

Introduction

If you've read through countless papers on TCP/IP and understand most of what you've read and yet subnetting still remains an enigmatic process, then this paper is for you. After reading this article, you should be able to do subnetting mostly in your head. The purpose of this paper is to simplify the subnetting process, nothing more.

Understanding What Subnetting Actually Does

What makes subnetting seem difficult to understand is its not physically tangible; all too often it's presented as a mathematical equation "you just have to learn". This paper is an adjunct of the three part video series explaining the 6 step subnetting process found at www.youtube.com/vegasrage inside are exercises you can work out for yourself allowing you to walk through each step so you can make subnetting easy for yourself. We'll first show how to determine your hosts and networks the easy way which is the core of 6 step process, then we'll do basic subnetting, then Variable Length Subnet Masking (VLSM).

Determining Hosts and Subnetworks

To me it seems the most fundamental concept in teaching subnetting is almost always overcomplicated, and that is people seem to forget they are just counting in binary. Every electronic bit has just two possible values “on” or “off” (numerically “0” or “1”) and counting the “possible” values in binary is just a matter of counting in powers of 2 which is nothing more than the process of doubling each answer by it’s own value 2, 4, 8, 16, 32, 64 etc. If you’ve been racking your brain trying to figure out how to subnet then you have used the 8-bit template in Figure 1 to determine the “actual” binary value is within an octet.

In determining the actual value you only count the bits which are turned on by adding up the turned on place values as shown (Figure 2).

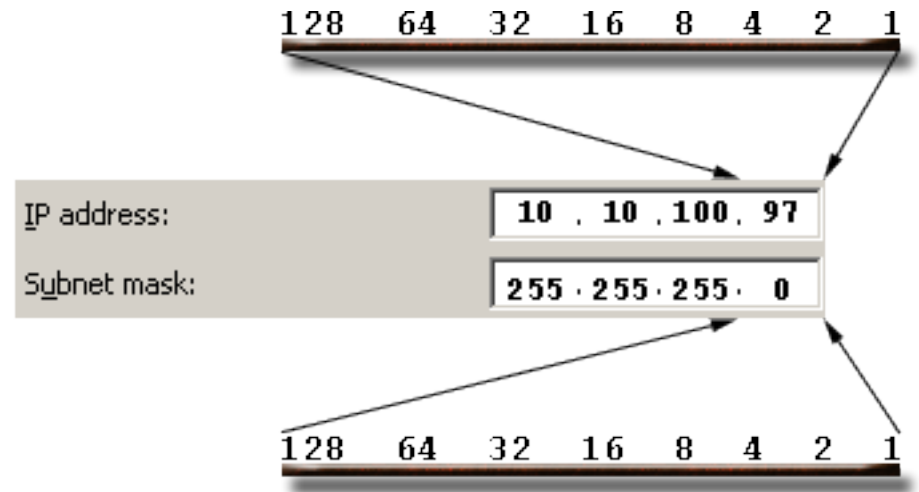


Figure 1

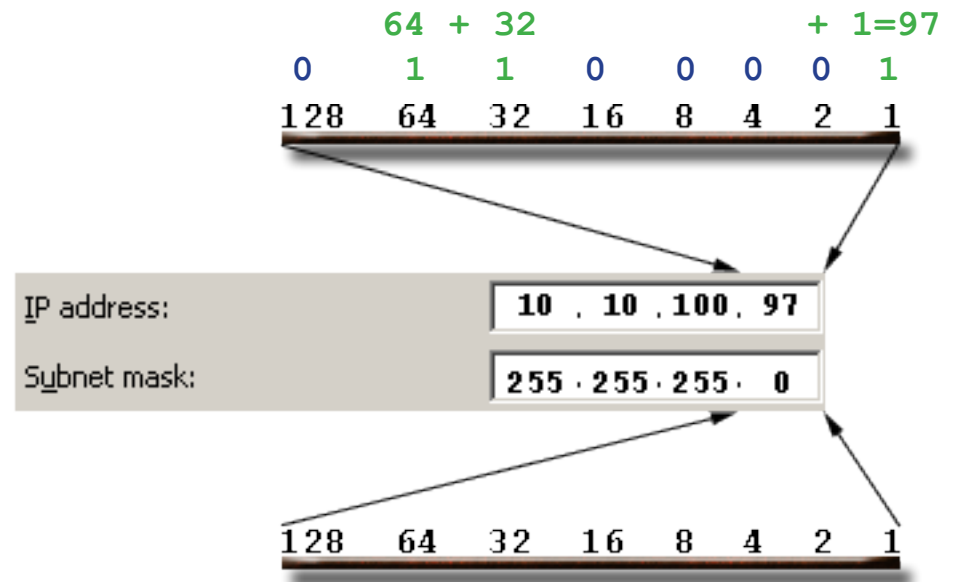


Figure 2

However to just know what the total maximum number of possible values are in a given number of bits is just as simple. For example by now you already know any octet's maximum possible value range is 0 to 255 or put differently can hold 256 possible values if you count 0. We can easily show this by counting in powers of 2 for all 8 bits inside an octet. For example lets say you needed 60 networks you would simply **double over each place holder** until you reached a value that accommodated your requirements. You can easily see (Figure 3) it took 6 bits to get to create 64 subnetworks on the 4th octet. Understanding just that allows you to easily answer 4 of the 6 question subnetting process.

