2. Процесорни архитектури

Васил Георгиев

 is.fmi.uni-sofia.bg/t3/

 v.georgiev@fmi.uni-sofia.bg

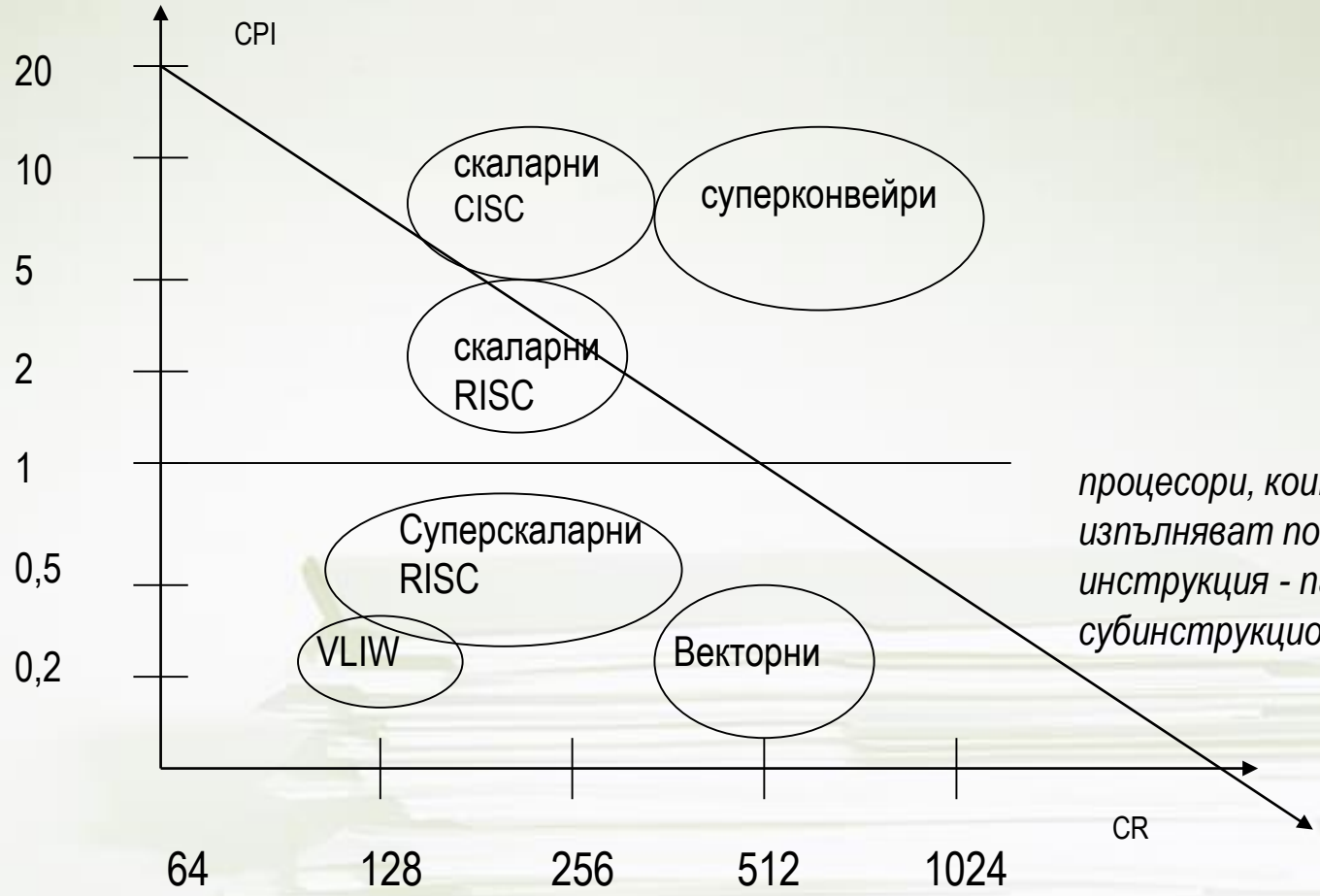
Съдържание

- Процесорни архитектури – технологично пространство
- Линейни и нелинейни конвейри
- Архитектура на набора инструкции
- Субинструкционен паралелизъм
- Суперскаларни и мултипроцесорни масови процесори

Процесорни архитектури

- **Main frame –** широк архитектурен клас от компютри, прилагат се някой от следните процесорни архитектури:
  - Скаларни процесори **CISC (Complex Instruction Set Computer); RISC (Reduced Instruction Set Computer);**
  - Суперскаларни **CISC; RISC,** само че **RISC** се използва по-често по технологични причини;
  - Процесори **VLIW – Very Long Instruction Word;**
  - Векторни;
  - Суперконвейрни **[super piperline];**
- Основни характеристики на всички архитектурни класове процесори:
  - Процесорни цикли **CPI;**
  - Тактова честота **CR.**
- Тези два параметъра на пръв поглед са независими, но между тях съществува корелация, която може да се представи в диаграма на технологичното пространство:

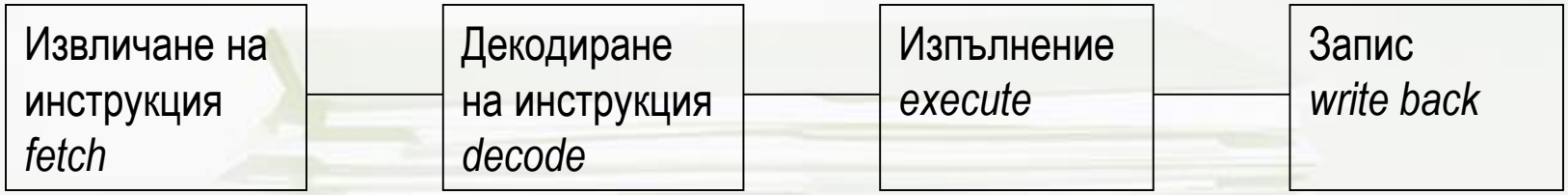
Диаграма на технологичното пространство [MHz]



процесори, които за един цикъл изпълняват повече от една инструкция - паралелизъм на субинструкционно ниво

Фази на инструкционен конвейер

- Процесорната обработка на типична инструкция реализира **MISD** паралелизъм на инструкционно ниво и минава през фазите извличане (от **cache** - обикновено 1 инстр. за цикъл), декодиране (установява функцията за изпълнение и необходимите ресурси - регистри, магистрали, устройства), издаване (резервира ресурсите чрез блокиране и извлича операндите от регисрите към устройствата), изпълнение (1 или повече фази), записване (**writeback** - на резултатите в регистрите).



Времедиаграма на инструкционен конвейер

- Закъснението между две последователни инструкции е една фаза при скаларните процесори

- Поради ресурсен конфликт между фазите на извличане и запис по-често се прилага закъснение на две фази (два субтакта) между инструкциите:

Синхронни линейни конвейрни процесори ЛКП

- ЛКП е каскада от **k** процесорни фази **(stages - Si)**, която изпълнява фиксирана функция върху данните, преминаващи през устройството от входа **(S1)** през последователните фази ($S_i \rightarrow S_{i+1}$) към изхода му **Sk.** Те не са динамично **(run-time)** настройваеми т.е. са статични. Изпълняват операционни, аритметични и обменни инструкции.
- Синхронните ЛКП са с интерфейс между фазите, който представлява синхронизиращи буферни ключове **(latches)** с общ такт. Фиг. **2.7.** Ключовете са регистри които изолират входовете от изходите и предават данните синхронно във всички фази. Фазата с най-голямо закъснение определя общия такт и общата производителност: $\tau=\tau_{max}+d_{latch}$; $P_{peak}=f=1/\tau$. Проявява се и фазово отместване **s (skew[ing])** на такта при предаване на тактовия сигнал между фазите. Затова се избира $\tau=\tau_{max}+d_{latch}+s$.

Асинхронни линейни конвейрни процесори ЛКП

- Асинхронните ЛКП контролират потока данни с “**Hand Shaking**” протокол - **Ready/Ack** между $S_i \rightarrow S_{i+1}$. Фиг. **2.8.** Подходящи за комуникационни канали при системи с обмен на съобщения. Производителността на отделните фази може да варира.

