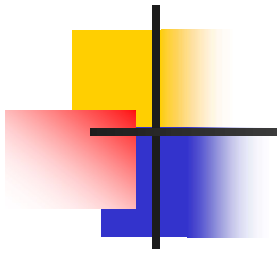
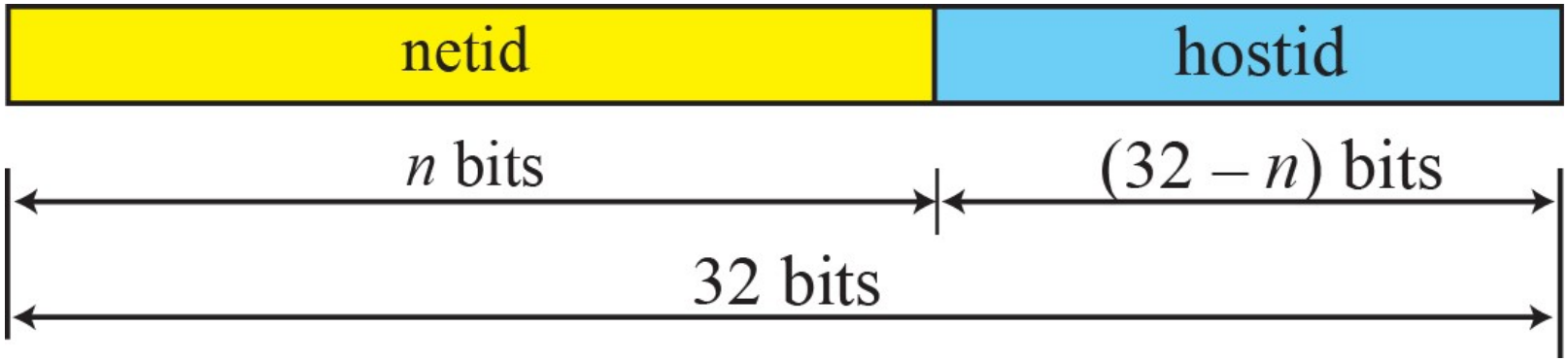


**Note**

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

# Net id and Host id Length

**Fixed netid's and host id's**




**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.

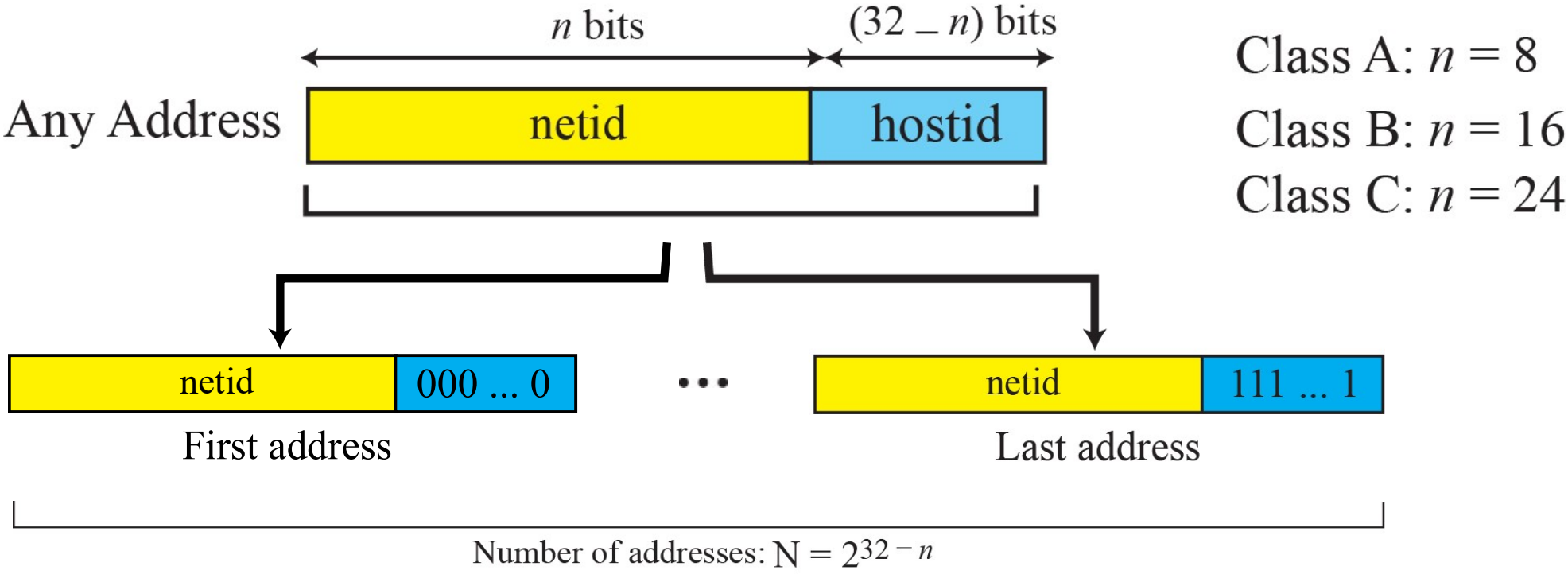
**(626) 3581301**

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

## Figure 5.15 *Information extraction in classful addressing*

**Extracting range of addresses from given block of address (of a class)**

**(number of hosts that can be connected in a network of address class)**




**Note: In a given block of address, 1st and last addresses are special addresses , not assigned to any host.**

# Example 5.13

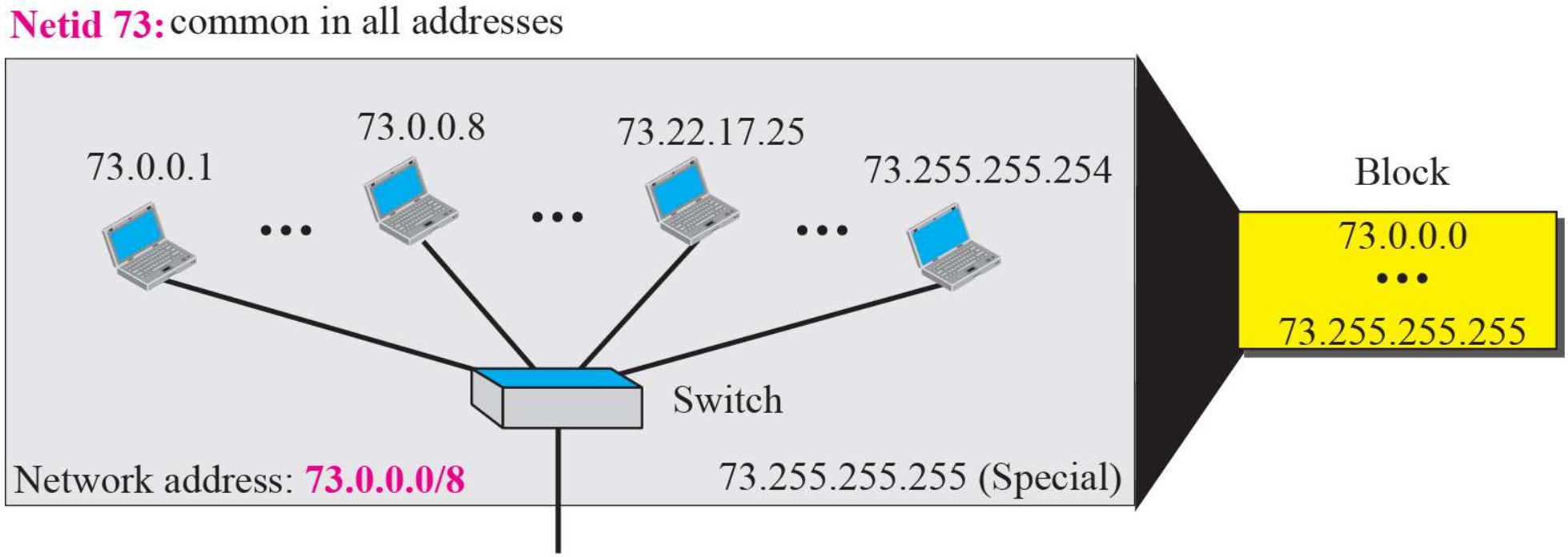
An address in a block is given as 73.22.17.25. Find the number of addresses in the block, the first address, and the last address.

*Solution*

Figure 5.16 shows a possible configuration of the network that uses this block. 73.22.17.25 Class A , hence n=8

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

## Figure 5.16 *Solution to Example 5.13*



# Example 5.14

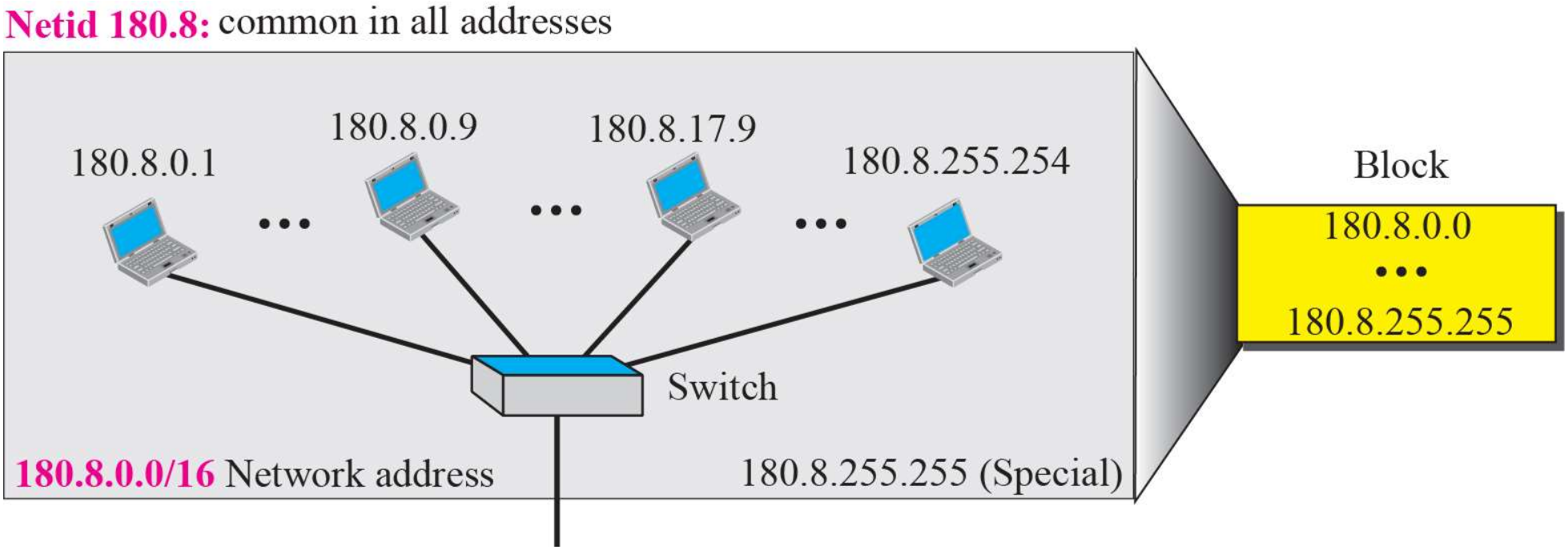
An address in a block is given as 180.8.17.9. Find the number of addresses in the block, the first address, and the last address.

*Solution*

Figure 5.17 shows a possible configuration of the network that uses this block. 180.8.17.9 is Class B, hence n=16

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(netid) 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

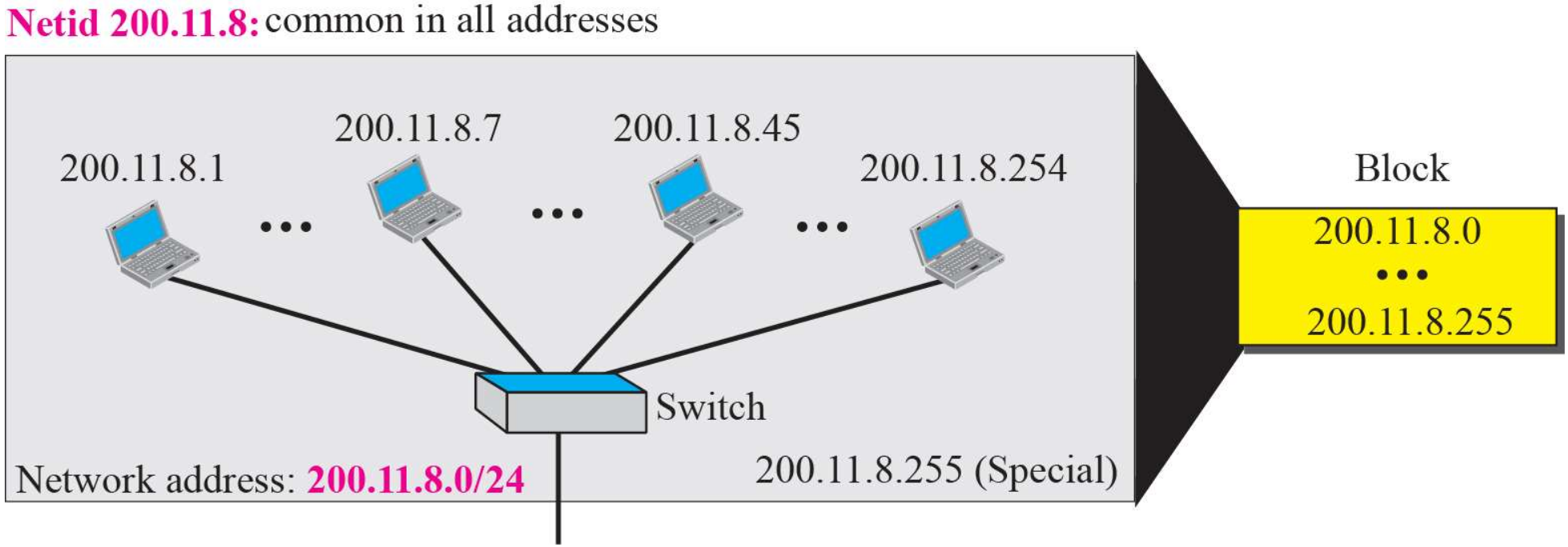
An address in a block is given as 200.11.8.45. Find the number of addresses in the block, the first address, and the last address.

*Solution*

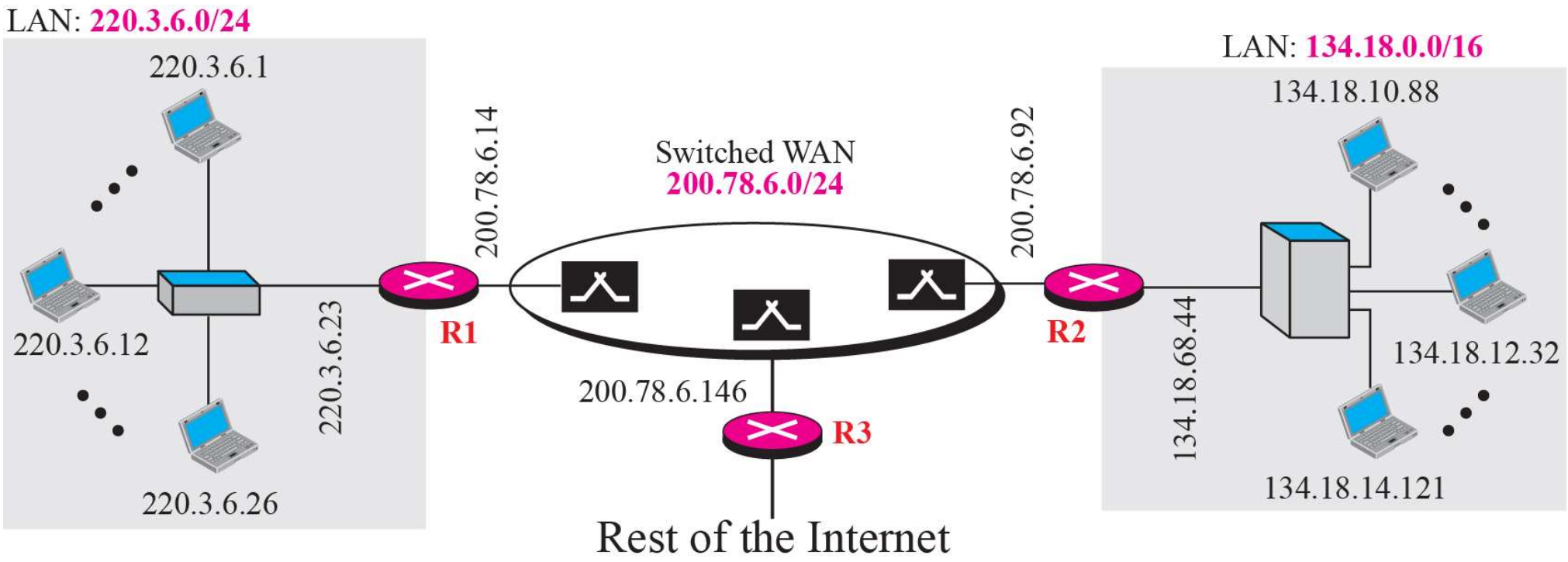
Figure 5.17 shows a possible configuration of the network that uses this block. Class C , so n=24

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.

