

Author's Profile 

Prof. B. Govindarajalu is presently Dean (Computer Studies) at Rajalakshmi Engineering College, Chennai. He is a B.E. in Electronics and Communication Engineering from Regional Engineering College, Trichirappali and M.Tech in Computer Science and Engineering from IIT Bombay. Before joining Rajalakshmi Engineering College he worked with Manipal Engineering College; IIT Bombay; ORG Systems (Baroda) Infotech Ltd. (Chennai); and Microcode (Chennai).

He has over 37 years of experience in computer hardware and IT industry covering design and development, manufacturing, maintenance, technical support, teaching and training. He has developed a series of microprocessor-based products such as the IBM System 360 Compatible Data Communications Multiplexer, VDU System Console and Terminal Processing Unit/Concentrator.

After 15 years of industrial experience, he founded Microcode, an IT firm specializing in PC hardware and networking training and development of diagnostic tools where he trained nearly 10,000 engineers in PC maintenance. He has developed a series of certification tests on computer architecture, PC hardware and networking. He has also taught personal computer hardware and architecture as visiting professor at several institutions such as Rajalakshmi Engineering College, Crescent Engineering College, Jaya Engineering College, Vellore Institute of Technology and Sastha Institute of Technology. He has conducted career courses on computer networking for Rajalakshmi Engineering College, Velammal Engineering College, Kongu Engineering College, J J College of Engineering and SSN College of Engineering.

He has been a specialist for a UNESCO project on curriculum development. He has also served as a member of the Board of Studies at PSG College of Technology, Coimbatore.

He has authored two computer books published by Tata McGraw Hill:

1. IBM PC and Clones: Hardware, Troubleshooting and Maintenance 2. Computer Architecture and Organization: Design Principles and Applications

He is working on a book titled Advanced Microprocessor Architectures. His interests include Tamil literature, spirituality and meditation.

Govindarajalu’s future plans include launching a technical magazine catering to the needs of engineering students. He may be contacted at bgrajulu@gmail.com.

B. Govindarajalu 

(Founder-CEO, MICROCODE, Chennai),

Dean (Computer Studies)

Rajalakshmi Engineering College

Chennai

Tata McGraw-Hill Publishing Company Limited NEW DELHI

McGraw-Hill Offices

New Delhi New York St Louis San Francisco Auckland Bogotá Caracas Kuala Lumpur Lisbon London Madrid Mexico City Milan Montreal San Juan Santiago Singapore Sydney Tokyo Toronto

Published by the Tata McGraw Hill Education Private Limited, 

7 West Patel Nagar, New Delhi 110 008.

Copyright © 2010, 2009, 2007, 2006, 2005 by Tata McGraw Hill Education Private Limited.

No part of this publication may be reproduced or distributed in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise or stored in a database or retrieval system without the prior written permission of the publishers. The program listings (if any) may be entered, stored and executed in a computer system, but they may not be reproduced for publication.

This edition can be exported from India only by the publishers,

Tata McGraw Hill Education Private Limited.

ISBN (13): 978-0-07-015277-9

ISBN (10): 0-07-015277-2

Vice President and Managing Director—Asia-Pacific Region: Ajay Shukla

Executive Publisher—Professional: R Chandra Sekhar

Production Executive: Rita Sarkar

Manager—Sales and Marketing: S Girish

Deputy Marketing Manager—Science, Technology & Computing: Rekha Dhyani

General Manager—Production: Rajender P Ghansela

Assistant General Manager—Production: B L Dogra

Information contained in this work has been obtained by Tata McGraw-Hill, from sources believed to be reliable. However, neither Tata McGraw-Hill nor its authors guarantee the accuracy or completeness of any information published herein, and neither Tata McGraw-Hill nor its authors shall be responsible for any errors, omissions, or damages arising out of use of this information. This work is published with the understanding that Tata McGraw-Hill and its authors are supplying information but are not attempting to render engineering or other professional services. If such services are required, the assistance of an appropriate professional should be sought.

Typeset at Script Makers, 19, A1-B, DDA Market, Paschim Vihar, New Delhi 110 063, and printed at Sai Printo Pack, A-102/4, Okhla Industrial Area, Phase II, New Delhi 110020.

Cover Design: Kapil Gupta

Cover Printer: Sai Printo Pack

RAZYCRBZDXQYY



To

my in-laws

Mr. N. Gopalakrishnan, my father-in-law and Mrs. Leelavathy Gopalakrishnan who have always encouraged and supported me

in all my new ventures

and

my recently discovered friend

Mr. G.R. Krishnamoorthy (EX OM, CCS Ltd. Chittoor)

whose simplicity, honesty and hardworking nature

have deeply impressed me



Foreword I

I am extremely happy to write the Foreword to the second edition of B. Govindarajalu’s book on Computer Architecture Organization: Design Principles and Applications. His earlier book, IBM PC and Clones: Hardware, Troubleshooting and Maintenance, was well received—not only in academic circles but also in the computer maintenance industry.

The first edition of this book was highly appreciated and lays emphasis upon learning of fundamental concepts through easily understandable text. Good selection of problems with their solutions helps students to grasp the intricacies of the subject and instill a sense of self study, creative thinking and review.

The second edition of this book covers important units of a computer in a very comprehensible manner and therefore, will be useful as a Level I book for undergraduate students pursuing courses in electrical engineering, electronics, computers and IT. All the chapters are concise and are supported by a detailed background. The author’s vast experience and sound theoretical and practical knowledge is reflected in the material covered. This book will serve as a valuable guide for students keen on mastering the organization of computer architecture.

Prof. S.S.S.P. Rao

Chief Mentor and Advisor, CMC Limited,

Hyderabad

Former Professor of Department of Computer Science and Engineering,

IIT Bombay

Visiting Professor of Department of Computer Science and Engineering,

IIT Hyderabad



Foreword II

There are many good books which cover the vast and ever-growing area of Computer Architecture but not all cover the entire gamut of emerging technologies in this domain. Also, at times the coverage as well as language do not suit Indian academic community.

Computer Architecture and Organization: Design Principles and Applications written by one of my colleagues, during my earlier stint, has addressed the issues lucidly and presented the technical content relevant to most of the Indian universities.

In the second edition, few additional chapters covering advanced topics such as Parallelism and Super Scalar Architecture have been included.

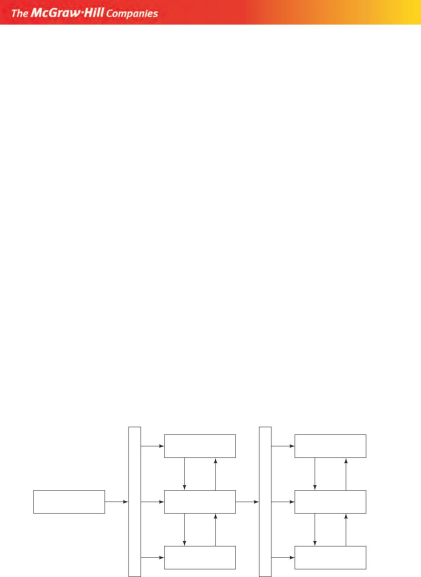
The author who has rich experience both in academia and industry has ensured that the overall pedagogic content is easy to follow and comprehend. I recommend this book for teachers, university students and professionals.

Dr. K. Sarukesi

Vice-Chancellor

Hindustan Institute of Technology and Science,

Chennai

Preface to the Second Edition 

Computer design or computer architecture is an interesting subject. With the rapid evolution of computer science, an increasingly diverse group of students is studying the subject of computer architecture. Though interesting and simple a majority of students consider it to be highly complex. Having worked both in industry and academic institutions with a range of computers starting from second generation systems, I decided to help in such a way that even a student of average IQ can understand it. The target audience of this book includes readers who want to learn basic computer organization as well as those who want to design a computer.

It can also be used by professionals as a reference. The book is intended as a text book for undergraduate students of computer science, information technology and electrical and electronics engineering. This book serves as a first level course on computer architecture/ organization. As a prerequisite, the reader is expected to be familiar with computer program ming. It is assumed that the reader does not know any specific programming language. In addition, exposure to digital electronics is helpful. For the benefit of those who have no knowledge of this, an annexure presents an overview of essential topics of digital electronics.

ORGANIZATION OF THE BOOK

This book is organized with a three layer structure shown here.

Basic Processor

Ch 4, 5, 6

Pipelining

Ch 12

Foundation Ch 1, 2, 3

Memory

Ch 7, 8, 9

Superscalar and

Advanced Processor

Ch 13, 14, 15

Layer 1

Foundation

I/O Subsystems

Ch 10, 11

Layer 2

Subsystems

Multiprocessors and Servers

Ch 16

Layer 3

Parallelism

xii Preface to the Second Edition

Though the chapters are organized in a linear sequence for reading, alternate layouts are possible. The first three chapters form the top layer that provides a foundation to computer architecture and organization. After studying these, the remaining chapters can be covered in any sequence shown in the figure. The middle layer consisting of chapters 4–11 describes the design of different subsystems of the computer. The last layer covers special topics that deal with parallelism and advanced computer architectures. This layer comprises chapters 12–16. This modular arrangement facilitates quick and standalone reference by professionals for select topics.

CHAPTER DESCRIPTION

Layer 1: Foundation

Chapter 1 introduces the basics of a present-day computer without describing the design aspects. It defines the computer's hardware and software layers and provides an overview of Interrupt Concept and I/O techniques. Chapter 1 also introduces techniques used in high performance computers in addition to system performance measurement.

Chapter 2 gives an overview of the history of computer and its evolution. Generations of computers are defined along with the new features of each generation. Chapter 3 describes the basic attributes of a computer such as instruction set, addressing modes and data types. It also distinguishes between RISC and CISC architectures.

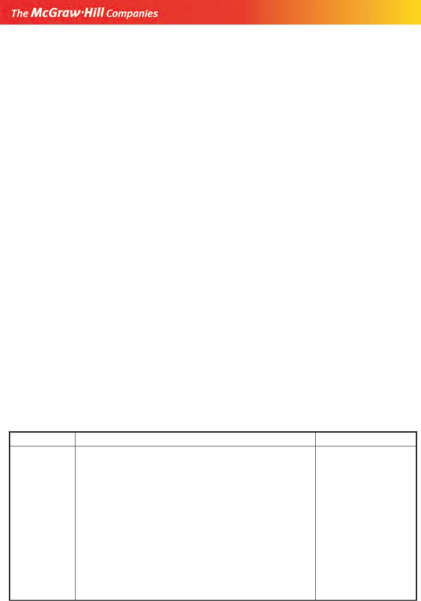
Layer 2: Subsystems

Chapter 4 focuses on data path organization and ALU design. Both fixed-point data path and floating-point data path are covered.

Chapter 5 describes the algorithms for fixed-point and floating-point arithmetic. It also defines serial and parallel adders.

Chapter 6 presents techniques of designing a control unit. Both hardwired and micro programmed control units are covered. It also describes micro operations and register transfer language. Single bus processor as well as multiple bus processor are discussed. Chapter 7 focuses on memory technologies and main memory design. Chapter 8 describes various memory enhancement techniques for tackling performance and capacity issues. Cache memory, virtual memory and memory interleaving are the major topics covered.

Chapter 9 deals with secondary storage devices such as magnetic disks and tapes apart from optical media. In addition, RAID levels and RAID controllers are discussed. Chapter 10 presents the sequence of communication between the internal units. Different I/O controllers, bus standards, Interrupt concept and I/O techniques are described. Chapter 11 describes the commonly used peripheral devices.

Preface to the Second Edition xiii

Layer 3: Parallelism and Advanced Architectures

Chapter 12 gives an elaborate study of parallelism and concurrency in computers with special emphasis on pipelining. The concept of RISC processors is also covered here. Chapters 13–16 describe advanced architectures found in uniprocessing and multiprocessing. Chapter 13 deals with the superscalar architecture of uniprocessors. Dynamic scheduling techniques are discussed.

Chapter 14 describes VLIW and EPIC architectures.

Chapter 15 deals with vector computing and array processing.

Chapter 16 describes multiprocessors and cache coherency in addition to reliability and fault tolerance concepts and techniques.

Additions in Second Edition

The Second Edition aims to:

· Enhance the suitability of the book according to the syllabi of the UG course on Computer Organization and Architecture offered by most universities, especially Anna University, Chennai.

· Update the technical content according to the changes in the industry since the publication of the first edition in 2003.

· Improve quality.

Two major changes have been carried out in the Second Edition:

· Chapters 11 and 12 of the first edition have been completely revised and reorga nized in the second edition as five different chapters (12 to 16).

