Классификация средств вычислительной техники и их сравнительный анализ. Области применения.

Средства вычислительной техники по представлению данных делятся на 3 группы:

1. Цифровые
2. Аналоговые
3. Гибридные

Цифровые средства вычислительной техники используют **дискретные данные**, **цифровые алгоритмы, цифровые элементы**. Аппроксимация непрерывного сигнала в дискретный сопровождается наличием ошибки квантования ($\Delta_{\text{квантования}}$). Эту ошибку можно уменьшить уменьшая время квантования ($\Delta_{\text{квантования}}$). Уменьшать ошибку квантования имеет смысл только до определённого значения (хотя теоретически мы можем уменьшать её очень много). С некоторого момента ошибка датчика (разница между реальным значением физической величины (температурой, давлением, влажностью) и показаниями датчика – зависит от элементной базы, наличия паразитных сопротивлений и ёмкостей и т.д.) становится выше ошибки квантования.

Вывод: цифровая техника – высокоточная, однако необходимо преобразовывать сложные алгоритмы к элементарным операциям. Во многих случаях точность результата зависит от времени вычисления $\sin(x) = x - \frac{x^2}{2!} + \frac{x^3}{3!} - \cdots$

Цифровая техника не требует настройки: правильная схема, собранная из правильных элементов обязана правильно работать. Все элементы взаимозаменяемые.

Аналоговые средства вычислительной техники используют **непрерывные сигналы, непрерывные алгоритмы, аналоговые элементы**. Сигнал (ток, напряжение) используется в качестве входного, выходного и промежуточного значения. Усложнение алгоритма не ведёт к замедлению работы схемы, но увеличивает количество элементов. Вычисления нельзя вести с точностью, превышающей технические ограничения (сопротивление резисторов) – зависит в том числе и от внешних условий (температура и т.д.). Существуют способы уменьшения этой ошибки (дифференциальный метод, настройка нуля), но полностью от неё избавиться невозможно.

Вывод: высокое быстродействие (сигнал на выходе равен времени прохождения самой длинной цепочки), но низкая точность, низкая ремонтопригодность, отсутствие алгоритмов проектирования, необходимость ручной настройки.

Гибридные средства вычислительной техники используют несколько аналоговых вычислительных машин (АВМ) и ЦВМ, соединённых **системой преобразователей**. Выделяют **аналоговую часть** (быстрое принятие решений с низкой точностью) и **цифровую часть** (получение точных результатов). Пример: наведение пушки – аналоговая часть определяет направление поворота (очень быстро), подаёт напряжение соответствующего знака на мотор, цифровая часть определяет угол поворота и регулирует напряжение, чтобы остановиться в нужный момент (Если использовать только ЦВМ – будет долго – движение начнётся только когда будет получен конечный угол поворота, хотя чтобы начать движение достаточно знать старший бит (знак) – направление поворота . Если использовать только АВМ – точность слишком низкая). Преобразователи – аналого-цифровой преобразователь (АЦП), преобразователь частоты в код (ПЧК), цифро аналоговый преобразователь (ЦАП).

Классификация по кругу решаемых задач:

1. Универсальные
2. Специализированные

Универсальные – должны уметь всё и желательно хорошо. Требования на уровне пожеланий, реальная система покупается исходя из бюджета. Интерфейс адаптирован для работы с человеком (монитор, мышь, клавиатура)

Специализированные – ориентированы на решение одной задачи или узкого класса задач. Требования жёсткие – низкая точность недопустима. Интерфейс ориентирован на работу с приборами, датчиками и исполнительными механизмами.

Области применения (напишите **любые** 4 из 6, из скобочек тоже не всё пишите):

1. Военное дело (системы навигации, системы радиолокационного опознавания и др.)
2. Промышленность (системы автоматизированного управления, автоматические системы планирования и управления производством, системы принятия решений и др.)
3. Научно-исследовательская деятельность (системы автоматического проектирования, моделирование физических явлений, системы принятия решений, обработка экспериментальных данных и др.)
4. Медицина (лазерная хирургия, искусственные органы, системы жизнеобеспечения и др.)
5. Социальная сфера (системы массового обслуживания, информационно-справочные системы и др.)
6. Связь (шифрование, модулирование и демодулирование сигнала)

Элементы и функциональные узлы ЭВМ.

