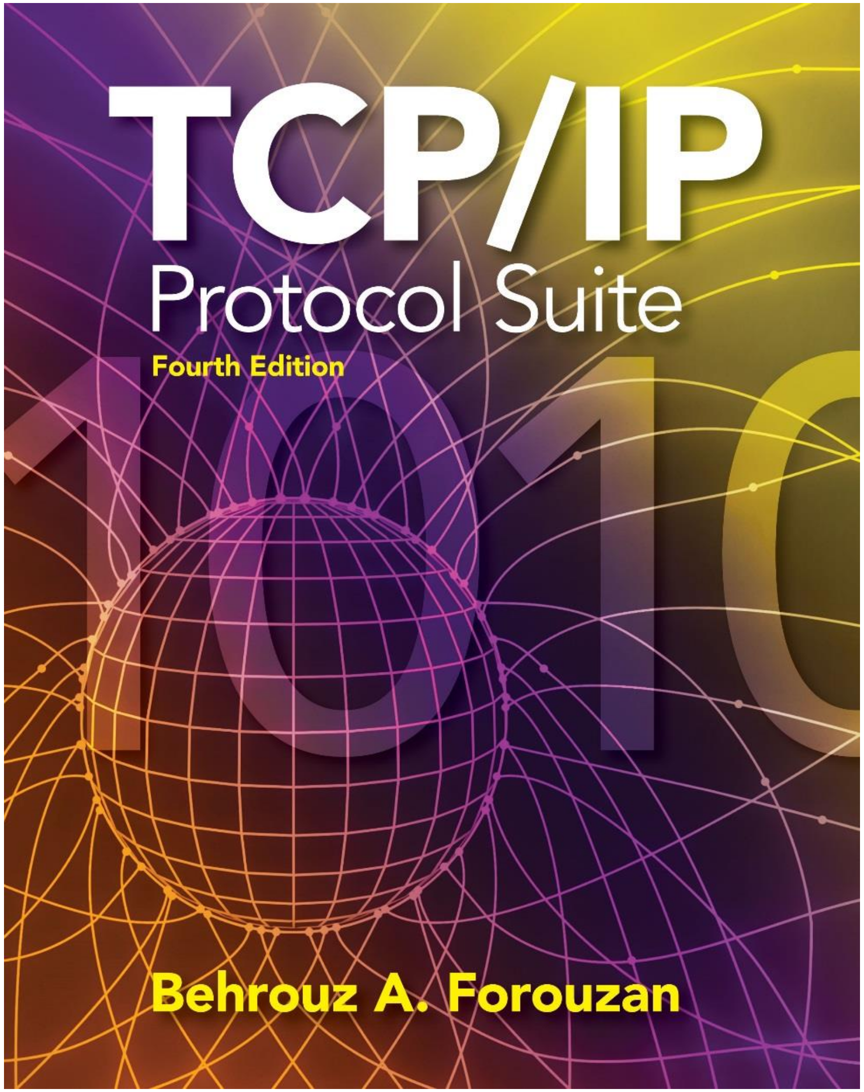
Chapter 5

IPv4 Addresses



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

OBJECTIVES:

- To introduce the concept of an address space in general and the address space of IPv4 in particular.
- To discuss the classful architecture and the blocks of addresses available in each class.
- To discuss the idea of hierarchical addressing and how it has been implemented in classful addressing.
- To explain subnetting and supernetting for classful architecture.
- To discuss classless addressing, that has been devised to solve the problems in classful addressing.
- To discuss some special blocks and some special addresses in each block.
- To discuss NAT technology and show how it can be used to alleviate(يخفف) of address depletion (نضوب).

Chapter Outline

5-1 INTRODUCTION

The identifier used in the IP layer of the TCP/IP protocol suite to identify each device connected to the Internet is called the Internet address or IP address. An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a host or a router to the Internet; an IP address is the address of the interface.

Topics Discussed in the Section

- ✓ **Notation**
- ✓ **Range of Addresses**
- ✓ **Operations**

Note

An IPv4 address is 32 bits long.

Note

The IPv4 addresses are unique and universal.

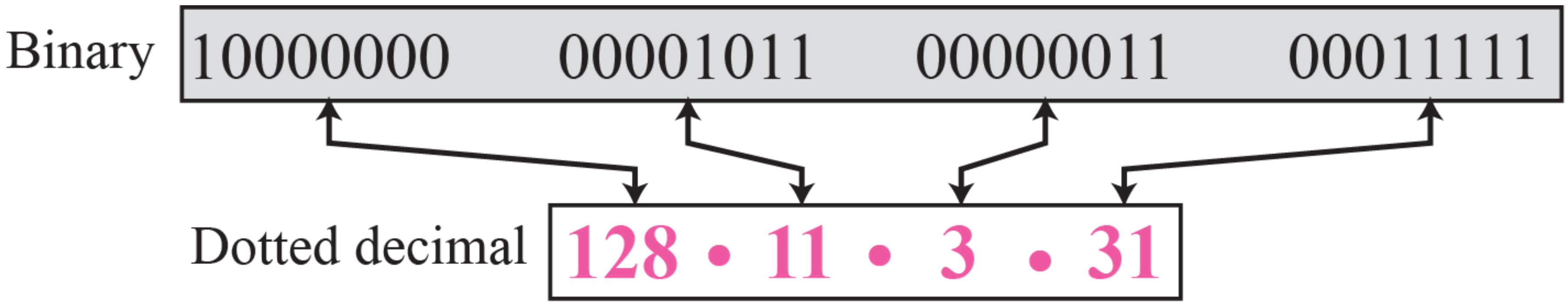
Note

The address space of IPv4 is 2^{32} or 4,294,967,296.

Note

Numbers in base 2, 16, and 256 are discussed in Appendix B.

Figure 5.1 *Dotted-decimal notation*



Example 5.1

Change the following IPv4 addresses from binary notation to dotted-decimal notation.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

c. 11100111 11011011 10001011 01101111

d. 11111001 10011011 11111011 00001111

Solution

We replace each group of 8 bits with its equivalent decimal number (see Appendix B) and add dots for separation:

a. 129.11.11.239

b. 193.131.27.255

c. 231.219.139.111

d. 249.155.251.15

Example 5.2

Change the following IPv4 addresses from dotted-decimal notation to binary notation.

a. 111.56.45.78
b. 221.34.7.82
c. *241.8.56.12*
d. *75.45.34.78*

Solution

We replace each decimal number with its binary equivalent:

a. 01101111 00111000 00101101 01001110
b. 11011101 00100010 00000111 01010010
c. 11110001 00001000 00111000 00001100
d. 01001011 00101101 00100010 01001110

Example 5.3

Find the error, if any, in the following IPv4 addresses:

a. 111.56.045.78

b. 221.34.7.8.20

c. 75.45.301.14

d. 11100010.23.14.67

Solution

a. There should be no leading zeroes (045).

b. We may not have more than 4 bytes in an IPv4 address.

c. Each byte should be less than or equal to 255.

d. A mixture of binary notation and dotted-decimal notation.

Example 5.4

Change the following IPv4 addresses from binary notation to hexadecimal notation.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

Solution

We replace each group of 4 bits with its hexadecimal equivalent. Note that 0X (or 0x) is added at the beginning or the subscript 16 at the end.

a. 0X810B0BEF or $810B0BEF_{16}$

b. 0XC1831BFF or $C1831BFF_{16}$

Example 5.5

Find the number of addresses in a range if the first address is 146.102.29.0 and the last address is 146.102.32.255.

Solution

We can subtract the first address from the last address in base 256 (see Appendix B). The result is 0.0.3.255 in this base. To find the number of addresses in the range (in decimal), we convert this number to base 10 and add 1 to the result..

$$\text{Number of addresses} = (0 \times 256^3 + 0 \times 256^2 + 3 \times 256^1 + 255 \times 256^0) + 1 = 1024$$

Example 5.6

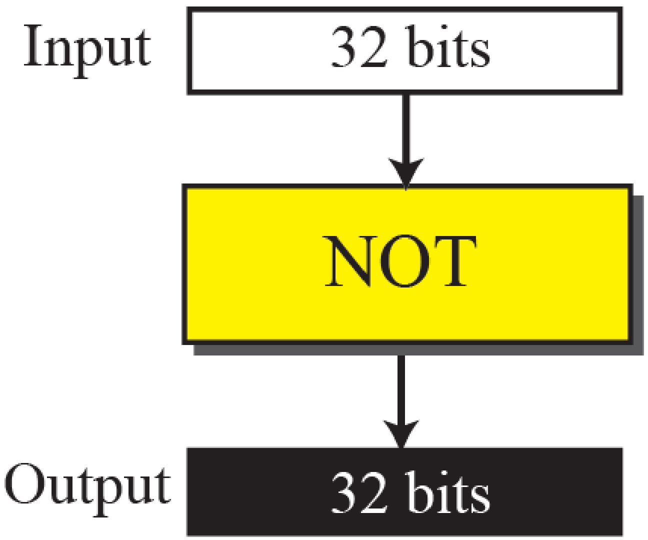
The first address in a range of addresses is 14.11.45.96. If the number of addresses in the range is 32, what is the last address?