· Part of the contents of Chapter 10 in the first edition, dealing with magnetic disk, tape and optical disk, have been shifted to Chapter 11 in the second edition. In addition, several minor additions have been made to the contents in almost all the chapters. Some of these are identified in the following table:

Chapter no. Topics added/modified Relevant section

1 System Performance Measurement and 1.13 Response Time

3 Hardware-Software Interface 3.8 4 Floating-point Numbers 4.4 5 Building Long Adder 5.3 6 Synchronous and Asynchronous Control Unit 6.7 6 Clocking and Synchronization 6.3 6 Single Cycle and Multicycle Design 6.3 6 Single Bus and Multiple Bus Processor 6.4, 6.5

9 End-to-End Data Protection 9.6.8 9 RAID 9.7 9 Magnetic Tape 9.8

10 PCI Bus 10.8 10 SCSI 10.8 (Contd.)

xiv Preface to the Second Edition

(Continued )

Chapter no. Topics added/modified Relevant section

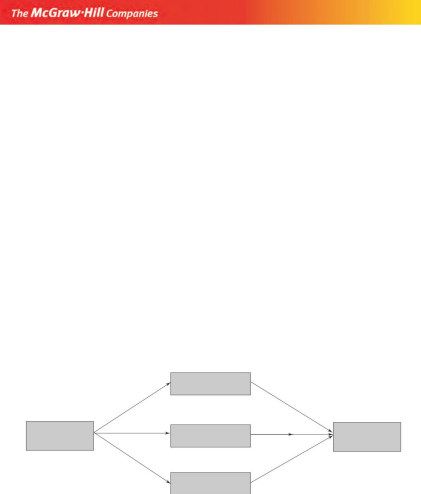
10 USB 10.8 11 Laser Printer 11.4 11 Inkjet Printer 11.4 12 RISC Pipeline 12.8 12 Operand Forwarding 12.9 12 Exception Handling 12.12

13 Loop Unrolling 13.7 13 Scoreboard 13.9 13 Tomasulo Algorithm 13.10

I welcome comments and suggestions from the readers for improvement of the book.

B. GOVINDARAJALU

bgrajulu@gmail.com

Preface to the First Edition 

Today computers find application in all spheres of life. And an increasingly diverse group of students are studying the subject. Though an interesting and simple subject, a majority of students known to me consider it highly complex, consisting of complicated and advanced topics. As I have worked with a range of computers, from the second generation to the modern systems both in the industry and the academia, I felt a need to provide a helping hand to such students. My aim is to simplify the subject and make it easy even to an average student.

This book serves as a first level course on computer architecture and organization. As a pre-requisite, the reader is expected to be familiar with computer programming. In addition to this, exposure to digital electronics will be helpful. For the benefit of those who have no prior knowledge of this, the book includes an overview of the essential topics of digital electronics in Annexure 3.

This book is organized in five modules with a three-layer structure shown in the following figure.

Processor 4, 5, 6

Memory 7, 8 Foundation

1, 2, 3 Parallelism 11,12

I/O 9, 10

The first three chapters form the top layer that provides a base to computer architecture and organization. After studying this, the remaining chapters can be covered in any of the three different paths shown in the figure. The middle layer, consisting of Chapters 4—10, describes the design of the different sub-systems of the computer. The last layer, consisting of Chapters 11 and 12, covers special topics that deal with parallelism and high performance of advanced computer architectures. This modular arrangement facilitates quick and stand-alone reference of selected topics by the professionals.

xvi Preface to the First Edition

Chapter 1 introduces the basics of the present day computer without dwelling on the design aspects. It defines the hardware and software layers in a computer and provides an overview of interrupt concept and I/O techniques.

Chapter 2 gives an overview of the history of computers and their evolution and defines the different generations of computers.

Chapter 3 describes the basic attributes of a computer such as instruction set, addressing modes and data types. It also distinguishes between RISC and CISC architectures. Chapter 4 describes the algorithms for fixed-point and floating-point arithmetic. It also defines serial and parallel adders.

Chapter 5 focuses on the datapath organization and ALU design. It also describes micro-operations and register transfer language.

Chapter 6 presents the techniques of designing a control unit. Both hardwired and microprogrammed control unit are covered.

Chapter 7 focuses on the memory technologies and main memory design. Chapter 8 describes the various memory enhancement techniques for tackling performance and capacity issues. Cache memory, virtual memory and memory interleaving are the major topics covered.

Chapter 9 presents the sequence of communication between the internal units. Different I/O controllers, bus standards, interrupt mechanism and I/O techniques are described.

Chapter 10 describes the commonly used peripheral devices.

Chapter 11 gives an elaborate study of parallelism and concurrency in computers, with a special emphasis on pipelining and vector computing.

Chapter 12 describes advanced architectures found in uni-processing and multi processing. RISC, Superscalar and VLIW architectures of uni-processor are covered. Multi-processors and cache coherency are also discussed, in addition to reliability and fault tolerance concepts and techniques.

I look forward to the comments and suggestions from the readers for further improving the book.

B. GOVINDARAJALU

Acknowledgements

When my first book IBM PC and Clones: Hardware, Troubleshooting and Maintenance was completed in 1991, I had no idea that I would be writing a book on Computer Architecture. The need for this book was felt during the special coaching I conducted in 2002 for a small group of students of Anna University, Chennai. I thank all my students of the past 30 years whose interactions have motivated me constantly to keep abreast of computer architecture and hardware concepts. I would like to extend special thanks to Ms Jamuna Maruthachalam who along with her classmates formed the pilot batch for this book.

I believe that my ability to author a book of this genre came from the opportunities and experiences I had in my career. To a large extent, my stay at IIT Bombay, both as a student and as an employee, had a strong impact on my technical and academic skills. I also thank Prof. James R Isaac and Dr. SSSP Rao, whose guidance from 1973 to 1979, at IIT Bombay, helped lay down the basic foundation for this book.

I sincerely thank our college management, especially Shri S. Meganathan, correspond ent and Dr.(Mrs.) Thangam Meganathan, chairperson, for providing excellent infrastruc ture and environment in Rajalakshmi Engineering College that was instrumental in suc cessfully bringing out the second edition.

I thank my ex-colleagues at Microcode (while work was in progress on the first edition) and present colleagues at Rajalakshmi Engineering College who supported me during the course of authoring this book.

I am also thankful to the Professional team of Tata McGraw Hill, for their encouragement and support, right from conception to publishing of this book in a short span of time. Writing a book requires extensive support from family. I thank my son Krishnakumar, and my daughter Padma Ananth, they reviewed certain portions and offered their comments. I also thank my wife Bhuvaneswari, and my daughter-in-law Manjula Krishnakumar, who assisted me in several ways, especially while I was preparing the figures. Special thanks to my grand son Vihaan, who frequently hindered progress of the book by his mischievous antics. It was he who provided unlimited entertainment and proved to be a welcome distraction whenever I needed it while authoring the second edition.

B. GOVINDARAJALU

Contents

Foreward I vii Foreward II ix Preface to the Second Edition xi Preface to the First Edition xv Acknowledgements xvii Abbreviations xxvii

