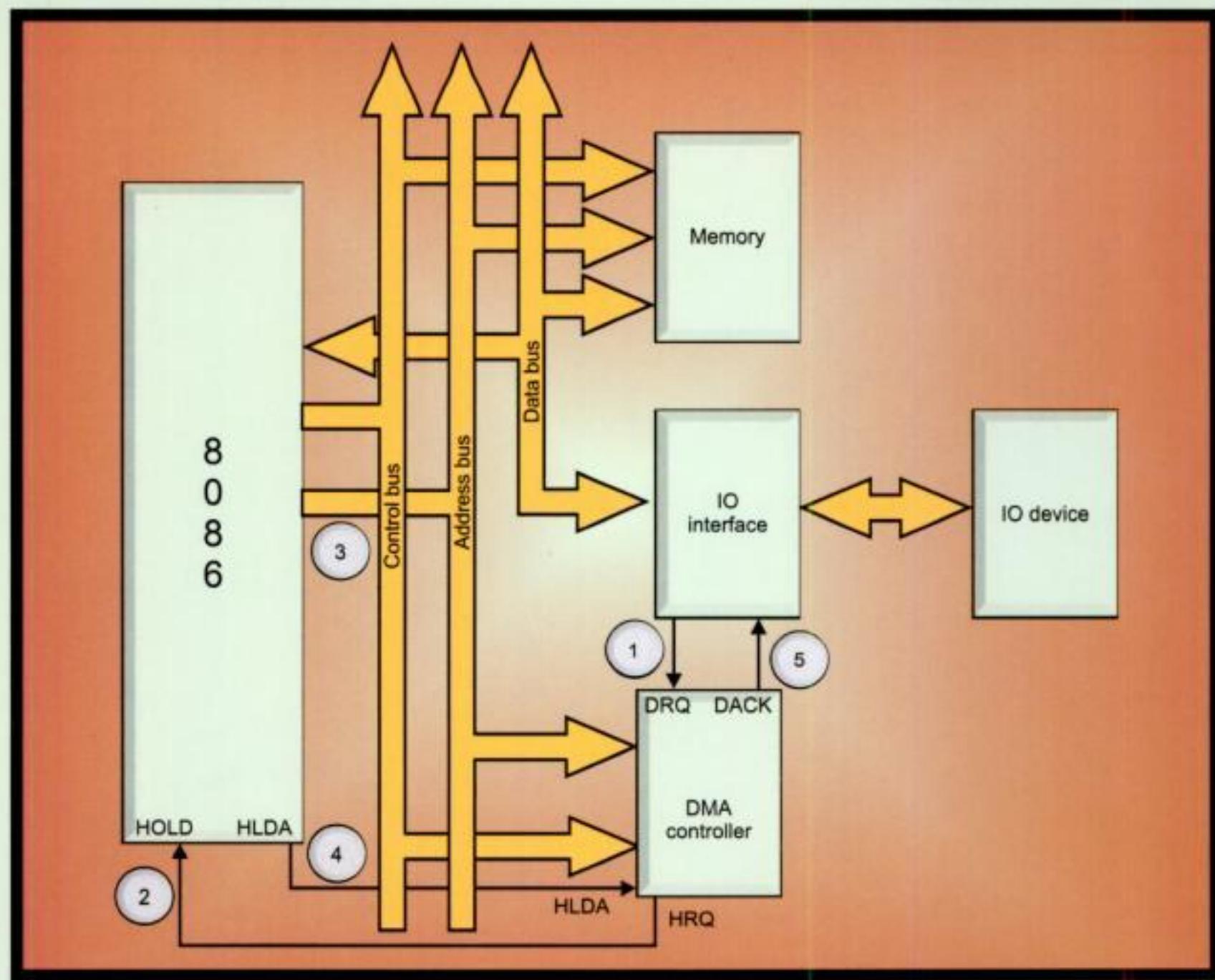


# Microprocessor 8086

## Architecture, Programming and Interfacing



**Sunil Mathur**

**₹ 450.00**

**MICROPROCESSOR 8086: Architecture, Programming and Interfacing**  
Sunil Mathur

© 2011 by PHI Learning Private Limited, New Delhi. All rights reserved. No part of this book may be reproduced in any form, by mimeograph or any other means, without permission in writing from the publisher.

**ISBN-978-81-203-4087-9**

The export rights of this book are vested solely with the publisher.

Published by Asoke K. Ghosh, PHI Learning Private Limited, M-97, Connaught Circus, New Delhi-110001 and Printed by Rajkamal Electric Press, Plot No. 2, Phase IV, HSIDC, Kundli-131028, Sonepat, Haryana.

# Contents

<i>Preface</i>	xix
<i>Acknowledgements</i>	xxiii

## PART I

<b>1. Architecture and Organization of Microprocessors and Microcomputers</b>	<b>3–26</b>
1.1 Introduction	3
1.2 Computer Languages	3
1.2.1 Low-level Language	3
1.2.2 Machine Language	4
1.2.3 Assembly Language	4
1.2.4 High-level Language	4
1.3 Microprocessor	4
1.3.1 Arithmetic and Logic Unit	5
1.3.2 Register Unit	5
1.3.3 Control Unit	5
1.4 Microprocessor Operations	5
1.4.1 Microprocessor Initiated Operations	6
1.4.2 Internal Data Operations	7
1.4.3 Peripheral or Externally Initiated Operations	8
1.5 8085 Functional Description	9
1.6 Internal Architecture of 8085	9
1.6.1 Register Unit	10
1.6.2 Control Unit	12
1.6.3 Arithmetic Logic Unit	12
1.6.4 System Bus of 8085	14

---

<b>1.7</b>	<b>8085 Pin Description</b>	<b>14</b>
1.7.1	Group 1: Power Supply and Frequency Signals	15
1.7.2	Group 2: Higher Order Address Bus A <sub>8</sub> –A <sub>15</sub>	16
1.7.3	Group 3: Multiplexed Address/Data Bus AD <sub>0</sub> –AD <sub>7</sub>	16
1.7.4	Group 4: Control and Status Signals	16
1.7.5	Group 5: Serial IO Signal	17
1.7.6	Group 6: Externally or Peripheral Initiated Signals	17
<b>1.8</b>	<b>Microprocessor System</b>	<b>19</b>
<b>1.9</b>	<b>How a Program is Executed</b>	<b>19</b>
<i>Exercises</i>	<i>20</i>	
	<i>Multiple Choice Questions</i>	<i>20</i>
	<i>Descriptive Questions</i>	<i>24</i>

**2. Introduction to 8086****27–49**

<b>2.1</b>	<b>Introduction</b>	<b>27</b>
<b>2.2</b>	<b>The 8086 Microprocessor</b>	<b>27</b>
2.2.1	Bus Interface Unit (BIU)	29
2.2.2	Execution Unit	31
<b>2.3</b>	<b>Pin Configuration of 8086</b>	<b>34</b>
2.3.1	Pin Details of 8086—Common to Both Minimum and Maximum Mode	35
2.3.2	Pin Details of 8086—(Minimum Mode)	36
2.3.3	Pin Details of 8086 S—(Maximum Mode)	37
<b>2.4</b>	<b>Memory Organization of 8086</b>	<b>39</b>
<b>2.5</b>	<b>Microprocessor 8088</b>	<b>41</b>
<i>Exercises</i>	<i>43</i>	
	<i>Multiple Choice Questions</i>	<i>43</i>
	<i>Descriptive Questions</i>	<i>48</i>

**3. 8086 Based System****50–73**

<b>3.1</b>	<b>Introduction</b>	<b>50</b>
<b>3.2</b>	<b>8086 Minimum Mode Configuration</b>	<b>50</b>
3.2.1	Demultiplexing of the Multiplexed Buses	51
3.2.2	Transceiver 8286	53
3.2.3	Generation of Control Signals	54
3.2.4	Clock Generator 8284 and Driver	55
3.2.5	Interfacing of Memory in Minimum Mode	58
<b>3.3</b>	<b>Maximum Mode Configuration of 8086</b>	<b>59</b>
3.3.1	Bus Controller 8288	60
3.3.2	Memory Interface of a Maximum-Mode 8086 System	64
<b>3.4</b>	<b>Bus Cycles of 8086</b>	<b>65</b>
3.4.1	Minimum Mode Bus Cycles	67
3.4.2	Maximum Mode Bus Cycles of 8086 System	67
3.4.3	Bus Request and Bus Grant Timings in Minimum and Maximum Mode Systems	69

<i>Exercises</i>	70	
<i>Multiple Choice Questions</i>	70	
<i>Descriptive Questions</i>	72	
<b>4. Instructions Set of 8086</b>		<b>74–136</b>
4.1 Introduction	74	
4.2 Addressing Modes of 8086	74	
4.2.1 Data Addressing Modes	75	
4.2.2 Address Addressing Modes	81	
4.3 Instruction Format	84	
4.4 Instruction Templates	86	
4.5 Instruction Set of 8086	92	
4.5.1 Data Transfer Instructions	92	
4.5.2 Arithmetic Instructions	98	
4.5.3 Logical Instructions	103	
4.5.4 Shift and Rotate Instructions	106	
4.5.5 String Instructions	110	
4.5.6 Adjustment Instructions	113	
4.5.7 Flag Related Instructions	114	
4.5.8 Control Transfer Instructions	115	
4.5.9 Processor-control Instructions	121	
<i>Exercises</i>	128	
<i>Multiple Choice Questions</i>	128	
<i>Descriptive Questions</i>	134	
<b>5. Assembler Directives</b>		<b>137–163</b>
5.1 Introduction	137	
5.2 Assembly Language	137	
5.3 Assembly Language Program Development Tools	138	
5.3.1 Editor	139	
5.3.2 Assembler	139	
5.3.3 Linker	140	
5.3.4 Loader	141	
5.3.5 Debugger	141	
5.4 TASM Assembler	143	
5.5 MASM Assembler	144	
5.6 Assembler Directives	146	
5.6.1 Data Defining Assembler Directives	146	
5.6.2 Segment Defining Directives	152	
5.6.3 Combining Segments	155	
5.6.4 Managing Large Programs	156	
5.6.5 Processor Directives	157	
5.6.6 Initialization of Program Memory Models	158	
<i>Exercises</i>	159	
<i>Multiple Choice Questions</i>	159	
<i>Descriptive Questions</i>	162	

<b>6. Programming of 8086</b>	<b>164–191</b>
6.1 Introduction	164
6.2 Flowchart	164
6.2.1 Guidelines for Drawing a Flowchart	164
6.2.2 Advantages of Using Flowcharts	166
6.2.3 Limitations of Using Flowcharts	166
6.3 Programming Steps	167
6.4 Solved Examples	167
<i>Exercises</i>	187
<i>Multiple Choice Questions</i>	187
<i>Descriptive Questions</i>	190
<b>7. Interrupts of 8086</b>	<b>192–236</b>
7.1 Introduction	192
7.2 Advantages of Interrupts	192
7.3 Interrupt Systems	193
7.3.1 Single Line or Single Level Interrupt System	193
7.3.2 Multilevel or Multiline Interrupt System	193
7.4 Classification of Interrupts	194
7.5 Interrupts of 8086	195
7.6 Interrupt Pointer Table	196
7.6.1 Dedicated Interrupts of 8086	198
7.6.2 Software Interrupts of 8086	202
7.6.3 Priority of Interrupts	202
7.7 Programmable Interrupt Controller (PIC) 8259A	203
7.7.1 Block Diagram and Functional Description of 8259A	203
7.7.2 Pin Diagram and Pin Description	205
7.8 Interrupt Sequence	206
7.9 Programming of 8259A	207
7.9.1 Initialization	207
7.9.2 Initialization Control Words (ICWs)	207
7.9.3 Operation Command Words (OCWs)	213
7.9.4 Operating Modes of OCW <sub>3</sub>	216
<b>7.10 Operatiang Modes of 8259A</b>	<b>217</b>
7.10.1 Automatic End of Interrupt (AEOI)	217
7.10.2 Automatic Rotation	218
7.10.3 Buffered Mode	218
7.10.4 Cascade Mode	219
7.10.5 End of Interrupt (EOI)	219
7.10.6 Fully Nested Mode	219
7.10.7 Special Fully Nested Mode	221
7.10.8 Special Mask Mode	221
7.10.9 Specific Rotation Mode	221
7.11 Expansion Past 64 Interrupts	221
<b>7.12 Interfacing of 8259A within IO Mapped IO Method</b>	<b>222</b>

<b>7.13 Programming 8259A</b>	<b>223</b>
7.14 Software Design Problems	224
<i>Exercises</i>	<i>230</i>
<i>Multiple Choice Questions</i>	<i>230</i>
<i>Descriptive Questions</i>	<i>234</i>

## PART II

<b>8. Math Coprocessor 8087</b>	<b>239–274</b>
8.1 Introduction	239
8.2 Introduction to the 8087	239
8.3 Escape Instructions	240
8.3.1 Host Response of Host Processor to Escape Instructions	241
8.3.2 Coprocessor Response to Escape Instructions	241
8.4 Queue Status	242
8.5 Data Types of 8087	243
8.5.1 Real or Floating Point Data Formats	243
8.5.2 Integer Data Format	246
8.5.3 Packed BCD Format	246
8.6 Pin Configuration of 8087	247
8.7 Block Diagram of 8087	248
8.7.1 Numeric Execution Unit	248
8.7.2 Registers of 8087	249
8.8 Interfacing of 8087 with 8086	255
8.9 Instruction Set of 8087	256
8.9.1 Data Transfer Instructions	257
8.9.2 Arithmetic Instructions	258
8.9.3 Compare Instructions	261
8.9.4 Transcendental Instructions (Trigonometric and Exponential Instructions)	262
8.9.5 Load Constant Instructions	263
8.9.6 Processor Control Instructions	263
8.10 Programming Examples of 8087	264
<i>Exercises</i>	<i>272</i>
<i>Multiple Choice Questions</i>	<i>272</i>
<i>Descriptive Questions</i>	<i>274</i>
<b>9. Multiprocessing and Multiprogramming</b>	<b>275–290</b>
9.1 Introduction	275
9.1.1 Multiprocessing and Multiprogramming	275
9.1.2 Benefits of Multiprocessing	276
9.2 Lock Facility	277
9.3 8086-based Multiprocessing Systems	279
9.4 Coprocessor Configuration	279
9.4.1 Escape (ESC) Instruction	281
9.4.2 Coprocessor Configuration Operation	282

9.5	Closely Coupled Configuration	282
9.6	Loosely Coupled Configuration	284
9.7	Bus Allocation Schemes	287
9.7.1	Daisy Chaining Scheme of Bus Allocation	287
9.7.2	Polling Scheme of Bus Allocation	288
9.7.3	Independent Request Scheme of Bus Allocation	288
	<i>Exercises</i>	289
	<i>Multiple Choice Questions</i>	289
	<i>Descriptive Questions</i>	290

## **10. Serial and Parallel Data Transfer** 291–339

10.1	Introduction	291
10.2	Parallel Transmission	291
10.3	Microprocessor Controlled Data Transfer	292
10.3.1	Programmed IO	292
10.3.2	Interrupt Driven IO	294
10.4	Direct Memory Access (DMA)	295
10.4.1	Burst or Block Transfer DMA	296
10.4.2	Cycle Steal or Single Byte Transfer DMA	297
10.5	Serial Transmission	297
10.6	Types of Communication Systems	299
10.6.1	Simplex	299
10.6.2	Duplex	299
10.7	Serial Transmission Format	300
10.7.1	Asynchronous Serial Data Transfer	300
10.7.2	Synchronous Serial Data Transfer	301
10.8	Baud Rate	303
10.9	Data Communication Over Long Distances	303
10.10	Software Based Serial I/O	303
10.10.1	Serial I/O Transfer Using Parallel Port	303
10.11	Universal Synchronous Asynchronous Receiver Transmitter (USART) 8251	306
10.11.1	Features of 8251 USART	306
10.11.2	Pin Description of 8251	306
10.12	Block Diagram Description of 8251	309
10.12.1	Data Bus Buffer	309
10.12.2	Read/Write Control Logic	310
10.12.3	Transmitter Section	311
10.12.4	Receiver Section	312
10.12.5	Modem Control	313
10.13	8251 Control Word	314
10.13.1	Mode Word Format	314
10.13.2	Command Word Format	316
10.13.3	Status Word	317
10.14	Data Transfer Operations of 8251	318
10.14.1	Asynchronous Mode—Transmission	319
10.14.2	Asynchronous Mode—Reception	320

10.14.3 Synchronous Mode—Transmission	321
10.14.4 Synchronous Mode—Reception	322
10.15 Initialization of 8251	323
10.16 8251 Interfacing in I/O Mapped I/O	325
<i>Exercises</i>	337
<i>Multiple Choice Questions</i>	337
<i>Descriptive Questions</i>	338
<b>11. IO and Memory Interfacing</b>	<b>340–380</b>
11.1 Introduction	340
11.2 IO Devices and Their Interfacing	340
11.3 IO Addressing	341
11.3.1 IO Mapped IO	341
11.3.2 Memory Mapped IO	342
11.4 Interfacing of Input Device	343
11.5 Interfacing of Output Device	345
11.6 Semiconductor Memory and Its Interfacing	350
11.7 ROM: Read Only Memory	353
11.7.1 Mask Programmed (ROM) Memory Circuits	353
11.7.2 Programmable Read-Only Memory (PROM)	355
11.7.3 Erasable Programmable Read-Only Memory (EPROM)	355
11.7.4 Electrically Erasable Programmable Read-Only Memory (EEPROM)	355
11.8 Random Access Memory (RAM)	356
11.8.1 Static Read/Write Memory (SRAM)	356
11.8.2 Dynamic Read-Write Memory (DRAM)	361
11.9 Basic Concepts in Memory Interfacing	362
11.9.1 Address Decoding	363
11.9.2 Interfacing Circuit	364
11.9.3 Address Decoding and Memory Addresses	365
11.9.4 Address Decoding Techniques	366
11.10 Memory Organization of Microprocessor 8086	368
11.11 Interfacing of ROM with 8086	371
<i>Exercises</i>	375
<i>Multiple Choice Questions</i>	375
<i>Descriptive Questions</i>	378
<b>PART III</b>	
<b>12. Programmable Peripheral Interfacing Chips</b>	<b>383–434</b>
12.1 Introduction	383
12.2 Programmable Peripheral Interfacing Chip 8255	383
12.2.1 Block Diagram of 8255	384
12.2.2 Pin Description of 8255	386
12.3 Operational Description of 8255	387
12.3.1 Bit Set Reset (BSR) Mode	388

12.3.2	IO Mode	389
12.3.3	Control Word Register of 8255 in IO Mode	390
12.4	Mode 0 (Simple Input/Output)	391
12.5	Mode 1 (Strobed Input/Output)	393
12.5.1	8255 in Input Mode of Mode 1	393
12.5.2	Input Control Signal Definition	393
12.5.3	Timing Diagram of Mode 1 (Input)	395
12.5.4	Data Transfer Modes	395
12.5.5	Mode 1 Output Mode	396
12.5.6	Control Signal Definition Output Mode	397
12.5.7	Data Transfer in Output Mode in Mode 1	397
12.5.8	Timing Diagram of Mode 1 (Output Mode)	398
12.5.9	Combination of Mode 1	398
12.6	Mode 2 (Strobed Bidirectional Bus I/O)	399
12.6.1	Mode 2 Different Combinations	400
12.7	8255 Interfacing	401
12.7.1	IO Mapped IO	402
12.7.2	Memory Mapped IO	402
12.8	Interfacing and Design Problems	404
12.9	Interfacing of Stepper Motor	415
12.9.1	Interfacing of DAC 0800	419
12.9.2	Interfacing of ADC0800 8-Bit A/D Converter	424
<i>Exercises</i>		431
	<i>Multiple Choice Questions</i>	431
	<i>Descriptive Questions</i>	432

### **13. 8253/54 Programmable Timer** **435–464**

13.1	Introduction	435
13.2	Functional Block Diagram of 8253/54	435
13.3	Pin Configuration of 8253/54	437
13.3.1	Pin Description of 8253/54	437
13.4	Programming the 8253/54	439
13.4.1	Control Word Format	439
13.5	Write Operations	439
13.6	Read Operations	441
13.6.1	Simple Read/Write Operations for the Desired Counter (Common for 8253 and 8254)	441
13.6.2	Counter Latch Command (Common for 8253 and 8254)	441
13.6.3	Read Back Command (only for 8254)	442
13.7	Modes of Operations	444
13.7.1	Mode 0: Interrupt on Terminal Count	444
13.7.2	Mode 1: Hardware Retriggerable One-Shot	446
13.7.3	Mode 2: Rate Generator	447
13.7.4	Mode 3: Square Wave Mode	447
13.7.5	Mode 4: Software Triggered Mode	448
13.7.6	Mode 5: Hardware Triggered Strobe (Retriggerable)	451

---

13.7.7	Operation Common to All Modes	451
13.7.8	Gate Pin Operation Summary	451
13.8	System Interface of 8253/54	451
13.9	Interfacing of 8253/54 with 8086	453
13.9.1	Interfacing of 8253/54 in IO Mapped IO Method	454
13.9.2	Interfacing of 8253/54 in Memory Mapped IO Method	455
13.10	Programming of 8253/54	457
	<i>Exercises</i>	462
	<i>Multiple Choice Questions</i>	462
	<i>Descriptive Questions</i>	463
<b>14.</b>	<b>DMA Controller 8257 and 8237</b>	<b>465–501</b>
14.1	Introduction	465
14.2	The DMA Controller	465
14.3	Functional Behaviour of a DMA Data Transfer	465
14.3.1	Burst or Block Transfer DMA	467
14.3.2	Cycle Steal or Single Byte Transfer DMA	467
14.3.3	Data Transfer DMA Operation	469
14.4	The Programmable DMA Controller 8257	470
14.5	The Pin Diagram of 8257	471
14.6	Block Diagram of 8257	474
14.6.1	DMA Channels	474
14.6.2	Data Bus Buffer	476
14.6.3	Read/Write Logic	476
14.6.4	Control Logic	477
14.6.5	Priority Resolver	477
14.6.6	Mode Set Register	477
14.6.7	Status Register	479
14.7	Programming and Reading the 8257 Registers	480
14.8	DMA Operation	481
14.8.1	Single Byte Transfers	481
14.8.2	Consecutive Transfers	481
14.8.3	Control Override	482
14.8.4	Not Ready	482
14.8.5	Speed	482
14.9	State Diagram of 8257	482
14.10	Operation of DMA Cycle	483
14.11	The 8237 DMA Controller	484
14.12	Functional Description	485
14.12.1	Registers of 8237	486
14.12.2	Current Address Register	487
14.12.3	Current Word Register	487
14.12.4	Base Address and Base Word Count Registers	487
14.12.5	Command Register	487
14.12.6	Mode Register	487
14.12.7	Request Register	487

14.12.8	Mask Register	489
14.12.9	Status Register	489
14.12.10	Temporary Register	489
14.12.11	Software Commands	490
14.13	DMA Cycles	491
14.13.1	Idle Cycle	492
14.13.2	Active Cycle	492
14.14	DMA Operating Modes	492
14.14.1	Single Transfer Mode	493
14.14.2	Block Transfer Mode	493
14.14.3	Demand Transfer Mode	493
14.14.4	Cascade Mode	493
14.15	Transfer Types	494
14.15.1	Peripheral Transfer	494
14.15.2	Memory-to-Memory	495
14.16	Operating Modes of 8237	495
14.16.1	Auto-initialize Mode	495
14.16.2	Priority Mode	496
14.16.3	Normal Mode	496
14.16.4	Extended Write Mode	496
14.16.5	Compressed Timing	496
14.17	Interfacing of DMA Controller	496
14.18	Solved Examples	497
	<i>Exercises</i>	499
	<i>Multiple Choice Questions</i>	499
	<i>Descriptive Questions</i>	500

## 15. Keyboard and Display Interfacing 502–541

15.1	Introduction	502
15.2	Keyboard	502
15.2.1	Key Debouncing	503
15.2.2	Hardware Key Debouncing	503
15.2.3	Software Key Debouncing	504
15.2.4	Roll-Over	504
15.2.5	2-Key Roll-Over	504
15.2.6	N-key Roll-Over	504
15.2.7	N-Key Lockout	504
15.2.8	Keyboard Interfacing Circuit	504
15.3	Displays	507
15.3.1	Numeric Displays	507
15.4	Pin Configuration of 8279	509
15.4.1	Microprocessor Interface Pins	510
15.4.2	Keyboard Input Lines	510
15.4.3	Display Output Lines	511
15.4.4	Scan Lines	511

15.5	Internal Block Diagram of 8279	512
15.5.1	Microprocessor Interface and Control Section	513
15.5.2	Display Section	514
15.5.3	Keyboard Section	514
15.6	Commands of 8279	515
15.6.1	Keyboard/Display Mode Set Command	516
15.6.2	Program Clock Command	517
15.6.3	Read FIFO/Sensor RAM Command	517
15.6.4	Read Display RAM Command	517
15.6.5	Write Display RAM Command	518
15.6.6	Display Write Inhibit/Blanking	518
15.6.7	Clear Command	519
15.6.8	End Interrupt/Error Mode Set Command	519
15.7	Operating Mode of 8279	519
15.7.1	Input Mode	520
15.7.2	Output Modes of 8279	524
15.8	<b>Initialization of 8279</b>	532
15.9	Interfacing of 8279 with Microprocessor 8086	533
	<i>Exercises</i>	540
	<i>Multiple Choice Questions</i>	540
	<i>Descriptive Questions</i>	541

## PART IV

<b>16.</b>	<b>80186 and 80286 Microprocessors</b>	<b>545–571</b>
16.1	Introduction	545
16.2	Intel 80186 Microprocessor	545
16.3	Internal Block Diagram of 80186	546
16.3.1	Clock Generator	546
16.3.2	Bus Interface Unit	547
16.3.3	Chip-select/Ready Generation Logic	547
16.3.4	DMA Channels	548
16.3.5	Timers	549
16.3.5	Interrupt Controller	549
16.4	Pin Configuration of 80186	549
16.5	Microprocessor 80286	551
16.6	Architecture of 80286	552
16.7	Pin Description of 80286	553
16.8	Registers of 80286	554
16.8.1	Flag Register of 80286	554
16.8.2	Machine Status Word	555
16.8.3	GDTR, LDTR and IDTR	556
16.8.4	Task Register	556
16.9	Memory Organization and Segmentation	557
16.10	Memory Operating Modes	557
16.10.1	Real Mode	557
16.10.2	Protected Mode	558

16.11	Protected Virtual Address Mode (PVAM)	559
16.11.1	Selector	560
16.11.2	Descriptors	560
16.11.3	Access Right Byte	561
16.12	Local and Global Descriptor Table	563
16.13	Multitasking in 80286	563
16.14	Privilege Levels	564
16.15	Task Switching and Task Gates	565
16.16	Interrupts and Exceptions	565
16.17	Interrupt Descriptor Table	567
<i>Exercises</i>		568
	<i>Multiple Choice Questions</i>	568
	<i>Descriptive Questions</i>	571
<b>17.</b>	<b>Intel's 32-bit Microprocessors</b>	<b>572–612</b>
17.1	Introduction	572
17.2	Microprocessor 80386	572
17.3	Architecture of 80386	573
17.3.1	Central Processing Unit	573
17.3.2	Memory Management Unit	575
17.3.3	Bus Interface Unit	575
17.4	Signal Descriptions of 80386	576
17.5	Modes of Operation	577
17.6	Register Organization of 80386	577
17.6.1	General Purpose Registers	578
17.6.2	Flag Register of 80386	578
17.6.3	Segment Descriptor Registers	580
17.6.4	Control Registers	581
17.6.5	System Address Registers	581
17.6.6	Debug and Test Registers	582
17.7	Addressing Modes	584
17.8	Memory Organization and Memory Management Unit of 80386	584
17.8.1	Logical and Linear Address	584
17.8.2	Real Address Mode of 80386	585
17.8.3	Protected Mode of 80386	586
17.9	Global and Local Descriptor Tables	588
17.9.1	GATE Descriptors	589
17.9.2	Task-state Segments and Task Gates	589
17.9.3	Format of Descriptors	590
17.10	Paging	590
17.10.1	Paging Unit	591
17.10.2	Paging Descriptor Base Register	591
17.10.3	Page Directory	591
17.10.4	Page Tables	591
17.11	Virtual 8086 Mode of 80386	594
17.11.1	Translating a Virtual Address to a Physical Address	595

17.12	Memory Protection	596
17.12.1	Memory Protection across Ring Boundaries	596
17.12.2	Memory Protection within the Same Ring	597
17.12.3.	The Mandatory Access Control on Data and Code Access	598
17.13	Microprocessor 80486	599
17.14	Pin Configuration of 80486	599
17.15	EFLAG Register of 80486	601
17.15.1	AC (Alignment Check) Flag	602
17.16	Memory Organization of 80486	602
17.16.1	Cache Memory	603
17.17	Memory Management of 80486	603
17.18	Interrupt and Exceptions of 80386 and 80486	604
17.18.1	Interrupt Descriptor Table (IDT)	605
17.18.2	Exception and Interrupt Handling	607
17.18.3	Exception- or Interrupt-Handler Procedures	607
17.18.4	Interrupt Tasks	607
	<i>Exercises</i>	608
	<i>Multiple Choice Questions</i>	608
	<i>Descriptive Questions</i>	611

## 18. Today's Processors 613–661

18.1	Introduction	613
18.2	Pentium Processor	614
18.3	Architecture	614
18.4	Pin Configuration of Pentium	616
18.5	Registers	618
18.5.1	User Registers	619
18.5.2	System Registers	620
18.6	Integer Pipeline	621
18.7	Superscalar Execution	622
18.8	Floating Point Unit of Pentium	623
18.8.1	Floating-point Pipeline Stages	623
18.8.2	FPU Architecture	624
18.9	Branch Prediction	625
18.10	Cache Organization	626
18.11	Memory Organization of Pentium	627
18.11.1	Memory Management System of Pentium Processor	627
18.11.2	Paging Unit of Pentium	630
18.12	Introduction to MMX	631
18.12.1	The MMX Registers	631
18.12.2	The MMX Data Types	632
18.12.3	MMX Technology Instructions	633
18.13	Pentium Pro Processor	636
18.14	Pentium Pro Micro Architecture	637
18.14.1	Fetch/Decode Unit	637
18.14.2	Dispatch/Execute Unit	638
18.14.3	Retire Unit	639

18.15	Pin Configuration of Pentium Pro Processor	640
18.16	Intel Pentium II Processor	643
18.17	The Pentium II Processor Pipeline	643
18.18	Pin Configuration of Pentium II Processor	644
18.19	Intel Pentium III Processor	646
18.20	Signals of Pentium III	647
18.21	Intel Pentium IV Processor	648
18.21.1	Front End	649
18.21.2	Out-of-order Execution Logic	651
18.21.3	Integer and Floating-point Execution Units	653
18.21.4	Memory Subsystem	654
18.22	Hyper-threading Technology	655
18.23	Signals of Pentium IV	655
<i>Exercises</i>		657
	<i>Multiple Choice Questions</i>	657
	<i>Descriptive Questions</i>	659

---

# Preface

*Microprocessor 8086: Architecture, Programming and Interfacing* is the result of more than 18 years of teaching experience and regular feedback from the students regarding the difficulties faced by them in grasping the subject. In the past I have taught all the topics of this book using different textbooks. Somehow, I was not able to find any single book which could explain clearly the different aspects of 8086, in a student-friendly fashion. Some of these books started at too high level of complexity, gives students most of the time trivia and details before they could grasp the fundamental concepts. Others covered materials without cohesion. I wrote this book to address these problems and to improve students ability to learn. I hope that I have successfully resolved the problems found in other texts. To facilitate easy reading and understanding, I have tried to write in a language that is understandable to the average students so that they do not get intimidated by the subject. The fundamental concepts and techniques needed for all the beginners in the field have been incorporated in this book. It provides complete instructions about the Intel 8086 microprocessor, its programming, and the concept of interfacing of memory, IO devices and programmable chips. Practical work and knowledge-check questions contribute to building a thorough understanding with a practical focus. Each topic has been articulated with numerous examples.

This book contains many hardware and software exercises students can do to solidify their knowledge of microprocessors. In response to feedback from many Professors from various Universities, this book contains the following salient features:

- Focused coverage of 8086 microprocessor.
- Examples of programs with assembly language using MASM Assembler.
- Interfacing examples of 8086 with various programmable chips.
- Coverage of the entire advance Processors of Intel for 80186 to Pentium 4.
- Multimedia technologies are also discussed.

This book will serve as a complete text on microprocessor for undergraduate students of Electronics and Communication Engineering, Computer Science and Engineering, Information Technology and Electrical Engineering in all the Engineering Colleges and Institutes of India.

The flow of the book follows the life of a microprocessor design. This book is organized in four parts:

Part I comprises seven chapters. It starts with the introduction of analog and digital computers, computer generations, microcomputers and the architecture, organization of microprocessor 8085 along with its instruction set. It explains the architecture and organization of microprocessor 8086, its minimum and maximum mode configurations. This also gives a detailed description of the microprocessor 8086 instructions set and the Assembler directives before explaining the programming techniques with programming examples. Interrupts, types of interrupts supported by 8086 processor, and Priority Interrupt Controller (PIC) 8259 are also discussed in this section.

Part II includes next four chapters. Chapter 8 shows the importance of math coprocessor, data formats supported by 8087, block diagram and pin descriptions of 8087, interfacing of 8087 with 8086, instruction set of 8087 and, finally, the programming example of 8087.

Chapter 9 covers interconnection topologies, software aspects of multiprocessor systems, Semaphore, coprocessor coupled, loosely coupled and tightly coupled systems along with bus arbitration and control logics.

Chapter 10 explains types of communication systems, different types of data transfer schemes, serial transmission formats software based serial, and hardware controlled data transfer using USART 8251, with the help of numerous examples. This chapter also explains the block and pin diagram of 8251, synchronous and asynchronous modes of operations of 8251, and interfacing of 8251 with 8086.

Chapter 11 explores memory concepts. Basic memory cells and their architecture are discussed before going to memory interfacing. It also explains the various address decoding techniques and circuits besides introducing the concept of fold back addressing.

Part III consists of four chapters which cover the Programmable Interfacing Chips. Independent chapters are devoted to each programmable chip.

Chapter 12 deals with explanation of 8255, the programmable peripheral interfacing chip, and then it discusses its interfacing with 8086 using both IO mapped IO and memory mapped IO techniques. Many solved examples such as interfacing with ADC, DAC, and stepper motor are discussed here.

Chapter 13 discusses in detail about the features of programmable interval timer 8253/54, its block and pin diagrams, modes of operation, and its read and write operations. The chapter also explains interfacing of 8253/54 with 8086 in IO mapped IO and in memory mapped IO techniques with the help of solved examples.

Chapter 14 begins with a discussion on the need of DMA controlled data transfer. The chapter also explains burst, cycle stealing and transparent modes of DMA controller in detail before discussing the programmable DMA controller 8257, its block diagram, pin diagram and operating modes of 8257. Then it goes on to give an analysis of programming and interfacing of 8257 with 8086 with sufficient examples. The chapter also discusses programmable DMA controller 8237, its block diagram, pin diagram and operating modes of 8237 along with programming and interfacing of 8237 with 8086.

Chapter 15 starts with a discussion on key switch mechanism, hardware key debouncing and software key debouncing. Then it explores key roll-over schemes, techniques of key

encoding, matrix type keyboard, numerical display, alphanumeric display and multi-digit seven segment display. At the end, it discusses 8279 programmable keyboard, display interface, pin diagram and block diagram of 8279, its commands and operating modes with the help of examples.

Part IV of the book comprises three chapters which cover the advance processors 80186 to Pentium 4. Chapter 16 deals with the 16-bit processor 80186 and 80286. Chapter 17 includes the 32-bit processor 80386 and 80486 and Chapter 18 is devoted to the Pentium processors. This part discusses the block diagram, pin configuration memory organization, interrupts of 80186 to Pentium processors, as well as the real, protected and virtual protected modes.

**Sunil Mathur**

# Acknowledgements

It is needed a great pleasure and a moment of immense satisfaction for me to express my gratitude towards my mother Shayma Devi, my wife Somprebha, my son Vaibhav and my daughter Aeshita for their priceless help and moral support during the writing of this book.

I pay my special thanks to Dr. Nand Kishore Garg, Chairman, Maharaja Agrasen Educational Trust and Prof. M.L. Goel, Director, Maharaja Agrasen Institute of Technology for providing me ample time and required resources to write this book.

I extend my special thanks and gratitude to the following professors, who boosted my morale, gave me great pedagogical tips, and tirelessly examined the entire book. They have been a great influence in the development of this book.

- Prof. A. Bhattacharyya (Head), Department of Electronics and Communication, Delhi College of Engineering.
- Mr. J. Panda Assistant Professor, Department of Electronics and Communication, Delhi College of Engineering.
- Prof. M.P. Tripathi (Head), Department of Electronics and Communication, Maharaja Agrasen Institute of Technology.

I also extend my thanks to Ms Rashmi Gupta and Ms Bhavana Aggarwal to review the contents of this book and to give valuable suggestions.

I remember so many students who participated actively in the teaching–learning process, which, in effect, gave me tips and suggestions on how this book should be. I thank them all and place on record my appreciation of their curiosity and determination to delve into the subject beyond mere superficiality.

Equally, I am thankful to my colleagues and students who compelled me to write this book. Finally, I would like to thank all the people who directly or indirectly have contributed to the completion of this book.

**Sunil Mathur**

# **Part I**

- 1. Architecture and Organization of Microprocessors and Microcomputers**
- 2. Introduction to 8086**
- 3. 8086 Based System**
- 4. Instructions Set of 8086**
- 5. Assembler Directives**
- 6. Programming of 8086**
- 7. Interrupts of 8086**

# 1

# Architecture and Organization of Microprocessors and Microcomputers

## 1.1 INTRODUCTION

The word *microprocessor* was introduced by Intel Corporation. The microprocessor is a single chip microcomputer, i.e. all the functional components of a computer are inbuilt on a single chip using VLSI technology. It consists of an arithmetic and logic unit, register unit and control unit. These three units are fabricated on a single chip. The microprocessors are generally characterized by speed, word length, architecture and instruction set.

8-bit, 16-bit and even 32-bit microprocessors are quite matured today. Several independent microprocessor families are available in the market. These microprocessors have been widely used in the design of new electronic equipments and computers.

In this chapter we will discuss computer languages and study the architecture and organization of microprocessor, microprocessor operations, internal architecture of a microprocessor, internal architecture of 8085, 8085 pin description, 8085 functional description, demultiplexing of buses, generation of control signals and how a program executed.

## 1.2 COMPUTER LANGUAGES

Computer languages are the languages which a computer can understand. Computer languages can be classified as high-level language and low-level language.

### 1.2.1 Low-level Language

Low-level languages are machine-dependent language. Machine dependent means that a programming language is designed for a particular computer and reflecting its internal machine code; low-level languages are, therefore, often described as *machine-oriented languages*.

They cannot be easily converted to run on a computer with a different central processing unit, and they are relatively difficult to learn because a detailed knowledge of the internal working of the computer is required. Low-level languages can further be classified as machine language and assembly language.

### 1.2.2 Machine Language

In machine language the instructions are written in binary. Machine languages are the only languages understood by computers. While easily understood by computers, machine languages are almost impossible for humans to use because they consist entirely of numbers. Machine languages, which are very hard to remember, write down, or correct, with short codes chosen to remind the programmer of the instructions they represent, are replaced by a mnemonic-based low-level language known as *assembly language*.

### 1.2.3 Assembly Language

Instructions in assembly language are written in English alphabets but in coded form. An assembly language contains the same instructions as a machine language, but the instructions and variables have names instead of being just numbers. For example, the binary-code instruction 13H (0001 0011) that means store the contents of the accumulator in memory may be replaced by the mnemonic STA.

As machine languages are the only languages understood by computers assembly languages are to be converted into machine languages. This conversion is done either by manually or by a software program known as *assembler*.

### 1.2.4 High-level Language

High-level languages are machine independent languages. Programs written for a particular machine can run on other machines also. These high-level languages are also known as *problem-oriented languages* because they are designed to solve particular problems. The main advantages of these high-level languages are that they are easier to read, write, and maintain.

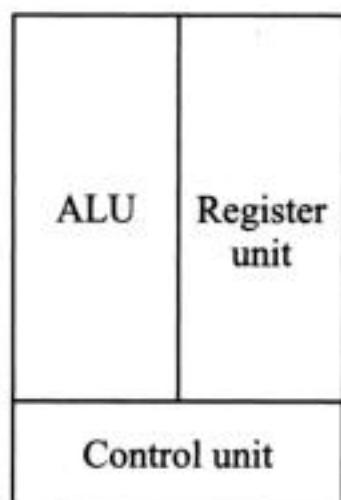
Programs written in high-level languages are converted into machine language by an interpreter or a compiler.

## 1.3 MICROPROCESSOR

In 1971 Intel Corporation fabricated the complete CPU of a computer system on a single chip and call it Microprocessor. The word micro was introduced to signify the size of the processing unit and not the operations it perform.

Microprocessor is a complex IC of sequential circuits. It is made up of VLSI technology. It is a programmable logic device, designed with registers, flip-flops, and timing elements. It has a set of instructions designed to manipulate data and communicate with peripherals. The process of data manipulation and communication with peripherals is determined by the logic design of the microprocessor. The logic design of the microprocessor is called microprocessor architecture. Figure 1.1 shows the diagram of a basic microprocessor. As shown in Figure 1.1

the microprocessor consists of three parts, viz. Arithmetic and Logic Unit (ALU), Register Unit (RU) and Control Unit (CU).



**Figure 1.1** Microprocessor unit.

### 1.3.1 Arithmetic and Logic Unit

In this area of the microprocessor, the computing functions are performed on the data. The ALU performs arithmetic operations such as addition and subtraction, and logic operations such as AND, OR and EX-OR results are stored in the registers or in memory unit or send to output unit.

