ЛЕКЦИЯ 5
ВИДОВЕ АДРЕСАЦИЯ

- Предназначение и същност
- Основни понятия
- Класификация
- Еднокомпонентни адресации
- Многокомпонентни адресации
- Позиционна независимост
- Блок за преобразуване на адресите

ВЪВЕДЕНИЕ

1. За да върши полезна работа **ЦП обработва данни**, запомнени **в ОП**. За целта в **МИ се съдържат адреси**, посочващи данните.
2. **Най-просто и най-естествено** е в адресните полета (**АП**) да се записва **пълен адрес**.
3. **Размерът на адреса** (**и АП**) определя и **максималния обем на ОП**.
4. **Цената на ОП** спада **наполовина** всеки **2–3 години** => **адресът** се увеличава **с 1 бит**.
5. Най-голямата и най-честата **грешка при конструиране** на ЦП е **малкият** (къс) **адрес**.

 Последна редакция 10 Септември 2005 г.

ПРОБЛЕМИ

Проблемите при задаване на пълен адрес са:

❶ **АП** трябва да бъде доста **голямо** (поне **32** бита**?**).

❷ затруднява се еднотипната обработка на **данни**, разположени **в последователни адреси** на ОП.

Пример за събиране на 100 числа при двуадресни МИ и КОП: 32 – събери, 42 – извади, 52 – преход по ≠ 0.

Данни:	**I вариант:**	**II вариант:**
3000) **С У М А**	1001) 32 3000 3001	1001) 32 3000 3002
3001 ÷ 3100) числа	1002) 32 3000 3002	1002) 32 1001 4001
4001) 0000000001	• • •	1003) 42 4002 4001
4002) 0000000099	1100) 32 3000 3100	1004) 52 4002 1001
		• • •

СЪЩНОСТ

Манипулирането с адреси от ОП, за което често възниква необходимост, **предполага допълнителни** по-удобни и ефективни **начини за определяне на адрес**, освен прякото му включване в АП на МИ.

Начинът, по който **от битовете на АП** ще бъде **определен адресът**, до който една МИ ще осъществи достъп, се нарича **вид адресация**.

Като **синоними** се използват още **режим на адресиране**, **способ за адресиране** и др.

Видът на **адресирането се определя от КОП**.

ПРЕДНАЗНАЧЕНИЕ

Предназначението на някои видове е:

1. да се разреши на МИ да осъществи достъп до клетка на ОП, чийто адрес се изчислява по време на изпълнение, осигурявайки ефективен достъп до масив, списък и други структури от данни;
2. да се манипулира с адреси във форма, която е най-подходяща за често ползвани структури от данни като стек и едномерен масив;
3. да се посочи пълен адрес от ОП с минимално количество битове с цел скъсяване на МИ;
4. да се изчисли адрес спрямо мястото на МИ в ОП с цел да стане възможно въвеждането на програмите за изпълнение в произволни области от ОП без промяна на техния код.

ОСНОВНИ ПОНЯТИЯ

Регистър, който съдържа адрес от ОП, ще наричаме адресен регистър.

При несъответствие в размера като адресен регистър се използва част от или двойка регистри на ЦП (H,L при I8080).

Адресът, който ЦП изпраща към ОП при изпълнение на дадена МИ, ще наричаме Ефективен (изпълнителен) Адрес (ЕА, ИА).

При обработващите МИ чрез ЕА ЦП чете или записва данни (операнди и резултати).

При управляващите МИ чрез ЕА се определя следващата МИ (т. е. той се записва в ПБ).

КЛАСИФИКАЦИЯ

Видовете адресация се класифицират по **два независими признака**:

❶ **брой на компонентите**, определящи ЕА:
- **еднокомпонентни** (или АП или 1 регистър);
- **многокомпонентни** (АП + регистри).

❷ **тълкуване** на ЕА:
- **преки** (ЕА определя **местоположението**);
- **косвени** (ЕА определя **къде е описано** крайното **местоположение**).

ЦП имат **различни подходи** при адресиране:
само преки адресации (тези от **първо поколение**);
само косвени (**IBM 360**; **Z8000** и др.);
всяка пряка има и съответната **косвена** (**PDP 11**).

КА-05 7/48

ПРИМЕР ЗА РАЗМЕРИ

Данни за **размера на адресите и регистрите** в брой битове при някои ЦП:

Процесор	Адрес	Регистър
IBM 360	24	32
PDP 11	16 (18)	16
M6809	16	8 и 16
Z8000	31	32
I8086	16 (20)	16
M68000	32	32
I80386	32	16 и 32

КА-05 8/48

ЧИСЛА И АДРЕСИ

Адресите на клетките в ОП **са числа без знак**.

Приема се, че **нулевият е и след** 11...1 (**max**).

Адреси **не се събират** (май+юни = **?** ☹💣).

Към/от **адрес** може да **се добави**/извади **цяло число** ☞ адрес на **клетката след**/преди (май+1=юни, юни–1=май, дек+1=яну ☺❣).

Адресите може да **се изваждат** ☞ **брой** на **клетките** между тях (юни–май=1 месец ❣).

Размерът на **число без знак** се увеличава **с** добавяне на **нули** (7 = 111, но и 0111 = 7).

А размерът на **число със знак** – **чрез** неговия **знак** (+3 = 011 и 0011, –1 = 111 и 11111).

ПРИМЕРИ ЗА ПРЯКА И КОСВЕНА АДРЕСАЦИЯ

При **преките** адресации **ЕА** директно **посочва** необходимите **данни/МИ**.

При **косвените** адресации това **става с посредничеството** на ОП или регистър.

ЕДНОКОМПОНЕНТНИ ВИДОВЕ АДРЕСАЦИЯ

Еднокомпонентните видове адресация са 8:

- регистрова;
- абсолютна;
- непосредствен операнд;
- косвена регистрова;
- автоувеличение;
- автонамаление;
- косвено автоувеличение;
- косвено автонамаление.

