

Indice generale

Prefazione	VII	Misurare le prestazioni	28
		Prestazioni della CPU	29
		Misura delle prestazioni associate alle istruzioni	30
		Equazione classica di misura delle prestazioni	31
1 Il calcolatore: astrazioni e tecnologia	1	1.7 Barriera dell'energia	35
1.1 Introduzione	1	1.8 Metamorfosi delle architetture: il passaggio dai sistemi uniprocessore ai sistemi multiprocessore	37
Tipi di calcolatori e loro caratteristiche	3	1.9 Un caso reale: la valutazione del Core i7 Intel	41
Benvenuti nell'era post-PC	4	Benchmark SPEC per la CPU	41
Che cosa si può imparare da questo libro	5	Benchmark SPEC sull'assorbimento di potenza	42
1.2 Otto grandi idee sull'architettura dei calcolatori	8	1.10 Errori e trabocchetti	43
Progettare tenendo conto della Legge di Moore	8	1.11 Note conclusive	46
Utilizzo delle astrazioni per semplificare il progetto	9	Organizzazione del testo	47
Rendere veloci le situazioni più comuni	9	1.12 Inquadramento storico e approfondimenti	48
Prestazioni attraverso il parallelismo	9	1.13 Esercizi	48
Prestazioni attraverso la pipeline	9	Risposte alle domande di autovalutazione	52
Prestazioni attraverso la predizione	9		
Gerarchia delle memorie	10	2 Le istruzioni: il linguaggio dei calcolatori	53
Affidabilità e ridondanza	10	2.1 Introduzione	53
1.3 Che cosa c'è dietro un programma	10	2.2 Operazioni svolte dall'hardware del calcolatore	54
Da un linguaggio ad alto livello al linguaggio dell'hardware	11	2.3 Operandi dell'hardware del calcolatore	58
1.4 Componenti di un calcolatore	13	Operandi allocati in memoria	60
Attraverso lo specchio	14	Operandi immediati o costanti	63
Touchscreen	15	2.4 Numeri con e senza segno	65
Dentro la scatola	16	Riepilogo	70
Un posto sicuro per i dati	19		
Comunicare con gli altri calcolatori	20		
1.5 Tecnologie per la produzione di processori e memorie	21		
1.6 Prestazioni	24		
Definizione delle prestazioni	25		

2.5 Rappresentazione delle istruzioni nel calcolatore	71	3 L'aritmetica dei calcolatori	151
Campi delle istruzioni RISC-V	73	3.1 Introduzione	151
2.6 Operazioni logiche	79	3.2 Somme e sottrazioni	151
2.7 Istruzioni per prendere decisioni	81	Riepilogo	154
Cicli	83	3.3 Moltiplicazione	155
Scorciatoie per il controllo dei confini di vettori e matrici	85	Versione sequenziale dell'algoritmo della moltiplicazione e sua implementazione hardware	156
Costrutto <i>case/switch</i>	85	Moltiplicazione di numeri dotati di segno	158
2.8 Supporto hardware alle procedure	86	Moltiplicazione veloce	159
Utilizzo di più registri	88	Moltiplicazione nel RISC-V	159
Procedure annidate	90	Riepilogo	160
Allocazione dello spazio nello stack per nuovi dati	92	3.4 Divisione	160
Allocazione dello spazio nello heap per nuovi dati	93	Un algoritmo della divisione e l'hardware che lo implementa	161
2.9 Comunicare con le persone	95	Divisione di numeri dotati di segno	164
Caratteri e stringhe in Java	98	Una divisione più veloce	165
2.10 Indirizzamento RISC-V di un campo immediato e di un indirizzo ampio	100	Divisione nel RISC-V	165
Operandi immediati ampi	100	Riepilogo	165
Indirizzamento nei salti	101	3.5 Numeri in virgola mobile	168
Riassunto delle modalità di indirizzamento del RISC-V	104	Rappresentazione in virgola mobile	169
Come decodificare il linguaggio macchina	104	Eccezioni e Interrupt	170
2.11 Parallelismo e istruzioni: la sincronizzazione	107	Standard IEE 754 per la virgola mobile	170
2.12 Tradurre e avviare un programma	110	Addizione in virgola mobile	174
Compilatore	110	Moltiplicazione in virgola mobile	178
Assemblatore	111	Istruzioni in virgola mobile nel RISC-V	181
Linker	113	Accuratezza dell'aritmetica	188
Loader	116	Riepilogo	190
Librerie a caricamento dinamico	116	3.6 Parallelismo e aritmetica dei calcolatori: parallelismo a livello di parola	192
Come avviare un programma Java	118	3.7 Un caso reale: le estensioni SIMD per lo streaming e le estensioni avanzate dell'x86 per il calcolo vettoriale	193
2.13 Un esempio riassuntivo in linguaggio C	119	3.8 Come andare più veloci: il parallelismo a livello di parola applicato alla moltiplicazione di matrici	194
Procedura <i>scambia</i>	120	3.9 Errori e trabocchetti	197
Procedura <i>ordina</i>	121	3.10 Note conclusive	200
2.14 Confronto tra vettori e puntatori	126	3.11 Inquadramento storico e approfondimenti	201
Versione della procedura <i>azzera</i> che utilizza vettore e indice	127	3.12 Esercizi	202
Versione della procedura <i>azzera</i> che utilizza i puntatori	128	Risposte alle domande di autovalutazione	205
Confronto tra le due versioni di <i>azzera</i>	129	4 Il processore	206
2.15 Approfondimento: compilazione del C e interpretazione di Java	130	4.1 Introduzione	206
2.16 Un caso reale: le istruzioni dell'architettura MIPS	130	Un'implementazione di base del RISC-V	207
2.17 Un caso reale: le istruzioni dell'architettura x86	131	4.2 Convenzioni del progetto logico	210
Evoluzione dell'Intel x86	131	Metodologia di temporizzazione	211
Registri e modalità di indirizzamento dell'x86	134	4.3 Realizzazione di un'unità di elaborazione	213
Operazioni su numeri interi dell'x86	134	Progettazione di un'unità di elaborazione unificata	218
Codifica delle istruzioni x86	138	4.4 Uno schema semplice di implementazione	220
Conclusioni sull'x86	139	Unità di controllo della ALU	221
2.18 Un caso reale: le altre istruzioni dell'architettura RISC-V	139	Progettazione dell'unità di controllo principale	223
2.19 Errori e trabocchetti	141	Funzionamento dell'unità di elaborazione	226
2.20 Note conclusive	143	Completamento dell'unità di controllo	228
2.21 Inquadramento storico e approfondimenti	144	Perché non si utilizzano più implementazioni a singolo ciclo?	230
2.22 Esercizi	146		
Risposte alle domande di autovalutazione	150		