1. Логические (без памяти) – выполняют функции алгебры логики и элементарные арифметические операции. Выходной сигнал есть только при наличии входного.

x_1	x_2	$x_1 \vee x_2$	$x_1 \& x_2$	$\overline{x_1}$	$x_1 \oplus x_2$	$x_1 \sim \vee x_2$	$x_1 \sim \& x_2$
		ИЛИ	И	НЕ	XOR	ИЛИ НЕ	И НЕ
0	0	0	0	1	0	1	1
0	1	1	0	1	1	0	1
1	0	1	0	0	1	0	1
1	1	1	1	0	0	0	0

2. Элементы с памятью (триггеры) – хранят 1 бит информации, обычно работают по фронту синхроимпульса с. **Фронт** импульса – переход из $0 \rightarrow 1$, **задний фронт** – переход $1 \rightarrow 0$. На выходе или сигнал Q, или два сигнала Q и $\bar{Q}$.

2.1 RS-триггер. R – Reset (установка 0), S – Set (установка 1).

S	R	Q	$\bar{Q}$
0	0	Сохранение состояния	
0	1	0	1
1	0	1	0
1	1	Запрещённое состояние	

2.2 JK-триггер. J – Jump (установка 1), K – Kill (установка 0).

J	K	Q
0	0	Сохранение состояния (Q)
0	1	0
1	0	1
1	1	Инверсия ($\bar{Q}$)

2.3 D-триггер – по фронту импульса передаёт значение со входа D (Data) на выход

D	Q	$\bar{Q}$
0	0	1
1	1	0

2.4 T-триггер – при T == 1 по фронту импульса меняет своё состояние на противоположное, при T == 0 – сохранение состояния.

T	Q	$\bar{Q}$
0	Q	$\bar{Q}$
1	$\bar{Q}$	Q

3. Узлы

3.1 Дешифратор (декодер) – преобразует n-рязрядный код в 2^nрязрядный сигнал. Например на входе десятичное число 2 (двоично число 10), тогда выходной сигнал будет иметь вид 0100 – единица во втором разряде. Для числа 5 (101) выходной сигнал будет 00100000. Используется для включения одного из 2^n устройств подачей кода нужного устройства на вход.

Входной код	n	2^n	Выходной сигнал
$2_{10} = 10_2$	2	4	0100
$5_{10} = 101_2$	3	8	0010 0000
$19_{10} = 10011_2$	5	32	0000 0000 0000 1000 0000 0000 0000 0000

3.2 Шифратор – устройство, выполняющее преобразование двоичного сигнала в код (операция, обратная дешифратору).

3.3 Мультиплексор – параллельная передача кода из нескольких направлений в одно. На вход подаётся k сигналов (разрядность всех сигналов равна $n, n \geq 1$) и адрес A (разрядность адреса m определяется следующим соотношением $2^m \geq k$). На выходе сигнал Q разрядности n, совпадающий с входным сигналом под номером A.

Пример: устройство имеет $k = 5$ входных сигналов данных $k_0 - k_4$ разрядностью 3 ($n = 3$), следовательно, разрядность адреса должна быть $m \geq \log_2 5, m \geq 2.3219, m = 3$.

Входные сигналы данных:					Адрес:	Выход:
k_0	k_1	k_2	k_3	k_4	A	Q
001	101	110	000	111	$010_2 = 2_{10}$	110
001	101	110	000	111	$100_2 = 4_{10}$	111

3.4 Демультиплексор – выполняет операцию, противоположную мультиплексору. Входные сигналы: Q – вход данных, A – вход адреса. Выходные сигналы - k_i, причём на выходе под номером A будет значение Q, на остальных – нули.

3.5 Арифметико-логическое устройство – выполняет одну из арифметико-логических операций над входными сигналами и передаёт результат на выход. Входные сигналы данных (обычно два): k_i, входной сигнал кода операции cop. Выходной сигнал Q – результат операции. Простейшее АЛУ строится на базе мультиплексора, на вход которого подаются результаты всех операций над входным сигналом, а код операции выступает в роли селектора результата.

3.6 Сдвигатель – выполняет сдвиг входного сигнала. Различают следующие виды сдвигов:

3.6.1 По направлению: влево и вправо.

3.6.2 По степени сдвига: на один разряд и на N разрядов.

3.6.3 По способу сдвига: арифметический (сохранение знака – сдвигаются все биты, кроме самого старшего) и логическое (сдвигаются все биты).

