

FORE
SYSTEMS





IP Version 4 Addressing

V2.1: Geoff Bennett



Contents

The Structure of an IP Address

IP Address Classes

Subnetting an IP Address

Applying Addresses

Reserved and Special Addresses

This tutorial is divided into five sections. The first three deal with the three layers of structure of an IP address.

These are the general address format, the “natural” address class and the subnet structure.

The last two sections deal with some consequences of IP usage in the real world.

Section 1

The Structure of an IP Address

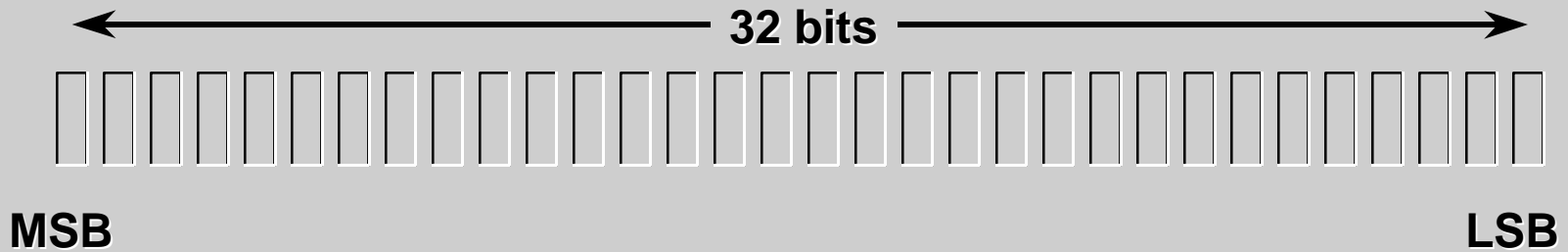
IP Address Classes

Subnetting an IP Address

Applying Addresses

Reserved and Special Addresses

In the first section I'll look at the basic structure of the IP address.





IPv4 addresses are always 32-bits long. Inside the computer, addresses will always end up as a collection of binary digits.

In addition, the other two layers of structure (“natural” address class and subnet structure) in IP addresses depend on binary, so we’ll look at the address in this way to start.

The Most Significant Bit (MSB) of the address is written on the left hand side of the page, with the Least Significant Bit (LSB) on the right.





 = binary 0
 = binary 1

In this explanation, I'll indicate a binary 0 by leaving the bit grey, and a binary 1 by coloring the bit blue.

IP addresses are never evaluated as a single, 32-bit number...

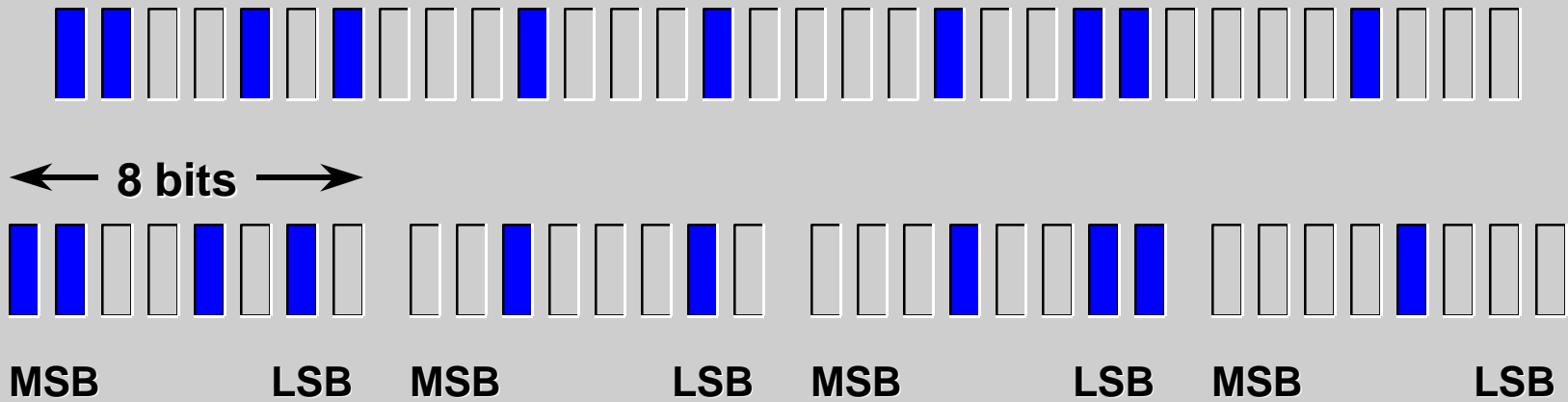


 = binary 0
 = binary 1

In this explanation I'll indicate a binary 0 by leaving the bit grey, and a binary 1 by coloring the bit blue.

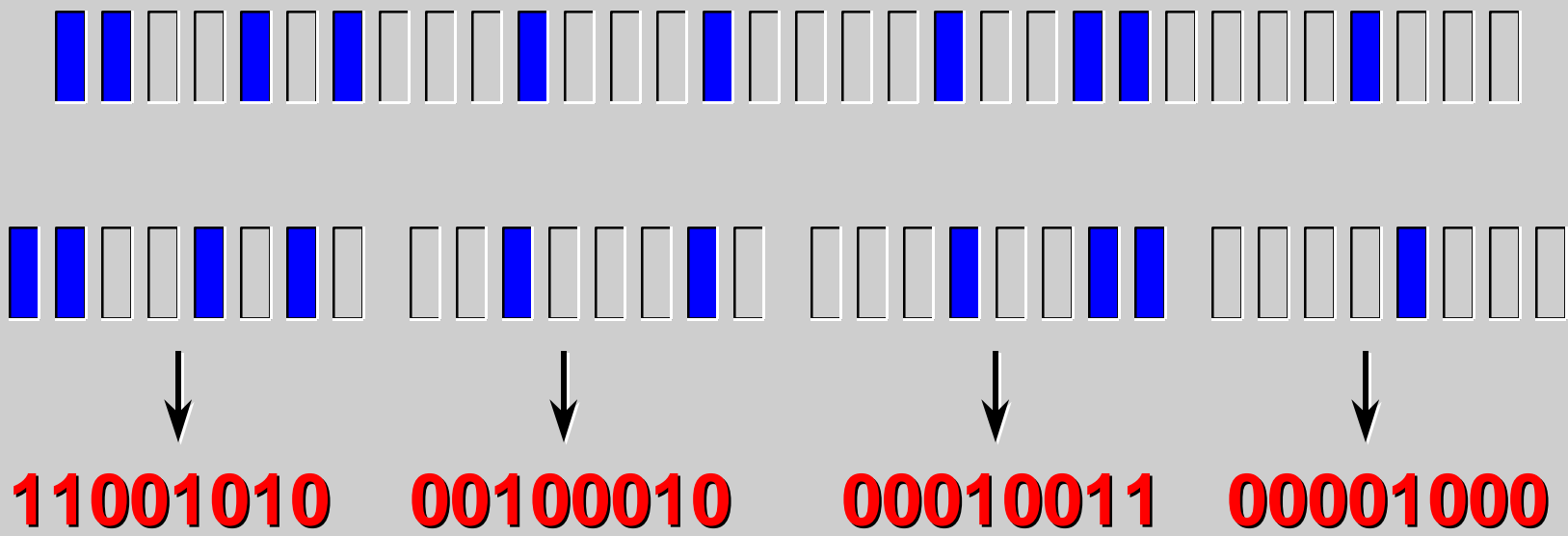
IP addresses are never evaluated as a single, 32-bit number...

...if they were, then this address would be 3391230728 in decimal.

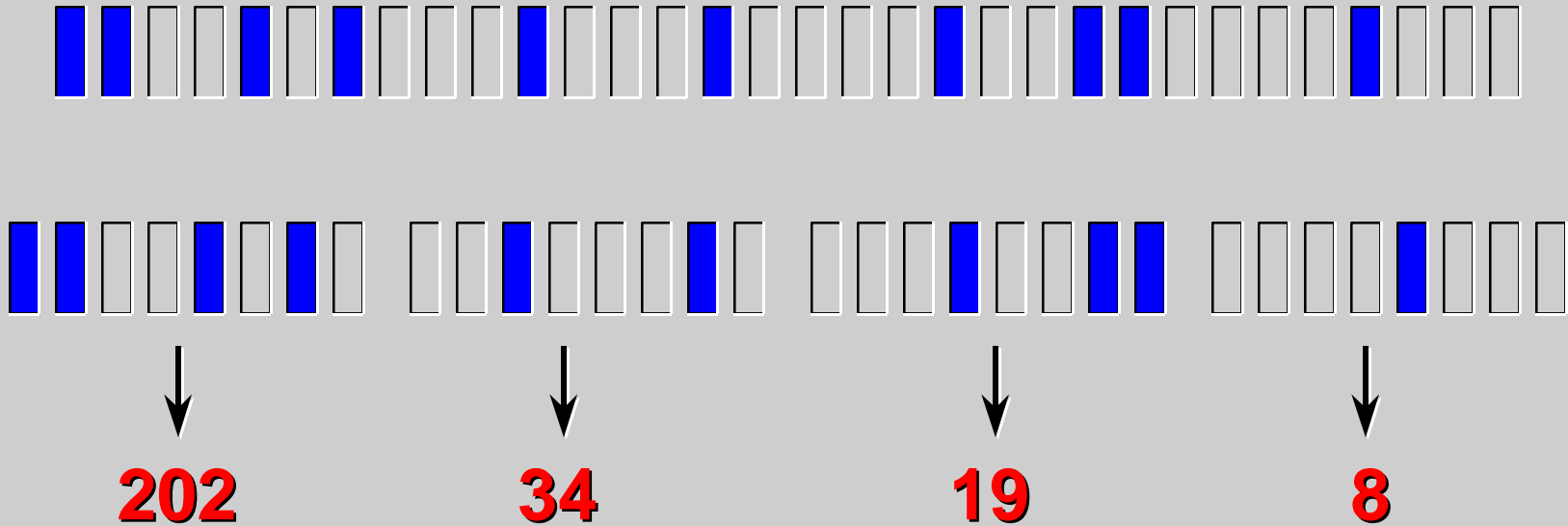


Instead, the first layer of structure in IP addresses is the fact that the 32 bits are divided into four, 8-bit blocks.

The MSB of each block is written on the left, and the LSB on the right.



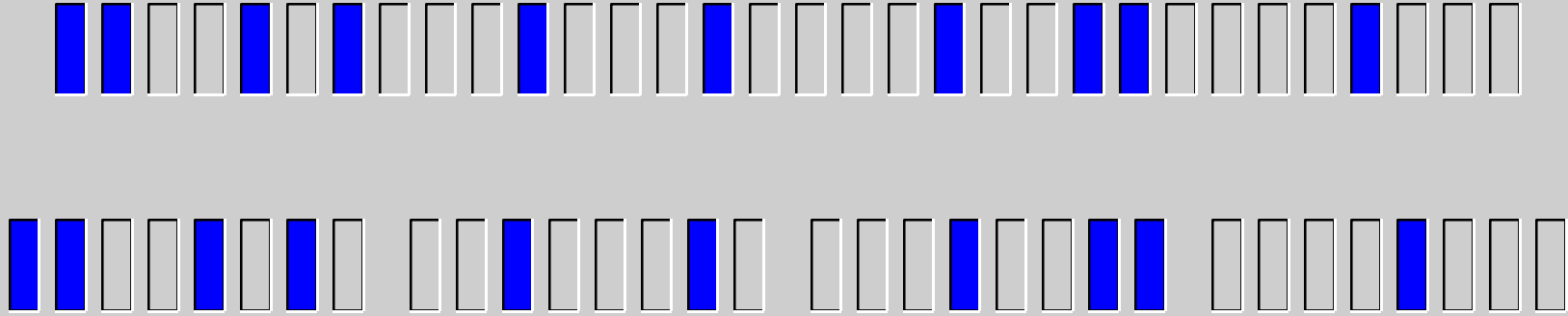
Here are the binary numbers...



...and this is what we get if we convert each byte to a decimal number.

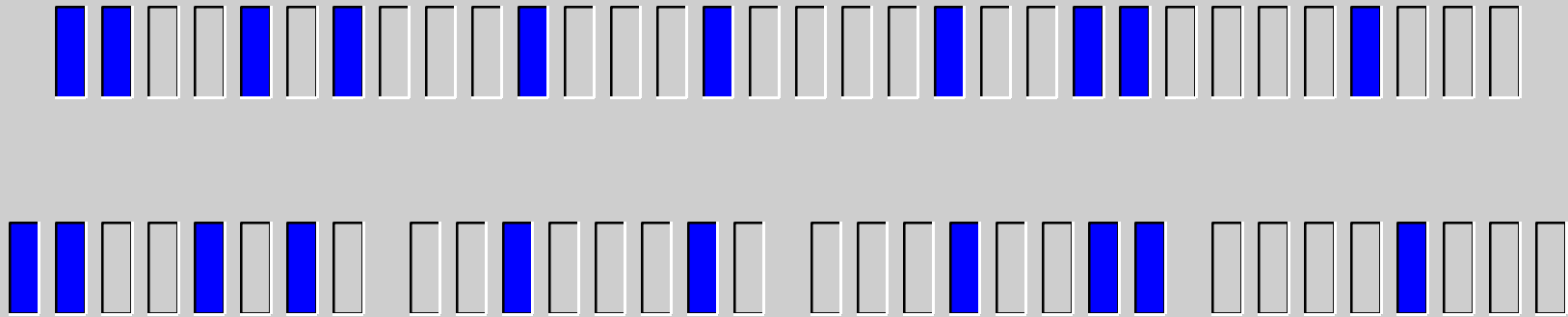
Notice that one of these numbers has three digits, two have two digits and one has one digit.

In other words...



20234198

...if we write the numbers down together, how do we know where the byte boundaries are?



202 . 34 . 19 . 8

To solve that problem we insert a dot (or period or decimal point) between the individual bytes.

We can now parse the IP address and translate back to binary inside a computer.



Decimal

Binary



See if you can work out the binary values for the decimal addresses I give you on the left.
On this first address, I'll only reveal one binary byte at a time.
Click to reveal the decimal number...



Decimal

Binary

192.32.20.4

. . .

. . .

. . .

. . .

. . .

. . .

Now translate the first byte, "192" into binary...
...click to reveal the answer...



Decimal

Binary

192.32.20.4

11000000



You're evaluating the decimal number 192. In terms of powers of 2, this is 128+64.
In binary, 128=10000000 and 64=01000000, so 192=11000000.
Click to reveal the next byte.



Decimal

Binary

192.32.20.4

11000000

00100000



32 can be represented with a single binary 1, 32=00100000.

Click to reveal the next byte.



Decimal

Binary

192.32.20.4

11000000

00100000

00010100



20 = 16 + 4.

Click to reveal the last byte.



Decimal

Binary

192.32.20.4

11000000

00100000

00010100

00000100



How did you do?

OK, for this next address I'll reveal one byte at a time again.

Click to see the decimal address...



Decimal

Binary

192.32.20.4

11000000

00100000

00010100

00000100

10.254.12.37



Now click to reveal the first binary byte.



Decimal

Binary

192.32.20.4

11000000



00100000



00010100



00000100

10.254.12.37

00001010



Click to reveal the second byte.



Decimal

Binary

192.32.20.4

11000000

00100000

00010100

00000100

10.254.12.37

00001010

11111110



If you got that one, then you're obviously catching on!
Click to reveal the next byte.



Decimal

Binary

192.32.20.4

11000000

00100000

00010100

00000100

10.254.12.37

00001010

11111110

00001100



...and click to reveal the last byte.



Decimal

Binary

192.32.20.4

11000000



00100000



00010100



00000100

10.254.12.37

00001010



11111110



00001100



00100101



...and that's it!

Before you get bored and click beyond the next examples, let me just highlight one point.



Decimal	Binary			
192.32.20.4	11000000	00100000	00010100	00000100
10.254.12.37	00001010	11111110	00001100	00100101

The zeroes that I've highlighted appear at the beginning of each byte, BEFORE the first "1". These are called *leading zeroes*.

In decimal numbers we don't normally write down leading zeroes, but we usually do in binary. That's because the *position* of binary digits in the byte is so important.

In IP addressing, we *know* that each number between the dots has eight binary digits, so if we see the binary number "1100", we can assume four leading zeroes and correctly evaluate the number as 12 decimal.



Decimal	Binary			
192.32.20.4	11000000	00100000	00010100	00000100
10.254.12.37	00001010	11111110	00001100	00100101
132.71.14.81	10000100	01000111	00001110	01010001
21.7.221.45	00010101	00000111	11011101	00101101
187.39.88.2	10111011	00100111	01011000	00000010
1.1.1.1	00000001	00000001	00000001	00000001

...and you thought I was going to make you go through all these examples byte-by-byte!
These are just examples of IP addresses. You can use the Windows Calculator to convert between decimal and binary if you would like more practice.
In theory we can put any number into an eight-bit field between 0 and 255.
In practice there are lots of reserved numbers and addresses, which I'll describe later in this tutorial.