КА-05 11/48

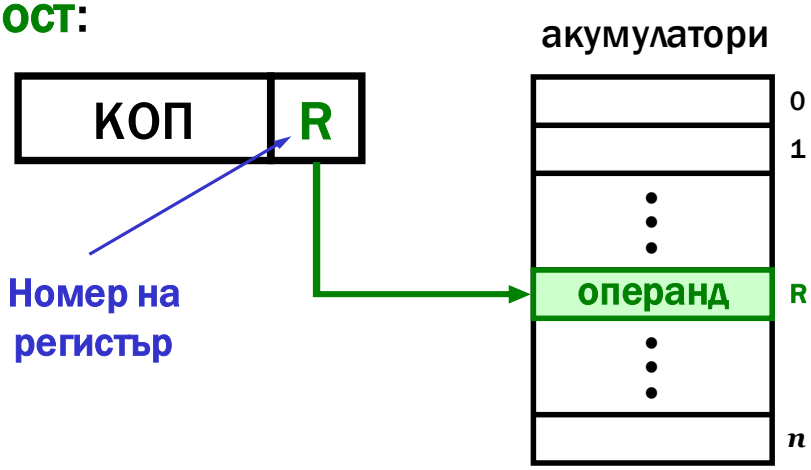
РЕГИСТРОВА АДРЕСАЦИЯ

Цел: **достъп до регистър** на ЦП.

Поява: при елиминиране на **първото АП**.

Използване: акумулатори – винаги, адресни регистри – ограничена (пренос, ±1).

Същност:



КА-05 12/48

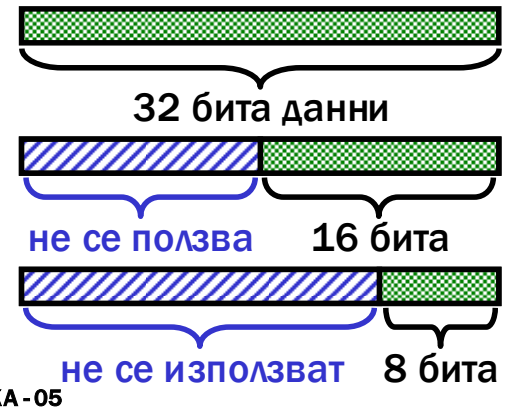
РАЗМЕР НА ДАННИТЕ

При **несъответствие** има **две** стратегии:

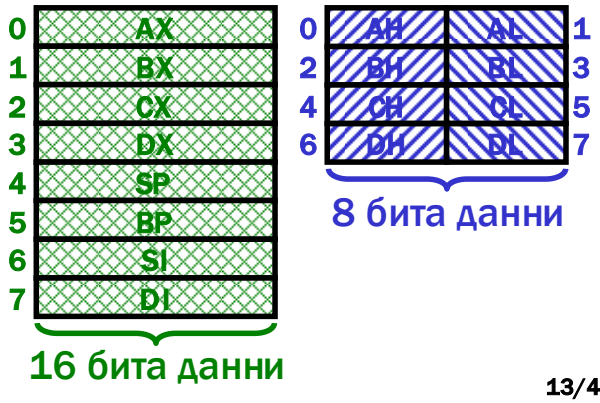
❶ използване на **част от регистъра**;

❷ **различно номериране** на регистрите.

❶ IBM, Motorola, DEC:
регистри с 32 (16) бита



❷ Intel 8086:
регистри с 16 бита



КА-05 13/48

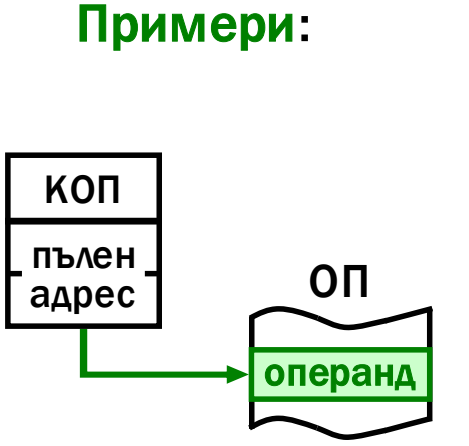
АБСОЛЮТНА АДРЕСАЦИЯ

Същност: **АП** задава **адрес от ОП**.

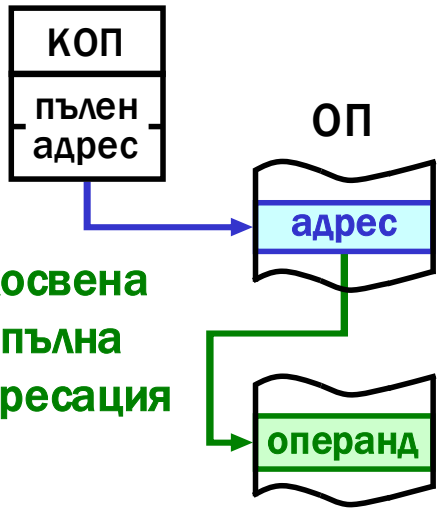
Поява: **най-естествената** адресация.

Видове: пълна, къса, косвена.

Примери:



пряка пълна адресация



косвена пълна адресация

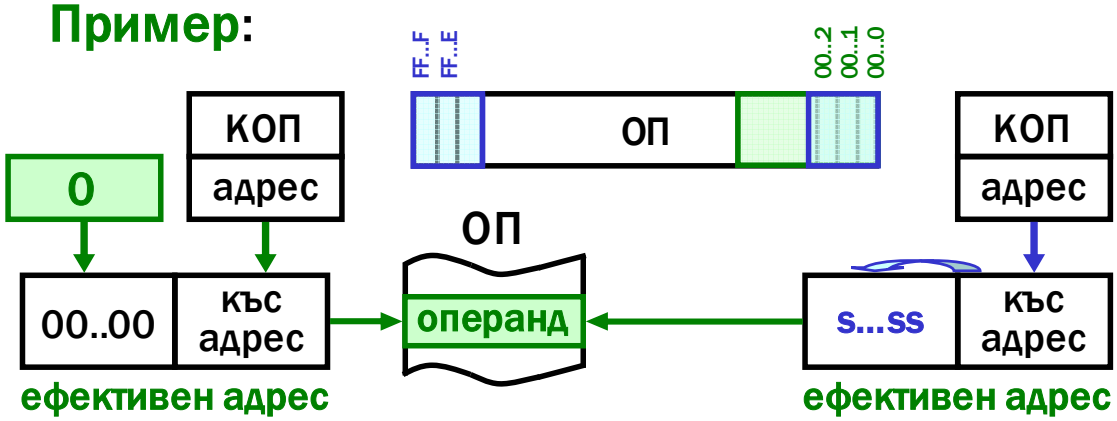
КА-05 14/48

КЪСА АБСОЛЮТНА

Цел: **съкращаване** на АП.