3.6.4 По значению: циклический (при сдвиге влево старший байт переходит в младший, при сдвиге вправо – младший в старший) и простой (при сдвиге влево младший бит заполняется нулём, при сдвиге вправо старший заполняется нулём)

4. Узлы с памятью (счетчики) – устройства с логикой, построенные на базе триггеров.

4.1 Сумматор – выполняет накопление. В качестве входов обычно имеет синхросигнал, так же может иметь: сигнал разрешения счёта, шину для загрузки начала счёта (и/или модуля счёта), сигнал сброса. Выходы: текущее значение, сигнал при переполнении (обнулении).

4.2 Вычитающий счётчик – аналогичен сумматору, но счёт происходит в обратном направлении.

4.3 Реверсивный счётчик – аналогичен сумматору, но присутствует дополнительный вход: направление счета.

Выполнение арифметических операций с фиксированной и плавающей точкой в ЭВМ.

Прямой код – просто двоичное представление числа $5_{10} = 101_2$

Дополнительный код – представление числа для арифметических операций. Запись положительного числа в дополнительном коде совпадает с записью числа в прямом коде $+5_{10} = 0101_2$(ноль вначале подставлен для удобства, чтобы подчеркнуть что число положительное). Запись отрицательного числа в дополнительном коде получается по следующему алгоритму:

1. Записать модуль числа в прямом коде $-5.625_{10} \rightarrow 101.101_2$
2. Инвертировать разряды слева направо до самой правой единицы (её и все что правее не трогать) $101.101_2 \rightarrow 010.011_2$ (другой пример $1010110101000_2 \rightarrow 0101001011000_2$)
3. Дописать к полученному результату единицу слева $010.011_2 \rightarrow 1010.011_2$ – запись числа в дополнительном коде (справа всегда единица).

Обратное преобразование:

1. Выбросить единицу справа $1010.011_2 \rightarrow 010.011_2$
2. Выполнить пункт 2 прямого преобразования $010.011_2 \rightarrow 101.101_2$

Вычитание выполняется как сложение чисел: $A - B \rightarrow A + (-B)$.

1. Сложение чисел с фиксированной точкой: -5.625 + 5.5 = -0.125

	знак в ДК	Целая часть			Дробная часть			
+	1	0	1	0	0	1	1	ДК
	0	1	0	1	1	0	0	ДК
	1	1	1	1	1	1	1	ДК
	-	0	0	0	0	0	1	ПК

2. Умножение чисел с фиксированной точкой: 1.25 * (-2.5) = -3.125

Модуль результата равен произведению модулей аргументов: 1.25 * 2.5 = 3.125

*		0	1	0	1		
				1	0	1	0
+				0	0	0	0
			0	1	0	1	
		0	0	0	0		
	0	1	0	1			
	0	1	1	0	0	1	0

Знак результата – сумма по модулю два знаков аргументов $1 \oplus 0 = 1$

3. Деление чисел с фиксированной точкой –аналогично умножению, модуль и знак вычисляются отдельно: $\frac{4.375}{1.25} = 3.5$

-	1	0	0	0	1	1	1	0	1	
		1	0	1			0	1	1	1
	-	0	1	1	1					
			1	0	1					

		-	0	1	0	1				
				1	0	1				
						0				

Числа с плавающей точкой представляются в виде: $x = S_x M_x 2^{P_x}$, где S_x – знак числа, M_x- мантисса числа, P_x – порядок числа (положение точки).

4. Сложение чисел с плавающей точкой

$$z = x + y$$

1) Выравнивание порядков $P_x - P_y = \Delta P$ (пусть $P_x > P_y$): $P_y = P_y + \Delta P$
2) Выравнивание мантисс $M_y = M_y * 2^{-\Delta P}$
3) Сложение (с учётом знаков), формирование знака результата
4) Возможно нарушение нормализации t при одинаковых знаках на один разряд ($0 \le t \le 1$) влево, при разных знаках на n разрядов $0 \le t \le n$ вправо.