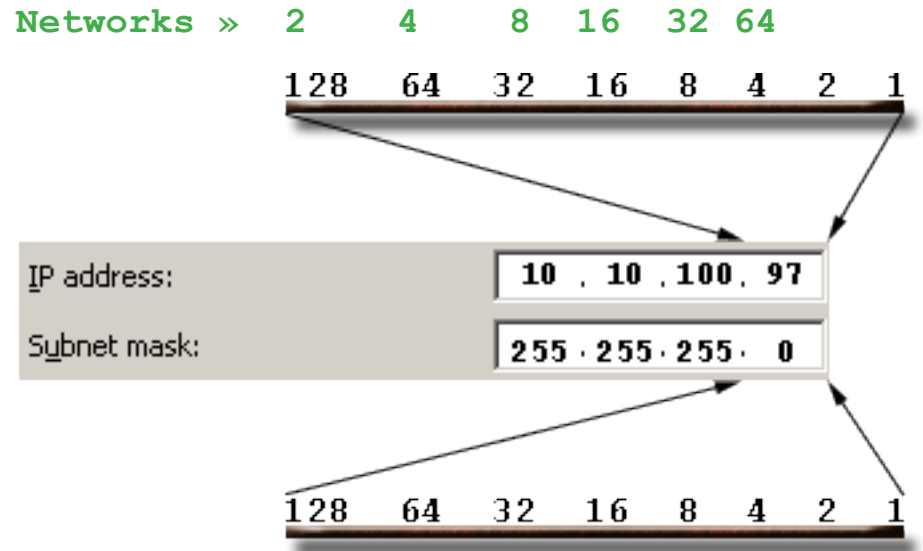


Figure 3

The process can be just as easily applied to hosts by counting right to left as shown in (Figure 4). The only difference is you must subtract 2 from the hosts for each subnet that is where the $2^n - 2$ formula comes into play, more on that later.

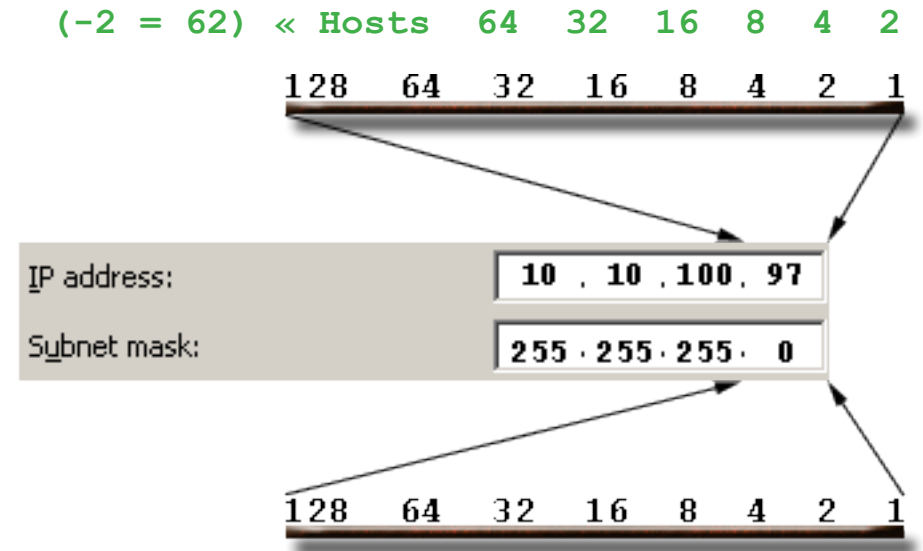


Figure 4

Basic Subnetting

Understanding it's just a matter of counting binary by doubling each resulting value you can now answer the 6 question subnetting process largely in your head by remembering the questions.

- 1) How Many Sub-Networks Do You Need?
- 2) How Many Bits Did You Have To Use?
- 3) What Is Your Subnet Mask?
- 4) What Is Your Block-Size?
- 5) What Are Your Subnets?
- 6) What Are The Number Of Hosts And IP Ranges For Each Subnet?

1) How Many Sub-Networks Do You Need?

We're going to create a minimum of 7 subnets from the fourth octet which means we just answered question 1. You don't need an IP address to do this but people like to see a network address for reference. Looking at the example (Figure 5), to come up with 7 subnets we need to think in binary, so we will apply the 8 bit template to the 4th octet. We needed 7 subnetworks but we count 8 by doubling the possible values.

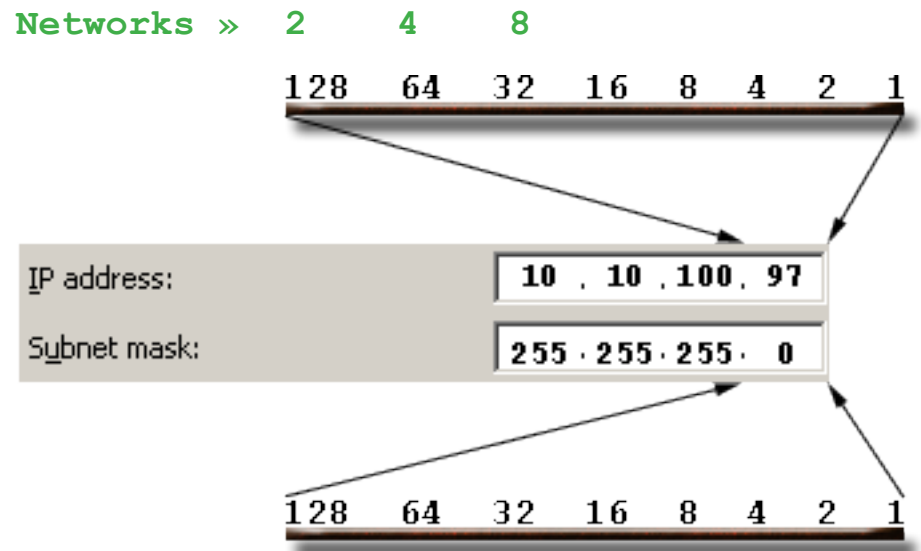


Figure 5

2) How Many Bits Did You Have To Use?

We used 3 bits in our doubling process giving us 8 subnets, on to the next question.