Нелинейни конвейрни процесори НЛКП

- Динамични, настройваеми, допуска се разклонение, обратна връзка **(feedback)** и пре-предаване **(feedforward)** на данните за обработка. Фиг. **2.9.1.** Изходът може да не е от последната фаза.
- **Карта на резервацията.** Тук не е тривиална като при ЛКП. За различните функции може да **варира по устройства и време (**тактове**).** Фиг. **2.9.2.** Тя се дава и съвместимостта на последователните функции по устройства т.е. зависимостта им по ресурси

Анализ на закъснението при НЛКП

- Закъснението **(latency)** се представя от броя процесорни тактове *k* между две последователни инициирания на функции.
- Опита за повече от едно инициране едновременно на едно устройство е колизия, която се избягва чрез *планиране* (*диспечеризация*, *scheduling*) на последователността от инициирания.
- Когато закъснението е такова, че предизвиква колизия, то е *забранено закъснение*; трябва да се избере последователност от закъснения, така че да не предизвиква колизия. Пример за две забранени закъснения с карта на резервацията - фиг. 2.10.
- *Цикъл на закъснението* е последователност от закъснения, която се повтаря неопределено дълго. Интервалите между две последователни инициирания на функции в цикъла на закъснението може да са еднакви, (*константен цикъл*), но може и да са различни, при което се изчислява *средно закъснение*. Чрез коефициента на запълване на цикъла се получава ефективността на конвейера.

Инструкционен конвейер

- ИК е специализиран за обработка на последователните инструкции в машинния код чрез припокриване **(overlapping)**
- типичната инструкция минава през фазите извличане (от **cache** - обикновено **1** за цикъл), декодиране (установява функцията за изпълнение и необходимите ресурси - регистри, магистрали, устройства), издаване (резервира ресурсите чрез блокиране и извлича операндите от регисрите към устройствата), изпълнение (**1** или повече фази), записване (**writeback** - на резултатите в регистрите).
- архитектурата на процесорния конвейер Фиг. **2.11**
- преподреждане на инструкциите за по-голям коефициент на запълване на цикъла фиг. **2.11.1**

Обработка на преходите

- Конвейризацията се лимитира от зависимостта по данни и от инструкциите за преход
- Производителността при програма с 20%/10% вероятност за условен преход между последователните инструкции, 50% вероятност за изпълнение на условието (т.е. на прехода; статистически обаче повечето условни преходи - 60% -се изпълняват) и 8-фазен конвейр е 41%/25% по-малка отколкото производителността при програма, в която поне едната вероятност е 0. Затова при конвейрни процесори е желателно алгоритъмът да се кодира с минимум условни преходи.
- Предвиждането на преходите се използва за да се отложи прехода докато се изпълнят опр. брой инструкции, независими от условието на прехода. То може да бъде базирано на кода на програмата - статично или на историята на изпълнението - динамично

Архитектура на набора инструкции

Разграничават се класовете **RISC** и **CISC** по следните параметри:

- формат на инструкцията и на данните
- режими на адресация
- регистърно адресиране (регистри с общо назначение)
- управление на изпълнението на програмата **(flow control instructions)**

CISC

- класическа архитектура (първите процесори са ограничен набор инструкции)
- увеличения набор инструкции настъпва с микропрограмирането с промяната на **SWcost/HWcost** (първосигнална реакция на семантичната ножица между **HLL** и процесорните архитектури/машинните езици)
- параметри:
  - **120 – 350** инструкции с няколко формата на инструкциите и данните
  - **32 – 64** регистъра с общо предназначение
  - **4 – 16** режима на адресиране
  - голяма част от изразите на **HLL** са микрокодирани (т.е. имат съответствие в набора инструкции)
- скаларни **CISC** процесори – за операции върху скаларни данни; частична конвейризация поради:
  - зависимост по данни между последователните инструкции
  - ресурсен конфликт
- фиг. **2.14.**

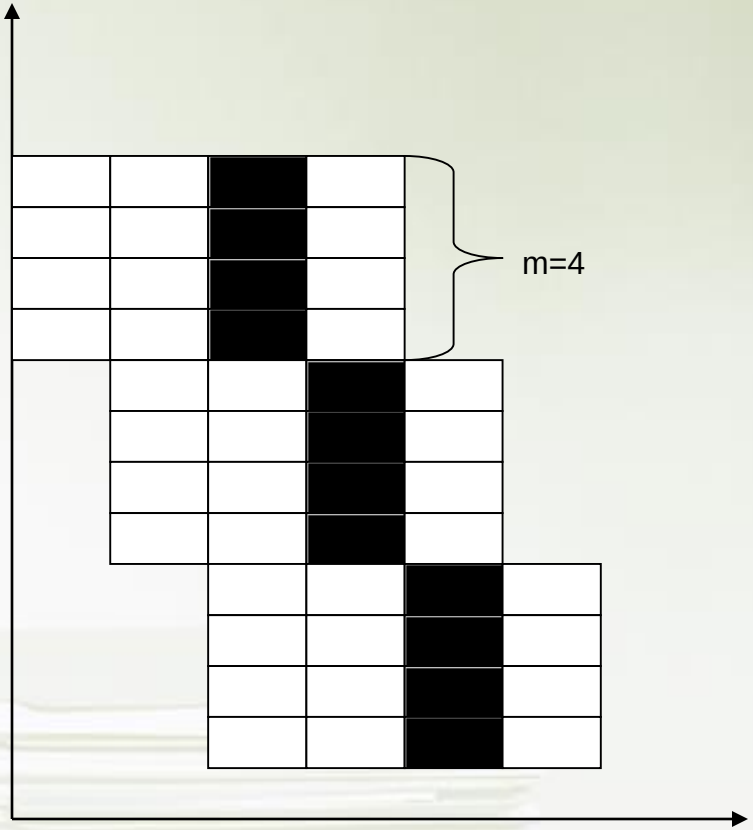
RISC

- **25%** от машинните инструкции кодират **90%** от **HLL** програмата и се изпълняват **95%** от процесорното време
- подходи за оптимизация:
  - трансформиране на микропоргмна памет в регистърен **cache**
  - **FPU** и други специализирани устройства на процесорния чип
  - суперскаларни процесори
  - броя на инструкциите е **< 100 –** с фиксиран формат (предимно регистър-регистър)
  - до **5** режима на адресиране, инструкциите са предимна от тип **load/store**
  - "регистърни фалове" **–** по **32+** вътрешни регистри за бързо превключване между процесите
  - едночипови, затова висока тактова честота **CR** и нисък **CPI** т.е. висок **MIPS** коефициент
  - скаларните **RISC** процесори са подобни на скаларните **CISC** но при еднаква тактова честота производителността може да е по-ниска поради по-малката плътност на кода
  - необходимост от ефективен компилатор за постигане на високо ниво конвейризация на ниво инструкция
  - суперскаларна **RISC** архитектура **–** фиг**. 2.15.**

Показатели	CISC	RISC скаларен
Брой инструкции	128-256-300	24-32
Формат на инструкции	16-64 бита, т.е. инструкцията е с плаваща дължина	32 бита, т.е. дължината е фиксирана
Формат на адреси	8-12 бита, различни начини на адресиране на операционната памет, къси/дълги	Регистър-регистър (много по-голям брой вътрешни регистри), регистерни файлове), 3-4 броя на регистерните формати
CPI брой просецорни тактове	8-20 процесорни такта, т.нар. инструкции с различна степен на сложност	3-6 процесорни такта, инструкциите са с фиксирана дължина – опростени
CM управляващ контролен модул	Базира се на микропрограмиране	С помоща на апаратна логика (АЛ) hardware control

Суперскаларни процесори (RISC и CISC)

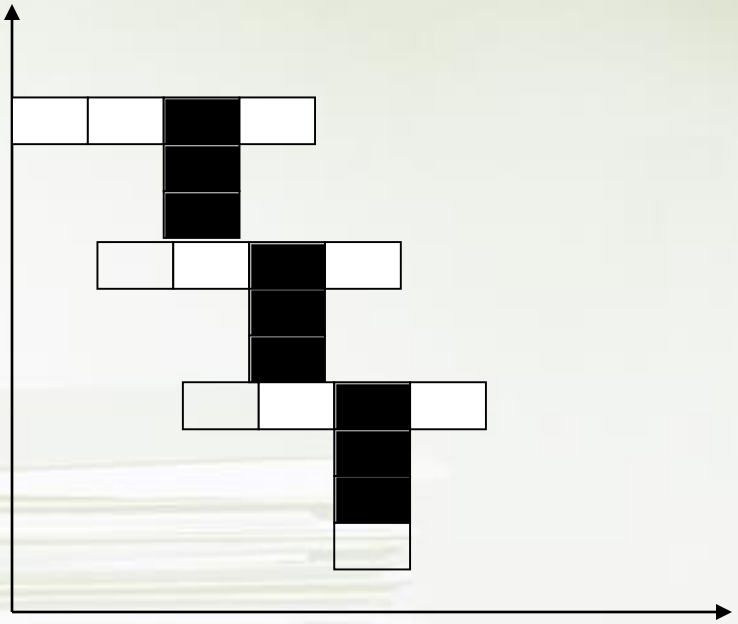
- Повече от **1** инструкция на такт поради наличието на няколко **(напр. 3)** инструкционни конвейера – съответно няколка резултата от всеки инструкционен цикъл
- модел **MIMD** инструкционно ниво
- разлика от векторните процесори, които реализират **SIMD** на инструкционно ниво
- паралелизма се реализира на инструкционно ниво – само между логически независими инструкции
- кратност на инструкцията **m = 2** до **5 (**при скаларните процесори **m = 1)**
- суперскаларен **RISC** процесор – фиг. **2.17.**



