Binair

D. Leeuw

4 december 2023
v.0.8.0

Over dit Document

0.1 Leerdoelen

Dit document leert je omgaan met binaire getallen. We behandelen wat binair is en hoe je ermee om gaat. Voor het rekenen met binair beperken we ons tot het optellen, aftrekken en vermenigvuldigen van binaire getallen.

0.2 Voorkennis

Voor een goed begrip van dit document is geen voorkennis vereist.

Inhoudsopgave

Hoofdstuk 1

Binaire getallen

Binair is een tweetalig stelsel, vandaar de bi in binair. We tellen dus met alleen de getallen 0 tot 1, dit in tegenstelling tot het decimale stelstel (deca is 10), waarbij we tellen van 0 tot 9.

Laten we bij het vertrouwde decimale stelsel beginnen. We tellen van 0 tot 9 en als er dan nog 1 bij komt (+1) dan zetten we een 1 voor de 0 (10) om aan te geven dat we eenkeer tot 9 hebben geteld. Het is dus tien. Tellen we nu weer door tot 9 (19) en tellen we er dan 1 bij op, dan wordt de voorste 1 een 2 (20) om aan te geven dat we voor de tweede keer tot 9 hebben geteld.

Dit zelfde principe kunnen we ook gebruiken voor binair, alleen tellen we van 0 tot 1 en daarna komt er een 1 voor, dan wordt het dus 10 en daarna 11 waarna we er weer een 1 voor moeten zetten, dus wordt het 100 en zo verder. De getallen 10 en 100 spreken we ook niet uit als tien en honderd, maar als één nul en één nul nul.

1.1 Binair naar decimaal

Stel we hebben een binair getal 11001110 en we willen weten wat het getal decimaal is. Om binaire getallen om te zetten naar decimaal is er een simpele reken opdracht met machten. Het binaire stelsel kent twee getallen, elke positie in het getal kunnen we weergeven met 2 tot de macht. De macht is dan de waarde van de positie van het bit gerekend vanaf rechts. Om dit wat duidelijker te maken kijken we eerst naar het decimale stelsel, daar ben je wat beter mee vertrouwd.

Het getal honderd is eigenlijk 10^2 want $10 * 10 = 100$ zo is 10 eigenlijk 10^1 en de getallen onder de 10 zijn eigenlijk 10^0 want $10^0 = 1$.

Een getal als 145 kunnen we dus schrijven als $1 * 10^2 + 4 * 10^1 + 5 * 10^0$. Deze logica kunnen we ook gebruiken bij het omzetten van binair naar decimaal.

In een binair systeem werken we met een tweetallig stelsel, dus met 2 tot de macht. Ons getal 11001110 kunnen we dus schrijven als: $1*2^7+1*2^6+0*2^5+0*2^4+1*2^3+1*2^2+1*2^1+0*2^0$ Omdat vermenigvuldigen met 0 0 is en vermenigvuldigen met 1 hetzelfde getal oplevert kunnen we het geheel dus nog veel simpeler maken: $2^7+2^6+2^3+2^2+2^1$. Dit sommetje wordt dus uiteindelijk een simpele optelling: $128+64+8+4+2=206$ En daarmee hebben we binair 11001110 naar decimaal 206.

1.2 Decimaal naar binair

Om een decimaal getal om te zetten naar decimaal moeten we van een 10-talig stelsel naar een 2-talig stelsel. Het makkelijkst kunnen we dat doen door te delen door 2. We delen het decimale getal dus steeds door 2 tot we op 0 zijn uitgekomen. Elke keer dat het lukt om door 2 te delen (zonder een breuk te maken) noteren we een 0, als het niet trekken we 1 van het (oneven) getal af en noteren deze 1. Daarna kunnen we het nieuwe evengetal weer delen. De genoteerde nullen en enen levert ons het binaire getal op dat we zoeken. Hieronder volgt een voorbeeld hoe we 1000 decimaal omzetten naar binair:

Bit 1:	1000 / 2	500	0
Bit 2:	500 / 2	250	0
Bit 3:	250 / 2	125	0
Bit 4:	125 / 2	62	1
Bit 5:	62 / 2	31	0
Bit 6:	31 / 2	30	1
Bit 7:	15 / 2	14	1
Bit 8:	7 / 2	6	1
Bit 9:	3 / 2	2	1
Bit 10:	1 / 2	0	1

Ons binaire getal noteren we van onder naar boven(!) dus als 1111101000.