The Structure of an IP Address

Section 2

IP Address Classes

Subnetting an IP Address

Applying Addresses

Reserved and Special Addresses

The next layer of structure in an IP address is the *Natural Class*.

202 . 34 . 19 . 8

Here is an IP address.

IP addresses contain a hierarchical structure similar to telephone numbers. Part of the telephone numbers we use everyday is an *Area Code*, and the rest of the number is the *Subscriber Number*.

This hierarchy can be used by telephone exchanges to route calls more efficiently.

202 . 34 . 19 . 8

Network ID

In IP terms, the area code is known as the *Network ID*. As the name suggests, this part of the address identifies the entire network segment. In other words, all the IP devices on the same network segment should have the same Network ID.

202 . 34 . 19 . 8

Network ID *Host ID*

The rest of the address is known as the *Host ID*.

The Host ID is used to uniquely identify a host machine within a network.

So IP routers can just use the Network ID to route traffic to the right segment, and then look at the Host ID to make sure traffic reaches the right host.

But how do IP devices know how much of the address to use as the Network ID?

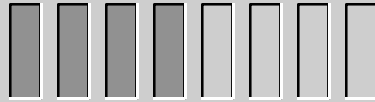


Remember I said how important binary is in IP addressing?

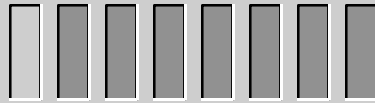
Well one reason for this is that we can use the value of the first few bits of the address to determine the size of the Network ID field.



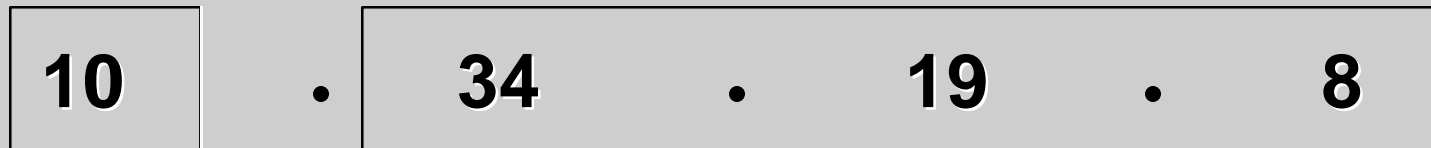
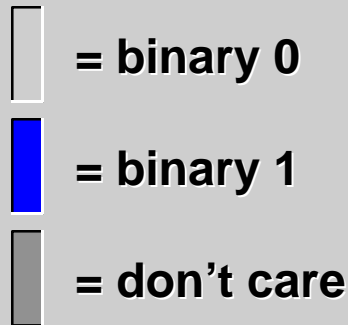
In particular I'll be looking at the values of the first 4 bits of the IP address.



In particular I'll be looking at the values of the first 4 bits of the IP address.
In fact I can throw away the other 24 bits for now.



Class A: Address Range 0..127

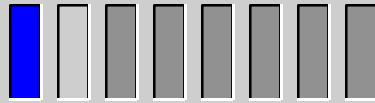


Network ID

Host ID

If the first bit is always set to zero, then no matter what the value of the other bits, the first byte can never be higher than 127.

These are known as *Class A Addresses*, and for a Class A address, the first 8-bits are assumed to be the Network ID, and the remaining 24-bits the Host ID.



Class B: Address Range 128..191

-  = binary 0
-  = binary 1
-  = don't care

132 . 34

Network ID

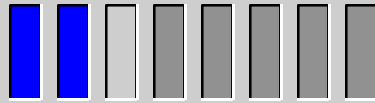
19 . 8

Host ID




If the first bit is set to binary “1”, and the second to “0”, then the first byte lies in the range 128 to 191.

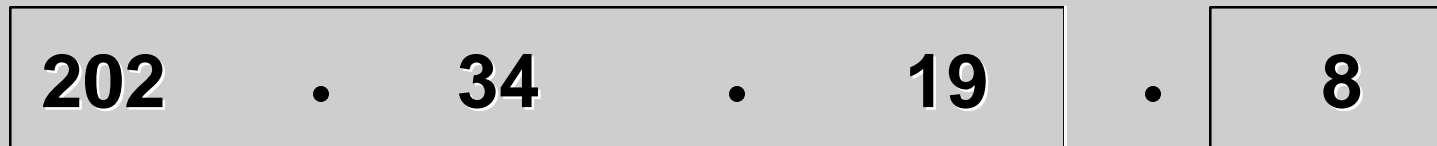
This is known as a Class B address.

The first 16 bits of a Class B address are assumed to represent the Network ID, and the remaining 16 bits the Host ID.



Class C: Address Range 192..223

-  = binary 0
-  = binary 1
-  = don't care



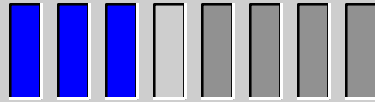
Network ID

Host ID

If the first two bits are set to binary “1”, and the third to “0”, then the first byte lies in the range 192 to 223.

This is known as a Class C address.

The first 24 bits of a Class C address are assumed to represent the Network ID, and the remaining 8 bits the Host ID.



Class D: Address Range 224..239



= binary 0



= binary 1



= don't care

If the first three bits are set to binary “1”, and the fourth to “0”, then the first byte lies in the range 224 to 239.

This is known as a Class D address.

Unlike Class A, B and C addresses, Class D addresses are reserved as *Multicast Addresses*. They do not use the concept of Network and Host ID.

A Review of Natural Address Class

Address Class	Value Range of 1st Byte	Network/Host ID Split
Class A	1..127	8 24
Class B		
Class C		

Class A addresses have their first byte in the range 1 to 127.

All the addresses in the 127.0.0.0 block (from 127.0.0.1 to 127.255.255.254) are reserved for loopback diagnostics, and are not valid IP Host addresses.

The first 8 bits of a Class A address are interpreted as the Network ID, and the remaining 24 bits as the Host ID.

A Review of Natural Address Class

Address Class	Value Range of 1st Byte	Network/Host ID Split
Class A	1..127	8 24
Class B	128..191	16 16
Class C		

Class B addresses have their first byte in the range 128 to 191.

The first 16 bits of a Class B address are interpreted as the Network ID, and the remaining 16 bits as the Host ID.

A Review of Natural Address Class

Address Class	Value Range of 1st Byte	Network/Host ID Split
Class A	1..127	8 24
Class B	128..191	16 16
Class C	192..223	24 8

Class C addresses have their first byte in the range 192 to 223.

The first 24 bits of a Class C address are interpreted as the Network ID, and the remaining 8 bits as the Host ID.

Why Have Different Address Classes?

Class A

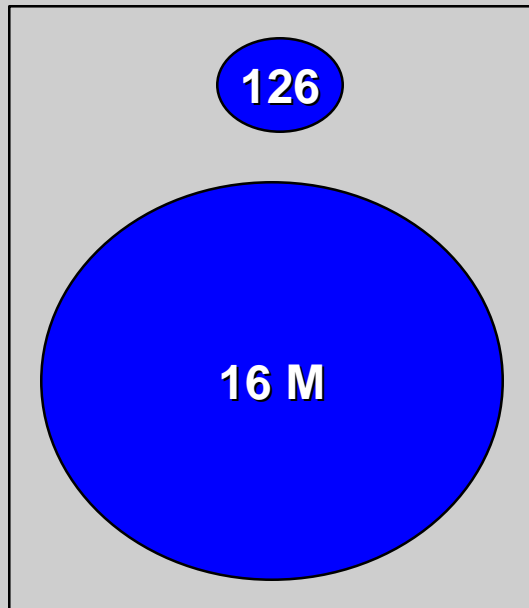


126

Remember Class A addresses take values between 1 and 127. Unfortunately, address 127.0.0.0 is reserved for a diagnostic function (loopback), and so there are 126 possible Class A networks.

Why Have Different Address Classes?

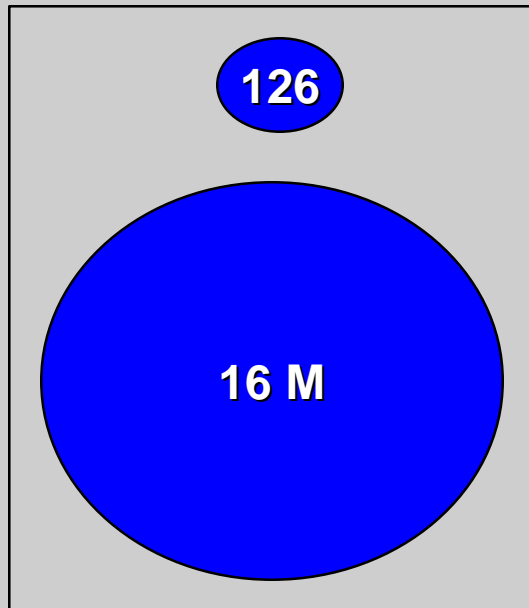
Class A



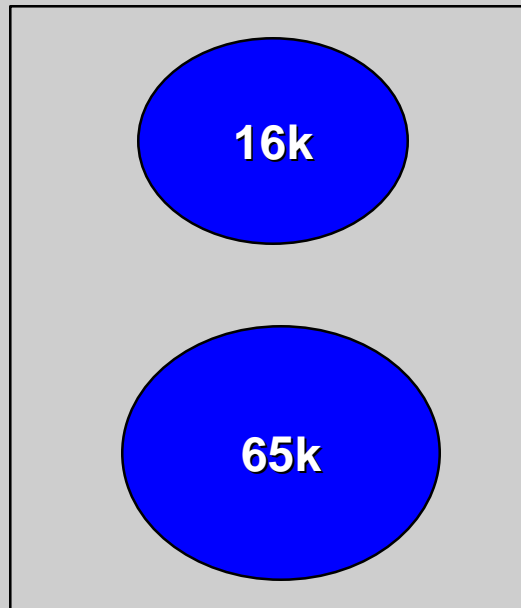
However, each of these networks can contain over 16 million hosts!

Why Have Different Address Classes?

Class A



Class B



Class B addresses are a little more balanced.

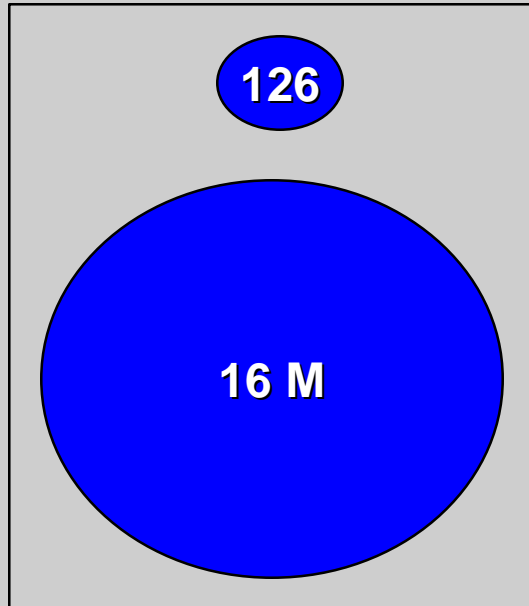
They take values between 128 and 191, but use the full eight bits of the second byte also.

So there are about 16,000 Class B networks available.

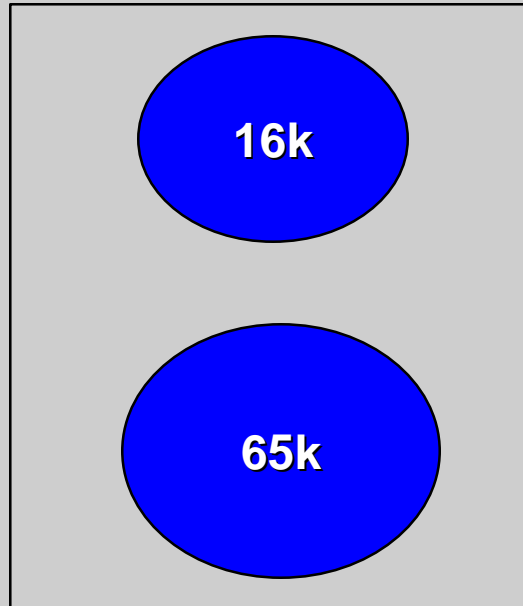
The remaining 16 bits of the address mean that each Class B network can support over 65,000 hosts.

Why Have Different Address Classes?

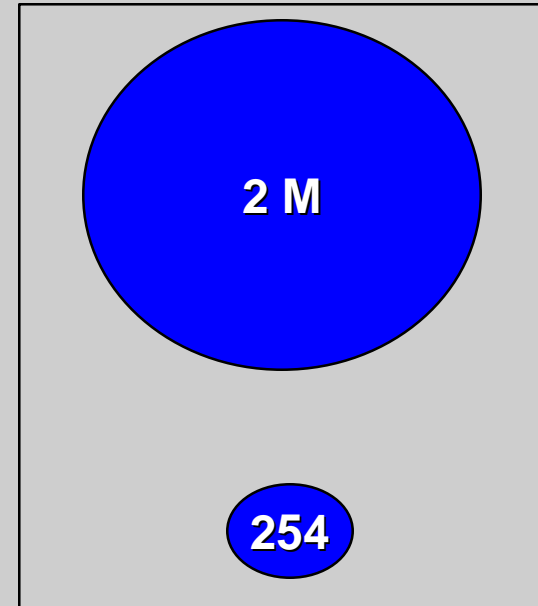
Class A



Class B



Class C



There are over 2,000,000 possible Class C addresses, but each address can only support 254 hosts.



The Structure of an IP Address

IP Address Classes

Section 3

Subnetting an IP Address

Applying Addresses

Reserved and Special Addresses

In this section we'll look at the most difficult concept in IP addressing, subnetting.



Subnetting is complicated, and so I'm going to give you a complete overview of the topic, and then go into the details...



- Subnetting is an additional level of address hierarchy used to identify the Network Segment, not the specific Host
- IP addresses are always 32-bits long, even when subnetted
- Natural address class still applies as the top level of hierarchy
- Therefore, subnet addressing must be “stolen” from the Host ID address space
- The amount of Host ID stolen is identified using a bit mask
- If we decide to subnet, then we type in a subnet bit mask to all of the hosts on the network segment
- We also type the same mask into the routers on the network segment, and this has to be done correctly





Class A: Address Range 0..127

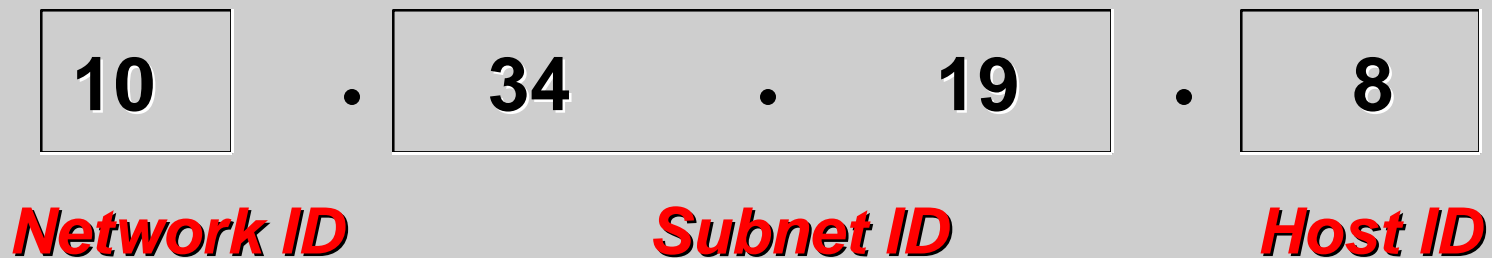
10 . 34 . 19 . 8

Network ID

Host ID

The end result of subnetting is that this Class A address, which has an 8-bit Network ID as its Natural Class...

Class A: Address Range 0..127



The end result of subnetting is that this Class A address, which has an 8-bit Network ID as its Natural Class...

...can be “transformed” into a *subnetted Class A address*, still retaining an 8-bit Host ID, but now having a 16-bit *Subnet ID* as part of the former Host ID field.

Contiguous Class A Network : 10.0.0.0

The effect on the network structure is to go from a contiguous Network ID capable of supporting almost 17,000,000 hosts...

Contiguous Class A Network : 10.0.0.0

Subnet
10.34.19.0

Subnet
10.34.20.0

Subnet
10.34.21.0

etc.

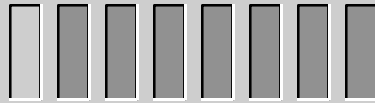
The effect on the network structure is to go from a contiguous Network ID capable of supporting almost 17,000,000 hosts...

...to a structured subnetted network. Each subnet can support up to 254 hosts, and there can be over 16,000 subnets. The yellow numbers on the addresses indicate the part of the address used as the Subnet ID.

