4 Booleaanse algebra en het 7-segmentsdisplay

In het vorige hoofdstuk hebben we geleerd dat digitale signalen vaak binair zijn en dat er logische poorten bestaan waarmee je kunt zorgen dat jouw schakeling bij bepaalde ingangswaarden de goede output geeft. Deze week gaan we de logische poorten in een schema combineren, zodat we ingewikkelder functionaliteiten kunnen realiseren. Om de elektrische schema's toch zo eenvoudig mogelijk te houden maken we gebruik van Booleaanse algebra. Tot slot leer je hoe je met binaire signalen decimale getallen kunt maken en hoe je deze afbeeldt op een 7-segmentsdisplay van de SMU.

Op de SMU zit een 7-segmentsdisplay dat gebruikt kan worden om cijfers of letters weer te geven. Uiteindelijk is het de bedoeling om dit via de microcontroller aan te sturen, maar deze week gaan we vast kijken hoe het 7-segmentsdisplay én de bijbehorende decoder werken.

4.1 Het 7-segmentsdisplay

Een 7-segmentsdisplay bestaat uit 7 losse ledjes die aan één kant verbonden zijn. Er zijn 2 manieren om de afzonderlijke ledjes aan te sturen: CA en CC.

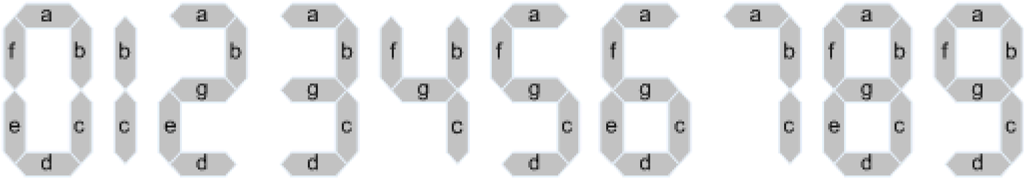
THUISOPDRACHT 1:

a) Zoek op waar CA en CC voor staat en schets het schema van beide aansluitingen.
b) Zoek in de datasheet van het 7-segmentsdisplay in je componentenset op van welk type (CC of CA) deze is.

LABOPDRACHT 1:

a) Zoek in de datasheet op wat de nominale stroom is die je door één van de LEDjes van het display dient te sturen.
b) Stel de stuurspanning in op 5 Volt en bereken de grootte van de voorschakelweerstand die je nodig hebt.
c) Wat denk je dat beter is: één weerstand naar de gnd en alle ingangen direct aansturen of één draad naar de ground en per LED een weerstand naar de stuurspanning?
d) Bouw de aansluiting met één weerstand aan de gnd op op je breadboard en meet de stroom en spanning door een segment van het display als er een 1 op het display staat én als er een 8 op het diplay staat.
e) Doe nu hetzelfde voor de andere manier van aansluiten: dus één weerstand aan ieder segment aangesloten en geen weerstand aan de gnd.
f) Wat is het verschil tussen de aansluitmethodes? Let op gemeten spanning en stroom, maar ook op wat je ziet!

Om het 7-segmentsdisplay aan te sturen kan je alle 7 LEDs direct aansturen. Je moet dan je microcontroller laten uitrekenen welke LEDs bij welke getallen aan moeten en kan dan de cijfers via 7 aansluitingen op het display laten verschijnen. Om het aantal aansluitingen en stroomverbruik te reduceren wordt er vaak gebruik gemaakt van een decoder.



4.2 De decoder

Om het 7-segmentsdisplay op de SMU aan te sturen wordt gebruik gemaakt van een decoder. Deze decoder zet een signaal dat het van de microcontroller krijgt om zodat de juiste LEDs van het display aan gaan. Het signaal dat de decoder van de microcontroller krijgt is een digitaal binair signaal, zoals we vorige week hebben besproken.