Основание: много адреси започват с **нули**.

Пример:



Следствие: **привилегирована област** от ОП.

Новости: много адреси започват **и с единици**.

НЕПОСРЕДСТВЕН ОПЕРАНД

Цел: да се облекчи **задаването** на **константи**.

Същност: **в АП** пряко се записват **данните**.

Не е необходима, но **ако липсва** се усложнява програмирането (**константите са в ОП**).

Ограничение: **не се използва** за посочване на **резултат и следваща** инструкция.

Примери:

Любима, при излишък на битове в МИ (**M68000**, **Z8000**).

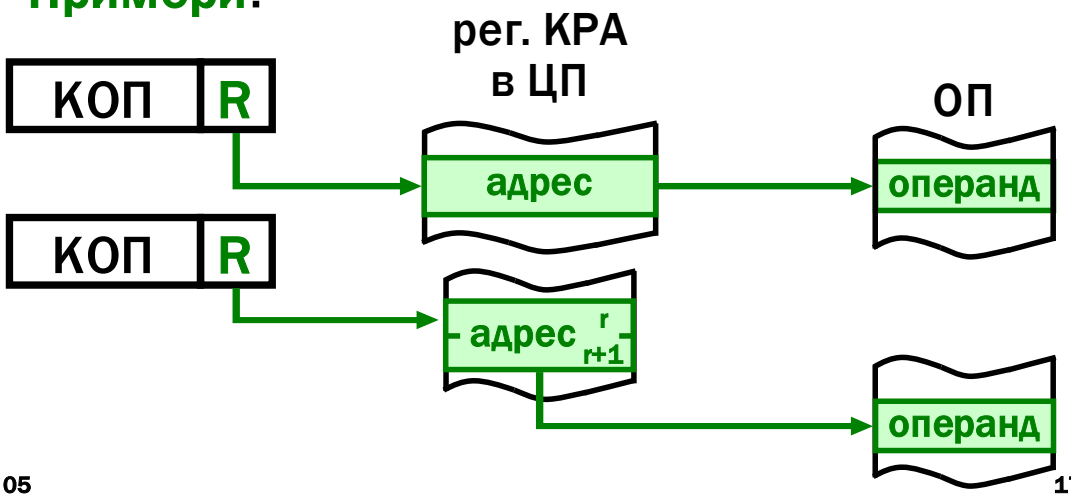
КОСВЕНА РЕГИСТРОВА

Цел: **динамична промяна** на адреса.

Същност: адресът е **в регистър** на ЦП.

Приложение: **списък**, обхождане на **масив**.

Примери:



КА-05 17/48

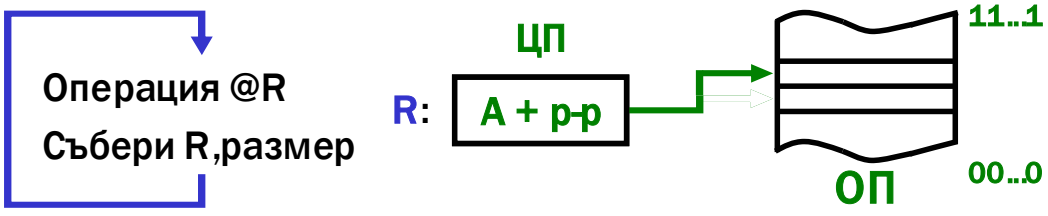
ДОСТЪП ДО ВЕКТОР

Косвена регистрова адресация може да се използва за **обхождане на вектор** в ОП.

За целта **след използване** на регистъра за достъп до данните, **стойността** му трябва да **се увеличи с размера** на данните.

Това води до **две** неприятни **следствия**:

1. изпълнение на **две МИ** (достъп + събиране);
2. възможност за **некоректно увеличаване**.



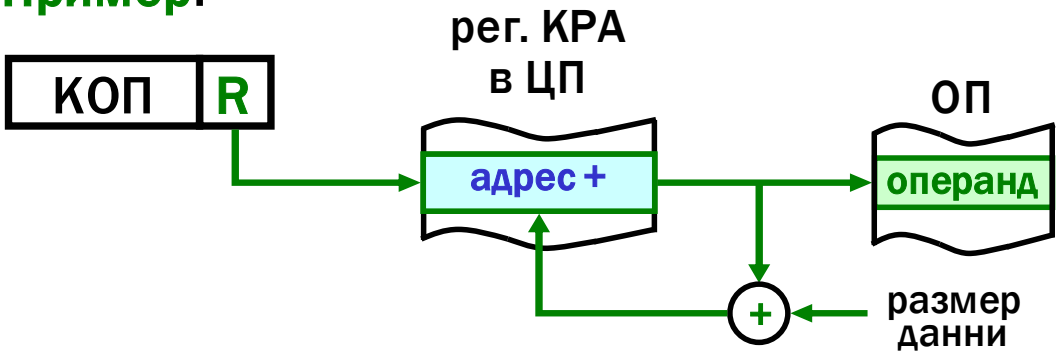
18/48

АВТОУВЕЛИЧЕНИЕ

Автоувеличението е **вариант на КРА**, който отстранява разгледаните проблеми.

Същност: **след използване** на стойността на **регистъра**, същата **се променя с добавяне на размера** на данните (в брой клетки).

Пример:



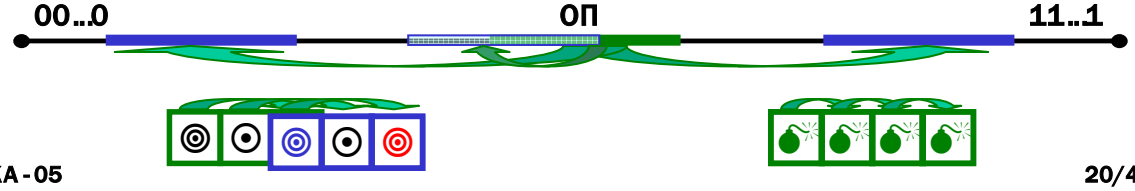
КА-05 19/48

ДОСТАТЪЧНО ЛИ Е АВТОУВЕЛИЧЕНИЕТО?