5. Умножение чисел с плавающей точкой

$$z = x * y$$

1) Знак результата $S_z = S_x \oplus S_y$
2) Мантисса $M_z = M_x * M_z$. Возможно нарушение нормализации на один разряд вправо.
3) Порядок $P_z = M_x + M_z + t$

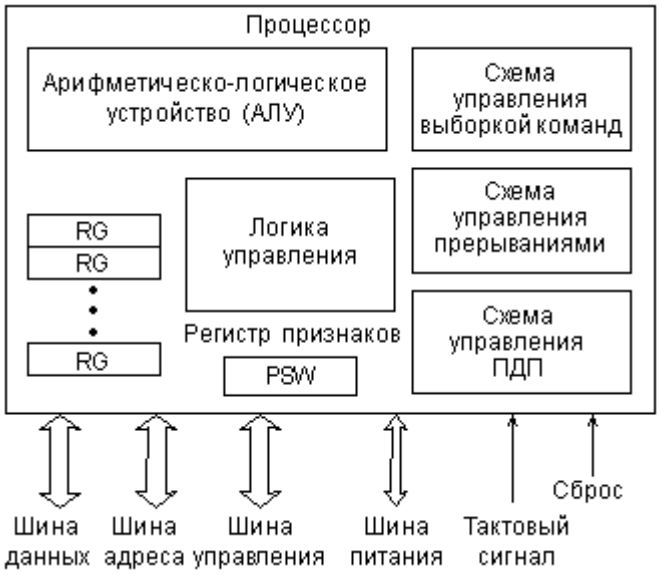
6. Деление чисел с плавающей точкой

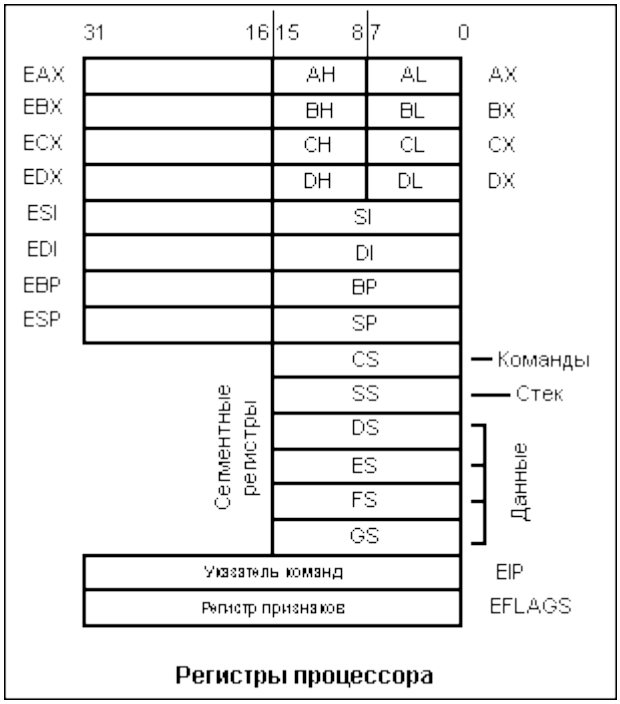
$$z = \frac{x}{y}$$

1) Знак результата $S_z = S_x \oplus S_y$
2) Мантисса $M_z = M_x / M_z$. Возможно нарушение нормализации на один разряд влево.
3) Порядок $P_z = M_x - M_z + t$

Структура процессора

http://arxitektura-pk.26320-004georg.edusite.ru/p31aa1.html





Регистры процессора

Схема управления и выбора команд – выполняет чтение команд из памяти и их дешифрацию. Выполняется параллельно с выполнением текущей команды в АЛУ. Для считанных команд используется конвейер типа FIFO.

АЛУ – выполняет команды, переданные процессору. Операция, аргументы, местоположение результата определяются командой. Если команда заключается в пересылке данных – АЛУ не используется. К операциям относятся логические («ИЛИ», «И», «Сумма по модулю 2») и арифметические (+, -, *, /).

Схема управления прерываниями – обрабатывает поступающий на процессор запрос прерывания. Определяет адрес начала программы обработки прерывания, обеспечивает выполнение этой программы после завершения текущей команды и сохранения текущего состояния регистров. После выполнения обработки прерывания возвращается к прерванной программе с восстановлением памяти.

Схема управления прямым доступом к памяти – служит для временного отключения процессора от шин и приостановления работы процессора на время предоставления доступа запросившему его устройства.

Логика управления - организует взаимодействие всех узлов процессора, синхронизирует работу процессора с внешними сигналами, реализует ввод и вывод информации.

Регистр признаков – содержит информацию о результате выполнения команды (флаг переноса, флаг нуля, флаг переполнения и т.д.)

