

2 Elementaire bewerkingen

1 BINAIRE GETALLEN

In het vorige hoofdstuk heb je gezien dat rijen bits worden gebruikt om lettertekens, getallen, kleuren, geluid en video voor te stellen. Met getallen moet de computer kunnen rekenen. In dit hoofdstuk leer je hoe te werken met binaire getallen en hoe hexadecimale getallen omgezet worden naar een decimaal getal en omgekeerd.

Vervolgens worden de logische operaties behandeld en leer je daarmee eenvoudige schakelingen te maken.

In het binair talstelsel is net als in het decimale talstelsel de positie van het cijfer bepalend voor de waarde van het cijfer in het getal. Het verschil tussen beide talstelsels is dat het binaire talstelsel een getal wordt voorgesteld door een rijtje van twee verschillende symbolen, 0 en 1, terwijl in het decimale talstelsel 10 symbolen gebruikt worden: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

1.1 VAN BINAIRE NAAR DECIMAAL

In het decimale stelsel is de positie van het cijfer bepalend voor de waarde. Het decimale stelsel is gebaseerd op machten van 10. In het getal 514 heeft het cijfer 4 de waarde 4, het cijfer 1 de waarde 10 en het cijfer 5 de waarde 500. Anders gezegd:

$$\begin{array}{r} 4 \times 10^0 = 4 \\ 1 \times 10^1 = 10 \\ 5 \times 10^2 = 500 \\ \hline 514 \end{array}$$

Het binair stelsel is gebaseerd op machten van 2. Ook in dit stelsel is de positie van het cijfer bepalend voor de waarde. Om het binair getal 11101 om te zetten naar een decimaal getal begin je rechts. Op deze positie staan de eenheden van het binair getal (2^0). Vanaf die positie werk je naar links. Op de volgende positie staan de tweetallen (2^1), links daarvan de viertallen (2^2) enzovoort.

Het binair getal 11101 wordt decimaal:

$$\begin{array}{r} 1 \times 2^0 = 1 \\ 0 \times 2^1 = 0 \\ 1 \times 2^2 = 4 \\ 1 \times 2^3 = 8 \\ 1 \times 2^4 = 16 \\ \hline 29 \end{array}$$

decimaal	binair
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

opdracht 1



1.2 VAN DECIMAAL NAAR BINAIR

Om een decimaal getal om te zetten in een binair getal deel je het getal door 2 en noteer je de rest van de deling. Dit herhaal je totdat de uitkomst gelijk is aan 0.

Het decimale getal 29 wordt binair:

```

0 ← 1 ← 3 ← 7 ← 14 ← 29
1   1   1   0   1

```

Je schrijft de bewerking van rechts naar links:

- 29 gedeeld door 2 is 14, rest 1
- 14 gedeeld door 2 is 7, rest 0
- 7 gedeeld door 2 is 3, rest 1
- 3 gedeeld door 2 is 1, rest 1
- 1 gedeeld door 2 is 0, rest 1

Het resultaat kun je nu van links naar rechts lezen.

opdracht 2

1.3 OPTELLEN

Om twee binaire getallen bij elkaar op te tellen zou je ze eerst kunnen omzetten naar decimale getallen, daarna bij elkaar optellen en het resultaat weer omzetten naar een binair getal. Deze methode is omslachtig en de computer rekt natuurlijk alleen in het binair stelsel.

Als je de getallen 29 en 57 bij elkaar op wilt tellen, gaat dat als volgt:

```

29
57 +
---
86

```

Als je deze som uitrekent, tel je eerst de eenheden, 9 en 7, bij elkaar op. Het resultaat is 16 ofwel 6 eenheden en 1 tiental, dat je moet 'onthouden'. De 6 van 16 wordt genoteerd. De 1 wordt bij de tientallen opgeteld. Dus heb je $2 + 5 + 1 = 8$ tientallen. De uitkomst van de optelling is 86.

