IPv6 Subnet Masking

Does IPv4 subnetting apply to IPv6? Yes, and it still uses binary values of 1 to define a mask, but there are two substantial differences due to the difference between IPv4 and IPv6 notation:

1. How the mask looks. (notation)
2. What gets masked. (which bits)

How an IPv6 Mask Looks (Notation)

An IPv6 subnet mask is written in hexadecimal, but let's start by explaining that IPv6 uses 128 binary digits for each IP address, as opposed to IPv4's 32 binary digits, and those 128 binary digits are divided into 16-bit words, like this:

0000000000000000.0000000000000000.0000000000000000.0000000000000000.0000000000000000.0000000000000000.0000000000000000.0000000000000000

It would be a little difficult to use IPv4's old octet notation for 128 bits, which might look like this:
182.222.101.003.255.074.112.200.000.010.135.016.208.192.136

So, a hexadecimal representation is used instead, which makes a little bit easier. Hexadecimal is a 16-digit numbering system, as opposed to binary's 2-digit system and decimal's 10-digit system. The 16 digits of hexadecimal run from zero to nine, then use the letters A to F: {0123456789ABCDEF}.

One 4-digit hex word represents 16 binary digits, like this:
Bin 0000000000000000 = Hex 0000 (or just 0)
Bin 1111111111111111 = Hex FFFF
Bin 1101010011011011 = Hex D4DB

So, this 128-bit binary address...
1111111111111111.1111111111111111.1111111111111111.1111111111111111.1111111111111111.1111111111111111.1111111111111111.1111111111111111

...would be represented by 8 hex words, separated by colons:
FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF

A full IPv6 subnet mask uses the same 8-hex-word format as an IPv6 address, although some tools allow you to specify only 1 hex word. We'll explain that option more in the next section...

What Gets Masked (Which Bits?)

Like IPv4, an IPv6 address has a network portion and a device portion. Unlike IPv4, an IPv6 address has a dedicated subnetting portion. Here's how the ranges are divided in IPv6:

Network Address Range
In IPv6, the first 48 bits are for Internet routing.
1111111111111111.1111111111111111.1111111111111111.0000000000000000.0000000000000000.00000000000
00000.0000000000000000.0000000000000000

Subnetting Range
The 16 bits from the 49th to the 54th are for defining subnets.
0000000000000000.0000000000000000.0000000000000000.1111111111111111.0000000000000000.00000000000
00000.0000000000000000.0000000000000000

Device (Interface) Range:
The last 64 bits are for device (interface) ID's.
0000000000000000.0000000000000000.0000000000000000.0000000000000000.1111111111111111.11111111111
11111.1111111111111111.1111111111111111

Let's say you want to break your corporate network into 64 subnets. The binary mask just for the subnetting range would be 1111110000000000 which translates to a hex value of FC00. Some IPv6 masking tools will work with just this one hex word, otherwise a full 128-bit hex mask would be FFFF:FFFF:FFFF:FC00:0:0:0:0.

For more info on basic masking, see definition of subnet masking.

If you play around with converting values in the Windows Calculator (in scientific mode), remember to convert between binary and hexadecimal, not decimal and hex.

Before you ask, yes, it is possible to use bits in the device range for additional subnet masking, but you shouldn't need it. The 16 binary digits dedicated to subnetting and 64 binary digits available for devices give 65,535 subnets with over 18 quintillion devices per subnet. In addition, if you use some of the 64 bits in the device range for subnetting, then you can't use autoconfiguration tools because they expect all of the 64 bits on the right side to be dedicated to devices. So don't use any of the device bits for subnetting if you need IPv6 Autoconfiguration and if you don't know whether or not you need autoconfiguration, assume you do. And even if you know you don't need autoconfiguration, it's a good standard to use a 64-64 split for network/lan vs. device.