Тъй като различните машинни алгоритми включват и еднакви микрооперации (нулиране на флагове, преместване в регистри и пр.), това позволява да се използва една и съща микроинструкция от постоянната памет за управление на такава микрооперация. Следователно, в микропрограмната памет на известни адреси са записани по една микроинструкция за всички поддържани микрооперации, а задача на БФА е да "комплектова" текущата микропрограма чрез задаване на необходимата последователност от адреси. Това позволява лесна модификация на управлението чрез подмяна на записаната в постоянната памет информация.

Глава трета
Организация на процесори

3.1. Принципна организация на процесори

3.1.1. Структура и действие

Процесорът е основното компютърно устройство, в което се извършва непосредствената обработка на информацията на базата на съхранена в оперативната памет (ОП) програма. Отделните инструкции от програмата се извличат от паметта и се изпращат в процесора за изпълнение. Изпълнението на текуща инструкция се осъществява като последователност от елементарни операции (микрооперации), реализирани съгласно заложения метод за управление. Инструкциите са свързани с обработката на данни, които също се намират в ОП. За да се извлекат тези данни и да се насочат към входовете на обработващото устройство (АЛУ) е необходимо да са известни адресите, на които са записани. За целта в инструкциите се отделя поле за информация, подпомагаща изчисляването на реалния адрес на операндите в ОП. Посочените особености при организация и реализация на компютърната обработка определят и базовата структура на процесора. Тъй като в един универсален компютър с традиционна организация обработката се извършва от един основен процесор, той се нарича *централен процесор*. Това означава, че в състава на КС може да има и други процесорни устройства, които обаче са със специализирани функции (например, за управление на входно-изходните операции).

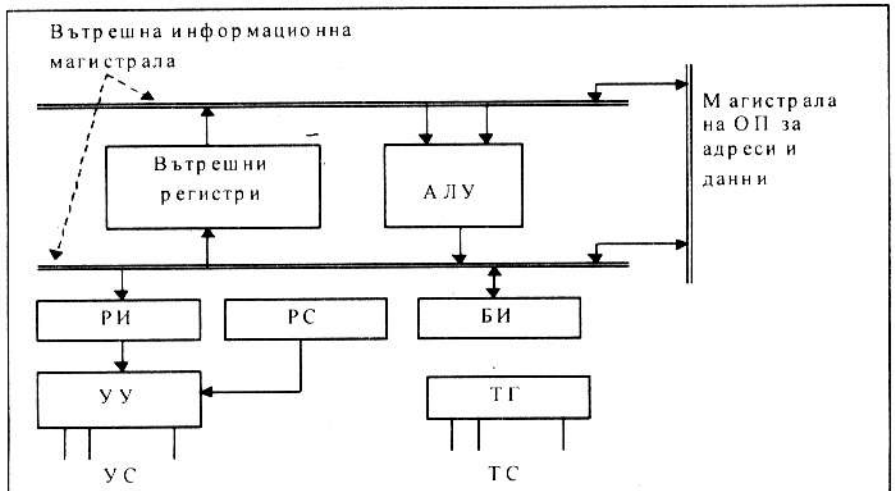
Структурата на типична процесорна конфигурация е показана на фиг.3.1 и включва:

- АЛУ за непосредствена обработка на операндите в зависимост от подаваните УС;
- УУ за реализация на инструкциите чрез определяне на заявената операция и генериране на точно определени УС в зависимост от признаци на състоянието (флагове);
- Блок вътрешни регистри, които се използват за поддържане на изчисленията с операнди, междинни резултати и др. Имат информационна връзка с АЛУ по вътрешната магистрала, която е свързана с магистралата на ОП за обмен на данни и адреси;

- Регистри на управлението, чрез които се реализира програмата от ОП: регистър на инструкцията (РИ), приемащ текуща инструкция за изпълнение; брояч на инструкциите (БИ), използван за адресиране на следваща инструкция от програмата в ОП; регистър на състоянието (РС), съхраняващ текущите ПС (флагове), формирани при изпълнението на всяка операция.