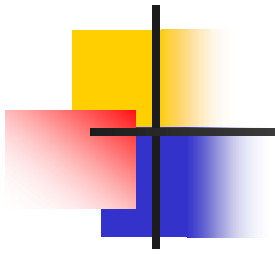
## Figure 5.18 *Solution to Example 5.15*



## Figure 5.19 *Sample Internet*



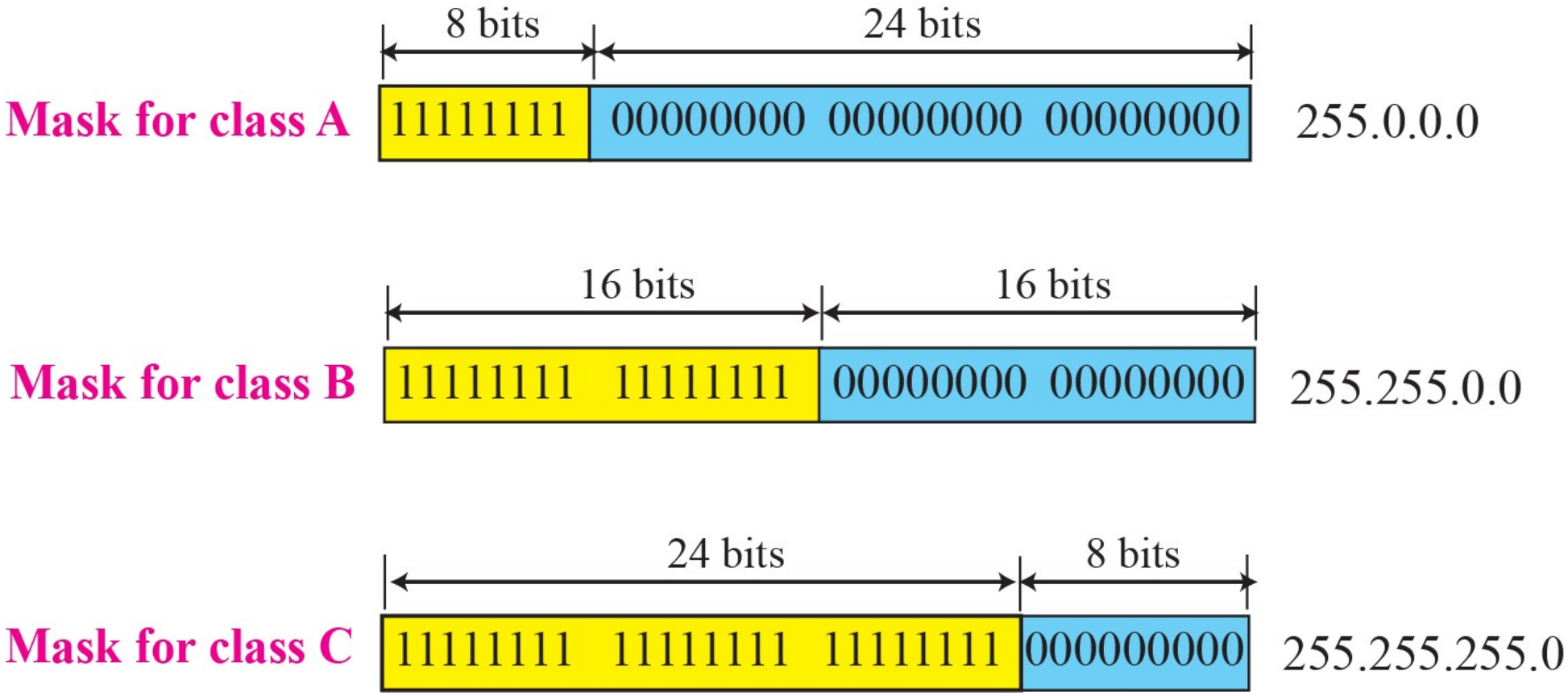

A switched WAN (class C), such as Frame Relay or ATM, that can be connected to many routers.
WAN connects LAN1 ,LAN2 and to the rest of internet.



Note

***The network address is the identifier of a network.***

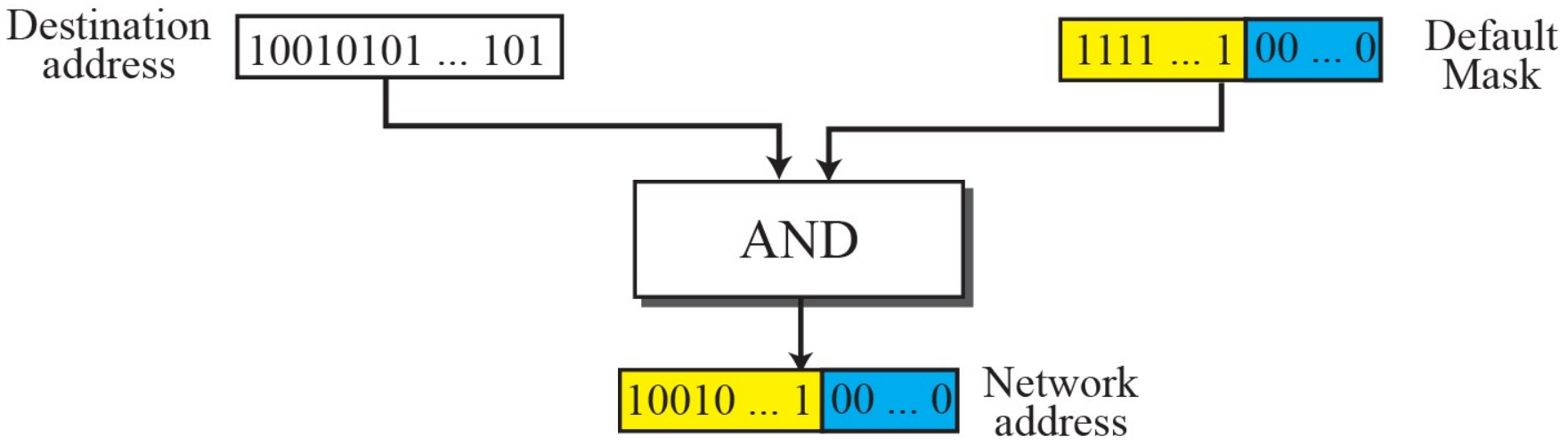
## Figure 5.21 *Network mask* - *Finding Network Address*




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.

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

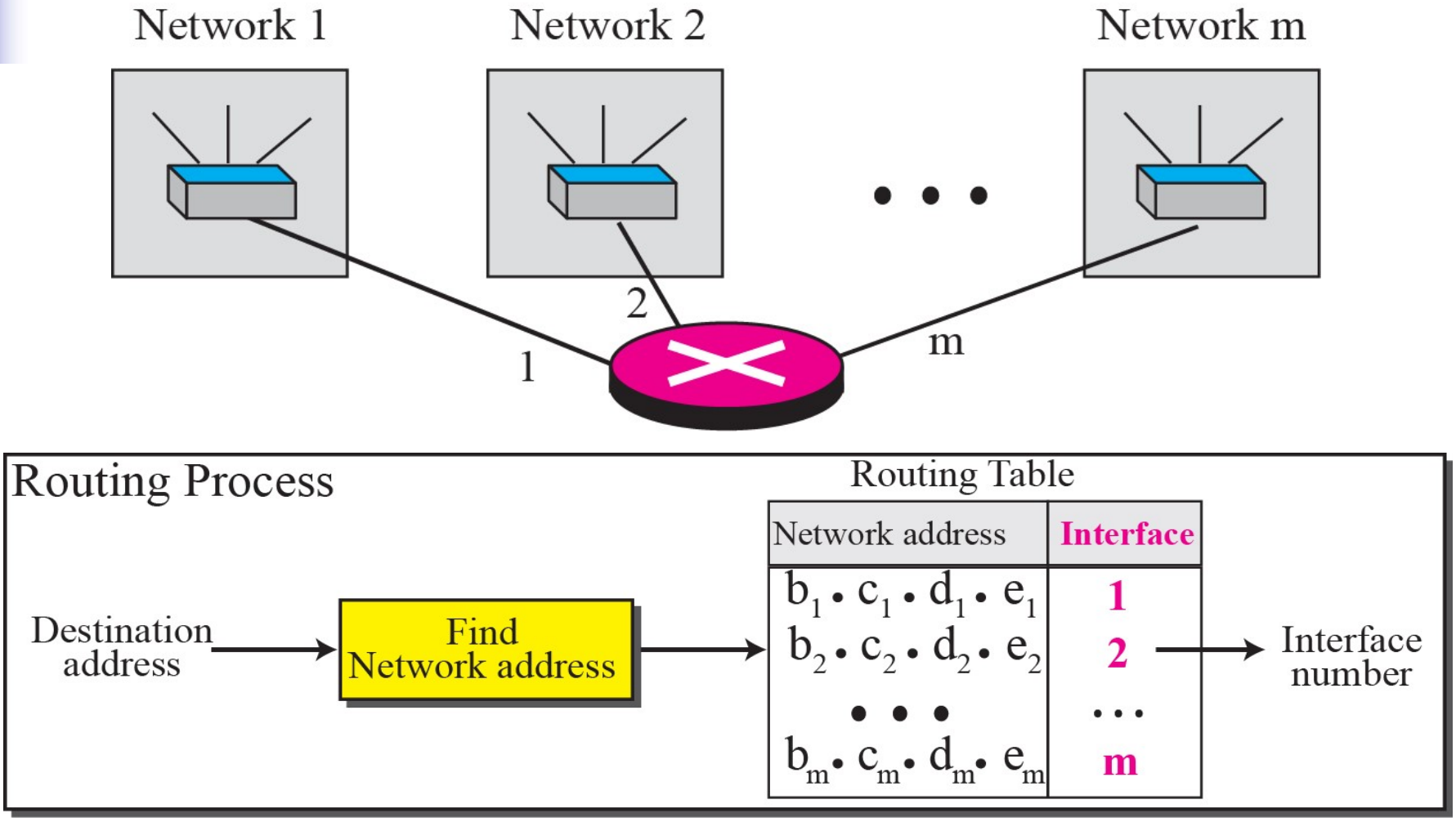
## USE of Network Mask




**Given an address & mask (or prefix length), we can find-**

**1st address (network address) & Last Address of the block to which given address belongs. Refer slide-Extracting Block Information(86-88)**

## Figure 5.20 *Network addresses*




**The network address is the identifier of a network.**
**Router consults Routing Table and decides through which interface of the Router the packet needs to be forwarded to reach a network identified by the netid.( network Address in the destination IP)**

# 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 B, we assume that the router applies the default mask for class B, 255.255.0.0 to find the network address.

Perform, Destination IP AND Default Mask(corresponding class of IP)

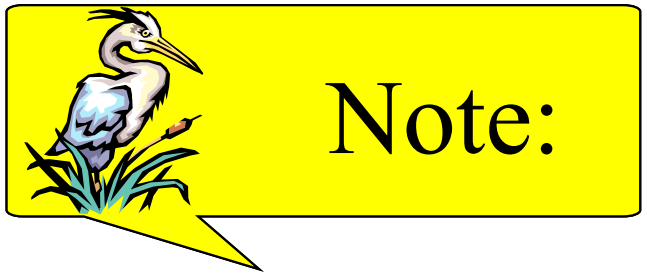
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Destination address | → | 201 | . | 24 | . | 67 | . | 32 |
| Default mask | → | 255 | . | 255 | . | 0 | . | 0 |
| Network address | → | 201 | . | 24 | . | 0 | . | 0 |

# SUBNETTING AND SUPERNETTING

*In the previous sections we discussed the problems associated with classful addressing. Specifically, the network addresses available for assignment to organizations are close to depletion. This is coupled with the ever-increasing demand for addresses from organizations that want connection to the Internet. In this section we briefly discuss two solutions: subnetting and supernetting.*

*The topics discussed in this section include:*

*Subnetting*
*Supernetting*
*Supernet Mask*
*Obsolescence*




***IP addresses are designed with two levels of hierarchy- Netid & Hostid***

# Three-Level Addressing: Subnetting

- The idea of **splitting an address block into smaller blocks** is referred to as **subnetting**.
- In **subnetting,** a network is divided into several smaller subnetworks (subnets) with each subnetwork having its own subnetwork address.

- **Why?**
  - for better security and management.

# 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 country code, 358 is the STD code, and 1301 is the subscriber connection.**

**Analogy to the Subnetting-3 Level hierarchy**

# Example 5.18

Figure 5.23 (next slide) shows a network using class B addresses(141.14.0.0) before subnetting.

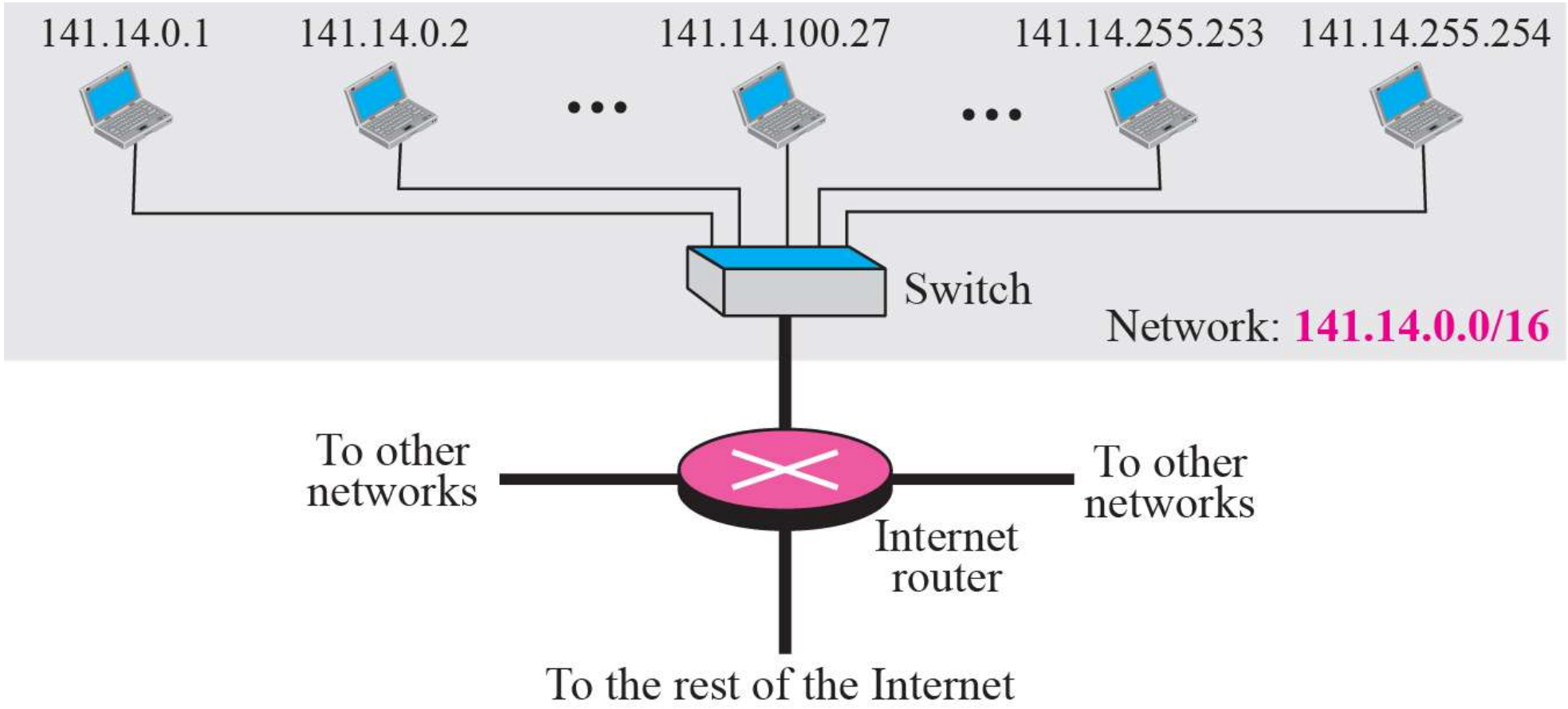
We have just one network with almost $2^{16}$ hosts.

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*

**Uses class B , Number of hosts = $2^{16=}$ = 65536**



# 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.

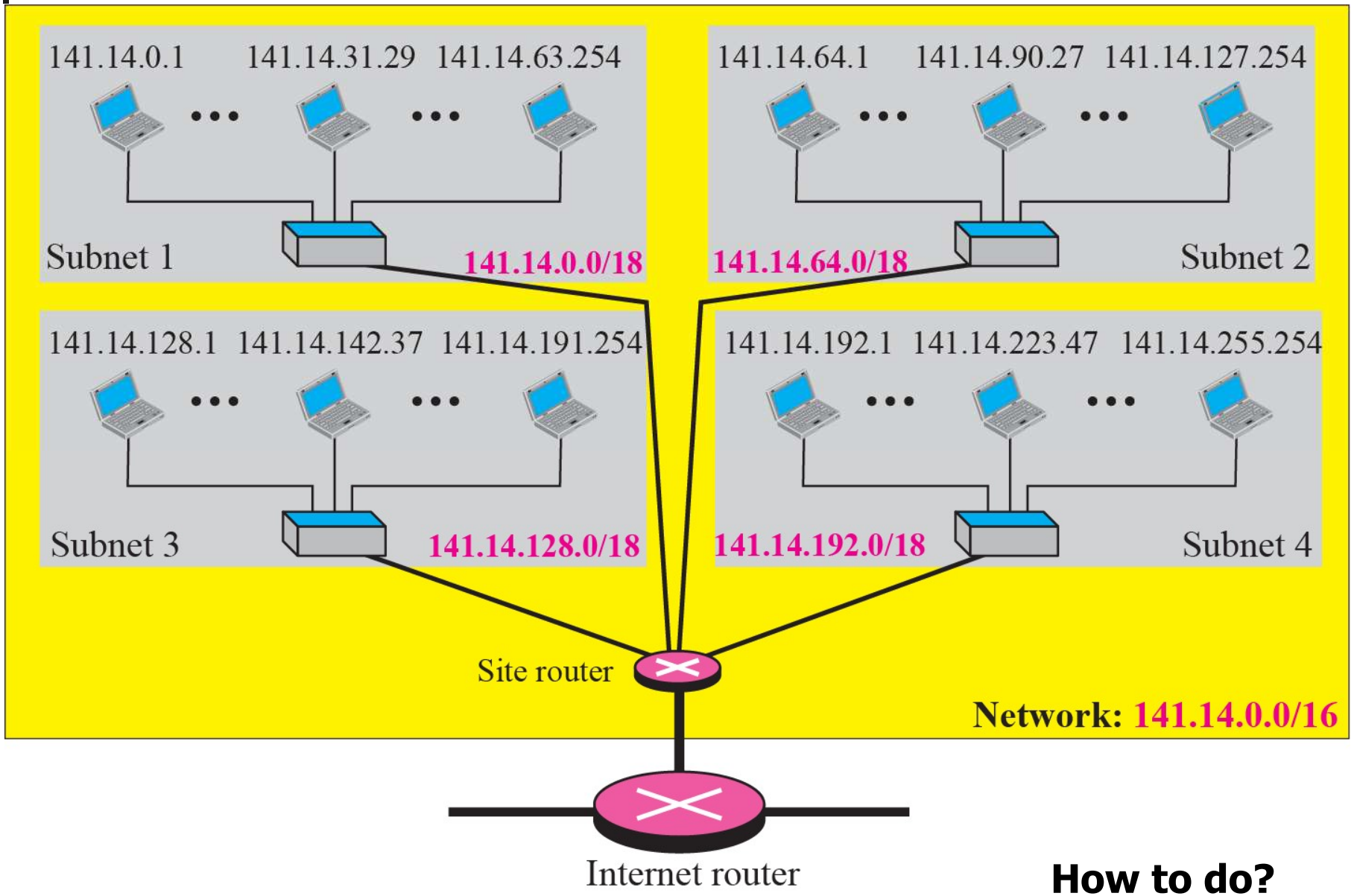
Each subnetwork can now have almost $2^{14}$ hosts.

