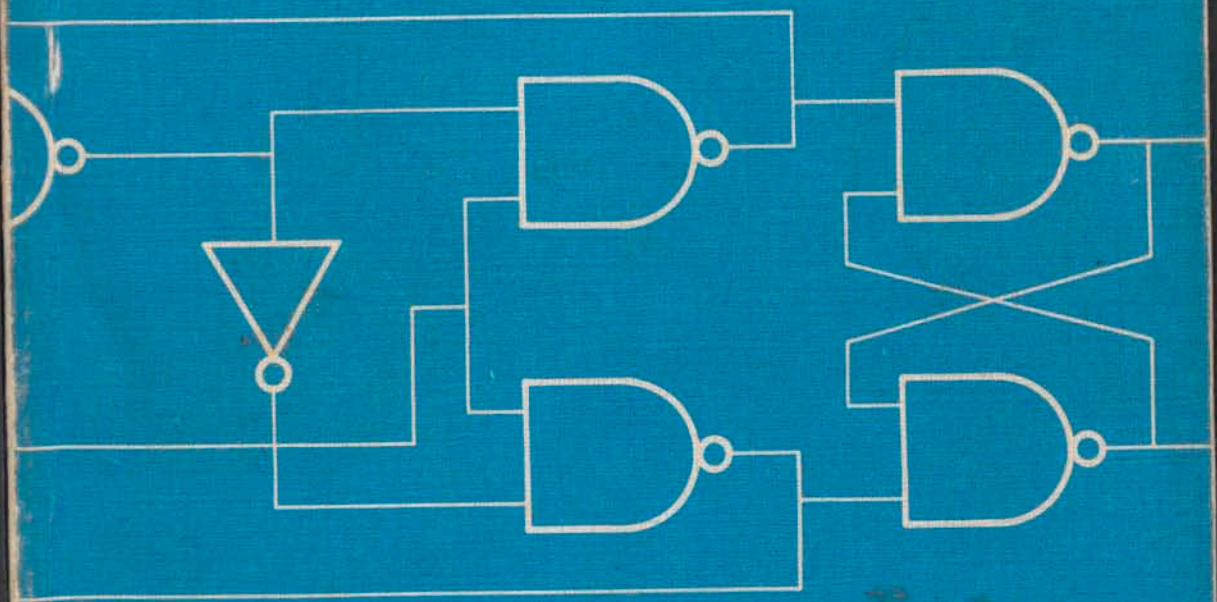
DIGITALE SCHAKELTECHNIEK

van probleemspecificatie tot realisatie

A.P. THIJSSEN, H.A. VINK
met medewerking van
C.H. EVERSDIJK



DEEL 1

DELFTSE UITGEVERS MAATSCHAPPIJ — 2e druk 1981

1673 5047

BIBLIOTHEEK TU Delft
P 1673 5047
C 862943

DIGITALE SCHAKELTECHNIEK

van probleemspecificatie tot realisatie

door ir. A.P. Thijssen en ir. H.A. Vink
met medewerking van lector ir. C.H. Eversdijk

1673 5047

BIBLIOTHEEK DER TECHNISCHE HOGESCHOOL
81
012303
DOELENSTR. 101
DELFT

LL

C10086
29436

Vereniging voor Studie- en Studentenbelangen te Delft
2e, verbeterde druk, 1981

VOORWOORD

Digitale schakeltechniek is lang, wellicht te lang, beschouwd als een vak dat slechts door ervaring kan worden geleerd. Vele leerboeken beperken zich tot een behandeling van de verkrijgbare componenten en tot het bespreken van schakelingen voor tellen, schuiven en andere standaardbewerkingen. Een systematische benadering van het ontwerpen van digitale schakelingen op elementair niveau werd node gemist.
De voor U liggende tekst is gegroeid uit een tweetal colleges aan de Technische Hogeschool te Delft, de colleges "Schakeltechniek Beknopte Cursus" en "Schakeltechniek III, Theorie van de Sequentiële Machines". Op suggestie van prof. dr. ir. R.M.M. Oberman heeft eerstgenoemde auteur zich sinds 1970 toegelegd op de bestudering van de mogelijkheden tot toepassing van de theorie van de sequentiële machines op het ontwerpen van digitale schakelingen. Weldra groeide het inzicht dat bij het onderwijs in de digitale schakeltechniek geen scheiding gemaakt dient te worden tussen de behandeling van de theoretische en praktische schakeltechniek. Een systematisch opgezette probleemspecificatie leidt vrijwel altijd tot een beter ontwerp, terwijl de voor het bou en van een betrouwbaar werkende proefschakeling benodigde tijd drastisch bekort kan worden. Bij de probleemspecificatie speelt de modelvorming een belangrijke rol. Het Moore en het Mealy model van een sequentiële schakeling zijn onmisbare hulpmiddelen, niet alleen bij de behandeling van de "level mode" schakeltechniek maar vooral bij de besturingsspecificatie van "clock mode" schakelingen. In deze tekst wordt veel aandacht besteed aan de probleemspecificatie.
Juist door het specificeren, op systematische wijze uitgevoerd, groeit het inzicht in de problematiek van het betreffende ontwerp. De mathematische relatie "compatibel" speelt een belangrijke rol bij de specificatie. Telkens en bij verschillende fasen van het ontwerp blijkt dat het opsporen van de compatibiliteitsrelatie tussen verschillende objecten, of dit nu besturingstoestanden, instelsignalen of andere grootheden zijn, te leiden tot een beter inzicht in de mogelijkheden om een schakeling voor de gegeven probleemstelling te realiseren.
Kenmerkend bij het beoefenen van de digitale schakeltechniek is het grote aantal varianten bij de oplossing van vrijwel elk probleem. De vrijheid die de ontwerper van logische schakelingen heeft is zeer groot. Een juist gebruik van deze vrijheid wordt vergemakkelijkt door een systematische probleemspecificatie. Daarbij zijn de reeds genoemde mathematische hulpmiddelen onontbeerlijk. Anderzijds is veel vrijheid in een ontwerp ook gevaarlijk. Men ziet gemakkelijk situaties over het hoofd die de juiste werking van een apparaat kunnen verstoren. Het opstellen van een toestandstabel en/of -diagram voor volgordeschakelingen brengt automatisch met zich mee dat over iedere situatie nagedacht wordt. Men komt veel minder in de verleiding bepaalde vooronderstellingen en randvoorwaarden bij een ontwerp niet te verifiëren. Fouten, die anders pas uit praktijkproeven naar voren komen, kunnen in een vroeg stadium van het ontwerp worden hersteld.

Dit boek behandelt de "klassieke schakeltechniek" en is in de eerste plaats bestemd voor degenen die zich beroepshalve met het ontwerpen van digitale apparatuur (gaan) bezighouden. Het niveau is afgestemd op H.T.S.-studenten en studenten van de T.H.'s in de eerste jaren van hun studie. Veel aandacht is besteed aan de didactische opbouw van de stof, zodat het boek ook geschikt is voor zelfstudie.
De behandeling van de diverse onderwerpen is elementair en vergt behalve een geringe kennis van de elektronica vrijwel geen voorkennis van andere vakgebieden. Om praktische redenen is afgezien van een vertaling van de ingeburgerde Engelse termen. In een vakgebied, dat zo sterk beheerst wordt door en berust op Engelstalige literatuur en handboeken, zou dit de toegankelijkheid van de internationale literatuur slechts schaden. Enige basiskennis van de Engelse taal is daarom gewenst, maar niet noodzakelijk. Voor die termen waarvoor een Nederlandse uitdrukking bestaat is deze in de tekst genoemd.
De behandelde stof biedt de noodzakelijke basiskennis om een cursus "microprocessoren" of "computerarchitectuur" met succes te kunnen volgen. Een belangrijk deel van deze stof vindt men in de hoofdstukken over besturingen en organisatie van digitale schakelingen in het tweede deel. De in deze hoofdstukken gepresenteerde stof is basiskennis voor het ontwerpen van logische schakelingen met bit-slice microprocessoren. Bij deze bouwstenen heeft men meer vrijheid in de structuur van een ontwerp dan bij de standaard microprocessoren.

Men zou vraagtekens kunnen plaatsen bij het opgenomen gedeelte over relais en kontaktschakelingen. De aanleiding tot deze twijfel is de overheersende rol die de geïntegreerde circuits spelen in de digitale schakeltechniek. Het weglaten van het gedeelte over kontaktschakelingen zou evenwel onjuist zijn omdat deze schakelingen o.a. in de energietechniek veelvuldig worden toegepast. De in hoofdstuk 3 behandelde stof, tezamen met de grondslagen van de level mode schakeltechniek in de hoofdstukken 6 en 7 maken dit werk ook voor energietechnici een geschikt leerboek. Zij kunnen na bestudering van deze stof in staat geacht worden om eenvoudige vergrendelschakelingen te ontwerpen, ook als deze een sequentieel karakter hebben. Overigens is het inzicht dat andere bouwstenen specifieke problemen met zich meebrengen, zoals sluipwegen bij kontaktschakelingen, op zich reeds een motivatie voor een behandeling.

In deel II worden in de hoofdstukken 11 – 13 algoritmen en schakelingen voor rekenkundige bewerkingen besproken. In de hoofdstukken 14 – 17 staat de organisatie van een digitale schakeling centraal. De behandeling is gebaseerd op de scheiding in datapad en besturing. Hoofdstuk 15 behandelt de specificatie van besturingen en hoofdstuk 16 de realisatie. In hoofdstuk 17 wordt dieper ingegaan op de structuur van digitale schakelingen, de organisatievorm van de schakeling. De laatste hoofdstukken van deel II zijn gewijd aan enkele onderwerpen uit de theorie van de sequentiële machines. Onder andere de compatibiliteitsrelatie, coderingsproblemen en de reductie van toestandstabellen worden besproken en van een mathematische basis voorzien.

Voor een introductiecursus digitale schakeltechniek kunnen enkele meer op de realisatie gerichte onderwerpen worden overgeslagen. Een suggestie voor een selectie uit deel 1 is: H 1 – 10 de volgende par.: 1.1 – 1.3; 2.1 – 2.3/4; 3.1 – 3.3; 4.1 – 4.7; 5.1 – 5.5 en 5.7; 6.1 – 6.2 (tot pag. 133); 6.3 – 6.5; 7.1; 8.1 – 8.4; 8.6 – 8.8; 9.1 – 9.6 en 10.1 – 10.3 en voorts enkele onderwerpen uit deel 2 die ter plaatse worden aangegeven.

De auteurs zijn veel dank verschuldigd aan ir. C.H. Eversdijk, wiens colleges de basis vormen van het voor U liggende werk en aan vele studenten, die via hun examens en taakopdrachten voor een welkome terugkoppeling hebben gezorgd. Mede door hun inbreng is de oorspronkelijke opzet op een aantal punten gewijzigd en aangevuld. De praktijk heeft geleerd dat een actieve beheersing van de hier gepresenteerde stof de tijd, die benodigd is om tot een betrouwbare proefschakeling te komen, drastisch kan bekorten. De tijd die men nodig heeft om zich deze stof eigen te maken wordt snel terugverdiend. Om een actieve beheersing van de stof te vergemakkelijken is een groot aantal opgaven toegevoegd. Sommige ervan gaan wat verder dan de behandelde theorie, of zijn er een aanvulling op. Een bundel met uitwerkingen van de opgaven verschijnt zodra deze gereed is.

De auteurs zijn zich bewust dat velen, direct of indirect, hebben bijgedragen aan het tot stand komen van deze tekst. Behalve aan de reeds genoemden is in het bijzonder dank verschuldigd aan ir. F. Handoko voor zijn medewerking bij het samenstellen van een voorloper van deze tekst, aan ir. R.B. Koolhaas voor zijn adviezen op het gebied van de hardware en aan ir. C.J. van Spronsen met wie deze tekst gebruikt is tijdens een bedrijfscursus in de digitale schakeltechniek.
Mevr. E.H. Wiersum-Bats heeft assistentie verleend bij het typen van de verschillende concepten van deel 1.

Een woord van waardering aan de medewerkers van de "Vereniging voor Studie- en Studentenbelangen te Delft" niet mag ontbreken. Zij zijn het die er in korte tijd in geslaagd zijn een van vele slordige correcties voorzien manuscript om te werken in persklare copy.

Voor opbouwende kritiek en suggesties tot verbetering hebben de auteurs een open oor. Uw suggesties worden met belangstelling afgewacht en zullen verwerkt worden in een naar zij hopen spoedig noodzakelijke tweede druk.

Delft, januari 1979.

A.P. Thijssen
H.A. Vink

INHOUD DEEL I

Inhoud DEEL II

Lijst van symbolen en afkortingen

$\oplus$	exclusieve OF	ζ	verzameling maximale compatibele klassen
+	logische OF		
·	logische EN	λ	uitgangsfunctie
$\bar{}$	aanduiding NIET	$\underline{\lambda}$	uitgebreide uitgangsfunctie
–	don't care	AND	logische EN-poort
$\wedge$	logische EN	BIN	binair
$\vee$	logische OF	BCD	binary coded decimal
$\cap$	doorsnijding	C	aanduiding klokingang
$\cup$	vereniging	C	command input
$\in$	behoort tot	CTn	count = n
=	gelijk aan	CTR	counter
$\neq$	ongelijk aan	D	ingang D(elay) flip-flop
$>$	groter dan	DTL	diode-transistor logica
$\geqslant$	groter of gelijk	d	don't care
$<$	kleiner dan	E_+	hoogste voedingsspanning
$\leqslant$	kleiner of gelijk	E_-	laagste voedingsspanning
$\neg$	uitsteloperator	EXOR	exclusieve-OF poort
$\sim$	compatibel met	F	false
$\not\sim$	incompatibel met	FF	flip-flop
$\rightarrow$	dynamic input	f_i	i-de functiewaarde
$\rightarrow\!\!\circ$	negatie aan ingang	f(x,y,z)	logische functie in x, y en z
$\rightarrow\!\!\mid$	polariteitsindicator	H	hoog
$\langle q,i\rangle$	geordend paar: $q \in Q$ en $i \in I$	HLD	hold
$[i_1 \ldots i_n]$	rij van n ingangssymbolen	I	ingangsverzameling
α	klokpuls	INV	invertor
β	klokpuls	i	ingangscombinatie
Γ_q	verzameling van voor toestand q gespecificeerde ingangswoorden	i	lopende variabele
		J	gespecificeerd ingangswoord
		J	ingang J-K flip-flop
Δt	vertragingstijd	K	ingang J-K flip-flop
Δt_{max}	maximale vertragingstijd	k	lopende variabele
δ	toestandsfunctie	L	laag
$\underline{\delta}$	uitgebreide toestandsfunctie	LD	load
		M_i	maxterm i
		m_i	minterm i

Mn	mode n
mod-2	modulo-2
NAND	NIET-EN poort
NOR	NIET-OR poort
n	variabele
^{n}	index, aanduiding n-de klokpuls
ns	nanoseconde
OR	OF-poort
PLA	programmable logic array
PROM	programmeerbare ROM
Q	toestandsverzameling
Q	uitgang van flip-flop
Q^n	uitgang na n-de klokpuls
q	toestand
R	resetingang
RePROM	herprogrammeerbare PROM
ROM	read-only memory
S	setingang
s	seconde
s_0,s_1	instelingangen van schuifregister
SHL	shift left
SHR	shift right
T	ingang T(oggle) flip-flop
T	periodeduur klokpuls
T	true
TTL	transistor-transistor logica
T_{skew}	effectief beschikbare t_{skew}
t	tijd
t_c	tijd dat signaal constant moet blijven
t_d	propagatietijd
t_h	hold time
t_i	insteltijd combinatoriek
t_p	propagatietijd
t_{PHL}	idem, bij Hoog-Laag overgang
t_{pLH}	idem, bij Laag-Hoog overgang
t_{puls}	pulsduur
t_{skew}	clock skew
U	uitgangsverzameling
$U/\overline{D}$	omschakelingang Op/Neer
V	Volt
x	ingangsvariabele
y	toestandsvariabele
Y	aanduiding geheugenelement
z	oude stand geheugenelement Z
Z	nieuwe stand geheugenelement Z
Z_0	trekker met overheersende reset
Z_1	trekker met overheersende set
Z_z	trekker met extra onthoud stand

INLEIDING

Wat omvat de digitale techniek?

Om ons heen zien we steeds meer digitaal werkende apparatuur verschijnen. Vele soorten van apparaten, die reeds lang in de handel zijn, worden steeds meer in een totaal andere techniek gerealiseerd. Zij worden "gedigitaliseerd". Voorbeelden zijn o.a. digitale klokken en horloges, meters, recorders, parkeermeters en parkeerautomaten, enz. Ook in de automatische telefonie breidt het aantal computergestuurde telefooncentrales zich sterk uit. Een ander voorbeeld is de intrede van de computer in ons bestaan. Door de computer is het menselijk leven zo ingrijpend gewijzigd, dat het niet onwaarschijnlijk is dat verre nageslachten de mens van de 20e eeuw zullen aanduiden als de "homo digitalis".
Om deze explosieve groei van het gebruik van digitaal werkende apparaten te kunnen verklaren moet in de eerste plaats gewezen worden op de techniek van het integreren van schakelingen. Deze techniek, vooral gestimuleerd door de ruimtevaart, heeft het mogelijk gemaakt complexe systemen in kleine afmetingen en tegen redelijke kosten te realiseren. De aanzet tot de groei van de digitale techniek stamt echter van voor de ruimtevaart. Voorbeelden vinden we reeds in de relais-automatieken van telefooncentrales. We kunnen de stormachtige intrede van de digitale technieken in ons bestaan niet geheel verklaren met de stimulans die uitgaat van de ruimtevaart. Er moeten meer redenen zijn die aan deze groei van het aantal toepassingen ten grondslag liggen. Aan de hand van een eenvoudig voorbeeld zullen we trachten er enkele op te sporen.

Voorbeeld. *Het op afstand zichtbaar maken van de stand van een as.*

Het op afstand zichtbaar maken van een bepaald gegeven (in dit geval de stand van een as) is een vaak voorkomend probleem. Van alle mogelijke middelen die ons hierbij ten dienste staan beperken we ons tot die middelen waarbij de benodigde informatie elektrisch wordt overgedragen.
De informatie wordt gerepresenteerd door een elektrische grootheid (stroom of spanning). Dit signaal brengen we via een kabelverbinding over. Op de plaats van bestemming moet de informatie weer zichtbaar gemaakt worden. Een van de mogelijkheden is als uitslag van een meter. Fig. 0.1 toont een (vrij primitieve) oplossing van het probleem.

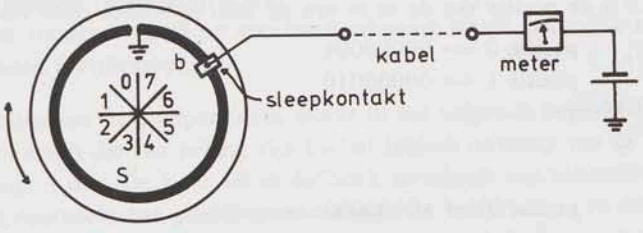


fig. 0.1. Continue positiebepaling met spanningsdeler.

Op de as is een schijf S gemonteerd, waarop een strook weerstandsmateriaal is bevestigd. De strook is aan een zijde geaard. De afstand van de borstel b tot het aardpunt bepaalt mede de totale weerstand in het metercircuit en dus de stroom door de meter. Indien de kabelweerstand gering is ten opzichte van de kringweerstand, dan zal de uitslag van de meter goed overeenkomen met de

positie van de as. De geschetste oplossing is een voorbeeld van een *continue registratiemethode.*

Bezien we deze oplossing van het probleem dan geven we als waardering:
- een ruime voldoende voor de eenvoud van de oplossing.
- een onvoldoende voor de betrouwbaarheid.

De betrouwbaarheid wordt negatief beïnvloed door:
- een directe afhankelijkheid van de batterijspanning.
- de kabelweerstand is meestal niet verwaarloosbaar en bovendien een functie van temperatuur e.d.
- overgangsweerstanden tussen borstel en weerstandsbaan zijn van grote invloed.

In verband met deze afhankelijkheden is een voortdurende ijking van het systeem noodzakelijk.

Fig. 0.2 geeft een oplossing die met betrekking tot de betrouwbaarheid een ruime voldoende scoort. Op de as zit wederom een schijf met daarop een geaarde strook ter lengte van één achtste van de omtrek. Acht vast opgestelde borstels bepalen nu de positie.

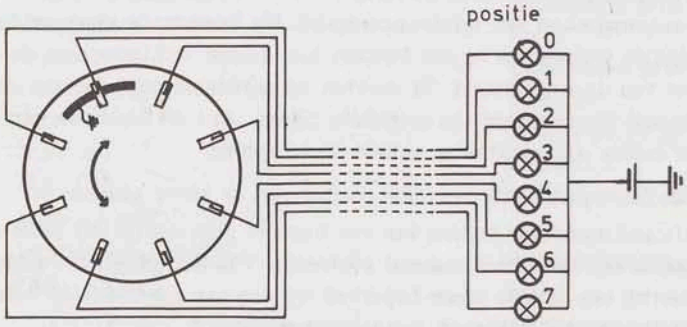


fig. 0.2. Discrete positiebepaling, parallelloplossing.

De bereikbare nauwkeurigheid van de oplossing met vast opgestelde borstels in fig. 0.2 lijkt geringer dan die van fig. 0.1. Dit is echter schijn, omdat het aantal borstels vergroot kan worden. Uiteraard wordt de realisatie daarmee duurder.

In fig. 0.2 is de positie van de as in een zg. *acht-eenheden code* vastgelegd:

positie 0 ↔ 00000001
positie 1 ↔ 00000010
.
.
.
positie 7 ↔ 10000000

Deze codering kan echter veel efficiënter geschieden. De fig. 0.3 en 0.4 geven twee uitvoeringen van een *codeschijf* waarmee de acht posities in een *drie-eenheden code* zijn vastgelegd.

Vergelijken we de oplossing in fig. 0.2 met die uit de figuren 0.3 en 0.4, dan maken beide laatste oplossingen een efficiënter gebruik van o.a. de kabelverbinding. De laatste twee oplossingen gebruiken slechts drie aders. Van deze twee heeft de oplossing in fig. 0.4 verre de voorkeur boven die in fig. 0.3. De reden is de volgende:

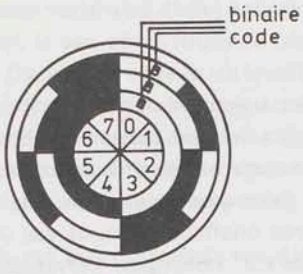


fig. 0.3. Codeschijf in de gewone binaire code.

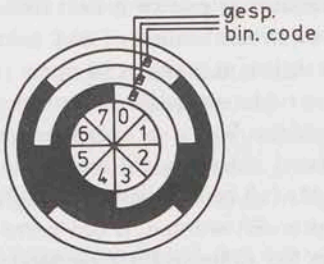


fig. 0.4. Codeschijf in de gespiegeld binaire

– Stel dat de borstels in fig. 0.3 staan in de stand 7, d.w.z. alle drie geaard. Bij verdraaiing naar de stand 0 (geen geaard) kunnen, afhankelijk van de exacte positie van de borstels, een aantal tussenmeldingen gegeven worden. Stel dat de borstels op de buitenste en binnenste ring niet geheel goed staan, zodat de stand 7 [111] overgaat in [011] en via [010] tenslotte in stand 0 [000]. De 1 geeft aan dat de borstel geaard is, de 0 dat de borstel dit niet is.

We zien dat tijdens de overgang van stand 7 naar stand 0 schijnbaar de standen 3 [011] en 2 [010] optreden.

De codeschijf in fig. 0.4 heeft geen last van deze overgangsverschijnselen dank zij de toegepaste *progressieve code*. Tijdens elke overgang verandert slechts op één positie het signaal. We moeten dus altijd kiezen tussen twee naburige posities, en dat is precies het gebied waar de as zich in bevindt.

Het bovenstaande voorbeeld beoogt niet het probleem van de standbepaling van een draaiende as voor eens en altijd op te lossen. Het voorbeeld duidt wel op een van de meest essentiële problemen van de digitale techniek: met betrouwbare componenten behoeft nog niet altijd een betrouwbare schakeling te ontstaan. Een juiste keuze van de systeemopzet is van essentieel belang voor het eindresultaat. Voor een goed ontwerp van een digitale schakeling is nodig:

– kennis van de beschikbare componenten (wat is mogelijk)
– inzicht in de factoren die een correcte werking van de schakeling nadelig kunnen beïnvloeden (storingsbronnen)
– ontwerpmethoden waarbij op voorhand rekening wordt gehouden met storingsbronnen (systeemopzet).

De behandeling van de componenten wordt in het volgende beperkt tot die eigenschappen ervan die van belang zijn bij het logisch ontwerp van de schakeling, de systeemopzet dus. De lezer zal in dit boek tevergeefs een uiteenzetting zoeken over bijvoorbeeld het geleidingsmechanisme in halfgeleiders en andere onderwerpen. Kennis hiervan gebruikt de ontwerper van digitale apparaten niet rechtstreeks bij zijn systeemopzet.

Tweewaardige componenten?

In de digitale techniek worden vrijwel uitsluitend tweewaardige componenten toegepast:

– een transistor geleidt of geleidt niet
– een relais is bekrachtigd of niet bekrachtigd
– een schakelaar is gesloten of open
enz. Het is echter mogelijk de stroom door een transistor over een vrij ruim gebied te variëren. Wat is de reden dat men slechts de twee uitersten gebruikt? Het antwoord moet gezocht worden in de gewenste betrouwbaarheid van het systeem. Als van een transistor uitsluitend de twee uitersten van de geleidingstoestand gebruikt worden, is een grote mate van onafhankelijkheid van de parameters van het element zoals versterkingsfactor e.d. verkregen. Vervanging en afregeling zijn dan uiterst eenvoudig. Door slechts de twee uiterste waarden te benutten is de uitwisselbaarheid van onderdelen optimaal. Men kan als bezwaar opwerpen dat een afrondingsfout wordt geïntroduceerd als men de waarde van grootheden met een continu waardebereik met discrete stappen vastlegt. Vergelijk o.a. het vastleggen van de stand van de as in de figuren 0.1 en 0.2. Echter, de waarde van elke continue grootheid is slechts binnen een bepaalde tolerantie bekend. Zo zijn variaties in de batterijspanning in fig. 0.1 van directe invloed op de nauwkeurigheid. De positiebepaling in fig. 0.2 kan dus even nauwkeurig geschieden als in fig. 0.1 indien men het aantal segmenten aan de omtrek groot genoeg maakt. Nu geschiedt de verdere verwerking van continue signalen altijd met een zekere onnauwkeurigheid. De verwerking van discrete grootheden in digitale schakelingen kan met een willekeurig te kiezen nauwkeurigheid worden uitgevoerd. Uiteraard kost een nauwkeurige verwerking van gegevens in een digitale schakeling meer tijd en/of materiaal.

Binaire talstelsel

In de digitale schakeltechniek worden om bovengenoemde redenen vrijwel uitsluitend tweewaardige componenten toegepast. De ingangs- en uitgangssignalen van deze componenten worden daarom *binaire signalen* genoemd.
Om meer dan twee niveaus van een continu signaal te kunnen onderscheiden zijn dus verscheidene tweewaardige signalen nodig. Aan elk signaalniveau wordt één (of soms meer dan één) combinatie van waarden van de tweewaardige signalen toegekend. Dat deze toekenning niet altijd probleemloos is leert ons een vergelijking van de fig. 0.3 en 0.4. Algemeen geldt dat met k binaire signalen of variabelen ten hoogste 2^k niveaus kunnen worden onderscheiden. Zo kunnen de decimale getallen 0 t/m 7 met drie binaire variabelen (bits) worden gecodeerd. Een van de meest gebruikte codes staat in tabel 0.1, het is de *gewone binaire code.*

getal/ combinatie	gewicht 2^2 2^1 2^0
0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1

tabel 0.1. Gewone binaire code.

De binaire variabelen (bits) krijgen van rechts naar links een gewicht toegekend dat gelijk is aan opeenvolgende machten van 2, *het grondtal van het binaire talstelsel.* De binaire code is uitbreidbaar tot een willekeurig aantal bits. Het corresponderende decimale getal wordt gevonden als die machten van twee bij elkaar opgeteld worden waarvoor een 1 genoteerd is in de desbetreffende kolom. Afhankelijk van bepaalde toepassingen worden ook andere codes gebruikt. Een voorbeeld hiervan is de toepassing van de *gespiegeld binaire code* in fig. 0.4.

De taak van een ontwerper van digitale schakelingen begint met het opstellen van een specificatie van de te realiseren schakeling. In deze specificatie wordt de gewenste werking vastgelegd. Vervolgens worden de realisatiemogelijkheden onderzocht, waarbij de beschikbaarheid van de bouwstenen een rol speelt. Ook aspecten zoals het afwegen van serie-oplossingen (veel tijd) versus parallel-oplossingen (veel materiaal) behoren hiertoe. Tenslotte de bouw van de schakeling, waarbij de fysische eigenschappen van de componenten een rol spelen. Hulpmiddelen, zoals de schakelalgebra en gedeelten van de theorie der sequentiële schakelingen, komen nog uitvoerig aan de orde. Zo ook de beperkingen van deze hulpmiddelen.

1. SCHAKELALGEBRA, EEN ALGEBRA MET NULLEN EN ENEN

1.1. Digitale schakelingen en symbolische logica

Ieder ontwerp van een digitale schakeling begint met het omschrijven van de gewenste werking. Zo'n eerste beschrijving zal als regel vaag zijn, terwijl ook het probleem meestal verre van compleet is onderzocht. In een aantal gesprekken tussen opdrachtgever en ontwerper ontstaat een meer gedetailleerd beeld van de bedoeling van de opdrachtgever. De ontwerper van digitale schakelingen heeft tot taak alle wensen van de opdrachtgever zo te omschrijven dat het mogelijk is een apparaat te bouwen, met tweewaardige bouwstenen, dat aan de bedoeling van de opdrachtgever tegemoet komt.
Wat kan men nu met één binaire (= tweewaardige) variabele beschrijven?
Is het bijvoorbeeld mogelijk met een binaire variabele A aan te geven of een voorwerp rood of groen is? Men kan stellen met A = 0 $\leftrightarrow$ rood en met A = 1 $\leftrightarrow$ groen de juiste kleur vast te leggen. Deze 0 en 1 corresponderen met de twee waarden die een binaire variabele kan aannemen. De gestelde vraag is hiermee op het eerste gezicht bevestigend beantwoord.

Het kan echter ook gebeuren dat bij onderzoek van het voorwerp de kleur oranje blijkt te zijn, dus noch rood noch groen. Er bestaat in dit geval een derde mogelijkheid, nl. geen van beide genoemde kleuren.

Conclusie.
Met één binaire variabele kan niet worden aangegeven of de kleur rood of groen is, indien andere kleuren niet zijn uitgesloten. Kenmerken die zich beter lenen om met één *binaire variabele* te worden vastgelegd zijn o.a.:

rood – niet rood
raam open – raam dicht
spanning hoog – spanning laag (precies twee niveaus mogelijk)
kontakt open – kontakt gesloten.

Sommige begrippen of kenmerken kunnen niet worden beschreven met binaire grootheden, zoals de intuïtieve begrippen: ongeveer, bijna, misschien, e.d.
Ook geldt dit voor de veel voorkomende overgangsverschijnselen in de techniek, i.h.a. *dynamische toestanden* genoemd. Het gedrag in de *statische toestanden* kan echter wel met binaire grootheden worden beschreven.

Definitie.
Een *propositie* is een uitspraak die precies twee waarden kan aannemen, welke meestal worden aangeduid met "**waar**" en "**niet waar**" ("**true**" en "**false**").

In het algemeen kan men stellen dat slechts die eigenschappen, kenmerken of verschijnselen met binaire grootheden kunnen worden beschreven waarop één of meer proposities "passen".
Dat wil zeggen dat de van belang zijnde kenmerken in proposities kunnen worden uitgedrukt. Meestal wordt het waar zijn van een propositie aangegeven met T (true), het niet waar zijn met F (false). In de digitale schakeltechniek gebruikt men vaak de logische 1 (waar) en de logische 0 (niet waar). Staat Z voor de uitspraak

Z: "Het regent"

dan betekent Z = T of Z = 1 dat het regent en Z = F of Z = 0 dat het niet regent. De logische 0 en 1 geven de *waarheidswaarde* van de betreffende propositie aan.

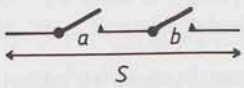
De begrippen propositie en waarheidswaarde stammen uit de *symbolische logica.* Als een belangrijke publicatie op dit gebied wordt beschouwd

"An Investigation of the Laws of Thought"

van George Boole in 1854. Naar hem wordt de nog te behandelen schakelalgebra wel de Boole algebra genoemd. Een wiskundige verstaat onder een Boole algebra echter wel iets meer dan de simpele collectie rekenregels die een schakeltechnicus eronder verstaat!

Bewerkingen in de symbolische logica

In fig. 1.1 is een *serieschakeling* van twee *maakkontakten* a en b getekend. De functie van de schakeling wordt vastgelegd met de naast de figuur gegeven uitspraken A, B en S.



A: "Het kontakt a geleidt"
B: "Het kontakt b geleidt"
S: "De keten geleidt"

fig. 1.1. Serieschakeling van kontakten.

Tussen de waarheidswaarden van A en B enerzijds en S anderzijds bestaat een direct verband. Dit verband is gegeven in de tabel 1.1 via T en F en in tabel 1.2 met 1 en 0. Tabellen, die het verband tussen de waarheidswaarde van verschillende uitspraken vastleggen, worden *waarheidstabellen* genoemd (*truth table*).

A	B	S
F	F	F
F	T	F
T	F	F
T	T	T

tabel 1.1. De operatie EN in waarheidswaarden.

F ↔ 0
T ↔ 1

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

tabel 1.2. De operatie EN in 0 en 1.

Het verband tussen de uitspraken A en B en S kan ook in een formule worden uitgedrukt via de *logische operatie* EN:

S = A EN B.

De operatie EN drukt uit dat S slechts waar is als zowel A als B waar is. We kunnen de tabellen 1.1 en 1.2 dus ook zien als een definitie van de logische EN. De operatie EN wordt met verschillende symbolen aangeduid in de literatuur:

EN, ∧, ·, *.

In deze tekst houden we ons aan de punt.

Behalve de logische operatie EN kennen we ook de logische operaties OF en NIET. Van de operatie OF zijn twee verschillende definities in omloop, de zg. "inclusieve OF" en de "exclusieve OF", gegeven in de tabellen 1.3 en 1.4.

A	B	S
F	F	F
F	T	T
T	F	T
T	T	T

$S = A + B$

tabel 1.3. De inclusieve OF.

A	B	S
F	F	F
F	T	T
T	F	T
T	T	F

$S = A \oplus B$

tabel 1.4. De exclusieve OF.

In het volgende wordt onder de operatie OF altijd de inclusieve OF verstaan, tenzij uitdrukkelijk is vermeld dat het de exclusieve OF betreft. Men komt de exclusieve OF niet vaak tegen. De inclusieve OF wordt vaak aangeduid met de symbolen

OF, v, +

de exclusieve OF met

EX-OF, $\oplus$.

In deze tekst is gekozen voor het "+" teken en het teken "$\oplus$".

De operatie NIET dient om de ontkenning van een uitspraak A aan te geven. Mogelijke schrijfwijzen zijn:

NIET A, $\bar{A}$, $\neg A$ en $\sim A$.

De operatie NIET wordt in deze tekst als regel met een streep boven de uitspraak aangegeven. Tabellen 1.5 en 1.6 geven het verband tussen de proposities A en $\bar{A}$ (lees: niet A):

A	$\bar{A}$
F	T
T	F

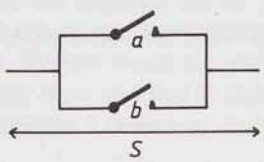
$0 \leftrightarrow F$
$1 \leftrightarrow T$

tabel 1.5. De operatie NIET in waarheidswaarden.

A	$\bar{A}$
0	1
1	0

tabel 1.6. De operatie NIET in 0 en 1.

Voorbeeld.
In fig. 1.2 is de *parallelschakeling* van twee kontakten a en b getekend. De logische functie wordt weer gekarakteriseerd door de proposities A, B en S zoals deze reeds zijn geïntroduceerd bij de serieschakeling van kontakten.



A: "Het kontakt a geleidt"
B: "Het kontakt b geleidt"
S: "De keten geleidt"

fig. 1.2. Parallelschakeling van kontakten.

Voor de parallelschakeling van kontakten bestaat een verband tussen de uitspraken A, B en S, zoals dit in de tabellen 1.7 en 1.8 is gegeven. Een vergelijking met tabel 1.3 leert dat de parallelschakeling van kontakten gezien kan worden als een realisatie van de logische operatie OF.

A	B	S
F	F	F
F	T	T
T	F	T
T	T	T

$0 \leftrightarrow F$
$1 \leftrightarrow T$

tabel 1.7. De operatie OF in waarheidswaarden.

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

tabel 1.8. De operatie OF in 0 en 1.

Voorbeeld. *Specificatie van een druktoets.*
De functie van een druktoets kan men beschrijven met de volgende twee proposities:

D: "De toets is ingedrukt"
G: "Het kontakt geleidt".

Is het verband tussen de waarheidswaarden van D en G als volgt (in nullen en enen):

$$D = 0 \leftrightarrow G = 0 \quad \text{en} \quad D = 1 \leftrightarrow G = 1$$

dan heeft drukken tot gevolg dat er een geleidende verbinding ontstaat.
Het betreffende kontakt wordt een *maakkontakt* genoemd.
Blijkt het verband echter te zijn:

$$D = 0 \leftrightarrow G = 1 \quad \text{en} \quad D = 1 \leftrightarrow G = 0$$

dan noemen we het kontakt een *verbreekkontakt.*

Voorbeeld. *Toekenning van waarheidswaarden.*
Van een elektronische bouwsteen is gegeven dat de in- en uitgangsspanningen twee niveaus kunnen aannemen, aangeduid met L en H (laag en hoog). Het verband tussen de twee ingangsspanningen Y en Z en de uitgangsspanning S is gegeven in tabel 1.9.

Y	Z	S
L	L	L
L	H	H
H	L	H
H	H	H

tabel 1.9. Functiebeschrijving

$L \leftrightarrow F$
$H \leftrightarrow T$

Y	Z	S
F	F	F
F	T	T
T	F	T
T	T	T

tabel 1.10. Waarheidstabel

$L \leftrightarrow T$
$H \leftrightarrow F$

Y	Z	S
F	F	F
F	T	F
T	F	F
T	T	T

tabel 1.11. Waarheidstabel

Spreken we af dat een lage uitgangsspanning overeenkomt met de waarheidswaarde F en een hoge met T, dan is tabel 1.10 de corresponderende waarheidstabel. Een vergelijking leert dat de schakeling de logische OF realiseert. Correspondeert daarentegen L (laag) met T (true), dan behoort tabel 1.11 bij tabel 1.9. Deze tabel beschrijft echter de logische EN.

Conclusie
Dezelfde schakeling kan soms meer dan één logische bewerking uitvoeren, afhankelijk van de afgesproken toekenning van T en F (of van 0 en 1). Het is daarom van essentieel belang vaste afspraken te maken over de interpretatie van logische spanningsniveaus. We komen op deze problematiek nog uitgebreid terug.

De toewijzing van binaire variabelen

Op relais kunnen maakkontakten, verbreekkontakten en/of wisselkontakten gemonteerd zijn. Meestal geven we in tekeningen het relais met een hoofdletter aan en de kontakten met kleine letters. Teneinde het verband tussen de bekrachtigingstoestand van het relais en het geleidend zijn van een erdoor bediend kontakt zo goed mogelijk te laten uitkomen, noteren we bij een maakkontakt van een relais Z de binaire variabele z en bij een verbreekkontakt de binaire variabele $\bar{z}$. Zie fig. 1.3.

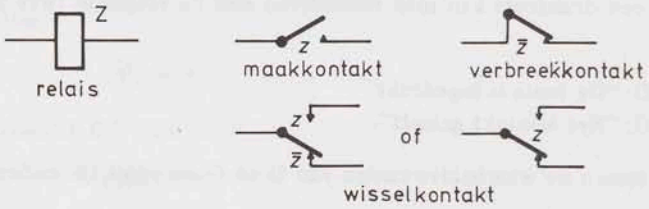


fig. 1.3. Toekenning van logische variabelen aan kontakten.