- Тактов генератор (ТГ), формиращ синхронизиращи тактови сигнали (ТС), необходими за реализация на синхронно управление на операциите в процесора.

Процесорът се свързва с ОП чрез съответна магистрала, която обединява определени линии за предаване на данни, адреси и за управление (старт, стоп, прекъсване и пр.).



Фиг. 3.1. Базова структура на процесор

Като основно устройство за автоматично изпълнение на машинните инструкции за обработка на данните, процесорът има следните *функции*:

- да формира адреси на инструкциите в ОП, да ги извлича и изпълнява чрез своите управляващи компоненти;

- да изчислява адреси на операнди в ОП, да ги прехвърля във вътрешните регистри и да организира обработката им чрез своите изчислителни ресурси;

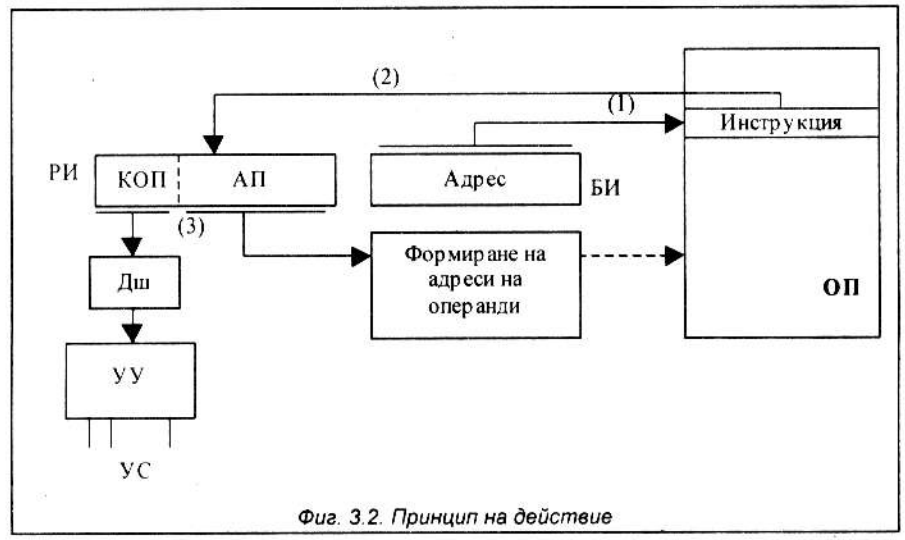
- да поддържа непрекъснато информация за състоянието на изчислението чрез формиране на признаци на състоянието и при необходимост да ги съхрани в паметта при евентуално прекъсване на обработката;

- да инициализира изпълнението на входно-изходните операции, като предаде управлението на съответния процесор или драйверна програма;

- да поддържа система за прекъсване, като приема и обработва заявки за прекъсване на текущата програма с осигуряване на възможност за нейното възстановяване;

- да контролира обменяната и обработваната информация, като обработва възникващите грешки за евентуалното им коригиране;

- да управлява диалога с потребителя чрез интерпретиране на системни команди от състава на операционната система (ОС).



Фиг. 3.2. Принцип на действие

Действието на централния процесор е свързано с принципа на програмното управлението и може да се представи по следния начин (фиг.3.2.):

1. В БИ се съдържа адрес, по който се определя разположението на текущата инструкция в ОП. Обикновено инструкциите от програмата са разположени в обособена област, наречена памет за програми. Адресирането се извършва чрез автоматично увеличаване на съдържанието на БИ с дължината (в байтове) на текущата инструкция. Така се осигурява извличане на последователните инструкции от програмата. Ако последователността на адресиране трябва да се наруши (например, при безусловен или условен преход), то текущата инструкция съдържа необходимата информация за това.

2. Адресираната инструкция се извлича от ОП и се съхранява в РИ за следваща обработка. Обикновено инструкциите съдържат задължително

поле за код на операцията (КОП) и евентуално адресно поле (АП). Последното участва при определяне разположението на операндите в ОП.