In het binair stelsel reken je op dezelfde manier:

```

  11101
+ 111001 +
-----
1010110

```

- Eerst worden weer de eenheden bij elkaar opgeteld: $1 + 1 = 10$ (0 eenheden, 1 tweetal). De 0 wordt genoteerd, de 1 moet je 'onthouden'.
- De 1 wordt bij de tweetallen opgeteld: $0 + 0 + 1 = 01$ (1 tweetal, 0 viertallen). De 1 wordt genoteerd.
- $1 + 0 = 01$ (1 viertal, 0 achttallen). De 1 wordt genoteerd.
- $1 + 1 = 10$ (0 achttallen, 1 zestiental). De 0 wordt genoteerd, de 1 moet je 'onthouden'.

- De 1 wordt bij de zestientallen opgeteld: $1 + 1 + 1 = 11$ (1 zestiental, 1 tweeëndertigtal). De 10 wordt genoteerd, de 1 moet je 'onthouden'.
- De 1 wordt bij de tweeëndertigtallen opgeteld: $1 + 1 = 10$ (1 tweeëndertigtal, 1 vierenzestigtal). De 10 wordt genoteerd.

opdracht 3

1.4 ANDERE BEWERKINGEN

Aftrekken van binaire getallen gaat in principe op dezelfde manier als optellen:

$$\begin{array}{r}
 1 \\
 0 - \\
 \hline
 1
 \end{array}
 \quad
 \begin{array}{r}
 11 \\
 1 - \\
 \hline
 10
 \end{array}
 \quad
 \begin{array}{r}
 10 \\
 1 - \\
 \hline
 1
 \end{array}
 \quad
 \begin{array}{r}
 101 \\
 11 - \\
 \hline
 10
 \end{array}$$

In de laatste twee gevallen moet je bij een hogere macht lenen.

Ook het vermenigvuldigen en delen van binaire getallen gaat op dezelfde manier als bij decimale getallen.

opdracht 4-11

2 HEXADECIMALE GETALLEN

Je hebt geleerd hoe je getallen kunt omzetten van het binaire stelsel naar het decimale stelsel en omgekeerd. Op dezelfde manier kun je ook getallen in andere talstelsels schrijven. Een ander talstelsel dat veel gebruikt wordt in bijvoorbeeld HTML-code en tekenprogramma's is het **hexadecimaal stelsel** (het zestientallig stelsel).

Ook in het hexadecimaal stelsel is de positie van het cijfer bepalend voor de waarde van het cijfer in het getal. Je hebt natuurlijk 16 verschillende symbolen nodig om de getallen te noteren. Daarvoor worden de cijfers 0 tot en met 9 en de letters A tot en met F gebruikt.

decimaal	hexadecimaal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F
16	10

2.1 VAN HEXADECIMAAL NAAR DECIMAAL

Het hexadecimaal stelsel is gebaseerd op machten van 16. Om het hexadecimale getal 52 om te zetten naar een decimaal getal begin je rechts. Op deze positie staan de eenheden van het hexadecimale getal (16^0). Vanaf die positie werk je naar links. Op de volgende positie staan de zestientallen (16^1) enzovoort.

Het hexadecimale getal 52 wordt decimaal:

$$\begin{array}{r}
 2 \times 16^0 = 2 \\
 5 \times 16^1 = 80 + \\
 \hline
 82
 \end{array}$$



Het hexadecimale getal B2 wordt decimaal:

$$\begin{array}{r} 2 \times 16^0 = 2 \\ 11 \times 16^1 = 176 \\ \hline 178 \end{array}$$

opdracht 12

2.2 VAN DECIMAAL NAAR HEXADECIMAAL

Om een decimaal getal om te zetten in een binair getal moest je het getal door 2 delen. Wil je een decimaal getal omzetten naar een hexadecimaal getal dan deel je het getal door 16 en noteer je de rest van de deling. Dit herhaal je totdat de uitkomst gelijk is aan 0.