Регистры процессора – ячейки быстрой памяти для хранения кодов данных и адреса. Регистры могут быть использованы под ЛЮБЫЕ нужны – назначение регистра определяется командой. Существуют назначения по умолчанию (их можно не указывать в команде): регистры общего назначения (EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP) – 32 разряда, можно обращаться к 16 разрядам

(AH, BX и т.д.) или к 8 разрядам (AL, AH и т.д.); сегментные регистры (CS – команды, SS – стека; DS, ES, FS, GS – данных), регистр указателя команд, регистр признаков.

Указатель счетчика команд изменяется следующим образом: при запуске системы в него заносится значение адреса программы начального запуска. Во время работы по мере выполнения команд значение счетчика автоматически инкрементируется (на единицу или на два). При выполнении команды перехода в указатель счетчика команд принудительно записывается адрес памяти. При вызове подпрограммы происходит подобная операция (старое значение счетчика сохраняется вместе с другими регистрами, чтобы по завершению подпрограммы можно было продолжить выполнение с момента прерывания).

Управляющие автоматы. Сравнительная характеристика. Принципы функционирования.

Управляющий автомат – конечный автомат, выполняющий управляющие функции над данными.

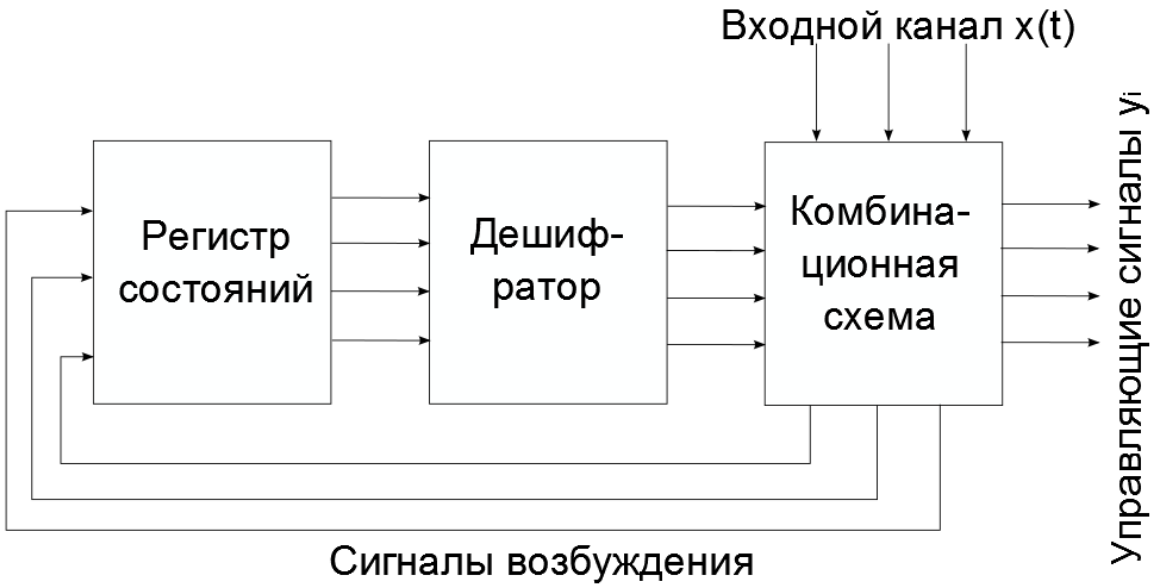
1. Управляющие автоматы с жёсткой логикой

+ компактны, быстродействие максимально

- не универсальны, рассчитаны только на свою систему микрокоманд. Изменения в системе микрокоманд приведут к необходимости изменения схемы.

Автомат МИЛИ $\varphi(a_i x_i)$

Автомат МУРА $\varphi(a_i)$ – проще и быстрее.