Адресация с автоувеличение дава **добри резултати** при обхождане **по нарастване** на адресите **за обработка** на данни.

Най-често обаче **обхождането е** просто **за преместване** (**пренос**) на данни от една област **на ОП** в друга.

При пренос в препокриваща област на ОП **понякога автоувеличението е безсилно**:



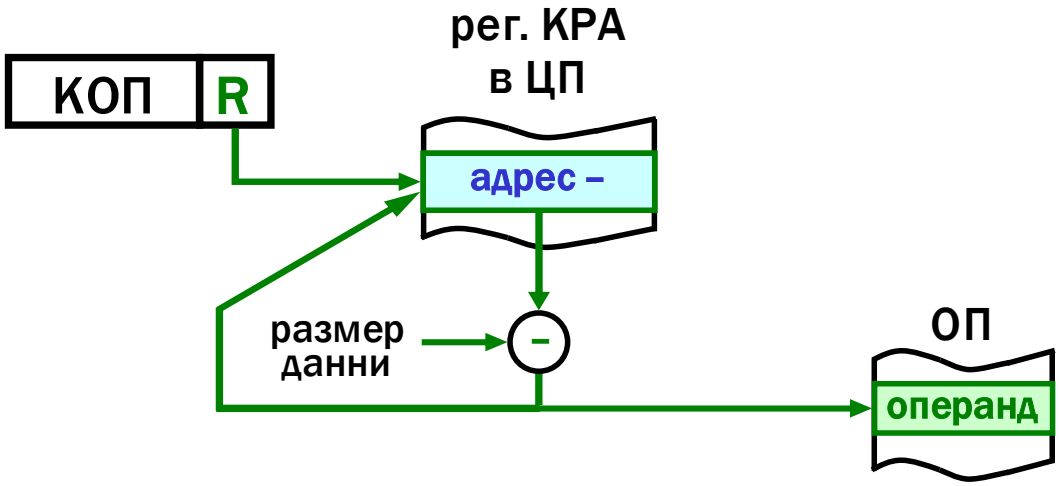
КА-05 20/48

АВТОНАМАЛЕНИЕ

Цел: да облекчи **пренасянето на блок** от ОП.

Същност: **точно обратна** на автоувеличение.

Пример:



КА-05 21/48

РАБОТА СЪС СТЕК

Едновременното наличие на автоувеличение и на автонамаление дава **възможност за** реализиране на **стек в ОП**, който:

❶ **се увеличава** (**запълва**) **към адрес 0** на ОП;

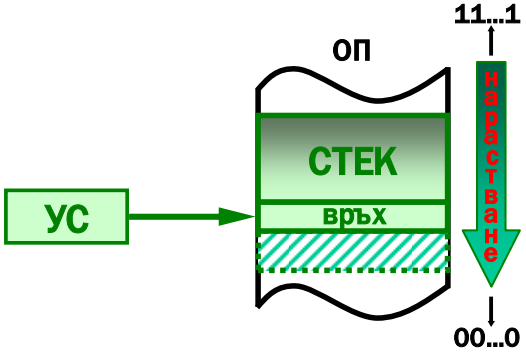
❷ има **указател към** намиращия се на **върха** на стека последен въведен елемент.

Операции със стека:

❶ **Запис** (**Push**) – **автонамаление** по УС.

❷ **Четене** (**Pull**, **Pop**) – **автоувеличение** по УС.

❸ **Достъп** до върха – **КРА**.



КА-05 22/48

ЗАБЕЛЕЖКИ

Пренасянето на блок от клетки на ОП с последователни адреси и **работата със стек** са **изключително важни** за нормална работа на компютърната система.

ЦП, които официално нямат описаните адресации, **предоставят**:

1. **специализирани инструкции за пренос на блок** от клетки на ОП (**I8086** и наследници);
2. **специализирани инструкции за** запис и четене от **стек** (практически **всички МП**).
3. регистър **Указател на Стек** (**УС**) – **Stack Pointer** (**SP**), използван от ЦП и за системни цели.

КА-05 23/48

КОСВЕНИ ВАРИАНТИ НА АВТОАДРЕСИРАНЕ

Автоувеличение и автонамаление са **косвени видове адресиране**.

Някои ЦП предоставят за тях **още една степен на косвеност**: косвено автоувеличение и косвено автонамаление.

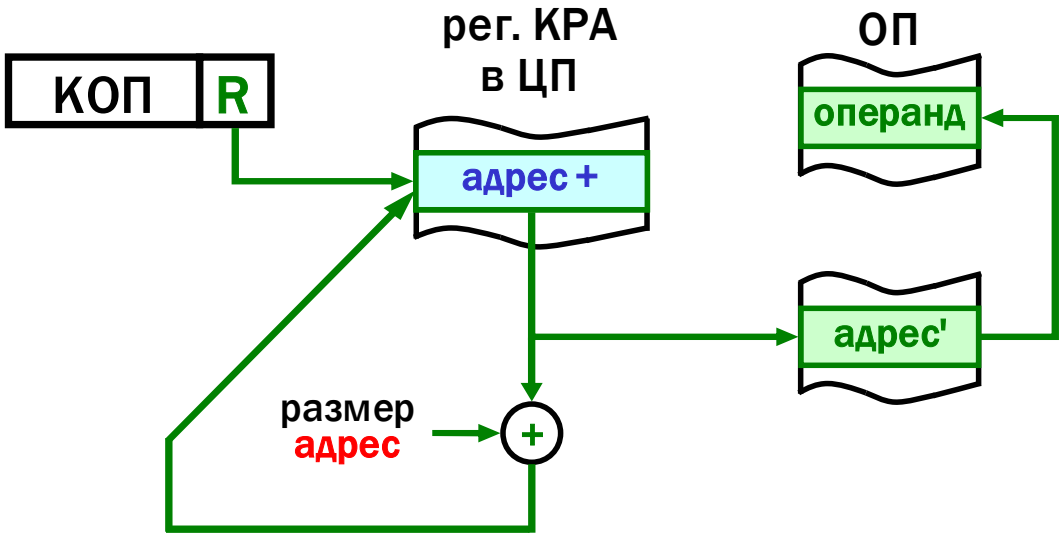
Целта на тези косвено косвени видове адресиране **е** да се улесни **обхождането на таблица с указатели** към данни.

