовых операций. Эти команды производят установку значения признака **C** в регистре состояний в соответствии со значением тестируемого бита **bn** в адресуемом операнде. В некоторых микропроцессорах по результату тестирования бита производится установка признака **Z**. Номер тестируемого бита **n** задается либо содержимым указанного в команде регистра, либо непосредственным операндом.

Команды данной группы реализуют разные варианты изменения тестируемого бита. Команда ***BT*** сохраняет значение этого бита неизменным. Команда ***BTS*** после тестирования устанавливает значение **bn**=1, а команда ***BTC*** - значение **bn**=0. Команда ***BTC*** инвертирует значение бита bn после его тестирования.

Операции управления программой. Для управления программой используется большое количество команд, среди которых можно выделить:

- команды безусловной передачи управления;
- команды условных переходов;
- команды организации программных циклов;
- команды прерывания;
- команды изменения признаков.

Безусловная передача управления производится командой ***JMP***, которая загружает в программный счетчик **PC** новое содержимое, являющееся адресом следующей выполняемой команды. Этот адрес либо непосредственно

указывается в команде ***JMP*** (прямая адресация), либо вычисляется как сумма текущего содержимого **PC** и заданного в команде смещения, которое является числом со знаком (относительная адресация). Так как **PC** содержит адрес очередной команды программы, то последний способ задает адрес перехода, смещенный относительно очередного адреса на заданное число байтов. При положительном смещении производится переход к последующим командам программы, при отрицательном смещении – к предыдущим.

Вызов подпрограммы также производится путем безусловной передачи управления с помощью команды ***CALL*** (или ***JSR***). Однако в этом случае перед загрузкой в ***PC*** нового содержимого, задающего адрес первой команды подпрограммы, необходимо сохранить его текущее значение (адрес очередной команды), чтобы после выполнения подпрограммы обеспечить возвращение к основной программе (или к предыдущей подпрограмме при вложении подпрограмм). Команды условных переходов (ветвлений программы) производят загрузку в **PC** нового содержимого, если выполняются определенные условия, которые обычно задаются в соответствии с текущим значением различных признаков в регистре состояния. Если условие не реализуется, то выполняется следующая команда программы.

Команды управления признаками обеспечивают запись - чтение содержимого регистра состояния, в котором хранятся признаки, а также изменение значений отдельных признаков. Например, в процессорах Pentium реализуются команды ***LAHF*** и ***SAHF***, которые выполняют загрузку младшего байта, где содержатся признаки, из регистра состояния **EFLAG** в младший байт регистра **EAX** и заполнение младшего байта **EFLAGS** из регистра **EAX**.. Команды **CLC, STC** осуществляют установку значений признака переноса CF=0, CF=1, а команда **CMC** вызывает инвертирование значения этого признака. Так как признаки определяют ход выполнения программы при условных переходах, то команды изменения признаков обычно используются для управления программой.

Команды управления процессором. К этой группе относятся команды останова, отсутствия операции и ряд команд, определяющих режим работы процессора или его отдельных блоков. Команда ***HLT*** прекращает выполнение программы и переводит процессор в состояние останова, выход из которого происходит при поступлении сигналов прерывания или перезапуска (Reset). Команда ***NOP*** (“пустая” команда), которая не вызывает выполнения каких-либо операций, служит для реализации программных задержек или заполнения пропусков, образовавшихся в программе.

Специальные команды ***CLI, STI*** запрещают и разрешают обслуживание запросов прерывания. В процессорах Pentium для этого используется бит управления (флаг) **IF** в регистре **EFLAGS.**

Многие современные микропроцессоры выполняют команду идентификации, которая позволяет пользователю или другим устройствам получить информацию о типе процессора, используемого в данной системе. В процессорах Pentuim для этого служит команда ***CPUID***, при выполнении которой необходимые данные о процессоре поступают в регистры **EAX, EBX, ECX, EDX** и могут затем считываться пользователем или операционной системой.

В зависимости от реализуемых процессором режимов работы и заданных типов обрабатываемых данных набор выполняемых команд может существенно расширяться.