VLIW процесори

Комбинират концепцията за хоризонтално микрокодиране и суперскаларна архитектура:

- дълги инструкции (стотици битове), които задават по няколко операции над операндите
- различават се от суперскаларните процесори по
  - бързо и просто декодиране на инструкциите понеже една VLIW инструкция замества няколко суперскаларни
  - по-ниска плътност на кода но по-висок паралелизъм на инструкционно ниво (понеже инструкциите са с фиксиран формат който може да включва и ниезпълними операции, а суперскаларните операции са само изпълними)
  - непреносим обектен код понеже нивото на паралелизма при различните процесори е различно а е заложено в самата дълга инструкция (супрескаларните архитектури са портабелни със скаларните) – само за специализирани компютри
  - инструкционния паралелизъм се задава на етапа компилация – т.е. статичен; няма динамична диспечеризация и синхронизация
  - VLIW процесор – фиг. 2.18.



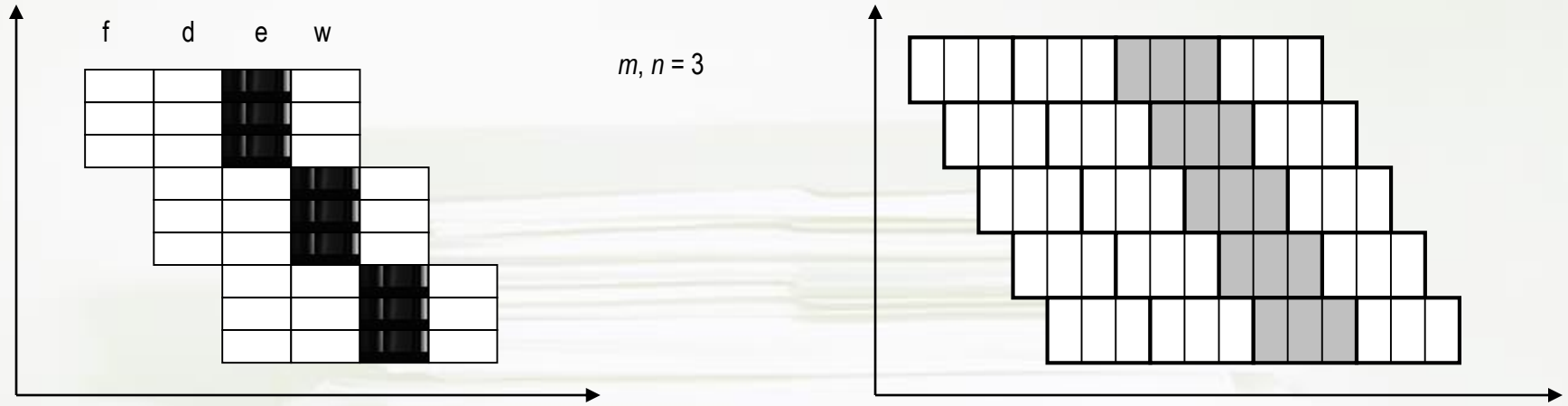
Векторни процесори

Специализирани копроцесори за векторни операции – операнодите в отделната инструкция са масив[и]

- дългите вектори (надвишаващи дължината на регистърните файлове) се сегментират
- инструкциите са тип
  - регистър-регистър – кратки; адресират се регистърни файлове
  - памет-памет – дълги (защото съдържат адреси от основната памет); те могат да обработват по-големи масиви с различна дължина
- типични векторни операции
  - зареждане на вектор от паметта на компютъра: $V_1 \leftarrow M_n$;
  - запис: $V_1 \rightarrow M_n$;
  - ескалиране: $S_1 \bullet V_1 \rightarrow V_2$;
  - векторна операция, при която и двете операнди са вектори и резултата е вектор: $V_1 \bullet V_2 \rightarrow V_3$;
  - редукция от векторни операнди и резултат скалар: $V_1 \bullet V_2 \rightarrow S_1$.
  - зареждане вектор-вектор: $\bullet V_1 \rightarrow V_2$.
  - редукция на единичен вектор: $\bullet V_1 \rightarrow S_1$.
  - аналогични инструкции от тип памет-памет – операндите са от вида $M_i(1 : n)$

Суперконвейрна архитектура

- При степен n цикъла на суперконвейера е $1/n$ от базовия цикъл на фазите. Фиг. 2.20.
- Закъснението за една операция е равно на базовия цикъл, но ILP е n. f
- $T(1, n) = k + (N\text{-}1)/n$
- $S(1, n) = \ldots \quad \rightarrow n$ за $N \rightarrow \infty$.
- Cray1: n=3.



Суперконвейрна суперскаларна архитектура

- Степента е (m, n) като m е кратността на едновременно издаваните инструкции (т.е. на супрескаларност) а n е кратността на супреконвейера (1/n от кратността на базовия цикъл между групите последователни инструкции). Фиг. 2.21. Закъснението за една операция е равно на базовия цикъл, но ILP е n.
- $T(m, n) = k + (N\text{-}m)/mn$
- $S(m, n) = \ldots \quad \rightarrow nm$ за $N \rightarrow \infty$.
- DEC Alpha: n=6, m=2.

Суперконвейирна суперскаларна архитектура – диаграма

SpeedupPipeline.xlsx - Excel

FILE | HOME | INSERT | PAGE LAYOUT | FORMULAS | DATA | REVIEW | VIEW

Файл с таблицата и диаграмата

J7 =C7+(D7-E7)/(E7*F7)

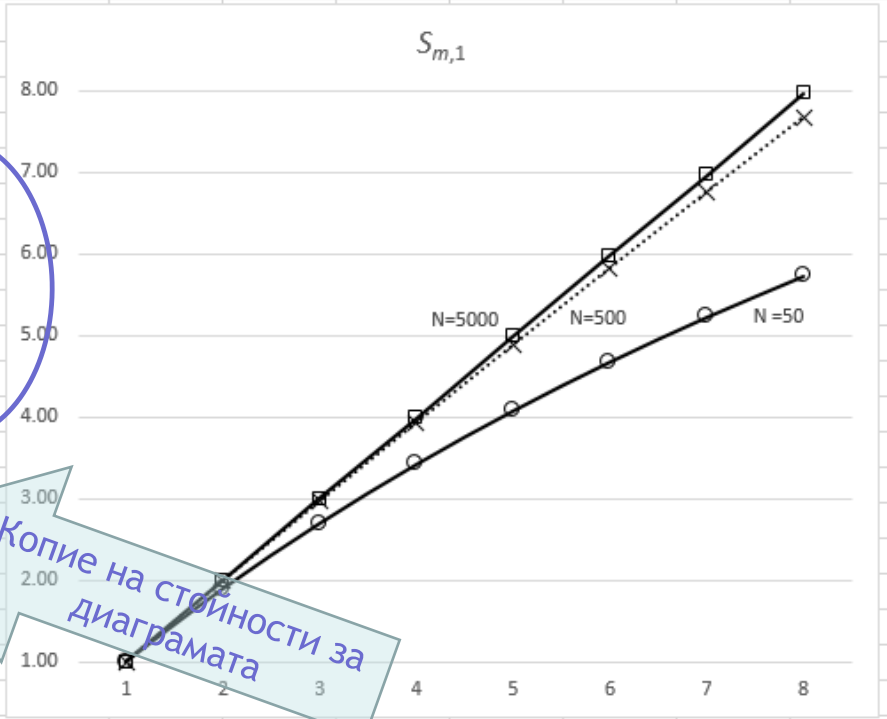
Formula Bar

Дефиниция на формулата

Дефиниция на стойности

	k=	N =	m	n	$T_{1,1}$	$T_{m,n}$	$S_{m,n}$
7	4	5000	1	1	5003	5003	1
8	4	5000	2	1	5003	2503	1.9988
9	4	5000	3	1	5003	1669.667	2.9964
10	4	5000	4	1	5003	1253	3.9928
11	4	5000	5	1	5003	1003	4.9880
12	4	5000	6	1	5003	836.3333	5.9821
13	4	5000	7	1	5003	717.2857	6.9749
14	4	5000	8	1	5003	628	7.9666