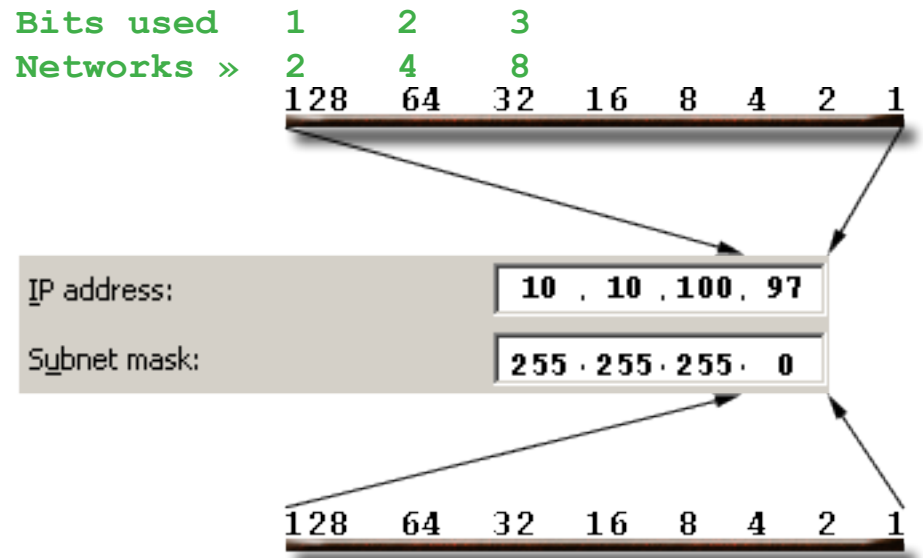


Figure 6

3) What Is Your Subnet Mask?

By adding the place holder values from the answer we arrived at in the previous question we know our mask, $128+64+32=224$.

The answer from question 2 also helps answer question 3.

Since we are dealing with the 4th octet our starting mask was 255.255.255 or known as a /24, we have extended our mask making it a /27, meaning we have committed 27 bits of the possible 32 bits address to network.

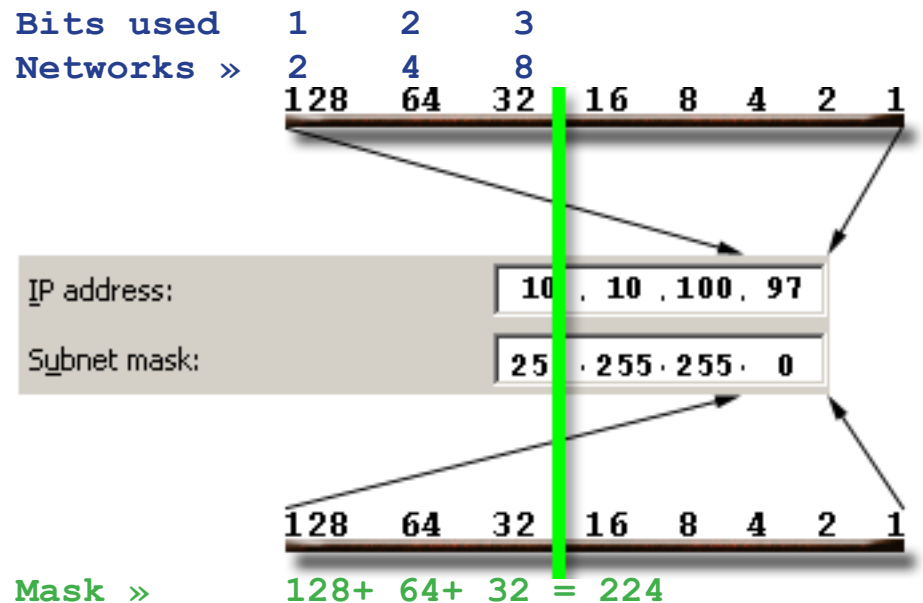


Figure 7

4) What Is Your Block-Size?

The block size is always the lowest place holder number that makes up your mask; again the previous question helps to answer the next. The block size helps define all 8 subnets which leads us to question 5.

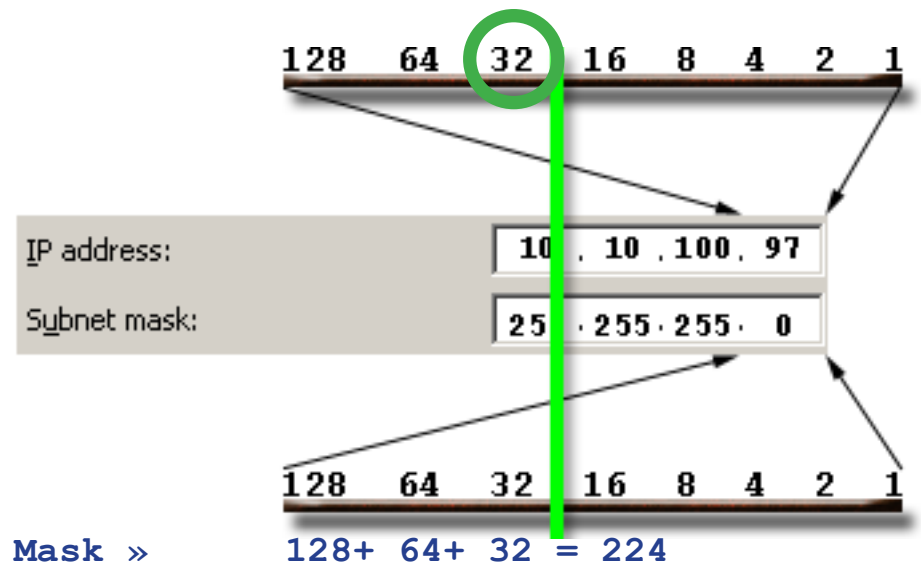


Figure 8

5) What Are Your Subnets?

In step 3 we determined the mask was 224, creating subnets is simply a matter of figuring out how many times the block size will go into the mask of 224 by incrementing the block size until we reach the value of the mask. In this case we just keep adding by 32 until we reach 224 giving us the expected 8 subnets.