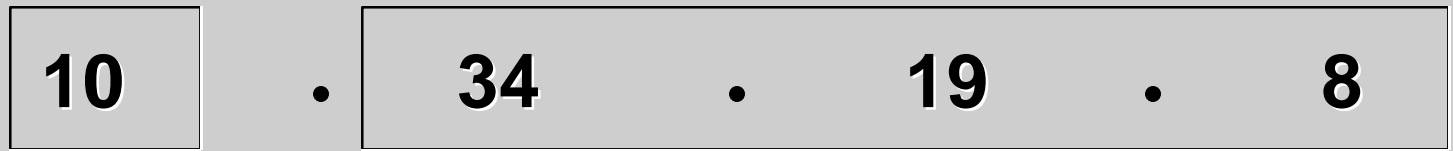
How do we decide subnet lengths?

Address Class	Value Range of 1st Byte	Network/Host ID Split
Class A	1..127	8 24
Class B	128..191	16 16
Class C	192..223	24 8

Remember that subnetting is applied *in addition* to the Natural Address Class.



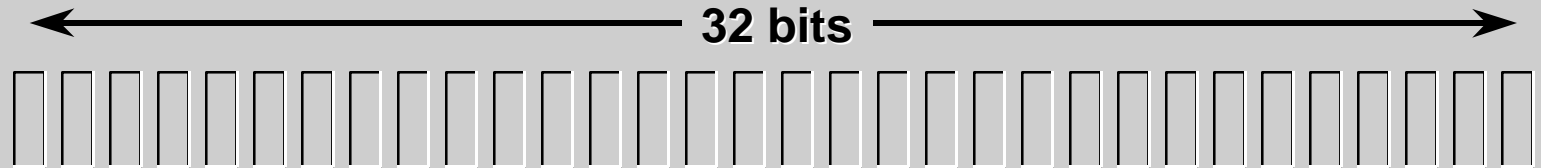
Class A: Address Range 0..127



Network ID

Host ID

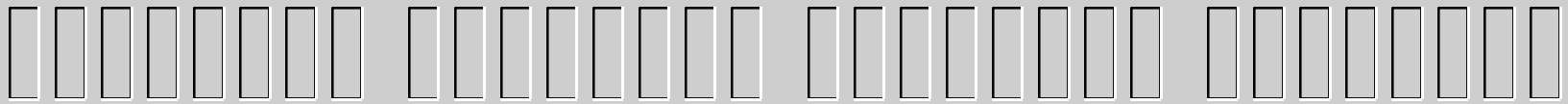
To leave the Natural Class intact, we “steal” subnet address space from the Host ID field. To do this we use a bit mask that we apply to the IP address. Let’s just quickly review IP address structure again...



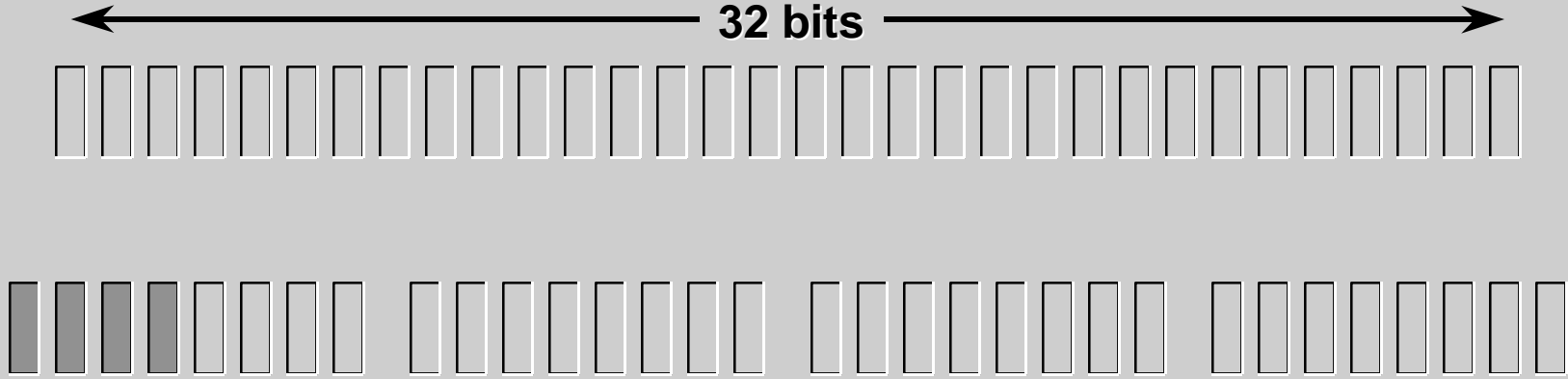
Subnetting is the third level of structure on the IP address.



← 32 bits →



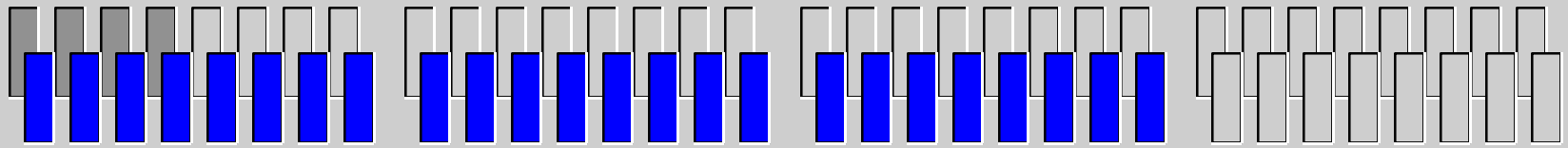
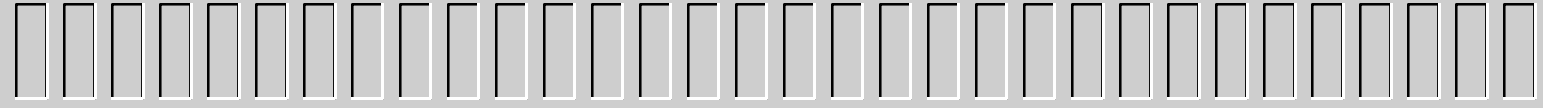
Remember the first level is the division of the binary address into four, 8-bit sections.



Then we use the value of the first four bits to determine the Natural Class (A, B, C or D) of the address.

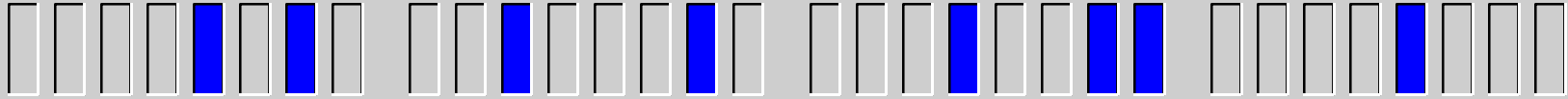



← 32 bits →



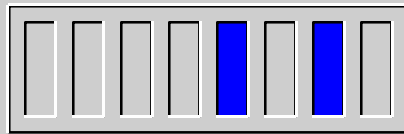
Binary Mask

To subnet the address, we'll apply a binary mask.
This is how it works...





 = binary 0  = binary 1

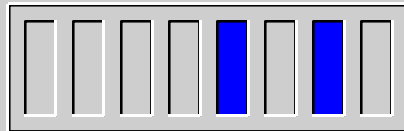
This is the binary representation of the Class A address 10.34.19.8.



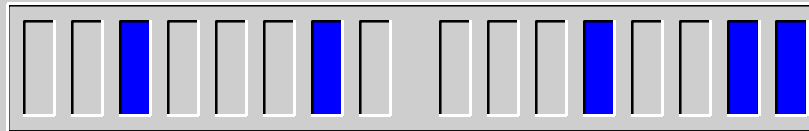
Network ID

 = binary 0  = binary 1

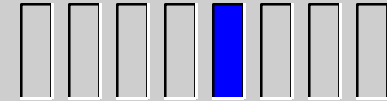
We already know this is a Class A address, and so the first 8-bits of the address represent the Network ID.





Network ID

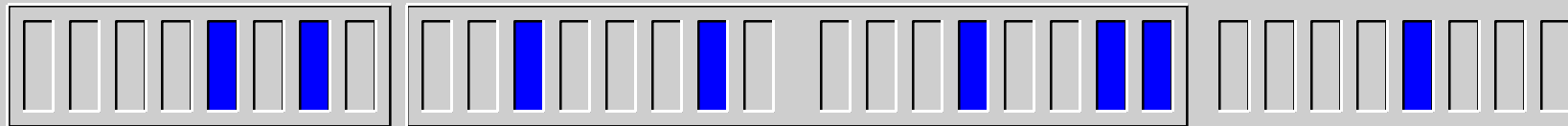


Subnet ID



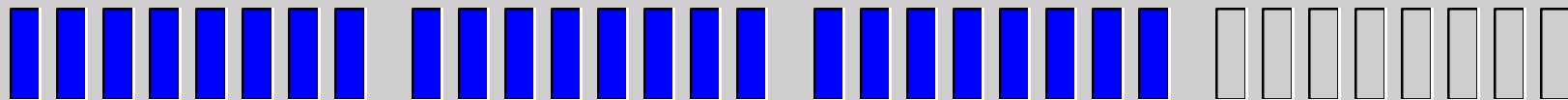
 = binary 0  = binary 1

Our objective is to use subnetting to create a 16-bit Subnet ID.



Network ID

Subnet ID



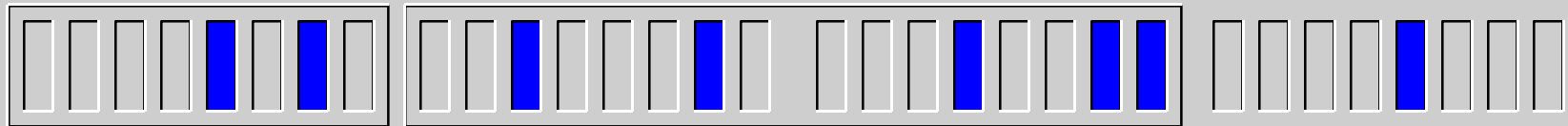
Binary Mask

 = binary 0  = binary 1

To do this we create a 32-bit binary mask. In the mask, we set bits to binary “1” if we want the same bit in the IP address to represent either the Network or Subnet ID.

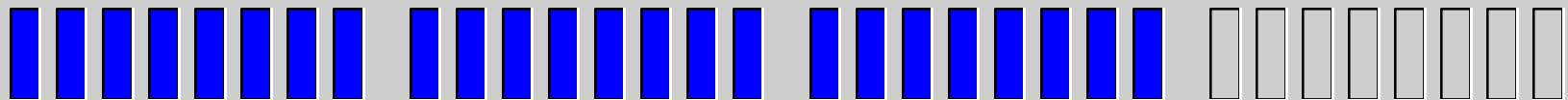
In other words, the only bits in the mask that should be binary “0” are those in the Host ID field.



When devices try to interpret the address structure, they need *both* the IP address *and* the subnet mask.



Network ID

Subnet ID

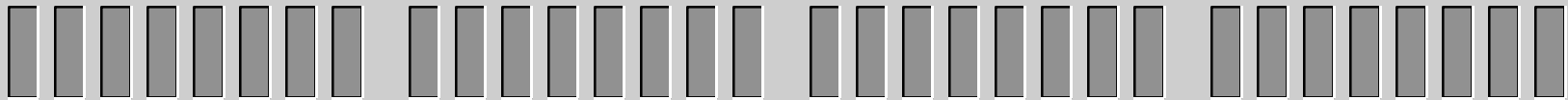
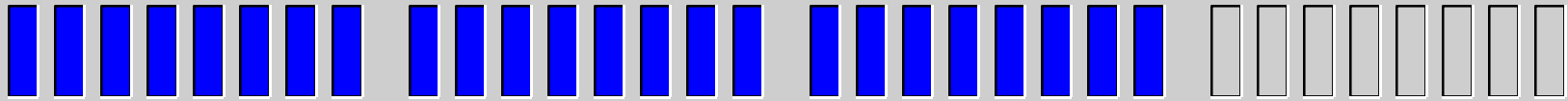
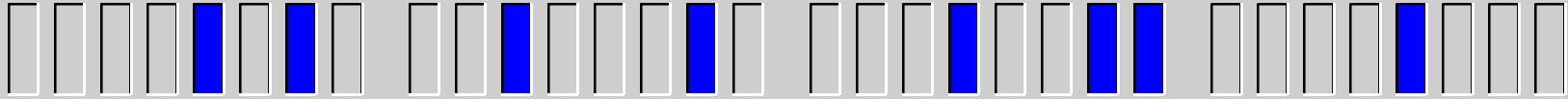





 = binary 0  = binary 1

When end stations interpret the IP address and mask, they use a binary AND process.

In a binary AND, two numbers are compared bit-by-bit (so they both have to be the same number of bits in length *including* leading zeroes).

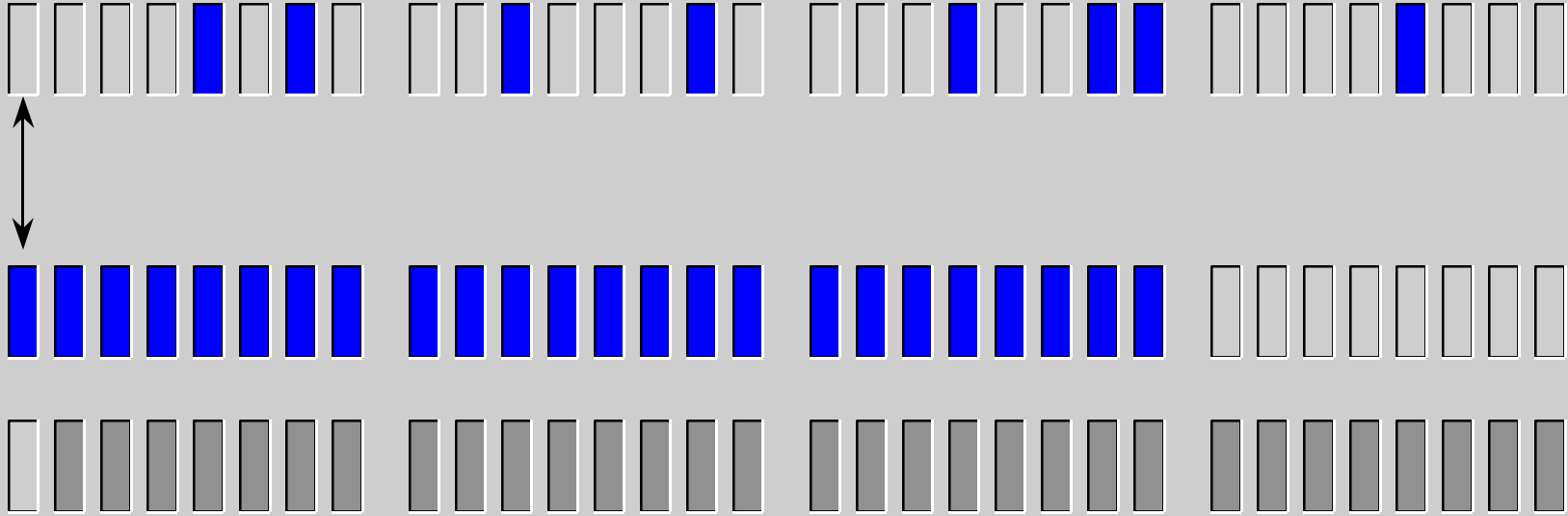
The number that results from this AND process is also 32-bits long.






 = to be decided  = binary 0  = binary 1

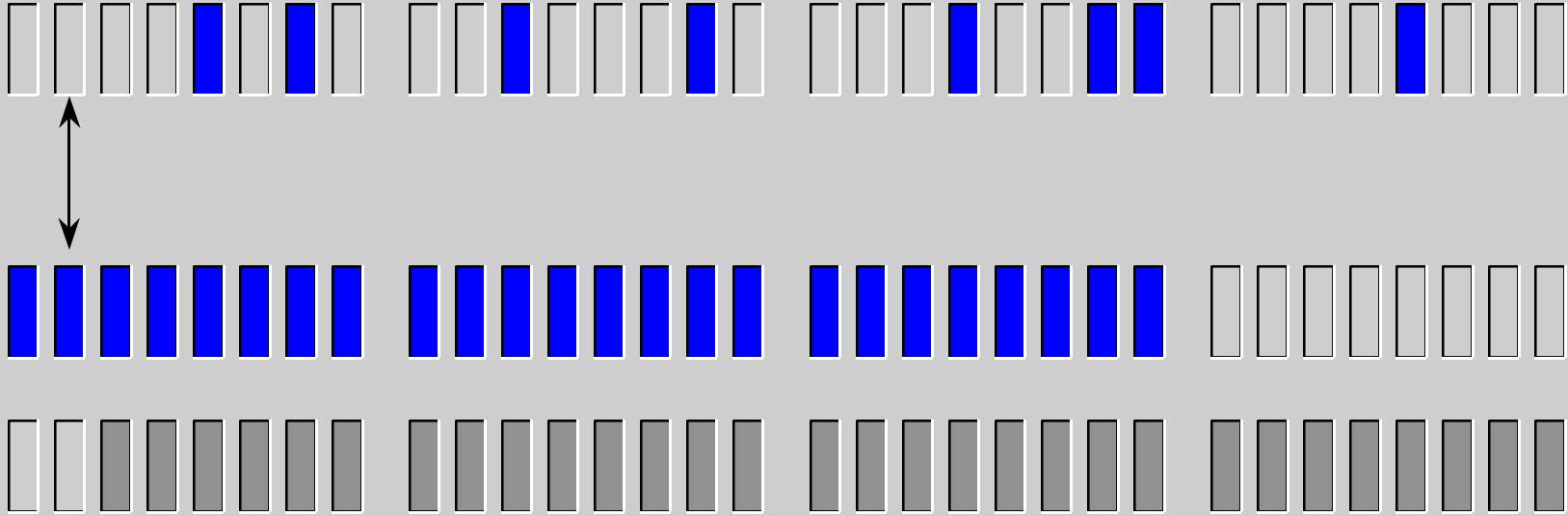
As each bit is compared, then the equivalent bit in the result is only set to binary “1” if both bits are set to “1” in the mask and the IP address. If either bit is set to “0” then the result bit is also set to 0.