### 1.3.2 Register Unit

This area of the microprocessor consists of various registers. The registers are used for temporary storage of data during execution of a program. Some of the registers are accessible to the users through various instructions.

### 1.3.3 Control Unit

The control unit provides necessary timing and control signals to all the operations in the microprocessor and peripherals including memory.

## 1.4 MICROPROCESSOR OPERATIONS

Microprocessor is a complex IC of sequential circuits. It is a programmable logic device, designed with registers, flip-flops, and timing elements. It has a set of instructions designed to manipulate data and communicate with peripherals. The process of data manipulation and communication with peripherals is determined by the logic design by the microprocessor, the logic design is called *architecture*.

All of the operations of the microprocessor can be classified into one of three types:

- Microprocessor initiated operations
- Internal operations
- Peripheral initiated operations.

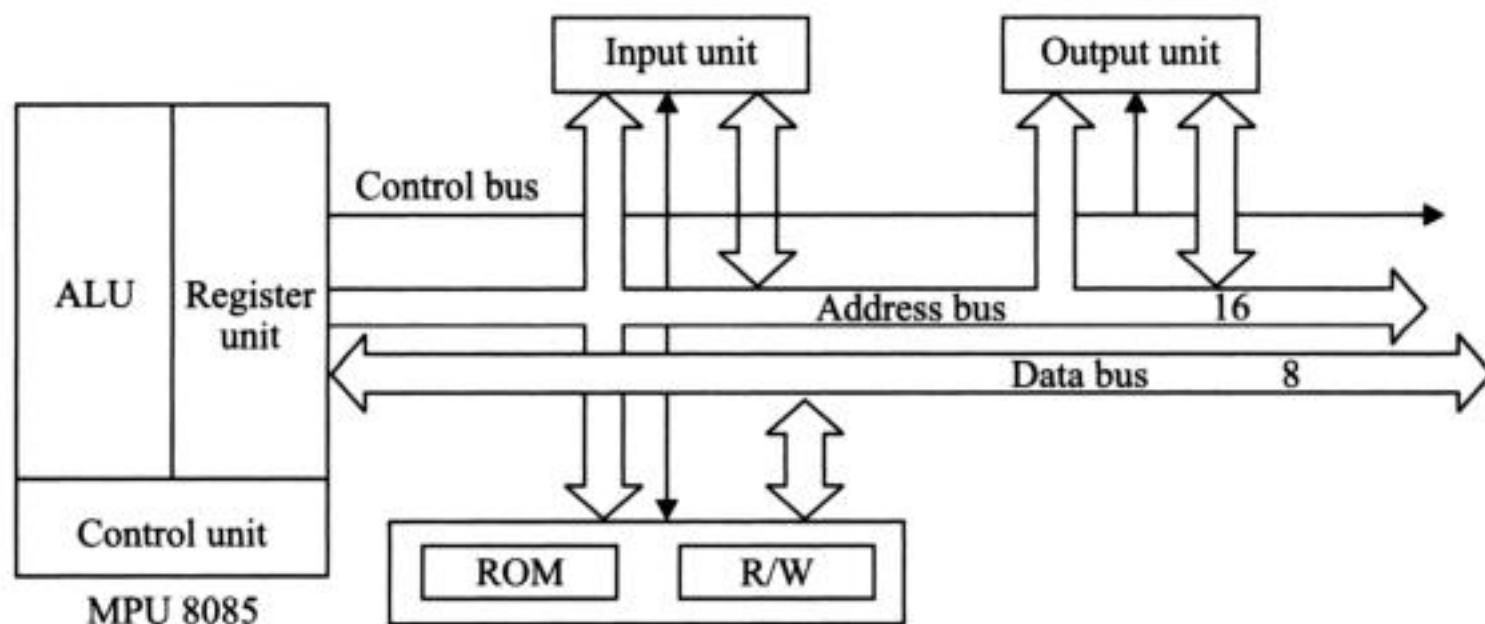
### 1.4.1 Microprocessor Initiated Operations

These are the operations that the microprocessor starts itself. These are the operations which are initiated by the microprocessor and the peripheral devices will execute these operations. These are usually one of the following operations:

1. Memory Read: Reads data from memory
2. Memory Write: Writes data into the memory
3. IOR: Accepts data from input devices
4. IOW: Sends data to output devices.

All these operations are part of the communication process between microprocessor and peripheral devices or memory. To perform these communication operations, microprocessor unit executes the following steps:

- Identify the address of memory location or the peripheral device
- Provide timing or synchronization signals
- Transfer the data.



**Figure 1.2** Bus system of 8085.

The microprocessor performs the above three functions through the communication lines called *system bus* (bunch of wires). As three types of information are communicated, hence there are three types of buses. These buses are:

1. **Address bus:** It carries the address of a memory location or I/O devices that the MPU wants to access. It is a unidirectional bus (from MPU to peripheral) generally denoted by A. In microprocessor 8085, it is 16-bit wide, i.e.  $A_0 - A_{15}$ .
2. **Data bus:** It is used to transfer data between the processor and memory and I/O devices. It is bidirectional in nature. In microprocessor 8085, there are 8 data lines denoted by  $D_0$  to  $D_7$ .
3. **Control bus:** It is used to carry control signals between MPU and various devices connected to it. It also carries synchronization and timing signals. The bus is somewhat confusing term for it. In fact, these are individual lines that provide signals to the devices connected to microprocessor.

Figure 1.2 shows the bus system of 8085.

### 1.4.2 Internal Data Operations

These are the operations which are internally performed by the microprocessor. The internal architecture of microprocessor determines how and what operations can be performed with the data. The internal operations performed by microprocessor are classified into five groups:

1. Store 8-bit data.
2. Perform arithmetic and logic operations.
3. Test for the conditions.
4. Sequence the execution of instructions.
5. Store data temporarily during the execution in the defined R/W memory locations called *stack*.

To perform these operations we require registers, ALU and control logic and path for communication, i.e. bus system.

To perform the first operation, i.e. to store 8-bit data, microprocessor has the register unit.

#### **Register unit**

Microprocessor 8085 has six 8-bit general purpose data registers. During program execution these registers are used to store 8 or 16-bit data. These registers are identified as B, C, D, E, H, and L. If during some operation microprocessor deals with 16-bit data, then these registers can be paired. This register pairing is fixed, i.e. register B will pair with register C, D with E and H with L. The higher byte of the 16-bit data will go to either B, D, and H registers whereas lower byte will go to C, E, and L.

These registers are programmable means that user can use them to load copy and move data from the registers by using instructions, e.g. MOV B, C. These registers can be considered as memory location as they are in the microprocessor, expects they are named by alphabet for the user convention. Some microprocessors do not have these registers so they use memory locations for this purpose.

To perform the second operations, i.e. to perform arithmetic and logic operations, microprocessor has ALU and a register called Accumulator A.

#### **Accumulator**

The accumulator is an 8-bit general purpose register which is the part of ALU. This register is used to store 8-bit data to perform arithmetic and logical operations. Accumulator participates in all the arithmetic and logical operations. After the operation the result is stored in the accumulator. Accumulator is the only register which communicates with IO. Accumulator is also identified as register A.

To perform the third operation, i.e. to test the condition, if any, microprocessor has another 8-bit register called *flag register*.

#### **Flag register**

The flag register is an 8-bit register. These 8-bits of this register are independent bits. Out of these 8-bits, five are used to represent five different data conditions. These are:

1. **Carry flag:** If after an arithmetic operation the result in the accumulator is greater than 8-bits, then carry flag is set, i.e. CY = 1, otherwise it is '0'.

2. **Zero flag:** If after an arithmetic operation the result in the accumulator is zero, then zero flag is set, i.e.  $Z = 1$ , otherwise ‘0’.
3. **Sign flag:** If after an arithmetic operation the MSB of the result in the accumulator is 1, then sign flag is set, i.e.  $S = 1$ , otherwise ‘0’.
4. **Parity flag:** If after an arithmetic operation the result contains even number of 1s, then parity flag is set, i.e.  $P = 1$ , otherwise ‘0’.
5. **Auxiliary flag:** This status flag is set when there is a carry from bit 3 to 4.

These five flip-flops or status flags are set or reset according to the result of operation. The microprocessor uses them to perform the testing or setting for data conditions.

To perform the fourth operation, microprocessor 8085 has another register called *program counter*.

### **Program counter**

This is a 16-bit register that deals with the fourth operation of the list, i.e. sequencing the execution of instructions. This register is a memory pointer. The function of the program counter is to point the memory address from which the next byte is to be fetched. When one byte is being fetched, the contents of the program counter is increased by 1 to point the next memory location. Memory locations have 16-bit address, that is why this register is of 16-bit.

To perform the fifth operation of the microprocessor internal operations, microprocessor 8085 has another register called *stack pointer* (SP).

### **Stack pointer**

The stack pointer is also a 16-bit memory address register used as memory pointer in the stack memory. The stack memory is predefined (defined by the user) in R/W memory. The stack memory is initialized by the programmer a 16-bit address in the stack pointer. The stack pointer is automatically incremented by two with each withdrawal of a 16-bit data and decremented by 2 by each loading of 16-bit data on to the stack memory.

### **1.4.3 Peripheral or Externally Initiated Operations**

External devices can initiate the following operations, for which individual pins on the microprocessor chip is assigned. These operations are:

1. **Reset:** When reset pin is activated, all internal operations are stopped and the program counter is reset to 0000. Program execution again begins from zero memory address.
2. **Interrupt:** Interrupt is a process in which microprocessor is asked to suspend its current operation and execute some emergency task or operation. Microprocessor 8085 has five different pins which are used to interrupt the processor. Whenever microprocessor receives a signal on any of these pins, the microprocessor suspends its current operation and executes the emergency task. The emergency program which is executed by the process in response to an interrupt signal is known as an Interrupt Service Routine (ISR). When the ISR is completed, the microprocessor returns to its previous operations and resumes its suspended operation.

3. **Ready:** The 8085 has a pin called *READY*. This pin is used to synchronize the speed of the microprocessor with the slower peripherals. When this *READY* signal is low, the microprocessor will remain in wait state.
4. **Hold:** Hold is a process in which microprocessor is asked by the peripheral device to suspend its current operation and release the bus system so that the requesting device can use the microprocessor buses. To initiate this Hold operation, the 8085 has a pin called *HOLD*.

## 1.5 8085 FUNCTIONAL DESCRIPTION

The 8085A is an 8-bit processor because its accumulator size is of 8-bit (It is the size of the accumulator which decides the word length of a processor). It operates on a single +5 volt power supply and at a frequency of 3 MHz.

The data bus of 8085A is multiplexed with lower-order address bus. So the address bus is divided, as the higher-order bus which carry the higher 8-bit address and the lower order multiplexed address/data bus which carries the lower 8-bit address. The address is sent out on the multiplexed lines during the first T state of the machine cycle. For the remaining T states of the machine cycle, the multiplexed address/data bus is used to carry the data for memory or I/O.

The 8085A generates three control signals RD<sup>-</sup>, WR<sup>-</sup>, and IO/M<sup>-</sup> for bus control. These three signals are used to generate the four control signals IOR<sup>-</sup>, IOW<sup>-</sup>, MEMR<sup>-</sup> and MEMW<sup>-</sup>. The 8085A also provides pins the serial input data (SID), and the serial output data (SOD) for serial communication.

The 8085 has six interrupt related signals which can cause five different hardware interrupts. These interrupts are INTR, RST 5.5, RST 6.5, RST 7.5, and TRAP. TRAP is a non-maskable interrupt which have the highest priority. The RST 5.5, RST 6.5, and RST 7.5 are maskable interrupt.

## 1.6 INTERNAL ARCHITECTURE OF 8085

The internal architecture of microprocessor 8085 can be divided into five parts. These five parts are:

1. Register unit
2. Arithmetic and logical unit
3. Control unit
4. Interrupt unit
5. Serial IO unit.

These five units themselves consist of other internal parts. The following section gives a detailed overview of these parts. Figure 1.3 shows the detailed block diagram of 8085.

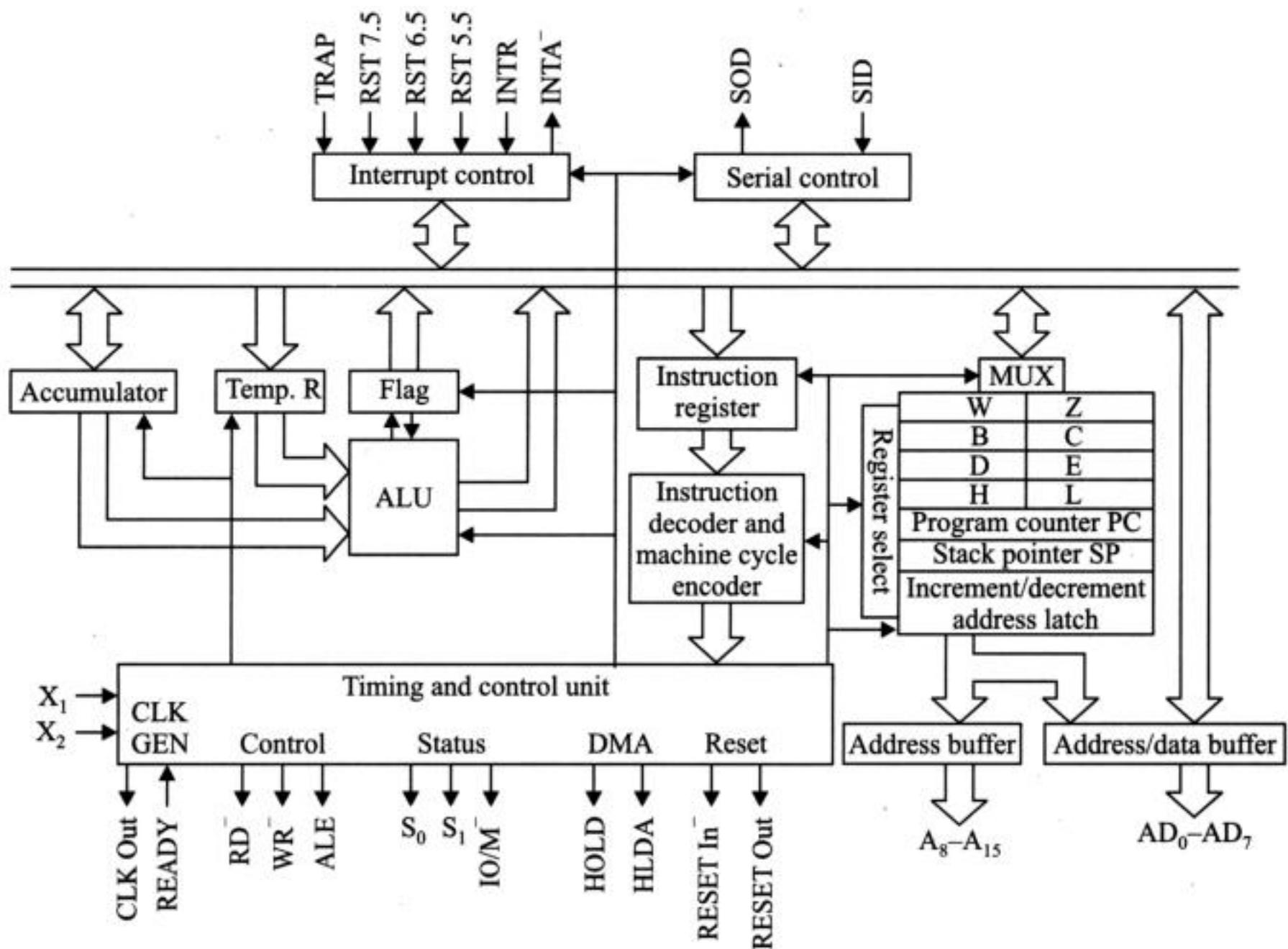


Figure 1.3 Internal architecture of 8085.

### 1.6.1 Register Unit

As shown in Figure 1.4, the register unit consists of six general purpose data registers B, C, D, E, H, and L, two internal registers W and Z, two 16-bit address register PC and SP, one increment/decrement counter register and one MUX/DEMUX.

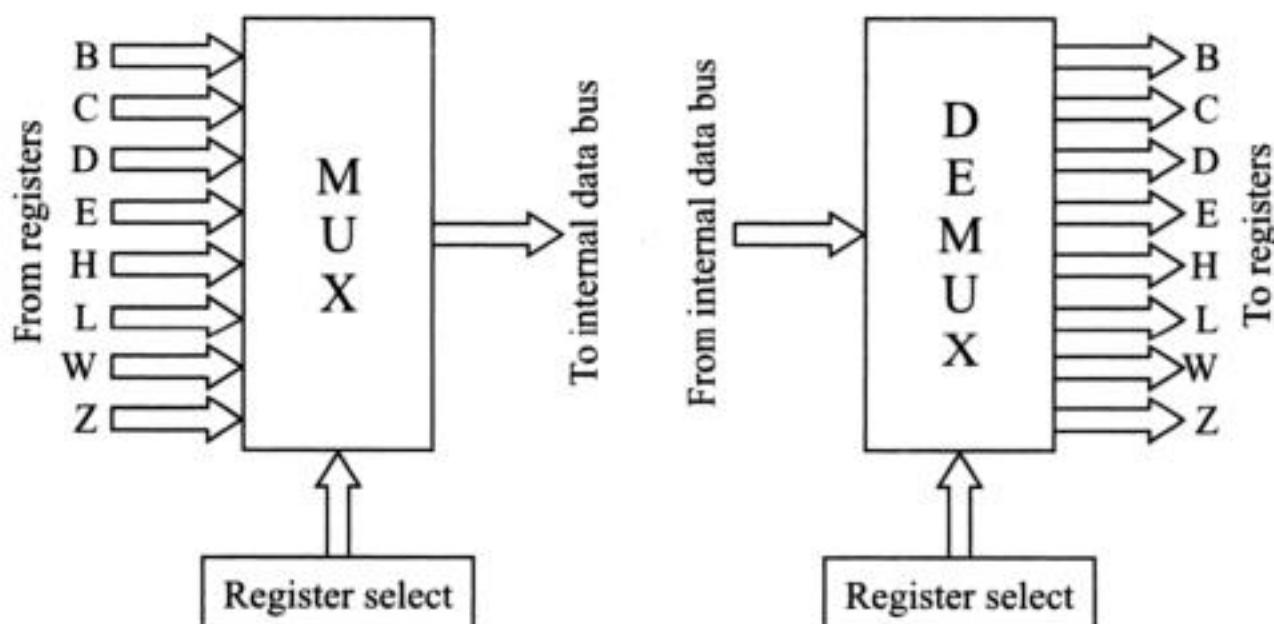
Accumulator A <sub>(8)</sub>	Flag register
B <sub>(8)</sub>	C <sub>(8)</sub>
D <sub>(8)</sub>	E <sub>(8)</sub>
H <sub>(8)</sub>	L <sub>(8)</sub>
Stack pointer (SP) <sub>16</sub>	
Program counter (PC) <sub>16</sub>	

Figure 1.4 Registers of 8085.

**1. General purpose data register:** The 8085/8080A has six general purpose data registers to store 8-bit data. These registers are named B, C, D, E, H, and L as shown in Figure 1.4. The user can use these registers to store or copy a data temporarily during the execution of a program by using data transfer instructions. Though these registers are of 8-bits but whenever the microprocessor has to handle 16-bit data these registers can be combined as register pairs: BC, DE, and HL.

Apart from these six general purpose data registers, the 8085 microprocessor also has two 8-bit internal data registers. These registers are named W and Z. These registers are only for internal operation of the microprocessor and not available to the user. Microprocessor uses these registers internally, for example, in case of CALL and XCHG instructions.

- 2. Program counter (PC):** The program counter (PC) is 16-bit register which is used to sequence the execution of instructions. This register is actually a memory pointer which always points to a memory location from where the microprocessor has to fetch the next byte. Memory locations have 16-bit addresses, and that is why this register is of 16-bits. When a byte (machine code) is being fetched by the microprocessor, the program counter is automatically incremented by one to point to the next memory location.
- 3. Stack pointer (SP):** As we have already discussed, the stack pointer is a 16-bit address register which is used as a memory pointer in the stack memory area.
- 4. Increment/decrement counter:** This counter register is used to increment or decrement the contents of the various registers available in the register unit. For example, every time microprocessor accesses a memory, its PC register is incremented.
- 5. Mux/demux unit:** This unit is used to select a register out of all the available registers. This unit (shown in Figure 1.5) behaves as a Mux when data is going from the register to the internal data bus. It behaves as a Demux when data is coming to a register from the internal data bus of the microprocessor. The register select will behave as the function selection lines of the Mux/Demux.



**Figure 1.5** Multiplexer unit.

- 6. Address buffer register and data/address buffer register:** These registers hold the address/data, received from PC (Program Counter)/internal data bus and then load the external address and data buses. These registers actually behave as the buffer stage between the microprocessor and external system buses.

### 1.6.2 Control Unit

As the name shows the control unit is responsible for the generation of the control signals within the microprocessor to execute an instruction. In reality it causes certain connections between blocks of the microprocessor to be opened or closed, so that data goes where it is required. It also tells the ALU about the operation which it has to perform. The control unit itself consists of three parts, first the instruction register; second the instruction decoder and machine cycle encoder and third the control and timing unit.

- 1. Instruction register (IR):** This register holds the machine code of the instruction. When microprocessor executes a program, it reads the opcode from the memory, this opcode is stored in the instruction register.
- 2. Instruction decoder and machine cycle encoder:** The IR sends the machine code to this unit. This unit, as its name suggests, decodes the opcode and finds out what is to be done in response of the coming opcode and how many machine cycles are required to execute this instruction.
- 3. Control and timing unit:** The control unit generates signals within microprocessor to carry out the instruction, which has been decoded. In reality it causes certain connections between blocks of the microprocessor to be opened or closed, so that data goes where it is required, and therefore ALU operations occur.

### 1.6.3 Arithmetic Logic Unit

The ALU consists of accumulator, flag register and temporary register. The Arithmetic Logic Unit (ALU), as its name suggests, performs the arithmetic operations such as Addition, Subtraction, Increment and Decrement and logic operations such as AND, OR, EXOR, etc. ALU uses data from memory/register and accumulator to perform arithmetic and always stores result of operation in accumulator.

- 1. Accumulator:** The accumulator is an 8-bit data register and is a part of arithmetic and logic unit (ALU). The accumulator is also identified as register A. It is just like the other data registers of 8085 with some additional features. These special features which only the Accumulator can perform are:
  - It participates in all the arithmetic and logical operations, i.e. out of the two operands one must be stored in accumulator.
  - After the operation, the result is stored in the accumulator.
  - The microprocessor always communicate with the IO's through accumulator only.
- 2. Flags:** The microprocessor 8085 consists of an 8-bit register called the flag register. The 8-bits of this register are independent with each other or we can say that it is made up of 8 flip flops. Out of these 8 independent flip-flops, five are used to represent five

different data conditions by setting or resetting these five flip-flops. These five flip-flops are called flag and are named Zero (Z), Carry (CY), Sign (S), Parity (P), and Auxiliary Carry (AC) flags. The bit positions of these flag bits in the flag register are shown in Figure 1.6. The microprocessor uses these flags to set and test data conditions. For example, after an addition of two numbers, if the sum in the accumulator is larger than eight bits, the flip-flop uses to indicate a carry – called the Carry flag (CY) – is set to one. When an arithmetic operation results in zero, the flip-flop called the Zero (Z) flag is set to one.

Figure 1.6 shows this 8-bit flag register, adjacent to the accumulator.

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
S	Z	X	AC	X	P	X	CY

X = Not specified

Figure 1.6 Format of the flag register.

These flags are very important in decision-making process of the microprocessor. The conditions (set or reset) of the flags are tested by the software instructions. For example, the instruction JNC (jump on no carry) is implemented to change the sequence of a program when CY flag is reset.

- (i) **Z (Zero) flag:** This flag indicates that the result of a mathematical or logical operation is zero or not. If the result of the current operation is zero then this flag will set, otherwise reset.
- (ii) **CY (Carry) flag:** This flag indicates that whether during an addition or subtraction operation, carry or borrow generated or not, if generated then this flag bit will set. (This flag may also be set before a mathematical operation as an extra operand to certain instructions.)
- (iii) **AC (Auxiliary carry) flag:** It shows carry propagation from D<sub>3</sub> position to D<sub>4</sub> position. To understand it better, consider the example.

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	1	1	0	0
0	0	1	0	1	0	1	1
1	0	1	1	0	1	1	1

In this example a carry generates from D<sub>3</sub> bit position and propagates to the D<sub>4</sub> position. This carry is called *auxiliary carry*. This flag is never used for setting or testing a condition.

- (iv) **S (Sign) flag:** Sign flag indicates that whether the result of a mathematical operation is negative or positive. If the result is positive, then this flag will reset and if the result is negative, this flag will set. This bit, in fact, is replica of the D<sub>7</sub> bit.

- (v) **P (Parity) flag:** This flag indicates whether the current result is of even parity (1) or of odd parity (0).

#### 1.6.4 System Bus of 8085

Typical system uses a number of buses which are nothing but collection of wires, which transmit binary numbers, one bit per wire. A typical microprocessor communicates with memory and other devices (input and output) using three buses: address bus, data bus and control bus.

1. **Address bus:** Address bus is a unidirectional group of lines, i.e. numbers only sent from microprocessor to memory, not other way. Microprocessor 8085 has 16 address lines, one wire for each bit, therefore  $16 \text{ bits} = 16 \text{ wires}$ . With the help of 16 bits we can generate  $2^{16}$  combinations, if one combination is allotted to a single memory location as an address that means 64 K memory locations can be addressed by 8085 microprocessor.
2. **Data bus:** Data bus is used to carry the data (in binary form) from microprocessor to the peripheral and vice versa. That is, the data can go from the microprocessor to the peripherals as well as from peripheral to microprocessor, so the data bus is bidirectional. In microprocessor 8085, there are 8-data lines. Apart from data, these lines also carries instructions from memory to the microprocessor. If the microprocessor has to carry 8-bit data, then the whole data is transferred in a single cycle but if the data is of 16-bits, then the lower 8-bits are transferred in the first cycle and the upper 8-bits in the next cycle.
3. **Control bus:** The control bus consists of individual lines which are used to carry the control information from the microprocessor to the peripherals or from peripherals to the microprocessor, e.g. Read line, Write line and the interrupt lines. The Read control line indicates that microprocessor wants to read from either the memory or from the IO. Similarly, the write control signal indicates that microprocessor wants to write something in the memory or in the IO. The control bus may also include clock line(s) for timing/synchronizing, ‘interrupts’, ‘reset’, etc. The control bus carries control signals partly unidirectional, partly bidirectional.

#### 1.7 8085 PIN DESCRIPTION

Microprocessor 8085 is a 40-pin IC which operates on +5 V power supply and 3 MHz frequency. These 40 pins are divided into six groups according to their functions. These groups are:

1. Frequency and power supply signals
2. Higher order address bus
3. Multiplexed address/data bus
4. Control and status signals
5. Serial IO signals
6. Externally or peripheral initiated signals.

The pin configuration of microprocessor 8085 is shown in Figure 1.7.

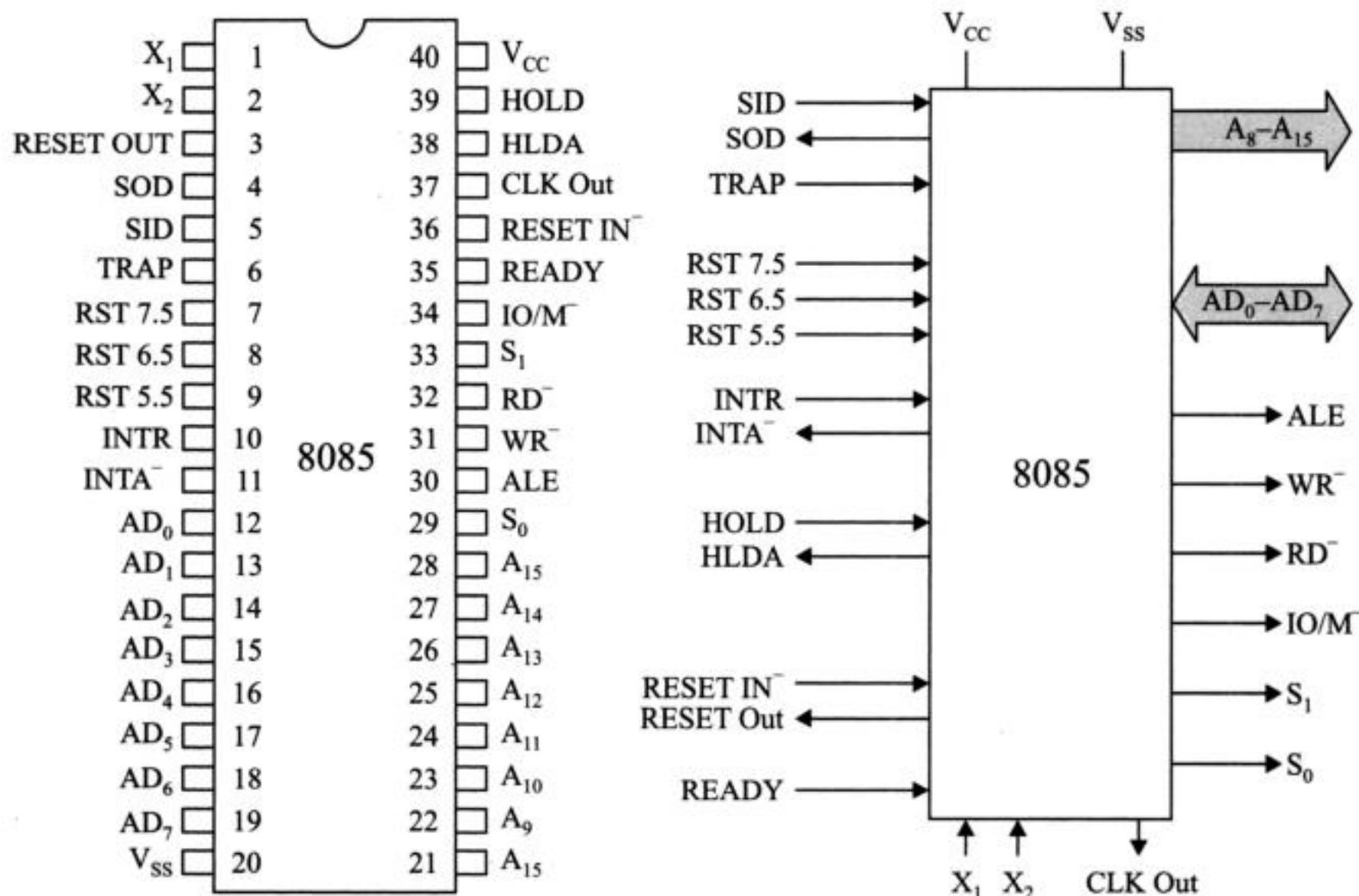


Figure 1.7 Pin configuration of 8085.

### 1.7.1 Group 1: Power Supply and Frequency Signals

- X<sub>1</sub>, X<sub>2</sub> (Input):** These are the two input lines across which a Crystal or R/C oscillator circuits is connected to provide the required clock frequency to the microprocessor. The frequency generated by the oscillator is divided by 2 to give the internal operating frequency of the microprocessor. The input frequency is divided by 2 because the frequency is applied to the system through a T flip-flop which divides the incoming frequency by 2 (as shown in Figure 1.8).
- V<sub>CC</sub>:** +5 volt supply.
- V<sub>SS</sub>:** Ground reference.

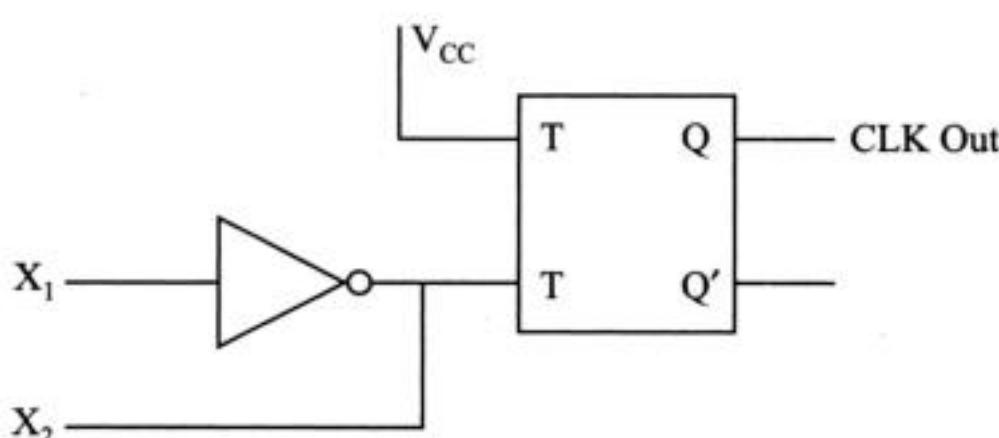


Figure 1.8 Internal clock.

### 1.7.2 Group 2: Higher Order Address Bus $A_8-A_{15}$

Instead of having 24 pins for address and data lines, 8085 has only 16 pins. Out of 16 pins, 8 pins are used to carry the higher order address and the other 8 pins are multiplexed to carry the address as well as the data. This multiplexing is done to keep the number of pin as minimum as possible. The most significant 8 bits of the memory address or the 8-bits of the IO address is carried by these lines. These lines are tri-stated during Hold and Halt modes.

### 1.7.3 Group 3: Multiplexed Address/Data Bus $AD_0-AD_7$

The 8085A uses a multiplexed data bus. These lines are time multiplexed with the lower 8-lines of the address bus. Lower 8 bits of the memory address or I/O address appear on the bus during the  $T_1$  state of a machine cycle. It then becomes the data bus during the second and third clock cycles. These lines are tri-stated during Hold and Halt modes.

### 1.7.4 Group 4: Control and Status Signals

1. **ALE (Address latch enable) (output):** ALE, as its name suggests, enables the address latch to store the address during the demultiplexing operation. It occurs during the first T state of every machine cycle. Whenever microprocessor sends a valid address on the multiplexed lines, it also makes the ALE signal high. ALE can also be used to strobe the status information. ALE is never tri-stated.
2. **RD<sup>-</sup> (read) (output):** Read is an active low output control signal. When this signal is low, it indicates that the microprocessor wants to read a data either from memory or from IO device. Microprocessor activates this signal during the  $T_2$  state of the machine cycle as in this T state the data bus is ready to carry the data. The read signal is tri-stated during Hold and Halt modes.
3. **WR<sup>-</sup> (write) (output):** Write is an active low output control signal. When this signal is low it indicates that the microprocessor wants to write a data either into memory or IO device. Microprocessor activates this signal during the  $T_2$  state of the machine cycle as in this T state the data bus is ready to carry the data. The write signal is tri-stated during Hold and Halt modes.
4. **IO/M<sup>-</sup> (output):** The Read and Write signals indicates that the microprocessor wants to read or write a data but do not specify from where this read or write operation will take place. This is indicated by the IO/M<sup>-</sup> signal. When this signal is low, it means a read or write operation will take place from/to memory. When this signal is high, it means the operation is with reference to IO. This signal is tri-stated during Hold and Halt modes.
5. **S<sub>0</sub>, S<sub>1</sub> (status signal) (output):** These are the two data bus status signal. The four combination of these signal give the information of what the microprocessor is doing. Table 1.1 shows the encoding of these two signals.

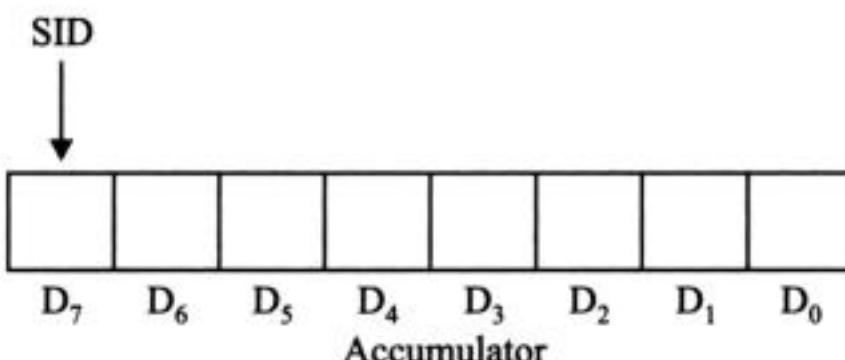
**Table 1.1** Encoded status of the status signals

<b>S<sub>1</sub></b>	<b>S<sub>0</sub></b>	<b>Operation</b>
0	0	Halt
0	1	Write
1	0	Read
1	1	Fetch

S<sub>1</sub> can be used as an advanced R/W status.

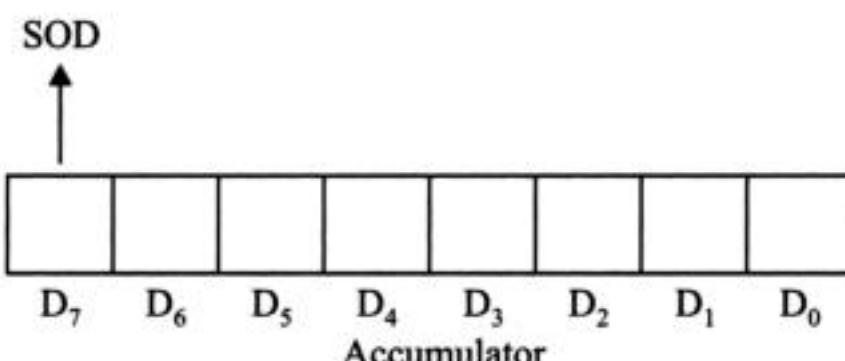
### 1.7.5 Group 5: Serial IO Signal

- 1. SID (input):** Serial input data line. This line is used in serial data communication. Through this pin the serial data is received by the processor. The data on this line is loaded into accumulator bit 7 whenever a RIM instruction is executed. Figure 1.9 shows the SID operation.



**Figure 1.9** SID operation.

- 2. SOD (output):** Serial output data line. This line is used in serial data communication. Through this pin the serial data is transmitted by the processor. The output SOD is set or reset as specified by the SIM instruction. Figure 1.10 shows the SOD operation.



**Figure 1.10** SOD operation.

### 1.7.6 Group 6: Externally or Peripheral Initiated Signals

- 1. TRAP (input):** Trap is a non-maskable interrupt which has the highest priority. At the same time it is recognized as INTR. This interrupt cannot be masked or disabled. This is a vectored interrupt. It is edge as well as level triggered.
- 2. RST 5.5, RST 6.5, RST 7.5: RESTRAT interrupt (inputs):** These three interrupts are called maskable interrupts. The mask can be set to any or all of these interrupts by SIM instruction. These interrupts can also be disabled by the software instruction DI.

All these three interrupts are vectored interrupts. The numbers 7.5, 6.5 and 5.5 correspond to their vector locations.

RST 7.5 has the highest priority and it is edge triggered.

RST 6.5 is level triggered.

RST 5.5 has the lowest priority and it is edge triggered.

The priority of these interrupts is ordered as shown above. These interrupts have a higher priority than the INTR.

**3. INTR (interrupt request) (input):** INTR is used as a general purpose interrupt. It is sampled only during the next to the last clock cycle of an instruction cycle. When it is active, the Program Counter (PC) will stop incremented and an interrupt acknowledge signal is issued by the processor. During this INTA cycle a RESTART or CALL instruction can be inserted to transfer the control to the interrupt service routine (ISR). The INTR is enabled and disabled by the software instructions EI (enable interrupt) and DI (disable interrupt). It is also disabled by Reset and immediately after an interrupt is accepted.

**4. INTA<sup>-</sup> (interrupt acknowledge) (output):** This signal is generated by microprocessor in response to the INTR. When microprocessor accepts the INTR, it executes an INTA machine cycle. This cycle is exactly the same as that of read cycle except that instead of RD<sup>-</sup> signal microprocessor sends the INTA<sup>-</sup> during the T<sub>2</sub> and T<sub>3</sub> states. It can be used to activate the 8259 Interrupt chip or some other interrupt port.

**5. READY (input):** This signal is used to synchronize the slower peripherals with microprocessor. Whenever the peripheral devices are ready to send or receive the data, they will make the Ready high. The microprocessor performs the read or the write operation and then enters into the wait state till the next time it receives a logic high on the Ready pin.

**6. HOLD (hold request) (input):** HOLD is bus request from a competent bus master. Whenever a competent bus master like DMAC (Direct memory access controller) wants to transfer data between memory and the IO, it sends a request on the HOLD pin of the microprocessor. On receiving the Hold request signal, the microprocessor suspends its current operation and relinquishes the buses as soon as the completion of the current machine cycle. Internal processing can continue. The processor regains the control of the buses when the Hold request is dropped by the DMAC. When the Hold is acknowledged, the Address, Data, RD<sup>-</sup>, WR<sup>-</sup>, and IO/M<sup>-</sup> lines are tri-stated. This signal is polled by the microprocessor in the last T state of every machine cycle. Figure 1.11 shows the HOLD operation.

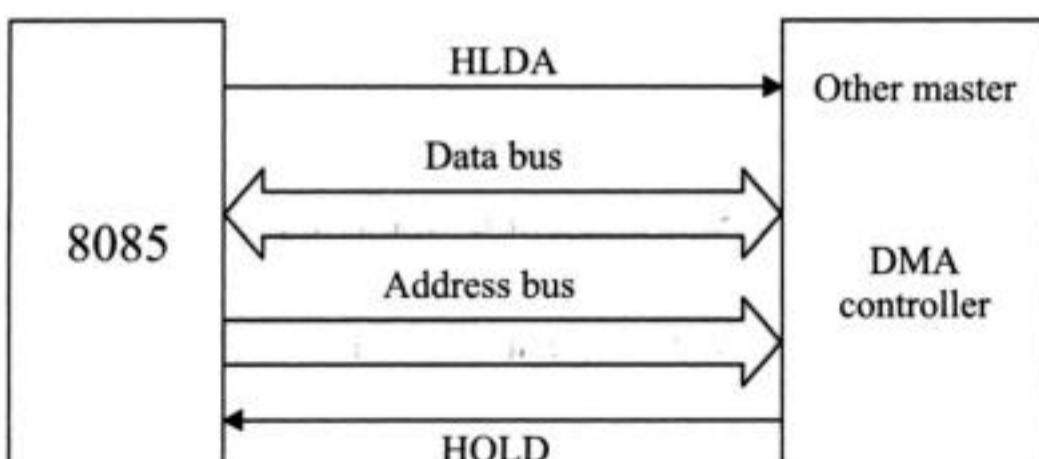


Figure 1.11 HOLD operation.