3. Текущата инструкция се обработва, като КОП се дешифрира (Дш) и се определя коя е заявената за изпълнение операция. В зависимост от това УУ генерира последователните УС за реализация на съставящите я микрооперации. В същото време на базата на АП се формират адресите на операндите. Последното зависи от метода за адресиране, заложен в самата инструкция.

4. Операндите се прочитат от ОП и се изпращат към входовете на АЛУ за обработка.

5. На базата на УС се изпълнява операцията в АЛУ и след приключване се съхранява готовият резултат (във вътрешен регистър или в ОП).

Структурата и организацията на централни процесори (ЦП) за различни компютри от различни, дори и от еднакви класове, силно се различават. Това се потвърждава от различните микропроцесорни фамилии, създавани и предлагани като ЦП за компютри от водещите фирми Intel Corp., Advance Micro Devices (AMD), Cyrix Corp. и др. Микропроцесорът е процесор с универсални функции, изграден от една или няколко големи интегрални схеми (ГИС). Евентуалната му специализация може да се постигне чрез програмиране на неговите функции. Възможни са различни структурни реализации на процесори, които се характеризират със своя вътрешен състав (процесор с акумулатор, със стек, с общи регистри и пр.).

Освен универсалните ЦП се разработват и специализирани процесори, които се включват като допълнителни или периферни устройства – канален процесор, матричен процесор, комуникационен процесор и пр.

3.1.2. Система от инструкции и адресиране

Инструкцията е основната управляваща единица, чрез която се реализира принципът на програмното управление. Тя съдържа необходимата информация за реализация на една операция от компютърното изчисление, заявено в дадена програма. Тази информация формира две основни полета – за код на операцията (КОП) и адресно поле (АП), които се третират по различен начин:

- КОП определя изпълняваната операция и съответната на нея микропрограма (последователност от микроинструкции) при микропрограмното управление;
- АП се използва при адресиране на операндите за текущата операция.

По *предназначение* инструкциите се разделят на:

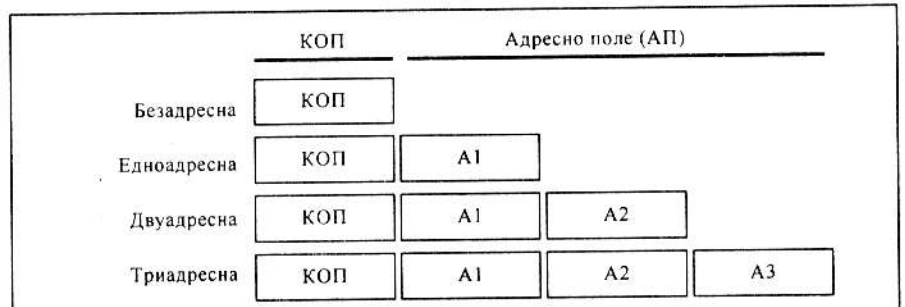
а) основни – за аритметична и логическа обработка на данни;

б) за обмен (между регистри и клетки от ОП);

в) за преход – управляват адресиране на инструкциите в програмите при безусловен, условен или преход към подпрограма;

г) за входно-изходни операции.

В зависимост от *структурата* и най-вече съдържанието на АП, инструкциите се разделят на безадресни, едноадресни, двуадресни и триадресни инструкции (фиг.3.3.).