Solution

We convert the number of addresses minus 1 to base 256, which is 0.0.0.31. We then add it to the first address to get the last address. Addition is in base 256.

$$\text{Last address} = (14.11.45.96 + 0.0.0.31)_{256} = 14.11.45.127$$

Figure 5.2 *Bitwise NOT operation*



NOT operation

Input	Output
0	1
1	0

Operation for each bit

Example 5.7

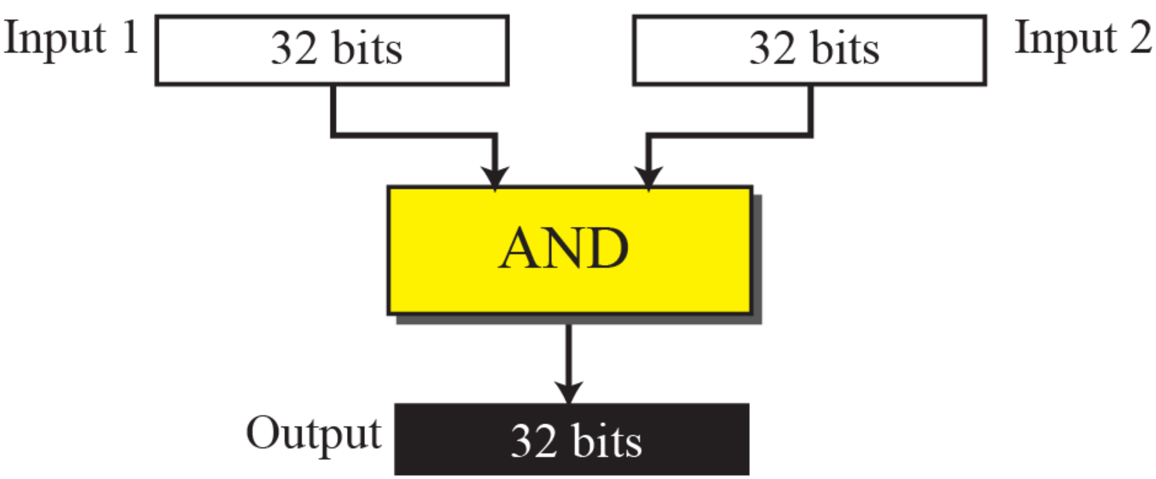
The following shows how we can apply the NOT operation on a 32-bit number in binary.

Original number:	00010001	01111001	00001110	00100011
Complement:	11101110	10000110	11110001	11011100

We can use the same operation using the dotted-decimal representation and the short cut.

Original number:	17 .	121 .	14 .	35
Complement:	238 .	134 .	241 .	220

Figure 5.3 *Bitwise AND operation*



AND

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

Operation for each bit

Example 5.8

First number:	00010001	01111001	00001110	00100011
Second number:	11111111	11111111	10001100	00000000
Result	00010001	01111001	00001100	00000000

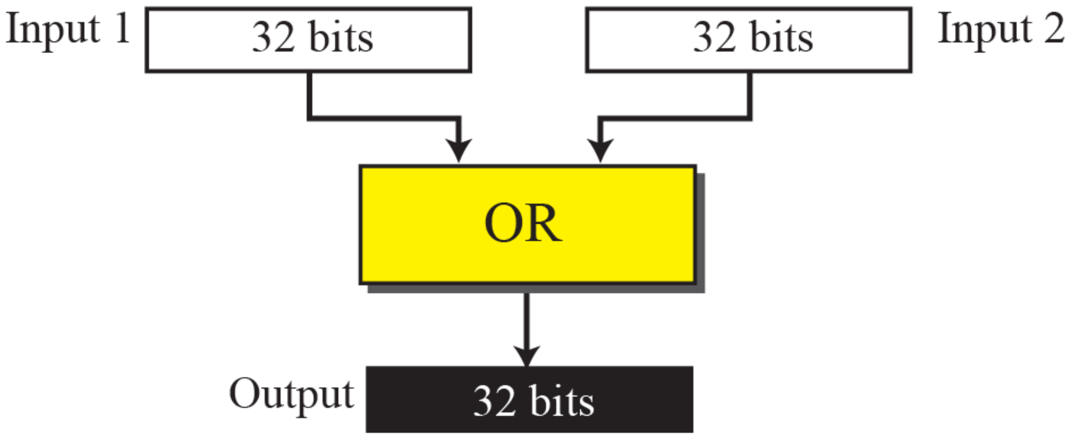
We can use the same operation using the dotted-decimal representation and the short cut.

First number:	17	.	121	.	14	.	35
Second number:	255	.	255	.	140	.	0
Result:	17	.	121	.	12	.	0

We have applied the first short cut on the first, second, and the fourth byte; we have applied the second short cut on the third byte. We have written 14 and 140 as the sum of terms and selected the smaller term in each pair as shown below.

Powers	2^7		2^6		2^5		2^4		2^3		2^2		2^1		2^0
Byte (14)	0	+	0	+	0	+	0	+	8	+	4	+	2	+	0
Byte (140)	128	+	0	+	0	+	0	+	8	+	4	+	0	+	0
Result (12)	0	+	0	+	0	+	0	+	8	+	4	+	0	+	0

Figure 5.4 *Bitwise OR operation*



OR

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	1

Operation for each bit

Example 5.9

The following shows how we can apply the OR operation on two 32-bit numbers in binary.

First number:	00010001	01111001	00001110	00100011
Second number:	11111111	11111111	10001100	00000000
Result	11111111	11111111	10001110	00100011

We can use the same operation using the dotted-decimal representation and the short cut.

First number:	17	.	121	.	14	.	35
Second number:	255	.	255	.	140	.	0
Result:	255	.	255	.	142	.	35

We have used the first short cut for the first and second bytes and the second short cut for the third byte.

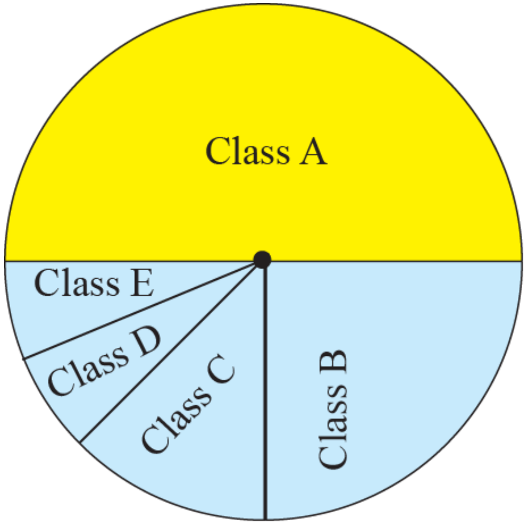
5-2 CLASSFUL ADDRESSING

IP addresses, when started a few decades ago, used the concept of classes. This architecture is called classful addressing. In the mid-1990s, a new architecture, called classless addressing, was introduced that supersedes (يحل محل) the original architecture. In this section, we introduce classful addressing because it paves(يمهد) the way for understanding classless addressing and justifies the rationale for moving to the new architecture. Classless addressing is discussed in the next section.

Topics Discussed in the Section

- ✓ Classes
- ✓ Classes and Blocks
- ✓ Two-Level Addressing
- ✓ Three-Level Addressing: Subnetting
- ✓ Supernetting

Figure 5.5 *Occupation of address space*



Class A: 2^{31} = 2,147,483,648 addresses, 50%

Class B: 2^{30} = 1,073,741,824 addresses, 25%

Class C: 2^{29} = 536,870,912 addresses, 12.5%

Class D: 2^{28} = 268,435,456 addresses, 6.25%

Class E: 2^{28} = 268,435,456 addresses, 6.25%

Figure 5.6 *Finding the class of address*

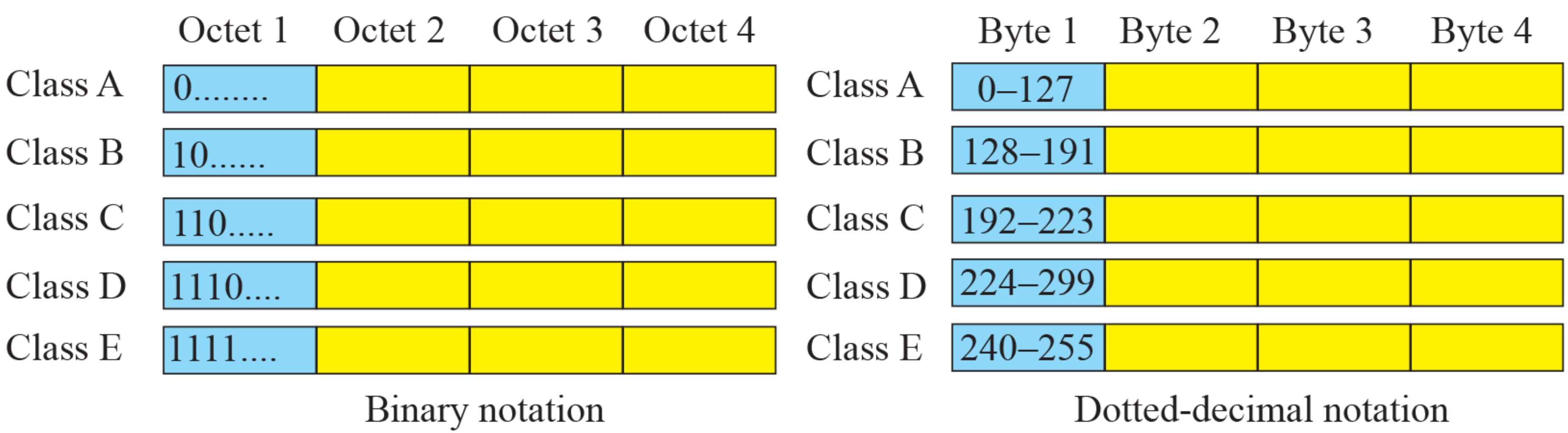
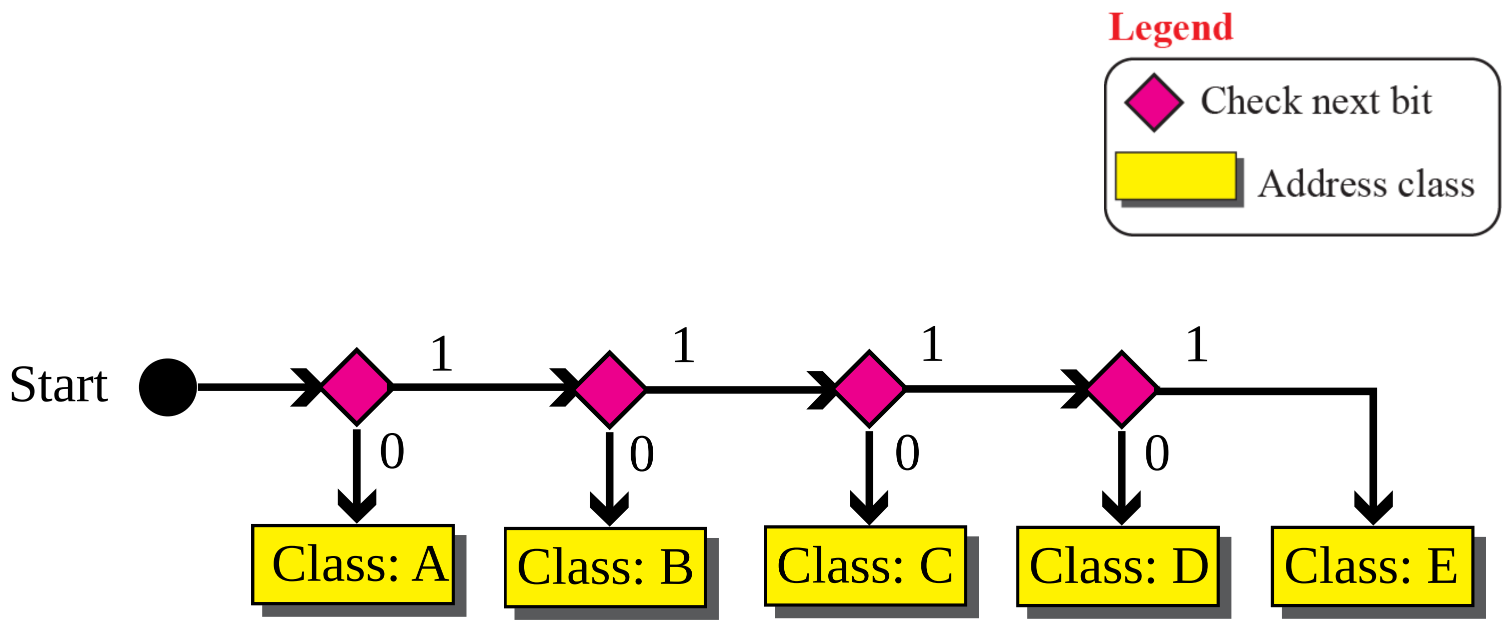