7. **HLDA: HOLD acknowledge (output):** On receiving the HOLD request the microprocessor completes the current machine cycle and then suspends its operation, release the buses and sends a HOLD acknowledge signal to the DMAC. HLDA goes low after the Hold request is dropped. The microprocessor takes the buses one-half clock cycle after HLDA goes low.
8. **RESET IN<sup>-</sup> (input):** This signal is used to reset the processor. When microprocessor receives a signal on this pin, it clears the Program Counter and resets the interrupt enable and HLDA flip-flop. Except the Instruction register all the general purpose data registers and the flag register remain unaffected by the reset signal. The microprocessor is held in the reset condition as long as reset is applied.
9. **RESET OUT (output):** This signal is used by the microprocessor to reset its peripheral devices. It can be used as a system RESET. The signal is synchronized to the processor clock.
10. **CLK (output):** Clock output is used as a system clock. This output can be used to apply the processor clock frequency to the other devices presented in the system. The period of CLK is twice the X<sub>1</sub>, X<sub>2</sub> input period.

## 1.8 MICROPROCESSOR SYSTEM

Figure 1.12 shows the complete diagram of a microprocessor 8085 with its control signals and demultiplexed bus system.

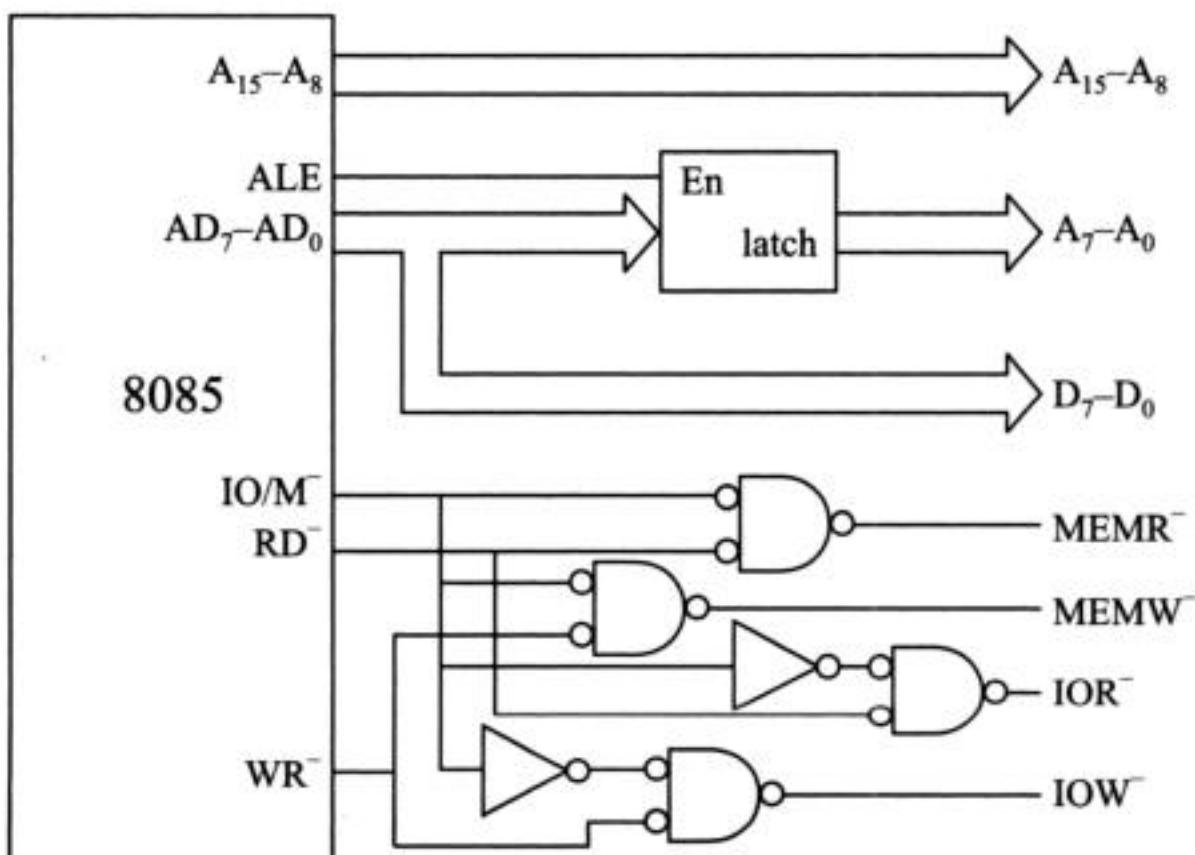


Figure 1.12 Microprocessor system.

## 1.9 HOW A PROGRAM IS EXECUTED

Intel Microprocessors work on the stored program concept, i.e. instructions or programs are stored in memory. To execute a program, microprocessor fetches the instructions one-by-one

and executes them in the same sequence. The flow of signal for the execution of a stored instruction is shown in Figure 1.13. Microprocessor performs the following steps to execute the stored program.

1. First the program counter loads the 16-bit memory address of the instruction on the address bus.
2. Then the control unit sends the  $\text{MEMR}^-$  signal to tell the memory that the microprocessor wants to read the memory and thereby memory enable the addressed memory location.
3. The addressed memory location then placed the 8-bit instruction code on the data bus and transferred this code to the instruction register (IR).
4. The opcode is then transferred to the instruction decoder and machine cycle encoder from instruction register. The instruction decoder and machine cycle encoder decoded the meaning of the opcode and the number of machine cycles required to execute the complete instruction.
5. The decoded opcode information is send to the control unit so that the control unit can generate the appropriate control signal to execute the instruction.

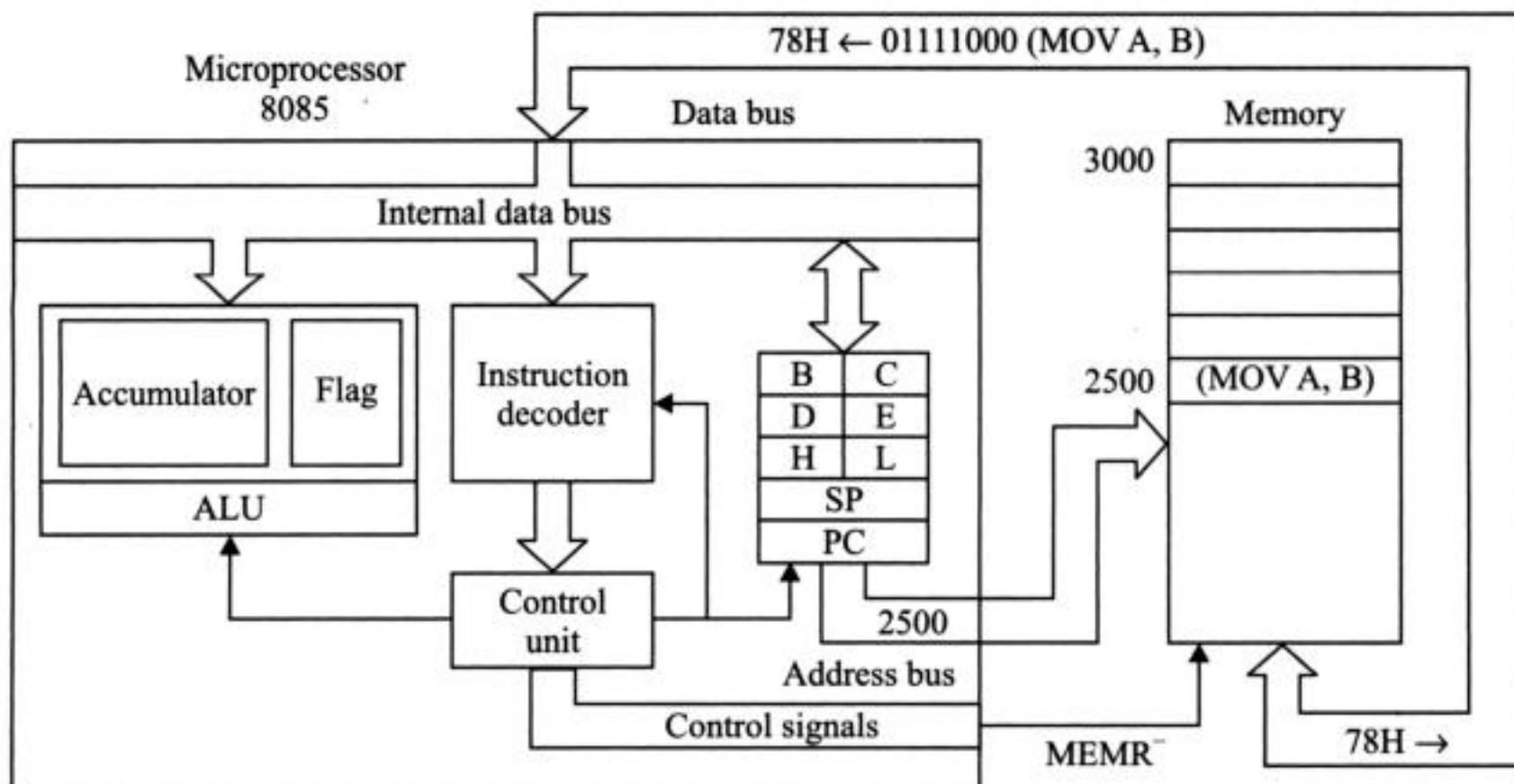


Figure 1.13 Flow of signal for the execution of a stored instruction.

## EXERCISES

### Multiple Choice Questions

1. Which generation of computer uses more than one microprocessor?
  - Second generation
  - Fifth generation
  - Third generation
  - None of the above.



14. The memory address register is used to store  
(a) Data to be transferred to memory  
(b) Data that has been transferred from memory  
(c) The address of a memory location  
(d) An instruction that has been transferred from memory.

15. A microprocessor is a processor with a reduced  
(a) Instruction set (b) Power requirement  
(c) MIPS performance (d) None of the above.

16. MPU stands for  
(a) Multi-Processing Unit (b) Micro-Processing Unit  
(c) Mega-Processing Unit (d) Major-Processing Unit.

17. Which of the following is not possible by a microprocessor?  
(a) Reading from memory (b) Writing into memory  
(c) Reading from input port (d) Writing into input port.

18. In which microprocessor does the concept of pipeline first introduced?  
(a) 8086 (b) 80286  
(c) 80386 (d) 80486.

19. A 32-bit microprocessor has the word length equal to  
(a) 1 byte (b) 2 byte  
(c) 4 byte (d) 8 byte.

20. If you wanted to find out whether an integer contained an even number of 1 bit, which status flag would be useful?  
(a) Carry (b) Overflow  
(c) Sign (d) Parity.

21. Within the CPU, all calculations and logic operations take place inside the .....  
(a) Registers (b) ALU  
(c) CU (d) MBU.

22. The three types of buses connected to the CPU are:  
(a) Data, address, control (b) Data, system, address  
(c) Address, control, memory (d) Fetch-decode, control, execution.

23. During which phase of the instruction execution cycle is the program counter incremented?  
(a) Decode (b) Execute  
(c) Operand fetch (d) Fetch.

24. The four parts of a CPU are:  
(a) Data bus, memory unit, control unit, arithmetic logic unit  
(b) Address bus, registers, control unit, arithmetic logic unit  
(c) Clock, memory unit, control unit, instruction fetch unit  
(d) Clock, registers, control unit, arithmetic logic unit.

25. Which flag is set when an unsigned value is too large to fit into a destination operand?  
(a) Sign (b) Carry  
(c) Overflow (d) Auxiliary carry.

26. What is Op-code?

- (a) The instruction that is to be executed
- (b) The value in which an operation acts upon
- (c) A mnemonic that defines a data size
- (d) The compiled assembly code.

27. Which of the following is not part of the processor?

- (a) The ALU
- (b) The CU
- (c) The registers
- (d) The system bus.

28. The memory address register is used to store

- (a) Data to be transferred to memory
- (b) Data that has been transferred from memory
- (c) The address of a memory location
- (d) An instruction that has been transferred from memory.

29. The memory data register is used to store

- (a) Data to be transferred to or from memory
- (b) Data to be transferred to the stack
- (c) The address of a memory location
- (d) An instruction that has been transferred from memory.

30. The instruction register stores

- (a) An instruction that has been decoded
- (b) An instruction that has been fetched from memory
- (c) An instruction that has been executed
- (d) The address of the next instruction to be executed.

31. TRAP is ..... interrupt

- (a) Synchronous
- (b) Asynchronous
- (c) Hardware
- (d) None of the above.

32. When the instruction RST4 is executed, the control jumps to location:

- (a) 0020
- (b) 0024
- (c) 0028
- (d) None of the above.

33. Which interrupt has the highest priority?

- (a) INTR
- (b) TRAP
- (c) RST6.5.

34. Name the 16-bit registers in 8085.

- (a) Stack pointer
- (b) Program counter
- (c) (a) and (b).

35. Which of the following is hardware interrupts?

- (a) RST5.5, RST6.5, RST7.5
- (b) INTR, TRAP
- (c) (a) and (b).

36. What is the RST for the TRAP?

- (a) RST5.5
- (b) RST4.5
- (c) RST4.

## **Descriptive Questions**

1. Explain the working principle of time-shared system and multi-programming system with example. Which is better and why?
  2. What is a microprocessor? What is the difference between a microprocessor and a CPU?

3. Define bit, byte, word and instruction.
4. Write a note on evolution of computers.
5. Discuss the comparison of the important characteristics of Intel 8-bit, 16-bit and 32-bit microprocessors.
6. Explain the various applications of microprocessor.
7. What do you mean by general purpose computers and single chip microcomputers?
8. Give a general block diagram of a microprocessor based system. Explain briefly the various blocks of the system. Give some examples of the types of devices used for each block.
9. What do you understand by a bus in a microprocessor system? Why is the data bus bidirectional? How many data lines are necessary in a 16-bit microprocessor, and what is the magnitude of the largest number that can be placed on its data bus? Why tri-state devices are required for a bus-oriented system?
10. Discuss the significance of flag register employed by different microprocessors.
11. Discuss the areas of application where microprocessor technology is in use.
12. Explain the difference between the following:
  - (a) Microprocessor and Microcomputer
  - (b) Assembler and Compiler
  - (c) High Level and Low Level Languages
13. Give the order of priority of the following signals with reason.
  - (a) HOLD
  - (b) RESET
  - (c) INTERRUPT.
14. What is stack memory? How is it implemented in an 8085 system? Discuss LIFO and FIFO type of operations in stack registers.
15. Explain the block diagram of 8085 in detail.
16. Discuss the functions of the address bus and direction of the information flow on the address bus.
17. Calculate the address lines for an 8 KB byte memory.
18. Why a 16-bit data is stored in memory in the reversed order: the low-order bytes first, followed by the high-order bytes?
19. Explain the functions of (1) HLDA, (2) READY, (3) SOD, (4) ALE, (5) RESET, and (6) HOLD pins of the 8085 microprocessor.
20. List the sequence of events that occurs when the 8085 microprocessor reads from the memory.
21. What are the various status flags provided in 8085? Discuss their roles.
22. Explain the requirement of a program counter, stack pointer and instruction register in the architecture of Intel 8085 microprocessor.



# 2

## Introduction to 8086

### 2.1 INTRODUCTION

Microprocessor 8086 is the next generation microprocessor. Architecturally it is totally different from its predecessor 8085 but functionally it is downward compatible with 8085. Microprocessor 8086 was introduced in 1978. Since its introduction, the architecture of 8086, the so-called X86 architecture has undergone five major evolutionary stages. The 8086, 8088, and 80186, 80188 are the members of the first generation of 80X86 family. The next generations are the 80286, followed by the 80386, and then the 80486. The Pentium is the fifth generation Intel microprocessor. Each generation built upon the basic concept of the previous with additional features and improved performance.

The 8086 was the first 80X86 families and is the basis for all Intel microprocessors that followed. It was a 16-bit microprocessor and significantly differ from the earlier 8-bit devices. It has 20 address lines and capable of addressing 1 Mbyte memory space. The various versions of the 8086 are operated on 5, 8, and 10 MHz clock frequencies.

In this chapter we discuss block and pin diagram of microprocessor 8086. The memory management system, logical and physical addresses advantages of segmented memory and even and odd memory concept is also covered in this chapter. Finally microprocessor 8088 and its differences with 8086 is discussed in this chapter.

### 2.2 THE 8086 MICROPROCESSOR

The block diagram of 8086 can be represented either as shown in Figure 2.1 or in Figure 2.2.

The 8086 consists of two main sections, the bus interface unit (BIU) and the execution unit (EU). These two units are independent of each other and behave as separate asynchronous operational processors. The EU contains the ALU, flags and general purpose registers. The EU carries out all the arithmetic and logical operations. All the registers in the 8086 are 16-bits wide, although 16-bit data registers can be used as two 8-bit data registers.

The BIU controls the address, data and control buses. The instruction fetching and queuing, operand fetch and store, and address relocation are the operations performed by the bus

interface unit. When the EU is decoding an instruction or executing instructions inside the microprocessor, the BIU prefetches instructions from memory and stores them in the instruction queue for faster processing. Up to 6 bytes of the instruction stream can be queued while waiting for decoding and execution. With the help of the queuing mechanism of instruction stream increases the efficiency of memory utility. Whenever there is room for at least 2 bytes in the queue, the BIU will fetch a word from memory and load it into the queue. Due to this queuing mechanism of instruction stream greatly reduces dead-time on the memory bus. The queue acts as a First-In-First-Out (FIFO) buffer between the BIU and the EU. The EU takes out instruction bytes as and when required. The first byte, into the queue, immediately goes to the EU when the queue is empty after the execution of a branch instruction. Also this is the only occasion when the processor has to wait for instruction (i.e. when the queue is flushed after a control transfer instruction). Otherwise, all the other times the execution unit receives pre-fetched instructions from the BIU queue.

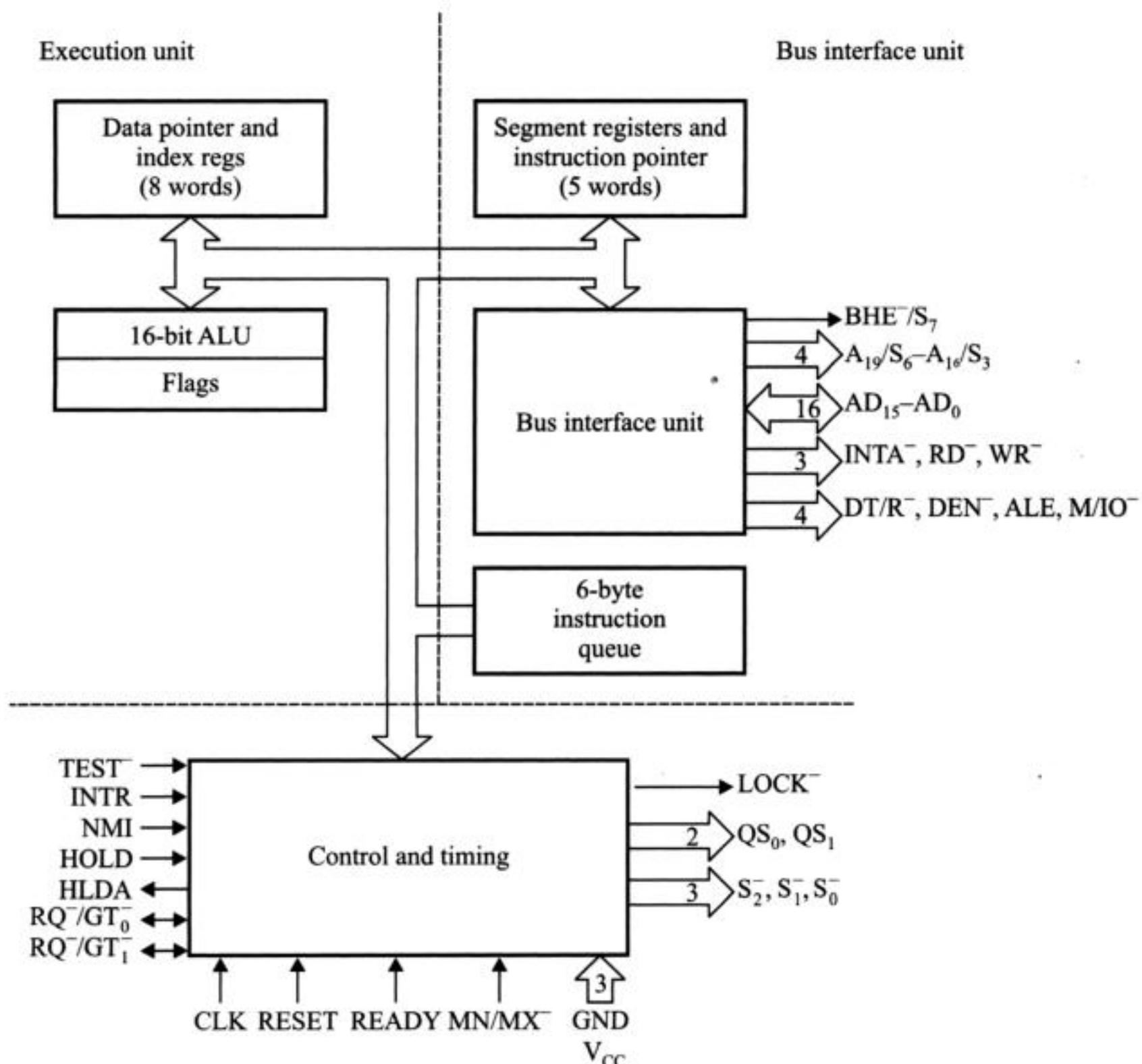


Figure 2.1 Block diagram of 8086.

## 2.2.1 Bus Interface Unit (BIU)

The Bus Interface Unit (BIU) consists of the following:

### Queue

Microprocessor 8086 consists of a FIFO (first in first out) registers set arranged like a pipe and called queue. The BIU continuously fetch operations from the memory while the processor is executing the current instruction. BIU unit stores the fetched bytes in the queue and the EU will read these bytes from the queue as and when it requires. The memory interface is usually much slower than the processor execution time, so this decouples the memory cycle time from the execution time.

### Segment registers

The memory of 8086 is of 1 MB which is divided into segments or we can say that the memory of 8086 is segmentized. Microprocessor 8086 can at a time access four segments. These segments are named Code Segment, Stack Segment, Data Segment and Extra Segment. Each segment is up to 64 K bytes long. Each segment is independent and separately addressable unit. Each segment is assigned a base address, which is its starting location in the memory space. All segments start on 16-bit memory boundaries. Segments may be adjacent, disjoint, partially overlapped, or fully overlapped.

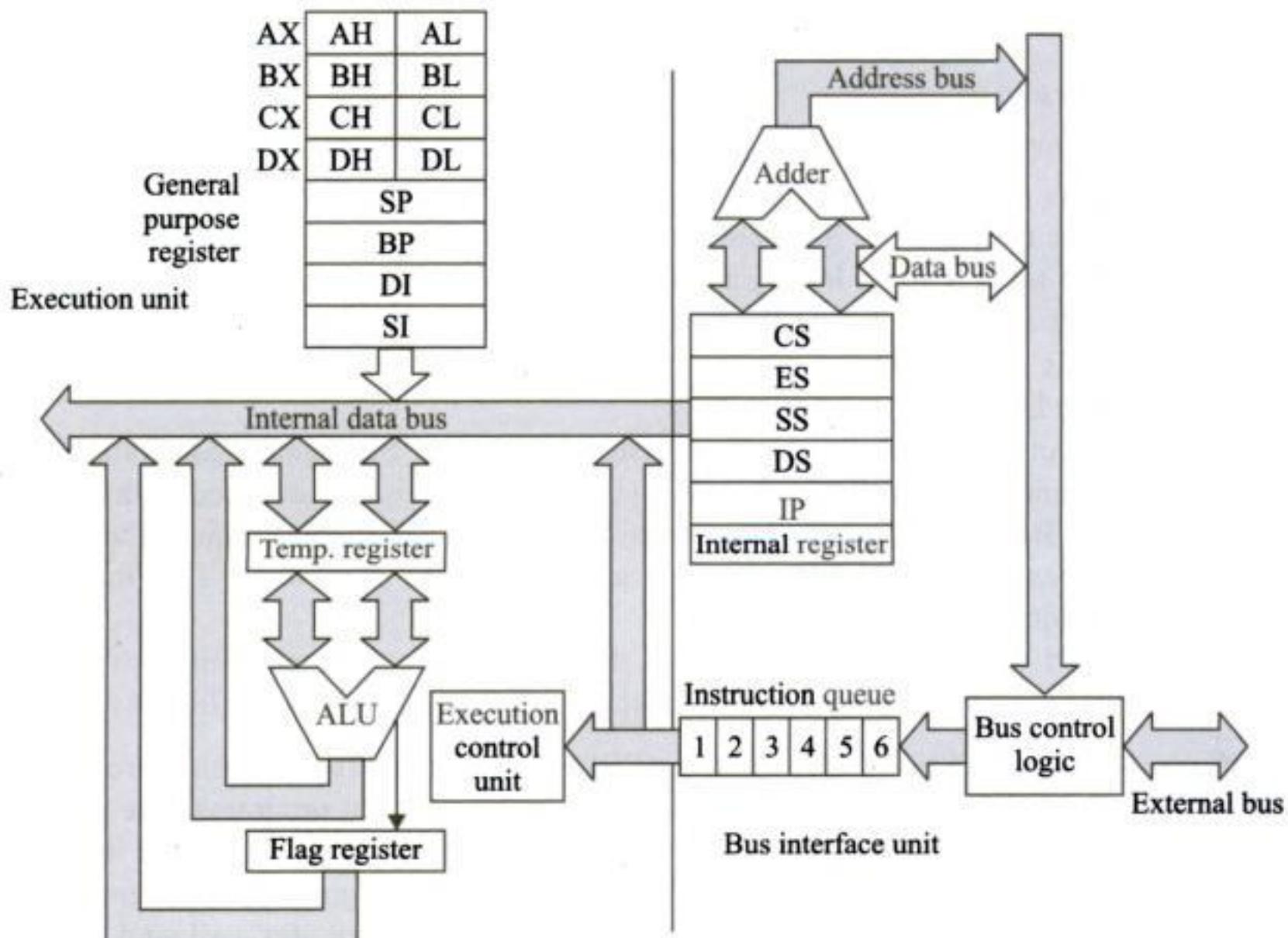
8086 consists of four 16-bit segment registers: the Code Segment (CS), Data Segment (DS), Stack Segment (SS) and Extra Segment (ES). These registers are used with the 16-bit Pointer, Index and Base registers to generate the 20-bit physical address required to allow the 8086/8088 to address 1 MB of memory. The segment registers point to the four immediately addressable segments.

The Segmented architecture was used in the 8086 to keep compatibility with earlier processors such as the 8085. It is one of the most significant elements of the Intel Architecture.

- (a) **Code segment (CS):** Code segment (CS) is a 16-bit register which stores the base address of 64 KB segment and microprocessor instructions or programs. The instruction pointer is the by default register used by the microprocessor to access the instructions from the CS. Like any other segment registers, the code segment (CS) register cannot be changed directly. During the execution of the far jump, far call and far return instructions, the CS register is automatically updated.
- (b) **Stack segment (SS):** Like Code segment, Stack segment (SS) is also a 16-bit register, containing the offset address of the 64 KB segment. This segment is used as a stack memory which operates on last in first out (LIFO). By default, the stack pointer (SP) and base pointer (BP) registers are the pointer registers. PUSH and POP are the main instructions to load and fetch a data from the stack segment (SS). This segment register (like other segment registers) cannot be initialized by loading the immediate value in the SS register. It can be changed directly using POP instruction.
- (c) **Data segment (DS):** The Data segment (DS) register is also a 16-bit register which holds the logical address of the 64 KB long data segment. The data segment is used to store the data. The by default registers of this segment are AX, BX, CX, DX and index register (SI, DI). This segment register (like other segment registers) cannot be

initialized by loading the immediate value in the DS register but can be changed directly using POP and LDS instructions.

- (d) **Extra segment (ES):** Similar to the other segment registers CS, SS, and DS, the Extra segment (ES) is also a 16-bit register which contains the starting address of 64 KB segment. The segment defined by ES register is used to store data. This segment, by default, is the destination location for the string data which are always pointed by the DI register. We cannot initialize the ES register by loading immediate value in it. It can be changed using POP and LES instructions.



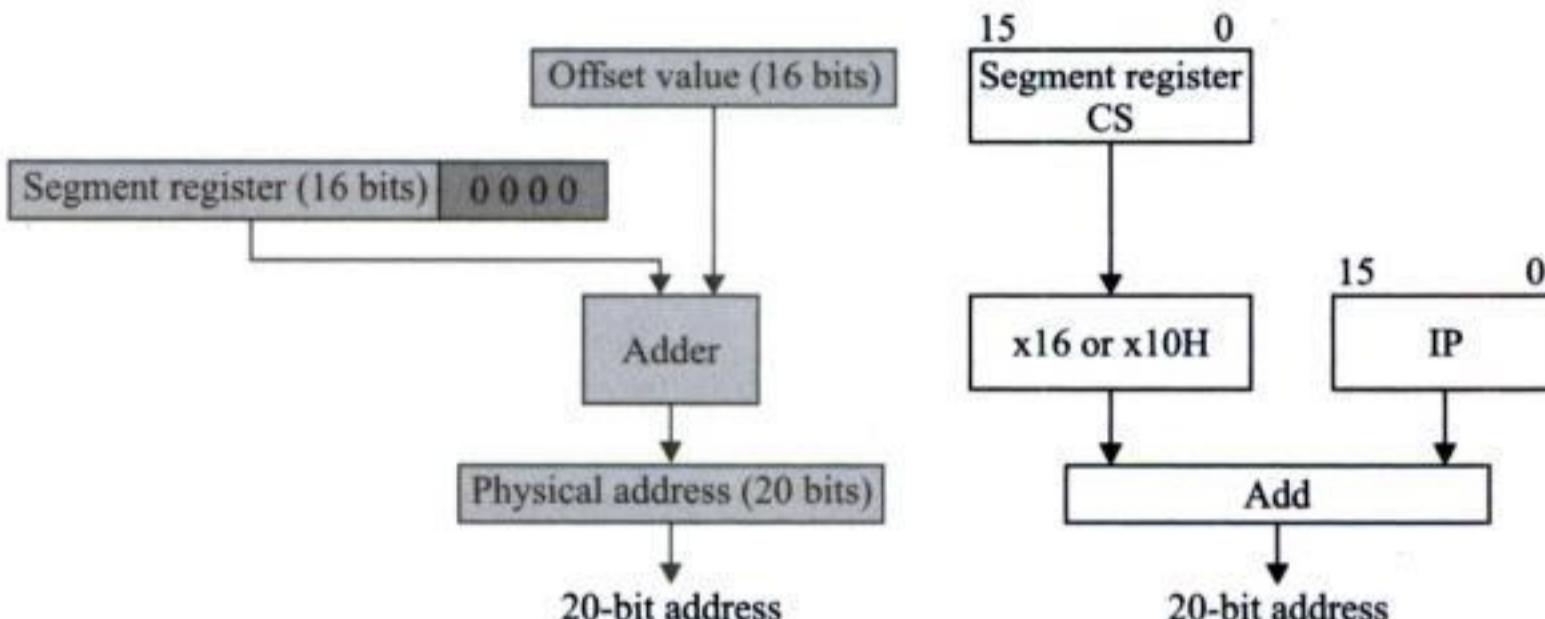
**Figure 2.2** Block diagram of 8086.

All the above stated segments have their own by default pointers but it is possible to change default segments (except IP) used by general and index registers by prefixing instructions with a CS, SS, DS or ES prefix followed by a colon.

#### **Instruction pointer (IP) and address summation**

The IP contains the offset or logical address of the next byte to be read from the code segment. In fact, it shows the distance of the current location, in bytes, from the base address given by the current Code Segment (CS) register. Figure 2.3 shows how this is done. The contents of the CS are shifted left by four. Bit 15 moves to the bit 19 position. The lowest four bits are filled with zeros or the CS register value is multiplied by decimal 16 or hexa decimal 10 H.

The resulting value is added to the Instruction Pointer contents to make up a 20-bit physical address. The CS makes up a segment base address and the IP is looked as an offset into this segment. This segmented model also applies to all the other general registers and segment registers in the 8086 device. For example, the SS and SP are combined in the same way to address the stack area in physical memory.



**Figure 2.3** Generation of 20-bit physical address.

### 2.2.2 Execution Unit

The execution unit consists of four 16-bit general purpose data registers which can be used as eight 8-bit data registers, four 16-bit pointers and base registers and one 16-bit flag register.

#### **General purpose data registers**

Microprocessor 8086 consists of four 16-bit data registers AX, BX, CX and DX. Each of these registers can be divided into two parts as higher and lower part to store 8-bit data. These are shown in Table 2.1.

**Table 2.1** Data Register of 8086

AX	AH	AL
BX	BH	BL
CX	CH	CL
DH	DH	DL

Apart from being general purpose registers, these registers also perform some specific functions. These functions are:

AX register is used as a 16-bit accumulator in 16-bit operations whereas AL is used as accumulator in 8-bit operations. These accumulator registers are used in division, multiplication, and shift and rotate instructions. All the IO and most of the string operations involve accumulator as one of the two operands.

The 16-bit base register (BX) is supposed to be made up of two 8-bit registers BL and BH. Apart from one 16-bit data register, BX register can be used as a memory pointer in data segment. It can be used for based, based indexed or register indirect addressing modes.

Count register (CX) or CL can be used as a counter in string manipulation and shift/rotate instructions. These registers are the default counter in loop instruction.

Data register (DX) is used in DIV instructions to hold higher word of the 32-bit operand and the remainder after division. It is used in multiplication operation to hold the higher word of the 32-bit result. When the IO address is of 16-bit than DX register is, by default, used to hold the 16-bit IO address.

**Pointers and base registers:** Microprocessor 8086 consists of four 16-bit pointers and base registers. These are SI, DI, SP and BP. These registers are used to hold offset or the logical addresses within a segment.

Stack Pointer (SP) is a 16-bit register pointing to program stack.

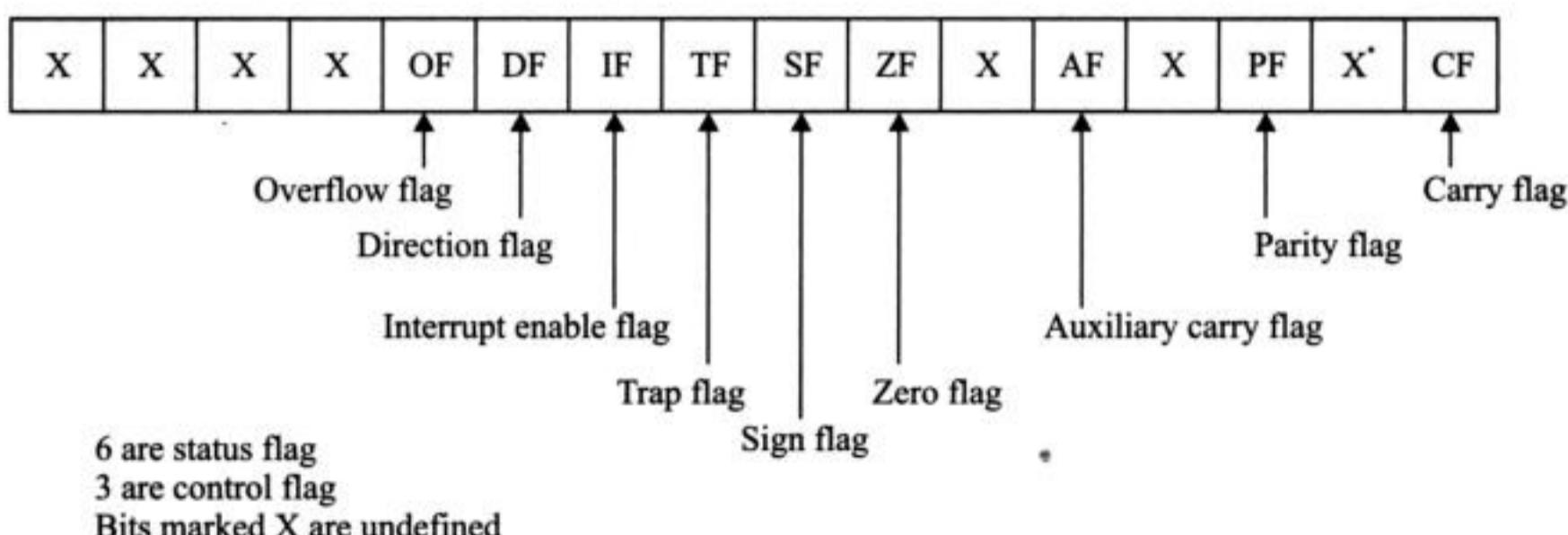
Base Pointer (BP) is a 16-bit register pointing to data in stack segment. BP register is usually used for based, based indexed or register indirect addressing.

**Source Index (SI)** is a 16-bit register. SI is used for indexed, based indexed and register indirect addressing, as well as a source data address in string manipulation instructions.

Destination Index (DI) is a 16-bit register. DI is used for indexed, based indexed and register indirect addressing, as well as a destination data address in string manipulation instructions.

### *Flag register*

Microprocessor 8086 consists of one 16-bit flag register. The flag register is a set of 16 independent flip-flops. Out of these 16 flip-flops, 6 flip-flops are used to indicate some data conditions and 3 flip-flops are used to control some machine control operations and the remaining 6 flip-flops are reserved for upcoming microprocessors. The format of the flag register is shown in Figure 2.4.



**Figure 2.4** Flag register format.

### (a) Status flags

- (i) Zero flag (Z)** The Zero flag indicates whether the result of a mathematical or logical operation is zero or non-zero. For a zero result this bit will be set, otherwise this bit will be clear.

- (ii) **Carry flag (CY)** This flag is set when an arithmetic carry or borrow has been generated out of the most significant bit position during an addition or subtraction operation. When used after an arithmetic operation it could be considered to be the unsigned equivalent of the overflow flag.
- (iii) **Sign flag (S)** This flag indicates whether the result of a mathematical operation is negative or positive. If the result is positive then this bit will be clear. Actually this bit represents the status of the D<sub>7</sub> bit of the last result.
- (iv) **Parity flag (P)** Parity flag indicates whether the number of set bits is odd or even in the binary representation of the result of the last operation. If the current result is of even parity, then this bit will be set. If the result is of odd parity, then this bit will be zero. This flag indicates whether the current result is of Even Parity (1) or of Odd Parity (0).
- (v) **Auxiliary carry flag (AC)** It shows carry propagation from D<sub>3</sub> position to D<sub>4</sub> position. This flag is used to convert the binary result into a BCD result. To understand it better, consider the example.

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	1	1	0	0
<hr/>							
0	0	1	0	1	0	1	1
1	0	1	1	0	1	1	1

- (vi) **Overflow flag** This flag is used in signed arithmetic operation. If the signed result is of more bits than the destination operand, then this flag will be set; otherwise it will be cleared. To understand better, consider the example.

Overflow →

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	1	0	0	1	1	0	0
<hr/>							
1	1	1	0	1	0	1	1
1	0	1	1	0	1	1	1

- (b) **Control flags** There are three control flags in 8086 which are used to control some operations of the processor. These flags are the Trap flag (TF), Interrupt enable flag (IF) and Direction flag (DF).

- (i) **Trap flag (TF)** If this flag is set, a single step interrupt occurs after the execution of the next instruction. This flag is used for single step debugging. This flag is set or cleared by software means.
- (ii) **Interrupt enable flag (IF)** This flag is used to mask or unmask the maskable interrupt. When this flag is set, maskable interrupts will cause the microprocessor to transfer its control to an interrupt vector specified location. This flag is set or cleared by software means.

- (iii) **Direction flag (DF)** This flag is used in string-related operations. It causes string instructions to auto decrement the appropriate index register (SI or DI) when set. If it is cleared, then the index registers will be in auto increment mode.

## 2.3 PIN CONFIGURATION OF 8086

The 8086 microprocessor is a 40 pin IC which operates on +5 V power supply and three clock rates: 5, 8, and 10 MHz. The 8086 operates in both single processor and multiple processor configurations to achieve high performance levels. During single processor mode, which is known as minimum mode, eight of its pins, from pin number 24 to 32 are having different definitions as that of multiple processor mode, known as maximum mode. The pin diagram of 8086 is shown in Figure 2.5.