The network can belong to a university campus with four different schools (buildings).

Example: After subnetting, each school has its own subnetworks, but still the whole campus is one network for the rest of the Internet.

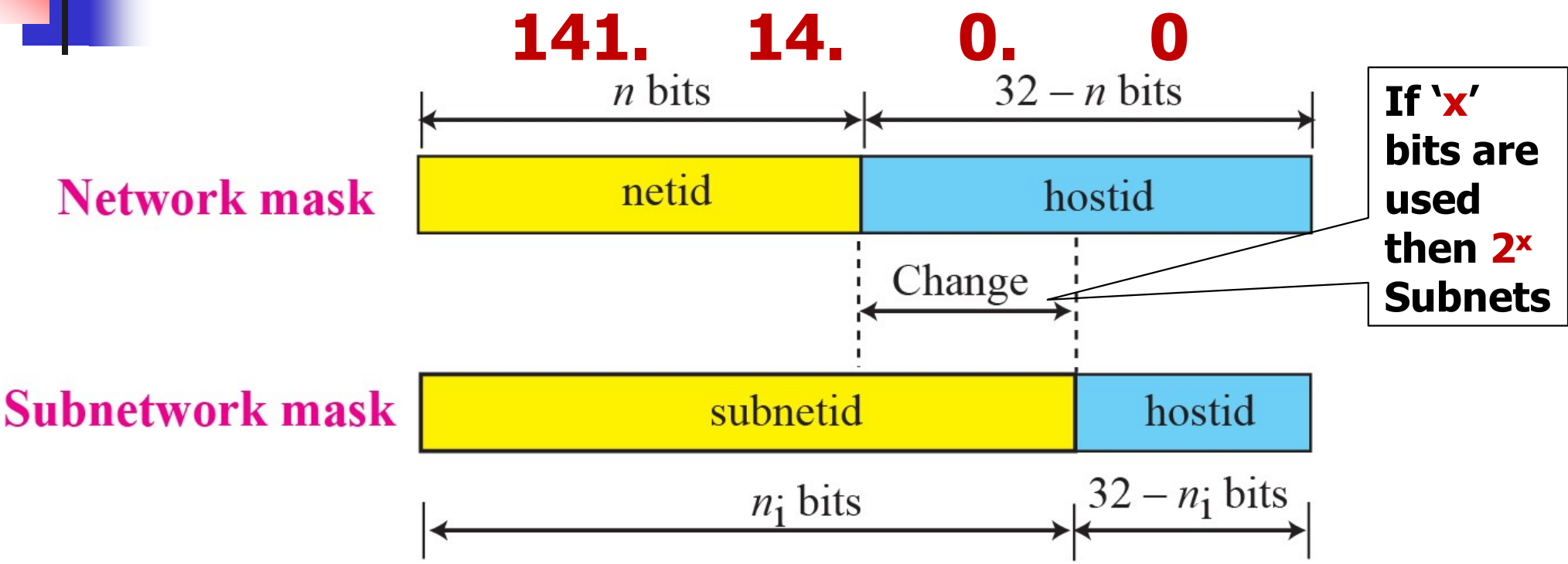
Note that /16 and /18 show the length of the netid and subnetids.

## Figure 5.24 *Example 5.19*




**How to do?**

**Figure 5.25** *Network mask and subnetwork mask*




**Number of Subnets we can create depends on how many bits we take from hostid (32-n).**

**Assume that, we have an address Block 141.14.0.0 to 141.14.255.255**
**We can see, how 2 or 4 subnets can be created by claiming(changing) 1 or 2 bits in 3rd Byte(left most bit from Host ID) from right**

**To create 2 subnets –**
**we have to use 1 rightmost bit of host id (right most 1 bit of 3rd Byte)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Given Network Address** | 10001101 | 00001110 | 0000000 | 00000000 | 141. | 14. | 0. | 0 |
| **Sub Net-1** | | | | | | | | |
| **1st Address** | 10001101 | 00001110 | 0000000 | 00000000 | 141. | 14. | 0. | 0 |
| **Last Address** | 10001101 | 00001110 | 1111111 | 11111111 | 141. | 14. | 127. | 255 |
| **Sub Net-2** | | | | | | | | |
| **1st Address** | 10001101 | 00001110 | 1000000 | 00000000 | 141. | 14. | 128. | 0 |
| **Last Address** | 10001101 | 00001110 | 1111111 | 11111111 | 141. | 14. | 255. | 255 |

Number of Host Addresses without subnet = 256*256 =65536

Number of Addresses in Subnet-1 =128*256 = 32768

Number of Addresses in Subnet-2 =128*256 = 32768

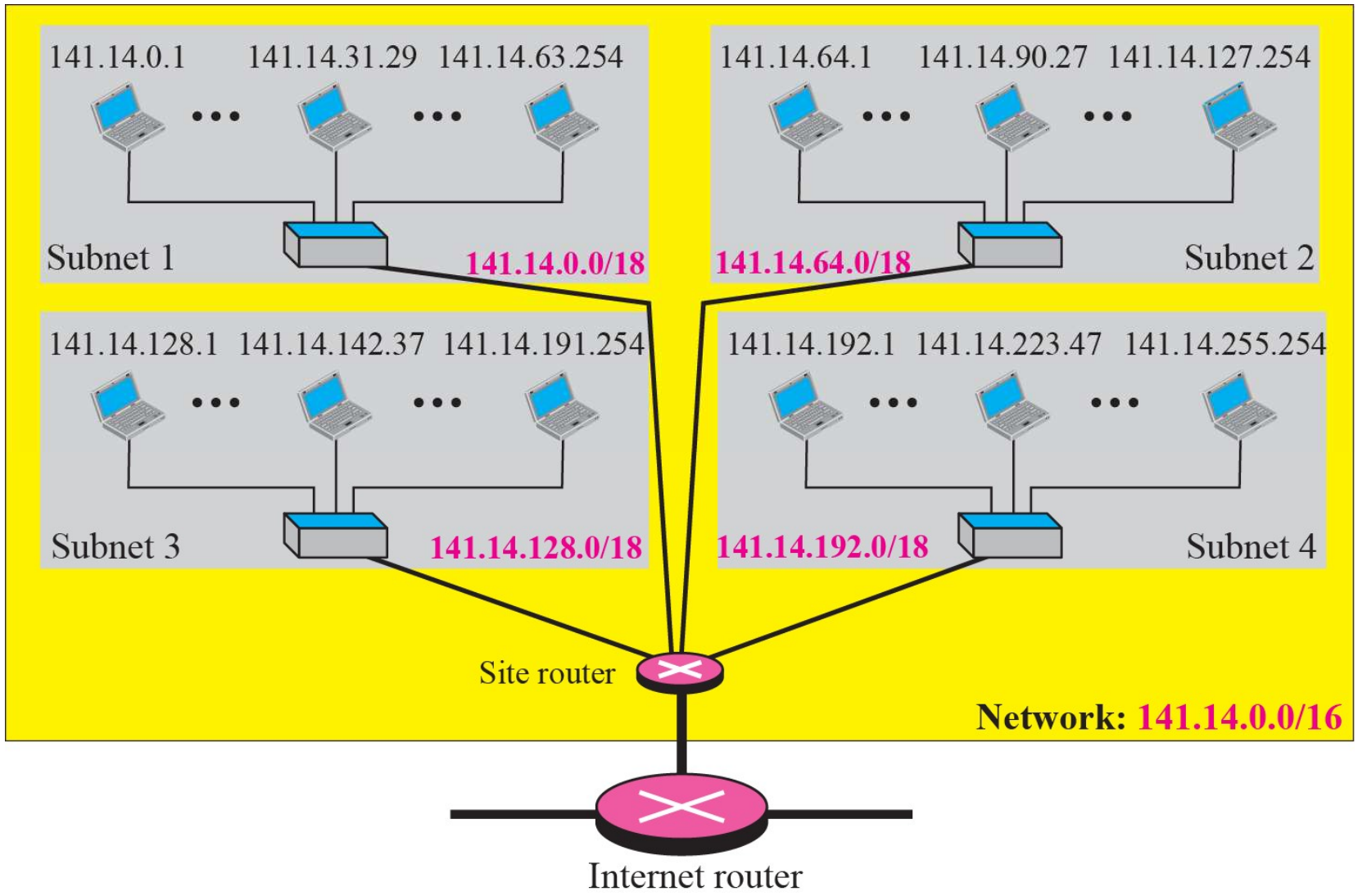
## To Create 4 subnets-

We need 2 right most bits from 3rd byte of the given address (Class B, host id is 3rd & 4th Bytes)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Given Network Address | | 10001101 | 00001110 | 00000000 | 00000000 | 141. | 14. | 0. | 0 |
| 1st Subnet | 1st Address | 10001101 | 00001110 | 00000000 | 00000000 | 141. | 14. | 0. | 0 |
| | Last Address | 10001101 | 00001110 | 00111111 | 11111111 | 141. | 14. | 63. | 255 |
| 2nd subnet | 1st Address | 10001101 | 00001110 | 01000000 | 00000000 | 141. | 14. | 64. | 0 |
| | Last Address | 10001101 | 00001110 | 01111111 | 11111111 | 141. | 14. | 127. | 255 |
| 3rd subnet | 1st Address | 10001101 | 00001110 | 10000000 | 00000000 | 141. | 14. | 128. | 0 |
| | Last Address | 10001101 | 00001110 | 10111111 | 11111111 | 141. | 14. | 191. | 255 |
| 4th subnet | 1st Address | 10001101 | 00001110 | 11000000 | 00000000 | 141. | 14. | 192. | 0 |
| | Last Address | 10001101 | 00001110 | 11111111 | 11111111 | 141. | 14. | 255. | 255 |

See the addresses of Subnets created in the Previous slide

## The Subnets looks as below-



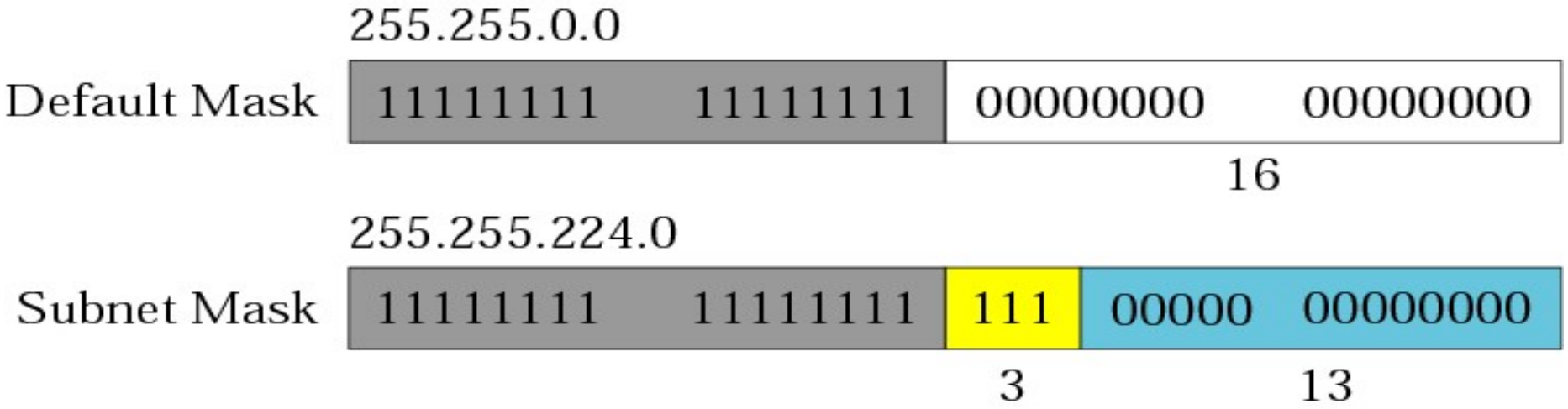

**How the packet is routed to a particular Subnet ?**

**In other words-**

**How site-router identifies each of subnets?**

# *Comparison of a default mask and a subnet mask*

**After creating Subnets, what is the Mask for each subnet?**




**3 bits are used means $2^3 = 8$ Subnets can be created.**

**s- number of subnets and**

**Number of bits required = $\log_2{}^{s} = \log_2{}^{8} = 3$ bits**

**If Given Address Block is Class B (netid=n=16) , then**

**Total bits Required=$n_{sub}=n+\log_2{}^{s}$ =16+3 =19 bits for subnetid and**

**32-19=13 bits for hostid**

**Therefore subnet mask is- Ninteen $1^s$ & remaining Thirteen $0^s$**

**255.255.224.0**

Subnetting increases the length of the netid and decreases the length of hostid.

When we divide a network to ***s*** number of subnetworks, each of equal numbers of hosts, we can calculate the subnetid for each subnetwork as -

$$n_{sub} = n + \log_2 s$$

Where ***n*** is the length of netid,

$n_{sub}$ is the length of each subnetid (net id of each subnet).

***s*** is the number of subnets which **must be a power of 2**.

# Example 5.20

## Finding Subnet Mask

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

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

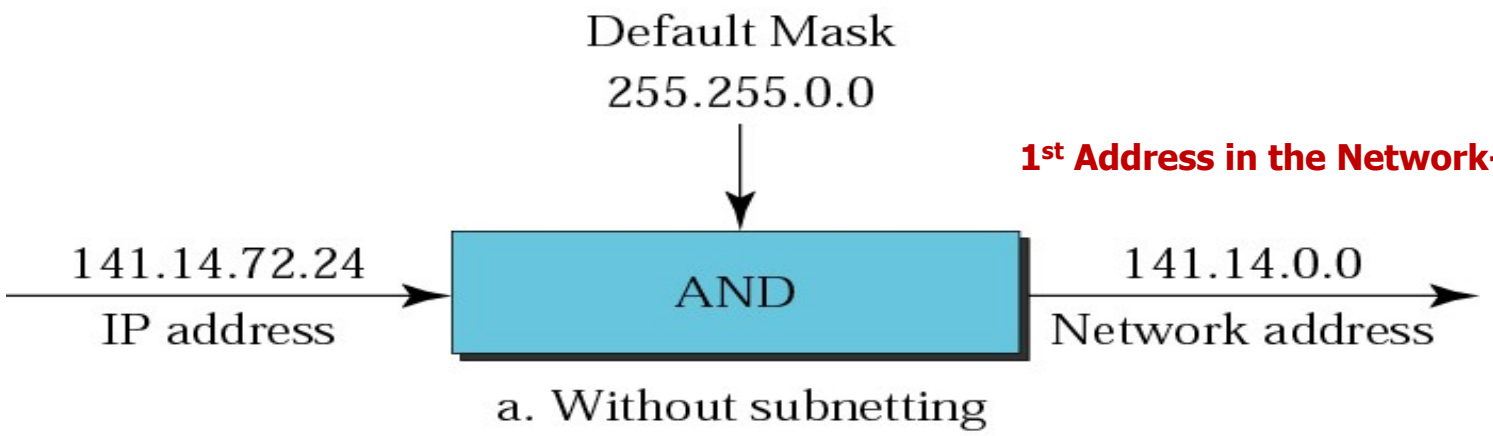
This means that the subnet mask has eighteen $1^s$ & fourteen $0^s$.

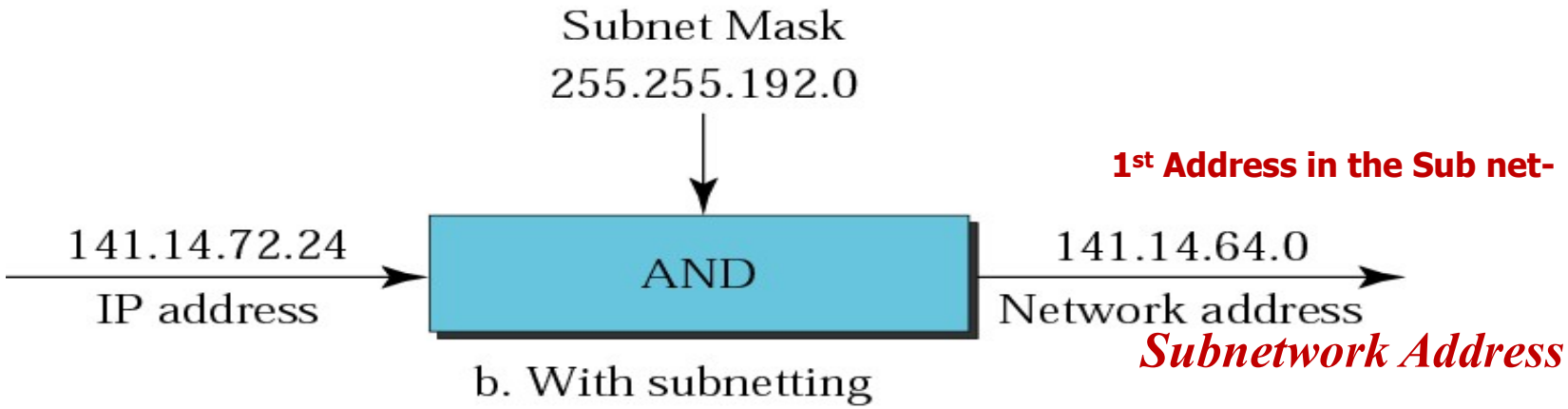
11111111 11111111 11000000 00000000

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).