Bij een wisselkontakt moet aan maakzijde z en aan verbreekzijde $\bar{z}$ worden genoteerd. Meestal volstaat men met z in het midden. De notatie $\bar{z}$ bij het verbreekkontakt heeft als voordeel dat de belettering in de formule niet afwijkt van die in de tekening.
Voor elektrotechnische tekeningen is de volgende afspraak van toepassing:

> Kontakten van relais worden in tekeningen getekend in de stand waarin zij staan als het bijbehorende relais niet is bekrachtigd. Bij druktoetsen is dit de stand waarin het kontakt staat als er niet gedrukt wordt. Van schakelaars met twee stabiele standen wordt de ruststand getekend; welke de rust- en welke de werkstand is wordt bij afspraak vastgelegd.

Opmerking
In de praktijk kan deze tekenregel ertoe leiden dat verschillende kontakten in een stand getekend staan waarin zij in het betreffende apparaat gedurende de werkcyclus nooit zullen staan.

Het volgende voorbeeld toont een probleem, dat via een uitdrukking in de geintroduceerde logische operatoren kan worden geformuleerd.

Voorbeeld
Iemand moet boodschappen doen en heeft op zijn boodschappenlijstje staan: boter (t), suiker (u), aspirine (v), drop (w), koekjes (x), brood (y) en vleeswaren (z). Tabel 1.12 geeft een overzicht waar de diverse artikelen verkocht worden.

		t	u	v	w	x	y	z
Kruidenier	A	×	×		×	×	×	×
Drogist	B			×	×			
Melkman	C	×			×	×		×
Bakker	D					×	×	
Slager	E							×

× = verkrijgbaar

tabel 1.12. Overzicht verkrijgbaarheid van boodschappen.

Zij S een binaire grootheid die voorstelt de propositie

S: "Boodschappenlijstje afgewerkt".

Om aan boter (t) te komen moet een bezoek worden gebracht aan de kruidenier of aan de melkman. Dit is in een formule uit te drukken:

$$S_{boter} = (A + C)$$

waarbij A aangeeft:

A: "Bezoek de kruidenier"
C: "Bezoek de melkman".

Evenzo moet suiker worden gekocht:

$$S_{boter \wedge suiker} = (A + C) \cdot A.$$

Alle boodschappen kunnen worden gekocht als de volgende uitdrukking waar is:

$$S = (A + C) \cdot A \cdot B \cdot (A + B + C) \cdot (A + C + D) \cdot (A + D) \cdot (A + C + E).$$

Uit tabel 1.12 vindt men bij nadere beschouwing weldra dat het voldoende is om zowel de kruidenier als de drogist een bezoek te brengen. Inderdaad blijkt bij invullen van "A is waar" en "B is waar" in de formule voor S ook S waar te zijn. De boodschappenlijst kan dus ook worden afgewerkt als

$$S = A \cdot B$$

is. Het stelsel rekenregels waarmee de formule voor S teruggebracht kan worden tot de laatst gegeven vorm wordt in de nu volgende paragraaf behandeld.

1.2. De schakelalgebra

In fig. 1.4.a en fig. 1.4.b zijn twee kontaktschakelingen getekend waarvan de werking in tabel 1.13 is beschreven.
Beide schakelingen leiden tot dezelfde tabel, en worden daarom *logisch equivalent* genoemd. Wordt gelet op overgangsverschijnselen, dan kunnen logisch equivalente schakelingen echter wel degelijk in gedrag verschillen. Niet echter in de statische toestand.
Omdat de beide formules tot dezelfde waarheidstabel leiden, moet het mogelijk zijn om ze met bepaalde rekenregels in elkaar om te rekenen:

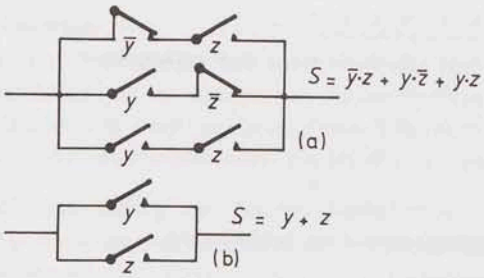


fig. 1.4. Equivalente schakelingen.

y	z	S
0	0	0
0	1	1
1	0	1
1	1	1

tabel 1.13.
Waarheidstabel.

$$
\begin{aligned}
S &= \bar{y}\cdot z + y\cdot\bar{z} + y\cdot z && \\
&= \bar{y}\cdot z + y\cdot z + y\cdot\bar{z} && \text{mits verandering van volgorde mag} \\
&= \bar{y}\cdot z + y\cdot z + y\cdot z + y\cdot\bar{z} && \text{mits } y\cdot z = y\cdot z + y\cdot z \\
&= (\bar{y} + y)\cdot z + y\cdot(z + \bar{z}) && \text{mits o.a. } \bar{y}\cdot z + y\cdot z = (\bar{y} + y)\cdot z \\
&= 1\cdot z + y\cdot 1 && \text{mits } \bar{y} + y = 1 \text{ en } z + \bar{z} = 1 \\
&= y + z && \text{mits } 1\cdot y = y.
\end{aligned}
$$

Achter elke regel in bovenstaande afleiding is vermeld onder welke voorwaarde(n) de betreffende stap is toegestaan. (De 1 in een formule duidt bij kontaktschakelingen aan dat de betreffende keten geleidt.)
Onderzoeken we de boven vermelde voorwaarden voor kontaktschakelingen dan blijkt dat daaraan altijd wordt voldaan. Hetzelfde zal blijken voor de nog te behandelen elektronische of mechanische realisaties van logische bewerkingen en functies.
Het stelsel rekenregels volgens welke bepaalde formules in andere (meestal eenvoudigere) formules kunnen worden omgerekend, wordt aangeduid met de naam *schakelalgebra*. Het volgende stelsel blijkt in de praktijk prettig hanteerbaar.

Schakelalgebra

Volgorde van bewerkingen
In de schakelalgebra wordt als volgorde voor de verschillende logische bewerkingen aangehouden:
1. Voer eerst alle inversies uit
2. Daarna alle logische bewerkingen EN, aangeduid met ·
3. Tenslotte de bewerkingen OF, aangeduid met +.

Deze volgorde kan worden gewijzigd met haken, zoals dit in de normale algebra ook geschiedt. Alle logische formules moeten volgens deze regels worden geïnterpreteerd.

Rekenregels voor constanten
Voor de constanten 0 en 1 gelden de volgende rekenregels:

$$
\begin{aligned}
0 + 0 &= 0 \qquad & 0\cdot 0 &= 0 & & \\
0 + 1 &= 1 \qquad & 0\cdot 1 &= 0 & \bar{0} &= 1 \\
1 + 0 &= 1 \qquad & 1\cdot 0 &= 0 & \bar{1} &= 0 \\
1 + 1 &= 1 \qquad & 1\cdot 1 &= 1 &
\end{aligned}
$$

De 0 en 1 zijn de enige constanten die voorkomen in de schakelalgebra. Zij komen overeen met "niet waar" en "waar" uit de symbolische logica.

Rekenregels voor logische variabelen

1. *De gelijksheidswet*

$$z + z = z \qquad 1.a$$
$$z \cdot z = z \qquad 1.b$$

Deze wet zegt dat in een logische som of produkt identieke termen mogen worden herhaald. Voor kontaktschakelingen betekent deze wet dat herhaling van dezelfde kontaktketen in serie- en parallelschakelingen is toegestaan, maar in wezen overbodig is. Voor de schakelalgebra betekent de gelijkheidswet dat coëfficiënten en machten van getallen anders dan 0 en 1 niet voorkomen.

2. *De associatieve wet*

$$(x + y) + z = x + (y + z) \qquad 2.a$$
$$(x \cdot y) \cdot z = x \cdot (y \cdot z) \qquad 2.b$$

De associatieve wet zegt dat bij de interpretatie van een logische formule de aangehouden volgorde van identieke logische bewerkingen onbelangrijk is.

3. *De commutatieve wet*

$$y + z = z + y \qquad 3.a$$
$$y \cdot z = z \cdot y \qquad 3.b$$

Volgens deze wet mag men de volgorde van termen in een logische som resp. produkt naar behoeve wijzigen.

4. *De distributieve wet*

$$x + y \cdot z = (x + y) \cdot (x + z) \qquad 4.a$$
$$x \cdot (y + z) = x \cdot y + x \cdot z \qquad 4.b$$

Deze wet geeft aan hoe haakjes binnen resp. uit een formule kunnen worden gebracht. Het is voor deze wet interessant de met deze logische uitdrukkingen overeenkomende kontaktschakelingen te bekijken. Fig. 1.5 geeft de schakeling behorend bij regel 4.a

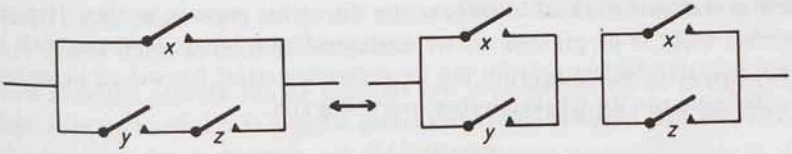


fig. 1.5. De distributieve wet in kontaktschakelingen.

De schakelingen zijn logisch equivalent. De eerste schakeling bezit drie kontakten en verdient uit het gezichtspunt van materiaalgebruik de voorkeur. De distributieve wet kan gebruikt worden als een hulpmiddel bij het vereenvoudigen van schakelingen.

5. *De modulus wetten*

$$0 + z = z \qquad 5.a$$
$$1 \cdot z = z \qquad 5.b$$
$$1 + z = 1 \qquad 5.c$$
$$0 \cdot z = 0 \qquad 5.d$$

De modulus wetten beschrijven hoe formules met constanten en logische variabelen moeten worden geïnterpreteerd.

6. *De negatie wetten*

$$z + \bar{z} = 1 \qquad 6.a$$
$$z \cdot \bar{z} = 0 \qquad 6.b$$
$$\overline{(\bar{z})} = z \qquad 6.c$$
$$\left.\begin{array}{l} \overline{y + z} = \bar{y} \cdot \bar{z} \qquad 6.d \\ \overline{y \cdot z} = \bar{y} + \bar{z} \qquad 6.e \end{array}\right\} \text{ De wetten van De Morgan}$$

De negatie wetten 6.a t/m 6.e beschrijven de grondregels, volgens welke de uitdrukkingen waarin inversies voorkomen moeten worden geïnterpreteerd. De wetten 6.d en 6.e worden de wetten van De Morgan genoemd. De Morgan behoort evenals Boole tot de grondleggers van de symbolische logica.

7. *De absorptie wetten*

$$z + y \cdot z = z \qquad 7.a$$
$$z \cdot (y + z) = z \qquad 7.b$$
$$y + \bar{y} \cdot z = y + z \qquad 7.c$$
$$y \cdot (\bar{y} + z) = y \cdot z \qquad 7.d$$
$$x \cdot y + \bar{x} \cdot z + y \cdot z = x \cdot y + \bar{x} \cdot z \qquad 7.e$$
$$(x + y) \cdot (\bar{x} + z) \cdot (y + z) = (x + y) \cdot (\bar{x} + z) \qquad 7.f$$

De absorptie wetten zijn vereenvoudigingswetten. Deze wetten geven aan dat een bepaalde term in een formule resp. een tak in een schakeling overbodig is. Soms echter gebruikt men deze wetten andersom. Door het toevoegen van termen is het soms mogelijk de invloed van hinderlijke overgangsverschijnselen uit te sluiten.

Onder de wetten genoemd onder 1 t/m 7 komen een aantal overbodige wetten voor, d.w.z. wetten die direct afleidbaar zijn uit eerder gegeven wetten. Het gegeven stelsel blijkt in de praktijk echter uitstekend te voldoen. Men kan zich veel werk besparen bij het gebruik van de absorptiewetten, hoewel zij in wezen voor de definitie van de schakelalgebra overbodig zijn.

Het bewijzen van schakelwetten

Een formeel bewijs voor de juistheid van de boven beschreven logische wetten kan men leveren via een waarheidstabel. Links van de vertikale streep zet men alle mogelijke combinaties van waarheidswaarden van de logische variabelen, rechts zet men in twee verschillende kolommen de waarheidswaarden van de logische uitdrukkingen links en rechts van het gelijkteken in de wet. Zijn deze twee kolommen gelijk, dan hiermee de wet bewezen.

Voorbeeld
Bewijs de wetten van De Morgan.

y	z	$\overline{y + z}$	$\bar{y} \cdot \bar{z}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

tabel 1.14.

y	z	$\overline{y \cdot z}$	$\bar{y} + \bar{z}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

tabel 1.15.

Het bewijs voor de absorptiewetten kan men ook geven op grond van de eerder gegeven wetten.

Voorbeeld
Bewijs van wet 7.a:

$z + y \cdot z = 1 \cdot z + y \cdot z$ (via 5.b)
$= (1 + y) \cdot z$ (via 3.b en 4.b)
$= 1 \cdot z$ (via 5.c)
$= z$ (via 5.b)

Bewijs van wet 7.d:

$y \cdot (\bar{y} + z) = y \cdot \bar{y} + y \cdot z$ (via 4.b)
$= 0 + y \cdot z$ (via 6.b)
$= y \cdot z$ (via 5.a)

Het begrip duaal in de schakelalgebra

In de Boole algebra kent men het begrip *duale vorm* van een formule. Als een bepaalde uitdrukking waar is, dan is ook de duale vorm ervan waar. De duale vorm van een uitdrukking ontstaat als volgt:
– Vervang elke + door · en elke · door + in een uitdrukking
– Vervang elke 1 door 0 en omgekeerd elke 0 door 1.
Het bewijs dat als een bepaalde uitdrukking of bewering juist is ook de duale vorm ervan juist is, wordt geleverd in de Boole algebra. In de schakelalgebra komt het begrip duaal veel voor. Alle schakelwetten hierboven zijn gegeven in paren. Bij onderzoek blijken de twee wetten in elk paar elkaars duale vorm te zijn. Een schijnbare uitzondering vormt wet 6.c. Van deze wet zijn de gegeven vorm en de duale vorm ervan aan elkaar gelijk. Vanwaar deze belangstelling voor het begrip duaal in de schakelalgebra?

Een mogelijk gebruik van de dualiteit van twee wetten bij de bewijsvoering ervan ligt voor de hand. Zo is wet 7.e gemakkelijker te bewijzen dan wet 7.f. Is echter 7.e bewezen dan is daarmee ook 7.f bewezen.

Het gebruik van het begrip duaal in de digitale techniek gaat echter veel verder. Vergelijken we tabel 1.9 met de tabellen 1.10 en 1.11 dan zien we dat de elektronische schakeling die door tabel 1.9 wordt beschreven in het ene geval de logische OF realiseert en in het andere geval de logische EN:

$$S = Y + Z \quad \text{resp.} \quad S = Y \cdot Z.$$

Beide logische formules zijn duaal ten opzichte van elkaar. Dit is geen toeval, zoals de volgende stelling aangeeft:

Stelling:

De logische functies die ontstaan bij de twee mogelijke interpretaties van H(oog) en L(aag) als T(rue) en F(alse) van een tabel die de elektrische werking van een bouwsteen beschrijft, zijn elkaars duale vorm.

Bewijs:

fig. 1.6.

Zij gegeven een element werkend volgens een beschrijving in H en L. Een beschrijving in F en T volgens de toekenning

$$F \rightarrow L,\ T \rightarrow H$$

ontstaat door de in- en uitgangen te voorzien van de getekende omzetters. Men bedenke echter dat het alternatief,

$$F \rightarrow H,\ T \rightarrow L$$

bereikt kan worden door vóór elke ingang en uitgang een z.g. *invertor* te plaatsen die de bewerking NIET uitvoert. In de schakelalgebra betekenen deze inversies

$$S = a \cdot b + c \cdot d \longrightarrow S^* = \overline{\bar{a} \cdot \bar{b} + \bar{c} \cdot \bar{d}} = (a + b) \cdot (c + d).$$

Nu zijn de schakelfuncties S en S^* echter elkaars duale vorm. Dit geldt ook voor een willekeurige schakelfunctie.

Het systeem van toekenning

$$F \rightarrow L \text{ en } T \rightarrow H$$

wordt het systeem van de *positieve logica* genoemd. Het alternatief

$$F \rightarrow H \text{ en } T \rightarrow L$$

het systeem van de *negatieve logica*. In de praktijk wordt tegenwoordig vrijwel uitsluitend het systeem van de positieve logica gevolgd.

1.3. Canonieke vormen van schakelfuncties

Voorbeeld
Zij gegeven een schakeling met drie binaire uitgangssignalen x, y en z. Aan deze signalen wordt het volgende gewicht toegekend:

$$x \leftrightarrow 2^2 \quad y \leftrightarrow 2^1 \quad z \leftrightarrow 2^0.$$

Zo correspondeert met de combinatie xyz = 110 het getal

$$g = 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 6.$$

Gevraagd wordt een schakeling te ontwerpen die signaleert of het met een bepaalde uitgangscombinatie corresponderende getal groter dan of gelijk aan 3 is. Tabel 1.16 specificeert de gewenste logische werking van de schakeling. (In plaats van T en F gebruiken we voortaan 1 en 0.)

g	x	y	z	S
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

tabel 1.16. Specificatie van een schakelfunctie.

x	y	z	S
0	0	0	f_0
0	0	1	f_1
0	1	0	f_2
0	1	1	f_3
1	0	0	f_4
1	0	1	f_5
1	1	0	f_6
1	1	1	f_7

tabel 1.17. Waarheidstabel van een willekeurige schakelfunctie.

De logische formule die behoort bij tabel 1.16 is

$$S = \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot \bar{z} + x \cdot \bar{y} \cdot z + x \cdot y \cdot \bar{z} + x \cdot y \cdot z.$$

Deze formule kan worden vereenvoudigd tot

$$S = x + y \cdot z.$$

Met deze laatste stap zijn we gekomen tot een beknopte formulering van de werkingsvoorwaarden van de te ontwerpen schakeling.

Opmerking
In het vervolg wordt de vermenigvuldigpunt, waarmee men de logische bewerking EN aanduidt, weggelaten wanneer dit de duidelijkheid van de formule niet schaadt.

De mintermvorm van schakelfuncties

In het bovenstaande voorbeeld is de specificatie van de gewenste werking van de schakeling gegeven in waarheidstabel 1.16. In het algemeen kan een uitgangssignaal twee waarden aannemen, zoals in waarheidstabel 1.17 is beschreven. Elke *functiewaarde* f_i (hier $0 \leqslant i \leqslant 7$) is een binaire grootheid, die afhankelijk van het beschreven probleem de waarde 0 of 1 aanneemt. Zo is van een schakeling die moet aangeven of het aangeboden getal in het vorige voorbeeld even (1) of oneven (0) is:

$$f_0 = f_2 = f_4 = f_6 = 1 \qquad f_1 = f_3 = f_5 = f_7 = 0.$$

Algemeen geldt:
Een waarheidstabel met n ingangsvariabelen kan 2^{2^n} verschillende schakelfuncties beschrijven. Niet elke functie in n variabelen is echter even zinvol. Zo is de schakelfunctie met f_0 t/m f_7 gelijk aan 1 identiek met de logische 1. En om deze voor te stellen zijn geen drie binaire variabelen nodig!

Tabel 1.17 leidt tot een standaardvorm (een som-van-produkten vorm) van een schakelfunctie in drie variabelen:

$$S = \bar{x}\bar{y}\bar{z}\cdot f_0 + \bar{x}\bar{y}z\cdot f_1 + \bar{x}y\bar{z}\cdot f_2 + \ldots + xy\bar{z}\cdot f_6 + xyz\cdot f_7$$
$$= m_0\cdot f_0 + m_1\cdot f_1 + m_2\cdot f_2 + \ldots + m_6\cdot f_6 + m_7\cdot f_7.$$

Bij elke functiewaarde f_i behoort één produkt van de variabelen x, y en z. Dit produkt neemt voor precies één ingangscombinatie van de schakeling de waarde 1 aan, voor alle overige combinaties heeft het de waarde 0. Een dergelijk produkt, dat als *adres* van de betreffende functiewaarde in de waarheidstabel kan worden geïnterpreteerd, wordt een *minterm* genoemd. Een minterm is een produkt van (ingangs)variabelen waarin alle ingangsvariabelen één maal voorkomen. Duidelijk is dat:

1. $m_i\cdot m_j = 0$ mits $i \neq j$
2. $m_i = 1$ voor slechts één ingangscombinatie
3. $\sum_{i=0}^{7} m_i = 1$.

Het rangnummer van de minterm resp. functiewaarde wordt als volgt bepaald: Ken de k ingangsvariabelen van een schakelfunctie de gewichten $2^0, 2^1, \ldots, 2^{k-1}$ toe. Tel die gewichten bij elkaar op waarvoor in de bijbehorende minterm de variabele niet geïnverteerd voorkomt. Deze som is het rangnummer van de minterm.

Venndiagrammen

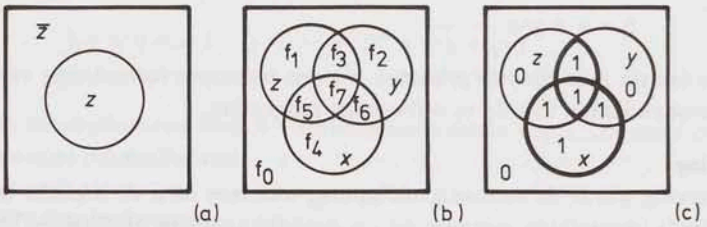


fig. 1.7. Venndiagrammen.

In fig. 1.7.a is een z.g. Venndiagram getekend voor één variabele. Het door de cirkel omsloten gebied correspondeert met z = 1, ook te interpreteren als het gebied waar de uitspraak Z waar is. Het gebied buiten de cirkel correspondeert met z = 0. Het gehele gebied, omsloten door de rechthoek wordt ook wel het *universum* genoemd, d.w.z. hierin zijn alle mogelijkheden omvat, zowel z = 0 als z = 1.
Fig. 1.7.b toont een Venndiagram voor een willekeurige schakelfunctie in drie

variabelen. In het diagram zijn acht verschillende gebieden aan te wijzen, die corresponderen met de acht mintermen van een schakelfunctie in drie variabelen. In elk vak is de bijbehorende functiewaarde geplaatst.

Venndiagrammen kunnen worden gebruikt bij de vereenvoudiging van schakelfuncties. In fig. 1.7.c zijn de functiewaarden ingevuld die corresponderen met de in tabel 1.6 gegeven specificatie van een schakeling. We zien onmiddellijk dat het gebied, waar de schakelfunctie de waarde 1 heeft, bestaat uit de vereniging van de gebieden x en yz. De meest eenvoudige vorm van de schakelfunctie is dus

$$S = x + yz.$$

Venndiagrammen voor meer dan vier variabelen worden weldra onoverzichtelijk en zijn voor het vereenvoudigen van schakelfuncties niet meer geschikt. In het volgende hoofdstuk wordt een meer gestileerde vorm van het Venndiagram geïntroduceerd speciaal ten behoeve van het vereenvoudigen van schakelfuncties. Nu het Venndiagram geïntroduceerd is, zal ook de betekenis van het woord min(imale)term duidelijk zijn.

De maxtermvorm van schakelfuncties

De mintermvorm van een schakelfunctie wordt veel gebruikt vanwege de directe correspondentie met de waarheidstabel. Soms wordt nog een andere standaardvorm (een produktvorm) van een schakelfunctie gebruikt, de z.g. *maxtermvorm*. Het verband met de mintermvorm is als volgt:
Zij gegeven een schakelfunctie in drie variabelen. De inverse van deze schakelfunctie ontstaat door alle functiewaarden te inverteren, d.w.z. van een NIET te voorzien. We kunnen de oorspronkelijke functie weer terug krijgen door het geheel nogmaals te inverteren. In formule:

$$\begin{aligned}
\bar{S} &= m_0 \cdot \bar{f}_0 + m_1 \cdot \bar{f}_1 + \ldots + m_6 \cdot \bar{f}_6 + m_7 \cdot \bar{f}_7 \\
S = \bar{\bar{S}} &= \overline{(m_0 \cdot \bar{f}_0 + m_1 \cdot \bar{f}_1 + \ldots + m_6 \cdot \bar{f}_6 + m_7 \cdot \bar{f}_7)} \\
&= (\bar{m}_0 + f_0) \cdot (\bar{m}_1 + f_1) \cdot (\ldots) \cdot (\bar{m}_6 + f_6) \cdot (\bar{m}_7 + f_7) \\
&= (M_0 + f_0) \cdot (M_1 + f_1) \cdot (\ldots) \cdot (M_6 + f_6) \cdot (M_7 + f_7).
\end{aligned}$$

Het verband tussen een minterm m_i en een z.g. maxterm M_i is dus:

$$m_i = \bar{M}_i \quad \text{of} \quad \bar{m}_i = M_i.$$

Een maxterm M_i vervult in een produktvorm dezelfde rol als m_i in een somvorm van de schakelfunctie. Voor elke combinatie van waarden van de binaire variabelen wordt de waarde van de functie S bepaald door precies één functiewaarde f_i. Werden daartoe in een somvorm alle overige termen 0, in een produktvorm echter moeten alle overige factoren van het produkt juist 1 zijn. Ook de naam maxtermvorm kan nu worden verklaard. Elke maxterm (maximale term) correspondeert met een zo groot mogelijk gebied in een Venndiagram, zodat het nog net niet samenvalt met het universum. Beide vormen van een schakelfunctie, de mintermvorm en de maxtermvorm, worden wel de *canonieke vormen* van een schakelfunctie genoemd. Beide vormen worden zelden direct in deze vorm in een schakeling gerealiseerd. Daarvoor kiest men liever een vereenvoudigde uitdruk-

king die minder materiaal kost. In het volgende hoofdstuk worden daartoe enkele vereenvoudigingsmethoden behandeld.

Voorbeeld
Bepaal de mintermvorm, de maxtermvorm en de meest eenvoudige somvorm en produktvorm van de volgende schakelfunctie

$$S = \bar{x}\bar{y}\bar{z} + x\bar{y} + x\bar{z} + yz.$$

Oplossing.
De mintermvorm vinden we door expansie, d.w.z. introductie van ontbrekende variabelen:

$$\begin{aligned} S &= \bar{x}\bar{y}\bar{z} + x\bar{y}(\bar{z} + z) + x(\bar{y} + y)\bar{z} + (\bar{x} + x)yz \\ &= \bar{x}\bar{y}\bar{z} + x\bar{y}\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + xy\bar{z} + \bar{x}yz + xyz \\ &= \bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z} + xyz. \end{aligned}$$

De overige termen ontbreken doordat de corresponderende functiewaarden 0 zijn. De functiewaarden met waarde 1 zijn dus:

$$f_0, f_3, f_4, f_5, f_6, f_7.$$

De functiewaarden die 0 zijn, zijn:

$$f_1, f_2.$$

Hieruit volgt direct de maxtermvorm:

$$\begin{aligned} S &= (\bar{m}_0 + f_0)\cdot(\bar{m}_1 + f_1)\cdot(\ldots)\cdot(\bar{m}_7 + f_7) \\ &= \bar{m}_1 \cdot \bar{m}_2 \\ &= (x + y + \bar{z})(x + \bar{y} + z). \end{aligned}$$

De eenvoudigste somvorm ontstaat als volgt uit de mintermvorm:

$$\begin{aligned} S &= \bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z} + xyz \\ &= (\bar{x}\bar{y}\bar{z} + x\bar{y}\bar{z}) + (\bar{x}yz + xyz) + x(\bar{y}\bar{z} + \bar{y}z + y\bar{z} + yz) \\ &= \bar{y}\bar{z} + yz + x. \end{aligned}$$

De eenvoudigste produktvorm van de gegeven functie is (toevallig) gelijk aan de maxtermvorm.

Opmerking
In veel boeken specificeert men een schakelfunctie door de functiewaarden te geven die 1 zijn of alleen hun indices. Voor bovenstaand voorbeeld zou dit zijn:

$$S = f_0, f_3, f_4, f_5, f_6, f_7$$

of

$$S = 0, 3, 4, 5, 6, 7.$$

1.4. Het ontbinden van een schakelfunctie naar zijn variabelen

Welke vorm van een schakelfunctie leidt tot een economische realisatie van de schakeling met de ter beschikking staande bouwstenen hangt van vele factoren af. Deze komen nog ter sprake in de hoofdstukken over kontaktschakelingen en poortschakelingen. Meestal berust een goede realisatie op een eenvoudige somvorm of produktvorm van de functie. Er bestaan echter een paar vormen van schakelfuncties die niet onbesproken mogen blijven in een hoofdstuk over schakelalgebra.

Wisselkontakten maken het mogelijk om een te realiseren schakelfunctie in twee subfuncties te splitsen, elk met één variabele minder. De volgende stelling is hierop van toepassing. Fig. 1.8 geeft een toelichting.

Stelling
Een schakelfunctie in (bijvoorbeeld) drie variabelen kan men schrijven met behulp van twee subfuncties in één variabele minder en wel als volgt:

$$S = f(x,y,z)$$
$$= \bar{x}\cdot f_1(0,y,z) + x\cdot f_2(1,y,z)$$

waarbij $f_1(0,y,z)$ uit $f(x,y,z)$ ontstaat door voor elke x in $f(x,y,z)$ de waarde 0 in te vullen (en voor elke $\bar{x}$ de waarde 1). Evenzo ontstaat $f_2(1,y,z)$ uit $f(x,y,z)$ door voor x de waarde 1 in te vullen.

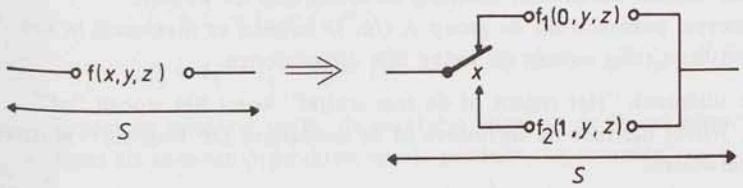


fig. 1.8. Ontwikkeling van een schakelfunctie naar één van de variabelen.

Bewijs:
1. $S = f(x,y,z)$ is gegeven in de mintermvorm. De helft der termen begint met $\bar{x}$, de andere helft met x. Haal $\bar{x}$ en x buiten haken en het bewijs is geleverd.
2. $S = f(x,y,z)$ is gegeven in een willekeurige vorm. Vul links en rechts van het gelijkteken $x = 0$ resp. $x = 1$ in. In beide gevallen zijn de functies links en rechts identiek:

$$f(x,y,z) = \bar{x}\cdot f_1(0,y,z) + x\cdot f_2(1,y,z)$$
$$x = 0: \quad f(0,y,z) = 1\cdot f_1(0,y,z) + 0\cdot f_2(1,y,z)$$
$$x = 1: \quad f(1,y,z) = 0\cdot f_1(0,y,z) + 1\cdot f_2(1,y,z).$$

Omdat x slechts twee waarden, 0 of 1, kan aannemen is hiermee de stelling ook bewezen.

De ontbinding van een schakelfunctie kan naar de overige variabelen worden voortgezet:

$$
\begin{aligned}
S &= f(x,y,z) \\
&= \bar{x}\cdot f(0,y,z) + x\cdot f(1,y,z) \\
&= \bar{x}\bar{y}\cdot f(0,0,z) + \bar{x}y\cdot f(0,1,z) + x\bar{y}\cdot f(1,0,z) + xy\cdot f(1,1,z) \\
&= \bar{x}\bar{y}\bar{z}\cdot f(0,0,0) + \bar{x}\bar{y}z\cdot f(0,0,1) + \ldots + xy\bar{z}\cdot f(1,1,0) + xyz\cdot f(1,1,1).
\end{aligned}
$$

De constanten f(0,0,0) t/m f(1,1,1) komen overeen met de functiewaarden f_0 t/m f_7 zoals deze hiervoor zijn geïntroduceerd.
De toepassing van deze ontbinding bij het ontwerpen van kontaktschakelingen wordt nader bestudeerd in hoofdstuk 3. Diverse ontwerpmethoden voor o.a. kontaktschakelingen berusten hierop.

Opgaven

1.1. Vier personen A, B, C en D komen in aanmerking om zitting te nemen in het bestuur van een vereniging. Er zijn echter enkele beperkingen:
- In verband met een andere functie in de vereniging, die niet kan worden gecombineerd met een bestuursfunctie, dient A of B buiten het bestuur te blijven.
- Wegens persoonlijke tegenstellingen is het niet wenselijk C en D samen in het bestuur te kiezen.
- Als A in het bestuur zit, dan moet C er ook in en omgekeerd.

a. Formuleer in een logische formule de condities waaronder het bestuur kan worden aangevuld, rekening houdend met de wensen.
b. Hoeveel personen uit de groep A t/m D kunnen er maximaal in het bestuur zitting nemen en welke zijn die personen.

1.2. In de uitspraak "Het regent of de zon schijnt" komt het woord "of" voor. Wordt hiermee de inclusieve of de exclusieve OF bedoeld? Motiveer uw antwoord.

1.3. Bij een kruising splitst de weg zich naar links en naar rechts. Een reiziger moet naar de stad A. Hij weet dat één van beide wegen naar A leidt. Hem is medegedeeld dat, als hij aan een van twee broers die bij het kruispunt wonen een vraag stelt, hij van één van de twee broers altijd een verkeerd antwoord krijgt. De andere broer geeft een correct antwoord op elke vraag.
a. Welke vraag moet hij aan één van de broers stellen om de juiste richting naar A te vinden?
b. Bewijs met behulp van een of meer proposities de geschiktheid van de vraag om de juiste richting vast te stellen.

1.4. De kluis van een bank mag slechts worden geopend door:
- De directeur alleen.
- De kassier en één van een groep van vier afdelingshoofden.
- Drie afdelingshoofden.

Elk heeft een eigen sleutel die, als deze in de bijbehorende gleuf wordt geplaatst, een "1" doet afgeven aan het slot.
Specificeer met een schakelformule de werking van een combinatorische schakeling, die het slot vrijgeeft (S = 1) als de kluis mag worden geopend.

1.5. Toon met behulp van de rekenregels van de schakelalgebra aan dat geldt:

$$(x + y)\cdot(x + z)\cdot(y + z) = x\cdot y + x\cdot z + y\cdot z.$$

Geldt ook dat

$$(w + x + y)\cdot(w + x + z)\cdot(w + y + z)\cdot(x + y + z) =$$
$$= w\cdot x\cdot y + w\cdot x\cdot z + w\cdot y\cdot z + x\cdot y\cdot z$$

1.6. Toon met behulp van de rekenregels uit de schakelalgebra aan dat de volgende twee vormen van de schakelfunctie S logisch equivalent zijn:

$$S_1 = \bar{x}\bar{y} + x\bar{z} + yz \qquad S_2 = \bar{x}z + xy + \bar{y}\bar{z}$$

1.7. Vereenvoudig de onderstaande schakelfuncties zo veel mogelijk:

$$S_1 = x(y + z) + x\bar{y}\bar{z}$$

$$S_2 = \bar{w}\bar{x}\bar{y} + \bar{w}\bar{x}z + \bar{x}y\bar{z} + wxy\bar{z}$$

Gebruik uitsluitend de rekenregels van de schakelalgebra en geef de eenvoudigste formule in de som-van-produkten vorm.

1.8. Geef de duale vorm van onderstaande vergelijking:

$$(x + y)(\bar{x} + z)(y + z) = xz + \bar{x}y$$

1.9. Een schakelfunctie in drie variabelen x, y en z wordt als volgt gespecificeerd:

$$f_0 = f_1 = f_3 = f_7 = 0$$

$$f_2 = f_4 = f_5 = f_6 = 1$$

Bepaal de minterm vorm, de maxterm vorm en de meest eenvoudige vorm als som-van-produkten en als produkt-van-sommen.

1.10. Stel de waarheidstabel op van de schakelfunctie:

$$S = (x + y)(x + \bar{y} + \bar{z})(\bar{x} + y + z)(\bar{y} + z)$$

1.11. Een bouwsteen met drie ingangen x, y en z, realiseert de logische functie

$$S = xy + \bar{z}$$

aan de uitgang.

a. Als voor elke ingang en uitgang een invertor wordt geplaatst ($0 \rightarrow 1$ en $1 \rightarrow 0$), welke logische functie wordt dan gerealiseerd?

b. Hoe is dit voor een willekeurige bouwsteen? Motiveer uw antwoord.

1.12. Lees de schakelfunctie

$$S = x \oplus y \oplus z$$

als

$$S = (x \oplus y) \oplus z$$

a. Schrijf S in de minterm vorm.

b. Toon aan dat

$$S = \bar{x} \oplus y \oplus z = x \oplus \bar{y} \oplus z = x \oplus y \oplus \bar{z} = \bar{x} \oplus \bar{y} \oplus \bar{z}$$

1.13. Gegeven is de schakelfunctie S:

$$S = xy + xz + yz$$

Toon aan dat

$$\bar{S} = \bar{x}\bar{y} + \bar{x}\bar{z} + \bar{y}\bar{z}$$

Geldt ook dat

$$S = vw + xyz \leftrightarrow \bar{S} = \bar{v}\bar{w} + \bar{x}\bar{y}\bar{z}$$

Motiveer uw antwoord.

Literatuur

1. S.T.M. Ackermans en J.H. van Lint, *Algebra en Analyse*, Wolters-Noordhoff NV; Groningen, 1970.
2. F.E. Hohn, *Applied Boolean Algebra*, The Macmillan Company; New York, 1966.
3. E. Mendelson, *Boolean Algebra and Switching Circuits*, McGraw-Hill Book Company; New York, 1970. (Aanbevolen)
4. S.R. Petrick, *A Direct Determination of the Irredundant Forms of a Boolean Function from the Set of Prime Implicants*, Computation Laboratory of the AFCRC, Bedford, Mass., TR-56-110, April 1956.
5. C.E. Shannon, *A Symbolic Analysis of Relay and Switching Circuits*, Transactions of the AIEE no. 57, 1938, pp. 713–723.

Voor een verdere studie op het gebied van de Boole Algebra, de Logica en de Algebra wordt verwezen naar:

6. G. Birkhoff and S. MacLane, *A Survey of Modern Algebra*, The MacMillan Company; New York, 1965.
7. G. Boole, *An Investigation of the Laws of Thought*, London, 1854.
8. Ph. Dwinger, *Introduction to Boolean Algebras*, Physica Verlag; Würzburg, 1961.
9. J.B. Fraleigh, *A First Course in Abstract Algebra*, Addison-Wesley Publ. Cie.: Reading, Mass., 1973.
10. P.R. Halmos, *Intuitieve Verzamelingenleer*, Het Spectrum, Aula 372; Utrecht, 1968.
11. P.R. Halmos, *Lectures on Boolean Algebras*, Van Nostrand, 1963.
12. E.J. Lemmon, *Beginning Logic*, Nelsons University Paperbacks, 1971. (*Moderne Logica*, Prisma-C. 56)
13. F. Loonstra, *Inleiding tot de Algebra*, Noordhoff; Groningen, 1963.

2. HET VEREENVOUDIGEN VAN SCHAKELFUNCTIES

2.1. Het Karnaughdiagram

Uit de in het hoofdstuk schakelalgebra gegeven voorbeelden blijkt dat het vereenvoudigen van schakelfuncties met behulp van de rekenregels van de schakelalgebra een moeizaam proces is. Er zijn uit de literatuur enkele systematische methoden bekend, waaronder die van Karnaugh en van Quine en McCluskey. De door Veitch voorgestelde en door Karnaugh verbeterde methode maakt gebruik van een grafische voorstelling van een schakelfunctie door middel van het *Karnaughdiagram*. De methode is zeer geschikt als handmethode. De door Quine-McCluskey beschreven methode is ontworpen om op een digitale rekenmachine te worden geprogrammeerd. Als eerste vereenvoudigingsmethode behandelen we het Karnaughdiagram.

