3 Digitale techniek Logische vergelijkingen

Logische vergelijkingen laten toe om een bepaalde logische beweringen eenduidig te beschrijven. Via booleaanse algebra kunnen we deze vergelijkingen vereenvoudigen en minimaliseren. Dankzij deze techniek kunnen we het aantal digitale IC's in digitale systemen tot een minimum beperken. Let echter wel op dat je bij het oplossen van booleaanse vergelijkingen de regels van de klassieke algebra niet altijd zomaar mag toepassen!

3.1 Boole algebra

3.1.1 Basiswetten

De drie basiswetten van de Booleaanse algebra zijn dezelfde als in de klassieke algebra: de *commutatieve* wetten, de *associatieve* wetten en de *distributieve* wetten.

Commutativiteit

De *volgorde* waarin de termen van een **som** staan in een logische vergelijking is onbelangrijk:

$$A + B = B + A$$

De *volgorde* waarin de factoren van een **product** staan in een logische vergelijking is onbelangrijk:

$$A \cdot B = B \cdot A$$

△ Fig.3.1 | Voorbeeld van commutativiteit.
Bron: W.Van Wichelen

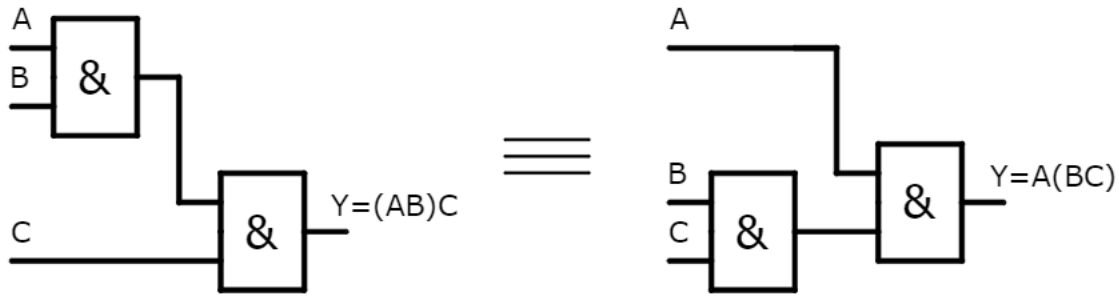
Associativiteit

Je mag steeds termen *groeperen* in een logische vergelijking:

$$(A+B)+C=A+(B+C)$$

Je mag steeds factoren *groeperen* in een logische vergelijking:

$$(A\cdot B)\cdot C=A\cdot(B\cdot C)$$

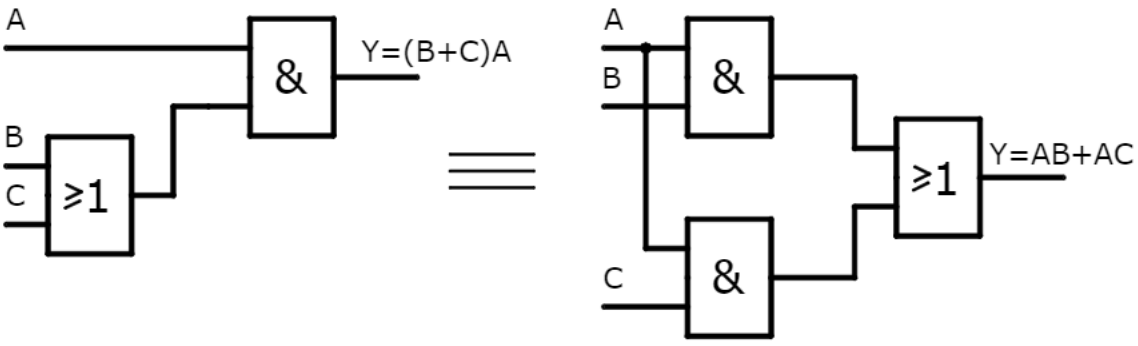


△ Fig.3.2 | Voorbeeld van associativiteit.
Bron: W.Van Wichelen

Distributiviteit

Je mag de haakjes uitwerken zoals bij de klassieke algebra:

$$A\cdot(B+C)=A\cdot B+A\cdot C$$



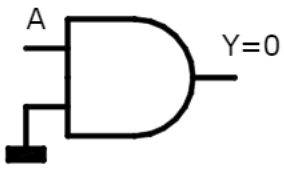
△ Fig.3.3 | Voorbeeld van distributiviteit.
Bron: W.Van Wichelen

3.1.2 Basisregels vermenigvuldiging

Logisch vermenigvuldigen met 0

Als in een product één of meer factoren '0' zijn, dan is het resultaat altijd '0':

$$A \cdot 0 = 0$$

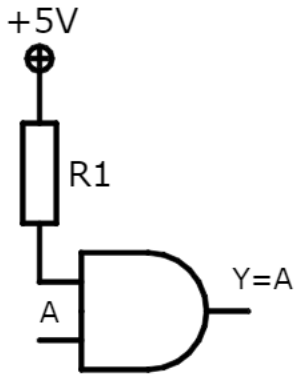


△ Fig.3.4 | Voorbeeld van logisch vermenigvuldigen met '0'.
Bron: W.Van Wichelen

Logisch vermenigvuldigen met 1

Als in een product één of meer factoren '1' zijn, dan hebben die factoren geen invloed op het resultaat:

$$A \cdot 1 = A$$

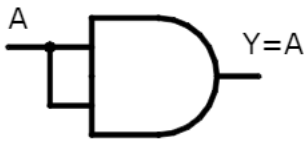


△ Fig.3.5 | Voorbeeld van logisch vermenigvuldigen met '1'.
Bron: W.Van Wichelen

Logisch vermenigvuldigen met zichzelf

Als in een product één of meer factoren *dezelfde* zijn, dan volstaat het die factoren slechts één keer te noteren:

$$A \cdot A = A$$

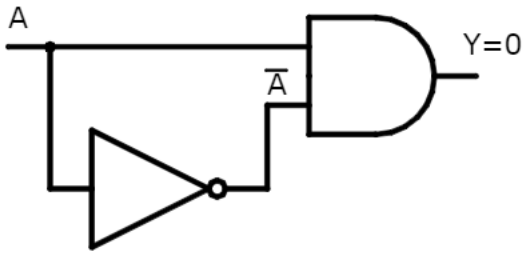


△ Fig.3.6 | Voorbeeld van logisch vermenigvuldigen met zichzelf. Bron: W.Van Wichelen