1. Basic Computer Organization 1

1.1 Introduction 3

1.2 Man and Computing 3

1.3 Digital Computer 5

1.4 Computer Organization and Functional Units 9

1.5 Main Memory 16

1.6 CPU Operational Concepts 22

1.7 Interrupt Concept 28

1.8 I/O Techniques 29

1.9 Bus Concept 31

1.10 System Software 35

1.11 Computer Types 36

1.12 Computer Performance Factors 39

1.13 System Performance Measurement 40

1.14 High Performance Techniques 43

1.15 Computer Design Process 49

1.16 Computer Structure 50

1.17 Computer Function 53

1.18 Architecture and Organization 55

SUMMARY 56

REVIEW QUESTIONS 58

EXERCISES 59

2. Evolution of Computers 61

2.1 Introduction 63

2.2 Dimensions of Computer Evolution 63

xx Contents

2.3 History of Computers 71

SUMMARY 85

REVIEW QUESTIONS 85

3. CPU Architecture and Instruction Set Principles 87

3.1 Introduction 89

3.2 Processor Role 89

3.3 Processor Design Goals 91

3.4 Processor Design Process 93

3.5 Datapath Organization 98

3.6 Control Unit Organization 98

3.7 Instruction Sequencing 99

3.8 Hardware–Software Interface 103

3.9 CISC vs RISC 105

3.10 Instruction Set Design 108

3.11 Instruction Types and Operations 118

3.12 Addressing Modes 123

3.13 Data Representation 130

3.14 Binary Data 131

SUMMARY 132

REVIEW QUESTIONS 134

EXERCISES 134

4. ALU Design and Datapath 137

4.1 Introduction 139

4.2 Arithmetic Types 139

4.3 Fixed-Point Numbers 140

4.4 Floating-Point Numbers 143

4.5 Fixed-Point Datapath 147

4.6 Design of Arithmetic Unit 153

4.7 Design of Logic Unit 154

4.8 Design of Shifter 155

4.9 ALU Design 157

4.10 Typical Minicomputer Datapath 159

4.11 Typical Mainframe Datapath 161

4.12 Main Memory Interface 162

4.13 Local Storage/Register File 163

4.14 Datapath for Simple Instructions 163

Contents xxi

4.15 Floating-Point Unit Datapath 169

4.16 Advanced Processors and Datapaths 173

SUMMARY 173

REVIEW QUESTIONS 174

EXERCISES 175

5. Binary Arithmetic Operations 177

5.1 Introduction 179

5.2 Hardware Components 179

5.3 Fixed-Point Arithmetic 188

5.4 Floating-Point Arithmetic 225

SUMMARY 235

REVIEW QUESTIONS 235

EXERCISES 236

6. Instruction Processing and Control Unit 237

6.1 Introduction 239

6.2 Basic Processing Concepts 239

6.3 Instruction Cycle and Processor Actions 244

6.4 Single-bus Processor 252

6.5 Multiple-bus Processor 264

6.6 Control Unit Design Options 267

6.7 Hardwired Control Unit 267

6.8 Microprogrammed Control Unit 276

SUMMARY 293

REVIEW QUESTIONS 294

EXERCISES 295

7. Main Memory Design 297

7.1 Introduction 299

7.2 Memory Parameters 299

7.3 Classification of Memory 301

7.4 ROM Subsystem Design 307

7.5 Static RAM IC 307

7.6 Dynamic RAM 313

7.7 Main Memory Allocation 327

7.8 Memory Hierarchy and Performance 329

xxii Contents

SUMMARY 331

REVIEW QUESTIONS 332

EXERCISES 332

8. Memory Management Techniques 333

8.1 Introduction 335

8.2 Main Memory Drawbacks 335

8.3 Instruction Prefetch 336

8.4 Memory Interleaving 338

8.5 Write Buffer 341

8.6 Cache Memory 342

8.7 Virtual Memory 368

8.8 Associative Memory 378

SUMMARY 380

REVIEW QUESTIONS 382

EXERCISES 383

9. Secondary Storage Devices 385

9.1 Introduction 387

9.2 Magnetic Storage Devices 387

9.3 Basic Principle of Magnetic Disk 388

9.4 Floppy Diskette 393

9.5 Overall Operation of Floppy Disk Subsystem 400

9.6 Hard Disk Drive (HDD) 404

9.7 RAID / DISK ARRAYS 431

9.8 Magnetic Tape Data Storage 437

9.9 Optical Disk Storage 446

SUMMARY 451

EXERCISES 451

10. I/O Concepts and Techniques 453

10.1 Introduction 455

10.2 Accessing I/O Devices 455

10.3 Interrupt Handling 466

10.4 I/O Techniques in PC 480

10.5 Bus Arbitration Techniques 496

10.6 Bus Concepts 500

10.7 Interface Circuits and I/O Controllers 528

10.8 Modern Standard I/O Interfaces 556

Contents xxiii

SUMMARY 596

REVIEW QUESTIONS 598

EXERCISES 599

11. I/O Devices 601

11.1 Introduction 603

11.2 Keyboard 603

11.3 CRT Display Monitor 608

11.4 Printer 613

11.5 Special Types of Disk Drives 625

11.6 Mouse and Trackball 627

11.7 Modem 628

11.8 Scanner 629

11.9 Digital Camera 630

11.10 Special Peripherals 630

SUMMARY 637

REVIEW QUESTIONS 637

EXERCISES 637

12. RISC, Parallelism and Pipelining 639

12.1 Introduction 641

12.2 RISC Systems and CISC Drawbacks 641

12.3 Techniques of RISC Systems 643

12.4 RISC Architecture in Modern Processors 646

12.5 Performance Enhancement Strategies 648

12.6 Classification of Parallelism 651

12.7 Multiple Functional Units 655

12.8 Pipelining 657

12.9 Pipeline Hazards 668

12.10 Influence on Instruction Sets 683

12.11 Data Path and Control Considerations 684

12.12 Exception Handling 684

SUMMARY 685

REVIEW QUESTIONS 686

EXERCISES 687

13. Superscalar Architecture 689

13.1 Introduction 691

13.2 Concept of Superscalar Architecture 691

xxiv Contents

13.3 Case Study 1—Intel Pentium Processor 696

13.4 Out-of-Order Execution 700

13.5 Register Renaming 701

13.6 Speculative Execution and Branch Prediction 704

13.7 Loop Unrolling 705

13.8 Dynamic Scheduling and Static Scheduling 707

13.9 Thornton Technique and Scoreboard 710

13.10 Tomasulo Algorithm and Reservation Stations 717

13.11 Case Study 2—Intel Pentium-pro Processor 730

13.12 NetBurst Architecture and Pentium 4 736

13.13 The Alpha Family 738

13.14 PowerPC 739

13.15 SPARC 740

SUMMARY 741

14. VLIW and EPIC Architecture 743

14.1 Introduction 745

14.2 VLIW Architecture 745

14.3 VLIW Compiler Techniques 746

14.4 VLIW vs Superscalar Processor 749

14.5 Tree Instruction 751

14.6 EPIC Architecture 752

SUMMARY 755

15. Vector Computing and Array Processing 757

15.1 Introduction 759

15.2 Vector Computing 759

15.3 Array Processor 769

SUMMARY 770

16. Multiprocessor Systems and Servers 773

16.1 Introduction 775

16.2 Classification of Multi-processors 775

16.3 Symmetric Multi-Processor (SMP) 777

16.4 Interconnection Structures 778

16.5 Clusters 782

16.6 Cache Coherence Problem 783

16.7 Fault Tolerance 786

Contents xxv

16.8 Server Systems 792

SUMMARY 797

Annexure 1 Number Systems and Conversions 799

A1.1 Introduction 799

A1.2 Common Number Systems 799

A1.3 Decimal Number System 801

A1.4 Conversion of Decimal Numbers into Other Number Systems 801 A1.5 Conversion of Other Systems into Decimal Number System 803

Annexure 2 Types of Computer Software 804

A2.1 Introduction 804

A2.2 System Software Types 804

A2.3 Programming Software 805

A2.4 Application Software 805

Annexure 3 ASCII Code 809

Annexure 4 Hardware Components and ICs 818

A4.1 Introduction 818

A4.2 Hardware Components: Discrete and Integrated 818

A4.3 Pulse Circuits and Waveforms 819

A4.4 Positive and Negative Logic 821

A4.5 High Active and Low Active Signals 821

A4.6 Combinational and Sequential Circuits 822

A4.7 Simple Gates 822

A4.8 Special Gates 823

A4.9 IC Families 824

A4.10 TTL Characteristics 827

A4.11 Open Collector TTL 828

A4.12 Tri-State Gates 829

A4.13 MOS Families 830

A4.14 Interfacing Logic Families 831

A4.15 Latch 832

A4.16 Flip-Flop 834

A4.17 Register 838

xxvi Contents

A4.18 Shift Register 838

A4.19 Counters 839

A4.20 Multiplexer 842

A4.21 Demultiplexer 844

A4.22 Decoder 845

A4.23 Encoder 846

A4.24 74S280 Parity Generator/Checker 847

Index 849

ABBREVIATIONS

mR Micro Routine

ms Micro second

hs Nano second

AC Accumulator

ACC Accumulator

ACK Acknowledgment

ADDR SEL Address Select

AGU Address Generation Unit

AHOLD Address Hold

ALU Arithmetic and Logic Unit

AOI And-Or-Inverter

AR Address Register

ASCII American Standard Code for Information Interchange ASIC Application Specific Integrated Circuit AT Advanced Technology

ATC Advanced Transfer Cache

b Bit

B Byte

BA Branch Address

BAL Bus Arbitration Logic

BCD Binary Coded Decimal

BCZ Byte Count Zero

BG Bus Grant

BIOS Basic Input-Output Control System

BIST Built-in Self Test

BOT Beginning Of Tape

BPI Bits Per Inch

BPL Branch Prediction Logic

BR Bus Request

BTB Branch Target Buffer

xxviii Abbreviation

BUN Branch Unconditionally

BZ Branch If Zero

C Carry

CAD Computer Aided Design

CAM Content Address Memory

CAS Column Address Strobe

CAW Channel Address Word

CCD Charge Coupled Device

CCW Channel Command Word

CD Compact Disc

CDB Common Data Bus

CD-R Compact Disc Recordable

CDRAM Cache DRAM

CD-ROM Compact Disc Read Only Memory CD-RW Compact Disc Rewritable

CISC Complex Instruction Set Computing CLA Carry Look-Ahead Adder

CLK Clock

CLV Constant Linear Velocity

CM Control Memory

CMAR Control Memory Address Register CMDR Control Memory Data Register CPI Cycles Per Instruction

CPS Characters Per Second

CPU Central Processing Unit

CR Carriage Return/Control Register CRC Cyclic Redundancy Check

CRCC Cyclic Redundancy Check Character CRT Cathode Ray Tube

CRTC CRT Controller

CU Control Unit

CWP Common Window Printer

DAT Digital Audio Tape

DCB Device Control Block/Device Command Block DCE Data Communication Equipment DDR SDRAM Double Data Rate Synchronous DRAM DEC Digital Equipment Corporation

Abbreviations xxix

DI Disable Interrupt

DIB Dual Independent Bus

DIMM Dual In-line Memory Module

DMA Direct Memory Access

DMA ACK DMA Acknowledgement

DMA REQ DMA Request

DMP Dot Matrix Printer

DPL Deep Pipeline Latency

DPU Data Processing Unit

DR Dynamic Reconfiguration/Data Register

DRAM Dynamic Random Access Memory

DRQ Data Request

DSDD Double-sided Double Density

DSHD Double-sided High Density

DSM Distributed Shared Memory

DSQD Double-sided Quad Density

DTE Data Terminal Equipment

DTP Desk Top Publishing

DVD Digital Versatile Disk

EADS Valid External Address

EAROM Electrically Alterable ROM

EBCDIC Extended Binary Coded Decimal Interchange Code ECC Error Checking and Correcting Code

ECP Extended Capability Port

EDC Electronic Digital Computer

EDP End-to-end Data Protection

EDVAC Electronic Discrete Variable Computer

EEPROM Electrically Erase Programmable ROM

EI Enable Interrupt

EIDE Enhanced IDE

EISA Extended Industry Standard Architecture

ENIAC Electronic Numeric Indicator and Computer

EO Execute Operation

EPIC Explicitly Parallel Instruction Computing

EPP Enhanced Parallel Port

EPROM Erase Programmable Read Only Memory

ESC Escape Sequence

xxx Abbreviation

EU Execution unit

f Frequency

FA Full Adder

FDC Floppy Disk Controller

FDD Floppy Disk Drive

FF Form Feed

FIFO First-In First-Out

FM Frequency Modulation

FPM Fast Page Mode

FRC Functional Redundancy Check FSK Frequency Shift Key

FU Fetch unit

GMR Giant Magneto Resistance

GPG Graphics Performance Characterization Group GPR General Purpose Register

GPU Graphics Processing Unit

GT Grant

GUI Graphical User interface

HA Half Adder

HDA Head Disc Assembly

HDC Hard Disk Controller

HDD Hard Disk Drive

Hexa Hexa Decimal

HLL High Level Language

HLT Halt

HMT Hardware Multi-Threading

HPG High Performance Group

HSYNC Horizontal Synchronization

Hz Hertz

I/O Input/Output

IAS Institute for Advanced Studies IB Instruction Buffer

IBM International Business Machines IBR Instruction Buffer Register

IC Integrated Circuit

ID Identifier/Instruction Decode

IDE Integrated Drive Electronics

Abbreviations xxxi

IE Interrupt Enable

IEEE Institute of Electrical and Electronics Engineers

IF Instruction Fetch

ILP Instruction Level Parallelism

IN Input

INC Increment

INTA Interrupt Acknowledgement

INTR Interrupt Request

IOAR I/O Address Register

IODR I/O Data Register

IOR I/O Read

IOW I/O Write

IR Instruction Register

IRET Return from Interrupt

IRG Inter-record Gap

IRQ Interrupt Request

ISA Industry Standard Architecture

ISR Interrupt Service Routine

IU Instruction Unit

KB Kilo Byte

KEN Cache Enable

KHz Kilo Hertz

KISS Keep It Short and Simple

LAN Local Area Network

LCD Liquid Crystal Display

LDA Load Accumulator

LED Light Emitting Diode

LF Line Feed

LFU Least Frequently Used

LM Local Memory

LMSW Load Machine Status Word

LPM Lines Per Minute

LQP Letter Quality Printer

LRU Least Recently Used

LSAR Local Storage Address Register

LSB Least Significant Bit

LSD Least Significant Digit

xxxii Abbreviation

LSDR Local Storage Data Register

LSH Least Significant Half

LSI Large Scale Integration

LSR/W Local Storage Read / Write Flag

MA Memory Address

MAR Memory Address Register

MB Mega Byte

MBR Memory Buffer Register

MCMAR Microprogram Control Memory Address Register MD Multiplicand

MDR Memory Data Register

MEMR Memory Read

MEMW Memory Write

MFC Memory Function Complete

MFM Modified Frequency Modulation

MHz Mega Hertz

MIG Metal In Gap

MIMD Multiple Instruction Stream, Multiple Data Stream MIPS Millions of Instructions Per Second MISD Multiple Instruction Stream, Single Data Stream ML Machine Language

MM Main Memory

MMU Memory Management Unit

MMX Multi-media Extension

MOB Memory Order Buffer

MODEM Modulator cum Demodulator

MQ Multiplier-Quotient

MR Memory Read/Magneto Resistance ms Milli Second

MSB Most Significant Bit

MSH Most Significant Half

MSI Medium Scale Integration

MTBF Mean Time Between Failures

MTTR Mean Time To Repair

MUX Multiplexor

MW Memory Write

NA Next Address

Abbreviations xxxiii

NAK Negative Acknowledgement

NAS Network Attached Storage

NCMAR Nanoprogram Control Memory Address Register NDP Numeric Data Processor

NLQ Near Letter Quality

NMA Next Microinstruction Address

NMI Non Maskable Interrupt

NNA Next Nanoinstruction Address

NOOP No Operation

NUMA Non-uniform Memory Access

OAC Operand Address Calculation

OCR Operation Code Register

OF Operand Fetch

OOO Out–Of–Order Execution

OS Operating System

OSG Open System Group

OUT Output

PC Personal Computer (or) Program Counter

PCB Printed Circuit Board

PCI Peripheral Component Interconnect

PCU Program Control Unit

PD Pre Decode

PF Pre Fetch

PIC Programmable Interrupt Controller

PLL Phase Lock Loop

PMR Perpendicular Magnetic Recording

PnP Plug and Play

POE Plan Of Execution

POR Power-On Reset

POST Power-On Self-Test

PPD Parallel Presence Detect

PPM Pages Per Minute

PR Processor

PROM Programmable Read Only Memory

PS Program Status

PSK Phase Shift Key

PSW Program Status Word

xxxiv Abbreviation

Q Quotient

QIC Quarter Inch

QS Queue Status

RA Register

RAID Redundant Array of Independent Disks

RAM Random Access Memory (or) Read/Write memory RAMDAC Random Access Memory Video Digital–to–Analog Converter RAS Reliability, Availability and Serviceability / Row Address Strobe RAT Register Alias Table

RAW Read After Write

Rclk Receive Clock

RDAT Rotary Head Digital Audio Tape

RDRAM Rambus DRAM

RISC Reduced Instruction Set Computing

RLL Run Length Limited

ROM Read Only Memory

RQ Request

RRF Retirement Register File

RTL Register Transfer Language (or) Register Transfer Level RV Reset Vector

RWM Read Write Memory

S Sign/Sum

SAN Storage Area Network

SATA Serial ATA

SC Sequence Counter

SCSI Small Computer System Interface

SDRAM Synchronous Dynamic RAM

SDT Segment Descriptor Table

SEL Select

SIMD Single Instruction stream, Multiple Data stream SIMM Single Inline Memory Module

SIN Serial In

SIPO Serial In Parallel Out

SIS Start Interrupt Service

SISD Single Instruction Stream, Single Data Stream SLSA Start Local Storage Access

