



南开大学

NANKAI UNIVERSITY, P.R. CHINA 1919

允公允能 日新月异



# 汇编语言与逆向技术

## 第6章 华为鲲鹏处理器体系结构

王志

zwang@nankai.edu.cn

南开大学 网络空间安全学院

2024-2025学年

大家是否使用过国产CPU?

- ☐ A 不知道有哪些国产的CPU
- ☐ B 听说过国产CPU，但是没有使用过
- ☐ C 使用过国产CPU
- ☐ D 在国产CPU上写过程序

提交



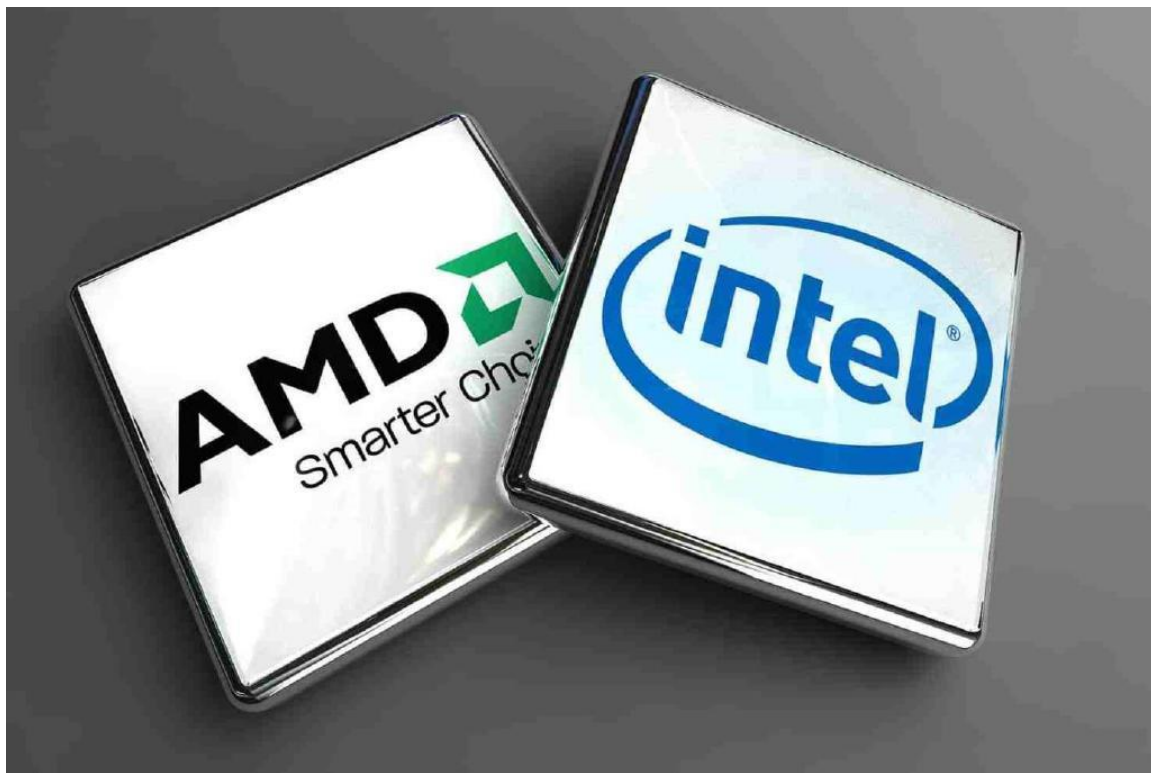
讨论题：大家知道哪些国产的CPU和操作系统？

作答



允公允能 日新月异

# 国外CPU vs. 国产CPU



南开大学  
Nankai University

思考题：我们为什么要发展国产CPU、为什么要学习国产CPU的编程？

作答

# 美国“对华芯片出口限制”

- 美国商务部宣布将华为及其70家公司列入出口管制实体名单之列
  - 如果没有美国政府的批准，华为将无法向美国企业购买元器件
  - Google、英飞凌、英特尔、高通、AMD、赛灵思等宣布停止和华为进行业务往来
  - 无线技术标准的Wi-Fi联盟、蓝牙技术联盟、制定SD存储卡标准的SD协会、PCIe标准组织PCI-SIG等将华为移出了会员名单。
- “**将中国冻结在落后的半导体发展状态**，并阻止中国公司获得美国的行业专业知识” --外交关系委员会（Council on Foreign Relations）高级研究员爱德华·奥尔登(Edward Alden)





允公允能 日新月异

# 鲲鹏CPU



南开大学  
Nankai University



允公允能 日新月异

# CPU架构

- x86、x64架构和ARM架构是最主流的两种CPU架构
  - x86、x64架构高度垄断
  - ARM架构开放授权模式
- 华为和飞腾已获得永久性Arm V8架构指令层级授权，自主研发CPU的设计根基。
  - 提升我国自主创新、安全可控、产业生态、芯片性能等综合实力，逐渐替代Intel等国外CPU芯片



南开大学  
Nankai University





允公允能 日新月异

# 本章知识点

- 服务器处理器体系结构
- 服务器处理器并行组织结构
- Intel处理器体系结构
- ARM处理器体系结构
- 华为鲲鹏处理器
- 基于ARMv8的处理器体系结构



南开大学  
Nankai University



南开大学

NANKAI UNIVERSITY, P.R. CHINA 1919

允公允能 日新月异

# 服务器处理器



允公允能 日新月异

**服务器（Server）：**专门用于提供服务的计算机系统，通常用于提供各种服务、存储和管理数据、执行计算任务，并在网络中起到协调和中继数据流量的作用。与个人计算机（Personal Computer，PC）相比，服务器通常具有**更高的性能和可靠性**。



