

1. Модели компютърни архитектури

Васил Георгиев



[i s . f m i . u n i - s o f i a . b g / t 3 /](https://www.is.fmi.uni-sofia.bg/t3/)



[v . g e o r g i e v @ f m i . u n i - s o f i a . b g](mailto:v.georgiev@fmi.uni-sofia.bg)

Съдържание

- ✦ Модели машинна архитектура и обработка: класификация и метрика
- ✦ Мултипроцесори: **UMA, NUMA, COMA**
- ✦ Векторни и потокови машини и систолични матрици
- ✦ Мултикомпютри

Класове компютърни архитектури

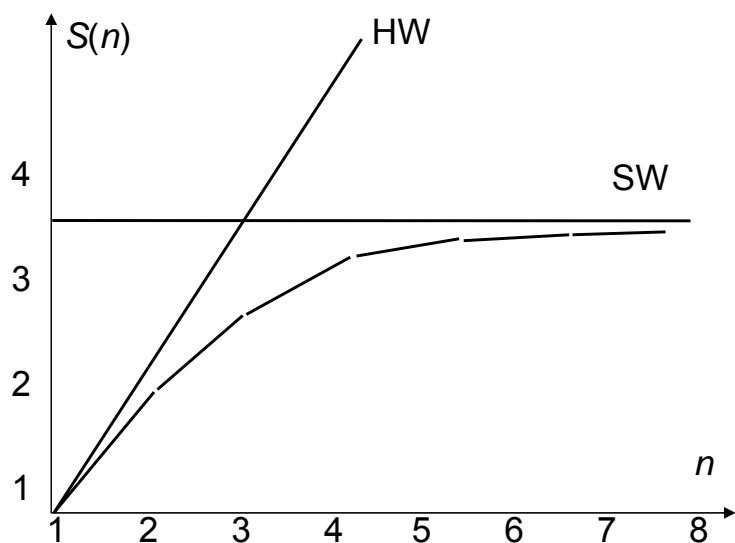
- архитектура – компоненти и организация на системата
- фон Нойманова (1.3.1) – на възли и мрежи и неklasическа организация (систолични, потокови, логически и редукионни модели и невронни мрежи)
- класификация на Michael Flynn (1966) по управление на потока инструкции и потока данни (операнди) – SISD, SIMD, MISD и MIMD архитектури – (1.3.2)
 - SIMD – за векторна обработка, фина грануларност
 - MISD – за конвейрна обработка (обработнащи фази върху вектор) – систолични масиви
 - MIMD – обикновено с локална и глобална памет; за средна и едра грануларност
- технологично-ориентирана таксономия на паралелните архитектури: мултипроцесори, мултикомпютри, потокови машини, матрични процесори, конвейерни векторни процесори и систолични матрици – частично съответствие с класовете на Флин (1.3.3)

HW/SW паралелизъм

- За паралелно изпълнение на програмни едновременно апаратна и програмна поддръжка.
 - Апаратен паралелизъм. Обуславя се от архитектурата и ресурсите, които са баланс между производителността и цената. Характеризират се с пикова производителност и средно натоварване. Той задава зависимостта по ресурси.
 - Програмен паралелизъм. Обуславя се от зависимостта и по управление. Реализира се като
 - паралелизъм по управление - конвейризация, мултиплициране на функционални възли. Обслужва се паралелно, прозрачно за програмиста.
 - паралелизъм по данни - типичен за SIMD, но и при MIMD.