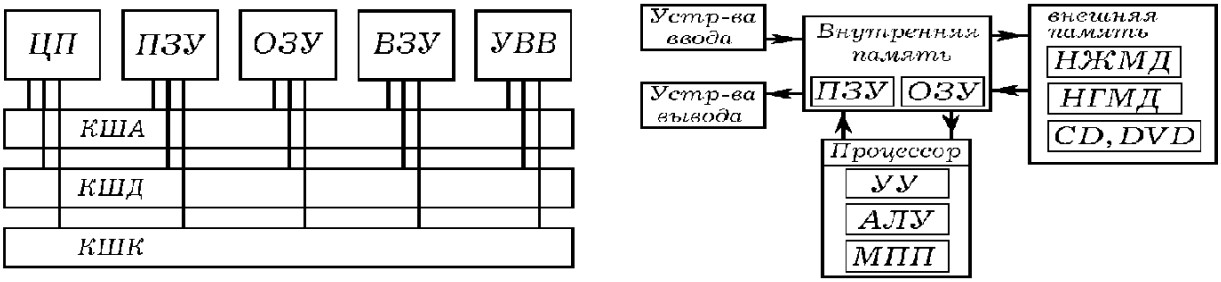
2. Управляющие автоматы с программируемой логикой

+ универсальны, перепрограммируемы

- медленная работа, часто избыточны

Код операции поступает на **преобразователь начального адреса**. По коду операции формируется начальный адрес микропрограммы, выполняющей эту операцию. **Блок формирования адреса** осуществляет последовательную выборку микрокоманд, нарушение которой может происходить в условных вершинах по сигналу и адресу. **Регистр микрокоманды** состоит из операционной и адресной части. **Операционная часть** - дешифратор микрокоманд и блок формирования управляющих сигналов, **адресная часть** – адрес (изменение естественного хода выборки команд).

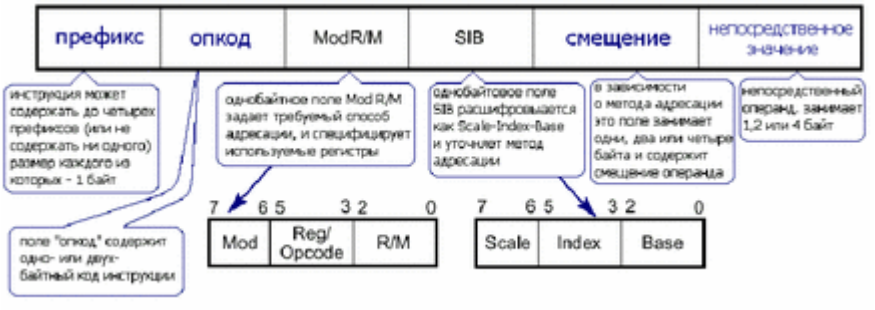
Типовая структура ЭВМ. Характеристики процессоров и структура команд.



ЦП – центральный процессор, ПЗУ – постоянно запоминающее устройства, ОЗУ – оперативное запоминающее устройство, ВЗУ – внешние запоминающие устройства, УВВ – устройства ввода-вывода, КША – кодовая шина адреса, КШД – кодовая шина данных, КШК – кодовая шина команд, НЖМД – накопитель на жёстких магнитных дисках, НГМД - накопитель на гибких магнитных дисках, УУ – устройство управление, АЛУ – арифметико-логическое устройство, МПП – микропроцессорная память.

Характеристики процессоров:

1. Разрядность шины адреса: 16, 32, 64 – определяет максимальное адресное пространство (обычно в байтах, но теоретически – минимальных адресуемых единиц) $2^{16} \rightarrow 64$ кБ, $2^{32} \rightarrow 4$ГБ, $2^{64} \rightarrow 16$ЭБ
2. Тактовая частота – определяет количество выполняемых операций в секунду. Измеряется в герцах (гигогерцах).
3. Размер кэша – размер быстрой памяти на кристалле процессора. Различают различные уровни кэша (чем меньше цифра, тем ближе к процессору – тем быстрее). Измеряется в байтах (мегабайтах).
4. Технический процесс – размер элементов на кристалле. Чем меньше элемент – тем больше их можно разместить на кристалле, так же маленькие элементы потребляют меньше энергии (и соответственно меньше греются). Изменяется в метрах (нанометрах).
5. Сокет – требуется для определения совместимости процессора с материнской платой. Не измеряется.