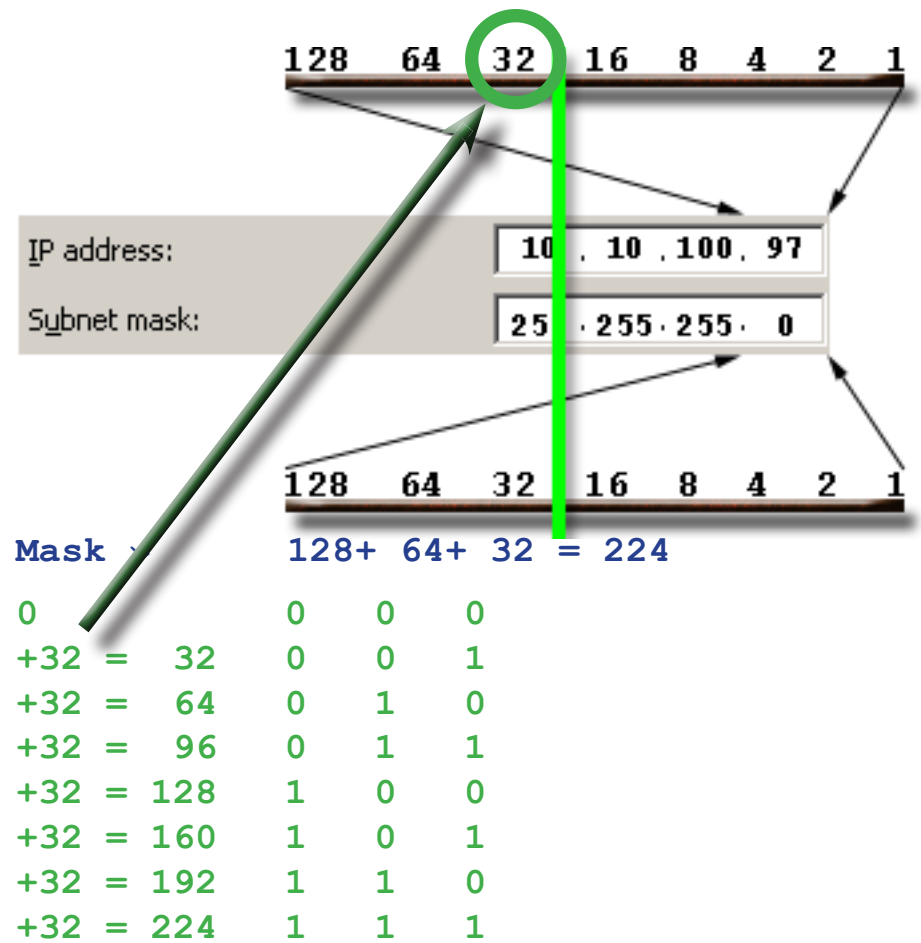
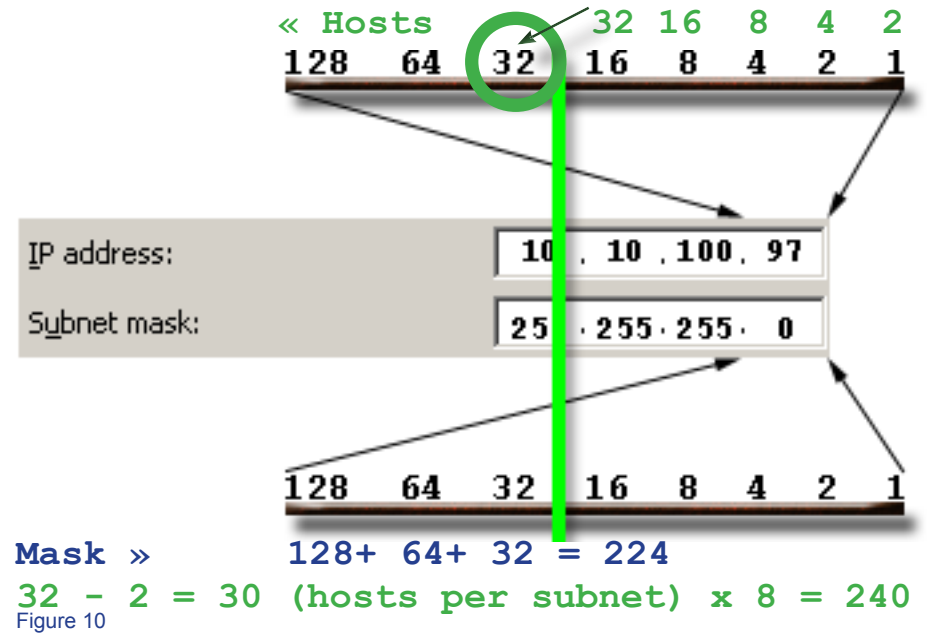


Figure 9

6) What Are The Number Of Hosts And IP Ranges For Each Subnet?

Question 6 is a two part question first determining how many hosts you get per subnetwork, and then detailing the actual host IP ranges. Since we're in the 4th octet we can determine how many hosts each subnet will have by looking at the block size of 32 and subtracting 2 ($2^n - 2$) giving you 30 hosts for each of the 8 subnets for a total of 240 hosts. You can confirm this by counting in binary the total possible values in the host range to the right of the block size as we did in beginning of this paper "Determining Hosts and Subnetworks" and then subtracting 2.



We subtract 2 because you cannot have a host of 0 or use a broadcast as a host. To visualize this, Figure 11 shows in binary the exclusions for the 0 and 32 subnets. In the 0 subnet, 0 equals no host and 31 is the last possible value before you move into the 32 subnet making it a broadcast (basically shouting to everyone) so you move up from 0 to 1 and down from 31 to 30 ($2^n - 2$). The same is true in the 32 subnet, 32 is effectively 0 in that subnet and 63 a broadcast so again you must move from 32 up to 33 and down from 63 to 62. Regardless of the subnet you are working on, your usable range of IP's will follow this process in the 4th octet.

