

Computer Systems Architecture

A Networking Approach

Second Edition

Rob Williams



PEARSON

Longman

Harlow, England • London • New York • Boston • San Francisco • Toronto • Sydney • Singapore • Hong Kong
Tokyo • Seoul • Taipei • New Delhi • Cape Town • Madrid • Mexico City • Amsterdam • Munich • Paris • Milan

Contents

	Preface	xiii
	Preface to the first edition	xv
	Recommended lab sessions	xxi
Part 1	Basic functions and facilities of a computer	
1	Introduction: the hardware–software interface	3
1.1	Computer systems – the importance of networking	4
1.2	Hardware and software – mutual dependence	5
1.3	Programming your way into hardware – VHDL, a language for electronic engineers	6
1.4	Systems administration – we all need to know	9
1.5	Voice, image and data – technological convergence	9
1.6	Windowing interfaces – WIMPs	11
1.7	The global Internet – connecting all the networks	13
1.8	Using the PC – a case study; more reasons to study CSA	16
2	The von Neumann Inheritance	23
2.1	Base 2 – the convenience of binary – 10110011100011110000	24
2.2	Stored program control – general-purpose machines	24
2.3	Instruction codes – machine action repertoire	26
2.4	Translation – compilers and assemblers	28
2.5	Linking – bringing it all together	28
2.6	Interpreters – executing high-level commands	30
2.7	Code sharing and reuse – let’s not write it all again!	31
2.8	Data codes – numeric and character	32
2.9	The operating system – Unix and Windows	36
2.10	Client–server computing – the way of the Net	40
2.11	Reconfigurable hardware – an alternative to fetch–execute	42

3	Functional units and the fetch–execute cycle	47
3.1	The naming of parts – CPU, memory, IO units	48
3.2	The CPU fetch–execute cycle – high-speed tedium	52
3.3	System bus – synchronous or asynchronous?	56
3.4	System clock – instruction cycle timing	59
3.5	Pre-fetching – early efforts to speed things up	61
3.6	Memory length – address width	63
3.7	Endian-ness – Microsoft vs. Unix, or Intel vs. Motorola?	65
3.8	Simple input–output – parallel ports	67
4	Building computers from logic: the control unit	73
4.1	Electronic Lego and logic – the advantage of modular units	74
4.2	Basic logic gates – truth tables for AND, OR, XOR and NOT	75
4.3	Truth tables and multiplexers – a simple but effective design tool	77
4.4	Programmable logic – reconfigurable logic chips	79
4.5	Traffic light controllers – impossible to avoid!	82
4.6	Circuit implementation from truth tables – some practical tips	83
4.7	Decoder logic – essential for control units and memories	85
4.8	CPU control unit – the ‘brain’	87
4.9	Washing machine controllers – a simple CU	88
4.10	RISC vs. CISC decoding – in search of faster computers	91
5	Building computers from logic: the ALU	97
5.1	De Morgan’s equivalences – logical interchangeability	98
5.2	Binary addition – half adders, full adders, parallel adders	98
5.3	Binary subtraction – using two’s complement integer format	101
5.4	Binary shifting – barrel shifter	103
5.5	Integer multiplication – shifting and adding	105
5.6	Floating-point numbers – from very, very large to very, very small	108
6	Building computers from logic: the memory	117
6.1	Data storage – one bit at a time	118
6.2	Memory devices – memory modules for computers	120
6.3	Static memory – a lot of fast flip-flops	121
6.4	Dynamic memory – a touch of analogue amid the digital	122
6.5	DRAM refreshing – something else to do	124
6.6	Page access memories – EDO and SDRAM	124
6.7	Memory mapping – addressing and decoding	127
6.8	IO port mapping – integration vs. differentiation	131