Het decimale getal 41 wordt hexadecimaal:

$$\begin{array}{r} 0 \leftarrow 2 \leftarrow 41 \\ 2 \quad 9 \end{array}$$

- 41 gedeeld door 16 is 2, rest 9
- 2 gedeeld door 16 is 0, rest 2

Het decimale getal 178 wordt hexadecimaal:

$$\begin{array}{r} 0 \leftarrow 11 \leftarrow 178 \\ B \quad 2 \end{array}$$

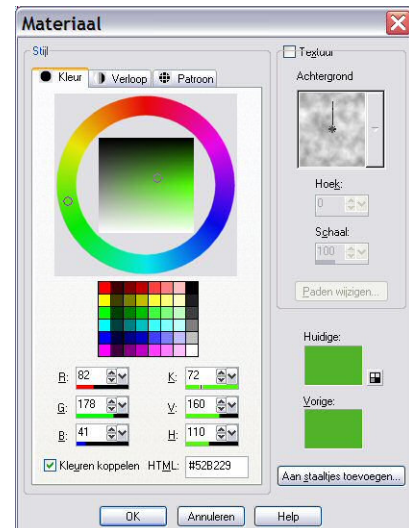
- 178 gedeeld door 16 is 11, rest 2
- 11 gedeeld door 16 is 0, rest 11 (=B)

opdracht 13-16

3 LOGISCHE OPERATIES

Je hebt gezien dat een computer in het binair stelsel rekt. Je kunt je afvragen hoe dit gaat. In een computer wordt gebruik gemaakt van digitale schakelaars, waarin maar twee waarden kunnen voorkomen: de 0 en 1. Een digitale schakelaar kan zo de waarde van een bit vastleggen. Deze schakelaars heten transistoren en daarvan zitten er miljoenen op een chip.

Om de werking duidelijk te maken kun je kijken hoe je een lamp met behulp van gewone schakelaars aan en uit kunt zetten. Zowel de lamp als de schakelaars stellen een bit voor.

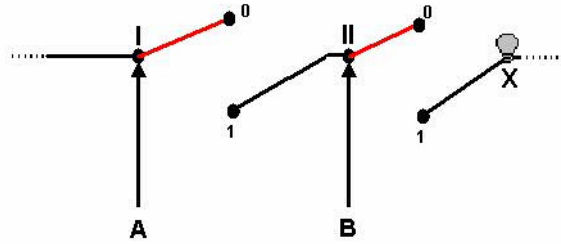


Achter HTML staat de hexadecimale code voor de RGB-kleuren:

$$\begin{array}{l} 82_{10} = 52_{16} \\ 178_{10} = B2_{16} \\ 41_{10} = 29_{16} \end{array}$$

3.1 DE AND-POORT

Als de schakelaars in positie 0 staan, brandt lamp X niet (0). Wanneer er een spanning op ingang A wordt gezet, gaat schakelaar I naar positie 1. Schakelaar II gaat naar positie 1 als er op ingang B een spanning wordt gezet. Alleen wanneer schakelaar A en B in positie 1 staan, gaat lamp X aan.

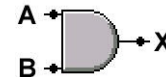


Daarom wordt deze schakeling een AND-poort genoemd.

Alle mogelijke varianten van deze schakeling staan in de tabel:

A	B	lamp
0	0	0
0	1	0
1	0	0
1	1	1

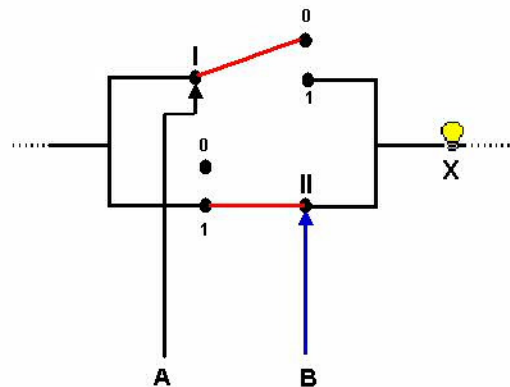
Een AND-poort wordt weergegeven door het volgende symbool:



opdracht 17

3.2 DE OR-POORT

Als de schakelaars in positie 0 staan, brandt lamp X niet (0). Wanneer er een spanning op ingang A wordt gezet, gaat schakelaar I naar positie 1 en gaat lamp X branden. Wanneer er een spanning wordt gezet op ingang B gaat schakelaar II naar positie 1 en dan gaat lamp X ook branden.

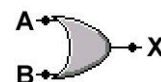


Deze schakeling wordt een OR-poort genoemd, want lamp X gaat branden als er een spanning staat op ingang A, op ingang B of op ingang A en B.