Logisch vermenigvuldigen met het inverse

Als in een product twee factoren elkaars *tegengestelde* zijn, dan is het resultaat steeds '0':

$$A \cdot \overline{A} = 0$$



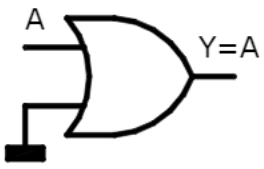
△ Fig.3.7 | Voorbeeld van logisch vermenigvuldigen met het inverse. Bron: W.Van Wichelen

3.1.3 Basisregels optelling

Logisch optellen bij 0

Als in een som één of meer termen '0' zijn, dan hebben die termen geen invloed op het resultaat:

$$A + 0 = A$$

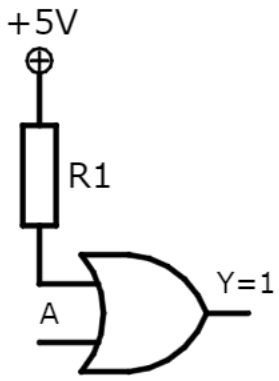


△ Fig.3.8 | Voorbeeld van logisch optellen met '0'.
Bron: W.Van Wichelen

Logisch optellen bij 1

Als in een som één of meer termen '1' zijn, dan is het resultaat altijd '1':

$$A + 1 = 1$$

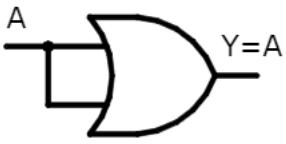


△ Fig.3.9 | Voorbeeld van logisch optellen met '1'.
Bron: W.Van Wichelen

Logisch optellen bij zichzelf

Als in een som één of meer termen *dezelfde* zijn, dan volstaat het die term slechts één keer te noteren:

$$A + A = A$$

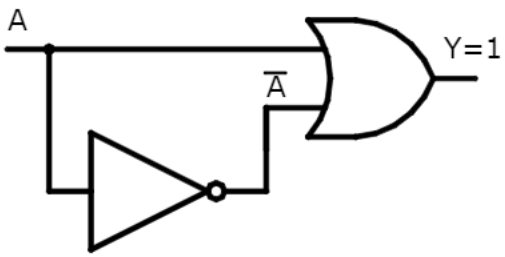


△ Fig.3.10 | Voorbeeld van logisch optellen met zichzelf.
Bron: W.Van Wichelen

Logisch optellen bij het inverse

Als in een som twee termen elkaars *tegengestelde* zijn, dan is het resultaat steeds '1':

$$A + \overline{A} = 1$$



△ Fig.3.11 | Voorbeeld van logisch optellen met inverse.
Bron: W.Van Wichelen

3.1.4 Tweemaal inverteren

Let op!

Een variabele die *tweemaal* geïnverteerd wordt, keert terug tot haar oorspronkelijke toestand:

$$\overline{\overline{A}} = A$$

3.2 Theorema van De Morgan

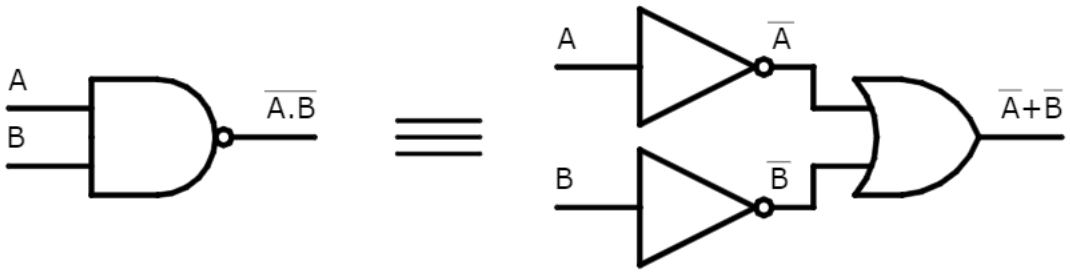
Het *theorema van De Morgan*[12] formuleert twee wetten in de booleaanse logica die een verband leggen tussen AND en OR en de negatie (NOT).

3.2.1 Van product naar som

Let op!

Met booleaanse algebra is het mogelijk een een logisch *product* om te vormen naar een logische *som*:

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$



△ Fig.3.12 Van logische som naar logisch produkt.
Bron: W.Van Wichelen

Uit bovenstaande uitdrukking kunnen we afleiden dat we met NOR-poorten een *AND-functie* kunnen realiseren:

$$\Rightarrow \overline{A \cdot B} = \overline{A} + \overline{B}$$

$$\Leftrightarrow \overline{\overline{A \cdot B}} = \overline{\overline{A} + \overline{B}}$$

$$\Leftrightarrow A \cdot B = \overline{\overline{A} + \overline{B}}$$

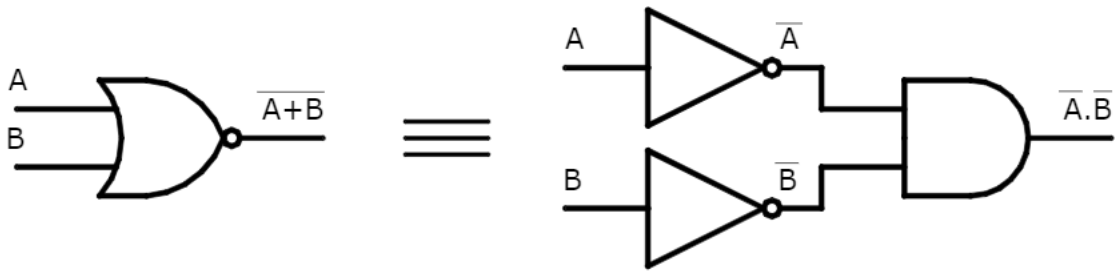
[12] Augustus De Morgan (1806–1871) was een Britse wiskundige en logicus. Bij het formuleren van zijn stelling werd hij beïnvloed door de ontwikkeling van de Booleaanse algebra door George Boole. Voor hem formuleerden Aristoteles en middeleeuwse denkers reeds deze wet.

3.2.2 Van som naar product

Let op!

