Liczba: 0|01111011|10011001100110011001101 = 0.1

Używamy typu float (4 bajty czyli 32 bity), więc przeznaczamy

1 bit na znak: $S = 0_2$ (jest to logiczne ponieważ 0.1 ma znak dodatni)

8 bitwów na cechę: $C = 01111011_2$

23 bity na mantysę: $F = 10011001100110011001101_2$

Oraz używamy stałej B = 127 ze względu na zakres zapisywanych na 4 bajtach liczb.

Wartości przeliczamy na dziesiętne:

$S = 0_2 = 0_{10}$

$C = 123_{10}$

$F = 2^{-1} + 2^{-4} + 2^{-5} + 2^{-8} + 2^{-9} + 2^{-12} + 2^{-13} + 2^{-16} + 2^{-17} + 2^{-20} + 2^{-21} + 2^{-23} = 0.5033165$

$(-1)^S * (1+F) * 2^{C-B} = (-1)^0 * (1 + 0.5033165) * 2^{123-127} = 1.5033165 * 2^{-4} = 0.09395728125$

Ponieważ liczba 0.1 nie ma dokładnej reprezentacji w postaci skończonej ilości bitów w systemie komputerowym musimy użyć przybliżenia. Liczba 0.09395728125 jest właśnie przybliżeniem komputerowym naszego 0.1 i właśnie dlatego zostało tak zapisane w pamięci komputera.

Największą potęgą dwójki nie większą od 0.1 jest $0.0625 = 2^{-4}$

Stąd:

$1.6 = (-1)^0 * (1.6) * 2^{-4} = (-1)^0 * (1 + 0.6) * 2^{123-127}$

S = 0 (Zgadza się)

C = 123 (Zgadza się)

0.6 binarnie zapisujemy jako: 0.(1001). Ponieważ ten zapis jest nieskończony będziemy zmuszeni zaokrąglać.

F = 10011001100110011001101 (Jest liczbą podobną do wcześniejszej mantysy, lecz przesuniętą o jeden bit)

$(-1)^0 * (1 + 0.10011001100110011001101) * 2^{01111011-127}$

Zapis tej liczby binarnie to:

0 01111011 10011001100110011001101

Ostatnim bitem jest jeden ponieważ w zapisie (1001) w okresie kolejną liczbą jest jeden i dlatego zaokrąglamy ostatnią liczbe w góre do 1