De algemene mintermvorm van een schakelfunctie in twee logische variabelen y en z is

$$S \doteq m_0 f_0 + m_1 f_1 + m_2 f_2 + m_3 f_3.$$
$$= \bar{y}\bar{z} f_0 + \bar{y} z f_1 + y\bar{z} f_2 + y z f_3$$

waarin de minterm m_i de waarde 1 heeft als de bijbehorende ingangscombinatie optreedt. In alle andere gevallen is $m_i = 0$. De functiewaarden f_i kunnen hierbij de logische waarde 0 of de logische waarde 1 aannemen, hetgeen afhangt van de specificatie van het te onderzoeken probleem. Het Karnaughdiagram voor een willekeurige functie in twee variabelen is getekend in fig. 2.1.a, terwijl in het diagram van fig. 2.1.b de schakelfunctie

$$S = \bar{y} z + y\bar{z} + y z = m_0 \cdot 0 + m_1 \cdot 1 + m_2 \cdot 1 + m_3 \cdot 1$$

geplaatst is. Deze Karnaughdiagrammen zijn zo opgesteld dat de functiewaarden

fig. 2.1. Karnaughdiagrammen voor functies in twee variabelen.

f_i en f_j welke behoren bij mintermen die slechts in één variabele verschillen, in *aangrenzende* hokjes geplaatst zijn. De randschriften geven aan bij welke minterm

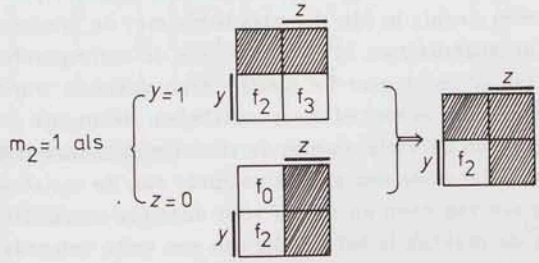


fig. 2.2. Het plaatsen van een functiewaarde in het Karnaughdiagram.

de in een bepaald hokje geplaatste functiewaarde behoort. In fig. 2.2 wordt stapsgewijs bepaald in welk hokje de bij de minterm $m_2 = y\bar{z}$ behorende functiewaarde f_2 moet worden geplaatst.
In een Karnaughdiagram wordt een duidelijk overzicht verkregen van alle mogelijkheden om de termen van een schakelfunctie te combineren. Bevatten twee aangrenzende vakjes een 1, dan resulteert dit in de gereduceerde vorm van de schakelfunctie in een nieuwe term die één variabele minder bevat.
Fig. 2.3 geeft de Karnaughdiagrammen voor willekeurige functies tot en met vier variabelen. Het diagram voor schakelfuncties in n variabelen ontstaat uit het diagram voor n − 1 variabelen door spiegeling rond een der randen.
Deze spiegellijnen zijn in fig. 2.3 dikker getekend. Bij het na spiegeling ontstane nieuwe gedeelte van het diagram wordt de volgende variabele als randschrift toegevoegd.

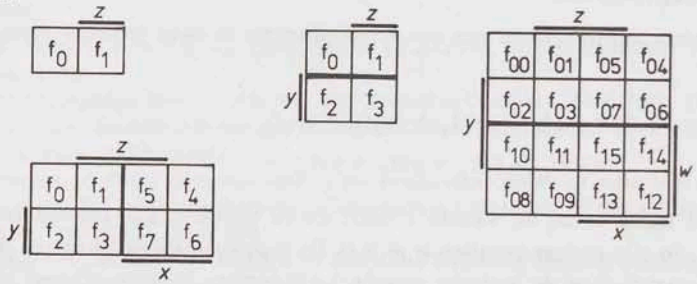


fig. 2.3. Karnaughdiagrammen tot functies in vier variabelen.

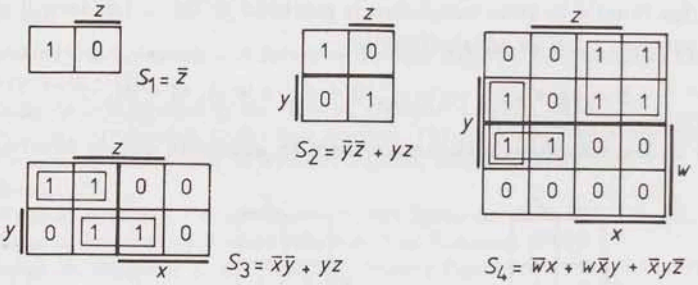
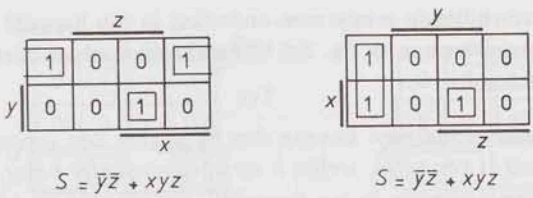


fig. 2.4. Karnaughdiagrammen van gegeven functies.

Fig. 2.4. geeft enkele voorbeelden van schakelfuncties en het bijbehorende patroon van nullen en enen in de verschillende Karnaughdiagrammen.
Het begrip *aangrenzende hokje* verdient voor Karnaughdiagrammen in meer dan twee variabelen een nadere toelichting. Zo verschilt de minterm $\bar{x}y\bar{z}$ van een functie in drie variabelen slechts in één der variabelen met de mintermen $\bar{x}yz$, $\bar{x}\bar{y}\bar{z}$ en $xy\bar{z}$. In het Karnaughdiagram in fig. 2.3 liggen de corresponderende hokjes niet alle naast elkaar, er moet *over de randen heen* gekeken worden. Hetzelfde geldt voor diagrammen in vier of meer variabelen. Belangrijk is het op te merken dat de volgorde van de variabelen in de randschriften niet essentieel is. In fig. 2.5 is aangegeven dat door een andere volgorde van de variabelen in de randschriften het patroon van enen en nullen voor dezelfde schakelfunctie zich sterk kan wijzigen. In de praktijk is het handig om een vaste volgorde van de variabelen in de randschriften aan te houden.



$S = \bar{y}\bar{z} + xyz$ $\qquad$ $S = \bar{y}\bar{z} + xyz$

fig. 2.5. Karnaughdiagrammen met verschillende volgorde van de variabelen in de randschriften.

Van de schakelfuncties S_1 en S_2 die in waarheidstabel 2.1 gespecificeerd zijn, staan de bijbehorende Karnaughdiagrammen in fig. 2.6 en fig. 2.7. De functie S_1 kan worden vereenvoudigd tot

$$S_1 = xy + xz + yz$$

terwijl de meest eenvoudige vorm van de functie S_2 is

$$\begin{aligned} S_2 &= \bar{x}\bar{y}z + \bar{x}yz + x\bar{y}z + xyz \\ &= \bar{x}z + xz \\ &= z. \end{aligned}$$

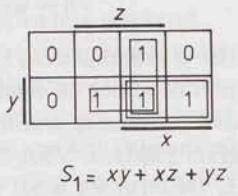
Het gebruik van het Karnaughdiagram berust in wezen op het herhaald toepassen van de schakelwetten

$$z + z = z \qquad \text{en} \qquad z + \bar{z} = 1 .$$

Volgens de eerste wet mag een term in een vereenvoudiging meer dan één keer gebruikt worden, zoals in fig. 2.6 is geïllustreerd. De tweede wet zegt dat twee termen tot één term kunnen worden gecombineerd als zij slechts in één variabele verschillen. Deze wet is in fig. 2.7 twee maal toegepast.

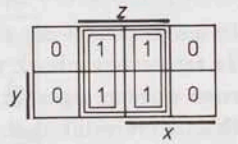
i	x	y	z	m_i	S_1	S_2
0	0	0	0	$\bar{x}\bar{y}\bar{z}$	0	0
1	0	0	1	$\bar{x}\bar{y}z$	0	1
2	0	1	0	$\bar{x}y\bar{z}$	0	0
3	0	1	1	$\bar{x}yz$	1	1
4	1	0	0	$x\bar{y}\bar{z}$	0	0
5	1	0	1	$x\bar{y}z$	1	1
6	1	1	0	$xy\bar{z}$	1	0
7	1	1	1	xyz	1	1

tabel 2.1 Waarheidstabel



$S_1 = xy + xz + yz$

fig. 2.6. Karnaughdiagram



$S_2 = z$

fig. 2.7. Karnaughdiagram

Uit fig. 2.7 volgt verder dat het soms mogelijk is vier verschillende mintermen tot één term samen te nemen. Gemakkelijk kan worden ingezien dat dit slechts

mogelijk is als vier verschillende mintermen onderling in ten hoogste twee variabelen verschillen. De vier termen in fig. 2.6 bijvoorbeeld voldoen niet aan dit criterium. In het algemeen:

> *2^k verschillende mintermen kunnen dan en slechts dan gecombineerd worden tot één term, welke k variabelen minder bevat, als deze 2^k termen onderling in ten hoogste k variabelen verschillen. De overige variabelen moeten in alle mintermen op gelijke wijze voorkomen, d.w.z. gewoon of geïnverteerd.*

Fig. 2.8 geeft enkele voorbeelden van gevallen waarbij vier mintermen in het Karnaughdiagram kunnen worden gecombineerd. Uit deze figuur blijkt tevens dat de meest eenvoudige vorm van de schakelfunctie niet altijd ondubbelzinnig bepaald is, soms is een keuze mogelijk uit twee of meer formules.

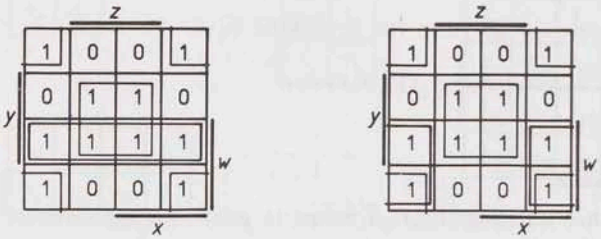


fig. 2.8. Verschillende omrandingen voor dezelfde functie.

Voorbeeld

Het nu volgende voorbeeld dient als toelichting op het werken met het Karnaughdiagram. Vereenvoudig de logische functie in vier variabelen w, x, y en z :

$$S = \bar{w}x + wx\bar{z} + wy\bar{z} + xy\bar{z} + \bar{w}\bar{x}\bar{y}z + w\bar{x}yz\ .$$

Het is mogelijk deze functie eerst tot de mintermvorm te expanderen om vervolgens de nullen en enen in het Karnaughdiagram te plaatsen. Na enige ervaring kan deze tussenstap worden overgeslagen en kan men de termen direct in het diagram plaatsen. Voor de term $\bar{w}x$ gaat dit bijvoorbeeld als volgt:

1. De term $\bar{w}x$ bevat uitsluitend mintermen waarvoor w = 0 is. De vakjes in de benedenhelft van het diagram in fig. 2.9.a vallen dus af.
2. De term $\bar{w}x$ bevat verder uitsluitend mintermen waarvoor x = 1 is. De linkerhelft van het diagram valt ook af.

Er is een gebied dat aan beide eisen voldoet, nl. de vier overblijvende vakjes in de rechterbovenhoek van fig. 2.9.a. Op dezelfde wijze kunnen in fig. 2.9.a de enen worden ingevuld die corresponderen met de overige termen van de gegeven functie (vergelijk ook fig. 2.2). In fig. 2.9.b zijn vervolgens de omrandingen aangebracht die corresponderen met de meest eenvoudige vorm van de gegeven schakelfunctie. Deze is :

$$S = \bar{w}x + x\bar{z} + \bar{w}\bar{y}z + w\bar{x}y\ .$$

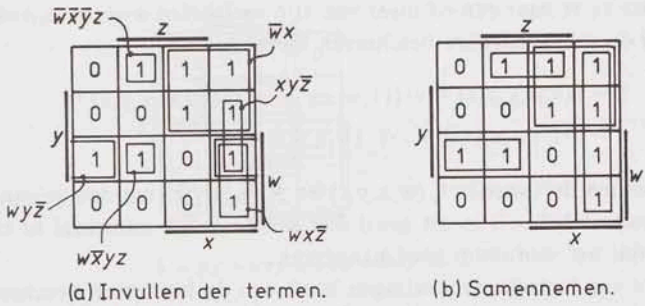
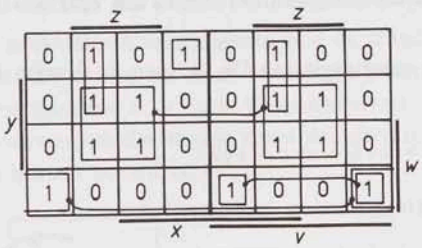


fig. 2.9. Voorbeeld van een vereenvoudiging door middel van het Karnaughdiagram.

Karnaughdiagrammen voor schakelfuncties in meer dan vier variabelen ontstaan uit het diagram voor vier variabelen door spiegeling rond een der randen. In fig. 2.10 is een voorbeeld gegeven van een vereenvoudiging van een functie in een diagram voor vijf variabelen. Het zal duidelijk zijn dat de methode met het Karnaughdiagram zijn aantrekkelijkheid verliest naarmate het aantal variabelen toeneemt.



$$S = yz + \bar{v}\bar{w}\bar{x}z + v\bar{w}xz + vw\bar{y}\bar{z} + w\bar{x}\bar{y}\bar{z} + \bar{v}\bar{w}x\bar{y}\bar{z}$$

fig. 2.10. Karnaughdiagram voor vijf variabelen.

In sommige boeken wordt het diagram voor vijf variabelen anders getekend. Deze variant is in fig. 2.11 weergegeven. De twee helften van het diagram zijn niet meer in spiegelbeeld getekend, maar dienen boven elkaar gedacht te worden. Bovendien zijn in fig. 2.11 de randschriften anders aangegeven. Het hangt geheel van een persoonlijke voorkeur af met welke variant van het diagram men het gemakkelijkst werkt. Het is een nuttige oefening na te gaan waar de nullen en enen uit fig. 2.10 in het diagram van fig. 2.11 geplaatst moeten worden.

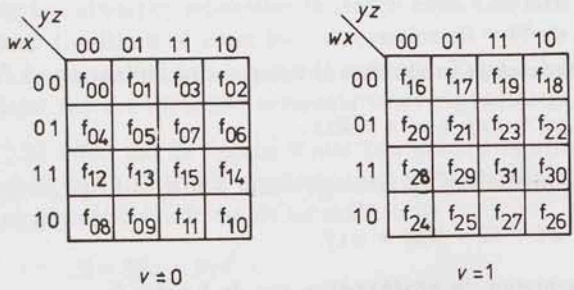


fig. 2.11. Alternatieve vorm van een Karnaughdiagram in vijf variabelen.

Voor functies in meer dan vijf à zes variabelen zijn Karnaughdiagrammen niet praktisch. Men moet dan overgaan op andere methoden. Als tussenstap kan de

gegeven functie eerst naar een of meer van zijn variabelen worden ontwikkeld volgens de bij de schakelalgebra beschreven methode:

$$S = f(v,w,x,y,z) = v \cdot f(1,w,x,y,z) + \bar{v} \cdot f(0,w,x,y,z) =$$
$$= v \cdot f_1(w,x,y,z) + \bar{v} \cdot f_2(w,x,y,z) .$$

Vervolgens kunnen de functies $f_1(w,x,y,z)$ en $f_2(w,x,y,z)$ worden geminimaliseerd. Het resultaat behoeft in dit geval niet noodzakelijk minimaal te zijn, maar zal meestal het minimum goed benaderen.
Een ontwerper van logische schakelingen heeft aan de hierboven beschreven reductiemethode met het Karnaughdiagram meestal voldoende gereedschap om de gangbare combinatorische problemen te kunnen beschrijven en op te lossen.

De som- en produktvorm van schakelfuncties

Tot nu toe is uitsluitend besproken hoe de meest eenvoudige vorm van een logische functie als som-van-produkten kan worden bepaald. Deze vorm van een logische functie wordt verder de *somvorm* genoemd. Voor sommige toepassingen is de representatie als produkt-van-sommen, hierna aan te duiden als *produktvorm*, eenvoudiger.
Wordt met behulp van de negatiewet van De Morgan de inverse functie berekend van de functie S:

$$S = w\bar{y}z + wy\bar{z} + xyz + x\bar{y}\bar{z}$$

dan wordt de inverse functie gevonden in de produktvorm:

$$\bar{S} = \overline{(w\bar{y}z + wy\bar{z} + xyz + x\bar{y}\bar{z})} =$$
$$= \overline{(w\bar{y}z)} \cdot \overline{(wy\bar{z})} \cdot \overline{(xyz)} \cdot \overline{(x\bar{y}\bar{z})} =$$
$$= (\bar{w} + y + \bar{z})\,(\bar{w} + \bar{y} + z)\,(\bar{x} + \bar{y} + \bar{z})\,(\bar{x} + y + z).$$

Dit gegeven kan worden gebruikt om een functie S, die gegeven is in de somvorm, om te zetten in de produktvorm.
De functie S moet dan in een Karnaughdiagram worden geplaatst, waarna men de nullen in plaats van de enen samen neemt. Op deze wijze wordt van de functie $\bar{S}$ de meest eenvoudige somvorm gevonden. Door inversie volgt hieruit de functie S in de produktvorm.

Voorbeeld
Bepaal de meest eenvoudige produktvorm van de schakelfunctie

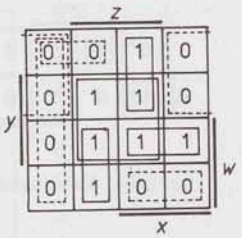
$$S = yz + wxy + w\bar{x}z + \bar{w}xz.$$

De eenvoudigste somvorm van de inverse functie $\bar{S}$ volgt uit fig. 2.12:

$$\bar{S} = \bar{w}\bar{z} + \bar{x}\bar{z} + \bar{w}\bar{x}\bar{y} + wx\bar{y}.$$

Door inversie volgt hieruit de produktvorm van de functie S:

$$S = (w + z)\,(x + z)\,(w + x + y)\,(\bar{w} + \bar{x} + y).$$



$$S = yz + wxy + w\bar{x}z + \bar{w}xz$$

$$\bar{S} = \bar{w}\bar{z} + \bar{x}\bar{z} + \bar{w}\bar{x}\bar{y} + wx\bar{y}$$

fig. 2.12. Omzetting van de somvorm naar de produktvorm.

Deze produktvorm bevat één variabele minder dan de equivalente somvorm. Of dit voor een realisatie van de schakeling van belang is, hangt mede af van het type bouwsteen dat gebruikt wordt. Op deze aspecten wordt in de hoofdstukken over combinatorische schakelingen nader ingegaan.

2.2. Schakelfuncties met enkele niet-gespecificeerde functiewaarden

In fig. 2.13 is schematisch weergegeven hoe de stand van een as wordt gedetecteerd door middel van een op de as bevestigde geaarde borstel. Deze borstel sleept langs drie lamellen x, y en z. Gevraagd wordt in de *black box* een schakeling te ontwerpen die ervoor zorgt dat de lamp L brandt als lamel y of lamel z via de borstel geaard is.

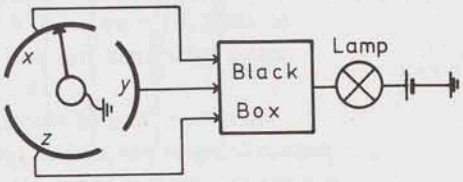


fig. 2.13. Hoekdetectie van een draaiende as.

Om dit probleem tot een oplossing te brengen wordt een waarheidstabel opgesteld. In deze tabel wordt het op te lossen probleem in "nullen en enen" geformuleerd. Aan de logische functie S in tabel 2.2 wordt de logische waarde 1 toegekend als voor de bijbehorende ingangscombinatie de lamp L moet branden. De aan de te ontwerpen schakeling te stellen eisen kan men zo interpreteren dat de lamp L uitsluitend moet kunnen branden (S = 1) als de ingangscombinatie $\bar{x}y\bar{z}$ (y geaard) of $\bar{x}\bar{y}z$ (z geaard) optreedt. Tabel 2.2 geeft de met deze interpretatie van het probleem overeenkomende specificatie.

Uit fig. 2.14 blijkt dat de functie S niet kan worden vereenvoudigd. Het detectieprobleem uit fig. 2.13 is dus opgelost als in de black box van fig 2.13 een schakeling geplaatst wordt die de schakelfunctie

$$S = \bar{x}\bar{y}z + \bar{x}y\bar{z}$$

realiseert.

x	y	z	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

tabel 2.2. Waarheidstabel

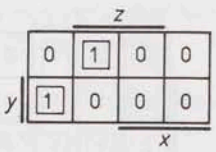


fig. 2.14. Karnaughdiagram

Voor dit eenvoudige detectieprobleem is een simpele oplossing bekend, bestaande uit een directe doorverbinding van de ingangen y en z van de black box met de uitgang ervan. Men komt tot de vraag of de schakelalgebra resp. de waarheidstabel ongeschikt is voor dit soort probleembeschrijvingen. Of is de probleemstelling verkeerd geïnterpreteerd?
In waarheidstabel 2.3 is de oplossing beschreven die met de doorverbindingen correspondeert.

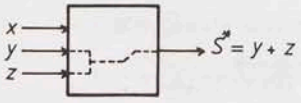


fig. 2.15. Oplossing met doorverbindingen

x	y	z	S*
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

tabel 2.3. Waarheidstabel

Een nadere analyse van de verschillen tussen de tabellen 2.2 en 2.3 leert dat verschillende functiewaarden alleen optreden bij de ingangscombinaties die niet voorkomen! Uit de probleembeschrijving volgt dat de volgende ingangscombinaties niet mogelijk zijn: xyz = 011, 101, 110 en 111. Achter deze ingangscombinaties zijn in de tabellen 2.2 en 2.3 wel functiewaarden gespecificeerd, hetgeen strikt genomen overbodig is. Het is toegestaan functiewaarden die behoren bij niet optredende ingangscombinaties geheel vrij te kiezen. De eenvoud van de te ontwerpen schakeling kan sterk afhangen van een juiste keuze van deze vrije functiewaarden, zoals boven wel is gebleken.
Een dergelijke vrije keuze van een functiewaarde in een waarheidstabel of Karnaughdiagram wordt meestal in een tabel of diagram aangeduid met een liggend streepje (–) of met de letter d van *don't care condition*. Het bovenstaand detectieprobleem is in tabel 2.4 en fig. 2.16 nogmaals gespecificeerd. Nu zijn uitsluitend die functiewaarden gespecificeerd die noodzakelijk uit de gestelde eisen volgen. De overige functiewaarden kunnen vrij gekozen worden. De reeds gevonden eenvoudige oplossing wordt gemakkelijk in fig. 2.16 herkend.

x	y	z	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	d
1	0	0	0
1	0	1	d
1	1	0	d
1	1	1	d

tabel 2.4. Waarheidstabel met don't care condities

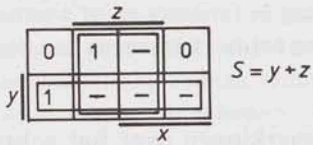


fig. 2.16. Karnaughdiagram met don't care condities.

Bij het specificeren van schakeltechnische problemen komen don't cares zeer vaak voor. Het niet specificeren van deze functiewaarden geeft een zo groot mogelijke vrijheid bij de definitieve realisatie van de schakeling. Tot slot nog enkele voorbeelden van don't care condities welke optreden bij het specificeren van schakeltechnische problemen.

Voorbeeld

Het komt vaak voor dat het aantal mogelijke combinaties van k binaire ingangsvariabelen kleiner is dan het theoretisch maximum 2^k. Zo kunnen schakelaars mechanisch gekoppeld zijn. Fig. 2.17 geeft hiervan een voorbeeld. Van de vier mogelijke combinaties van de schakelaars y en z komen er slechts drie voor, nl. yz = 00, yz = 01 en yz = 10. Denk in dit verband ook aan keuzeschakelaars van radio's, e.d..
Ook in elektronische systemen is het mogelijk door middel van vergrendelingen het aantal optredende combinaties van een groep logische variabelen te beperken.

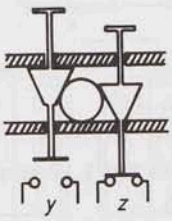


fig. 2.17. Mechanisch gekoppelde schakelaars.

Voorbeeld

Een don't care conditie kan soms moeilijk herkend worden. Bij automatieken bijvoorbeeld komt het veel voor dat een deel van de schakeling in bedrijf moet komen na ontvangst van een voor dat deel specifieke rij ingangssymbolen. Zo is het mogelijk dat een bepaalde schakeling in werking moet komen na ontvangst van een rij van drie symbolen, mits deze gelijk is aan 010. (Hoe deze rij symbolen wordt ontvangen, blijft op dit moment buiten beschouwing.)
Als nu bovendien gegeven is dat in elke rij van drie symbolen slechts één 1 voorkomt, dan reduceert het aantal mogelijke ingangsrijen zich van acht tot drie. Dit geeft reeds aanleiding tot don't care conditities. Moeilijker is het ook te onderkennen dat, onder de gegeven beperking van het aantal mogelijke rijen symbolen tot 100, 010 en 001, het voor de detectie slechts van belang is te onderzoeken of het tweede symbool een 1 is. Het eerste en derde symbool zijn in dit geval niet relevant voor de te nemen beslissing!

Uit dit voorbeeld blijkt dat het efficiënt specificeren van schakeltechnische problemen niet altijd even gemakkelijk is. Het omzetten van een woordelijke beschrijving in formules en/of waarheidstabellen is niet eenvoudig, vooral met betrekking tot het herkennen van don't care condities.

2.3. Opmerkingen over het gebruik van Karnaughdiagrammen

De keuze van de don't care condities

Het diagram van fig. 2.18 specificeert een schakelfunctie S. Een nadere beschouwing van dit diagram leert dat er twee groepen van vier enen kunnen worden gevormd. De eenvoudigste somvorm van de functie S is

$$S = \bar{w}x + \bar{y}\bar{z}.$$

Een van de don't care functiewaarden wordt dus als 1 gekozen, de andere als 0 ($f_4 = 1$, $f_{11} = 0$). Het is een misvatting te menen dat voor alle don't care condities van een schakelfunctie dezelfde waarde dient te worden gekozen!

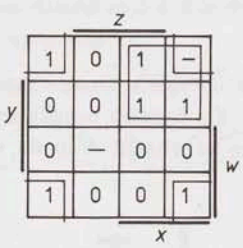


fig. 2.18. Karnaughdiagram

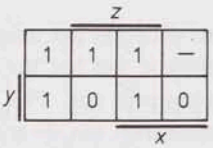


fig. 2.19. Karnaughdiagram

Niet equivalente vormen van schakelfuncties

Voor de in fig. 2.19 gespecificeerde functie zijn de eenvoudigste som- en produktvorm

$$S = \bar{y} + \bar{x}\bar{z} + xz \qquad \text{en} \qquad S = (x + \bar{y} + \bar{z})\,(\bar{x} + z).$$

Bij de somvorm is de don't care functiewaarde als 1 geïnterpreteerd, terwijl dit bij de produktvorm juist omgekeerd is. Hoewel vanuit schakeltechnisch standpunt bezien beide vormen hetzelfde probleem beschrijven, moet worden geconstateerd dat deze gereduceerde formules algebraïsch niet equivalent zijn. Ook bij het bepalen van de functie S en zijn inverse $\bar{S}$ uit een Karnaughdiagram met don't care functiewaarden kan dit optreden. Als voor de don't cares in beide gevallen niet dezelfde waarde wordt gekozen, dan zijn de formules voor S en $\bar{S}$ algebraisch niet elkaars inverse. De verschillen bestaan uiteraard alleen voor de don't care functiewaarden.

Het uitlezen van Karnaughdiagrammen

Het patroon van nullen en enen in fig. 2.20 staat bekend onder de naam "de molenwieken". Men komt bij het uitlezen van dit diagram in de verleiding eerst de term yz te omranden, teneinde in één keer zoveel mogelijk enen mee te nemen.

Vervolgens worden de termen $\bar{w}xz$, $\bar{w}\bar{x}y$, $w\bar{x}z$ en wxy gevormd. In de somvorm

$$S = yz + \bar{w}xz + \bar{w}\bar{x}y + w\bar{x}z + wxy$$

is de term yz echter overbodig. Elke 1 van de term yz wordt ook door een van de andere omrandingen omvat. Deze redundante termen kunnen vermeden worden, als bij het uitlezen een zekere volgorde in acht wordt genomen.

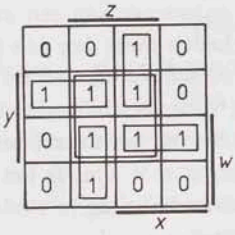


fig. 2.20. Karnaughdiagram.

Van de in fig. 2.21.a gespecificeerde schakelfunctie moet de eenvoudigste somvorm worden bepaald. De volgorde van het aanbrengen van omrandingen is hierbij:

1. Omrand eerst die enen die niet met andere samengenomen kunnen worden (term $\bar{w}\bar{x}\bar{y}\bar{z}$ in fig. 2.21.b).
2. Omrand vervolgens alle termen die uit twee enen bestaan, waarvan er minstens een op slechts één manier met een andere kan worden gecombineerd (de termen $\bar{w}yz$ en $wy\bar{z}$ in fig. 2.21.c).
3. Doe hetzelfde met alle groepen van vier enen, enz.

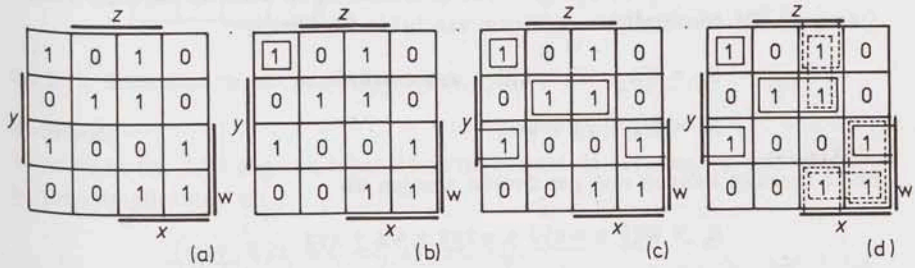


fig. 2.21. Het aanbrengen van omrandingen.

Soms zijn na deze drie stappen alle enen omrand. In fig. 2.21 is dit niet zo. De overblijvende functiewaarden in fig. 2.21.c kunnen elk op meer dan een wijze omrand worden.

Voor de functie in fig. 2.21 bestaan er drie equivalente minimale oplossingen:

$$S = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}yz + wy\bar{z} + \bar{w}xz + wx\bar{y}$$

$$S = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}yz + wy\bar{z} + x\bar{y}z + wx\bar{y}$$

$$S = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}yz + wy\bar{z} + x\bar{y}z + wxz$$

Het uitlezen van Karnaughdiagrammen kan men formeel aanpakken. De schakelalgebra blijkt dan weer een uitstekend hulpmiddel voor het formuleren van het

probleem. In de praktijk kan men voor diagrammen tot ca. vijf variabelen met bovenstaande regels volstaan. Een meer formele behandeling van dit probleem is echter op zijn plaats, omdat het daarbij optredende dekkingsprobleem ook in vele andere schakeltechnische algoritmen optreedt. De hierbij gebruikte *Petrickfunctie* wordt behandeld in de volgende paragraaf.

Logische schakelingen met meer dan één uitgang

Het komt vaak voor dat bij het ontwerpen van een digitale schakeling op dezelfde verzameling binaire ingangsgrootheden meer dan één logische functie is gedefinieerd. Tijdens het bouwen van de schakeling bestaat dan meestal de mogelijkheid gedeelten van de schakeling voor verschillende uitgangsfuncties te benutten, hetgeen kan leiden tot een reductie van het aantal benodigde bouwstenen. Vooral wanneer er don't cares in het spel zijn, is het niet altijd even gemakkelijk de meest economische schakeling te vinden.

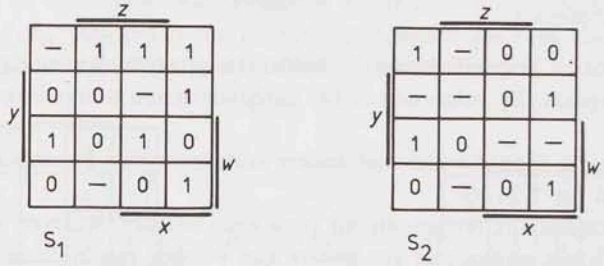


fig. 2.22. Specificatie van een schakeling met twee uitgangen S_1 en S_2.

De schakelfuncties S_1 en S_2 zijn beide gedefinieerd op dezelfde ingangsverzameling {w,x,y,z}. De eenvoudigste somvorm van beide functies is

$$S_1 = \bar{w}x + \bar{w}\bar{y} + x\bar{y}\bar{z} + xyz + w\bar{x}y\bar{z}$$

$$S_2 = \bar{w}\bar{x} + y\bar{z} + wx\bar{z}$$

Beide functies kunnen ook geschreven worden als

$$S_1 = \underline{\bar{w}\bar{x}\bar{y}} + \underline{wx\bar{y}\bar{z}} + \underline{w\bar{x}y\bar{z}} + \bar{w}x + xyz$$

$$S_2 = \underline{\bar{w}\bar{x}\bar{y}} + \underline{wx\bar{y}\bar{z}} + \underline{y\bar{z}}.$$

Het is nu mogelijk enkele termen van beide schakelfuncties gecombineerd te realiseren. Deze termen zijn onderstreept. Bij een realisatie van een schakeling moet men dus niet automatisch van de eenvoudigste vorm van de formule uitgaan. De te gebruiken bouwstenen spelen bij dit keuzeproces uiteraard een belangrijke rol. In het vervolg zal daarom bovenstaand probleem nog herhaaldelijk aan de orde komen.

2.4. Het formuleren van dekkingsproblemen

Bij het bepalen van de eenvoudigste somvorm van een schakelfunctie treedt het probleem op dat een minterm vaak op meer dan een wijze bevat kan worden in een produktterm. Gevraagd wordt in dit geval een kleinste verzameling produkten te selecteren, zodat alle mintermen waarvoor de bijbehorende functiewaarde $f_i = 1$ is minstens één maal bevat zijn. Dit type dekkingsproblemen komt

bij vele algoritmen in de schakeltechniek als deelprobleem voor, hetgeen reden is voor een aparte behandeling ervan. Als toepassing wordt het uitlezen van een Karnaughdiagram in vier variabelen genomen.

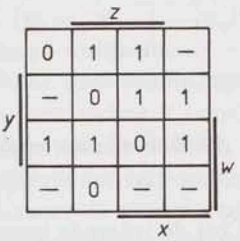
De diverse produkttermen van een schakelfunctie zijn niet alle even relevant voor de minimale somvorm. Deze termen worden daarom onderscheiden als:

minterm (minterm) ⟷ een produktterm waarin alle variabelen één maal voorkomen.

produktterm of *implicant (implicant)* ⟷ een tot de functie behorend produkt van een of meer der variabelen.

priemterm of *priemimplicant (prime implicant)* ⟷ een produktterm die niet met andere produkttermen kan worden gecombineerd tot een produktterm in minder variabelen.

essentiële priemterm of *priemimplicant (essential prime implicant)* ⟷ een term die noodzakelijk voorkomt in de minimale somvorm van een schakelfunctie.



Priemimplicanten:

$w\bar{x}y$ $\bar{w}\bar{y}z$ $\bar{w}x$ $w\bar{z}$ $x\bar{y}$ $x\bar{z}$ $y\bar{z}$

fig. 2.23. Schakelfunctie met priemimplicanten.

Voorbeeld

Voor de in fig. 2.23 gegeven schakelfunctie bestaat de verzameling priemtermen of priemimplicanten uit

$$\{w\bar{x}y,\ \bar{w}\bar{y}z,\ \bar{w}x,\ w\bar{z},\ x\bar{y},\ x\bar{z},\ y\bar{z}\},$$

terwijl essentiële priemimplicanten zijn

$$\{w\bar{x}y,\ \bar{w}\bar{y}z,\ \bar{w}x\}.$$

Worden de essentiële priemimplicanten nog even buiten beschouwing gelaten, dan kan het dekkingsprobleem als volgt worden geformuleerd:

> Gegeven de verzameling mintermen waarvan de corresponderende functiewaarde $f_i = 1$ is. Selecteer uit de verzameling priemimplicanten een zo klein mogelijk aantal priemimplicanten zodat elke minterm in tenminste één priemimplicant is bevat.

Tabel 2.5 geeft een overzicht van de mogelijkheden. Een dergelijke tabel wordt *dekkingstabel (cover table)* of *priemimplicantentabel* genoemd.

mintermen		m_1	m_5	m_6	m_7	m_{10}	m_{11}	m_{14}
priemimpl.		$\bar{w}\bar{x}\bar{y}z$	$\bar{w}x\bar{y}z$	$\bar{w}xy\bar{z}$	$\bar{w}xyz$	$w\bar{x}y\bar{z}$	$w\bar{x}yz$	$wxy\bar{z}$
A	$w\bar{x}y$	•	•	•	•	×	×	•
B	$\bar{w}\bar{y}z$	×	×	•	•	•	•	•
C	$\bar{w}x$	•	×	×	×	•	•	•
D	$w\bar{z}$	•	•	•	•	×	•	×
E	$x\bar{y}$	•	×	•	•	•	•	•
F	$x\bar{z}$	•	•	×	•	•	•	×
G	$y\bar{z}$	•	•	×	•	×	•	×

tabel 2.5. Dekkingstabel

Uit tabel 2.5 volgen de volgende voorwaarden:
- priemimplicant B ($\bar{w}\bar{y}z$) moet deel uitmaken van de selectie, omdat minterm m_1 uitsluitend door priemimplicant B gedekt kan worden;
- tenminste één van de priemimplicanten B, C of E moet gekozen worden voor de dekking van minterm m_5;

  •
  •
  •
  •

- tenminste één van de termen D, F of G moet gekozen worden voor de dekking van minterm m_{14}.

Deze voorwaarden kunnen worden gecombineerd tot de volgende logische functie:

$$S = B \cdot (B + C + E) \cdot (C + F + G) \cdot C \cdot (A + D + G) \cdot A \cdot (D + F + G)$$

De logische variabelen A tot en met G krijgen hierbij de waarde 1 als de corresponderende priemimplicant tot de gekozen dekking behoort. Het reductieprobleem voor de gegeven schakelfunctie is opgelost voor die combinaties van de variabelen A tot en met G, waarvoor de gedefinieerde functie S de logische waarde 1 heeft. Door middel van bovenstaande logische functie is het dekkingsprobleem elegant geformuleerd.

De rekenregels uit de schakelalgebra zijn van toepassing op de gedefinieerde logische functie:

$$\begin{aligned} S &= B \cdot (B + C + E) \cdot (C + F + G) \cdot C \cdot (A + D + G) \cdot A \cdot (D + F + G) = \\ &= B \cdot C \cdot A \cdot (D + F + G) = \\ &= A \cdot B \cdot C \cdot D + A \cdot B \cdot C \cdot F + A \cdot B \cdot C \cdot G \end{aligned}$$

Uit deze laatste vereenvoudigde somvorm van de functie S volgt dat er drie combinaties van de variabelen A tot en met G bestaan waarvoor S de waarde 1 aanneemt:

A = B = C = D = 1	d.w.z.:	de dekking bestaat uit de priemimplicanten $w\bar{x}y$, $\bar{w}\bar{y}z$, $\bar{w}x$ en $w\bar{z}$.
A = B = C = F = 1	d.w.z.:	de dekking bestaat uit de priemimplicanten $w\bar{x}y$, $\bar{w}\bar{y}z$, $\bar{w}x$ en $x\bar{z}$.
A = B = C = G = 1	d.w.z.:	de dekking bestaat uit de priemimplicanten $w\bar{x}y$, $\bar{w}\bar{y}z$, $\bar{w}x$ en $y\bar{z}$.

Het idee om dekkingsproblemen als logische functie te formuleren is afkomstig van Petrick. Een dergelijke functie wordt daarom wel aangeduid met de naam *Petrickfunctie*. De Petrickfunctie is niet alleen van nut bij het uitlezen van Karnaughdiagrammen. Ook bij het hierna te behandelen Quine-McCluskey algoritme, bij het coderen van toestanden van nog te behandelen level mode en clock mode schakelingen enz., is een nuttig gebruik ervan mogelijk. Bij het formuleren van dekkingsproblemen zijn in het bovenstaande nog enkele aspecten niet naar voren gekomen. Hierover de volgende opmerkingen. We formuleren eerst het dekkingsprobleem algemeen.