Your usable range of hosts are:

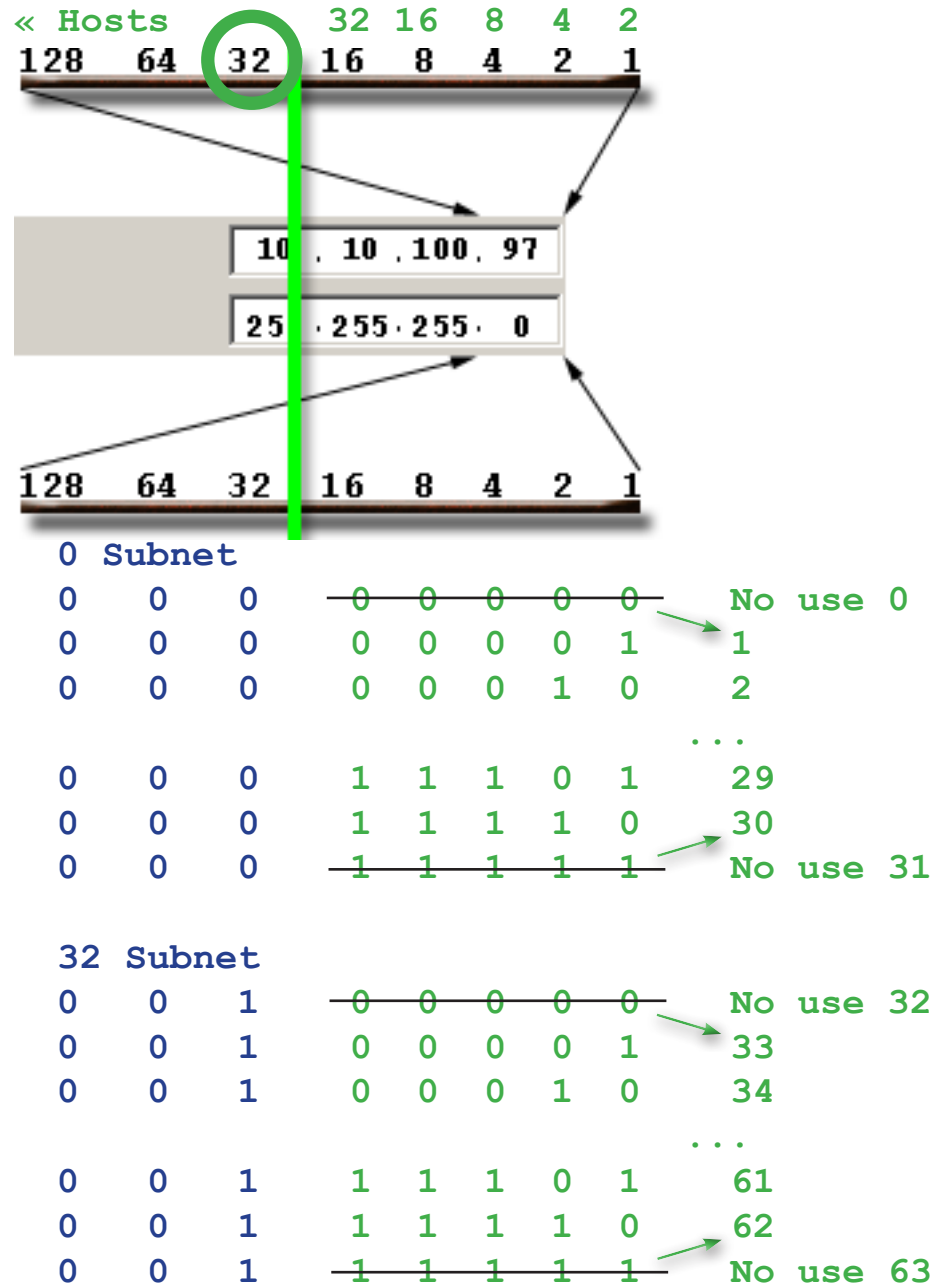
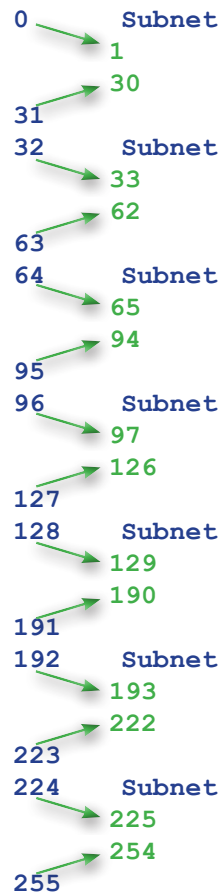


Figure 11

Create a minimum of 20 networks from the 3rd octet

See if you can through the process. Let's say you need 20 additional subnets minimum, you may end up with more but you will discover this by going through each step. Go back through the previous pages to work through each step if necessary. The answers are on the following pages.

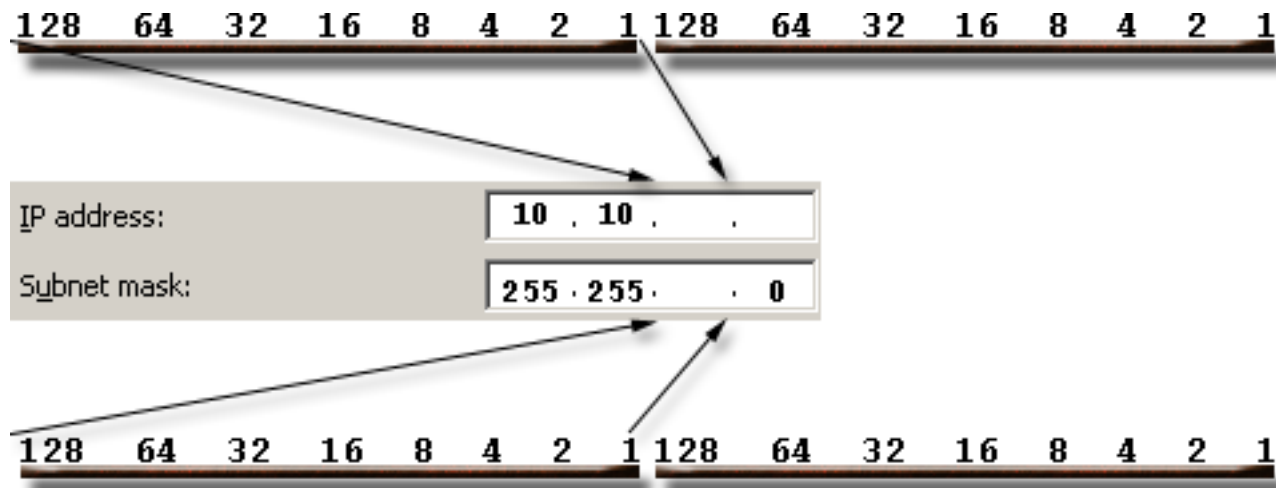


Figure 12

- 1) How Many Sub-Networks Do You Need?
- 2) How Many Bits Did You Have To Use?
- 3) What Is Your Subnet Mask?
- 4) What Is Your Block-Size?
- 5) What Are Your Subnets?
- 6) What Are The Number Of Hosts And IP Ranges For Each Subnet?