De waarheidstabel van de OR-poort ziet er als volgt uit:

A	B	lamp
0	0	0
0	1	1
1	0	1
1	1	1

Een OR-poort wordt weergegeven door het volgende symbool:

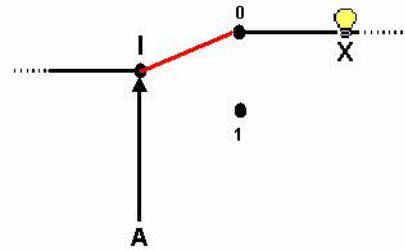


opdracht 18

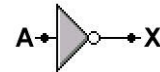


3.3 DE NOT-POORT

Als de schakelaar in positie 0 staat, brandt lamp X (1). Wanneer er een spanning op ingang A wordt gezet, gaat schakelaar I naar positie 1 en brandt lamp X niet (0). De stand van de schakelaar is dus omgekeerd aan de toestand van lamp X. Deze schakeling wordt een NOT-poort of Inverter (keert de invoer om) genoemd.



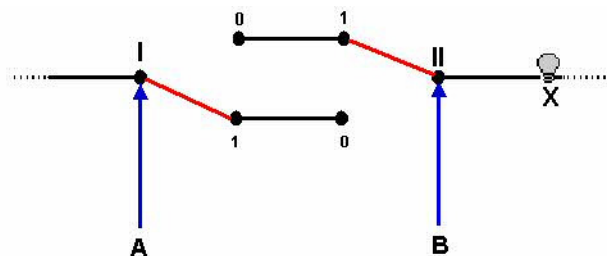
Een NOT-poort wordt weergegeven door het volgende symbool:



opdracht 19

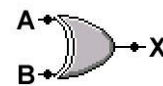
3.4 DE XOR-POORT

Als de schakelaars in positie 0 staan, brandt lamp X niet (0). Wanneer er een spanning op ingang A wordt gezet, gaat schakelaar I naar positie 1 en gaat lamp X branden. Wanneer er een spanning wordt gezet op ingang B gaat schakelaar II naar positie 1 en dan gaat lamp X branden.



Deze schakeling wordt een XOR-poort genoemd, want lamp X gaat branden als er een spanning staat op ingang A en niet op ingang B of niet op ingang A en wel op ingang B.

Een XOR-poort wordt weergegeven door het volgende symbool:

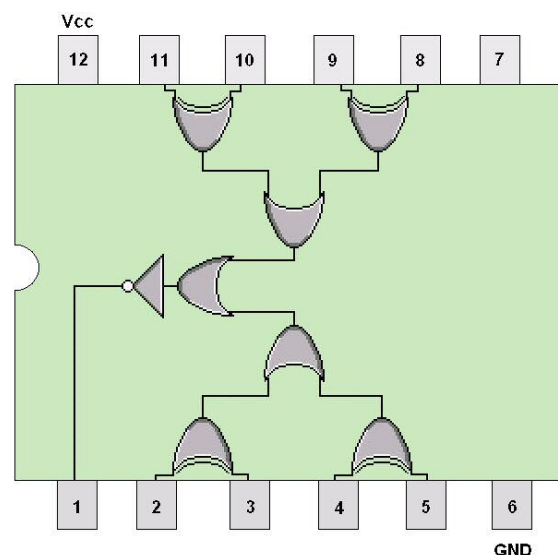


opdracht 20-22

4 CHIPS

In de praktijk kom je geen losse poorten tegen, maar zijn ze geïntegreerd in een chip. Chips of IC's (geïntegreerde circuits) bestaan uit een stukje silicium waarop enkele poorten zijn aangebracht. Ze worden gemonteerd in een plastic of keramische behuizing. Langs de rand bevinden zich rijen pinnen, waarmee de chip wordt bevestigd op een printkaart.

In de afbeelding rechts zie je een schematisch ontwerp van een chip die twee invoerwaarden met een lengte van vier bits met elkaar vergelijkt. De pinnen 2, 3, 4, 5, 8, 9, 10 en 11 zijn pinnen voor de input en pin 1 is voor de output. Daarnaast heeft de chip een pin voor de voeding (Vcc) en een pin voor de aarde (GND), die door alle poorten worden gebruikt. Pin 7 is een lose pin. De inkeping bij pin 1 zit er om de oriëntatie aan te geven.