> Zij gegeven een verzameling V met elementen $\{v_1, v_2, v_3, \ldots, v_k\}$. Ook is gegeven een verzameling W waarvan de elementen niet-disjuncte (is mogelijk elkaar overlappende) deelverzamelingen van V zijn,
> $W = \{w_1, w_2, w_3, \ldots, w_l\}$
> Gevraagd wordt nu:
> 1. Welke deelverzamelingen van W bevatten tesamen alle elementen van de verzameling V tenminste één keer?
> 2. Welke van de onder 1. gevonden deelverzamelingen van W zijn de kleinste deelverzamelingen, die nog juist V bedekken?
> 3. Als aan elk element van W een kostenfactor wordt toegekend, welke zijn dan de deelverzamelingen van W met de laagste totale kosten waarmee V nog juist bedekt kan worden?

Het overzicht hoe de elementen van de verzameling V bevat zijn in elementen van de verzameling W wordt gegeven in de vorm van een dekkingstabel, waarvan tabel 2.5 een voorbeeld is. Van tabel 2.5 bestaat de verzameling V uit alle mintermen waarbij een functiewaarde 1 behoort; de verzameling W bestaat uit de verzameling priemimplicanten van de gegeven functie.

Het onder 1. gestelde probleem kan geformuleerd worden met de Petrickfunctie. Voor het uitwerken resp. vereenvoudigen van deze functie kan weer gebruik worden gemaakt van de schakelalgebra en het Karnaughdiagram. Daarna is een overzicht van alle mogelijkheden bij de hand.

Het onder 2. gestelde probleem kan worden opgelost door van de onder 1. opgestelde Petrickfunctie die priemimplicanten te selecteren welke een zo gering mogelijk aantal variabelen bevatten. (Een systematische methode om priemimplicanten te bepalen is onderdeel van het Quine-McCluskey algoritme.)

Keren we nogmaals terug naar tabel 2.5. Het meenemen van priemimplicant B hierin is noodzakelijk, omdat anders minterm m_1 niet gedekt is. Hetzelfde geldt voor de priemimplicanten C voor minterm m_7 en A voor minterm m_{11}. Tabel 2.6 geeft aan welke mintermen er door de essentiële priemimplicanten A, B en C gedekt zijn. Het blijkt dat na het aangeven van de essentiële priem-

implicanten en de consequenties ervan voor het dekkingsprobleem het oorspronkelijke probleem uit tabel 2.5 sterk vereenvoudigd is. De "nieuwe" Petrickfunctie wordt

$$S = A \cdot B \cdot C \cdot (D + F + G)$$

	m_1	m_5	m_6	m_7	m_{10}	m_{11}	m_{14}
A	•	•	•	•	⊗	⊗	•
B	⊗	⊗	•	•	•	•	•
C	•	⊗	⊗	⊗	•	•	•
D	•	•	•	•	×	•	×
E	•	×	•	•	•	•	•
F	•	•	×	•	•	•	×
G	•	•	×	•	×	•	×

⊗ noodzakelijk
⊗ gevolg van een gedane keuze

tabel 2.6. Dekkingstabel met essentiële priemimplicanten.

Het is daarom in het algemeen zinvol een dekkingsprobleem in twee fasen op te lossen: Zoek eerst de essentiële elementen op uit verzameling W en bepaal welke elementen van verzameling V hierdoor gedekt zijn. Stel daarna voor de dekking van de overige elementen van verzameling V een Petrickfunctie op.

Een ander aspect van het dekkingsprobleem is dat sommige rijen van een dekkingstabel *dominant* zijn over andere rijen. Een voorbeeld hiervan geeft tabel 2.7. In deze tabel is de rij achter w_i dominant over de rij achter w_j.

	v_1	v_2	v_3	v_4	v_5	v_6
w_i	×	×	•	×	×	×
w_j	•	×	•	×	•	•

tabel 2.7. Dekkingstabel.

Bij het bepalen van een minimale dekking kan de rij w_j buiten beschouwing gelaten worden, tenminste wanneer de waarde van de kostenfunctie van w_i kleiner dan of gelijk is aan die van w_j. Het verdient daarom aanbeveling na te gaan of er dominante rijen in een tabel bestaan (nadat de essentiële elementen uit W bepaald zijn). Het hangt dan van de toegevoegde kostenfunctie af of er rijen buiten beschouwing kunnen blijven.

Voorbeeld
Gegeven is de schakelfunctie

$$S_1 = 0, 6, 7, 8, 13, 14$$

$$S_d = 1, 3, 5$$

waarbij achter S_1 nummers van mintermen staan waarvoor $f_i = 1$ is en achter

S_d nummers van mintermen waarvoor f_i niet gespecificeerd is. De priemimplicantentabel voor deze functie is tabel 2.8. Vergelijk ook fig. 2.24.

		m_0	m_6	m_7	m_8	m_{13}	m_{14}
$\bar{x}\bar{y}\bar{z}$	A	⊗	·	·	⊗	·	·
$\bar{w}\bar{x}\bar{y}$	B	×	·	·	·	·	·
$\bar{w}\bar{z}$	C	·	·	×	·	·	·
$\bar{w}xy$	D	·	×	×	·	·	·
$xy\bar{z}$	E	·	⊗	·	·	·	⊗
$x\bar{y}z$	F	·	·	·	·	⊗	·

tabel 2.8. Dekkingstabel.

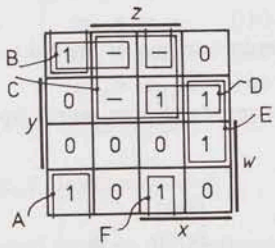


fig. 2.24. Karnaughdiagram.

In tabel 2.8 is rij A dominant over rij B en is rij D dominant over rij C. Tevens is A een essentiële priemterm. Bij gelijke realisatiekosten van de gegeven priemimplicanten mogen de rijen B en C dus geschrapt worden. Echter, priemimplicant C bevat één variabele minder dan priemimplicant D. Bij de nog te behandelen poortschakelingen kost realisatie van D één ingang meer dan een realisatie van C. Soms leidt dit tot een kostenbesparing. Kost in een realisatie een term in twee variabelen k_1 kosteneenheden en een term in drie of vier variabelen k_2 kosteneenheden, met $k_2 > k_1$, dan mag rij B buiten beschouwing blijven, maar rij C niet.

Uit dit voorbeeld blijkt dat in het behandelde dekkingsalgoritme realisatie-aspecten, zoals kostenfactoren, in principe meegenomen kunnen worden.

2.5. Het Quine-McCluskey algoritme

Karnaughdiagrammen zijn een goed hulpmiddel bij het vereenvoudigen van schakelfuncties tot zes variabelen. Daarboven moet het probleem systematischer worden aangepakt. De door Quine-McCluskey opgestelde tabellarische methode leent zich goed voor berekeningen met een computer. Het zal blijken dat er in wezen niet veel verschillen bestaan tussen het werken met een Karnaughdiagram en met de door Quine-McCluskey gegeven methode.
Het reduceren van schakelfuncties geschiedt meestal in twee stappen:

1. Het bepalen van de verzameling der priemimplicanten.
2. Het selecteren van een (minimale) deelverzameling ervan, waarin alle mintermen m_i met $f_i = 1$ tenminste één maal bevat zijn.

Als voorbeeld wordt van de in fig. 2.23 gespecificeerde schakelfunctie de verzameling der priemimplicanten bepaald. Voor deze functie geldt:

$$S_1 = 1, 5, 6, 7, 10, 11, 14$$

$$S_d = 2, 4, 8, 12, 13$$

Beide verzamelingen mintermen van deze functie zijn in tabel 2.9 opgenomen, waarbij met d is aangegeven als de minterm afkomstig is van S_d, d.w.z. van het niet gespecificeerde deel van de schakelfunctie. De mintermen in tabel 2.9 zijn vervolgens in drie groepen ingedeeld. Mintermen uit groep I kunnen slechts met mintermen uit groep II worden gecombineerd en mintermen uit groep III slechts met die uit groep II. Het criterium bij deze indeling is het aantal geïnverteerde variabelen in een (min-)term.
Voor produkttermen wordt een *ternaire* notatie ingevoerd:

a. Komt een variabele gewoon voor in een produkt, dan wordt dit aangeduid met 1.
b. Komt een variabele geïnverteerd voor in een produkt, dan wordt dit aangeduid met 0.
c. Ontbreekt een variabele in een produkt, dan wordt op de betreffende plaats een streepje (–) genoteerd.

$\bar{w}\bar{x}\bar{y}z$	m_1	0001	√	groep I
$\bar{w}\bar{x}y\bar{z}$	m_2	0010	√ d	
$\bar{w}x\bar{y}\bar{z}$	m_4	0100	√ d	
$w\bar{x}\bar{y}\bar{z}$	m_8	1000	√ d	
$\bar{w}x\bar{y}z$	m_5	0101	√	groep II
$\bar{w}xy\bar{z}$	m_6	0110	√	
$w\bar{x}y\bar{z}$	m_{10}	1010	√	
$wx\bar{y}\bar{z}$	m_{12}	1100	√ d	
$\bar{w}xyz$	m_7	0111	√	groep III
$w\bar{x}yz$	m_{11}	1011	√	
$wx\bar{y}z$	m_{13}	1101	√ d	
$wxy\bar{z}$	m_{14}	1110	√	

tabel 2.9. Mintermen met corresponderende ternaire notatie.

Men kan minterm m_1 uit groep I combineren met minterm m_5 uit groep II tot de produktterm (implicant) $\bar{w}\bar{y}z$, welke in de ternaire notatie wordt genoteerd als 0–01. Achter de mintermen m_1 en m_5 in tabel 2.9 wordt dan met √ aangegeven dat zij reeds in een grotere implicant bevat zijn. (Het begrip groter hier op te vatten als: omvat meer mintermen.) Tabel 2.10 bevat het resultaat van alle mogelijke combinaties van mintermen uit groep I met die uit groep II en van mintermen uit groep II met die uit groep III. Het blijkt dat alle mintermen uit tabel 2.9 minstens één maal bevat zijn in een der implicanten in tabel 2.10. In deze tabel zijn de implicanten geordend naar afwezigheid van variabelen. Dit vermindert de hoeveelheid werk bij verdere combinatiepogingen.

$-\bar{x}y\bar{z}$	$m_2 + m_{10}$	-010 √	I
$-x\bar{y}\bar{z}$	$m_4 + m_{12}$	-100 √ d	
$-x\bar{y}z$	$m_5 + m_{13}$	-101 √	
$-xyz$	$m_6 + m_{14}$	-110 √	
$\bar{w}-\bar{y}z$	$m_1 + m_5$	$0-01$	II
$\bar{w}-y\bar{z}$	$m_2 + m_6$	$0-10$ √	
$w-\bar{y}\bar{z}$	$m_8 + m_{12}$	$1-00$ √ d	
$w-yz$	$m_{10} + m_{14}$	$1-10$ √	
$\bar{w}x-\bar{z}$	$m_4 + m_6$	$01-0$ √	III
$\bar{w}x-z$	$m_5 + m_7$	$01-1$ √	
$w\bar{x}-\bar{z}$	$m_8 + m_{10}$	$10-0$ √	
$wx-\bar{z}$	$m_{12} + m_{14}$	$11-0$ √	
$\bar{w}x\bar{y}-$	$m_4 + m_5$	$010-$ √	IV
$\bar{w}xy-$	$m_6 + m_7$	$011-$ √	
$w\bar{x}y-$	$m_{10} + m_{11}$	$101-$	
$wx\bar{y}-$	$m_{12} + m_{13}$	$110-$ √ d	

tabel 2.10. Implicantentabel.

Vervolgens kan geprobeerd worden de implicanten uit tabel 2.10 paarsgewijs te verenigen. Het zal duidelijk zijn dat dit slechts mogelijk is binnen de groepen I, II, III en IV in tabel 2.10, d.w.z. dat twee implicanten slechts dan verenigd kunnen worden als de ontbrekende variabele in beide dezelfde is. Tabel 2.11 bevat het resultaat. Opgemerkt wordt nog dat elk der implicanten in tabel 2.11 twee maal gegenereerd wordt. Ga dit na!

$--y\bar{z}$	$m_2 + m_6 + m_{10} + m_{14}$	$--10$
$\bar{w}x--$	$m_4 + m_5 + m_6 + m_7$	$01--$
$-x\bar{y}-$	$m_4 + m_5 + m_{12} + m_{13}$	$-10-$
$-x-\bar{z}$	$m_4 + m_6 + m_{12} + m_{14}$	$-1-0$
$w--\bar{z}$	$m_8 + m_{10} + m_{12} + m_{14}$	$1--0$

tabel 2.11. Implicantentabel.

Het blijkt dat geen der implicanten uit tabel 2.11 met een der andere verenigd kan worden, alle implicanten uit deze tabel zijn priemimplicanten. Het hierboven beschreven proces wordt daarom gestopt.

Samenvatting

Bij het genereren van priemimplicanten wordt als volgt te werk gegaan:

1. Ga uit van de verzameling mintermen waarvoor de functiewaarde $f_i = 1$ is of waarvoor f_i niet gespecificeerd is. Merk deze laatste mintermen met d. Maak een tabel van alle implicanten die ontstaan door de vereniging van twee mintermen. Geef met d aan als de functiewaarde van *beide* mintermen niet gespecificeerd is.

2. Verenig zoveel mogelijk de implicanten die uit twee mintermen bestaan tot implicanten die vier mintermen omvatten. Merk deze implicanten met d als *alle* omvatte mintermen een don't care functiewaarde bezitten.
3. Herhaal het proces k maal tot implicanten ontstaan (elk 2^k mintermen omvattend) die niet meer met andere kunnen worden gecombineerd.

Tijdens de stappen 1–3 worden alle implicanten die tenminste éénmaal in een grotere implicant bevat zijn, met √ gemerkt. Alle implicanten die niet met √ gemerkt worden, kunnen niet met andere worden gecombineerd, het zijn de priemimplicanten.

In bovenstaand voorbeeld worden de volgende priemimplicanten gevonden:

$\bar{w}\bar{y}z$ en $w\bar{x}y$ (uit tabel 2.10)

en

$\bar{w}x$, $w\bar{z}$, $x\bar{y}$, $x\bar{z}$ en $y\bar{z}$ (uit tabel 2.11)

Ontstaan er priemimplicanten die uitsluitend mintermen bevatten met niet gespecificeerde functiewaarden, dan kunnen deze worden weggelaten.

Vervolgens moet uit de verzameling priemimplicanten een (minimale) deelverzameling worden geselecteerd, waardoor de gehele functie bedekt wordt. Dit probleem is reeds in de vorige paragraaf besproken.

Bovenstaande methode voor het genereren van priemimplicanten beschrijft globaal de door Quine-McCluskey ontwikkelde methode voor het reduceren van schakelfuncties. In deze uiteenzetting is niet gestreefd naar volledigheid. Zo is bijvoorbeeld een vraag of het noodzakelijk is een gegeven schakelfunctie eerst in de mintermvorm te schrijven voordat systematisch de priemimplicanten kunnen worden bepaald. Of kan deze tussenstap geheel of gedeeltelijk worden overgeslagen? Zie hiervoor bijvoorbeeld Tison.
Voor schakelingen met meer dan één uitgang kan een aangepaste versie van de methode van Quine-McCluskey ontwikkeld worden.
Op deze en andere details wordt niet dieper ingegaan.

Opgaven

2.1 Plaats in een Karnaughdiagram voor drie variabelen het cijfer 0 in het vakje van de minterm $\bar{x}y\bar{z}$. Plaats het cijfer 1 in de vakjes die behoren bij mintermen welke slechts in één variabele verschillen met de minterm $\bar{x}y\bar{z}$. Doe hetzelfde met de cijfers 2 en 3 voor mintermen die in twee resp. drie variabelen verschillen met $\bar{x}y\bar{z}$.

2.2 Als bij opgave 2.1 voor een Karnaughdiagram in vijf variabelen. Plaats de cijfers 0 t/m 5 uitgaande van de minterm $\bar{v}\bar{w}\bar{x}\bar{y}\bar{z}$.

2.3 Teken in een Karnaughdiagram de schakelfuncties:

$S_1 = xz + \bar{w}xy + w\bar{x}y + \bar{w}\bar{x}\bar{y}\bar{z}$

$S_2 = w \oplus x \oplus y \oplus z$

$S_3 = wxy \oplus wxz \oplus wyz \oplus xyz$

2.4 Geef in een Karnaughdiagram aan voor welke combinaties van de variabelen w, x, y en z de schakelfunctie S:

$$S = \{\overline{w} + (x + \overline{z})(x + y)\}\{y + x(w + \overline{z})\}$$

de waarde 1 aanneemt.

2.5 Op welke twee rekenregels uit de schakelalgebra berust het vereenvoudigen van schakelfuncties met het Karnaughdiagram?

2.6 Toon met behulp van een Karnaughdiagram aan dat elke schakelfunctie in drie variabelen op slechts één manier is te schrijven als:

$$S = c_0 \cdot 1 \oplus c_1 \cdot z \oplus c_2 \cdot y \oplus c_3 \cdot x \oplus c_4 \cdot yz \oplus c_5 \cdot xz \oplus c_6 \cdot xy \oplus c_7 \cdot xyz$$

Deze ontwikkeling van een schakelfunctie wordt genoemd de Reed-Muller ontwikkeling van een schakelfunctie. De coëfficiënten c_i zijn 0 of 1. Deze ontwikkeling is niet ondubbelzinnig bepaald als ook inverse variabelen in de formule mogen voorkomen. Toon dit aan.

2.7 Welke is de Reed-Muller ontwikkeling van de schakelfunctie

$$S = \bar{x}\bar{y} + x\bar{z} + yz.$$

2.8

S_1

y \ z				
	0	1	1	0
	0	–	0	1
	1	1	0	–
	0	1	1	0

w \ x

S_2

y \ z				
	0	–	0	1
	1	0	0	–
	–	1	0	–
	1	–	1	0

w \ x

S_3

y \ z				
	1	1	–	–
	–	–	1	0
	1	0	1	1
	1	1	1	0

w \ x

Bepaal de meest eenvoudige somvorm en produktvorm van de in de Karnaughdiagrammen gespecificeerde schakelfuncties S_1 t/m S_3.

2.9 Bepaal voor de in 2.8 gegeven schakelfuncties de verzamelingen der priemimplicanten. Stel vervolgens een dekkingstabel op en bepaal minimale dekkingen.

2.10 Een schakeling heeft drie signalen S_1, S_2 en S_3 als ingangssignaal. Gegeven is dat S_1 en S_2 als volgt van twee binaire variabelen afhangen:

$$S_1 = yz \qquad \text{en} \qquad S_2 = y + z.$$

De formule voor het uitgangssignaal P van de schakeling is:

$$P = S_1 + S_2 S_3$$

a. Komen alle ingangscombinaties voor? Licht uw antwoord toe.
b. Bepaal de eenvoudigste formule voor $\bar{P}$.

Literatuur

1. M. Karnaugh, *The Map Method for Synthesis of Combinational Logic Circuits,* Trans. AIEE Vol. 72, part I, 1953, pp. 593–598.
2. G.A. Maley and J. Earle, *The Logic Design of Transistor Digital Computers,* Prentice-Hall Inc., Englewood Cliffs, N.J., 1963.
3. E.J. McCluskey, *Minimization of Boolean Functions,* Bell System Tech. J., Vol. 35, no. 5, November 1956, pp. 1417–1444.
4. S.R. Petrick, *A Direct Determination of the Irredundant Forms of a Boolean Function from the Set of Prime Implicants,* Computation Laboratory of the AFCRC, Bedford, Mass., TR–56–110, April 1956.
5. W.V. Quine, *The Problem of Simplifying Truth Functions,* Am. Math. Monthly, Vol. 59, no. 8, October 1952, pp. 521–531.
6. P. Tison, *Generalization of Consensus Theory and Application to the Minimization of Boolean Functions,* IEEE Trans. on Electronic Comp., Vol. EC–16, no. 4, August 1967, pp. 446–456.
7. E.W. Veitch, *A Chart Method for Simplifying Truth Functions*, Proc. ACM, Pittsburgh, Pa., May 2–3 1952, pp. 127–133.

Voor verdere studie wordt verwezen naar:

8. T.C. Bartee, *Computer Design of Multiple-Output Logical Networks,* IRE Trans. on Electronic Computers, Vol. EC–10, no. 1, March 1961, pp. 21–30.
9. E. Mendelson, *Boolean Algebra and Switching Circuits,* McGraw-Hill Book Company, New York 1970.
10. E. Morreale, *Partitioned List Algorithms for Prime Implicant Determination from Canonical Forms,* IEEE Trans. on Electronic Computers, Vol. EC–16, no. 5, October 1967, pp. 611–620.
11. R. McNaughton and B. Mitchell, *The Minimality of Rectifier Nets with Multiple Outputs Incompletely Specified,* J. Franklin Inst. Vol. 264, December 1957, pp. 457–480.
12. J.R. Slagle, Chin-Liang Chang, R.C. Lee, *A New Algorithm for Generating Prime Implicants,* IEEE Trans. on Computers, Vol. C–19, no. 4, April 1970, pp. 304–310.
13. D.C. Schmidt and L.E. Druffel, *An Extension of the Clause Table Approach to Multi-Output Combinational Switching Networks,* IEEE Trans. on Computers, Vol. C–23, no. 4, April 1974, pp. 338–346.
14. H.A. Vink, B. van den Dolder and J. Al, *Reduction of CC-tables Using Multiple Implication,* IEEE Trans. on Computers, Vol. C–27, no. 10, October 1978, pp. 961–966.

3. KONTAKTSCHAKELINGEN

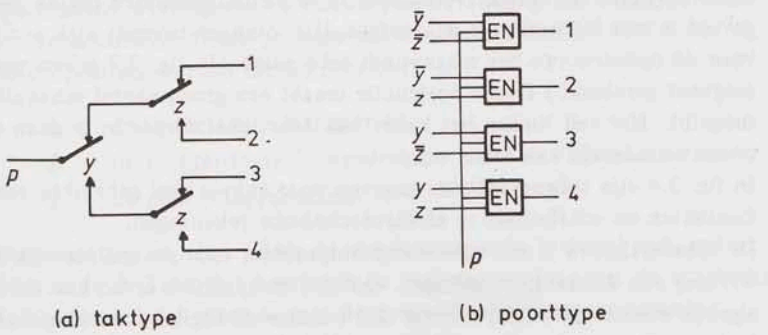
3.1. Inleiding

Digitale schakelingen worden onderscheiden in combinatorische schakelingen en sequentiële schakelingen. Bij een *combinatorische schakeling* wordt het uitgangssignaal op een bepaald tijdstip uitsluitend bepaald door de waarde van de ingangssignalen op datzelfde tijdstip. Een *sequentiële schakeling* onderscheidt zich van een combinatorische schakeling doordat niet in alle gevallen de huidige combinatie van ingangssignalen de waarde van het uitgangssignaal vastlegt. Ook in het verleden aangeboden ingangscombinaties en de volgorde ervan zijn van invloed. Een sequentiële schakeling moet daarom geheugenwerking bezitten. Bovenstaande indeling in combinatorische en sequentiële schakelingen is alleen juist indien men afziet van overgangsverschijnselen, welke o.a. een gevolg zijn van de traagheid van de gebruikte componenten. Deze traagheid, welke berust op fysische verschijnselen, veroorzaakt dat er enige tijd verloopt tussen het moment waarop een ingangsverandering plaatsvindt en het moment waarop de uitgang zich aanpast. Bij het beschrijven van de logische werking van digitale schakelingen wordt daarom uitgegaan van de *statische toestand* van een schakeling. Dit is een situatie waarin de in- en uitgangssignalen niet meer veranderen. Bij een realisatie van een digitale schakeling moet het *dynamische gedrag* van de bouwstenen echter mede in beschouwing genomen worden. Voorlopig blijft het gedrag in *overgangstoestanden* e.d. buiten beschouwing.

Combinatorische schakelingen kunnen naar hun structuur in twee groepen worden ingedeeld:

a. *Schakelingen van het taktype.*
Bij deze schakelingen kan op een of andere wijze een geleidend pad gevormd worden tussen een ingangsklem en een uitgangsklem. De gewenste informatie wordt via het wel of niet geleidend zijn van het netwerk overgedragen. Schakelingen van het taktype worden meestal gerealiseerd met kontakten van relais, schakelaars, enz. De behandeling van dit type van digitale schakelingen vormt het onderwerp van dit hoofdstuk.

b. *Schakelingen van het poorttype.*
Deze schakelingen hebben als kenmerk dat de informatie wordt overgedragen via het wel of niet aanwezig zijn van een bepaald spanningsniveau aan de uitgang, hetgeen bestuurd wordt door één of meer ingangssignalen.



(a) taktype (b) poorttype

fig. 3.1. Verdeler

Schakelingen met elektronische componenten zijn meestal van het poorttype. De behandeling ervan komt in het volgende hoofdstuk aan de orde.
Fig. 3.1 geeft twee uitvoeringen van een schakeling, waarmee een signaal p naar keuze op één van de vier uitgangen kan worden gezet. Bij schakeling 3.1.a wordt de juiste weg door middel van wisselkontakten ingesteld.
In schakeling 3.1.b bepalen twee ingangssignalen y en z welke poort het signaal p doorlaat.

Combinatorische schakelingen van het taktype zijn meestal samengesteld uit kontakten van elektromagnetische schakelaars (relais) en/of uit schakelaars welke met de hand of bijvoorbeeld pneumatisch worden bediend. In de techniek komen vele uitvoeringsvormen voor. Het maakt nogal wat verschil uit of met een kontakt een signaallampje op een tableau wordt bediend, of dat een stroom van ca. 100 A onderbroken wordt. Fig. 3.2 geeft een schets van een doorsnede van een *relais*. Wordt de spoel van het relais bekrachtigd, dan wordt het anker aangetrokken en worden de kontakten gesloten resp. geopend. .

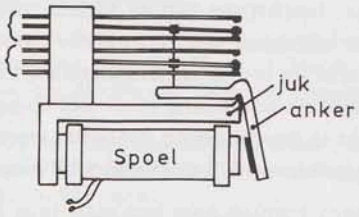


fig. 3.2. Relais met twee wisselkontakten.

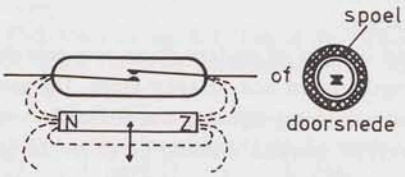


fig. 3.3. Reedrelais met maakkontakt.

In fig. 3.3 is een z.g. *reedrelais* getekend. De werking ervan berust op het feit dat twee kontaktveren van magnetiseerbaar materiaal elkaar aantrekken onder invloed van een magnetisch veld. Verdwijnt dit veld, dan wordt de verbinding verbroken ten gevolge van de voorspanning van de kontaktveren. Bij een reedrelais bevinden de kontaktveren zich in een dichtgesmolten glazen buisje, dat gevuld is met bijvoorbeeld stikstofgas. Hier omheen bevindt zich een spoel die voor de opbouw van het magnetisch veld zorgt. (In fig. 3.3 is een permanente magneet getekend.) Deze constructie maakt een groot aantal schakelbewegingen mogelijk. Het valt buiten het kader van deze tekst dieper in te gaan op constructiedetails van relais e.d.
In fig. 3.4 zijn tekensymbolen gegeven voor enkele veel gebruikte vormen van kontakten en schakelaars in elektrotechnische tekeningen.
De schakelalgebra is een uitstekend hulpmiddel voor de analyse van de logische werking van kontaktschakelingen. Ook bij de synthese ervan kan de schakelalgebra worden gebruikt, hoewel dan behalve de logische werking ook andere factoren mede in beschouwing genomen moeten worden, zoals betrouwbaarheid,

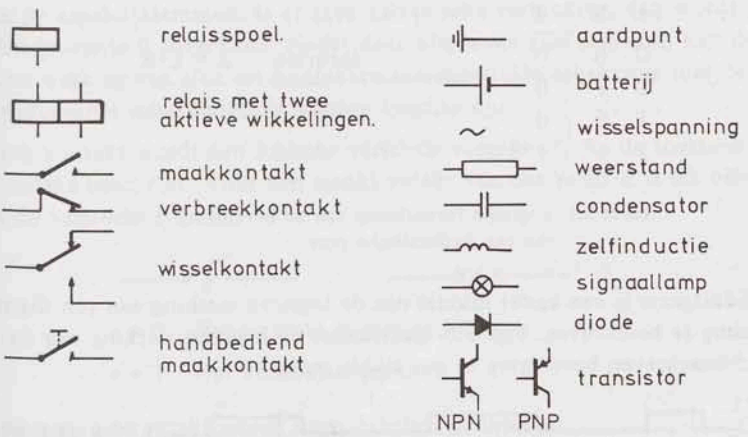


fig. 3.4. Overzicht van enkele symbolen voor elektrotechnische tekeningen.

kostprijs van onderdelen, gemakkelijk onderhoud, enz.. Met het volgende voorbeeld zal worden nagegaan op welke verschillende manieren de logische werking van een digitale schakeling kan worden beschreven.

Voorbeeld

Veiligheidsvoorschrift bij het bedienen van een hydraulische pers is, dat de pers niet mag kunnen werken als de bedieningsman zijn hand onder het bewegende gedeelte van de pers heeft. Deze beveiliging kan worden gerealiseerd door de bediening van de pers zo te ontwerpen dat de bedieningsman met beide handen op twee knoppen moet drukken, die aan weerszijden van het werktableau gemonteerd zijn. Fig. 3.5 geeft schematisch weer hoe deze beveiliging is uitgevoerd.

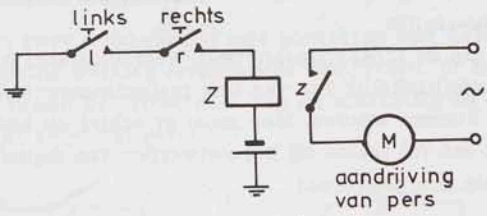


fig. 3.5. Beveiliging hydraulische pers, elektrisch schema.

Uit fig. 3.5 blijkt dat een serieschakeling van kontakten de logische functie EN realiseert. De logische werking van de ontworpen beveiliging kan ook door middel van een *waarheidstabel* of een *formule* worden beschreven.
De volgende proposities worden hierbij gebruikt:

L : de linker knop is ingedrukt.
R : de rechter knop is ingedrukt.
Z : de pers is ingeschakeld.

Zijn deze proposities waar dan wordt de waarheidswaarde 1 toegekend, anders de waarheidswaarde 0. Tabel 3.1 beschrijft de logische werking van de schakeling uit fig. 3.5. Naast tabel 3.1 staat de met deze specificatie overeenkomende logische formule, waarin · de logische EN aanduidt.

L	R	Z
0	0	0
0	1	0
1	0	0
1	1	1

formule: $Z = L \cdot R$

tabel 3.1. Logische formulering van de beveiliging van een hydraulische pers.

Een tijddiagram is een ander middel om de logische werking van een digitale schakeling te beschrijven. Fig. 3.6 specificeert de logische werking van de in fig. 3.5 beschreven beveiliging in een *tijddiagram*.

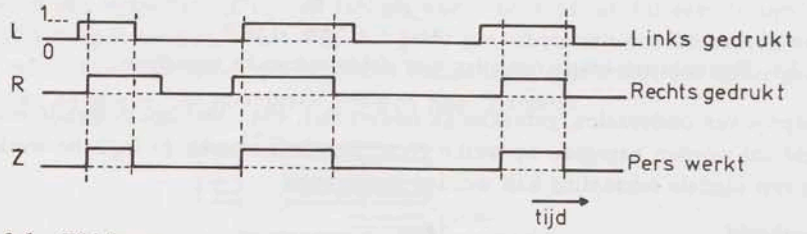


fig. 3.6. Tijddiagram.

De *logische werking* van een digitale schakeling in de statische toestand kan dus op verschillende manieren beschreven worden. Het hangt van de toepassing af voor welke vorm (waarheidstabel, formule, tijddiagram, elektrisch blokschema) wordt gekozen. Zo geeft een *elektrisch schema* meer informatie over bijvoorbeeld de gevolgen van overgangsverschijnselen, terwijl een formule de logische werking efficiënter beschrijft.
Een groot voordeel van de schakelalgebra resp. waarheidstabel is, dat deze beschrijvingswijzen onafhankelijk zijn van vele realisatie-aspecten, en daardoor universeel toegepast kunnen worden. Men moet er echter op bedacht zijn dat ook andere factoren een rol spelen bij het ontwerpen van digitale schakelingen, die niet in de formule zijn uitgedrukt.

3.2. De analyse van schakelingen met kontakten

Bij het in de vorige paragraaf gegeven voorbeeld van een beveiliging moest aan twee voorwaarden tegelijk voldaan worden. Het is gebleken dat dit met een logische EN-functie kan worden uitgedrukt. Een serieschakeling van twee kontakten vormde de realisatie van de logische EN. Het is niet moeilijk in te zien dat een logische OF gerealiseerd kan worden door een parallelschakeling van kontakten.
Voor schakelingen met kontakten wordt in het vervolg aan de proposities

"Het kontakt is gesloten"

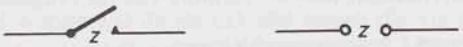
respectievelijk

"De keten van kontakten is gesloten"

de waarheidswaarde 1 toegekend als er een galvanische verbinding bestaat

tussen de aansluitklemmen. Is er geen galvanische verbinding, dan wordt de waarheidswaarde 0 toegekend. Nadat deze afspraken zijn gemaakt, kan de logische werking van elke uit kontakten samengestelde schakeling met de reeds geïntroduceerde schakelalgebra worden beschreven.

Aan elk kontakt wordt een logische variabele toegekend, die de toestand van het kontakt beschrijft. Voor een maakkontakt van een relais Z is dit bijvoorbeeld de variabele z:



z = 0 : Het kontakt is niet gesloten.

z = 1 : Het kontakt is gesloten.

Bij een verbreekkontakt wordt de variabele $\bar{z}$ genoteerd

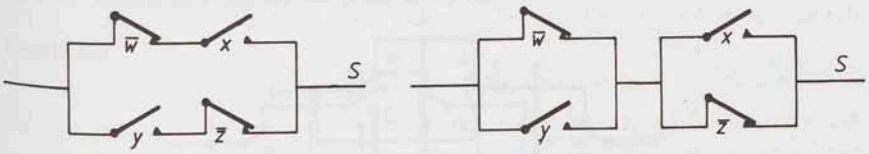
$z = 0 \rightarrow \bar{z} = 1$ Het kontakt is gesloten.

$z = 1 \rightarrow \bar{z} = 0$ Het kontakt is niet gesloten.

Soms wordt bij een verbreekkontakt de niet geïnverteerde variabele z genoteerd als uit de tekening blijkt dat het een verbreekkontakt betreft. De belettering in de tekening verschilt dan met die van de formule, hetgeen tot misverstanden aanleiding kan geven.

Bij een wisselkontakt moet eigenlijk de variabele z aan de maakzijde en $\bar{z}$ aan de verbreekzijde worden genoteerd. In dit geval wordt de variabele $\bar{z}$ aan de verbreekzijde meestal weggelaten.

In fig. 3.7 staan twee schakelingen met kontakten met daaronder formules waarmee de logische werking ervan wordt beschreven. Er bestaat een ondubbelzinnig verband tussen de "structuur" van de schakeling en die van de formules. Dit is echter niet altijd het geval.



$S = \bar{w} \cdot x + y \cdot \bar{z}$ $S = (\bar{w} + y)(x + \bar{z})$

fig. 3.7. Kontaktschakelingen met bijbehorende logische formules.

Voor de in fig. 3.8 gegeven *brugschakeling* kan de bijbehorende formule worden bepaald door alle mogelijke wegen tussen de punten 1 en 2 op te zoeken. Bij de brugschakeling behoort de logische formule

$$S = z \cdot w + z \cdot x \cdot v + y \cdot x \cdot w + y \cdot v$$

Deze formule is niet representatief voor de structuur van het netwerk, maar wel voor de logische werking ervan.

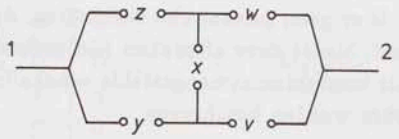


fig. 3.8. Brugschakeling.

In fig. 3.9 is een schakeling getekend die voor wat zijn structuur en de logische werking betreft wel overeenkomt met de formule van de brugschakeling.

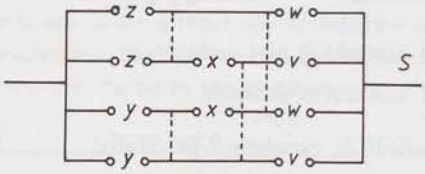


fig. 3.9. Schakeling welke de "brugformule" realiseert.

De beide kontakten y in fig. 3.9 kunnen worden gecombineerd, alsmede de beide kontakten z. De nieuwe structuur wordt dan beschreven door de formule

$$S = z\cdot(w + x\cdot v) + y\cdot(x\cdot w + v)$$

Ook de beide kontakten w en v kunnen worden teruggebracht tot één kontakt. Zie de stippellijnen. Hierna ontstaat een schakeling die identiek is met de brugschakeling in fig. 3.8.

De analyse van de logische werking van een schakeling met kontakten is niet altijd eenvoudig. Soms komen er in een netwerk wegen voor die gemakkelijk over het hoofd gezien worden. Dergelijke wegen worden *sluipwegen* genoemd. In fig. 3.10 is een sluipweg aangegeven die met de term $x\bar{y}z$ in de formule correspondeert. Voor deze gevallen is het gewenst te beschikken over een systematische methode voor het analyseren van kontaktschakelingen.
Een op *knooppuntseliminatie* berustende methode volgt hieronder.

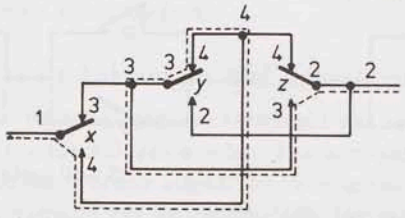


fig. 3.10. Kontaktschakeling met sluipweg.

De eerste stap in het proces is het nummeren van alle *knooppunten*. Onder een knooppunt wordt in dit geval verstaan dat deel van een kontaktschakeling dat permanent galvanisch is verbonden. In fig. 3.10 zijn er vier knooppunten, aangeduid met 1 t/m 4. Vervolgens wordt de schakeling anders getekend. Alle rechtstreekse verbindingen tussen de diverse knooppunten worden hierbij opgespoord en genoteerd. Het verbindingspatroon tussen de diverse knooppunten in fig. 3.10 staat in fig. 3.11.
De tussenpunten worden vervolgens één voor één geëlimineerd. Het weglaten

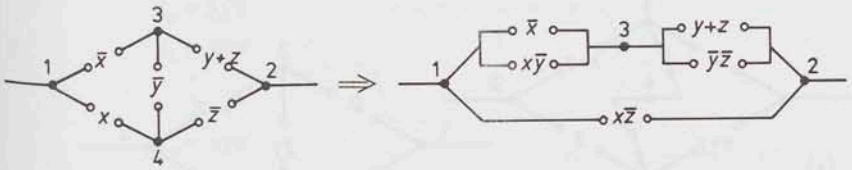


fig. 3.11. Overzicht van de verbindingen tussen de knooppunten.

fig. 3.12. Situatie na eliminatie van één knooppunt.

van knooppunt 4 stelt wel de eis dat alle wegen die via punt 4 lopen op een of andere wijze behouden blijven. In ons voorbeeld zijn dit de weg $x\bar{y}$ van knooppunt 1 via 4 naar 3, de weg $\bar{y}\bar{z}$ van knooppunt 3 via 4 naar 2 en de weg $x\bar{z}$ van knooppunt 1 via 4 naar 2. Fig. 3.12 geeft het resultaat na eliminatie van knooppunt 4. De wegen die vervallen na weglating van knooppunt 4 zijn rechtstreeks aangebracht. Dat dit zo gemakkelijk kan wordt mogelijk gemaakt door het feit dat kontakten de stroom in twee richtingen geleiden.
Tussentijdse vereenvoudiging van de formules die de verschillende verbindingsmogelijkheden tussen de diverse knooppunten aangeven, kan nuttig zijn.
Voor fig. 3.12 geeft dit:

tussen de punten 1 en 3: $S = \bar{x} + \bar{y}$
tussen de punten 3 en 2: $S = 1$
tussen de punten 1 en 2: $S = x\bar{z}$

Het blijkt dat de punten 2 en 3 in de *statische toestand* altijd doorverbonden zijn. Fig. 3.13 beschrijft de situatie die ontstaat na eliminatie van knooppunt 4 respectievelijk knooppunt 3. Hiermee is de bij het netwerk behorende logische formule bepaald.