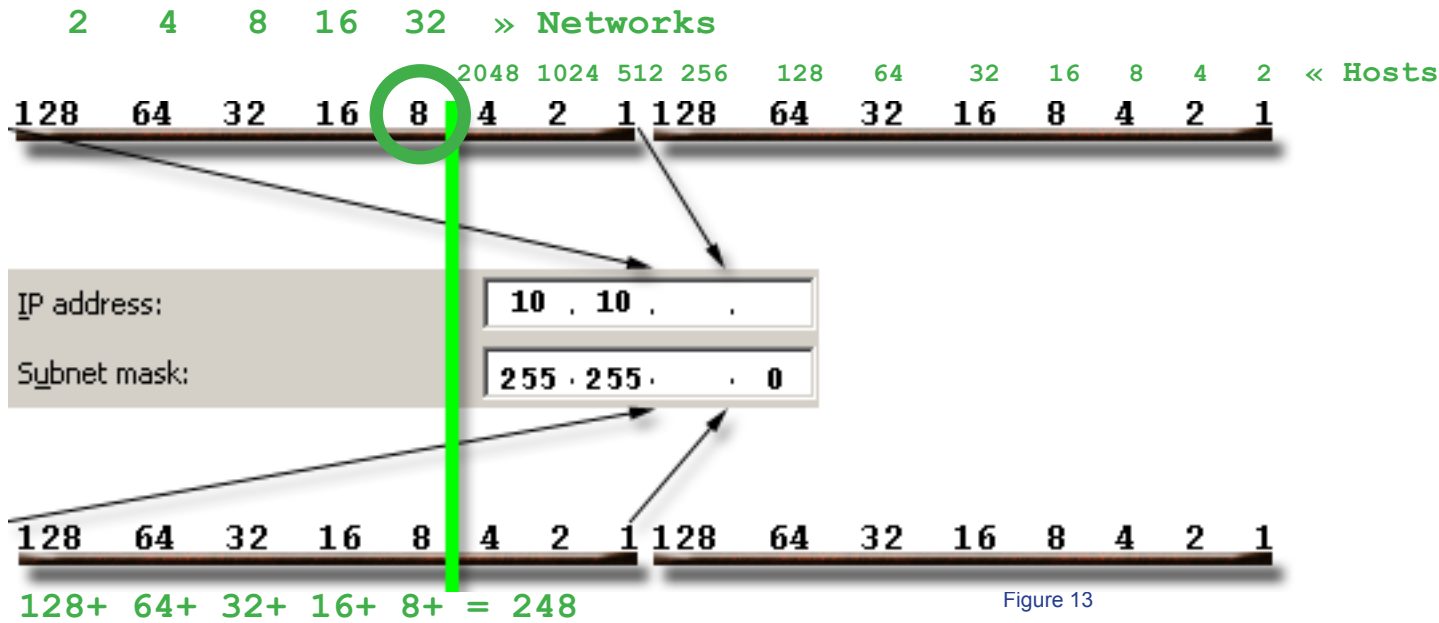


Figure 13

1) How Many Sub-Networks Do You Need?

2, 4, 8, 16, 32
 You needed 20 but got 32

2) How Many Bits Did You Have To Use?

5
 You doubled 5 times

3) What Is Your Subnet Mask?

248 (Remember 8 is always 248)
 128+64+32+16+8=248
 (Aka /21 network - 21 bits used on the network side of the subnet mask)

4) What Is Your Block-Size?

8
 Always the lowest number in the mask

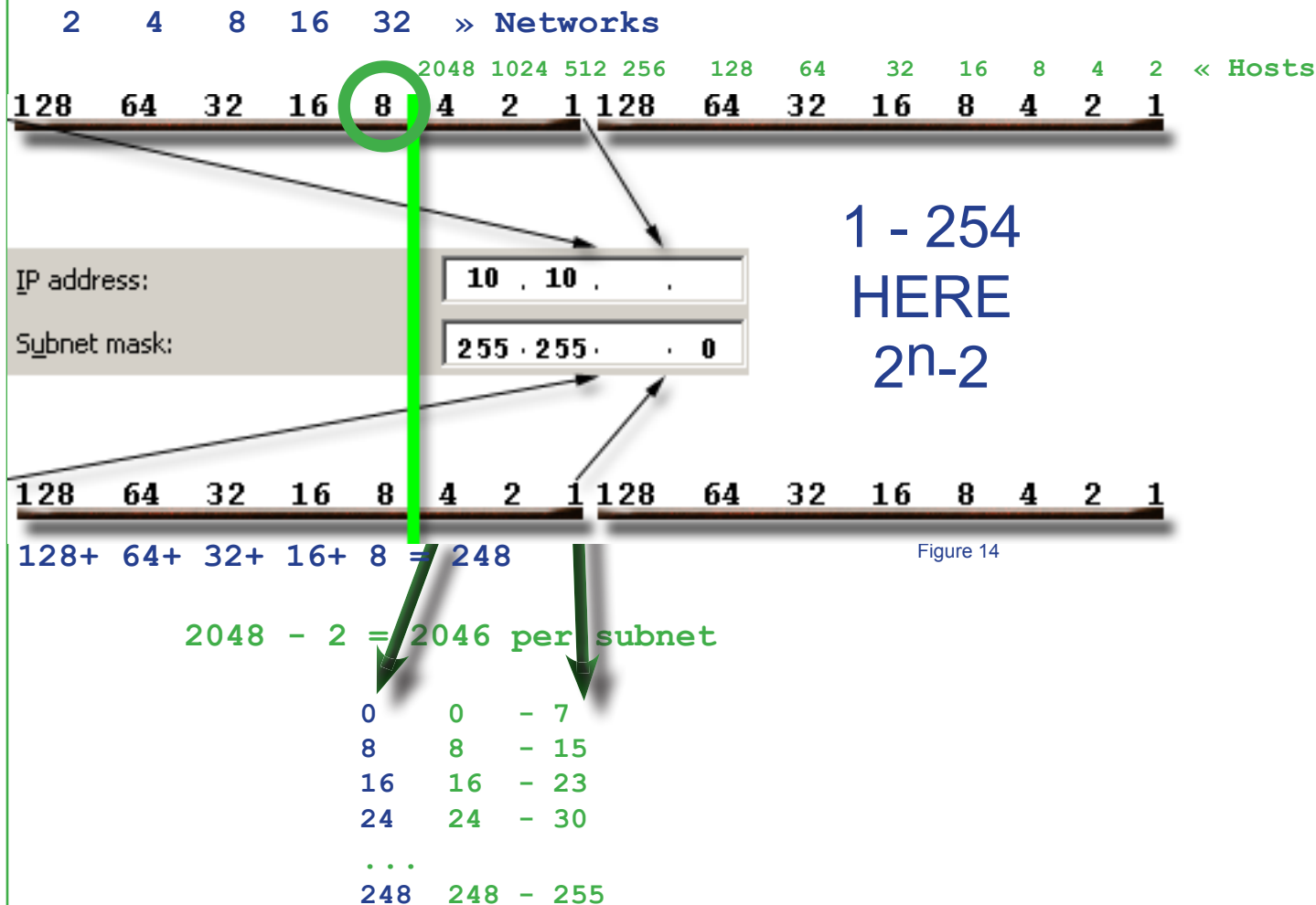
5) What Are Your Subnets?

