Chapter 1 ISA的七个维度：① Class of ISA; ② Memory Addressing; ③ Addressing Modes; ④ Types and sizes of operands; ⑤ Operations; ⑥ Control flow inst.; ⑦ Encoding an ISA. 18-24年11note

3 Walls: ILP Wall; Memory Wall; Power Wall

SISD: ILP; SIMD, MISD; MIMD(TLP)

Dynamic Power: $Power_{dy} = \frac{1}{2} \times Capacitive\ load \times Voltage^2 \times Frequency\ Switched$ (switching transistors); $Energy_{dy} = Cap\text{-}load \times Voltage^2$

Static Power: $Power_{static} = current\ static \times Voltage$.

MTTF: To Failure; MTTR: To Repair; FIT = 1/MTTF

MTBF = MTTF + MTTR; Availibity $= \frac{MTTF}{MTBF}$

$MIPS = \frac{1}{1000000}\left(\frac{\#\ of\ inst.}{benchmark} \times \frac{benchmark}{total\ runtime}\right)$ 每秒百万条指令.

CPU Time = IC × CPI × CC time (= 1/rate); IPC: 每时钟周期指令数.

Amdahl's Law = $Speed\ up = \frac{1}{(1-a) + a/k}$.

Chapter 2 trace cache: 动态指令序列、一个line保存一条或多条指令。

DRAM(Dynamic Random): 高密度，便宜、慢、定时刷新、主存。

SRAM(Static Random): 低密度，high power、贵、快。cache。

Cache原理: 时间、空间局部性. Temporal / Spatial

Cache 四个问题: Q1: Block Placement (FA, DM, SA)

Q2: Block Identification (Tag / Block)

Q3: Block Replacement (LRU / Random / FIFO)

Q4: Write Strategy (Write Back / Write Through)

Write Buffer. 阻塞 → L2 Cache

• Write Policy (write miss): Write Allocate / Write Around

CPU Ex Time = (CPU CC + MemStall cycle) × CC time

MemStall Time = IC × Mem refs per inst. × MR × MP.

AMAT = Hit Time + MR × MP. 单次访问存储.

提升Cache性能: ① Reduce Hit Time: Small and simple first-level caches, way prediction; avoiding address translation (TLB / VIPT), trace cache | ② Increase Bandwidth: Pipelined caches, multibanked caches, non-blocking caches (L2以候支持)

③ Reduce Miss Penalty: multilevel caches, read miss prior to writes; critical word first (early restart), merging write buffers, victim caches

④ Reduce Miss Rate: larger block size, large cache size, higher associativity (way prediction, pseudo-associativity); compiler optimizations. (Merging arrays, Loop Interchange, Loop Fusion, Blocking).

⑤ Reduce miss penalty or miss rate via parallelization: Hardware or compiler prefetching.

提升DRAM性能 (FPM: Fast Page Mode DRAM; Synchronous DRAM; DDR; RAMBUS).

VIPT: 要求 pgoffset ≥ block offset + index. 避免别名问题.
多路组相连, cache大小可扩大n倍. 不用虚地址 idx, tag, ofs.

多级: $AMAT = HT_{L1} + MR_{L1}(HT_{L2} + MR_{L2} \times MP_{L2})$.

第④个: Block size↑, compulsory↓; Cache Larger↑, capacity↓; Higher Associativity↑, conflict↓. MR – Block size: 一开始 compulsory↓, 后来 conflict↑.

Chapter 3

Latency: 一个指令产生结果到另一个指令可以使用结果的cycle差值.

Initiation Interval: FU接受两条指令cycle差值. 完全流水: 1, 完全非流水: latency+1

Latency = FU Time − 1 cycle.

RAW: True Dependence; WAW: output −; WAR: anti −.

ILP: Basic Block中ILP比较小 | 软件: Loop LP

两个基本原则: ① exception behavior ② data flow. 程序正确性.

软件方法: Loop unrolling; Static Branch Prediction; Static Multiple Issue (VLIW); Advanced Compiler (软件流水线、全局代码调度). 硬件辅助: conditional or Predicated Inst., compiler speculation, hw support

硬件方法: Dynamic Scheduling.