Метрика: ускорение и эффективность

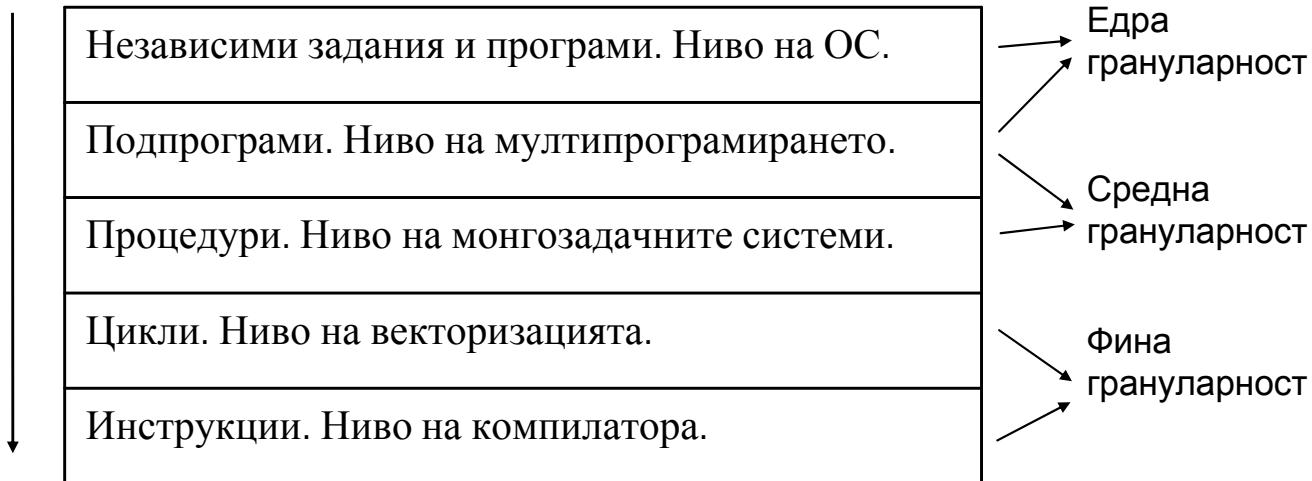
- ускорение $S(n) = T_1/T_n$; лимитиращи фактори
- ефективност $E(n) = S(n)/n$



Делене на обработката: грануларност

Ниво на паралелизма

Комуникационен свръхтовар



SIMD

- обобщения модел включва контролно устройство и еднотипни обработващи модули с достъп към обща памет – (1.7.1)
- програмно-апаратна зависимост на паралелизма/ускорението – пример за изпълнение на програма на SIMD машина (1.7.2)
- процесорните елементи изпълняват операциите във формат битове или думи
- локалната памет за данните може да бъде разпределена, обща или йерархична (със свързваща мрежа) (1.7.3)
- особености:
 - опростена архитектура спрямо MIMD поради общото контролно устройство (за дешифриране и зареждане на инструкциите) и съответно поддържане само на едно копие от кода за инструкции
 - скаларните операции (включително контролната логика) се изпълняват от контролното устройство – евентуално конкурентно на паралелната обработка на данни в обработващите устройства
 - имплицитна синхронизация между отделните обработващи устройства (при MIMD – експлицитна)
- примери – фамилия Connection Machine на Thinking Machine Co.

MISD

- това е архитектурния принцип на всички конвейри – вкл. на процесорния конвейер – обработката се разделя на последователни фази; обработката на следващата инструкция (при най-фина грануларност) или на следващия процес започва веднага щом предходния процес освободи първата фаза – (1.8.1)
- прилагат се и функционални (или циклични) конвейри например с фазите (1.8.2):
 - четене на инструкциите от обща памет
 - зареждане в обработващото устройство с евентуално буфериране
 - обработка
 - пренос на резултата към общата памет (буфериране)
 - запис в общата памет
- инструкционно, субсистемно (обикн при аритметична обработка – нелинейни конвейри с фази `add, mul, div, sort...`) и с и с т е м н о н (процеси, също и програмна организация) на конвейризация

Систолични матрици (Systolic Arrays)

- представляват модификация на **MISD** на субсистемно ниво, специализирана архитектура за определени алгоритми – с многодимензионни конвейри т.е. фиксирана мрежа от обработващи устройства
- ограничено приложение – ЦОС (цифрова обработка на сигнали – **DSP**), обработка на образи и др.
- опростени процесорни елементи и комутираща съобщителна мрежа с ограничен набор шаблони
- управлението е по инструкции (**control flow** – не **data flow**) но програмирането е като при потоките архитектури
- архитектурата включва обработващ масив (с комутатор) и управляващ модул, който настройва масива, предава данните и извлича резултатите (+ контролен възел – хост) – (1.9.1)
- производителността се понижава значително при интензивен вход/изход
- топологични шаблони:
 - систолични вектори – по същество конвейри
 - двумензионни масиви – обикновено регулярни с коеф. на съседство най-често 4 или 6 (1.9.2)