Met booleaanse algebra is het mogelijk een een logische *som* om te vormen naar een logisch *product*:

$$\overline{A+B} = \overline{A} \cdot \overline{B}$$



△ Fig.3.13 | Van logisch produkt naar logische som.
Bron: W.Van Wichelen

Uit bovenstaande uitdrukking kunnen we afleiden dat we met NAND-poorten een *OR-functie* kunnen realiseren:

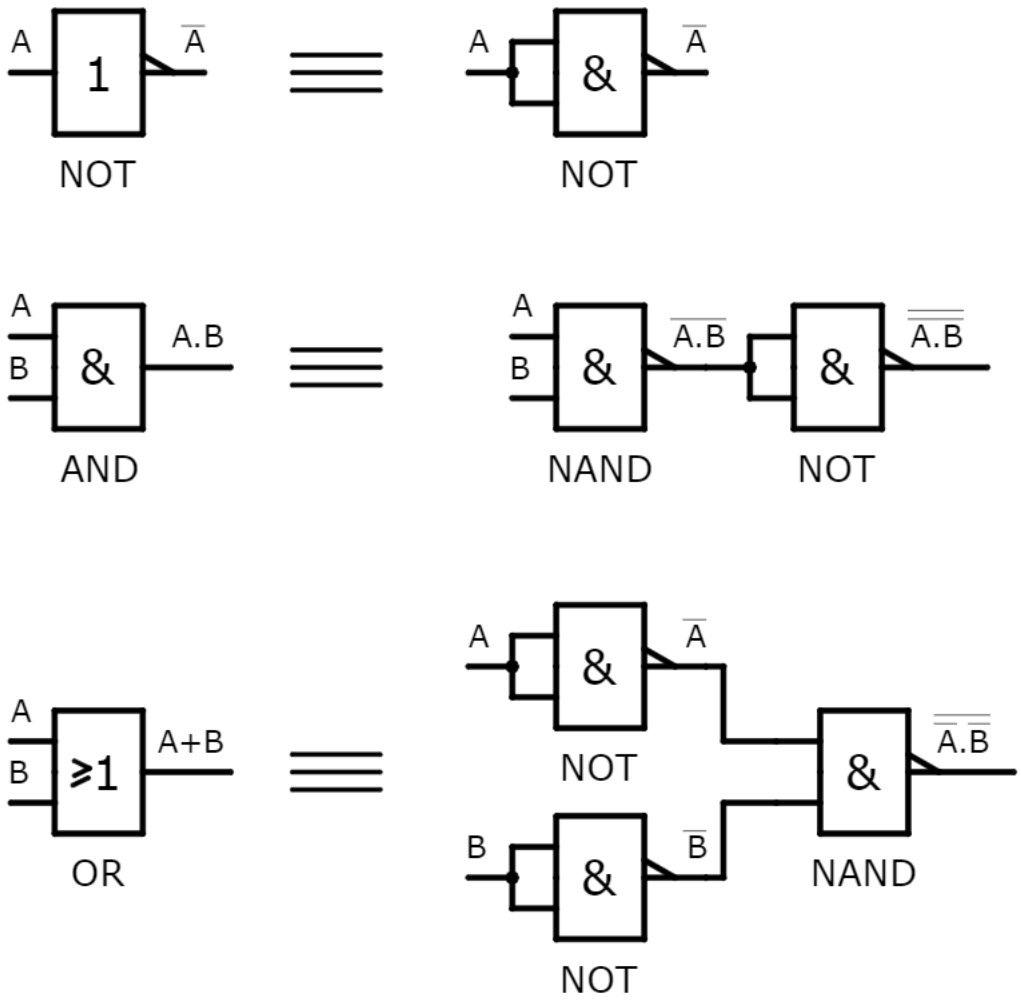
$$\Rightarrow \overline{A+B} = \overline{A} \cdot \overline{B}$$

$$\Leftrightarrow \overline{\overline{A+B}} = \overline{\overline{A} \cdot \overline{B}}$$

$$\Leftrightarrow A+B = \overline{\overline{A} \cdot \overline{B}}$$

3.2.3 Praktisch nut

Omdat we booleaanse functies gemakkelijk kunnen omvormen kiezen fabricanten van geïntegreerde schakelingen (IC's) ervoor om de schakellogica uit te voeren met enkel **NAND**-poorten. Hier hebben ze uiteraard wel goede redenen voor. NAND-technologie is makkelijker te procuceren en dus ook goedkoper. Figuur 3.14 toont de realisatie van de drie basisfuncties NOT, OR en EN met enkel NAND-poorten.

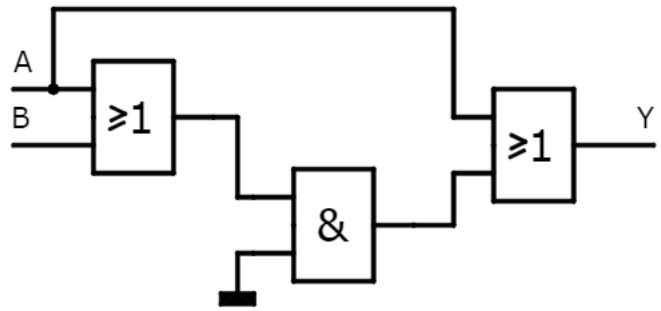


△ Fig.3.14 | De drie basisfuncties door enkel gebruik te maken van NAND-poorten. Bron: W.Van Wichelen

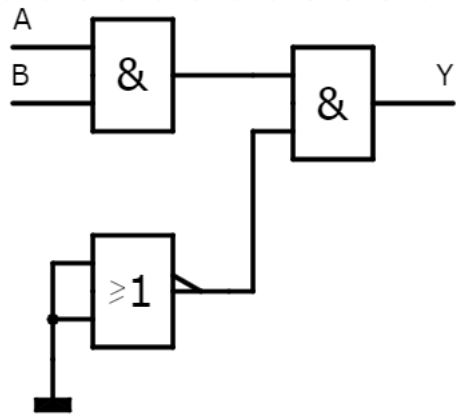
3.2.4 Oefeningen

Reeks 1

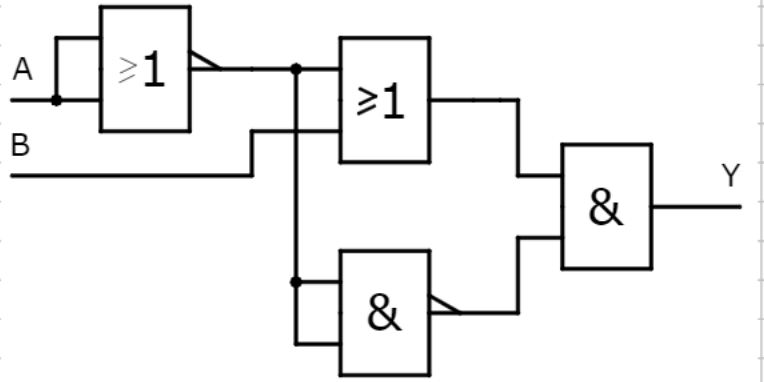
1. Bestudeer onderstaande logische schakeling.
   a. Stel de logische vergelijking op.
   b. Vereenvoudig door gebruik te maken van de basisregels.