南开大学  
Nankai University



允公允能 日新月异

# 服务器体系结构

**基本架构：**中央处理器、内存、存储设备、总线、输入/输出接口等基本组件

**操作系统：**如Windows、Linux或其他服务器操作系统，以管理硬件资源、提供服务和支持应用程序运行

**接口：**标准的通用接口和协议，例如通用序列汇流排协议（Universal Serial Bus, USB）、以太网协议（Ethernet）等，以便连接外部设备和网络



南开大学  
Nankai University

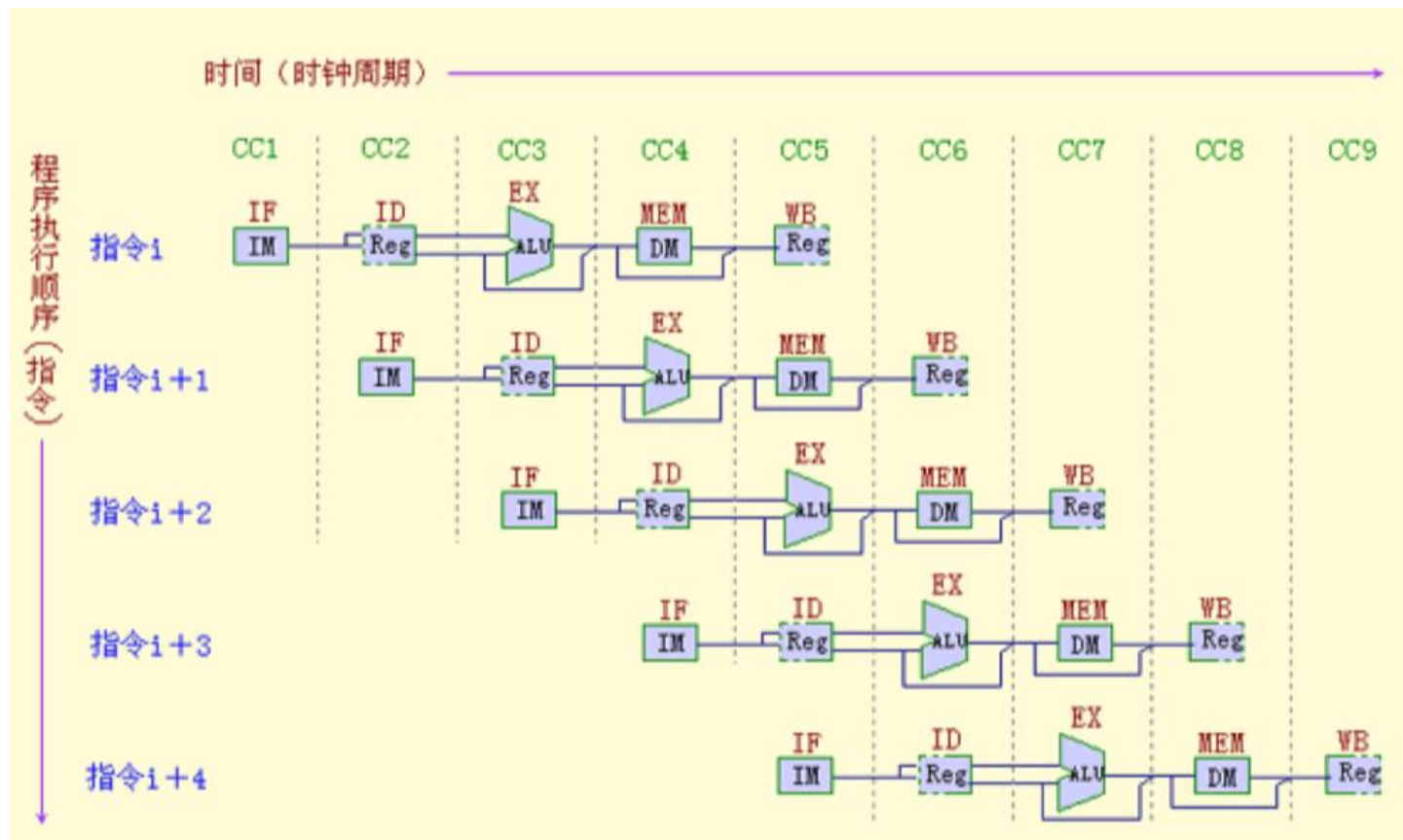
如果计算任务大于我们目前电脑的计算能力，有哪些方法可以提升计算能力？

作答

# 服务器处理器并行组织结构

## 1. 指令流水线

将指令处理过程分割为多个阶段，并在处理器中的不同部件上同时执行不同阶段的方法。

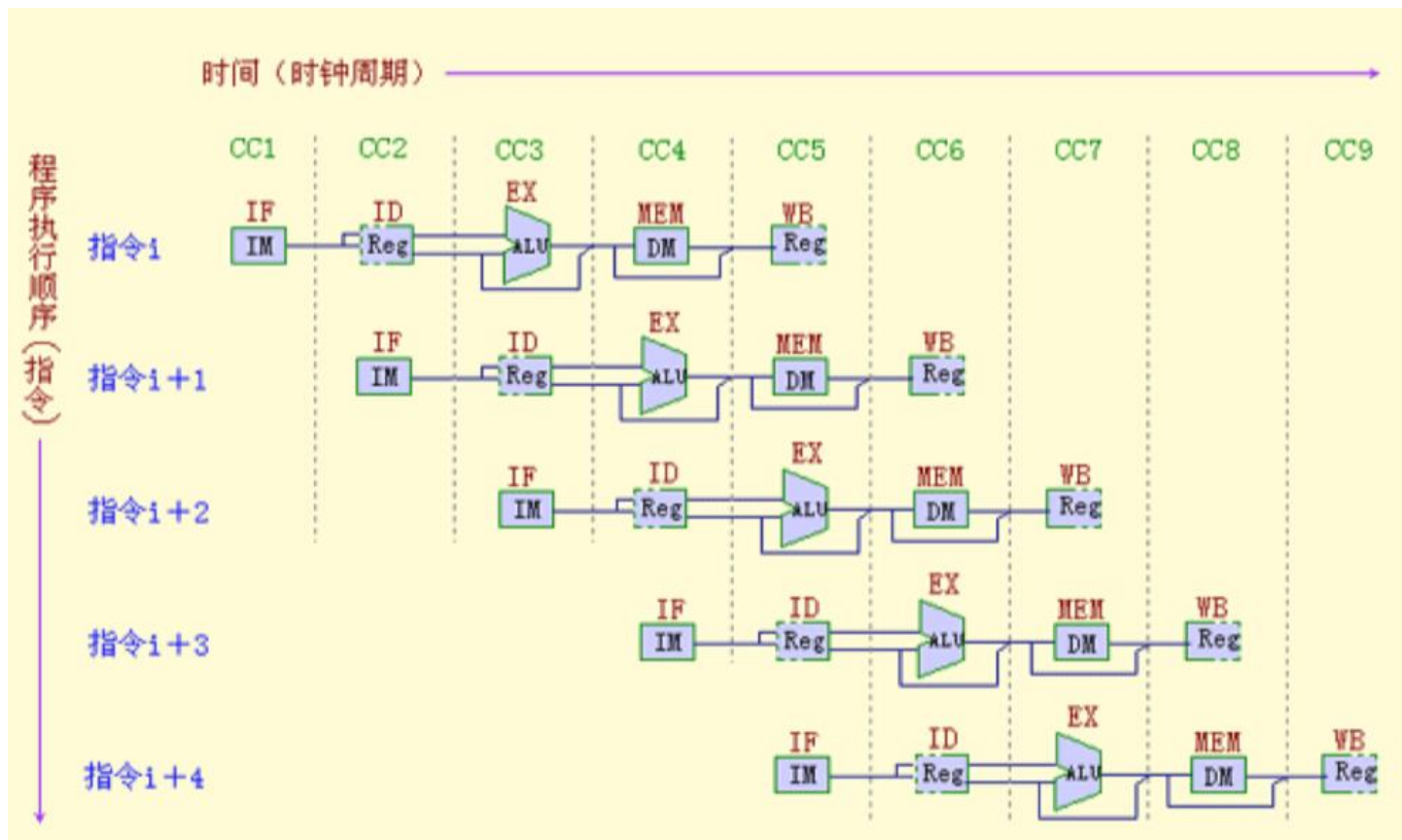




# 服务器处理器并行组织结构

指令流水线的常见阶段包括

取指令（IF - Instruction Fetch）、  
译码（ID - Instruction Decode）、  
执行（EX - Execute）、  
访存（MEM - Memory）、  
写回（WB - Write Back）等。



# 服务器处理器并行组织结构

## 2. 多处理器系统与多计算机系统

**多处理器系统**是一种在单个计算机系统中包含多个处理器的架构。这些处理器共享同一物理内存和其他系统资源，可以同时执行不同的指令。

**多计算机系统**是由多个独立的计算机系统（节点或处理器）组成的，这些计算机通过消息传递的方式进行通信。每个计算机系统都有自己的物理内存和资源。





允公允能 日新月异

# 服务器处理器并行组织结构

## 3. 多线程处理器

多个指令流能共享同一个支持多线程的处理器。使用同时多线程技术（Simultaneous Multi-Threading, SMT），允许在一个处理器的时钟周期内执行来自多个线程的指令。