1	1.00	1.00	1.00
2	1.89	1.99	2.00
3	2.69	2.96	3.00
4	3.42	3.93	3.99
5	4.08	4.88	4.99
6	4.68	5.83	5.98
7	5.23	6.76	6.97
8	5.73	7.68	7.97



Копие на стойности за диаграмата

Технологии на процесорите

- Суперскаларните архитектури са по-подходящи за паралелизъм по данни - многократни операции се изпълняват конкурентно на няколко еднотипни устройства (блокове за изпълнение, портове към регистрови файлове ...). Затова необходимост от по голяма интеграция в чипа – **CMOS** технология. Комбинират се с **RISC** архитектура на процесорната дума.
- Суперконвейрите се базират на паралелзъм по управление, поради което съществено при тях е прилагането на устройства с висока тактова честота – т.е. **TTL** технология.

Intel Pentium

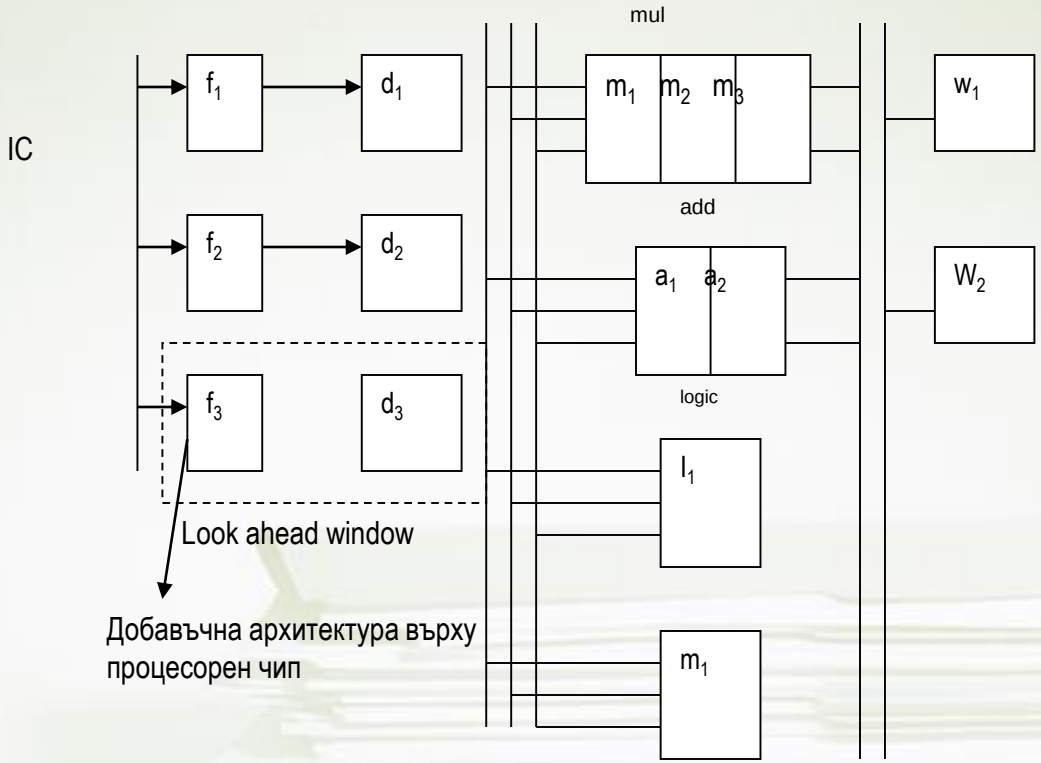
С въвеждането на Пентиум архитектурата **(1995)** Интел прилага предимствата на некласическа паралелна архитектура в производството на процесор, предназначен типично за масови компютри:

- суперскаларен процесор с ниво на инструкционния паралелизъм **m=2 (3** за P**4) –** едновременна обработка на **2** целочислени операнда по модела **MIMD (**когато последователните инструкции нямат зависимост по данни или управление**!);**
- всеки инструкционен конвейер се състои от **5** фази**:** извличане**,** декодиране**,** адресна генерация **(**типичен **CISC** процесор с много режими на адресация на ОП**),** изпълнение и запис
- изпълнението на последователни инструкции от всеки конвейер е със закъснение **1** фаза **(**извличане**):** два самостоятелни **I cache + D cache** по **8** кБ

…Intel Pentium

- за ефективно съчетаване на работата на конвейрите (т.е. за избягване на някои от случаите на конфликт) работата на двата инструкционни конвейра е «дефазирана» със стъпка от 1 фаза (първата фаза «извличане»)
- със същата цел Dcache е с двупортова организация – по един самостоятелен порт за всеки от инструкционните конвейри
- cache буферите са с асоциативна организация на достъпа: асоциативната памет има 32-байтов TLB с последните адреси така че търсенето на зарежданата страница става в 32 адреса (вместо 8К)
- планирането на активните страници в cache е по дисциплината LRU
- изискването за свързаност (кохерентност) между данните в cache и в ОП се постига чрез специален протокол – MESI – което позволява изграждането на мултипроцесорни архитектури (т. нар. симетричен мултипроцесинг – хомогенна мултипроцесорна архитектора с обща памет между процесорите)
- интегрирано FPU устройство с 8-фазов конвейер (извличане, декодиране, адресна генерация изпълнение, обработка мантиса, обработка експонента, обработка приближение и запис, който може да изпълнява и две FP инструкции едновременно (когато едната от тях е присвояване).

…Intel Pentium



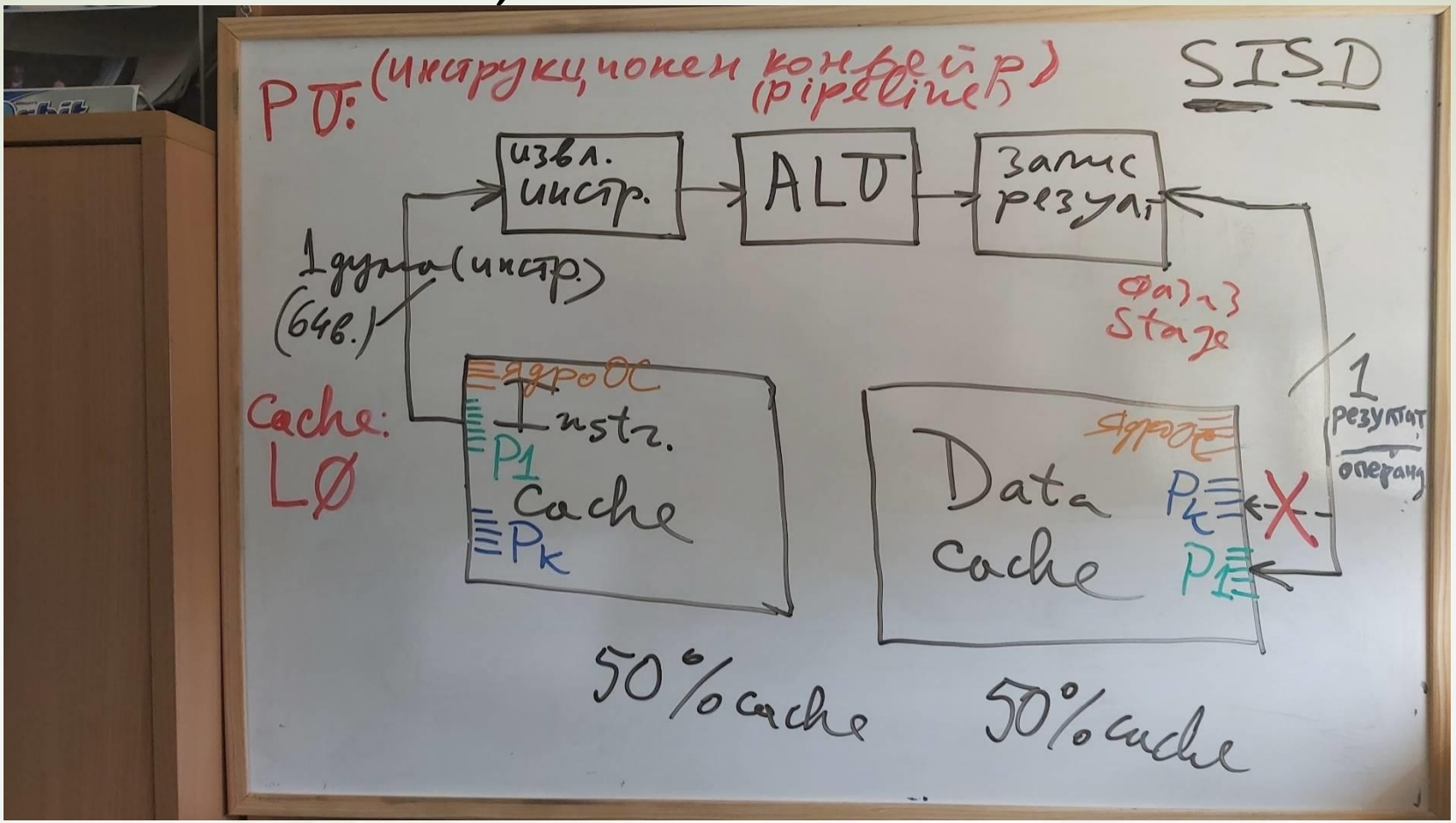
Добавъчна архитектура върху процесорен чип