Figure 5.7 *Finding the class of an address using continuous checking*



Example 5.10

Find the class of each address:

a. 00000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

c. 10100111 11011011 10001011 01101111

d. 11110011 10011011 11111011 00001111

Solution

See the procedure in Figure 5.7.

a. The first bit is 0. This is a class A address.

b. The first 2 bits are 1; the third bit is 0. This is a class C address.

c. The first bit is 1; the second bit is 0. This is a class B address.

d. The first 4 bits are 1s. This is a class E address.

Example 5.11

Find the class of each address:

a. 227.12.14.87

b. 193.14.56.22

c. 14.23.120.8

d. 252.5.15.111

Solution

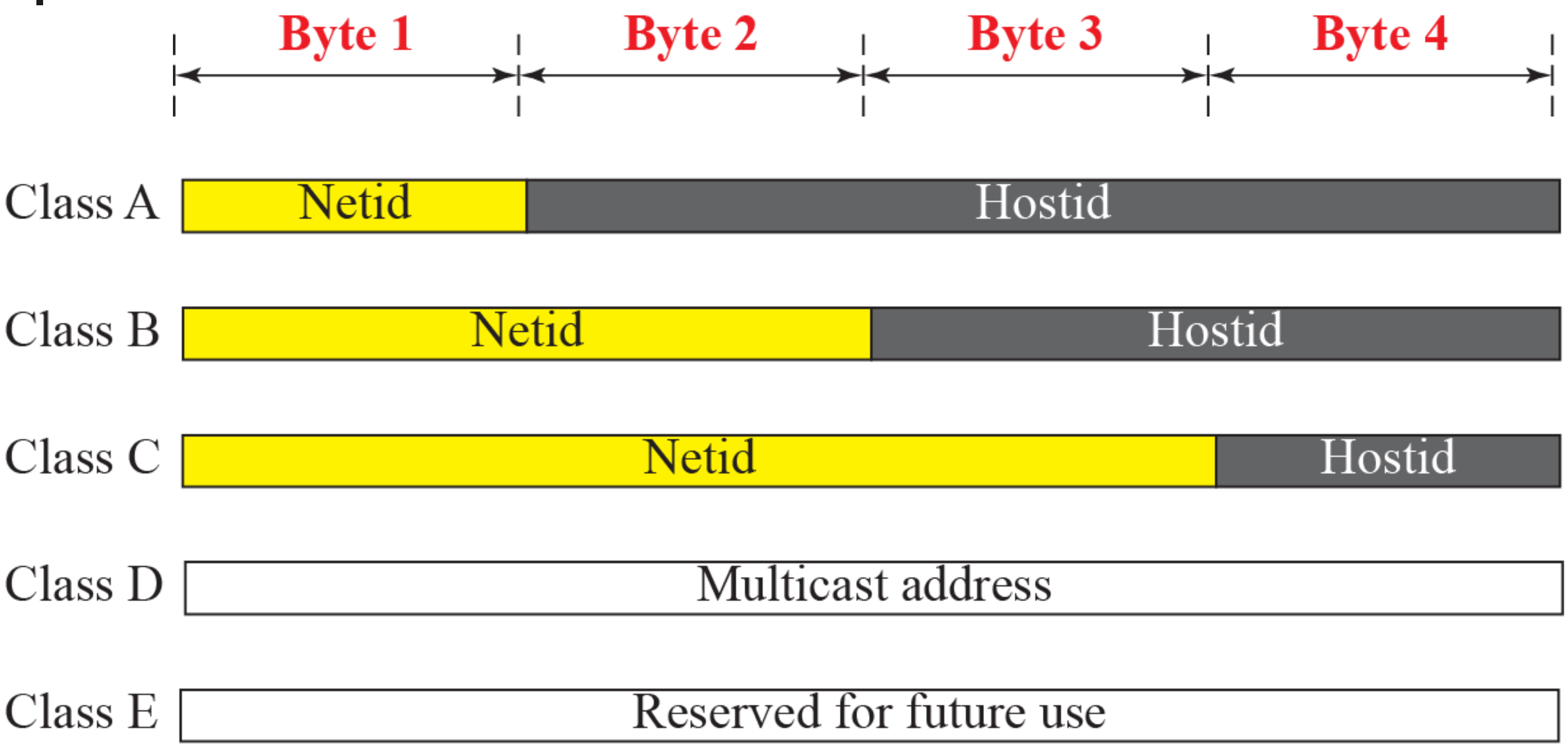
a. The first byte is 227 (between 224 and 239); the class is D.

b. The first byte is 193 (between 192 and 223); the class is C.

c. The first byte is 14 (between 0 and 127); the class is A.

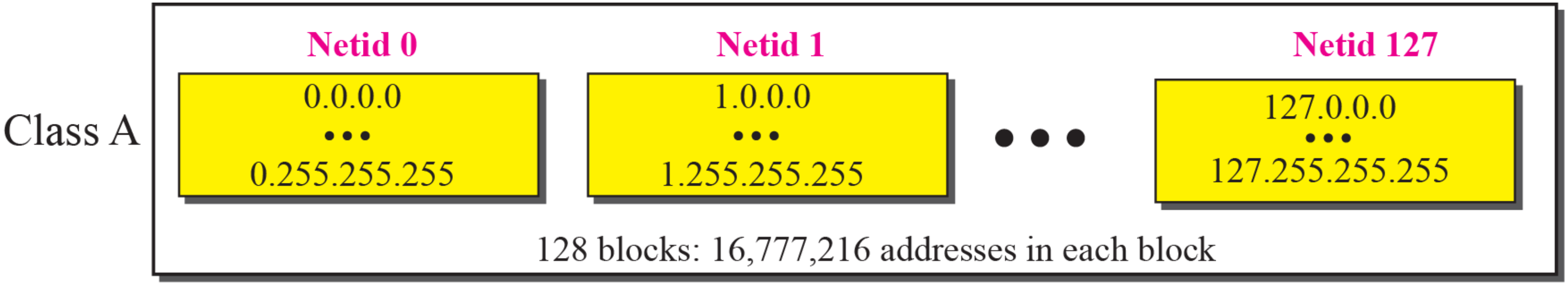
d. The first byte is 252 (between 240 and 255); the class is E.

Figure 5.8 *Netid and hostid*



❖ In classful addressing, an IP address in classes A, B, and C is divided into **netid and hostid**.

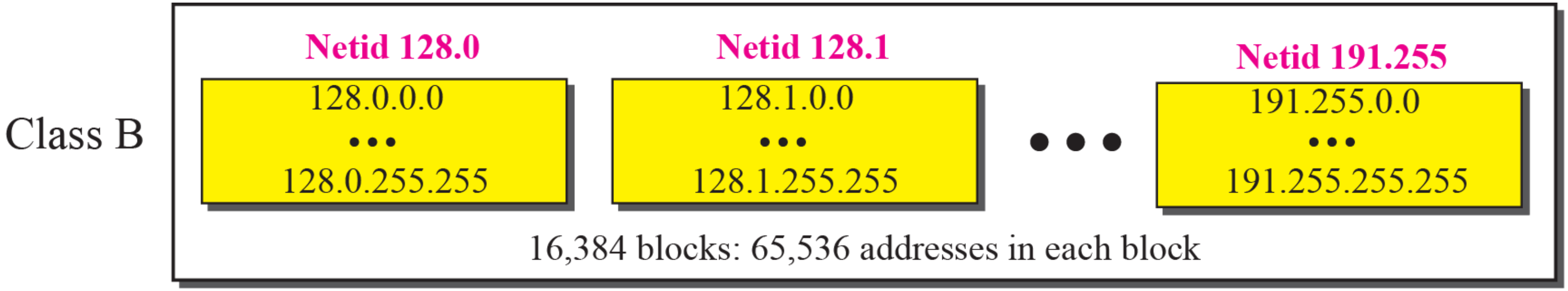
Figure 5.9 *Blocks in Class A*



Note

Millions of class A addresses are wasted.