http://citforum.ru/book/cook/intel.shtml

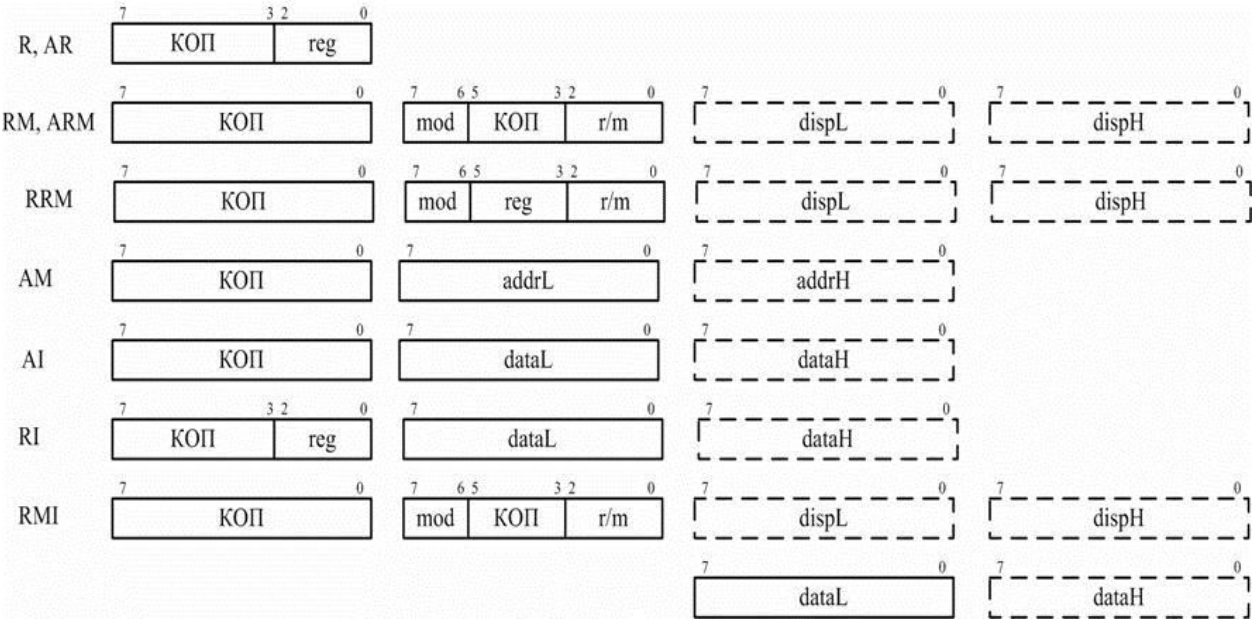
Префиксы: блокировки и повторения, переопределения сегмента, переопределения размеров операнда, переопределения размера адреса.

Опкод – код инструкции (ADD, XOR, SUB и т.д.) и несколько бит параметров инструкции (размер операнда, порядок операндов и т.д.)

Форматы команд и способы адресации. Основные этапы выполнения команд.

Формат команд достаточно разнообразен и сильно зависит от типов аргументов.

Форматы команд (достаточно написать 2-3 штуки и соответствующие обозначения под картинкой):



Обозначения полей команд:
КОп - код операции;
reg - адрес регистра (данных или указателей);
mod - режим адресации;
r/m - при mod ? 11 - указатель регистров, участвующих в формировании адреса, при mod=11 - адрес регистра;
dispL, dispH - младший и старший байты смещения;
addrL, addrH - младший и старший байты адреса;
dataL, dataH - младший и старший байты непосредственного данного.
Операции, задаваемые различными типами команд:
R – операция над регистром reg;
AR – операция над регистром reg и аккумулятором AX;
RM – операция над регистром r/m или словом памяти;
RRM – операция над регистром reg и регистром r/m или словом памяти;
ARM – операция над AX и регистром r/m или словом памяти;
AM – операция над AX и словом памяти [addr];
AI – операция над AX и непосредственным одно- или двухбайтовым операндом;
RI – операция над регистром reg и непосредственным операндом;
RMI – операция над регистром r/m или словом памяти и непосредственным операндом.

Способы адресации (не переписывайте примеры вслепую! Поставьте свои чиселки, так же вместо 12H можно писать 0x12)

1. Регистровая – операнды находятся в регистрах, указанных полями reg и r/m
2. Непосредственная – в команде определяется непосредственный операнд (константа).

Пример. MOV DX, 0DH.

3. Абсолютная – команда содержит физический адрес памяти.
4. Косвенно-регистровая – в регистре содержатся не данные, а EA операнда.

Пример. MOV DX,[DI].

5. Базовая адресация - предполагается, что база содержит начальный адрес структуры данных, а смещение определяет элемент этой структуры.

Пример: MOV DX, [BP]10H.