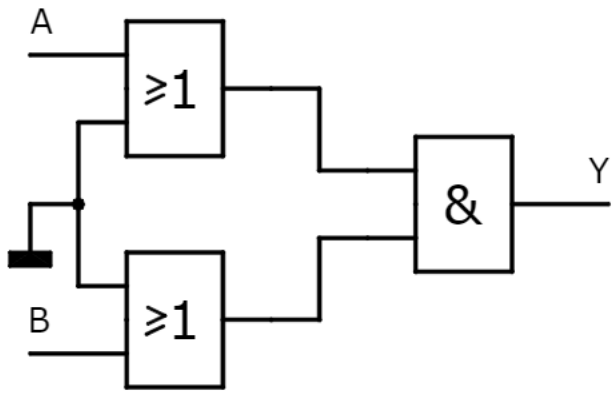
2. Bestudeer onderstaande logische schakeling.
   a. Stel de logische vergelijking op.
   b. Vereenvoudig door gebruik te maken van de basisregels.



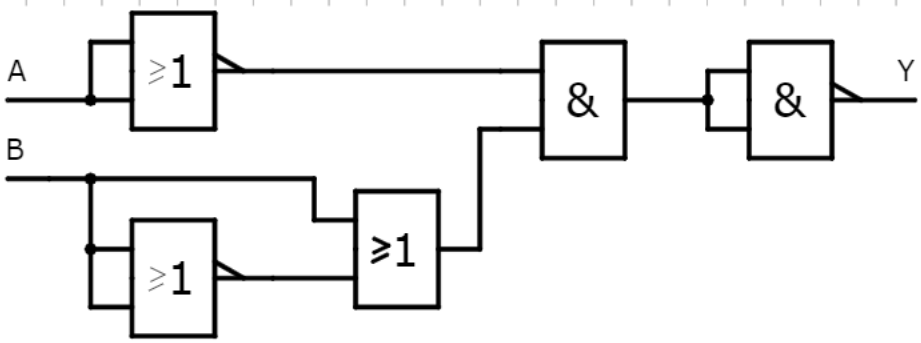
3. Bestudeer onderstaande logische schakeling.
   a. Stel de logische vergelijking op.
   b. Vereenvoudig door gebruik te maken van de basisregels.



4. Bestudeer onderstaande logische schakeling.
   a. Stel de logische vergelijking op.
   b. Vereenvoudig door gebruik te maken van de basisregels.



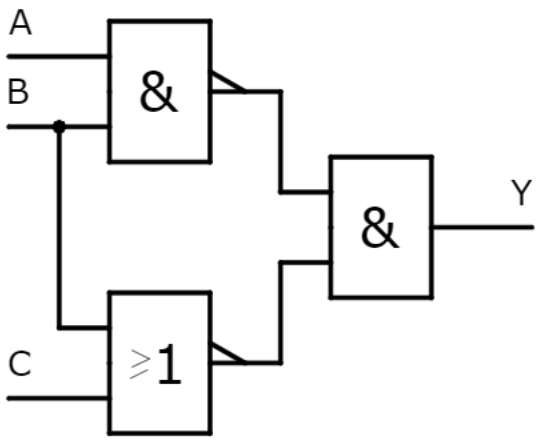
5. Bestudeer onderstaande logische schakeling.
   a. Stel de logische vergelijking op.
   b. Vereenvoudig door gebruik te maken van de basisregels.



Reeks 2

1. Realiseer een digitale schakeling die voldoet aan de vergelijking: $Y = \overline{A} \cdot B + C$. Je mag enkel gebruik maken van het IC 74LS00. Maak gebruik van de stelling van De Morgan.

2. Bestudeer onderstaande logische schakeling.
   a. Stel de logische vergelijking op.
   b. Pas het theorema van De Morgan toe en vereenvoudig de vergelijking.
   c. Teken het vereenvoudigde schema met de basispoorten.
   d. Realiseer de schakeling met 4 TTL poorten van het IC 74LS00.

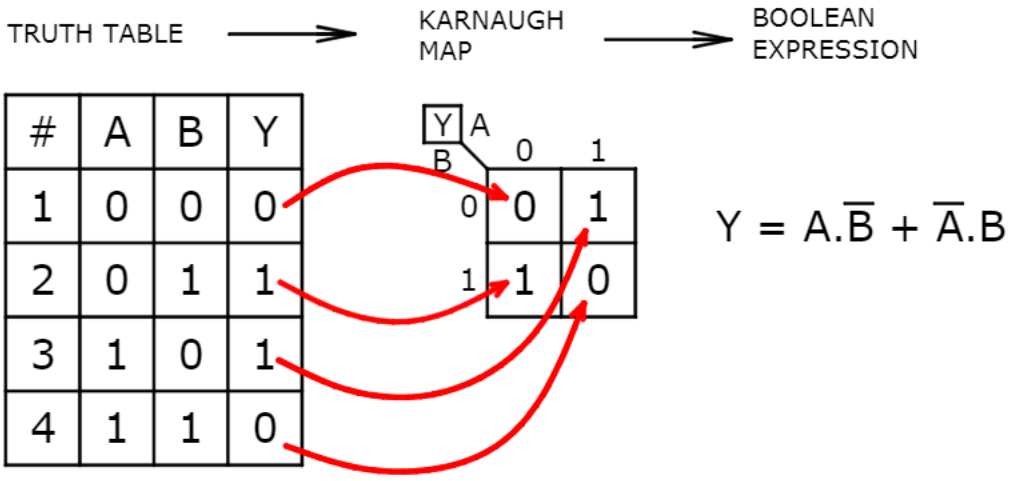


3.3 Karnaugh-diagrammen

Eerder leerden we om logische vergelijkingen te vereenvoudigen met booleaanse algebra. ICT-technici - die niet erg vertrouwd zijn met de regels en theorema's van de Boole-algebra - maken liever gebruik van een *grafische* methode om tot de meest eenvoudige logische uitdrukking te komen. Doordat het menselijk brein gemakkelijk patronen kan herkennen, helpt een zg. **karnaugh-diagram**[13] om snel op te zoeken welke termen gecombineerd kunnen worden. We kunnen nu visueel logische vergelijkingen, met maximaal 6 logische variabelen, minimaliseren. Het stelt ons in staat om digitale logica op te leveren met een minimum aantal elektronische componenten, waardoor de kans op defecte poorten aanzienlijk vermindert. De alzo ontworpen schakelingen bezitten kleinere afmetingen en zijn goedkoper om te produceren.