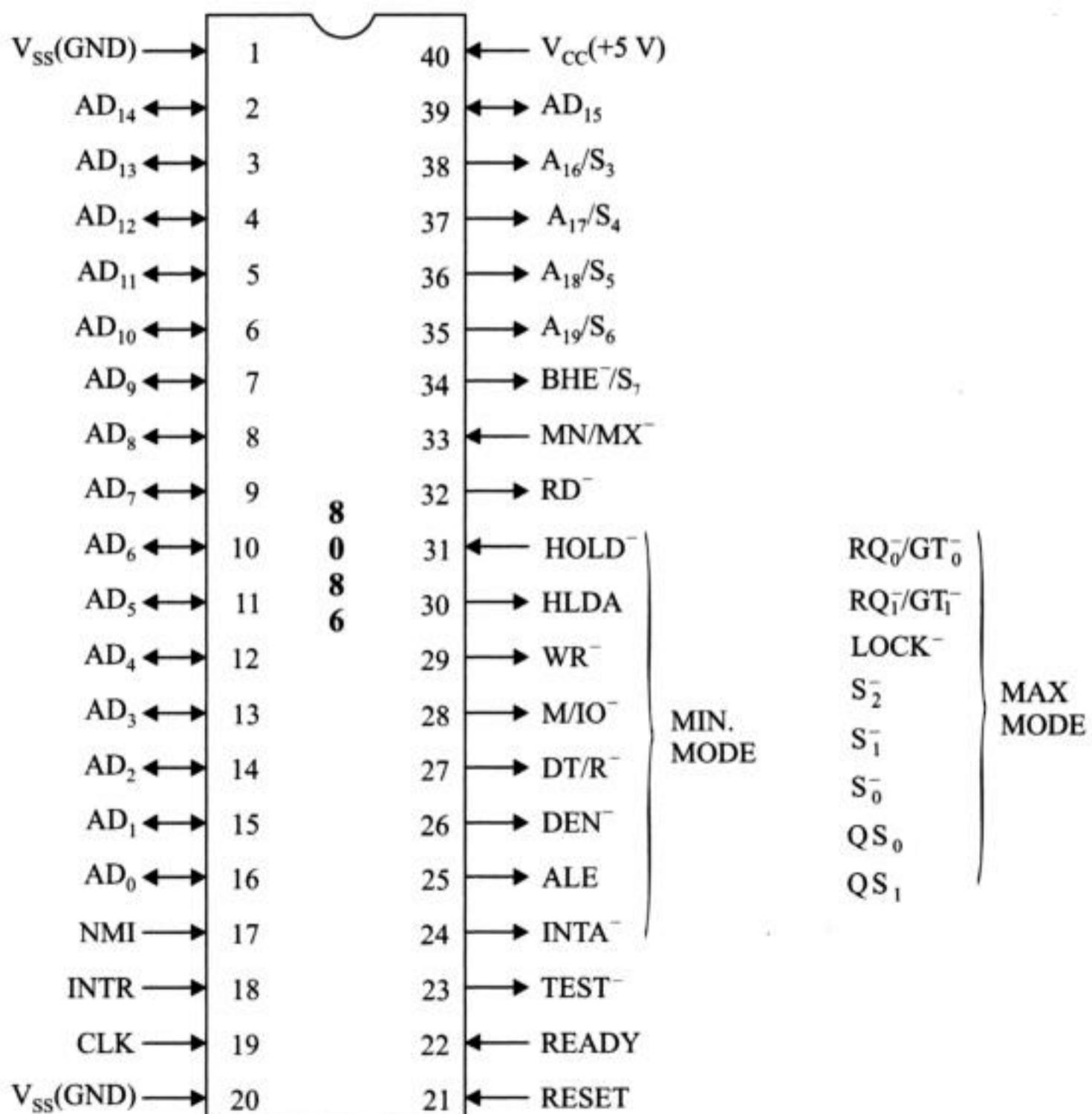


Figure 2.5 Pin configuration of 8086.

### 2.3.1 Pin Details of 8086—Common to Both Minimum and Maximum Mode

- AD<sub>0</sub>–AD<sub>15</sub> (address data bus)** These lines are multiplexed address/data lines. During the T<sub>1</sub> state of every machine cycle, these lines carry the address and for the rest of the T states (T<sub>2</sub>, T<sub>3</sub>, TW, T<sub>4</sub>), these lines carry the data. The A<sub>0</sub> line along with BHE<sup>-</sup> defines whether the microprocessor will access the lower byte or the higher byte or the word.
- A<sub>19</sub>/S<sub>6</sub>, A<sub>18</sub>/S<sub>5</sub>, A<sub>17</sub>/S<sub>4</sub>, A<sub>16</sub>/S<sub>3</sub> (address and status lines)** These are the upper four address lines of the microprocessor. These lines are multiplexed with the status signals S<sub>4</sub>, S<sub>5</sub>, S<sub>6</sub> and S<sub>7</sub>. During T<sub>1</sub> state, these acts as address lines for memory operations. During I/O operations, these lines are low. For the remaining T states, these lines show the status information. S<sub>4</sub> and S<sub>3</sub> give the information about the segment which is currently used by the processor. These lines are encoded as shown in Table 2.2.

**Table 2.2** Encoding of S<sub>4</sub> and S<sub>3</sub>

S <sub>4</sub>	S <sub>3</sub>	Segment in use
0	0	Alternate data (ES)
0	1	Stack (SS)
1	0	Code (CS) or none
1	1	Data (DS)

The status signal S<sub>5</sub> keeps the value of Interrupt Enable flag. The status of the interrupt enable flag bit (S<sub>5</sub>) is updated at the beginning of each CLK cycle. The S<sub>6</sub> status signal indicates whether 8086 is bus master or any other competent device is bus master. The S<sub>6</sub> is always low (logical) indicating 8086 is on the bus. If it is tristated, another bus master has taken control of the system bus.

- BHE<sup>-</sup>/S<sub>7</sub> (bus high enable/status)** The BHE<sup>-</sup> signal is used to enable the higher or the odd memory bank. During T<sub>1</sub> state of every machine cycle, the bus high enable signal (BHE<sup>-</sup>) is used to enable data onto the D<sub>15</sub>–D<sub>8</sub> data lines which are connected with the odd memory bank. The BHE<sup>-</sup> along with the A<sub>0</sub> address line is used to the even or the odd or both the banks. These two signals are encoded as per Table 2.3. During the rest of the T state (i.e. T<sub>2</sub>, T<sub>3</sub> and T<sub>4</sub>), this line sends the status signal S<sub>7</sub>. The status signal S<sub>7</sub> is used by 8087 numeric coprocessor to determine whether the CPU is an 8086 or 8088.

**Table 2.3** Encoding of BHE<sup>-</sup> and A<sub>0</sub>

BHE <sup>-</sup>	A <sub>0</sub>	Operation
0	0	Word (16-bit) will be access
0	1	Upper byte or odd byte will be access
1	0	Lower byte or even byte will be access
1	1	None

4. **RD<sup>-</sup> (read)** It is an active low output signal. It indicates that the microprocessor is performing a memory or I/O read operation. This signal is used to read devices which are connected to the 8086 local bus. RD<sup>-</sup> is activated during T<sub>2</sub>, T<sub>3</sub> and T<sub>W</sub> of any read machine cycle, and remain high in the other T states.
5. **READY** This signal is used to synchronize the slower peripherals. This is an active high input signal. When a peripheral device is ready to receive/transmit the data, it will send the READY signal to the microprocessor. On receiving this signal microprocessor, release the data and enter into the wait state till the next READY signal. The READY signal from memory/IO is synchronized by the 8284A Clock Generator to form READY.
6. **INTR (interrupt request)** It is an interrupt request signal. This is an active high level triggered input signal. INTR is sampled by the processor in the last clock cycle of each instruction. If microprocessor finds this signal as high, then the processor enter into an interrupt acknowledge operation. It can be internally masked by software resetting the interrupt enable bit.
7. **TEST<sup>-</sup>** This is an active low input signal. It is generally connected to the BUSY pin of the math coprocessor 8087. When microprocessor executes the ESCAPE instruction, it enters into the wait state. During the wait state, microprocessor polls the TEST<sup>-</sup> pin to check out up to when it has to be in the wait state. If the TEST<sup>-</sup> input is low, execution continues, otherwise the processor will remain in the wait state. This input is synchronized internally during each clock cycle on the leading edge of CLK.
8. **NMI (non-maskable interrupt)** NMI is the non maskable interrupt. This interrupt cannot be masked or denied. It is an edge triggered input which causes a type 2 interrupt. An interrupt service routine is vectored via an interrupt vector table located in the first 1 KB memory space from 00000H to 0003FFH.
9. **RESET** RESET causes the processor to immediately terminate its current operation. The signal must be active high for at least four clock cycles.
10. **CLK (clock)** The CLK is an input to the microprocessor and it provides the basic timing to the microprocessor and bus controller. This signal is generated by the 8284 clock generator.
11. **MN/MX<sup>-</sup> (minimum/maximum)** This signal indicates that in which mode the processor is to operate. If this pin is high, it means that the microprocessor will be in single processor mode and if it is low, then the microprocessor will be in multiprocessor operation mode.
12. **V<sub>CC</sub>** This pin is connected to the +5 V power supply.
13. **GND** It is the GROUND pin.

### 2.3.2 Pin Details of 8086—(Minimum Mode)

The following pin function descriptions are for the 8086 in minimum mode (i.e. MN/MX<sup>-</sup> = V<sub>CC</sub>).

1. **M/IO<sup>-</sup> (status line)** It is an output signal. It is used to distinguish whether microprocessor is going to access a memory or an IO. If M/IO<sup>-</sup> becomes zero, it means it will be an IO operation, otherwise a memory operation. M/IO<sup>-</sup> floats to 3-state OFF in local bus “hold acknowledge”.

2. **WR<sup>-</sup> (write)** It is an active low output signal. It indicates that the microprocessor is performing a write operation either from memory or I/O, depending on the status of the M/IO<sup>-</sup> signal. WR<sup>-</sup> is active for T<sub>2</sub>, T<sub>3</sub> and TW of any write machine cycle.
3. **INTA<sup>-</sup> (interrupt acknowledge)** It is an active low output signal. Microprocessor sends this signal in response to an interrupt request signal. It is active LOW during T<sub>2</sub>, T<sub>3</sub> and TW of each interrupt acknowledge cycle.
4. **ALE (address latch enable)** It is an active high output signal. It is provided by the processor to latch the address into the 8282/8283 address latch. It is a high pulse active during T<sub>1</sub> of any machine cycle. The T<sub>1</sub> state is sufficient enough to latch the address on the multiplexed AD<sub>0</sub>–AD<sub>15</sub> lines. Whenever the processor sends a valid address on the multiplexed lines it also makes the ALE high. ALE is never floated.
5. **DT/R<sup>-</sup> (data transmit/receive)** It is used to control the direction of data flow through the transceiver. If it is high, then data will be transmitted, otherwise data will come to the microprocessor. This signal is used only in minimum mode. It is used to select the direction of the transceiver 8286.
6. **DEN<sup>-</sup> (data enable)** It is an active low output signal. It is provided as an output enable for the 8286/8287 in a minimum system which uses the transceiver. DEN<sup>-</sup> is active during each memory and I/O access and for INTA<sup>-</sup> cycles. For a read or INTA<sup>-</sup> cycle, it is active from the middle of T<sub>2</sub> until the middle of T<sub>4</sub>, while for a write cycle, it is active from the beginning of T<sub>2</sub> until the middle of T<sub>4</sub>.
7. **HOLD (hold request)** It is an active high input signal. This pin is used by external devices (like DMA) to gain control of the buses. When the HOLD signal is activated by an external device, the microprocessor suspends current execution after the completion of the current machine cycle and stops using the buses. After releasing the buses, it sends the HLDA signal as an acknowledgement. This would allow external devices to control the buses. HOLD is not an asynchronous input. External synchronization should be provided if the system cannot otherwise guarantee the setup time.
8. **HLDA (hold acknowledge)** It indicates that the microprocessor has received the Hold request and that it will relinquish the buses in the next clock cycle. HLDA goes low after the Hold request is removed. The microprocessor takes the buses one-half clock cycles after HLDA, goes low. With the issuance of HLDA, the processor will float the local bus and control lines.

### 2.3.3 Pin Details of 8086 S—(Maximum Mode)

The following pin function descriptions are for the 8086/8288 system in maximum mode:

1. **S<sub>2</sub><sup>-</sup>, S<sub>1</sub><sup>-</sup>, S<sub>0</sub><sup>-</sup> (status)** These are three status output signals. These three signals are applied to the Bus Controller 8288 to generate the various control signals and the INTA signal in maximum mode. These status lines are encoded, as shown in Table 2.4, by the bus controller.

These status signals are active during T<sub>1</sub>, T<sub>2</sub>, and T<sub>4</sub> and are returned to the passive state (1, 1, 1) during T<sub>3</sub> or during T<sub>W</sub>. Any change by S<sub>2</sub>, S<sub>1</sub>, or S<sub>0</sub> during T<sub>4</sub> is used to indicate the beginning of a bus cycle, and return to the passive state in T<sub>3</sub> or T<sub>W</sub> is used to indicate the end of a bus cycle.

**Table 2.4** Encoding of  $S_2$ ,  $S_1$  and  $S_0$ 

$S_2$	$S_1$	$S_0$	Operation
0	0	0	Interrupt acknowledge
0	0	1	Read I/Q port
0	1	0	Write I/Q port
0	1	1	Halt
1	0	0	Code access
1	0	1	Read memory
1	1	0	Write memory
1	1	1	Passive

2. **RQ<sup>-</sup>/GT<sub>0</sub><sup>-</sup>, RQ<sup>-</sup>/GT<sub>1</sub><sup>-</sup> (I/O REQUEST/GANT)** These signals are the same as that of HOLD and HLDA in minimum configuration. These pins are used by other local bus masters to force the processor to release the local bus at the end of the processor's current machine cycle. Each pin is bidirectional with RQ/GT<sub>0</sub> having higher priority than RQ/GT<sub>1</sub>.

The request/grant sequence is as follows:

- The local bus master sends a pulse of 1T duration to the processor for the bus request.
- During a T<sub>1</sub> or T<sub>4</sub> clock cycle, the microprocessor 8086 send a pulse of 1T duration to the requesting master to indicate that the 8086 has allowed the local bus to float and it sends the hold acknowledgement signal. The microprocessor disconnects all its non-DMA devices from the local bus during hold acknowledge.
- In the last the requesting master sends a pulse of 1T duration to indicate to the 8086 that the hold request is about to end and that the 8086 can regain the local bus at the next CLK.

Each master-master exchange of the local bus is a sequence of 3 pulses. There must be one dead CLK cycle after each bus exchange. Pulses are active low.

If the request is made while the microprocessor is performing a memory cycle, it will release the local bus during T<sub>4</sub> of the cycle.

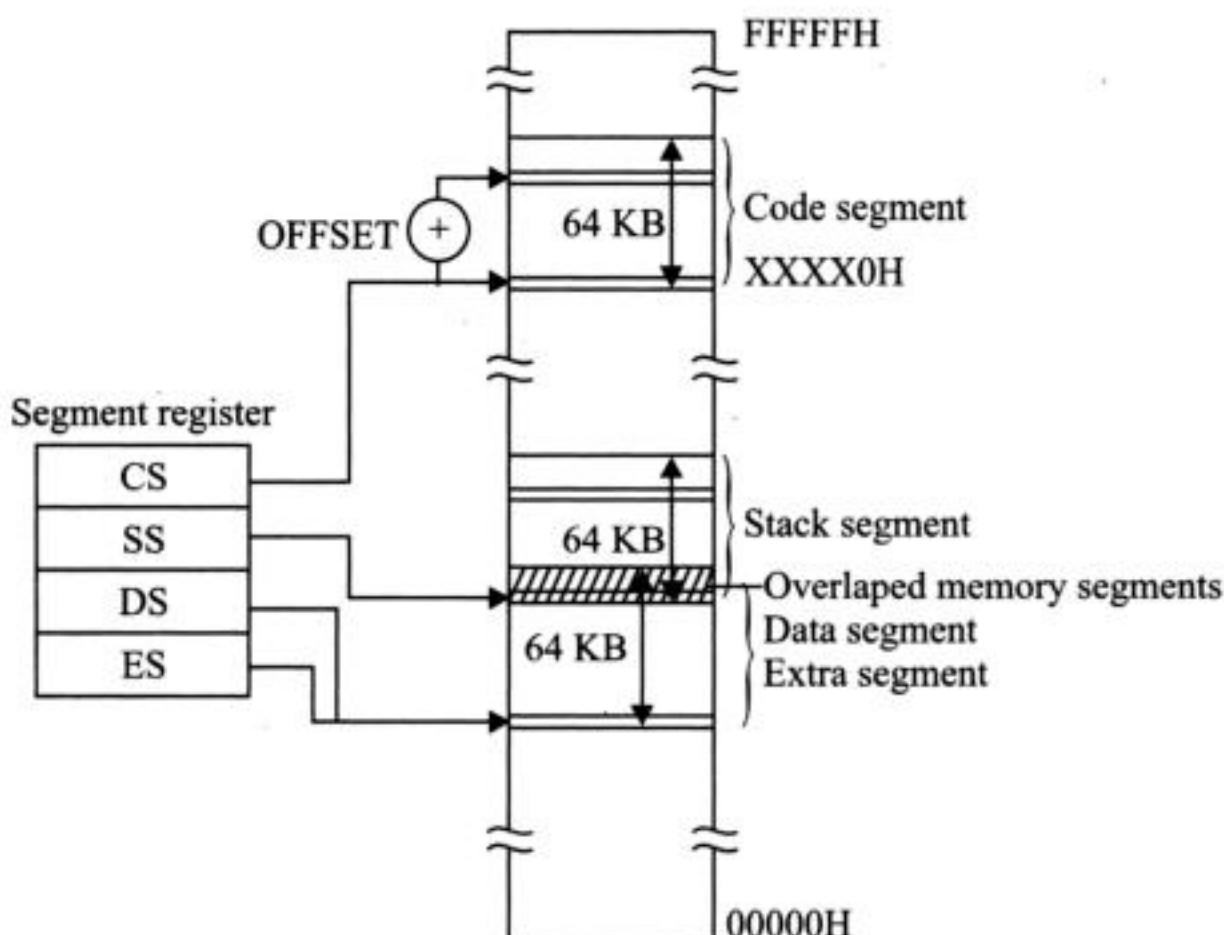
3. **LOCK<sup>-</sup>** It is an active low output signal. If this signal is active, then the other bus masters will not be allowed to take control over the system buses. The LOCK signal is activated by the LOCK instruction and remains active until the completion of the next instruction. The LOCK instruction is actually prefix to an instruction.
4. **QS<sub>1</sub>, QS<sub>0</sub> (queue status)** QS<sub>1</sub> and QS<sub>0</sub> provide status to allow external tracking of the internal 8086 instruction queue. These status signals are interfaced with the status signals (of the same name) of the math coprocessor 8087. By these two status signals, the math coprocessor tracks the queue of 8086. The queue status signals QS<sub>1</sub> and QS<sub>0</sub> are encoded in Table 2.5.

**Table 2.5** Encoding of  $QS_1$  and  $QS_0$ 

<b><math>QS_1</math></b>	<b><math>QS_0</math></b>	<b>Characteristics</b>
0	0	No operation
0	1	First byte of op-code from queue
1	0	Empty the queue
1	1	Subsequent byte from queue

## 2.4 MEMORY ORGANIZATION OF 8086

The memory is logically divided into code, data, extra and stack segments each of 64 KB. These four segments can partially or fully overlap with each other. These four segments are shown in Figure 2.6.

**Figure 2.6** The segmented memory.

Here in this figure the code segment is separately defined, whereas the stack segment is partially overlapped with the data and extra segment. The data and the extra segments are fully overlapped with each other. This can be done by loading the same base value in the DS and ES registers. It is also shown in the figure that the LSB nibble of the starting address of each segment is always 0.

All memory references are made relative to base addresses contained in segment registers. The segment types were chosen based on the addressing needs of programs. The segment register to be selected is automatically chosen according to the specific rules of Table 2.2. All information in one segment type shares the same logical attributes (e.g. code or data). That means the code segment always contains the programs or the instructions whereas the stack segment is used to store the data or the addresses during the execution of the PUSH, POP,

CALL or the interrupt operations. The data segment and the extra segments are used to store the data. For string operations the DS contains the source strings whereas the ES contains the destination strings. Certain locations in memory are reserved for specific microprocessor operations. Table 2.6 shows these reserved locations.

**Table 2.6** Assignment of segment registers

Type of memory references	Default segment base	Alternate segment base	Offset
Instruction fetch	CS	None	IP
Stack operation	SS	None	SP
Variable (except following)	DS	CS, ES, SS	Effective address
String source	DS	CS, ES, SS	SI
String destination	ES	None	DI
BP used as base register	SS	CS, DS, ES	Effective address

The segmentation of memory may be confusing at the first glance but it has certain advantages. These advantages are:

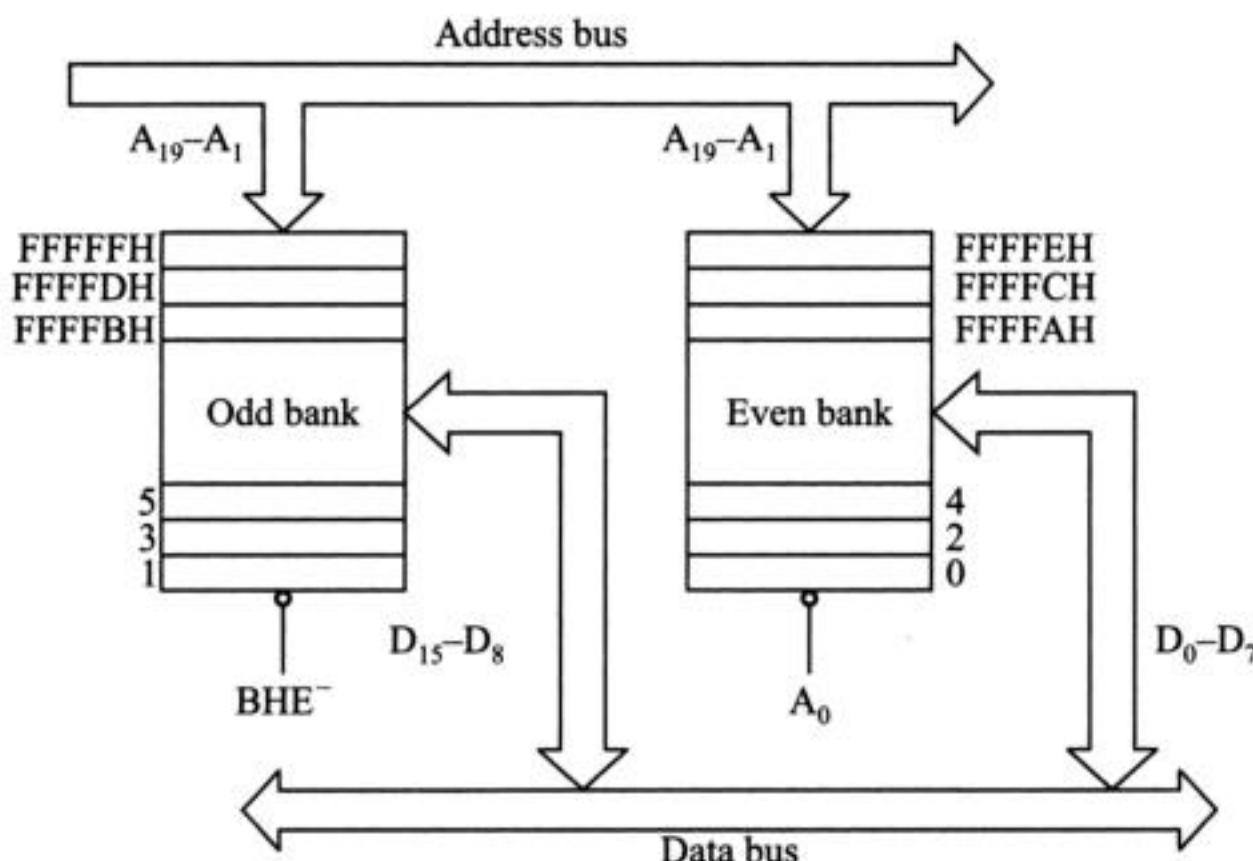
- (a) We can address 1 MB memory with only 16-bit registers.
- (b) Allow the instruction, stack or data part of a program to be more than 64 KB long by using more than one code, stack and data segment.
- (c) We will have separate area for code, stack and data information.
- (d) We can relocate a program by simply changing the content of the code segment register.
- (e) Program can work on several data sets by reloading the DS register.

The 8086 microprocessor provides a 20-bit address to memory. The memory is organized as a linear array of up to 1 MB, addressed from 00000H to FFFFFH. Though microprocessor 8086 is a 16-bit microprocessor, still its memory is of 8-bit wide. This is because of two reasons:

- (a) Most of our IO devices are of 8-bit wide.
- (b) There are many instructions which are of one byte. The other instructions are from two to seven bytes. So by being able to access individual bytes, these odd lengthed instructions can be handled.

Physically, the memory is organized as an odd bank ( $D_{15}-D_8$ ) of 512 KB and an even bank ( $D_7-D_0$ ) of 512 KB addressed in parallel by the processor's address lines. The odd memory bank is the memory having odd addresses, similarly the even bank memory is the memory having the even addresses. Data in even bank is transferred on the  $D_7-D_0$  bus lines; whereas data to the odd bank is transferred on the  $D_{15}-D_8$  bus lines. The processor provides two enable signals,  $BHE^-$  and  $A_0$ . These two signals selectively allow reading from or writing into either an odd byte location, even byte location, or both.  $BHE^-$  is used to enable the odd bank and the  $A_0$  address line is used to enable the even bank. Table 2.3 shows how the  $BHE^-$  and  $A_0$  select the odd byte or even byte or a word.

To access a Byte of data in even bank (also known as Low bank), valid address is provided via address pins  $A_1$  to  $A_{19}$  together with  $A_0 = '0'$  and  $BHE^- = '1'$ . Similarly to access a byte of data in odd bank (also known as High bank), valid address in pins  $A_1$  to  $A_{19}$ ,  $A_0 = '1'$  and  $BHE^- = '0'$  are required to access the data through  $D_8$  to  $D_{15}$  of the data bus. These signals disable the Low bank and enable the High bank to transfer (In/Out) data through  $D_8$  to  $D_{15}$  of the data bus.



**Figure 2.7** Memory banks.

For odd-addressed words two bus-cycles are required to access the Word-data. During the first bus-cycle, odd addressed LSB of the word is accessed from the High-memory. During the second bus-cycle, IP is auto-incremented to access the even address MSB of the word from the Low bank. For even-addressed words, only one bus-cycle is needed to access the word, as both low and high banks are activated at the same time using  $A_0 = 0$  and  $BHE^- = 0$ . During this bus-cycle, all 16-bit data is transferred via  $D_0$  to  $D_{15}$  of the data bus.

## 2.5 MICROPROCESSOR 8088

Intel 8088 microprocessor was released in 1979, one year after the Intel 8086 microprocessor. The 8088 MPU is also a 16-bit processor designed around the 8086 internal structure. Most internal functions of the 8088 are identical to the equivalent 8086 functions. The 8088 handles only 8-bits at a time because it has 8-bit external data bus. 16-bit operands are fetched or written in two consecutive bus cycles which makes the 8088 processor to run slower. On the plus side hardware changes to the 8088 microprocessor made it compatible with 8080/8085 support chips. The internal register structure of 8088 is identical to 8086.

Like 8086, the 8088 microprocessor also has 16-bit registers, 16-bit internal data bus and 20-bit address bus. The memory space of 8088 processor is of 1 MB. The memory of 8088

is also segmented into 64 K segments and it can use four segments (CS, SS, DS and ES) at a time. Unlike the memory of 8086, the memory of 8088 is not divided in even and odd memory banks. As there are no memory banks the BHE<sup>-</sup> signal is not required in 8088.

Intel 8087 numeric coprocessor is also used as a math coprocessor by 8088, like that of 8086. The 8088 processor (Floating-Point) recognize all 8087 instructions, and, when necessary, it calculates memory address for FP instruction operand.

The major differences between 8088 and 8086 are outlined below:

- I. The queue length is 4 bytes in the 8088, whereas the 8086 queue comprises 6 bytes.
- II. The 8088 BIU will fetch a new instruction to load into the queue as soon as it finds a byte space available in the queue. The 8086 waits until a 2 byte space is available.
- III. The 8088 has 8-bit external data bus.
- IV. BHE<sup>-</sup> signal is replaced by SS<sub>0</sub> status signal.
- V. All 16-bit fetches and writes from/to memory take an additional four clock cycles (or one additional machine cycle).

The hardware interface of the 8088 has some major differences as compared to the 8086. The pin assignments are nearly identical, however, with the following functional changes.

1. A<sub>8</sub>-A<sub>15</sub>: These pins are only address outputs on the 8088. These address lines are latched internally and remain valid throughout a bus cycle in a manner similar to the 8085 upper address lines.
2. SS<sub>0</sub><sup>-</sup> provides the S<sub>0</sub><sup>-</sup> status information in the minimum mode. This output occurs on pin 34 in minimum mode only. DT/R<sup>-</sup>, IO/M<sup>-</sup> and SS<sub>0</sub><sup>-</sup> provide the complete bus status in minimum mode. This is shown in Table 2.7.
3. BHE<sup>-</sup> has no meaning on the 8088 and has been eliminated.
4. IO/M<sup>-</sup> has been inverted, i.e. (in 8086, this pin is IO<sup>-</sup>/M).
5. ALE is delayed by one clock cycle in the minimum mode when entering HALT to allow the status to be latched with ALE.

**Table 2.7** Status information of 8088 in minimum mode

<b>IO/M<sup>-</sup></b>	<b>DT/R<sup>-</sup></b>	<b>SS<sub>0</sub><sup>-</sup></b>	<b>Characteristics</b>
0	0	0	Code access
0	0	1	Read memory
0	1	0	Write memory
0	1	1	Passive
1	0	0	Interrupt acknowledge
1	0	1	Read I/O port
1	1	0	Write I/O port
1	1	1	Halt

**EXERCISES****Multiple Choice Questions**

1. In 8086 microprocessor, which one of the following statements is not true?
  - (a) Coprocessor is interfaced in MAX mode.
  - (b) Coprocessor is interfaced in MIN mode.
  - (c) I/O can be interfaced in MAX/MIN mode.
  - (d) Supports pipelining.
2. The 8086/8088 use a multiplexed address and data bus because
  - (a) 40 pins is a good size for the IC.
  - (b) Multiplexing is supported by 8085 Microprocessor.
  - (c) Multiplexing reduces the number of lines between the processor and the auxiliary ICs
  - (d) All of the above.
3. The word size of an 8086 processor is
  - (a) 8 bits
  - (b) 16 bits
  - (c) 32 bits
  - (d) 64 bits.
4. The data bus size of an 8088 processor is
  - (a) 8 bits
  - (b) 16 bits
  - (c) 32 bits
  - (d) 64 bits.
5. What is the size of instruction queue in 8086?
  - (a) 2 bytes
  - (b) 4 bytes
  - (c) 6 bytes
  - (d) 8 bytes.
6. What is the size of instruction queue in 8088?
  - (a) 2 bytes
  - (b) 4 bytes
  - (c) 6 bytes
  - (d) 8 bytes.
7. Which is faster: Reading word size data whose starting address is at
  - (a) Even address
  - (b) Odd address
  - (c) Memory address don't matter.
8. The SP register is typically used for accessing
  - (a) Strings
  - (b) Memory
  - (c) Stack
  - (d) Data segment.
9. The read/write line
  - (a) Belongs to the data bus
  - (b) Belongs to the control bus
  - (c) Belongs to the address bus
  - (d) CPU bus.
10. How many 16-bit registers are available in 8086?
  - (a) 8
  - (b) 12
  - (c) 14
  - (d) 16.
11. Which flags of 8086 are not present in 8085?
  - (a) OF
  - (b) DF
  - (c) TF
  - (d) All of the above.





- (a) Trap (b) Direction flag  
(c) Interrupt enable (d) Overflow flag.

38. The flag which indicates whether a string will be copied from source to destination starting from top of the string (lower order address) to bottom of the string or vice versa is .....  
(a) Trap (b) Direction flag  
(c) Interrupt enable (d) Overflow flag.

39. The flag which indicates whether the result of an arithmetic operation has exceeded the maximum number that can be represented in a given word size or not is .....  
(a) Trap (b) Direction flag  
(c) Interrupt enable (d) Overflow flag.

40. The starting address of memory containing program is stored in ..... register.  
(a) Code segment (b) Data segment  
(c) Index register (d) Pointer register.

41. The least significant four bits of a 16-bit number to be moved into a segment register should be  
(a) 0000 (b) 1111  
(c) can be any four-bit number (d) None.

42. The starting address of memory containing data is stored in ..... register.  
(a) Code segment (b) Data segment  
(c) Index (d) Pointer register.

43. The starting address of memory used for storing and retrieving data quickly, using PUSH and POP instructions are stored in ... register.  
(a) Code segment (b) Data segment  
(c) Stack segment (d) Pointer register.

44. The starting address of memory used for storing and retrieving a string of data quickly, is stored in ... register.  
(a) Code segment (b) Data segment  
(c) Stack segment (d) Extra segment.

45. The number of 8-bit general purpose registers of 8086 is  
(a) 8 (b) 4  
(c) 5 (d) 6.

46. The ... byte of status register of 8086 contains the status flags as that of 8085.  
(a) Lower (b) Higher  
(c) Both (d) None.

47. The input of 8086 which determines whether 8086 is used for multiprocessing application is  
(a) MN/MX (b) LOCK  
(c) BUSY (d) None.

48. The signal which is used to latch the address in the multiplexed addresss data bus of 8086 in minimum mode is  
(a) ALE (b) READY  
(c) TEST (d) BHE.



## **Descriptive Questions**

1. Draw the block diagram of 8086 and explain the functions of GPRs.
  2. With a neat diagram explain the architecture of 8086 microprocessor along with function of each block and register.
  3. How many address lines does an 8086 have? How many memory addresses does this number of address lines allow the 8086 to access directly?
  4. At any given time, the 8086 works with 4 segments in its address space. How many bytes are contained in each segment?
  5. Draw the programmer's model of 8086/8088 register set.
  6. What is the length of the instruction queue in 8086? Discuss the use of the queue. Explain the reason for limiting the length of queue.
  7. What is the minimum number of segment registers that are necessary to provide segmentation? How do access common data for different programs using segmentation?
  8. Discuss the function of segment registers of 8086 with examples.
  9. Explain various parts of BIU in 8086.
  10. Explain what are the advantages of the memory segmentation. Discuss various segment registers in 8086.
  11. Explain the physical memory organization in 8086. How is it differing from 8088?
  12. What is pipelining? How is it achieved in 8086? What are the advantages?
  13. Explain the functions of different registers in 8086. Explain with examples, the various flags of 8086 and their conditions in various instances.

14. List out segmentation registers of 8086. Explain how 8086 provides 1 MB memory address space using the segment registers? What is the purpose of extra segment?
15. Which are the default segment bases: offset pairs?
16. With a neat diagram explain the pin configuration of 8086.
17. Describe the function of the following pins and their use in 8086 based system.
  - (a) NMI
  - (b) LOCK
  - (c) TEST
  - (d) RESET.
18. What are the two modes of 8086? List out various signals generated by the CPU in these two modes respectively.
19. Explain in detail about memory access mechanism in 8086.
20. Explain the functions of the following 8086 signals.
  - (a) HLDA
  - (b)  $RQ_0^-/GT_0^-$
  - (c)  $DEN^-$
  - (d) ALE.
21. In addition to the function of a general purpose register, what other functions are performed by the register BX, BP and CX? Explain why the PTR attribute operator is sometimes necessary.
22. What  $S_7$ ,  $S_6$  and  $S_5$  status signals show?
23. Explain the 8086 conditional flags.
24. What is the function of TF flag of 8086?
25. Why the memory of 8086 is divided into even and odd memory banks? How these memory banks are selected for byte and word access?
26. What is the difference between 8086 and 8088 microprocessors?
27. Explain the following signals of 8086
  - (a)  $RQ_1^-/GT_1^-$
  - (b) MN/MX $^-$
  - (c) READY
  - (d) INTR
  - (e) DT/R $^-$
28. How does 8086 convert a logical address to physical address? Explain with an example.
29. Explain with the suitable diagram how 8086 access a byte or word from EVEN and ODD memory banks.

# 3

## 8086 Based Minimum/ Maximum Systems

### 3.1 INTRODUCTION

Microprocessor 8086 can work in two modes, i.e. minimum mode and the maximum mode. In these two modes the working environment of the processor changed. The minimum mode is intended for small- and medium-sized systems employing only a single processor whereas the maximum mode is designed for larger systems and employs more than one processor. Some of the pins of 8086 (pin number 24 to 32) have different definitions in the two modes. The mode of the processor is determined by the MN/MX<sup>-</sup> pin of the processor.

In this chapter we will discuss the minimum and maximum mode configurations of 8086 processor, demultiplexing of the multiplexed buses, generation of control signals and logic diagram of transceivers. This chapter also covers the block and pin diagrams of clock generator and bus controller. The concept of memory interfacing along with the timing diagram of 8086 processor in minimum and maximum modes is covered in this chapter.

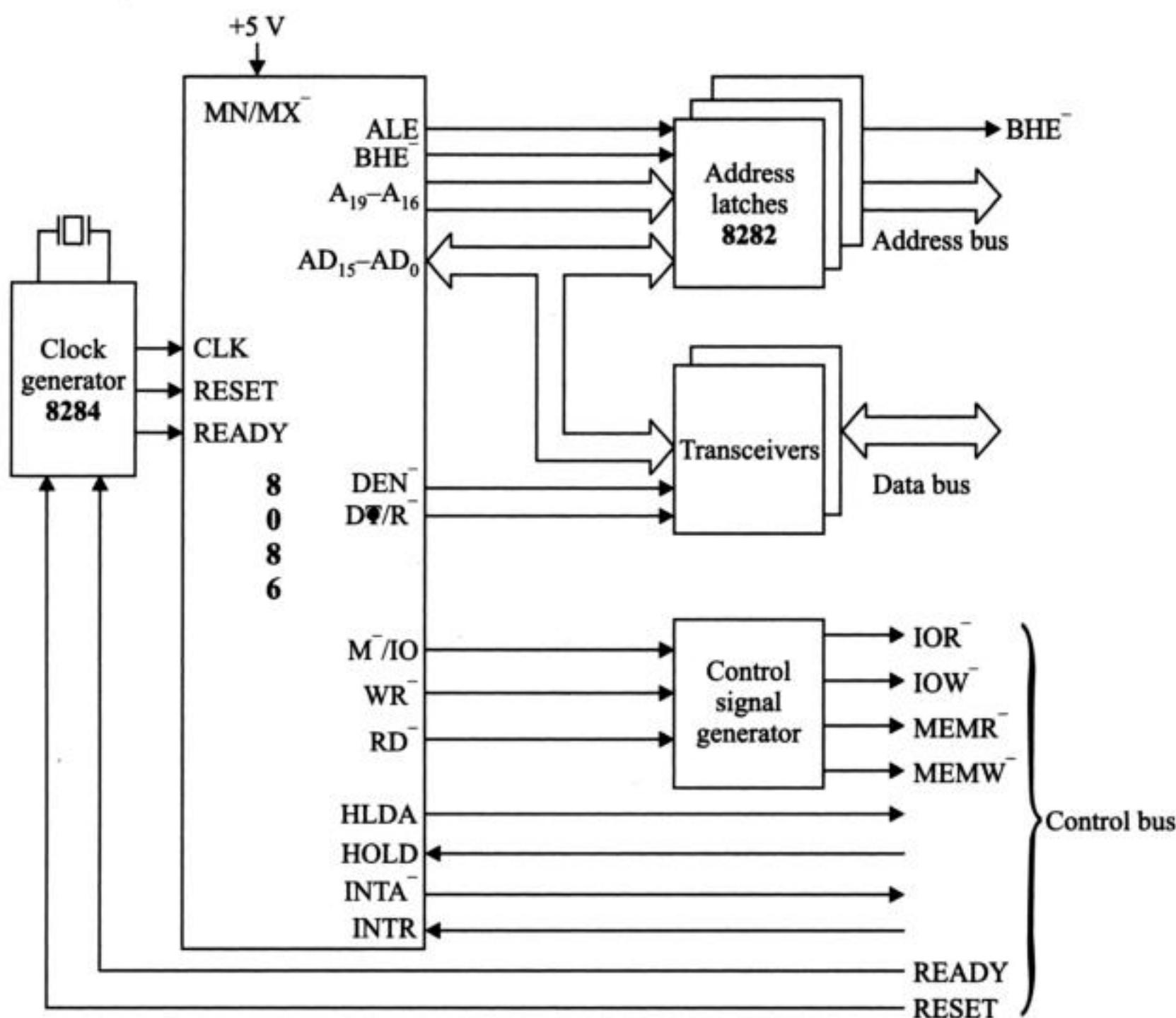
### 3.2 8086 MINIMUM MODE CONFIGURATION

When the MN/MX<sup>-</sup> pin of the 8086 is wired to a +5 volt power supply, the processor operates in minimum mode. This mode is intended for small- and medium-sized systems employing only a single processor. All the control signals are generated by the 8086 processor itself. Figure 3.1 shows such a minimum mode configurations system of 8086.

The minimum mode configuration of 8086 consists of:

1. Three 8-bit latches (IC 8282)
2. Two 8-bit transceivers (IC 8286)
3. One control signal generator
4. One clock generator (8284).

The latches are D-type flip-flops whose outputs are generally buffered. The common examples of latches are 8282 and 74LS373. These latches are used to demultiplexed the



**Figure 3.1** Minimum mode configuration.

multiplexed lines, i.e.  $AD_0-AD_{15}$ ,  $A_{16}/S_3-A_{19}/S_6$  and  $BHE^-/S_7$ . The ALE signal controls the D-FF in the latches.

The transceivers are used to separate the data bus from the multiplexed address/data bus. This chip consists of bidirectional buffers. The control signal generator is used to generate the four control signals, i.e.  $IOR^-$ ,  $IOW^-$ ,  $MEMR^-$  and  $MEMW^-$ . The clock generator is responsible for the generation of the required clock frequency and to synchronize the READY and RESET signals.

### 3.2.1 Demultiplexing of the Multiplexed Buses

In 8086 the address lines are not dedicated to carry the address but are time multiplexed with data bus and the status signals. This multiplexing is done to keep the number of pin as minimum as possible. These lines are demultiplexed to generate the independent address, status as well as data bus.