# Finding Network Address

## Default mask and Subnet Mask




141.14.72.21/18 is a **class B** address –

its **Default Mask** 11111111 11111111 00000000 00000000 **255.255.0.0**

Since we are using 2 right most bits of 3rd Byte to create 4 subnets, the mask for each subnets is as below-

it is called as **Subnet Mask**

11111111 11111111 11000000 00000000 **255.255192.0**

# Example 5.21

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

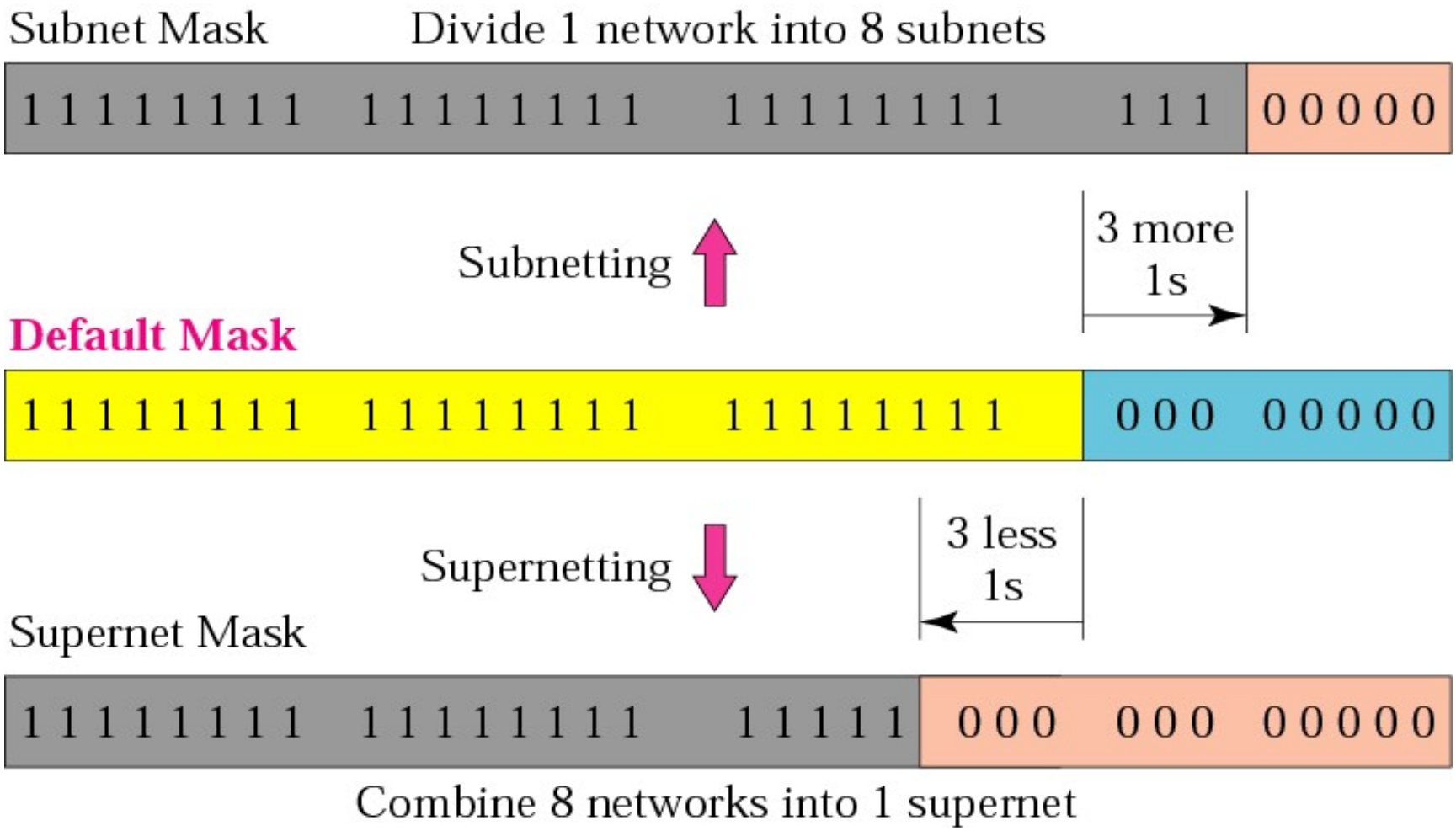
| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Address | → | 141 | . | 14 | . | 120 | . | 77 |
| Mask | → | 255 | . | 255 | . | 192 | . | 0 |
| Subnet Address | → | 141 | . | 14 | . | 64 | . | 0 |

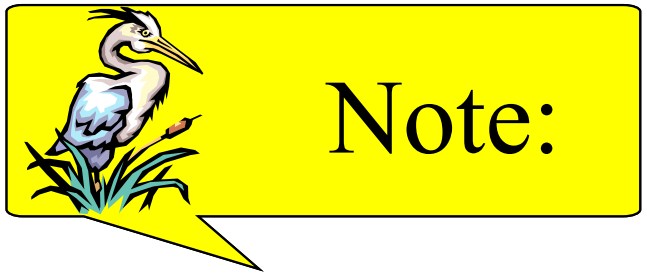
**Shortcut to do AND**

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. (Refer Slide 17 &18) or use calculator to find AND

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

## *Comparison of subnet, default, and supernet masks*





Note:

***The idea of subnetting and supernetting of classful addresses is almost obsolete.***

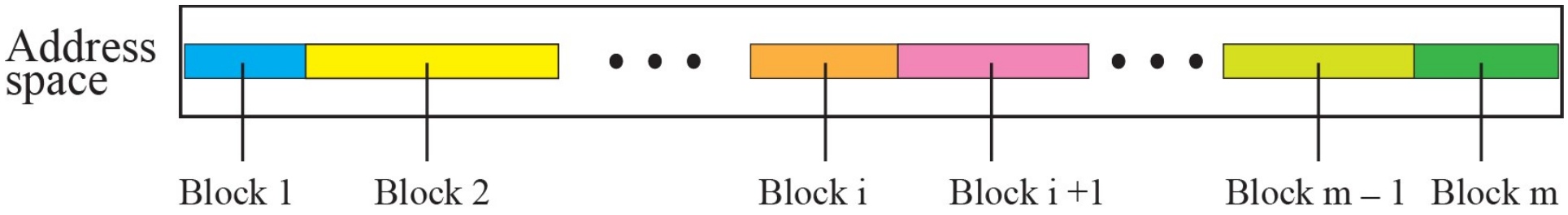
# 5-3 CLASSLESS ADDRESSING

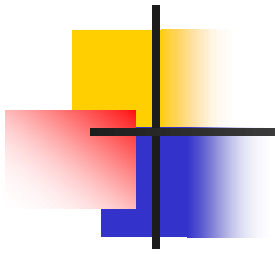
**Subnetting and supernetting in classful addressing did not really solve the address depletion problem. With the growth of the Internet, it was clear that a larger address space was needed as a long-term solution. Although the long-range solution has already been devised and is called IPv6, a short-term solution was also devised to use the same address space but to change the distribution of addresses to provide a fair share to each organization. The short-term solution still uses IPv4 addresses, but it is called *classless addressing*.**

# *Topics Discussed in the Section*

- ✓ **Variable –Length Blocks**
- ✓ **Two-Level Addressing**
- ✓ **Block Allocation**
- ✓ **Subnetting**

## Figure 5.27 *Variable-length blocks in classless addressing*

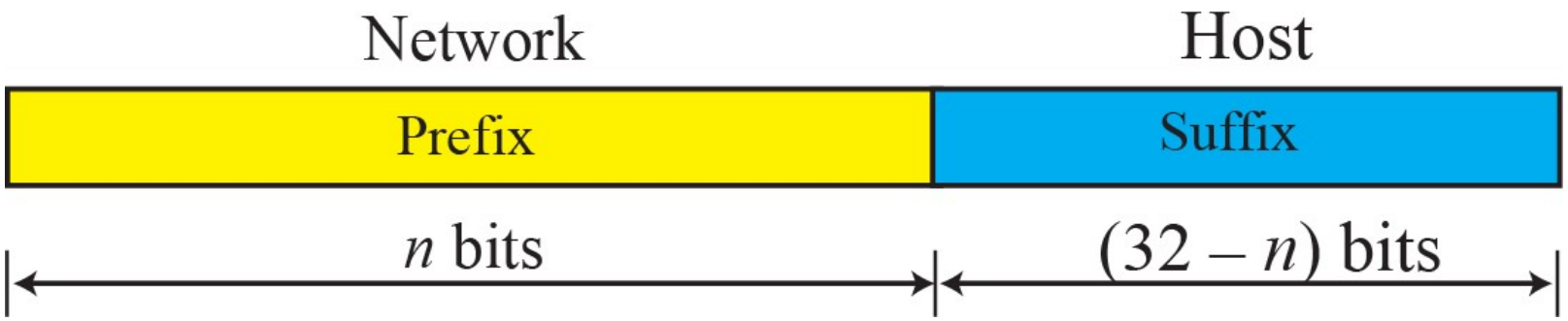




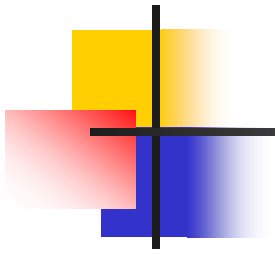
**Note**

***In classless addressing, the prefix defines the network and the suffix defines the host.***

## Figure 5.28 *Prefix and suffix*




***prefix length –number of bits used to define prefix(network) i.e. n bits***



Note

**The prefix length in classless addressing can be 1 to 32.**

# Example 5.22

What is the prefix length and suffix length if the whole Internet is considered as one single block with 4,294,967,296 addresses?

*Solution*

In this case, the prefix length is 0 and the suffix length is 32. All 32 bits vary to define $2^{32}$ = 4,294,967,296 hosts in this single block.

# Example 5.23

What is the prefix length and suffix length if the Internet is divided into 4,294,967,296 blocks and each block has one single address?

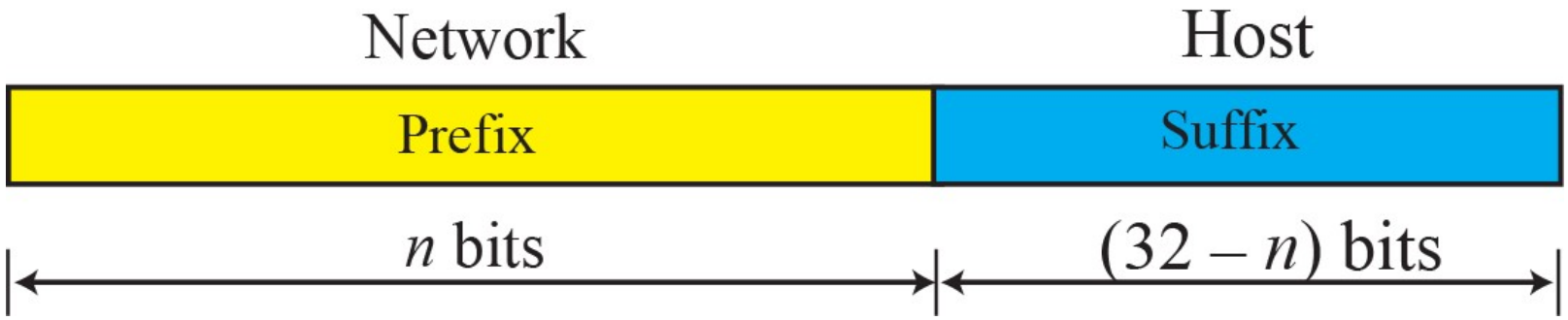
*Solution*

In this case, the prefix length for each block is 32 and the suffix length is 0(zero). All 32 bits are needed to define $2^{32}$ = 4,294,967,296 blocks. The only address in each block is defined by the block itself.

# Example 5.24

**The number of addresses in a block is inversely related to the value of the prefix length, *n*.**

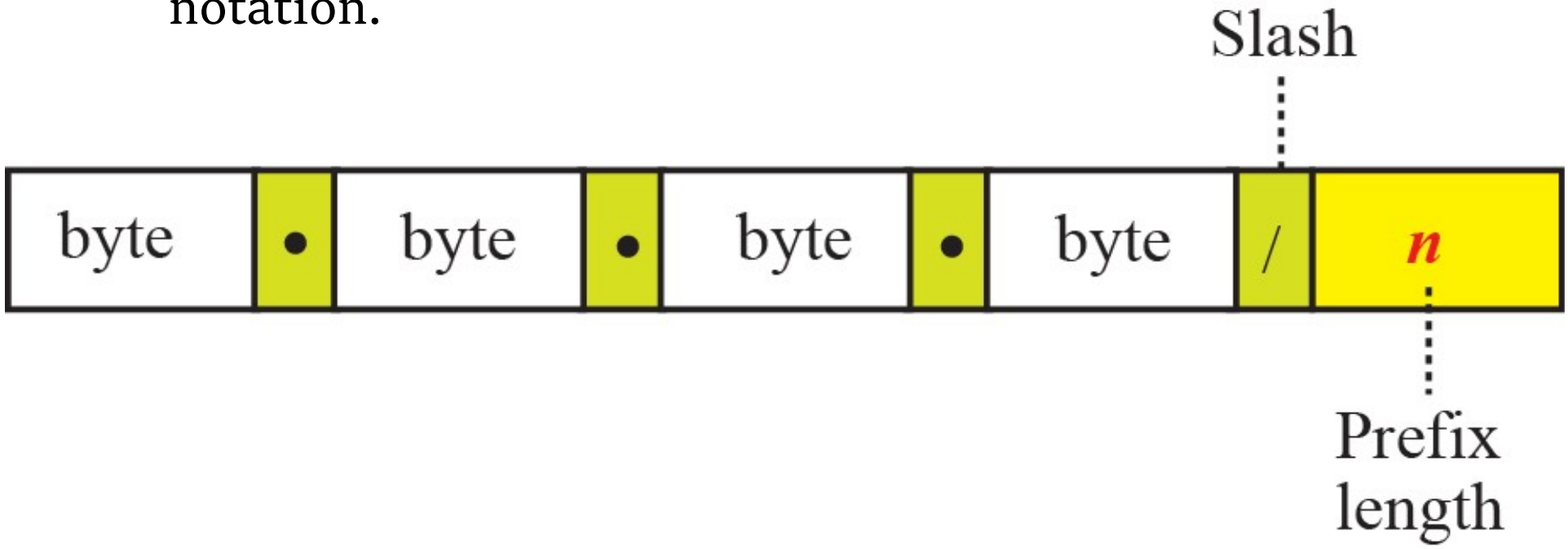
**A small *n* means a larger block; a large *n* means a small block.**




**n (prefix) small means 32-n (Suffix) is large and vice-versa**

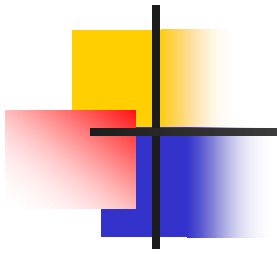
## Figure 5.29 *Slash notation*

The slash notation is formally referred to as **Classless InterDomain Routing** or **CIDR** (pronounced cider) notation.




### *Network Mask*

The idea of network mask in classless addressing is the same as the one in classful addressing. A network mask is a 32-bit number with the ***n* leftmost bits all set to 1s** and the **rest of the bits all set to 0s (zeros).**



**Note**

***In classless addressing, we need to know one of the addresses in the block and the prefix length to define the block.***

# Example 5.25

In classless addressing, an address cannot per se define the block the address belongs to.

For example, the address **230.8.24.56** can belong to many blocks some of them are shown below with the value of the prefix associated with that block:

| | | | | | |
|---|---|---|---|---|---|
| Prefix length:16 | → | Block: | 230.8.0.0 | to | 230.8.255.255 |
| Prefix length:20 | → | Block: | 230.8.16.0 | to | 230.8.31.255 |
| Prefix length:26 | → | Block: | 230.8.24.0 | to | 230.8.24.63 |
| Prefix length:27 | → | Block: | 230.8.24.32 | to | 230.8.24.63 |
| Prefix length:29 | → | Block: | 230.8.24.56 | to | 230.8.24.63 |
| Prefix length:31 | → | Block: | 230.8.24.56 | to | 230.8.24.57 |

# Extracting Block Information

An address in slash notation (CIDR) contains all information we need about the block: (this method **can be applied to classful address also**, because classful is a special case of classless where prefix length is fixed as /8,/16,/24)

1. The **number of addresses** in the block .
2. The **first address** (network address) in the block.
3. The **last address** of the Block.

These three pieces of information can be found as follows:

# Extracting Block Information

**1. The number of addresses in the block.**

The number of addresses in the block can be found as:

$$N=2^{32-n}$$

in which **n** is the prefix length and
**N** is the number of addresses in the block.

**Example: If a given address is** 167.199.170.82/27
n=27 , therefore number of addresses $=2^{32-27}=32$

Similarly, if given address is 167.199.170.82
Since prefix length is not given , let us assume as classful(special case of classless) address. The address is class B address , therefore Prefix length(netid)=16

Number of addresses= $2^{32-16}=2^{16}=65536$

# Extracting Block Information

**2. The first address (network address) in the block.**

The first address (network address) in the block can be found by ANDing (Slide 17,18) the given address with the network mask:

*Alternatively, we can keep the **n** leftmost bits of any address in the block and set the **32- n** bits to **0s** to find the first address.*

**First address = (Any address in the Block) AND (network mask)**

# Extracting Block Information

### 3. The last address of the Block.

The last address in the block can be found by either adding the first address with the number of addresses (256 base number system Arithmetic) or, directly, by ORing (slide 19,20) the address with the complement (NOTing) of the network mask:

**Last address =**

**(any address in the block) OR [NOT (network mask)]**

# Example 5.26

The following addresses are defined using slash notations.

**a.** In the address 12.23.24.78/8, the network mask is 255.0.0.0. The mask has eight 1s and twenty-four 0s. The prefix length is 8; the suffix length is 24.

**b.** In the address 130.11.232.156/16, the network mask is 255.255.0.0. The mask has sixteen 1s and sixteen 0s.The prefix length is 16; the suffix length is 16.

**c.** In the address 167.199.170.82/27, the network mask is 255.255.255.224. The mask has twenty-seven 1s and five 0s. The prefix length is 27; the suffix length is 5.

# Example 5.27

One of the addresses in a block is 167.199.170.82/27. Find the number of addresses in the network, the first address, and the last address.

***Solution***

The value of n is 27. The network mask has twenty-seven 1s and five 0s (Zeros). It is 255.255.255.240.