3.3.1 Karnaugh-diagram voor 2 variabelen

De waarheidstabel toont de status (0 of 1) van de uitgang (Y) voor iedere mogelijke combinatie van de ingangsvariabelen (A en B). Een karnaugh-diagram geeft dezelfde informatie maar iets anders voorgesteld. Elke ingangscombinatie uit de waarheidstabel stemt overeen met een welbepaald vakje in het karnaugh-diagram. Figuur 3.15 illustreert hoe dit in zijn werk gaat.



△ Fig.3.15 Omzetting van een waarheidstabel naar een karnaugh-diagram voor 2 booleaanse variabelen. Bron: W.Van Wichelen

[13] Het karnaugh-diagram werd uitgevonden in 1950 door Maurice Karnaugh, een telecommunicatie-ingenieur bij Bell Labs.

3.3.2 Karnaugh-diagram voor 3 variabelen

#	A	B	C	Y
1	0	0	0	0
2	0	0	1	1
3	0	1	0	0
4	0	1	1	1
5	1	0	0	1
6	1	0	1	1
7	1	1	0	1
8	1	1	1	1

Y / C \ AB	00	01	11	10
0	0	0	1	1
1	1	1	1	1

$$Y = A.B.\overline{C} + A.\overline{B}.\overline{C} + \overline{A}.\overline{B}.C + \overline{A}.B.C + A.B.C + A.\overline{B}.C$$

△ Fig.3.16 Omzetting van een waarheidstabel naar een karnaugh-diagram voor 3 booleaanse variabelen. Bron: W.Van Wichelen

3.3.3 Karnaugh-diagram voor 4 variabelen

#	A	B	C	D	Y
1	0	0	0	0	0
2	0	0	0	1	1
3	0	0	1	0	0
4	0	0	1	1	1
5	0	1	0	0	1
6	0	1	0	1	1
7	0	1	1	0	1
8	0	1	1	1	1
9	1	0	0	0	0
10	1	0	0	1	1
11	1	0	1	0	0
12	1	0	1	1	1
13	1	1	0	0	0
14	1	1	0	1	0
15	1	1	1	0	0
16	1	1	1	1	0

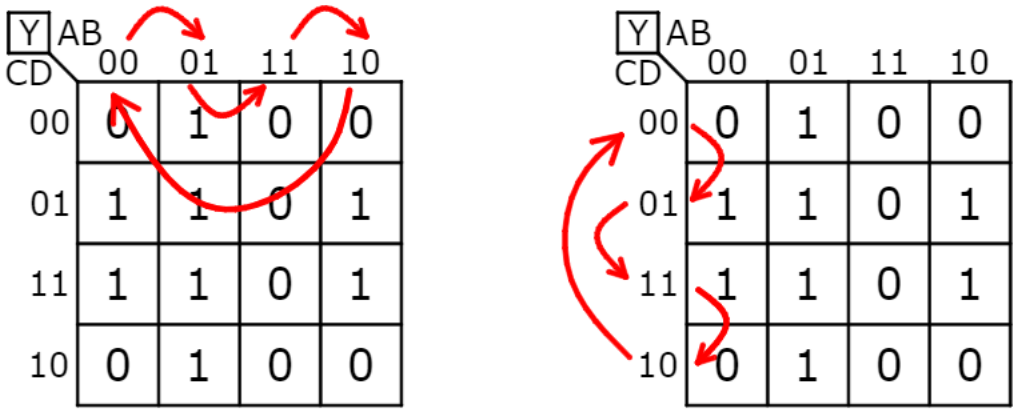
Y / CD \ AB	00	01	11	10
00	0	1	0	0
01	1	1	0	1
11	1	1	0	1
10	0	1	0	0

$$Y = \overline{A}.B.\overline{C}.\overline{D} + \overline{A}.\overline{B}.\overline{C}.D + \overline{A}.B.\overline{C}.D + A.\overline{B}.\overline{C}.D + \overline{A}.\overline{B}.C.D + \overline{A}.B.C.D + A.\overline{B}.C.D + \overline{A}.B.C.\overline{D}$$

△ Fig.3.17 Omzetting van een waarheidstabel naar een karnaugh-diagram voor 4 booleaanse variabelen. Bron: W.Van Wichelen

3.3.4 Regels voor minimalisatie

Wanneer je een karnaugh-diagram van naderbij bekijkt dan merk je dat bij de overgang van twee naast elkaar gelegen vakken er slechts één variabele is die van toestand wijzigt. Ook voor de grensvakken (uiterst links, uiterst rechts, bovenaan en onderaan) is dit het geval. Dankzij deze schikking is het mogelijk om de logische vergelijking te vereenvoudigen of te minimaliseren.



△ Fig.3.18 | Schikking van de vakken in een karnaugh-diagram voor 4 booleaanse variabelen.
Bron: W.Van Wichelen

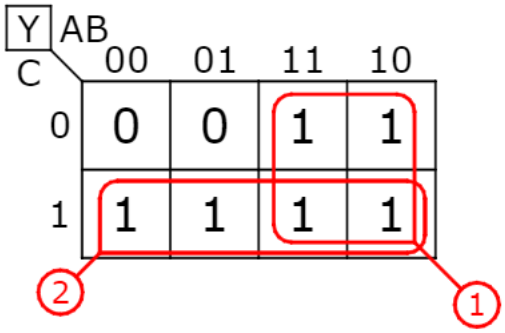
Aangrenzende vakken vormen een zg. **lus**. Het is de bedoeling om zo groot mogelijke lussen te maken in een poging om zoveel mogelijk enen te vangen. Hierbij houden we ons aan volgende regels:

Regels voor minimalisatie

- We maken steeds een zo *groot mogelijke* lus met vakken die **enen** bevatten en we herhalen dit tot *alle* enen in een bepaalde lus zijn opgenomen.
- Elke lus moet een *rechthoek* of een *vierkant* vormen met **1**, **2**, **4** of **8** aangrenzende vakken.
- Noteer enkel de variabelen die *constant* zijn binnen alle vakken in een lus.
- Lussen mogen elkaar *overlappen*.
- Je moet alle enen tenminste één keer aflezen.