4.5 Introduzione alla pipeline	230	Un esempio di memoria cache: il processore FastMATH Intrinsicity	342
Progettazione dell'insieme di istruzioni per architetture dotate di pipeline	235	Riepilogo	344
Hazard nelle pipeline	235	5.4 Come misurare e migliorare le prestazioni di una cache	345
Hazard sul controllo	239	Riduzione delle miss di una cache utilizzando un posizionamento più flessibile dei blocchi	348
Riepilogo sulla pipeline	243	Come trovare un blocco nella cache	353
4.6 Unità di elaborazione con pipeline e unità di controllo associata	244	Come scegliere il blocco da sostituire	354
Rappresentazione grafica delle pipeline	253	Ridurre la penalità di miss utilizzando una cache multilivello	355
Unità di controllo della pipeline	257	Ottimizzazione software mediante elaborazione a blocchi	357
4.7 Hazard sui dati: propagazione o stallo	261	Riepilogo	362
Hazard sui dati e stallo	268	5.5 Affidabilità delle gerarchie delle memorie	363
4.8 Hazard sul controllo	272	Definizione di malfunzionamento	363
Ipotizzare che il salto condizionato non sia eseguito	273	Codice di Hamming per la correzione di errori singoli e identificazione di errori doppi (SEC/DED)	365
Ridurre i ritardi associati ai salti condizionati	273	5.6 Macchine virtuali	369
Predizione dinamica dei salti	275	Requisiti del monitor di una macchina virtuale	370
Riepilogo sulla pipeline	278	Mancanza di supporto alle macchine virtuali da parte dell'architettura dell'insieme di istruzioni	371
4.9 Le eccezioni	278	Protezione e architettura dell'insieme delle istruzioni	371
Gestione delle eccezioni nelle architetture RISC-V	280	5.7 Memoria virtuale	372
Eccezioni e loro gestione nella pipeline	281	Come individuare la posizione di una pagina e come ritrovarla	376
4.10 Parallelismo a livello di istruzioni	285	Page fault	377
Concetto di speculazione	286	Memoria virtuale per un insieme ampio di indirizzi virtuali	380
Parallelizzazione statica dell'esecuzione	287	Che cosa succede in scrittura?	382
Processori dotati di parallelizzazione dinamica dell'esecuzione	291	Come rendere più veloce la traduzione degli indirizzi: il TLB	382
Efficienza energetica e pipeline avanzate	296	TLB del processore FastMATH Intrinsicity	384
4.11 Un caso reale: la pipeline del Cortex-A53 ARM e del Core i7 Intel	297	Integrazione della memoria virtuale, dei TLB e delle cache	387
Cortex-A53 ARM	297	Meccanismi di protezione basati sulla memoria virtuale	388
Core i7 920 di Intel	299	Gestione delle miss del TLB e dei page fault	390
Prestazioni del Core i7 920 Intel	302	Riepilogo	393
4.12 Come andare più veloci: parallelismo a livello di istruzioni e moltiplicazione di matrici	304	5.8 Schema comune per le gerarchie delle memorie	395
4.13 Argomenti avanzati: un'introduzione alla progettazione digitale con un linguaggio di progettazione dell'hardware e un modello di pipeline, e approfondimenti sulla pipeline	306	Domanda 1: dove può essere posizionato un blocco?	395
4.14 Errori e trabocchetti	307	Domanda 2: come si individua un blocco?	396
4.15 Note conclusive	308	Domanda 3: quale blocco deve essere sostituito in caso di miss della cache?	397
4.16 Inquadramento storico e approfondimenti	309	Domanda 4: come vengono gestite le scritture?	398
4.17 Esercizi	309	Le tre C: un modello intuitivo per comprendere il comportamento delle gerarchie delle memorie	399
Risposte alle domande di autovalutazione	318	5.9 Come utilizzare una macchina a stati finiti per controllare una cache semplificata	401
5 Grande e veloce: la gerarchia delle memorie	320	Una cache semplificata	401
5.1 Introduzione	320	Macchine a stati finiti	402
5.2 Tecnologie delle memorie	325	FSM per il controllore semplificato della cache	404
Tecnologia SRAM	325	5.10 Parallelismo e gerarchie delle memorie: coerenza delle cache	405
Tecnologia DRAM	326	Schemi di base per garantire la coerenza	407
Memorie flash	328	Protocolli di snooping	407
Memorie a disco	328	5.11 Parallelismo e gerarchie delle memorie: i dischi RAID	409
5.3 Principi base delle memorie cache	330	5.12 Argomenti avanzati: come implementare i controllori delle cache	409
Accesso alla cache	333		
Gestione delle miss della cache	339		
Gestione della scrittura	340		