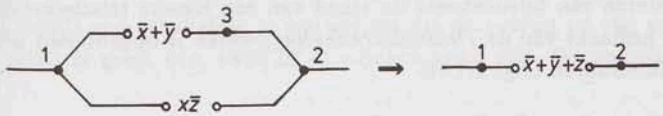


fig. 3.13. Situatie na eliminatie van beide tussenpunten.

Voorbeeld

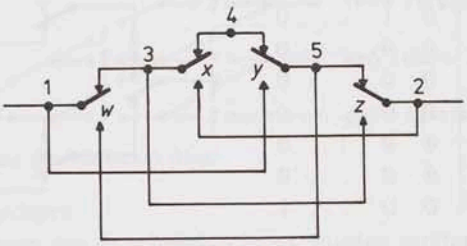


fig. 3.14. Kontaktschakeling.

In fig. 3.14 is een kontaktschakeling getekend, waarin de sluipweg $w\bar{x}\bar{y}z$ voorkomt. In de figuren 3.15.a tot en met 3.15.c wordt met knooppuntseliminatie de formule bepaald die bij dit netwerk behoort.

Opmerking:
Vaak wordt bij de analysemethode met knooppuntseliminatie een matrix-

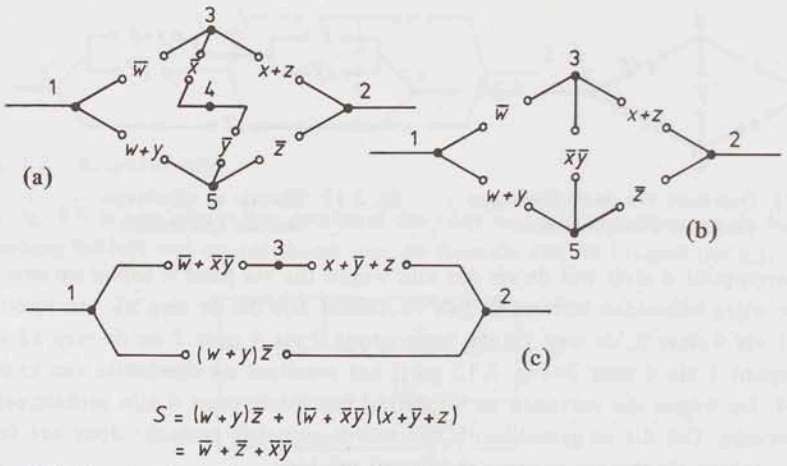


fig. 3.15. Voorbeeld van knooppuntseliminatie.

notatie gebruikt. Voor een schakeling met vier knooppunten is dit een vier-bij-vier matrix. Op de i^e rij in de j^e kolom wordt dan door middel van een logische formule aangegeven welke wegen van knooppunt i naar knooppunt j lopen. Door bewerkingen op deze matrix worden dan de knooppunten geëlimineerd. Deze methode wordt niet verder besproken.

3.3. Enkele voorbeelden van kontaktschakelingen

Voor een aantal problemen in de schakeltechniek zijn "pasklare" oplossingen bekend. Enkele ervan volgen hieronder.

Decodeerschakelingen.
Bij het decoderen van bijvoorbeeld de stand van een binaire telschakeling wordt veel gebruik gemaakt van de *Piramideschakeling*, welke is opgebouwd uit een aantal wisselkontakten (fig. 3.16):

i	x	y	z	S_0	S_1	...	S_7
0	0	0	0	1	0	...	0
1	0	0	1	0	1	...	0
2	0	1	0	0	0	...	0
3	0	1	1	0	0	...	0
4	1	0	0	0	0	...	0
5	1	0	1	0	0	...	0
6	1	1	0	0	0	...	0
7	1	1	1	0	0	...	1

tabel 3.2. Waarheidstabel

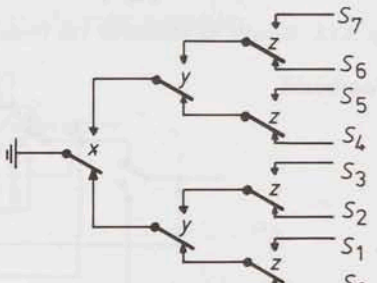


fig. 3.16. Piramide schakeling.

Tabel 3.2 specificeert de werking van zo'n decodeerschakeling. Een logische 1 in de uitgangskolommen betekent dat voor die ingangscombinatie de betreffende uitgang geaard moet zijn.
De logische formules voor S_0 tot en met S_7 zijn :

$$S_0 = \bar{x}\bar{y}\bar{z}$$
$$S_1 = \bar{x}\bar{y}z$$
$$\vdots$$
$$S_7 = xyz$$

De schakeling in fig. 3.16 realiseert het gewenste logische gedrag. Het basisidee van de piramideschakeling kan bij vele decodeerproblemen worden gebruikt. Enkele toepassingen volgen hierna. Over de verdeling van de wisselkontakten over de verschillende relais kan het volgende worden opgemerkt. Bij de in fig. 3.16 gegeven schakeling is de verdeling als volgt:

relais X : één wisselkontakt.
relais Y : twee wisselkontakten.
relais Z : vier wisselkontakten.

De volgende verdeling kan onder omstandigheden gunstiger zijn:

Voor het decoderen van een binaire teller met 16 standen kan bij een geschikt gekozen toewijzing bijvoorbeeld worden volstaan met maximaal vijf wisselkontakten van eenzelfde relais.

Voorbeeld
Gevraagd wordt een schakeling te ontwerpen die detecteert of van een groep van drie relais er geen, één, twee of drie bekrachtigd zijn. De oplossing staat in fig. 3.17.

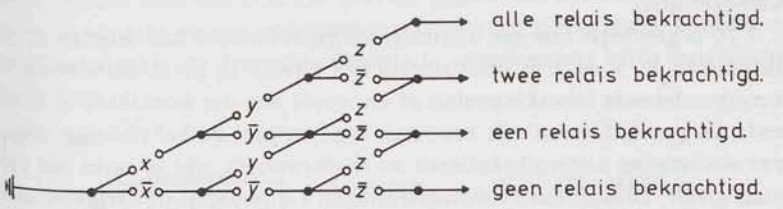


fig. 3.17. Toepassing piramideschakeling.

Vergrendelschakelingen.
In vele toepassingen van de schakeltechniek worden vergrendelingen gebruikt, die ervoor moeten zorgen dat bijvoorbeeld bedieningsfouten geen desastreuze gevolgen kunnen hebben. Zo moet bij sterkstroomtoepassingen verhinderd worden dat een onder spanning staande geleider geaard wordt, omdat er anders ernstige kortsluitingen ontstaan. Dit kan worden bereikt door in serie met de (handbediende) aardschakelaar een verbreekkontakt op te nemen, dat geopend wordt als er spanning aanwezig is (fig. 3.18). Anderzijds moet het, als de geleider geaard is, onmogelijk zijn de spanning in te schakelen.

fig. 3.18. Voorbeeld van vergrendeling.

Voorbeeld

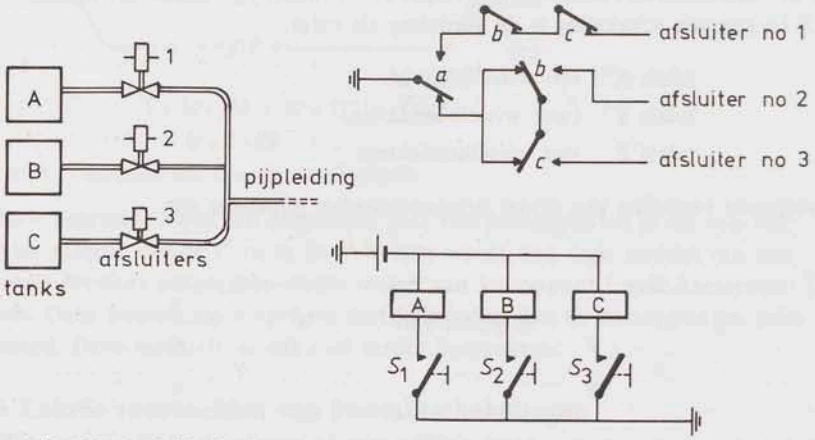


fig. 3.19. Vergrendeling in een transportsysteem.

In fig. 3.19 is een voorbeeld gegeven van een onderlinge vergrendeling van afsluiters in een transportleiding voor vloeistoffen. Hierbij is als eis gesteld dat slechts één afsluiter tegelijk mag worden geopend. Wordt abusievelijk ook de schakelaar voor een der andere omgezet, dan moet het gehele transport worden onderbroken. De getekende schakeling voldoet aan deze eis.

Alarmschakelingen.

In fig. 3.20 is geschetst hoe een alarmsignaal geproduceerd kan worden als van drie automaten X, Y en Z er door storing een uitvalt. In geval van storing valt het corresponderende bewakingsrelais af en wordt een der kontakten x, y of z geopend. Als gevolg hiervan valt het relais R af en gaat de bel rinkelen. *Het verdient aanbeveling alarmschakelingen zo te ontwerpen, dat in geval van storing een relais afvalt.* Draadbreuk, voedingsstoringen e.d. worden in dat geval ook gedetecteerd.

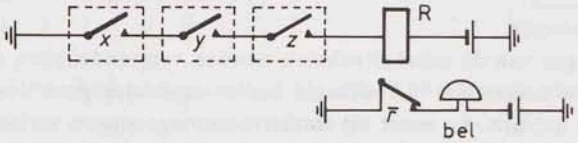


fig. 3.20. Alarmschakeling.

Vergelijkschakelingen

Bij een aantal toepassingen moet worden gedetecteerd of de standen van twee delen van een uit relais en/of kontakten opgebouwde schakeling gelijk zijn.

Hierbij kan bijvoorbeeld gedacht worden aan een schakeling die pulsen telt en na een instelbaar aantal ontvangen pulsen een detectiesignaal moet afgeven.

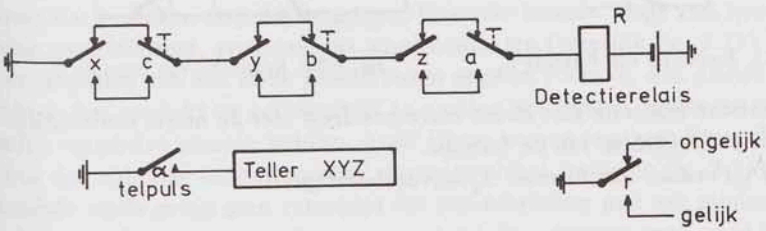


fig. 3.21. Vergelijkschakeling.

In fig. 3.21 is een oplossing geschetst, waarbij met drie wisselkontakten a, b en c de te detecteren stand ingesteld kan worden.

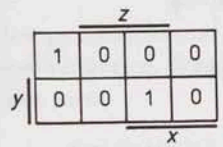
Voor verdere voorbeelden wordt verwezen naar de literatuuropgaven achter dit hoofdstuk.

3.4. Het ontwerpen van schakelingen met kontakten

Het is meestal niet gemakkelijk een eenvoudige realisatie in kontakten te ontwerpen voor een gegeven logische functie. Vooral het mogelijk optreden van sluipwegen, welke vaak niet als weg tussen de ingang en de uitgang van de schakeling bedoeld zijn, kan er oorzaak van zijn dat het gewenste gedrag niet wordt gerealiseerd. Ook overgangsverschijnselen moeten in beschouwing worden genomen. In dit hoofdstuk wordt volstaan met enkele opmerkingen over deze problemen.

Bij het ontwerpen van kontaktschakelingen kan men uitgaan van bijvoorbeeld de minimale somvorm van een schakelfunctie. Deze formule kan dan rechtstreeks gerealiseerd worden als parallelschakeling van een aantal in serie geschakelde ketens, en wel één tak voor elke produktterm in de formule. Het is echter meestal zinvol te streven naar het gebruik van zoveel mogelijk wisselkontakten, omdat deze een efficiënt gebruik maken van het beschikbare materiaal. Een maakkontakt en een verbreekkontakt bestaan elk uit twee kontaktveren, terwijl het slechts uit drie veren bestaande wisselkontakt beide mogelijkheden combineert.

Voorbeeld



Somvorm : $S_1 = \bar{x}\bar{y}\bar{z} + xyz$

Produktvorm: $S_2 = (x + \bar{y})(y + \bar{z})(\bar{x} + z)$

$S_3 = (x + \bar{z})(\bar{x} + y)(\bar{y} + z)$

fig. 3.22. Karnaughdiagram.

In fig. 3.22 is een logische functie gespecificeerd. Voor een realisatie in kontakten kan bijvoorbeeld worden uitgegaan van de somvorm S_1. Een met deze vorm van de schakelfunctie overeenkomende realisatie staat in fig. 3.23.

Een alternatieve schakeling die met formule S_2 overeenkomt staat in fig. 3.24. Hierbij zijn weer enkele kontakten tot wisselkontakten gecombineerd.

Bij het ontwerpen van kontaktschakelingen komt het vaak voor dat de meest

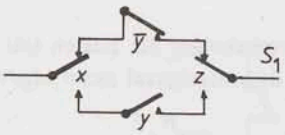


fig. 3.23. Realisatie van formule S_1.

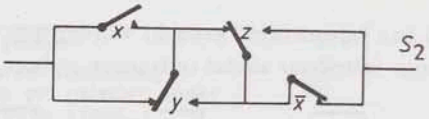


fig. 3.24. Realisatie van formule S_2.

economische realisatie niet direct correspondeert met de meest eenvoudige som- of produktvorm van de formule.
Verder uitwerken van formule S_2 bijvoorbeeld geeft:

$$S_2^* = (x + \bar{y})(yz + \bar{x}\bar{z})$$

Fig. 3.25 geeft de bij deze vorm van de formule behorende realisatie. Deze kost slechts negen veren (drie wisselkontakten).

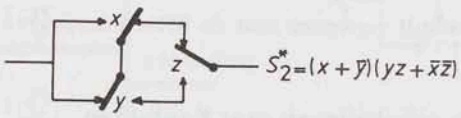


fig. 3.25. Realisatie met drie wisselkontakten.

Voorbeeld

Een bekend probleem uit de klassieke schakeltechniek is het in- en uitschakelen van apparatuur op meer dan één bedieningsplaats (de verlichting in trappenhuizen). Fig. 3.26 geeft twee varianten van een schakeling waarmee op drie verschillende bedieningsplaatsen in- en uitgeschakeld kan worden. De oplossing in fig. 3.26.a maakt gebruik van *kruisschakelaars*, die in fig. 3.26.b is samengesteld uit wisselkontakten.

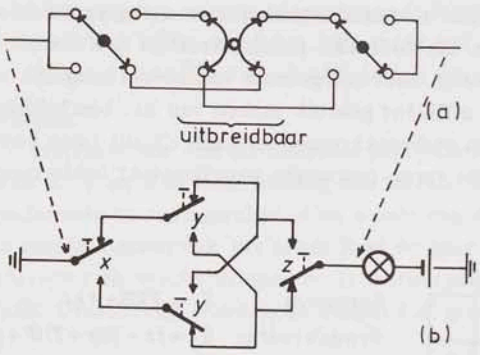


fig. 3.26. Schakeling voor trappenhuisverlichting.
Hotelschakeling.

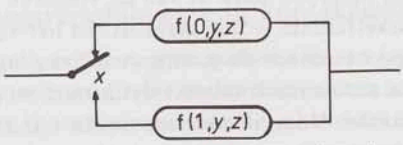
Het voordeel van de in fig. 3.26.a geschetste oplossing ten opzichte van die in fig. 3.26.b is, dat de eerste eenvoudig uitbreidbaar is. De oplossing met wisselkontakten gaat snel zeer veel materiaal kosten bij toenemend aantal bedieningsplaatsen. De verificatie van deze laatste bewering wordt aan de lezer overgelaten.

Uit deze voorbeelden blijkt dat het ontwerpen van combinatorische schakelingen met kontakten een gecompliceerd probleem is. Er zijn wel enige vuistregels te formuleren, waarmee redelijke resultaten behaald kunnen worden. Bij een in de

somvorm gegeven formule bijvoorbeeld worden vaak eenvoudige schakelingen gevonden als men zoveel mogelijk termen met dezelfde variabele (of diens inverse) laat beginnen resp. laat eindigen. Daarmee kunnen reeds veel kontakten worden gecombineerd, eventueel tot wisselkontakten (vergelijk fig. 3.23). Wordt uitgegaan van een in de produktvorm gegeven formule, dan dienen de termen in het produkt zo gerangschikt te worden dat termen, welke zo veel mogelijk variabelen gemeen hebben, naast elkaar komen te staan. Eventueel kunnen dan naburige termen worden vermenigvuldigd (vergelijk fig. 3.25). Genoemde regels geven geen zekerheid dat een schakeling met een minimaal aantal kontakten wordt gevonden, maar wel zal dit minimum vaak goed benaderd worden.

Realisatie van schakelfuncties via splitsing naar variabelen

Er bestaan systematische methoden voor het ontwerpen van kontaktschakelingen. In het volgende voorbeeld wordt een van deze methoden, welke berust op de splitsing van een schakelfunctie naar één van zijn variabelen, globaal behandeld. De splitsing van een schakelfunctie naar een van zijn variabelen (zie het hoofdstuk Schakelalgebra) kan direct gerealiseerd worden via een wisselkontakt:



$$f(x,y,z) = \bar{x}\cdot f(0,y,z) + x\cdot f(1,y,z)$$

fig. 3.27. Het splitsen van een schakelfunctie met een wisselkontakt.

Vooral wanneer de te realiseren schakelfunctie is uitgedrukt in een groot aantal variabelen resp. termen kan met vrucht gebruik worden gemaakt van deze mogelijkheid. Onderstaand voorbeeld geeft globaal deze realisatiemethode weer.

Voorbeeld

De schakelfunctie

$$S = \bar{w}\bar{x}z + \bar{w}\bar{y}z + wx\bar{z} + wy\bar{z} + \bar{x}\bar{y}z + xy\bar{z}$$

kan worden gerealiseerd met behulp van de ontwikkeling van de functie naar één of meer variabelen. Wordt begonnen met een splitsing naar de variabele w, dan vindt men:

$$\begin{aligned} S &= \bar{w}\cdot f(0,x,y,z) + w\cdot f(1,x,y,z) = \\ &= \bar{w}\cdot(\bar{x}z + \bar{y}z + \bar{x}\bar{y}z + xy\bar{z}) + w\cdot(x\bar{z} + y\bar{z} + \bar{x}\bar{y}z + xy\bar{z}) = \\ &= \bar{w}\cdot(\bar{x}z + \bar{y}z + xy\bar{z}) + w\cdot(x\bar{z} + y\bar{z} + \bar{x}\bar{y}z). \end{aligned}$$

In een schakeling wordt dit (fig. 3.28):

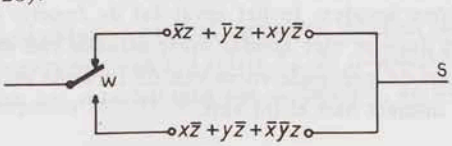


fig. 3.28. Splitsing naar één variabele.

Wordt de splitsing voortgezet naar bijvoorbeeld de variabele x, dan vindt men:

$$S = \bar{w}\bar{x}\cdot f(0,0,y,z) + \bar{w}x\cdot f(0,1,y,z) + w\bar{x}\cdot f(1,0,y,z) + wx\cdot f(1,1,y,z)$$
$$= \bar{w}\bar{x}\cdot z + \bar{w}x\cdot(\bar{y}z + y\bar{z}) + w\bar{x}\cdot(\bar{y}z + y\bar{z}) + wx\cdot\bar{z}.$$

Hieruit blijkt dat $f(0,1,y,z) = f(1,0,y,z)$. In dit geval mogen deze takken worden gecombineerd. De hiermee overeenkomende schakeling staat in fig. 3.29.

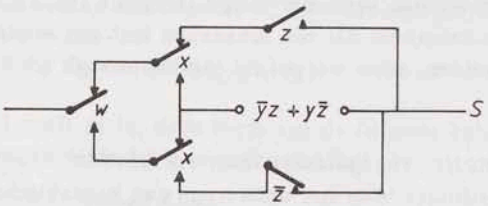


fig. 3.29. Situatie na afsplitsing van twee variabelen.

In de bovenste en onderste tak van fig. 3.29 blijft slechts de variabele z over. De definitieve schakeling, welke nu vrij gemakkelijk kan worden gevonden, is gegeven in fig. 3.30.

Het eindresultaat hangt bij deze methode sterk af van de volgorde van de variabelen waarnaar de gegeven schakelfunctie wordt gesplitst. In het algemeen moet men alle mogelijke volgorden proberen om de gunstigste schakeling te vinden. Tevens wordt bij deze methode automatisch bereikt dat zoveel mogelijk gebruik gemaakt wordt van wisselkontakten. Voor schakelfuncties in een groot aantal variabelen is het zinvol te streven naar een splitsing naar die variabelen waarbij zoveel mogelijk identieke "subfuncties" *(residuen)* optreden.

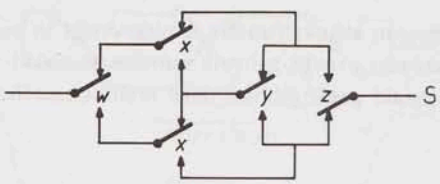


fig. 3.30. Resulterende schakeling.

Vergelijk hierboven de subfuncties $f(0,1,y,z)$ en $f(1,0,y,z)$. Deze kunnen dan onder zekere condities tot één tak worden gecombineerd. Ook behoeft in verschillende takken de verdere splitsing niet steeds naar dezelfde variabele te worden voortgezet.

Opmerking 1

Schakelfuncties met don't care condities vormen een apart probleem. Het is mogelijk bij elke subfunctie alle don't care condities te noteren. Na elke splitsing dient onderzocht te worden of er een combinatie van keuzes van don't care's mogelijk is, waarbij de diverse takken zoveel mogelijk gecombineerd kunnen worden. In het geval dat de functie don't care condities bezit, moet men daarom niet zonder meer uitgaan van een *minimale formule*. Het verband tussen de minimale vorm van de formule en de meest economische schakeling ligt immers niet altijd vast.

Opmerking 2

Indien na een aantal splitsingen er twee of meer gelijke *subfuncties* (of combineerbare in het geval van don't care conditions) overblijven moet onderzocht worden of het combineren tot één tak geen sluipwegen introduceert! In fig. 3.29 was dit niet het geval omdat één der takken een kontakt x bevatte en de andere begon met $\bar{x}$. Dit is echter niet altijd het geval. Zou men door het samennemen sluipwegen introduceren, dan moet men:

a. òf deze twee takken niet combineren
b. òf door middel van een of meer diodes er voor zorgen dat de stroom slechts in de gewenste richting de schakeling doorloopt (zie fig. 3.31).

fig. 3.31. Het verwijderen van sluipwegen.

Zie voor verdere details van de boven geschetste methode de literatuuropgave van Scheinman en Roginsky.

3.5. Overgangsverschijnselen in kontaktschakelingen

Als een digitaal probleem door middel van een logische formule is beschreven, dan kan men over de diverse realisatie-aspecten gaan nadenken. Een ervan is de keuze der componenten, waarmee de schakeling moet worden gerealiseerd. Heeft men bijvoorbeeld de beschikking over samengeperste lucht (werkplaatsen e.d.) dan kan men denken aan pneumatisch werkende bouwstenen.
Vaak zullen elektronische bouwstenen (geïntegreerde circuits – losse componenten) de voorkeur verdienen. Bij al deze types componenten treedt als realisatieprobleem op de vraag in hoeverre het door middel van de logische formules gespecificeerde gedrag ook werkelijk gerealiseerd wordt. *Overgangsverschijnselen* en hun gevolgen spelen hierbij een belangrijke rol.

Voorbeeld

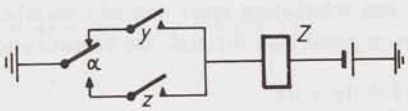


fig. 3.32. Kontaktschakeling.

Bij de in fig. 3.32 getekende schakeling komt het relais Z op via de verbreekzijde van het wisselkontakt a, indien aan zekere voorwaarden is voldaan. Deze voorwaarden zijn hier aangeduid met het maakkontakt y. De bedoeling van de schakeling is, dat er na het opkomen van het relais Z een zogenaamde *houdketen* wordt opgebouwd via de maakzijde van het wisselkontakt a en het eigen kontakt z van het relais Z.
Uit fig. 3.32 blijkt dat, als het wisselkontakt a relatief langzaam schakelt (in verhouding tot de opkom- en afvalvertraging van het relais Z), de mogelijkheid bestaat dat het relais Z afvalt tijdens het omschakelen van a. Immers, de bekrach-

tigingsketen wordt tijdelijk onderbroken. Duurt deze onderbreking lang genoeg, dan valt het relais Z af en opent het kontakt z. De gewenste houdketen komt niet tot stand.

Er bestaan voor dit type veel voorkomende problemen twee oplossingen, een mechanische en een schakeltechnische. Het is mogelijk een relais zo te construeren dat van een wisselkontakt eerst de maakzijde gesloten wordt, *voordat* de verbreekzijde opent. Een dergelijk kontakt wordt een *maak-voor-verbreekkontakt* genoemd:

$$\begin{matrix} \bar{a} = 1 \\ a = 0 \end{matrix} \Rightarrow \begin{matrix} \bar{a} = 1 \\ a = 1 \end{matrix} \Rightarrow \begin{matrix} \bar{a} = 0 \\ a = 1 \end{matrix}$$

Duidelijk is dat dit gedrag niet zonder meer past binnen het kader van de schakelalgebra. Immers, dan geldt in alle gevallen dat $a \cdot \bar{a} = 0$.

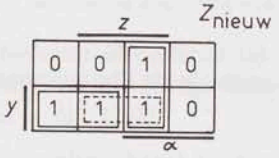


fig. 3.33. Karnaughdiagram.

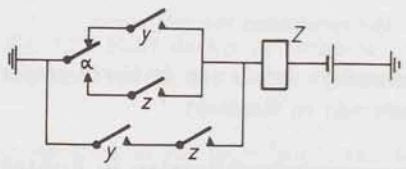


fig. 3.34. Kontaktschakeling zonder overgangsverschijnselen.

Het diagram in fig. 3.33 specificeert de gewenste logische werking van de schakeling uit fig. 3.32. Als de combinatie ($a = 0$; $y = 1$) optreedt, dan moet volgens de specificatie het relais Z opkomen. Dit wordt aangeduid met twee enen links beneden in het diagram, welke corresponderen met de weg $\bar{a}y$ in de schakeling. De twee overige enen corresponderen met de houdketen az in de schakeling. Uit dit Karnaughdiagram blijkt dat de term yz het omschakelen van het a-kontakt als het ware overbrugt omdat deze term onafhankelijk is van a. In fig. 3.34 is deze extra weg aan de schakeling toegevoegd, waarmee het overgangsverschijnsel is geëlimineerd!
Het is niet altijd mogelijk overgangsverschijnselen op deze wijze te overbruggen, met name niet wanneer in een schakeling meer dan één variabele tegelijk verandert. In het boven beschreven geval kon dit wel. De vereenvoudigingswet:

$$\bar{a}y + az + yz = \bar{a}y + az$$

is hier omgekeerd toegepast.
Een ander overgangsverschijnsel van kontakten, het nadenderen van kontakten die schakelen, heeft tot gevolg dat er vaak bij het drukken op drukknoppen e.d. niet één, maar verscheidene pulsen gegeven worden. Dit probleem kan worden omzeild met de *anti-dender schakeling* (anti-bounce). Deze schakeling komt in het volgende hoofdstuk aan de orde.

Conclusie
Bij het ontwerpen van kontaktschakelingen moet men rekening houden met overgangsverschijnselen. Als uit de analyse van een ontworpen kontaktschakeling volgt dat deze op enkele punten kritisch is, dan zijn speciale maatregelen

nodig. Deze maatregelen kunnen bestaan uit elektro-mechanische ingrepen (maak-voor-verbreekkontakten, traag opkomende en/of afvallende relais, enz.), of men kan trachten het probleem door een andere logische formulering te omzeilen. Een Karnaughdiagram, waarin de nieuwe toestand van het relais gespecificeerd is als functie van de oude toestand en de huidige ingangssignalen, bewijst hierbij goede diensten.
Met overgangsverschijnselen e.d. gekoppelde realisatieproblemen komen in volgende hoofdstukken nog uitvoerig ter sprake. Het zal dan blijken dat in schakelingen met geheugenwerking deze problemen nog gecompliceerder zijn dan hierboven is geschetst.

Opgaven

3.1. Gegeven is de schakelfunctie:

$$S = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}yz + wx\bar{y}\bar{z} + wxyz$$

a. Formuleer in woorden de taak van de schakeling, zoals deze in de gegeven schakelfunctie is geformuleerd.
b. Ontwerp een zo eenvoudig mogelijke schakeling met behulp van (wissel-)kontakten.

3.2. In fig. 3.19 is een vergrendelschakeling van een transportsysteem gegeven. Modificeer de schakeling zodanig dat, als de afsluiter i geopend is (i = 1, 2 of 3), de andere twee afsluiters niet meer kunnen worden bediend. De schakeling vergrendelt dan volgens het principe "De eerste wint".

3.3. Ontwerp een eenvoudige kontaktschakeling voor de formule:

$$S = w + \bar{w}x + \bar{w}\bar{x}y + \bar{w}\bar{x}\bar{y}z$$

S = 1 als de schakeling een doorverbinding vormt.

3.4. Idem voor:

$$S = w\bar{x} + \bar{w}z + x\bar{y} + y\bar{z}$$

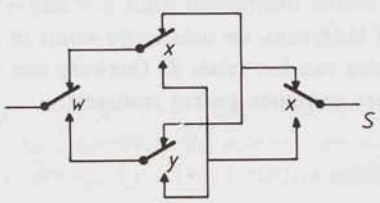
3.5. Idem voor:

$$S = uv + wx + xy + wz + yz.$$

3.6. Idem voor:

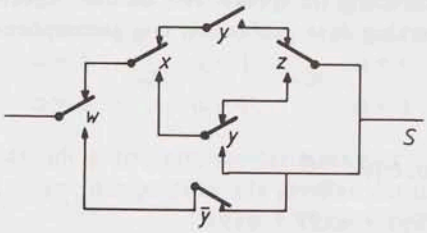
$$S_1 = \bar{y}z + y\bar{z}$$
$$S_2 = x\bar{y} + x\bar{z} + \bar{x}y + \bar{x}z + y\bar{z} + \bar{y}z$$
$$S_3 = w\bar{x} + w\bar{y} + w\bar{z} + x\bar{y} + x\bar{z} + y\bar{z}.$$

3.7. Bepaal de schakelfunctie van het getekende netwerk.

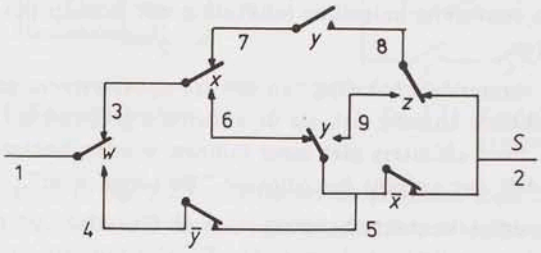


3.8. Gegeven zijn de vier relais W, X, Y en Z. Ontwerp een uit de kontakten van de relais W t/m Z opgebouwde schakeling, die aangeeft dat het aantal bekrachtigde relais even is. De schakeling moet een doorverbinding vormen als dit zo is.

3.9. Vereenvoudig de onderstaande schakeling met behulp van knooppuntseliminatie:



3.10. Bepaal de schakelfunctie van onderstaand netwerk met behulp van knooppuntseliminatie:



3.11. Op twee punten, aangeduid met A en B, staan twee spanningen. Deze spanningen zijn gelijk aan 0 Volt of V_A resp. V_B. Aan de klemmen A en B is, als V_A resp. V_B ongelijk aan 0 Volt zijn, voldoende vermogen beschikbaar om een of enkele relais te bekrachtigen. Gevraagd wordt een schakeling te ontwerpen die:

- V_A doorgeeft als $V_A \neq 0$ Volt is.
- V_B doorgeeft als $V_B \neq 0$ Volt is en $V_A = 0$ Volt is.
- De nulpotentiaal (aarde) doorgeeft als $V_A = V_B = 0$ Volt is.

a. Formuleer het probleem in een schakelformule.

b. Realiseer een schakeling m.b.v. relais en kontakten ervan. Maak de schakeling zodanig dat de opzet ervan uitbreidbaar is.

3.12. Gegeven zijn reedrelais met één maakkontakt. Tevens een handbediende drukknop die ook één maakkontakt bevat. De schakelformule ervan is dus S = a. Ontwerp een handbediende verbreekschakelaar met deze bouwstenen: $S = \bar{a}$.

3.13. De werking van een relais wordt beschreven door $S = ax\bar{y} + \bar{a}\bar{y}z$. Het a-kontakt schakelt relatief langzaam, de omslagtijd komt in de buurt van de opkom- en afvalvertraging van het relais Z. Ontwerp een betrouwbaar werkende schakeling die het gewenste gedrag realiseert.

Literatuur

1. J. Appels en B. Geels, *Handboek der Relaisschakeltechniek,* Philips Technische Bibliotheek, Eindhoven 1965.
2. R.M.M. Oberman, *Disciplines in Combinational and Sequential Circuit Design,* McGraw-Hill, New York 1970.

Voor verdere studie wordt verwezen naar 3.–8. Ref. 4. bevat een tabel met minimale oplossingen voor kontaktschakelingen tot en met vier variabelen. Deze tabel is geschikt als een bron voor oefenmateriaal.

3. M.A. Gavrilov, *Relaisschaltèchnik,* Berlin 1953.
4. M.A. Harrison, *Introduction to Switching and Automata Theory,* McGraw-Hill, New York 1965.
5. F.E. Hohn, *A Matrix Method for the Design of Relay Circuits,* IRE Trans. on CT., Vol. CT–2, 1955, pp. 154–161.
6. W.N. Roginsky, *Grundlagen der Structursynthese von Relaisschaltungen,* Oldenburg, München 1962.
7. W.N. Roginsky, *A Graphical Method for the Synthesis of Multiterminal Networks,* Proc. International Symposium on the Theory of Switching, 1957.
8. A.H. Scheinman, *A Numerical-Graphical Method for Synthesising Switching Circuits,* Trans. AIEE, 1958.

4. POORTSCHAKELINGEN

4.1. Inleiding

Hoofdstuk 3 behandelde enkele aspecten van het realiseren van combinatorische schakelingen met kontakten. Het informatietransport geschiedt via het wel of niet geleidend zijn van het netwerk.
Combinatorische schakelingen, welke zijn opgebouwd uit elektronische componenten, dragen de gewenste informatie meestal over via het wel/niet aanwezig zijn van een bepaald spanningsniveau aan de uitgang van de schakeling. De meeste van deze elektronische componenten kennen in de statische toestand twee uitgangsniveaus. Een ervan wordt als logische "1", de andere als logische "0" geïnterpreteerd. Het ontwerpen van combinatorische schakelingen met elektronische poorten zal in twee fasen worden behandeld:

- Het logisch ontwerp van de schakeling in de statische toestand.
- Aspecten van het ontwerp die samenhangen met overgangsverschijnselen in de dynamische toestand.

De elektronica van poortschakelingen zal zeer summier worden behandeld. Gestreefd wordt te komen tot een inzicht in die verschijnselen, die de correcte werking van een schakeling nadelig kunnen beïnvloeden.

Een combinatorisch probleem wordt meestal geformuleerd met een logische formule. In deze formule komen de logische operaties EN, OF en NIET voor. Het ligt daarom voor de hand logische bouwstenen te ontwerpen die deze drie elementaire logische bewerkingen kunnen uitvoeren. Een logische formule kan men dan direct vertalen in een schakeling. Er zijn echter andere factoren die een rol spelen bij de keuze der bouwstenen. Een ervan is de gebruikte realisatietechniek. Als voorbeeld mag genoemd worden de veel gebruikte AND-OR-INVERT combinatie in TTL-techniek. Een ander aspect is de wens dat iedere poortschakeling met een versterking van het signaal eindigt (zie par. 4.6). Omdat het "herstel" van het logische niveau een omkering van de polariteit met zich meebrengt, L → H en H → L, is in een poort een tweede versterkertrap (invertorversterker) nodig om het signaal de juiste polariteit te geven. In de praktijk wordt de tweede inversie vaak weggelaten, waardoor poortschakelingen ontstaan met de logische functie EN-NIET en OF-NIET. Deze poorten worden meestal aangeduid met de naam NAND en NOR, afkortingen van NOT-AND en NOT-OR.

Vele andere bouwstenen zijn ontstaan doordat voor veel voorkomende schakelhandelingen speciale circuits zijn ontworpen. Voorbeelden hiervan zijn optellers, code-omzetters, vermenigvuldigers, enz. Deze meer complete bouwstenen worden in volgende hoofdstukken behandeld. Een aparte vermelding verdient de EX-OR poort, die de exclusieve OF realiseert. Deze bouwsteen, met de logische functie $S = \bar{y}z + y\bar{z}$, kan worden gebruikt als *bestuurbare invertor.* Een van de twee signaalingangen wordt dan als stuuringang beschouwd. Is $z = 0$, dan is $S = y$, maar als $z = 1$ is dan is $S = \bar{y}$. Van deze mogelijkheid kan vaak een handig gebruik gemaakt worden.

Symbolen

Enkele van de vele symbolen voor poortschakelingen zijn in fig. 4.1 getekend. Links staan de symbolen die veel gebruikt worden in de catalogi, o.a. in die

van Texas Instruments. Rechts staan de symbolen volgens het normblad NEN 5152. In deze tekst worden voor poortschakelingen de symbolen uit de T.I. catalogus gebruikt.

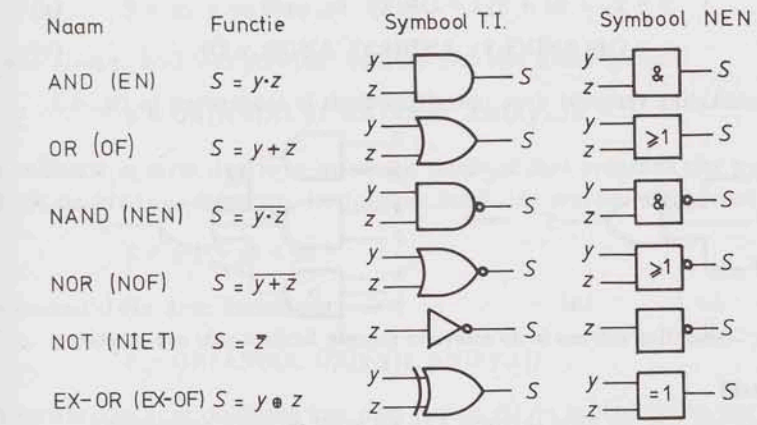


fig. 4.1. Symbolen voor poortschakelingen.

Opmerking

Als een signaal moet worden geïnverteerd, dan kan men dit in een tekening aangeven met een invertorsymbool. Het is echter gebruikelijk om dit aan de ingangen van poortschakelingen e.d. met een cirkeltje te doen. Hetzelfde geldt voor de uitgangen van poorten. Vergelijk daartoe de symbolen voor de AND en de NAND. Zo staat het symbool in fig. 4.2 voor de logische formule

$$S = \overline{\bar{x} \cdot y \cdot \bar{z}} = x + \bar{y} + z$$

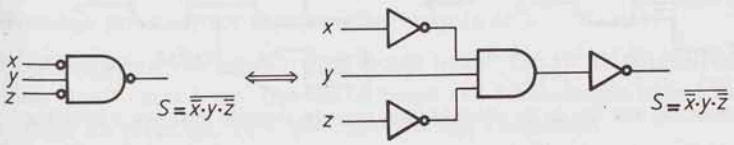


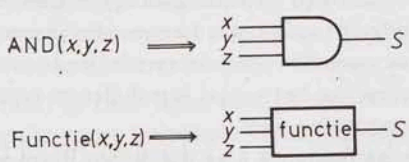
fig. 4.2. Aanduiding van inversies in poortschakelingen.

4.2. Het ontwerpen met de bouwstenen AND en OR

Een formule in de somvorm kan men rechtstreeks realiseren met de bouwstenen AND en OR. Fig. 4.3 geeft een voorbeeld van de hierbij te volgen procedure. Verondersteld is hierbij dat de inverse signaalwaarden beschikbaar zijn. Dezelfde formule, maar nu als een produkt geschreven, wordt gerealiseerd in fig. 4.4.