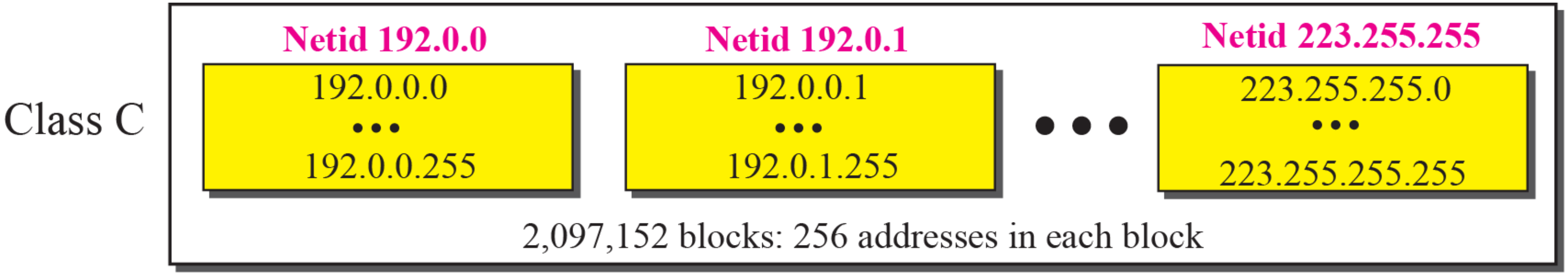
Figure 5.10 *Blocks in Class B*



Note

Many class B addresses are wasted.

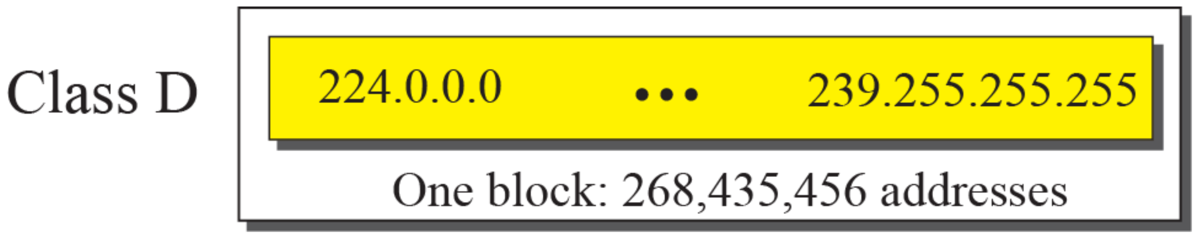
Figure 5.11 *Blocks in Class C*



Note

Not so many organizations are so small to have a class C block.

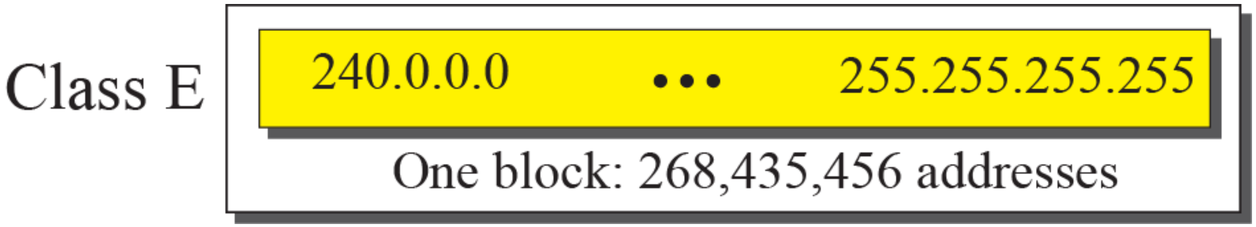
Figure 5.12 *The single block in Class D*



Note

Class D addresses are made of one block, used for multicasting.

Figure 5.13 *The single block in Class E*



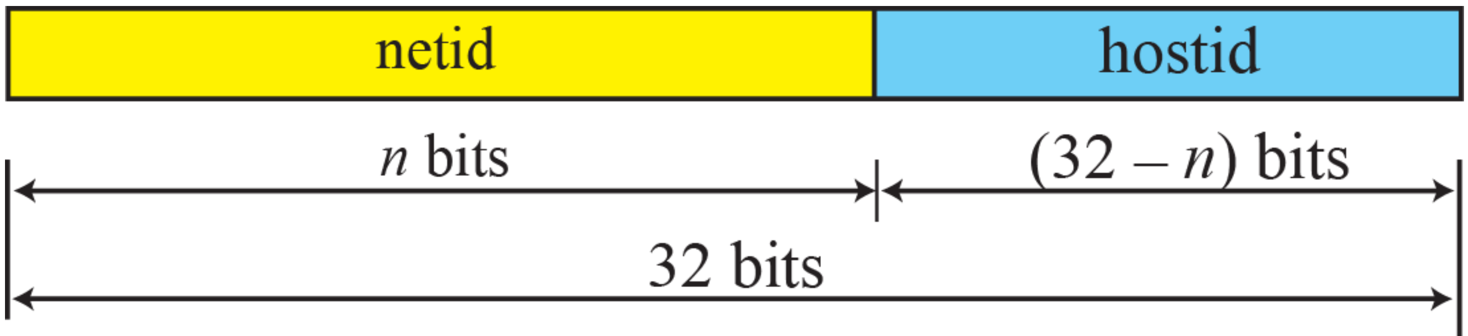
Note

The only block of class E addresses was reserved for future purposes.

Note

The range of addresses allocated to an organization in classful addressing was a block of addresses in Class A, B, or C.

Figure 5.14 *Two-level addressing in classful addressing*



Class A: $n = 8$

Class B: $n = 16$

Class C: $n = 24$

Example 5.12

Two-level addressing can be found in other communication systems. For example, a telephone system inside the United States can be thought of as two parts: area code and local part. The area code defines the area, the local part defines a particular telephone subscriber in that area.

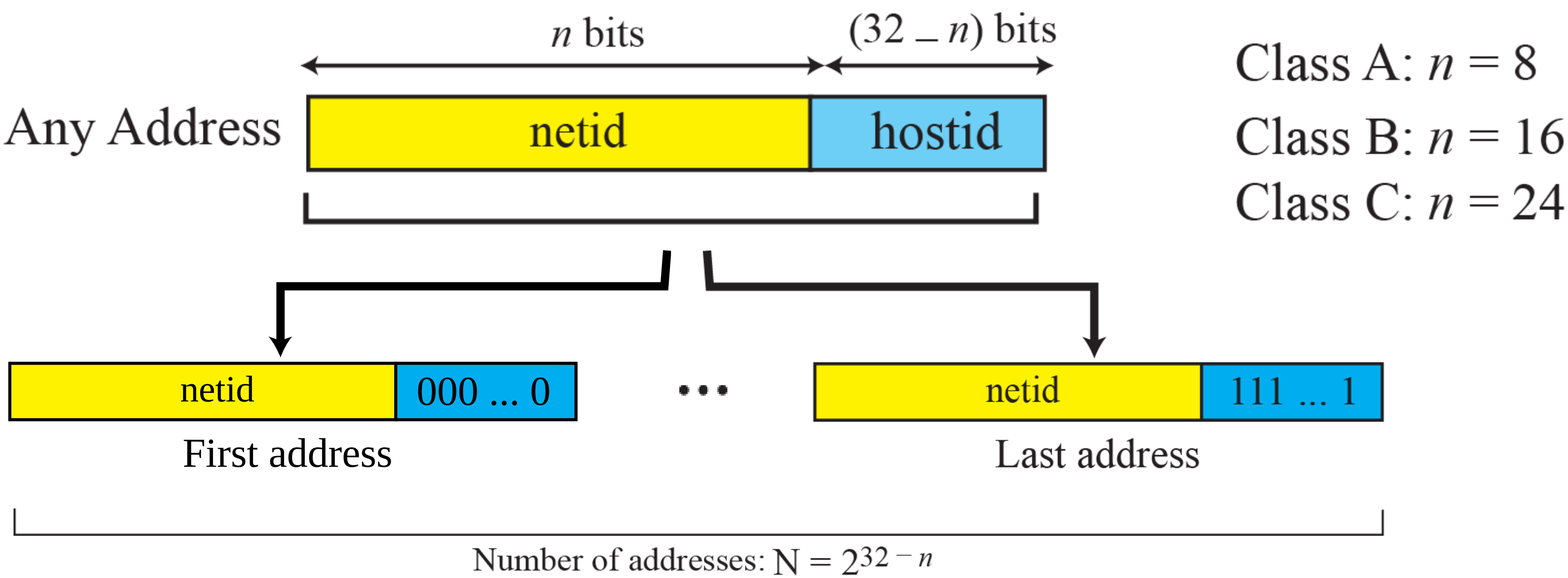
(626) 3581301

The area code, 626, can be compared with the netid, the local part, 3581301, can be compared to the hostid.

Extracting Information in a Block by:

1. The number of addresses in the block, N, can be found using N $=2^{32-n}$, Where n= no. of netid bits's
2. To find the first address, we keep the n leftmost bits and set the (32 - n) rightmost bits all to 0s.
3. To find the last address, we keep the n leftmost bits and set the (32 - n) rightmost bits all to 1s.