During  $T_1$  state of every machine cycle the multiplexed lines carry the address part. In this  $T_1$  state, ALE signal is also high to indicate that these lines carry a valid address. After  $T_1$

state, the microprocessor 8086 will remove the address contents from these lines and use these lines as data lines and status signals for next three clock cycles. The demultiplexing of these lines is done by using three 8-bit D-latch IC along with the ALE signal as shown in Figure 3.2. When ALE is high it will trigger the D-FF's through the STB<sup>-</sup> terminal of the latch. The address signals will get latched in the three 8-bit latches and the output of latch will provide A<sub>0</sub>–A<sub>19</sub> address contents and also BHE<sup>-</sup>.

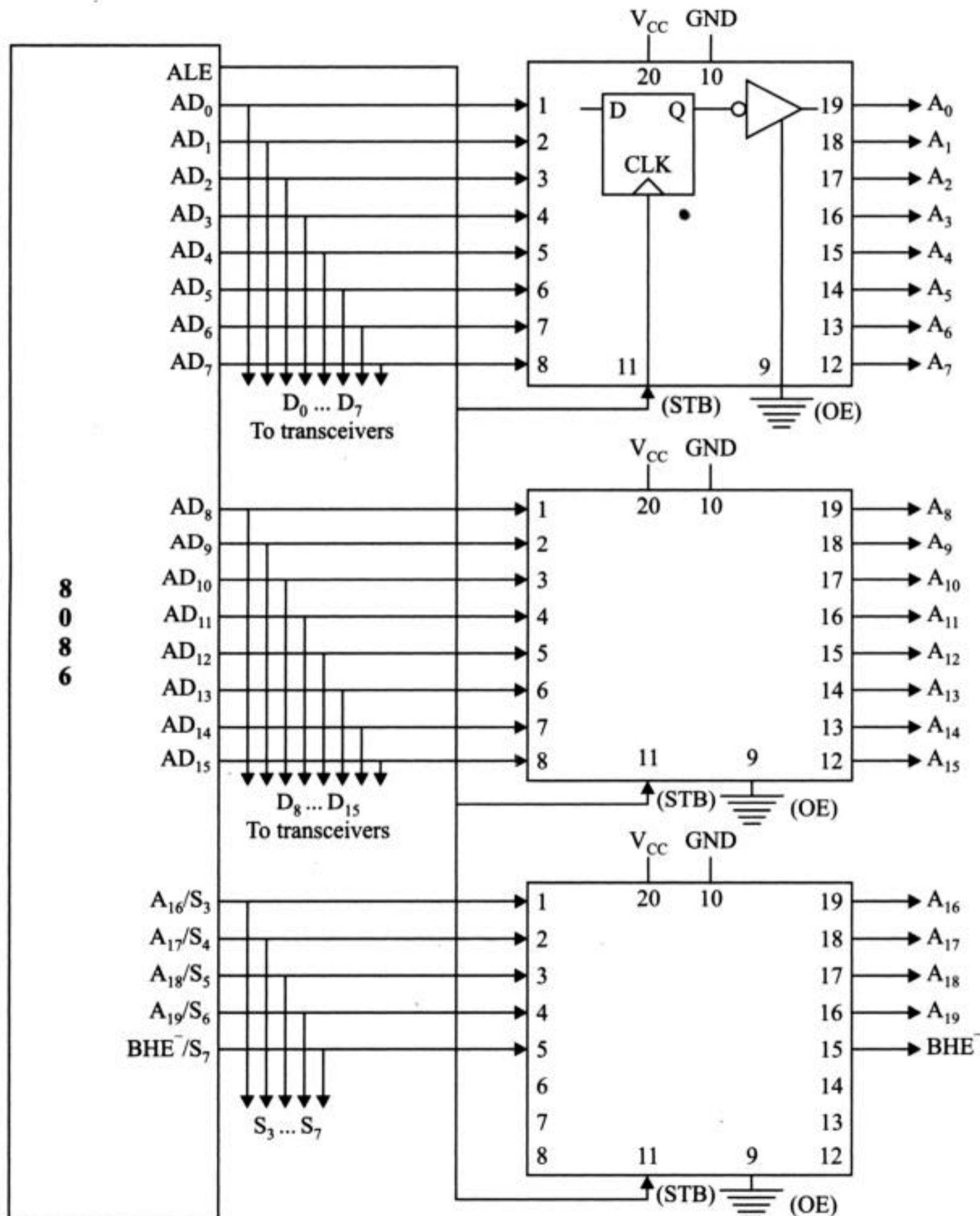


Figure 3.2 Demultiplexing of Address/data bus.

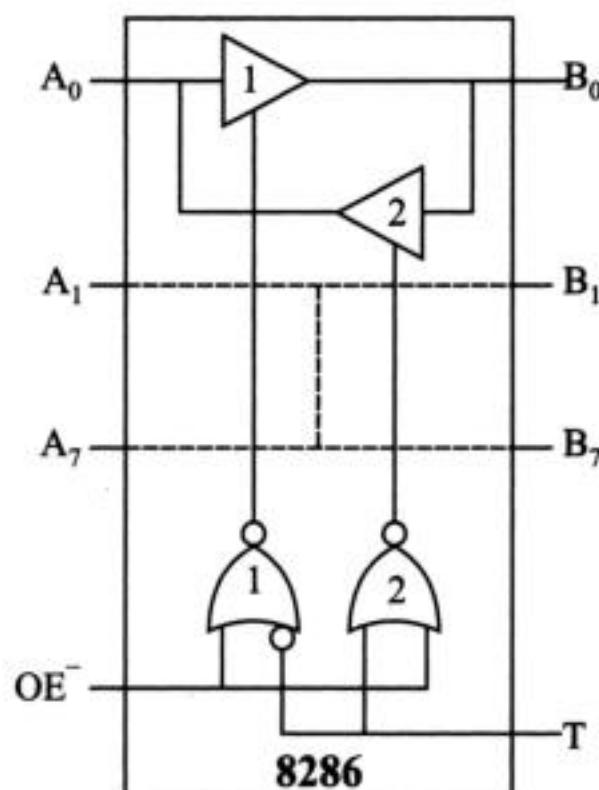
As shown in Figure 3.2, internally the latch IC also consists of tri-stated buffers. These buffers are used to increase the driving capabilities of the buses. These buffers are permanently enabled by grounding the  $OE^-$  terminal.

In the  $T_2$ ,  $T_3$ , and  $T_4$  clock cycles the  $AD_0-AD_{15}$  lines are ready to carry the data whenever an appropriate  $RD^-$  or  $WR^-$  signal is generated, the  $A_{16}/S_3-A_{19}/S_6$  and  $BHE^-/S_7$  lines carry the status signals  $S_3-S_7$ .

The separated lines  $A_0-A_{19}$  will now behave as dedicated address bus of the system and the data lines  $D_0-D_{15}$  are applied to the transceiver.

### 3.2.2 Transceiver 8286

Transceivers are data transmitter and receiver. The transceiver chip 8286 consists of eight bidirectional tri-stated buffers and control logic. Figure 3.3 shows the logic diagram of 8286. The control logic will control the direction of flow of data. In 8286 there are two control signals  $OE^-$  and  $T$ . The  $OE^-$  input is used to enable the buffer operation, whereas  $T$  input selects the direction of intended data transfer. While interfacing with 8086, the  $OE^-$  is connected with  $DEN^-$  and  $T$  is connected with  $DT/R^-$  signals of 8086.



**Figure 3.3** Logic diagram of transceiver.

Assume that the device is enabled by applying  $OE^- = 0$ . Now if  $T$  is set to logic 0, i.e.  $DT/R^-$  is 0, the output of OR gate 2 will be 1 and the buffers numbered as 2 will be enabled. So the data path from  $B_n$  to  $A_n$  will be enabled and hence data will be received by the processor. But the output of OR gate 1 will be logic 0 and the buffers numbered 1 will be disabled and hence the data path  $A_n$  to  $B_n$  will also be disabled. Reverse action will take place when  $DT/R^-$  is at logic 1. Table 3.1 shows the data flow on the basis of  $OE^-$  and  $T$ .

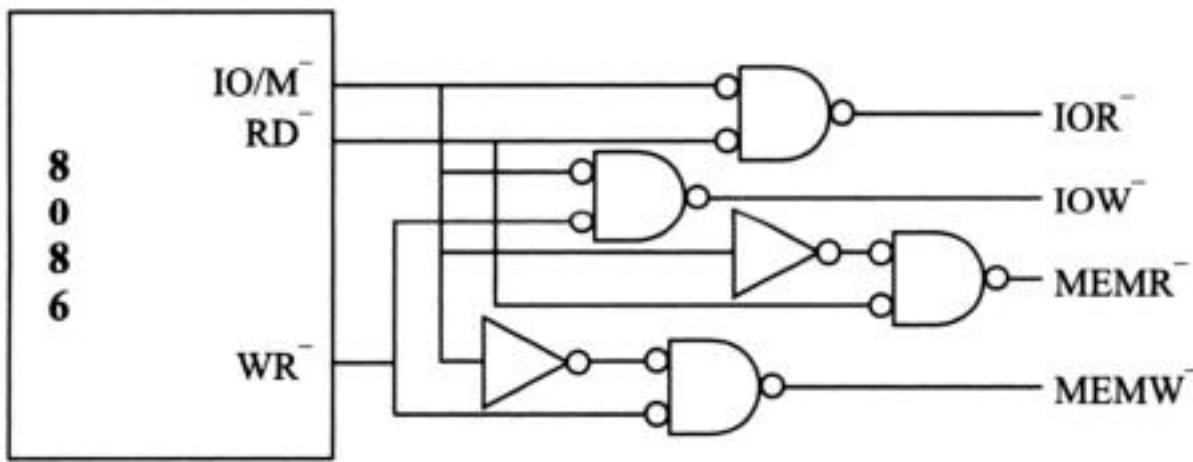
**Table 3.1** Data direction in the transceiver

<b>OE<sup>-</sup> or DEN<sup>-</sup></b>	<b>T or DT/R<sup>-</sup></b>	<b>Data direction</b>
0	1	$A_n \rightarrow B_n$
0	0	$B_n \rightarrow A_n$
1	X	8286 will disable

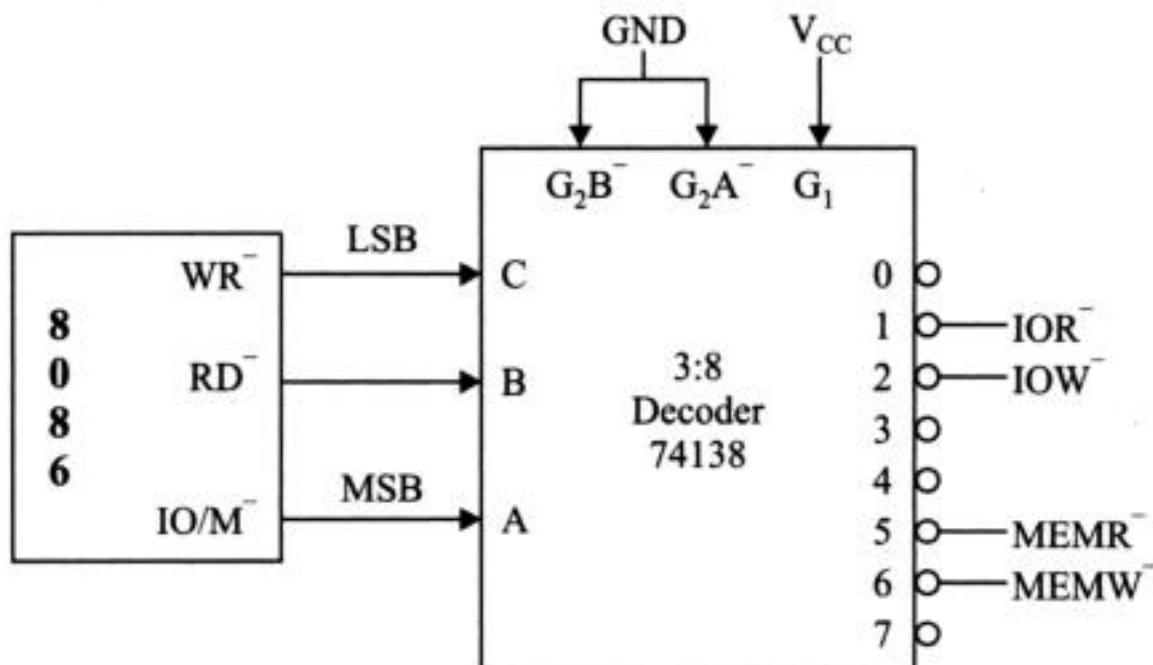
Two 8286s are connected in parallel to achieve 16-bit bidirectional data bus transceiver buffer operation.

### 3.2.3 Generation of Control Signals

Microprocessor 8086 does not give the four control signals, i.e. IOR<sup>-</sup>, IOW<sup>-</sup>, MEMR<sup>-</sup> and MEMW<sup>-</sup> directly. These four control signals are generated from the three control signals: RD<sup>-</sup>, WR<sup>-</sup> and IO<sup>-</sup>/M as shown in Figure 3.4. In Figure 3.4(a), a bubbled NAND gate logic is used to decode the three control signals from the processor as per Table 3.2 to generate the four control signals. From De Morgan's theorem, a bubbled NAND gate is equal to an OR gate, so here in this logic circuit the bubbled NAND gates can also be replaced by OR gate.



(a) With logic gates



(b) With decoder circuit

**Figure 3.4** Generation of control signal.

**Table 3.2** Decoding of control signal

<b>IO<sup>-</sup>/M</b>	<b>RD<sup>-</sup></b>	<b>WR</b>	<b>Control signal</b>
0	0	1	IOR <sup>-</sup>
0	1	0	IOW <sup>-</sup>
1	0	1	MEMR <sup>-</sup>
1	1	0	MEMW <sup>-</sup>

The control signals can also be generated by using a 3\*8 decoder as shown in Figure 3.4(b).

### 3.2.4 Clock Generator 8284 and Driver

8284 is an 18-pin chip which is designed to provide the clock frequency, Ready and Reset signals to 8088/86 microprocessors. The READY and RESET signals are applied to the clock generator and it synchronizes the READY and RESET signals with clock and then apply them to the microprocessor. The block diagram of 8284 is shown in Figure 3.5 and the pin diagram is shown in Figure 3.6. The interfacing of the clock generator 8284 is shown in Figure 3.7.

#### ***Operation of the clock section***

The clock logic generates the three output signals, namely OSC, CLOCK, and PCLK. Crystal oscillator has two inputs X<sub>1</sub> and X<sub>2</sub>. The oscillator generates a square wave signal at the same frequency as a crystal. This square wave is applied to AND gate and also to an inverted buffer (NOT gate) that provides an OSC output signal. Sometimes this OSC signal can be used as EFI input to other 8284 clock generators.

The AND gate is used to apply the oscillator output to the divided by 3 counter, when F/C<sup>-</sup> is at logic 0. When F/C<sup>-</sup> is at logic 1, then EFI is passed through the counter. The output of the divided by 3 counter generates the timing for ready and reset signals.

If in a system there is more than one 8284, then those entire 8284 clock generators need to be synchronized. This synchronization in multiple 8284 is provided by the Clock Synch (CSYNC) input. This input is grounded in case of a single 8284.

The peripheral clock (PCLK) frequency signal is used to provide required clock for supporting peripheral chips like 8253/54. This output signal is acquired by dividing clock frequency by 2.

#### ***Operation of the RESET section***

The reset section of the 8284 consists of Schmitt trigger buffer and single D type FF. This circuit applies the reset signal to microprocessor on the negative edge of each clock. An RC circuit provides logic 0 to the RES input pin when power is first applied to the system. The RES input becomes a logic 1 because the capacitor charges towards + 5 V through the resistor.

Correct reset timing requires the reset input to become logic 1 no later than 4 clocks after system power is applied and to be held high for 50 microseconds. The FF makes certain reset goes high in 4 clocks and RC time constant ensures that it stays high for at least 50 microseconds.

#### ***Operation of the READY section***

The ready section generates the ready signal for 8086/8088. The microprocessor introduces a wait state between T<sub>3</sub> and T<sub>4</sub> states of the machine cycle whenever the READY signal is low.

The ready logic is indicated in Figure 3.5. Whenever the logic level of  $RDY_1$  and  $SEN_1^-$  becomes 01 or the logic level of  $RDY_2$  and  $AEN_2^-$  becomes 01, the ready output becomes low.

The  $AEN_1^-$  is used for generating wait states in the 8086/8088 bus cycle, whereas  $RDY_1$  is used for generating the wait state in the DMA bus cycle.

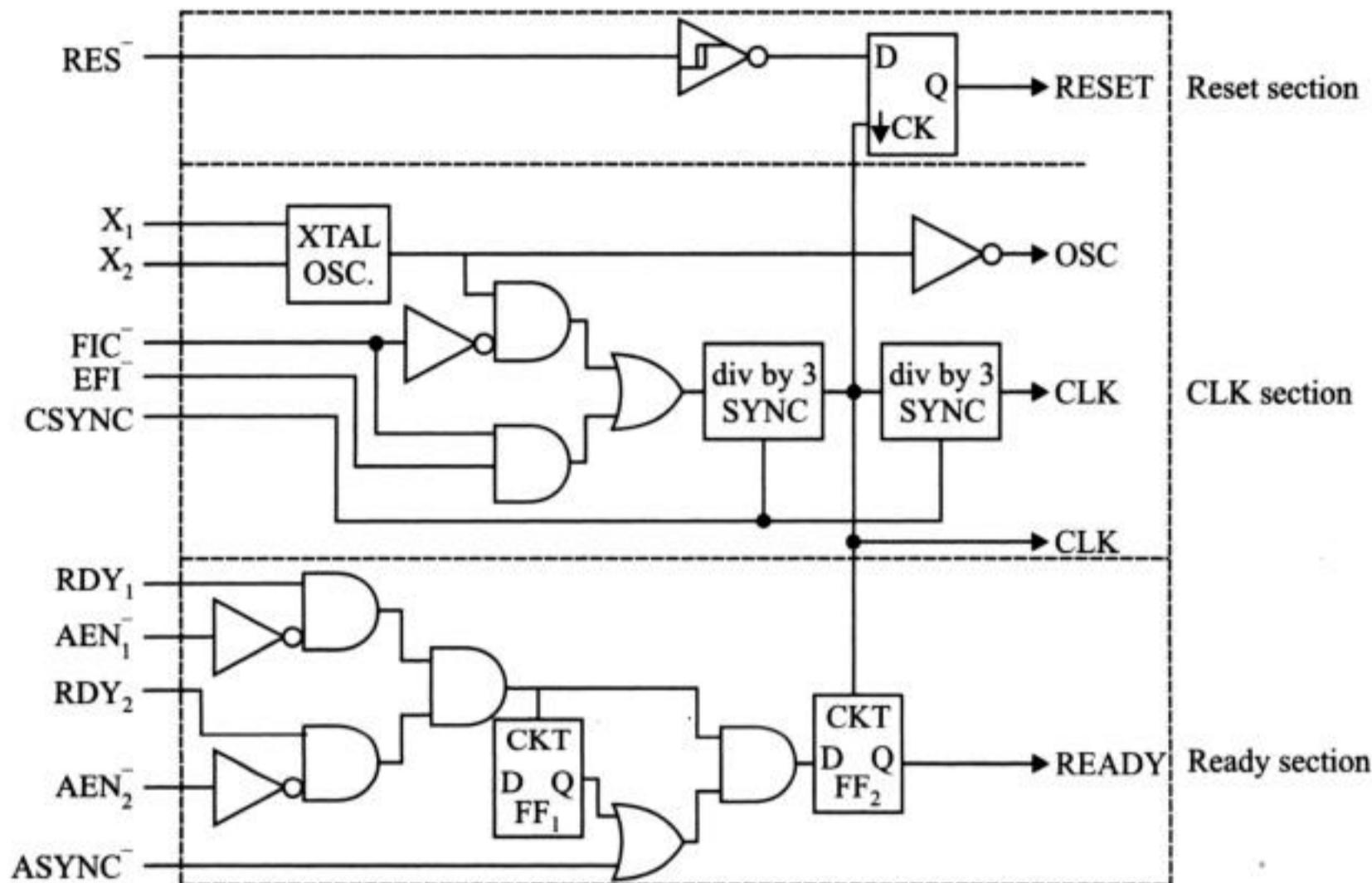


Figure 3.5 Block diagram of 8284 clock generator.

### Pin configuration of clock generator

The following sections discuss the pins of 8284.

- 1. RES (RESET IN):** RES is an active low input signal which is used to generate RESET IN signal for 8284. This is connected to the power supply of the microprocessor. When the microprocessor wakes up, at that time a low signal is generated on the RES pin of 8284. In response to this, the 8284 activates the RESET pin which in turn resets the microprocessor and the processor boot from this state. This booting of the processor is named cold boot.
- 2. X<sub>1</sub> and X<sub>2</sub> (crystal in):** These are the two input pins through which a crystal is attached to the crystal oscillator to generate the desired frequency. The crystal frequency is three times the desired frequency for the microprocessor. The maximum crystal for the 8284A is 24 MHz and 30 MHz for the 8284A-1.
- 3. F/C<sup>-</sup>:** In 8284 the clock may be generated either internally with the help of internal crystal oscillator or with an external source. The F/C<sup>-</sup> pin is used to select whether an internal clock is generated or the external clock is to be used. When this pin is low

the clock is generated by the 8284 and if it is high, then an external source will generate the clocks and applied through the EFI pin.

4. **EFI (external frequency in):** When the F/C<sup>-</sup> is high, then this pin is used through which the external clock frequency is applied. This pin is generally not used in case of microprocessor because in microprocessor we use the internal clock frequency.
5. **CSYNC (clock synchronization):** The clock synchronization is an active-high input signal. It is used to synchronize when many 8284 chips are used together in a system. It remains low when 8284 is used with microprocessor 8086 alone.
6. **RDY<sub>1</sub> and AEN<sub>1</sub><sup>-</sup>:** These two signals are the input signals to 8284. The RDY<sub>1</sub> (Ready) is active high and AEN<sub>1</sub><sup>-</sup> (Address Enable) is active low. These two signals used together to provide the READY signal to the microprocessor. This Ready signal is used by the processor to synchronize it with the slower peripherals.
7. **RDY<sub>2</sub> and AEN<sub>2</sub><sup>-</sup>:** These two signals are an additional set of ready and address enable signals and the function of these pins are the same as that of RDY<sub>1</sub> and AEN<sub>1</sub><sup>-</sup>. These additional RDY and AEN<sup>-</sup> signals are used in multiprocessing system.
8. **RESET:** RESET is an active-high output signal which is connected to the RESET signal of 8088/86 microprocessor.

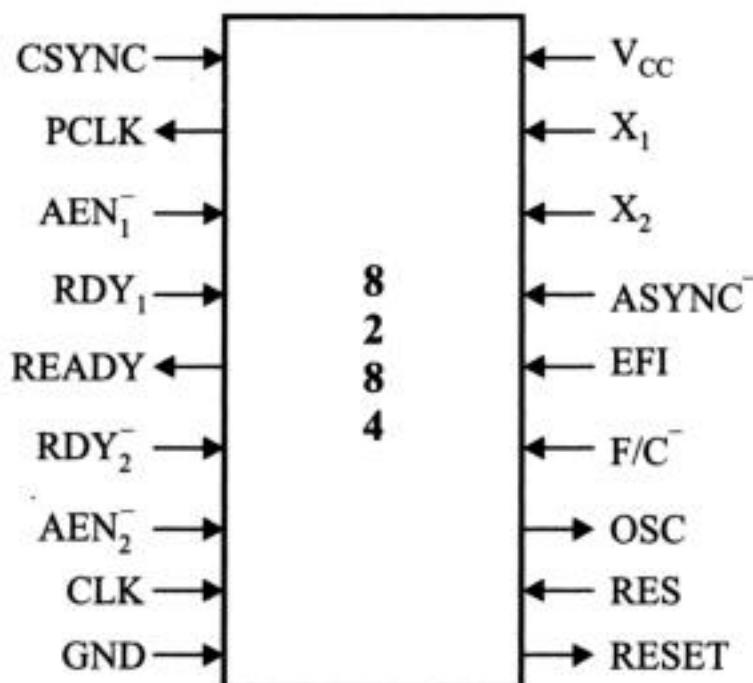
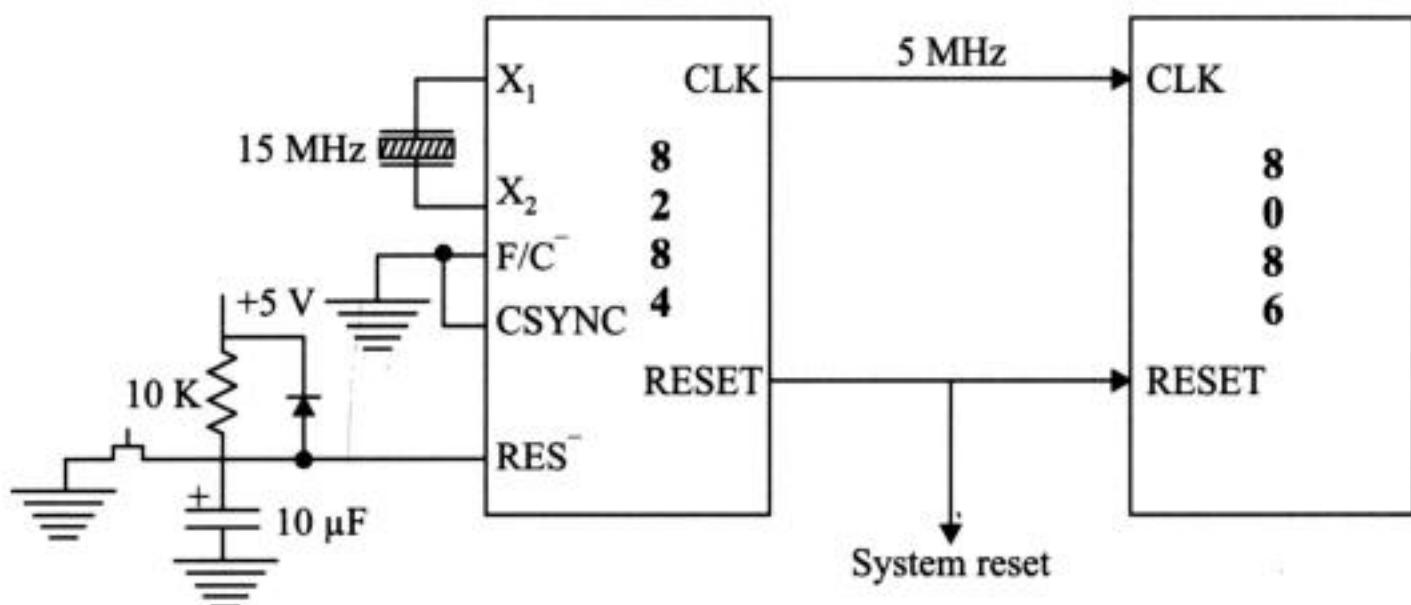


Figure 3.6 Pin configuration of 8284.

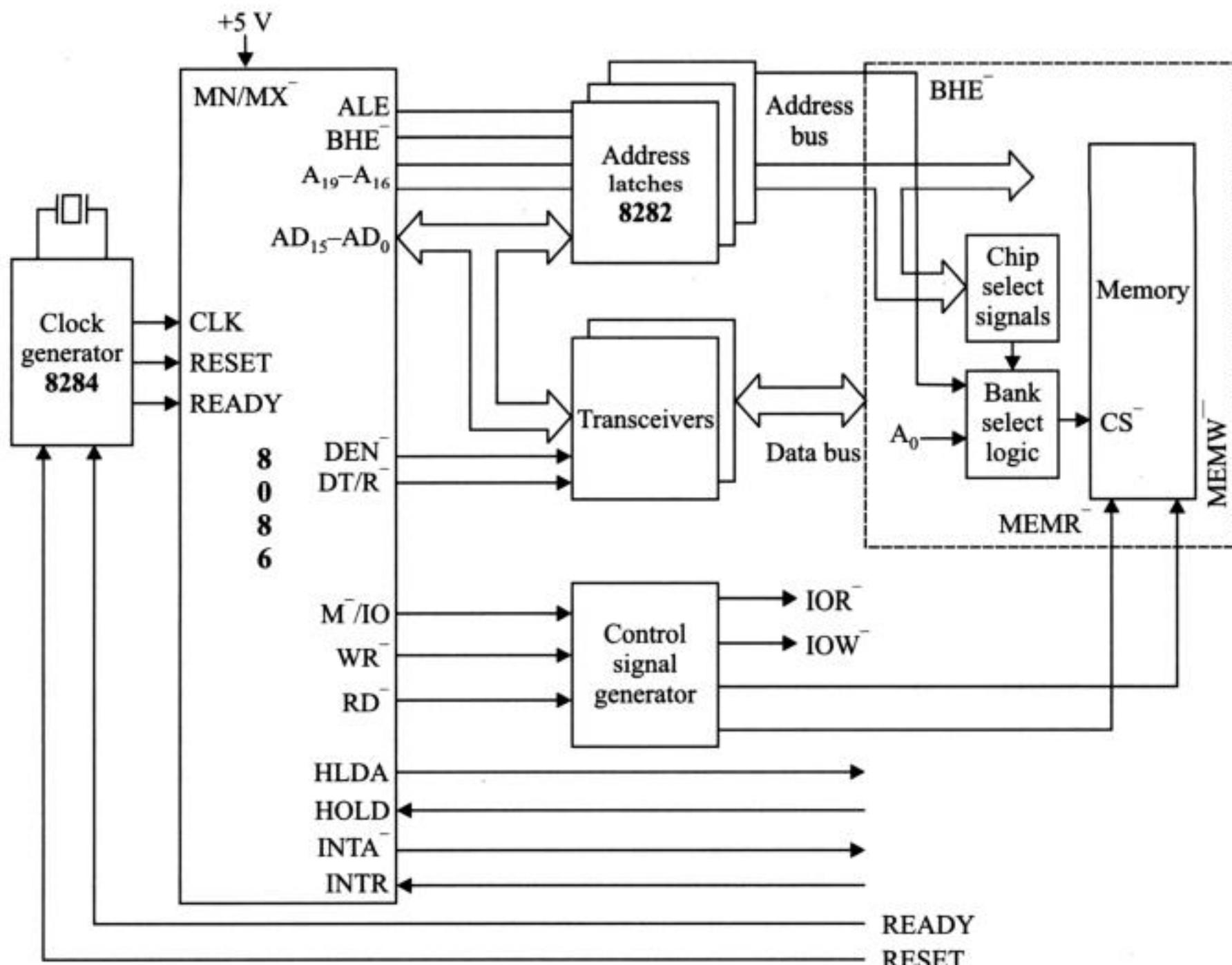
9. **OSC (oscillator):** Oscillator is the output signal and its clock frequency is the same as that of the crystal oscillator.
10. **CLK (clock):** This output signal provides one-third of the crystal oscillator frequency as the clock frequency. This output is connected to the clock input of the processor. It also provides the clock frequency to all other devices in the system and which are to be synchronized with the processor.
11. **PCLK (peripheral clock):** This output signal is used to provide clock signal to the peripherals like 8254. This frequency is one-half of CLK (or one-sixth of the crystal) with a duty cycle of 50%.
12. **READY:** This output signal is connected to READY signal of the processor.



**Figure 3.7** Interfacing of clock generator 8284 with 8086.

### 3.2.5 Interfacing of Memory in Minimum Mode

The block diagram in Figure 3.8 shows the memory interface circuit of an 8086 based system operating in minimum mode.

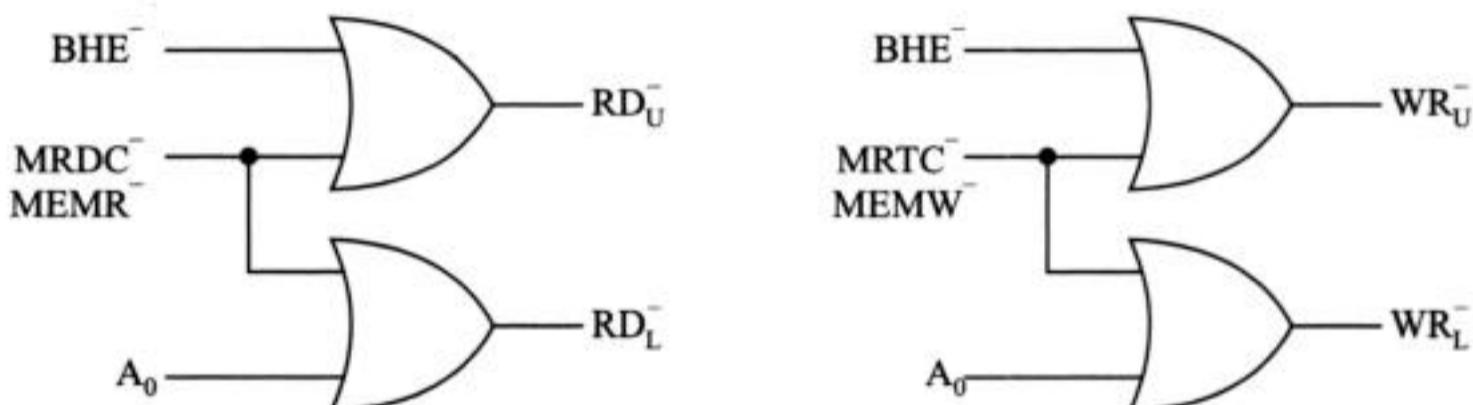


**Figure 3.8** Memory interfacing in minimum mode.

The block diagram in Figure 3.8 is just an extension of Figure 3.1. The dotted block shows the memory interfacing concept of 8086 in minimum mode configuration. This block shows that some of the address lines (depending on the size of the memory chip) from  $A_1 - A_{19}$  are connected to the memory chip and the remaining address lines will go to the chip select signal generator. The output of the chip select signal is applied to the bank select logic to generate the  $CS^-$  signals for the two memory banks.

The internal diagram of the bank select logic is shown in Figure 3.9. The  $BHE^-$  and  $A_0$  along with the control signal  $MEMR^-$  (in minimum mode) or  $MRDC^-$  (in maximum mode) generate the control signals  $RD_U^-$  (Read upper bank) and  $RD_L^-$  (Read lower bank). Similarly  $A_0$ ,  $BHE^-$  and  $MEMW^-$  (in minimum mode) or  $MWTC^-$  (in maximum mode) generate the control signals  $WR_U^-$  (write upper bank) and  $WR_L^-$  (write lower bank).

Bank read and write control logic circuits enables even and odd address byte transfer, as per logic levels of  $BHE^-$  and  $A_0$  signals.



**Figure 3.9** Read and write bank control logic circuits.

### 3.3 MAXIMUM MODE CONFIGURATION OF 8086

In the maximum mode of operation of 8086, more than one processor is present in the system, i.e. another processor is interfaced with 8086. The other processor may be either a numeric coprocessor 8087 or any other independent processor like 8086 or 8088. All the resources like memory, address bus, data buses are shared between the two processors.

The block diagram of the maximum mode configuration system of 8086 is shown in Figure 3.10. In maximum mode three 8-bit latches (IC 8282), two 8-bit transceivers (IC 8286) and one clock generator (8284) are used along with the bus controller 8288.

The latches are used to demultiplex the multiplexed address/data lines and also address/status signals. The two transceivers are used to enable the data flow and direction of the data flow. The clock generator is used to generate the clock and also synchronize the READY and RESET signals.

The control signals for maximum mode of operation are generated by the Bus Controller chip 8288. The three status outputs  $S_0^-$ ,  $S_1^-$ ,  $S_2^-$  from the processor are input to 8288. The outputs of the bus controller are the Control Signals, namely DEN, DT/R $^-$ , IORC $^-$ , IOWTC $^-$ , MWTC $^-$ , MRDC $^-$ , ALE, etc. These control signals perform the same task as the minimum mode operation. However, the DEN is an active HIGH signal which has to be converted to active LOW by means of an inverter.

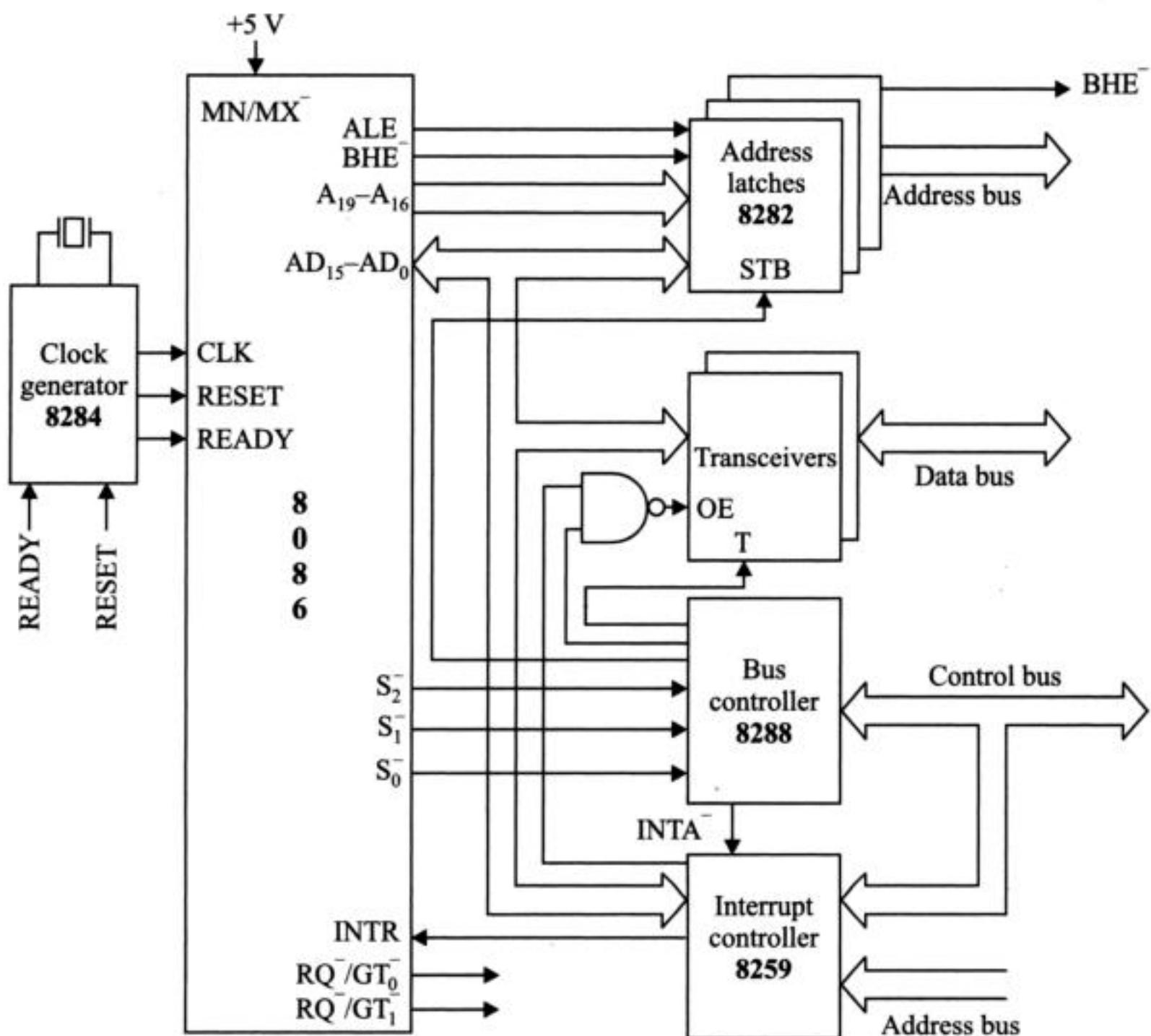


Figure 3.10 Maximum mode configuration.

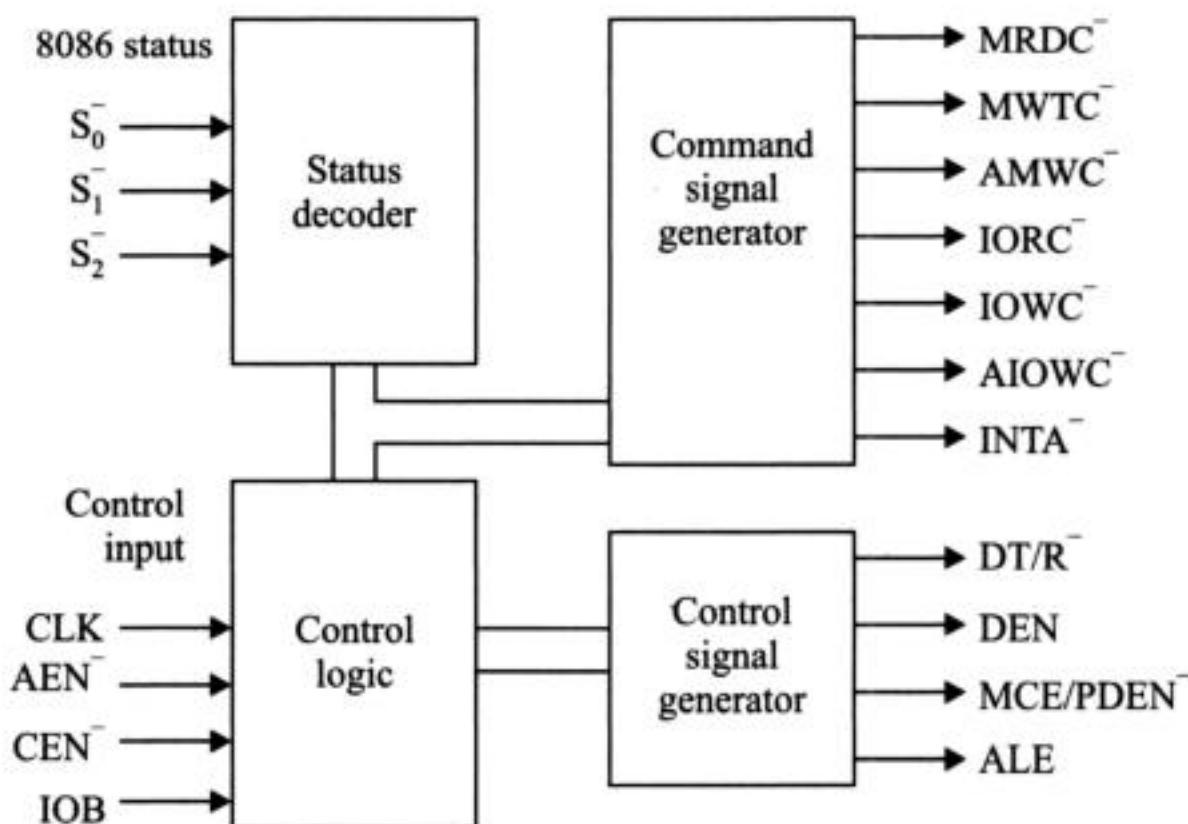
### 3.3.1 Bus Controller 8288

The 8288 provides the control and command timing signals for 8086, 8088, 8089, 80186, and 80188 based systems. The block diagram of the bus controller 8288 is shown in Figure 3.11. The bus controller 8288 works in two operating modes, i.e. IO Bus mode and System Bus mode. These modes are selected by the three control inputs CEN, IOB and AEN<sup>-</sup> as per Table 3.3.