...Систолични матрици (Systolic Arrays)

- ✦ тенденцията е към елементи за фина грануларност – на инструкционно ниво – снабдени с няколко високоскоростни дуплексни серийни канали (броя на които определя валентността – коеф. на съседство)
- ✦ пример: iWrap серия на Интел и университета Carnegie-Mellon (1.10.1) – процесорната клетка се състои от
 - ✦ iWrap компонент с изчислителен и комуникационен агент и
 - ✦ страницирана памет с директен интерфейс към компонента
- ✦ пример: умножение на матрици в двумерен систоличен масив с коеф. на съседство 6 (1.10.2)

MIMD

- това е архитектурния принцип на всички мултипроцесори и мултикомпютри:
- процесорите са автономни и могат да изпълняват различни програми (вкл. локално копие на ОС!)
- имат общ ресурс с разпределен конкурентен достъп – памет или комуникационна среда
- организация: по памет / по комуникации

автономни (локална памет)	общо адресно пространство (общодостъпна памет)
магистрални	комутационни

- универсални, отказоустойчиви, по-едра грануларност
- обикновено се изграждат с масови процесори (вместо специализирани процесорни елементи с ограничени функции)

...MIMD

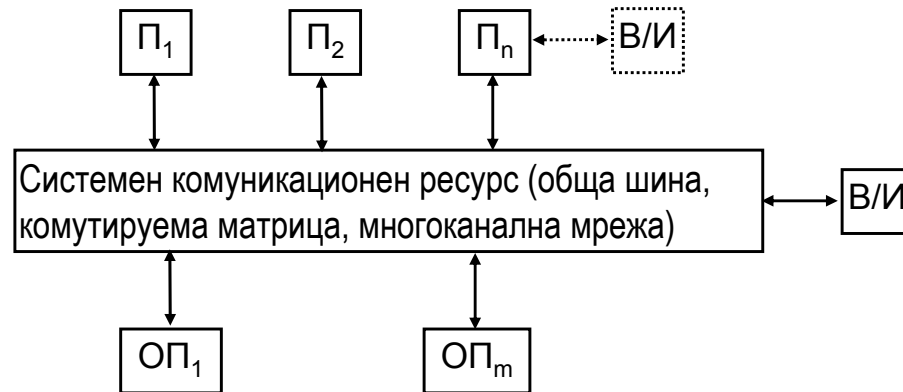
- наличието на автономна локална памет ги разделя на:
 - системи с обща памет; синоними: мултипроцесори | [shared-memory | tightly-coupled] systems | Global-Memory MIMD, GM-MIMD | Uniform Memory Access System – UMA
 - системи с обмен на съобщения; синоними: мултикомпютри, [distributed-memory | loosely-coupled] systems | Local-Memory MIMD, LM-MIMD | Non-Uniform Memory Access System – NUMA (поради наличието на локална и отдалечена памет)
- глобално и локално адресно пространство; виртуалната памет поддържа глобално адресно пространство на страниците (не на ниво думи), което се управлява от разпределена ОС (РОС) за МП и хомогенните МК. При МК общата виртуална памет се поддържа и с обмен на съобщения – 1.12.3
- хетерогенните МК използват мрежови ОС (МОС), при които нивото на достъп е разпределена файлова система (напр. базирана на DNS) с ползване на примитиви от типа `read`, `write`...

Мултикомпютри

- Разпределената обща памет (**distributed shared memory DSM**): програмната имплементация на обща памет в система с автономни възли (и адресни пространства)
 - виртуално общо адресно пространство от страници (не думи) – 4/8 kB – (което позволява програмиране за мултикомпютъра като за виртуален уникompютър)
 - при отсъствие на страница от локалната памет възниква вътрешно прекъсване (**memory trap**) и зареждане на страницата в локалната от отдалечената памет
 - възможно е репликиране на страници само за четене (**read only**);
 - ако страницата е и за запис, се прилагат различни мерки за поддържане на свързаност
- Системи с обмен на съобщения – **Message passing distributed systems**