4.2.1 Binaire getallen

Om een getal van de microcontroller naar de decoder te versturen maken we gebruik van binaire getallen. Binaire getallen zijn opgebouwd uit enen en nullen, waarvan de positie van de enen overeenkomt met een macht van twee (de grootte van de macht hangt af van de plek van de één):

Binair getal	Macht	Decimaal getal
00001	2^0	1
00010	2^1	2
00100	2^2	4
01000	2^3	8
10000	2^4	16

Stel dat er meer dan één keer een ‘1’ in een binair getal staat, dan bepaal je alle decimale getallen van elke ‘1’ en neem je de som van alle decimale getallen:

000101 = $2^2 + 2^0 = 4 + 1 = 5$

Je kunt ook gebruik maken van een tabel, waarin je op de bovenste regel alle machten van 2 opschrijft (2^0 helemaal rechts). Daaronder schrijf je je binaire getal op. Vervolgens vermenigvuldig je de vakken die onder elkaar liggen met elkaar en tel je alles bij elkaar op:

2^4	2^3	2^2	2^1	2^0	
1	**0**	**1**	**1**	**0**	
$\mathbf{2^4 \times 1 = 16}$	$2^3 \times 0 = 0$	$\mathbf{2^2 \times 1 = 4}$	$\mathbf{2^1 \times 1 = 2}$	$2^0 \times 0 = 0$	$16 + 4 + 2 = 22$

Om van een decimaal getal een binair getal te maken gebruik je ook de machtreeks van twee. Je begint met te bekijken wat de grootste macht van twee is die in jouw decimale getal past. Stel we willen het getal 39 omzetten, dan is de grootste macht van twee die daar in past 2^5 = 32. Vervolgens trek je deze 32 af van je begingetal: 39-32 = 7. Van dit getal ga je weer kijken wat de hoogste macht van twee is die erin past: $2^2 = 4$. Deze trek je er weer van af en zo ga je door tot je bij 0 uitkomt. Als het goed is vindt je dan dat het getal 39 is opgebouwd uit $2^5 + 2^2 + 2^1 + 2^0$. Het binaire getal voor 39 is dus 100111.

THUISOPDRACHT 2:

Reken de volgende getallen om van binair naar decimaal en vice versa:

a) 1010
b) 00101101
c) 10010110
d) 23
e) 48
f) 78

4.2.2 De decoder toegepast

De decoder die zich op de SMU bevindt, vind je ook in je componentenset. Dit is een IC (integrated circuit) van de HEF4000 serie.

LABOPDRACHT 2:

Zoek op internet de datasheet van deze decoder een HEF4511 van de firma Philips/NXP. Beantwoord de volgende vragen:

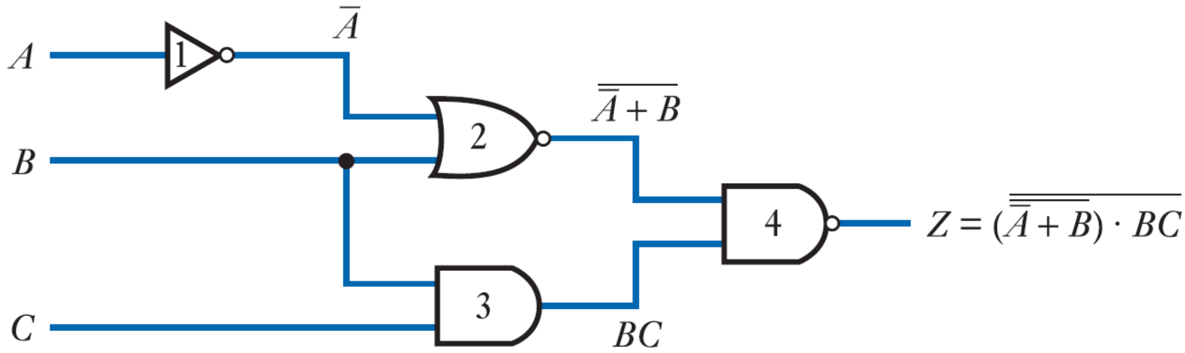
a) Maak een overzicht welk pootje van het IC welke functie heeft.
b) Welke van de twee typen displays (CC of CA) kan het beste door dit IC aangestuurd worden? Verklaar je antwoord. Tip: check de application information in de datasheet.
c) Waartoe dienen de aansluitingen LE, $\overline{BL}$ en $\overline{LT}$? Waar knoop je deze signalen aan vast?
d) Wat gebeurt er als je deze drie ingangen niet aansluit?
e) Teken een schema waarin de 4511 en een 7 segmentdisplay gecombineerd worden. Gebruik hier bij de voorschakel weerstand waarvan je de grootte hebt uitgerekend in opdracht 1.
f) Laat je schema controleren.
g) Bouw je schema met deze decoder en het 7-segment display op een breadboard.
h) Probeer alle cijfers uit en check of je schakeling het goed doet!