Обхождането на подобни **таблици в двете посоки не е** жизнено **важно** – **косвеното автоувеличение е** напълно **достатъчно**.

КА-05 24/48

КОСВЕНО АВТОУВЕЛИЧЕНИЕ

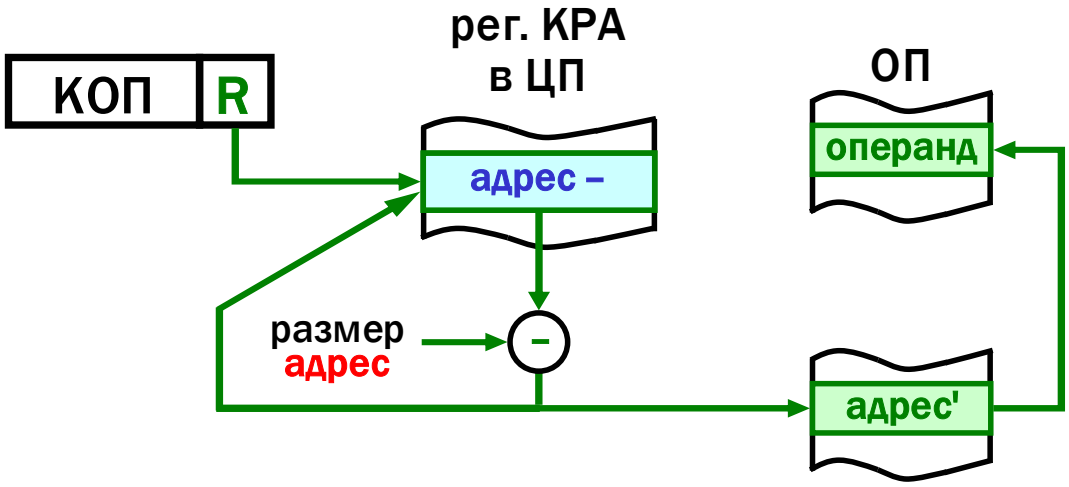
Съществува в PDP–11, VAX, M6809 и др.



КА-05 25/48

КОСВЕНО АВТОНАМАЛЕНИЕ

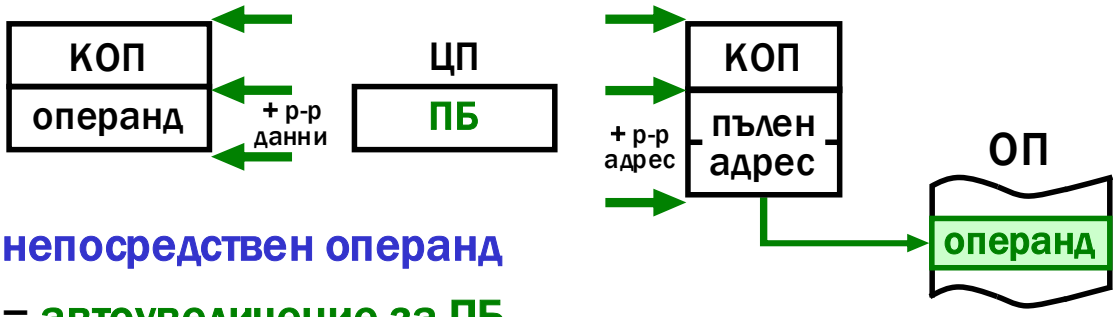
Съществува в PDP–11, M6809 и др.,
но е премахната във VAX!



КА-05 26/48

ЕКВИВАЛЕНТНИ АДРЕСАЦИИ

Някои от разгледаните еднокомпонентни **видове адресиране са еквивалентни**:



непосредствен операнд

= автоувеличение за ПБ

абсолютна адресация **= косвено автоувеличение**

КА-05 27/48

МНОГОКОМПОНЕНТНИ ВИДОВЕ АДРЕСАЦИЯ

Многокомпонентните видове адресация са по-малко от еднокомпонентните – само 5:

- 📖 стрàнична;
- 📖 с индексиране;
- 📖 по база;
- 📖 по база с индексиране;
- 📖 относителна.

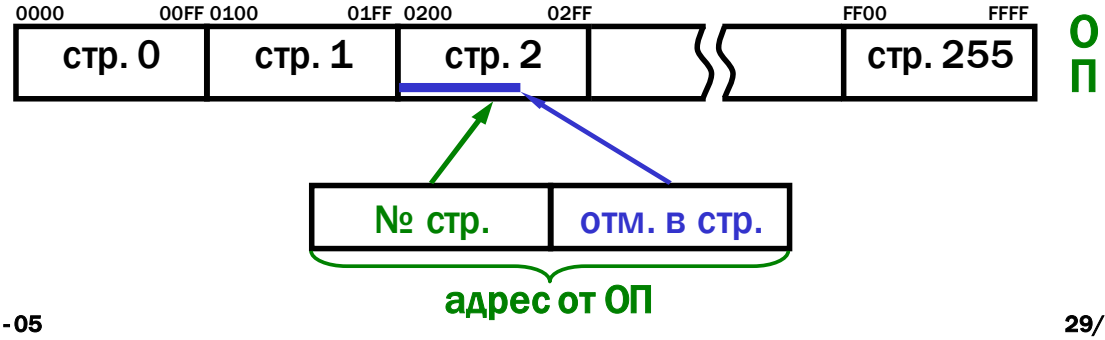
КА-05 28/48

СТРÀНИЧНА АДРЕСАЦИЯ

Цел: **съкращаване** на АП.

Същност: ОП се дели на **страници с равен размер** и в АП се задава **само отместване** в избраната страница.