- 0
- 8
- 16
- 24
- ...
- 248

If you feel like doing all 32 you can, I'll do the first 4 here.

6) What Are The Number Of Hosts And IP Ranges For Each Subnet?

Intentionally I had you subnet the 3rd octet to make the point again that the $2^n - 2$ formula only applies to the forth octet, really just farthest right bit in the address where the number 1 is counted. The 4th octet will be 1 through 254, $2^n - 2$ will take care of itself in the address ranges, you just need to remember it when determining the number of hosts. As for the actual ranges you can use all the numbers in the third octet and not worry about subtracting 2.



Variable Length Subnet Masking (VLSM) Subnetting

Create a minimum of 3 networks from a /21 classless network

Having finished subnetting the /21 network you are ready to subnet it again and create even more subnets using the same 6 step process as before. I hear a maniacal laugh in the distance. The only difference between basic subnetting and Variable Length Subnet Masking (VLSM) is the starting place you begin creating subnets (in this case your 248 mask is where you will start). You now have a classless /21 network that you are going to further segment into smaller networks. The previous network you created is now written in stone and you must forge ahead to the right and encroach into host territory (pillaging as you go, of course).

1) How Many Sub-Networks Do You Need?

2, 4

You needed 3 but got 4

2) How Many Bits Did You Have To Use?