4.3 De decoder van binnen

Binnen in de decoder bevindt zich een elektronische schakeling die ervoor zorgt dat elk LEDsegment bij de juiste inputwaardes aan gaat. Hoe die elektronische schakeling eruit ziet kunnen we afleiden uit de waarheidstabel. Daarvoor hebben we wel eerst wat theoretische kennis nodig over combinatorische logica.

4.3.1 Logisch schema en Booleaanse expressie

Met behulp van logische poorten (AND, OR, NOT, NAND, enz.) is het mogelijk om logische systemen op te bouwen. Hieronder zie je van zo'n systeem een **logisch schema**:



Om de Booleaanse vergelijking te vinden die bij dit schema hoort begin je aan de linkerkant en schrijf je na elke logische poort op wat er met het ingangssignaal is gebeurd. In het schema hierboven is de volgorde waarin je de poorten langsgaat aangegeven met de nummers 1 t/m 4. Doe je dit voor elke poort, dan kom je uiteindelijk uit bij output Z: $Z = \overline{\overline{\bar{A} + B} \cdot BC}$.

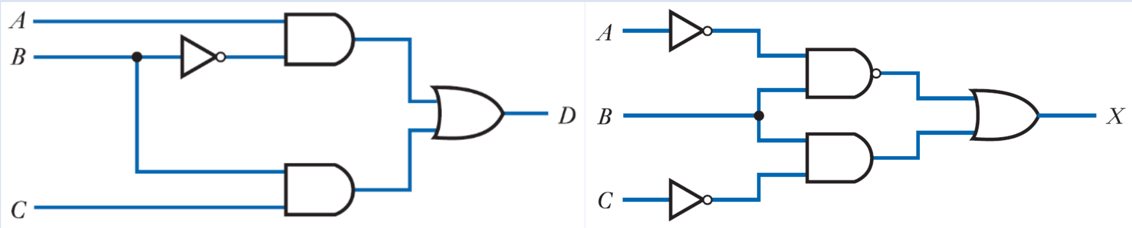
Om van de Booleaanse uitdrukking naar het bijbehorende logische schema te komen zoek je eerst het grootste element in de vergelijking. Voor $Z = \overline{\overline{\bar{A} + B} \cdot BC}$ is dat de NAND-poort $\overline{\ldots \cdot \ldots}$. Aangezien de strepen werken als een soort haakjes, kan je ook eerst bekijken wat de bovenste streep doet en van daaruit naar beneden werken. We starten dus met een NAND-poort en tekenen die aan de rechterkant van het schema. Vervolgens kijk je wat de ingangen van deze NAND-poort moeten zijn. In dit geval is dat voor de ene kant $\overline{\bar{A} + B}$ en voor de andere kant $B \cdot C$. Kijk weer wat de grootste elementen hiervan zijn: voor $\overline{\bar{A} + B}$ is dat een NOR-poort en voor $B \cdot C$ een AND-poort. Deze teken je dus links van de NAND-poort. Tot slot blijft er nog één poort over, want in de NOR-poort moet $\bar{A}$. Hiervoor teken je voor de NOR-poort nog een NOT-poort en dan is je schema af.