Let’s go through the first 8-bits...



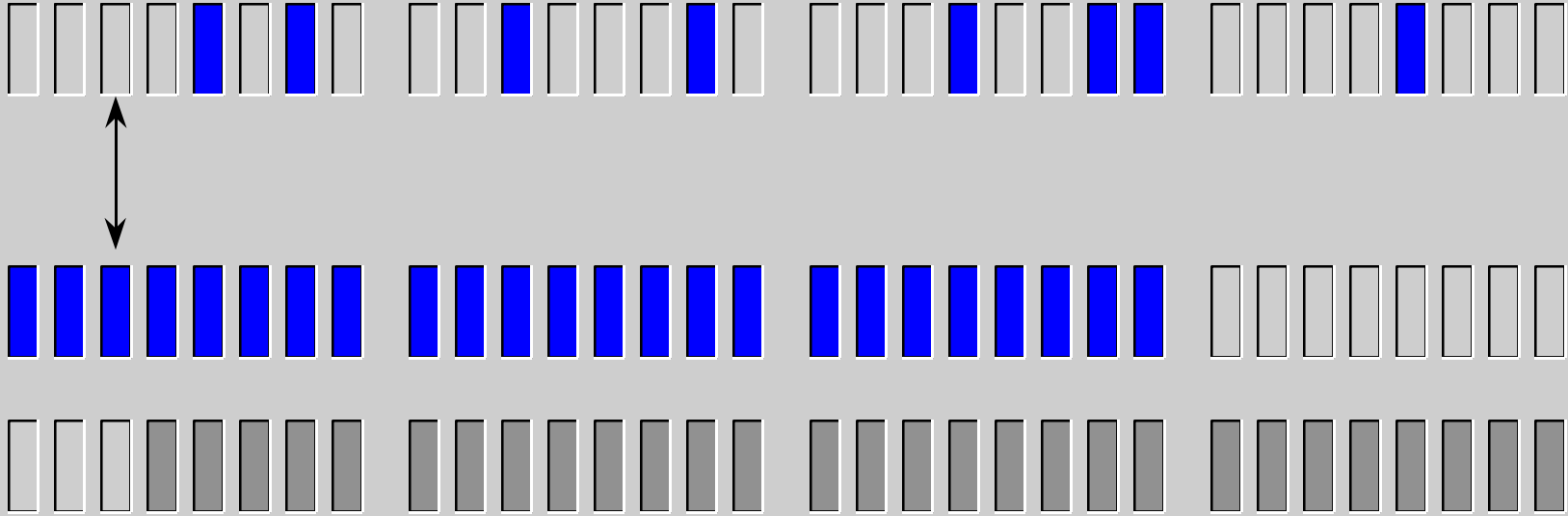
 = to be decided  = binary 0  = binary 1




0 AND 1 = 0



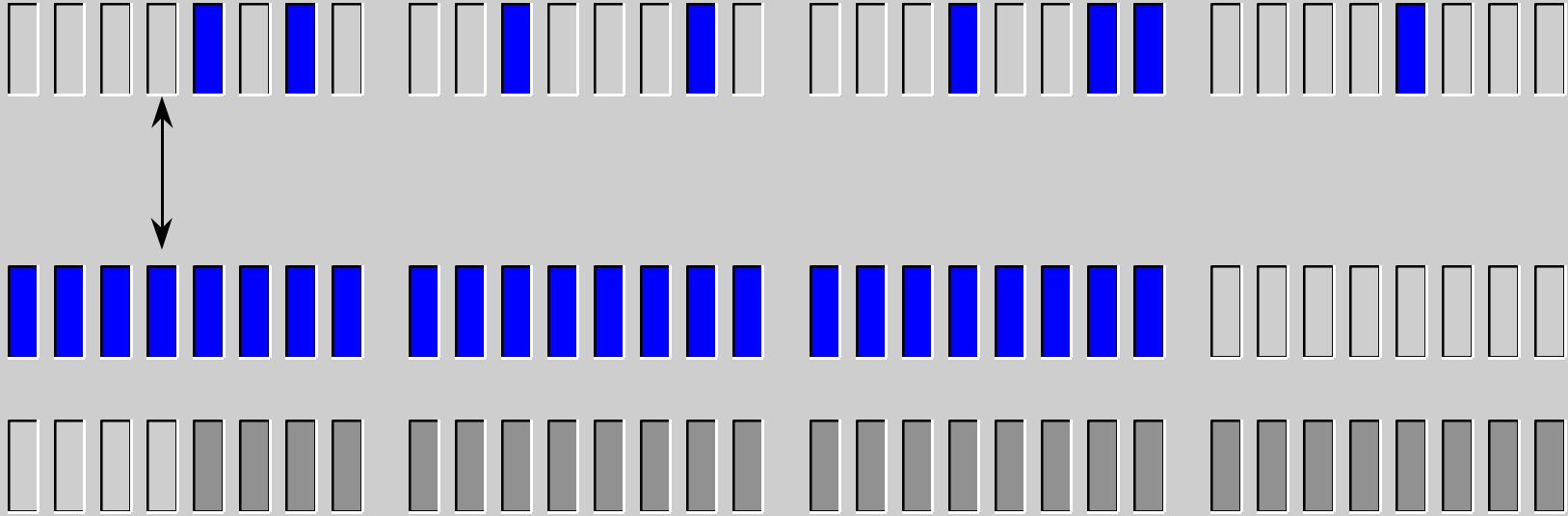
 = to be decided  = binary 0  = binary 1




0 AND 1 = 0



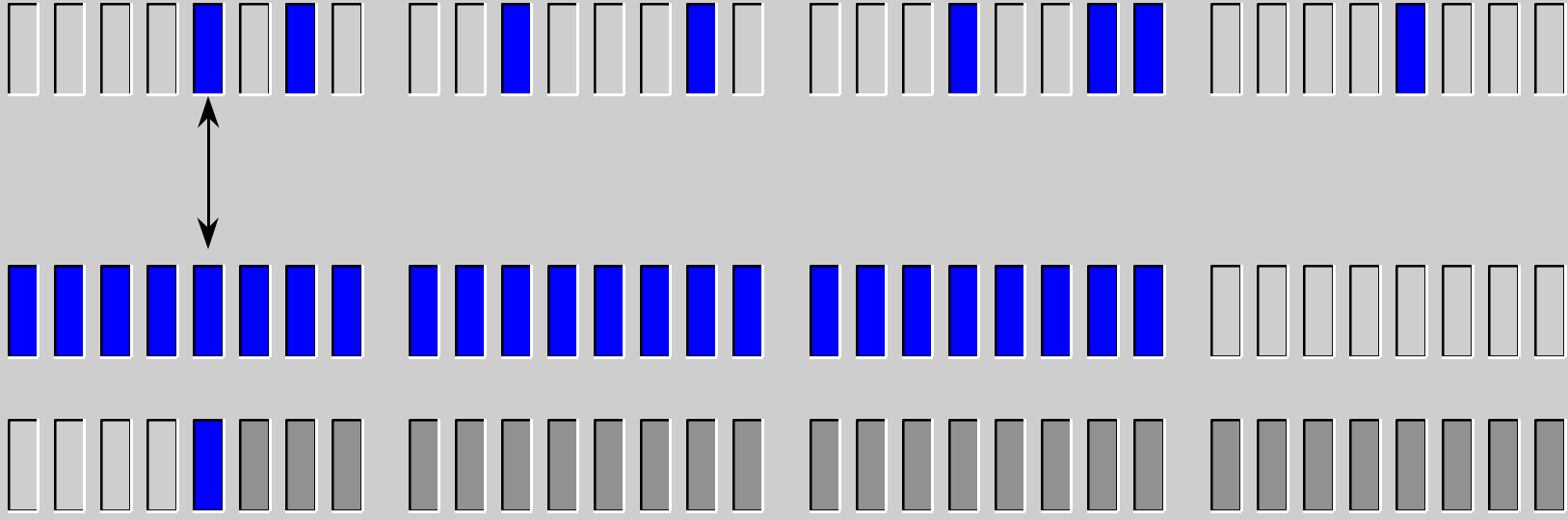
 = to be decided  = binary 0  = binary 1




0 AND 1 = 0



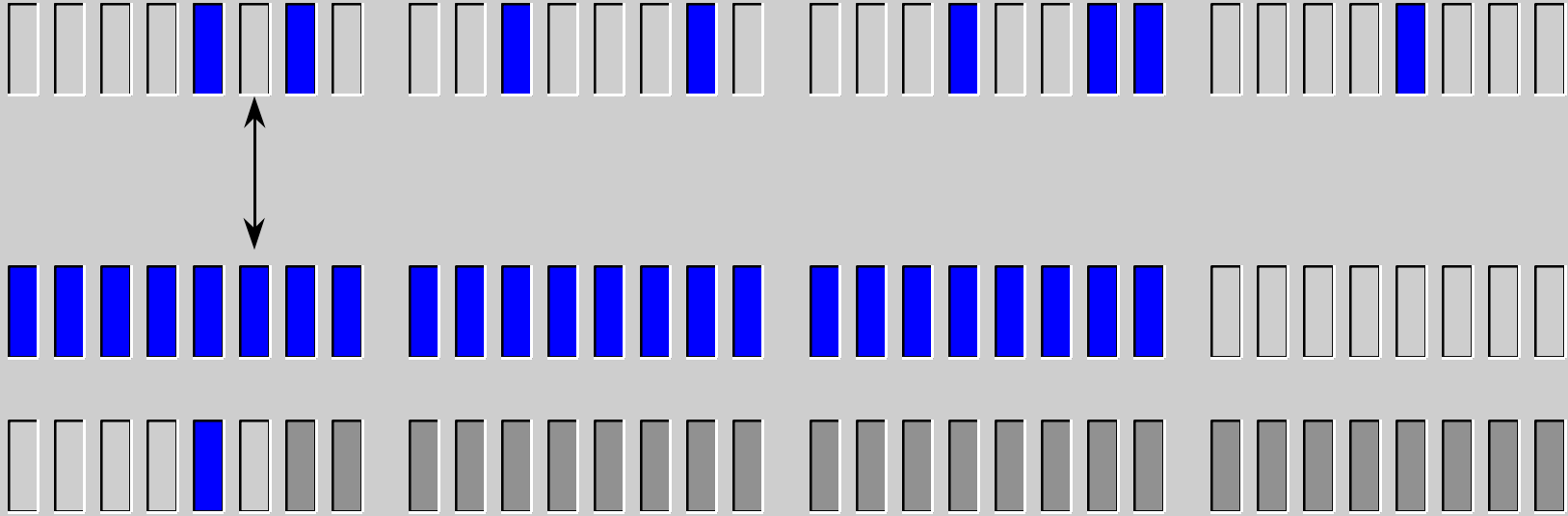
 = to be decided  = binary 0  = binary 1




0 AND 1 = 0



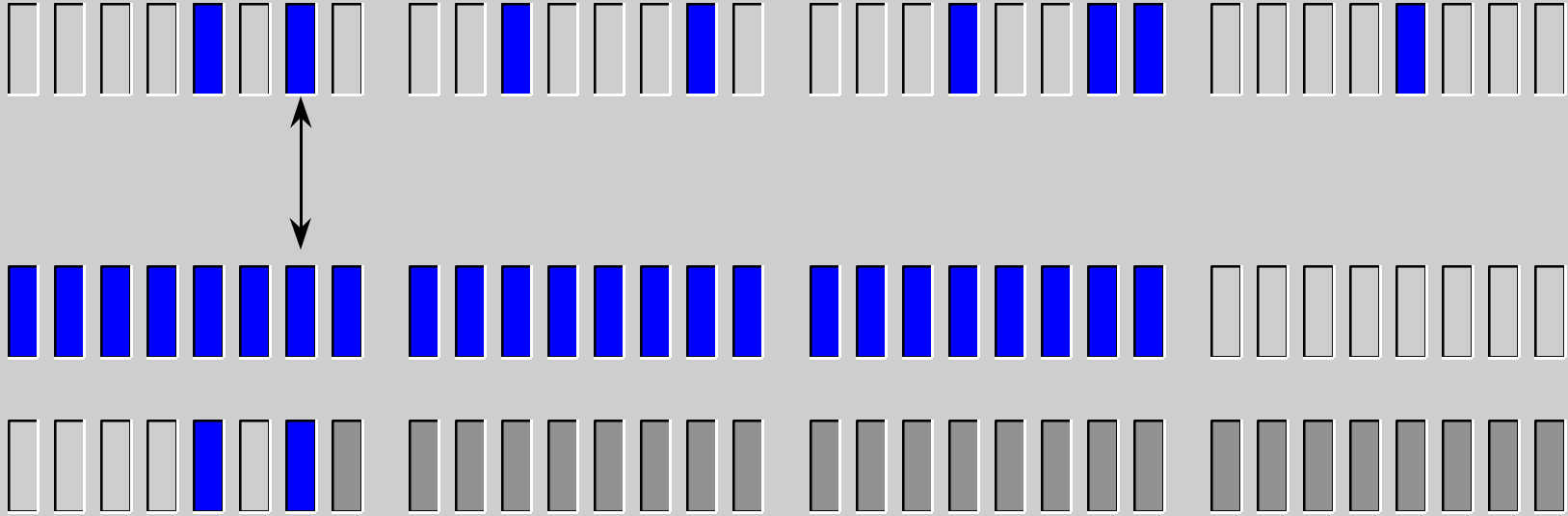
 = to be decided  = binary 0  = binary 1




1 AND 1 = 1



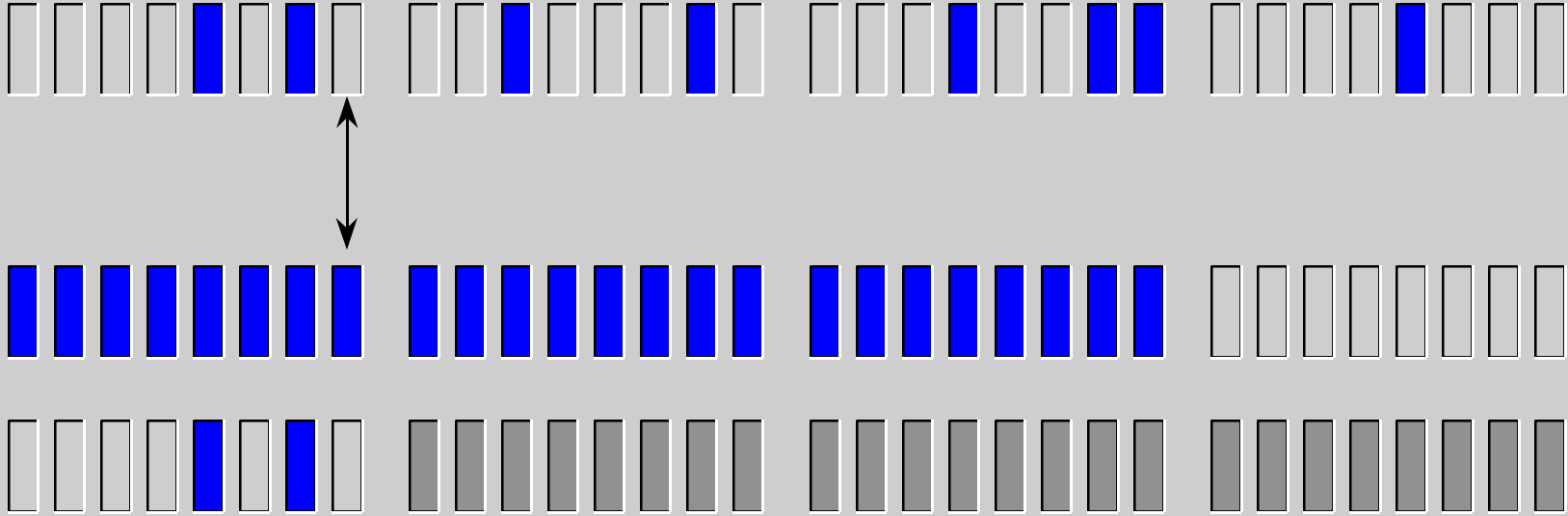
 = to be decided  = binary 0  = binary 1