Архитектура с обща памет (мултипроцесори) – UMA

- **UMA (uniformly shared memory access)** - еднакъв достъп на процесорите - силно свързани системи: архитектура
 - обща шина - разширение от унипроцесинг към мултипроцесинг
 - комутируема матрица (crossbar switch)
 - многоканални мрежи

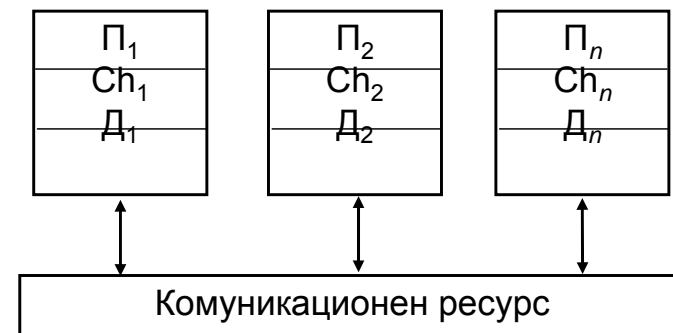
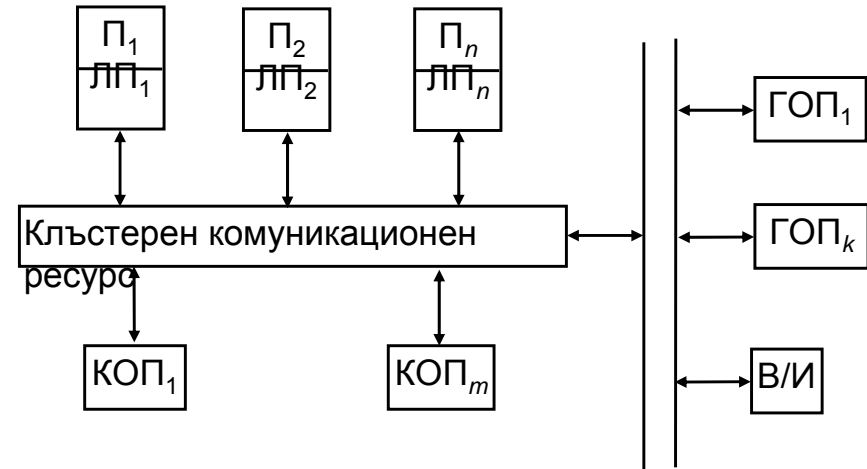


- синоним: симетричен (централизиран В/И) и а с и м е т р и ч е с к и е с и с т е м и (специализиран процесор за В/И) мултипроцесинг - обикновено хомогенни системи.

NUMA и COMA

➤ **NUMA (non-uniformly shared memory access)** - йерархия на общата памет - локални, глобални и/или клъстерни памет

➤ **COMA (cache only shared memory access)** - паметта е локална (cache) но йерархията и позволява част от нея (“директория”) да се адресира отдалечено.



Потокови архитектури (Data Flow)

- при класическите фон Нойманови архитектури (вкл. модификациите по Флин) програмата е последователност от инструкции, която се изпълнява от контролно устройство – **control flow**
- при потоковите архитектури операциите се изпълняват веднага при наличие на операндите (и наличие на операционен ресурс) – контрола се осъществява чрез планиране на операндите т.е. данните; концептуално всички инструкции с готови операнди могат да се изпълнят паралелно (на практика конкурентно)
- програмите за потокови архитектури се представят с потокови графи (обикн. с текстов синтаксис) – възлите представят операции, а дъгите – информационните връзки на операндите; нивото на паралелизъм обикновено е инструкционно – **1.16.1**
- допълнителни особености на потоковите архитектури: реконфигурация, буфериране на данните, комплементиране на операндите
- наличие на управляващ процесор, който пакетира операндите и инструкцията в блок – **token** – и го предава на някой от обработващите процесори