THUISOPDRACHT 3:

Teken de logische schema's voor de volgende Booleaanse functies:

a) $X = \overline{A + B} \cdot C$
b) $Y = A\bar{B}C + \bar{A}D + C\bar{D}$
c) $Z = \overline{A \cdot B + \overline{C + D}}$

THUISOPDRACHT 4:
Geef de Booleaanse uitdrukkingen voor de volgende schema's:



4.3.2 Logisch schema vanuit een waarheidstabel

In praktijk is aanvankelijk vaak nog niet duidelijk wat de Booleaanse expressie van een systeem is, terwijl je er wel graag een schakeling voor wilt bouwen. In dat geval kan je de werking van het systeem meestal in een waarheidstabel samenvatten. Vervolgens kunnen we uit de waarheidstabel een Booleaanse expressie afleiden en aan de hand daarvan een logisch schema opstellen.

Stel, we willen een systeem maken dat gedefinieerd wordt door onderstaande waarheidstabel:

A	B	C	D	*minterms*
0	0	0	0	
0	0	1	1	$\bar{A}\bar{B}C$
0	1	0	0	
0	1	1	0	
1	0	0	1	$A\bar{B}\bar{C}$
1	0	1	0	
1	1	0	0	
1	1	1	1	ABC

Om de Booleaanse expressie van dit systeem te vinden gaan we op zoek naar de **minterms** in de waarheidstabel. De mintermen zijn af te leiden uit de regels van de waarheidstabel waarvoor de output gelijk is aan '1'. De minterm is opgebouwd uit de inputwaardes die horen bij de outputwaardes die '1' zijn: is de inputwaarde '0', dan wordt de input geïnverteerd ($\bar{A}$); is de inputwaarde '1', dan wordt de input niet geïnverteerd (B). Voor de eerste regel waarop D gelijk is aan '1' zijn inputwaarden ABC gelijk aan '001'. Hierbij hoort de minterm $\bar{A}\bar{B}C$.

LET OP: de streepjes boven de variabelen van de minterm zijn altijd gescheiden en mag je niet doortrekken, $\bar{A}\bar{B} \neq \overline{AB}$!

Wanneer je alle mintermen van je systeem gevonden hebt kan je de volledige Booleaanse expressie opstellen. Deze is namelijk gelijk aan de som van alle mintermen: $D = \bar{A}\bar{B}C + A\bar{B}\bar{C} + ABC$.

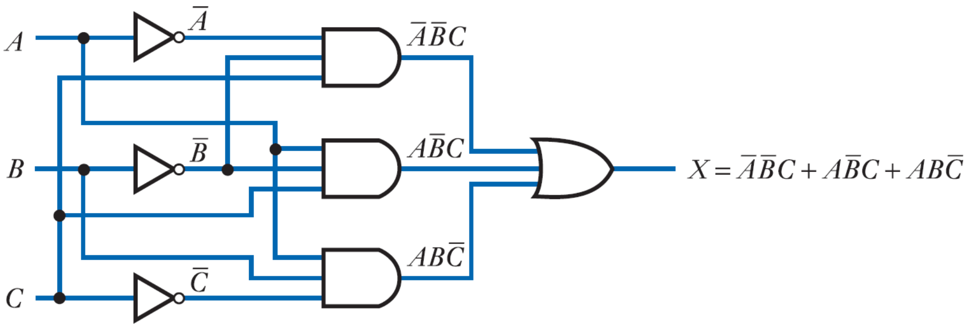
Vervolgens kan je de Booleaanse expressie weer omzetten naar een logisch schema zoals in de vorige paragraaf besproken.

THUISOPDRACHT 5:
Teken de logische schema's van de systemen gedefinieerd door onderstaande waarheids-tabellen.

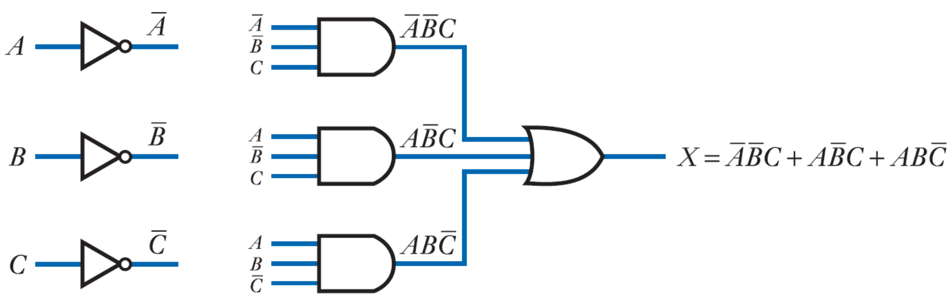
A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Je ziet dat de vorm van het logisch schema op deze manier steeds hetzelfde is: een OR-poort aan het eind, met een aantal AND-poorten daarvoor, aangevuld met wat NOT-poorten. Naar mate je meer minterms hebt, wordt dit schema steeds moeilijker te lezen. Daarom mag het schema ook anders getekend worden:



wordt dan



4.3.3 Schakelalgebra

Hoe meer minterms je Booleaanse expressie heeft, hoe ingewikkelder en uitgebreider het schema wordt, en hoe meer logische poorten je ook nodig hebt om je systeem te bouwen. Vanuit praktisch oogpunt willen we die Booleaanse expressie dus graag zo kort mogelijk houden: simpel schema, minder componenten nodig. Het zou dus fijn zijn als we de lange Booleaanse vergelijkingen die we via de mintermen krijgen kunnen verkorten. Een manier om dat te bereiken is gebruik maken van Booleaanse algebra.

Bij het vereenvoudigen van Booleaanse expressies via Booleaanse algebra maken we gebruik van een aantal wetten en regels, hieronder samengevat in een tabel.

Boolean identities

AND function	**OR function**	**NOT function**
$0 \cdot 0 = 0$	$0 + 0 = 0$	$\overline{0} = 1$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$\overline{1} = 0$
$1 \cdot 0 = 0$	$1 + 0 = 1$	$\overline{\overline{A}} = A$
$1 \cdot 1 = 1$	$1 + 1 = 1$	
$A \cdot 0 = 0$	$A + 0 = A$	
$0 \cdot A = 0$	$0 + A = A$	
$A \cdot 1 = A$	$A + 1 = 1$	
$1 \cdot A = A$	$1 + A = 1$	
$A \cdot A = A$	$A + A = A$	
$A \cdot \overline{A} = 0$	$A + \overline{A} = 1$	

Boolean laws

Commutative law	**Absorption law**
$AB = BA$	$A + AB = A$
$A + B = B + A$	$A(A + B) = A$
Distributive law	**De Morgan's law**
$A(B + C) = AB + AC$	$\overline{A + B} = \overline{A} \cdot \overline{B}$
$A + BC = (A + B)(A + C)$	$\overline{A \cdot B} = \overline{A} + \overline{B}$
Associative law	**Note also**
$A(BC) = (AB)C$	$A + \overline{A}B = A + B$
$A + (B + C) = (A + B) + C$	$A(\overline{A} + B) = AB$

Een paar punten van aandacht:

a) Let op het gebruik van haakjes
b) De AND functie gaat voor de OR functie (net als vermenigvuldigen gaat voor optellen)
c) Let goed op hoe lang de streep wordt/is boven een geïnverteerde variabele

Wanneer we beginnen vanuit een waarheidstabel kunnen we een expressie afleiden die bestaat uit een aantal mintermen. Bijvoorbeeld:

$$Z = ABC + \bar{A}B\bar{C} + A\bar{B}C$$

Deze kunnen we vereenvoudigen door op zoek te gaan naar termen die grotendeels gelijk aan elkaar zijn, maar waarvan één variabele anders is. Deze ene variabele kunnen we dan binnen haakjes plaatsen. In het voorbeeld hierboven geldt dat voor de eerste en de laatste term:

$$Z = AC(B + \bar{B}) + \bar{A}B\bar{C}$$

Nu maken we gebruik van de regel: $A + \bar{A} = 1$:

$$Z = AC \cdot 1 + \bar{A}B\bar{C}$$

Nu kunnen we de regel $A \cdot 1 = A$ toepassen:

$$Z = AC + \bar{A}B\bar{C}$$

Zo hebben we de vergelijking ingekort van drie naar twee termen, waarvan één term ook nog minder variabelen heeft. Als we dit omzetten naar logische poorten zijn voor de oorspronkelijke vergelijking 6 AND-poorten, 3 NOT-poorten en 2 OR-poorten nodig. Voor de vereenvoudigde vergelijking hebben we nog maar 3 AND-poorten, 2 NOT-poorten en 1 OR-poort nodig. De helft bespaard dus!