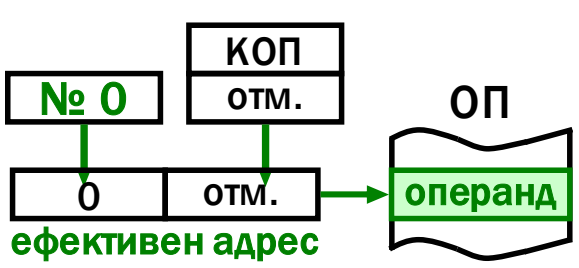
Видове: в **страница 0**, в **текуща страница**, чрез **стрàничен регистър**.



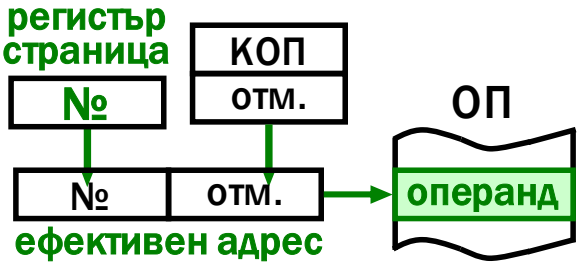
КА-05 29/48

СТРÀНИЧНИ АДРЕСАЦИИ

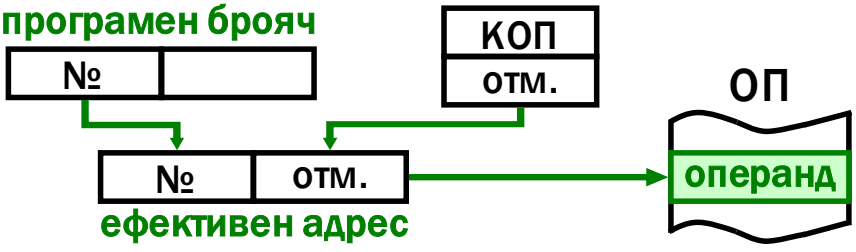
❶ адресация в **страница нула**:



❸ адресация чрез **стрàничен регистър**:



❷ адресация в **текущата страница**:



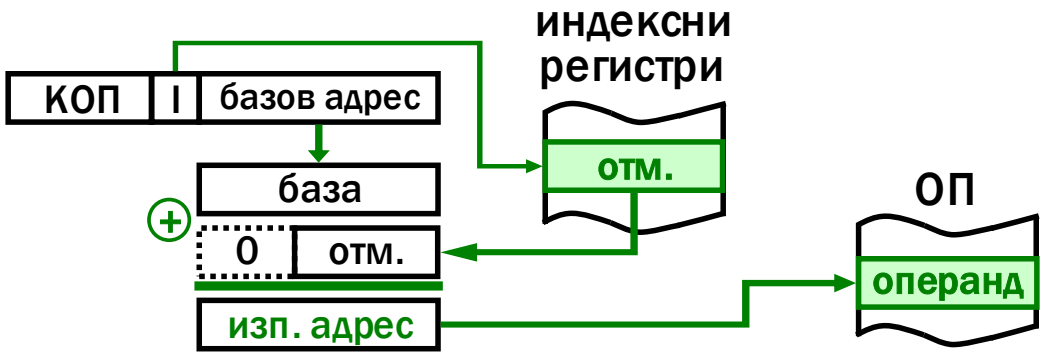
КА-05 30/48

АДРЕСАЦИЯ С ИНДЕКСИРАНЕ

Цел: **динамично модифициране** на адрес.

Същност: **базов адрес в АП** се модифицира с **добавяне** на **отместване в регистър** на ЦП.

Приложение: **достъп до** елемент на **масив**.



КА-05 31/48

КОСВЕНИ АДРЕСАЦИИ С ИНДЕКСИРАНЕ

Адресирането с индексиране по принцип **е пряка адресация**, тъй като базовият адрес се посочва явно в АП на МИ.

Някои процесори дават възможност и за **два вида косвени адресации с индексиране**:

❶ **прединдексна**: **полученият след сумирането** на базата и отместването **адрес е косвен** и посочва в ОП **адрес** на данните **вместо данни**;

❷ **слединдексна**: **базовият адрес** в МИ **е косвен** и в сумирането участва прочетеният чрез него от ОП **истински** базов **адрес** на масива.

КА-05 32/48

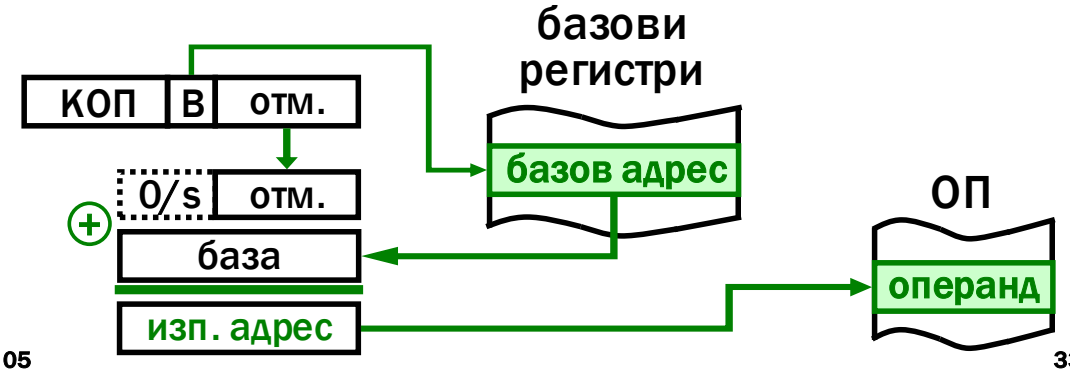
АДРЕСАЦИЯ ПО БАЗА

Цел: **скъсяване на АП**.

Основание: **обръщенията** към ОП са **съсредоточени в малка област** от адреси.

Същност: базов **регистър посочва началото** или средата **на** достъпната **област от ОП**.

Приложение: **достъп до поле** от запис.



КА-05 33/48

ПО БАЗА ИЛИ С ИНДЕКСИРАНЕ

Двата вида адресации – по база и с индексиране, си приличат защото при тях се **сумират две компоненти**.

При **адресация по база**:

- **пълният** базов **адрес** е **в регистър** на ЦП.
- късото **отместване** е **в АП** на МИ (**икономия**).
- **стойността** на регистъра **не се мени**.