**a.** The number of addresses in the network is $2^{32-n} = 32$.

**b.** We use the AND operation to find the first address (network-address). The first address is 167.199.170.64/27.

167.199.170.82 AND 255.255.255.240 = 167.199.170.64

| | |
|---|---|
| Address in binary: | 10100111 11000111 10101010 01010010 |
| Network mask: | 11111111 11111111 11111111 11100000 |
| First address: | 10100111 11000111 10101010 01000000 |

# Example 5.27 *Continued*

c. To find the last address, we first find the complement of the network mask and then OR it with the given address:

(how to Perform OR slide 19 & 20)

The last address is 167.199.170.95/27.

| | |
|---|---|
| Address in binary: | 10100111 11000111 10101010 01010010 |
| Complement of network mask: | 00000000 00000000 00000000 00011111 |
| Last address: | 10100111 11000111 10101010 01011111 |

**167.199.170.82 OR NOT(255.255.255.240)=**

```
   167.199.170. 82
OR   0.   0.   0. 15
=====================
   167.199.170. 95
```

**(refer notes section for 82 OR 15 or slide 19,20)**

**if we use 256 base addition**

167.199.170.64 + **0.0.0.15=**167.199.170.95

# Example 5.28

One of the addresses in a block is 17.63.110.114/24. Find the number of addresses, the first address, and the last address in the block.

***Solution***

The network **mask is 255.255.255.0.**

a. The number of addresses in the network is $2^{32-24} = 256$.

b. To find the **first address**, we use the short cut methods discussed early in the chapter. The **first address** is **17.63.110.0/24.**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Address: | 17 | . | 63 | . | 110 | . | 114 |
| Network mask: | 255 | . | 255 | . | 255 | . | 0 |
| First address (AND): | 17 | . | 63 | . | 110 | . | 0 |

# Example 5.28 *Continued*

**c.** To find the last address, we use the complement of the network mask and the first short cut method we discussed before.

The last address is 17.63.110.255/24.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Address: | 17 | . | 63 | . | 110 | . | 114 |
| Complement of the mask (NOT): | 0 | . | 0 | . | 0 | . | 255 |
| Last address (OR): | 17 | . | 63 | . | 110 | . | 255 |

# Example 5.29

One of the addresses in a block is 110.23.120.14/20.

Find the number of addresses, the first address, and the last address in the block.

***Solution***

The network mask is 255.255.240.0.

a. The number of addresses in the network is

$$2^{32-20} = 4096.$$

**b.** To find the first address, we apply the first short cut to bytes 1, 2, and 4 and the second short cut to byte 3. The **first address** is 110.23.112.0/20.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Address: | 110 | . | 23 | . | 120 | . | 14 |
| Network mask: | 255 | . | 255 | . | 240 | . | 0 |
| First address (AND): | 110 | . | 23 | . | 112 | . | 0 |

# Example 5.29 *Continued*

**c.** To find the **last address**, we apply the first short cut to bytes 1, 2, and 4 and the second short cut to byte 3 (slide 19 &20).

The OR operation is applied to the complement of the mask.

The **last address** is 110.23.127.255/20.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Address: | 110 | . | 23 | . | 120 | . | 14 |
| Network mask: | 0 | . | 0 | . | 15 | . | 255 |
| Last address (OR): | 110 | . | 23 | . | 127 | . | 255 |

**if we use 256 base addition**

110.23.112.0 (1st Address)
\+ 0. 0. 15.255
==========
**110.23.127.255**

## Another Method

## Using 256 number system Addition

## Slide 93- 96

## EXAMPLE 5

**What is the first address in the block if one of the addresses is 140.120.84.24/20?**

### Solution

**The first, second, and fourth bytes are easy; for the third byte we keep the bits corresponding to the number of 1s in that group.**

**The first address is 140.120.80.0/20.**

*140. 120. 84. 24*

*10001100 01111000 01010100 00011000 140.120.84.24/20*

*20 Bits as it is*

*140. 120 80 0*

*10001100 01111000 01010000 00000000 140.120.80.0/20*

## EXAMPLE 7

**Find the number of addresses in the block if one of the addresses is 140.120.84.24/20.**

**Solution**

**The prefix length is 20. The number of addresses in the block is $2^{32-20}$ or $2^{12}$ or 4096. Note that this is a large block with 4096 addresses.**

## EXAMPLE 8

**Find the last address in the block if one of the addresses is 140.120.84.24/20.**

### Solution

**We found in the previous examples that the first address is 140.120.80.0/20 and the number of addresses is 4096.**

**To find the last address, we need to add 4095 (4096 – 1) to the first address.**

See Next Slide

## EXAMPLE 8 (Continued)

**To keep the format in dotted-decimal notation, we need to represent 4095 in base 256, and do the calculation in base 256. We write 4095 as 15.255. We then add the first address to this number (in base 256) to obtain the last address as shown below:**

```
   140 . 120 .  80 .   0
+             15  .  255
-----------------------
   140 . 120 .  95 .  255
```

**The last address is 140.120.95.255/20.**

# Block Allocation

The ultimate responsibility of block allocation is given to a global authority called the Internet Corporation for Assigned Names and Addresses (**ICANN**).

For the proper operation of the CIDR, three restrictions need to be applied to the allocated block.

**1.** The number of requested addresses, ***N***, needs to be a **power of 2**.

This is needed to provide an integer value for the prefix length, *n* (see the second restriction). The **number of addresses** can be **1, 2, 4, 8, 16**, and **so on**.

**2.** The value of **prefix length (*n*)** can be found from the number of addresses in the block. Since $N = 2^{32-n}$, then $n = \log2\ (2^{32}/N) = 32 - \log_2 N$. That is the reason why *N* needs to be a power of 2.

**3.** The requested block needs to be allocated where there are a **contiguous** number of unallocated addresses in the address space. However, there is a restriction on **choosing the beginning addresses** of the block. The beginning address needs to be **divisible by the number of addresses** in the block.

## EXAMPLE 1

*Which of the following can be the beginning address of a block that contains 16 addresses?*

*a. 205.16.37.32* *b.190.16.42.44*

*c. 17.17.33.80* *d.123.45.24.52*

### Solution

*Only two are eligible (a and c). The address 205.16.37.32 is eligible because 32 is divisible by 16. The address 17.17.33.80 is eligible because 80 is divisible by 16.*

## EXAMPLE 2

*Which of the following can be the beginning address of a block that contains 256 addresses?*

a. 205.16.37.32 b. 190.16.42.0

c. 17.17.32.0 d. 123.45.24.52

### Solution

*In this case, the right-most byte must be 0. As we know, the IP addresses use base 256 arithmetic.* $256_{(10)} = 0.0.0.255_{(256)}$ *When the right-most byte is 0(zero), the total address is divisible by 255. Only two addresses are eligible (b and c).*

## EXAMPLE 3

*Which of the following can be the beginning address of a block that contains 1024 (Note: more than 256) addresses?*

a. 205.16.37.32

b. 190.16.42.0

c. 17.17.32.0

d. 123.45.24.52

### Solution

*In this case, we need to check two bytes because 1024 = 4 × 256 and 256 base equivalent number is 0.0.4.0 The right-most byte must be divisible by 0. The second byte (from the right) must be divisible by 4. Only one address is eligible (c).*

# Example 5.30

**An ISP has requested a block of 1000 addresses. The block 18.14.12.0/22 is granted.**

**a) How many addresses ICANN allot.**

**b) Find prefix length, mask.**

**c) Which addresses can be a 1st , last address of the allotted block.**

# Example 5.30

a. Since 1000 is not a power of 2, 1024 addresses are granted ($1024 = 2^{10}$). Suffix length=N=10

b. The prefix length for the block is calculated as

$$n = 32 - \log_2 1024 = 22.$$

c. The beginning address is chosen as 18.14.12.0 (which is divisible by $1024_{(10)} = 0.0.4.0_{(256)}$

The granted block is 18.14.12.0/22. The first address is 18.14.12.0/22 and the last address is 18.14.15.255/22.

# Classful address - Special case of classless address

**Table 5.1** *Prefix length for classful addressing*

| *Class* | *Prefix length* | *Class* | *Prefix length* |
|---|---|---|---|
| A | /8 | D | /4 |
| B | /16 | E | /4 |
| C | /24 | | |

**Classful address scheme is a special case of classless address scheme where prefix length is fixed as 8 for Class A, 16 for Class B , 24 for Class C. 4 for Class D and 4 for Class E**

# Example 5.31

**Assume an organization has given a class A block as 73.0.0.0 in the past. If the block is not revoked by the authority, the classless architecture assumes that the organization has a block 73.0.0.0/8 in classless addressing.**

Note

***The restrictions applied in allocating addresses for a subnetwork are parallel to the ones used to allocate addresses for a network.***

# *Designing Subnets*

The sub networks in a network should be carefully designed to enable the routing of packets. We assume the total number of addresses granted to the organization is ***N***, the prefix length is ***n***, the assigned number of addresses to each sub network is ***N*sub,** the prefix length for each sub network is ***n*sub,** and the total number of sub networks is ***s***. Then, the following steps need to be carefully followed to guarantee the proper operation of the sub networks.

1. The **number of addresses in each sub network** should be a **power of 2**.
2. The **prefix length for each sub network** should be found using the following formula:

$$n_{sub} = n + \log_2 (N/N_{sub})$$

3. The s**tarting address in each subnetwork** should be **divisible by the number of addresses in that subnetwork**. This can be achieved if we **first assign addresses to larger networks.**

$$n_{sub} = n + \log_2 (N/N_{sub})$$

**N**-Total number of addresses in the granted block

$\mathbf{N_{sub}}$ - Number of addresses required in the subnet (has to be power of 2).

**n**- Prefix length of granted address block

$\mathbf{\log_2(N/N_{sub})}$ - Gives Extra bits (apart from n bits) needed to create subnet.

# Example 5.32

An organization is granted the block 130.34.12.64/26. The organization needs four subnetworks, each with an equal number of hosts. Design the subnetworks and find the information about each network.

**Solution**

The number of addresses for the whole network can be found as $N = 2^{(32 - 26)} = 64$. The first address in the network is 130.34.12.64/26 and the last address is 130.34.12.127/26. We now design the sub networks:

**1.** We grant 16 addresses for each sub network to meet the first requirement (16 is a power of 2).

**2.** The subnetwork mask for each subnetwork is:

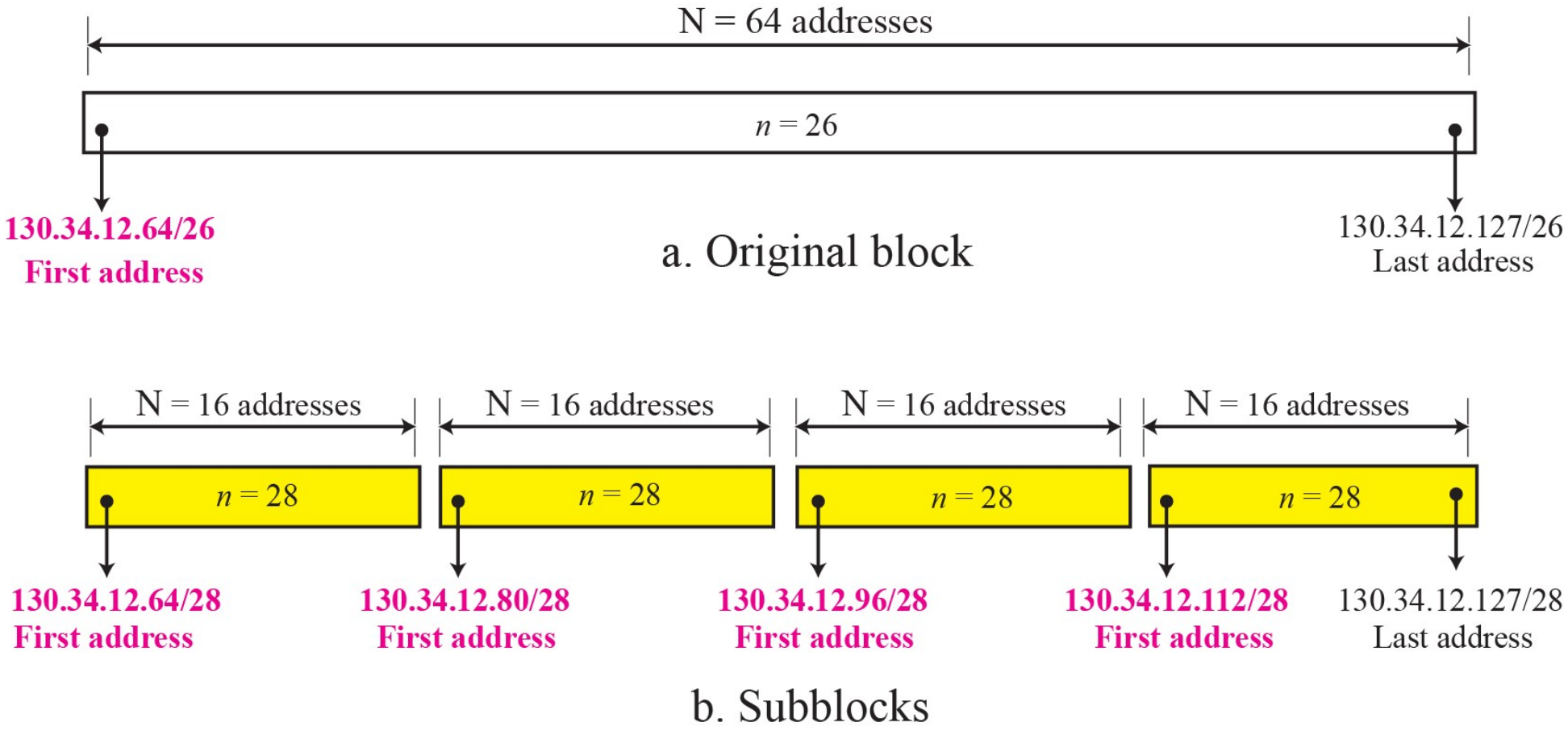
$$n_1 = n_2 = n_3 = n_4 = n + \log_2 (N/N_i) = 26 + \log_2 4 = 28$$

# Example 5.32 *Continued*

**3. We grant 16 addresses to each subnet starting from the first available address. Figure 5.30 shows the sub-block for each subnet.**

**Note that the starting address in each subnetwork is divisible by the number of addresses in that subnetwork.**

## Figure 5.30 *Solution to Example 5.32*



# Example 5.33

An organization is granted a block of addresses with the beginning address **14.24.74.0/24**. The organization needs to have 3 subblocks of addresses to use in its three subnets as shown below:

- One subblock of **120** addresses.
- One subblock of **60** addresses.
- One subblock of **10** addresses.

***Solution***

There are $\mathbf{2^{32-24} = 256}$ addresses in this block. The first address is **14.24.74.0/24**; the last address is **14.24.74.255/24**.

a. The number of addresses in the first subblock is not a power of 2. We allocate 128(=$2^7$) addresses. The prefix length is 25 (see Notes section) . **14.24.74.0** divisible by 128

b. The first address is 14.24.74.0/25; the last address is 14.24.74.127/25 (see Notes section).

# Example 5.33 Continued

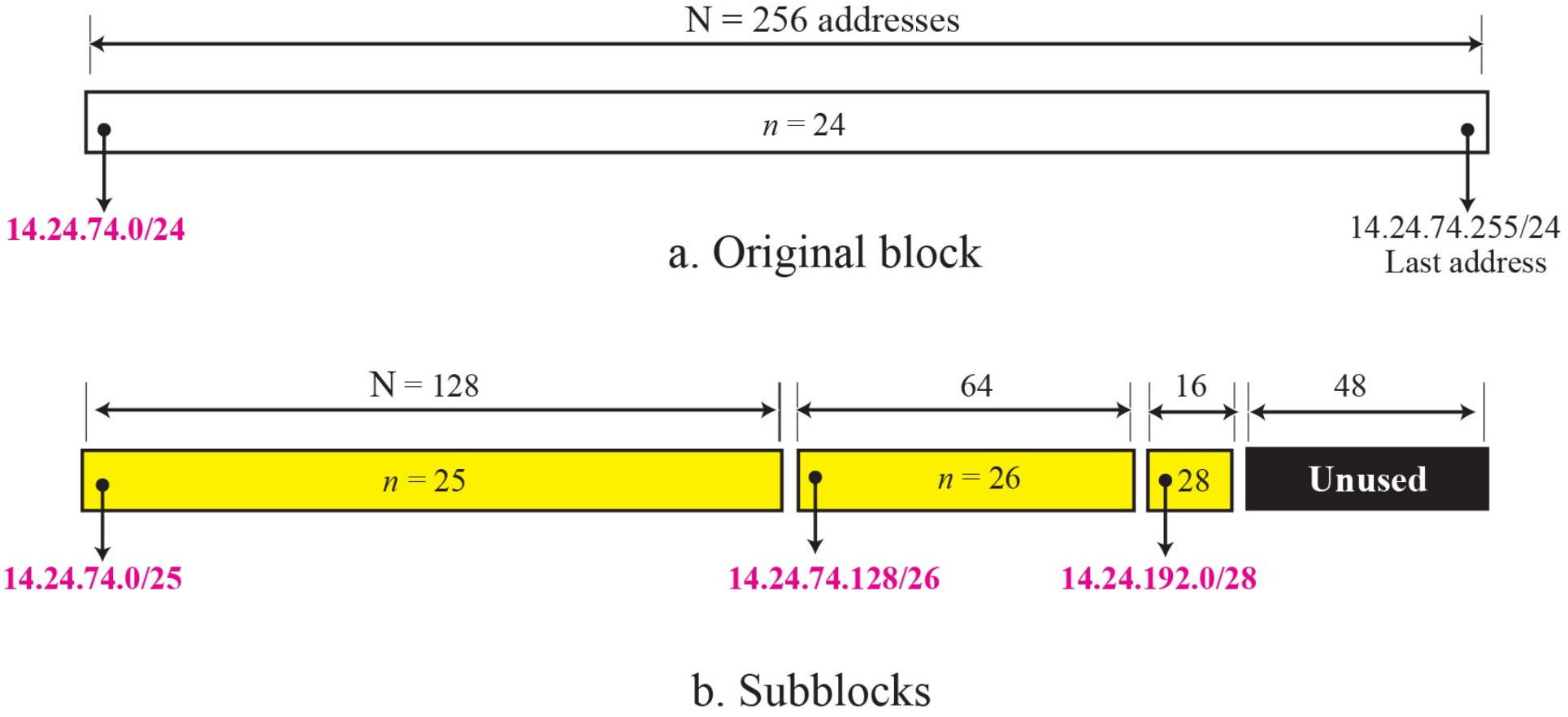
b. The number of addresses in the second subblock is not a power of 2 either. We allocate 64 addresses. The subnet length is 26. 1st address 14.24.74.128 divisible by 64. The first address in this block is 14.24.74.128/26; the last address is 14.24.74.191/26.

c. The number of addresses in the third subblock is not a power of 2 either. We allocate 16 addresses. The subnet mask is 28. The first address in this block is 14.24.74.192/28 ( Divisible by 16); the last address is 14.24.74.207/28.

d. If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.209. The last address is 14.24.74.255.

e. Figure 5.31 shows the configuration of blocks. We have shown the first address in each block.