SMI System Management Interrupt

Abbreviations xxxv

SMMA Start Main Memory Access

SMP Symmetric Multiprocessor

SOHO Small Office Home Office

SOUT Serial Out

SP Stack Pointer

SPARC Scalable Processor Architecture

SPD Serial Presence Detect

SPEC System Performance Evaluation Corporation

SPM Scratch-Pad Memory

SPP Standard Parallel Port

SR Store Result

SRAM Static Random Access Memory

SSE2 Streaming SIMD Extensions 2

SSI Small Scale Integration

STA Store Accumulator

SUB Subtract

T Time Period / Temporary Register

TB Tera Byte

TC Terminal Count

Tclk Transmit Clock

TLB Translation Look Aside Buffer

TPI Tracks Per Inch

Tpw Pulse width

UART Universal Asynchronous Receiver Transmitter

UMA Uniform Memory Access

UNIVAC Universal Automatic Computer

USB Universal Serial Bus

VCD Video CD

VESA Video Electronics Standard Association

VGA Video Graphics Array

VLIW Very Long Instruction Word

VLSI Very Large Scale Integration

VRAM Video RAM

VSYNC Vertical Synchronization

WAR Write After Read

WAW Write After Write

WB Write Back

xxxvi Abbreviation

WE Write Enable WIW Wide Issue Width XOR Exclusive OR XR Transceiver Z Zero

CHAPTER 1 

BASIC COMPUTER ORGANIZATION

7, 8, 9

Memory

1

Basics

History 2

3 4, 5, 6

Instruction

set

12 to 61

Advanced

topics CPU

I/O

1 11 0,

Chapter Flow Diagram

OBJECTIVES

Stored program concept Basic structure of computer Functional units

Basic operational concepts Hardware

Software

Operating system

Memory addressing

Memory operations

Bus structures

Computer types

Performance measurement High performance techniques

2 Computer Architecture and Organization: Design Principles and Applications KEYWORDS

Memory Access Time Clock Frequency

Instruction Cycle

Memory Address Space Word length

Random Access Memory Volatile Memory

Booting

Mainframe Computer Minicomputer

Microcomputer

Supercomputer

Workstation

Personal Computer Portable Computer Application Software System Software

Operating System

Compiler

Interpreter

Machine Language High Level Language I/O Port

Hardware

Software

Microprocessor

Architecture

Organization

Multiprogramming Cache Memory

General purpose register Multiprocessing

Pipelining

SPEC Rating

Assembly Language

Basic Computer Organization 3 

1.1 Introduction

Today computers are used in almost all walks of life: banks, hospitals, schools, shops, libraries, factories, courts, universities, prisons, etc. Computer buyers consider a number of factors, before choosing a product, such as good performance, low cost, easy programmability, low maintenance cost, less downtime, etc.

The design of a computer is highly sophisticated and complex. Over the years, a variety of concepts and techniques have been applied in designing and building it. The study of these concepts and techniques is the objective of this book.

This chapter serves as a foundation to the following chapters. It outlines the basic organization of a computer in order to prepare the reader for learning the various architectural and design issues covered in following chapters.

1.2 Man and Computing

The word computing means ‘an act of calculating’. The human race has so far seen three different types of computing:

1. Fully manual computing using brain and fingers.

2. Manual computing using simple tools such as slide rule, abacus, etc. 3. Automatic computing using a computer.

Table 1.1 compares the three types.

TABLE 1.1 Comparison of computing types

S. no. Parameter Manual Manual Automatic computing computing computing

with simple tools

1 Speed Low Medium High 2 Reliabilty Poor; varies Medium High with individual

3 Problem complexity Low Medium High possible

4 Extent of human effort Very high Medium Very low 5 Consistency Poor due to Medium Very high tiredness or

moodiness

6 Impact Some problems Slightly better Any reasonable cannot be than manual problem can

solved in computing be solved

reasonable time

4 Computer Architecture and Organization: Design Principles and Applications 

The computer is a tool for solving problems (Fig. 1.1). Its basic ability is to perform arithmetic calculations. The computer is used to solve problems in several fields such as scientific, research, commerce, administration, manufacturing, etc. The following are some simple uses:

Problem Solution Computer

User

Fig. 1.1 User and computer

1. Calculating the average mark of students of a class.

2. Preparing monthly salary statement for a factory.

3. Designing a building.

4. Calculating the tax payable by a business firm.

5. Preparing materials list for production shop floor.

6. Preparing the results of an university examination.

The various advantages of using a computer are as follows:

1. Quick processing (calculations)

2. Large information storage

3. Relieving manual efforts

4. Messaging and communication

Combination of processing and storage has resulted in several new types of uses such as Multimedia, Internet, Desk Top Publishing (DTP), etc. in recent years.

1.2.1 Characteristics of a Computer

The main characteristics of a computer are listed below:

1. Very high speed of computation.

2. Consistency of behavior, unaffected by fatigue, boredom, likes and dislikes, etc.

Basic Computer Organization 5 

3. Large storage capacity (for data and programs).

4. High accuracy of computation.

5. General purpose machine which can be programmed as per user’s requirement.

1.3 Digital Computer

Most of the present day computers are digital computers though some are also analog computers. An analog computer monitors (senses) input signals whose values keep changing continuously. It generally deals with physical variables such as voltage, pressure, temperature, speed, etc. A digital computer operates on discrete (digital) information such as numbers. It uses binary number system in which there are only two digits 0 and 1. Each is called a bit (binary digit). Annexure 1 discusses the different number systems: decimal, binary, octal and hexa decimal. The digital computer is designed using digital circuits in which there are two different levels for an input or output signal. These two levels are known as logic 0 and 1. The modern computer is a digital computer capable of performing arithmetic (mathematical) and logical operations on data and generates a result (Fig. 1.2).

Input data Output result

Computer circuits

Computer

Fig. 1.2 Computer as an electronic machine

1.3.1 Program

The computer behaves as directed by the ‘program’ developed by the programmer. To solve a problem, the user inputs a program to the computer (Fig. 1.3). The program has a sequence of instructions which specify the various operations to be done. Thus, the program gives a step-by-step method to solve a problem. A computer can carry out a

Program

User Result

Computer

Fig. 1.3 Solving a problem

6 Computer Architecture and Organization: Design Principles and Applications 

variety of instructions such as ADD, SUBTRACT, etc. The entire list of instructions for a computer is known as instruction set. A program uses combinations of the instructions ac cording to the method (algorithm) to solve the problem. A program can be defined as a sequence of instructions with appropriate data for solving a problem (Fig. 1.4). The compu ter analyses each instruction and performs action on data.

1.3.1.1 Machine Language Program

The digital computer is an electronic machine. Hence, the computer program should con sist of only 1’s and 0’s. The operations are specified in the instructions as binary informa

tion. The operands (data) are given as binary numbers. Such a program is called a machine language program. Though it is tedious to write programs in machine lan guage, all early programs were written in machine lan guage as there was no other choice. Different computer models had different machine languages and hence, a machine language program for one computer can not run on any other computer with different instruction set. Present day programmer uses high level languages due to simplicity of the high level languages. A high level language program consists of statements whereas a ma

Instructions Data

Add

Move

Multiply Store

Jump

3000

6000

8

3

chine language program has instructions. The compu ter converts the statements into instructions.

1.3.2 Hardware and Software

Fig. 1.4 A computer program

The term hardware generally refers to the electronic circuits in the computer. The main hardware modules are shown in Fig.1.5. In practice, the term hardware is used for all physi cal items in a computer including mechanical, electrical and electronic assemblies and com ponents. Figure 1.6 shows different hardware parts in a computer.

Hardware

System box Keyboard CRT monitor Disk drive Printer Other

peripherals

Fig. 1.5 Major hardware modules

Basic Computer Organization 7

Computer components

Electrical Mechanical Electronics

1. Motors 

2. Power supplies 3. Transformers 4. Relays

5. Fans

6. PCBs

7. Wires

8. Cables

1. Switches 2. Panels 3. Covers 4. Chassis 5. Nuts

6. Screws

1. Resistors

2. Capacitors 3. Coils

4. Diodes

5. Transistors 6. IC

7. Crystals

8. LED

9. Optoisolators 10. CRT

11. Speaker

12. Photosensors 13. Delay lines

Fig. 1.6 Hardware components of a computer

Any program is a software. The software is developed to solve a problem and it controls the hardware when the program is executed. In other words, the hardware obeys the soft ware as shown in Fig. 1.7. The hardware can be seen visually whereas the software is a logical action plan that is not visually noticeable. Here, when we say hardware, we don’t mean the physical units but the functional units.

Instruction

Software

Result Hardware

Program Circuits

Fig. 1.7 Hardwaresoftware interface

1.3.3 Layers in Modern Computer

A modern computer is not a mere electronic machine. It is a system consisting of multiple layers. The innermost layer is the hardware unit that is surrounded by other layers of software as shown in Fig. 1.8. A programmer writes an application program in a high level language using decimal numbers and English statements. The compiler is a language translator which converts the high level language program into equivalent machine language program, consisting of instructions and binary numbers (Fig. 1.9).

8 Computer Architecture and Organization: Design Principles and Applications 

OS,

Compiler

Hardware

BIOS

BIOS Basic input output

æ

control s

ystem

User application program

OS Operating system æ

Fig. 1.8 Layers in a computer

The operating system is a set of programs that provide a variety of functions with the objective of offering an efficient and friendly environment to the users and programmers. The following are important functions of operating system:

Machine language

HLL statements

Compiler

instructions

Fig. 1.9 Use of compiler

1. Handling computer user requests for various services

2. Scheduling of programs

3. Managing I/O operations

4. Managing the hardware units

The Basic Input-Output control System (BIOS) is collection of I/O drivers (programs for performing various I/O operations) for different peripheral devices in the computer. These programs can be called by other programs whenever an I/O operation has to be done. The BIOS programs are requested by other programs to perform I/O operations whenever needed. Figure 1.10 illustrates the role of BIOS in a computer system. Chapter 10 discusses the three–tier interface between the device controller and the application program.

Basic Computer Organization 9 Device

DOS

Application program

Call Command

BIOS I/O

controller

Return Status

interface

Device

Fig. 1.10 Three-tier interface 

1.3.4 Application Software and System Software

Computer software is classified into two types: application software and system software. An application program is a program for solving an users problem. Some typical examples are: payroll program, inventory control program, tax calculator, class room scheduler, library management software, train reservation software, billing software and game programs. Each of these programs is an application program since its objective is tackling a specific application (user requirement). A system program is a program which helps in efficient utilization of the system by other programs and the users. It is generally developed for a given type of computer and it is not concerned with specific application or user. The operating system and compiler are examples of system software. Annexure 2 provides additional information regarding various types of application software and system software.

1.4 Computer Organization and Functional Units

A modern computer is a computer system consisting of hardware and software. The hardware has five different types of functional units (Fig. 1.11): memory, arithmetic and logic unit (ALU), control unit, input unit and output unit. The program and data are

Memory

Input unit

CPU

Output unit

Control unit ALU Control signal Instruction/data

Fig. 1.11 Functional units in computer

10 Computer Architecture and Organization: Design Principles and Applications 

entered into the computer through the input unit. The memory stores the program and data. The control unit fetches and analyzes the instructions one-by-one and issues control signals to all other units to perform various operations. The ALU is capable of performing several arithmetic and logical operations. For a given instruction, the exact set of operations required is indicated by the control signals. The results of the instructions are stored in memory. They are taken out of the computer through the output unit. A computer can have several input units and output units. Figures 1.12(a) and (b) show examples of various input and output units. Table 1.2 gives brief summary of the functions of these devices. Some of them have dual functions: input and output.

TABLE 1.2 Peripheral devices and functions

S. No. Peripheral device Function/operation Remarks

1 Keyboard Detects the key pressed and sends A common input device the corresponding code

2 Mouse Coordinates of the mouse movement Easy to operate are sensed and sent to control the pointing device; ideal

cursor; pressing a button enables for a graphical user

selection of an item or action interface (GUI)

3 Printer Produces output on paper in Several types are readable form available

4 Floppy disk drive Magnetic recording of data on A low cost secondary rotating floppy diskette by read/write storage device;

head; two surfaces are present each becoming obsolete

handled by a separate head due to popularity of CD

and flash memory

5 Hard disk drive Magnetic recording similar to floppy Better reliability and disk but flying heads record at higher higher speed than

density. Multiple disks of hard surface floppy disk

rotate at high speed

6 Scanner Converts pictures and text into a Useful for publishing stream of data and multi-media

applications

7 CRT Display (CRT Displays characters and graphics Internal operation has monitor) similarity with the picture tube in TV

8 Magnetic tape Magnetic recording of data along Common with older drive the length of the tape computers; nowadays used for data back-up

9 Plotter Creates drawings on paper; A costly output device graphics output device used for special

applications such as

CAD

(Contd.)