3.3.5 Minimalisatie met een karnaugh-map

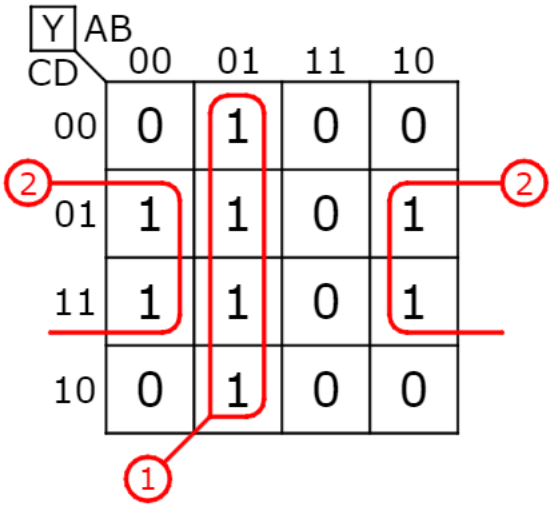
Voorbeeld 1



△ Fig.3.19 | Minimalisatie met een karnaugh-diagram voor 3 booleaanse variabelen. Bron: W.Van Wichelen

Bij het aflezen van lus 1 brengen we enkel de constanten in rekening: $Y_1 = A$. Zo levert lus 2: $Y_2 = C$. Finaal kunnen we schrijven: $Y = A + C$.

Voorbeeld 2



△ Fig.3.20 | Minimalisatie met een karnaugh-diagram voor 4 booleaanse variabelen. Bron: W.Van Wichelen

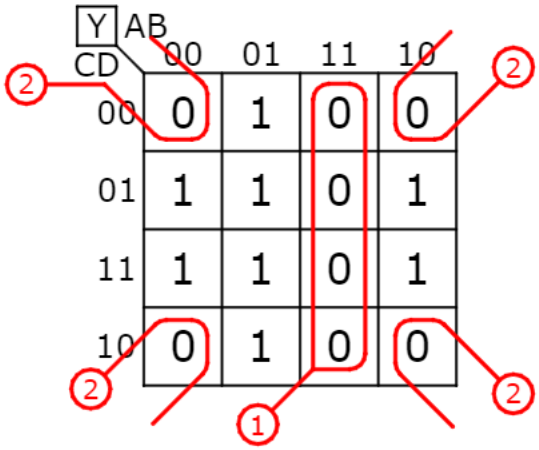
We lezen lus 1 af: $Y_1 = \overline{A} \cdot B$

We lezen lus 2 af: $Y_2 = \overline{B} \cdot D$

De minimale logische vergelijking wordt: $Y = \overline{A}B + \overline{B}D$

3.3.6 Inverse van een functie

Via een karnaugh-diagram is het eenvoudig het inverse van een bepaalde booleaanse uitdrukking te bepalen. Het volstaat om de nullen af te lezen in plaats van de enen. Het voorbeeld van figuur 3.21 maakt dit duidelijk.



△ Fig.3.21 Afleiden van de inverse functie met een karnaugh-diagram voor 4 booleaanse variabelen. Bron: W.Van Wichelen

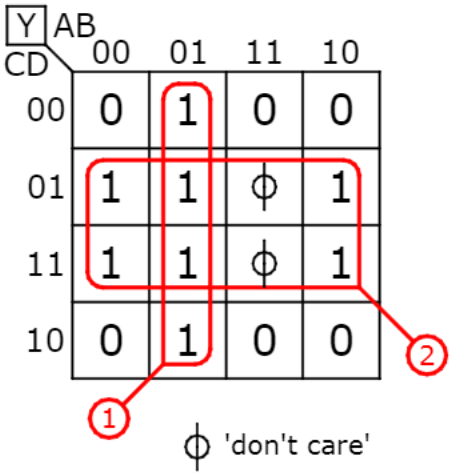
We lezen lus 1 af: $\overline{Y_1} = A \cdot B$

We lezen lus 2 af: $\overline{Y_2} = \overline{B} \cdot \overline{D}$

De minimale inverse logische vergelijking wordt: $\overline{Y} = AB + \overline{B}\overline{D}$

3.3.7 Gebruik van DON'T CARE

Bij het ontwerpen van digitale combinatorische logica kan het voorkomen dat niet alle vakken van een karnaugh-map bepaald zijn. Er kunnen toestanden voorkomen die zowel '0' als '1' mogen zijn. Dit zijn de zg. *DON'T CARE's*. Bij het uitlezen mag je zelf beslissen of een '1' of een '0' plaats in het betreffende vak.



△ Fig.3.22 Uitlezen van een karnaugh-diagram voor 4 booleaanse variabelen met 'don't care's'. Bron: W.Van Wichelen

We lezen lus 1 af: $Y_1 = \overline{A} \cdot B$

We lezen lus 2 af: $Y_2 = D$

De minimale logische vergelijking wordt: $Y = \overline{A} \cdot B + D$

3.3.8 Uitgewerkt voorbeeld

Digitale stemmachine

Een deliberatiecommissie moet beslissen tot het al dan niet geven van een herexamen (Y). De commissie bestaat uit 4 leden: 3 leerkrachten (A, B en C) en de directeur (D). Indien de directeur 'ja' ('1') stemt, dan volstaat nog 1 extra ja-stem om de te delibereren leerling een herexamen te geven. Zegt de directeur 'nee' ('0'), dan moeten alle leerkrachten 'ja' stemmen om onze arme student toch nog zijn herexamen te kunnen geven. Ontwerp een digitale schakeling om de stemming automatisch te laten verlopen.

Waarheidstabel & karnaugh-map

#	A	B	C	D	Y
1	0	0	0	0	0
2	0	0	0	1	0
3	0	0	1	0	0
4	0	0	1	1	1
5	0	1	0	0	0
6	0	1	0	1	1
7	0	1	1	0	0
8	0	1	1	1	1
9	1	0	0	0	0
10	1	0	0	1	1
11	1	0	1	0	0
12	1	0	1	1	1
13	1	1	0	0	0
14	1	1	0	1	1
15	1	1	1	0	1
16	1	1	1	1	1

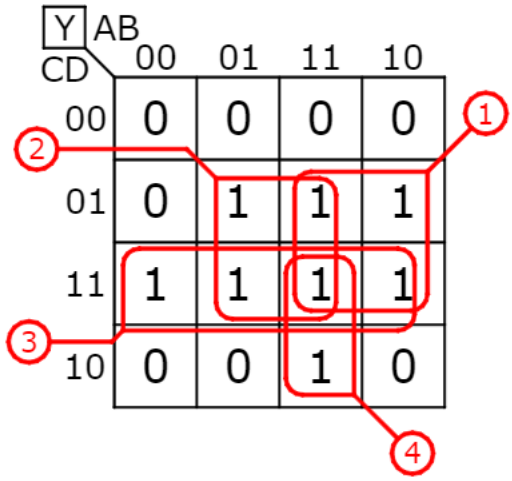
Y / CD \ AB	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	1	1	1	1
10	0	0	1	0

△ Fig.3.23 | Truth table en karnaugh-map van de digitale stemmachine. Bron: W.Van Wichelen