0 AND 1 = 0



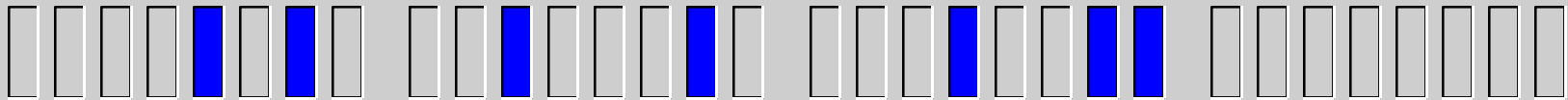
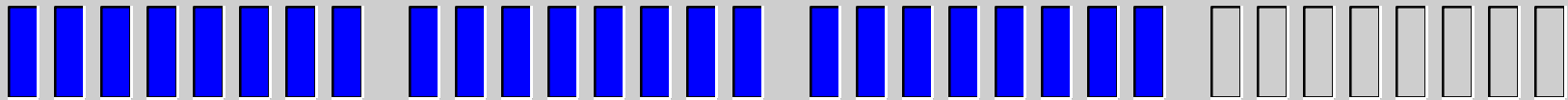
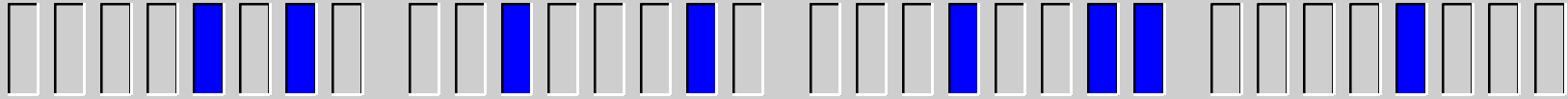
 = to be decided  = binary 0  = binary 1


1 AND 1 = 1



 = to be decided  = binary 0  = binary 1

0 AND 1 = 0

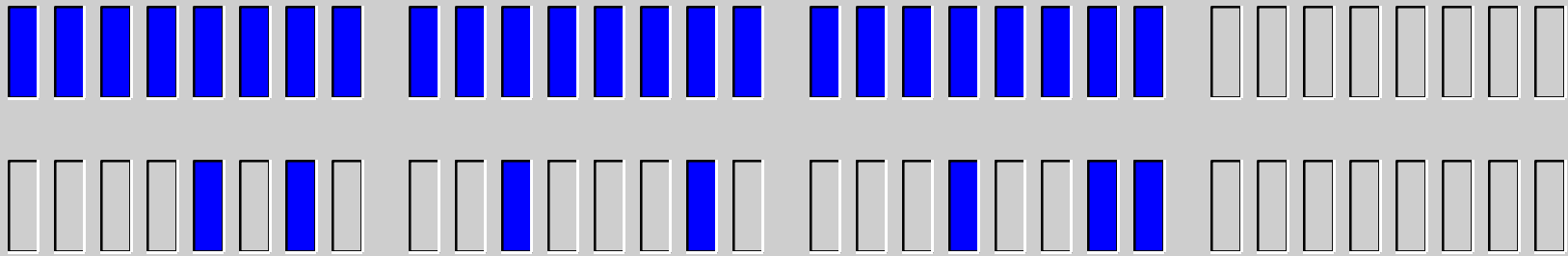


 = binary 0  = binary 1

And here's the completed AND operation.



10 . 34 . 19 . 8

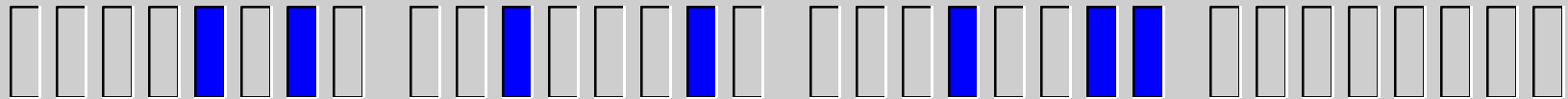




 = binary 0  = binary 1

Here's the original IP address in decimal...

10 . 34 . 19 . 8

255 . 255 . 255 . 0



 = binary 0  = binary 1

Here's the original IP address in decimal...
... here's the subnet mask...

10 . 34 . 19 . 8

255 . 255 . 255 . 0

10 . 34 . 19 . 0

Here's the original IP address in decimal...

... here's the subnet mask...

... and the resulting 32-bit number.

10 . 34 . 19 . 0

Note that this number is *not* an IP address.

It represents the part of the address that should be considered as the “area code” (in effect the combination of Network ID and Subnet).



The Structure of an IP Address

IP Address Classes

Subnetting an IP Address

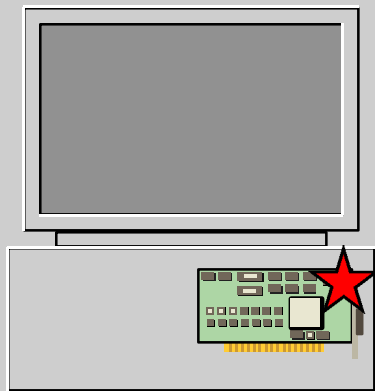
Section 4

Applying Addresses

Reserved and Special Addresses

Let's look at how we use IP addresses.

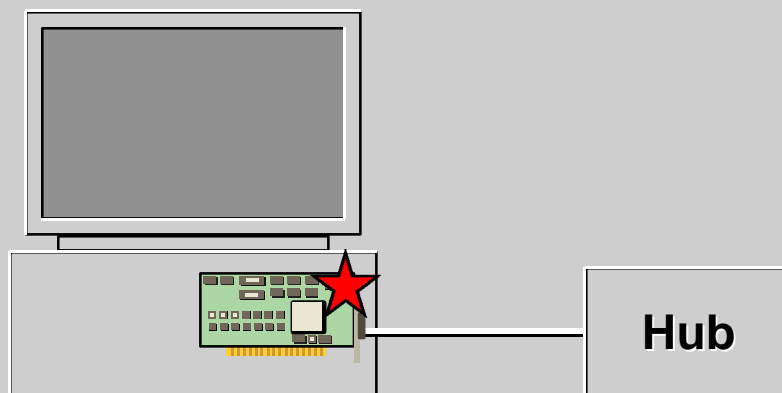
Where do IP Addresses Go? ★



When we install a LAN adapter in a PC or workstation, we give an IP address to this interface.

In effect, the IP address represents the physical end point of the network connection into the workstation.

Where do IP Addresses Go? ★

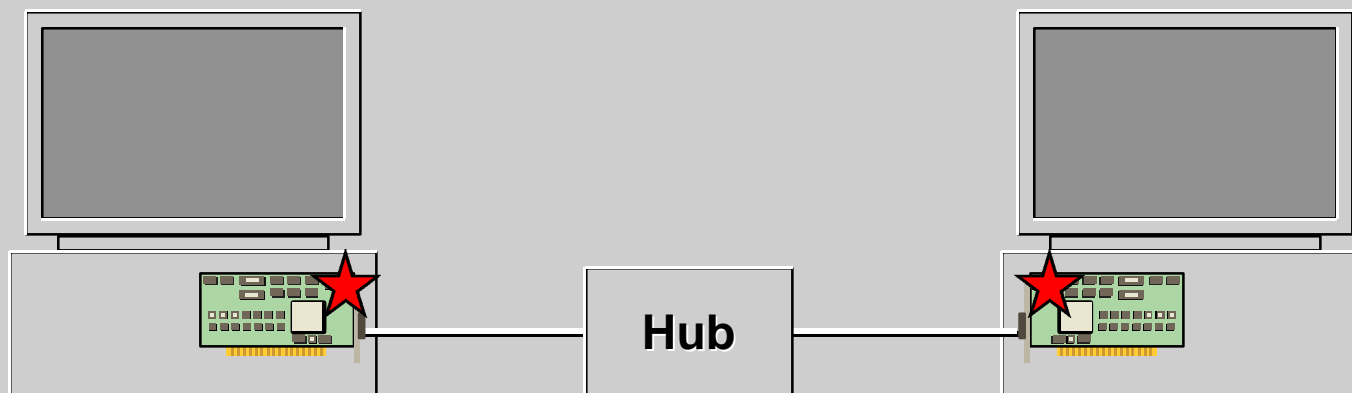


In a typical Structured Cabling environment, our workstation will connect over UTP cable to a Cable Hub.

Cable Hubs *do not need IP addresses* to perform their basic function.

However, if we want to manage a Cable Hub using SNMP, we do need to allocate an IP address to the hub.

Where do IP Addresses Go? ★

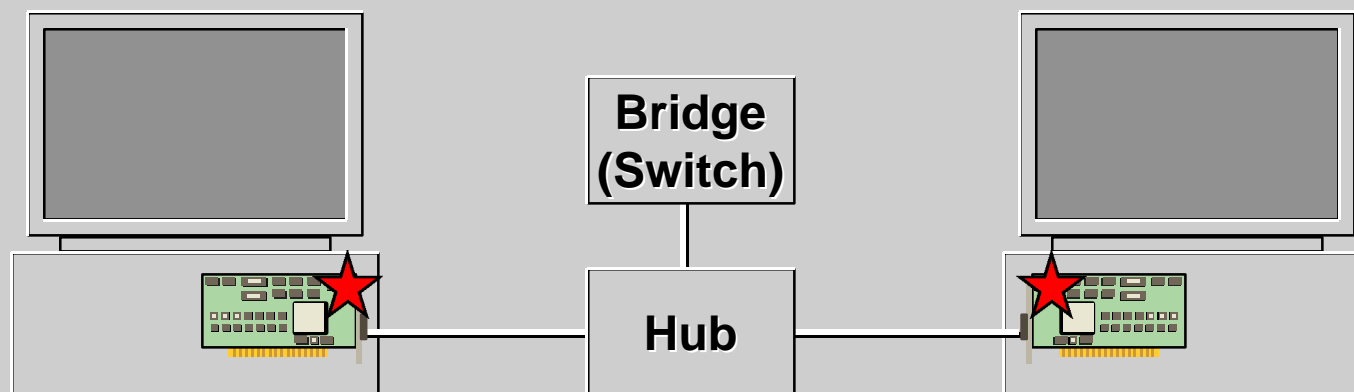


Each of the hosts attached to the same cabling hub will need an IP address.

If the hub is configured so that these hosts are on the same LAN segment, they *must* be configured with the same Network ID, or the same Network/Subnet ID *and* identical subnet masks.

If the address and mask values are not configured in a consistent way, then there will be problems with the communication between these hosts,

Where do IP Addresses Go? ★

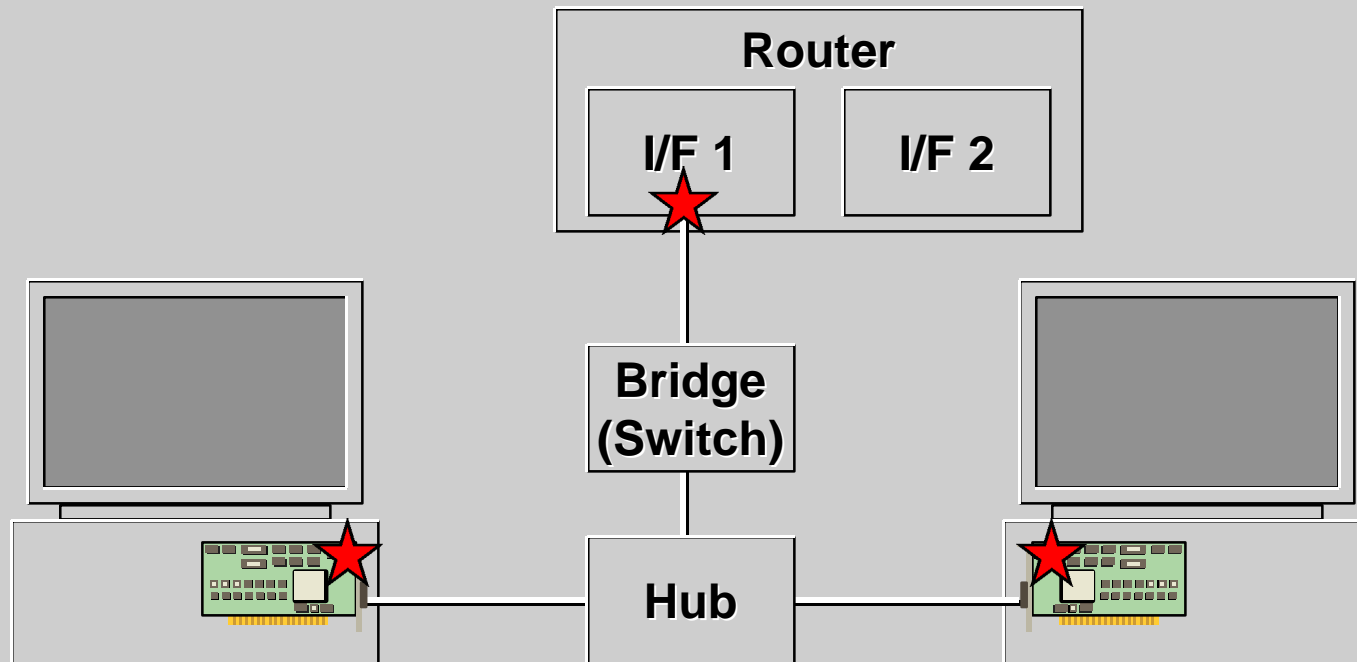


Like Cable Hubs, MAC Layer bridges (or Switches) do not need an IP address to function, but they do need an address to be managed by SNMP.

Again, like cabling hubs, hosts that are connected through a MAC Layer bridge (or Switch) must adopt the same addressing convention as though they were on the same LAN segment.

We can summarize this by saying that REPEATERS, HUBS, BRIDGES and SWITCHES are *transparent* to IP addressing.

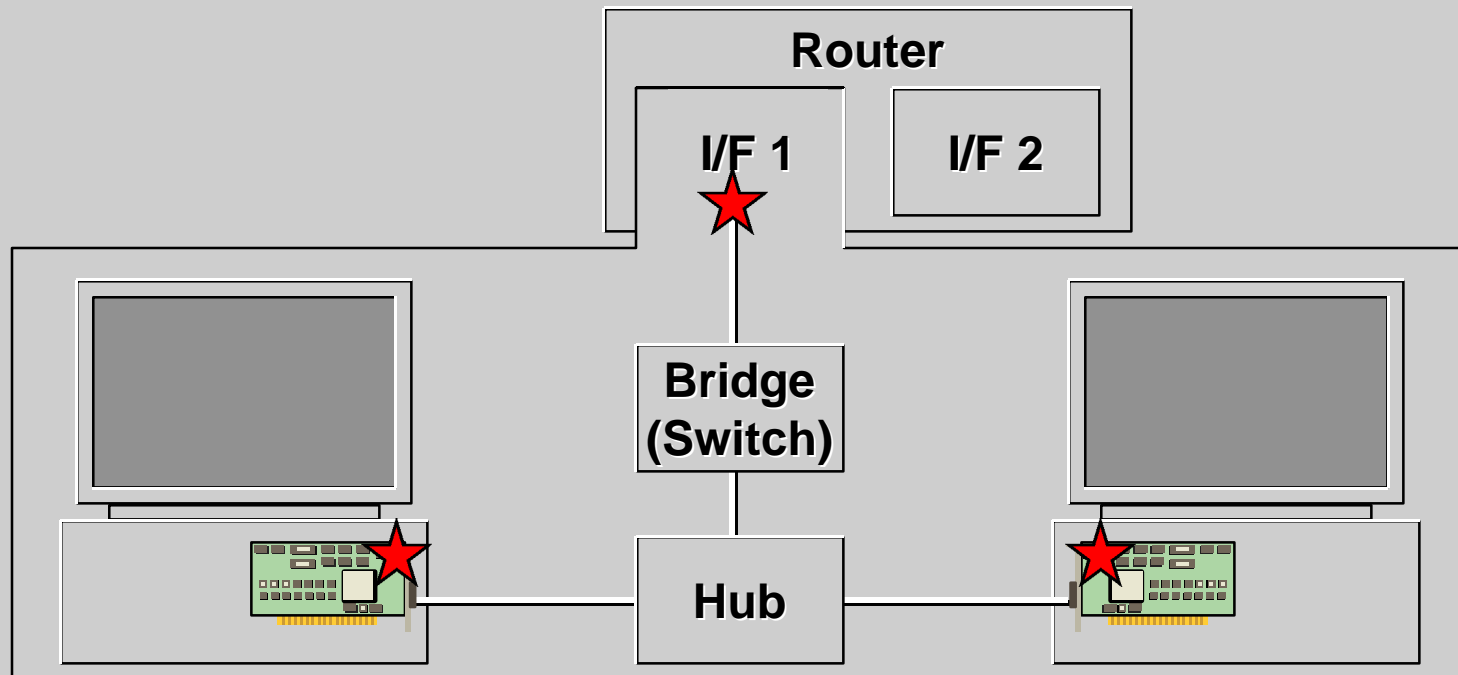
Where do IP Addresses Go? ★



Routers are full Network Layer devices.