Notatie

Indien wordt gesproken over de realisatie van een logische formule, dan zal soms van de volgende notatiewijze gebruik worden gemaakt:



Voorbeeld
De formule S kan op de volgende wijze worden gerealiseerd:

$$S = \bar{x}\bar{y} + \bar{x}z + xy\bar{z} = OR(\bar{x}\bar{y},\ \bar{x}z,\ xy\bar{z}) \qquad (a)$$
$$= OR(AND(\bar{x},\bar{y}),\ AND(\bar{x},z),\ AND(x,y,\bar{z})) \qquad (b)$$

In de schakeling verloopt deze omzetting zoals is aangegeven in fig. 4.3.

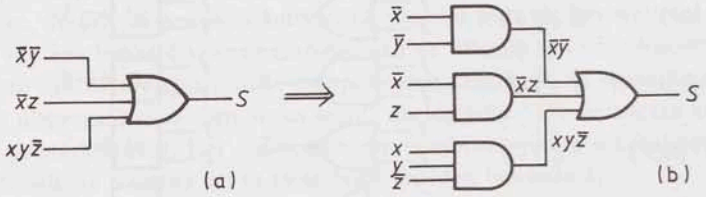


fig. 4.3. Omzetting van een in de somvorm gegeven formule naar een schakeling.

Voorbeeld
Een formule S in de produktvorm kan als volgt worden gerealiseerd:

$$S = (\bar{x} + y)(\bar{x} + \bar{z})(x + \bar{y} + z)$$
$$= AND((\bar{x} + y),\ (\bar{x} + \bar{z}),\ (x + \bar{y} + z)) \qquad (a)$$
$$= AND(OR(\bar{x},y),\ OR(\bar{x},\bar{z}),\ OR(x,\bar{y},z)) \qquad (b)$$

De schakeling staat in fig. 4.4.

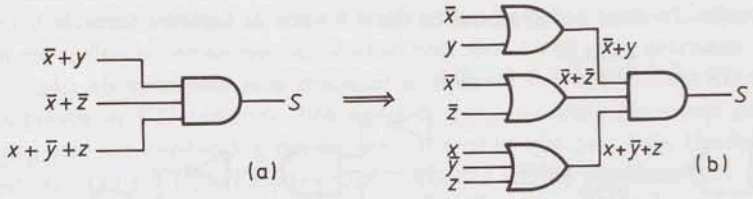


fig. 4.4. Omzetting van een in de produktvorm gegeven formule naar een schakeling.

De schema's in de figuren 4.3 en 4.4 geven twee mogelijkheden om een willekeurige schakelfunctie in de zg. *two level* vorm te realiseren. Wanneer de inverse signaalwaarden niet voorhanden zijn, dan moet een derde niveau met invertoren worden toegevoegd. Het aantal poortniveaus achter elkaar bepaalt o.a. de reactiesnelheid van de schakeling.

Het optimaliseren van poortschakelingen

Te ontwerpen poortschakelingen kunnen naar verschillende criteria worden geoptimaliseerd. Dit kan kostenaspecten betreffen (prijs der onderdelen, montagekosten, magazijnbeheer) of technische aspecten (minder niveaus is een snellere schakeling, minder onderdelen kan de storingskans verlagen, enz.). Meestal kan niet naar verschillende criteria tegelijk worden geoptimaliseerd en zal men een prioriteitsvolgorde moeten opstellen voor de verschillende criteria. Ook speelt een belangrijke rol de keuze van het aantal verschillende types bouwstenen die men de ontwerper ter beschikking stelt.
De oplossingsschema's in de figuren 4.3 en 4.4 leiden daarom niet altijd tot de optimale schakeling bij de gestelde criteria. Enkele voorbeelden hiervan volgen nu.

Voorbeeld
Een rechtstreekse realisatie van de schakelfunctie

$$S = xy + xz + yz$$

in twee niveaus kost vier poorten, waarvan één met drie ingangen:

$$S = OR(AND(x,y), AND(x,z), AND(y,z)).$$

Een realisatie in meer dan twee niveaus is mogelijk met eveneens vier poorten met elk slechts twee ingangen. De formule moet dan worden geschreven als:

$$S = x(y + z) + yz$$

Een realisatie van deze formule is:

$$S = OR(AND(x, OR(y,z)), AND(y,z))$$

Een variant van deze oplossing kan men vinden via de produktvorm van de formule:

$$S = (x + y)(x + z)(y + z)$$
$$= (x + yz)(y + z)$$

Een realisatie hiervan kost vier poorten met twee ingangen:

$$S = AND(OR(x, AND(y,z)), OR(y,z))$$

Ook deze schakeling is opgebouwd uit drie niveaus. Uit dit voorbeeld volgt reeds dat het optimaliseren naar snelheid (aantal niveaus) en lage kostprijs (eenvoudige poorten) tot tegenstrijdige eisen leidt.

Het optimaliseren van een ontwerp wordt moeilijker als de schakelfunctie een of meer don't cares bezit. Het aantal mogelijke schakelingen neemt dan snel toe, zeker als meer dan twee poortniveaus zijn toegestaan.

Voorbeeld

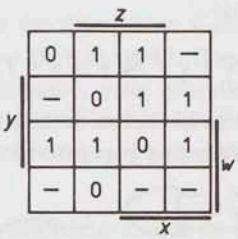


fig. 4.5. Karnaughdiagram.

De in fig. 4.5 gespecificeerde schakelfunctie bevat enkele don't care condities. De inverse signaalwaarden zijn beschikbaar. Gevraagd wordt een zo gunstig mogelijk ontwerp te maken voor de schakeling. Hierbij moet men rekening houden met de onder a, b of c geformuleerde eisen/wensen:

a. De schakeling moet in twee lagen worden opgebouwd, met een minimum aantal ingangen per poort (minimale vertraging, lage montagekosten).
b. Het totale aantal poortingangen moet zo klein mogelijk zijn (lage montagekosten, goedkope poorten).

c. De schakeling moet worden gerealiseerd met poorten die slechts twee ingangen bezitten. Dit probleem komt o.a. voor in sommige series pneumatische bouwstenen.

Ontwerp

Voor het onder a. gestelde criterium worden optimale schakelingen gevonden via de eenvoudigste somvorm S_1 en de produktvorm S_2 van de formule:

$$S_1 = \bar{w}\bar{y}z + w\bar{x}y + \bar{w}x + x\bar{z}$$

$$S_2 = (\bar{w} + \bar{x} + \bar{z})(w + x + \bar{y})(\bar{w} + y)(y + z)$$

In de formule S_1 mag de term $x\bar{z}$ vervangen worden door $y\bar{z}$ of door $w\bar{z}$. De realisatieschema's staan in fig. 4.6.

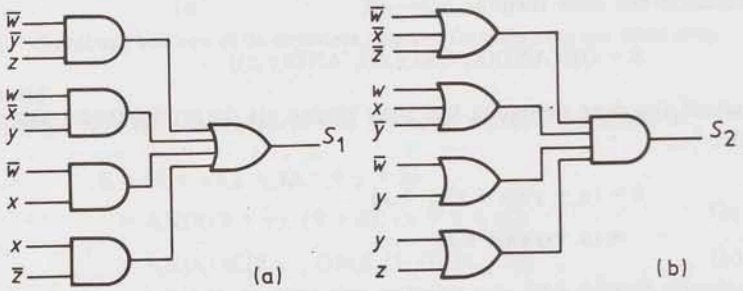


fig. 4.6. Poortschakelingen.

De via de somvorm en de produktvorm van de formule verkregen schakelingen zijn in het bovenstaande geval gelijkwaardig. Dit is niet altijd het geval. In het algemeen moeten beide vormen van de formule worden onderzocht.

Een gunstige schakeling voor het onder b. gestelde criterium wordt gevonden uit formule S_3:

$$S_3 = (\bar{w} + \bar{z} + \bar{x}y)(w + x + \bar{y}z)$$

Men kan deze formule vinden via:

$$S = (\bar{w} + \bar{x} + \bar{z})(\bar{w} + y + \bar{z})(w + x + \bar{y})(w + x + z)$$

welke vorm van de formule *niet* gelijk is aan de minimale produktvorm. De volgens criterium b. optimale schakeling is gegeven in fig. 4.7.

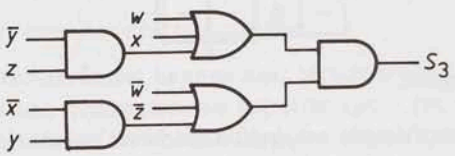


fig. 4.7. Poortschakeling.

Optimaliseren volgens criterium c. is moeilijker. Een oplossing is:

$$S_4 = \bar{w}(x + \bar{y}z) + w(\bar{z} + \bar{x}y)$$

De schakeling staat in fig. 4.8. Het aantal poorten is zeven, twee meer dan in

de vorige figuren. Bovendien is het aantal schakelniveaus groot. Hieruit blijkt dat een beperking van het aantal ingangen zeer zwaar telt.

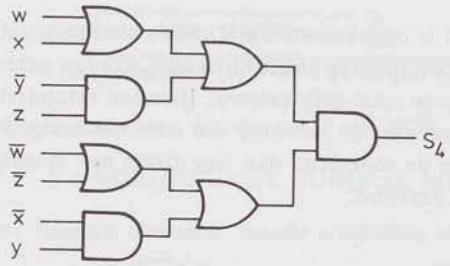


fig. 4.8. Poortschakeling.

Het ontwerpen van verscheidene poortschakelingen tegelijk

In de praktijk moeten binnen een ontwerp vaak vele combinatorische schakelingen tegelijk worden gerealiseerd. Deze schakelingen hebben soms dezelfde, soms een gedeeltelijk overlappende set ingangssignalen. Bij het ontwerp tracht men zoveel mogelijk deelschakelingen te combineren. Voor *two-level* schakelingen zijn hiervoor methoden ontwikkeld, o.a. uitbreidingen van de methode van Quine-McCluskey voor *multiple-output* schakelingen. Schakelingen met meer dan één uitgang zullen we in de volgende hoofdstukken herhaaldelijk tegenkomen.
Een belangrijk aspect van het samenstellen van combinatorische schakelingen moet echter nog worden genoemd. De vereenvoudigingswetten van de schakelalgebra zeggen o.a. dat de schakelformules

$$S = x(\bar{x} + \bar{y}) \quad \text{en} \quad S = x + \bar{x}\bar{y}$$

vereenvoudigd kunnen worden tot

$$S = x\bar{y} \quad \text{en} \quad S = x + \bar{y}$$

Het komt echter vaak voor dat de signalen $\bar{x}$ en $\bar{y}$ niet beschikbaar zijn, maar wel bijvoorbeeld de signalen $(\bar{x} + \bar{y})$ of $\bar{x}\bar{y}$. Door in deze gevallen niet uit te gaan van de eenvoudigste formule, maar gebruik te maken van reeds aanwezige signalen, kunnen besparingen op poorten worden bereikt. De risico's die dit met zich mee kan brengen worden nog besproken.

Conclusie
Bij het ontwerp van poortschakelingen leidt de eenvoudigste formule niet altijd tot de meest optimale schakeling. Afhankelijk van de gestelde criteria wordt aan een bepaalde realisatie de voorkeur gegeven. Het is niet altijd even gemakkelijk de optimale schakeling te vinden.

4.3. Het ontwerpen met de bouwstenen NAND en NOR

Iedere willekeurige schakelfunctie kan met uitsluitend gebruik van NAND's resp. NOR's worden gerealiseerd. Daarbij komt dat het werken met deze bouwstenen nauwelijks gecompliceerder is dan met AND's en OR's, terwijl de AND-OR combinatie met de invertor moet worden aangevuld indien de inverse

signaalwaarden niet beschikbaar zijn.

De realisatie van een willekeurige schakelfunctie in twee niveaus met NAND's komt als volgt tot stand:

Een schakeling die geheel is opgebouwd uit NAND's eindigt altijd met een inversie. Dit moet in de formulering tot uitdrukking worden gebracht. De formule wordt daartoe twee maal geïnverteerd. Hiermee verandert de formule niet. Zie de schakelalgebra voor de juistheid van deze bewering. Passen we dit toe op een formule in de somvorm, dan kan direct een mogelijke realisatie in NAND's worden herkend.

Voorbeeld
Zij gegeven de schakelfunctie

$$S = wx + wz + \bar{w}y\bar{z}$$

dan geldt

$$S = \overline{\overline{(wx + wz + \bar{w}y\bar{z})}} = \overline{(\overline{wx})\cdot(\overline{wz})\cdot(\overline{\bar{w}y\bar{z}})} \ .$$

De termen $\overline{(wx)}$, $\overline{(wz)}$ en $\overline{(\bar{w}y\bar{z})}$ kunnen elk met één NAND worden gerealiseerd. De functie S ontstaat dan door de uitgangen van deze drie NAND's te verbinden met de ingangen van een NAND op het tweede niveau. De uitgang van deze poort levert de gevraagde functie S.

$$S = \text{NAND}(\text{NAND}(w,x),\ \text{NAND}(w,z),\ \text{NAND}(\bar{w},y,\bar{z})) \ .$$

Het poortschema dat hiermee overeenkomt staat in fig. 4.9.

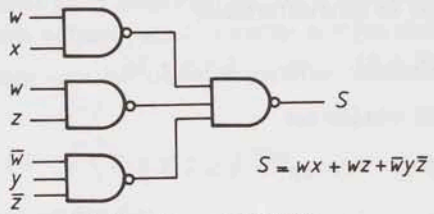


fig. 4.9. Realisatie van een schakelfunctie met NAND's.

Opmerking
Een vergelijking met het in fig. 4.3 gegeven realisatieschema voor een formule in de somvorm met de bouwstenen AND en OR leidt tot de conclusie dat de "structuur" van beide schakelingen dezelfde is. De poorten in fig. 4.3 kan men vervangen door NAND's, de logische werking blijkt identiek te zijn. Dit geldt echter niet meer als poortschakelingen in meer dan twee niveaus worden opgebouwd. Vergelijk daartoe fig. 4.7.

Het realisatieschema voor logische functies in twee niveaus met NOR's wordt gemakkelijk gevonden, wanneer wordt uitgegaan van de produktvorm van de formule. Weer moet de schakeling eindigen met een inversie. Deze inversie kan weer in de formule gebracht worden door een dubbele inversie.

Voorbeeld
Voor de schakelfunctie

$$S = (w + x)(w + z)(\overline{w} + y + \overline{z})$$

vinden we dan:

$$S = \overline{\overline{(w + x)(w + z)(\overline{w} + y + \overline{z})}}$$
$$= (\overline{\overline{(w + x)} + \overline{(w + z)} + \overline{(\overline{w} + y + \overline{z})}}) .$$

In deze formule herkennen we vier maal een NOR:

$$S = NOR(NOR(w,x), NOR(w,z), NOR(\overline{w},y,\overline{z})) .$$

De met deze formule overeenkomende schakeling staat in fig. 4.10.

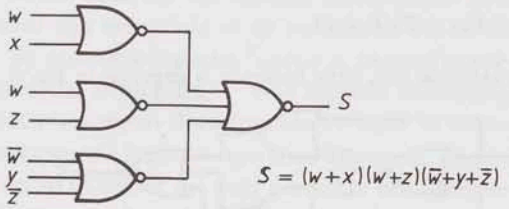


fig. 4.10. Realisatie van een schakelfunctie met NOR's.

Tot slot nog enkele voorbeelden van combinatorische schakelingen met NAND's en NOR's.

Voorbeeld
Realiseer de schakelfunctie

$$S = \bar{x}y + \overline{w}z + wx\bar{y}$$

met uitsluitend gebruik van NOR's. Inverse signaalwaarden zijn beschikbaar.
De eenvoudigste produktvorm van de formule is:

$$S = (\overline{w} + x + y)(\overline{w} + \bar{x} + \bar{y})(w + \bar{x} + z)(w + y + z) .$$

Een realisatie hiervan volgens het voorgaande oplossingsschema kost vijf poorten, waarvan vier met drie ingangen en één met vier ingangen:

$$S = NOR(NOR(\overline{w},x,y), NOR(\overline{w},\bar{x},\bar{y}), NOR(w,\bar{x},z), NOR(w,y,z)) .$$

Een eenvoudigere oplossing, maar wel in drie niveaus, wordt afgeleid van de somvorm van de formule:

$$S = \bar{x}y + \overline{w}z + wx\bar{y} = \overline{\overline{\bar{x}y + \overline{w}z + wx\bar{y}}}$$
$$= \overline{(x + \bar{y})(w + \bar{z})(\overline{w} + \bar{x} + y)} .$$

De functie

$$\bar{S} = (x + \bar{y})(w + \bar{z})(\overline{w} + \bar{x} + y)$$

kan worden gerealiseerd volgens het gebruikelijke oplossingsschema, waarna het uitgangssignaal nog één maal moet worden geïnverteerd. Ook dit kan met een NOR worden uitgevoerd. De totale schakeling wordt dan:

$$S = NOR(NOR(NOR(x,\bar{y}), NOR(w,\bar{z}), NOR(\overline{w},\bar{x},y))) .$$

Het aantal poorten is weer vijf, het aantal ingangen is echter vijf minder.

Voorbeeld
Aan een schakeling worden drie signalen x, y en z aangeboden. De inversen ervan zijn niet beschikbaar. De schakelfunctie die moet worden gerealiseerd luidt:

$$S = \bar{x}\bar{y}\bar{z} + xy\bar{z} .$$

Voor de realisatie staan NOR's ter beschikking.
De produktvorm van de schakelfunctie is:

$$S = \bar{z}(x + \bar{y})(\bar{x} + y) .$$

Een eenvoudige realisatie van deze formule is gegeven in fig. 4.11.

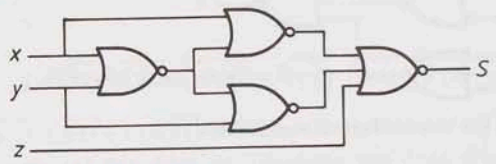


fig. 4.11. Poortschakeling met NOR's.

In fig. 4.11 zijn twee vereenvoudigingen aangebracht ten opzichte van het voor NOR's gebruikelijke oplossingsschema. De beide invertoren voor $\bar{x}$ en $\bar{y}$ zijn in één poort gecombineerd. Dit berust op de schakelwet $x + \bar{x}\bar{y} = x + \bar{y}$.
Verder is het signaal z direct aan de poort op het uitgangsniveau toegevoerd.

Staan ook andere bouwstenen ter beschikking, dan kan men een aardige oplossing vinden, als men de operatie EX-OR in de formule herkent:

$$S = \bar{z}(x + \bar{y})(\bar{x} + y) = \bar{z}(xy + \bar{x}\bar{y}) = \bar{z}\overline{(x \oplus y)} = \overline{z + (x \oplus y)} .$$

De schakeling staat in fig. 4.12. Dit is een elegante oplossing van het gestelde probleem.

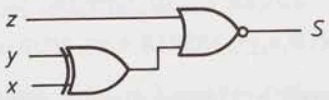


fig. 4.12. Poortschakeling.

Niet gebruikte ingangen bij poorten

De poorten in de getekende schema's bezitten precies het gewenste aantal ingangen. In de praktijk is dit vaak niet het geval, enkele ingangen kunnen overbodig zijn. In het algemeen verdient het geen aanbeveling overbodige ingangen open te laten. Dit o.a. met het oog op de storingsgevoeligheid (antenne-effect) van de schakeling.
Een van de mogelijke oplossingen is het met elkaar verbinden van een aantal ingangen. Dit kan bij poorten die symmetrisch zijn in de ingangsvariabelen, zoals de AND, OR, NAND, NOR, enz. Een nadeel is dat de uitgang van de sturende schakeling dan zwaarder wordt belast, hetgeen gevolgen heeft voor het maximale aantal poorten dat uit één uitgang kan worden gestuurd.
Ook is het mogelijk niet gebruikte ingangen, al of niet via een weerstand, met

de voedingsspanningen te verbinden. Bij een AND-poort moet op de niet gebruikte ingangen een spanning gezet worden die ongeveer hetzelfde niveau heeft als een logische "1". Een ingang van een OR moet worden verbonden met de spanning die overeenkomt met de logische "0".

Opmerking
Een bekend effect bij TTL-schakelingen is, dat de poort sneller reageert op een ingangsverandering naarmate er meer ingangen met elkaar zijn verbonden.

4.4. Combinatorische schakelingen en standaard bouwstenen

Enkele aspecten van het ontwerp van logische schakelingen voor problemen van combinatorische aard zijn behandeld in de paragrafen 4.2 en 4.3. Hierbij is er van uitgegaan dat de gewenste logische functie is gespecificeerd in de vorm van een logische formule, die is uitgedrukt in de logische bewerkingen EN, OF en NIET. In de realisatiefase wordt deze formule *vertaald* in bouwstenen, die de in de formule voorkomende logische operaties uitvoeren. Daarbij is gebleken dat ook NAND's en NOR's voor dit doel geschikte bouwstenen zijn. Voor het op deze wijze realiseren van logische schakelingen staat een zeer gevarieerde collectie bouwstenen ter beschikking, zeker als van geïntegreerde circuits gebruik wordt gemaakt:

AND, met 2, 3 of 4 ingangen
NAND, met 2, 3, 4, 8 of 13 ingangen
OR, met 2 ingangen
NOR, met 2, 3, 4 of 5 ingangen
EX-OR, met 2 ingangen.

In deze collectie moet ook genoemd worden de in TTL-schakelingen veel voorkomende AND-OR-INVERT combinatie. Zie fig. 4.13. Deze poorten zijn er met o.a. de volgende aantallen ingangen:

(2, 2), (3, 3), (2, 2, 2, 2), (2, 3, 3, 2), enz.

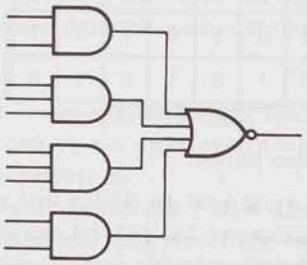


fig. 4.13. (2, 3, 3, 2) AND-OR-INVERT poortcombinatie.

Bij het op de hiervoor geschetste wijze realiseren van combinatorische schakelingen werd niet of nauwelijks gebruik gemaakt van de systematiek die in vele problemen verscholen zit. Deze is vaak moeilijk te herkennen nadat het probleem is omgezet in een logische formule. Het volgende voorbeeld geeft een toelichting.

Voorbeeld. *Pariteitscontrole*

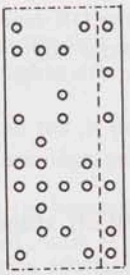


fig. 4.14. Ponsband met oneven pariteitscontrole.

Bij het vastleggen van karakters in ponsband komt de wens naar voren een eenvoudige controle uit te voeren op correcte ponsingen. Een van de mogelijkheden is het toevoegen van een extra bit, en wel zo dat het aantal ponsingen per rij even of oneven is. Een dergelijk bit wordt wel *pariteitsbit* genoemd, en dit systeem van foutendetectie wordt aangeduid met *parity check code.*
Bij het uitlezen kunnen enkelvoudige fouten worden getest op de volgende wijze:

Het aantal gaten per rij is even resp. oneven $\leftrightarrow$ fout resp. goed

Gevraagd wordt een controleschakeling te ontwerpen die deze test uitvoert. De controleschakeling heeft vijf ingangen, de signalen v, w, x, y en z. Elk van deze signalen is 1 als op het bemonstertijdstip in de band op de betreffende plaats een gat zit. Gaan we uit van even pariteit, dan specificeert het Karnaughdiagram in fig. 4.15 de logische werking van de controleschakeling op de tot nu toe gebruikelijke wijze. De bijbehorende formule in de somvorm bevat 16 produkten, elk in vijf variabelen.

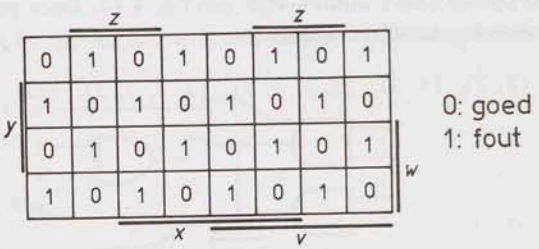


fig. 4.15. Karnaughdiagram, even pariteit.

Een realisatie in AND-OR vorm kost 16 AND's met vijf ingangen en een OR met 16 ingangen. Nog afgezien van het feit dat een OR met 16 ingangen niet in de handel is, is de geschetste oplossing zeer onpraktisch.

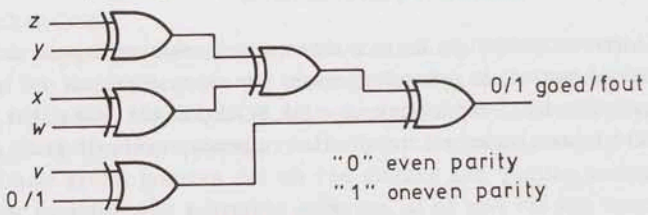


fig. 4.16. Schakeling voor pariteitscontrole.

Een nadere analyse van het probleem leert, dat uitsluitend getest moet worden of het aantal gaten per rij even resp. oneven is. Voor de bits y en z betekent dit dat de combinaties 00 en 11 gescheiden moeten worden van de combinaties 01 en 10. Dit betekent, dat getest moet worden of y en z gelijk of ongelijk zijn. Deze detectie is als logische bewerking identiek met de EXCLUSIEVE-OF voor twee bits. Zie de schakelalgebra.
De EX-OR poort is dus geschikt om het gelijk of ongelijk zijn van twee signalen te detecteren. Fig. 4.16 geeft de definitieve schakeling, die bestaat uit vijf EX-OR poorten. In deze schakeling kan naar keuze op even of oneven pariteit getest worden.

De conclusie die uit dit voorbeeld volgt is, dat men bij het ontwerpen van digitale schakelingen zo goed mogelijk gebruik moet maken van de in het probleem aanwezige systematiek. Voor een aantal meer complexe problemen zijn standaard schakelingen ontwikkeld en als geïntegreerd circuit verkrijgbaar. Als voorbeeld kan genoemd worden de 8 of 9 bit even/oneven parity generator/checker. Vaak "past" een probleem geheel of gedeeltelijk op zo'n standaard bouwsteen. In dat geval is de schakeling vaak beter dan een die via de reeds eerder besproken algemene ontwerpmethode is gerealiseerd.

Standaard bouwstenen

Voorbeelden van meer complexe geïntegreerde circuits voor speciale toepassingen zijn:

1. *Parity generators/checkers.*
2. *Code-omzetters.* Deze dienen om informatie die op een bepaalde wijze gecodeerd is om te zetten in een andere code, welke beter aansluit bij de verwerking van de informatie.
3. *Vergelijkschakelingen (Comparatoren).* Met deze schakelingen kan van twee getallen worden bepaald welke de grootste is, of dat zij gelijk zijn.
4. *Datakiezers (Selectors/Multiplexers).* Hiermee kan men kiezen welke lijn van een aantal ingaande lijnen met de uitgangslijn wordt verbonden. Deze keuze wordt via een aantal *select-signalen* ingesteld.
5. *Decoders (Demultiplexers).* Hiermee wordt één ingaande lijn met één van vele uitgaande lijnen doorverbonden, wederom onder besturing van enkele select-signalen.

Van deze opsomming, die beslist niet volledig is, verdient vooral de datakiezer de aandacht omdat deze ook bij het ontwerp van willekeurige combinatorische schakelingen universeel toepasbaar is.
In fig. 4.17 is het principeschema van een 1-uit-4 datakiezer getekend. Door middel van twee binaire signalen y en z wordt één van de vier datalijnen D_0 t/m D_3 geselecteerd en met de uitgang doorverbonden.

Datakiezers zijn verkrijgbaar in 2:1, 4:1, 8:1 en 16:1 uitvoering. Worden de programmeerdraden y en z als informatie-ingangen beschouwd, dan bepalen in fig. 4.17 de ingangen D_0 t/m D_3 hoe de uitgang S van de variabelen y en z afhangt. Tabel 4.1 geeft een voorbeeld van de realisatie van de schakelfunctie $S = y + \bar{z}$ met een 4:1 datakiezer. De mintermvorm van de schakelfunctie biedt hierbij een goed uitgangspunt:

y	z	$S = y + \bar{z}$
0	0	$D_0 = 1$
0	1	$D_1 = 0$
1	0	$D_2 = 1$
1	1	$D_3 = 1$

tabel 4.1. Realisatie van S met datakiezer.

$$S = m_0 f_0 + m_1 f_1 + m_2 f_2 + m_3 f_3$$
$$= \bar{y}\bar{z}\cdot f_0 + \bar{y}z\cdot f_1 + y\bar{z}\cdot f_2 + yz\cdot f_3$$

Is de functiewaarde $f_i = 1$, dan moet op de corresponderende D_i-ingang een 1 worden aangeboden, anders een 0.

Hoewel datakiezers als bouwsteen duurder zijn dan losse poorten, verdienen zij toch een nadere beschouwing. Het gebruik ervan als *programmeerbare poortschakeling* heeft als voordeel lage ontwerpkosten en lage montagekosten. Vooral bij weinig gestructureerde formules, de z.g. *random logic*, kan het gebruik van programmeerbare poortschakelingen voordeel bieden. De programmering geschiedt bij datakiezers via de bedrading. Zie ook de volgende paragraaf.

In het algemeen is het mogelijk met datakiezers met 2^k data-ingangen en k selectsignalen een willekeurige schakelfunctie in k + 1 variabelen te realiseren. Als voorbeeld de realisatie van de schakelfunctie

$$S = xy + xz + yz + \bar{x}\bar{y}\bar{z} .$$

We gebruiken hiervoor een 4:1 datakiezer. Iedere schakelfunctie in drie variabelen is te schrijven als:

$$S = \bar{y}\bar{z}\cdot f(x,0,0) + \bar{y}z\cdot f(x,0,1) + y\bar{z}\cdot f(x,1,0) + yz\cdot f(x,1,1) .$$

Voor de beschouwde functie leidt dit tot:

$$S = \bar{y}\bar{z}\cdot\bar{x} + \bar{y}z\cdot x + y\bar{z}\cdot x + yz\cdot 1 .$$

De gewenste schakelfunctie wordt dan gerealiseerd als $D_0 = \bar{x}$, $D_1 = x$, $D_2 = x$ en $D_3 = 1$ gemaakt worden.

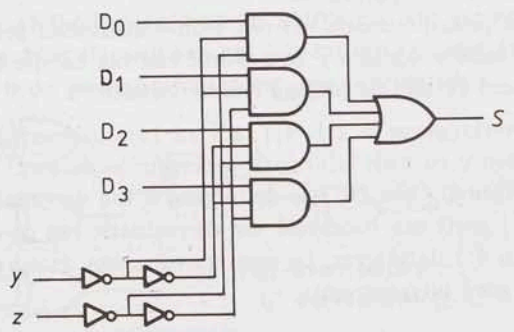


fig. 4.17. 4:1 Datakiezer.

Voorbeeld

Een uitbreiding van het aantal variabelen in een functie kan worden opgevangen door datakiezers in meer dan één niveau te schakelen. Als voorbeeld de realisatie van de schakelfunctie

$$S = \bar{x}y + \bar{v}\bar{w}y + v\bar{y}z + \bar{w}x\bar{y}\bar{z} + wx\bar{y}z$$

met behulp van 4:1 datakiezers. (Merk op dat dit zou kunnen met één 16:1 datakiezer).

Fig. 4.18 geeft een mogelijke oplossing. Een besparing van één datakiezer is mogelijk als op het uitgangsniveau de variabelen x en y worden aangeboden, zoals in fig. 4.18 wordt getoond.

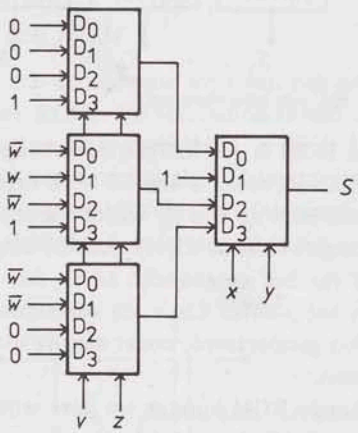


fig. 4.18. De realisatie van een functie met datakiezers.

Opmerking

Combinatorische schakelingen, die volgens het principe van fig. 4.18 zijn samengesteld, hebben ten opzichte van schakelingen die uit losse componenten zijn samengesteld enige nadelen. Datakiezers zijn onder meer trager dan gewone poorten, zodat een schakeling met datakiezers over het algemeen trager zal zijn dan een schakeling uit losse componenten. Bovendien worden de ingangen op verschillend niveau aangesloten, hetgeen aanleiding kan geven tot hinderlijke overgangsverschijnselen. Zie ook par. 4.8.

4.5. Combinatorische schakelingen en leesgeheugens

Tot nu toe zijn combinatorische schakelingen steeds gerealiseerd met bouwstenen die een vaste logische bewerking uitvoeren. De keuze van de componenten varieerde van de logische AND en OR enerzijds en complexe bouwstenen zoals de parity checker anderzijds. *De gewenste logische functie wordt in wezen gerealiseerd via de bedrading tussen de componenten. Met een ander bedradingspatroon realiseert dezelfde set bouwstenen als regel een andere logische functie.* Er is echter een geheel andere benadering mogelijk van het ontwerpen van combinatorische schakelingen. Deze is mogelijk gemaakt door goedkope *halfgeleidergeheugens.* Het ontwerpen bestaat uit het inlezen van de waarheidstabel van de gewenste logische functie in een geheugen. Bij aanbieding van een bepaalde combinatie van de ingangsvariabelen van de schakeling als adres aan het geheu-

gen wordt de corresponderende regel van de waarheidstabel gelezen. In fig. 4.19 is de opbouw van een dergelijke geheugenschakeling *(Read Only Memory, ROM)* weergegeven.

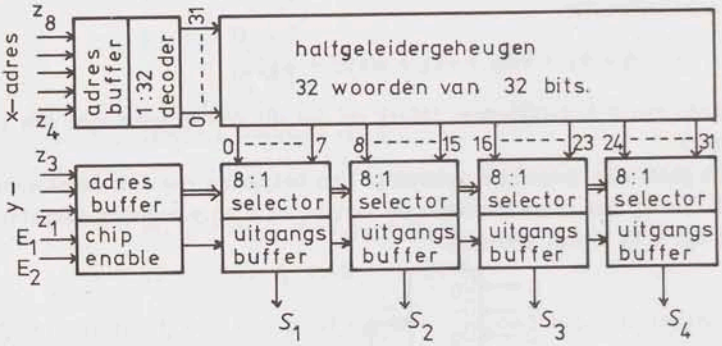


fig. 4.19. Organisatieschema van een leesgeheugen.

Het centrale deel van een ROM is een halfgeleidergeheugen, georganiseerd in k woorden van n bits. In het voorbeeld is een 32 × 32 organisatie verondersteld. Door middel van vijf adreslijnen z_4 t/m z_8 wordt via een 1:32 decoder één regel van het geheugen aangewezen (x-adres). Aan de uitgang van het geheugen verschijnt dan de inhoud van het aangewezen adres, een woord van 32 bits. Van dit woord wordt via het y-adres d.m.v. de adreslijnen z_1 t/m z_3 uit iedere groep van acht bits één bit geselecteerd, zodat aan de uitgang van de schakeling vier bits tegelijk verschijnen.
Met de in fig. 4.19 getekende ROM kunnen op deze wijze vier schakelfuncties in ten hoogste acht ingangsvariabelen worden gerealiseerd. De schakelfuncties zijn S_1 t/m S_4, de ingangsverzameling bestaat uit de variabelen z_1 t/m z_8.

Voorbeeld

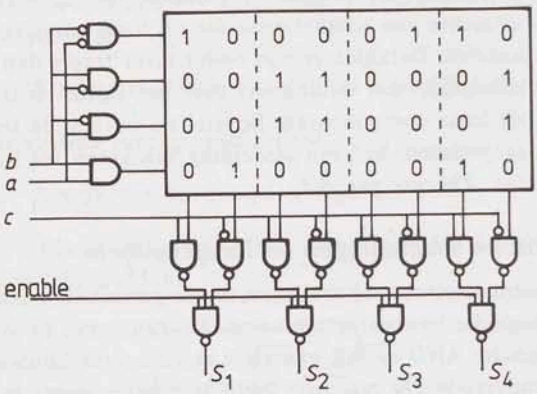


fig. 4.20. Vulling van een ROM.

In fig. 4.20 is voor een (hypothetische) 32-bit ROM (4 woorden van 8 bits) aangegeven hoe deze moet worden geprogrammeerd voor vier schakelfuncties in de variabelen a, b en c:

$$S_1 = ab\bar{c} + \bar{a}\bar{b}c$$

$$S_2 = a \oplus b$$

$$S_3 = \overline{\bar{a}\bar{b}\bar{c}}$$

$$S_4 = a \oplus b \oplus c$$

Het is mogelijk de adressering van een ROM zo te ontwerpen dat de waarheidstabellen van de vier functies direct kunnen worden ingelezen. De aanwezige *chip enable* ingang dient om het geheugen te kunnen blokkeren. Hierdoor wordt het mogelijk met verschillende ROM's een ROM met een grotere geheugencapaciteit (meer adresvariabelen) samen te stellen.

De geheugencel van een ROM

Fig. 4.21 geeft schematisch de opbouw weer van een geheugencel van een PROM (Programmeerbare ROM). De verbinding tussen de x-draad en de y-draad kan met een stroomstoot worden opgeblazen. Is de verbinding verbroken, dan zal de y-draad de inhoud van deze cel als "0" interpreteren. Is de verbinding nog aanwezig, dan zal de y-draad bij bekrachtiging van de x-draad de spanning V_E voeren. Deze spanning wordt dan aan de uitgang in het niveau van de logische 1 omgezet.

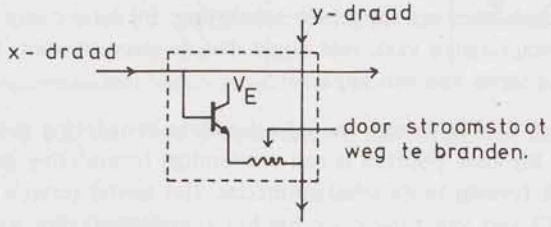


fig. 4.21. Geheugencel van een PROM.

Read Only Memories zijn in velerlei uitvoeringen verkrijgbaar. In omvang variëren zij van een 32 × 8 organisatie tot een 2048 × 8 organisatie. Met deze laatste kunnen acht schakelfuncties in maximaal elf variabelen worden gerealiseerd. Bepaalde uitvoeringen kunnen alleen bij de fabricage worden geprogrammeerd, waarna de inhoud voor altijd vastligt. Andere uitvoeringen zijn *"Field Programmable"*, zij kunnen ter plaatse worden geprogrammeerd (ingebrand). Ook bestaan er uitvoeringen, waarvan de programmering niet destructief geschiedt, zodat het geheugen opnieuw kan worden geprogrammeerd. Afhankelijk van hun uitvoering worden zij aangeduid als

ROM	Read Only Memory
PROM	Programmable Read Only Memory
RePROM	Reprogrammable Read Only Memory.

Het laatste type wordt ook wel aangeduid als EPROM (erasable PROM).

Voorbeeld

Bij het meten van bepaalde grootheden heeft men vaak last van de niet-lineariteit van omzetters. Een voorbeeld is het meten van een temperatuur met een thermokoppel. De door het thermokoppel afgegeven gelijkspanning wordt met

behulp van een analoog-digitaal omzetter (A-D omzetter) omgezet in een binair getal. Zie het schema in fig. 4.22. Voor een 8-bit binair getal kan men met een 256 × 8 PROM het meetresultaat gemakkelijk corrigeren. Voor elke combinatie van acht bits, zijnde een meetresultaat, wordt de juiste waarde in het geheugen opgeslagen.

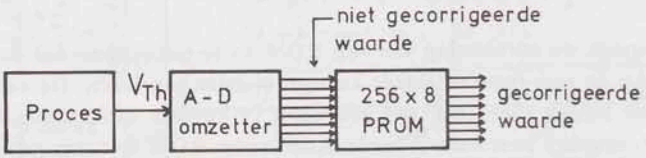


fig. 4.22. Toepassing van een PROM bij correctie van metingen.