## Figure 5.31 *Solution to Example 5.33*




**Solve the same problem by allocating addresses in the order 64, 16 & 120 addresses.**

# Example 5.34

Assume a company has three offices: Central, East, and West. The Central office is connected to the East and West offices via private WAN lines.

The company is granted a block of 64 addresses with the beginning address 70.12.100.128/26, Last address 70.12.100.191/26 .

The management has decided to allocate 32(power of 2) addresses for the Central office and divides the rest of addresses equally between the two other offices.

**1.** The number of addresses are assigned as follows:

$$\text{Central office } N_c = 32 \qquad \text{East office } N_e = 16 \qquad \text{West office } N_w = 16$$

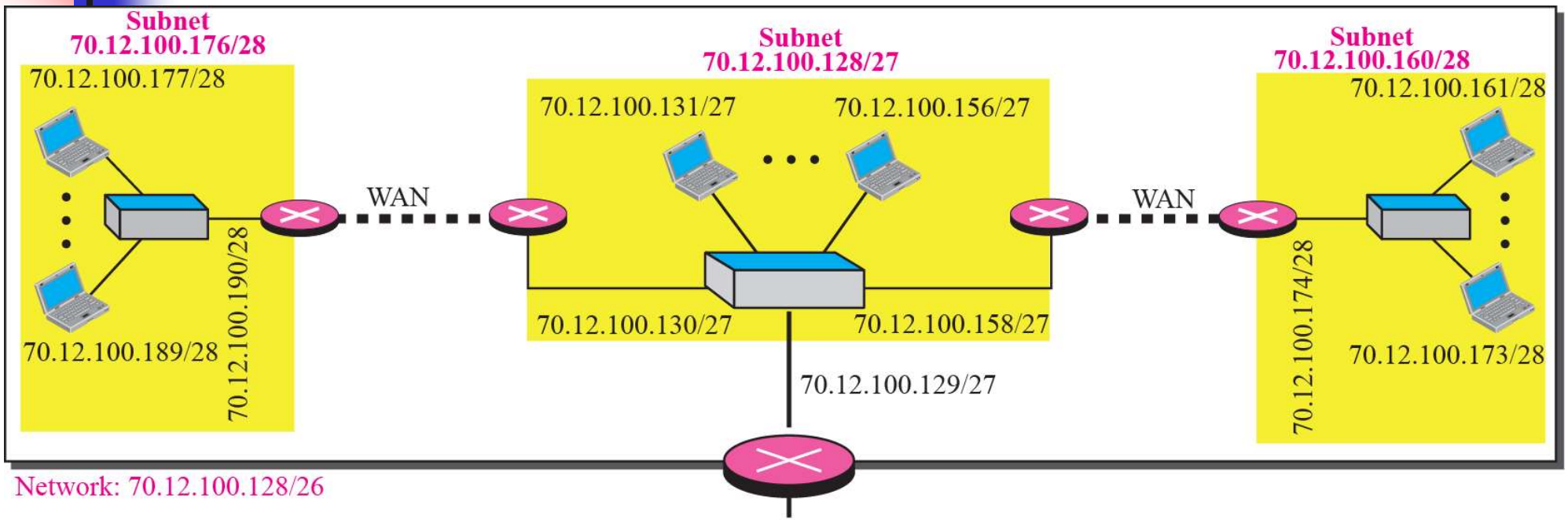
**2. We can find the prefix length for each subnetwork:**

$$n_c = n + \log_2(64/32) = 27 \qquad n_e = n + \log_2(64/16) = 28 \qquad n_w = n + \log_2(64/16) = 28$$

# Example 5.34 Continued

3. Figure 5.32 shows the configuration designed by the management. The Central office uses addresses 70.12.100.128/27 ( Divisible by 32 ) to 70.12.100.159/27. The company has used three of these addresses for the routers and has reserved the last address in the sunblock. The East office uses the addresses 70.12.100.160/28 ( Divisible by 16) to 70.12.100.175/28. One of these addresses is used for the router and the company has reserved the last address in the sub block. The West office uses the addresses 70.12.100.160/28 to 70.12.100.175/28. One of these addresses is used for the router and the company has reserved the last address in the sub block. The company uses no address for the point-to-point connections in WANs.

## Figure 5.32 *Example 5.34*




All addresses from 70.12.100.128 to 70.12.100.191
are delivered to this network

# Example 5.35

An ISP is granted a block of addresses starting with 190.100.0.0/16 (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:

- The first group has 64 customers; each needs approximately 256 addresses.
- The second group has 128 customers; each needs approximately 128 addresses.
- The third group has 128 customers; each needs approximately 64 addresses.

We design the subblocks and find out how many addresses are still available after these allocations.

# Example 5.35 Continued

***Solution***

**Let us solve the problem in two steps. In the first step, we allocate a subblock of addresses to each group. The total number of addresses allocated to each group and the prefix length for each subblock can found as**

$$\text{Group 1: } 64 \times 256 = 16{,}384 \qquad n_1 = 16 + \log_2 (65536/16384) = 18$$

$$\text{Group 2: } 128 \times 128 = 16{,}384 \qquad n_2 = 16 + \log_2 (65536/16384) = 18$$

$$\text{Group 3: } 128 \times 64 = 8192 \qquad n_3 = 16 + \log_2 (65536/8192) = 19$$

**Figure 5.33 shows the design for the first hierarchical level. Figure 5.34 shows the second level of the hierarchy. Note that we have used the first address for each customer as the subnet address and have reserved the last address as a special address.**

**Group 1**

**1st Address** **190.100.0.0/16**

**n=18**

**Mask** **255.255.192.0**

**Complement of Mask** **0. 0. 63.255**

**Last address** **190.100.0.0**

**OR** **0.0.63.255**

==========

**190.100. 63.255**

**Group 2**

Next address is **190.100.64.0**, it can be 1st address of Group 2, if and is divisible by number of addresses - 16384

**n=18 ,**

**Mask** **255.255.192.0**

**Complement of Mask** **0. 0. 63.255**

**Last address** **190.100.64.0**

**OR** **0.0.63.255**

==========

**190.100.127.255**

**Group 3**

Next address is **190.100.128.0**, it can be 1st address of Group 3, if and is divisible by number of addresses - 8192

**n =19,**

**Mask** **255.255.224.0**

**Complement of Mask** **0. 0. 31.255**

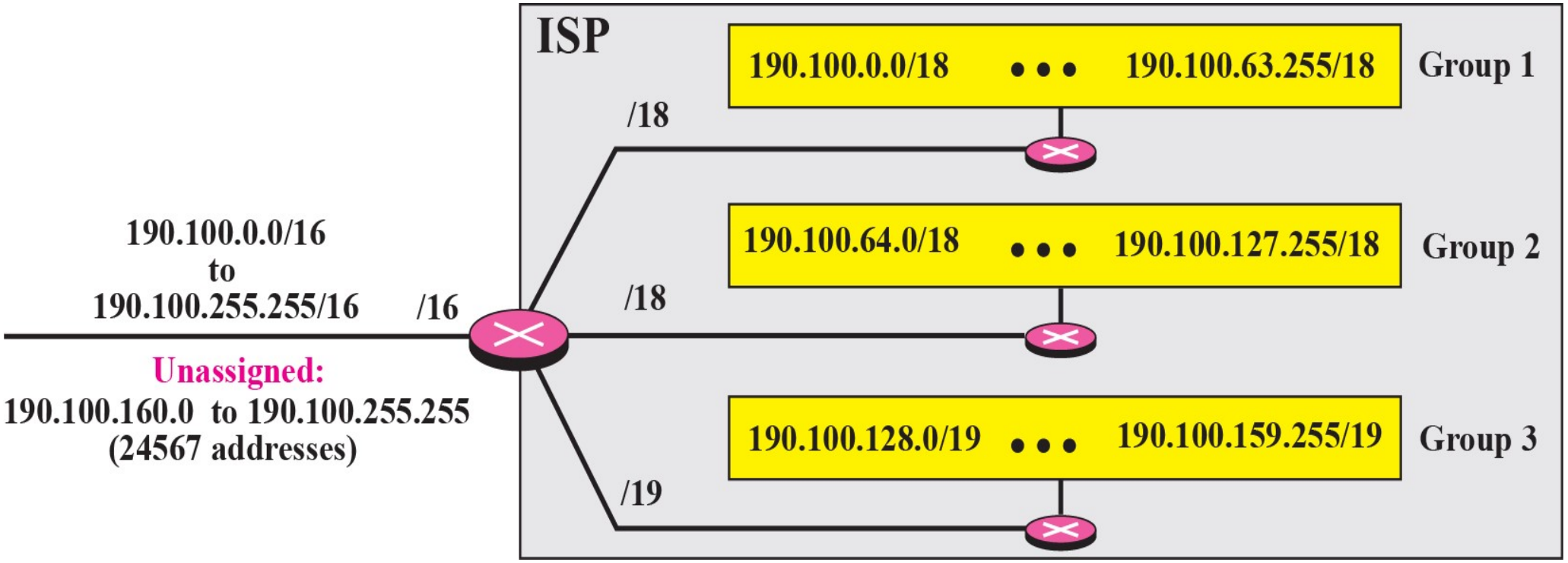
**Last address** **190.100.128.0**

**OR** **0.0.31.255**

==========

**190.100.159.255**

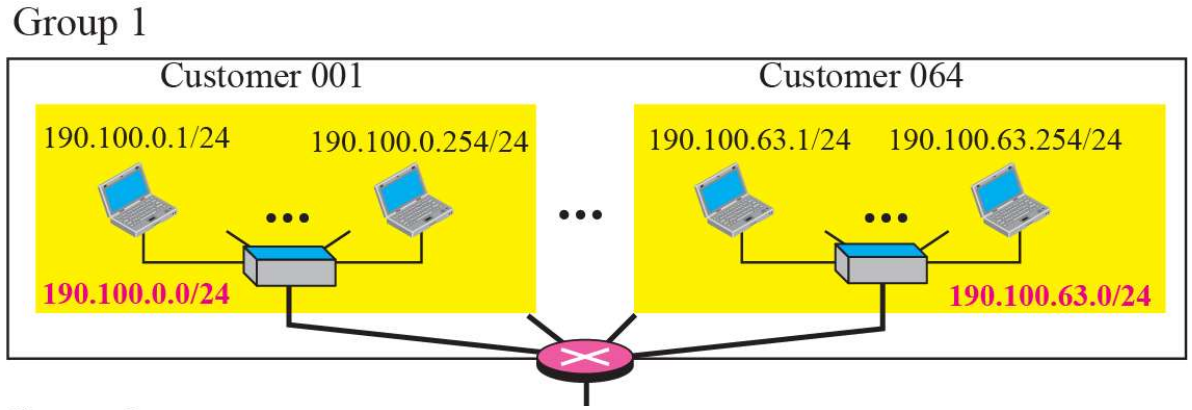
## Figure 5.33 *Solution to Example 5.35: first step*



## Figure 5.34 *Solution to Example 5.35: second step*

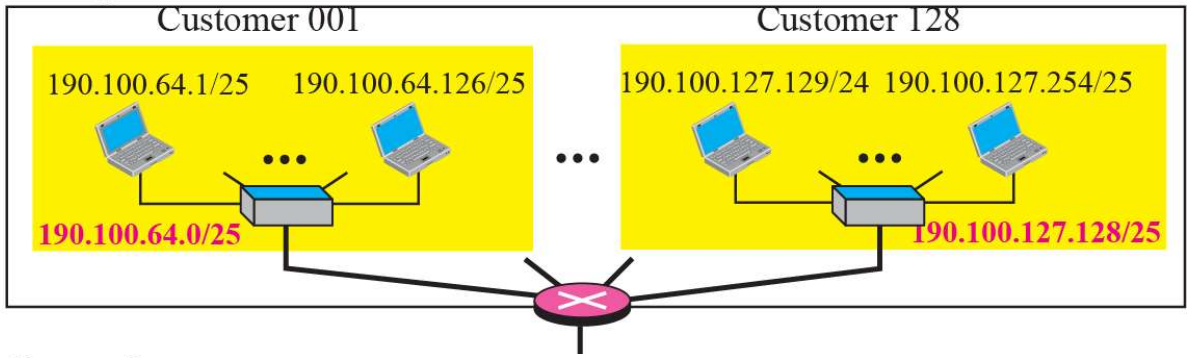
Group: $n = 18$

Subnet: $n = 18 + \log_2 (16385/256) = 24$



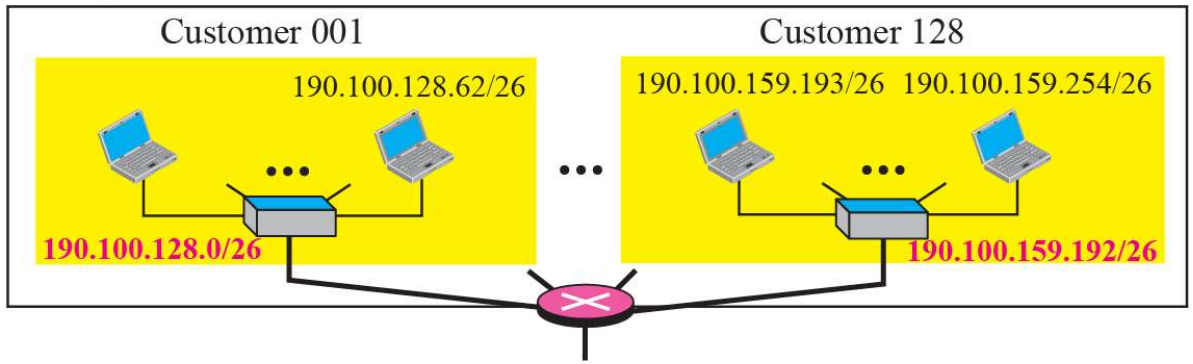

Group: $n = 18$

Subnet: $n = 18 + \log_2 (16385/128) = 25$




Group: $n = 19$

Subnet: $n = 19 + \log_2 (8192/64) = 26$



# 5-4 SPECIAL ADDRESSES

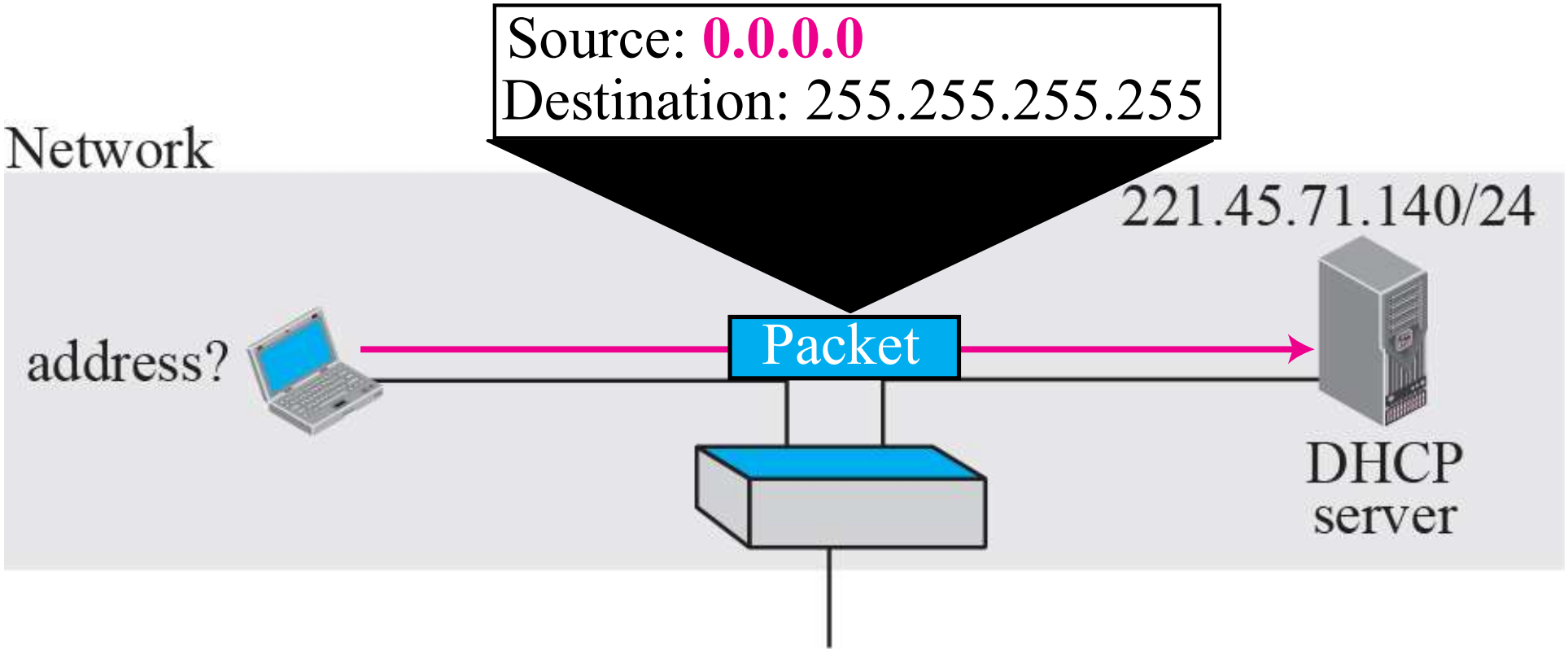
**In classful addressing some addresses were reserved for special purposes. The classless addressing scheme inherits some of these special addresses from classful addressing.**