Each interface on a router has an IP address assigned to it. To communicate with hosts on that segment, the Network ID (or Network/Subnet ID) must be the same as the hosts.

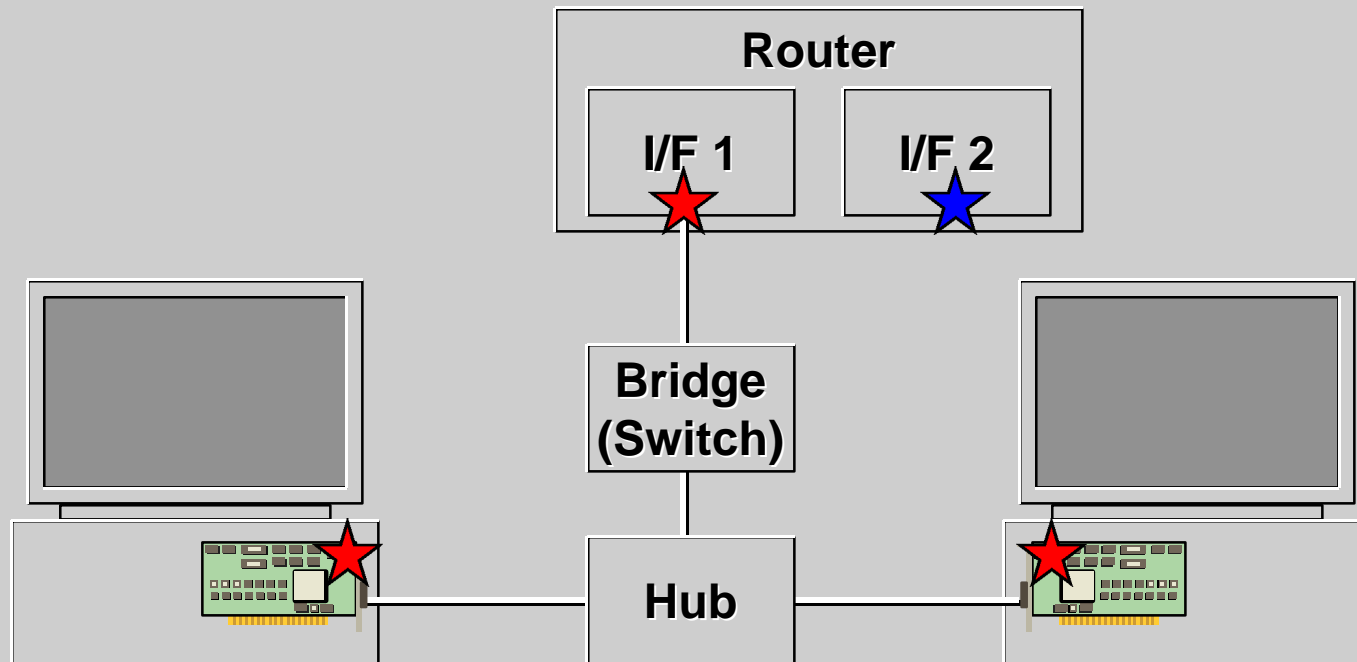
Where do IP Addresses Go? ★



In other words, all the IP addresses in this logical area must have the same Network ID, or if subnetting used used the same Network/Subnet ID and Subnet Mask.

(Note: Sometimes we can “cheat” with subnet masks on routers because they know more about the structure of the network than hosts. This technique is called *ARP Subnet Routing*. It’s beyond this basic introductory tutorial.)

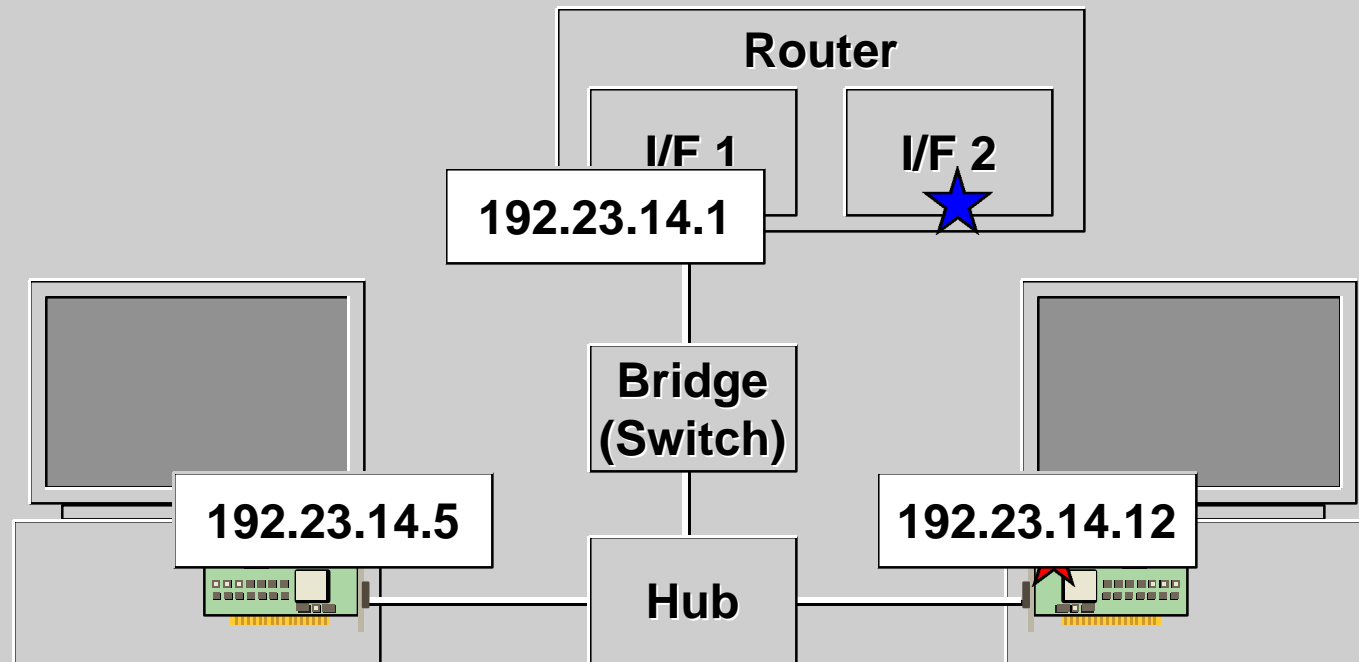
Where do IP Addresses Go? ★



Conversely the IP addresses assigned to other ports on the router *must* be allocated *different* Network IDs (or Subnet IDs if subnetting is used).

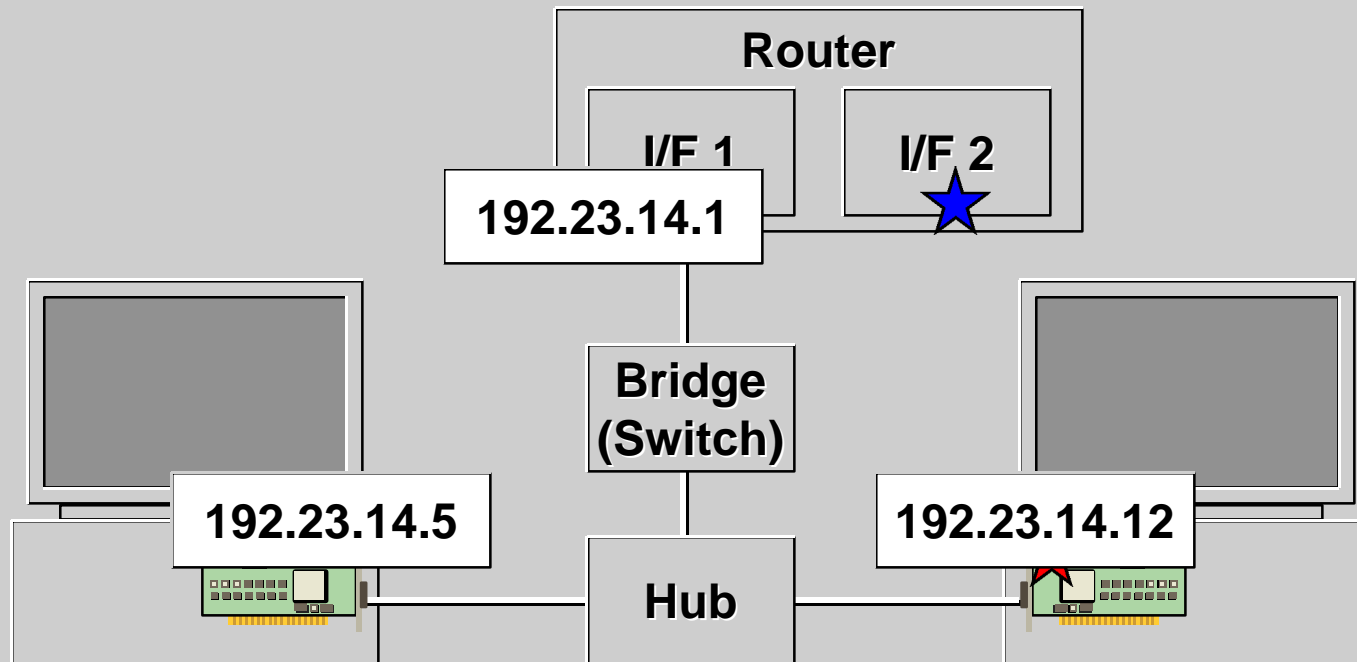
Let me show you an example...

Example: Class C, No Subnet



This is a Class C addressing scheme, with no subnetting.

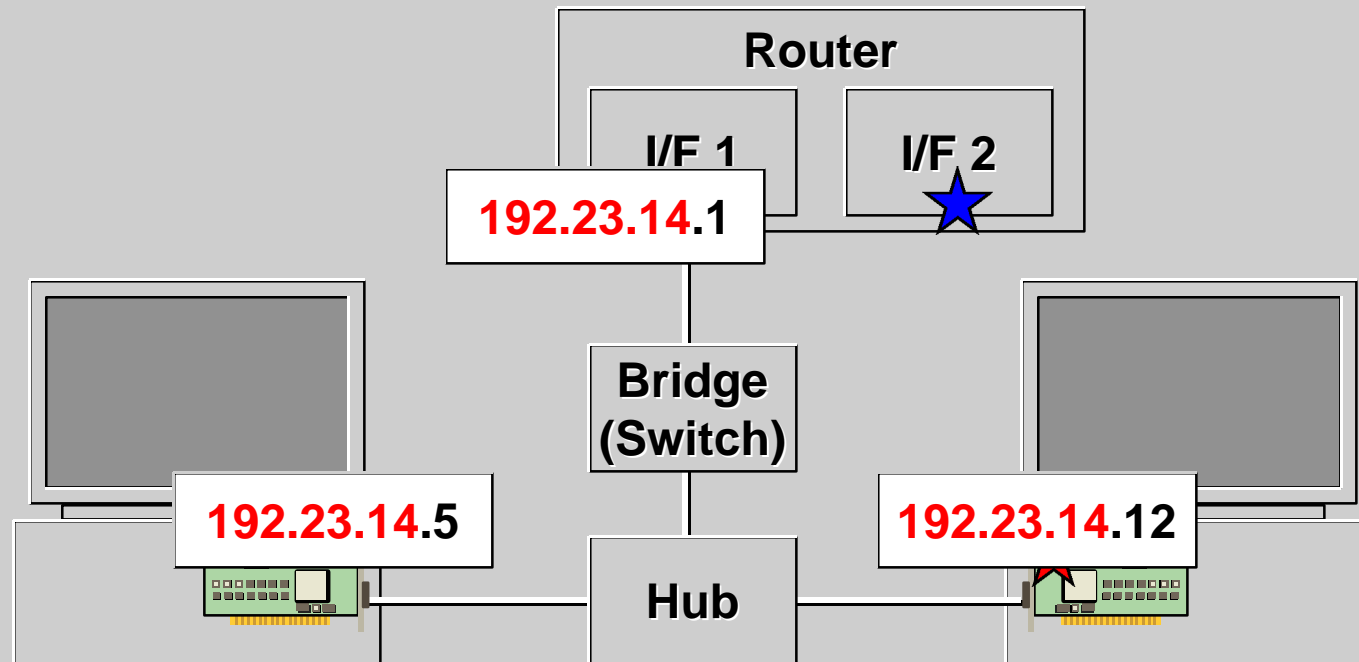
Example: Class C, No Subnet



This is a Class C addressing scheme, with no subnetting.

Remember with a Class C address, we take the first 24 bits as Network ID, and the remaining 8 bits as Host ID.

Example: Class C, No Subnet

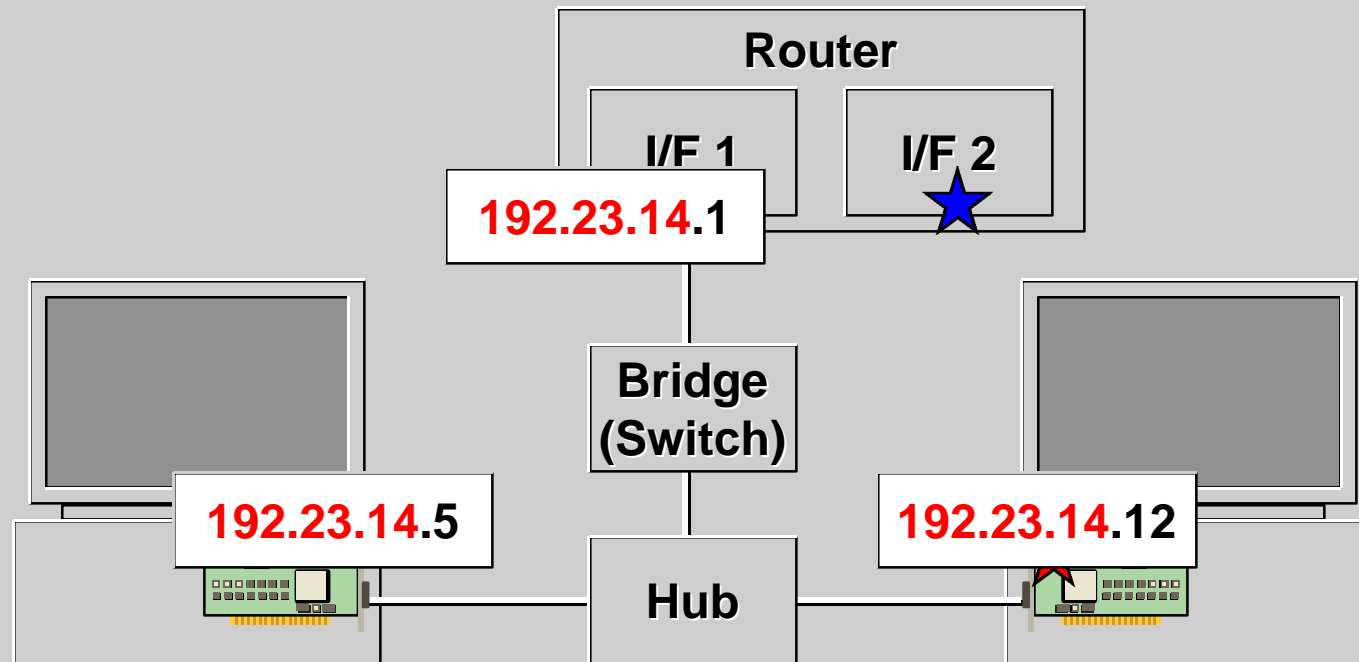


This is a Class C addressing scheme, with no subnetting.

Remember with a Class C address, we take the first 24 bits as Network ID, and the remaining 8 bits as Host ID.

So the red-shaded section of the address is the Network ID. You'll notice that all the Network IDs are identical.