Voor het realiseren van *random logic* bieden ROM's en PROM's vele mogelijkheden. In het bovenstaande voorbeeld is het zeer onpraktisch om de correcties op de gemeten waarden in logische functies uit te drukken en deze vervolgens in losse poorten te realiseren. De schakeling zou te omvangrijk worden. Met PROM's echter is het zelfs mogelijk rekening te houden met de karakteristiek van elk thermokoppel afzonderlijk.
Als nadeel van PROM's en ROM's moet worden genoemd dat zij relatief traag zijn. Het grote aantal schakelniveaus is hiervan de oorzaak. Ook de hogere prijs is (nog) een bezwaar, hoewel deze in de praktijk vaak wordt gecompenseerd door lagere montagekosten van de gehele schakeling. Bij *losse-componenten-logica* zijn de montagekosten vaak veel hoger dan de circuitkosten. Dit is zeker het geval bij kleine series van een apparaat.

Het ontwerpen met ROM's vereist een geheel andere benadering van de probleemspecificatie. Bij losse poorten is een eenvoudige formulering gewenst, met zo weinig mogelijk termen in de schakelfuncties. Het aantal termen is bij het realiseren in ROM's niet van belang, slechts het aantal variabelen waarop de schakelfunctie is gedefinieerd telt. Vereenvoudigingsmethoden zoals het Karnaughdiagram en de methode van Quine-McCluskey worden minder belangrijk.

Programmable Logic Arrays (PLA's)

Een van de kostenbepalende factoren bij *losse-componenten-logica* is het aantal termen in de schakelfunctie, alsmede het aantal variabelen per term. Bij ROM's is dit het aantal ingangsvariabelen. Grotere aantallen dan 10 à 12 ingangsvariabelen leiden tot onpraktische schakelingen. De beperkende factor is het adresgedeelte van het geheugen, dat zowel in omvang als in vertragingstijd gaat toenemen naarmate het aantal variabelen groter wordt. Zie fig. 4.19.
Een tussenvorm is de PLA *(Programmable Logic Array)*. Fig. 4.23 geeft het principeschema ervan. Een PLA bestaat uit twee "matrices". In de bovenste wordt een aantal AND-functies gevormd. Aan elke AND kan elke ingangsvariabele of zijn inverse worden aangeboden. Op de met ⊹ gemerkte kruispunten kan een verbinding worden gemaakt.

In de onderste matrix van fig. 4.23 worden de gevormde produkten gesommeerd. Elk produkt kan in verschillende OR-poorten worden meegenomen. Op deze wijze kunnen verschillende functies tegelijk worden gerealiseerd. De verzameling ingangsvariabelen waarop deze zijn gedefinieerd is groter dan bij een ROM gebrui-

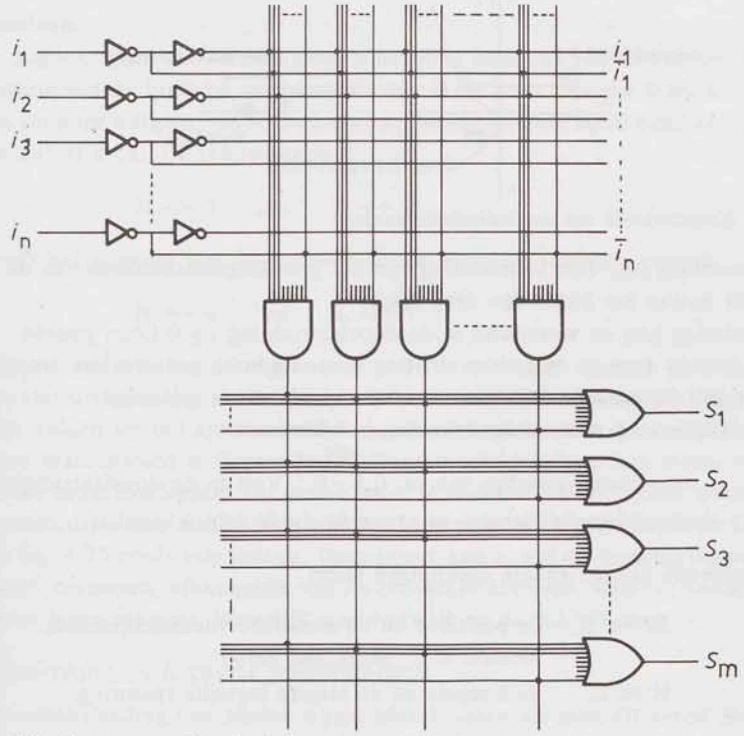


fig. 4.23. Programmeerbaar logisch array.

kelijk is. Er is o.a. een *field programmable* uitvoering van een PLA verkrijgbaar met

16 ingangen
8 uitgangsfuncties
48 produkttermen

Het optimaliseren van een ontwerp met PLA's is vrij lastig. In feite moeten verschillende schakelfuncties tegelijk worden geminimaliseerd. Dit minimaliseren betreft uitsluitend het aantal produkttermen van de schakelfuncties. Het totale aantal produkten is beslissend, niet de omvang der produkten. Een aangepaste versie van een multiple output Quine-McCluskey algoritme kan goede diensten bewijzen.

4.6. De realisatie van poortschakelingen met diodes

De belangrijkste componenten van logische schakelingen zijn de halfgeleiderdiode en de transistor. Tot een tiental jaren geleden kwamen zij in logische schakelingen nog als losse componenten voor. Tegenwoordig zijn zij onderdeel van een geïntegreerde schakeling. In fig. 4.24 is de *karakteristiek* van een *halfgeleiderdiode* getekend. Ook is het symbool ervan voor elektrotechnische tekeningen aangegeven.
De diode is een element dat in één richting de stroom goed geleidt, dus een lage weerstand bezit. We noemen deze richting de *doorlaatrichting*. In de omgekeerde richting is de weerstand veel groter. De pijl bij het symbool geeft de

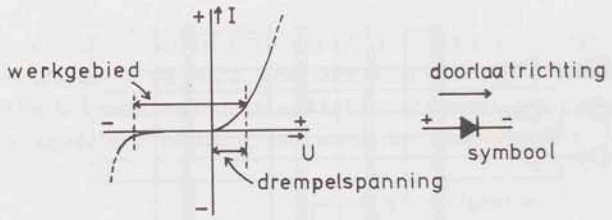


fig. 4.24. Karakteristiek van een halfgeleiderdiode.

doorlaatrichting aan. Een behandeling van het geleidingsmechanisme van de diode valt buiten het kader van deze tekst.
Als benadering kan de weerstand in de doorlaatrichting op 0 Ohm gesteld worden, terwijl deze in de andere richting oneindig hoog gedacht kan worden. Een dergelijk (hypothetisch) element wordt *ideale diode* genoemd.
In de praktijk moet men echter rekening houden met:

- een spanningsverlies van ca. 0,2 - 0,5 Volt in de doorlaatrichting
- een zekere lekstroom, wanneer de diodé geblokkeerd is.

In het volgende komen enkele spanningen voor:

E_+ en E_- , de positieve en de negatieve voedingsspanning. Meestal is E_- gelijk aan aarde.

H en L, de hoogste en de laagste logische spanning.

Het verband tussen deze spanningen is: $E_+ > H > L > E_-$

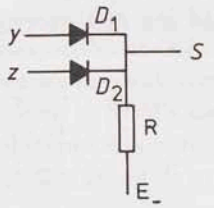


fig. 4.25. Diodepoort.

y	z	S
L	L	L
L	H	H
H	L	H
H	H	H

(a) spanningen in Volt.

L ↔ 0
H ↔ 1

y	z	S
0	0	0
0	1	1
1	0	1
1	1	1

(b) waarheidswaarden, positieve logica.

tabel 4.2. Werking van een diodepoort.

Fig. 4.25 geeft het schema van een *diodepoort* met twee ingangen.
Tabel 4.2.a beschrijft de elektrische werking van deze poort, uitgedrukt in logische spanningen H en L. Het blijkt dat de uitgang S de meest positieve van de op de y- of z-ingang aangeboden spanningen volgt. Wanneer de ingangsspanningen ongelijk zijn, dan geleidt de diode van de meest positieve ingang. De andere ingangsdiode staat geblokkeerd. Tabel 4.2.a veronderstelt ideale diodes, in de praktijk is de uitgangsspanning wat lager.

Positieve en negatieve logica

Wordt de logische spanning H als logische 1 geïnterpreteerd, en L als logische 0, dan beschrijft tabel 4.2.b de logische werking van de getekende diodepoort. Deze waarheidstabel is identiek aan die van de logische OF. De getekende poort realiseert de logische bewerking OF.
De omgekeerde interpretatie, H als 0 en L als 1, leidt tot de waarheidstabel van de logische bewerking EN.

Conclusie

De logische functie van een poortschakeling hangt af van de afgesproken interpretatie van de logische spanningsniveaus H en L als logische 0 en 1. In catalogi ziet men de werking van logische schakelingen daarom vaak beschreven in tabellen met H en L. De interpretatie

$$H \leftrightarrow 1 \quad \text{en} \quad L \leftrightarrow 0$$

wordt het systeem van de *positieve logica* genoemd. Het alternatief,

$$H \leftrightarrow 0 \quad \text{en} \quad L \leftrightarrow 1$$

heet het systeem van de *negatieve logica.* Deze namen spreken voor zich.
Om verwarring te voorkomen verdient een vaste afspraak de voorkeur. In deze tekst volgen we het systeem van de positieve logica, welk systeem bijna algemeen geaccepteerd is. Sommige fabrikanten echter volgen nog steeds het systeem van de negatieve logica. Tot welke moeilijkheden dit leidt bij het koppelen van apparatuur, welke in een verschillend systeem is beschreven geeft de diodepoort van fig. 4.25 reeds een indruk. Deze poort kan zowel de logische bewerking EN als OF realiseren, afhankelijk van de gemaakte afspraak. Ook bij andere bouwstenen komt dit voor. Vergelijk o.a. de NOR en de NAND, enz.

Versterking in logische schakelingen

Het schakelgedrag van diodes is niet ideaal, zeker als men dit vergelijkt met kontakten. In de geleidende richting resteert er een niet te verwaarlozen restspanning. Na enkele schakelniveaus moet het logische niveau weer hersteld worden. Omdat we slechts twee logische niveaus onderscheiden, H en L, is een zekere marge rond beide niveaus toegestaan.
Bij diodepoorten treedt nog een extra probleem op. Fig. 4.26 beschrijft twee schakelingen met diodepoorten. Onder het elektrisch schema staan deze schakelingen vertaald in de gebruikelijke poortsymbolen. Het gebruik van positieve logica is hierbij verondersteld, zoals is afgesproken.

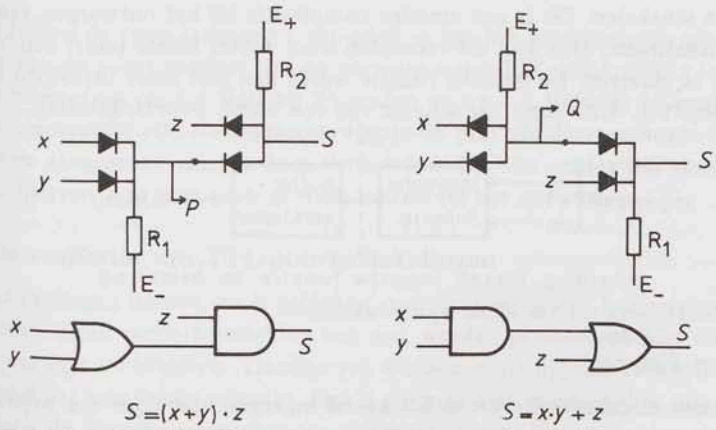


fig. 4.26. Schakelingen met diodepoorten.

In de schakeling zijn de volgende spanningen aanwezig:

$E_+ = +15$ Volt, $E_- = -10$ Volt } Voedingsspanningen

$H = 5$ Volt, $L = 0$ Volt } Logische niveaus

De diodes worden ideaal verondersteld. De schakeling in fig. 4.26.a realiseert de logische functie:

$$S = (x + y) \cdot z$$

mits voldaan is aan de eis dat $R_1 \leqslant \frac{2}{3} R_2$ is. Immers, zijn de ingangsspanningen

$$U_x = U_y = 0 \text{ Volt} \quad \text{en} \quad U_z = 5 \text{ Volt}, \quad (x = y = 0 \text{ en } z = 1)$$

dan is de gewenste uitgangsspanning

$$S = (x + y) \cdot z = (0 + 0) \cdot 1 = 0, \quad \text{d.w.z. } U_S = 0 \text{ Volt.}$$

Deze uitgangsspanning kan alleen worden bereikt, indien $R_1 \leqslant \frac{2}{3} R_2$ is. De lezer verifiëre deze bewering. Deze verhouding zal bovendien nog veranderen als op de punten P of S nog andere poorten worden aangesloten.

Op geheel overeenkomstige wijze volgt uit fig. 4.26.b dat deze schakeling slechts correct werkt als $R_2 \leqslant \frac{2}{3} R_1$ is. Dit valt in te zien bij de ingangscombinatie

$$S = x \cdot y + z = 1 \cdot 1 + 0 = 1 .$$

Voor de twee getekende configuraties gelden tegengestelde eisen. Bovendien moet in beide gevallen de gevonden verhouding worden aangepast als er nog andere poorten worden aangesloten.
We stuiten hier op het verschijnsel dat de elektrische werking van een poort, en dus ook de logische werking, kan veranderen onder invloed van de *belasting*. Dit betekent dat men deze poorten niet in een willekeurige configuratie achter elkaar kan schakelen. Dit is een ernstige complicatie bij het ontwerpen van logische schakelingen. Men kan dit vermijden door achter iedere poort een buffer-versterker te plaatsen. De logische functie wordt dan niet meer beïnvloed door de belasting. Fig. 4.27 geeft het schema van een *ideale poortschakeling*.

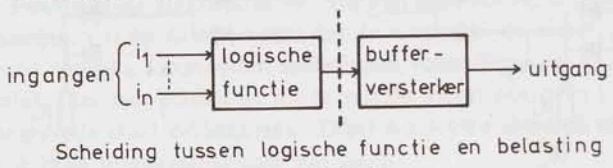


fig. 4.27. Blokschema van een ideale poortschakeling.

Fan-in en Fan-out

Ook met een uitgangsversterker is het aantal ingangen, dat door een bepaalde poort kan worden gestuurd, beperkt. Om aan te geven waar de grenzen liggen wordt binnen een bepaalde serie bouwstenen een *standaardbelasting* gedefinieerd. Heeft een poort een *fan-out* van 10, dan wil dat zeggen dat deze poort met maximaal 10 van deze standaardbelastingen mag worden belast. Wanneer een

ingang een grotere belasting vormt dan één eenheid, dan wordt dit aangegeven met een *fan-in* van bijvoorbeeld 2.
Bij geïntegreerde circuits treft men aan de ingangen van een circuit vaak een buffer-versterker aan. Dit wordt gedaan als een ingangssignaal op verschillende plaatsen in het circuit doorwerkt (kloksignalen, e.d.). De gebufferde ingang heeft dan naar buiten toe een fan-in van 1, de belasting ervan voor de sturende schakeling is identiek aan de standaardbelasting.
Bij vele types logische schakelingen ziet men de aanduiding *"TTL-compatible"* in de specificaties staan. Dit betekent dat de ingangs- en uitgangsschakelingen van deze componenten zo zijn gedimensioneerd dat zij met TTL-schakelingen vergelijkbaar zijn m.b.t. de logische niveaus en de impedanties. Vooral het compatibel zijn van de impedanties dient te worden geverifieerd. Afwijkingen komen voor.

De keuze van de logische spanningen

Verschillende factoren bepalen de keuze van de niveaus van de logische spanningen. Enerzijds worden de mogelijkheden beperkt door de componenten, anderzijds kan bijvoorbeeld een gewenste storingsdrempel in een gestoorde omgeving ook eisen stellen waaruit een minimaal verschil tussen de logische niveaus voortvloeit. In het algemeen wordt rond elk van de logische niveaus een zekere marge aangehouden. Ligt de aangeboden spanning binnen een van deze marges, dan wordt deze als logische 0 resp. logische 1 geïnterpreteerd.
Bij TTL-schakelingen zijn de voedingsspanningen 0 Volt (aarde) en +5 Volt, waarop naar boven en naar beneden ca. 0,25 Volt mag worden afgeweken. De logische niveaus zijn +0,2 en +3,3 Volt, waarbij de marges voor dit type bouwsteen zijn (positieve logica):

aan de ingang:		*aan de uitgang:*	
$U_{in} \leqslant 0{,}8$ Volt	$\rightarrow 0$	$U_{uit} \leqslant 0{,}4$ Volt	$\rightarrow 0$
$U_{in} \geqslant 2{,}0$ Volt	$\rightarrow 1$	$U_{uit} \geqslant 2{,}4$ Volt	$\rightarrow 1$

Ergens tussen de twee aangegeven gebieden in ligt het ingangsniveau waarbij de uitgang van de poort omslaat. Bij de normale series TTL-schakelingen ligt deze *omslagdrempel* op ca. 1,4 Volt bij 25 graden en op 1,2 Volt bij 80 graden omgevingstemperatuur. De *storingsmarge* ligt in de praktijk daarom hoger dan de hierboven aangegeven marges doen vermoeden. Dit zijn echter wel storingsmarges, hiervan mag men geen gebruik maken bij het ontwerpen.

4.7. De realisatie van TTL-poortschakelingen

Poortschakelingen dienen, zo is gebleken, aan de uitgang een buffer-versterker te bezitten. Deze versterkerwerking kan niet worden gerealiseerd met diodes, echter wel met *transistoren.* Transistoren worden in de digitale techniek vooral als *open-dicht schakelaar* gebruikt. Het is daarom niet noodzakelijk uitvoerig in te gaan op de fysische eigenschappen van transistoren. Wanneer overgangsverschijnselen en andere afwijkingen van het ideale gedrag buiten beschouwing blijven, dan beschrijft fig. 4.28 de schakelaarwerking van een transistor.

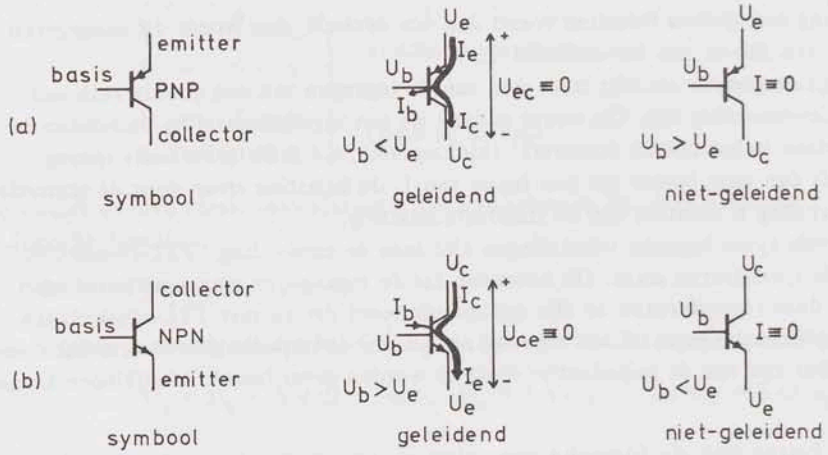


fig. 4.28. De transistor als schakelaar.

Transistoren als schakelaars

De werking van een transistor hangt af van drie spanningen:

U_c : de collectorspanning,

U_b : de basisspanning,

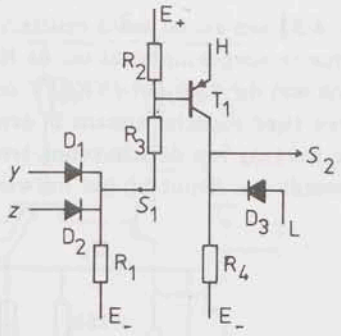
U_e : de emitterspanning.

Voor een PNP-transistor geldt normaal dat $U_e > U_c$ is. Is nu $U_b < U_e$, dan geleidt de transistor. Een klein deel van de emitterstroom I_e verdwijnt als basisstroom I_b, het restant is de collectorstroom I_c. Zie fig. 4.28.a onder de geleidende toestand.
Is $U_b \geqslant U_e$, dan is de transistor in de niet-geleidende toestand. Er loopt dan slechts een zeer geringe lekstroom. Over de transistor staat dan het gehele spanningsverschil tussen U_e en U_c.
In de geleidende toestand, $U_b < U_e$, staat er een spanningsval van ca. 0,2 - 0,5 Volt in het emitter-collector pad over de transistor.

Zoals fig. 4.28.b toont is er een belangrijk verschil tussen een PNP- en een NPN-transistor. Deze laatste geleidt juist als $U_b > U_e$ is. Hier geldt dat $U_c > U_e$ is onder normale condities. Bij het PNP-type was dit juist omgekeerd.

De transistor als versterker

De *versterkerwerking* van transistoren berust op het feit dat met een geringe basisstroom een veel grotere emitter-collectorstroom kan worden geschakeld. Deze verhouding varieert afhankelijk van het type van 1:10 tot 1:200. Daardoor wordt de belasting van het gedeelte van de poort waarin de logische functie wordt gerealiseerd, belangrijk gereduceerd. Fig. 4.29 geeft het schema van een NOR in losse componenten. Deze schakeling is opgebouwd uit een diodepoort en een invertor-versterker.
De diodes D_1 en D_2 vormen samen met R_1 een diodepoort, die in positieve logica een OR realiseert. De weerstanden R_2 en R_3 vormen een spanningsdeler, waardoor $U_b > U_e$ is als S_1 een spanning H voert en $U_b < U_e$ is als S_1 een spanning L voert. Daardoor wordt de transistor in de geleidende of de niet-



Diode-transistor poort.

fig. 4.29. Schema DTL NOR (positieve logica).

geleidende toestand gebracht. Geleidt de transistor, dan voert S_2 een spanning H, terwijl in de niet-geleidende toestand S_2 een spanning L voert. De diode D_3 zorgt ervoor dat de uitgangsspanning niet zakt beneden de spanning L.

Door toepassing van transistor T_1 is een scheiding gerealiseerd tussen het deel dat de logische functie realiseert en de belasting van de poort. Tevens wordt door deze versterkertrap een omkering van de logische spanning geïntroduceerd. Zoals reeds is aangetoond is het ontwerpen van combinatorische schakelingen met NAND's en NOR's nauwelijks moeilijker dan met de AND en de OR als bouwsteen.

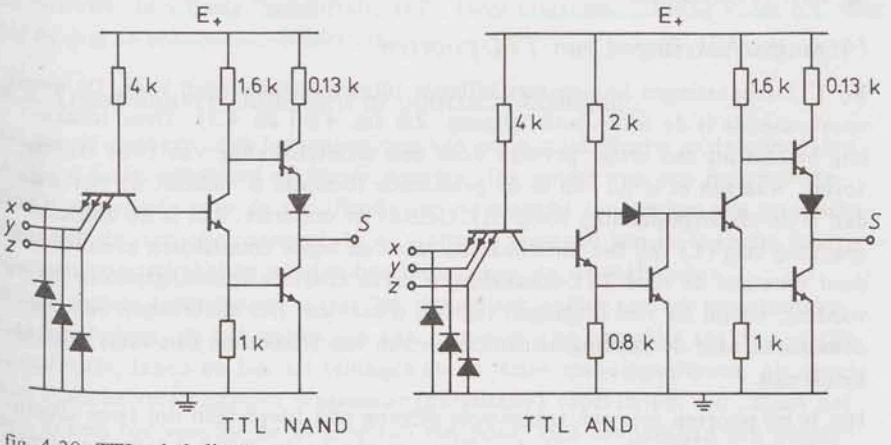


fig. 4.30. TTL-schakelingen.

In fig. 4.30 zijn de schema's getekend van een NAND en een AND, zoals deze in TTL-techniek worden geïntegreerd. De AND-functie wordt in beide gevallen gerealiseerd via een *multi-emitter* transistor aan de ingang. Daarna volgt bij de NAND een versterker-invertor tesamen met een uitgangsschakeling. Deze uitgangsconstructie wordt wel *totem-pole* genoemd. In een AND moet een tussentrap worden geïntroduceerd, die een inverterende werking heeft en aldus de inverterende werking van de uitgangstrap opheft. Daardoor neemt wel de *vertragingstijd* t_p van de poort (*propagation delay*) toe.
In fig. 4.31 is het schema gegeven van een NOR in TTL-techniek. Deze poort is duidelijk gecompliceerder dan de NAND. Dit is de reden van de sterke voorkeur voor de NAND als component in grotere TTL-bouwstenen.

Met stippellijnen is in fig. 4.31 een aantal extra emitters op de ingangstransistoren getekend. Door deze toe te voegen ontstaat uit de NOR de veel voorkomende AND-NOR combinatie, ook wel de *AND-OR-INVERT combinatie* genoemd. In feite zijn in deze bouwsteen twee logische niveaus in één schakelniveau gerealiseerd. Dit komt een snelle werking van de schakeling ten goede. Deze eigenschap van de AND-NOR wordt o.a. benut bij het ontwerpen van snelle optellers. Zie hoofdstuk 11.

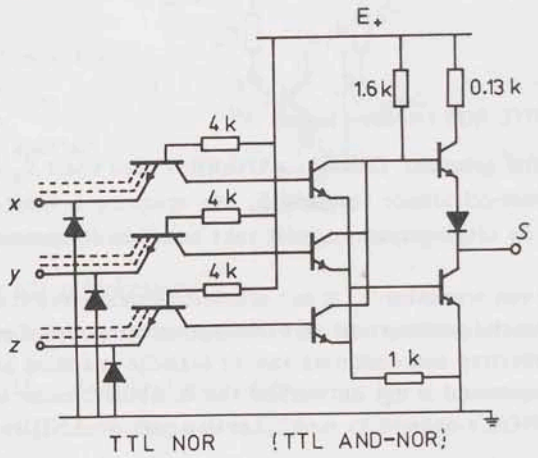


fig. 4.31. TTL-schakeling.

Uitgangsschakelingen van TTL-poorten

Bij TTL-schakelingen komen verschillende uitgangsschakelingen voor. De meest voorkomende is de *totem-pole* uitgang. Zie fig. 4.30 en 4.31. Deze schakeling bestaat uit een *driver* gevolgd door een serieschakeling van twee transistoren, waarvan er altijd één in de geleidende toestand is. Geleidt de bovenste, dan is de uitgangsspanning hoog (H). Geleidt de onderste, dan is de uitgangsspanning laag (L). Bij het omschakelen geleiden beide transistoren even. Hierdoor ontstaan de voor TTL-schakelingen karakteristieke spanningspieken op de voeding, vooral als veel uitgangen tegelijk schakelen. Het aanbrengen van condensatoren over de voedingsaansluitingen van een schakeling kan soms noodzakelijk zijn.

Het is bij poorten met een totem-pole uitgang niet toegestaan om twee uitgangen direct met elkaar te verbinden. Zou een van beide naar het logische niveau H gestuurd worden en de andere naar L, dan ontstaat er een kortsluiting van de voeding via beide transistoren. De uitgangstransistoren kunnen hier als regel niet lang tegen en raken defect. Toch wil men vaak uitgangen met elkaar verbinden, o.a. bij verbindingssystemen (bussen). Zie hoofdstuk 16.
In dat geval gebruikt men poorten met een *open-collector* uitgang. Deze uitgangsschakeling ontstaat uit de "totem-pole" door weglating van de bovenste transistor in de uitgangstrap. Zie fig. 4.32.
Hierdoor is het mogelijk een aantal uitgangen met elkaar te verbinden. Het is dan wel noodzakelijk een *trekweerstand* naar E_+ aan te brengen, omdat anders de uitgang bij niet-geleidende uitgangstransistor zal zweven. Zonder deze trekweerstand wordt het logische niveau H niet bereikt. Verbindt men een aantal open-collector uitgangen met elkaar, dan ontstaat de *wired-AND*.

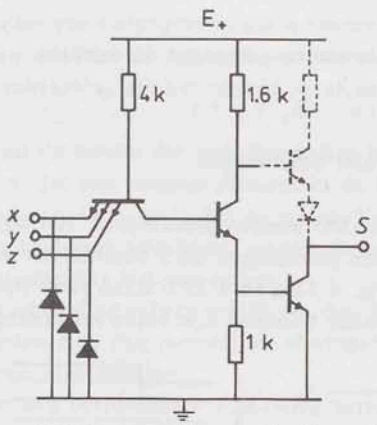


fig. 4.32. Open-collector uitgangsschakeling.

Het derde type uitgangsschakeling is de *3-state* uitgang, ook wel de *free-state* uitgang genoemd. Deze uitgangsschakeling bestaat o.a. uit twee transistoren. Een ervan dient om de uitgang op het logische niveau H te brengen. De andere brengt de uitgang op het logische niveau L. Geleidt geen van beide transistoren, dan is de uitgang noch H noch L. De lijn, waarmee deze uitgang is verbonden ziet deze uitgang dan als een hoogohmige belasting. Hiermee wordt de betreffende uitgang "geneutraliseerd". Deze uitgangsschakeling komt o.a. voor bij de nog te behandelen *busdrivers*.

4.8. Overgangsverschijnselen in poortschakelingen

De beschouwingen over het ontwerpen van poortschakelingen in de paragrafen 4.1 – 4.5 zijn gebaseerd op *ideale poorten*. Dit model van een poortschakeling is voldoende voor de specificatie van de logische functie van een poortschakeling in de *statische toestand*. In de statische toestand kan de logische functie van een poortschakeling worden beschreven met de schakelalgebra.
Bij de meeste toepassingen is ook het *dynamisch gedrag* van een poortschakeling van belang. Bij het ontwerpen kan men zich niet beperken tot de logische specificatie, fan-in en fan-out tellingen en de juiste spanningsniveaus. Als gevolg van fysische verschijnselen, waaronder (parasitaire) capaciteiten e.d., duurt het een zekere tijd voordat de uitgang van een poort zich aanpast aan een ingangsverandering. Bovendien verloopt deze aanpassing vaak niet vloeiend, maar treden overgangsverschijnselen als *overshoot* en *spikes* op. Om de invloed hiervan enigszins te kunnen analyseren dient een poortmodel te worden geïntroduceerd, waarin behalve de logische functie ook andere factoren kunnen worden meegenomen. In een gangbaar model wordt achter een poort een vertragingselement opgenomen. Alle overgangsverschijnselen worden gedacht in dit element geconcentreerd te zijn. We nemen in eerste instantie aan dat het vertragingselement ideaal is, d.w.z. het ingangssignaal een tijd Δt later onvervormd doorgeeft. Met dit eenvoudige model kan reeds een aantal gevolgen van overgangsverschijnselen worden bestudeerd.

Voorbeeld
Realiseer twee schakelingen die aan de uitgangen de signalen

$$S_1 = \bar{y} + \bar{z} \quad \text{en} \quad S_2 = \bar{y} + z$$

afgeven. Inverse signalen zijn niet beschikbaar.
Uitwerking:
Het signaal S_1 kan met één NAND worden gerealiseerd. Het signaal S_2 zou ook met één NAND kunnen worden gerealiseerd als $\bar{z}$ beschikbaar was. Hiervoor is een extra invertor nodig. In fig. 4.33.a en 4.33.b staan twee oplossingen van het gestelde probleem. In de statische toestand zijn beide schakelingen identiek.

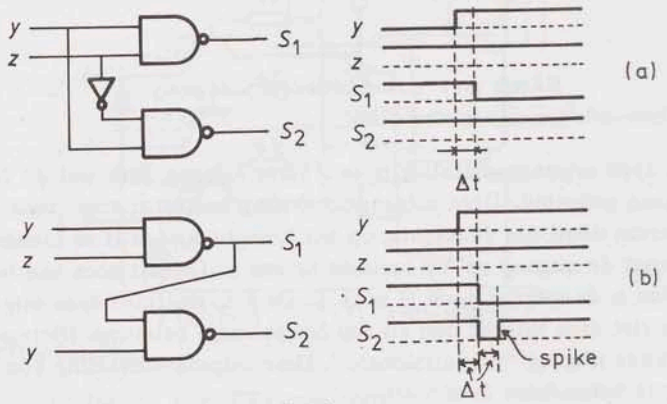


fig. 4.33. Gevolgen van overgangsverschijnselen.

In de dynamische toestand bestaat er een groot verschil tussen beide schakelingen. In de onder a. beschreven schakeling zal na een verandering van y van $0 \rightarrow 1$ ($z = 1$) het signaal S_1 een tijd Δt later een $1 \rightarrow 0$ overgang laten zien. Hierna is de schakeling weer in rust.
De onder b. beschreven schakeling berust op:

$$S_2 = \bar{y} + z = \overline{y \cdot \bar{z}} = \overline{y \cdot (\bar{y} + \bar{z})} \; .$$

In de schakeling is $\bar{z}$ niet beschikbaar, maar $\bar{y} + \bar{z}$ wel. Neemt, uitgaande van de situatie $yz = 01$ het signaal y de waarde 1 aan, dan bereikt in beide schakelingen het signaal S_1 een tijd Δt later zijn nieuwe eindwaarde $S_1 = 0$. In fig. 4.33.a behoudt S_2 de gewenste waarde. In fig. 4.33.b echter treedt in S_2 een ongewenste overgang op. Aanvankelijk gaat S_2 naar 0 als $y = 1$ wordt. Een tijd Δt later gaat S_1 naar 0, waardoor S_2 weer naar de waarde 1 gestuurd wordt. Ga dit na.

Kortdurende overgangsverschijnselen, zoals het signaal S_2 hierboven vertoont, worden *spikes* (pieken, naalden) genoemd. Zij ontstaan o.a. als gevolg van *looptijdverschillen* in poortschakelingen. Sommige ontwerpers huldigen het standpunt dat daarom poortschakelingen zo mogelijk in twee lagen moeten worden gerealiseerd. De looptijdverschillen zijn dan zo gering dat verschijnselen als spikes binnen de perken blijven. Dit standpunt is op zichzelf niet onjuist. Het berust op het uitgangspunt dat de ingangssignalen van de schakeling tegelijk veranderen. Maar of dit zo is moet dan terdege worden geverifieerd.

Behalve ten gevolge van looptijdverschillen binnen een schakeling kunnen spikes ook ontstaan doordat de ingangssignalen van de schakeling niet tegelijk veranderen. De schakeling die het signaal S_2 afgeeft in fig. 4.33.b is hiervan een voorbeeld.
In de praktijk valt de hinder die men van spikes heeft wel mee. Meestal duren ze relatief kort, zodat een volgend element in de keten er niet op reageert. De vermogensinhoud van de spike is dan te gering. Veranderen de ingangssignalen tegelijk, hetgeen heel vaak voorkomt, en telt de schakeling slechts enkele niveaus, dan heeft men nauwelijks last van spikes.
Treden er in een schakeling echter spikes op, dan is het opsporen ervan als regel zeer lastig. Zij uiten zich dan meestal als storingen die zich op volstrekt willekeurige tijdstippen manifesteren.
Het ontwerp van een betrouwbare schakeling vereist daarom dat men de mogelijke gevolgen van spikes a priori in beschouwing neemt bij het opstellen van de ontwerpeisen. Meestal kan men dit reeds aan de hand van het eenvoudige poortmodel in fig. 4.34. Een poort bestaat volgens dit model uit een blokje dat de logische bewerking verricht, gevolgd door een blokje met vertraging Δt. Deze vertraging moet zo worden geïnterpreteerd, dat pas na een tijd Δt het uitgangssignaal zijn eindwaarde bereikt. Ervoor is het signaal niet noodzakelijk representatief voor de ingangscombinatie. Via een zorgvuldige organisatie van de schakeling moet deze noodzakelijke wachttijd worden ingebouwd. De schakeling mag pas naar de uitgang van een poort kijken als diens uitgang stabiel is geworden. Over dit aspect van het ontwerpen meer in de volgende hoofdstukken.

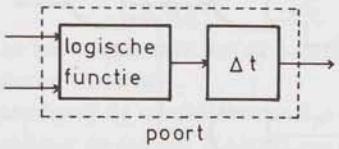


fig. 4.34. Poortmodel.

Opmerking
Op het model in fig. 4.34 zijn verfijningen mogelijk. Een model dat wel wordt gebruikt bij de analyse en simulatie van level mode schakelingen (zie hoofdstukken 6/7) bestaat uit:

t_0: tijdstip der ingangsverandering.

t_0 tot t_1: de uitgang is nog stabiel ≡ zeker.

t_1 tot t_2: de uitgang kan veranderen ≡ onzeker.

$t > t_2$: de uitgang is stabiel ≡ zeker.

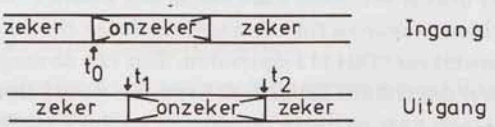
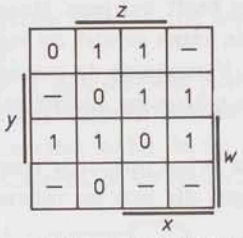


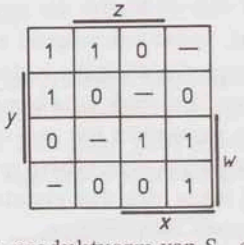
fig. 4.35. Simulatiemodel.

Hierin is de tijd tussen t_0 en t_1 de *minimale propagatietijd* van het signaal door een poort en de tijd tussen t_0 en t_2 de *maximale propagatietijd*, rekening houdend met exemplaarspreiding, omgevingstemperatuur en belasting.

Opgaven

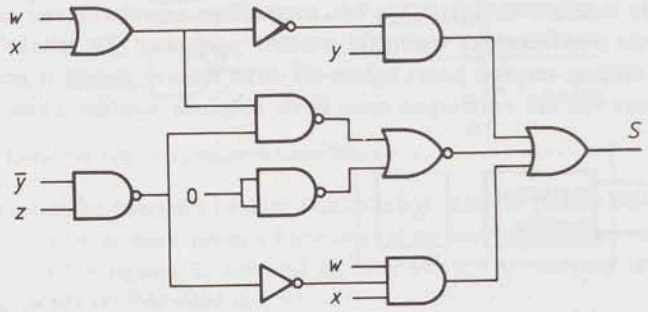
4.1. In de Karnaughdiagrammen zijn twee schakelfuncties S_1 en S_2 gespecificeerd. Van de signalen w t/m z zijn ook de inversen beschikbaar.



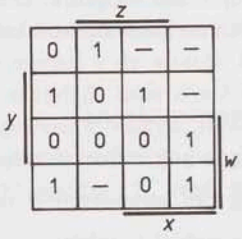


a. Bepaal de eenvoudigste somvorm en produktvorm van S_1 en S_2.
b. Realiseer de schakelingen met AND en OR poorten. Optimaliseer achtereenvolgens naar de volgende criteria:
   1. Een minimum aantal poorten.
   2. Een minimum aantal ingangen.
   3. Een minimale propagation delay door de schakeling.

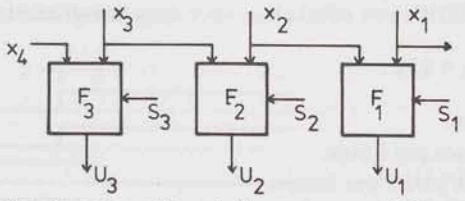
4.2. Welke schakelfunctie behoort bij het getekende poortnetwerk?



4.3. Ontwerp een poortschakeling voor de schakelfunctie S. Beschikbaar zijn AND en OR poorten met slechts twee ingangen.



4.4. Voor een zeker doel is het nodig alle enen in een woord (= rij nullen en enen) naar rechts te schuiven tot een aaneengesloten rij. Zo moet het woord 1011001 na bewerking 0001111 opleveren. Een van de mogelijkheden om dit te bereiken is om telkens het gedeelte van een woord links van de meest rechtse 0 één plaats naar rechts te verplaatsen en links een 0 in te schuiven. De verplaatsing over één positie kan worden gerealiseerd via een combinatorische schakeling, die uit een aantal identieke cellen bestaat.

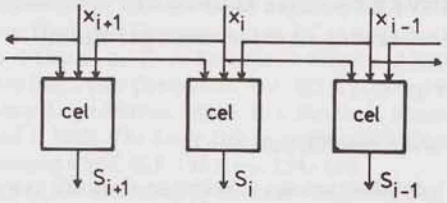


De ingangssignalen van cel no. i zijn x_i en x_{i+1} (bits van het woord) en een signaal S_i dat aangeeft of er rechts van x_i een 0 voorkomt. $S_i = 1$ als er geen nullen rechts van x_i staan.

a. Geef een eenvoudige formulering voor S_i.
b. Specificeer de schakelfunctie F_i van de cel.