Look ahead window – прозорец за предварителен преглед на инструкциите.
Mul – модул за изпълнение на умножението;
Add – модул за събиране;
Log – логически модул;

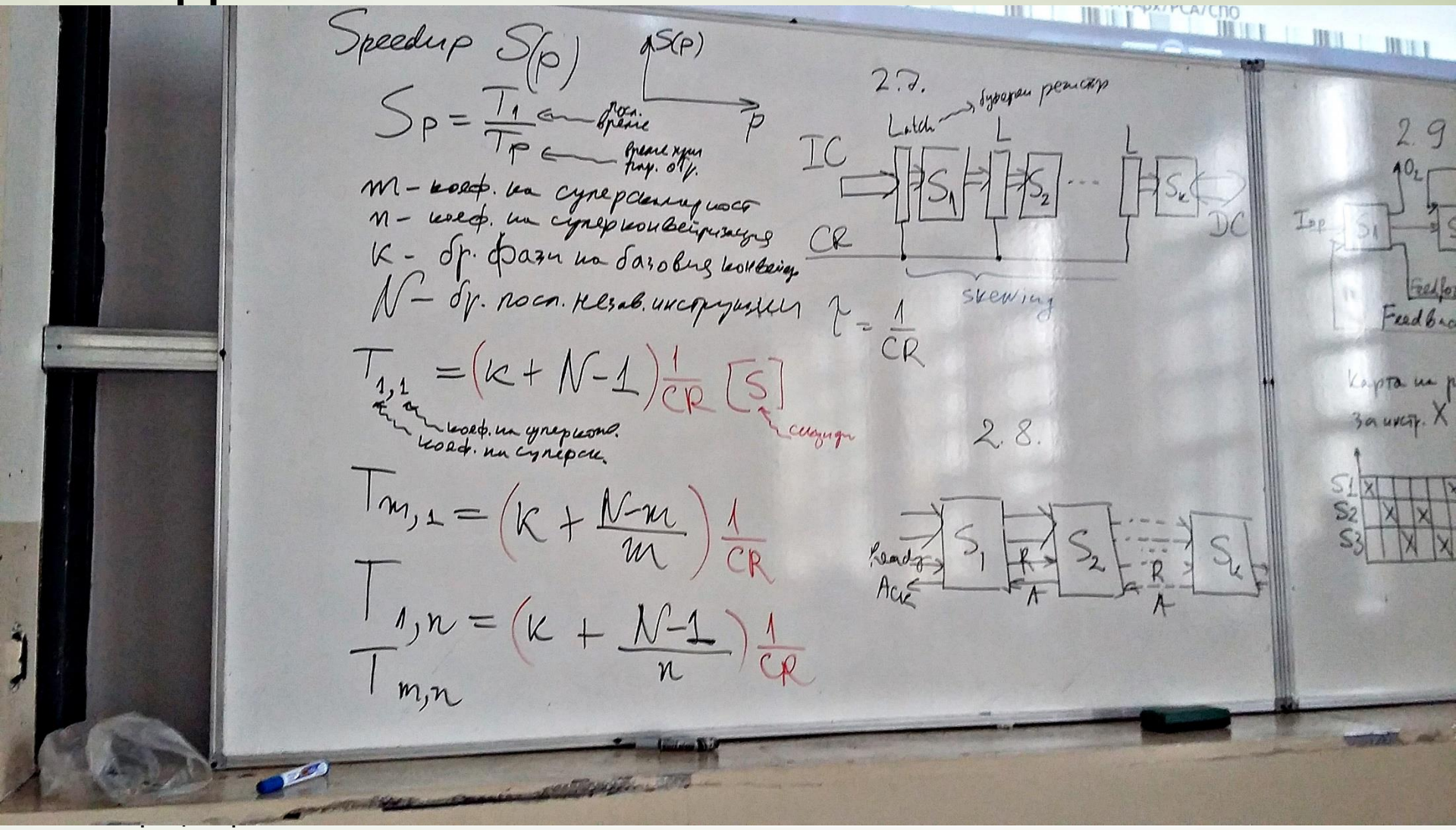
Intel Multicore

- Intel Core Microarchitecture технология прилага интегриране на машинната архитектура на симетричния мултипроцесинг в микропорцесор
- суперконвейрни суперскаларни ядра: 14 фази (коеф. на супреконвейрност 2); по 4 инструкции
- Две/четири независими ядра – NUMA мултипроцесинг с локализиран L1-cache за всяко ядро и общ L2-cache
- елементи на VLIW и RISC (едновременно!):
  - поддържа x86 CISC набор инструкции, като декодира част от тях до 2 и повече конвейрни микроинтрукции – RISC
  - т.нар. микрооперации обединяват няколко често срещани последователности от машинни инструкции за изпълнение като една инструкция – напр. проверка на стойност и преход по флаг са обединени в “микрооперацията” условен преход
- интегриран арбитраж на достъпа до L2-магистралата
- разширени възможности на управление на енергийното потребление и версии

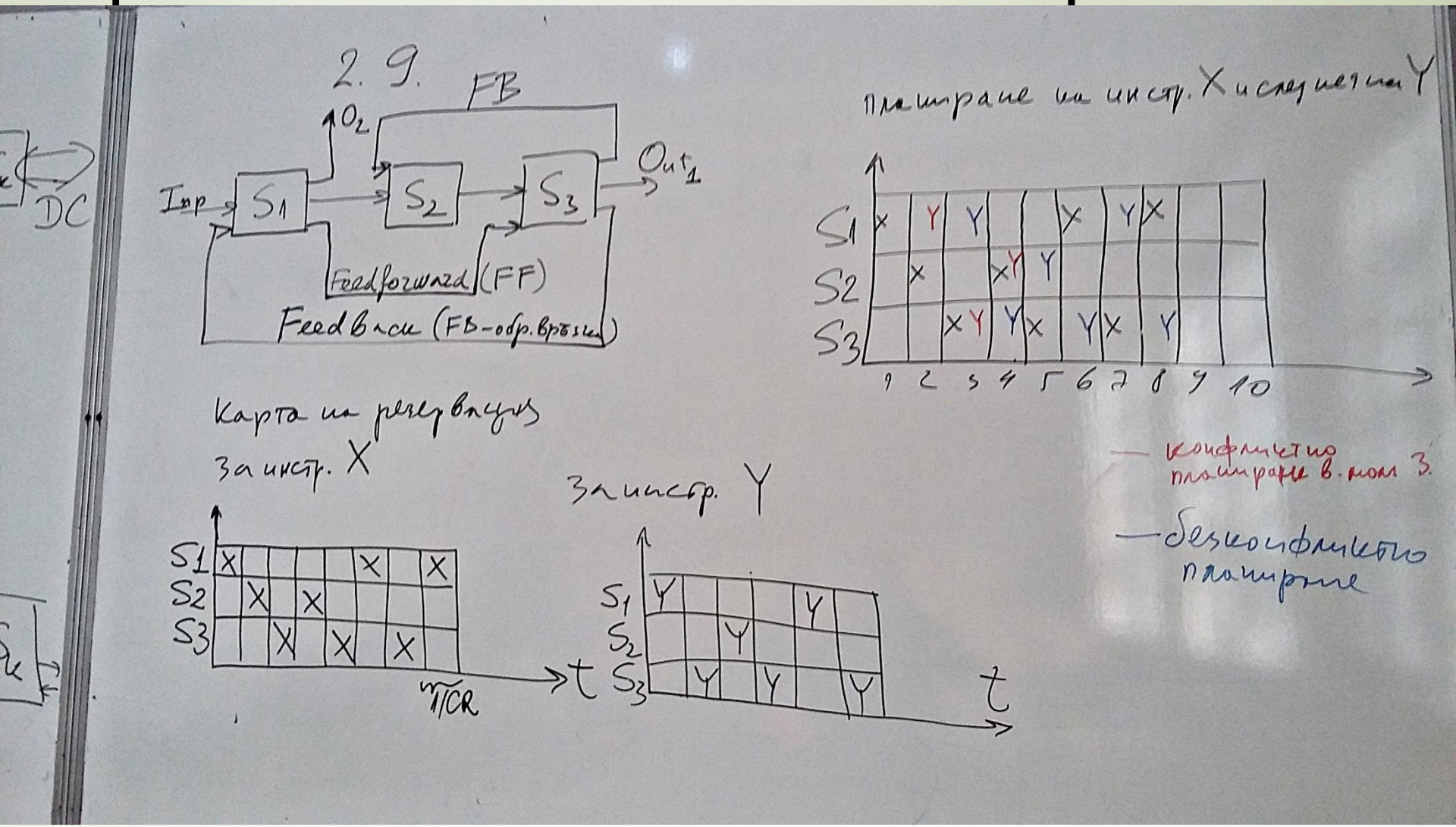
Фиг. 2.5 – инструкционен конвейер с 3 фази и двуделен кеш (Harvard architecture)



