计算机体系结构 A4 叶振宇 3090103433

计算机设计基础
Computer Architecture plays an important role in performance improvement.
Pipeline, dynamic schedueing, ooo, branch prediction,
speculation, superscalar, VLIW, prediction
Instructions and so on.
微处理器四个阶段：Microprocessors-Quantitative Architecture-Instruction-Level Parallelism-Thread-level/Data-level parallelism
体系的三个分类：Instruction set architecture-Microarchitecture-System Design
ISA七个方面：Class of ISA-Memory addressing-Addressing modes-Types and sizes of operands-Operations-Control flow instructions-Encoding an ISA
Three Wall：ILP（指令级并行）Memory Power

计算机分类：根据指令和数据流
SISD,MISD(少),SIMD(llliac-IV CM-2),MIMD(SPARCCenter T3D)
MIMD的两种编程模式：SPMD(单程序),MPMD
桌面计算-Optimized price- performance
服务器-dependability可靠scalability-efficient throughput
嵌入式Embedded-Real time-Strict resource constraints
Register machine vs Stack machine vs
Accumulater machine RISC vs. CISC
技术趋势：Cost decrease rate ~ density increase rate
Technology improves continuously, an impact of this improvements can be in discrete leaps.

电能趋势：（电压低更省电性能没降多少，多核降电压提性能）
挑战：distributing the power-removing the heat-preventing hot spot

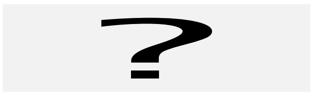
Cost趋势：因素：time volume commodification
Component direct（加工） indirect（渠道）ASP list price（仓储利润）

可靠性The quality of the delivered service such that quality of the delivered service such that reliance can justifiably be placed on this service.
reliability, maintainability, availability
量度标准：MTTF MTTR FIT = 1/MTTF
MTBF = MTTF+MTTR
Availability= MTTF/MTBF
解决：time/resource redundancy

测度性能：
Wall-clock-time（response/elapse）唯一性能通用指标
CPU time(不含IO) = user + system
Throughout：Amount of work done in a given time（带宽可能重要于延迟）
Response升throughout升 换CPU
Throughout升response不一定升 多CPU

同指令集对比

SPEC The System Performance Evaluation Cooperative
Maximizing performance means minimizing response (execution) time
等价权重计算 1/(ti*SUM(1/tj))
Geometric mean does NOT predict run time

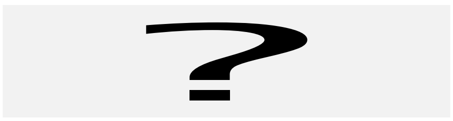
定量原则：利用流水线（最重要）流水线CPU（指令级） 组关联cache 流水功能部件（操作级）
Locality：时空
Focus on the common case：（重要 普遍pervasive）
Simple Is fast Amdahl's law
Speedup = 增强后表现/增强前表现
=


Total speedup no more than 1/(1-f)
影响：
Program ic
Compiler ic cpi
Instruction set ic cpi
Organization cpi cc
Technology cc
TPC-C （TransactionProcessing PerformanceCouncil）: standard industry benchmark for OLTP

Benchmark不可能永远有效
Peak performance tracks observed performance. X

指令集架构 Instruction Set Architecture

ISA分类：stack accumulator gpr
Gpr中：alu中memory操作数个数
RR lwsw 0；RM-1（src only）；MM-2或3

C = A + B

寄存器更快，寄存器可以存储变量，编译器可以直接使用，减小代码密度（寄存器用更少的位）
RR 指令条数最多，密度最低，但最简单，固定指令长度，固定编码复杂度，小CPI

内存编址 word bit byte Intel小端 小处为低位 对齐

寻址模式 模式越多，降低指令条数，体系变复杂，CPI增大

Register	Add R4, R3
Immediate	Add R4, #3
Displacement	Add R4, 100(R1)
Register indirect	Add R4, (R1)
Indexed	Add R3, (R1+R2)
Direct or absolute	Add R1, (1000)
Memory indirect	Add R1, @(R3)
Autoincrement	Add R1, (R2)+
Autodecrement	Add R1, -(R2)
Scaled	Add R1,100(R2)[R3]

流行模式：displacement（12-16bits） 立即数（8-16bits） 间接寄存器

操作数大小类型：DSP需要宽寄存器加速定点运算

指令集操作：Arithmetic and logical-Data transfer-Control
All machines MUST provide instruction support for basic system functions.
Floating point instructions are optional but are commonly provided.