Nog een voorbeeld:

$$E = B\bar{C}\bar{D} + \bar{A}BD + ABD + BC\bar{D} + \bar{B}CD + \bar{A}\bar{B}\bar{C}D + A\bar{B}\bar{C}D$$

Vergelijkbare termen buiten haakjes halen geeft:

$$E = B\bar{D}(\bar{C} + C) + BD(\bar{A} + A) + \bar{B}CD + \bar{B}\bar{C}D(\bar{A} + A)$$

$$E = B\bar{D} + BD + \bar{B}CD + \bar{B}\bar{C}D$$

Nogmaals vergelijkbare termen buiten haakjes halen:

$$E = B(\bar{D} + D) + \bar{B}D(\bar{C} + C)$$

$$E = B + \bar{B}D$$

Nu maken we gebruik van de wet $A + \bar{A}B = A + B$:

$$E = B + D$$

Er blijft nu nog maar één poort over.

THUISOPDRACHT 6:
Vereenvoudig de volgende algebraïsche functies:

a) $F = A + AB + AC$
b) $F = \bar{A} + AB + BC$
c) $F = \overline{A + B} \cdot B$
d) $F = \bar{A}B\bar{C}D + AB\bar{C}D + \bar{A}BCD + ABCD + \bar{A}BC\bar{D} + \bar{A}\bar{B}C\bar{D} + ABC\bar{D} + A\bar{B}C\bar{D}$

4.3.4 Logica in de decoder

De Booleaanse logica die we net in theorie hebben geoefend is ook in de decoder terug te vinden. In de datasheet van de decoder staat een waarheidstabel (Functional Description). Hierin staat wat er met de outputs gebeurt voor welke inputs.

LABOPDRACHT 3:

Zoek de waarheidstabel van de decoder (HEF4511B) op in de dataheet van dit onderdeel. Gebruik hiervoor de datasheet van NXP.

a) Op de eerste regel van de waarheidstabel vind je de waardes ‘L’, ‘X’, en ‘H’. Waar staan deze waardes voor en wat betekenen ze?

b) Geef de Booleaanse uitdrukking voor Q_c als functie van de ingangen D_0 t/m D_3. Ga ervan uit dat LE, BL en LT op de waardes staan waarbij er cijfers op het display verschijnen.
Aangezien deze output bestaat uit veel ‘H’-tjes en niet zoveel ‘L’-etjes is het in dit geval handiger om niet naar de mintermen, maar juist naar de maxtermen te kijken (regels waarbij de output 0 is). Je vergelijking wordt dan:
$\overline{Q_c} = \text{maxterm} + \text{maxterm} + \cdots$

c) Laat zien dat deze vergelijking te vereenvoudigen is tot:
$Q_c = \overline{D_1} \cdot \overline{D_2} + \overline{D_1} \cdot \overline{D_3} + D_2 \cdot \overline{D_3} + D_0 \cdot \overline{D_3}$

Op de datasheet staat ook een logisch schema (Logic Diagram) van de schakeling.

d) Leid aan de hand van dit schema een Booleaanse vergelijking af voor Q_c. Je mag hierbij aannemen dat de ingangen ‘LE’, LT en BL op de waarden staan waarbij cijfers worden afgebeeld op het display.

e) Komt deze vergelijking overeen met wat je uit de waarheidstabel hebt afgeleid?

f) Is de kortst mogelijke vergelijking geïmplementeerd? Waarom wel/niet?

VOORTGANGSOPDRACHT 3:

Zodra je er klaar voor bent kan je je docent vragen om de derde voortgangsopdracht die afgetekend moet worden om mee te mogen doen aan de eindtoets.

Voor deze opdracht moet je laten zien dat je een zo compact mogelijke logische schakeling kan ontwerpen op basis van een waarheidstabel.