При **адресация с индексиране**:

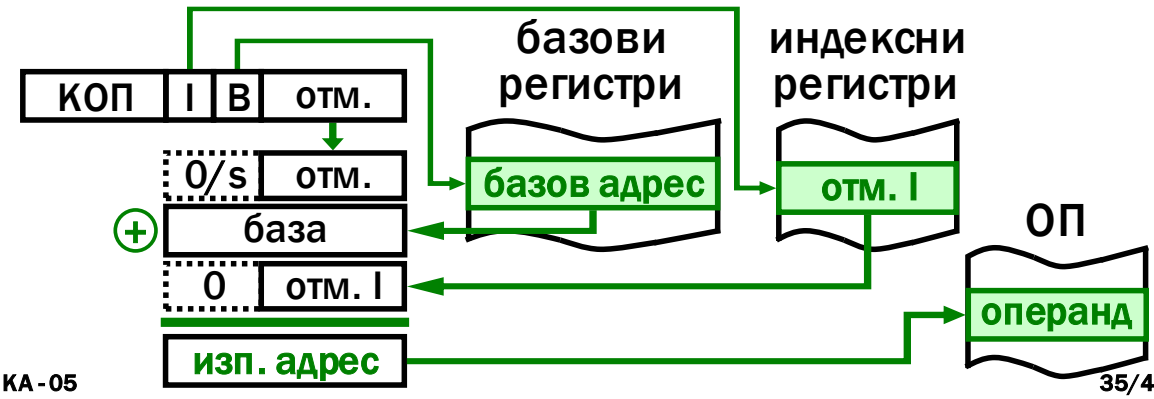
- **пълният** базов **адрес** е **в АП** на МИ.
- късото **отместване** е **в регистър** на ЦП.
- **стойността** на регистъра **се мени** (**динамична**).

КА-05 34/48

АДРЕСАЦИЯ ПО БАЗА С ИНДЕКСИРАНЕ

Двете популярни адресации имат **различна цел** и за удобство **често се обединяват**.

Базовият регистър и отместването в МИ определят **началото на вектор**, което се **индексира за достъп** до негов елемент.



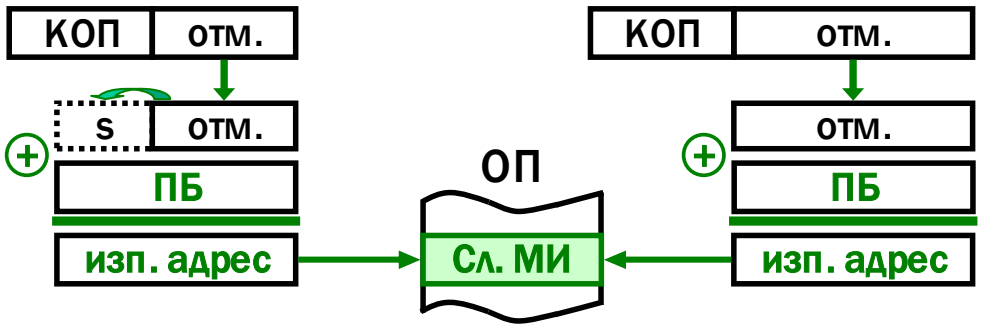
КА-05 35/48

ОТНОСИТЕЛНА АДРЕСАЦИЯ

Цел: да определи **адрес относно МИ**.

Приложение: **управляващи МИ**.

Основание: 90% от **преходите са наблизо**.



къс преход (8 бита)

пълен преход (M6809, M68000)

КА-05 36/48

МОДИФИКАЦИИ

❶ При PDP–11 дължината на МИ е кратна на 2 и всички МИ започват от четен адрес в ОП: отместването се умножава по 2 преди сумирането (добавя се 0 отзад).

❷ Някои МЕ имат инструкция за край на цикъл (SOB в PDP–11 и DJNZ в Z8000), при която преходът ще бъде назад в ОП: отместването е без знак и се вади от ПБ.

❸ Адресацията рядко се ползва за достъп до данни: в M68000 това е разрешено само за операндите, но не и за резултата.

КА-05 37/48

ПОЗИЦИОННО-НЕЗАВИСИМА ПРОГРАМА

Програма, която може да работи коректно на произволно място в ОП, се нарича позиционно-независима.

Позиционната независимост е два вида:

❶ статична: програмата може да бъде въведена на (заредена от) произволен адрес на ОП.

❷ динамична: по време на изпълнение програмата може да бъде преместена.

КА-05 38/48

СТАТИЧНА НЕЗАВИСИМОСТ

Основна характеристика на статичните позиционно-независими програми е, че в тях **не** се **съдържат абсолютни адреси**.

Два вида **адресиране** осигуряват подобна независимост: **относително** и **по база**.

При адресиране по база **адресните константи** трябва да са **спрямо адреса на зареждане**.

Абсолютна адресация може да се използва **само при достъп до външни** за програмата **адреси** (например в **ПП**, **ОС** и др.).

КА-05 39/48

ДИНАМИЧНА НЕЗАВИСИМОСТ

Динамична позиционно-независима програма се създава изключително **трудно**, защото се осигурява **само при относително адресиране**, а то не винаги е разрешено.

Динамична позиционна независимост **се постига** само **при ЦП**, които прилагат **скрито** (от програмата) **адресиране по база**.

Служебните базови регистри са под контрола на **ОС**, която единствена има интерес да **мести програми** при тяхното изпълнение.

КА-05 40/48

БЛОК ЗА ПРЕОБРАЗУВАНЕ НА АДРЕСИТЕ

Някои КС използват **Блок за Преобразуване на Адресите** (**БПА**) със следните функции:

Увеличаване на обема **на ОП** над този на ЦП.

Разделяне на голяма по обем **ОП**, така че в нея да се поместят **няколко програми**, които да бъдат **изолирани помежду си**.

БПА (**MMU – Memory Management Unit**) може да бъде **външно устройство** (**PDP–11**) или стандартна **част от ЦП** (**I80x86**).