# *Topics Discussed in the Section*

- ✓ **Special Blocks**
- ✓ **Special Addresses in each Block**

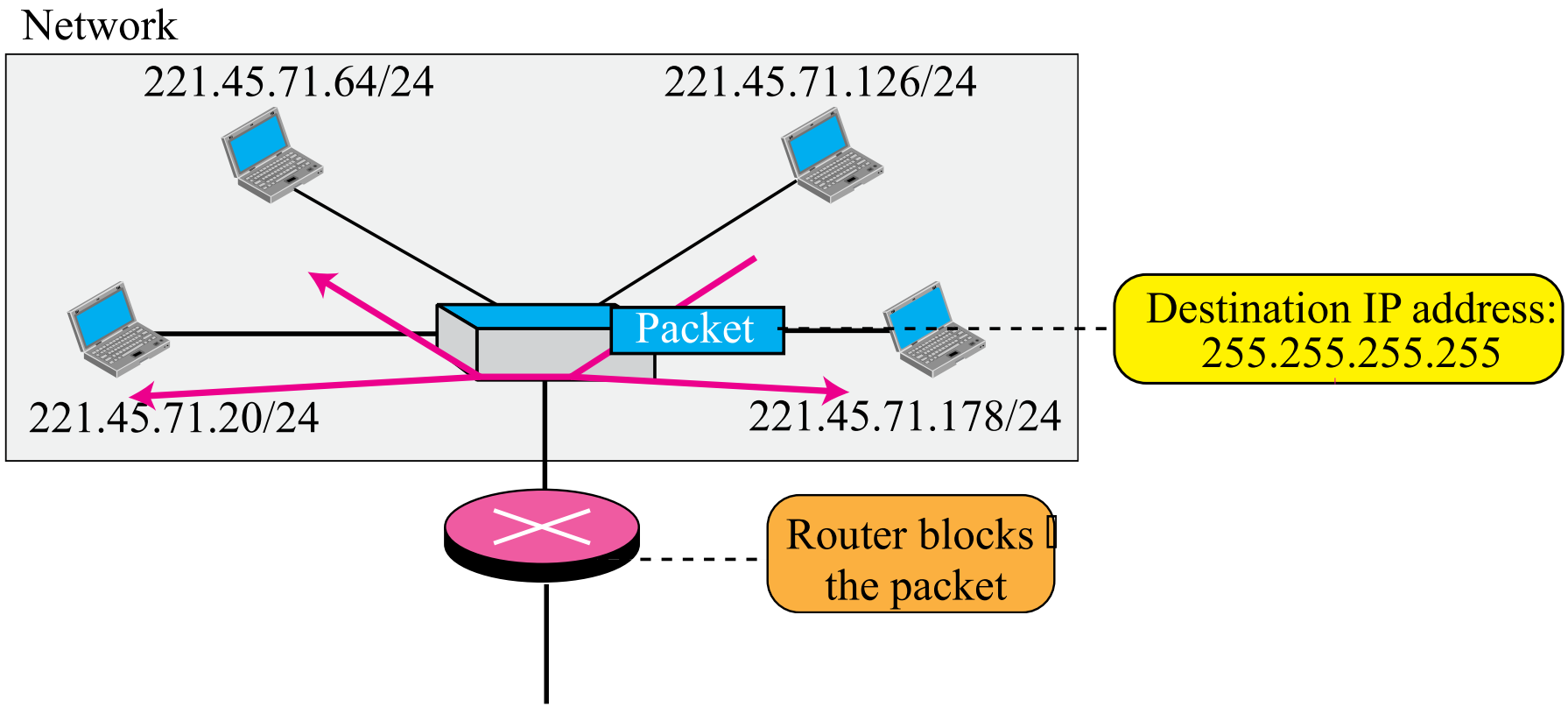
## Figure 5.35 *Example of using the all-zero address*

**The block 0.0.0.0/32, which contains only one single address**



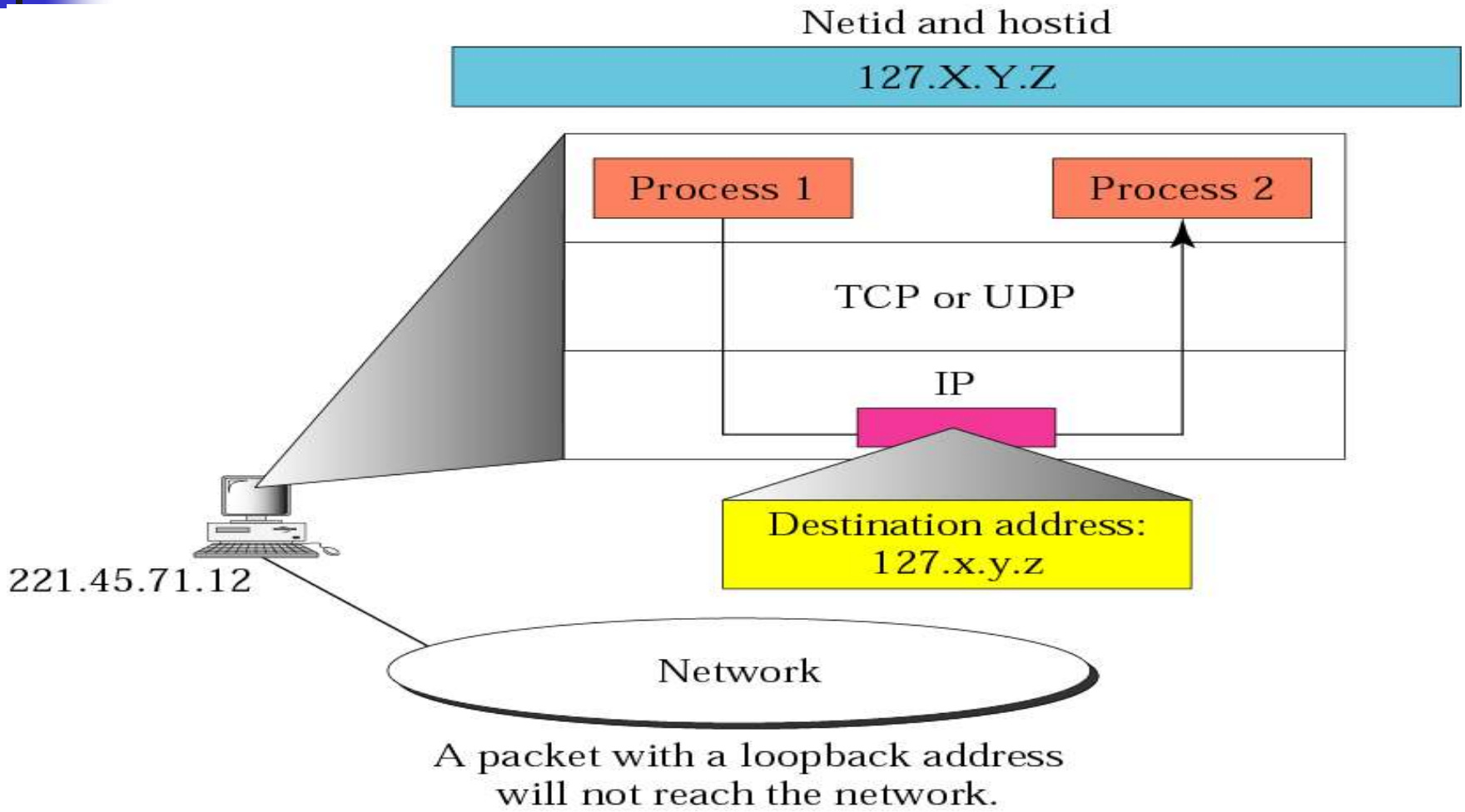

**Normally used by a host A as the Source IP during bootstrap time when it does not know its IPv4 address**

## Figure 5.36 *Example of limited broadcast address*



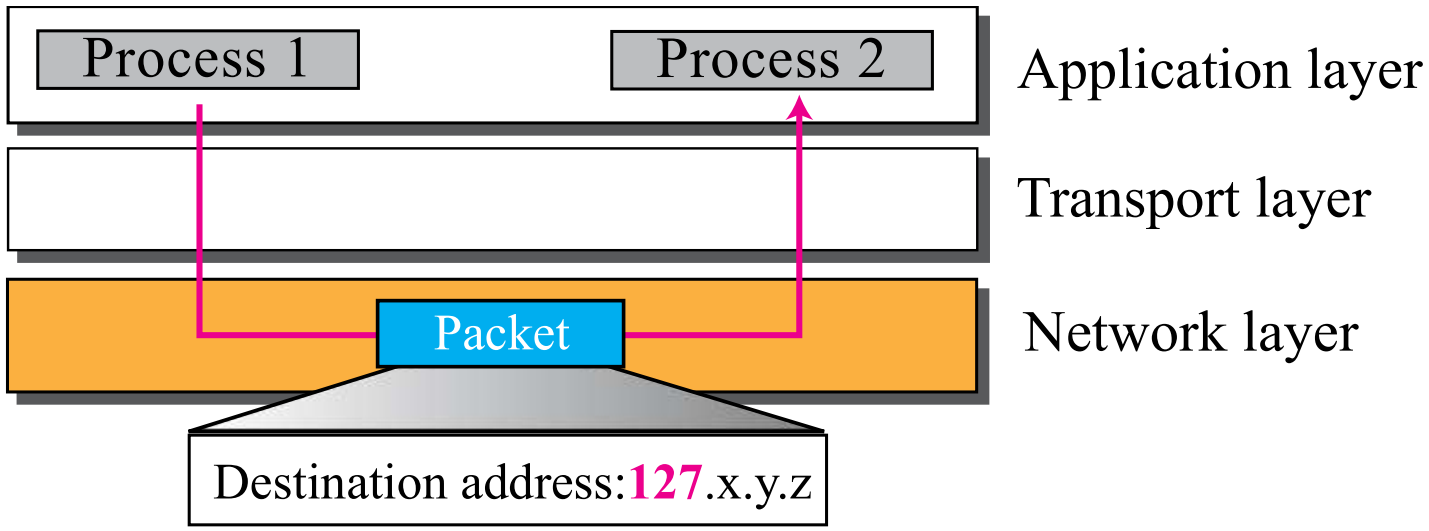

A host that wants to send a message to every other host can use this address as a destination address in an IPv4 packet. However, a router will block a packet bearing this as destination address..

## *Example of loopback address*




As an example, the loopback address can be used by a *client process* (a running application program) to send a message to a server process on the same machine.

## Figure 5.37 *Example of loopback address*




**loopback address,** which is an address used to test the software on a machine

**Table 5.2** *Addresses for private networks*

| Block | Number of addresses | Block | Number of addresses |
|---|---|---|---|
| 10.0.0.0/8 | 16,777,216 | 192.168.0.0/16 | 65,536 |
| 172.16.0.0/12 | 1,047,584 | | |

A number of blocks are assigned for private use. They are not recognized globally. These blocks are depicted in Table 5.2.

These addresses are used either in isolation or in connection with **Network Address Translation (NAT)** techniques

# Multicast Addresses

The block **224.0.0.0/4** is reserved for multicast communication

# Special Addresses in Each block

It is not mandatory, but it is recommended, that some addresses in a block be used for special addresses.

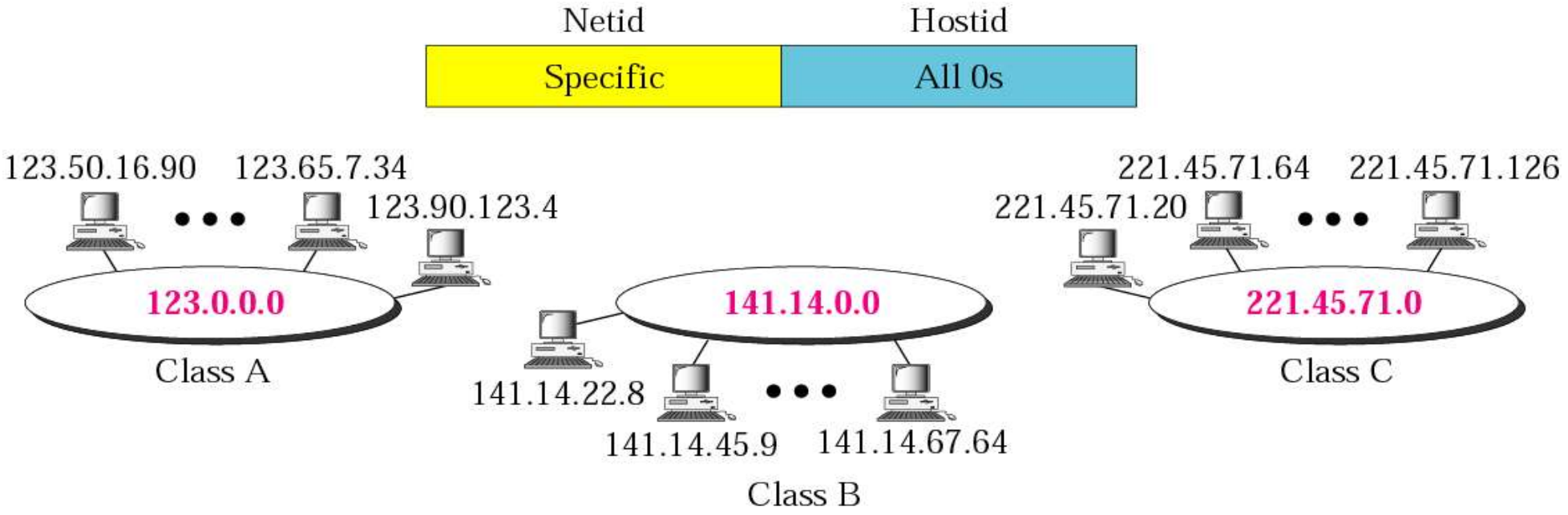
These addresses are not assigned to any host.

- Network Address
- Direct Broadcast

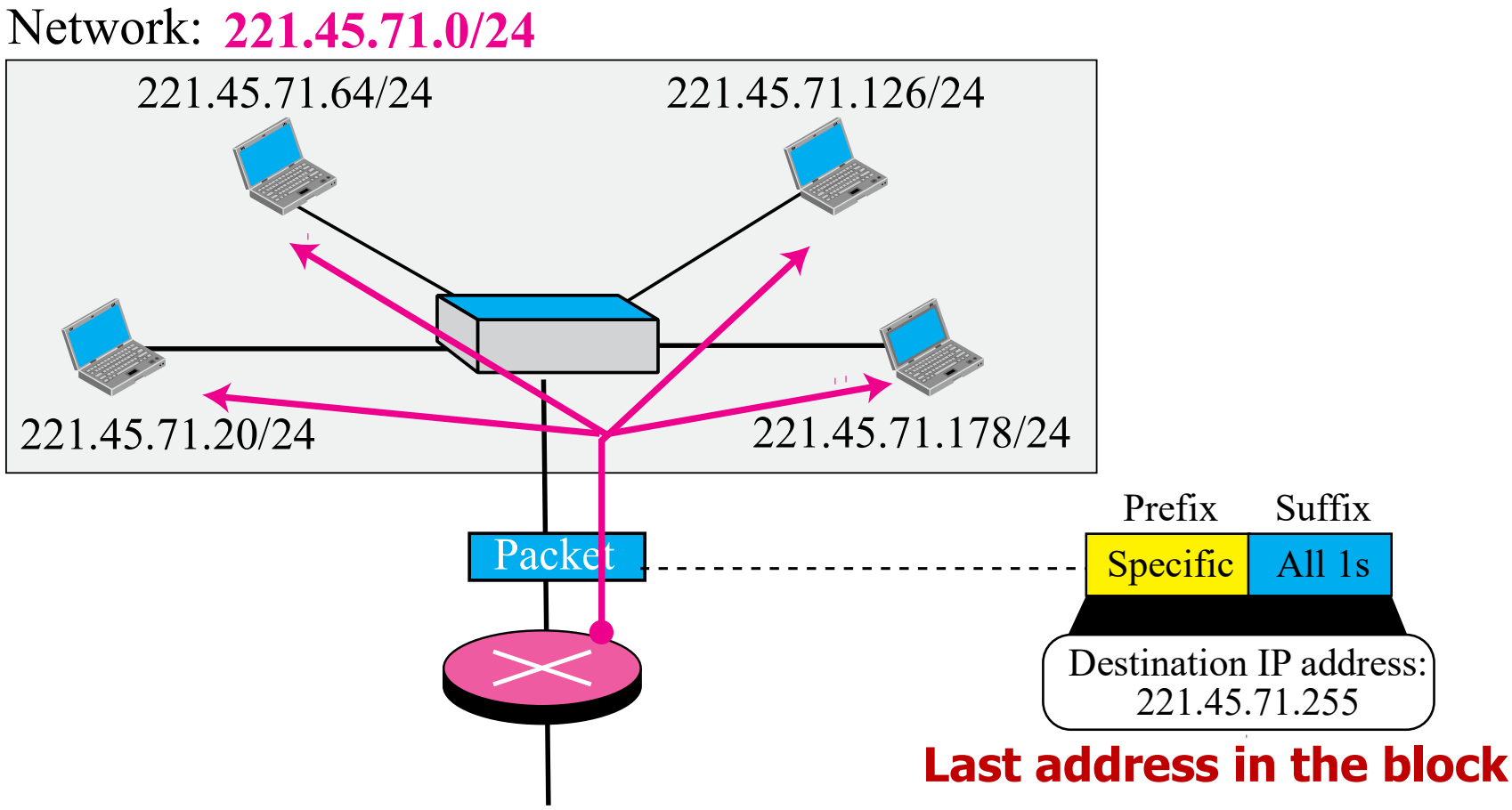
## Network Address

The first address (with the suffix set all to 0s) in a block defines the network address. It actually defines the network itself (cabling) and not any host in the network. first address in a subnetwork is called the subnetwork address and plays the same role.

***Network address***



## Figure 5.38 *Example of a directed broadcast address*




**Directed Broadcast address is the local subnet broadcast address**

**This address is usually used by a router to send a packet to all hosts in a specific network**

# Different types of Addresses

- Communication on the Internet can be achieved using unicast, multicast or broad cast address.
- **Unicast communication** is one to one: A packet is sent from one individual source to one individual destination.
- **Multicast** is one to many: A packet is sent from an individual source to a group of destinations.
  - Class D address is a multicast address.
  - The entire address defines a group ID.
  - A system in the internet can have more the one multicast address.
  - Note: **Class D** address can be used as a Destination Address only.
- **Broadcast** communication is one to all.
- The internet **allows broadcasting only in the local level** and not in the global level. This means that a system can not send a message to all the hosts and the routers in the internet.
- Two types of broad cast in local level are: The limited broadcast address and direct broadcast address.