Basic Computer Organization 11 

S. No. Peripheral device Function/operation Remarks 10 Digital camera Captures digital images and stores A modern popular in its internal memory and transfers input device

to computer

11 Compact disc (CD) Provides large storage capacity Slower than hard disk (700 MB) using optical technology; but faster than floppy

laser beam is used for storing disk; ideal for multi

information media applications

12 Digital versatile disk New type of CD with storage Achieves theatre quality (DVD) capacity of 17 GB video and sound 13 Card reader Senses holes on punched (paper) Obsolete today cards

14 Card punch Punches holes on (paper) cards Obsolete today 15 Paper tape reader Senses holes on paper tape Obsolete today 16 Paper tape punch Punches holes on paper tape Obsolete today 17 Magnetic drum Data is recorded on rotating Slower than hard disk;

magnetic drum obsolete today

18 Web camera A low cost digital camera Helps posting attachable to Internet photographs directly

to web sites

19 Pen drive/RAM stick A special kind of electrically Very small size; portable erasable memory known as flash memory

memory

20 Modem Links a computer to a telephone Has become a line so as to communicate with a household device

remote computer / device thanks to the Internet

Input

devices

Obsolete

devices

Active

devices

Fig. 1.12(a) Input units

Card reader

Paper tape reader Keyboard

Mouse

Magnetic tape

Floppy disk

Hard disk

Scanner

Joystick

Compact disk

Digital versatile disk

12 Computer Architecture and Organization: Design Principles and Applications 

The ALU and control unit have usually some temporary storage units known as registers. Each register can be considered as a fast memory with single location. Such registers tem porarily store certain information such as instruction, data, address, etc. Storing in registers is advantageous since these can be read quickly compared to fetching them from external memory.

The ALU and control unit are together known as Central Processing Unit (CPU) or processor. The memory and CPU consist of electronic circuits and form the nucleus of the computer. The input and output units are electromechanical units consisting of both electronic circuits and mechanical assemblies. The input and output units are known as peripheral devices.

Card punch

Paper tape punch

Console typewriter

Output devices

Obsolete devices Active devices

CRT display

Printer

Plotter

Magnetic tape

Floppy disk

Hard disk

Compact disk(writable)

Digital

versatile disk (DVD)

Fig. 1.12(b) Output devices

1.4.1 Stored Program Concept

All modern computers use the stored program concept which was initially conceived by the design team of ISA computer led by Von Neumann. Hence, it is commonly known as Von Neumann concept. The essentials of stored program concept are as follows:

1. The computer has five different types of units: memory, ALU, control unit, input unit and output unit.

Basic Computer Organization 13 

2. The program and data are stored in a common memory.

3. Once a program is in memory, the computer can execute it automatically without manual intervention.

4. The control unit fetches and executes the instructions in sequence one by one. This sequential execution can be modified by certain type of instructions. 5. An instruction can modify the contents of any location in memory. Hence, a pro gram can modify itself; instruction–execution sequence also can be modified.

1.4.2 Main Memory and Auxiliary Memory

The memory from which the CPU fetches the instructions is known as main memory or primary memory. Hence, to run a program, it has to be brought into the main memory. The auxiliary memory is external to system nucleus and it can store large amount of programs and data. The CPU does not fetch instructions of a program in the auxiliary memory. Several programs are stored in an auxiliary memory and the program that should be executed is brought into the main memory (Fig. 1.13). The auxiliary memory is cheaper compared to main memory and hence a computer generally has limited amount of main memory and large amount of auxiliary memory. The auxiliary memory is also known as secondary storage. Figure 1.14 lists several types of auxiliary memory. These are connected to the computer as Input–Output (I/O) devices.

Main

Memory Input

Program

memory Auxiliary storage

devicesCPU Output devices

Fig. 1.13 Auxiliary storage

14 Computer Architecture and Organization: Design Principles and Applications Auxiliary memory

Obsolete type Active type

Magnetic drum

Paper tape

Punch card

Floppy disk

Hard disk

Optical disk

Magnetic tape

Pen drive

(Flash memory)

CD DVD

Fig. 1.14 Auxiliary memory types

1.4.3 Device Controller

A peripheral device is linked to the system nucleus (CPU and memory) by a device control ler, also known as I/O controller. Figure 1.15 shows different device controllers. The

Device controller

Keyboard interface

CRT

controller

Printer

controller

Floppy disk controller

Hard disk controller

Fig. 1.15 Common device controllers 

main function of a device controller is transfer of information (program and data) between the system nucleus and the device. Physically, a device controller can exist in three different forms: as a separate unit, as integrated with the device, as integrated with CPU. A basic device controller has five sections as shown in Fig. 1.16. It communicates with a device through the device interface that carries the signals between a device controller and a de vice. All device controllers communicate with CPU or memory through the system inter face (Fig.1.17). The system interface is identical to all device controllers.

Command

register

CPU and

System

Device

Device I/O device interface System

system memory

interface

interface logic

Data buffer

Status register Device controller

interface logic

Fig. 1.16 Device interface and system interface

Memory

CPU

System interface

Basic Computer Organization 15

FDC FDC - Floppy disk controller

HDC HDC - Hard disk controller

CRTC CRTC - CRT controller

Fig. 1.17 Common system interface 

The command register stores the command given by the software. The data buffer tempo rarily stores the data being transferred with the device. The status register stores the status of the device and controller. It is read by the software.

1.4.4 Device Interface Signals

There are three types of signals (Fig. 1.18) between a device and a device controller: data, control signals and status signals. The control signals are issued by the device controller

Data

Controller Control signals

Status signals

Fig. 1.18 Device interface singals

Device

demanding certain actions by the device. For example, the RESET signal asks the device to get reset, i.e., clear the internal condition inside the device. The status signals are sent by the I/O device reporting certain internal conditions (status) to the device controller. For exam ple, the ERROR signal reports that there is an error in the device. The data signals may be

16 Computer Architecture and Organization: Design Principles and Applications 

sent either serially (Fig. 1.19), on one wire, bit-by-bit or in parallel (Fig. 1.20), on eight wires, carrying all eight bits of a byte simultaneously.

Data wire

Controller Device

Fig. 1.19 Serial interface

Data bit 0

Controller Device

Data bit 7

Fig. 1.20 Parallel interface

1.4.5 I/O Drivers

An I/O driver for a device consists of routines for performing various operations: read, write etc. Each routine gives appropriate commands to the I/O controller and the device controller issues necessary control signals to the device. To support a device in a system, three items are required:

1. A device controller which logically interfaces the device to the system nucleus. 2. A device interface cable which physically connects the device to the device controller. 3. An I/O driver.

An I/O driver (device driver) is a collection of I/O routines for various operations for a specific I/O device. It takes care of issuing commands to a device controller, verifying the status of the device controller and device and handling input/output operations. The oper ating system and other programs use the device driver routines for doing I/O operations. The I/O driver is a system program which can be asked by any other program to ‘serve’. The I/O drivers of all the I/O devices are collectively known as BIOS.

1.5 Main Memory

The memory stores instructions, data and results of the current program being executed by the CPU. It is called program memory since the CPU fetches instructions only from this memory. The main memory is functionally organized as a number of locations. Each loca-

Basic Computer Organization 17

tion stores a fixed number of bits. The term word length of a memory indicates the number of bits in each location. The total capacity of a memory is the number of locations multi plied by the word length. Each location in memory is identified by a unique address as shown in Fig. 1.21. Two different memories with the same capacity may have different organization as illustrated by Fig. 1.22. Both have same capacity of 4 kilo bytes but they differ in their internal organizations.

Address

0000 First location

0001 0010

1111

Second location 

Third location

Last location (16 ) th

Fig. 1.21 Main memory locations

16 Bits

.

.

2 K locations 1 K locations .

.

32 Bits

.

.

.

.

(a) 2048 X16 (b) 1024 X 32 Fig. 1.22 Memory capacity and organization

18 Computer Architecture and Organization: Design Principles and Applications 

Example 1.1 A computer has a main memory with 1024 locations of each 32-bits. Calculate the total memory capacity:

Word length = 32 bits = 4 bytes;

No. of locations = 1024 = 1 kilo = 1K;

Memory capacity = 1K ´ 4 bytes = 4 KB

1.5.1 Access Time and Cycle Time

The time taken by the memory to read the contents of a location (or write information into a location) is called its access time. Generally, main memory should have uniform access time for all its locations irrespective of its address (first, middle, last, etc.). Modern computers use semiconductor memory whose access time is around 50 hs. Old computers use magnetic core memory whose access time was around 1 ms. After the access is over, the memory needs settling time after which the next access can be started. The settling time for semicon ductor and core memory are in the order of 10 and 200 hs, respectively. The cycle time is the minimum time interval from the beginning of one access to the beginning of next access. It is the sum of access time and settling time.

1.5.2 Random Access Memory (RAM)

A memory which has equal access time for all its locations is called a Random Access Memory (RAM). Both semiconductor memory and magnetic core memory are RAM. Though core memory has become obsolete, its non-volatile nature is interesting. The semi conductor memory is volatile: its contents are lost when power supply is removed.

Example 1.2 A main memory has an access time of 45 hs. A 5 hs time gap is necessary from the completion of one access to beginning of next access. Calculate the bandwidth of the memory.

Access time = 45 hs; settling time = 5 hs;

Cycle time = access time + settling time = 45 hs + 5 hs = 50 hs;

Bandwidth = 1/cycle time = 1/50 hs = 20 MHz.

1.5.3 Read Only Memory (ROM)

In a modern computer, usually a small part of the main memory is a Read Only Memory (ROM). The CPU can read from the ROM but cannot write into it. Generally, some permanently required control programs and BIOS are stored in ROM by the manufacturer.

Basic Computer Organization 19 

1.5.4 Memory Operations

The CPU addresses the memory both for a memory read operation and for a memory write operation. During a memory read operation, the CPU first sends the address of the location and then sends the memory read signal. On receiving the memory read signal, the memory starts reading from the location pointed by the address. After the access time, the content of the location is put by the memory on the data lines (Fig. 1.23). During a memory write operation, the CPU first sends the address of the location and then sends

Address

Memory read

CPU Memory

Data

Fig. 1.23 Memory read operation

the data to be written and the memory write signal (Fig. 1.24). On receiving the memory write signal, the memory starts writing operation in the location corresponding to the ad dress. Till the access time is over, the memory is busy with the write operation. The

Address

Memory write

CPU Memory

Data

Fig. 1.24 Memory write operation

20 Computer Architecture and Organization: Design Principles and Applications 

CPU uses two registers for communication with memory (Fig. 1.25). During read/write operations, the CPU puts the memory address in Memory Address Register (MAR). The Memory Buffer Register (MBR) is used to store the data from CPU during a write operation and the data from memory during a read operation. The MBR may be also called as Memory Data Register (MDR).

Address

MAR

Data

MBR

CPU Memory

Fig. 1.25 CPU registers for memory access

1.5.5 Asynchronous and Synchronous Memory Interface

Two types of CPU–memory interfaces are implemented in different computer systems: synchronous and asynchronous. In synchronous interface (Fig.1.26), the time taken by the memory for read/write operation is known to the CPU. Hence, there is no

Address

Memory read

CPU Memory

Memory write

Data

Fig. 1.26 Synchronous memory interface

Basic Computer Organization 21

Address

Memory read

Memory write

CPU Memory 

Data

MFC

Fig. 1.27 Asynchronous memory interface

feedback from memory to CPU to indicate the completion of read/write operation. In asyn chronous interface (Fig. 1.27), the memory informs about the completion of the read/write operation by sending a status signal, Memory Function Complete (MFC). This signal is also called as Memory Operation Complete.

1.5.6 Memory Addressability

The number of bits in the memory address determines the maximum number of memory addresses possible for the CPU. Suppose a CPU has n bits in the address, its memory can have a maximum of 2n (2 to the power of n) locations. This is known as the CPU’s memory addressability. It gives theoretical maximum memory capacity for a CPU. In practice, the physical memory size is less than this due to cost consideration.

Example 1.3 A CPU has a 12 bit address for memory addressing: (a) What is the memory addressability of the CPU ? (b) If the memory has a total capacity of 16 KB, what is the word length of the memory?