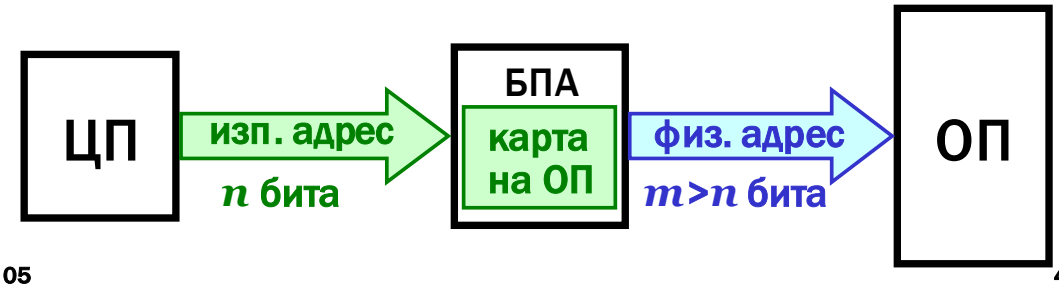
КА-05 41/48

ПРИНЦИП НА БПА

❶ **ЦП изчислява логически** изпълнителен **адрес**, който изпраща **към ОП**.

❷ **БПА** прехваща логическия адрес и **го преобразува във физически** адрес на ОП.

❸ За преобразуването **БПА използва карта** за разпределение на паметта (**PDP–11**) **или** специални **регистри на ЦП** (**I80x86**).



КА-05 42/48

КАРТА НА ПАМЕТТА

❶ Чрез карта на паметта работещата програма не може да адресира текущо повече от допустимите за ЦП адреси.

❷ За достъп до друга област на ОП със същия обем трябва да се измени картата.

❸ БПА дели логическия адрес на две части: № на регистър и отместване спрямо него.

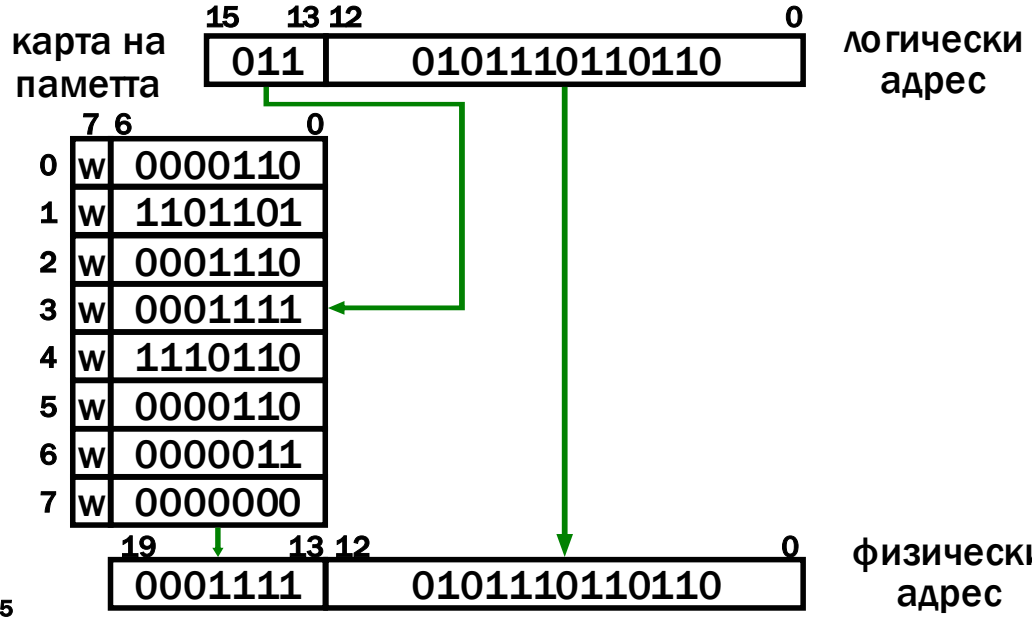
❹ Определеният чрез номера си регистър дава старшата част на физ. адрес, към която се дописва наличното отместване.

❺ Така БПА реализира адресация по база.

КА-05 43/48

ПРИМЕРНА КАРТА

16-битов логически адрес се разширява до 20-битов физически чрез 8 регистрова карта.



КА-05 44/48

БПА ПРИ ИНТЕЛ

При производството на своите МП от 1980 г. фирмата Интел осигурява апаратна съвместимост отдолу нагоре.

Така всички МП от фамилията I80x86 поддържат съвместимост с първия от тях, а именно I8086.

I8086 определя 16-битов ЕА. Логическият ЕА се разширява до 20 бита, чрез БПА, който е неотменна част на ЦП. Полученият 20-битов физически адрес се изпраща към ОП.

КА-05 45/48

ПОЛУЧАВАНЕ НА АДРЕСА

Генерираният от ЦП ЕА се явява отместване, което се добавя към 20-битов начален адрес, получен от програмно достъпен 16-битов сегментен регистър, умножен по 16.

Така по време на своето изпълнение всяка програма има достъп до 4 сегмента – кодов, (CS) стеков (SS), даннов (DS) и допълнителен (ES), всеки от които е с размер 64 килобайта.

сегментен регистър (16)	0000
+ ефективен адрес на ЦП (16)	
краен физически адрес в ОП (20)	

КА-05 46/48

СЕГМЕНТНИ РЕГИСТРИ

За да не се налага непрекъснато **уточняване** кой сегментен регистър съдържа базовия адрес **се използва система за подразбиране**.

Специална **еднобайтова МИ може да измени подразбирания** от ЦП сегментен **регистър**.

обръщение към ОП за	стандарт	алтернатива	лог. адрес е
извличане на МИ	CS	няма	IP
работа със стека	SS	няма	SP
работа с данни	DS	CS, ES, SS	ЕА на ЦП
BP като базов р-р	SS	CS, ES, SS	ЕА на ЦП
низ-източник (спец. МИ)	DS	CS, ES, SS	SI
низ-приемник (спец. МИ)	ES	няма	DI

КА - 05 47/48

БЛАГОДАРЯ ВИ ЗА ВНИМАНИЕТО!

БЪДЕТЕ С МЕН И В СЛЕДВАЩАТА ЛЕКЦИЯ, КОЯТО ЩЕ НИ ОТВЕДЕ В НЕВЕРОЯТНИЯ СВЯТ НА КОМПЮТЪРНИТЕ ОПЕРАЦИИ