Figure 5.15 *Information extraction in classful addressing*



Example 5.13

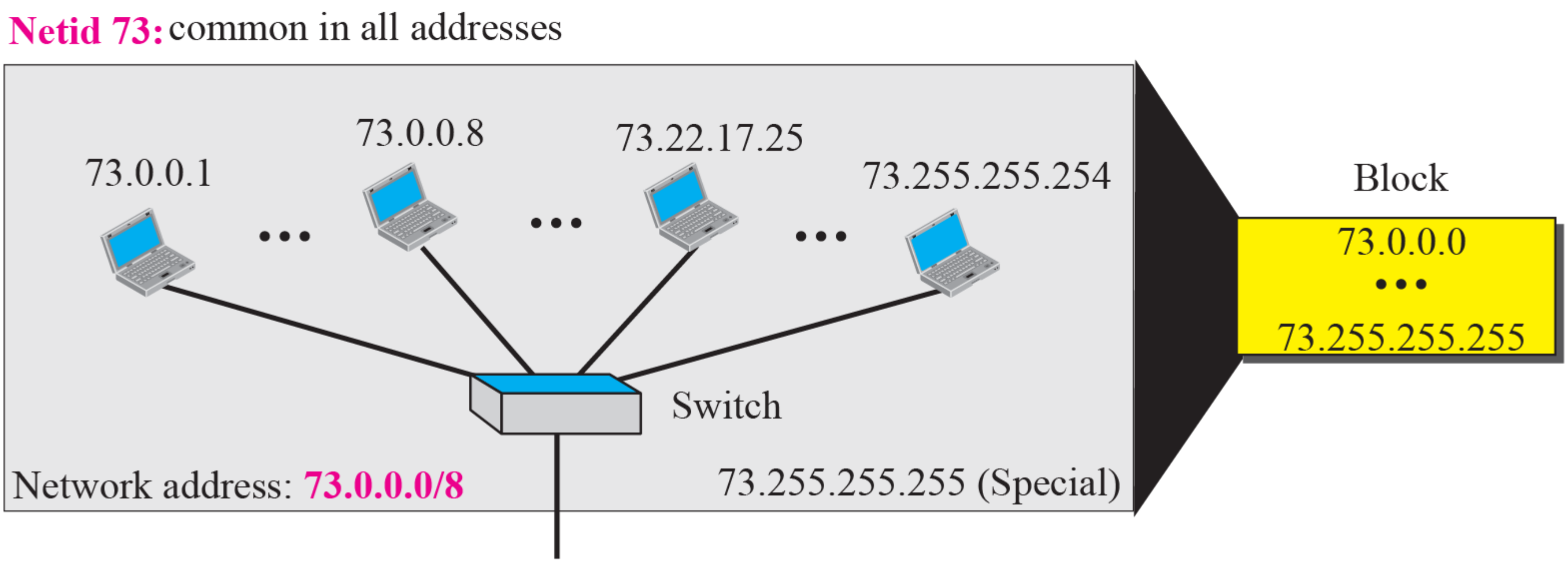
An address in a block is given as 73.22.17.25. Find the number of addresses in the block, the first address, and the last address.

Solution

Figure 5.16 shows a possible configuration of the network that uses this block. Since the block from class A, so:

1. The number of addresses in this block is $N = 2^{32-n} = 2^{32-8} = 2^{24} = 16,777,216$.
2. To find the first address, we keep the leftmost 8 bits and set the rightmost 24 bits all to 0s. The first address is 73.0.0.0/8, in which 8 is the value of *n*.
3. To find the last address, we keep the leftmost 8 bits and set the rightmost 24 bits all to 1s. The last address is 73.255.255.255.

Figure 5.16 *Solution to Example 5.13*



Example 5.14

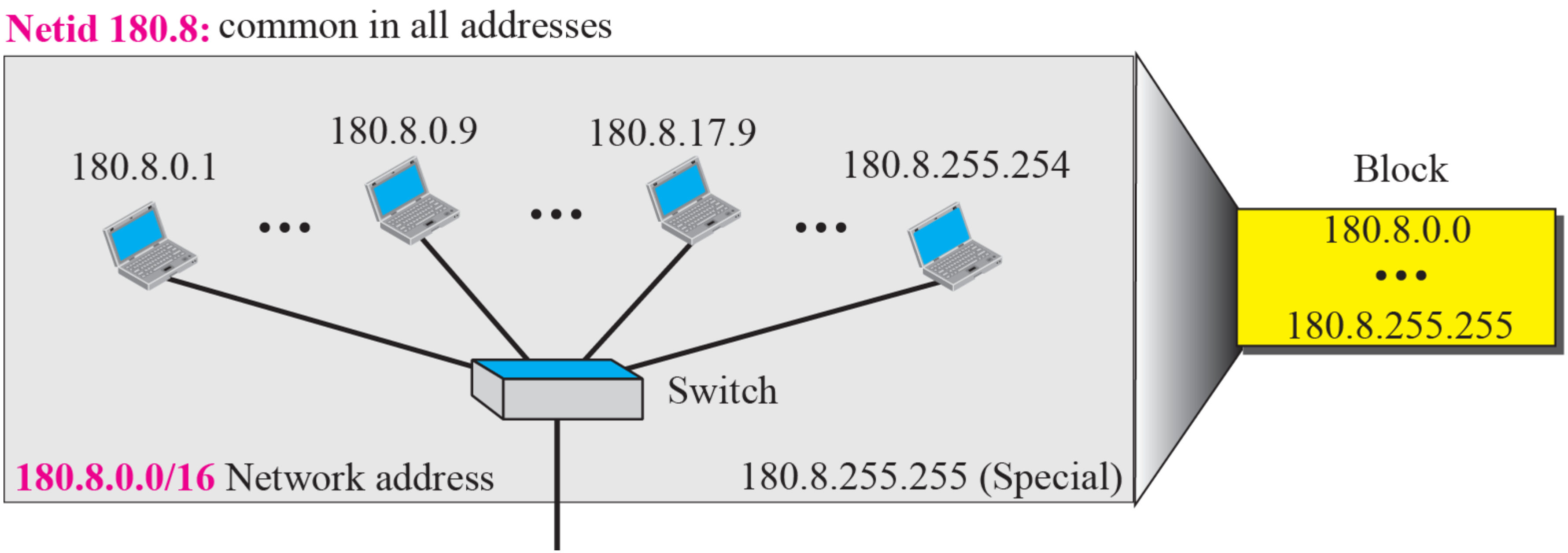
An address in a block is given as 180.8.17. 9 Find the number of addresses in the block, the first address, and the last address.

Solution

Figure 5.17 shows a possible configuration of the network that uses this block. Since the block from class B, so:

1. The number of addresses in this block is $N = 2^{32-n} = 2^{32-16}$ =65,536.
2. To find the first address, we keep the leftmost 16 bits and set the rightmost 16 bits all to 0s. The first address is 180.8.0.0/16, in which 16 is the value of *n*.
3. To find the last address, we keep the leftmost 16 bits and set the rightmost 16 bits all to 1s. The last address is 180.8.255.255.

Figure 5.17 *Solution to Example 5.14*



Example 5.15

An address in a block is given as 200.11.8.45. Find the number of addresses in the block, the first address, and the last address.

Solution

Figure 5.17 shows a possible configuration of the network that uses this block.

1. The number of addresses in this block is $N = 2^{32-n} = 256$.
2. To find the first address, we keep the leftmost 24 bits and set the rightmost 8 bits all to 0s. The first address is 200.11.8.0/24, in which 24 is the value of *n*.
3. To find the last address, we keep the leftmost 24 bits and set the rightmost 8 bits all to 1s. The last address is 200.11.8.255/24.