Некоторые процессоры производят арифметические операции с двоично-десятичными числами или выполняют специальные команды коррекции результата при обработке таких чисел. В состав многих высокопроизводительных процессоров входит FPU - блок обработки чисел с “плавающей точкой”.

В ряде современных процессоров реализована групповая обработка нескольких целых чисел или чисел с “плавающей точкой” с помощью одной команды по принципу SIMD (“Single Instruction – Multiple Data”) - «Одна команда – Множество данных». Одновременное выполнение операций над несколькими операндами существенно повышает производительность процессора при работе с видео- и аудиоданными. Такие операции широко используются для обработки изображений, звуковых сигналов и в других приложениях. Для выполнения этих операций в состав процессоров введены специальные блоки, реализующие соответствующие наборы команд, которые в различных типах процессоров (Pentium, Athlon) получили название **MMX** (“Milti-Media Extension”) – Мультимедийное Расширение, **SSE** (“Streaming SIMD Extension”) – Потоковое SIMD – расширение, **“3D – Extension”** – Трехмерное Расширение.

Характерной особенностью процессоров компании Intel, начиная с модели 80286, является приоритетный контроль при обращении к памяти, который обеспечивается при работе процессора в режиме защищенных виртуальных адресов – “Protected Mode” (защищенный режим). Для реализации этого режима используется специальные группы команд, которые служат для организации защиты памяти в соответствии с принятым алгоритмом приоритетного обращения.

4.3. Лабораторные работы

№ п/п	*Номер раздела дисциплины*	*Наименование тем лабораторны занятий*	*Объем (час.)*	*Вид занятия в интерактивной, активной, инновационной формах, (час.)*
1	**9.**	Регистры микропроцессора	4	2
2	**13.**	Система прерываний	4	3
3	**6.**	Производительность ЭВМ	3	2
4	**5.**	Отложенный запуск команд	4	3
5	**16.**	Изучение программного обеспечения управления роботами на примере программы Робот PASCAL DELTA 1-3X-USB+.	3	2
		ИТОГО	**18**	**12**

4.4. Практические занятия

Учебным планом не предусмотрено.

4.5. Контрольные мероприятия: курсовой проект (курсовая работа), контрольная работа, РГР, реферат

Учебным планом не предусмотрено.

5. МАТРИЦА СООТНЕСЕНИЯ РАЗДЕЛОВ УЧЕБНОЙ ДИСЦИПЛИНЫ К ФОРМИРУЕМЫМ В НИХ КОМПЕТЕНЦИЯМ И ОЦЕНКЕ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Компетенции *№, наименование разделов дисциплины*	*Кол-во часов*	*Компетенции* *ПК* *6*	*Σ комп.*	$t_{ср}$, *час*	*Вид учебных занятий*	*Оценка результатов*
1	2	3	5	6	7	8
1.Введение. История развития микропроцессоров.	6	+	1	6	Лк, СРС	Экзамен
2.Внутренняя структура микропроцессоров. Принципы фон Неймана.	6	+	1	6	Лк, СРС	Экзамен
3.Классификация микропроцессоров.	8	+	1	8	Лк, СРС	Экзамен
4.Внутренняя структура микропроцессоров. Арифметико-логический блок.	6	+	1	6	Лк, СРС	Экзамен
5.Устройство управления.	10	+	1	10	Лк, ЛР, СРС	Экзамен
6.Система команд микропроцессора.	9	+	1	9	Лк, ЛР, СРС	Экзамен
7.Режимы работы микропроцессора.	6	+	1	6	Лк, СРС	Экзамен
8.Принципы формирования адресного пространства.	4	+	1	4	Лк, СРС	Экзамен
9.Система адресации.	10	-	1	10	Лк, ЛР, СРС	Экзамен
10.Память как функциональный узел.	6	-	1	6	Лк, СРС	Экзамен
11.Принципы организации памяти.	6	-	1	6	Лк, СРС	Экзамен
12.Виртуальная память.	8	-	1	8	Лк, ЛР, СРС	Экзамен
13.Прерывания.	9	-	1	9	Лк, СРС	Экзамен
14.Поддержка многозадачности.	5	-	1	5	Лк, СРС	Экзамен
15.Программы-отладчики.	4	-	1	4	Лк, СРС	Экзамен
16.Принципы программирования микропроцессоров.	9	-	1	9	Лк, ЛР, СРС	Экзамен
17.Команды языка Ассемблер	5	-	1	5	Лк, СРС	Экзамен
всего часов	**117**	**117**	**1**	**117**		