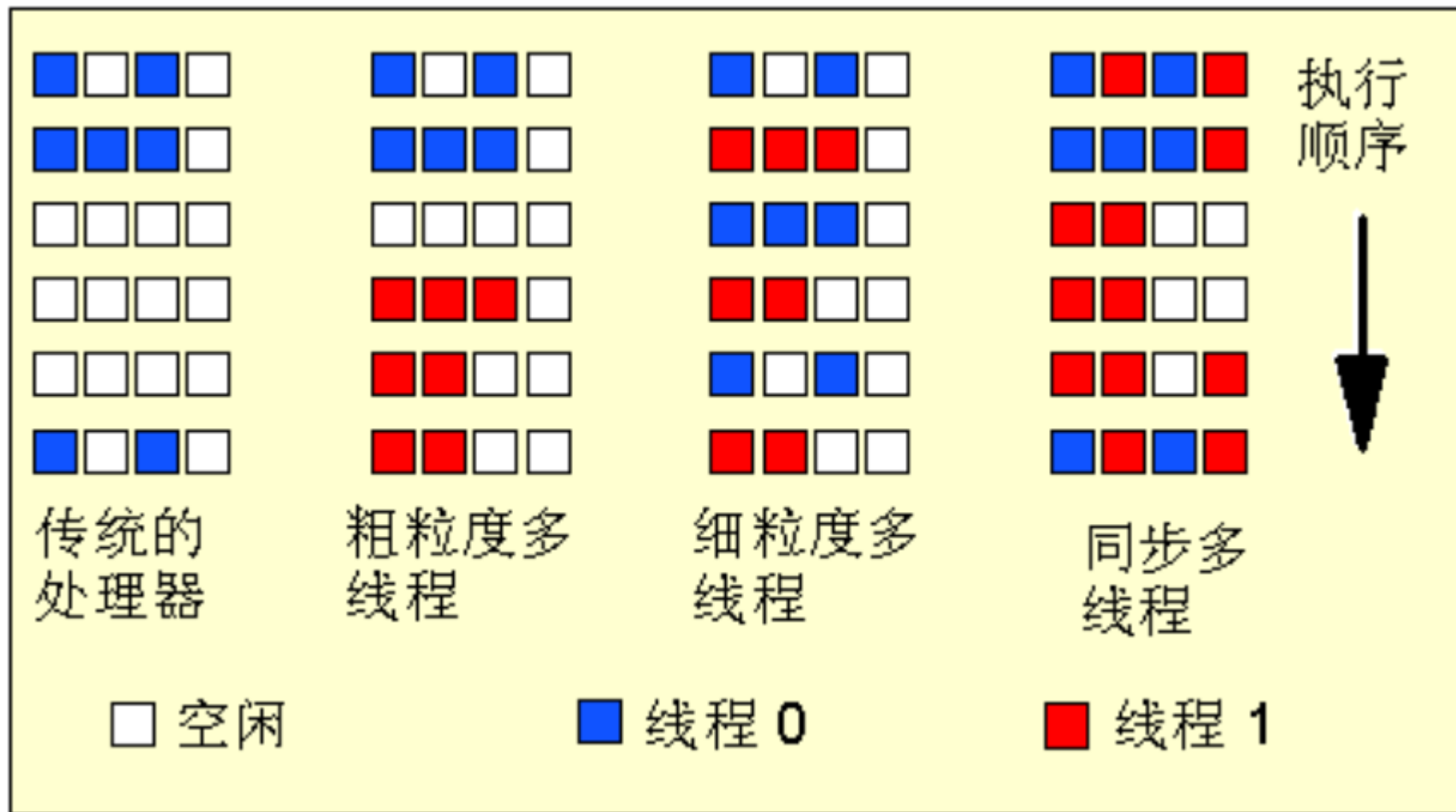


南开大学  
Nankai University



允公允能 日新月异

# 服务器处理器并行组织结构





允公允能 日新月异

# 服务器处理器并行组织结构

## 4. 多核处理器

多核处理器也称片上多处理器（Chip Multi-Processor, CMP），是一种在单个芯片或芯片组上集成多个处理器核心的计算机架构，多个处理器核心被集成到同一物理芯片上，以提供更高的并行性和整体性能。

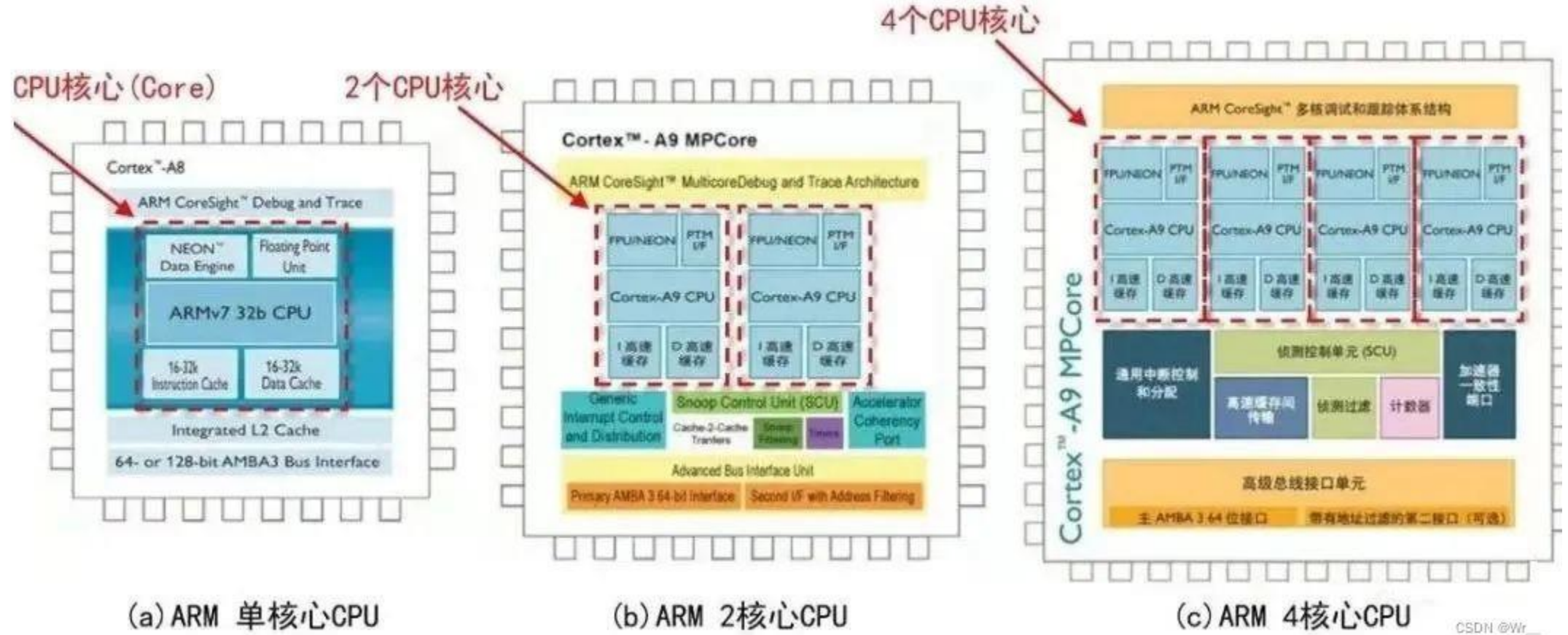


南开大学  
Nankai University





# 服务器处理器并行组织结构



CSDN @W...