Figure 5.18 *Solution to Example 5.15*

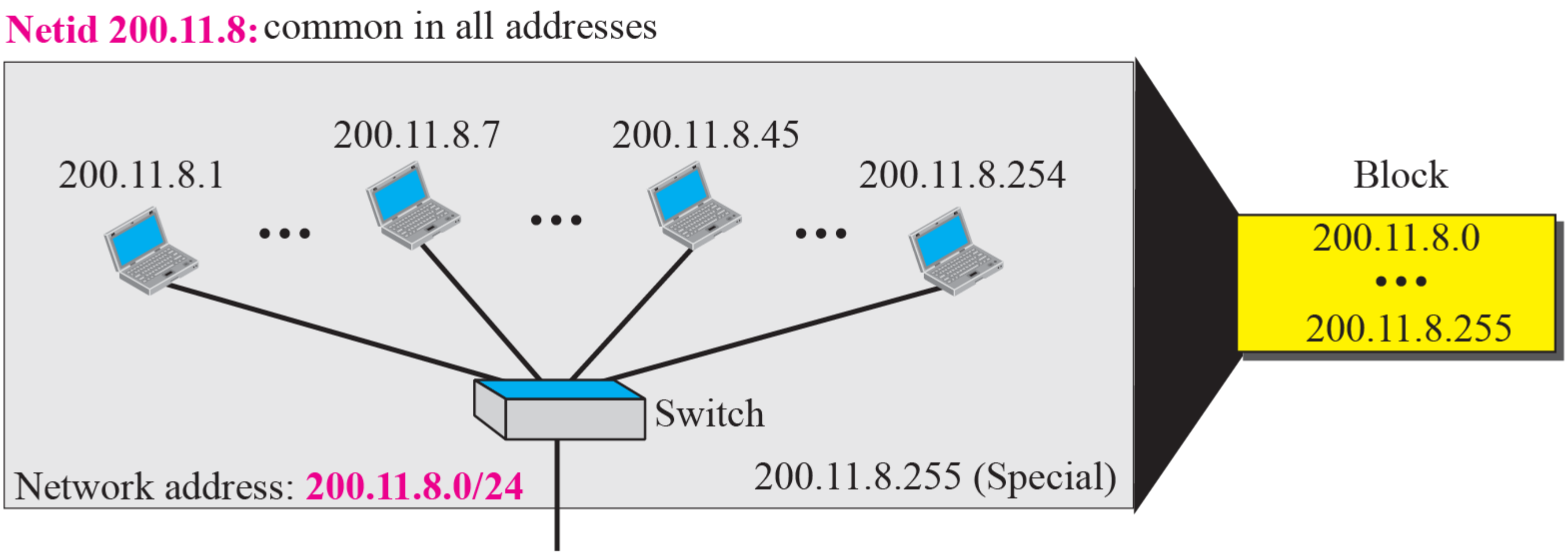
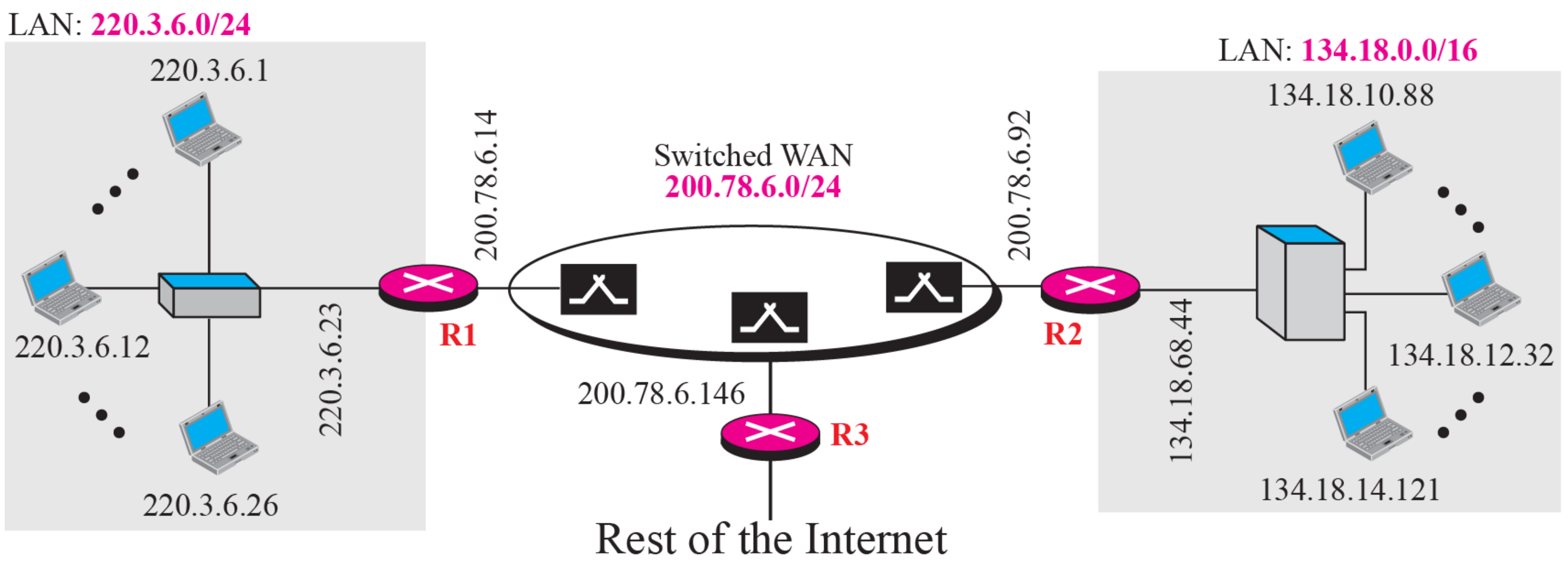


Figure 5.19 *Sample Internet*



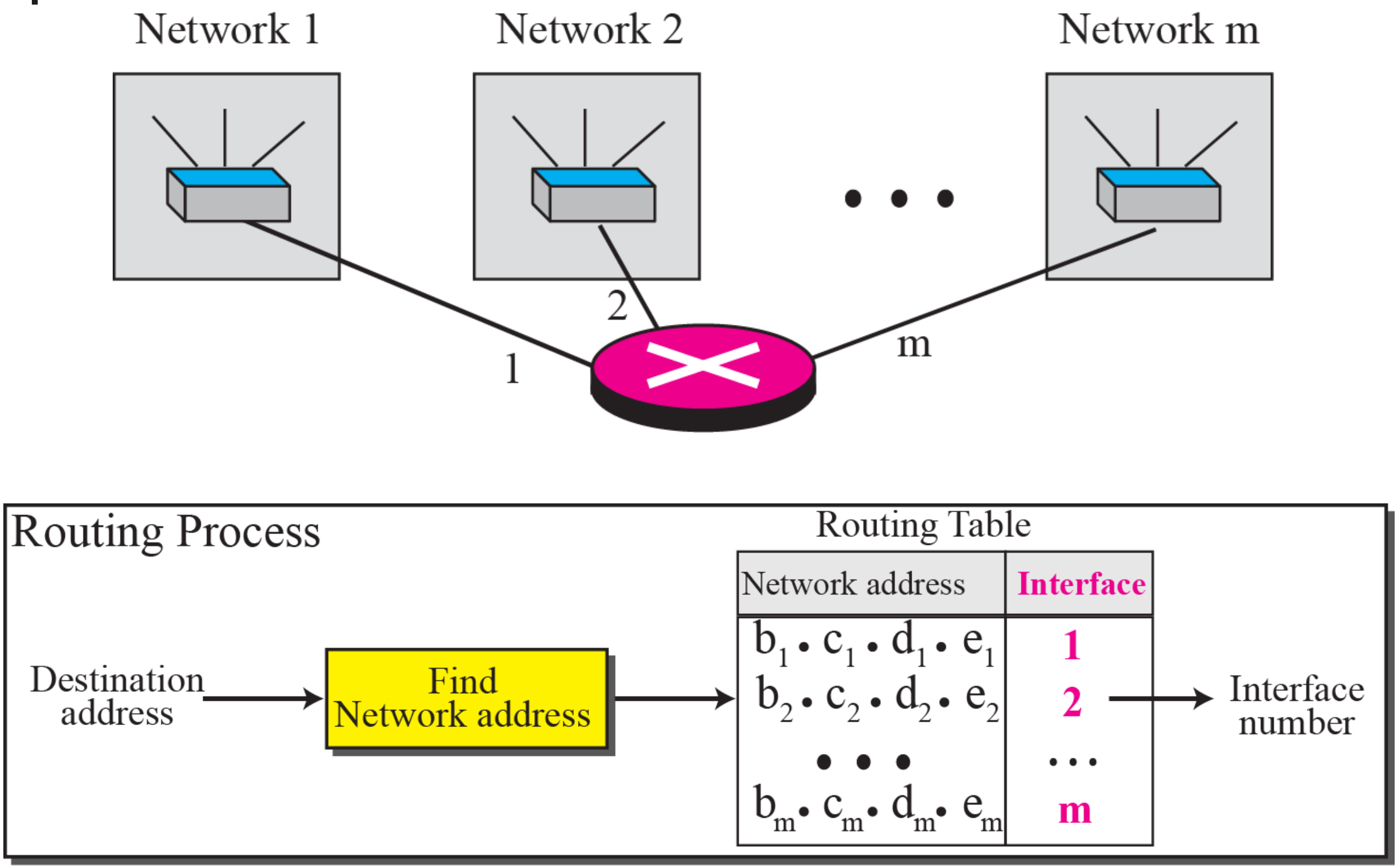
Note

The network address is the identifier of a network.

Note

Network address, is particularly important because it is used in routing a packet to its destination network.

Figure 5.20 *Network addresses*



Network mask

The routers in the Internet normally use an algorithm to extract the **network address** from the **destination address** of a packet. To do this, we need a network mask. A network mask or a default mask in classful addressing is a 32-bit number with **n leftmost bits all set to 1s** and (32 - n) **rightmost bits all set to 0s**. Since n is different for each class in classful addressing, we have three default masks in classful for (A, B and C) classes.

- **The default network mask is used when a network is not subnetted.**

- **When we divide a network to several subnetworks, we need to create a subnetwork mask (or subnet mask) for each subnetwork.**