6. ПЕРЕЧЕНЬ УЧЕБНО-МЕТОДИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем., табл. ;[Электронный ресурс].- URL: http://biblioclub.ru/index.php?page=book&id=458701 (с.1-158)

7. ПЕРЕЧЕНЬ ОСНОВНОЙ И ДОПОЛНИТЕЛЬНОЙ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

№	*Наименование издания*	*Вид занятия*	*Количество экземпляров в библиотеке,* **шт.**	*Обеспеченность, (экз./ чел.)*
1	**2**	**3**	**4**	**5**
Основная литература				
1.	Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем., табл. ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=458701	Лк, ЛР	ЭР	1
Дополнительная литература				
1.	Пильщиков, В.Н. Программирование на языке ассемблера IBM PC : учебное пособие / В.Н. Пильщиков. - М. : Диалог-МИФИ, 2014. - 288 с. : ил. - Библиогр. в кн. - ISBN 5-86404-051-7 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=447687	ЛР	ЭР	1
2.	Гуров, В.В. Архитектура микропроцессоров : учебное пособие / В.В. Гуров. - М. : Интернет-Университет Информационных Технологий, 2010. - 272 с. : табл., схем. - (Основы информационных технологий). - ISBN 978-5-9963-0267-3 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=233074	Лк	ЭР	1

8. ПЕРЕЧЕНЬ РЕСУРСОВ ИНФОРМАЦИОННО - ТЕЛЕКОММУНИКАЦИОННОЙ СЕТИ «ИНТЕРНЕТ» НЕОБХОДИМЫХ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1. Электронный каталог библиотеки БрГУ http://irbis.brstu.ru/CGI/irbis64r_15/cgiirbis_64.exe?LNG=&C21COM=F&I21DBN=BOOK&P21DBN=BOOK&S21CNR=&Z21ID=.
2. Электронная библиотека БрГУ http://ecat.brstu.ru/catalog .
3. Электронно-библиотечная система «Университетская библиотека online» http://biblioclub.ru .
4. Электронно-библиотечная система «Издательство «Лань»

http://e.lanbook.com .
5. Информационная система "Единое окно доступа к образовательным ресурсам" http://window.edu.ru .
6. Научная электронная библиотека eLIBRARY.RU http://elibrary.ru .
7. Университетская информационная система РОССИЯ (УИС РОССИЯ) https://uisrussia.msu.ru/ .
8. Национальная электронная библиотека НЭБ http://xn--90ax2c.xn--p1ai/how-to-search /.

9. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

9.1. Методические указания для обучающихся по выполнению лабораторных работ

Лабораторная работа № 1
Регистры микропроцессора.

Цель работы:

Изучить регистры и регистры общего назначения.

Вид занятия в интерактивной, активной форме: выполнить задание и ознакомиться с составом и характеристиками.

Задание:

1. Изучить состав регистров.
2. Изучить характеристики регистров.

Порядок выполнения:

На основании задания, полученного у преподавателя, произвести заданные манипуляции с регистрами микропроцессора.

Форма отчетности:

Отчет не предусмотрен.

Задания для самостоятельной работы:

Изучить основные характеристики и состав регистров.

Рекомендации по выполнению заданий и подготовке к занятию

Ознакомиться с теоретическим материалом, представленным в 9-м разделе данной дисциплины и учебном пособии.

Основная литература

1. Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем, табл.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=458701

Дополнительная литература

1. Пильщиков, В.Н. Программирование на языке ассемблера IBM PC : учебное пособие / В.Н. Пильщиков. - М. : Диалог-МИФИ, 2014. - 288 с. : ил. - Библиогр. в кн. - ISBN 5-86404-051-7 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=447687

Контрольные вопросы для самопроверки

1. Что такое микропроцессор?
2. Что такое регистры?
3. Какие виды регистров существуют?

Лабораторная работа № 2
Система прерываний.

Цель работы:

Изучить прерывания.