Zonder kennis van vereenvoudigingstechnieken zou de schakeling bestaan uit een OR-poort met 8 ingangen waarbij elke ingang connectie maakt met een AND-poort met 4 ingangen. Uit de waarheidstabel leiden we immers volgende vergelijking af:

$$Y = \overline{A}\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BCD + A\overline{B}\overline{C}D + A\overline{B}CD + AB\overline{C}D + ABC\overline{D} + ABCD$$

Minimalisatie voor Y



△ Fig.3.24 | Aflezen van de enen in de karnaugh-map van de digitale stemmachine. Bron: W.Van Wichelen

We lezen lus 1 af: $Y_1 = A \cdot D$

We lezen lus 2 af: $Y_2 = B \cdot D$

We lezen lus 3 af: $Y_3 = C \cdot D$

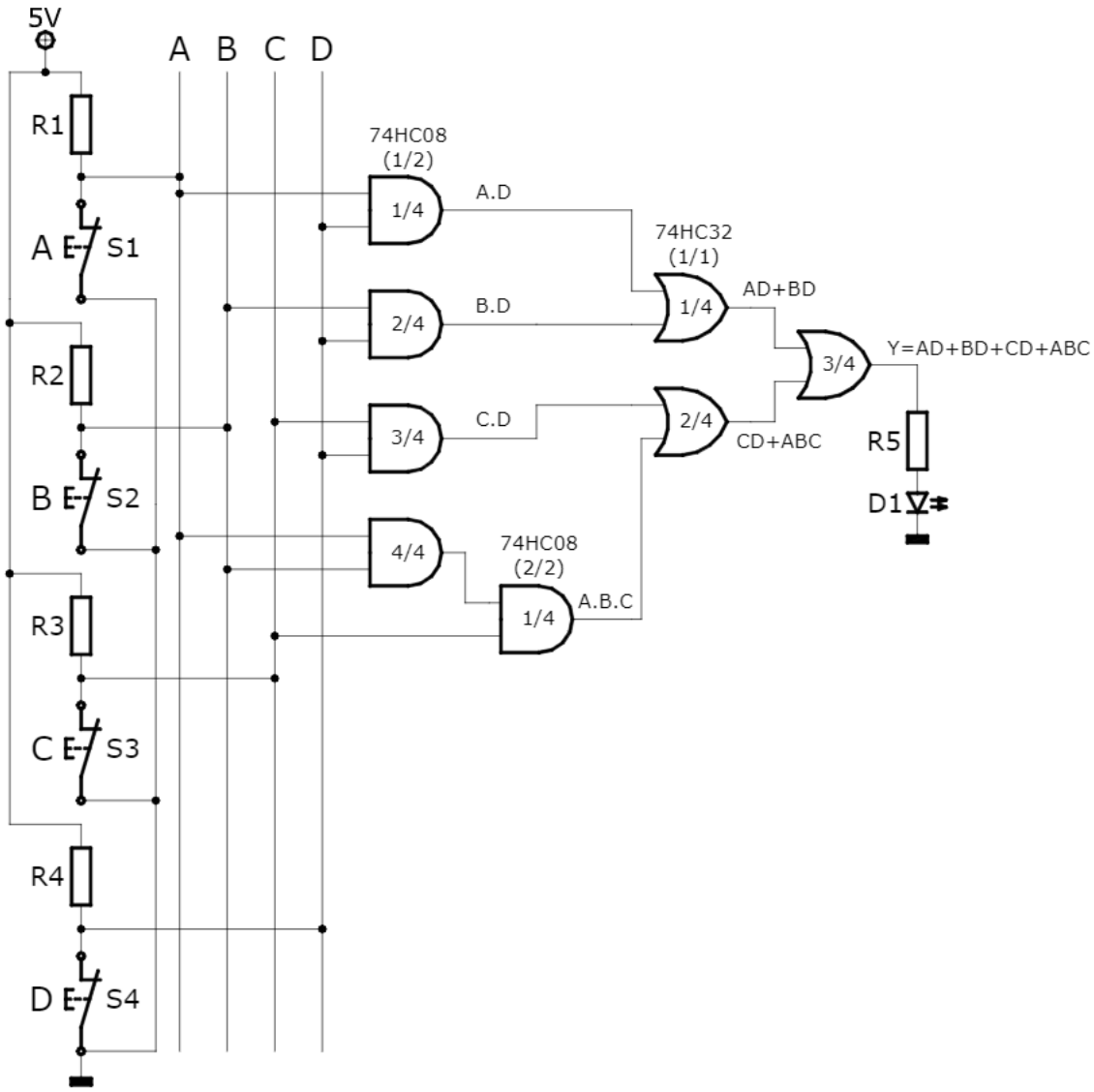
We lezen lus 4 af: $Y_4 = A \cdot B \cdot C$

De minimale logische vergelijking wordt: $Y = AD + BD + CD + ABC$

Laboproef 2

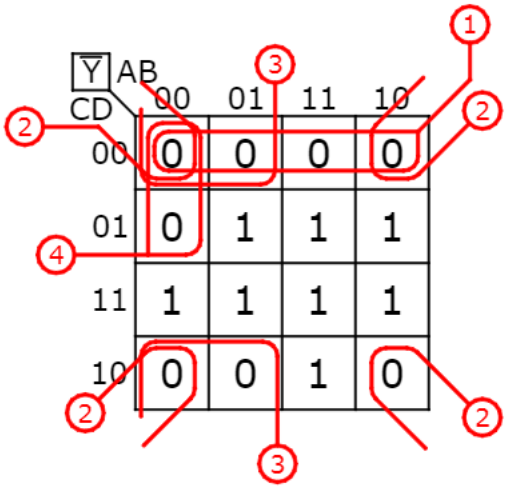
Ontwerp een digitale stemmachine om een deliberatie-commissie te automatiseren. De commissie bestaat uit drie vakleerkrachten (A, B en C) en de directeur (D). Wanneer de directeur zich onthoudt, dan beslist de meerderheid van de leerkrachten tot het geven van een herexamen (Y). Wanneer de directeur meestemt dan is de uitkomst steeds een herexamen. Stel de waarheidstabel op en verifieer ze met metingen. Visualiseer de logische toestand van de uitgang met een blauwe LED. Pas het principe van 'current sinking' toe. Maak gebruik van miniatuur drukknoppen (S1,..S4).