Figure 5.21 *Network mask*

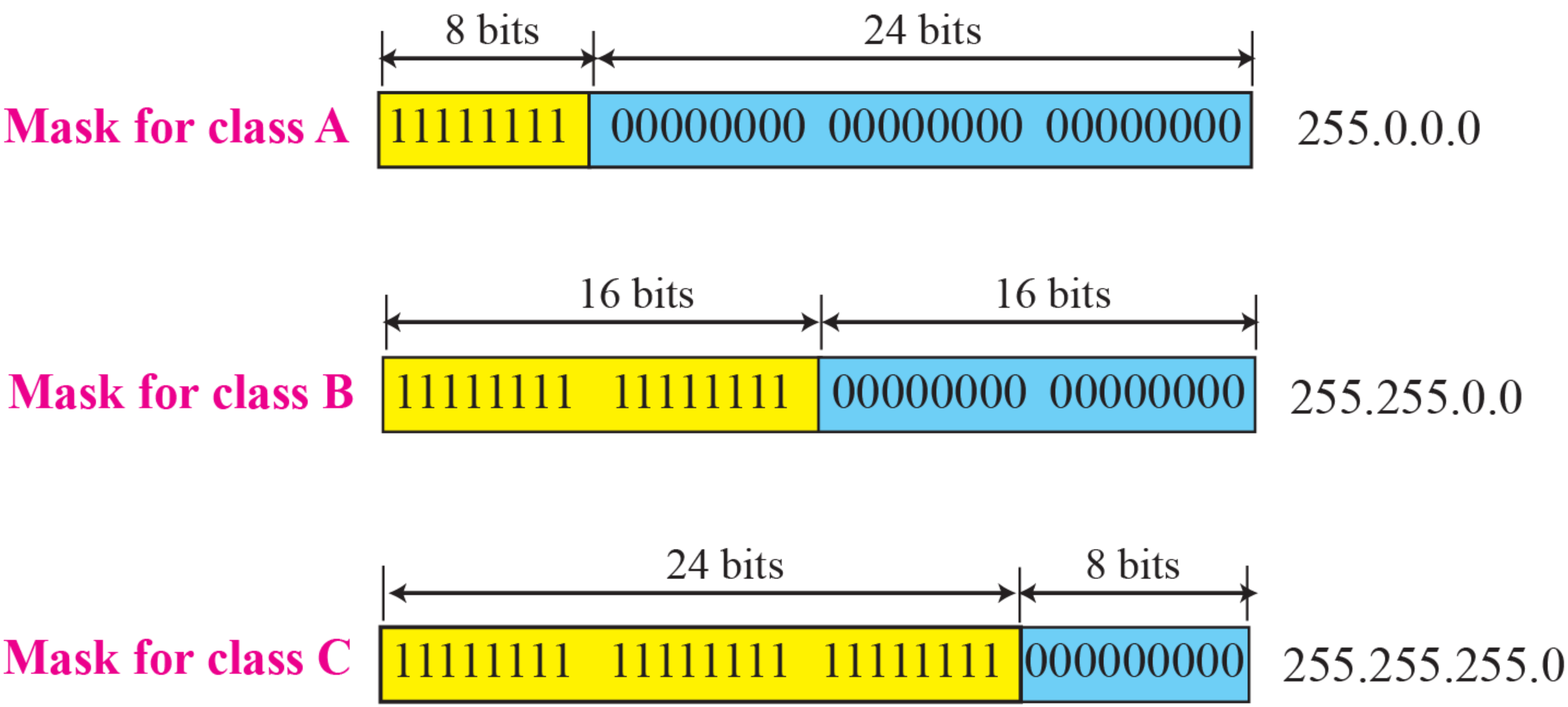
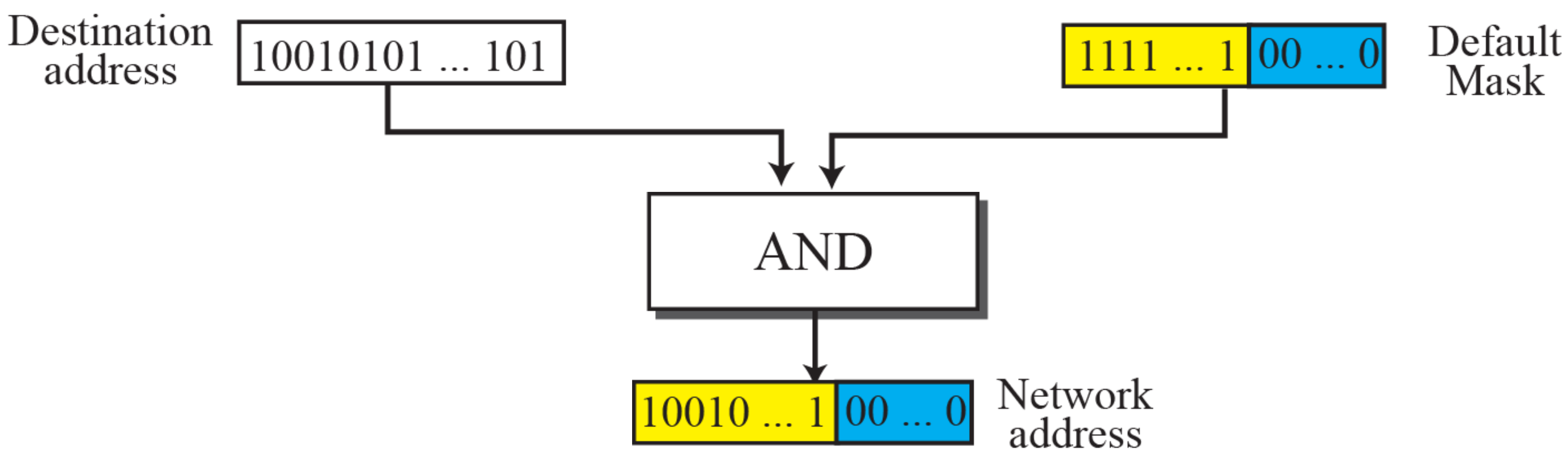


Figure 5.22 *Finding a network address using the default mask*



Example 5.16

A router receives a packet with the destination address 201.24.67.32. Show how the router finds the network address of the packet.

Solution

Since the class of the address is C, we assume that the router applies the default mask for class C, 255.255.255.0 to find the network address.

Destination address	→	201	.	24	.	67	.	32
Default mask	→	255	.	255	.	255	.	0
Network address	→	201	.	24	.	67	.	0

Example 5.17

Three-level addressing can be found in the telephone system if we think about the local part of a telephone number as an exchange and a subscriber connection:

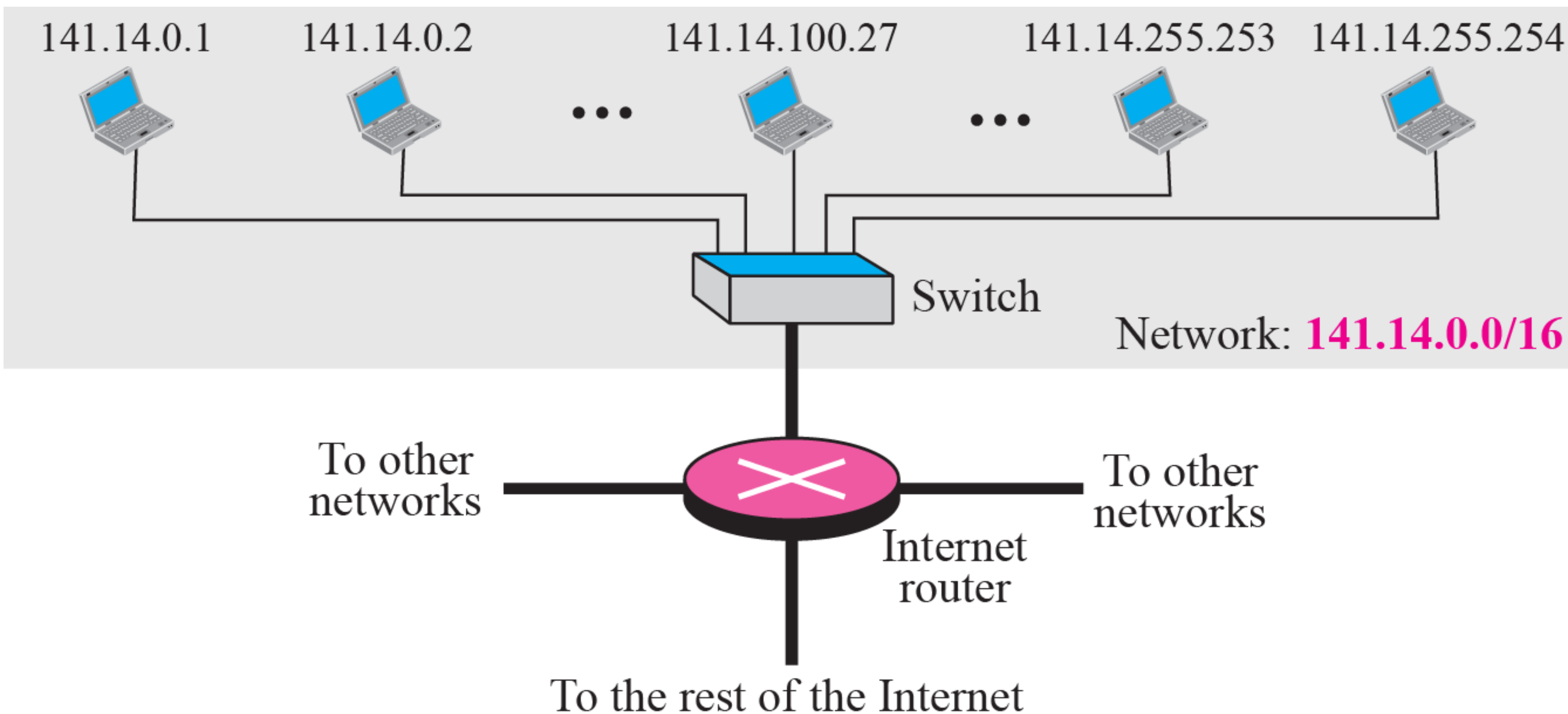
(626) 358 - 1301

in which 626 is the area code, 358 is the exchange, and 1301 is the subscriber connection.

Example 5.18

Figure 5.23 shows a network using class B addresses before subnetting. We have just one network with almost 2^{16} hosts. The whole network is connected, through one single connection, to one of the routers in the Internet. Note that we have shown /16 to show the length of the netid (class B).

Figure 5.23 *Example 5.18*



Example 5.19

Figure 5.24 shows the same network in Figure 5.23 after subnetting. The whole network is still connected to the Internet through the same router. However, the network has used a private router to divide the network into four subnetworks. The rest of the Internet still sees only one network; internally the network is made of four subnetworks. Each subnetwork can now have almost 2^{14} hosts. The network can belong to a university campus with four different schools (buildings). After subnetting, each school has its own subnetworks, but still the whole campus is one network for the rest of the Internet. Note that /16 and /18 show the length of the netid and subnetids.