Вид занятия в интерактивной, активной форме: выполнить задание и ознакомиться с составом и характеристиками.

Задание:

1. Изучить систему прерываний.
2. Программирование прерываний.

Порядок выполнения:

На основании задания, полученного у преподавателя, произвести заданные прерывания системы.

Форма отчетности:

Отчет не предусмотрен.

Задания для самостоятельной работы:

Изучить основные характеристики и виды прерываний.

Рекомендации по выполнению заданий и подготовке к занятию

Ознакомиться с теоретическим материалом, представленным в 13-м разделе данной дисциплины и учебном пособии.

Основная литература

1. Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем, табл.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=458701

Дополнительная литература

1. Пильщиков, В.Н. Программирование на языке ассемблера IBM PC : учебное пособие / В.Н. Пильщиков. - М. : Диалог-МИФИ, 2014. - 288 с. : ил. - Библиогр. в кн. - ISBN 5-86404-051-7 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=447687

Контрольные вопросы для самопроверки

1. Что такое прерываниие?
2. Какие виды прерываний существуют?

Лабораторная работа № 3
Производительность ЭВМ.

Цель работы:

Вычислить производительность ЭВМ.

Вид занятия в интерактивной, активной форме: выполнить задание и ознакомиться с составом и характеристиками.

Задание:

1. Изучить элементы, отвечающие за производительность ЭВМ.
2. Программирование задачи производительности.

Порядок выполнения:

На основании задания, полученного у преподавателя, произвести замеры производительности ЭВМ.

Форма отчетности:
Отчет не предусмотрен.

Задания для самостоятельной работы:
Изучить основные характеристики, отвечающие за производительность ЭВМ.
Рекомендации по выполнению заданий и подготовке к занятию
Ознакомиться с теоретическим материалом, представленным в 6-м разделе данной дисциплины и учебном пособии.

Основная литература

1. Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем, табл.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=458701

Дополнительная литература

1. Пильщиков, В.Н. Программирование на языке ассемблера IBM PC : учебное пособие / В.Н. Пильщиков. - М. : Диалог-МИФИ, 2014. - 288 с. : ил. - Библиогр. в кн. - ISBN 5-86404-051-7 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=447687

Контрольные вопросы для самопроверки

1. Чем оценивается производительность ЭВМ?
2. Какие шины имеет ЭВМ?

Лабораторная работа № 4
Отложенный запуск команд.

Цель работы:
Написать программу с использованием отложенного запуска команд.

Вид занятия в интерактивной, активной форме: выполнить задание и ознакомиться с составом и характеристиками.
Задание:

1. Программирование задачи с использованием отложенного запуска команд.

Порядок выполнения:
На основании задания, полученного у преподавателя, произвести отложенный запуск команд.

Форма отчетности:
Отчет не предусмотрен.

Задания для самостоятельной работы:
Изучить команды отложенного запуска команд.
Рекомендации по выполнению заданий и подготовке к занятию
Ознакомиться с теоретическим материалом, представленным в 5-м разделе данной дисциплины и учебном пособии.

Основная литература

1. Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем, табл.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=458701

Дополнительная литература

1. Пильщиков, В.Н. Программирование на языке ассемблера IBM PC : учебное пособие / В.Н. Пильщиков. - М. : Диалог-МИФИ, 2014. - 288 с. : ил. - Библиогр. в кн. - ISBN 5-86404-051-7 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=447687

Контрольные вопросы для самопроверки

1. Чем такое отложенный запуск команд?
2. Какие задачи можно решать с помощью отложенного запуска команд?

Лабораторная работа № 5
Изучение состава и характеристик робота PASCAL DELTA 1-3X-USB+.

Цель работы:

Изучить состав и характеристики робота PASCAL DELTA 1-3X-USB+.

Вид занятия в интерактивной, активной форме: выполнить задание и ознакомиться с составом и характеристиками.

Задание:

1. Изучить состав звеньев и оборудования робота.
2. Изучить характеристики робота.

Порядок выполнения:

На основании представленного робота определить состав звеньев, их функции, а также состав оборудования для данного робота. Изучить и описать характеристики робота, его функциональные возможности.

Форма отчетности:

Отчет не предусмотрен.