7	The Intel Pentium CPU	137
7.1	The Pentium – a high-performance microprocessor	138
7.2	CPU registers – temporary store for data and address variables	143
7.3	Instruction set – introduction to the basic Pentium set	148
7.4	Structure of instructions – how the CU sees it	149
7.5	CPU status flags – very short-term memory	151
7.6	Addressing modes – building effective addresses	153
7.7	Execution pipelines – the RISC speedup technique	155
7.8	Pentium 4 – extensions	157
7.9	Microsoft Developer Studio – using the debugger	158
8	Subroutines	167
8.1	The purpose of subroutines – saving space and effort	168
8.2	Return address – introducing the stack	169
8.3	Using subroutines – HLL programming	170
8.4	The stack – essential to most operations	172
8.5	Passing parameters – localizing a subroutine	173
8.6	Stack frame – all the local variables	176
8.7	Supporting HLLs – special CPU facilities for dealing with subroutines	179
8.8	Interrupt service routines – hardware-invoked subroutines	179
8.9	Accessing operating system routines – late binding	180
9	Simple input and output	185
9.1	Basic IO methods – polling, interrupt and DMA	186
9.2	Peripheral interface registers – the programmer’s viewpoint	187
9.3	Polling – single-character IO	191
9.4	Interrupt processing – service on demand	197
9.5	Critical data protection – how to communicate with interrupts	205
9.6	Buffered IO – interrupt device drivers	209
9.7	Direct memory access (DMA) – autonomous hardware	210
9.8	Single-character IO – screen and keyboard routines	212
10	Serial Connections	219
10.1	Serial transmission – data, signals and timing	220
10.2	Data format – encoding techniques	221
10.3	Timing synchronization – frequency and phase	224
10.4	Data codes and error control – parity, checksums, Hamming codes and CRCs	227
10.5	Flow control – hardware and software methods	235

10.6	The 16550 UART – RS232	237
10.7	Serial mice – mechanical or optical	244
10.8	Serial ports – practical tips, avoiding the frustration	246
10.9	USB – Universal Serial Bus	246
10.10	Modems – modulating carrier waves	252

11 Parallel connections 263

11.1	Parallel interfaces – better performance	264
11.2	Centronics – more than a printer port but less than a bus	264
11.3	SCSI – the Small Computer Systems Interface	267
11.4	IDE – Intelligent Drive Electronics	271
11.5	AT/ISA – a computer standards success story	272
11.6	PCI – Peripheral Component Interconnection	275
11.7	Plug-and-Play – automatic configuration	278
11.8	PCMCIA – Personal Computer Memory Card International Association	280

12 The memory hierarchy 285

12.1	Levels of performance – you get what you pay for	286
12.2	Localization of access – exploiting repetition	288
12.3	Instruction and data caches – matching memory to CPU speed	293
12.4	Cache mapping – direct or associative	295
12.5	Virtual memory – segmentation and demand paging	299
12.6	Address formulation – when, where and how much	304
12.7	Hard disk usage – parameters, access scheduling and data arrangement	306
12.8	Performance improvement – blocking, caching, defragmentation, scheduling, RAM disk	310
12.9	Optical discs – CD-DA, CD-ROM, CD-RW and DVDs	312
12.10	DVD – Digital Versatile Disc	316
12.11	MPEG – video and audio compression	316
12.12	Flash sticks – the new floppy disk	323

Part 2 Networking and increased complexity

13 The programmer's viewpoint 329

13.1	Different viewpoints – different needs	330
13.2	Application user – office packages	331

13.3	Systems administration – software installation and maintenance	333
13.4	HLL programmer – working with Java, C++, BASIC or C#	337
13.5	Systems programming – assembler and C	340
13.6	Hardware engineer – design and hardware maintenance	344
13.7	Layered virtual machines – hierarchical description	345
13.8	Assemblers – simple translators	346
13.9	Compilers – translation and more	347
14	Local area networks	353
14.1	Reconnecting the users – email, printers and database	354
14.2	PC network interface – cabling and interface card	359
14.3	Ethernet – Carrier Sense, Multiple Access/Collision Detect	363
14.4	LAN addressing – logical and physical schemes	367
14.5	Host names – another layer of translation	370
14.6	Layering and encapsulation – TCP/IP software stack	371
14.7	Networked file systems – sharing files across a network	372
14.8	Interconnecting networks – gateways	374
14.9	Socket programming – an introduction to WinSock	374
15	Wide area networks	383
15.1	The Internet – origins	384
15.2	TCP/IP – the essential protocols	386
15.3	TCP – handling errors and flow control	390
15.4	IP routing – how packets find their way	392
15.5	DNS – Distributed Name Database	398
15.6	World Wide Web – the start	401
15.7	Browsing the Web – Netscape Navigator	403
15.8	HTTP – another protocol	407
15.9	Search engines – Google	409
15.10	Open Systems Interconnect – an idealized scheme	412
16	Other networks	419
16.1	The PSTN – telephones	420
16.2	Cellnets – providers of mobile communications	426
16.3	ATM – Asynchronous Transfer Mode	435
16.4	Messaging – radio paging and packet radio networks	440
16.5	ISDN – totally digital	442
16.6	DSL – Digital Subscriber Line	446
16.7	Cable television – facilities for data transmission	447