No. of address bits = 12;

Memory addressability = 212 = 4 kilo locations;

Memory capacity = 16 KB;

Word length = memory capacity/no. of locations = 16 KB/4K = 4B = 4 bytes. Some popular CPU’s and their memory addressability is given below:

CPU No. of address bits Memory addressability IBM System 360/40 24 16 mega Intel 8080 16 64 kilo

Intel 8088 20 1 mega

Pentium 32 4 giga

‘Unknown’ 40 1 tera

22 Computer Architecture and Organization: Design Principles and Applications 

1.6 CPU Operational Concepts

The function of the CPU is executing the program stored in the memory. For doing this, the CPU fetches one instruction at a time, executes it and then takes up next instruction. This action is done repeatedly and is known as instruction cycle. As shown in Fig. 1.28, the instruction cycle consists of two phases: fetch phase and execute phase. In fetch phase, an instruction is fetched from memory. In execute phase, the instruction is analysed and rel evant operations are performed.

Fetch instruction

Next

instruction

Execute instruction

Fig. 1.28 Instruction cycle phases

1.6.1 Instruction Format and Instruction Cycle

The general format of an instruction is shown in Fig. 1.29. The operation code (opcode) field identifies the operation specified and the operand field indicates the data. Generally

Operation code Operand

Fig. 1.29 Instruction format

the operand field gives the address of the memory location where the operand (data) is stored. The complete sequence involved for the fetch and execute phases of one instruction is known as instruction cycle. Consider an ADD instruction whose format is shown in Fig. 1.30. The bit pattern in the opcode field identifies it as an ADD instruction. The

ADD opcode I operand address II operand address

Fig. 1.30 ADD instruction format

Basic Computer Organization 23

other two fields identify the location where the two operands (data) are available. Figure 1.31 gives the steps involved in the instruction cycle and Table 1.3 defines each step. Figure 1.32 shows the formats of three more simple instructions: JUMP, NOOP and HALT.

Fetch

instruction

Decode

instruction

Next

instruction

Fetch

operands

Do

addition

Store

result

Fig. 1.31 Instruction cycle steps for ADD 

Opcode Instruction address

(a) JUMP instruction

Opcode

(b) HALT instruction

Opcode

(c) NOOP instruction

Fig. 1.32 Formats of common instructions

24 Computer Architecture and Organization: Design Principles and Applications 

TABLE 1.3 Instruction cycle steps and actions for ADD

S. no. Step Action responsibility Remarks 1 Instruction fetch Control unit; external Fetches next instruction from action main memory

2 Instruction decode Control unit; internal Analyses opcode pattern in action the instruction and identifies

the exact operation specified

3 Operand fetch Control unit: external Fetches the operands, one by (memory) or internal one, from main memory or

action depending on the CPU registers and supply them

location of operands to ALU

4 Execute (ADD) ALU; internal action Specified arithmetic or logical operation is done

5 Result store Control unit; external Stores the result in memory or or internal action registers

1.6.2 CPU States

The CPU has two major operating states: running and halt. In running state, the CPU per forms instruction cycles. In halt state, the CPU does not perform instruction cycle. When a computer is powered on, it goes to running state. The CPU goes from running state to halt state when any of the following actions take place:

1. Software Halt: A halt instruction is fetched and being executed by the CPU and hence the CPU halts.

2. Halt Input: The CPU receives halt input signal (Fig. 1.33) and halts. This signal may be generated either by the operator pressing the HALT (or PAUSE) switch in the front panel or by the circuits external to CPU.

Halt

CPU

Fig. 1.33 Halt input signal

3. Auto Shutdown: During instruction cycle, the CPU encounters a serious abnormal situation and hence, does shutdown halting the instruction cycle. (This shutdown is different from system shutdown.)

Basic Computer Organization 25 

1.6.3 CPU Registers

The CPU consists of following major registers (Fig. 1.34):

1. Accumulator (AC)

2. Program Counter (PC)

PC MBR MAR

IR

GPRs

A

B

C AC

IODR IOAR

Fig. 1.34 CPU registers

3. Memory Address Register (MAR)

4. Memory Buffer Register (MBR)

5. Instruction Register (IR)

6. General Purpose Registers (GPR)

7. I/O Data Register (IODR)

8. I/O Address Register (IOAR)

The adder performs arithmetic operations such as addition, subtraction etc. The accumu lator register holds the result of previous operation in ALU. It is also used as an input register to the adder. The program counter (instruction address counter) contains the ad dress of the memory location from where next instruction has to be fetched. As soon as one

26 Computer Architecture and Organization: Design Principles and Applications 

instruction fetch is complete, the contents of PC are incremented so as to point to the next instruction address. The instruction register stores the current instruction fetched from memory. The MAR contains the address of memory location during a memory read/write operation. The MBR contains the data read from memory (during read) or data to be writ ten into memory (during write). The GPRs are used for multiple purposes: storing oper ands, addresses, constants etc. In addition to GPRs, the CPU also has some working regis ters known as scratch pad memory. These are used to keep the intermediate results within an instruction cycle for complex instructions such as MULTIPLY, DIVIDE etc.

1.6.4 CLOCK

The clock signal is used as a timing reference by the control unit. The clock signal is a periodic waveform since the waveform regularly repeats itself. The rate at which the periodic waveform repeats is known as the frequency (f ). It is specified as cycles per second (cps) or Hz. The clock frequency is an indication of the internal operating speed of the processor. The fixed interval at which the periodic signal repeats is called as its time period (T ). The relationship between the frequency and the period is

f = 1/T

The pulse width (tpw) indicates the duration of the (clock) pulse. Another related term is duty cycle. The duty cycle is the ratio (expressed in percentage) of pulse width to period, i.e., duty cycle = (tpw /T ) × 100%. Fig. 1.35 illustrates the terms, period, frequency, and duty cycle.

tpw

T T

T

= Time period

tpw = Pulse width

Frequency )= 1/

(f T

Duty cycle = ( ) 100 pw ¥ t /T

Fig. 1.35 Periodic waveform

Basic Computer Organization 27 

Example 1.4 A clock signal has a frequency of 10 MHz with a duty cycle of 50%. Calculate its period and pulse width.

Data: f = 10 MHz; duty cycle = 50%;

Time period , (T ) = 1/f = 1/(10 × 106) = 100 hs;

Duty cycle = (tpw /T ) × 100 = 50%;

tpw = 0.5 T = 50 hs.

1.6.5 Macro-operation and Micro-operation

Every instruction specifies a macro-operation to be performed by the CPU. The opcode indicates the exact macro-operation. The instruction set lists all the possible macro operations for a computer. A machine language program is also called macro-program. To execute an instruction, the CPU has to perform several micro-operations. The micro operations are elementary operations inside the computer. The following are some examples of micro-operations: clearing a register, incrementing a counter, adding two inputs, transferring a register contents into another register, setting a flip–flop, complementing a register contents, shifting a register contents, reading from memory and writing into memory. Each micro-operation is done when the corresponding control signal is issued by the control unit. When a control signal is generated by the control unit, a control point in the hardware is activated resulting in the micro-operation. Table 1.4 gives some examples of micro-operations. Additional details of micro operations are discussed in Chapter 6.

TABLE 1.4 Sample microoperations

S. No. Control signal Micro-operation Remarks

1 MAR ¬ PC Contents of PC are copied The first micro-operation (transferred) to MAR in instruction fetch

2 PC ¬ PC + 1 Contents of PC are incremented The PC always points to next instruction address

3 IR ¬ MBR Contents of MBR are copied to IR The last micro-operation in instruction fetch

4 MBR ¬ AC Contents of accumulator are The first micro-operation copied to MBR in result store

28 Computer Architecture and Organization: Design Principles and Applications 

1.7 Interrupt Concept

An interrupt is an event inside a computer system requiring some urgent action by the CPU. In response to the interrupt, the CPU suspends the current program execution and branches in to an Interrupt Service Routine (ISR). The ISR is a program that services the interrupt by taking appropriate actions. After the execution of ISR, the CPU returns back to the interrupted program. On returning from ISR, the CPU resumes the old interrupted program as if nothing has taken place. This means, the CPU should continue from the place (instruction address) when interruption has occurred and the CPU status at that time should be the same. For this purpose, the condition of the CPU should be saved before taking up ISR. Before returning from ISR, this saved status should be loaded back into the CPU. Figure 1.36 illustrates the interrupt concept. There are several types of interrupts

1

Interrupt o ere ccurs h

2

3

4

5

6

7

.

.

.

Current program, A

Branch

Return

Interrupt service

routine (ISR)

l Interrupt occurs after commencement of the 4th instruction. The CPU recognizes the interrupt before fetching the 5th instruction. The CPU suspends the current program execution and branches to ISR. After completion of ISR, the CPU returns to the interrupted program, A. Now the 5th instruction is fetched.

Fig. 1.36 Interrupt concept

Basic Computer Organization 29 

TABLE 1.5 Common interrupt causes and actions

S. No. Interrupt type Cause for interrupt Action by ISR

1 I/O completion A device controller reports Control is given to I/O driver that it has completed an I/O

operation i.e., a command

execution is over

2 Data transfer A device controller reports Control is given to I/O driver that either it has a data byte which performs data transfer

ready (for input operation),

or it needs a data byte (for

output operation)

3 Overflow The result of an operation is Control is given to OS which greater than the capacity may cancel the program or

of accumulator or request user action

for various needs. Table 1.5 defines the causes for common interrupts. Chapter 10 covers interrupt handling in detail.

Nested interrupts: If the CPU is interrupted while executing an ISR, it is known as interrupt nesting. As a result, the CPU branches to new ISR. On completion of the new ISR, the CPU returns to the interrupted (old) ISR.

1.8 I/O Techniques

Input and output devices help us to give/take the program, data, and results to/from the computer. When an input operation is performed, we move information from an input device to memory (or CPU). Similarly, an output operation moves information from memory (or CPU) to an output device. An I/O routine (a program) takes care of input/ output operations. It interacts with the device controller to perform information transfer (input/output).

An I/O routine can follow three different methods for performing data transfer as shown in Fig. 1.37 (a). In software methods, the I/O routine transfers every piece (byte) of data, through the CPU, in two steps as shown in Fig. 1.37 (b). An IN or OUT instruction is

Data transfer

Through CPU (software method)

Bypassing CPU (hardware method)

Programmed mode DMA mode Interrupt mode Fig. 1.37(a) Methods of data transfer

30 Computer Architecture and Organization: Design Principles and Applications 

2 2

Memory

Input device Output device

1b 1a

1a 1b

CPU

1a, 1b Two steps in software method; 2 hardware method æ æ

Fig. 1.37(b) Principles of data transfer

used for transfer of data between CPU and a device. Data transfers from high speed devices such as hard disk and floppy disk is difficult to handle by programmed mode or interrupt mode since transfer rate of these modes is slower than the rate at which data is coming from these devices. Hence, DMA mode is essential for high speed devices. Slow speed devices can be handled by programmed mode or interrupt mode. Interrupt mode is ideal for very slow speed devices so that CPU time is not wasted in waiting for device readiness between succes sive bytes transfer. Different I/O drivers follow different techniques to suit the device control ler/devices. Chapter 10 covers the I/O techniques in detail.

1.8.1 I/O Ports

An I/O port is a program addressable unit which can either supply data to the CPU or which can accept data given by the CPU. An input port supplies data (when addressed) where as an output port accepts data when addressed. An IN instruction is used to accept data from an input port (Fig. 1.38). An OUT instruction is used to supply data to an output port (Fig. 1.39).

IN

Opcode

Port address

Fig. 1.38 In instruction

Each port has an unique address just like each memory location having an unique address. Physically, a port may exist in many forms: an independent unit, together with another port, or part of a special purpose component such as interrupt controller or device controller. Functionally, a port serves as a window for communication with CPU through data bus. For instance, the position (ON/OFF) of eight switches can be sensed through an eight bit input port. An IN instruction can transfer these status to a CPU register. Similarly, a set of eight LEDs can

Basic Computer Organization 31

be interfaced by an output port. An OUT instruction can supply the desired pattern by sending a CPU register contents to the output port. Chapter 10 covers I/O techniques in detail.

OUT

Opcode

Port address

Fig. 1.39 Out instruction 

1.9 Bus Concept

Connecting paths are necessary inside a computer for carrying following types of informa tion between the subsystems (CPU, memory, and I/O controllers):

1. Instruction from memory to CPU.

2. Data from memory to CPU.

3. Data from CPU to memory.

4. Memory Address from CPU to memory.

5. Port Address from CPU to I/O controllers.

6. Command from CPU to I/O controllers.

7. Status from I/O controllers to CPU.

In mainframe computers, there is a separate path from each source register to each destination register (Fig. 1.40) for connecting the source and destination. Every source is directly

Memory

PRC

CPU

HDC

Fig. 1.40 Non-bus communication

connected to every destination. This results in huge cost since, there are lot of wires and driver/receiver circuits. In mini computers and microcomputers, bus concept is used for inter connection of signals between the subsystems in a computer. The bus is a shared common path for several sources and destinations (Fig. 1.41). A single wire carries a signal to multiple units. However, at any instant only two units are logically linked: one source and one destination. The other units are logically disabled though physically connected.