Задания для самостоятельной работы:

Изучить основные характеристики и состав роботов-манипуляторов.

Рекомендации по выполнению заданий и подготовке к занятию

Ознакомиться с теоретическим материалом, представленным в 16-м разделе данной дисциплины и учебном пособии.

Основная литература

1. Сажнев, А.М. Цифровые устройства и микропроцессоры : учебное пособие / А.М. Сажнев, И.С. Тырышкин ; Новосибирский государственный аграрный университет, Инженерный институт. - Новосибирск : ИЦ НГАУ «Золотой колос», 2015. - 158 с. : схем, табл.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=458701

Дополнительная литература

1. Пильщиков, В.Н. Программирование на языке ассемблера IBM PC : учебное пособие / В.Н. Пильщиков. - М. : Диалог-МИФИ, 2014. - 288 с. : ил. - Библиогр. в кн. - ISBN 5-86404-051-7 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=447687

Контрольные вопросы для самопроверки

1. Что такое робот?
2. Что такое манипулятор?
3. Какие звенья входят в состав робота?
4. Назвать основные составляющие системы управления робота.

10. ПЕРЕЧЕНЬ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, ИСПОЛЬЗУЕМЫХ ПРИ ОСУЩЕСТВЛЕНИИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

Информационно-коммуникационные технологии (ИКТ) – преподаватель использует для:

- получения информации при подготовке к занятиям,
- создания презентационного сопровождения лекций;
- интерактивного общения;
- ОС Windows 7 Professional;
- Microsoft Office 2007 Russian Academic OPEN NO Level;
- Антивирусное программное обеспечение Kaspersky Security;
- ПО “Антиплагиат”.

11. ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

Вид занятия	***Наименование аудитории***	***Перечень основного оборудования***	***№ ЛР или ПЗ***
1	**2**	**3**	**4**
ПЗ	Лаборатория робототех-ники	Персональные компью-теры	ПЗ № 1-5
СР	ЧЗЗ	-	-

Приложение 1

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

1. Описание фонда оценочных средств (паспорт)

№ компетенции	Элемент компетенции	Раздел	Тема	ФОС
ПК-6	способность производить расчеты и проектирование отдельных блоков и устройств систем автоматизации и управления и выбирать стандартные средства автоматики, измерительной и вычислительной техники для проектирования систем автоматизации и управления в соответствии с техническим заданием	**1.** Введение. История развития микропроцессоров.	1.1 Первое поколение ЭВМ. 1.2 Второе поколение ЭВМ. 1.3 Третье поколение ЭВМ. 1.4 Четвертое поколение ЭВМ.	Вопросы к экзамену
		2.Внутренняя структура микропроцессоров. Принципы фон Неймана.	2.1 Определение микропроцессора. 2.2 История развития микропроцессоров. 2.3 Архитектура фон Неймана. 2.4 Принципы фон Неймана.	Вопросы к экзамену
		3. Классификация микропроцессоров.	3.1 Классификация микропроцессоров по числу больших интегральных схем. 3.2 Классификация микропроцессоров по назначению. 3.3 Классификация микропроцессоров по виду обрабатываемых сигналов. 3.4 Классификация микропроцессоров по характеру временной организации. 3.5 Классификация микропроцессоров по организации структуры. 3.6 Классификация микропроцессоров по количеству выполняемых программ.	Вопросы к экзамену
		4. Внутренняя структура микропроцессоров. Арифметико-логический блок.	4.1 Арифметико-логический блок. 4.2 Основные характеристики микропроцессора. 4.3 Структура типового микропроцессора. Выполнение элементарных операций	Вопросы к экзамену
		5. Устройство управления.	5.1 Принцип управления. 5.2 Организация устройств управления. 5.3 Устройство управления микропроцессора. 5.4 Особенности программного и микропрограммного управления	Вопросы к экзамену
		6. Система команд микропроцессора.	6.1 Оптимальная система команд.	Вопросы к экзамену