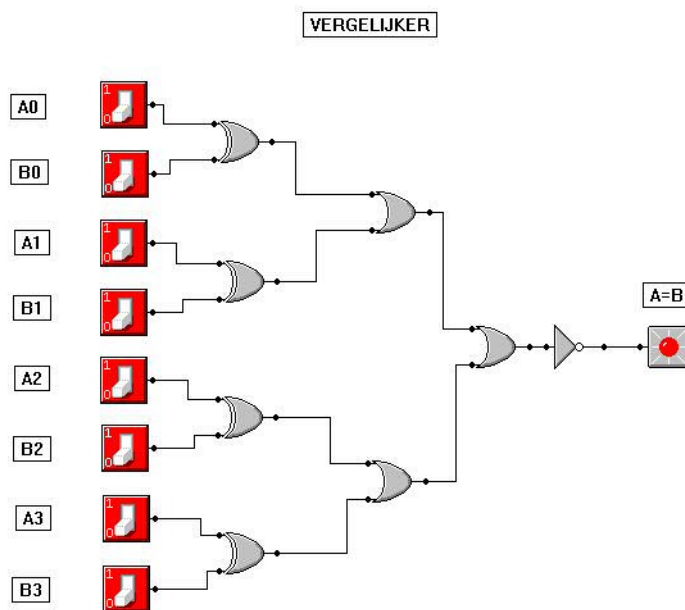
Chips worden in vier klassen verdeeld op grond van het aantal poorten dat ze bevatten:

SSI-chips (Small Scale Integrated):	1 tot 10 poorten
MSI-chips (Medium Scale Integrated):	10 tot 100 poorten
LSI-chips (Large Scale Integrated):	100 tot 100.000 poorten
VLSI-chips (Very Large Scale Integrated):	meer dan 100.000 poorten

We gaan het ontwerp van twee relatief eenvoudige chips bekijken: de **vergelijker** uit het voorbeeld en de **opteller**.

4.1 DE VERGELIJKER

Als voorbeeld van een simpele chip is een vergelijker getekend. De basispoort voor deze schakeling is de XOR, die een 0 als uitvoer geeft als de invoer gelijk is en een 1 als ze ongelijk zijn. Als twee waarden gelijk zijn, dan leveren alle vier de XOR-poorten een 0. Deze vier signalen worden met OR-poorten gecombineerd. Als het resultaat van die combinatie 0 is dan zijn de invoerwaarden gelijk, anders niet. Door als laatste poort een NOT te gebruiken wordt het resultaat omgekeerd: is het resultaat een 1 (de LED brandt) dan zijn de waarden gelijk.



opdracht 23-24



4.2 DE OPTELLER

Met behulp van deze logische schakelingen kun je een eenvoudige 'opteller' bouwen. Kijk eerst naar het optellen van 1-bits getallen.

$$\begin{aligned} 0 + 0 &= 00 \\ 0 + 1 &= 01 \\ 1 + 0 &= 01 \\ 1 + 1 &= 10 \end{aligned}$$

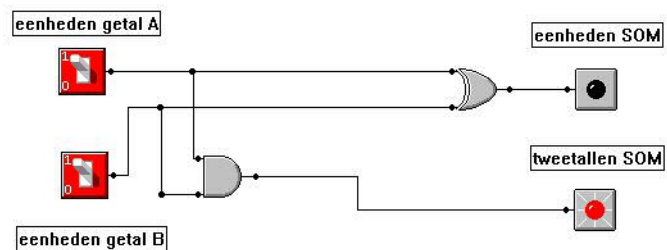
Je zou deze bewerkingen ook in een tabel kunnen zetten.

A	B	Tweetallen	Eenheden
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Wanneer je de uitkomst vergelijkt met de waarheidstabellen van de verschillende poorten, dan blijkt dat de eenheden worden bepaald door een XOR-poort. De tweetallen worden bepaald door een AND-poort.

Met behulp van één XOR-poort en één AND-poort kun je de rekenmachine bouwen voor het optellen van 1-bits getallen.

Deze schakeling wordt een half adder (adder=opteller) genoemd. Met een **half adder** kun je alleen twee 1-bits getallen bij elkaar optellen. Bij het optellen van grotere getallen heb je het probleem van de overdracht.