Статични потокови архитектури

- **статични** – програмния (потоковия) граф е фиксиран. За изпълнение на повече от една програма се използват различни варианти на зареждането на данните, които се генерират на етапа компилация
 - този модел не поддържа процедури, рекурсия и обработка на масиви
 - организация 1.17.1
- **статични с реконфигурация** – логическите връзки между процесорните елементи се установяват на етапа зареждане на програмата: топологията на връзките се решава от компилатора и след зареждане на програма остава фиксирана при изпълнението; особености:
 - физическите канали съществуват, но са комутират
 - броя алоцирани (заредени) процесори обикновено е по-малък от инсталираните процесори поради ограничения в комутацията – логическата връзка между процесорите е дърво, не всички процесори в листата на което се използват
 - пример – **MIT Data Flow Machine** – клетките памет съответстват на информацията във възлите на потоковия граф – т.е. инструкционните блокове (**tokens**) – когато блока е комплектован с операнди, той се предава като операционен пакет към елемент за обработка; пакета с резултата се връща в клетъчната памет – 1.17.2

Динамични потокови архитектури

- базират се на логически канали между процесорите, които могат да се реконфигурират по време на изпълнение подобно на система с обмен на съобщения – с маркирани блокове (**tagged tokens**)
 - дъгите в потоквия граф могат да съдържат повече от един блок едновременно (но с различни марки!)
 - операциите се извършват когато възела получи блокове (с еднакви марки) на всичките си входящи дъги
 - циклични итерации могат да бъдат изпълнявани паралелно: за целта всяка итерация се представя като отделен субграф като маркировката се разширява с номера на итерацията – 1.18.1 (само при информационна независимост на итерациите!)
 - пример – **Manchester Data Flow Machine MDM**: цикличен конвейер, в който блоковете циркулират и се управляват от ключов модул. Компонентите са (1.18.2)
 - Блоков буфер (**token queue**) – за съхраняване на междинни резултати (ако се произвеждат по-бързо отколкото е последващата им обработка) – капацитет 32К блока и производителност 2.5 МБлока/Сек
 - Комплементираща памет (**matching store**) – за комплементиране на блоковете с еднакви марки – процеса е апаратен и поддържа до 1.25 МБлока
 - Памет инструкции (**instruction store**) – n -торките (обикновено 2ки) операнди-блокове се пакетират с инструкции и адрес (етикет) на резултата и се предават за изпълнение
- обработващ модул (**processing unit**) – 20 процесора (24-битова дума и 4Кдуми вътрешна памет)

Съпоставка на компютърните архитектури

Тип	Принцип на действие	Интерфейс	Приложимост	Сложност	Ефективност
SIMD	<i>спонтанен</i>	<i>директен</i>	<i>средна</i>	<i>висока</i>	<i>висока</i>
MIMD	<i>сложна абстракция</i>	<i>най-сложна организация</i>	<i>висока (универсални)</i>	<i>висока</i>	<i>средна</i>
MISD	<i>спонтанен</i>	<i>директен</i>	<i>ниска</i>	<i>ниска</i>	<i>висока</i>
Систолични	<i>сложна абстракция</i>	<i>директен</i>	<i>ниска</i>	<i>средна</i>	<i>висока</i>
Потокови	<i>сложна абстракция</i>	<i>сложна организация</i>	<i>висока</i>	<i>висока</i>	<i>висока</i>

Мрежи за връзка

- ✦ осъществяват комуникациите между процесорните възли
п р и в с и ч к и в и д о в е М-Статични и динамични
(базират се на [каскади от] комутируми блокове -
ключове)
- ✦ топологии на свързване
 - ✦ пълен граф
 - ✦ линия и пръстен
 - ✦ двудименсинна циклична и ациклична мрежа
 - ✦ хиперкуб (n-куб)
 - ✦ двоично дърво
 - ✦ **shuffle exchange**

Характеристики на мрежите за връзка

- ✦ разстояние d_{ij}
- ✦ диаметър на мрежата $D = \max\{d_{ij}, \forall (i, j)\}$ - изисква по-голям брой канали между възлите, респ. валентност
- ✦ валентност на възлите (degree)
- ✦ сечение (bisection width) $S = \min\{\text{AllLinks}(X, Y) : ||X| - |Y|| \leq 1\}$
- ✦ разширяемост

ТОПОЛОГИЯ	брой ВЪЗЛИ	валентн ост
линия и пръстен	d	2
двоично дърво	$2^d - 1$	3
shuffle exchange	2^d	3
двудименсионна мрежа	d^2	4
хиперкуб	2^d	d
пълен граф	N	N-1