			6.2 Формат команды. 6.3 Классификация команд. Примеры кодирования команд.	
		7. Режимы работы микропроцессора.	7.1 Режимы работы микропроцессора. Защищенный режим. Реальный режим. 7.2 Режим системного управления. Переключение между режимами.	Вопросы к экзамену
		8. Принципы формирования адресного пространства.	8.1 Среда выполнения задачи. Архитектурные решения микропроцессоров.	Вопросы к экзамену
		9. Система адресации.	9.1 Режимы адресации. 9.2 Способы адресации. 9.3 Режимы адресации операндов. Возможности микропроцессоров по адресации.	Вопросы к экзамену
		10. Память как функциональный узел.	10.1 Внутренняя память компьютера. 10.2 Технология производства интегральных схем. 10.3 Основные характеристики полупроводниковой памяти.	Вопросы к экзамену
		11. Принципы организации памяти.	11.1 Организация физической памяти. ОЗУ, ПЗУ и FLASH – память. 11.2 Логическая структура основной памяти.	Вопросы к экзамену
		12. Виртуальная память.	12.1 Модели памяти. 12.2 Понятие о сегментированной модели памяти. 12.3 Понятие о страничной модели памяти. 12.4 Сегментно-страничная модель памяти. 12.5 Плоская модель памяти.	Вопросы к экзамену
		13. Прерывания.	13.1 Понятие прерывания. 13.2 Классификация прерываний. 13.3 Внешние прерывания. 13.4 Система прерываний. Аппаратные и программные средства системы прерываний. Обработка прерываний в реальном режиме.	Вопросы к экзамену

		14. Поддержка многозадачности.	14.1 Многозадачность – определение, история развития. 14.2 Режимы многозадачности. 14.3 Невытесняющая многозадачность. 14.4 Многозадачность в защищенном режиме.	Вопросы к экзамену
		15. Программы-отладчики.	15.1 Методы и средства отладки. Отладка на примере процессора 486.	Вопросы к экзамену
		16. Принципы программирования микропроцессоров.	16.1 Программирование микропроцессора. Машинно-ориентированные языки. 16.2 Языки высокого уровня. Специализированные языки.	Вопросы к экзамену
		17. Команды языка Ассемблер	17.1 Основные группы операций. Мнемокоды команд процессора Pentium.	Вопросы к экзамену

2. Экзаменационные вопросы

№ п/п	Компетенции		ЭКЗАМЕНАЦИОННЫЕ ВОПРОСЫ	№ и наименование раздела
	Код	Определение		
1	2	3	4	5
1.	ПК-6	способность производить расчеты и проектирование отдельных блоков и устройств систем автоматизации и управления и выбирать стандартные средства автоматики, измерительной и вычислительной техники для проектирования систем автоматизации и управления в соответствии с техническим заданием	1.1 Первое поколение ЭВМ. 1.2 Второе поколение ЭВМ. 1.3 Третье поколение ЭВМ. 1.4 Четвертое поколение ЭВМ. 2.1 Определение микропроцессора. 2.2 История развития микропроцессоров. 2.3 Архитектура фон Неймана. 2.4 Принципы фон Неймана. 3.1 Классификация микропроцессоров по числу больших интегральных схем. 3.2 Классификация микропроцессоров по назначению. 3.3 Классификация микропроцессоров по виду обрабатываемых сигналов. 3.4 Классификация микропроцессоров по характеру временной организации. 3.5 Классификация микропроцессоров по организации структуры. 3.6 Классификация микропроцессоров по количеству выполняемых программ. 4.1 Арифметико-логический блок. 4.2 Основные характеристики микропроцессора. 4.3 Структура типового микропроцессора. Выполнение элементарных операций 5.1 Принцип управления. 5.2 Организация устройств управления. 5.3 Устройство управления микропроцессора. 5.4 Особенности программного и микропрограммного управления 6.1 Оптимальная система команд. 6.2 Формат команды. 6.3 Классификация команд. Примеры кодирования команд. 7.1 Режимы работы микропроцессора. Защищенный режим. Реальный режим. 7.2 Режим системного управления. Переключение между режимами. 8.1 Среда выполнения задачи. Архитектурные решения микропроцессоров. 9.1 Режимы адресации.	**1.** Введение. История развития микропроцессоров. **2.**Внутренняя структура микропроцессоров. Принципы фон Неймана. **3.** Классификация микропроцессоров. **4.** Внутренняя структура микропроцессоров. Арифметико-логический блок. **5.** Устройство управления. **6.** Система команд микропроцессора. **7.** Режимы работы микропроцессора. **8.** Принципы формирования адресного про-