Figure 5.24 *Example 5.19*

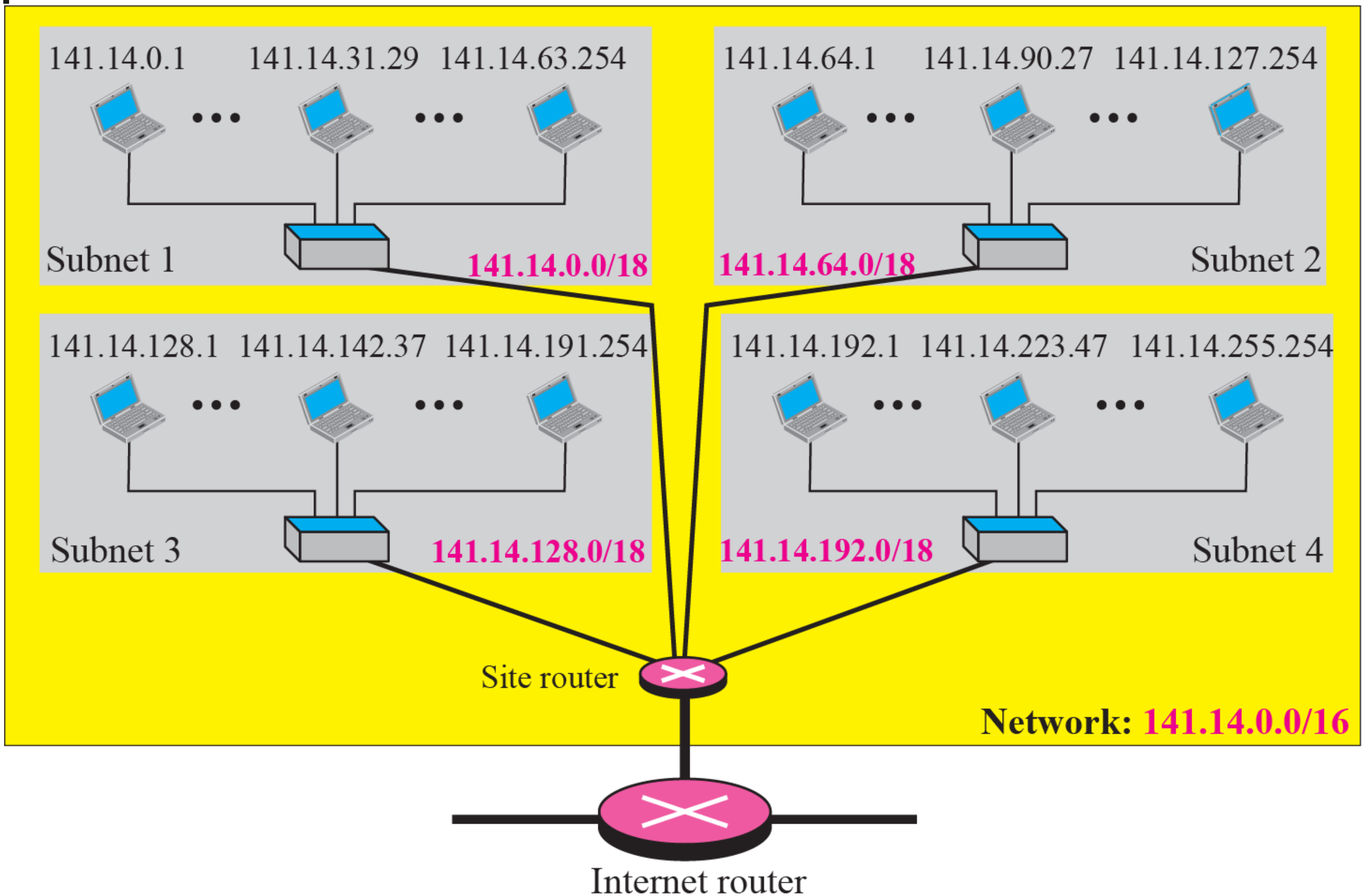
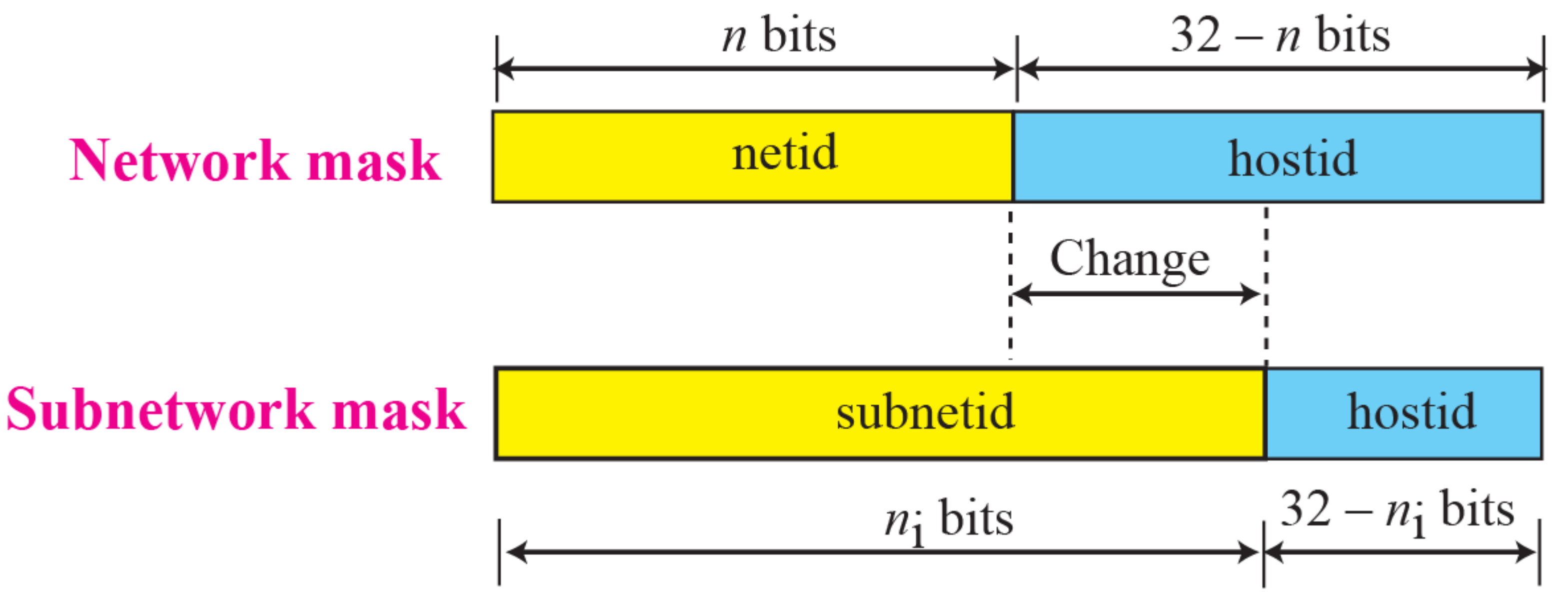


Figure 5.25 *Network mask and subnetwork mask*



Example 5.20

In Example 5.19, we divided a class B network into four subnetworks. The value of n = 16 and the value of

$$n_1 = n_2 = n_3 = n_4 = 16 + \log_2 4 = 18.$$

This means that the subnet mask has eighteen 1s and fourteen 0s. In other words, the subnet mask is 255.255.192.0 which is different from the network mask for class B (255.255.0.0).

Example 5.21

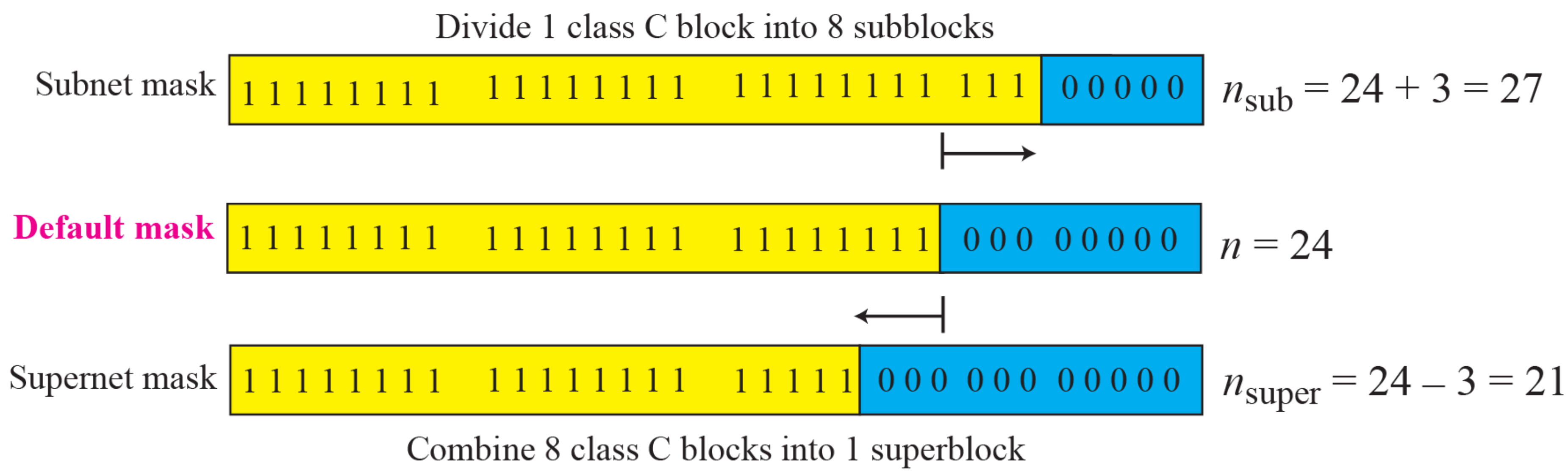
In Example 5.19, we show that a network is divided into four subnets. Since one of the addresses in subnet 2 is 141.14.120.77, we can find the subnet address as:

Address	→	141	.	14	.	120	.	77
Mask	→	255	.	255	.	192	.	0
Subnet Address	→	141	.	14	.	64	.	0

The values of the first, second, and fourth bytes are calculated using the first short cut for AND operation. The value of the third byte is calculated using the second short cut for the AND operation.

Address (120)	0	+	64	+	32	+	16	+	8	+	0	+	0	+	0
Mask (192)	128	+	64	+	0	+	0	+	0	+	0	+	0	+	0
Result (64)	0	+	64	+	0	+	0	+	0	+	0	+	0	+	0

Figure 5.26 *Comparison of subnet, default, and supernet mask*



200.100.0.0
200.100.1.0
200.100.2.0
200.100.3.0

200.100.00000000.00000000 block1
200.100.00000001.00000000 block2
200.100.00000010.00000000 block3
200.100.00000011.00000000 block4

Supernetmask :
255.255.11111100.000000000
255.255.252.0