Example: Class C, No Subnet



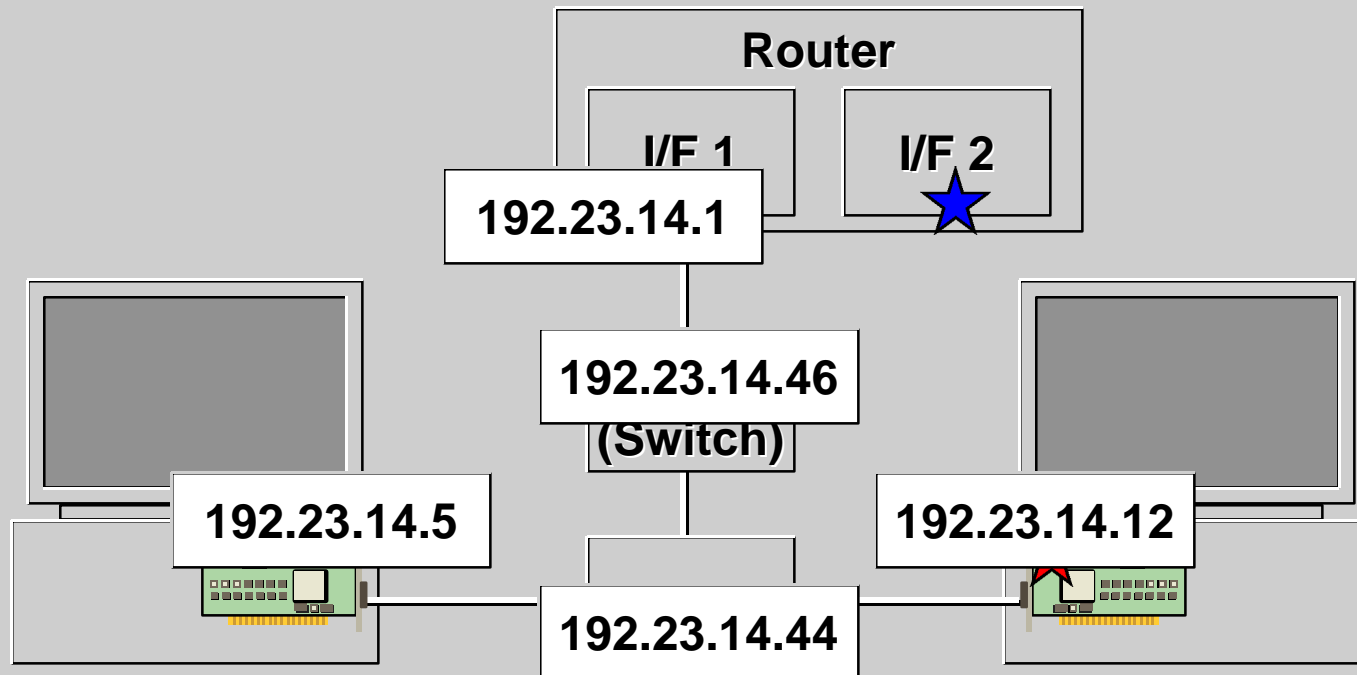
This is a Class C addressing scheme, with no subnetting.

Remember with a Class C address, we take the first 24 bits as Network ID, and the remaining 8 bits as Host ID.

So the red-shaded section of the address is the Network ID. You'll notice that all the Network IDs are identical.

And all the Host IDs are unique.

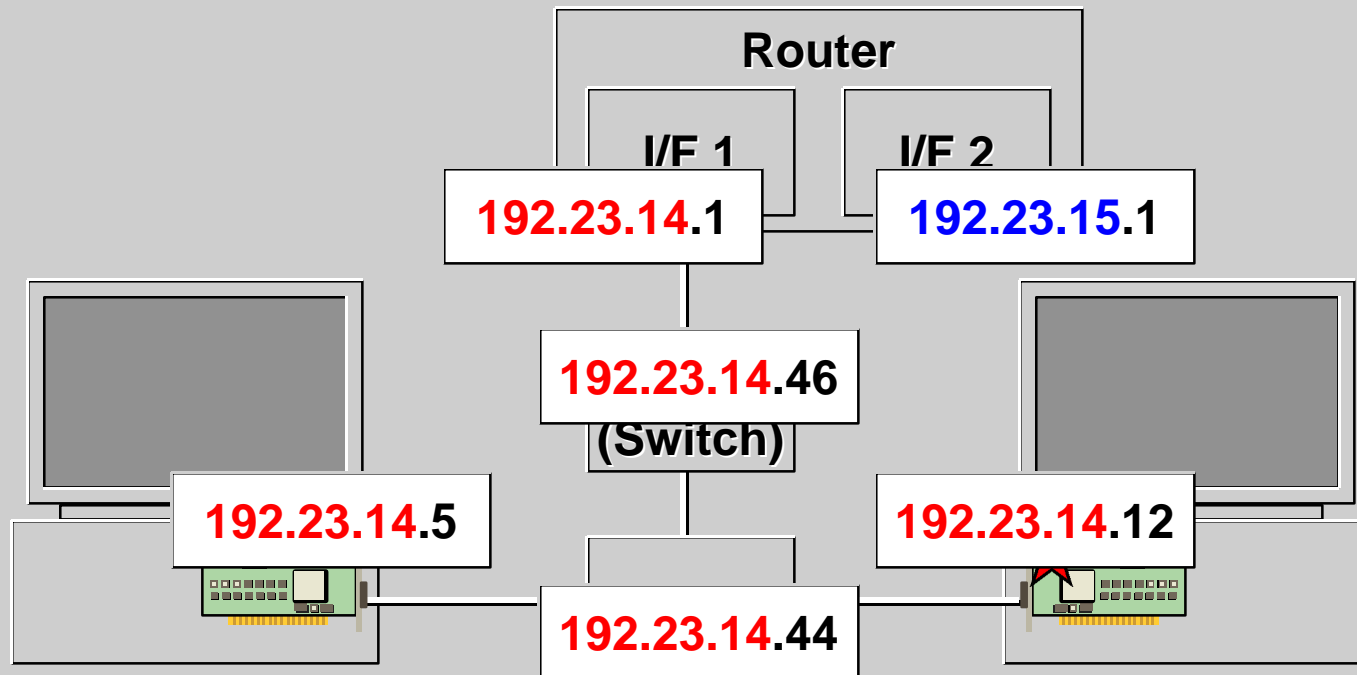
Example: Class C, No Subnet



Because we want to manage the Hub and Bridge with SNMP, we allocate IP addresses to these devices too.

But remember; Repeaters, Hubs, Bridges and Switches don't need an IP address to perform their basic function.

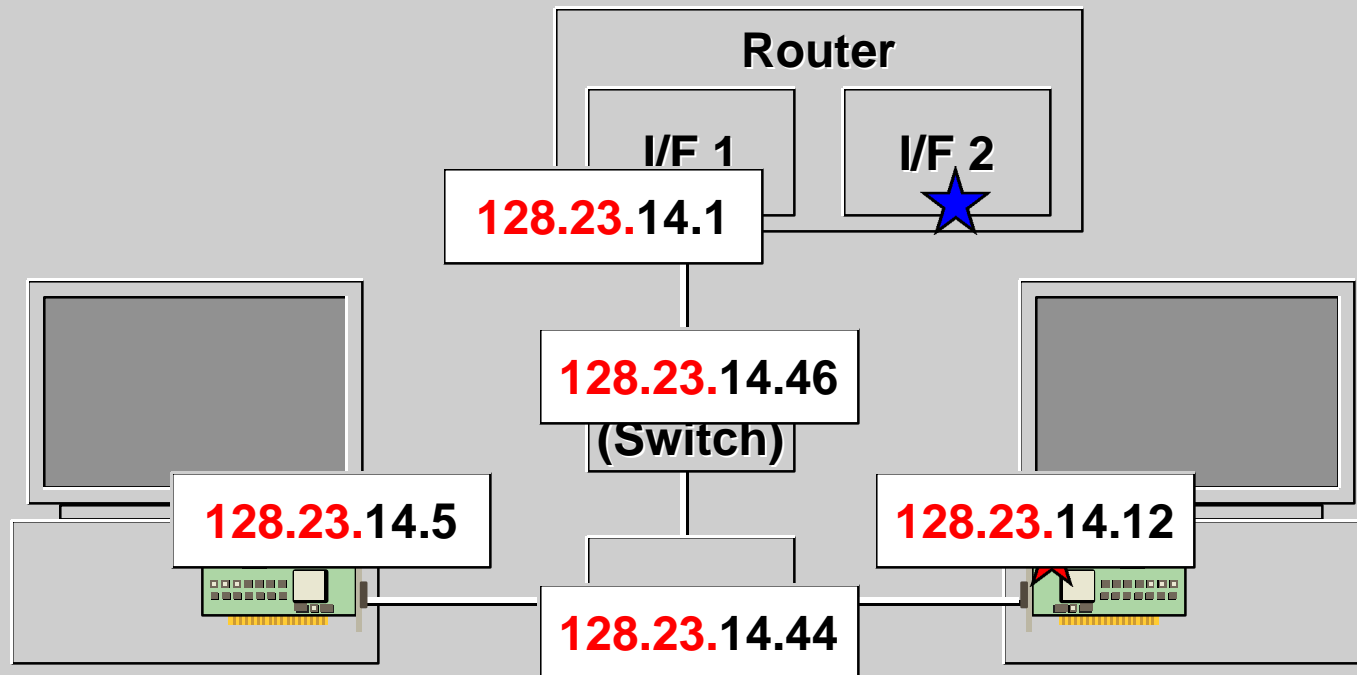
Example: Class C, No Subnet



The IP address of IF/2 on the router *must* have a different Network ID.

The blue-highlighted section of the address is the Network ID and you can see it's different from the red-highlighted sections of the addresses on IF/1.

Example: Class B with 8-bit Subnet



In this example I'm using a Class B address.

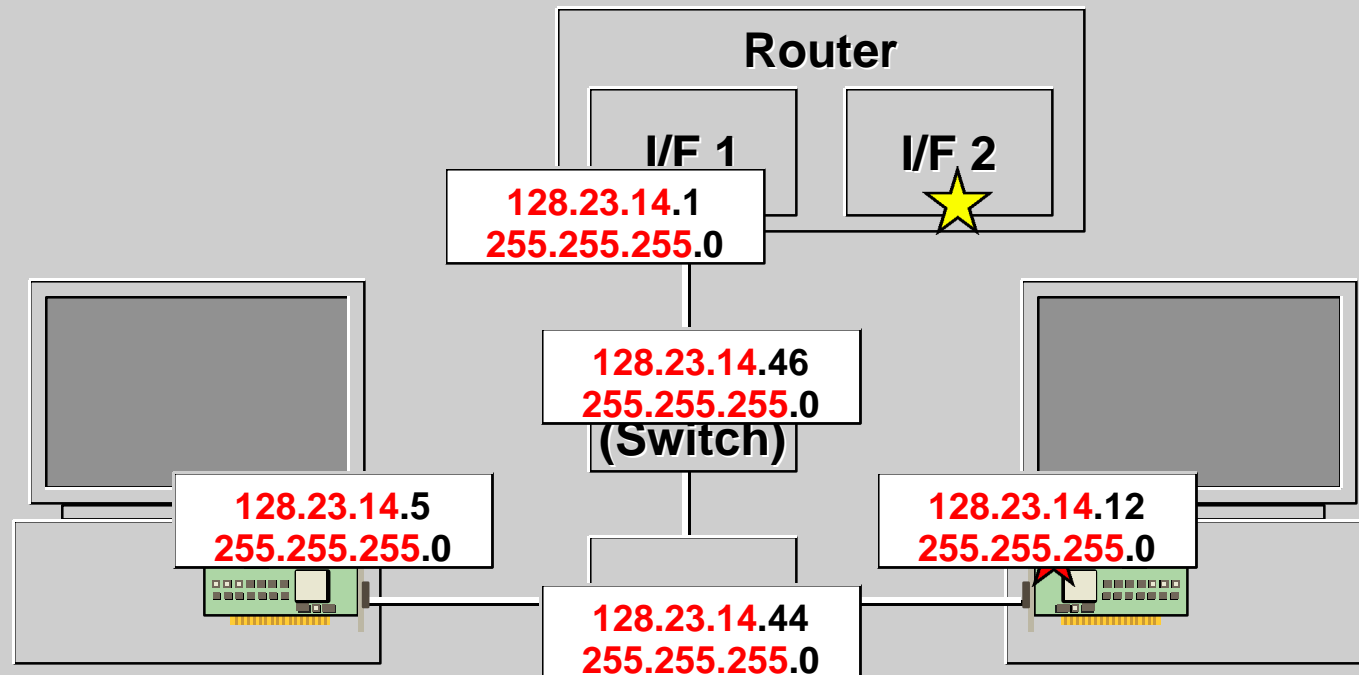
Class B networks have the first 16-bits of the address to represent the Network ID. I've red-highlighted the Network IDs in this example.

You'll notice the same two rules apply as in the Class C example.

First, all the Network IDs are the same, and second the Host IDs are unique.

However, we want to subnet this Class B network.

Example: Class B with 8-bit Subnet

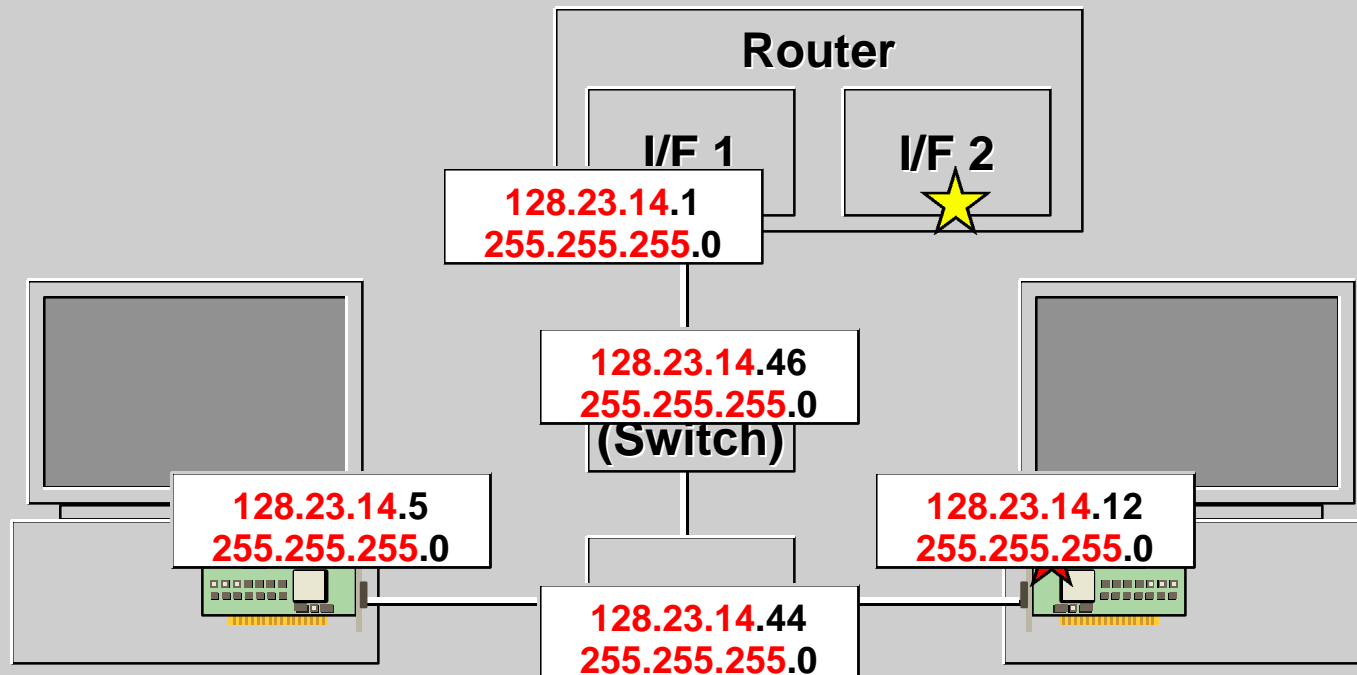


We need to type in the *same subnet mask* (255.255.255.0) to each of the hosts in the subnet.

By applying the mask, you can see that I've also changed the red-highlighted part of the address to include the third byte.

Note that the rule still applies that all hosts *within the same subnet* must have identical Subnet IDs, but unique Host IDs.