		9.2 Способы адресации. 9.3 Режимы адресации операндов.Возможности микропроцессоров по адресации.	странства. **9**. Система адресации.
		10.1 Внутренняя память компьютера. 10.2 Технология производства интегральных схем. 10.3 Основные характеристики полупроводниковой памяти.	**10**. Память как функциональный узел.
		11.1 Организация физической памяти. ОЗУ, ПЗУ и FLASH – память. 11.2 Логическая структура основной памяти.	**11.** Принципы организации памяти.
		12.1 Модели памяти. 12.2 Понятие о сегментированной модели памяти. 12.3 Понятие о страничной модели памяти. 12.4 Сегментно-страничная модель памяти. 12.5 Плоская модель памяти.	**12.** Виртуальная память.
		13.1 Понятие прерывания. 13.2 Классификация прерываний. 13.3 Внешние прерывания. 13.4 Система прерываний. Аппаратные и программные средства системы прерываний. Обработка прерываний в реальном режиме.	**13.** Прерывания.
		14.1 Многозадачность – определение, история развития. 14.2 Режимы многозадачности. 14.3 Невытесняющая многозадачность. 14.4 Многозадачность в защищенном режиме.	**14.** Поддержка многозадачности.
		15.1 Методы и средства отладки. Отладка на примере процессора 486.	**15.** Программы-отладчики.
		16.1 Программирование микропроцессора. Машинно-ориентированные языки. 16.2 Языки высокого уровня. Специализированные языки.	**16.** Принципы программирования микропроцессоров.
		17.1 Основные группы операций. Мнемокоды команд процессора Pentium.	**17.** Команды языка Ассемблер

3. Описание показателей и критериев оценивания компетенций

Показатели	Оценка	Критерии
Знать **ПК-6:** – состав и средства микропроцессорных систем; – способы управления микропроцессорными системами **Уметь** **ПК-6:** - выбирать методы и средства управления микропроцессорными системами; - выбирать методы управления микропроцессорными системами **Владеть** **ПК-6:** -навыками программирования и управления микропроцессорными системами	**отлично**	Обучающийся должен во время ответа показать знания: микропроцессорных систем управления. Обучающийся должен иметь навыки владения: программирования микропроцессоров.
	хорошо	Ответ содержит неточности. Дополнительные вопросы требуется, но обучающийся с ними справляется отлично.
	удовлетворительно	Ответил только на один вопрос, либо слабо ответил на оба вопроса. На дополнительные вопросы отвечает неуверенно.
	неудовлетворительно	На оба вопроса обучающийся отвечает неубедительно. На дополнительные вопросы преподавателя также не может ответить.

4. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и опыта деятельности

Дисциплина Микроконтроллеры и микропроцессоры в системах управления направлена на изучение основ микропроцессорной техники, методов контроля, обработки, анализа теоретических и экспериментальных исследований в сфере профессиональной деятельности.

Изучение дисциплины предусматривает:

- лекции,
- лабораторные занятия,
- самостоятельную работу,
- экзамен

В ходе освоения раздела 1 «Введение. История развития микропроцессоров» обучающиеся должны историю развития.

В ходе освоения раздела 2 «Внутренняя структура микропроцессоров. Принципы фон Неймана» обучающиеся должны знать принципы построения ЭВМ .

В ходе освоения раздела 3 «Классификация микропроцессоров» обучающиеся должны знать основные виды микропроцессоров.

В ходе освоения раздела 4 «Внутренняя структура микропроцессоров. Арифметико-логический блок» обучающиеся должны знать основу структур микропроцессоров.

В ходе освоения раздела 5 «Устройство управления» обучающиеся должны изучить структуру и основные составляющие микропроцессоров.

В ходе освоения раздела 6 «Система команд микропроцессора» обучающиеся должны уметь использовать систему команд.

В ходе освоения раздела 7 «Режимы работы микропроцессора» обучающиеся должны уметь отличать режимы работы ЭВМ.

В ходе освоения раздела 8 «Принципы формирования адресного пространства» обучающиеся должны овладеть знаниями по формированию адресов микропроцессоров.