控制流指令：branch call jump return 7bits
Three techniques to specify the branch conditions：
condition code（指令顺序）/register（占用寄存器）/and branch（相当于两条指令）
Caller can deliver variables to callee via callee save registers
只要程序用到某寄存器，其在调用时，都保存

指令系统编码：影响编译系统的大小，CPU的实现
Key factor：The range of addressing modes；The degree of independence between opcodes and addressing modes
定长编码：程序大，代码密度低，易实现

编译器角色：design architectures to be compiler targets .
至少16个寄存器 keep it simple less Is more
all addressing modes apply to all data transfer instructions

MIPS：
Sp29 gp28 fp30 ra31 lui ori 配合使用，记入32位数于寄存器a0-a3传参 v0-v1返回 jal记录返回地址于31

CISC 存储资源少，着重编译器优化

RISC 降低cpi 降低指令集 ls结果

Pipeline流水线
降低CPU时间，增大throughout，增大资源利用率
exploits parallelism among the instructions in a sequential instruction stream
machine cycle = 最长的某stage的时间
Machine cycle > latch latency + clock skew
理想speedup = 流水级
不能级数太多：无法分，级之间有延迟，逻辑复杂，指令相关，Lots of complications.
slt set when less than 三参数
针对单周期，流水线降低ct,针对多周期，降低cpi
各stage执行时间不同降低性能，冲突
latch作用，不同指令间不影响，数据传递
the memory system must deliver 5 times the bandwidth over the unpipelined version.
结构：These are conflicts over hardware resources
数据：Instruction depends on result of prior computation which is not ready (computed or stored) yet
控制：branch condition and the branch PC are not available in time to fetch an instruction on the next clock
stall法
jmp jal没有rs；wreg&m2r为lw； wreg则为alu或lw
alurr beq bne sw 将rt作为源寄存器；branch则为branch
cpi = 1 + 平均每条指令stall数

结构、数据hazard
Stall（bubble），最简单delays all instructions issued after the instruction that was stalled, while other instructions in the pipeline go on proceeding，不再取新指令

结构hazard：replicate hardware：分开指令数据memory
Multiple memory port/instruction buffer
Double bump（双重触发）先写后读 fully pipeline，多冗余（浮点部件）
允许结构竞争（影响不大，不常见）：cost（memory bandwidth，性能是否足够提高）；latency

数据hazard：
读和写冲突，停两拍
Stall-add hardware interlock，
逻辑判断：IDEX源和后面的EXMEM及IDEX目的对比，若冲突，则让IDEX成为NOP，禁写

Forward path（bypass short-circuiting）
EX/MEM.ALUoutput → ALU input port（相邻）
MEM/WB.ALUoutput → ALU input port（隔一拍）
MEM/WB LMD→ALUinput（隔一拍，旧指令为lw）
旧指令的目标寄存器不能为0，对于MEMWB端，要先排除EXMEM端的冲突
Branch在ID结束，则需要forward到ID，产生无法解决的stall
Load stall不能解决lw之后紧跟（不含sw）冲突（必停一拍）
编译器（compile scheduling），在lw后跟上一条无关指令

Control hazard：造成很大的性能损失，超过数据
the deeper the pipeline, the worse the branch penalty in clock cycles
A higher CPI processor can afford to have more expensive branches
解决方案：freeze or flush the pipeline
MEM结束停3拍，代价固定，软件不能改，损失了not-taken，
Predict-not-taken
MEM结束决定地址时已经进入三条指令，nop（来得

及）
Not taken 0stall；taken 3stall
Perf = 1+ br% * take% *3
代码更改，if下的代码执行可能性更高
Predict-taken 实际上60%都是taken
ID确定是否taken，MEM出最终地址，地址一算好就取指令（仅先知道目标地址后知道是否taken才有效），对5流水线基本无优点
必停一拍（无法确定taken地址）
taken 停1拍，not taken 3拍
perf=1+Br%*(taken%*1+untaken%*3)
Delayed branch ID结束一切 中间未知1拍可插无关指令
the branch delay slot 编译器
From before 总是改进/target 可能要重复指令变大程序/fall-through 不taken时有改进，对taken不能有影响
考虑编译器复杂性，slot往往只有一个即使停多拍

Branch prediction
Static 是否taken；profile information from previous run
Dynamic 1/2/N bit/correlating（关联） branch prediction
1位，预测错误变位
2位，连续两次预测错误变位
Correlate 结合全局预测位和局部预测位