5.13	Due casi reali: la gerarchia delle memorie del Cortex-A53 ARM e del Core i7 Intel	410	6.8	Introduzione alle topologie delle reti di calcolatori	466
	Prestazioni della gerarchia delle memorie del Cortex-A53 e del Core i7	412		Implementazione delle topologie di rete	468
5.14	Un caso reale: il resto del sistema RISC-V e le istruzioni speciali	414	6.9	Come comunicare con il mondo esterno: le reti dei cluster	469
5.15	Come andare più veloci: blocchi di cache e moltiplicazione tra matrici	415	6.10	Benchmark per i multiprocessori	470
5.16	Errori e trabocchetti	417		Modelli delle prestazioni	472
5.17	Note conclusive	422		Modello roofline	474
5.18	Inquadramento storico e approfondimenti	423		Confronto tra due generazioni di Opteron	475
5.19	Esercizi	423	6.11	Un caso reale: il confronto mediante il modello roofline tra un Core i7 960 Intel e una GPU Tesla NVIDIA	480
	Risposte alle domande di autovalutazione	433	6.12	Come andare più veloce: processori multipli e moltiplicazione di matrici	484
6	Processori paralleli: dai client al cloud	434	6.13	Errori e trabocchetti	487
6.1	Introduzione	434	6.14	Note conclusive	489
6.2	Le difficoltà nel creare programmi a esecuzione parallela	436	6.15	Inquadramento storico e approfondimenti	492
6.3	SISD, MIMD, SIMD, SPMD e processori vettoriali	441		Bibliografia	492
	SIMD negli x86: le estensioni multimediali	443	6.16	Esercizi	492
	Architetture vettoriali	443		Risposte alle domande di autovalutazione	499
	Confronto tra architetture vettoriali e scalari	445			
	Processori vettoriali ed estensioni multimediali	446			
6.4	Multithreading hardware	448	Indice analitico		501
6.5	I multicore e gli altri multiprocessori a memoria condivisa	451	Manuale di riferimento RISC-V		511
6.6	Introduzione alle unità di elaborazione grafica	455	Appendici		
	Introduzione alle architetture GPU di NVIDIA	457	Appendice A	The Basics of Logic Design	
	Strutture di memoria delle GPU NVIDIA	458	Appendice B	Mapping Control to Hardware	
	La prospettiva delle GPU	460	Appendice C	La grafica e il calcolo con la GPU	
6.7	Cluster, calcolatori per centri di calcolo e altri multiprocessori a scambio di messaggi	462	Appendice D	A Survey of RISC Architectures for Desktop, Server and Embedded Computers	
	Calcolatori per grandi centri di calcolo	463			