1.3 Bits en bytes

In de computertechniek spreken we bij binair van een 1 of een 0 als een bit. Zetten we 4 bits als achter elkaar dan hebben we een nibble en 8 bits zijn een byte. Het zijn allemaal woordspelingen op eten. Een bit kan vertaald worden als een stukje of een hapje. To nibble is knabbelen en to bite is bijten/afbijten. Omdat bite te veel leek op bit werd het byte. Het woord nibble voor 4-bits komt niet zo veel meer voor.

Moderne computers werken met 64 bits, wat dus 8 bytes is.

In afkortingen is afgesproken dat de bit een b is en de byte een B, dus de grootste hoeveelheid is de grootste letter.

1.4 Duizenden of miljoenen

Volgens de standaard is een hoeveelheid van 1000 een kilo en van een miljoen een mega. Denk hierbij aan bijvoorbeeld een kilogram. Analoog daaraan is een kb 1000 bits en is kB een 1000 bytes.

Een duizendtal binair is eigenlijk 1024. Voorheen leverde dit verwarringen op. Een hardeschijf van 1024 bits is wat anders dan een hardeschijf van 1000 bits. Om dit onderscheid duidelijk te maken is er voor het binaire 1024 de benaming Kibi bedacht inplaats van kilo. Afgekort wordt het KiB voor 1024 bytes.

Kilo	k	Kibi	Ki
Mega	M	Mebi	Mi
Giga	G	Gibi	Gi
Peta	P	Pebi	Pi
Exa	E	Exbi	Ei
Zetta	Z	Zebi	Zi

Hoofdstuk 2

Binair rekenen

2.1 Optellen

Als we twee getallen willen optellen zullen we dus steeds rekening moeten houden dat we heel snel door moeten schuiven met een 1 ervoor. Als we simpel beginnen dan komen we eerst tegen het optellen van twee nullen:

$$0 + 0 = 0$$

Dat is niet anders dan in het decimale stelsel. Ook het optellen van 0 en 1 is niet anders:

$$0 + 1 = 1$$

$$1 + 0 = 1$$

Voor het optellen van twee enen verandert de zaak wel:

$$1 + 1 = 10$$

Nu we dit weten kijk dan eens of de volgende optelling klopt:

$$\begin{array}{rl} 11001110 & \\ 01000101 & + \\ \hline 100010011 & \end{array}$$

2.2 Aftrekken

Als we twee getallen van elkaar willen aftrekken dan is ook weer het aftrekken van twee nullen heel simpel:

$$0 - 0 = 0$$

Voor het aftrekken van een nul van één is het ook nog eenvoudig:

$$1 - 0 = 1$$

Het wordt moeilijk bij het aftrekken van één van nul. We zouden dan uit moeten komen bij -1, maar dat kan in binair niet. Het binaire stelsel kent geen getallen onder nul. We kunnen dus alleen kleine getallen van grote getallen aftrekken zolang de uitkomst niet lager wordt dan 0. We kunnen dus wel 1 aftrekken van 10:

$$10 - 01 = 01$$

Voor het aftrekken van twee enen is de zaak weer simpel:

$$1 - 1 = 0$$

Nu we dit weten kijk dan eens of de volgende aftrekking klopt:

$$\begin{array}{rl}
11001110 & \\
01000101 & \text{-} \\
\hline
10001001 &
\end{array}$$

2.3 Vermenigvuldigen

Het vermenigvuldigen van binaire getallen is niets relatief simpel, vandaar dat computers ook zulke goede rekenaars zijn. Vermenigvuldigen is schuiven en optellen. Het is waarschijnlijk het makkelijkst met een voorbeeld:

$$\begin{array}{rl}
11001110 & \\
01000101 & \text{x} \\
\hline
11001110 & \\
000000000 & \\
1100111000 & \\
00000000000 & \\
000000000000 & \\
0000000000000 & \\
11001110000000 & + \\
\hline
11011110000110 &
\end{array}$$

Voor elk bit uit het getal waarmee we vermenigvuldigen schuift het getal dat vermenigvuldigt wordt op naar links en krijgt er een 0 achter. Heeft het getal waarmee we vermenigvuldigen een 1 staan dan kom het opgeschoven getal er te staan, staat er een 0 van is de opteller allemaal nullen, kortom er gebeurt niets.