下列哪些是服务器所使用的并行技术？

- ☒ A 指令流水线
- ☒ B 多线程处理器 (SMT)
- ☒ C 多核处理器 (CMP)
- ☒ D 多处理器和多计算机系统



南开大学

NANKAI UNIVERSITY, P.R. CHINA 1919

允公允能 日新月异



# 处理器体系结构



允公允能 日新月异

# Intel处理器体系结构



## 32位处理器体系架构

**IA-32**（Intel Architecture, 32-bit），也称为**x86**，是Intel公司推出的32位指令集架构，广泛应用于Windows、Linux等操作系统。

- 8086处理器和8088处理器是IA-32架构的早期16位处理器，具有16位寄存器
- Intel 386处理器是IA-32架构家族的第一款32位处理器，引入了32位寄存器，用于保存操作数和寻址
- 从1993年诞生的奔腾（Pentium）处理器开始，IA-32开始采用RISC处理器的设计思想，采用超标量流水技术，允许两条指令同时执行



南开大学  
Nankai University



允公允能 日新月异

# Intel处理器体系结构

## 64位处理器体系架构

1994年，Intel和惠普公司合作设计**IA-64**（Intel Architecture, 64-bit）架构，后命名为**安腾架构**。安腾架构作为一种64位处理器架构，支持64位的寻址和数据处理，并完全放弃了既有的CISC架构，转向了RISC架构。

2000年，AMD公司首次提出了一种新的64位扩展架构，称为**AMD64**。该架构不仅提供了64位的寻址和数据处理能力，还保持了对32位x86指令集的完全兼容，所以该架构也被称为**x86-64**架构。Intel于2004年发布了首款支持x86-64的处理器。



南开大学  
Nankai University



允公允能 日新月异

# ARM处理器体系结构

**ARM处理器：** ARM体系结构的发展经历了ARMv1~ARMv9

**ARMv8之前：** 主要使用的是ARM指令集、Thumb指令集和Thumb-2指令集。

- ARM指令集——基于32位的原始ARM体系结构的指令集
- Thumb指令集——ARM指令集的一种压缩形式，通过使用16位的指令来减小程序的体积
- Thumb-2——对Thumb指令集的扩展，引入了一些32位的指令



南开大学  
Nankai University



允公允能 日新月异

# ARM处理器体系结构

**1. ARM指令集：** ARM体系结构的发展经历了ARMv1~ARMv9

**ARMv8之后：** ARM引入了新的64位指令集，并重新命名了原有的指令集。

- A64指令集——ARMv8-A架构中64位执行状态（AArch64）下的指令集
- T32和A32指令集——ARMv8-A架构中32位执行状态（AArch32）下的两个指令集
  - A32与之前的32位ARM架构兼容
  - T32是Thumb-2指令集的一个变种。



南开大学  
Nankai University

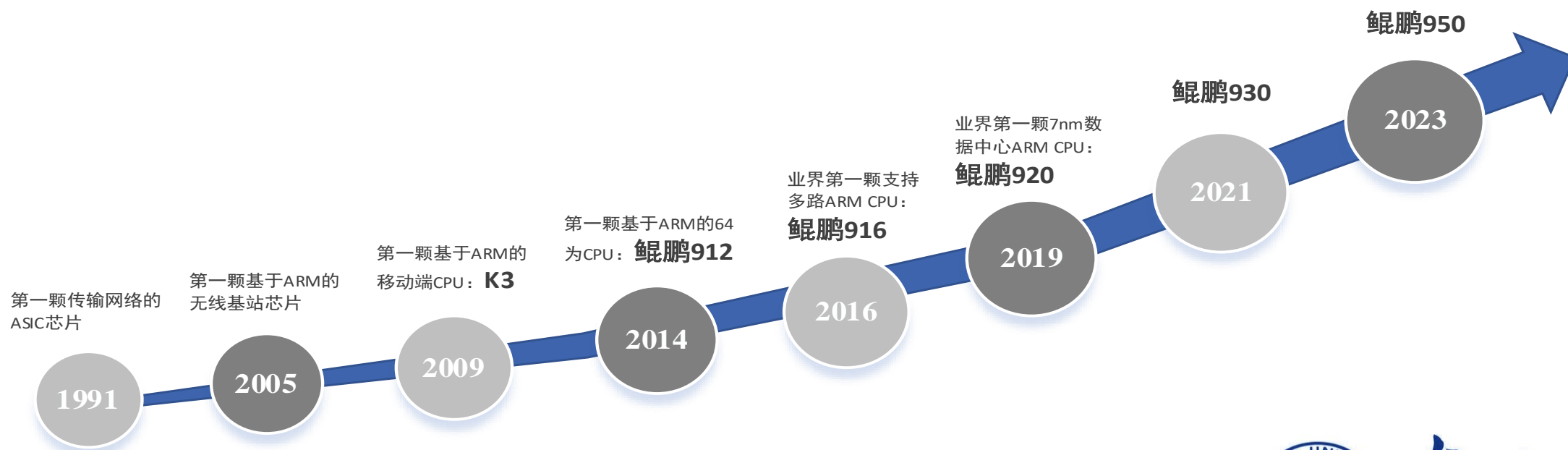


	架构版本	发布年份	数据/地址位宽	主要特征	ARM处理器产品	第三方处理器产品
	ARMv1	1985	32/26	<ul style="list-style-type: none"> <li>32位指令宽度；</li> <li>16个32位整数寄存器</li> <li>仅有基本的数据处理指令，无乘除指令</li> </ul>	ARM1（未商用）	
	ARMv2	1986	32/26	<ul style="list-style-type: none"> <li>增加了乘法指令；</li> <li>支持协处理器</li> </ul>	ARM2 ARM3 ARM2As	Amber
	ARMv3	1990	32/32	<ul style="list-style-type: none"> <li>地址线扩展至32位；</li> <li>增加两种处理器模式；</li> <li>支持存储管理单元（MMU）</li> </ul>	ARM6系列 ARM7系列	
	ARMv4 ARMv4-T	1993	32	<ul style="list-style-type: none"> <li>增加半字加载/存储指令；</li> <li>Armv4-T增加16位；Thumb指令集</li> </ul>	ARM8 ARM7TDMI ARM9TDMI	StrongARM
	ARMv5 ARMv5-TE ARMv5-TEJ	1998	32	<ul style="list-style-type: none"> <li>改进了ARM与Thumb指令集的互操作性；</li> <li>扩充DSP指令；</li> <li>支持浮点运算VFP20</li> </ul>	ARM7EJ ARM9E ARM9EJ ARM10E	XScale
	ARMv6	2001	32	<ul style="list-style-type: none"> <li>增加对Thumb-2指令集的支持；</li> <li>提供SIMD指令</li> </ul>	ARM11系列	
	ARMv6-M	2004	32	<ul style="list-style-type: none"> <li>专注于嵌入式微控制器，裁剪了指令集</li> </ul>	Cortex M0/M0+/M1	
	ARMv7-M	2004	32	<ul style="list-style-type: none"> <li>仅支持Thumb-2指令集的子集</li> </ul>	Cortex M3/m4	STM32系列
	ARMv7-R ARMv7-A	2004	32	<ul style="list-style-type: none"> <li>支持高级SIMD技术；</li> <li>支持矢量浮点运算VFP30</li> <li>配置32个64位寄存器</li> <li>支持1TB物理地址空间</li> <li>ARMv7-R支持存储保护，ARMv7-A支持虚拟存储器</li> </ul>	Cortex R4/R5/R7 Cortex A5/A7/A8 /A9/A15/A17	
	ARMv8-A	2011	64/32	<ul style="list-style-type: none"> <li>使用64位通用寄存器</li> <li>支持64位处理和扩展的虚拟寻址</li> <li>支持指令集A64、A32、T32</li> <li>兼容ARMv7</li> <li>支持虚拟化技术</li> </ul>	Cortex A53/A57/A72 /A73/A76 Neoverse NI	华为鲲鹏920系列 Mavell/Cavium ThunderX系列 三星ExynosM系列 苹果Ax系列 亚马逊Gravito 富士通A64FXn
	ARMv9-A	2021	64	<ul style="list-style-type: none"> <li>引入了SVE2用于支持计算密集型应用</li> <li>引入了新的安全内核级别Secure EL2</li> <li>硬件安全增强</li> <li>高级的虚拟化支持</li> </ul>	Cortex A510/A710 /A715 Cortex X2/X3 Neoverse E2 Neoverse N2 Neoverse V2	