32 Computer Architecture and Organization: Design Principles and Applications 

Suppose the CPU sends an eight bit data to floppy disk controller. The eight bits are sent on eight lines. Each line is common to all controllers and all of them receive the data pattern. Thus, each data bit is a bus and we have eight buses for data. In other words the width of the data bus is 8-bits. Though all controllers receive the data, only one controller is logically connected to the CPU. The other controllers do not use the data. The main advantage of bus is reducing cost of connecting paths (wires) and associated driver/receiver circuits. The disadvantage of bus is low speed since the bus is shared. At any time, only two units can communicate. Any other unit wanting to communicate has to wait.

Universal bus

CPU Memory PRC FDC HDC

Fig. 1.41 Bus concept

Fig. 1.42 shows a simple bus structure. There are three different buses: data bus, address bus and control bus. The data bus is used for multiple purposes:

Control bus Data bus Address bus

Output port/

IOW

MEMR

MEMW

MEMR

MEMW

IOR

IOW

IOR

MEMR memory read

æ

IOR I/O read

æ

device

Memory

CPU

Input port/

device

MEMW memory write

æ

IOW I/O write

æ

Not all control bus signals are shown.

Fig. 1.42 Simple bus structure

Basic Computer Organization 33

1. Communication between CPU and memory: instruction, operand and result. 2. Communication between CPU and I/O controllers: data, command and status.

The address bus is used by the CPU to send the address of the memory or I/O controller with whom it wants to communicate. The control bus has several control signals each

Memory Input port Output port

Read Write Read Write

Memory read

Memory write

Microprocessor

I/o read

I/o write

Ready

Fig. 1.43 Control bus signals

forming a bus. Some of them are: memory read, memory write, I/O read, I/O write, clock, reset and ready. Figure 1.43 shows the control bus signals and Table 1.6 defines them. 

TABLE 1.6 Control bus signals definition

S. No. Bus control signal Signal Direction Description

1 Memory read Output of CPU Indicates that the addressed memory should do memory read operation (and

put data on data bus)

2 Memory write Output of CPU Indicates that the addressed memory should (take data from data bus and) do

a memory write operation

3 I/O read Output of CPU Indicates that the selected input port should put data on data bus

4 I/O write Output of CPU Indicates that the selected output port should take data from data bus

5 Ready Input of CPU Indicates that the addressed subsystem (memory or I/O port) has completed the

read/write operation

34 Computer Architecture and Organization: Design Principles and Applications 

Microprocessors have two different bus structures (Fig. 1.44):

1. Internal bus which links the internal components of the microprocessor. 2. External bus which links the microprocessor with external units such as memory, I/O controllers, support chips etc.

Internal bus Address bus

Data bus

PC

IR

ALU AC

MAR MBR

Memory read I/O write

Ready

Interrupt

Control bus

Fig. 1.44 Internal and external bus in microprocessor

1.9.1 Bus Cycle

In a bus based computer, communication between the CPU and other subsystems take place on the bus. A clearly defined protocol is followed on the bus so that there is no interference from other subsystems when the CPU is communicating with any one subsystem. Normally, the CPU can initiate a communication sequence whenever it wants. A sequence of events, performed on the bus, for transfer of one byte (or word), through the data bus, is called a bus cycle. There are four major types of bus cycles:

1. Memory read bus cycle: CPU reads from a memory location;

2. Memory write bus cycle: CPU writes data into a memory location; 3. I/O Read bus cycle: CPU reads (accepts) data from an input port; 4. I/O Write bus cycle: CPU writes (sends) data to an output port.

Basic Computer Organization 35 

An instruction cycle may involve multiple bus cycles of different types. Fetching of an instruction involves memory read bus cycle. Storing a result in memory, needs memory write bus cycle. Fetching operands involve memory read bus cycles. An IN or OUT instruc tion execution involves I/O read bus cycle or I/O write bus cycle. Chapter 10 discusses the details of various bus cycles.

The CPU starts a bus cycle by sending the address (of memory or port) on the address bus. All the subsystems connected to the bus decode this address. Only the addressed subsystem will get logically connected to the bus and others do not interfere. The CPU indicates the type of bus cycle by sending the appropriate control signal (memory read, memory write, I/O read, I/O write) on the control bus. In a given bus cycle, the CPU sends one control signal which distinctly identifies the type of bus cycle. Memory responds to memory read or memory write whereas I/O ports respond to I/O read or I/O write. During output bus cycles, CPU puts data on the data bus and the addressed subsystem takes it. During input bus cycles, the addressed subsystem puts data on data bus and the CPU takes (reads) it.

1.10 System Software

A computer without system software is similar to a multi-storey building without lift. The main objective of any system software is simplifying system usage /operation and maximiz ing efficiency. A variety of system programs are available with different functions useful to various types of computer users: operators, programmers, maintenance engineers, etc.

1.10.1 Booting Sequence and OS

A computer system is always under the control of the operating system. Any other software gets control when the operating system assigns CPU to it. When a computer is installed, the operating system resides on hard disk. Immediately after powered on, the operating system should be loaded into memory and it should be given control of the CPU. The process of loading operating system (after powering on) into memory is called booting.

The following sequence of actions take place in the computer immediately after switching on:

1. The power-on action resets the CPU.

2. The CPU fetches its first instruction from a fixed location in main memory. This ad dress is known as the reset vector for the processor. Usually this address belongs to ROM. In most computers, a Power – on Self – Test (POST) program starts at this address. The POST checks and verifies proper functioning of different hardware units.

3. The ROM contains a short program called boot strap loader which reads the boot record (track 0, sector 1) from the disk and stores it in main memory.

36 Computer Architecture and Organization: Design Principles and Applications 

4. Then the boot strap loader gives control to the contents of boot record which is a pro gram called boot program. The boot program loads the operating system from the disk to main memory in read/write area and gives control to the operating system. 5. The operating system takes over and issues message to the user.

Physically, the POST and boot strap loader programs are present in a ROM that also contains BIOS. Hence, this ROM is usually called as BIOS ROM.

1.10.2 Types of Operating Systems

In a simple computer system, the OS is small in size. In a powerful computer system, the OS consists of several program modules: kernel (supervisor), scheduler, process manager and file manager. The different types of OS are listed in Table 1.7. In a real computer system, the OS is hybrid one: a combination of different types.

TABLE 1.7 Operating system types

No. Type Remarks

1 Batch OS Programs are input and run one by one 2 Inter-active OS During program execution input of data is supported

3 Time-sharing OS The system is shared by multiple users with terminals for interaction with OS

4 Multitasking/Multiprogramming OS More than one task or program is in memory. The CPU keeps switching between programs.

5 Real-time OS The OS periodically monitors the various inputs and accordingly different tasks are performed.

6 Multiprocessor OS Runs many programs simultaneously on many CPUs in a single computer system.

1.11 Computer Types

Based on performance, size, cost and capacity, the digital computers are classified into four different types (Fig. 1.45): mainframe (or maxi) computer, minicomputer, microcomputer

Computers

Supercomputer Mainframe Minicomputer Microcomputer Fig. 1.45 Types of computers

Basic Computer Organization 37 

and supercomputer. Due to constant change in technology and non-uniform standards among different computer manufacturers, any definition for these computer types is not easily accepted by everyone. Table 1.8 should be read with this caution in mind. A large system (mainframe) is physically distributed into more than one or two cabinets. It is an expensive computer that can be purchased only by large organizations. The minicomputers were developed with the objective of bringing out low cost computers so that smaller organizations could afford them. Some of the hardware features available in mainframes were not included in the minicomputer hardware in order to reduce the cost. Some features which were handled by hardware in mainframe computers were done by software in minicomputers. As a result, the performance of minicomputer is less than that of mainframe. The invention of microprocessor (single chip CPU) gave birth to the microcomputer that is several times cheaper than minicomputer but at the same time it offers everything available in minicomputer at a low speed. This has practically killed the minicomputer market.

TABLE 1.8 Classification of computers

Computer type Specific computers Typical feature Typical word length Mainframe B 5000, IBM S/360 Large size, high cost 32 bits and above Minicomputer PDP-8, HP 2100 A, Medium size, low cost 8/16 bits TDC-312

Microcomputer IBM PC, TRS 80 Very small size, Very 4/8/ 16/32 bits low cost

Supercomputer CRAY-1, Cyber Extremely high speed 64 bits and above 203/205

Any computer that is designed with a microprocessor is a microcomputer. A wide range of microcomputer is available today. The microcomputers are classified in two ways: 1. Based on functional usage or application

2. Based on shape or size

Figure 1.46 illustrates the different models in both the classifications. Tables 1.9 and 1.10 define these models briefly. The portable computers operate on battery and are designed to consume low power. Special engineering and hardware design techniques are adopted to make the portable computers strong and light weight. Hence, these are costlier though smaller in size.

38 Computer Architecture and Organization: Design Principles and Applications Micro computer

Shape/size Function/use

Desktop Palmtop

Laptop Notebook Pocket computer

Personal

computer

Pen

computer

Workstation

Embedded System

Server

Computer

Fig. 1.46 Microcomputer classification 

TABLE 1.9 Functional classification of microcomputers

S. No. Model Remarks

1 Personal Computer (PC) A low cost general purpose microcomputer; used for common applications by various professionals and

small and medium scale organizations; both initial

price and ongoing maintenance cost is low.

2 Workstation A special purpose microcomputer designed exclusively for a specific application such as CAD, DTP, Multimedia

etc; provides high performance but costlier than PC.

3 Embedded System A small dedicated/special computer (with control software) normally housed inside an equipment or

device such as washing machine, digital set-top box,

cell phone, automobile etc.

4 Server Several computers (clients) are supported by a central server computer with powerful hardware and software

resources common to all clients. Individual computers

have limited essential resources such as CPU and

memory. For the other required hardware/software,

the server will provide service on request basis.

TABLE 1.10 Physical classification of microcomputers

S. No. Model Remarks

1 Desktop Also known as tabletop; a common type

2 Laptop Can be kept on lap of the user; a portable computer 3 Notebook Appears like a notebook in shape; a portable computer 4 Palmtop Fits on the palm of the user

5 Pocket Like a pocket calculator

6 Pen Looks like a pen

Basic Computer Organization 39 

1.12 Computer Performance Factors

The performance of a computer is measured by the amount of time taken by the computer to execute a program. One may assume that the speed of the adder is a indicator of its performance, but it is not so. This is because the instruction cycle for ADD instruction involves not only add operation but also other operations such as instruction fetch, instruc tion decode, operand fetch, storing result, etc. The factors that contribute to the speed of these operations are as follows:

1. Instruction fetch: memory access time

2. Instruction decode: control unit speed

3. Operand address calculation: (1) GPRs access time/memory access time (2) address addition time

4. Operand fetch: memory access time/GPRs access time

Execute: addition time

5. Store result: main memory access time/GPRs access time

For two computers operating at the same clock rate, the execution time for ADD instruction can be different if they have different internal organization. Also for a given computer, the time taken for different instructions are not equal. For instance, MULTIPLY instruction will take more time than ADD instruction whereas LOAD instruction will take less time. Hence, the type of instructions and the number of instructions executed by the CPU, while running a program, decides the time taken by the computer for a program. The execution time for a program (TP) is related to the clock speed and the actual program by the following equation:

TP = NF

ie ´ CPI

where Nie is the number of instructions executed (encountered by the CPU; not total in structions in the program), CPI, the average number of clock cycles needed for an instruc tion and F, the clock frequency. It should be noticed that Nie is not equal to the total number of instructions in the program. Some of the instructions in the program may not be executed at all i.e. they may be skipped because of program logic. Similarly, some instructions may be executed several times because of loops.

From the above performance equation, it appears that to reduce program execution time Tp, the following approaches can be taken: reducing Nie, reducing CPI and increasing F. Reducing Nie involves having less instructions in the compiled program which is related to the compiler efficiency and the instruction set. Reducing CPI involves better CPU design to shorten instruction cycle time. Increasing F involves higher clock frequency which depends

40 Computer Architecture and Organization: Design Principles and Applications 

on technology. However, while designing a computer system, these points have to be considered together since improving one parameter may affect other parameters. Some high performance techniques are outlined in Section 1.14.