17	Introduction to operating systems	455
17.1	Historic origins – development of basic functions	456
17.2	Unix – a landmark operating system	459
17.3	Outline structure – modularization	462
17.4	Process management – initialization and dispatching	463
17.5	Scheduling decisions – time-slicing, demand preemption or cooperative	469
17.6	Task communication – pipes and redirection	471
17.7	Exclusion and synchronization – semaphores and signals	473
17.8	Memory allocation – <code>malloc()</code> and <code>free()</code>	479
17.9	User interface – GUIs and shells	481
17.10	Input–output management – device handlers	482
18	Windows XP	491
18.1	Windows GUIs – responding to a need	492
18.2	Win32 – the preferred user API	494
18.3	Processes and threads – multitasking	495
18.4	Memory management – virtual memory implementation	496
18.5	Windows Registry – centralized administrative database	496
18.6	NTFS – Windows NT File System	498
18.7	File access – ACLs, permissions and security	499
18.8	Sharing software components – OLE, DDE and COM	502
18.9	Windows NT as a mainframe – Winframe terminal server	502
19	Filing systems	507
19.1	Data storage – file systems and databases	508
19.2	The PC file allocation table – FAT	515
19.3	Unix inodes – they do it differently	518
19.4	Microsoft NTFS – complexity and security	523
19.5	RAID configuration – more security for the disk subsystem	525
19.6	File security – access controls	526
19.7	CD portable file system – multi-session contents lists	528
20	Visual output	533
20.1	Computers and graphics – capture, storage, processing and redisplay	534
20.2	PC graphics adapter cards – graphics coprocessors	541
20.3	Laser printers – this is mechatronics!	547
20.4	Adobe PostScript – a page description language	549
20.5	WIMPs – remodelling the computer	554

20.6	Win32 – graphical API and more	555
20.7	The X Window system – enabling distributed processing	557
20.8	MMX technology – assisting graphical calculations	558
21	RISC processors: ARM and SPARC	563
21.1	Justifying RISC – increased instruction throughput	564
21.2	Pipeline techniques – more parallel operations	569
21.3	Superscalar methods – parallel parallelism	571
21.4	Register files – many more CPU registers	572
21.5	Branch prediction methods – maintaining the pipelines	574
21.6	Compiler support – an essential part of RISC	576
21.7	The ARM 32 bit CPU – origins	576
21.8	StrongARM processor – a 32 bit microcontroller	585
21.9	The HP iPAQ – a StrongARM PDA	588
21.10	Puppeteer – a StrongARM SBC	590
21.11	Sun SPARC – scalar processor architecture as RISC	592
21.12	Embedded systems – cross-development techniques	594
22	VLIW processors: the EPIC Itanium	601
22.1	Itanium 64 bit processor – introduction	602
22.2	Itanium assembler – increasing the control of the CPU	609
22.3	Run-time debugging – gvd/gdb	613
22.4	Future processor design – debate	615
23	Parallel processing	619
23.1	Parallel processing – the basis	620
23.2	Instruction-level parallelism (ILP) – pipelining	623
23.3	Superscalar – multiple execution units	623
23.4	Symmetric, shared memory multiprocessing (SMP) – the future?	623
23.5	Single-chip multiprocessors – the IBM Cell	626
23.6	Clusters and grids – application-level parallelism	629
	Appendix: MS Visual Studio 8, Express Edition	635
	Glossary	647
	Answers to end-of-chapter questions	661
	<u>References</u>	713
	Index	717