2

You doubled 2 times

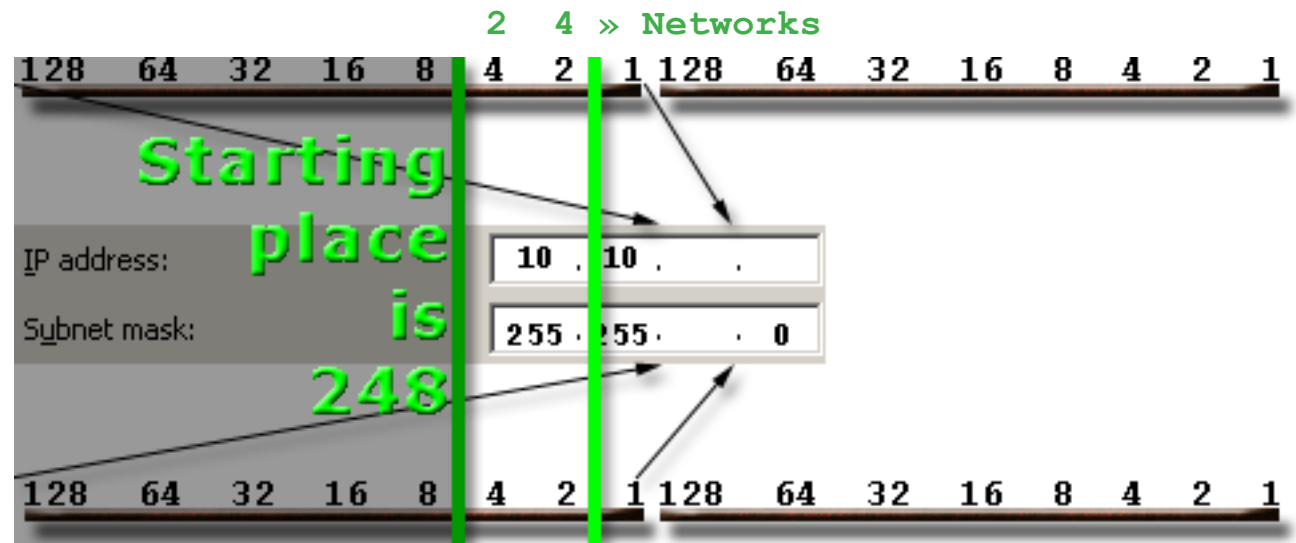


Figure 15

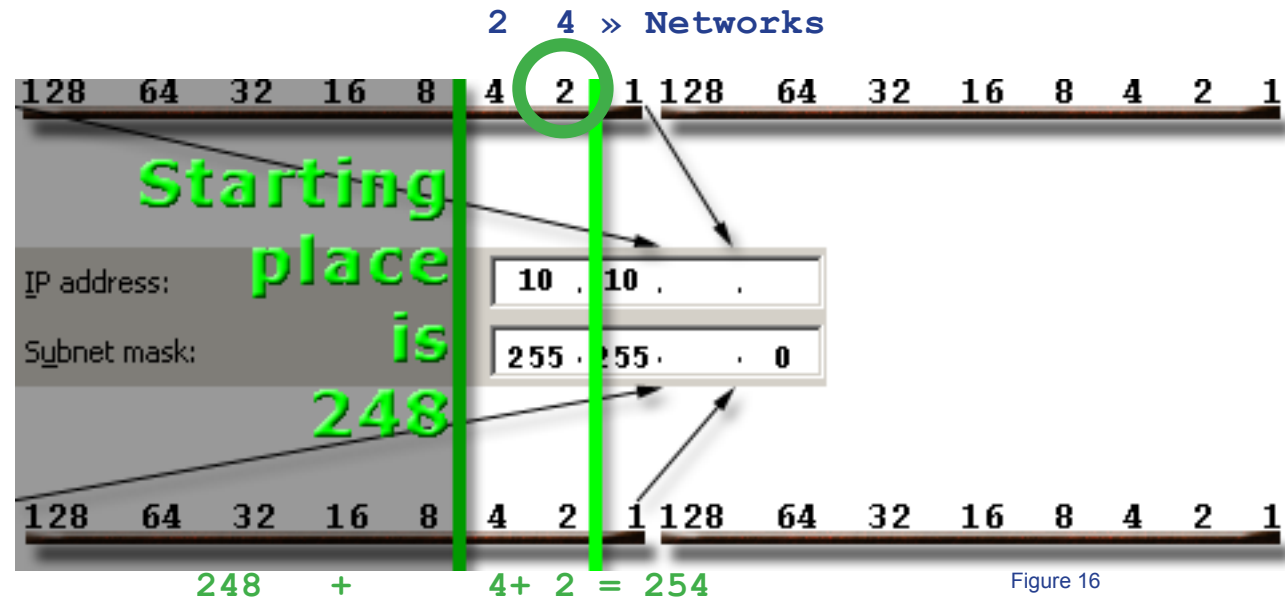


Figure 16

3) What Is Your Subnet Mask?

Since we left off with a 248 mask we simply continue extending our mask from 248 by adding the place values 4 + 2 on to the mask and now we have a mask of 254. If by chance you needed 8 subnets then you would extend the mask one more bit and your mask would become 255, you would still have 8 subnets. To the unsuspecting it would appear to be a /24 class C address but in reality it is not.

Start with 248

$248 + 4 + 2 = 254$

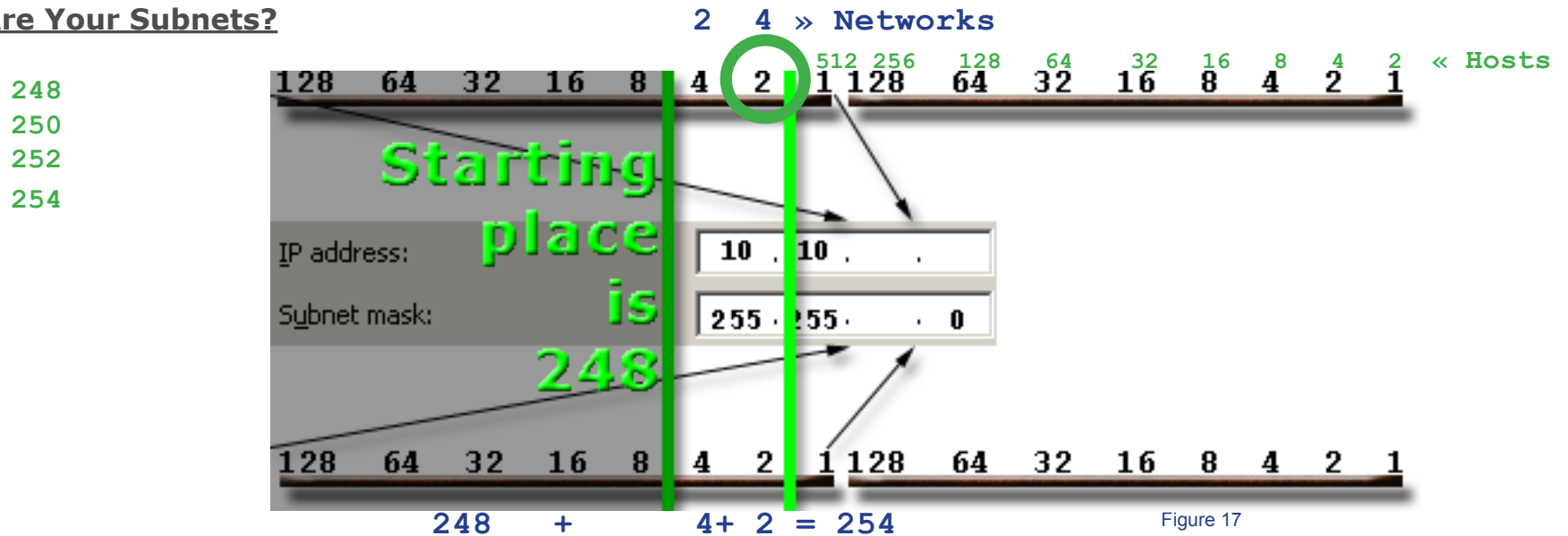
(Aka /23 network - 23 bits used on the network side of the subnet mask)

4) What Is Your Block-Size?

2

Always the lowest number in the mask

5) What Are Your Subnets?



| | | | | |
|------|-----|---|---|------------------------|
| MASK | 248 | 0 | 0 | (Effectively 0 subnet) |
| +2 = | 250 | 0 | 1 | |
| +2 = | 252 | 1 | 0 | |
| +2 = | 254 | 1 | 1 | |

6) What Are The Number Of Hosts And IP Ranges For Each Subnet?

Counting the number of hosts right to left you can see you have 512 possible values per subnet, subtract 2 from each subnet and you have 510 hosts for each subnet. As for the host ranges you only have 1 bit left in the 3rd octet capable of holding 2 possible values 0 and 1 (ON or OFF), each subnet can use either 0 or 1, the 4th octet however will have values of 1 – 254 just as it did in the previous example. -- That's it.

| | | | |
|-----|-----|---|-----|
| 248 | 248 | - | 249 |
| 250 | 250 | - | 251 |
| 252 | 252 | - | 253 |
| 254 | 254 | - | 255 |