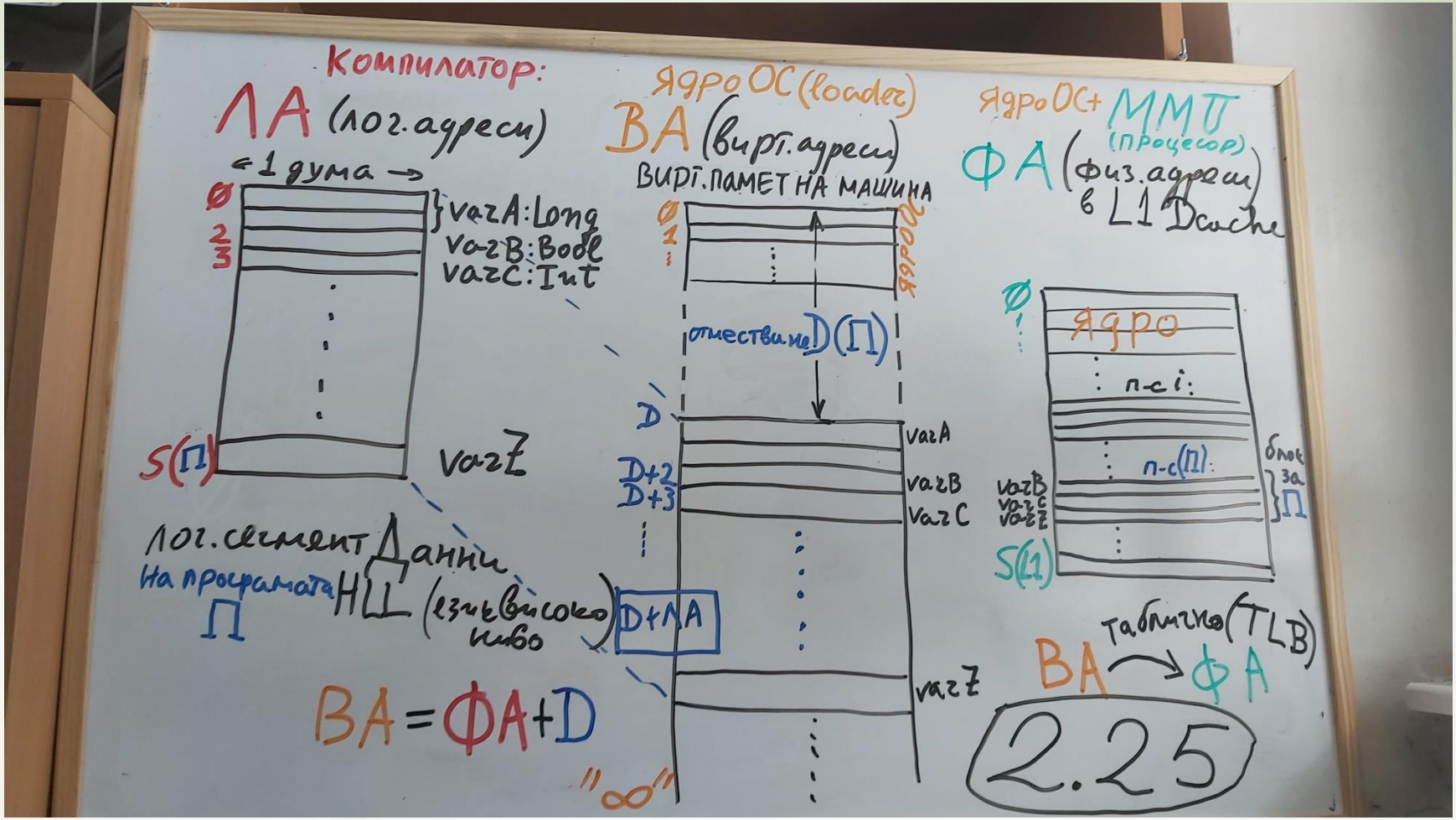
Фиг. 2.7 и 2.8; изчисление на модела в 2.21