拓展
Latency 功能完成时钟-1；
initiation interval 两条同指令间隔
对于全流水，后者为1，对于非流水后者为前者加1
浮点乘7步，浮点加4步，浮点除25步（非流水）
新的latch，注意memory的带宽
WAW型数据冒险乱序写，操作延迟变大RAW更频繁
写口冲突：增加写口，ID插stall（移位寄存器记录写时刻，加重数据冒险），MEM或WB插stall（易检测，但两个地方stall）
数据冒险类型：RAW；WAW（在多个stage写能发生），WAR（少，读很早，复杂编址中可能）
解决WAW：Stall an instruction；Cancel the WB phase of the earlier instruction（都可在ID完成）
寄存器有两套，结构竞争有所不同
RAW stall： source registers are no longer listed as destinations；source registers are no longer listed as the destination of a load in the EX/MEM register.
WAW：Check instructions in A1, ..., A4, Divide, or M1, ...,M7 for 相同目的。

Exceptions and Interrupts
turn off all writes for the faulting instruction and any instruction that issued after the faulting instruction
saves the PC of the offending instruction
precise exceptions：All instructions before the faulting instruction complete，运行慢
旧指令的中断优先级高

设计流水线指令集：
Avoid variable instruction lengths and running times
Avoid sophisticated addressing modes
Don't allow self modifying code
Avoid implicitly setting CCs in instructions

长流水：requires additional forwarding hardware
more complex hazard detection to find dependencies
好处：stage may be shorter；clock cycle can be shorter；allowing more instructions to be issued in a fixed time；

CACHE

内存技术：DRAM-dynamic random access memory
High density, low power, cheap, slow，refreshed regularly
SRAM 非失电下永存
Temporal Locality：Keep most recently accessed data items closer to the processor
Spatial Locality：Move blocks consists of contiguous words to the faster levels

Cache：Small, fast storage used to improve average access time to slow memory

块替换策略：LRU：
Psedo LRU：V NV两个位（next victim/victim）
另一种：3个位一个set：for AB；CD；AB/CD

写策略：Write-through adv：Read misses don't result in writing back；memory hierarchy is consistent and it is simple to implement.
Write back ad v：Writes occur at speed of cache；main memory bandwidth is smaller when multiple writes occur to the same block.

Write stall：CPU等待写通过，buffer解决，过多写可能有saturation（饱和）

Write allocate/around（back对应allocate；through对应around）

Unified cache 命中率低，硬件少
Split cache没有指令块和数据块的冲突miss

Supervisor/user space bit当0，用户和系统都可以访问

AMAT（平均内存访问时间）：指令和数据要分开算，性能间接度量，比miss rate好

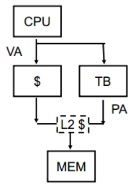
AMAT包含了指令本身的运行时间，故有下式

Miss rate和硬件速度无关
乱序执行要去掉重叠时间

改进cache性能

1.Reduce the time to hit in the cache

small and simple caches：直接映射更快，片上cache
way prediction：预测位，下一次访问哪个block P4
预测成功，则cache时延；反之多一个时延改变预测
avoiding address translation：TLB（使用VA）



进程标识号，避免切换时flush；Virtual cache（一致性问题）
用部分pageoffset做index，可以同时查索引和VAPA转换；cache小于pageoffset的2幂次，用多组关联
trace caches：多指令并行，branch，复杂的地址转换，没有I-cache misses，prediction miss，packet breaks

2.Increase cache bandwidth

pipelined cache access：将流水线、cache访问、使一级cache命中的有效时延分散到多个周期；命中时间长，失败代价大
non-blocking caches：乱序执行CPU，处理器等待数据cache返回数据时，可以继续从指令cache取指令；缺失时保持数据cache命中。乱序执行若能在二级cache命中，可隐藏一级的cache缺失代价，但不能隐藏二级的。
multibanked caches：多块consecutive对半；interleaving奇偶；group interleaving余12和余34

3.Reduce the miss penalty

multilevel caches：一级升命中，二级减代价

一级的全局缺失和局部缺失一样；
critical word first and early restart（wrapped fetch）：对大block有效；和空间局部性有冲突
read miss prior to writes：写操作完成前就进行读
写通过：先查看buffer的内容，解决RAW；写回，read miss替换出要写，替换写到buffer后，读就可以开始
merging write buffers：同地址块在buffer中只占一项
victim caches：小全关联，最近被替换，miss后检查