#### I/O bus mode

When the IOB pin is high, then the 8288 will work in IO mode. In this mode, all I/O command lines, i.e. IORC<sup>-</sup>, IOWC<sup>-</sup>, AIOWC<sup>-</sup> and INTA<sup>-</sup> are always enabled and independent of AEN signal. In the maximum mode, the transceivers are enabled by the PDEN<sup>-</sup> and DT/R<sup>-</sup>. These signals are generated by the 8288 as soon as an I/O command is initiated by the processor. In IO bus mode the system bus is not controlled by the I/O command lines due to the non-availability of bus arbitrator logic in this mode. In this mode the microprocessor need not to wait for the IO buses whenever it requires them. In a multiprocessor environment, when

a processor has its own dedicated IO or peripherals, then it is advantageous to use the IOB mode.



**Figure 3.11** Block diagram of bus controller 8288.

**Table 3.3** Selection of operating modes of 8288

CEN	IOB	AEN <sup>-</sup>	Description
1	1	X	I/O bus mode
1	0	1	System bus mode, but all control signals are disabled
1	0	0	System bus mode, but all control signals are enabled
0	X	X	Open circuited, i.e. all command outputs and DEN and PDEN are disabled

### **System bus mode**

When the IOB line of 8288 is low, at that time the 8288 will work in the system bus mode. In this mode, command is issued after a specified time period after the activation of AEN signal. The system bus mode is used in a multiprocess environment when only one bus exists and both I/O and memory are shared by more than one processor.

Bus arbitration logic is used in this mode for bus arbitration. The bus arbitration logic informs the bus controller whenever the buses are free for use through the AEN line. Both memory and I/O commands wait for bus arbitration.

### **Functional block diagram of 8288**

The block diagram of 8288 consists of four blocks. These blocks are:

- (i) Status decoder
- (ii) Command signal generator
- (iii) Control logic
- (iv) Control signal generator.

The 8288 bus controller receives seven input signals. Out of these seven input signals, four inputs ( $S_0^-$ ,  $S_1^-$ ,  $S_2^-$  and CLK) are from the 8086. The command logic decodes the three 8086 status lines ( $S_0^-$ ,  $S_1^-$ , and  $S_2^-$ ) to determine what command is to be issued (see Table 3.4). The CLK input provides the desired clock frequency to the bus controller.

The remaining three input signals are the control signals CEN, IOB and AEN<sup>-</sup>. These control signals determine the operating modes of 8288. Table 3.4 shows how these signals decide the operating modes.

**Table 3.4** Decoding of the status signals

<b><math>S_2^-</math></b>	<b><math>S_1^-</math></b>	<b><math>S_0^-</math></b>	<b>Processor state</b>	<b>Command</b>
0	0	0	Interrupt acknowledge	INTA <sup>-</sup>
0	0	1	Read I/O port	IORC <sup>-</sup>
0	1	0	Writer I/O port	IOWC <sup>-</sup> , AIOWC <sup>-</sup>
0	1	1	Halt	None
1	0	0	Code access	MRDC <sup>-</sup>
1	1	1	Read memory	MRDC <sup>-</sup>
1	1	0	Write memory	MWTC <sup>-</sup> , AMWC <sup>-</sup>
1	1	1	Passive	None

The 8288 produces two types of outputs, viz. the command outputs and the control outputs.

### **Command outputs**

The 8288 bus controller produces all the control signals in the maximum mode configuration system in 8086. The following command outputs are generated by 8288:

- (i) MRDC<sup>-</sup> — Memory read command
- (ii) MWTC<sup>-</sup> — Memory write command
- (iii) IORC<sup>-</sup> — I/O read command
- (iv) IOWC<sup>-</sup> — I/O write command
- (v) AMWC<sup>-</sup> — Advanced memory write command
- (vi) AIOWC<sup>-</sup> — Advanced I/O write command
- (vii) INTA<sup>-</sup> — Interrupt acknowledge.

The advanced memory and IO write commands are used to initiate the write operation slightly before the actual commencement of the commands in the machine cycle. This signal can be used to avoid the processor from entering a redundant wait state.

As the INTR line is available in both the minimum mode as well as maximum mode, there is no Interrupt acknowledge in the maximum mode. So the INTA<sup>-</sup> (interrupt acknowledge) signal is generated by the bus controller. It is used to inform an interrupting device that its interrupt is being acknowledged.

### **Control outputs**

The data enable (DEN<sup>-</sup>), data transmit/receive (DT/R<sup>-</sup>) and master cascade enable/peripheral data enable (MCE/PDEN<sup>-</sup>) are the various control outputs of the 8288.

The  $\text{DEN}^-$  signal is used to enable the transceiver and determines when the external bus should be enabled. The  $\text{DT/R}^-$  is used to set the direction of the data flow by enabling one of the two buffers in the transceiver. The function of the  $\text{MCE/PDEN}^-$  pin depends on the two modes of the 8288. This signal serves as the  $\text{PDEN}^-$  signal in the IOB mode. In this mode this signal serves as a dedicated data enable signal for the I/O or Peripheral System bus.

The  $\text{MCE/PDEN}^-$  pin serves as the MCE signal in the System bus mode. This signal is used during an interrupt acknowledge cycle. This signal is used when a Priority Interrupt controller, in their cascade mode, is interfaced with the processor. Whenever there is an INTR signal, the system generates two INTA cycles. During the first INTA cycle, the MCE signal is masked and there is no data or address transfer. Before the starting of the second INTA cycle, the MCE signal allows a master priority interrupt controller's (PIC) cascade address onto the processor's local bus. In the second INTA cycle, the addressed slave PIC gates an interrupt vector onto the system data bus.

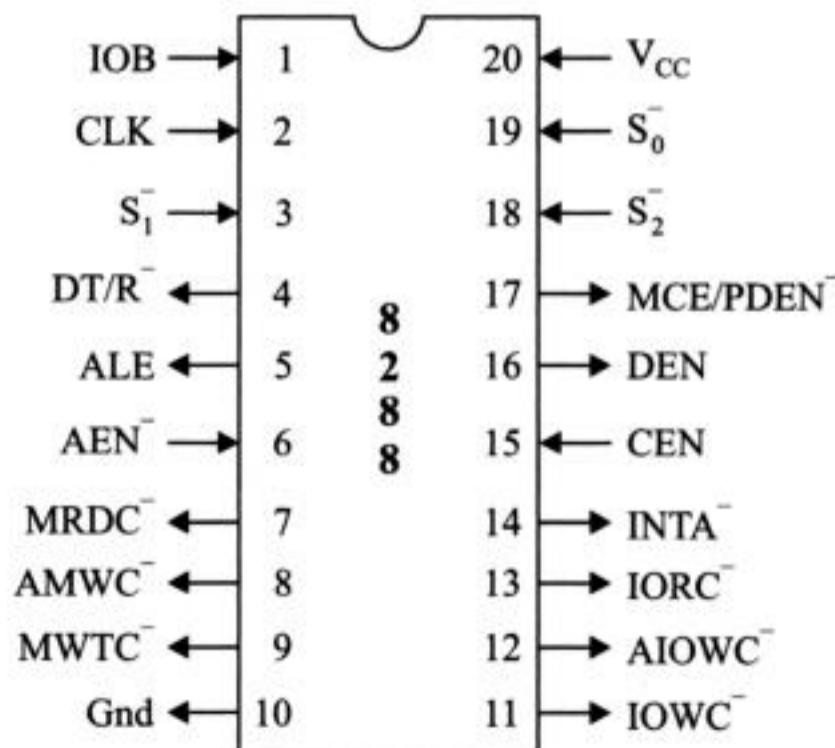
In the maximum mode the address latch enable (ALE) signal is generated by the 8288. Its function is similar to that of the ALE in minimum mode. The status  $S_0^-$ ,  $S_1^-$  and  $S_2^-$  are also latched by the ALE signal for halt state decoding.

The command enable (CEN) input is something like an enable or chip select signal for the 8288. The 8288 functions normally or activated when the CEN pin is high. If this signal is low, then all command lines are held in their inactive state but not tri-stated.

### **Pin configuration of bus controller 8288**

Figure 3.12 shows the pin configuration of 8288. 8288 is a 20-pin chip. These signals are explained below.

**$S_2^-$ ,  $S_1^-$ , and  $S_0^-$  (Status signals):** These are three status signals which are input to the 8288. These signals are connected to the output status signals of 8086 in its maximum mode. These signals are decoded by 8288 to generate the control and timing signals for the system in maximum mode configuration.



**Figure 3.12** Pin diagram of bus controller 8288.

**CLK (Clock input):** The clock input signal provides the clock signal for the internal timings. This CLK signal is connected to the CLK output pin of the clock generator 8284A.

**ALE (Address latch enable):** This signal is the same as that of ALE signal of 8086 in minimum mode and it performs the same function as that of ALE signal, i.e. demultiplex the multiplexed address/data/status signals.

**DEN (Data enable):** The DEN controls the bidirectional data bus buffer or the transceivers in the system. This is an active high signal unlike the DEN<sup>-</sup> signal in minimum mode of 8086 which is active low.

**DT/R<sup>-</sup> (Data transmit/receive):** This is an output signal from 8288. It is used to control the direction of the bidirectional data buffers in the transceiver chip. This is similar to the DT/R<sup>-</sup> signal of the 8086 in minimum mode.

**AEN<sup>-</sup> (Address enable):** This active low input signal is used by 8288 to enable the memory control signals.

**CEN (Control enable):** This active high input signal is used to enable the command output signals on the 8288. When it is low, the command output signals are disabled. It is used in system bus mode.

**IOB (IO bus mode):** This active high input signal is used to select either the IO bus mode or the system bus mode. When it is high, the 8288 will work in IO mode.

**AIOWC<sup>-</sup> (Advance IO write command):** This is an active low command output signal. It is used to provide the advance IO write control signal.

**IORC<sup>-</sup> (IO Read command):** This is active low IO read command signal generated by 8288 in maximum mode. This is just like IOR<sup>-</sup> signal of minimum mode.

**IOWC<sup>-</sup> (IO Write command):** This is active low IO write command signal generated by 8288 in maximum mode. This is the same as that of IOW<sup>-</sup> signal of minimum mode.

**AMWT<sup>-</sup> (Advance memory write command):** This active low output signal provides an advance memory write control signal.

**MWTC<sup>-</sup> (Memory write command):** This active low memory write command signal provides the normal memory write control signal.

**MRDC<sup>-</sup> (Memory read command):** The memory read control signal provides memory with read control signal.

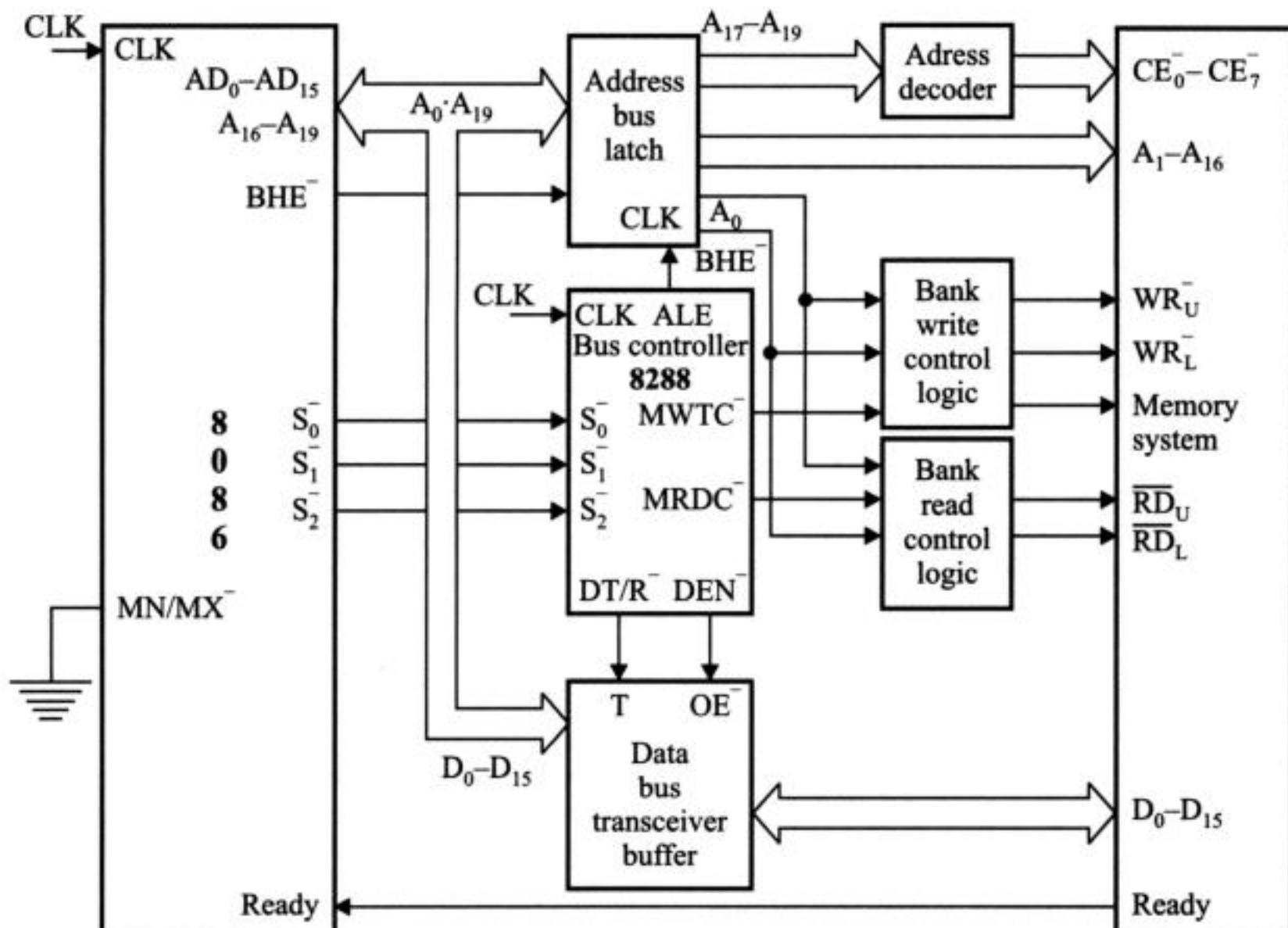
**INTA<sup>-</sup> (Interrupt acknowledge):** The interrupt acknowledge output signal is the acknowledgement of the INTR signal in maximum mode configuration system.

**MCE/PDEN<sup>-</sup> (Master cascade/peripheral data):** This output signal selects cascade operation for an interrupt controller if IOB is grounded, and enable the IO bus transceiver if IOB is high.

### 3.3.2 Memory Interface of a Maximum-Mode 8086 System

The block diagram in Figure 3.13 shows the memory interface circuit of an 8086 based system operating in maximum mode.

The bus controller is introduced here due to the support of multiprocessor environment of maximum mode. The decoder is used to select the desired memory chips. The remaining components of this circuit are similar to 8086 minimum mode circuit.



**Figure 3.13** Memory interfacing in maximum mode configuration.

The bank high enable signal is used to control the access of even or odd memory banks of 8086 system.

### 3.4 BUS CYCLES OF 8086

Instruction cycle is defined as the time taken by the processor to execute an instruction. Each processor has different cycles based on different instruction sets. Typically all the processor utilizes the following five stage cycles.

**Stage 1: Fetch the instruction from main memory:** The processor loads the contents of the Instruction Pointer (IP) on the address bus. The processor then generates the MEMR control signal. On receiving the memory read control signal, the memory loads the opcode on the data bus. This opcode is then placed into the Instruction Register (IR).

**Stage 2: Decode the instruction:** The instruction decoder decodes what is to be done in response of an opcode.

**Stage 3: Fetch data from main memory:** Read the required data from main memory to be processed and placed into registers.

**Stage 4: Execute the instruction:** From the instruction register, the data forming the instruction is decoded by the Instruction decoder machine cycle encoder unit of microprocessor. This unit sends the decoded information to the control unit. The control unit then sends the

decoded information to the relevant functional units of the microprocessor to perform the operation.

**Stage 5: Store results:** The result obtained by the execution of the operation is stored in the destination location. During this whole operation the content of the instruction pointer is incremented to point towards the next memory location or a different address is loaded into the instruction pointer in case of control transfer instructions.

All these operations are performed with respect to the clock. Microprocessor performs an operation in a specific time period, i.e. specific clock cycles known as *T-state*. T-state is the time period of a single cycle of the clock frequency as shown in Figure 3.14.

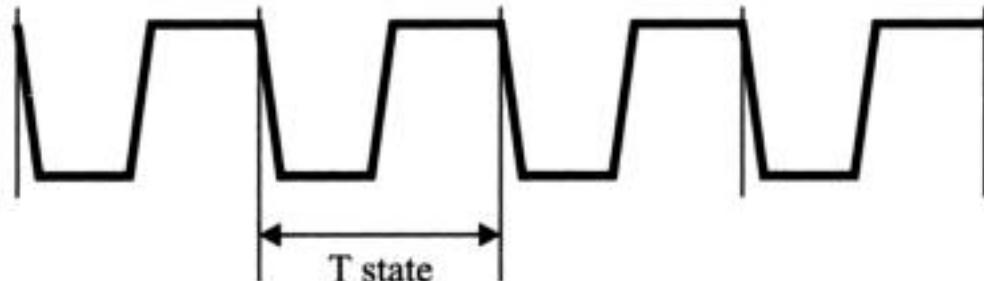


Figure 3.14 T-state.

The number of T-state required to access a peripheral is called machine cycle. Access a peripheral means to perform a read or a write operation either from memory or an IO. Instruction cycle is the total number of machine cycles required to execute a complete instruction. Or the number of machine cycles required to fetch and execute an instruction is called an instructions cycle.

In 8086 a memory read or write bus cycle requires 4 T-states. The status of the various signals with respect to the clock frequency is as follows:

#### **T<sub>1</sub>-state**

The bus cycle starts with T<sub>1</sub> state. During this time control signals are generated to give the required logic values for IO<sup>−</sup>/M, ALE, DT/R<sup>−</sup> and a valid address is placed onto the address bus.

#### **T<sub>2</sub>-state**

During this time the RD<sup>−</sup> or WR<sup>−</sup> control signals are issued, DEN<sup>−</sup> is asserted and in the case of a write, data is put onto the data bus. The DEN<sup>−</sup> turns on the data bus buffers to connect the microprocessor to the external data bus. The READY input to the microprocessor is sampled at the end of T<sub>2</sub> and if READY is low, wait state (T<sub>w</sub>) is inserted before T<sub>3</sub> begins.

#### **T<sub>3</sub>-state**

This clock period provides sufficient time to allow memory to access the data. If the bus cycle is a read cycle, the data bus is sampled at the end of T<sub>3</sub>-state.

#### **T<sub>4</sub>-state**

In T<sub>4</sub>-state all the bus signals are deactivated or in high impedance state for the preparation of the next clock cycle. The 8086 also finishes sampling the data (in a read cycle) in this period. For the write cycle, the trailing edge of the WR<sup>−</sup> signal transfers data to the memory.

### 3.4.1 Minimum Mode Bus Cycles

The timing diagram for 8086 minimum mode memory read operation is shown in Figure 3.15.

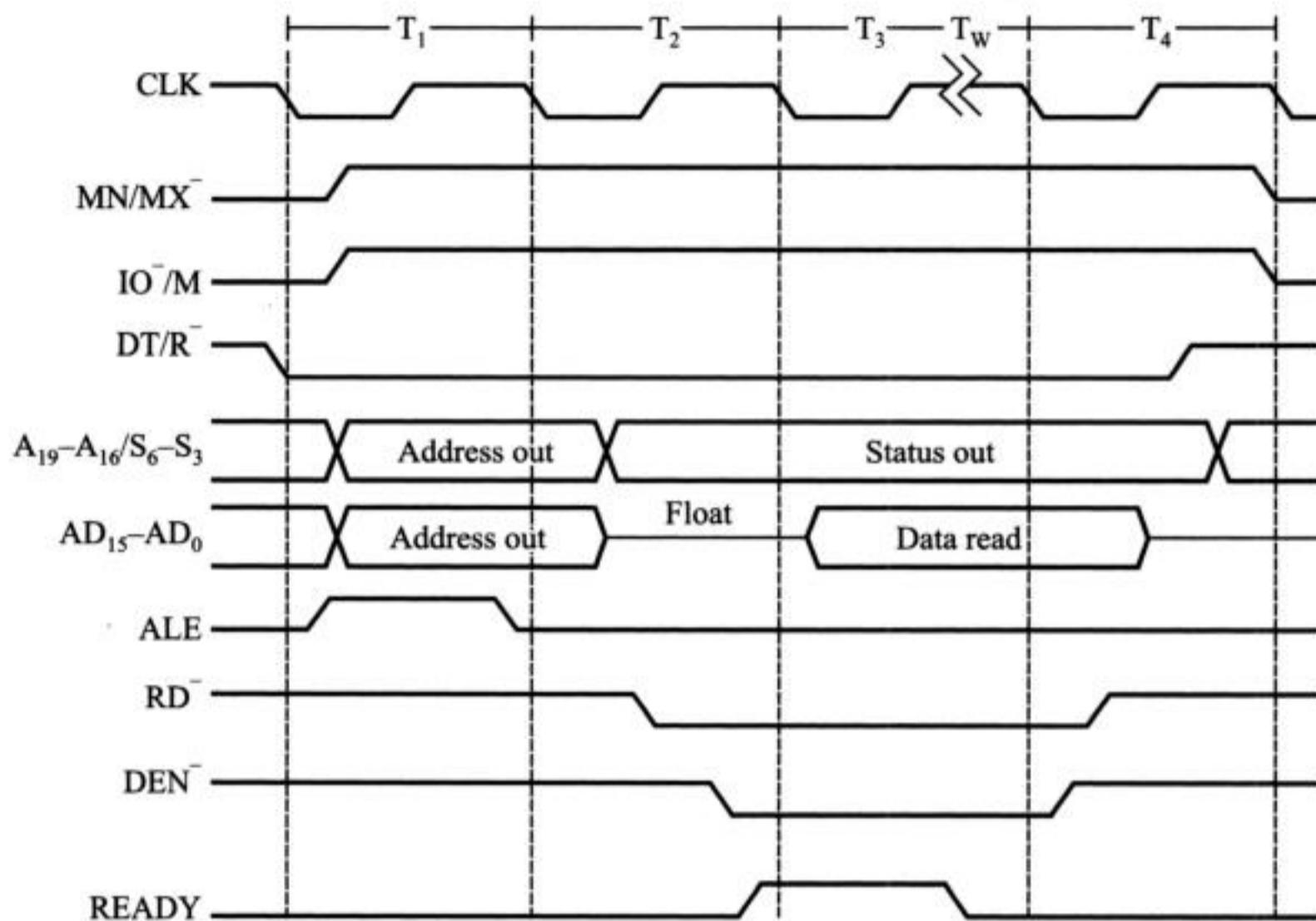


Figure 3.15 Timing diagram for 8086 minimum mode memory read.

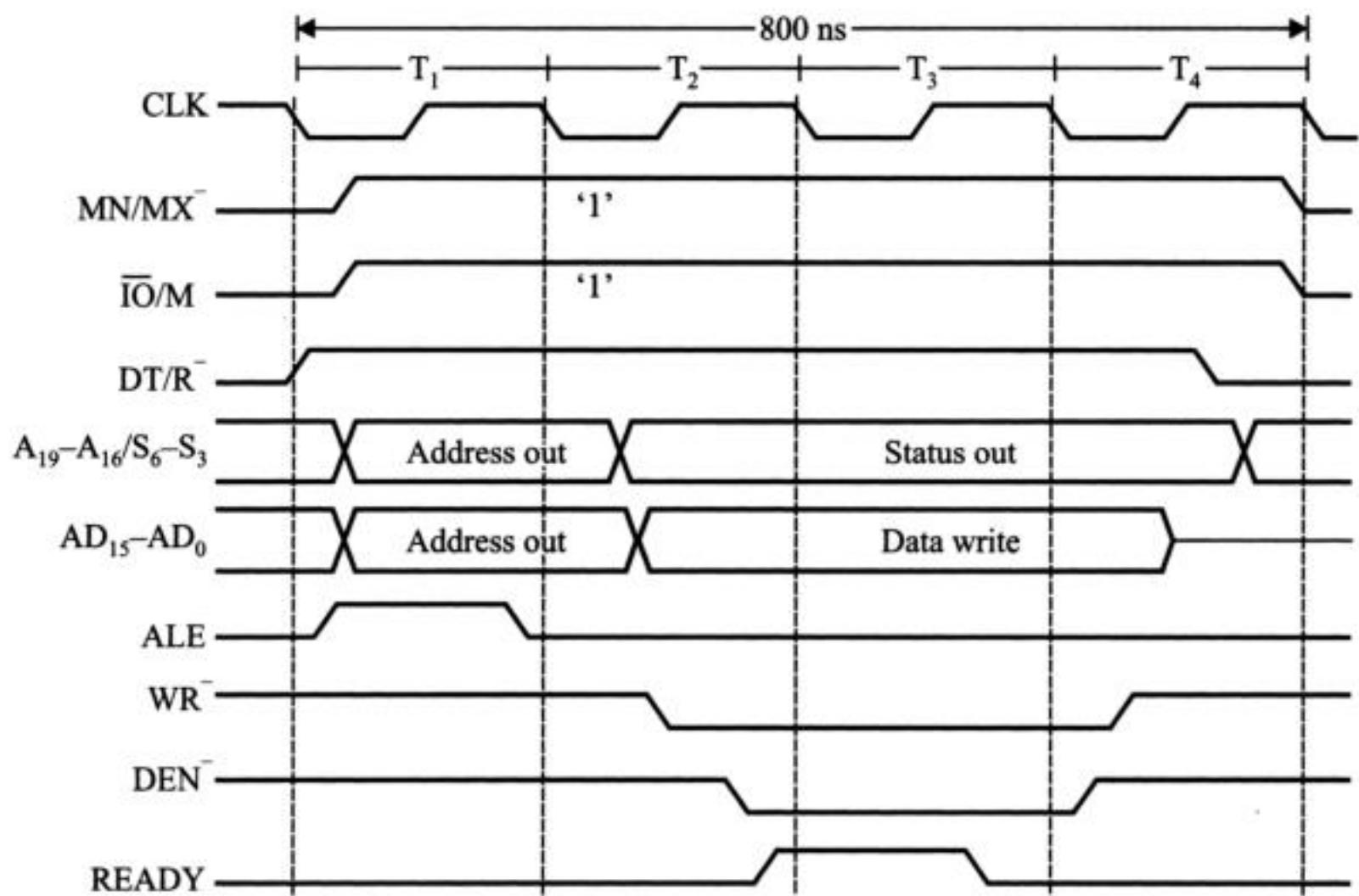
During  $T_1$  or the 1st clock pulse, the read bus cycle starts and valid address is latched together with setting minimum and maximum mode, input output or memory interface, data transmit and receive mode of buffer IC. During  $T_2$ , the Read control signals are issued and data enable signal is asserted. Note that during this state the READY signal is also checked to insert wait status, if needed. During  $T_3$ , the data from the memory is read by sampling the data bus at the end of  $T_3$ . During  $T_4$ , all bus signals are deactivated in preparation for the next clock cycle.

The timing diagram for 8086 minimum mode memory write operation is shown in Figure 3.16. This is the same as read cycle Timing Diagram except that the DT/R<sup>-</sup> line goes high indicating it is a Data Transmission operation for the processor to memory/peripheral. Again DEN<sup>-</sup> line goes low to validate data and WR<sup>-</sup> line goes low, indicating a Write operation.

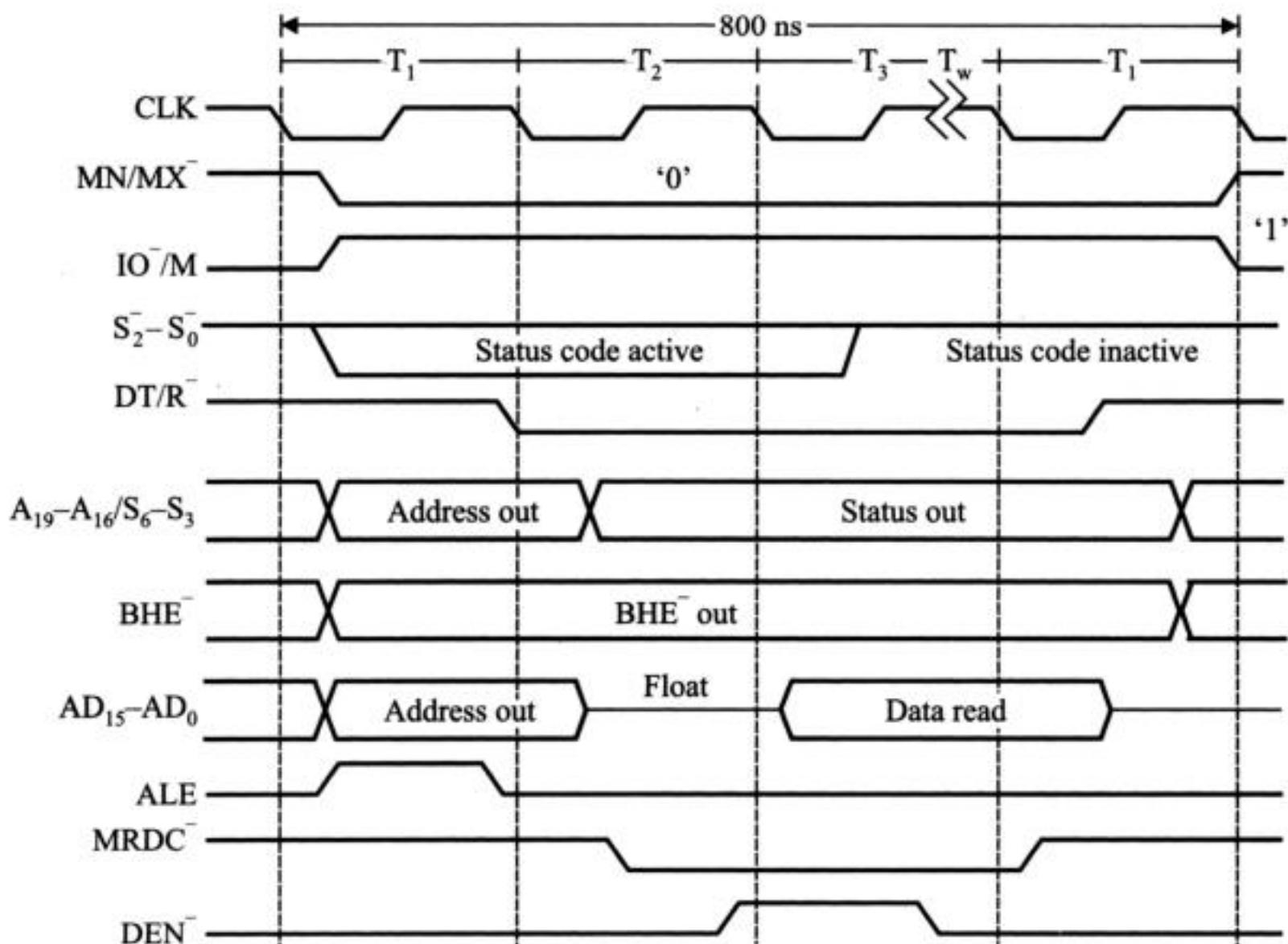
Note that the control signal logic levels and timing diagram are similar to that of read operation, except for data transmit or receive mode, read and write signals. The response of the ready signal is not shown here, as it behaves in the same way as data read process.

### 3.4.2 Maximum Mode Bus Cycles of 8086 System

The timing diagram for 8086 maximum mode memory read operation is shown in Figure 3.17. In maximum mode, status codes need to be active to generate control signals from bus



**Figure 3.16** The timing diagram for 8086 minimum mode memory write.



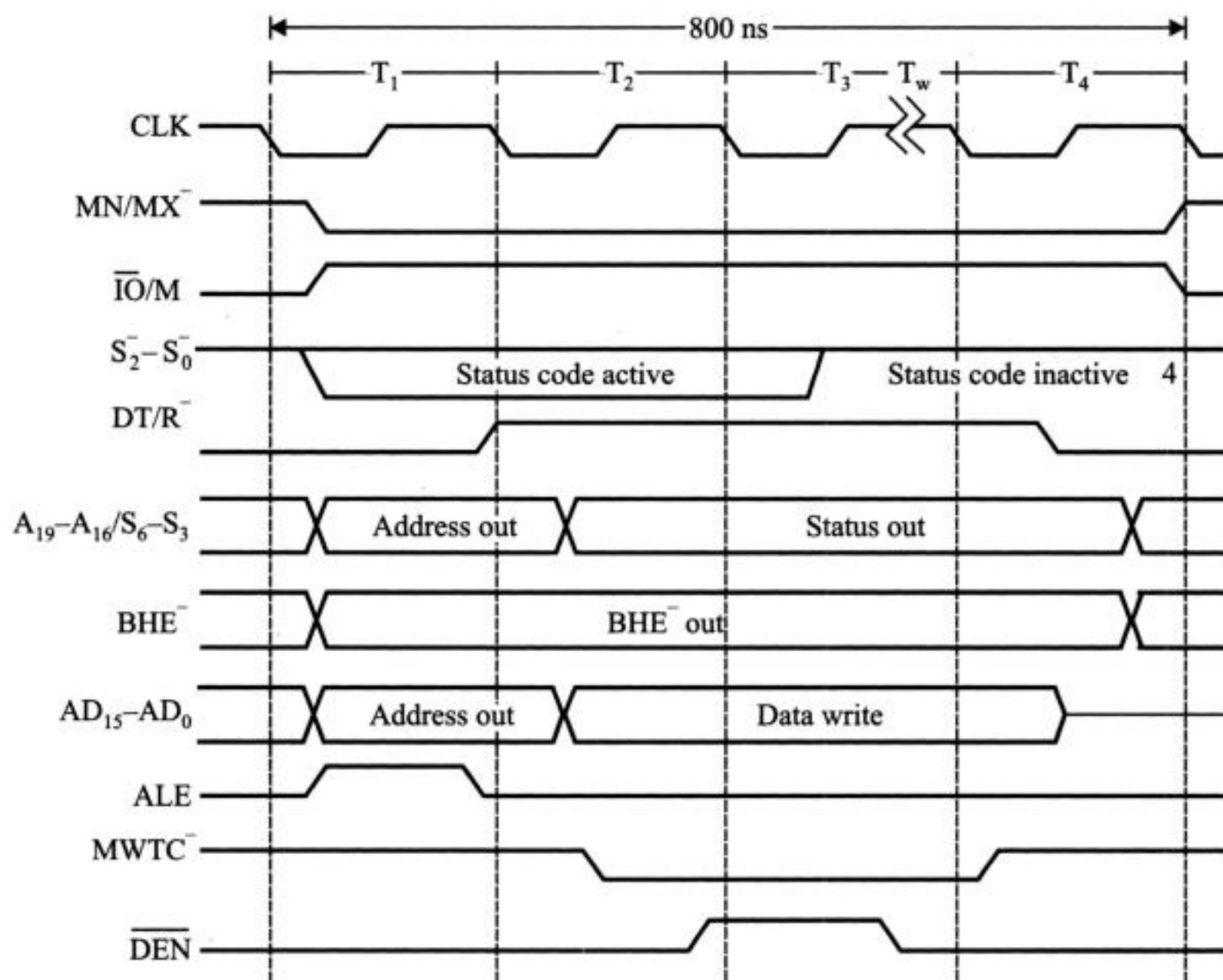
**Figure 3.17** Maximum mode memory read bus cycle of 8086 system.

controller. Here MRDC-signal is used instead of RD<sup>-</sup> as in case of minimum mode S<sub>0</sub><sup>-</sup>, S<sub>1</sub><sup>-</sup> and S<sub>2</sub><sup>-</sup> are active and are used to generate control signal through the bus controller.

The timing diagram for 8086 maximum mode memory write operation is shown in Figure 3.18.

The control signal logic levels and timing diagram are similar to that of read operation, except for data transmit and receive, memory read and write signals.

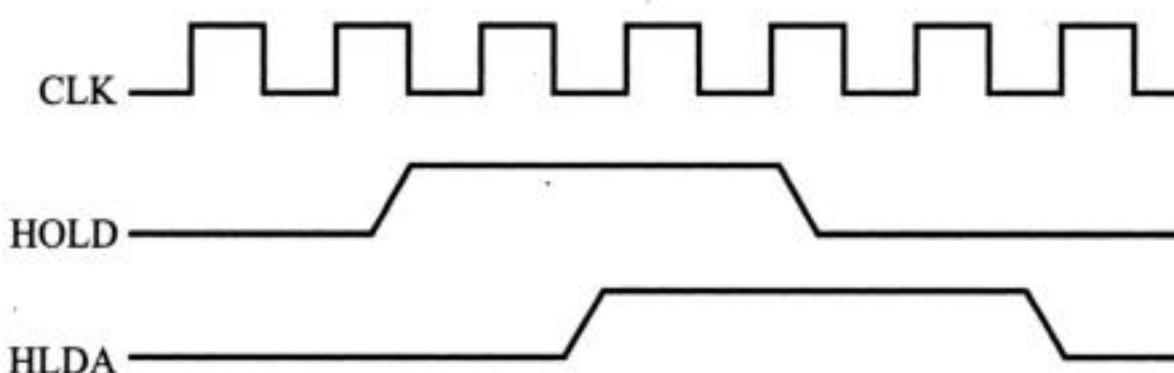
Here the T-states correspond to the time during which DEN<sup>-</sup> is low, WRITE control goes low, DT/R<sup>-</sup> is high and data output is available from the processor on the data bus.



**Figure 3.18** Maximum mode memory write bus cycle of 8086 system.

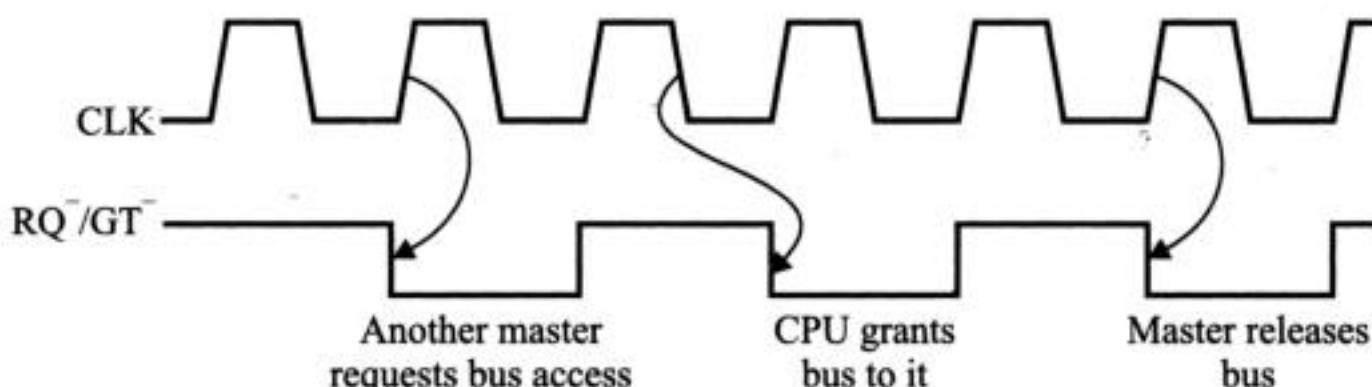
### 3.4.3 Bus Request and Bus Grant Timings in Minimum and Maximum Mode Systems

In the minimum mode configuration the request for the buses is made on the HOLD line of the processor. The microprocessor outputs the HOLD acknowledgement signal on the HLDA line. When a competent bus master required the buses it first sends the request in response of that the processor sends a logic high signal on the HLDA line. Figure 3.19 shows the response of HOLD and HLDA with respect to clock.



**Figure 3.19** Bus request and bus grant timings in minimum mode system.

In maximum mode the bus request as well as grant is performed by the same pin, i.e.  $RQ^-/GT^-$ . Here both the request and the grant signals are active low. During a bus request and grant cycle, first the Request signal which is input to processor goes low. At the next falling edge of the clock, the processor grants the request by sending a low on the request/grant pin. Figure 3.20 shows the response of  $RQ^-/GT^-$  with respect to clock.