фиг. 2.9 – нелинеен конвейер



фиг. 2.25 – генерация и преобразуване ЛА → ВА → ФА

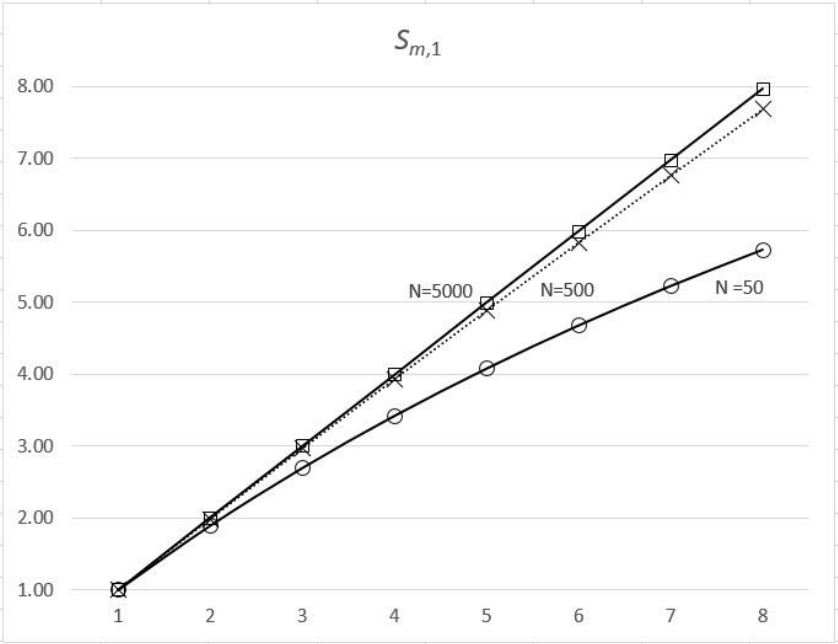


Числов анализ на модела в 2.21

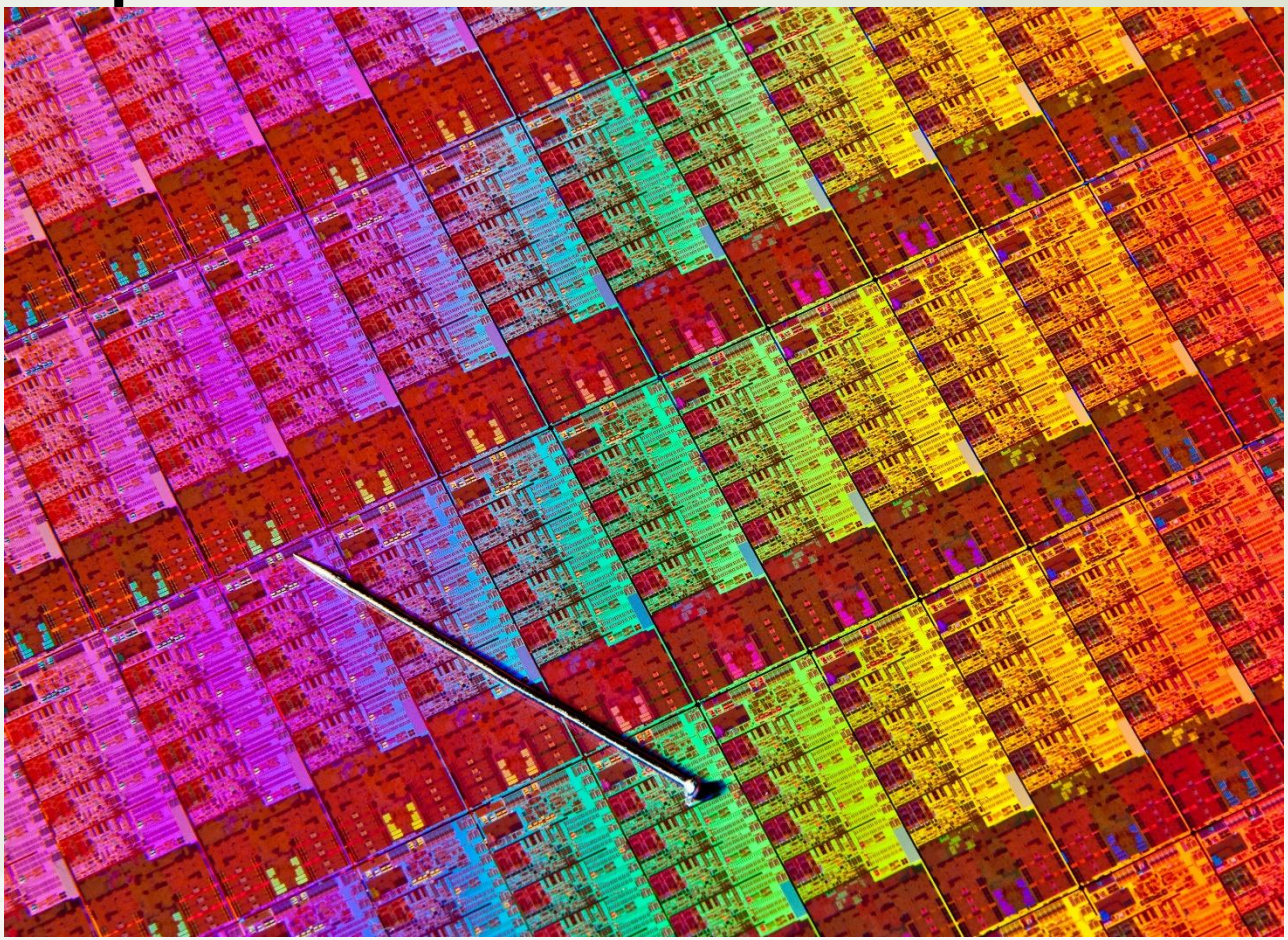
J7 | =C7+(D7-E7)/(E7*F7)

	k=	N =	m	n	$T_{1,1}$	$T_{m,n}$	$S_{m,n}$
7	**4**	**5000**	1	1	5003	5003	1
8	4	5000	2	1	5003	2503	1.9988
9	4	5000	3	1	5003	1669.667	2.9964
10	4	5000	4	1	5003	1253	3.9928
11	4	5000	5	1	5003	1003	4.9880
12	4	5000	6	1	5003	836.3333	5.9821
13	4	5000	7	1	5003	717.2857	6.9749
14	4	5000	8	1	5003	628	7.9666

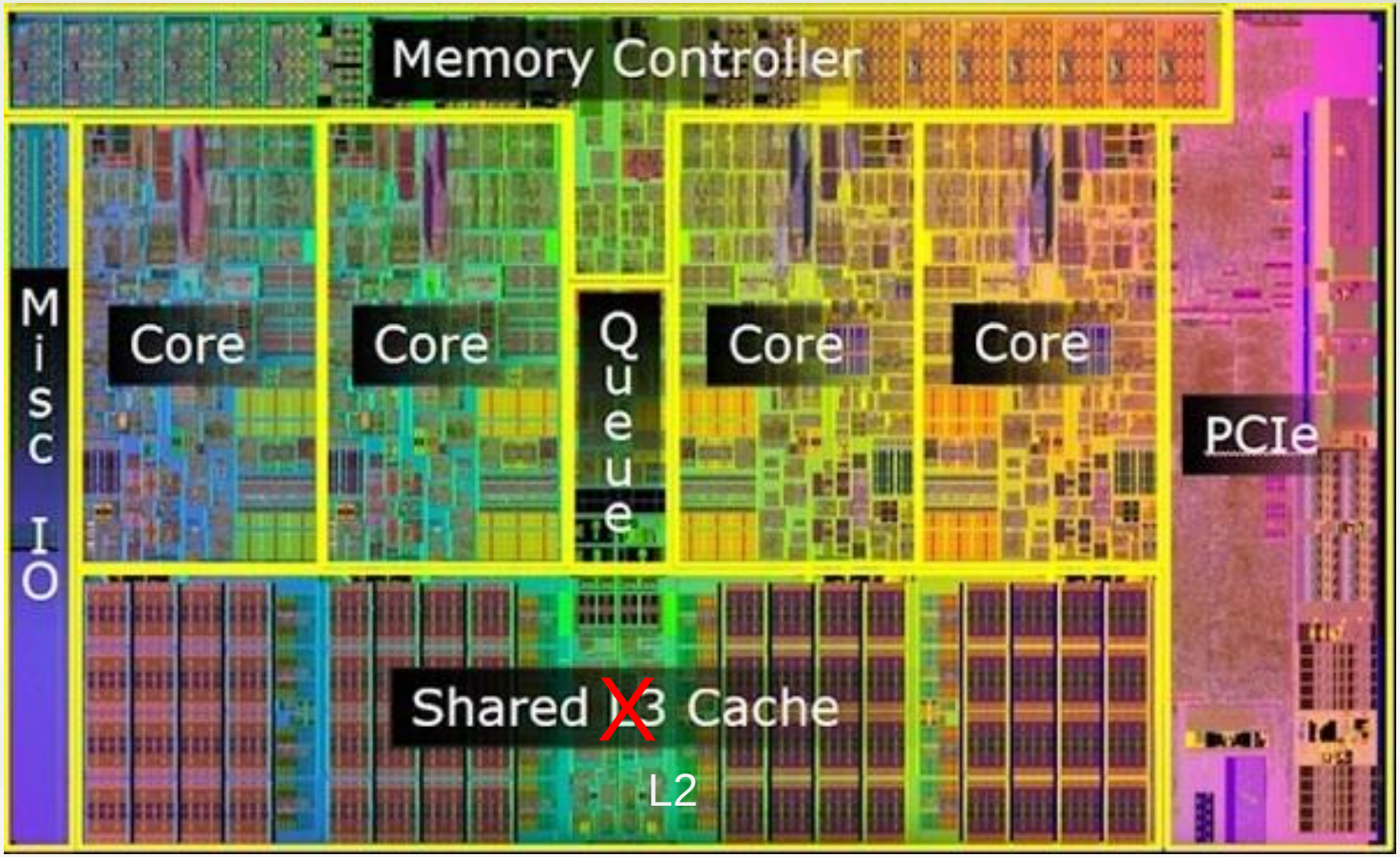
1	1.00	1.00	1.00
2	1.89	1.99	2.00
3	2.69	2.96	3.00
4	3.42	3.93	3.99
5	4.08	4.88	4.99
6	4.68	5.83	5.98
7	5.23	6.76	6.97
8	5.73	7.68	7.97