# ARM处理器体系结构

## 华为鲲鹏处理器

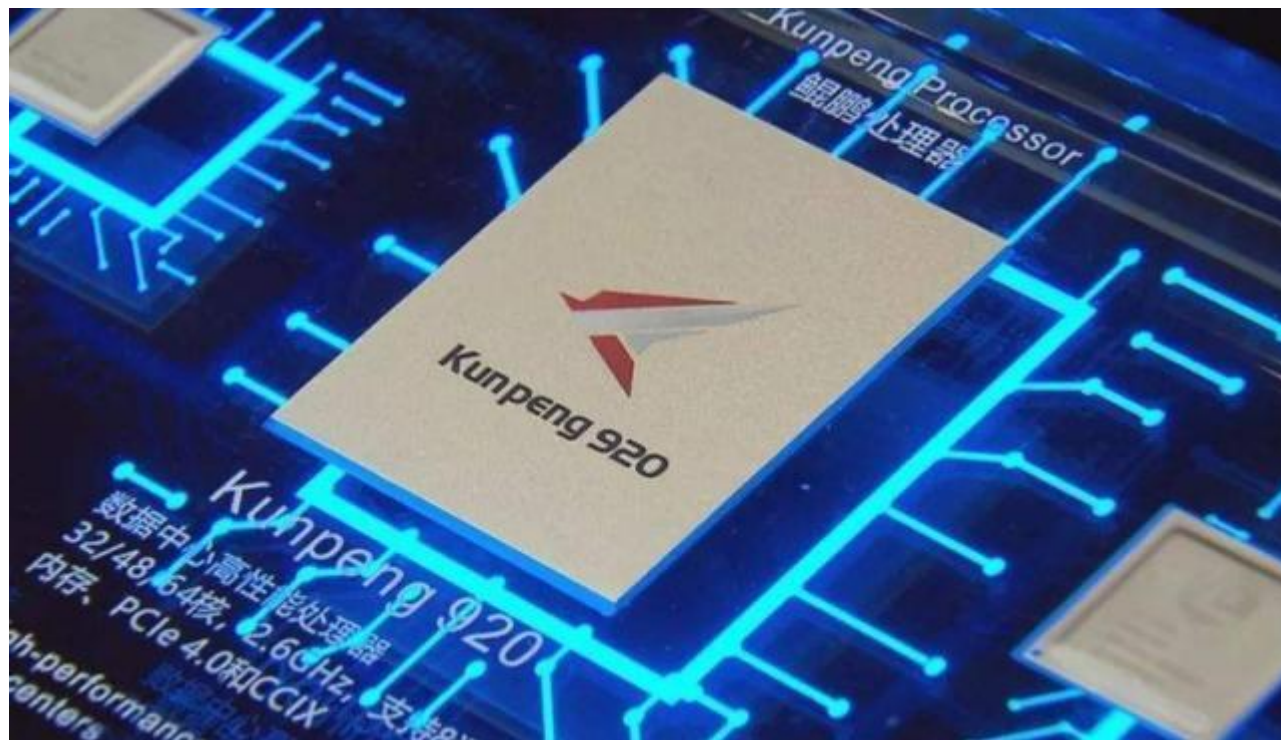
2004年，华为公司开始基于ARM技术自研芯片。目前，华为自主研发的处理器系列产品已经覆盖“算、存、传、管、智”（计算、存储、传输、管理、人工智能）五个应用领域。



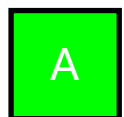
# ARM处理器体系结构

## 华为鲲鹏处理器

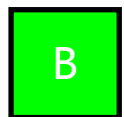
2019年诞生的**鲲鹏920处理器**是业界首款基于7nm工艺的数据中心ARM处理器，具有高性能、高吞吐率、高集成度和高能效的特点。



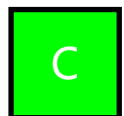
处理器有哪些架构?