Фиг. 3.3. Основни структури на инструкциите

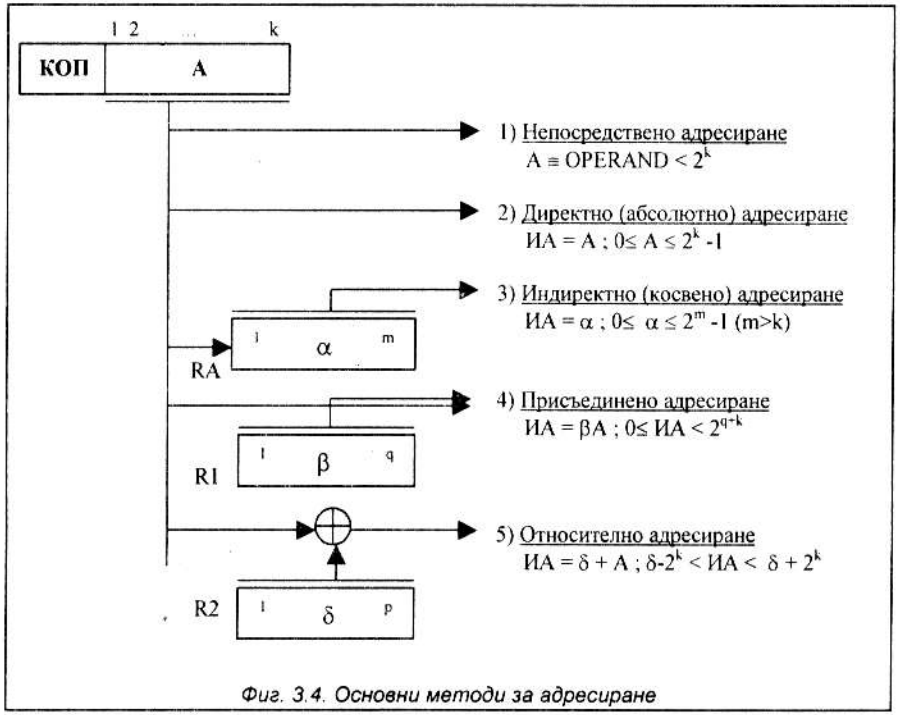
Безадресните инструкции се използват за задаване на операции, при които е известен източникът или приемникът на информацията и неговото адресиране е излишно. Такава операция е нулиране на служебен регистър или установяване на даден флаг в избрано състояние.

Най-близка до традиционната операция е триадресната инструкция, при която могат да се зададат адресите на двата операнда и адресът, на който ще се съхрани резултата, т.е. (A1) + (A2) → A3. Този тип инструкции, обаче, са най-дълги и изискват повече байтове за съхраняването им, поради което не са предпочитани. За икономия на памет при съхраняване на програмата най-често се използват двуадресните и едноадресните. Отсъствието на информация за адресирането при тях се компенсира чрез допълнителни похвати. При двуадресната инструкция е прието резултатът да се записва на мястото на първия операнд: (A1) + (A2) → A1. Едноадресните инструкции обикновено се използват при процесори, в които има служебен вътрешен регистър, наречен акумулатор (АК). В него винаги се съхранява първият операнд и се записва разултатът след изпълнение на операцията: (АК) + (A1) → АК.

.Независимо от структурата на инструкцията, при наличие на АП се преминава към неговата обработка, която позволява да се изчисли изпълнителния (физическия) адрес на операнда в ОП. Това се извършва за всяка адресна част от АП на базата на съответен метод за адресиране.

Адресирането в паметта се извършва на базата на няколко основни метода (фиг.3.4), които могат да се комбинират помежду си. В процесорите

се поддържа достатъчно голямо разнообразие от начини за адресиране, които се задават от специално поле, допълващо КОП.



Фиг. 3.4. Основни методи за адресиране

Адресирането е свързано с начина на интерпретиране на информация А в адресното поле. На фиг.3.4 са представени следните методи:

1. *Непосредствено адресиране*, при което в инструкцията се съдържа самият операнд. Този метод е най-бърз, но ограничава максималната стойност на операнда до 2^k-1.

2. *Директното адресиране* третира А като самия изпълнителен адрес (ИА), което позволява директно обръщане към паметта. И тук недостатък е ограничеността на адресното поле, чрез което могат да се задават не повече от 2^k адреса. Ето защо директното адресиране се прилага най-често при обръщане към регистровата памет (вътрешните регистри), която по правило е с ограничен обем, но с максимално бързодействие. Тогава, при *регистровото адресиране* дължината от k бита на адресното поле ще е достатъчна за задаване на адреса А на един от регистрите.

3. При *индиректното адресиране* се използва посредник RA (служебен регистър или клетка от паметта), чието съдържание α се третира като ИА. Информацията А от адресното поле се използва за адресиране на

посредника. Така може значително да се увеличи размера на адресното пространство ($2^m>2^k$), но нараства и времето за адресиране (поради междинното адресиране).