4.Reduce the miss rate
Miss来源：compulsory（第一次冷启动）capacity（容量，只能做大）conflict（索引冲突）

larger block size：降冷，U曲线（增缺失代价，增冲突容量），取决于底层的延迟和带宽
large cache size：始终下降（翻倍下降25%，因为冷降不到底）增大hit时间，AMAT是U线
higher associativity：降冲突，direct是2路组两倍，多路组cc（AMAT）变大了
way Prediction and Pseudo-Associative Cache：引入预测位，在哪一组，使hit不要太长
compiler optimizations：merge arrays（空间局部性）；loop interchange（让列序号在内循环，空局）；loop fusion（时局，多个相关循环并成一个），block（矩阵子块计算，全关联一直降，直接映射U）

5.Reduce the miss penalty and miss rate via parallelism
hardware prefetching：在CPU需要前就预取，降冷
compiler prefetching：插入预取指令

内存技术

Latency难降bandwidth好升
提高内存带宽
wider main memory（增bus）：错误纠正复杂，需要MUX
Simple interleaved memory：parallelism of having many chips in a memory system；优化顺序访问；width of the bus and cache need NOT change；各块可以同时读，按序传，access时间可以重叠
Independent memory banks：独立地址线数据总线
Number of banks >=Number of clock cycles to access word in bank
Avoiding Memory bank conflicts:编译器：expand the size of the array so that it is not a power of 2;硬件：Using a prime number of memory banks 2n-1 or 2n+1

Access time:time between when a read is requested and when the desired word arrives;
Cycle time:minimum time between requests to memory
后者大于前者，地址线信号要稳定一会

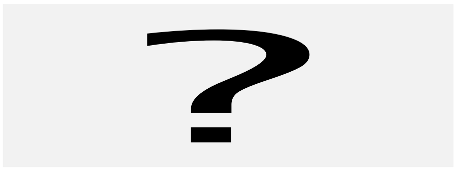
改进DRAM性能
Fast page mode dram（FPM）：空局，行地址缓存
Synchronous dram：add a clock signal to the DRAM interface, so that the repeated transfers would not bear that overhead.
Double data rate（DDR）：上下降沿都可以传输数据
ddr266 时钟只有133，带宽266
New DRAM interface（rambus，rdram）：Each chip has interleaved memory and a high speed interface；allows other accesses over the bus between the sending of the address and return of the data；a separate row- and column-command buses

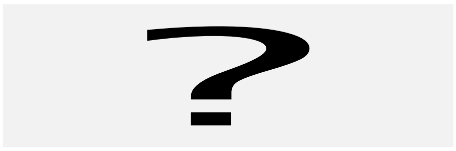
Multiprocessors
Performance Cost Efficiency Scalability Fault tolerance
Flynn分类：SISD MISD（不使用） SIMD（低代价，灵活） MIMD（灵活，使用off-the-shelf微处理器Not superscalar, each node is superscalar）

Memory类型

Shared（SMP/UMA）：延迟同，lwsw直接交流
Distributed：延迟不同，message交流
DSM: (physically) distributed, (logically) shared memory
Multiple computer：message passng（同步异步）

UMA/NUMA：理想：延迟带宽都好；现实：带宽有限，延迟随系统增长而增长，增加bank





Centralized shared memory
Decentralized memory：get more memory bandwidth, lower memory latency；Drawback: Longer communication latency；Software model more complex
共享地址空间，但是内存是分布式的
multiple computers：内存私有，无法之间互访

并行框架
Programming Model：Multiprogramming（工作多，无交流）；Shared address space；Message passing；Data Parallel
Communication Abstraction：
Shared address space：lwsw直接访问，拓展有限，数据一致性和保护；同步问题及复杂硬件
Model of choice for uniprocessors, small-scale MPs
Ease of programming，Lower latency
Easier to use hardware controlled caching

Message passing：显式IO交流，独立地址空间，简单硬件，Difficult programming Model，Communication patterns explicit and precise；Focuses attention on costly non-local operations

---

趋势：Shared Memory，Small-to-medium size UMA；Larger NUMAs

Limited program parallelism：新算法
Long communication latency：caching shared data to lower the remote access frequency；restructuring the data to make more accesses local；Synchronization；latency hiding techniques

---

Cache Coherence：
Snooping Solution (Snoopy Bus)：广播通知，小型
Write Invalidate Protocol:写时通知失效，read miss时从cache中找最新的
三状态，invalid，shared，exclusive；四事件：CPU/BUS W/R
Write Broadcast Protocol (typically write through)：同时广播更新所有副本

Directory-Based Schemes：一开始就记录了重复数据的位置，点对点通知，拓展好
Directory: track state of every block in memory, and change the state of block in cache according to directory
block三状态：shared/uncached/exclusive
Bit vector表示哪个处理器有副本，dirty bit