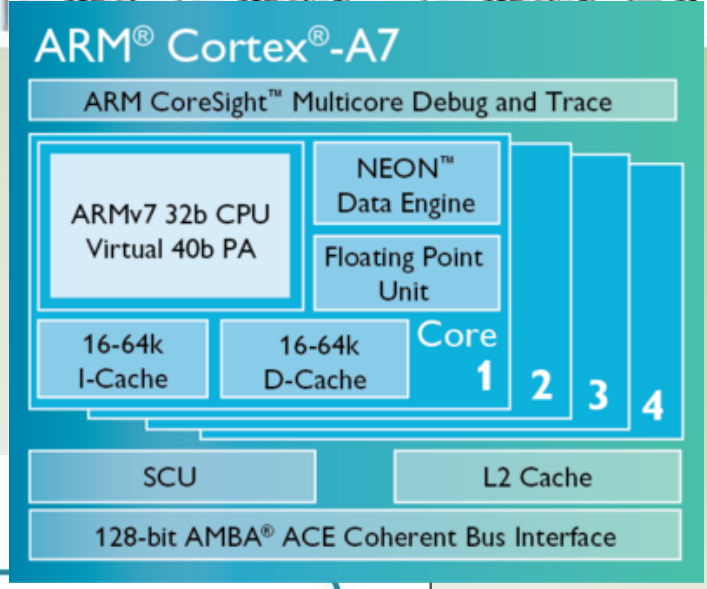
Hasswell Intel microarchiecture – силициев диск с десетки 4-ядрени процесори



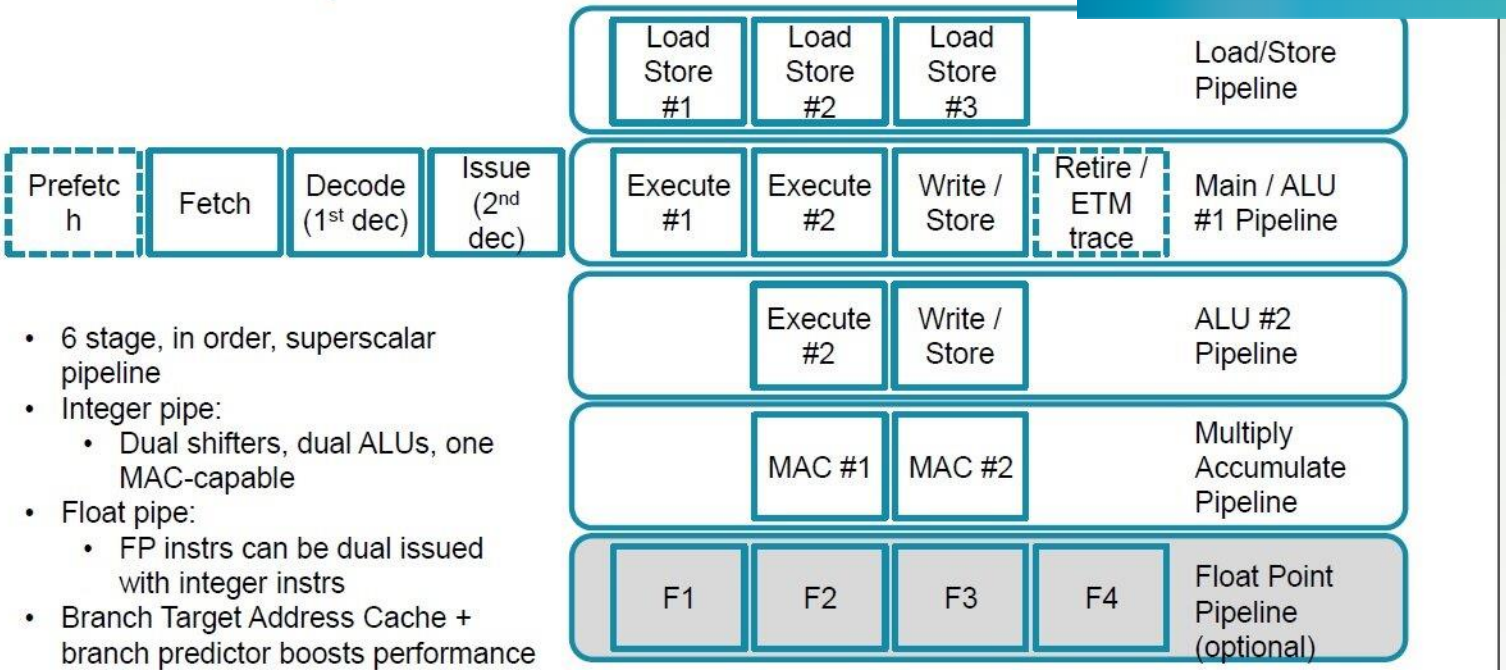
i5 – i7 кристал



RISC процесорите ARM



Cortex-M7 Pipeline

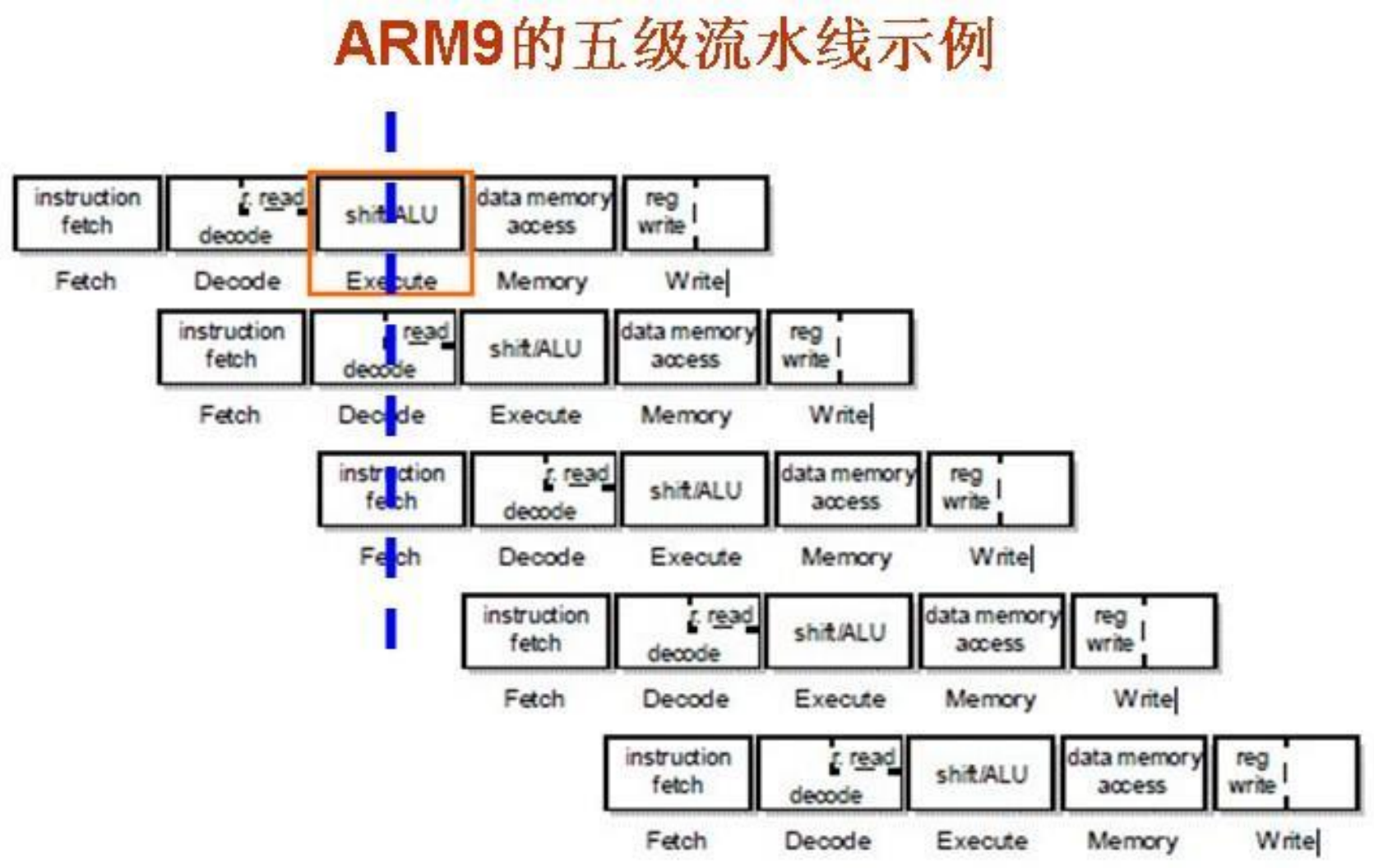


- 6 stage, in order, superscalar pipeline
- Integer pipe:
  - Dual shifters, dual ALUs, one MAC-capable
- Float pipe:
  - FP instrs can be dual issued with integer instrs
- Branch Target Address Cache + branch predictor boosts performance

35

ARM

RISC процесорите ARM – „карта на резервацията“ (времезиаграма) на инструкционния конвейер



Процесорна архитектура на NUMA-сървер