4. *Присъединеното адресиране* позволява формиране на ИА от две части: старшата част β се взема от служебен регистър R1 (например, регистър за начален адрес на страница в паметта), а младшата – от текущата инструкция. Така ИА ще се комплектова от q разряда на β и k разряда на А, което ще позволи адресиране в диапазона [0, 2^{q+k}-1].

5. При *относителното адресиране* ИА се изчислява чрез сумиране на стойността А от адресното поле на инструкцията със съдържанието δ на служебен регистър R2. Използва се при предаване на управлението (преход) в програмата, както и при т.нар. *адресиране по база*, където δ се нарича базов адрес и се съхранява в сегментен регистър R2 (съхранява начален адрес на сегмент от паметта). За това, обаче, в поле на инструкцията трябва да се укаже този сегментен регистър. Такава адресация се нарича още *адресиране с отместване (offset)* и при нея ИА=$\delta\pm2^k$.

В табл.3.1. са дадени примери за типични начини за адресиране, илюстрирани чрез примерна двуадресна инструкция за сумиране (мнемоничен запис) – в колоната 'изпълнение' чрез скобите се указва съдържанието на посочената клетка или регистър.

Таблица 3.1.

Адресиране	*Инструкция*	*Изпълнение*
1) Непосредствено	ADD R1,#450	(R1)+450→R1
2) Директно (абсолютно)	ADD R1,(450)	(R1)+(mem[450])→R1
3) Директно (регистрово)	ADD R1,R2	(R1)+(R2)→R1
4) Индиректно (чрез регистър)	ADD R1,(R2)	(R1)+(mem[(R2)])→R1
5) Индиректно (чрез паметта)	ADD R1,@(R2)	(R1)+(mem(mem[(R2)]))→R1
6) С отместване	ADD R1,100(R2)	(R1)+(mem[100+(R2)])→R1

В различните процесори са заложени различни методи за адресиране. Изборът на определено множество зависи и от организацията на паметта. Двата традиционни начина на организация са следните:

а) *странична организация* – паметта е условно разделена на равни по размер (брой адреси) области, наречени *страници*; обикновено размерът е четен брой Кбайта (например, 2KB, 4KB и пр.) и зависи от общия размер на ОП;

б) *сегментна организация*, при която отделните области, наречени *сегменти*, могат да са с различен размер; обикновено в процесора са предвидени няколко сегментни регистъра, чрез които се адресира в ралични

сегменти на ОП, а максималният адрес на даден сегмент зависи от размера на съответния сегментен регистър.

3.2. Организация на CISC процесори

3.2.1. Особености на CISC-технологията

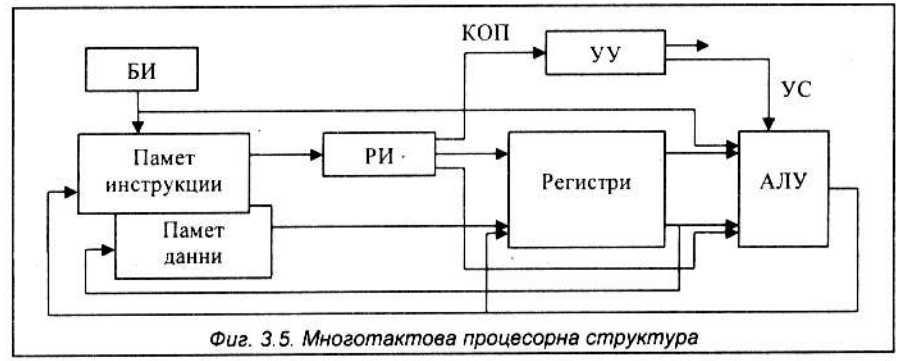
Технологията CISC (Complex Instruction Set Computer) е свързана с традиционните процесори, при които се поддържат множество инструкции, изпълнявани за различно време. Това време е в зависимост от типа на инструкцията, дължината, метода за адресиране на операндите и пр. В някои процесори изчисляването на резултата става на части поради разликата между дължината на реалните операнди и размера на входовете на АЛУ. По принцип изпълнението на една инструкция преминава през следните стъпки:

1. Извличане на инструкция от паметта.
2. Декодиране на инструкцията.
3. Формиране на адреса на операнда.
4. Извличане на операнда от паметта.
5. Изпълнение на операцията, кодирана чрез КОП в инструкцията.
6. Съхраняване на резултата.