Scoreboard: IF, IS: 避免结构冲突, WAW冲突.
RO: 直到两个都可用: 动态解决 RAW冲突.
EX, WB: 解决 WAR冲突.

计分牌三张表: Inst. Status: 指令阶段; RRS (FU将写的寄存器);
FUS (busy, op, Fi 目标, Fj, Fk 源, Qj, Qk 从哪个FU得数, Rj Rk 是否可用).

限制: ILP; issued window 不跨基本块; FU; WAR和WAW stalls.

特点: Multiple Funcs, Issue in order, OOO complete, Scoreboard centralized control. Stall when WAW, WAR.

Tomasulo: Fewer Func, unpipelined, issue in order, complete OOO, FP op. queue, Reservation Station, LD/ST buf, CDB. Renam: no WAW, WAR. CDB: forwarding, decentralized. 不局限于基本块

保留站 (busy, op, Vj, Vk 源, Qj, Qk 产生源操作数的保留站, A 地址).

(IF) IS: 保留站空闲, 发射

(EX) (WB): 监听总线, 两个都准备好了, 执行.

好处: 分布式, 消除 WAW/WAR. 缺点: CDB瓶颈, 一条总线, 非精确中断.

Scoreboard 显式寄存器重命名: 分配新的物理寄存器, translation table.

改后: IS: 要分配一个新 rd. 若没有 free rd, 不IS.

Branch Prediction: (动态) 1-bit, 2-bit, Correlating, Tournament, Branch Target Buffer, Integrated Inst. fetch Units, Return Addr Predictors.

Correlating: (m, n) 预测器: 最近m个行为, 在2^m个n位预测器中选. (m: global, n: local)

Tournament: 用地址索引局部预测器和选择器, 用分支历史索引全局预测器.

gshare: 全局分支历史与分支地址异或做索引.

Integrated: branch prediction, inst. prefetch, inst. memory access and buffering.

Return Addr: 用栈存储 ra.

Speculation: 投机, 乱序执行, 按序提交. Tomasulo 加 ROB, 取消 store buffer, ROB 换名.

ROB Entry: op, rd, value, ready, exception vector.

Commit: 跳错: 刷新ROB重新开始. 其他: 更新 reg/mem, 移除ROB Entry.
会停顿在 Issue (RS和ROB结构冲突).

	Issue Structure	Hazard Detection	Scheduling	Characteristic
Superscalar (static)	Dynamic	HW	Static	顺执
Superscalar (dynamic)	Dynamic	HW	Dynamic	乱执
Superscalar (speculative)	Dynamic	HW	Dynamic with 投机	乱执+投机
VLIW/LIW	Static	SW	Static	packet内无Hazard
EPIC	Mostly static	mostly SW	mostly Static	明显hazard被compiler标注.

静态多发射: 对指令序列要求高, 依赖逻辑复杂, 需要考虑结构冒险.
VLIW: 字槽 (ILP开发), 二进制兼容性差.

Chapter 4 | DLP: Vector Processor, GPU.

Vector Processors: memory-memory (VMMA) (相比读叠指令, 检查内存地址的依赖, 启动延迟更高, 内存带宽要求更高).

vector-register.

向量处理器 Pros: 向量内无依赖. SIMD. Regular Mem access pattern, Fewer branches

Cons: 并行少, 性能就差; 内存带宽瓶颈.

三种寻址方式: unit stride (连续), Non-unit (constant) stride. Indexed (gather-scatter).

向量长度 > MVL, Strip Mining: 先处理 N mod MVL, 后面每段……

opt1: Vector chaining: 有 RAW 时可用. Convoy -> chime -> chain.

opt2: Conditional. Exec / Masked Vector Inst.

opt3: Sparse Matrix: Indexed load (gather), Indexed store (scatter).

opt4: Multi-Lane.

GPU: Threads are organized into blocks, blocks are organized to a grid. GPU handles thread management.

与向量的区别: 无标量processor, 用 multi-threading 来隐藏 memory latency. 有很多 FU.