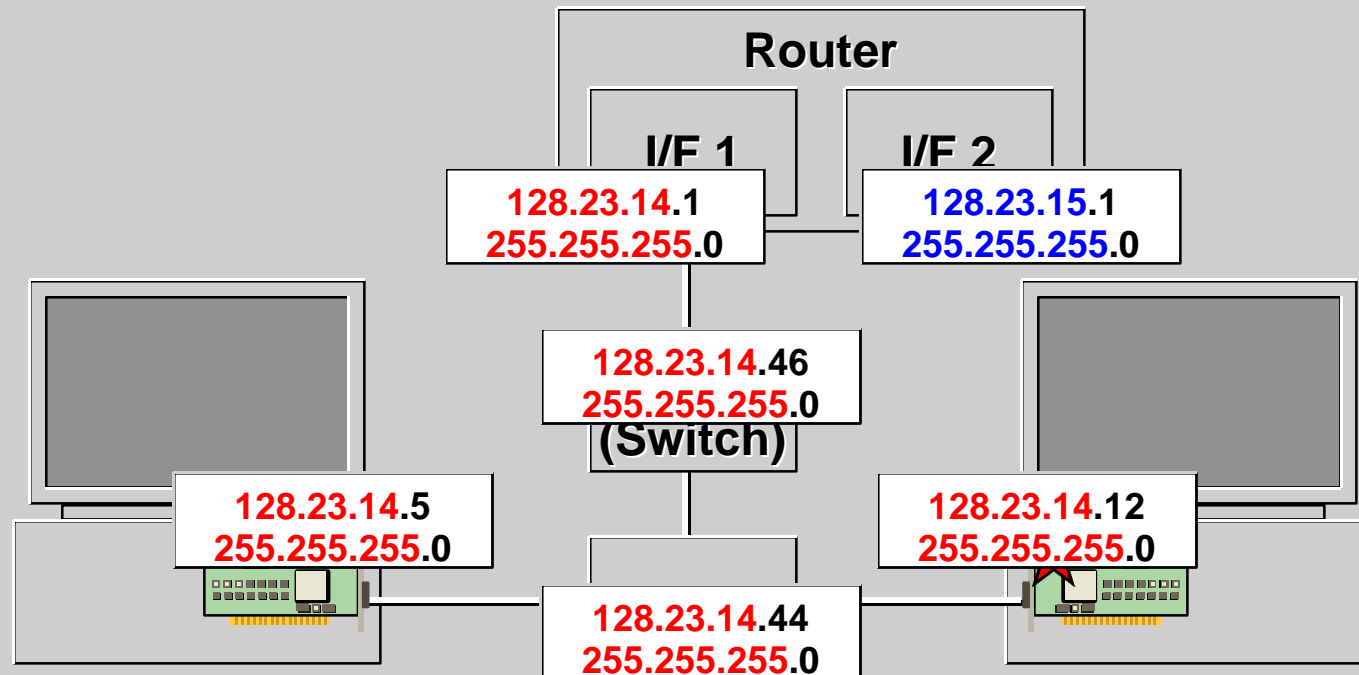
Example: Class B with 8-bit Subnet



Subnetting gets really interesting at the router.

Remember I said that each interface on the router *must* have a different Network ID? Well that rule applies in non-subnetted networks. If we're subnetting at the router...

Example: Class B with 8-bit Subnet



...then the Network IDs of the two interfaces can be the same, but the Subnet IDs must be different.

The logical view of the network is that every device hanging off I/F 1 is part of Subnet 14, and every device hanging off I/F 2 is part of Subnet 15.



The Structure of an IP Address

IP Address Classes

Subnetting an IP Address

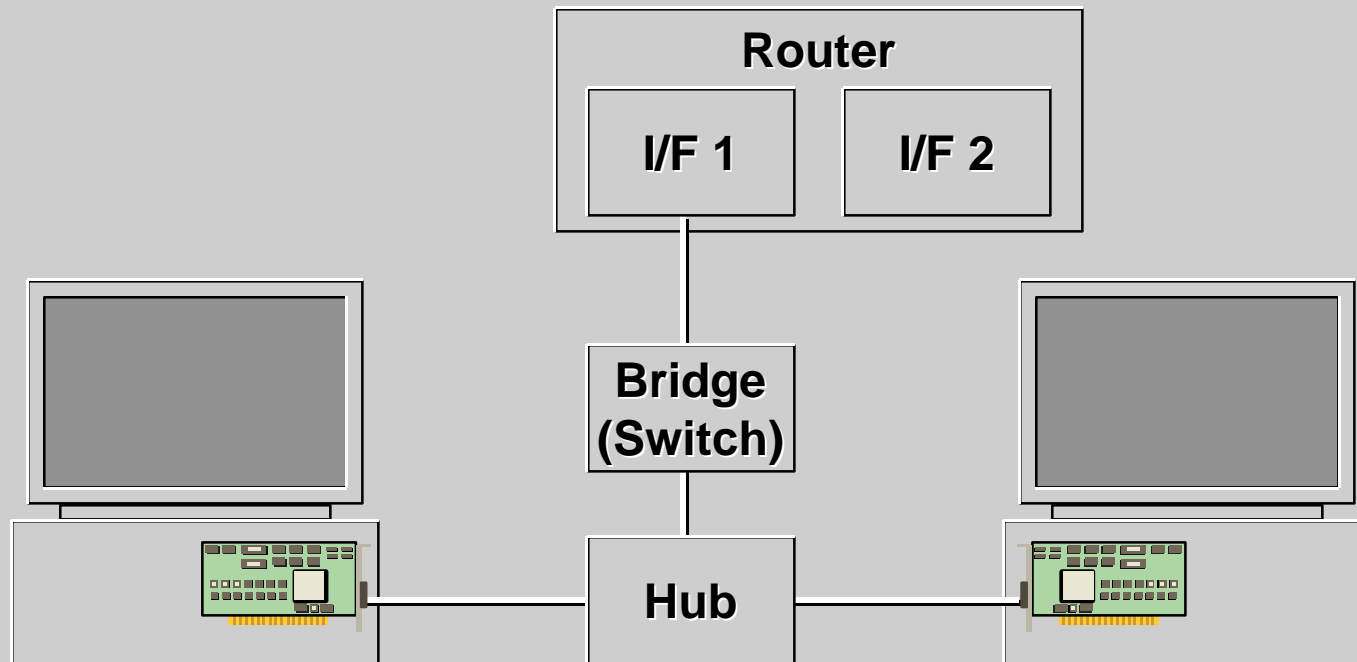
Applying Addresses

Section 4

Reserved and Special Addresses

In this final section I'll look at IP address conventions, and addresses that are reserved for specific purposes.

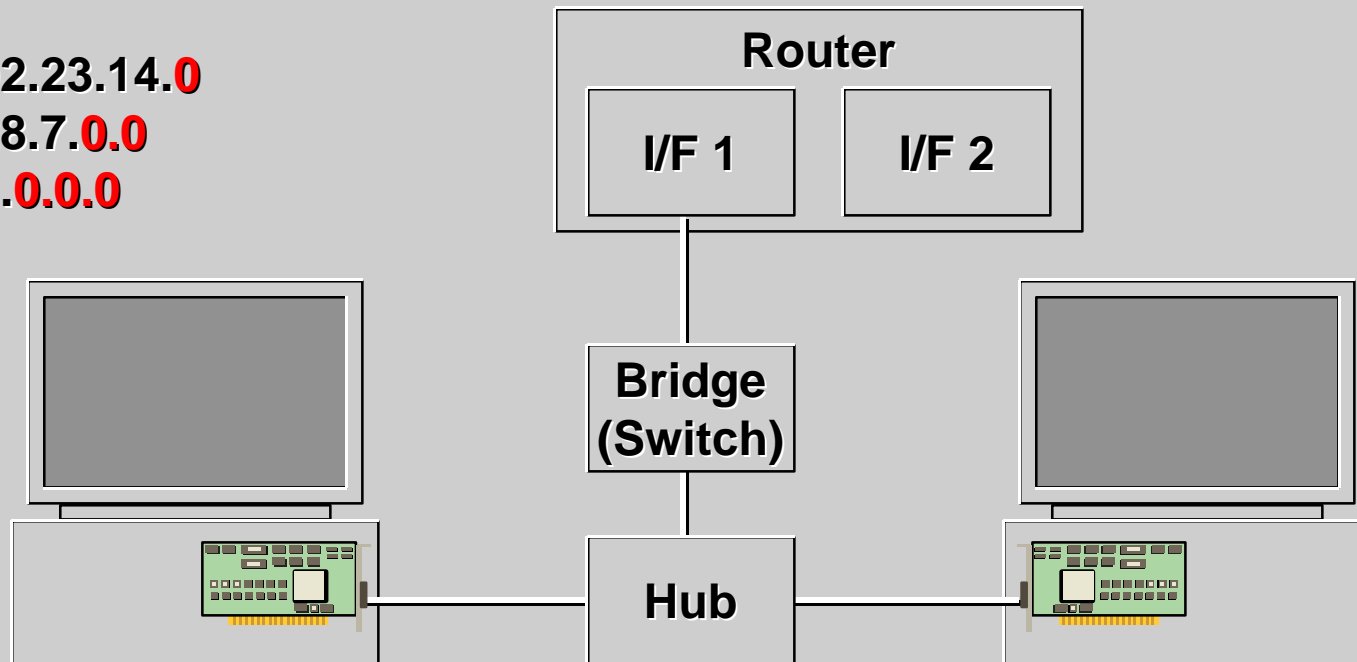
Special Address Conventions



There are a two special address conventions used in IP addresses.

Special Address Conventions

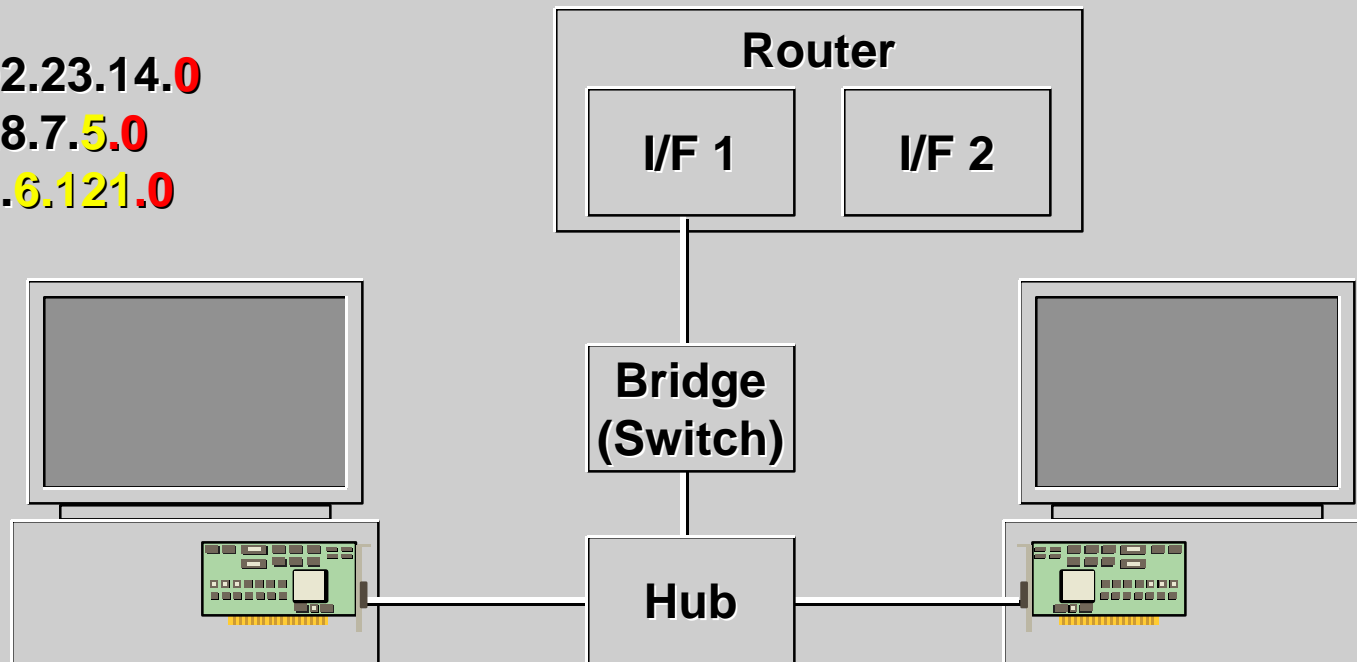
192.23.14.**0**
128.7.**0.0**
21.**0.0.0**



If I use a Host ID of zero (red-highlighted on the diagram for different address classes), then it is *written convention* meaning "any host on this network".

Special Address Conventions

192.23.14.**0**
128.7.**5.0**
21.**6.121.0**

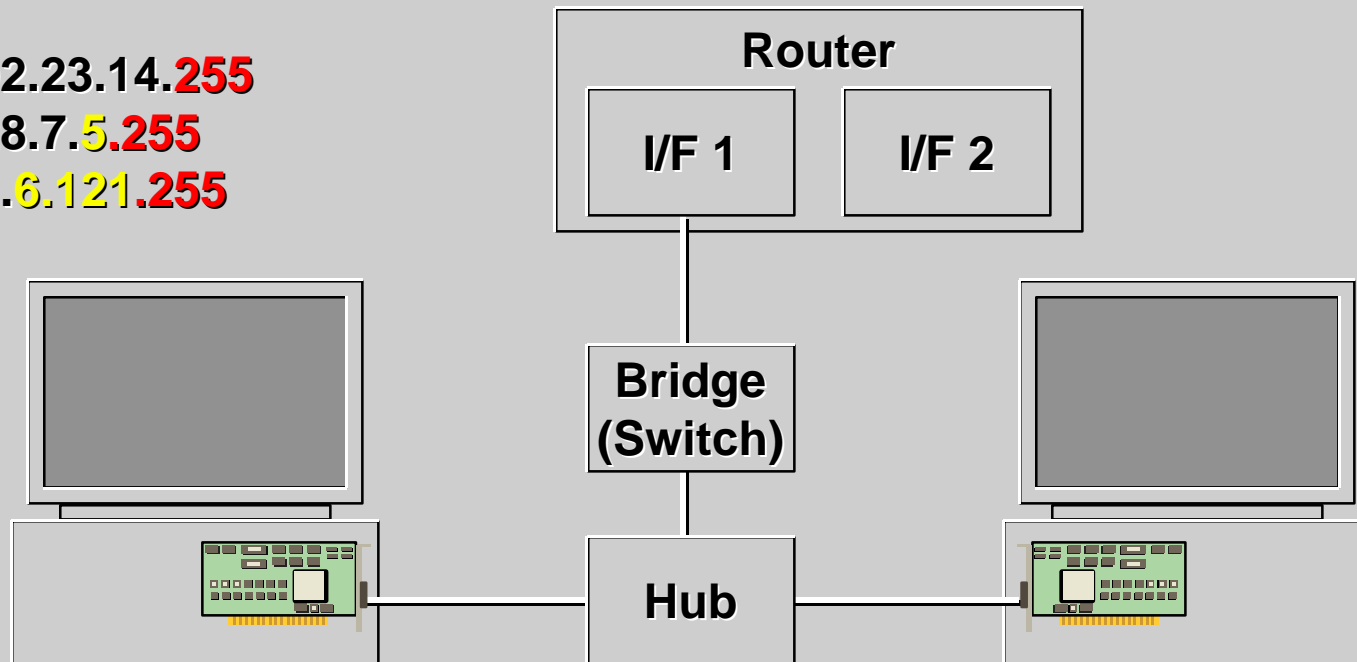


If I used a Host ID of zero (red-highlighted on the diagram for different address classes), then it is *written convention* meaning “any host on this network”.

If I write a zero Host ID inside a subnetted address (subnet space highlighted in yellow), then it is a *written convention* meaning “any host on this subnet”.

Special Address Conventions

192.23.14.**255**
128.7.**5.255**
21.6.121.**255**



All-ones Host IDs represent an all-hosts *broadcast*.

Note the difference between a broadcast "all-ones" and the written convention of "all hosts".

Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**

I mentioned this earlier. When we transmit an IP datagram with this destination address, all IP devices in the network (eg., routers) understand that it is to be looped back to the original source segment.



Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**
- **Network number of all-zero's (e.g., 0.0.0.24)**

An all-zero Network ID is defined as “class-less”, and is used for a host to indicate that it doesn't know the Network ID of the segment it's attached to. Sometimes referred to as “this network”.



Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**
- **Network number of all-one's (e.g., 0.0.0.24)**
- **Network number of all-one's (e.g., 255.255.255.24)**

All-one's Network ID is reserved, and has no current meaning.



Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**
- **Network number of all-one's (e.g., 0.0.0.255)**
- **Network number of all-zero's (e.g., 255.255.255.255)**
- **Host number 0**

A Host ID of 0 is the written convention meaning “any host on this network”. You should not see this address in a real IP packet, just in documentation or configuration files.

Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**
- **Network number of all-one's (e.g., 0.0.0.255)**
- **Network number of all-one's (e.g., 255.255.255.255)**
- **Host number 0**
- **Host number all-one's (e.g., 192.23.14.255)**

All-one's Host IDs represent an all-hosts *broadcast*.

You will see this in real IP packets that are intended to be copied to every host in this Network ID, in this case all the hosts on 192.23.14.0.

Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**
- **Network number of all-one's (e.g., 0.0.0.255)**
- **Network number of all-zero's (e.g., 255.255.255.0)**
- **Host number 0**
- **Host number all-one's (e.g., 192.23.14.255)**
- **Address 0.0.0.0**

Address 0.0.0.0 is reserved to be used to advertise a Default Gateway.

Reserved IP Addresses

- **Network Number 127.0.0.0 reserved for loopback**
- **Network number of all-one's (e.g., 0.0.0.24)**
- **Network number of all-one's (e.g., 255.255.255.24)**
- **Host number 0**
- **Host number all-one's (e.g., 192.23.14.255)**
- **Address 0.0.0.0**
- **Address 255.255.255.255**

Address 255.255.255.255 is known as the "local wire broadcast". It appears as an actual IP Destination Address in a packet that is meant to be copied to all hosts on this LAN segment.

(NOTE: The concept of "all hosts on this LAN segment" is subtly different from "all hosts in this Network ID").



The End

This concludes the tutorial.

If you aren't viewing this on the FORE Systems Web Site, then you can become a member of the ATM Academy free of charge and have access to many more of these tutorials.

You can find details at:

www.fore.com/atm-academy