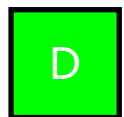
x86



x86-64



ARM



IA-64

提交

64位的ARM指令集是？

- ☐ A Thumb指令集
- ☒ B A64指令集
- ☒ C T32指令集
- ☐ D A32指令集

提交



南开大学

NANKAI UNIVERSITY, P.R. CHINA 1919

允公允能 日新月异



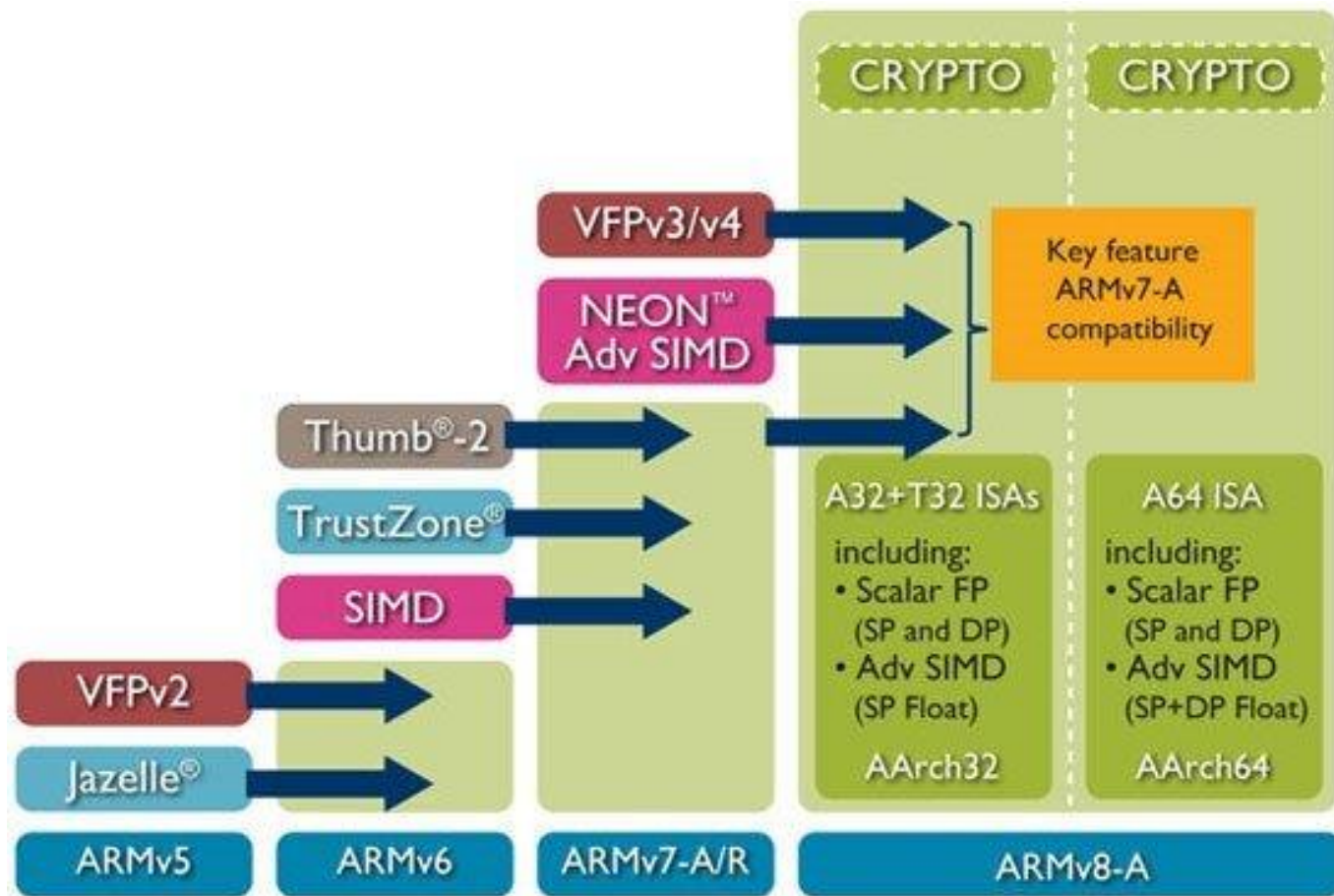
# ARMv8处理器体系结构





允公允能 日新月异

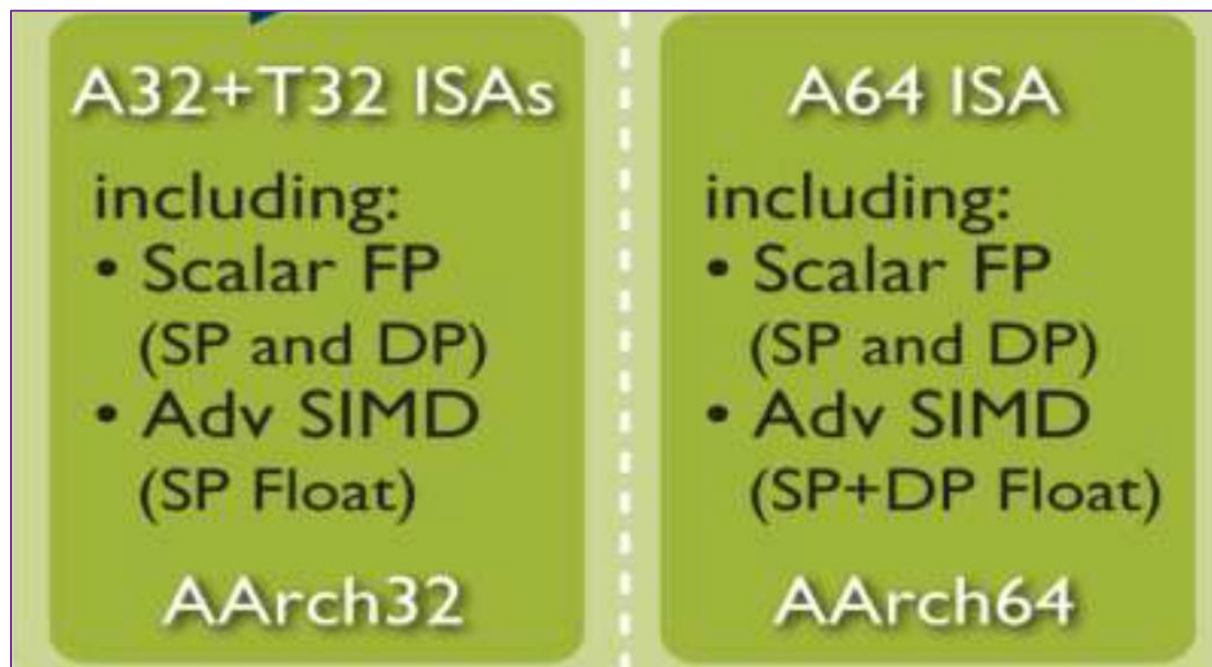
# ARMv8的发展过程





# ARMv8-A的2种执行状态（Execution States）

- AArch32架构
- AArch64架构
- 指令集、寄存器不同





允公允能 日新月异

# ARMv8支持的指令集

- AArch64执行状态下
  - 只能使用A64指令集：32位等长指令字
- AArch32执行状态下
  - 可以使用A32指令集：32位等长指令字
  - 或T32指令集：16位和32位可变长指令字





# ARMv8-A的执行状态（Execution States）

- AArch32（32位架构）
  - 支持A32（ARM）和T32（Thumb）两种指令集
  - 提供13个32位通用寄存器（R0-R12）
  - 提供1个32位程序计数器PC、1个栈指针SP、1个链接寄存器LR
  - 提供32个64位寄存器 用于增强SIMD向量和标量浮点运算
  - 一组处理状态PSTATE参数用于保存处理单元状态





允公允能 日新月异

# ARMv8-A的执行状态

- AArch64（64位架构）
  - 支持单一的A64指令集
    - 固定长度32bits的指令集
  - 提供31个64bit通用寄存器（X0-X30）
  - 提供1个64位程序计数器PC、若干堆栈指针SP寄存器、若干异常链接寄存器ELR
  - 提供32个128位寄存器用于增强SIMD向量和标量浮点运算
  - 一组处理状态PSTATE参数用于保存处理单元状态







# ARMv8-A的执行状态

Execution State	Note
AArch32	提供13个32bit通用寄存器R0-R12，一个32bit PC指针 (R15)、堆栈指针SP (R13)、链接寄存器LR (R14)
	提供一个32bit异常链接寄存器ELR, 用于Hyp mode下的异常返回
	提供32个64bit SIMD向量和标量floating-point支持
	提供两个指令集A32 ( 32bit ) 、T32 ( 16/32bit )
	兼容ARMv7的异常模型
	协处理器只支持CP10\CP11\CP14\CP15
AArch64	提供31个64bit通用寄存器X0-X30 ( W0-W30 ) ，其中X30是程序链接寄存器LR
	提供一个64bit PC指针、堆栈指针SPx 、异常链接寄存器ELRx
	提供32个128bit SIMD向量和标量floating-point支持
	定义ARMv8异常等级ELx ( x<4 ) ,x越大等级越高，权限越大
	定义一组PE state寄存器PSTATE ( NZCV/DAIF/CurrentEL/SPSel等 ) ，用于保存PE当前的状态信息
	没有协处理器概念





# ARMv8支持的数据类型

四字（Quadword）	128位
双字节（ <b>DoubleWord</b> ）	64位
字（ <b>Word</b> ）	在ARM体系结构中，字的长度为 <b>32位</b>
半字（Half-Word）	在ARM体系结构中，半字的长度为 <b>16位</b>
字节（Byte）	在ARM体系结构中，字节的长度为8位





允公允能 日新月异

# ARMv8支持的数据类型

- 三种浮点数据类型
  - 半精度（Half-precision）浮点数据
  - 单精度（Single-precision）浮点数据
  - 双精度（Double-precision）浮点数据



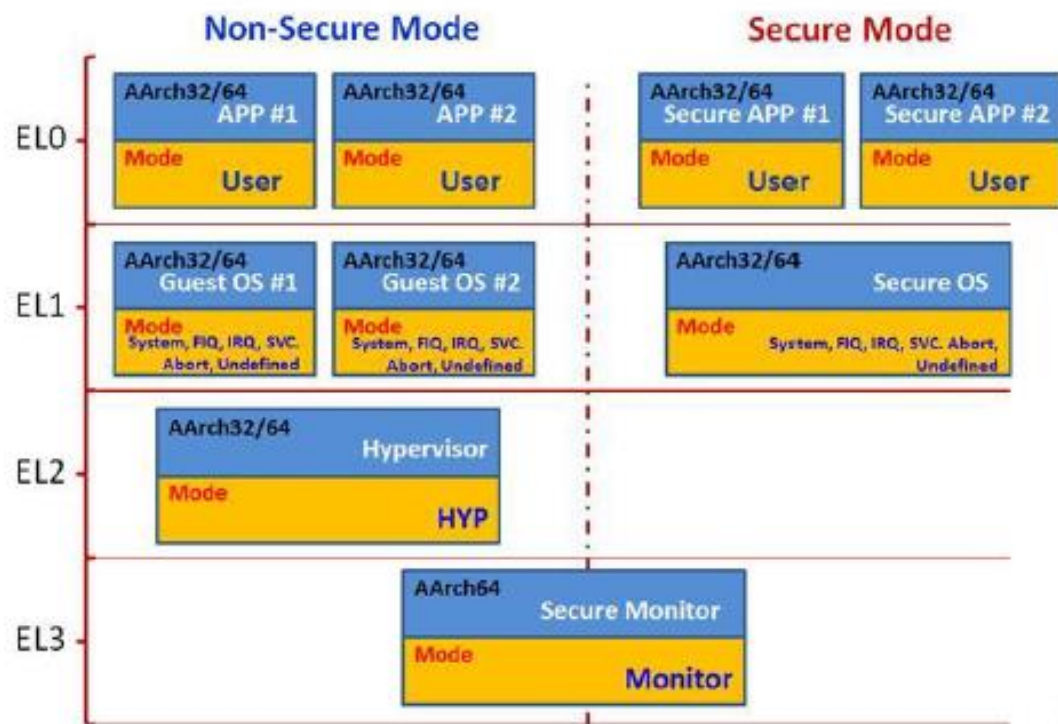
南开大学  
Nankai University

# ARMv8架构的异常等级与安全模型

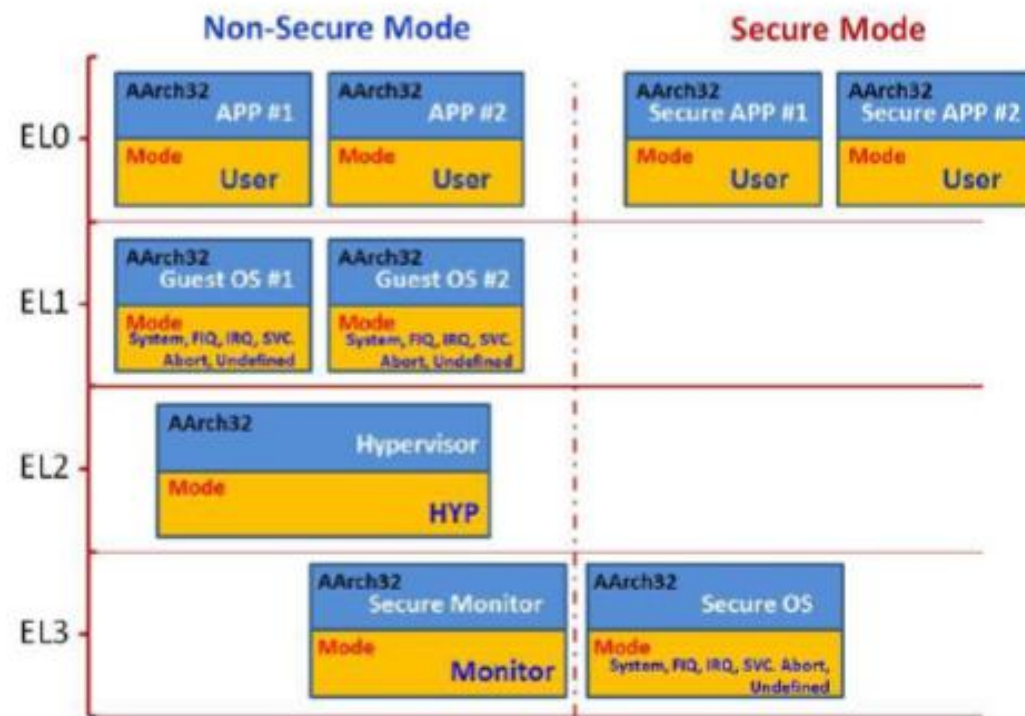
- 支持多级执行权限：异常等级的数字越大，软件执行权限越高

Exception Level	
EL0	Application
EL1	Linux kernel- OS
EL2	Hypervisor (可以理解为上面跑多个虚拟OS)
EL3	Secure Monitor (ARM Trusted Firmware)
Security	
Non-secure	EL0/EL1/EL2, 只能访问Non-secure memory
Secure	EL0/EL1/EL3, 可以访问Non-secure memory & Secure memory,可起到物理屏障安全隔离作用

# ARMv8架构的异常等级与安全模型



当EL3使用AArch64时



当EL3使用AArch32时





# AArch32架构的寄存器

寄存器类型	Bit	描述
R0-R14	32bit	通用寄存器，但是ARM不建议使用有特殊功能的R13，R14，R15当做通用寄存器使用。
SP_x	32bit	通常称R13为堆栈指针，除了User和Sys模式外，其他各种模式下都有对应的SP_x寄存器：x={und/svc/abt/irq/fiq/hyp/mon}
LR_x	32bit	称R14为链接寄存器，除了User和Sys模式外，其他各种模式下都有对应的SP_x寄存器：x={und/svc/abt/svc/irq/fiq/mon},用于保存程序返回链接信息地址，AArch32环境下，也用于保存异常返回地址，也就说LR和ELR是公用一个，AArch64下是独立的。
ELR_hyp	32bit	Hyp mode下特有的异常链接寄存器，保存异常进入Hyp mode时的异常地址。
PC	32bit	通常称R15为程序计算器PC指针，AArch32 中PC指向取指地址，是执行指令地址+8，AArch64中PC读取时指向当前指令地址。
CPSR	32bit	记录当前PE的运行状态数据,CPSR.M[4:0]记录运行模式，AArch64下使用PSTATE代替。
APSR	32bit	应用程序状态寄存器，EL0下可以使用APSR访问部分PSTATE值。
SPSR_x	32bit	是CPSR的备份，除了User和Sys模式外，其他各种模式下都有对应的SPSR_x寄存器：x={und/svc/abt/irq/fiq/hpy/mon}，注意：这些模式只适用于32bit运行环境。
HCR	32bit	EL2特有，HCR.{TEG,AMO,IMO,FMO,RW}控制EL0/EL1的异常路由。
SCR	32bit	EL3特有，SCR.{EA,IRQ,FIQ,RW}控制EL0/EL1/EL2的异常路由，注意EL3始终不会路由。
VBAR	32bit	保存任意异常进入非Hyp mode & 非Monitor mode的跳转向量基地址。
HVBAR	32bit	保存任意异常进入Hyp mode的跳转向量基地址。
M_vBAR	32bit	保存任意异常进入Monitor mode的跳转向量基地址。
ESR_ELx	32bit	保存异常进入ELx时的异常综合信息，包含异常类型EC等，可以通过EC值判断异常class。
PSTATE		不是一个寄存器，是保存当前PE状态的一组寄存器统称，其中可访问寄存器有：PSTATE.{NZCV,DAIF,CurrentEL,SPSel},属于ARMv8新增内容，主要用于64bit环境下。



# AArch64架构的寄存器

寄存器类型	Bit	描述
X0-X30	64bit	通用寄存器，如果有需要可以当做32bit使用：WO-W30
LR (X30)	64bit	通常称X30为程序链接寄存器，保存跳转返回信息地址
SP_ELx	64bit	若PSTATE.M[0] ==1，则每个ELx选择SP_ELx，否则选择同一个SP_EL0
ELR_ELx	64bit	异常链接寄存器，保存异常进入ELx的异常地址（x={0,1,2,3}）
PC	64bit	程序计数器，俗称PC指针，总是指向即将要执行的下一条指令
SPSR_ELx	32bit	寄存器，保存进入ELx的PSTATE状态信息
NZCV	32bit	允许访问的符号标志位
DIAF	32bit	中断使能位：D-Debug，I-IRQ，A-SError，F-FIQ，逻辑0允许
CurrentEL	32bit	记录当前处于哪个Exception level
SPSel	32bit	记录当前使用SP_EL0还是SP_ELx，x= {1,2,3}
HCR_EL2	32bit	HCR_EL2.{TEG,AMO,IMO,FMO,RW}控制EL0/EL1的异常路由 逻辑1允许
SCR_EL3	32bit	SCR_EL3.{EA,IRQ,FIQ,RW}控制EL0/EL1/EL2的异常路由 逻辑1允许
ESR_ELx	32bit	保存异常进入ELx时的异常综合信息，包含异常类型EC等.
VBAR_ELx	64bit	保存任意异常进入ELx的跳转向量基地址 x={0,1,2,3}
PSTATE		不是一个寄存器，是保存当前PE状态的一组寄存器统称，其中可访问寄存器有：





允公允能 日新月异

# ARMv8架构的异常处理

- 异常：现代处理器必备的程序随机切换机制
  - 最常见的异常是由外部事件引起的中断服务过程
  - 衡量一个CPU实时性就是最短响应中断时间以及单位时间内响应中断次数
- ARMv8-A的异常类型
  - 同步异常（Synchronous exception）：直接由执行指令或者尝试执行指令引起，且异常返回地址指明了引起异常的特定指令细节，例如debugging
  - 异步异常（Asynchronous exception）：由IRQ、FIQ这两个中断请求管脚引起的中断及系统错误引起的异常，例如网卡接收网络数据



南开大学  
Nankai University



# ARMv8架构的异常处理

Synchronous(同步异常)	
异常类型	描述
Undefined Instruction	未定义指令异常
Illegal Execution State	非法执行状态异常
System Call	系统调用指令异常 ( SVC/HVC/SMC )
Misaligned PC/SP	PC/SP未对齐异常
Instruction Abort	指令终止异常
Data Abort	数据终止异常
Debug exception	软件断点指令/断点/观察点/向量捕获/软件单步 等Debug异常
Asynchronous(异步异常)	
类型	描述
SError or vSError	系统错误类型，包括外部数据终止
IRQ or vIRQ	外部中断 or 虚拟外部中断
FIQ or vFIQ	快速中断 or 虚拟快速中断





允公允能 日新月异

# ARMv8架构的异常处理

- AArch64状态下，引起异常的几类事件：

- 终止

- 指令取指错误，发生在CPU流水线取指阶段，如果目标指令地址是非法地址进入该异常

- 复位Reset

- 最高等级异常，CPU刚上电时或按下Reset重启键

- 执行异常产生指令

- 系统调用指令，引起软中断，用于用户程序申请访问硬件资源

- 例如屏幕输出“Hello World”，申请使用显示器，使用软件中断指令切换到内核态，系统内核代码来访问外设硬件

- 中断

- IRQ（一般中断请求）和FIQ（快速中断请求），后者优先级更高



南开大学  
Nankai University

# ARMv8架构的异常处理

## • ARMv8-A的异常处理

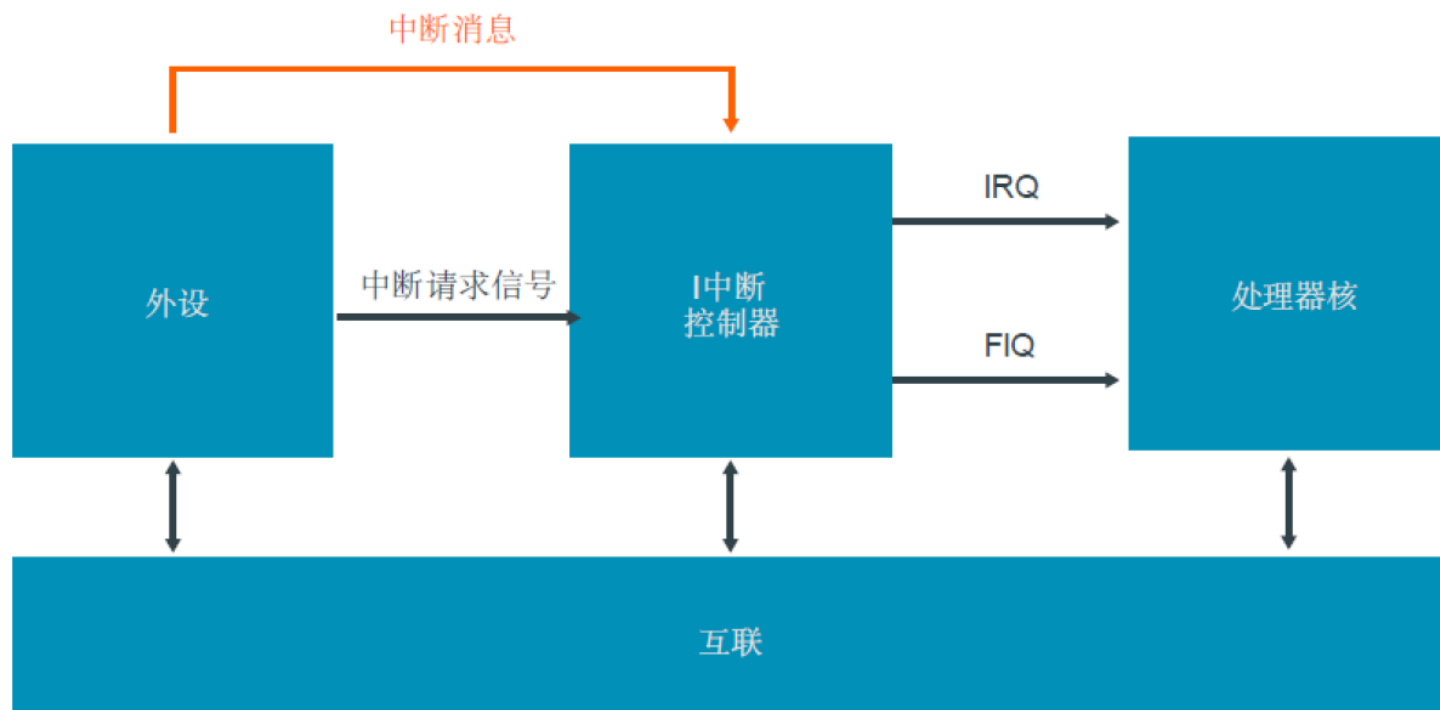
1. 更新备份程序状态寄存器**SPSR** (Save Program Status Register)，以保存异常处理结束返回时恢复现场必须的PSTATE信息
2. 用新的处理器状态信息更新程序状态**PSTATE**
3. 将异常处理结束返回的地址保存在异常链接寄存器**ELR\_ELn**中

```
type ProcState is (
    bits (1) N,      // Negative condition flag
    bits (1) Z,      // Zero condition flag
    bits (1) C,      // Carry condition flag
    bits (1) V,      // oVerflow condition flag
    bits (1) D,      // Debug mask bit [AArch64 only]
    bits (1) A,      // Asynchronous abort mask bit
    bits (1) I,      // IRQ mask bit
    bits (1) F,      // FIQ mask bit
    bits (1) SS,     // Software step bit
    bits (1) IL,     // Illegal execution state bit
    bits (2) EL,     // Exception Level (see above)
    bits (1) nRW,    // not Register Width: 0=64, 1=32
    bits (1) SP,     // Stack pointer select: 0=SP0, 1=SPx [AArch64 only]
    bits (1) Q,      // Cumulative saturation flag [AArch32 only]
    bits (4) GE,     // Greater than or Equal flags [AArch32 only]
    bits (8) IT,     // If-then execution state bits [AArch32 only]
    bits (1) J,      // J execution state bit [AArch32 only, RES0 in ARMv8]
    bits (1) T,      // T32 execution state bit [AArch32 only]
    bits (1) E,      // Endian execution state bit [AArch32 only]
    bits (5) M       // Mode field (see above) [AArch32 only]
)
```

# ARMv8架构的中断

- ARMv8架构的通用中断控制器（GIC）

—ARM架构下，系统通过GIC实现中断请求的仲裁、优先级排队和向处理器中断申请等操作



ARMv8-AArch32的执行状态可用的通用寄存器数量是多少个？

- ☐ A 8
- ☒ B 13
- ☐ C 16
- ☐ D 4



ARMv8的AArch64执行状态可用的通用寄存器数量是多少个？

- ☐ A 8
- ☐ B 13
- ☐ C 16
- ☒ D 31

提交

ARMv8中的Double Word数据类型占用几个字节?

- ☐ A 1
- ☐ B 2
- ☐ C 4
- ☒ D 8

提交

ARMv8的A32和A64指令集的指令字长度是固定的吗？

- ☒ A 是
- ☐ B 不是

提交



南开大学

NANKAI UNIVERSITY, P.R. CHINA 1919

允公允能 日新月异



# 汇编语言与逆向技术

## 第6章 华为鲲鹏处理器体系结构

王志

zwang@nankai.edu.cn

南开大学 网络空间安全学院

2024-2025学年



允公允能 日新月异

# 本章知识点

- 服务器处理器体系结构
- 服务器处理器并行组织结构
- Intel处理器体系结构
- ARM处理器体系结构
- 华为鲲鹏处理器
- 基于ARMv8的处理器体系结构



南开大学  
Nankai University