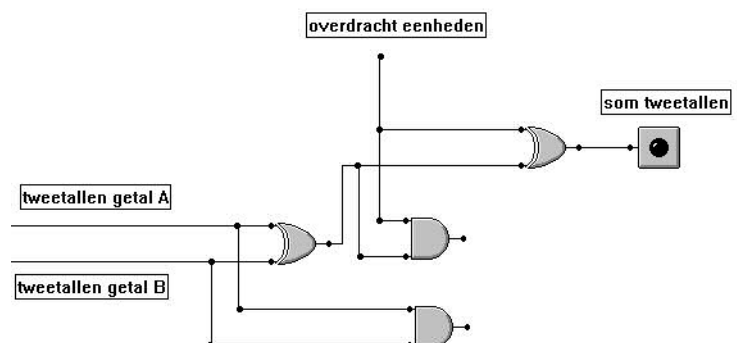


opdracht 25

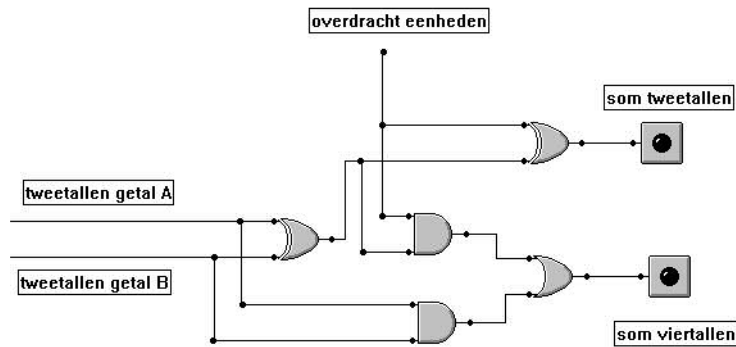
In de praktijk wil je grotere getallen kunnen optellen. In dat geval heb je het probleem van de overdracht. Wanneer je bijvoorbeeld de binaire getallen 01 en 11 bij elkaar wilt optellen gaat dat als volgt:

$$\begin{array}{r} 01 \\ 11 + \\ \hline 100 \end{array} \quad \begin{array}{l} \text{Tel eerst de eenheden bij elkaar op: } 1 + 1 = 10; \\ 0 \text{ opschrijven; } 1 \text{ onthouden (de overdracht)} \\ \text{Dan tel je de tweetallen bij elkaar op en daarbij de overdracht: } 0 + 1 + 1 = 10 \end{array}$$

Met één half adder kun je de eenheden bij elkaar optellen. Voor het optellen van de tweetallen moet je eerst de tweetallen bij elkaar optellen en bij de uitkomst daarvan de overdracht. Dit kan door twee half adders achter elkaar te zetten.



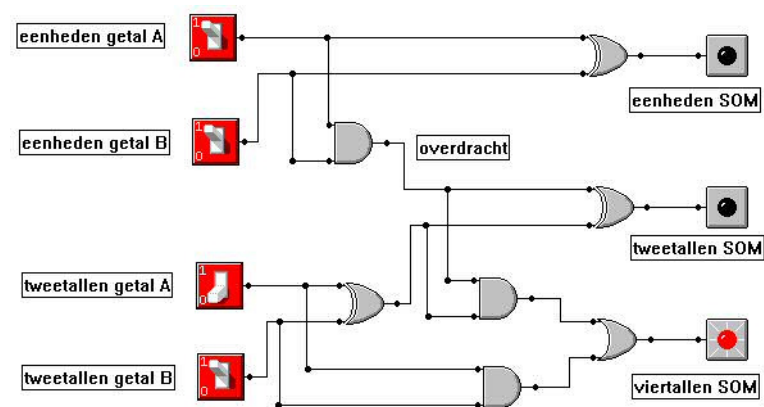
Om de viertallen te bepalen moet er een OR-poort worden toegevoegd, die "kijkt" of er een viertal is.



Deze schakeling wordt de **full adder** genoemd.

opdracht 26

Om een 2-bits opteller te maken moet je een half adder en een full adder combineren.



opdracht 27-33