**Figure 3.20** Bus request and bus grant timings in maximum mode system.

## EXERCISES

### Multiple Choice Questions

- What is the name of the time delay in a CPU caused by differences between the speed of the CPU, the system bus, and memory circuits?
  - Wait cycle
  - Wait state
  - Dead cycle
  - Memory write cycle.
- List the three primary steps of the instruction execution cycle, in sequential order:
  - Fetch, decode, memory write
  - Fetch, memory read, execute
  - Decode, fetch, execute
  - Fetch, decode, execute.
- In 8086, the example for Non-maskable interrupts is
  - Trap
  - RST6.5
  - INTR
  - NMI.
- What does microprocessor speed depends on?
  - Clock
  - Data bus width
  - Address bus width.

5. Can ROM be used as stack?  
(a) Yes (b) No  
(c) Sometimes yes, sometimes no.
6. Which processor structure is pipelined?  
(a) All 80 processors (b) All 85 processors  
(c) All 86 processors.
7. In 8086 the overflow flag is set when  
(a) The sum is more than 16-bits  
(b) Signed numbers go out of their range after an arithmetic operation  
(c) Carry and sign flags are set  
(d) During subtraction.
8. BHE<sup>-</sup> of 8086 microprocessor signal is used to interface the  
(a) Even bank memory (b) Odd bank memory  
(c) I/O (d) DMA.
9. In 8086 microprocessor one of the following statements is not true.  
(a) Coprocessor is interfaced in MAX mode  
(b) Coprocessor is interfaced in MIN mode  
(c) I/O can be interfaced in MAX/MIN mode  
(d) Supports pipelining.
10. 8088 microprocessor differs from 8086 microprocessor in  
(a) Data width on the output (b) Address capability  
(c) Support of coprocessor (d) Support of MAX/MIN mode.
11. Which is not the input signal to the bus controller 8288?  
(a) CLK (b) CEN  
(c) IOB (d) DEN.
12. Which is not the output signal of the bus controller 8288?  
(a) ALE (b) DT/R<sup>-</sup>  
(c) IOB (d) DEN.
13. How many operating modes are there of bus controller 8288?  
(a) 2 (b) 3  
(c) 4 (d) 5.
14. The bus controller 8288 works in which of the operating modes  
(a) IO bus mode (b) System bus mode  
(c) Both of these (d) None of these.
15. How many functional blocks are there in bus controller 8288?  
(a) 2 (b) 3  
(c) 4 (d) 5.
16. The 8282 chip is used as  
(a) Bus controller (b) Data latch  
(c) Clock generator (d) Transceiver.

## **Descriptive Questions**

1. What are the contents of the data bus and the status of  $A_0$  and BHE when the following instructions are executed in 8086?
    - (a) CPU writes a byte 11H at memory location 1000H : 0002H.
    - (b) CPU writes a word 2211H at memory location 1000H : 0003H.
  2. What is the purpose of ALE, BHE<sup>-</sup>, DT/R<sup>-</sup> and DEN<sup>-</sup> pins of 8086? Show their timing in the system bus cycle of 8086.
  3. Sketch and explain the 8086 bus activities during read machine cycle.
  4. What is the difference between system bus cycle and bus idle cycle? Draw the timing diagram of bus idle cycle.
  5. Draw and explain a block diagram showing 8086 in maximum mode configuration.
  6. Draw typical 8086 minimum mode configuration and explain the function of two signals used in minimum mode.
  7. Draw the maximum mode module of 8086 showing clearly address latches, transceivers, and clock generator. Neatly label the diagram. Terminate unused pins properly.
  8. Design the required logic to generate read, write control signals for memory and I/O in a system using 8086 microprocessor. Generate bank select signals for even and odd address memory banks.
  9. Draw the block diagram of the clock generator 8284.
  10. Draw and explain the interfacing of the clock generator 8284 with 8086.

11. Draw and explain the timing diagram for 8086 minimum mode memory write cycle.
12. Draw and explain the timing diagram for 8086 maximum mode memory write cycle
13. Draw and explain the timing diagram for 8086 minimum mode memory read cycle.
14. Draw and explain the timing diagram for 8086 maximum mode memory read cycle.
15. Draw the functional block diagram of 8288.
16. Explain the command signals and control signals of 8288.
17. Explain the concept of interfacing memory with 8086 in minimum mode. Draw a neat sketch of interfacing of memory and the processor.
18. Explain the concept of interfacing memory with 8086 in maximum mode. Draw a neat sketch of interfacing of memory and the processor.

# 4

## Instructions Set of 8086

### 4.1 INTRODUCTION

Program is a set of instructions written to solve a problem. Instructions are the directions which a microprocessor follows to execute a task or part of a task.

Broadly computer language can be divided into two parts as high-level language and low-level language. Low-level languages are machine specific. Low-level language is further divided into machine language and assembly language.

Machine language is the only language which a machine can understand. Instructions in this language are written in binary bits as a specific bit pattern. The computer interprets this bit pattern as an instruction to perform a particular task. The entire program is a sequence of the binary numbers. This is a machine-friendly language but not user friendly. Debugging is another problem associated with machine language.

To overcome these problems, programmers develop another way in which instructions are written in English alphabets. This new language is known as Assembly language. The instructions in this language are termed *mnemonics*. As microprocessor can only understand the machine language so mnemonics are translated into machine language either manually or by a program known as *assembler*.

Efficient software development for the microprocessor requires a complete familiarity with the instruction set, their format and addressing modes. This chapter is devoted to develop an understanding regarding the addressing modes and instruction formats of all the instructions of microprocessor 8086.

This chapter can be divided into three parts. The first part deals with the addressing modes of 8086. The second part discusses the instruction formats and instruction templates of 8086. Finally the third part covers the instructions of 8086 microprocessor.

### 4.2 ADDRESSING MODES OF 8086

The addressing modes are the ways of specifying an operand in an instruction. In 8086 the addressing modes are broadly categorized into two groups, i.e. data addressing modes and

address addressing modes. Data addressing modes are for defining a data operand in the instruction whereas address addressing modes are the ways of specifying a branch address in control transfer instructions.

#### 4.2.1 Data Addressing Modes

The 8086 microprocessor introduces many new techniques to access the memory by introduction of many more types of addressing modes. With these new memory related addressing modes, it can access memory in many different ways. These addressing modes provide flexibility to the processor to access memory, which in turn allows the user to access variables, arrays, records, pointers, and other complex data types in a more flexible manner. Mastery of the 8086 addressing modes is the first step towards the understanding of 8086 assembly language.

The microprocessor 8086 has all the five data addressing modes available in 8085, i.e. implied, register, immediate, direct and register indirect. The register indirect addressing mode in 8086 works with SI, DI, BX and BP registers. Apart from these data addressing modes, 8086 has five more addressing modes. These are:

1. Base addressing mode
2. Index addressing mode
3. Based indexed addressing mode
4. Based indexed with displacement addressing mode
5. String addressing mode

Different addressing modes may take differing amounts of time to compute the effective address. Complex addressing modes take more time to compute the effective address than the simpler addressing modes. Complexity of an addressing mode will go on increasing with the number of terms in the addressing mode. For example, [BX] [DI] is more complex than [DI]. Similarly disp [BX] [DI] is more complex than [BX] [DI].

The displacement in all the memory-related addressing modes can be a signed 8-bit constant or a signed 16-bit constant. For 8-bit displacement the offset is in the range of -128 ... +127 and the instruction will be shorter and faster as compared to the instructions which uses the 16-bit signed displacement. It is always preferable to use a small displacement (8-bit) and a large number in the register(s) instead of using large displacement (16-bit) and small value in the register(s) because the size of the value in the register does not affect the execution time or size.

When the BIU of microprocessor calculates the effective address, and finds that the address sum is more than 16-bit, i.e. address value is greater than FFFFH, then the microprocessor discards the overflow and the result wraps around back to zero. For example, if DI contains 1000H, then the instruction MOV CL, 0FFFFH [DI] will load the CL register from location DS: 1000H.

##### ***Immediate addressing mode***

In immediate addressing mode the operands are specified within the instruction itself. The immediate operand can only be the source operand. For example,

MOV AX, 2500H

Here the immediate data is 2500H.

### **Register addressing mode**

Most 8086 instructions can operate on the 8086's general purpose register set. The content of a register can be accessed by specifying the name of the register as an operand to the instruction. For example, the following MOV instruction of 8086 copies the data from the source operand to the destination operand.

```
MOV      AX, BX
MOV      DL, AL
MOV      SI, DX
```

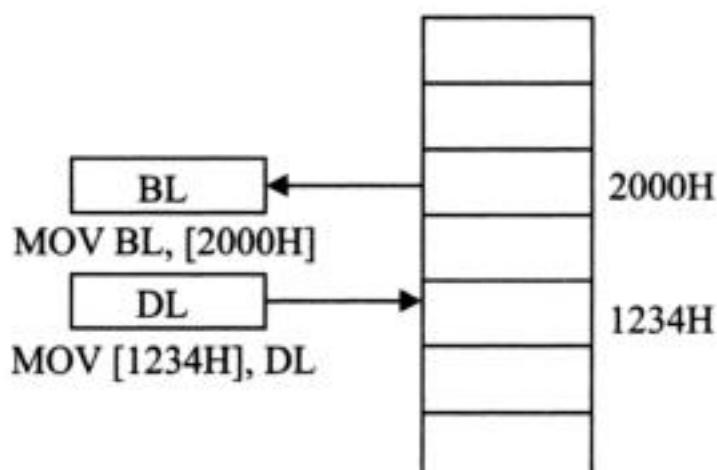
The 8- and 16-bit registers are the valid operands for this instruction. The only restriction is that both operands must be of the same size.

The registers are the best place to keep often used variables. Instructions using the registers are shorter and faster than those that access memory.

Segment registers can never be used as data registers to hold arbitrary values. They should only contain segment addresses.

### **Direct addressing mode or the displacement only addressing mode**

Direct addressing mode displacement only addressing mode may be defined as the addressing mode in which the address of the memory is specified in the instruction itself. In this addressing mode the instruction consists of a 16-bit memory address or an 8-bit IO address. The 16-bit memory address is always written inside the square brackets. For example, the instruction `MOV BL, [2000H]`, transfers the content of the memory location `2000H` in the `BL` register. Similarly, the instruction `MOV [1234H], DL` transfers the content of the `DL` register in the memory location specified by `1234H`. Figure 4.1 shows the direct addressing mode.



**Figure 4.1** Direct addressing mode.

By default, all the direct addressing mode point in the data segment. The segment override prefix is to be used before address if we have to point any other memory segment. For example, to access location `4321H` in the extra segment `ES` the instruction will be of the form `MOV AX, ES: [4321H]`. Similarly, to access this location in the code segment, the instruction will be `MOV AX, CS: [4321H]`.

### **Register indirect addressing modes**

This addressing mode is also used in concern with memory and IO. In this addressing mode, the memory address is specified by some pointer, index or base registers. These registers are

written inside the square brackets. There are four forms of this addressing mode on the 8086, best demonstrated by the following instructions:

```
MOV    DX, [BX]
MOV    DX, [BP]
MOV    DX, [SI]
MOV    DX, [DI]
```

These four addressing modes refer the word at the offset found in the BX, BP, SI or DI registers, respectively. The [BX], [SI], and [DI] modes use the DS segment by default. The [BP] addressing mode uses the stack segment (SS) by default.

To access data from other than the default segment, the segment override prefix symbols are to be used. The following instructions demonstrate the use of these overrides:

```
MOV    AX, CS:[BX]
MOV    AX, DS:[BP]
MOV    AX, SS:[SI]
MOV    AX, ES:[DI]
```

### **Base addressing mode**

In this mode 8-bit or 16-bit displacement is added to the contents of a base register (BX or BP); the resulting value is a pointer to location where data resides. In this addressing mode, the memory location is calculated by adding the signed 8-bit or 16-bit displacement to either BX or BP register.

$$\text{Memory location} = \begin{cases} \text{BX} & \text{8-bit displacement} \\ \pm & \\ \text{BP} & \text{16-bit displacement} \end{cases}$$

For example if BX = 2000H, the instruction is

```
MOV    AL, [BX + 15]
```

In this example, the contents of the memory location 200FH (2000H + 0FH (equivalent to decimal 15)) is transferred to AL register. The maximum 8-bit displacement can be  $\pm 127$  and the maximum 16-bit displacement can be  $\pm 32767$ .

There are four possible combinations of the base addressing modes, i.e.

$$\text{Memory location} = \begin{array}{l} \text{BX} \pm \text{8-bit displacement} \\ \text{BX} \pm \text{16-bit displacement} \\ \text{BP} \pm \text{8-bit displacement} \\ \text{BP} \pm \text{16-bit displacement} \end{array}$$

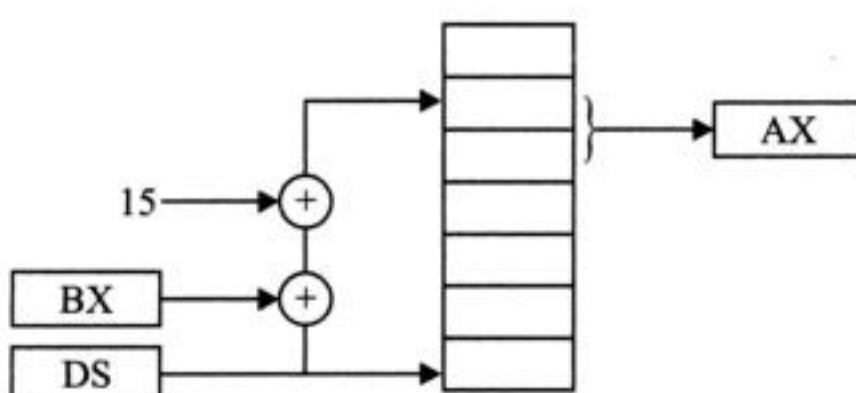
The displacement can also be written as

```
MOV    AL, DISP [BX]
```

The addressing modes involving BX, use the data segment, the addressing mode involving [BP] uses the stack segment by default. As with the register indirect addressing modes, the segment override prefixes can be used to specify a different segment:

MOV AL, SS: DISP [BX]  
 MOV AL, ES: DISP [BP]

Figure 4.2 shows how the offset address will be calculated for the instruction `MOV AX, [BX + 15]`.



**Figure 4.2** Calculation of offset address in base addressing mode.

### ***Index addressing mode***

This addressing mode is similar to base addressing mode with the difference that in this mode the 8-bit or 16-bit displacement is added to the contents of an index register (SI or DI). In this addressing mode, the memory location is calculated by adding the signed 8-bit or 16-bit displacement to either SI or DI register.

$$\text{Memory location} = \begin{cases} \text{SI} & \text{8-bit displacement} \\ \pm & \\ \text{DI} & \text{16-bit displacement} \end{cases}$$

There are four possible combinations of the base addressing modes, i.e.

$$\text{Memory location} = \begin{array}{l} \text{SI} \pm \text{8-bit displacement} \\ \text{SI} \pm \text{16-bit displacement} \\ \text{DI} \pm \text{8-bit displacement} \\ \text{DI} \pm \text{16-bit displacement} \end{array}$$

Data segment is the default segment for this addressing mode. As with the register indirect and base addressing modes, the segment override prefixes can be used to specify a different segment:

MOV AL, CS: DISP [SI]  
 MOV AL, SS: DISP [DI]

The offset address in this addressing mode will be calculated as shown in Figure 4.2 by replacing the BX contents by SI or DI contents.

Note that Intel still refers to these addressing modes as based addressing and indexed addressing. Intel's literature does not differentiate between these modes with or without the constant.

### **Based indexed addressing mode**

The based indexed addressing modes are simply the combinations of the register indirect addressing modes. In based indexed addressing mode, the contents of a base register (BX or BP) are added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.

$$\text{Memory location} = \begin{pmatrix} \text{SI} & \text{BX} \\ & + \\ \text{DI} & \text{BP} \end{pmatrix}$$

The allowable forms for these addressing modes are:

MOV AL, [BX][SI]  
 MOV AL, [BX][DI]  
 MOV AL, [BP][SI]  
 MOV AL, [BP][DI]

For example, if BX = 2000H and SI = 5400H, the instruction is

MOV AL, [BX + SI]

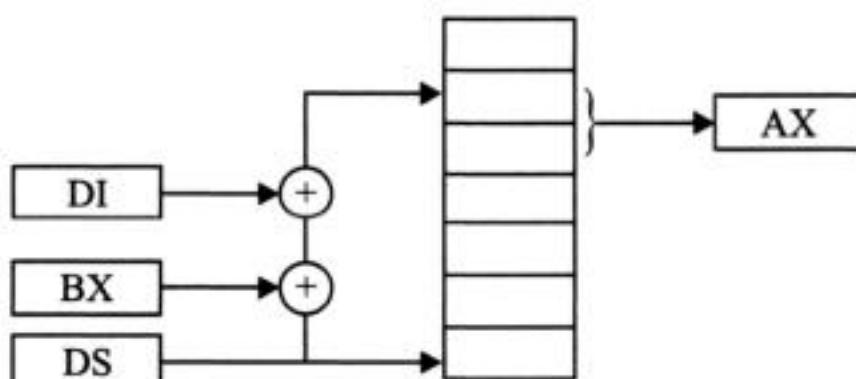
In this example the contents of the memory location 7400H (2000H + 5400H) is transferred to AL register.

The addressing modes that do not involve bp use the data segment by default. Those that have bp as an operand use the stack segment by default.

There are four possible combinations of the base addressing modes, i.e.

$$\text{Memory location} = \begin{matrix} \text{SI} + \text{BP} \\ \text{SI} + \text{BX} \\ \text{DI} + \text{BP} \\ \text{DI} + \text{BX} \end{matrix}$$

Figure 4.3 shows how the offset address will be calculated for the instruction MOV AX, [BX + DI].



**Figure 4.3** Calculation of offset address in base index addressing mode.

### **Based indexed with displacement addressing mode**

In this addressing mode, the offset address is generated by the sum of Base register and Index registers along with 8-bit or 16-bit displacement. In this addressing mode, 8-bit or 16-bit

displacement is added to the contents of a base register (BX or BP) and index register (SI or DI); the resulting value is a pointer to location where data resides.

$$\text{Memory location} = \begin{pmatrix} \text{SI} & \text{BX} & \text{8-bit displacement} \\ + & \pm & \\ \text{DI} & \text{BP} & \text{16-bit displacement} \end{pmatrix}$$

In this addressing mode the memory location is calculated by adding the signed 8-bit or 16-bit displacement to the sum of the content of SI + BP or SI + BX or DI + BP or DI + BX.

Considering the same example, i.e. if BX = 2000H and SI = 5400H, the instruction is

`MOV AL, [BX + SI + 15]`

In this example, the contents of the memory location 740FH (2000H + 5400H + 0FH (equivalent to decimal 15)) is transferred to AL register. Again the maximum 8-bit displacement can be  $\pm 127$  and the maximum 16-bit displacement can be  $\pm 32767$ .

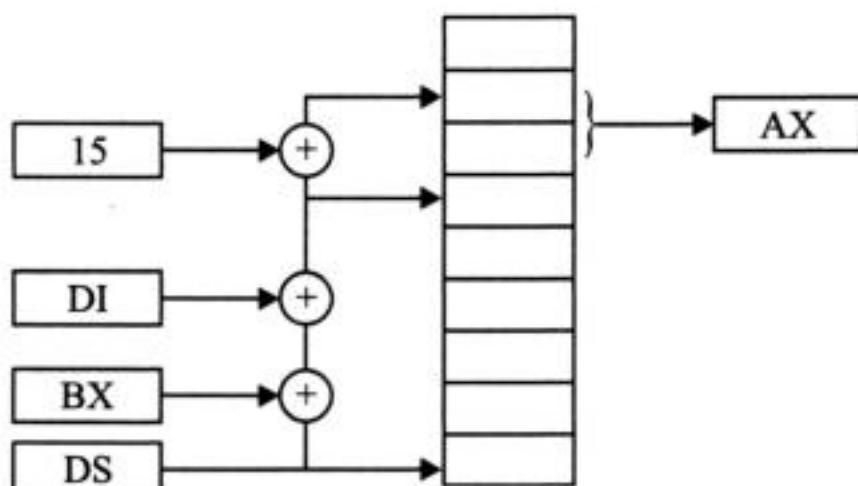
There are eight possible combinations of the base index with displacement addressing mode, i.e.

$$\begin{aligned} &\text{SI} + \text{BP} \pm \text{8-bit displacement} \\ &\text{SI} + \text{BX} \pm \text{8-bit displacement} \\ &\text{DI} + \text{BP} \pm \text{8-bit displacement} \\ \text{Memory location} = &\text{DI} + \text{BX} \pm \text{8-bit displacement} \\ &\text{SI} + \text{BP} \pm \text{16-bit displacement} \\ &\text{SI} + \text{BX} \pm \text{16-bit displacement} \\ &\text{DI} + \text{BP} \pm \text{16-bit displacement} \\ &\text{DI} + \text{BX} \pm \text{16-bit displacement} \end{aligned}$$

The following are some of the examples of these addressing modes:

`MOV AL, DISP [BX][SI]`  
`MOV AL, DISP [BX + DI]`  
`MOV AL, [BP + SI + DISP]`  
`MOV AL, [BP][DI][DISP]`

Figure 4.4 shows how the offset address will be calculated for the instruction `MOV AX, [BX + DI + 15]`.



**Figure 4.4** Calculation of offset address in relative base index addressing mode.

### String addressing modes

This mode uses index registers. The string instructions automatically assume SI to point to the first byte or word of the source operand and DI to point to the first byte or word of the destination operand.

The segment register for the source is DS and may be overridden. The segment register for the destination must be ES and cannot be overridden.

The contents of SI and DI are automatically incremented by clearing DF (Direction Flag) to 0 by CLD instruction or automatically decremented by setting DF to 1 by STD instruction.

Table 4.1 summarizes all the 32 possible data addressing modes of 8086.

**Table 4.1** Summary of data addressing modes

(BX) + (SI)	(BX) + (SI) + d8	(BX) + (SI) + d16	AL	AX
(BX) + (DI)	(BX) + (DI) + d8	(BX) + (DI) + d16	CL	CX
(BP) + (SI)	(BP) + (SI) + d8	(BP) + (SI) + d16	DL	DX
(BP) + (DI)	(BP) + (DI) + d8	(BP) + (DI) + d16	BL	BX
(SI)	(SI) + d8	(SI) + d16	AH	SP
(DI)	(DI) + d8	(DI) + d16	CH	BP
d16	(BP) + d8	(BP) + d16	AH	SP
(BX)	(BX) + d8	(BX) + d16	BH	DI

### 4.2.2 Address Addressing Modes

These addressing modes indicate the branch addresses in the call and jump instructions. In 8086 there are four types of address addressing modes, i.e. intrasegment direct, intrasegment indirect, intersegment direct and intersegment indirect. In intrasegment the branching is within the segment and in intersegment the branching is outside the segment. Intersegment is a synonym for far, intrasegment is a synonym for near.

#### Intrasegment direct

In this addressing mode the effective branch address will be the sum of the signed 8- or 16-bit displacement and the content of the IP. When the displacement is of 8-bit, then this is referred to as short jump or short call, otherwise long jump or long call. For example, consider the following two jump instructions

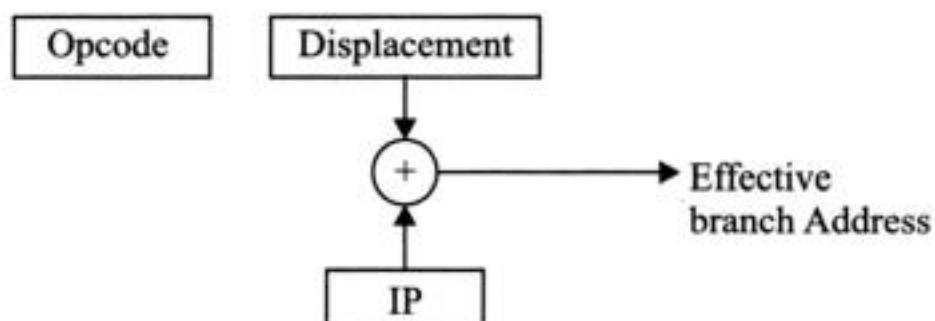
JMP 8-bit displacement; direct intrasegment, 8-bit displacement  
JMP 16-bit displacement; direct intrasegment, 16-bit displacement

These two forms of the direct intrasegment jump are the same except their length. In the case of JMP 8-bit displacement instruction, the microprocessor sign extends this 8-bit displacement into 16-bits and then add it to the IP register to point the jump location. With the help of this instruction, microprocessor can transfer the control either to a branch location to 128 locations before the JMP instruction location or to 127 locations ahead of the JMP instruction. Or, we can say that with 8-bit displacement the control can be transferred within a memory location space of -128 to +127. The JMP 8-bit displacement is a two-byte long instruction.

The second form of the intrasegment jump is three bytes long with the first byte as the opcode and the next two bytes as displacement. This instruction allows a memory range of

-32,768 to +32,767 locations. This instruction can transfer control to anywhere in the current code segment. The microprocessor adds the two byte displacement to the IP register contents to point to the jump location.

The 8-bit displacement can be used with conditional as well as unconditional call and jump instructions but the 16-bit displacement can be used only in unconditional instructions. Figure 4.5 shows a graphical means to compute the effective address of the branched address.



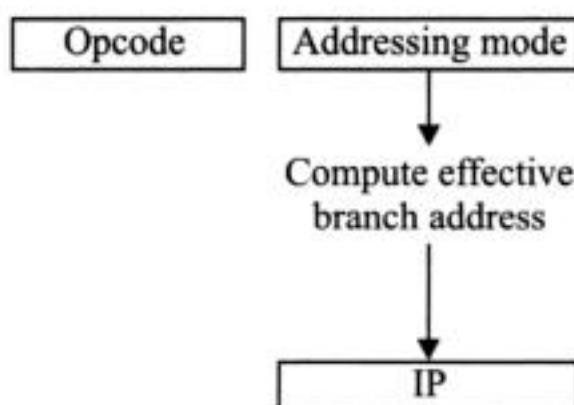
**Figure 4.5** Calculation of the EA of the branched address in intrasegment direct addressing mode.

### Intrasegment indirect

In this addressing mode the effective address is specified by any of the register or memory contents. The memory location can be specified by any of the memory related data addressing modes. In this addressing mode the content of the IP is replaced by the effective branch address. This addressing mode is used only for unconditional branch instructions. Consider the example

JMP Disp [BX]; Disp is an array of words

This addressing mode fetches the word from location disp + BX and copies this value to the IP register. Figure 4.6 shows a graphical means to compute the effective address of the branched address.

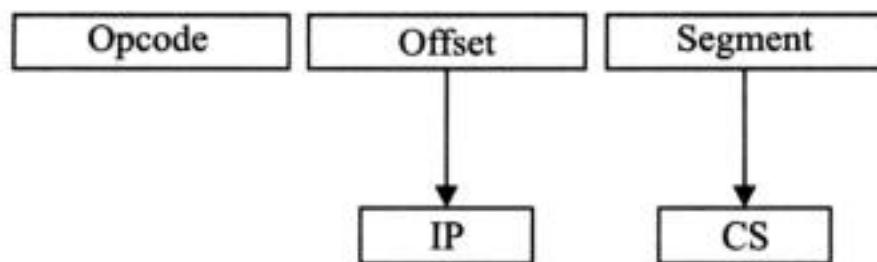


**Figure 4.6** Calculation of the EA of the branched address in intrasegment indirect addressing mode.

### Intersegment direct

In this addressing mode the contents of IP are replaced by a part of the instruction and the contents of the CS is replaced by another part of the instruction. The 32-bit operand is loaded into the IP and CS register. The direct intersegment jump or call is five bytes long, the last four bytes containing a segmented address (the offset in the second and third bytes, the segment in the fourth and fifth bytes). This instruction copies the offset into the IP register and the segment into the CS register. Execution of the next instruction continues at the new address in CS:IP. The address following the opcode is the absolute memory address of the target instruction. This instruction loads CS:IP with a 32-bit immediate value. Intersegment

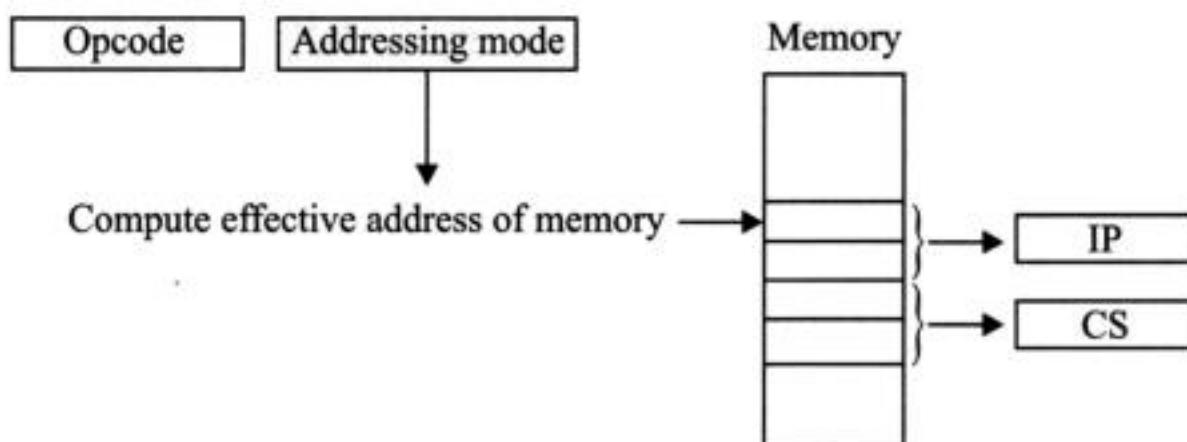
Direct does not use relative addressing. Figure 4.7 shows a graphical means to compute the effective address of the branched address.



**Figure 4.7** Calculation of the EA of the branched address in intersegment direct addressing mode.

### Intersegment indirect

In this addressing mode the contents of IP and CS are replaced by the contents of the four consecutive memory locations pointed by any of the memory-related data addressing modes except the immediate and register addressing mode. Figure 4.8 shows a graphical means to compute the effective address of the branched address.



**Figure 4.8** Calculation of the EA of the branched address in intersegment indirect addressing mode.

### EXAMPLE 4.1 Given that

$$BX = 2500H \quad SI = 5000H \quad \text{Displacement} = 1000H \quad IP = 2000H$$

Determine the effective address for the following addressing modes:

- |                                 |                             |              |
|---------------------------------|-----------------------------|--------------|
| (i) Immediate                   | (ii) Register using SI      | (iii) Direct |
| (iv) Register indirect using BX | (v) Base                    | (vi) Index   |
| (vii) Base index                | (viii) Base index relative. |              |

### Solution

- (i) For immediate addressing mode effective address is the content of the IP, so  $EA = 2000H$ .
- (ii) For register addressing using SI, the  $EA = 5000H$
- (iii) For direct addressing using the displacement, the  $EA = 1000H$
- (iv) For register indirect addressing using BX, the  $EA = 2500H$
- (v) For base addressing using displacement, the  $EA = 2500H + 1000H = 3500H$
- (vi) For index addressing using displacement, the  $EA = 5000H + 1000H = 6000H$
- (vii) For base index addressing the  $EA = BX + SI = 2500H + 5000H = 7500H$
- (viii) For base index relative addressing the  $EA = BX + SI + Disp = 2500H + 5000H + 1000H = 8500H$ .

**EXAMPLE 4.2** Find the branch address for the following address addressing modes:

- (i) Intrasegment direct
- (ii) Intrasegment indirect using BX register
- (iii) Intrasegment indirect using base addressing
- (iv) Intrasegment direct
- (v) Intrasegment indirect using BX register using relative addressing

Assume the following:

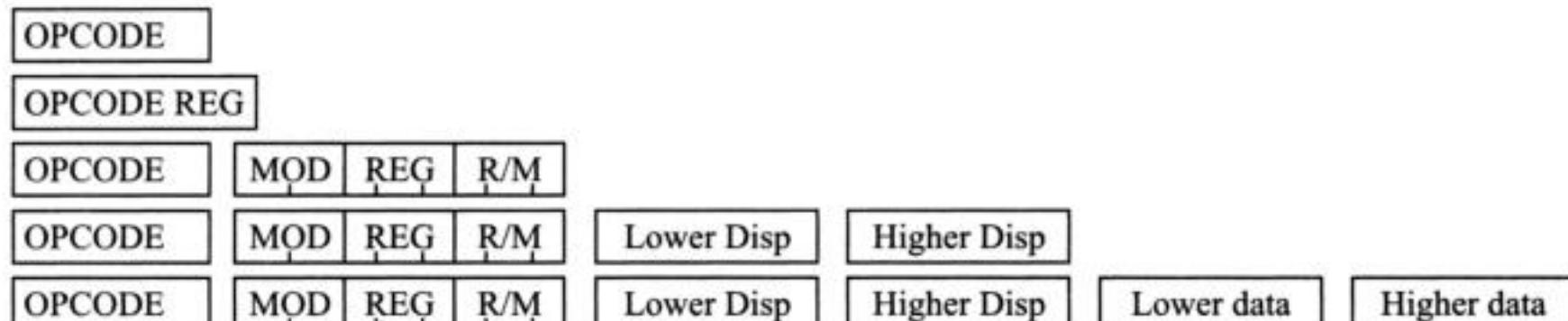
IP = 2500H CS = 2000H BX = 3000H DS = 4000H Immediate value (16-bit) = 1500H.  
Immediate value (32-bit) = 1500:3500H [43000H] = 40FD3598H [44500H] = 05F6H.

### Solution

- (i) For intrasegment direct addressing mode, the branch address will be the sum of the 16-bit immediate value present in the instruction and the content of IP.  
So branch address = 1500H + 2500H = 4000H.
- (ii) For intrasegment indirect addressing mode, using BX register and register addressing, the branch address will be the value present in the BX register.  
So branch address = 3000H.
- (iii) For intrasegment indirect addressing mode, using base addressing, the branch address will be the value present in the memory location pointed by [DS:BX] plus the displacement.  
So branch address = [DS \* 10H + BX + Disp] = [44500H] = 05F6H.
- (iv) For intersegment direct addressing mode the branch address will be the immediate value present in the instruction. The first word of the immediate value will go to the IP and the remaining will go to the CS register.  
So branch address will be IP = 1500H and CS = 3500H.
- (v) For intersegment indirect addressing mode using BX register and register addressing the branch address will be the content of the memory location pointed by the [DS:BX] register. The first word will go to the IP and the second will go to the CS register.  
So branch address = [DS \* 10H + BX] = [43000H] = 40FD3598H  
So IP = 40FDH and CS = 3598H.

### 4.3 INSTRUCTION FORMAT

The instructions of 8086 may be one to six byte long. These instructions have different formats. Figure 4.9 shows some of the instruction formats.



**Figure 4.9** Examples of Instruction format.

The first byte always consists of the opcode. The opcode may be of 8-bit or may occupy MSB six bits of the first bytes and it defines the operation to be carried out by the instruction. The remaining two bits are any of the following bits.

1. Direction bit (D) defines whether the register operand in byte 2 is the source or destination operand.  
 D = 1 specifies that the register operand is the destination operand.  
 D = 0 indicates that the register is a source operand.
2. Data size bit (W) defines whether the operation to be performed is an 8 bit or 16 bit data.  
 W = 0 indicates 8-bit operation  
 W = 1 indicates 16-bit operation
3. Sign bit (S) is used for sign extension of an 8-bit 2's compliment number to a 16-bit 2's compliment number. This is done by making all the bits in high-order byte same as that of MSB in the lower-order byte. This bit appears with the W bit in add, subtract and compare instructions. For 8-bit operation S:W bits are 00 and these bits are 01 for 16-bits operation with 16-bit immediate operand.  
 S:W bits are 11 for 16-bit operation with a sign-extended 8-bit immediate operand.
4. V-bit is used in shift and rotate instruction to determine the number of shifts.  
 V = 0 indicates that the shift count is 1  
 V = 1 indicates that CL register contains the shift count
5. The Z-bit is used in REP instruction. The Z-bit is matched with the zero flag bit. The REP instruction goes on executing till the Z-bit does not match with the zero flag.

A summary of these bits encoding is shown in Table 4.2.

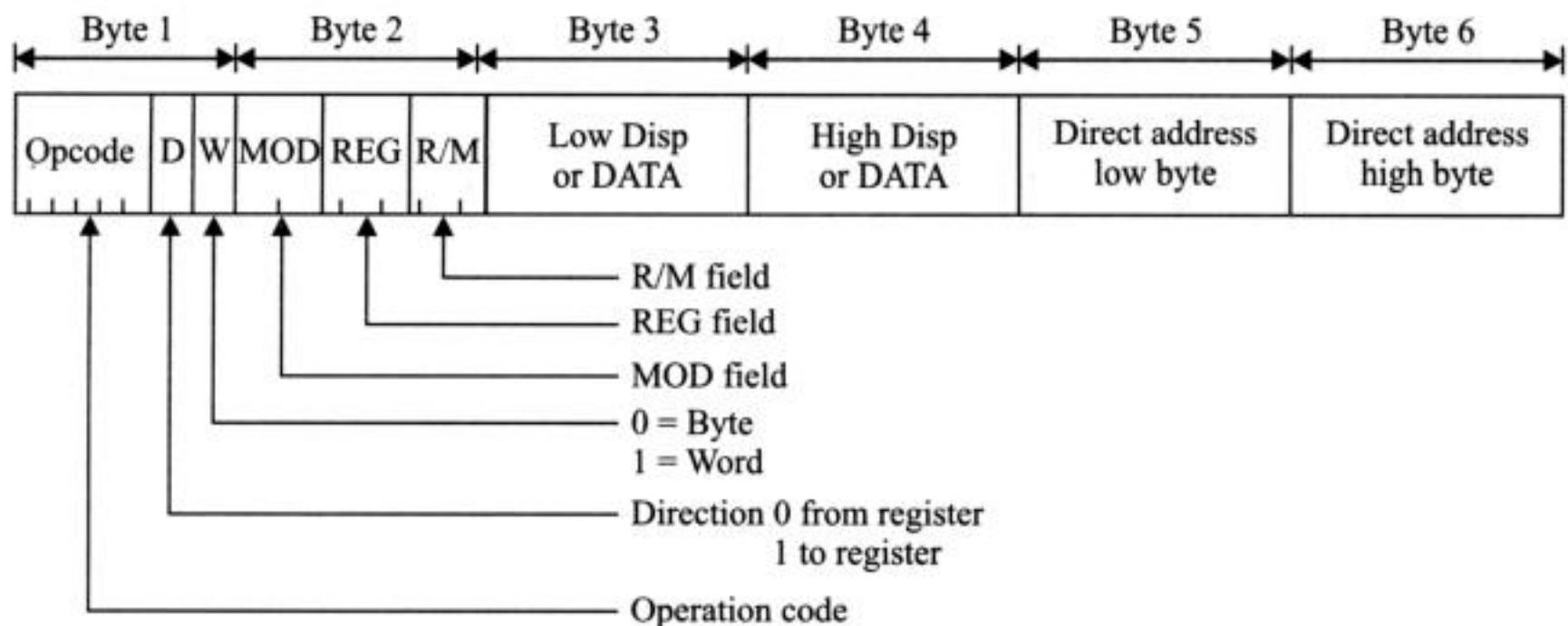
**Table 4.2** Single-bit field encoding

Field	Value	Function
S	0	No sign extension
	1	Sign extend 8-bit immediate data to 16-bits if W = 1
W	0	Instruction operates on byte data
	1	Instruction operates on word data
D	0	Instruction source is specified in REG field
	1	Instruction destination is specified in REG field
V	0	Shift/rotate count is one
	1	Shift/rotate count is specified in CL register
Z	0	Repeat/loop while zero flag is clear
	1	Repeat/loop while zero flag is set

As shown in Figure 4.9, depending on the instruction, the opcode byte may be the only byte in the instruction or may be followed by

- One or two byte long immediate data
- One or two byte long displacement
- One or two byte long displacement and then one or two byte long immediate data
- Two byte long direct address
- Two byte long displacement and then two byte long segment address

The presence of these additional bytes depends on the opcode byte of the instruction. Figure 4.10 shows an instruction format of MOV instruction.



**Figure 4.10** Instruction template for the MOV instruction.

#### 4.4 INSTRUCTION TEMPLATES

8085 has total 246 opcodes. These opcodes can be printed on a single paper sheet but such is not the case with 8086 microprocessor.

For example, consider the case of the MOV AX, Source instruction. As there are altogether 32 different addressing modes so the source can be specified by any of those 32 ways. Similarly, if AX becomes the source, then again there will be 32 different ways to specify the destination. Hence there are 64 opcodes for MOV instruction involving AX register at source and destination location. Still there will be another 64 opcodes if AL is used and 64 more opcodes can be generated by using AH register. In a nutshell there will be 192 opcodes only for MOV instruction involving AX, AL, and AH registers.

Microprocessor 8086 has about 13000 opcodes which require 60 pages to tabulate these opcodes.

It becomes very tedious to find out the opcodes of the 8086 instructions from a book of 60 pages. So instruction templates are used for each basic instruction to generate the opcodes by filling the bits in the templates corresponding to those instructions. In other words, the opcodes are generated on a bit by bit basis. Figure 4.10 shows the instruction template for the MOV instruction to transfer data between registers or between register and memory location specified by any of the addressing mode.

In the MOV instruction template the MSB six bits of the first byte is of the op code and the LSB two bits are D and W bits. The second byte of the instruction usually identifies whether one of the operands is in memory or whether both are registers. This byte contains 3 fields. These are the mode (MOD) field, the register (REG) field and the Register/Memory (R/M) field.

These three fields are encoded as per Table 4.3.