Като правило в традиционният CISC-процесор инструкцията се реализира като група от няколко байта в паметта, като управлението се извършва от микропрограмно устройство. Последното извлича конкретна микропрограма, съответстваща на изпълняваната инструкция, след което реализира последователност от действия - елементарни операции, определяни от включените в микропрограмата микроинструкции. Тази последователност се нарича "вътрешен цикъл". Така всяка инструкция се изпълнява за различен брой стъпки (тактове) в зависимост от броя на микроинструкциите в тази микропрограма. В съвременните CISC-процесори микропрограмната памет може да достигне до големи обеми. От времето на първия компютър с микропрограмна реализация Atlas микропрограмните УУ значително са еволюирали и са достигнали съвършенство, но принципът на действие на микропрограмния процесор се отразява неизбежно върху неговата архитектура.

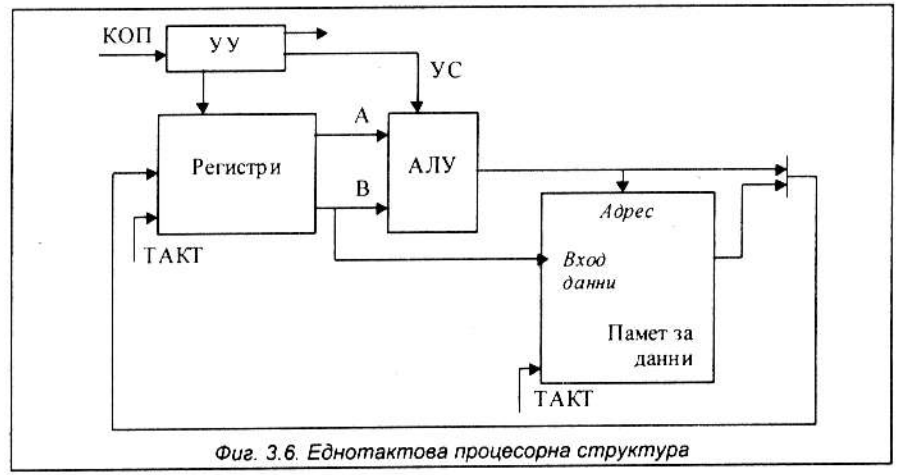
Примерна обобщена структура на CISC процесор с *многостъпково изпълнение* на инструкциите е показана на фиг.3.5. Обикновено инструкциите се съхраняват отделно от данните в област, условно наречена памет за инструкции (програми). Адресирането им става чрез брояча на инструкциите (БИ). Отделните инструкции се изпълняват за различен брой тактове, като в рамките на един такт се изпълява отделна стъпка.

Дължината на такта се определя от най-дългата по изпълнение стъпка. При организация на работата от съществено значение е общото управление. Необходимо е да се разграничават инструкциите от данните (при оперирane), както и достъпът до паметта (запис или четене на съответната информация). За целта, освен УС към АЛУ, се формират допълнителни управляващи сигнали, които имат служебно предназначение. Те управляват посоката по шините, начина на мултиплексиране при въвеждане и пр.



Фиг. 3.5. Многотактова процесорна структура

Друга възможна организация е *еднотактовата процесорна структура* (фиг.3.6), при която всяка инструкция се изпълнява за 1 такт (един машинен цикъл). В този случай дължината на машинния цикъл се определя от най-бавната инструкция (с най-голяма продължителност).



Фиг. 3.6. Еднотактова процесорна структура

Обикновено при едностъпковото изпълнение на дадена инструкция АЛУ обработва 32-битови операнди, а нейната продължителност се определя като сума от различни индивидуални времена. Например, нека са