All GPU loads are gather inst. All GPU stores are scatter inst.

Chapter 5 | TLP. Multiprocessors.

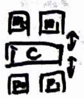
Flynn分类法: SISD, SIMD, MISD, MIMD.

硬件下的内存: Shared / Distributed.

软件下的内存: Shared Memory / Message bypassing.

- Centralized shared mem (UMA/SMP).
- Distributed memory
  - Distributed shared mem (DSM/NUMA).
  - multiple computer: msg bypassing {同步 / 异步}

UMA: uniform mem access.



Small multiprocessor only.

NUMA:

more scalable. 难实现.

不同处理器访问内存的时间……

Programming Model: multiprogramming: 无交流; Shared addr space (通过mem共享), message passing (消息传递). data parallel (不同数据同时处理, 全局交换信息).

Communication Abstraction: Shared addr space (ld/st/atomic swap). msg passing (send/receive library calls)

CPI = Base CPI + Remote Request Rate × Remote Request Cost.

Cache Coherence →
- P1 先写X后读X, return 新值
- P1 先写X, P2 后读, ----
- P1 先写, P2 后写, 序列化.

(一) Snooping Protocol.

- Write Invalid Protocol
- Write Broadcast Protocol.

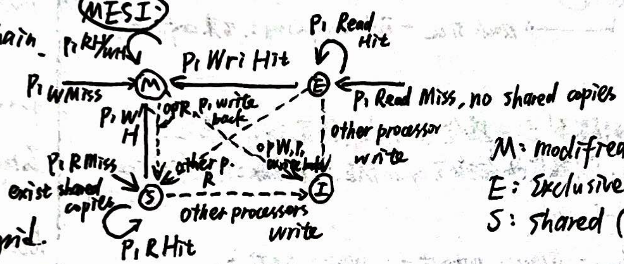
串行化的实现: Bus.

V=0, D=x -> Invalid

V=1, D=0 -> Shared (not dirty)

V=1, D=1 -> Exclusive (dirty).

MESI:



M: modified (private, != mem).

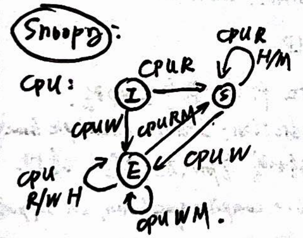
E: Exclusive (private, = mem).

S: shared (shared, = mem)

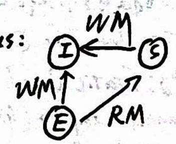
Invalid.

Snoopy:

CPU:



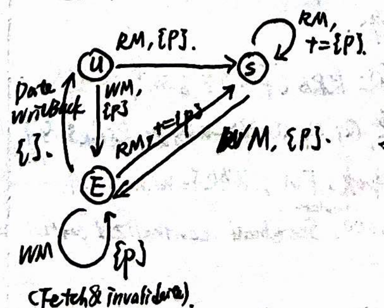
Bus:



(二) Directory-Based.

Block Status: Shared, Uncached, Exclusive

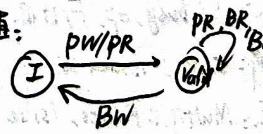
Local node / Home node / Remote node



NAK/ACK

Limited buffering

如果写直:



同步: Atomic Exchange / Test-and-Set / Fetch and Increment

```
try: mv x3, x4
     lr x2, x1
     sc x3, 0(x1)
     bnez x3, try
     mv x4, x2.
```

spinlock

```
try:  li x3, 1
lock: lw x3, 0(x1)
      bnez x3, lock
      exch x2, 0(x1)
      bnez x2, try.
```

```
try: lr x2, x1
     addi x3, x2, 1
     sc x3, 0(x1)
     bnez x3, try.
```

Barrier 改进 → Local_sense

```
local sense = !local sense.
if (count == total) { count = 0;
    release = lo_s }
else unlock; spin(release == lo_s).
```

Models of Memory Consistency (连贯性).

- Sequential Consistency: in order; interleaved. (bad performance).
- Relaxed. 共享变量写操作……
  - L> W-R
  - L> W-W -