В ходе освоения раздела 9 «Система адресации» обучающиеся должны уметь использовать различные способы адресации.

В ходе освоения раздела 10 «Память как функциональный узел» обучающиеся должны изучить структуру памяти.

В ходе освоения раздела 11 «Принципы организации памяти» обучающиеся должны знать принципы организации памяти.

В ходе освоения раздела 12 «Виртуальная память» обучающиеся должны уметь использовать виртуальную память.

В ходе освоения раздела 13 «Прерывания» обучающиеся должны изучить систему прерывания.

В ходе освоения раздела 14 «Поддержка многозадачности» обучающиеся должны знать принцип многозадачности.

В ходе освоения раздела 15 «Программы-отладчики» обучающиеся должны уметь проводить отладку с помощью различных программных средств.

В ходе освоения раздела 16 «Принципы программирования микропроцессоров» обучающиеся должны уметь программировать микропроцессор.

В ходе освоения раздела 17 «Команды языка Ассемблер» обучающиеся должны уметь пользоваться командами.

В процессе выполнения практических занятий происходит закрепление знаний, формирование умений и навыков реализации представления о различных вероятностно-статистических методах исследования параметров.

Работа с литературой является важнейшим элементом в получении знаний по дисциплине. Прежде всего, необходимо воспользоваться списком рекомендуемой по данной дисциплине литературой. Дополнительные сведения по изучаемым темам можно найти в периодической печати и Интернете.

К экзамену допускаются студенты, которые выполнили все практические работы.

Оценка знаний, умений, навыков осуществляется в процессе промежуточной аттестации обучающихся по дисциплине, которая осуществляется в виде экзамена. Для оценивания знаний, умений, навыков используются ФОС по дисциплине, содержащий вопросы к экзамену.

АННОТАЦИЯ
рабочей программы дисциплины
Микроконтроллеры и микропроцессоры в системах управления

1. Цель и задачи дисциплины

Целью изучения дисциплины является: приобретение умений и навыков исследования проблем в своей предметной области, выбора методов и средств их решения, анализа результатов теоретических и экспериментальных исследований.

Задачей изучения дисциплины является: формирование способностей анализа результатов исследований, выбора методов и средств решения проблем в своей предметной области.

2. Структура дисциплины

2.1 Распределение трудоемкости по отдельным видам учебных занятий, включая самостоятельную работу: лекции – 36 ч., лабораторные занятия -18 ч., самостоятельная работа – 63 ч.

Общая трудоемкость дисциплины составляет 144 часа, 4 зачетные единицы.

2.2 Основные разделы дисциплины:

1- Введение. История развития микропроцессоров.
2- Внутренняя структура микропроцессоров. Принципы фон Неймана.
3- Классификация микропроцессоров.
4-Внутренняя структура микропроцессоров. Арифметико-логический блок.
5-Устройство управления.
6-Система команд микропроцессора.
7-Режимы работы микропроцессора.
8-Принципы формирования адресного пространства.
9-Система адресации.
10-Память как функциональный узел.
11-Принципы организации памяти.
12-Виртуальная память.
13-Прерывания.
14-Поддержка многозадачности.
15-Программы-отладчики.
16-Принципы программирования микропроцессоров.
17-Команды языка Ассемблер.

3. Планируемые результаты обучения (перечень компетенций)

Процесс изучения дисциплины направлен на формирование следующих компетенций:
-ПК-6 способность производить расчеты и проектирование отдельных блоков и устройств систем автоматизации и управления и выбирать стандартные средства автоматики, измерительной и вычислительной техники для проектирования систем автоматизации и управления в соответствии с техническим заданием.

4. Вид промежуточной аттестации: экзамен

Приложение 3

Протокол о дополнениях и изменениях в рабочей программе
на 20___-20___ учебный год

1. В рабочую программу по дисциплине вносятся следующие дополнения:
_____
_____

2. В рабочую программу по дисциплине вносятся следующие изменения:
_____
_____
_____

Протокол заседания кафедры №____ от «___» _____ 20 ____ г.,
(разработчик)

Заведующий кафедрой _____ _____
(подпись) *(Ф.И.О.)*