# NAT
# TCP/IP Protocol Suit
# Forouzan

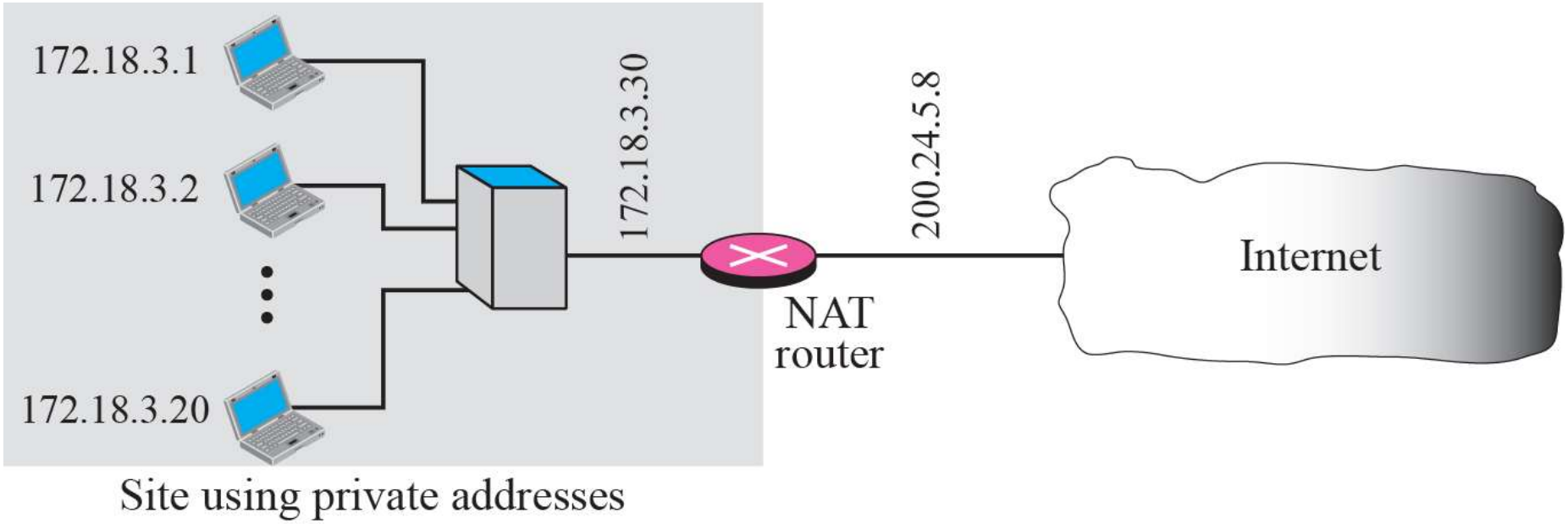
# 5-5 NAT

**The distribution of addresses through ISPs has created a new problem. If the business grows or the household needs a larger range, the ISP may not be able to grant the demand because the addresses before and after the range may have already been allocated to other networks. In most situations, however, only a portion of computers in a small network need access to the Internet simultaneously. A technology that can help in this cases is *network address translation (NAT)*.**

# *Topics Discussed in the Section*

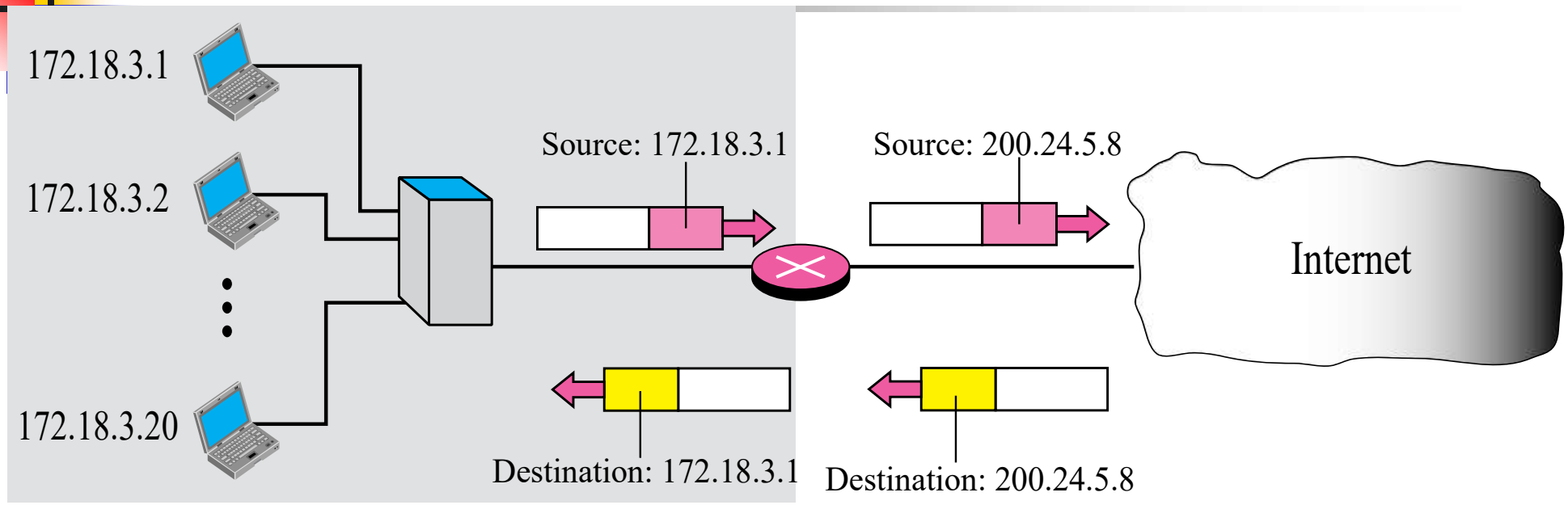
- ✓ **Address Translation**
- ✓ **Translation Table**

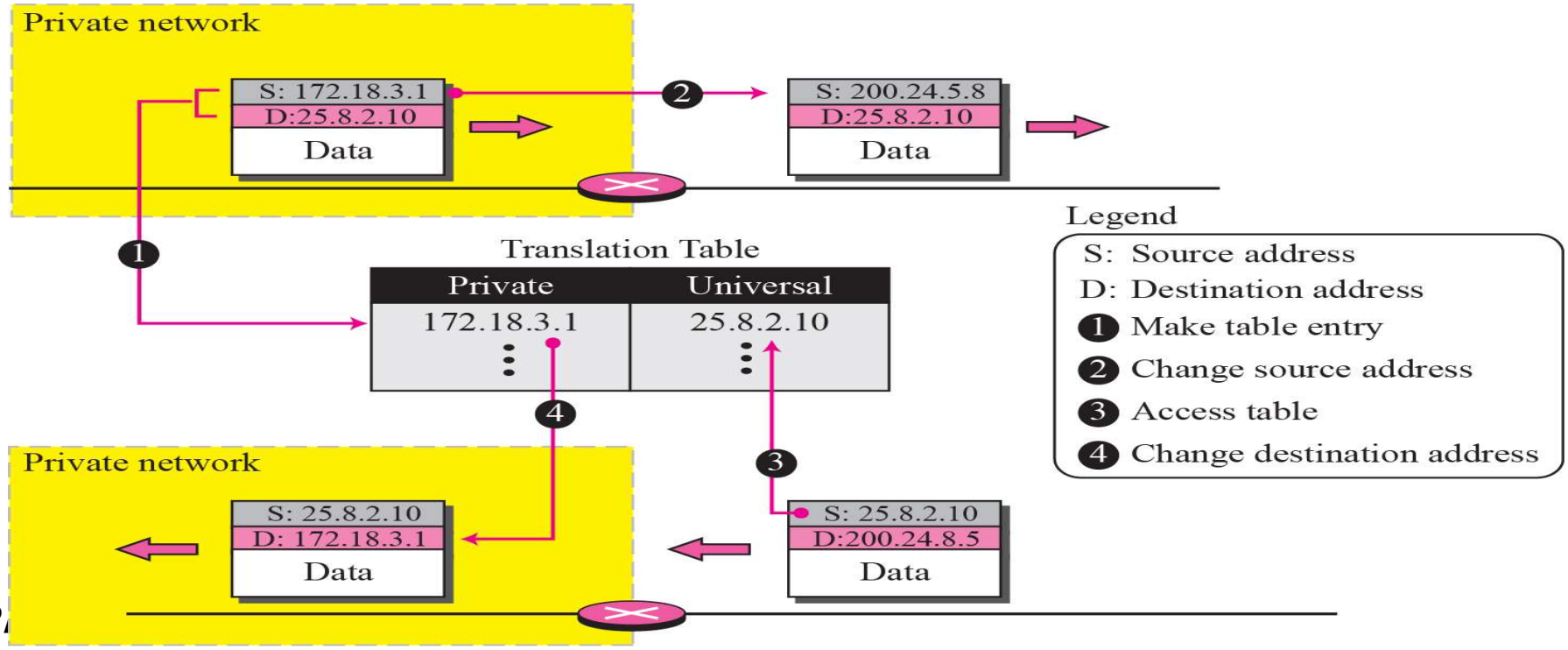
## Figure 5.39 *NAT*

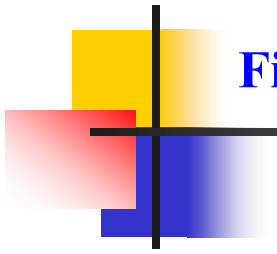



**The rest of the Internet sees only the NAT router with the address 200.24.5.8.**

# Figure 5.40 *Address resolution*



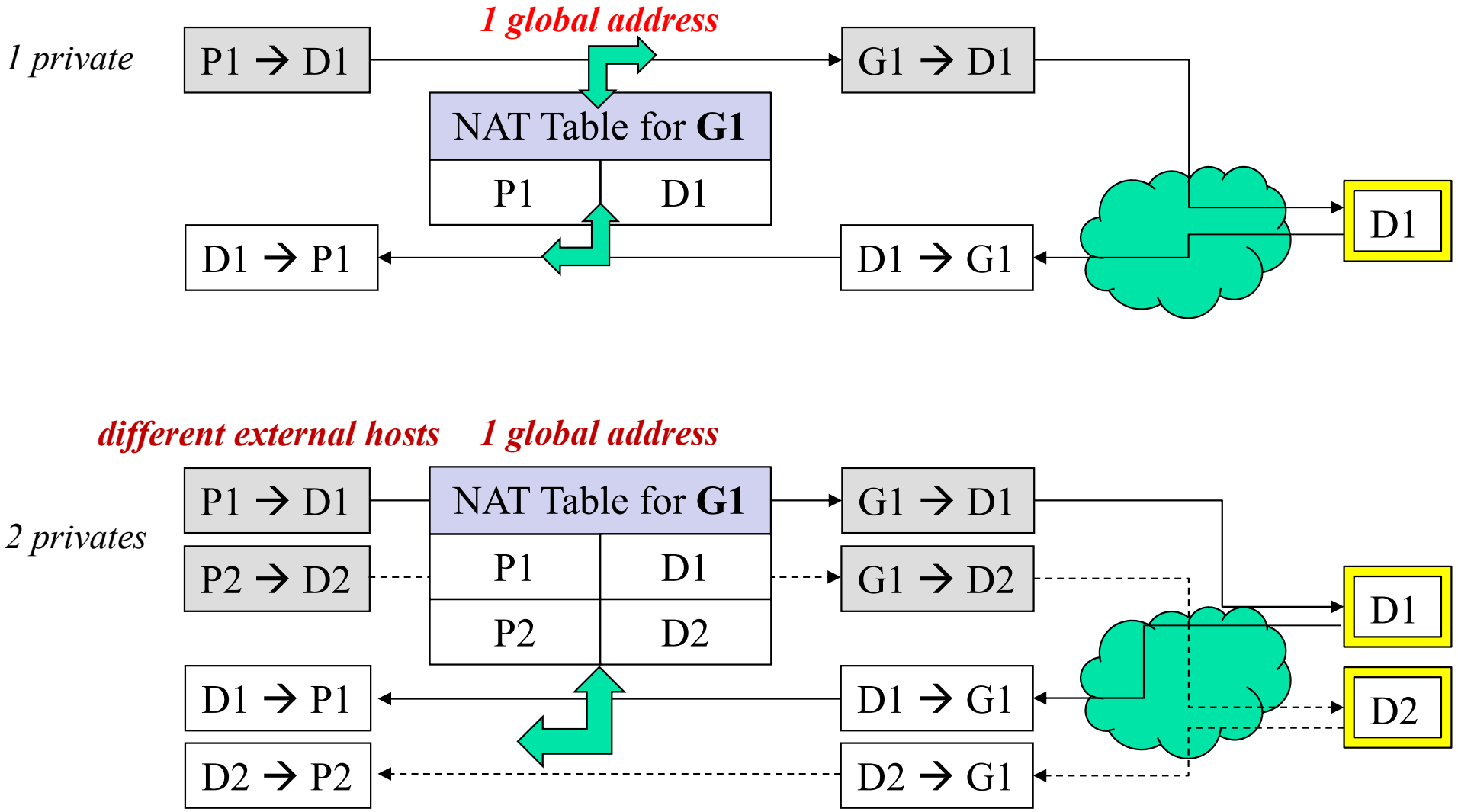


## Figure 5.41 *Translation*

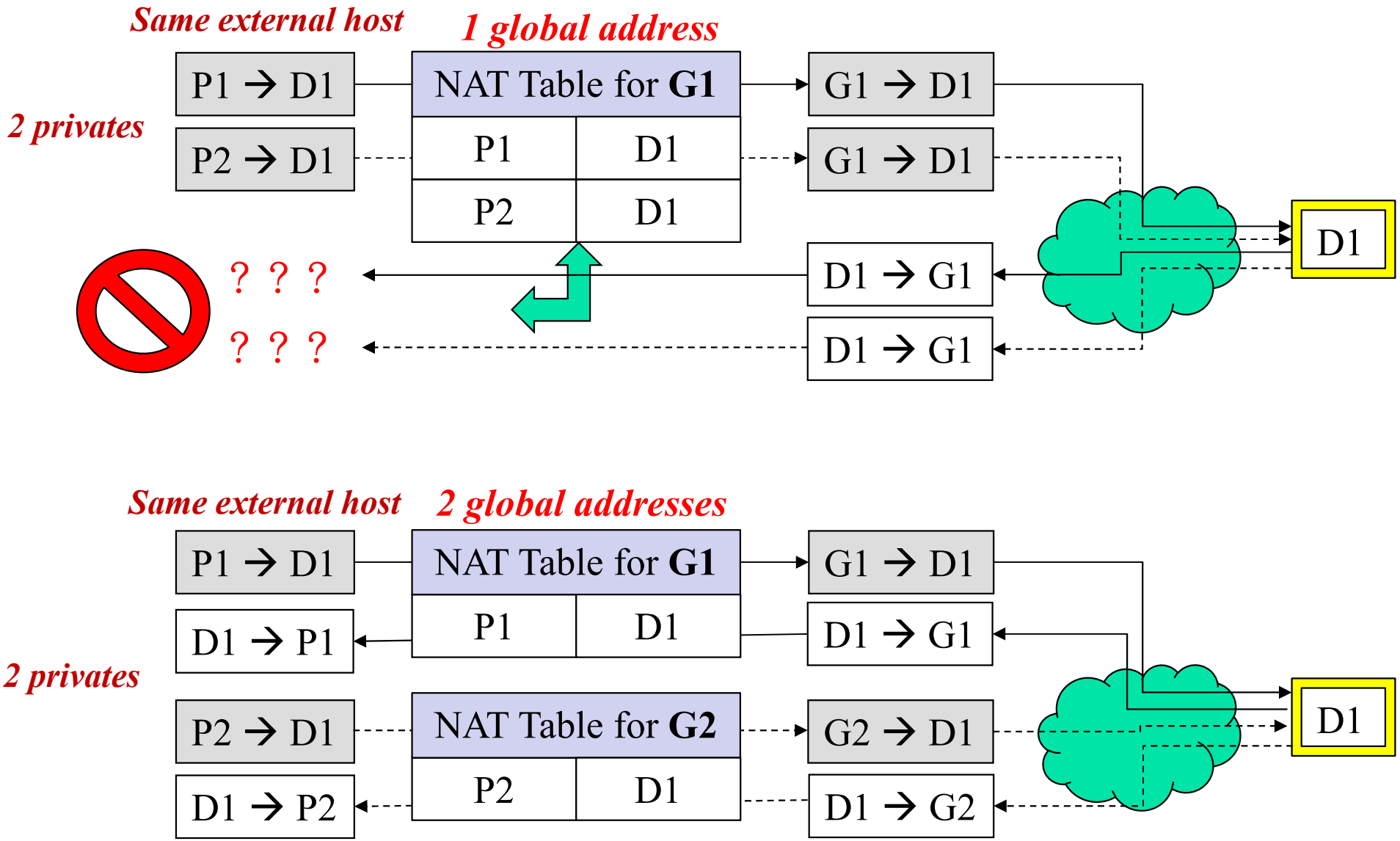
- **Communication must always be initiated by the private network.**
- **A Private network cannot run a server program for clients outside of its network if it is using NAT technology.**

# NAT Table with only IP addresses (1)

**P1- Source address(private Address) , D1-Destination address, G1-Global IP**



# NAT Table with only IP addresses (2)



# NAT Table with only IP addresses (3)

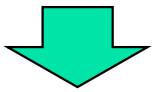
- If using only one global address
  - Only one private-network host to access the same external host
- If using a pool of global addresses (e.g. 4 addr)
  - No more than 4 connections can be made to the same destination
  - Two private-network hosts cannot access the same external server program at the same time (by using the same global address)

# NAT Table with IP address & Port # (4)

- NAT Table in the textbook
  - This may not be the best way to understand NAT

***Server***

| *Private Address* | *Private Port* | *External Address* | *External Port* | *Transport Protocol* |
|---|---|---|---|---|
| 172.18.3.1 | 1400 | 25.8.3.2 | 80 | TCP |
| 172.18.3.2 | 1401 | 25.8.3.2 | 80 | TCP |
| … | … | … | … | … |



***Must be unique***

***Otherwise reassign a new private port and cache the mapping (the new port --> the old port)***

**END of CHAPTER**