1.13 System Performance Measurement

Knowing the performance level of a computer system is a common requirement for compu ter users (buyers, programmers, etc). Performance comparison of two or more computers is a frequent need for making a choice between several computers.

For a given computer, there are two simple measurements that give us an idea about its performance:

1. Response time or execution time:

This is the time taken by the computer, to execute a given program, from the start to the end of the program. The response time for a program is different for different computers.

2. Throughput:

This is the work done (total number of programs executed) by the computer during a given period of time.

When we compare two computers, A and B, we say A is faster than B if the response time for a program is less on A than on B. Similarly, we say ‘A is n times faster than B ’ if

Execution time for B

Execution time for A = n

In other words, n = Performance of A

Percormance of B

1.13.1 MIPS

In the early days, the term MIPS (Millions of Instructions executed Per Second) was commonly used to indicate the speed of a computer. Since instruction cycle time is different for different instructions, the MIPS value will differ for different instruction mix in the program. Hence, the MIPS value of a computer gives only a rough idea about system performance.

Basic Computer Organization 41

1.13.2 Benchmark Programs

Benchmark programs are evaluation programs which are used to compare the performance of different computers. The time taken by a computer system to execute a benchmark pro gram serves as a measure of its performance. This is used for decision making by buyers, developers and marketing teams. There are two important aspects to be considered when we use benchmark programs:

1. Different instruction types have different execution times.

2. Number of occurrence of an instruction type varies with different types of programs.

The ability of a benchmark program to give reasonably accurate information about the performance of a computer depends on the instruction mix used in the benchmark program. Since application programs of different types (scientific, commercial, etc.) have different in struction mix, it is very difficult to develop a benchmark program which will truly stimulate all real life programs. Hence, separate benchmark programs are developed with different in struction mix for each type of application: scientific, commercial, educational, etc. While no benchmark can fully represent overall system performance, the results of a group of carefully selected benchmarks can give valuable information about real performance.

1.13.3 Benchmark Suites

A Benchmark suite is a collection of different types of benchmark programs. Running a benchmark suite on a computer gives a reasonably good idea about the performance level of a computer. Since there are a variety of application programs in a benchmark suite, weaknesses of one or more programs are compensated by the strengths of certain other programs in the benchmark suite. Thus, a benchmark suite helps in determining relative performances of two or more computers. However, the extent of accuracy of the result depends on the nature of the individual programs in the benchmark suite.

1.13.4 SPEC Rating

The SPEC (System Performance Evaluation Corporation) is a non-profit organization dedicated for performance evaluation. It selects typical application programs for different application areas and publishes (announces) the performance results for important commercial computers. The performance of one of the commercially available computer is taken as the reference computer. The performance of other computers are rated as a relative measure of the standard computer. The following are some of the standards released by SPEC:

42 Computer Architecture and Organization: Design Principles and Applications 

1. Desktop Benchmarks: SPEC89, SPEC2000, SPEC2006

2. Java Benchmarks: JSB2000, JVM98

3. Web server Benchmarks: SPECweb99, SPECweb96

4. Mail server Benchmark: MAIL2001

5. NFS Benchmark: SFS97

The SPEC rating for a computer X is specified as follows:

SPEC rating for X = Program execution time for standard computer Program execution time for X

In practice, a suite of programs of various types is run and the SPEC rating is calculated for each. The SPEC rating is a reflection of multiple factors: CPU performance, Memory performance, System organization, Compiler efficiency, Operating system performance etc.

SPEC does two different roles: developing benchmarks and publishing results. Developing suites of benchmarks: These suites have sets of benchmark programs with source codes and tools. The programs in the suites are developed by SPEC from programs donated by various sources (generally system manufacturers). SPEC aims on portability and creates tools and meaningful functions for the benchmarks programs.

Publishing news and benchmark results: Along with performance results, SPEC also provides additional information such as submitted results, benchmark descriptions, background in formation, and tools; these help in performance comparisons.

1.13.5 SPEC Groups

Presently SPEC has become an umbrella organization with three groups: OSG, HPG, and GPC.

Open Systems Group (OSG): OSG group develops benchmarks for desktop systems, workstations and multi-user servers supporting open operating system environments. High Performance Group (HPG): HPG group develops benchmarks for symmetric multiprocessor systems, workstation clusters, distributed memory parallel systems, and traditional vector and vector parallel supercomputers. These benchmarks represent large, real applications, in scientific and technical computing.

Graphics Performance Characterization Group (GPC): Responsible for industry standard graphics benchmarks for graphical and multimedia applications, subsystems, OpenGL, etc.

Basic Computer Organization 43 

Example 1.5 A benchmark program titled ‘equake’ (part of SPEC2000 suite) was run on two different computers: A with OPTERON processor and B with INTEL Itanium 2 processor. The computers A and B measured execution times of 72.6 seconds and 36.3 seconds respectively. The same program takes 1300 seconds on the Sun Ultra 5 – the reference computer of SPEC2000. Calculate the spec ratio of these two processors and comment on the relative performance.

SPEC ratio of OPTERON system = 1300

72.6 = 17.92

SPEC ratio of Itanium 2 system = 1300

36.3 = 35.78

Comment on relative performance: SPEC ratio of Itanium 2 is twice higher than SPEC ratio of OPTERON.

1.13.6 SPEC Suite and Mean

A SPEC suite has multiple application programs and each program gives a different SPEC ratio for a given computer, say X. For example, the SPEC 2000 suite has 14 benchmark programs. The mean of all the SPEC ratios of these programs gives a typical idea about the suite’s performance on computer X. Suppose s1, s2, ….. s14 are the SPEC ratios of different benchmark programs for X, then the geometric mean for the suite for X is calculated as

n

Õ

Geometric mean =

n s i

i

=

1

where si is the SPEC ratio for program i.

Example 1.6 SPEC 2000 was run on Intel Itanium 2 and the following SPEC ratio figures were calculated for the 14 benchmark programs of SPEC 2000 as 28.53, 43.85, 27.36, 41.25, 12.99, 72.47, 123.67, 35.78, 21.86, 16.63, 18.76, 16.09, 15.99, and 11.27 respectively. Calculate the geometric mean.

Applying the formula, Geometric mean = 27.12

1.14 High Performance Techniques

Several techniques have been developed over the years to increase the system performance. A brief discussion of these is given here and exhaustive coverage of them is presented in later chapters.

44 Computer Architecture and Organization: Design Principles and Applications 1.14.1 High Performance Computer Architecture

The designer can follow any of the following two approaches for enhancing the perform ance of a processor:

1. Increasing the clock frequency.

2. Increasing the number of operations performed during each clock cycle. Increasing the clock frequency depends upon IC fabrication process. For performing more operations per clock cycle, any of the two strategies are followed: 1. Providing multiple functional units in a single processor chip.

2. Executing multiple instructions concurrently by fully loading the functional units. As discussed earlier in Section 1.12, the performance of a computer (i.e. the time taken to execute a program) depends on three factors:

1. Number of instructions to be executed

2. Average number of clock cycles required per instruction

3. Clock cycle time

As mentioned earlier, reducing the clock cycle time depends on the IC manufacturing technology. Research and newer inventions in this area continue. A convenient technique for obtaining high performance is parallelism. It is achieved by replicating basic units in the computer system. Parallelism has been successfully used since the early days in a variety of ways in several computer systems. Some common techniques followed by computer archi tects for achieving parallelism are as follows:

1. Using multiple ALUs in a single CPU.

2. Connecting multiple memory units to a single CPU to increase the bandwidth. 3. Connecting multiple processors to one memory for enhancing the number of instructions to be executed in a unit time.

4. Connecting multiple computers (as a cluster) that can share the program load.

1.14.1.1 Reducing Memory Access Time

If the memory access time is large, the instruction cycle takes more time. To resolve this, there are many techniques:

Cache Memory: Having a small and fast memory (between CPU and main memory) known as cache memory (Fig. 1.47) in which contents of certain main memory locations can be temporarily stored in advance. Whenever any of these locations are accessed, CPU need not access main memory since these will be supplied by cache memory very fast. Hence, CPU fetches an instruction or operand in shorter time than main memory access time. It is obvious that at any time the cache memory can keep only some location contents depend-

Basic Computer Organization 45 

ing on its capacity. Generally cache memory is costlier as compared to main memory and hence its size should not be too big. Also the result of an instruction can be stored in cache memory instead of storing in main memory so that instruction cycle time for CPU is short ened.

Main memory Large and slow

Cache fill (multiple words)

Cache memory Small and fast

Cache read (one word)

CPU

Fig. 1.47 Cache memory

Instruction Pre-fetch: Fetching an instruction in advance when previous instruction execu tion is not yet completed is known as instruction pre-fetch (Fig. 1.48). In this technique, two consecutive instruction cycles are overlapped. As a result, the instruction fetch time in the instruction cycle is reduced to zero.

Memory

Pre-fetch

Instruction

queue

IRCPU

Fig. 1.48 Instruction prefetch

46 Computer Architecture and Organization: Design Principles and Applications

Memory Interleaving: Organizing the main memory into two independent banks with one containing odd addresses and the other containing even addresses is known as memory interleaving (Fig. 1.49). When CPU is accessing more than one consecutive location, both odd and even banks can perform simultaneous read/write operation. This effectively cuts the access time by half.

ADDRESS ADDRESS

Odd bank

0000 0010

1110

Even bank Address

0001

0011

1111

MR MW

Data/instruction

CPU

Fig. 1.49 Memory interleaving

1.14.1.2 Reducing Instruction Decode time

MR æ Memory read MW Memory write æ

The time taken for decoding depends on hardware circuit technology. It can be overlapped with the execution time of previous instruction if it has been already fetched. This technique is known as pre-decode or instruction look-ahead (Fig. 1.50). 

PF PD

Prefetch section Predecode section

Instruction Decoded output

Fig. 1.50 Predecode

Basic Computer Organization 47 

1.14.1.3 Instruction Pipelining

Even a fast adder needs some time to produce result. However, the CPU can be divided into different sections so that each stage of an instruction cycle can be done by an independ ent section. All sections can work simultaneously for different instructions. This technique is known as instruction pipelining (Fig. 1.51). The net effect of this full overlap of instruction cycle is that average instruction cycle time is equal to the time taken by the longest stage. Chapter 12 covers elaborately the technique of instruction pipelining.

PD OA OF EX SR PF

PF Prefetch æ OA Operand address calculation æ EX Execute æ

PD Predecode æ OF Operand fetch æ SR Store result æ

Fig. 1.51 Instruction pipelining

1.14.1.4 Superscalar Architecture

If more than one adder is available in a CPU, all adders can work simultaneously for con secutive instructions. This technique is known as superscalar architecture (Fig. 1.52). Gener ally two or more pipelines are present in a single CPU. Chapter 12 discusses the superscalar architecture.

1.14.2 Instruction set: CISC and RISC

There are two popular architectural standards for CPU design:

Complex Instruction Set Computing (CISC)

Reduced Instruction Set Computing (RISC)

The CISC is an old concept where as the RISC is a modern concept. A CISC CPU has a large number of instructions of which many are powerful. Its control unit is highly complex. A RISC CPU has a small number of instructions all of which are simple. Its control unit is very simple. All older systems are CISC systems but today’s systems comprise both types. RISC systems are popular today due to their higher performance but they are costlier. Hence, they are used only for special applications where a special need for high perform ance or reliability arises. Chapter 3 discusses the CISC and RISC concepts briefly and Chapter 13 reviews the RISC systems in detail.

48 Computer Architecture and Organization: Design Principles and Applications 

1.14.3 Compiler Efficiency

A good compiler generates a short machine language program (object code) for a given high level language program (source code) with minimum number of instructions. To what extent the instruction count can be small depends also on the CPU architecture. However the CPI also should be small so that the program execution time is small. This again de pends on the CPU architecture. An optimizing compiler for certain advanced processors is a special compiler that uses several techniques to minimize the program execution time. Chapters 13 and 14 discuss about the special compilers.

The overall performance of a computer system can be improved by certain advanced system designs discussed in the following chapters. Two popular common design techniques are multiprogramming and multiprocessing which are outlined in Table 1.11.

Some of the high performance techniques are visible to operating system whereas others are invisible. The invisible design techniques are not part of system architecture but part of system organization.

PF

(prefetch)

D1

(first decode)

D2

(second decode)

2 instructions together 2 instructions together

D2

(second decode)

U pipe V pipe

E

(execute)

WB

(write back) result

E

(execute)

WB

(write back) result

Fig. 1.52 Superscalar architecture (Pentium)