Praktische realisatie - current sourcing



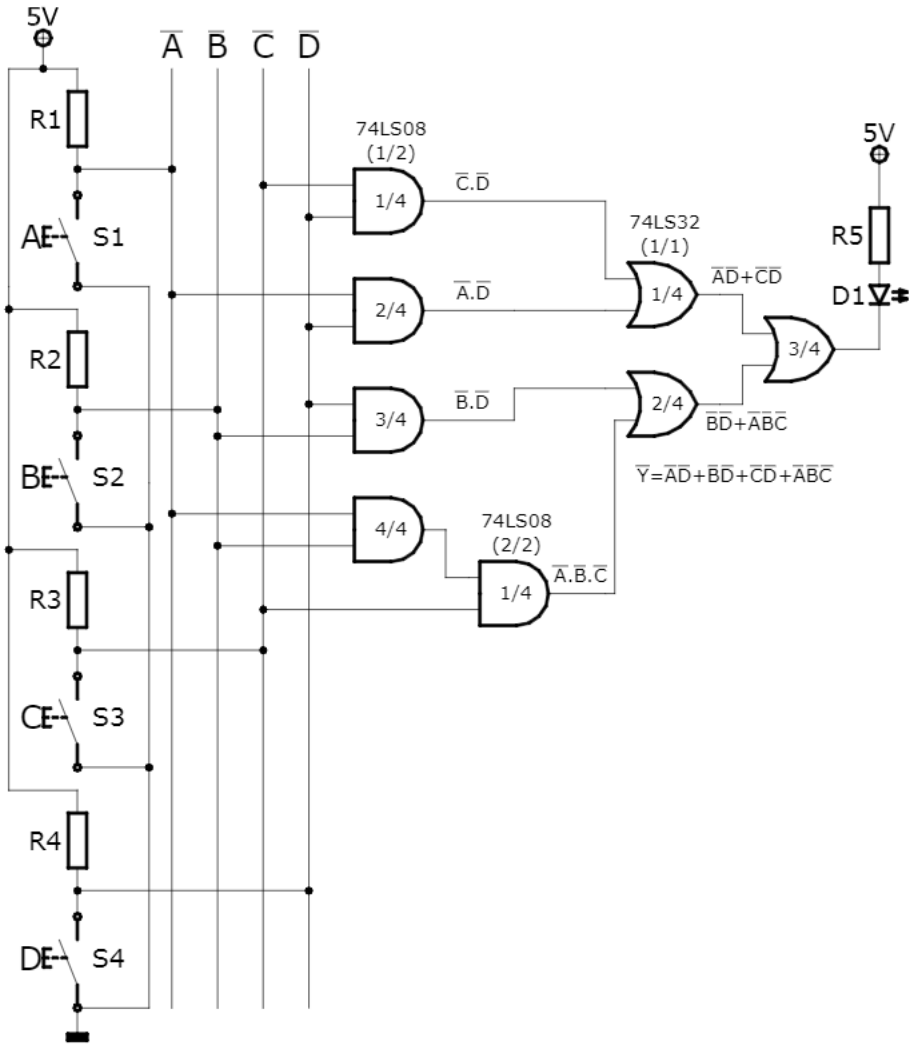
△ Fig.3.25 | Praktische realisatie met CMOS-poorten van de digitale stemmachine. Bron: W.Van Wichelen

Minimalisatie voor inverse functie /Y



△ Fig.3.26 | Aflezen van de nullen in de karnaugh-map van de digitale stemmachine. Bron: W.Van Wichelen

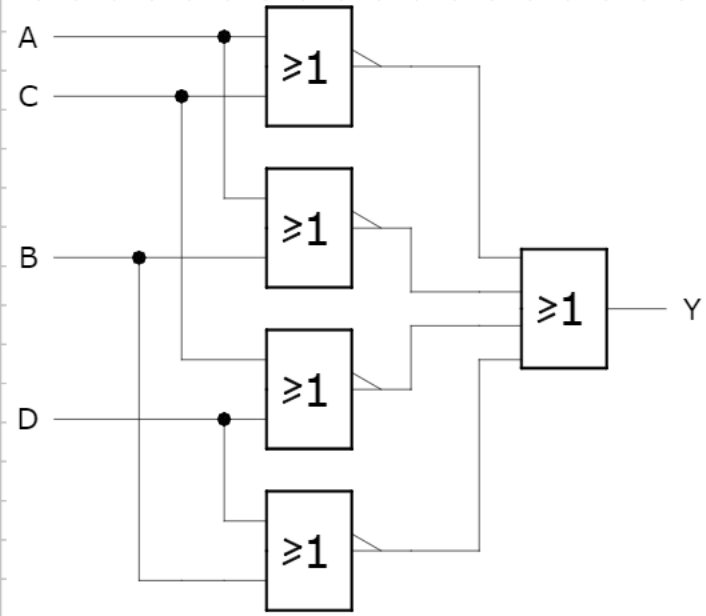
Praktische realisatie - current sinking



△ Fig.3.27 | Praktische realisatie met TTL-poorten van de digitale stemmachine. Bron: W.Van Wichelen

3.3.9 Oefeningen

1. Bestudeer onderstaande logische schakeling.
   a. Stel de waarheidstabel op.
   b. Stel de logische vergelijking op.
   c. Vereenvoudig m.b.v. een karnaugh-map.
   d. Teken de eenvoudigste realisatie.



2. Een toestel (Y) wordt bediend door twee drukknoppen (A en B). Ontwerp een combinatorische NAND-poortschakeling die er voor zorgt dat het toestel enkel en alleen inschakelt als de twee schakelaars een verschillende stand aannemen.
   a. Stel de waarheidstabel op.
   b. Noteer het karnaugh-diagram.
   c. Stel de vergelijking op.
   d. Realiseer de schakeling met enkel NAND-poorten.

3. Twee verbruikers (X en Y) worden bediend door drie schakelaars (A, B en C). Verbruiker X moet werken zodra A en B ingedrukt zijn. Verbruiker Y moet werken zodra A en C ingedrukt zijn. Ontwerp een combinatorische schakeling die enkel gebruik maakt van het IC 74LS00.

   a. Stel de waarheidstabel op.

   b. Noteer het karnaugh-diagram.

   c. Stel de vergelijkingen op.

   d. Realiseer de schakeling met enkel NAND-poorten.

4. Een elektronische thermometer meet de temperatuur van een aquarium. Het uitgangssignaal bestaat uit 4 bits en is de binaire voorstelling van de temperatuur van 0°C tot 15°C. Ontwerp een combinatorische schakeling waarvan de uitgang (Y) hoog wordt voor temperaturen hoger dan 9°C en laag wordt voor temperaturen lager dan 10°C. Maak gebruik van de basispoorten met actief hoge ingangen.

   a. Stel de waarheidstabel op.

   b. Maak een karnaugh-map.

   c. Stel de vergelijking op.

   d. Teken de realisatie met basispoorten.