**Table 4.3** Encoding of MOD and R/M field

MOD/R/M	MOD 00	MOD 01	MOD 10	MOD 11	
				W = 0	W = 1
000	(BX) + (SI)	(BX) + (SI) + d8	(BX) + (SI) + d16	AL	AX
001	(BX) + (DI)	(BX) + (DI) + d8	(BX) + (DI) + d16	CL	CX
010	(BP) + (SI)	(BP) + (SI) + d8	(BP) + (SI) + d16	DL	DX
011	(BP) + (DI)	(BP) + (DI) + d8	(BP) + (DI) + d16	BL	BX
100	(SI)	(SI) + d8	(SI) + d16	AH	SP
101	(DI)	(DI) + d8	(DI) + d16	CH	BP
110	d16	(BP) + d8	(BP) + d16	DH	SI
111	(BX)	(BX) + d8	(BX) + d16	BH	DI
← Memory Mode (EA Calculation) →				← Register Mode →	

In the second byte the MSB two bits  $D_7$  and  $D_6$  are defined as MOD field. The MOD field defines whether the R/M field is for register or memory and if it is for memory then is there no displacement, or an 8-bit displacement or a 16-bit displacement. These two bits are encoded as:

MOD = 00: R/M for memory with no displacement

MOD = 01: R/M for memory with 8-bit displacement

MOD = 10: R/M for memory with 16-bit displacement

MOD = 11: R/M for a register.

Register field occupies 3 bits  $D_3$ ,  $D_4$ , and  $D_5$ . This field defines the register for the first operand which is specified as source or destination by the D bit.  $D_3$ ,  $D_4$ , and  $D_5$  are encoded as per Table 4.4.

**Table 4.4** Encoding of REG field

REG	W = 0	W = 1
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

The R/M field occupies LSB 3 bits:  $D_0$ ,  $D_1$  and  $D_2$ . The R/M field along with the MOD field defines the second operand as shown in Table 4.5 for MOD = 11. In this table, for example, suppose the second operand is AX, then R/M field will be 000.

If the second operand is memory then the MOD field will be either 00 or 01 or 10 depending on how the memory is addressed. Table 4.6 shows the encoding of the R/M field along with the MOD field for MOD = 00, 01 and 10.

**Table 4.5** Encoding of R/M field for MOD = 11

R/M	W = 0	W = 1
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

**Table 4.6** Encoding of R/M field for MOD = 00, 01, 10

R/M	MOD 00	MOD 01	MOD 10
000	(BX) + (SI)	(BX) + (SI) + D <sub>8</sub>	(BX) + (SI) + D <sub>16</sub>
001	(BX) + (DI)	(BX) + (DI) + D <sub>8</sub>	(BX) + (DI) + D <sub>16</sub>
010	(BP) + (SI)	(BP) + (SI) + D <sub>8</sub>	(BP) + (SI) + D <sub>16</sub>
011	(BP) + (DI)	(BP) + (DI) + D <sub>8</sub>	(BP) + (DI) + D <sub>16</sub>
100	(SI)	(SI) + D <sub>8</sub>	(SI) + D <sub>16</sub>
101	(DI)	(DI) + D <sub>8</sub>	(DI) + D <sub>16</sub>
110	Direct address	(BP) + D <sub>8</sub>	(BP) + D <sub>16</sub>
111	(BX)	(BX) + D <sub>8</sub>	(BX) + D <sub>16</sub>

In the above, encoding of the R/M field depends on how the mode field is set. If MOD = 11 (register to register mode), this R/M identifies the second register operand. MOD selects memory mode, then R/M indicates how the effective address of the memory operand is to be calculated. Bytes 3 through 6 of an instruction are optional fields that normally contain the displacement value of a memory operand and/or the actual value of an immediate constant operand.

### Segment override prefix

Segment override prefix (SOP) is used when a default offset register is not used with its default base segment register, but with a different base register. In instruction format one byte is appended as suffix, for segment override before the opcode byte. The format of this byte is shown in Figure 4.11.

Format of the segment override instruction

0	0	1	S	R	1	1	0
---	---	---	---	---	---	---	---

Encoding of segment register

SR	Segment register
00	ES
01	CS
10	SS
11	DS

**Figure 4.11** Format of the segment override instruction.

In this format the SR indicates the segment register. The four segment register is encoded as shown in Figure 4.11. For example, to use DS as the new register in place of default register we have to use 3EH as a prefix byte. Table 4.6 shows the offset registers and their respective default segment registers and segment override prefix.

In Table 4.7 it is shown that IP and SP can never be associated with any other segment registers apart from their respective default segments. Similarly when DI is used as an implied memory pointer for string instructions, then it can only be used with its default segment register, i.e. ES. The other offset registers can be used with any of the segment registers using segment override prefix.

**Examples**    MOV AX, DS: [BP],  
              LODS ES: DATA 1

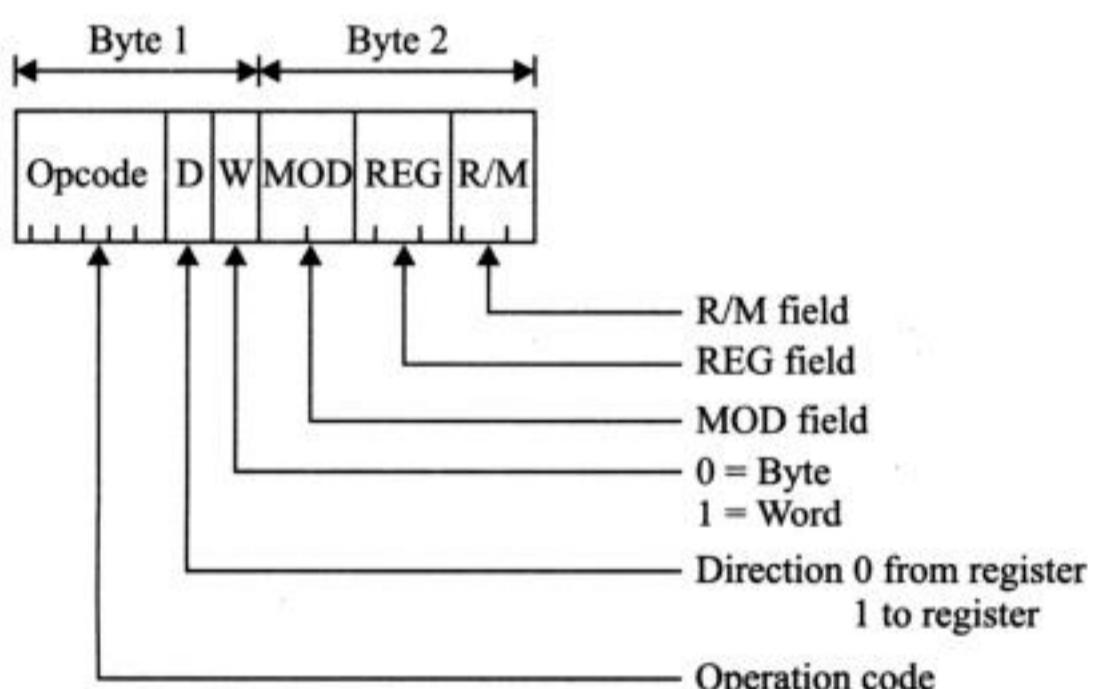
**Table 4.7** Probable override prefixes

Operand register	Default	With override prefix
IP (Code address)	CS	Never
SP (Stack address)	SS	Never
BP (Stack address)	SS	BP + DS or ES or CS
SI or DI (not including strings)	DS	ES, SS or CS
SI (Implicit source address for strings)	DS	ES, SS or CS
DI (Implicit destination address for strings)	ES	Never

**EXAMPLE 4.3** Construct the machine code for the instruction MOV BL, CH.

**Solution** This instruction copies the 8-bit content of CH to BL.

The instruction template of this instruction will be of two bytes only and shown in Figure 4.12.



**Figure 4.12** Template for MOV BL, CH.

The 6-bit opcode for this instruction is 100010.

Since this instruction transfer only 8-bit, hence W = 0 shows a byte operation.

D-bit indicates whether the register specified by the REG field of byte 2 is a source or destination operand.

D = 0 indicates CH is a source operand.

In byte 2, since the second operand is a register, MOD field will be 11.

The R/M field = 011 (BL)

Register (REG) field = 101 (CH)

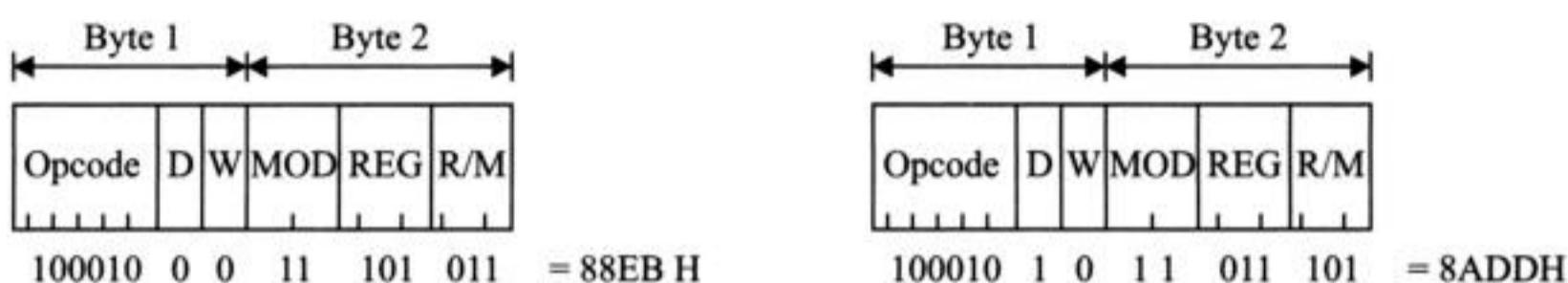
Hence the machine code for MOV BL, CH will be 88EBH.

The same instruction can be coded by another way by considering BL register in REG field. In this case the D bit is defined in concern with the BL register and now it will be D = 1 because the BL register is destination.

The R/M field = 101 (CH)

Register (REG) field = 011 (BL)

Hence the machine code for MOV BL, CH will be 8ADDH



**Figure 4.13** Machine code for MOV BL, CH.

The machine code for the instruction MOV BL, CH is shown in Figure 4.13. These two machine codes will perform the same function, i.e. MOV BL, CH.

**EXAMPLE 4.4** Construct the machine code for the instruction MOV 1234 [SI], AX.

**Solution** Here REG field will specify the AX register, the D bit must be 0, indicating the AX is the source register. The REG field must be 000 to indicate AX register. The W bit must be 1 to indicate it is a word operation. 1234 [SI] is specified using MOD value of 10 and R/M value of 100 and a displacement of 1234H. The 4-byte code for this instruction would be 89 8434 12H and is shown in Figure 4.14.

Opcode	D	W	MOD	REG	R/M	LB displacement	HB displacement
100010	0	1	10	000	100	34H	12H

**Figure 4.14** Machine code for the instruction MOV 1234 [SI], AX.

**EXAMPLE 4.5** Construct the machine code for MOV DS: 43 [BP], CX.

**Solution** Here we have to specify CX using REG field. The D bit must be 0, indicating that CX is the source register. The REG field must be 001 to indicate CX register. The W bit must be 1 to indicate it is a 16-bit operation. 43 [BP] is specified with MOD = 01 and R/M = 110 and displacement = 43H.

Whenever BP is used to generate the Effective Address (EA), the default segment would be SS. But here in this instruction segment override is used, i.e. the segment register DS is used. In such cases segment override prefix byte (SOP byte) is to be used. The SOP byte is 001 SR 110, where SR value is provided as per Table 4.8 shown below.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



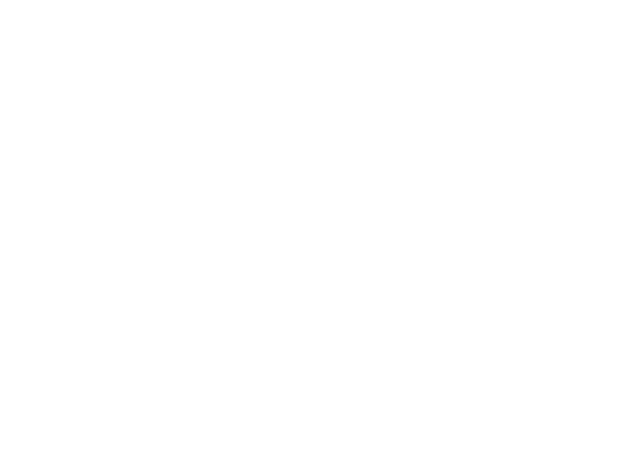
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



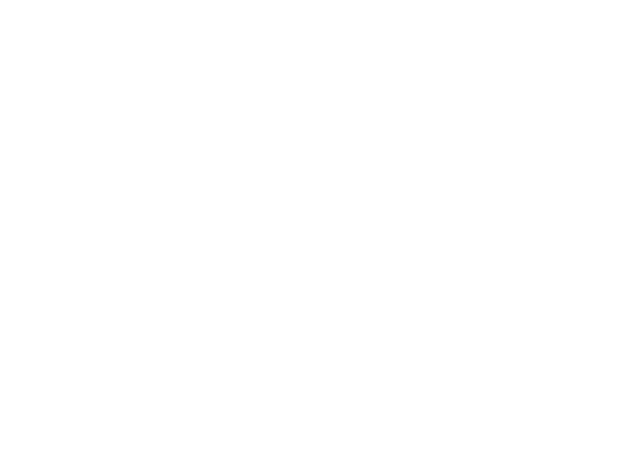
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

# Index

- 2-Key lockout mode, 521  
2-Key roll-over, 504  
8085  
    functional description, 9  
    pin description, 14  
8086  
    microprocessor, 27  
    minimum mode configuration, 50  
8086-based multiprocessing systems, 279  
8237 DMA controller, 484  
8251  
    control word, 314  
    interfacing in I/O mapped I/O, 325  
8255  
    in input mode of mode 1, 393  
    interfacing, 401
- AAA, 113  
AAD, 113  
AAM, 114  
AAS, 114  
Absolute address decoding, 366  
AC (alignment check) flag, 602  
AC (auxiliary carry) flag, 13  
Accumulator, 7, 12  
ACL approach, 597  
Active cycle, 492  
ADC, 98  
ADC0808, 426  
ADC0809, 426  
ADD, 98  
Address  
    addressing modes, 81  
    buffer register, 12  
    bus, 6, 14  
    decoding, 363, 365  
    decoding techniques, 366  
    unit, 552  
Addressing  
    modes, 584  
    modes of 8086, 74  
    in protected mode, 588  
Adjustment Instructions, 113  
Advanced programmable interrupt controller (APIC), 615  
Advantages of  
    interrupts, 192  
    using flowcharts, 166  
ALE, 16  
ALIGN, 150  
Allocator, 651  
AM (alignment mask), 620  
AND, 103  
Architecture, 614  
    of 80286, 552  
    of 80386, 573  
Arithmetic  
    instructions, 98  
    and logic unit, 5, 12  
Assemble, 141  
Assembler, 74, 139  
    directives, 146  
Assembly language 4, 137  
    program development tools, 138  
Assume, 153  
Asynchronous  
    mode, 319  
    reception, 320  
parallel data transfer, 293  
serial data transfer, 300  
Auto  
    EOI mode, 217  
    load mode, 479

- Auto-initialize mode, 495  
 Automatic  
     end of interrupt (AEOI), 217  
     rotation, 218  
 Auxiliary flag, 8
- Base  
     address, 487  
     addressing mode, 75, 77  
     registers, 32  
     word count registers, 487
- Based indexed  
     addressing mode, 75, 79  
     with displacement, 75  
         addressing mode, 79
- Basic concepts in memory interfacing, 362  
 Baud rate, 303  
 Benefits of multiprocessing, 276  
 BHE-, 35  
 Bit set reset (BSR) mode, 388  
 Block diagram and functional description of 8259A, 203  
 Block diagram description of 8251, 309  
 Block diagram of  
     80186, 546  
     8086, 28, 30  
     8087, 248  
     8237 DMA controller, 485  
     8253/54, 437  
     8255, 384  
     8257, 474  
     8279, 512  
     math coprocessor 8087, 249  
     pentium II processor, 644  
     pentium IV, 649  
     the interfacing of 8086 with 8259, 200
- Block transfer  
     DMA, 467  
     mode, 493
- Branch prediction, 625  
 Breakpoint or type 3 interrupt (INT3), 199  
 Buffered mode, 218  
 Burst, 467  
     or block transfer DMA, 296  
 Bus allocation schemes, 287  
 Bus controller 8288, 60  
 Bus cycles of 8086, 65  
 Bus interface  
     in mode 0, 392  
     unit, 29, 575  
 Bus interfacing in  
     mode 1, 393  
     mode 2, 400  
 Bus request and bus grant timings in minimum and, maximum mode systems, 69
- Bus system, 6  
 Bus unit (BU), 553
- Cache memory, 603  
 Cache organization, 626  
 CALL, 115  
 Capability  
     approach, 597  
     list, 597  
 Carry flag, 7  
 Cascade mode, 219, 493  
 CBW, 102  
 Central processing unit, 573  
 Chip-select/Ready generation logic, 547  
 Classification of interrupts, 194  
 CLC, 115  
 CLD, 115  
 Clear command, 519  
 CLI, 115  
 Clock generator 546  
     8284 and driver, 55  
 Closely coupled, 279  
     configuration, 282  
 CMC, 115  
 CMP, 104  
 CMPSB, 111  
 CMPSW, 111  
 .CODE, 154  
 Code segment (CS), 29  
 Combination of mode 1, 398  
 Combining segments, 155  
 Command  
     outputs, 62  
     register, 487  
     word format, 316  
 Commands of 8279, 515  
 Compare instructions, 261  
 Compressed timing, 496  
 Computer languages, 3  
 Consecutive transfers, 481  
 .CONST, 155  
 Control  
     bus, 6, 14  
     flags, 33  
     outputs, 62  
     override, 482  
     register, 250, 581  
     section, 513  
     signal definition output mode, 397  
     and status signals, 16  
     timing unit, 12  
     transfer instructions, 115  
     unit, 5, 12, 248  
     word format, 439  
     word register of 8255 in IO mode, 390

- Coprocessor, 279  
 configuration, 279  
 operation, 282  
 response to escape instructions, 241
- Counter, 435, 451  
 Counter latch command, 441
- CPL, 565
- Current  
 address register, 487  
 word register, 487
- CWD, 103
- CY (carry) flag, 13
- Cycle steal, 297, 467
- DAA, 114  
 Daisy chaining scheme of bus allocation, 287
- DAS, 114  
 .DATA, 154
- Data  
 addressing model, 75  
 bus, 6, 14  
 buffer, 309  
 communication over long distances, 303  
 defining assembler directives, 146  
 register, 250  
 segment (DS), 29  
 size bit (W), 85  
 types of 8087, 243
- Data transfer  
 DMA operation, 469  
 with handshake signals, 293  
 instructions, 92, 257  
 modes, 395  
 operations of 8251, 318  
 in output mode in mode 1, 397  
 with ready signal, 293
- Data/Address buffer register, 12  
 DB (define byte), 146  
 DD (define double word), 147  
 DE (debugging extensions), 621  
 Debounce and control, 515  
 Debug and test registers, 582  
 Debugger, 141  
 commands, 141
- DEC, 100  
 Dedicated interrupts of 8086, 198  
 Delay, 435  
 Demand transfer mode, 493  
 Demultiplexing of the multiplexed buses, 51  
 DEN-, 37  
 Descriptors, 560  
 Detailed block diagram of pentium IV, 650  
 Direct, 75  
 addressing mode, 76  
 memory access (DMA), 295
- Direction  
 bit (D), 85  
 flag (DF), 34  
 Dispatch ports of pentium IV, 652  
 Dispatch/execute unit, 638  
 Display, 507  
 section, 512, 514  
 write inhibit/blanking, 518
- Displacement only addressing mode, 76  
 DIV, 100  
 Divide by zero interrupt or type 0 (INT0), 198
- DMA  
 channels, 474  
 controller, 465  
 cycles, 491  
 operating modes, 492  
 operation, 481  
 read 494  
 verify, 494  
 write, 494
- DOSSEG, 152  
 DPL, 565  
 DT (define ten bytes), 148  
 DT/R-, 37  
 DUMP, 142  
 DUP (duplicate), 149  
 Duplex, 299  
 DW (define double byte or define word), 146  
 Dynamic branch prediction, 614  
 Dynamic RAM, 353, 356  
 Dynamic read-write memory (DRAM), 361
- Edge triggered interrupts, 194  
 Editor, 139  
 Eflag register of 80486, 601  
 Electrically erasable programmable read-only memory, 355  
 Electrically erasable PROM (EEPROM), 352  
 Encoded mode, 521  
 END, 153  
 End interrupt/Error mode set command, 519  
 End of interrupt (EOI), 219  
 ENDS, 153  
 ENTER, 142  
 EPL, 565  
 EQU (equate), 149  
 Erasable programmable read-only memory (EPROM), 355  
 Erasable PROM (EPROM), 352  
 ESC, 121  
 Escape (ESC) instruction, 240, 281  
 EVEN, 151  
 Exception  
 and interrupt handling, 607

- mask bits, 251  
 pointer of 8087, 255
- Exception- or interrupt-handler procedures, 607
- Execution unit, 31, 553
- Exit, 159  
 from SMM, 630
- Expansion past 64 interrupts, 221
- Extended write mode, 478, 496
- Externally or peripheral initiated signals, 17
- Extra segment (ES), 30
- EXTRN, 156
- F2XM1, 262
- FABS, 261
- FADD, 258
- FADD (Floating-point adder section), 625
- .FARDATA, 155
- FBLD, 258
- FBSTP, 258
- FCHS, 261
- FCLEX/FNCLEX, 263
- FCOM, 261
- FCOMP, 261
- FDECSTP, 264
- FDISI/FNDISI, 263
- FDIV, 260
- FDIV (Floating-point divider), 625
- FDIVP, 260
- FDIVR, 260
- Features of 8251 USART, 306
- Features of DAC 0800, 419
- FENI/FNENI, 263
- Fetch/decode unit, 637
- FEXP (Floating-point exponent section), 624
- FFREE, 264
- FIADD, 259
- FICOM, 262
- FICOMP, 262
- FICR, 624
- FIDIV, 260
- FIFO/RAM status and clear control, 515
- FILD, 258
- FIMUL, 260
- FINCSTP, 264
- FINIT/FNINT, 263
- FIST, 258
- FISTP, 258
- FISUB, 259
- Flag 12  
 register, 7, 32  
 of 80286, 554  
 of 80386, 578
- related instructions, 114
- FLD, 257
- FLD2T, 263
- FLDCW source, 263
- FLDENV, 264
- FLDI, 263
- FLDL2E, 263
- FLDLG2, 263
- FLDPI, 263
- FLDZ, 263
- Floating point unit of pentium, 623
- Floating-point pipeline stages, 623
- Flowchart, 164
- FMUL, 260
- FMUL (Floating-point multiplier section), 624
- FMULP, 260
- FNOP, 264
- Format of descriptors, 590
- Format of the  
 command register, 488  
 mask register, 489  
 mode set register, 488  
 request register, 488  
 status register, 253, 489
- FPATAN, 262
- FPREM, 261
- FPTAN, 262
- FPU architecture, 624
- FRNDINT, 261
- Front end, 649
- FRSTOR, 264
- FSAVE/FNSAVE, 264
- FSQRT, 260
- FST, 257
- FSTCW/FNSTCW, 263
- FSTENV/FNSTENV, 264
- FSTP, 257
- FSTSW/FNSTW, 263
- FSUB, 259
- FSUBP, 259
- FSUBR, 259
- FTST, 262
- Full duplex, 300
- Full or absolute address decoding, 366
- Fully nested mode, 219
- Functional behaviour of a DMA data transfer, 465
- Functional block diagram of  
 8253/54, 435  
 8288, 61
- Functional description, 485
- FWAIT, 264
- FXAM, 262
- FXCH, 257
- FXTRACT, 261
- FYL2X, 262
- FYL2XP1, 262

- Gate, 451  
GATE descriptors, 589  
GDTR, LDTR and IDTR, 556  
General purpose  
    data register, 11, 31  
    registers, 578  
Generation of control signals, 54  
Global and local descriptor tables, 588  
GO, 142  
Group, 155
- Half duplex, 299  
Hardware interrupts, 195  
Hardware key debouncing, 503  
High-level language, 4  
Higher order address bus, 16  
HLDA, 19, 37  
HOLD, 18, 37  
Host response of host processor to escape instruct, 241  
Hyper-threading technology, 655
- I/O bus mode, 60  
ID identification, 620  
IDIV, 101  
Idle cycle, 492  
Immediate, 75  
Immediate addressing mode, 75  
Implementing INTR as edge triggered, 200  
Implied, 75  
IMUL, 102  
IN, 93  
IN instruction, 341  
INC, 99  
INCLUDE, 156  
Increment/decrement counter, 11  
Independent request scheme of bus allocation, 288  
Index addressing mode, 75, 78  
Index of capabilities, 598  
Infinity control bit (bit 12), 252  
Initialization, 207  
    of 8251, 323  
    of 8279, 532  
    sequence of 8259A, 208  
Initialization control  
    word1 (ICW1), 208  
    word2 (ICW2), 210  
    word3 (ICW3), 211  
    word4 (ICW4), 212  
    words (ICWS), 207  
Input control signal definition, 393  
Input mode, 520  
Instruction  
    decoder, 12  
format, 84  
pointer (IP), 30  
register (IR), 12  
set of 8086, 92  
set of 8087, 256  
templates, 86  
unit (IU), 553  
INTA—(interrupt acknowledge), 18  
Integer  
    data format, 246  
    and floating-point execution units, 653  
pipeline, 621  
transfers, 258  
Intel 80186 microprocessor, 545  
Intel pentium II processor, 643  
Intel pentium III processor, 646  
Intel pentium IV processor, 648  
Interfacing, 340  
    circuit, 364  
    in input mode, 394  
Interfacing of  
    8087 with 8086, 255  
    8253/54 with 8086, 453  
    8259A within IO mapped IO method, 222  
    8279 with microprocessor 8086, 533  
    ADC0800 8-bit A/D converter, 424  
    DAC 0800, 419  
    DMA controller, 496  
    DMA controller with processor, 497  
    input device, 343  
    memory in minimum mode, 58  
    output device, 345  
    ROM with 8086, 371  
    stepper motor, 415  
Internal  
    architecture of 8085, 10  
    data operations, 7  
Interrupt  
    acknowledge machine cycle in  
        maximum mode, 201  
        minimum mode, 201  
    descriptor table, 567, 605  
    driven  
        input, 395  
        IO, 292, 294  
        output, 397  
    enable flag (IF), 33  
    and exceptions of 80386 AND 80486, 604  
    pointer table, 196  
    sequence, 196, 206  
    systems, 193  
    tasks, 607  
    vector/pointer table, 197  
Interrupts and exceptions, 565  
Interrupts of 8086, 195

- Intersegment  
 direct, 82  
 indirect, 83
- Initialization  
 format 1, 440  
 format 2, 440
- INTn, 116
- INTO, 116
- INTR (interrupt request), 18, 199
- Intrasegment  
 direct, 81  
 indirect, 82
- Introduction  
 to the 8087, 239  
 to MMX, 631
- IO**  
 addressing, 341  
 devices, 340  
 mapped IO, 341, 402  
 method, 454  
 mode, 389
- IO/M-, 16
- IRET, 116
- JMP, 117
- Key debouncing, 503
- Keyboard, 502  
 interfacing circuit, 504  
 section, 512, 514
- Keyboard/display mode set command, 516
- LABEL, 150
- LAHF, 97
- LDS, 94
- LEA, 95
- Left entry mode, 524
- LES, 95
- Level-triggered interrupts, 194
- Limitations of using Flowcharts, 166
- Linker, 140
- Load constant instructions, 263
- Loader, 141
- Local and global descriptor table, 563
- Lock facility, 277
- LOCK-, 38
- LODSB, 111
- LODSW, 111
- Logical and linear address, 584
- LOOP, 120  
 conditional, 120  
 unconditionally, 120
- LOOPE, 120
- LOOPNE, 120
- LOOPNZ, 121
- LOOPZ, 121
- Loosely coupled, 279  
 configuration, 284
- Low-level language, 3, 74
- Lower memory CS, 548
- M/I/O-, 36
- Machine  
 cycle encoder, 12  
 language, 4, 74  
 status word, 555
- Managing large programs, 156
- Mandatory access control on data and code access, 598
- Mask programmed ROM, 352  
 memory circuits, 353
- Mask register, 489
- Maskable interrupts, 194
- MASM, 137  
 assembler, 144
- Matrix keyboard, 506
- Maximum mode  
 bus cycles of 8086 system, 67  
 configuration of 8086, 59  
 memory read bus cycle of 8086 system, 68  
 memory write bus cycle of 8086 system, 69
- MCE (machine check enable), 621
- Memory  
 addresses, 365  
 addressing in real mode, 586  
 hierarchy of pentium IV, 654  
 interface of a maximum-mode 8086 system, 64  
 interfacing with fold back address, 371  
 interfacing without fold back address, 371  
 management of 80486, 603  
 system of pentium processor, 627  
 unit, 575
- mapped IO, 342, 402  
 method, 455
- operating modes, 557
- organization, 352  
 of 8086, 39  
 of 80486, 602  
 and memory management unit of, 584  
 of microprocessor 8086, 368  
 of pentium, 627  
 and segmentation, 557
- protection, 596  
 across ring boundaries, 596  
 within the same ring, 597

- reference ESCAPE instruction, 241  
 subsystem, 654
- Memory-to-Memory**, 495
- Memory/Peripheral control**, 547
- Microprocessor**, 4  
 controlled data transfer, 292  
 initiated operations, 6  
 operations, 5  
 system, 19
- Microprocessor**  
 80286, 551  
 80386, 572  
 80486, 599  
 8088, 41
- Microsoft assembler (MASM)**, 137
- Mid-range memory CS**, 548
- Minimum mode bus cycles**, 67
- MMX**  
 arithmetic instructions, 635  
 comparison instructions, 635  
 data  
     packing instructions, 636  
     transfer instructions, 633  
     types, 632  
 logical instructions, 634  
 registers, 631  
 shift and rotate instructions, 634  
 state management instruction, 633  
 technology instructions, 633
- MN/MX-**, 36
- Mnemonics**, 74, 137
- Mode**  
 register, 487  
 set register, 477  
 word format, 314
- Mode 0**, 384
- Mode 0 (simple input/output)**, 391
- Mode 0: Interrupt on Terminal Count**, 444
- Mode 1**, 384
- Mode 1 (strobed input/output)**, 393
- Mode 1 output mode**, 396
- Mode 1 status word format**, 398
- Mode 1: hardware retriggerable one-shot**, 446
- Mode 2**, 384
- Mode 2 (strobed bidirectional bus I/O)**, 399
- Mode 2 different combinations**, 400
- Mode 2: rate generator**, 447
- Mode 3: square wave mode**, 447
- Mode 4: software triggered mode**, 448
- Mode 5: hardware triggered strobe (Retriggerable)**, 451
- .MODEL**, 158
- Modem control**, 313
- Modes of operations**, 444
- MOV**, 92  
**MOVE**, 142  
**MOVSB**, 111  
**MOVSW**, 111  
**MUL**, 101  
**Multilevel**, 193  
**Multiline interrupt system**, 193  
**Multiplexed address/data bus**, 16  
**Multiprocessing and multiprogramming**, 275  
**Multitasking** in 80286, 563  
**Mux/Demux unit**, 11
- N-Key**  
 lockout, 504
- N Roll-over**, 504, 522
- NE (numeric error)**, 621
- NEG**, 102
- Non-maskable**  
 interrupts, 195  
 type 2 interrupt (NMI), 199
- Non-memory reference ESCAPE instruction**, 240
- Non-vectored interrupts**, 195
- NOP**, 121
- Normal mode**, 496
- NOT**, 104
- Not ready**, 482
- Numeric displays**, 507
- Numeric execution unit**, 248
- NVRAM**, 356
- OFFSET**, 148
- Operating mode**  
 of 8279, 519  
 of OCW3, 216
- Operation**, 424  
 of 8255, 387  
 of the clock section, 55  
 command  
     word2 (OCW2), 214  
     word3 (OCW3), 215  
     words (OCWs), 213  
 common to all modes, 451  
 of DMA cycle, 483  
 of the Ready section, 55  
 of the RESET section, 55
- OR**, 105
- ORG**, 152
- OUT**, 94  
 instruction, 342
- Out-of-order execution logic**, 651
- Output modes of 8279**, 524

- P (Parity) flag, 14  
 Packed decimal transfers, 258  
**Page**  
 directory, 591  
 tables, 591  
**Paging**, 575, 590  
 descriptor base register, 591  
 unit, 591  
 of pentium, 630  
**Parallel transmission**, 291  
**Parity flag**, 8  
**Partial address decoding**, 366  
**Pentium II processor pipeline**, 643  
**Pentium pro**  
 micro architecture, 637  
 processor, 636  
**Pentium processor**, 614  
**Peripheral or externally initiated operations**, 8  
**PF (Prefetch)**, 621  
**Pin configuration**  
 of 80186, 549  
 of 80486, 599  
 of 8086, 34  
 of 8087, 247  
 of 8237, 486  
 of 8253/54, 437  
 of 8279, 509  
 of bus controller 8288, 63  
 of clock generator, 56  
 of pentium, 616  
 II processor, 644  
 pro processor, 640  
 of 80286, 553  
 and block diagram of DAC 0800, 420  
**Pin description**  
 of 8251, 306  
 of 8253/54, 437  
 of 8255, 386  
**Pin diagram**  
 of 8257, 471  
 of bus controller 8288, 63  
 and pin description, 205  
**Pipelined floating-point unit**, 614  
**Pointers**, 32  
**Polled mode**, 216  
**Polling scheme of bus allocation**, 288  
**POP**, 96  
**POPF**, 97  
 Precision control bits (bits 9 and 8), 252  
**Priority mode**, 496  
**Priority of interrupts**, 202  
**Privilege levels**, 564  
**Processor**  
 control instructions, 121, 263  
 directives, 157  
**Program**  
 clock command, 517  
 counter, 8, 11  
**Programmable**  
 DMA controller 8257, 470  
 peripheral interfacing chip 8255, 383  
 read-only memory (PROM), 355  
 ROM (PROM), 352  
**Programmed IO**, 292  
**Programming** 8259A, 223  
 of 8259A, 207  
 and reading the 8257 registers, 480  
 steps, 167  
**Protected mode**, 558  
 of 80386, 586  
**Protected virtual address mode (PVAM)**, 559  
**PSE (page size extensions)**, 621  
**PTR**, 148  
**PUSH**, 96  
**PUSHF**, 97  
**PVI (protected mode virtual interrupts)**, 621  
  
**QS0**, 38  
**QS1**, 38  
**Queue**, 29  
 status, 242  
**QUIT**, 142  
  
**Random access memory (RAM)**, 356  
**RCL**, 106  
**RCR**, 107  
**RD-**, 16  
**Read back command**, 442  
 format, 442  
**Read display RAM command**, 517  
**Read FIFO/sensor RAM command**, 517  
**Read operations**, 441  
**Ready**, 18  
**Real**  
 address mode of 80386, 585  
 or floating point data formats, 243  
 mode, 557  
**Receiver section**, 312  
**Register**, 75, 142  
 addressing mode, 76  
 indirect, 75  
 indirect addressing modes, 76  
 organization of 80386, 577  
 renaming, 651  
 unit, 5, 7, 10  
**Register alias table (RAT)**, 638  
**Registers**, 618  
 of 80286, 554

- of 8087, 249  
of 8237, 486  
unit, 10
- REP (conditionally), 113  
REP (unconditionally), 113  
REPE/REPZ, 113  
REPNE/REPNZ, 113  
Request register, 487  
RESET OUT, 19  
RESET-IN-, 19  
RET, 116  
Retire unit, 639  
Return lines, 510  
RF resume, 580  
Right entry mode, 528  
ROL, 107  
Roll-over, 504  
ROM (read only memory), 353  
ROR, 108  
Rotating priority, 477  
Rounding control bits (bits 11 and 10), 252  
RPL, 565  
RQ-GT0-, 38  
RQ-GT1-, 38  
RRND (floating-point rounder), 625  
RST 5.5, RST 6.5, RST 7.5, 17
- S (sign) flag, 13  
SAHF, 97  
SAL/SHL, 108  
SAR, 109  
SBB, 99  
Scan lines, 511  
Scanned  
  keyboard mode, 520  
  sensor matrix mode, 520, 522  
SCASB, 112  
SCASW, 112  
SEGMENT, 153  
Segment  
  defining directives, 152  
  descriptor registers, 580  
  override prefix, 88  
  registers, 29  
Segmentation, 575  
Selector, 560  
Semiconductor memory, 350  
Serial IO signal, 17  
Serial I/O transfer using parallel port, 303  
Serial transmission, 297  
  format, 300  
Shift and rotate instructions, 106  
SHR, 110  
SID, 17
- Sign  
  bit(s), 85  
  flag, 8
- Signal descriptions of 80386, 576  
Signals of pentium IV, 655  
Simple data transfer, 293  
Simple read/write operations for the desired count, 441  
Simplex, 299  
Single byte transfer, 481  
  DMA, 297, 467  
Single level interrupt system, 193  
Single line, 193  
Single step or type 1 interrupt (INT1), 198  
Single transfer mode, 493  
SMM entry, 628  
SMRAM, 629  
SMRAM state save map, 629  
SOD, 17
- Software  
  based serial I/O, 303  
  interrupts, 195  
  interrupts of 8086, 202  
  key debouncing, 504
- Special  
  fully nested mode, 221  
  mask mode, 221, 216
- Specific rotation mode, 221
- Speed, 482
- SRAM read circuitry, 359  
SRAM write circuitry, 358  
.STACK, 154
- Stack  
  pointer, 8, 11  
  segment (SS), 29
- STARTUP, 158
- State diagram of 8257, 482
- Static  
  RAM, 353, 356  
  read/write memory (SRAM), 356
- Status  
  driven data transfer, 395, 398  
  flags, 32  
  read operation, 217  
  register, 479, 489  
  register of 8087, 252  
  word, 317
- STC, 115  
STD, 115  
STI, 115  
STOSB, 112  
STOSW, 112
- String  
  addressing mode, 75, 81  
  instructions, 110
- Strobed input mode, 520, 524

- STRUCT, 152  
SUB, 99  
Superscalar architecture, 614  
Superscalar execution, 622  
Synchronous data transfer (unconditional), 293  
Synchronous mode  
    with delay, 293  
    reception, 322  
    transmission, 321  
    serial data transfer, 301  
System bus  
    mode, 61  
    of 8085, 14  
interface of 8253/54, 451  
memory management mode (SMM), 628  
registers, 620
- Tag register of 8087, 254  
Task gates, 589  
register, 556  
state segments, 589  
switching and task gates, 565  
TASM assembler, 143  
TC stop mode, 479  
Temporary register, 489  
TEST, 105  
Timing diagram  
    for 8086 minimum mode memory  
        write, 68  
        read., 67  
    of 8257, 484  
    in mode 0 (port B), 392  
        of mode 1, 395, 398  
Trace, 143  
Transceiver 8286, 53  
Transcendental instructions, 262
- Translating a virtual address to a physical address, 595  
Translation lookaside buffers (TLB), 626  
Transmitter Section, 311  
TRAP, 17  
Trap flag (TF), 33  
Trigonometric and exponential instructions, 262  
TSD (time stamp disable), 621\*
- Turbo assembler (TASM), 137
- Unassemble, 143  
Universal synchronous asynchronous receiver transmitter (USART), 306  
UOP scheduling, 652  
Upper memory CS, 548  
User registers, 619
- Vectored interrupts, 195  
VIP (virtual interrupt pending), 620  
Virtual 8086 mode of 80386, 594  
Virtual mode, 588  
VM virtual 8086 mode, 580  
VME (virtual 8086 mode extensions), 621
- WAIT, 121  
WP (write protect), 621  
WR-, 16  
Write display RAM command, 518  
Write operations, 439
- XLAT, 97  
XOR, 105
- Z (zero) flag, 8, 13  
Z-bit, 85

# **Microprocessor 8086**

## **Architecture, Programming and Interfacing**

**Sunil Mathur**

Primarily intended for the undergraduate students of electronics and communication engineering, computer science and engineering, and information technology, this book skilfully integrates both the hardware and software aspects of the 8086 microprocessor. It offers the students an up-to-date account of the state-of-the-art microprocessors and therefore can be regarded as an incomparable source of information on recently developed microprocessor chips. The book covers the advanced microprocessor architecture of the Intel microprocessor family, from 8086 to Pentium 4.

The text is organized in four parts. Part I (Chapters 1–7) includes a detailed description of the architecture, organization, instruction set, and assembler directives of microprocessor 8086. Part II (Chapters 8–11) discusses the math coprocessor, multiprocessing and multiprogramming, the different types of data transfer schemes, and memory concepts. Part III (Chapters 12–15) covers programmable interfacing chips with the help of extensive interfacing examples. Part IV (Chapters 16–18) deals with advanced processors—from 80186 to Pentium 4.

This well-organized and student-friendly text should prove to be an invaluable asset to the students as well as the practising engineers.

### **KEY FEATURES**

- ◆ Gives elaborate programming examples to develop the analytical ability of students.
- ◆ Provides solved examples covering different types of typical interfacing problems to develop the practical skills of students.
- ◆ Furnishes chapter-end exercises to reinforce the understanding of the subject.

### **THE AUTHOR**

**SUNIL MATHUR** is Assistant Professor, Department of Electronics and Communication Engineering, Maharaja Agrasen Institute of Technology, Guru Gobind Singh Indraprastha University, Delhi. He is also a visiting faculty in Delhi Technological University. He has 18 years of teaching experience. He is a life member of Institution of Engineers (India) and Institute of Electronics and Telecommunication Engineers.

His areas of interest are microprocessors, computer architecture and digital signal processing, image processing, and consumer electronics.

### **Our other useful books**

*Microprocessors and Microcontrollers: Architecture, Programming and System Design 8085, 8086, 8051, 8096*, Krishna Kant

*Embedded System Design*, Santanu Chattopadhyay

*Microprocessors, PC Hardware and Interfacing*, N. Mathivanan

**₹ 450.00**

[www.phindia.com](http://www.phindia.com)

ISBN: 978-81-203-4087-9



9 788120 340879