6. Индексная адресация - смещение в команде определяет фиксированный начальный адрес структуры. Содержимое индексного регистра определяет элемент этой структуры.

Пример: MOV DX, 10H[SI].

7. Базово-индексная адресация - удобна для обращения к сложной структуре данных, например к матрице. Пример. MOV DX, 10H [BP] [DI].

8. Относительная адресация - применяется для команд переходов и циклов.

Этапы выполнения команд:

В регистре IP записан начальный адрес программы в ОП. Микропроцессор обращается к ОП и осуществляет выборку команды по адресу из IP. Затем загружает ее в регистр команд RGK и дешифрирует ее с одновременным инкрементом IP для адресации следующей по порядку команды. Если команда не является командой перехода, то выполняем ее и переходим к следующей по порядку команде. Команды выполняются последовательно до тех пор, пока не встретится команда перехода. Если встречается команда безусловного перехода, то изменяется естественный порядок следования команд путем замещения содержимого IP адресом, определяемым самой командой перехода. Команды условных переходов замещают или не замещают содержимое IP в зависимости от результатов предыдущих команд, т.е. от состояния регистра FLAG. Если, например, после команды «вычитания» находится команда «перехода по нулю», переход осуществляется, если в регистре FLAG флажок ZF=1. Если же ZF=0, то переход не производится. Когда реализован переход, начинается новая последовательность команд с адреса, к которому осуществлен переход.

16-битные и 32-битные микропроцессоры. Программная модель и форматы команд.

Программная модель intel 8086 включает в себя следующие группы регистров: сегментные регистры (CS, DS, SS, ES), указатель команд IP, регистр связи, регистры данных (AX, BX, CX, DX – можно получить доступ к старшему (high) или младшему (low) байтам – AH, BH, CL, DL и т.д.), указатели (SP, BP, SI, DI), регистр флагов F.

Вся область памяти делится а сегменты: CS – код сегмента, DS – сегмент данных, SS – сегмент стека, ES – дополнительный сегмент. На шину адреса поступает адрес, получаемых по следующей формуле: $\text{ША} = CS * 2^4 + IP$. Таким образом при разрядности шины данных 16 бит, разрядность шины адреса – 20 бит.

Форматы команд (http://frolov-lib.ru/books/bsp/v33/ch10_4.htm)

1. Обращение к памяти – КОП1 и КОП2 определяют команду и её вид; MOD, R/M и смещения – положение аргумента (аргументов). Например, MOD = 000, R/M = 00, означают, что в качестве аргументов будут использованы BX и SI.

1 байт		1 байт			1 байт	1 байт
11011	КОП1	MOD	КОП2	R/M	Смещение 1	Смещение 2

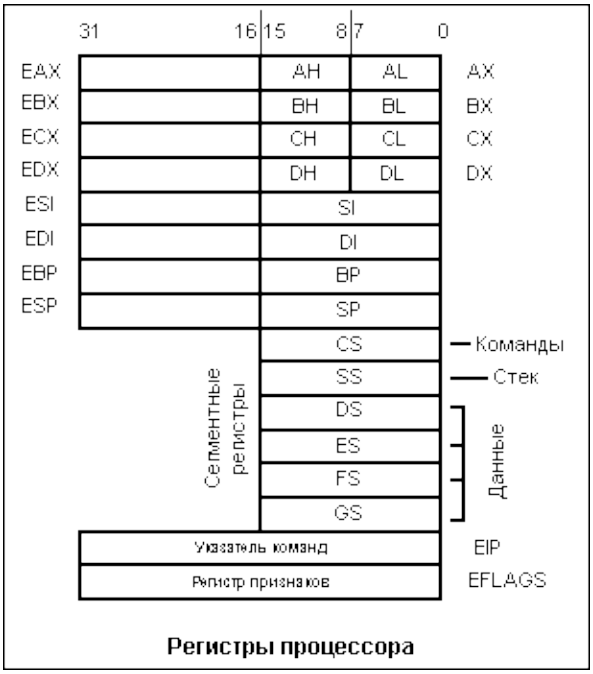
2. Обращение к регистру

1 байт		1 байт		
11011	КОП1	11	КОП2	STi

3. Команда без явного указания операндов

1 байт		1 байт	
11011	КОП1	11	КОП2

Программная модель intel 8086 ➔



Регистры процессора

Формат операций: