# X86、RISC、ARM架构服务器的比较

廖汉龙

北京理工大学计算机学院07111507班 北京 10081

([liaohanlong@outlook.com](mailto:liaohanlong@outlook.com))

**Comparison of X86, RISC, and ARM architecture servers**

Hanlong Liao

Class 07111507,School of Computer Science, Beijing Institute of Technology, Beijing 100081)

**Abstract**: in the past 10 years, with the rapid popularization of smart phones, the computing capacity of mobile smart terminals has begun to receive wide attention. Headed by the ARM chip in the intelligent terminal equipment accounts for more than 95% of the market, this also has been at the edge of Intel x86 architecture server caused a certain impact, but in the field of the desktop, the ARM is difficult to break through. This article aims to compare the performance of RISC and CICS architectures by exploring their features. By selecting X86 and ARM architectures, the performance and advantages of each processor are analyzed in detail.

**Key words**：x86; ARM; RISC; CISC; server architecture；

**摘要**：近10年来，随着智能手机的迅速普及，移动智能终端的运算处理能力开始得到大家的广泛关注。以ARM架构为首的处理芯片在智能终端设备占有了95%以上的市场，这也对一直处于优势的Intel公司的x86架构服务器造成了一定的冲击，但是在桌面领域，ARM还是难以有所突破。本文旨在通过探讨RISC和CICS架构的特点，比较其性能的优劣。通过选取X86、ARM架构具体分析各处理器的性能和各自所占的优势。

关键字：x86; ARM; RISC; CISC; 服务器架构；

ARM与X86服务器架构的服务器是目前市场上占有率最高的服务器，其中，ARM是基于精简指令集RISC设计，x86则是基于复杂指令集CISC设计，其中包含有较多的复杂的体系结构。X86架构的逻辑结构相对复杂很多。在目前的移动终端发展较为迅速和大数据、人工智能等也在处于数据量与数据复杂度快速上升的阶段，基于ARM与x86架构的服务器各占有一定的优势。

目前国产的CPU中，相对较为成功的有国防科技大学开发的飞腾系列CPU, 五十六所的申威系列CPU，以及中国科学院计算所开发的龙芯系列CPU。目前性能最好的属于飞腾系列，但是其是基于ARM架构的基础上开发的。而龙芯系列芯片则是基于MIPS精简指令集开发的。

从设计之初，两种架构模式的服务器芯片的设计思路就背道而驰。由于x86系列芯片的高复杂度，其功耗相对ARM架构的芯片要大很多。目前，intel公司宣称已经可以将x86架构芯片的功耗降低到数百毫瓦数量级，但是前提是在 CPU闲置进入C5或者C6 state时，而这样的功耗的数量级和大部分低频ARM芯片正常工作功耗数量持平。

但是在较为复杂的应用场景时，例如高性能计算，领域，ARM架构的服务器则显得力不从心，难以胜任。

目前，已经有很多的学者发表了很多文章对与CSIC, RSIC，MIPS进行比较，对于x86，ARM架构服务器进行了较有深度的探讨。

由于指令集的不同，使得ARM架构的服务器和x86架构的服务器在性能，成本，功耗，应用场景等多方面都显示了较大的差异，本文将对这几方面进行比较。

## 1.指令集的发展与简介

### 1.1 指令集分类

指令集是CPU中用来计算和控制计算机系统的一套指令的集合。每一种新型的CPU在设计时规定了一系列与其他硬件电路相配合的指令系统。而指令集的先进与否，也关系CPU的性能发挥，它是CPU性能体现的一个重要标志。

按照大类来分，指令集可以分为精简指令集（RISC）和复杂指令集（CISC）

RISC指令集的特点是指令数目少，每条指令都采用标准字长、执行时间短、中央处理器的实现细节对于机器级程序是可见的。

而在基于CISC架构的微处理器中，程序的各条指令是按照顺序串行执行的，每条指令中的各个操作也是按顺序串行执行的。

顺序执行的优点是控制简单，但计算机各部分的利用率不高，执行速度慢。

一般理解为RISC指令集是针对CISC指令集中的一些常用指令进行优化设计，放弃了一些复杂的指令，对于复杂的功能，需要通过组合指令来完成。

由于指令集的设计的差异，两者的使用场合不一样。对于复杂的系统，CISC更合适。否则，RISC更合适，且功耗低。在服务器早期发展阶段没有RISC和CISC的区别。历史上，Intel x86的第一个CPU定义了第一套指令集，这是最原始的指令集，后来一些研发企业发现很多指令并不常用，他们发现计算机指令集中也符合著名的“28定理”。几乎只有20%的指令就能够满足80%的工作。设计一套简介高效的指令集成为了一种趋势，为了与原来的相对较为复杂的指令集区分开来，这一套新的指令集被称为RISC指令集，而将原来的Intel x86指令集定义为CISC指令集。

典型的RISC指令集的CPU有:ARM、MIPS等。

典型的CISC指令集的CPU有：Intel的x86指令集，以及现在的AMD的x86-64指令集。

### 1.2 指令集发展

从上个世纪80年代以来，RISC与CISC指令各有发展，尤其是作为复杂指令集的CISC。RISC指令的发展主要体现在了在以ARM为主的嵌入式的设备终端，而CISC则主要是以Intel公司和AMD公司的X86架构服务器为主。

RISC的指令集由于用于嵌入式的开发，并且指令集寻求精简，所以其并没有像CSIC那样变化迅速,近些年来，RISC变化较小。

CSIC则发展和变化较为迅速。

较早出现的是x86指令集，早年电脑中为了提高浮现数据处理能力而增加的x87芯片系列数学协处理器使用了另外的x87指令。后来将x86与x87指令集统称为x86指令集。

CSIC指令集的发展与扩充包括了MMX指令集（多媒体指令集），包含有57条多媒体指令；SSE系列指令集，增强了对于浮点运算，视频媒体，线程同步等更多的指令，3D Now！系列扩展指令集，EM64T指令集，以及最近的AVX指令集，AVX是在之前的128bit扩展到和256bit的SIMD（single instruction multiple data）。而Sandy Bridge的SIMD演算单元扩展到256bits的同时数据传输也获得了提升，所以从理论上看CPU内核浮点运算性能提升到了2倍。

## 2.RISC与CISC架构比较

CISC体系的指令特征使用微代码。指令集可以直接在微代码记忆体（比主记忆体的速度快很多）里执行，新设计的处理器，只需增加较少的电晶体就可以执行同样的指令集，也可以很快地编写新的指令集程式，更加庞大的指令集。可以减少编程所需要的代码行数，减轻程式师的负担。高阶语言对应的指令集：包括双运算元格式、寄存器到寄存器、寄存器到记忆体以及记忆体到寄存器的指令。

CISC体系能够有效缩短新指令的微代码设计时间，允许设计师实现CISC体系机器的向上相容。新的系统可以使用一个包含早期系统的指令超集合，也就可以使用较早电脑上使用的相同软体。

另外微程式指令的格式与高阶语言相匹配，因而编译器并不一定要重新编写。

但是CISC指令集以及晶片的设计比上一代产品更复杂，不同的指令，需要不同的时钟周期来完成，执行较慢的指令，将影响整台机器的执行效率。

RISC体系的指令特征是精简指令集：包含了简单、基本的指令，透过简单、基本的指令组合成复杂指令。

在RISC架构中，每条指令的长度都是相同的，可以在一个单独操作里完成。大多数的指令都可以在一个机器周期里完成，并且允许处理器在同一时间内执行一系列的指令。

在使用相同的晶片技术和相同运行时钟下，RISC系统的运行速度将是CISC的2～4倍。由于RISC处理器的指令集是精简的，它的记忆体管理单元、浮点单元等都能设计在同一块晶片上，RISC处理器比相对应的CISC处理器设计更简单，所需要的时间将变得更短，并可以比CISC处理器应用更多先进的技术，开发更快的下一代处理器。 但是RISC多指令的操作使得程式开发者必须小心地选用合适的编译器，而且编写的代码量会变得非常大。

另外由于RISC体系的处理器需要更快记忆体，这通常都集成于处理器内部，就是L1 Cache（一级缓存）。

CISC与RISC指令集架构的差异，总结来说可以归纳为以下几点：

1、指令的形成。CISC因指令复杂，采用微指令码控制单元的设计，而RISC的指令90%是由硬体直接完成，只有10%的指令是由软体以组合的方式完成，因此指令执行时间上RISC较短，但RISC所须ROM空间相对的比较大而RAM使用大小应该与程序的应用比较有关系。

2、定址模式。CISC指令集架构需要较多的定址模式，而RISC只有少数的定址模式，因此CPU在计算记忆体有效位址时，CISC占用的汇流排周期较多。

3、指令周期。CISC指令的格式长短不一，执行时的周期次数不统一，而RISC结构则相反，采用了管线处理架构的设计，进而达到了平均一周期完成一条指令。

在设计上RISC较CISC简单，同时因为CISC的执行步骤过多，闲置的单元电路等待时间增长，不利于平行处理的设计，所以就效能而言RISC较CISC还更优，但RISC因指令精简化后造成应用程式码变大，需要较大的程式记忆体空间，且存在指令种类较多等等的缺点。

4．CISC指令大量使用到了寄存器。由于复杂指令多位高阶指令，这些指令需要使用寄存器到寄存器，内存到寄存器的指令，这将对寄存器有一个较大的需求。但是精简指令集架构则不存在这样的问题，其最大的开销，是对于程式的存取的开销。

## 3.X86与ARM架构服务器比较

ARM 架构和 X86 架构是彼此借鉴和共同创新的，这类似于 iOS 和 Android。例如 X86 架构开始引入 RISC 的一些设计例如一级缓存和微指令，而 RLISC 也开始搭载一些复杂的指令集。X86 架构的优势在于复杂的指令集所带来的高性能，与此对应的则是芯片 的复杂度和面积提升所带来的高功耗，而 ARM 架构的优势则是高效，灵活的指令集搭配所带来 的低功耗便利性，缺点则是执行复杂程序时性能上要明显弱于 X86 和对带宽的高要求。

### 3.1 x86架构模式

x86架构于1978年推出的Intel 8086中央处理器中首度出现，它是从Intel 8008处理器中发展而来的，而8008则是发展自Intel 4004的。8086在三年后为IBM PC所选用，之后x86便成为了个人电脑的标准平台，成为了历来最成功的CPU架构。

其他公司也有制造x86架构的处理器，计有Cyrix（现为威盛电子所收购）、NEC集团、IBM、IDT以及Transmeta。Intel以外最成功的制造商为AMD，其早先产品Athlon系列处理器的市场份额仅次于Intel Pentium。

8086是16位处理器；后来出现的一系列的处理器表示了32位架构的细微改进，推出了数种的扩充，直到2003年AMD对于这个架构发展了64位的扩充，并命名为AMD64。后来英特尔也推出了与之兼容的处理器，并命名为Intel 64。两者一般被统称为x86-64或x64，开创了x86的64位时代。

x86架构是重要的可变指令长度的CISC（复杂指令集电脑，Complex Instruction Set Computer）。字组（word, 4位组）长度的存储器访问允许不对齐存储器地址，字组是以低位字节在前的顺序存储在存储器中。在较新的微架构中，x86处理器会把x86指令转换为更像RISC的微指令再予执行，从而获得可与RISC比拟的超标量性能，但保持向前兼容。x86架构的处理器一共有四种执行模式，分别是真实模式，保护模式，系统管理模式以及虚拟V86模式。

目前市场上主流的x86架构的服务器都是64位的，以64位x86处理器为例，x86架构服务器定址是通过寄存器或者内存单元数值，通过CPU地址生成单元参与计算的，而不是通过ALU参与计算，所以在流水线上不会与上下文的算术逻辑指令产生流水相关。

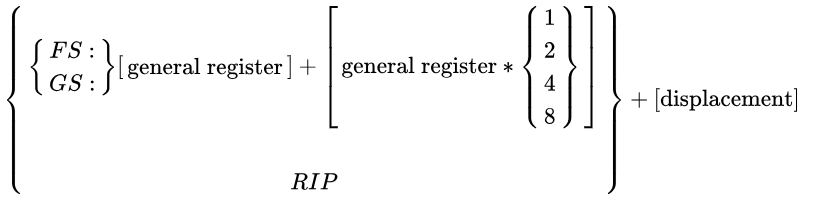


图1 X86-64位寻址模式

值得注意的是，在定址过程中，使用到的LEA指令具有单时钟特性，执行效率很高。

X86架构服务器的寄存器有从16位到64位等，从SSE系列芯片以来，出现了128位，256位，512位的寄存器。

### 3.2 ARM架构模式

ARM架构，过去称作高级精简指令集机器（Advanced RISC Machine，Acorn RISC Machine），是一个32位精简指令集（RISC）处理器架构，其广泛地使用在许多嵌入式系统设计。但在其他领域上也有很多作为，由于节能的特点，ARM处理器非常适用于移动通信领域，匹配其主要设计目标为低成本、高性能、低耗电的特性。另一方面，超级计算机消耗大量电能，ARM同样被视作更高效的选择。

至2009年为止，ARM架构处理器占市面上所有32位嵌入式RISC处理器90%的比例，使它成为占全世界最多数的32位架构之一。ARM处理器可以在很多消费性电子产品上看到，从便携式设备（PDA、移动电话、多媒体播放器、掌上型电子游戏和计算机）到电脑外设（硬盘、桌面型路由器），甚至在导弹的弹载计算机等军用设施中都有他的存在。在此还有一些基于ARM设计的衍伸产品，重要产品还包括Marvell的XScale架构和德州仪器的OMAP系列。

在ARM架构中，CPU定义了8中模式，包括用户模式，系统模式，SVC模式，Abort模式，未定义模式，干预模式，快速干预模式以及Hyp模式，其中，系统模式是仅无需例外进入的特权模式。仅以执行明确写入CPSR的模式位的指令进入。

ARM指令集讲求精简又快速的设计方式，整体电路不采用微码，其指令集的架构方式为读取/存储架构，不支持地址不对齐存储器访问（ARMv6内核现在已经支持），ARM架构拥有大量的1632-bit寄存器堆，32位操作码长度固定，编码数量的耗费较低，流水线负担较小，大部分指令均在一个CPU周期运行，拥有精简但是快速的双优先级子系统，具有可切换的寄存器组支持快速叶端函数调用的链寄存器。

ARM比较有特色地使用了有条件运行的方式，即使用4-bit条件编码在每个指令前，表示每个指令的运行是否为有条件式的。这样的方式大大减低了在存储器访问指令时用到的编码位。并且，ARM架构下，能够将shift和rotate等功能合并成数据处理型指令。

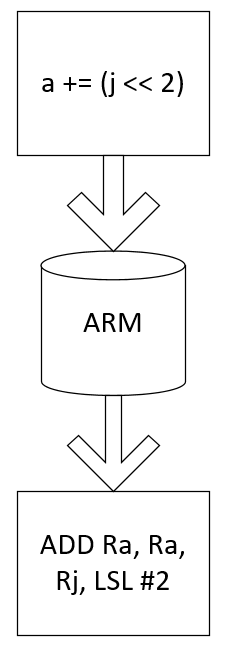


图2 ARM架构下的shift,rotate指令合成

这种方式让一般的ARM程序变得更加紧密，而不需经常使用存储器访问，流水线也可以更有效地使用。即使在ARM以一般认定为慢速的速度下运行，与更复杂的CPU设计相比它仍能运行得不错。

ARM7和大多数较早的设计具备三阶段的流水线：提取指令、解码，并运行。较高性能的设计，如ARM9，则有更深阶段的流水线： Cortex-A8有13阶段的流水线。提高性能的额外方式，包含一颗较快的加法器，和更广的分支预测逻辑线路。例如，在ARM7DI核与ARM7DMI核之间的差异，是一种改进的乘法器（因此添加的“M”）。

ARM架构服务器的寄存器不像X86架构寄存器复杂。

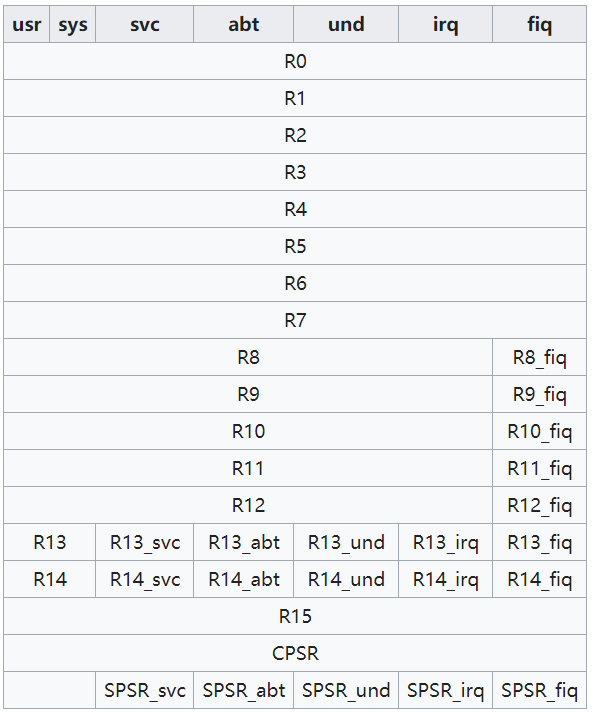


图3 ARM架构寄存器

其中，寄存器 R0-R7 对于所有CPU模式都是相同的，它们不会被分块。对于所有的特权CPU模式，除了系统CPU模式之外，R13和R14都是分块的。也就是说，每个因为一个异常（exception）而可以进入模式，有其自己的R13和R14。这些寄存器通常分别包含堆栈指针和函数调用的返回地址。

### 3.3 X86与ARM架构服务器比较

1.指令集长度。由于 X86 指令集是属于 CISC类型的指令集， 其每条指令的长度是不固定的，而且有几种不同的格式，造成了 X86 处理器的解码工作非常复杂。

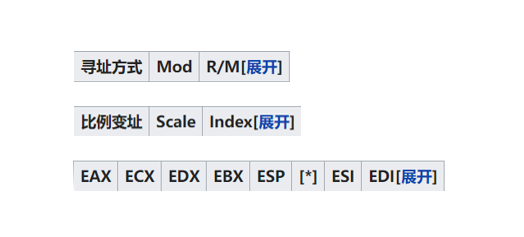


图4 X86指令示例

X86处理器通过延长处理器中的流水线来提高工作频率，但是这带来的弊端是如果出现预测出错的情况，CPU工作停滞时间较长。ARM 指令集适合于使用流水线技术，因为ARM架构大多数指令采用相同的字节长度， 并且在字边界上对齐， 字段位置固定， 特别是操作码的位置。基于ARM架构的流水线允许在当前指令译码阶段去取其下一条指令，并且在流水过程中被拆分成几个更小的、 能够被流水线并行执行的单元，可获得最高的吞吐率；而X86指令集的执行需要调用微代码的一个微程序，在执行速度上不如 ARM 指令集。

ARM并行处理能力强于x86的架构，精简指令使得其非常适合于采用处理器的流水线、超流水线和超标量技术，从而实现指令级并行操作， 提高处理器的性能。然而 X86 指令集却给 VLSI（硬件电路设计） 设计带来很大的设计负担，不利于单片集成。

2.浮点运算能力。X86构架处理器中的FPU (Floating Point Unit）浮点运算单元的运算能力较差。由于 X86 指令集中使用一个操作数堆栈，在计算过程中，如果出现计算量太大，则需要将中间构成存储到堆栈，则需要较多的时间来处理FXCH指令来将正确的数据放到堆栈的顶部。

在ARM中，完成一个浮点运算，通常需要数千时钟周期。ARM处理器不支持浮点运算,浮点运算运行在一个特殊的速度较慢的浮点模拟器。

另外，X86架构服务器指令可访问内存地址，由于指令更加复杂，以及内存与处理器之间的处理指令速度的严重不协调，使得工作效率的降低。ARM处理器使用 Load/Store 的存储模式， 意味着只有 Load 和 Store 指令才能从内存中读取数据到寄存器，所有其他指令只对寄存器中的操作数进行计算。因此，每条指令中访问的内存地址不会超过 1 个， 指令访问内存的操作不会与算术操作混在一起，相对来说，涉及到读物内存的指令，其效率要高很多。

3.外部扩展。ARM架构可以支持非常丰富的外部扩展。但是X86 指令对于扩展部件是不友好的。在前些年， X86 架构的处理器对于 4GB 的内存容量上限制， 目前已经提高。但是其发展的速度远远满足不了日益增长的需求。

4. X86架构的处理器为了提高性能，而出现寄存器重命名、缓冲器巨大、乱序执行、分支预测、X86 指令转化等等现象，这增加了成本，也限制了处理器工作频率的进一步提升。此外，未来解决x86的指令问题， 处理器必须集成数目庞大的晶体管。而ARM 指令集可以大大简化处理器的控制器和其他功能单元的设计， 不必使用大量专用寄存器， 特别是允许以硬件线路来实现指令操作， 从而节约的处理器的制造成本，核心面积小。

5.软件应用。Arm结构的电脑系统因为硬件性能的制约、操作系统的精简、以及系统兼容等问题的制约，造成Arm结构的电脑系统不可能像X86架构的系统那样有众多的编程工具和第三方软件可供选择及使用，ARM的编程语言大多采用C和JAVA。

但随着android系统的出现,ARM的兼容性问题得到了很好的解决。由于x86系统已经退出了较长的时间，在用户的应用、软件配套、软件开发工具的配套及兼容等很多方面已经形成了完善的体系。

6.功耗。X86架构的发展思路是同时考虑到性能和运行效率。复杂的指令集导致了其功耗是同等效率ARM架构服务器的几倍甚至几十倍。这与ARM结构的服务器基本无法相比。ARM的设计及发展思路则完全不同，其低功耗的特性也是的其在移动终端上占据了绝对的统治地位。

## 结束语

本文就CISC, RISC指令集进行了介绍，对x86和ARM架构服务器的发展与现状进行了分析。本文对x86和ARM架构的服务器的指令集的长度，并行性能，浮点数的计算能力，地址访问方式，外部扩展的能力，基于ARM与x86的操作系统与软件的应用以及两种架构的功耗等方便进行了分析与对比，得到了一系列结论。随着X86与ARM架构的服务器进一步发展，两种模式势必走向互相融合，取长补短的方向，对于x86与ARM架构的服务器各方面的性能的比较仍然具有现实意义。

**参考文献**

1. 贾少波 《基于X86平台的ARM指令集模拟器的设计》[J]电子设计工程 年份：2013.09

2.友文《让老电脑用上64位Win7》[J]《电脑知识与技术：经验技巧》 2015.06

胡品平《基于专利分析的移动智能终端产业发展策略研究》 《科技管理研究》 年份：2014 作者：

3高天宇《ARM新星速死之谜》[J]《董事会》2014/04/03

4Blem E R, Menon J, Sankaralingam K, et al. Power struggles: Revisiting the RISC vs. CISC debate on contemporary ARM and x86 architectures[C]. high-performance computer architecture, 2013: 1-12.

5Younan Y, Philippaerts P, Piessens F, et al. Filter-resistant code injection on ARM[C]. computer and communications security, 2009: 11-20.

6姜贵福,《试析单片机编程中的相关技巧》[J]《中国电子商务》2016