4.5. Een schakeling doorloopt de getallen 0 t/m 15 (binair).
Ontwerp met NAND's een schakeling die detecteert of de aangeboden combinatie tussen 10 en 15 ligt. S = 1 als het getal in het gebied ligt, de grenzen ervan inbegrepen.

4.6. Van een rij nullen en enen moet bit x_i van plaats wisselen met bit x_{i-1} als $x_i x_{i-1} = 10$. Bit x_i moet van plaats wisselen met bit x_{i+1} als $x_{i+1} x_i = 10$.



a. Kan het voorkomen dat de paren $x_i x_{i-1}$ en $x_{i+1} x_i$ tegelijk van plaats moeten wisselen?
b. Specificeer de schakelfunctie S_i van de cel.
c. Realiseer de cel met NAND's resp. met NOR's als bouwstenen.

4.7. Realiseer met NAND's de schakelfuncties:

$$S_1 = xy\bar{z} + x\bar{y}z + \bar{x}yz$$

$$S_2 = \bar{w}yz + w\bar{x}\bar{y} + w\bar{x}\bar{z}$$

$$S_3 = \bar{w}z + w\bar{x}y + wx\bar{y}\bar{z}\,.$$

De inverse signaalwaarden zijn niet beschikbaar.

4.8. Van vier signalen w t/m z zijn de inversen niet beschikbaar. Realiseer met NAND's de schakelfunctie:

$$S = wx\bar{y} + wx\bar{z} + \bar{x}yz\,.$$

Gebruik zo weinig mogelijk poorten.

4.9. Van vijf signalen v t/m z zijn ook de inversen beschikbaar. Realiseer de schakelfunctie:

$$S = (v + w)(x + y + z)$$

met NAND's. Gebruik zo weinig mogelijk poorten.

4.10. Ontwerp met NOR's een schakeling voor de schakelfunctie:

$$S = xy + \bar{x}\bar{y}\bar{z} .$$

Beschikbaar zijn:
- Invertoren, zes per huisje,
- 2-input NOR's, vier per huisje,
- 3-input NOR's, drie per huisje.

Hoeveel huisjes kost dan uw oplossing?

4.11. Van drie signalen x, y en z zijn de inversen niet beschikbaar.

a. Realiseer de schakelfunctie

$$S = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz .$$

Beschikbaar zijn:
- Invertoren, zes per huisje,
- 2-input NAND's, vier per huisje,
- 3-input NAND's, drie per huisje,
- 4-input NAND's, twee per huisje,
- AND-OR-INVERT poorten (3–3–3–3).

b. Hoe wordt de schakeling als elders reeds het signaal

$$p = xy + xz + yz$$

en de inverse ervan beschikbaar zijn?

4.12. Ontwerp met 2:1 datakiezers een schakeling voor elk van de volgende logische functies. Inverse signaalwaarden zijn beschikbaar.

$$S_1 = yz$$

$$S_2 = y + z$$

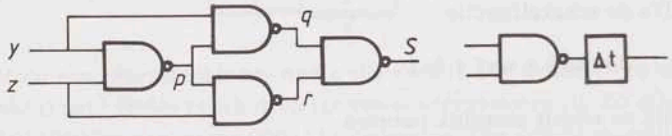
$$S_3 = x\bar{y} + yz$$

$$S_4 = xy + xz$$

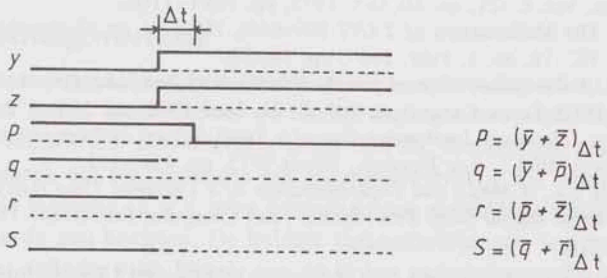
$$S_5 = wx + yz$$

$$S_6 = wxy + \bar{x}\bar{y}\bar{z} + xz .$$

4.13. Gegeven is een schakeling die uit vier NAND's is opgebouwd. Deze schakeling realiseert de EX-OR functie.



Vul het volgende tijddiagram aan.



4.14. Welk verband bestaat er tussen de begrippen positieve logica en negatieve logica enerzijds en het begrip duaal in de schakelalgebra anderzijds? Licht uw antwoord toe.

Literatuur

1. W.N. Carr and J.P. Mize, *MOS/LSI Design and Application,* Texas Instruments, McGraw-Hill Book Company, New York 1972, pp. 229–258.
2. D.T. Ellis, *A Synthesis of Combinational Logic with NAND and NOR Elements,* IEEE Tr. on Electronic Computers, Vol. EC–14, no. 5, Oct. 1965, pp. 701–705.
3. L. Hellerman, *A Catalog of Three-Variable Or-Invert and And-Invert Logical Circuits,* IEEE Tr. on Electronic Computers, Vol. EC–12, June 1963, pp. 198–223.
4. J.W. Jones, *Array Logic Macros,* IBM J. Res. Develop., March 1975, pp. 120–126.
5. G.A. Maley and J. Earle, *The Logic Design of Transistor Digital Computers,* Prentice-Hall, Englewood Cliffs, N.J. 1963, pp. 114–159.
6. E.J. McCluskey Jr., *Minimization of Boolean Functions,* Bell Syst. Tech.J., Nov. 1956, pp. 1417–1444.
7. B. Norris e.a., *Digital Integrated Circuits and Operational-Amplifier and Opto-electronic Circuit Design,* Texas Instruments Electronics Series, McGraw-Hill, New York 1976.
8. J.A. Scarlett, *Transistor-Transistor Logic and its Interconnections,* Van Nostrand, London 1972.
9. T.F. Tabloski and F.J. Mowle, *A Numerical Expansion Technique and its Application to Minimal Multiplexer Logic Circuits,* IEEE Tr. on Computers, Vol. C–25, July 1976, pp. 684–702.
10. H. Taub and D. Schilling, *Digital Integrated Electronics,* McGraw-Hill, New York 1977.
11. *The TTL Data Book for Design Engineers,* Second Edition 1977, Texas Instruments.
12. H.C. Torng, *Switching Circuits, Theory and Logical Design,* Addison-Wesley, Reading, Mass., 1972, pp. 103–130.

Voor een verdere bestudering van het onderwerp combinatorische schakelingen worden de volgende artikelen en boeken aanbevolen:

13. R.L. Ashenhurst, *The Decomposition of Switching Functions,* Annals Computation Lab., Harvard Univ., Vol. 29, Cambridge, Mass., 1959, pp. 74–116.
14. S.G. Chappell, *Simulation of Large Asynchronous Logic Circuits using an Ambiguous Gate Model,* Proc. Fall Joint Computer Conference, 1971, pp. 651–660.
15. H.A. Curtis, *Design of Switching Circuits,* Van Nostrand, Princeton, N.J. 1962.
16. M. Davio and J.P. Deschamps, *Symmetric Discrete Functions,* Philips Res. Repts no. 27, 1972, pp. 405–445.
17. J.P. Deschamps, *Partially Symmetric Switching Functions,* Philips Res. Repts no. 28, 1973, pp. 245–264.
18. E.B. Eichelberger, *Hazard Detection in Combinational and Sequential Switching Circuits,* IBM J. Res. Develop., March 1965, pp. 90–99.
19. H. Fleisher and L.I. Maissel, *An Introduction to Array Logic,* IBM J. Res. Develop., March 1975, pp. 98–109.

20. J. Frackowiak, *The Synthesis of Minimal Hazardless TANT Networks,* IEEE Tr. on Computers, Vol. C–21, no. 10, Oct. 1972, pp. 1099–1108.
21. J.F. Gimpel, *The Minimization of TANT Networks,* IEEE Tr. on Electronic Computers, Vol. EC–16, no. 1, Febr. 1967, pp. 18–38.
22. E. Kjelkerud, *A Computer Program for the Synthesis of Switching Circuits by Decomposition,* IEEE Tr. on Computers, Vol. C–21, June 1972, pp. 568–572.
23. J.C. Logue e.a., *Hardware Implementation of a Small System in Programmable Logic Arrays,* IBM J. Res. Develop., March 1975, pp. 110–119.
24. S.A. Szygenda e.a., *A Model and Implementation of a Universal Time Delay Simulator for Large Digital Nets,* Proc. Spring Joint Computer Conference 1970, pp. 207–216.
25. H.A. Vink, B. van den Dolder and J. Al, *Reduction of CC-tables Using Multiple Implication,* IEEE Trans. on Computers, Vol. C–27, no. 10, Oct. 1978, pp. 961–966.
26. H.A. Vink, *Minimal TANT Networks of Functions with DON'T CARE's and Some Complemented Input Variables,* IEEE Trans. on Computers, Vol. C–27, no. 11, Nov. 1978, pp. 1073–1078.

5. ENKELVOUDIGE GEHEUGENELEMENTEN

5.1. Geheugenwerking

Het uitgangssignaal van een *combinatorische schakeling* wordt in de statische toestand geheel bepaald door de huidige ingangscombinatie. Bij andere schakelingen, *sequentiële schakelingen*, is dit niet zo. Sequentiële schakelingen hebben als kenmerk dat bij tenminste één ingangscombinatie het uitgangssignaal in de statische toestand (d.w.z. geen veranderingen meer in de schakeling) meer dan één waarde kan bezitten. De huidige ingangscombinatie legt het uitgangssignaal dan niet volledig vast. Enkele van de in het verleden aangeboden ingangscombinaties, alsmede de volgorde ervan, beïnvloeden mede het huidige uitgangssignaal. Om dit mogelijk te maken dient een sequentiële schakeling *geheugenwerking* te bezitten. Geheugenwerking en schakelingen waarmee men deze kan realiseren zijn het onderwerp van dit hoofdstuk.
De eerste vraag die men zich bij de bestudering van geheugenwerking stelt is, welke functie(s) een geheugenelement moet kunnen realiseren. Enkele voorbeelden zullen deze vraag toelichten.

Voorbeeld. *Inbraakalarm.*
Een schakeling voor een inbraakalarm bevat o.a. een deurkontakt. Bij het openen moet de alarmbel gaan rinkelen. Uiteraard moet deze bel blijven rinkelen, ook al wordt de deur onmiddellijk weer gesloten. Hiervoor is een geheugenelement nodig. De installatie bevat een aparte resetknop voor het afzetten van het alarm.

Voorbeeld. *Verlichting van drukknoppen in liften.*
Bij sommige liftinstallaties gaat achter de knop van de gewenste verdieping een lampje branden nadat deze knop is ingedrukt. Wanneer de gewenste verdieping is bereikt gaat het lampje weer uit. Hetzelfde geldt voor de verlichting van de oproepknoppen voor omhoog en omlaag buiten de lift.

Voorbeeld. *Autobusschakeling.*
Als in een bus een passagier op zijn stopknop drukt gaat een (meestal rode) lamp branden. Hij geeft hiermee aan de chauffeur te kennen op de eerstvolgende halteplaats te willen uitstappen. De lamp blijft branden als de passagier zijn knop loslaat (geheugenwerking). Na de halte kan de chauffeur de schakeling resetten, waardoor de lamp weer dooft. Hij drukt daartoe op een resetknop.

Deze voorbeelden, en men bedenke zelf vele andere, hebben het volgende gemeenschappelijk:
- Er is één situatie (deur openen, verdieping aangeven, stopknop drukken) waarbij een lamp moet gaan branden en vervolgens moet blijven branden.
- De lamp wordt gedoofd als op een resetknop wordt gedrukt. Bij een liftinstallatie is dit meestal een sleepkontakt.
- Er bestaat een toestand (deur gesloten, niemand drukt) waarbij de lamp wel of niet brandt, afhankelijk van de laatst gepleegde actie.

Proberen we de gewenste geheugenwerking te specificeren met twee uitspraken S en R:

"De setknop wordt gedrukt" $S = 1 \leftrightarrow$ waar

en

"De resetknop wordt gedrukt" $R = 1 \leftrightarrow$ waar

dan blijkt dit niet te lukken. Als S = R = 0 is, dan hangt het van de voorgeschiedenis af of de lamp brandt. Behalve de twee ingangssignalen S en R, welke de waarheidswaarde van genoemde uitspraken vastlegt, moeten we nog een derde grootheid introduceren. We noemen deze de *toestand van het geheugenelement* dat de lamp bedient. Geven we de toestand ervan aan met Z:

"Het geheugenelement staat in de werkstand" Z = 1 ↔ waar

en geven we het branden van de lamp aan met L:

"De lamp brandt" L = 1 ↔ waar

dan kunnen we tabel 5.1 opstellen. Deze tabel beschrijft het gewenste uitwendige gedrag van de schakeling. Hieruit zullen de werkingseisen van het geheugenelement Z worden afgeleid.

S	R	Z	L	
0	0	0	0	onthouden
0	0	1	1	
0	1	0	0	resetten
0	1	1	0	
1	0	0	1	setten
1	0	1	1	
1	1	0	–	nog te bepalen
1	1	1	–	

tabel 5.1. Waarheidstabel met onthoudfunctie.

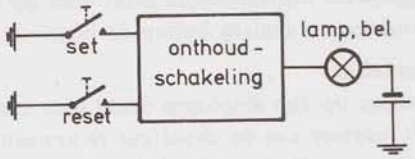


fig. 5.1. Schakeling met onthoudfunctie.

De eerste twee regels van tabel 5.1 geven weer dat als S = R = 0 het geheugenelement Z bepaalt of de lamp moet branden. De volgende twee regels specificeren dat als SR = 01 de lamp moet doven, en wel onafhankelijk van Z. Is de ingangscombinatie SR = 10, dan moet de lamp (gaan) branden. Dit laatste ook weer onafhankelijk van de stand van Z.
De laatste twee regels van tabel 5.1 geven geen specificatie van de gewenste uitkomst als tegelijk op de setknop en op de resetknop wordt gedrukt. Het zal bij een alarminstallatie duidelijk zijn dat het setsignaal moet overheersen. Bij een lift is dit precies andersom. De busschakeling stelt misschien wat minder hoge eisen, zowel de set als de reset mag hier overheersen. Afhankelijk van het doel van de schakeling kunnen we de specificatie dus aanvullen.

In tabel 5.1 is gespecificeerd wanneer de lamp moet branden als op de set- of de resetknop wordt gedrukt. Daarbij bleek een geheugenelement Z noodzakelijk te zijn om uit te maken wat er moet gebeuren als er in het geheel niet gedrukt

wordt. Uit deze tabel moeten nu de werkingseisen van het geheugenelement worden afgeleid:

- Uit de eerste twee regels van tabel 5.1 volgt dat het geheugenelement (tenminste) in twee standen moet kunnen staan. Deze standen zijn aan te duiden met "0" en "1" en corresponderen met het branden van de lamp. Als de lamp brandt, dan is L = 1 en Z = 1.
- De volgende twee regels geven de resetsituatie weer. Bij het drukken op de resetknop moet de lamp ophouden met branden. Nadat de resetknop weer losgelaten is moet de lamp gedoofd blijven. Het drukken van de resetknop moet dus tot gevolg hebben dat Z = 0 wordt.
- Evenzo heeft het drukken op de setknop tot noodzakelijk gevolg dat Z in de stand 1 komt. Immers, alleen dan weet de schakeling bij het loslaten van de setknop wat er met de lamp moet gebeuren.
- Wordt tegelijk op de set- en de resetknop gedrukt, dan zal het van het gebruiksdoel van de schakeling afhangen wat er moet gebeuren. Mogelijkheden zijn een overheersende set, een overheersende reset of we beslissen dat de voorlaatste actie bepalend is, een extra onthoudstand dus.

Hiermee zijn we gekomen tot drie mogelijke varianten van het geheugenelement. Tabel 5.2 geeft de specificatie ervan. In deze tabel is onder z de huidige toestand van het geheugenelement Z vermeld en onder Z de nieuwe. De kolom onder Z_0 beschrijft het geheugenelement met overheersende reset, die onder Z_1 met overheersende set en onder Z_z wordt het geheugenelement met een extra onthoudstand beschreven.

S	R	z	Z_0	Z_1	Z_z	
0	0	0	0	0	0	onthouden
0	0	1	1	1	1	
0	1	0	0	0	0	reset
0	1	1	0	0	0	
1	0	0	1	1	1	set
1	0	1	1	1	1	
1	1	0	0	1	0	extra stand
1	1	1	0	1	1	

tabel 5.2. Waarheidstabellen voor geheugenelementen.

Notatie

In het vervolg wordt de oude of huidige stand van een geheugenelement aangegeven met een kleine letter; de gewenste nieuwe stand met een hoofdletter. Is de oude stand gelijk aan de nieuwe, dan is Z = z, anders is Z = $\bar{z}$.

De codering van de set- en resetcombinatie

De set, reset en onthoudstand zijn hierboven gecodeerd als

SET : SR = 10
RESET : SR = 01
ONTHOUD : SR = 00

Men kan deze codering niet willekeurig kiezen. De onthoudstand moet zowel

uit de set als uit de resetcombinatie *zonder overgangsverschijnselen* kunnen worden bereikt. Dit betekent dat een codering zoals

SET : SR = 00
RESET : SR = 10
ONTHOUD : SR = 11

niet kan worden gebruikt. Ga dit na! In het vervolg spreken we af dat de setcombinatie wordt gecodeerd als SR = 10, de resetcombinatie als SR = 01 en de ingangscombinatie waarbij moet worden onthouden als SR = 00. De vierde combinatie SR = 11 bestemmen we voor één van de genoemde mogelijkheden, afhankelijk van de omstandigheden.

Opmerking
Het signaal S wordt vaak het *setsignaal* genoemd en R het *resetsignaal*. In feite bepaalt echter de *combinatie* van S en R wat het geheugenelement gaat doen. Men zij dus bedacht op een mogelijke spraakverwarring in deze.

5.2. Realisaties van geheugenelementen

De drie Karnaughdiagrammen van fig. 5.2 volgen rechtstreeks uit tabel 5.2 en geven exact dezelfde informatie als deze tabel. De randschriften duiden de huidige ingangscombinatie en de huidige of oude toestand van het geheugenelement aan. In elk vakje is de gewenste nieuwe stand van het geheugenelement genoteerd, die behoort bij de oude toestand ervan en de aangeboden ingangscombinatie.

In feite hebben we hier te maken met twee *binaire ingangsvariabelen* s en r, welke de ingangscombinatie coderen, en een *binaire inwendige* of *toestandsvariabele* die de stand van het geheugenelement aangeeft. In het vervolg duiden we de signalen S en R (en andere) aan met hoofdletters. De bijbehorende binaire variabelen met kleine letters.

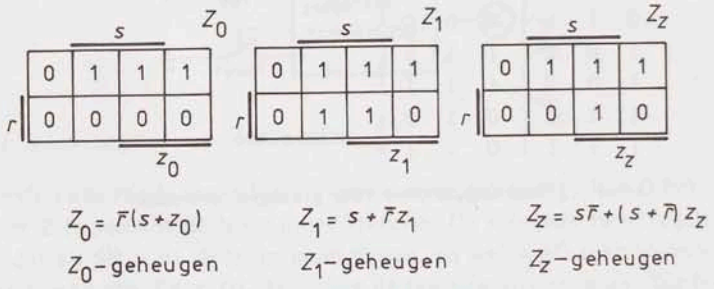


fig. 5.2. Karnaughdiagrammen van geheugenschakelingen.

De bovenstaande geheugenschakelingen kan men in principe realiseren met OR- en AND-poorten. Het Z_0-*geheugenelement* ziet er dan uit als in fig. 5.3.
Aan deze opzet van het Z_0-geheugenelement kunnen bezwaren kleven. Stel dat op zeker moment de ingangscombinatie SR = 10 gevolgd wordt door SR = 00. Het geheugenelement Z_0 staat dan zolang als SR = 00 blijft in de werkstand, gecodeerd als Z_0 = 1. De in fig. 5.4 getekende situatie treedt dan op. Deze situatie is slechts stabiel als het rondgaande signaal in de geheugenlus niet verzwakt wordt (denk aan diodepoorten). Ergens in de lus moet versterking aanwezig zijn, d.w.z. herstel van de spanning tot de gedefinieerde logische niveaus.

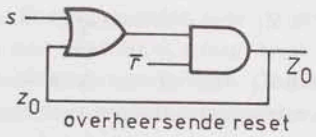


fig. 5.3. Realisatie van het Z_0-geheugenelement.

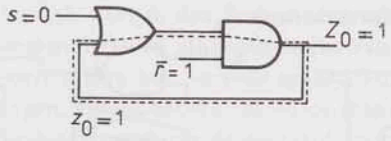


fig. 5.4. Z_0-geheugenelement in de onthoudstand $Z_0 = 1$.

Deze versterking kan op elke plaats in de lus worden aangebracht. Onder andere twee in serie geschakelde invertoren in deze lus kunnen de gewenste versterking leveren. (TTL-poorten bezitten altijd versterking.)

Men kan de schakeling uit fig. 5.3 bouwen met twee identieke NOR's. Fig. 5.5 geeft de omzetting aan. Deze symmetrisch opgebouwde schakeling is ook te vinden met formule manipulaties:

$$Z_0 = \bar{r}(s + z_0) = \overline{\bar{r}(s + z_0)} = \overline{r + \overline{(s + z_0)}} = \mathrm{NOR}(r,\mathrm{NOR}(s,z_0)).$$

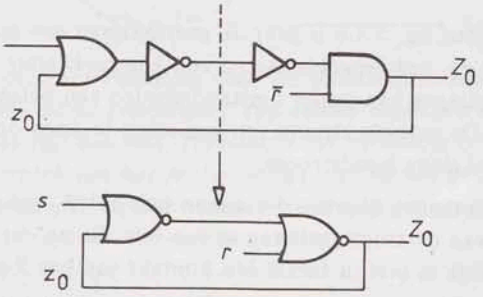


fig. 5.5. Z_0-geheugenelement met NOR's.

Op dezelfde wijze kunnen schakelingen ter realisatie van het *Z_1-geheugenelement* worden ontworpen. Fig. 5.6 geeft twee realisaties. De realisatie met de OR- en AND-poort volgt uit de formule voor Z_1, de symmetrische schakeling met NAND's is weer met formule manipulaties te vinden. (Merk op dat de NAND-oplossing vergt dat de inverse waarden van het set- en resetsignaal worden aangeboden.)

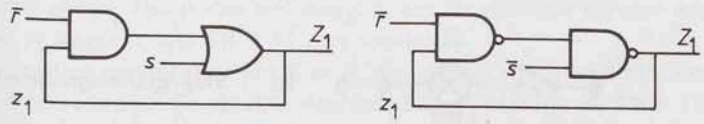


fig. 5.6. Realisaties van het Z_1-geheugenelement.

Een realisatie van het *Z_z-geheugenelement* kost wat meer poorten. Vergelijken we de formule voor Z_z met die voor Z_1, dan valt direct een overeenkomst op:

$$Z_1 = s + \bar{r}z_1 \qquad Z_z = s\bar{r} + (s + \bar{r})z_z \ .$$

Vervangt men s in de formule voor Z_1 door $s\bar{r}$ en r door $\bar{s}r$, dan volgt hieruit dat het Z_z-geheugenelement kan worden gerealiseerd met één Z_1-geheugenelement en twee poorten. Een voorwaarde is dat de variabelen s en $\bar{r}$ beschikbaar zijn (zie fig. 5.7.a). De schakeling in fig. 5.7.b heeft dit bezwaar niet. Een geïntegreerd circuit is voldoende om deze schakeling te realiseren. Ga na hoe deze schakeling volgt uit de formule voor Z_z.

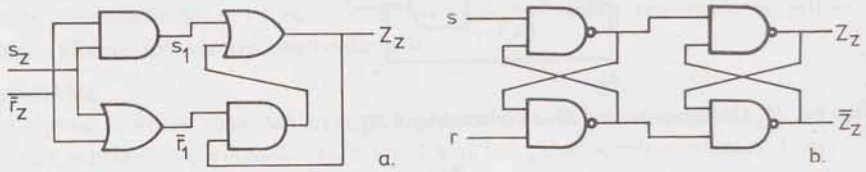


fig. 5.7. Realisaties van het Z_z-geheugenelement.

Realisatie van geheugenelementen met behulp van relais

De gedefinieerde Z_0, Z_1 en Z_z-geheugenelementen kunnen met behulp van één relais worden gerealiseerd. Fig. 5.8 geeft twee uitvoeringen van het Z_1-geheugenelement met één relais. Het ontwerp in fig. 5.8.a is een directe realisatie van de formule

$$Z_1 = s + \bar{r}z_1 \ .$$

Bij de uitvoering volgens fig. 5.8.b is gebruik gemaakt van een relais met gescheiden wikkelingen, een zg. *opkomwikkeling* en een *houdwikkeling*. Het gebruik van gescheiden wikkelingen kan onder omstandigheden een belangrijke energiebesparing opleveren. De vereiste stroom om een relais te doen opkomen is veel groter dan de noodzakelijke houdstroom.

In fig. 5.8.c is een schakeling gegeven die samen met het Z_1-geheugenelement een realisatie vormt van de stopsignalering in een bus. Ga na dat een stopsignalering ook mogelijk is met in totaal één kontakt van het Z-relais.

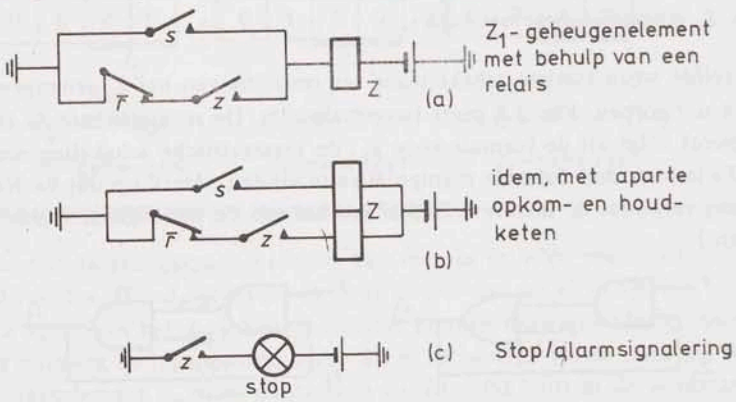


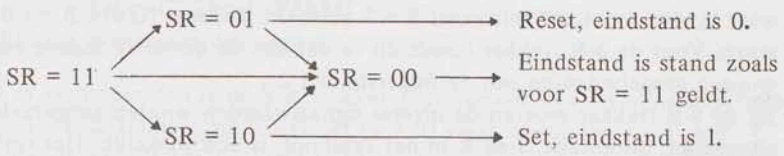
fig. 5.8. Z_1-geheugenelement met relais.

5.3. Het SR = 11 probleem bij geheugenelementen

De uitgangen van de geheugenelementen Z_0, Z_1 en Z_z verschillen uitsluitend van elkaar als de set- en resetsignalen beide 1 zijn, dus bij de "overbodige"

ingangscombinatie. Bovendien geldt voor de geheugenelementen Z_0 en Z_1 in de uitvoeringen met twee NOR's resp. twee NAND's dat als SR = 11 is beide uitgangen niet elkaars inverse zijn. Onder omstandigheden kan dit erg hinderlijk zijn. Voor de overige ingangscombinaties zijn de beide uitgangen nl. wel elkaars inverse.
Er kleeft echter nog een bezwaar aan het gebruik van de SR = 11 combinatie bij geheugenelementen. Stel dat beide signalen S en R de waarde 1 hebben. Worden zij nu tegelijk weer 0, dan leidt dit tot een zg. *raceconditie*. Het zal van toevallige omstandigheden afhangen welke van de twee signalen het eerst de drempel passeert tussen het 1-niveau en het 0-niveau van de corresponderende poortingangen. Onder andere spelen de (capacitieve) belasting van een uitgang en de omslagdrempel van de gestuurde poortingang hierbij een rol.
Al deze omstandigheden veroorzaken dat het meestal onzeker is welk signaal het eerst als 0 wordt geïnterpreteerd, hetgeen resulteert in een onzekerheid over de te bereiken toestand van het geheugenelement.

Zij gegeven een Z_z-geheugenelement. Bij de ingangscombinatie SR = 11 onthoudt dit geheugenelement. Gaat SR = 11 naar SR = 00 dan kan er het volgende geschieden:



Men kan proberen een oplossing voor dit probleem te zoeken door een van de ingangen, de set- of de resetingang, een andere omslagdrempel te geven. De uitslag van de race ligt dan vast. Hiermee is het probleem echter niet opgelost! *Immers, de oorzaak van het probleem ligt niet bij het geheugenelement, maar in het feit dat S en R tegelijk veranderen.* En dit hangt af van de sturende schakeling.

Conclusie

In de praktijk dient men zoveel mogelijk te vermijden dat de SR = 11 ingangscombinatie optreedt in verband met de onzekerheid die er dan bestaat over de te bereiken eindstand van het geheugenelement.
Bij de in par. 5.1 gegeven voorbeelden is het niet mogelijk de SR = 11 combinatie uit te sluiten. De stand van een geheugenschakeling in een bus bijvoorbeeld bij gelijktijdig drukken door passagier en chauffeur hangt af van degene die het eerst zijn knop loslaat. En dat dit wel eens tot problemen leidt weet iedere geregelde gebruiker van het openbaar vervoer. Intern in schakelingen is dit geheel anders. Het is dan wel mogelijk aan de sturende signalen een beperking op te leggen zodat SR = 11 niet voorkomt.
De drie andere combinaties van S en R zijn precies toereikend om het geheugenelement te bedienen. We spreken daarom af in de praktijk de SR = 11 combinatie indien mogelijk te mijden. Uitzonderingen komen echter voor.

Houden we ons aan deze afspraak bij het ontwerpen van schakelingen met geheugenwerking, dan is het geïntroduceerde verschil tussen het Z_0, Z_1 en Z_z-geheugenelement van theoretische aard. We komen dan tot twee vormen van

het geheugenelement, de *S-R trekker* (Am.: *S-R latch*) en de $\bar{S}$-$\bar{R}$ *trekker* (Am.: $\bar{S}$-$\bar{R}$ *latch*) in de figuren 5.9 en 5.10. Vergelijk ook fig. 5.5 en fig. 5.6.

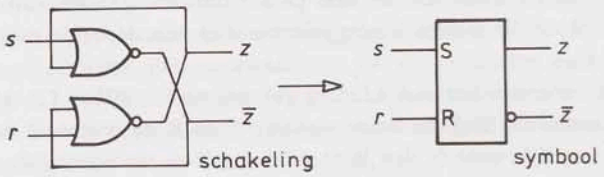


fig. 5.9 Schakeling en symbool van S-R trekker.

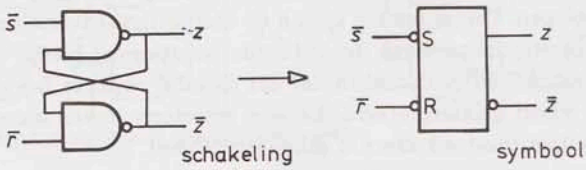


fig. 5.10. Schakeling en symbool van $\bar{S}$-$\bar{R}$ trekker.

De symbolen behoeven een toelichting. Als de uitspraak:

"De trekker moet geset worden"

waar is, dan moet het setsignaal S = 1 gemaakt worden, terwijl R op 0 blijft staan. Voor de S-R trekker houdt dit in dat aan de bovenste ingang een 1 moet worden aangeboden en aan de onderste een 0.
Bij de $\bar{S}$-$\bar{R}$ trekker moeten de inverse signaalwaarden worden aangeboden. Het alternatief, de notatie $\bar{S}$ en $\bar{R}$ in het symbool, is ook mogelijk. Het verband met de waarheidswaarde van de uitspraak:

"De trekker moet geset worden"

is dan minder duidelijk.

5.4. Toepassingen

De S-R en de $\bar{S}$-$\bar{R}$ trekker worden relatief weinig als geheugenelement gebruikt. De reden is dat in een aantal toepassingen racecondities kunnen optreden, die mogelijk een onjuiste werking van de schakeling tot gevolg hebben. We beschouwen deze problemen in volgende hoofdstukken.
Bij sommige toepassingen bewijzen trekkers als geheugenelement echter goede diensten. Enkele voorbeelden volgen hieronder.

Voorbeeld. *Het dendervrij maken van wisselkontakten.*
Wanneer met drukknoppen en/of relaiskontakten een apparaat wordt bediend, dat uit relatief snelle elektronische componenten is samengesteld, ontstaan vaak moeilijkheden t.g.v. het *denderen* van de kontakten. Met denderen bedoelt men het verschijnsel dat kontaktveren bij het omleggen van een kontakt vaak nog enkele keren lossprингen alvorens tot rust te komen. Snelle elektronische schakelingen registreren dan als regel meer dan één puls bij slechts één keer drukken op een drukknop. Fig. 5.11 geeft een schakeling voor het *ontdenderen* van een wisselkontakt (anti-bounce circuit).
De toegepaste $\bar{S}$-$\bar{R}$ trekker registreert slechts één verandering bij elke omlegging van het wisselkontakt, ongeacht het aantal keren dat het kontakt aan de maak- resp. verbreekzijde dendert.

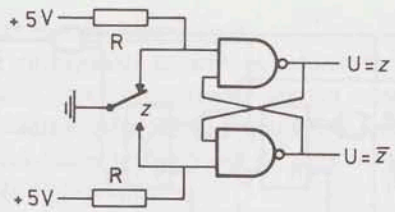


fig. 5.11. Anti-dender schakeling.

Voorbeeld. *Detectie draairichting van een as.*
Van een langzaam draaiende as moet de draairichting zichtbaar gemaakt worden. Een oplossing staat in fig. 5.12. Op de as zit een geaarde borstel die langs twee sleepkontakten a en b beweegt.

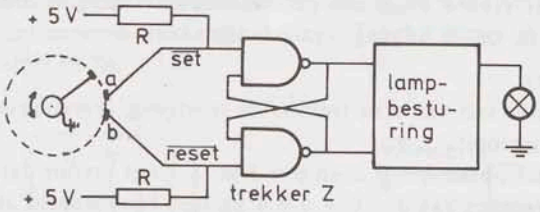


fig. 5.12. Richting detectie van draaiende as.

Draait de as rechtsom dan is de $\bar{S}$-$\bar{R}$ trekker Z gedurende een omwenteling slechts kort in de 1-stand. Immers, na iedere set (a = 0) volgt direct een reset (b = 0). De lamp brandt dan slechts kort. Bij linksom draaiende as zal de lamp lang branden. Aan het kort of lang oplichten van de lamp, resp. zwak of fel branden, kan men de draairichting aflezen. De trekker Z heeft een dubbele functie in deze schakeling:
- als anti-denderschakeling gedurende het schakelen.
- als onthoudschakeling erna.

Voorbeeld. *Pulsgever.*
Men wil bij het testen van schakelingen de mogelijkheid bezitten om via handpulsen de schakeling stap voor stap te testen. Dit kan via een pulsgever, bestaande uit een drukknop gevolgd door een anti-denderschakeling. Het is echter juister om van de klokpulsreeks, waar het te testen apparaat op moet gaan werken, bij iedere keer drukken telkens één puls door te laten. De testpuls heeft dan dezelfde breedte als de normale puls! Fig. 5.13. specificeert de gewenste werking.

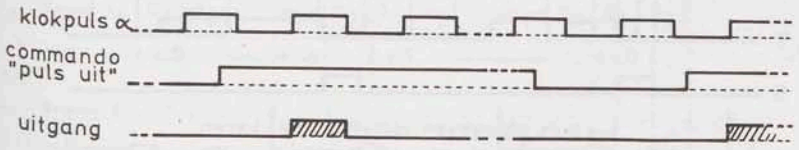


fig. 5.13. Specificatie van pulsgever.

We gaan er bij het oplossen van dit probleem van uit dat het drukken op de knop veel langer duurt dan één periode van de klokpuls α. Een mogelijke oplossing voor dit probleem staat in fig. 5.14.

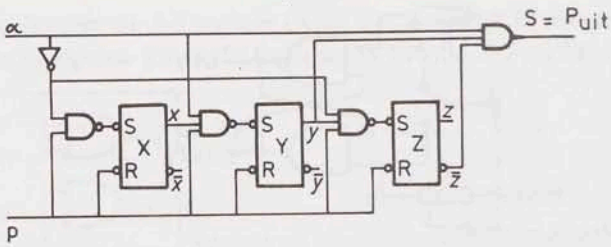


fig. 5.14. Pulsgever.

De logische werking van de pulsgever kan gemakkelijk worden geanalyseerd met een tijddiagram. We onderscheiden de volgende situaties:

– p = 0 (niet gedrukt).
Elke trekker X, Y en Z krijgt een permanente reset aangeboden. Omdat trekker Y = 0 is, zal de uitgang S ook 0 zijn.

– p = 1 (gedrukt).
Wordt p = 1, dan valt van elke trekker de reset weg. Vervolgens komen X, Y en Z na elkaar op:
Trekker X komt op als $a = 0$ is en p = 1 is. X zorgt ervoor dat trekker Y opkomt *bij de overgang van* $a = 0 \rightarrow a = 1$ en niet bijvoorbeeld als a reeds 1 is. Hiermee wordt voorkomen dat smallere pulsen dan de a-puls worden doorgegeven.
Trekker Y zet de uitgangspoort open, zodat de a-puls kan worden doorgegeven.
Trekker Z komt op onder de conditie dat Y = 1 is als a van $1 \rightarrow 0$ gaat. Hiermee wordt gesignaleerd dat er reeds een puls is afgegeven. Z blokkeert alle volgende pulsen.

Het tijddiagram in fig. 5.15 geeft een toelichting. Tijdsverschuivingen ten gevolge van de eigen traagheid van de poorten zijn uit het diagram weggelaten. De cyclus herhaalt zich telkens als p = 0 geweest is.

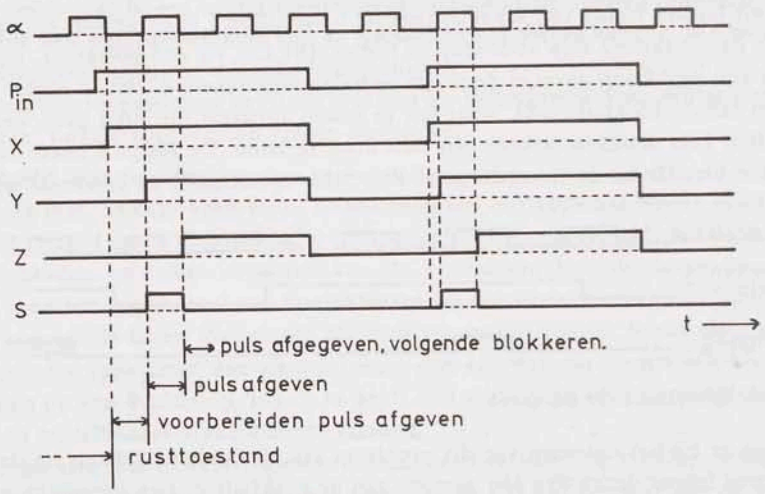


fig. 5.15. Tijddiagram van een pulsgever.

Voorbeeld. *Trekker met blokkeeringang.*
Soms is het nodig op bepaalde tijdstippen informatie vast te leggen en deze enige tijd te bewaren. Denk bijvoorbeeld aan het aangeven van rondetijden en tussentijden bij schaatswedstrijden gedurende T.V.-uitzendingen.
Na enige tijd vastgehouden te zijn loopt de tijdklok op het scherm weer mee met de klok van de tijdwaarneming.
Fig. 5.16 geeft een voorbeeld van een schakeling die één bit informatie kan vasthouden gedurende bepaalde tijd, of de schakeling volgt het ingangssignaal. De gewenste werking wordt ingesteld met een besturingssignaal C.
Deze schakeling staat bekend als de *geklokte trekker* (Am.: *Clocked latch*). In verband met de nog te behandelen flip-flop geheugenelementen is het beter te spreken van *trekker met blokkeeringang* (Am.: *Gated latch*). Flip-flop geheugenelementen reageren essentieel anders op de nog te introduceren klokpuls dan de hier gegeven trekkerschakeling. Tabel 5.3 specificeert de werking van de schakeling in fig. 5.16.

c	a	z	Z	
0	0	0	0	Z = z
0	0	1	1	
0	1	0	0	
0	1	1	1	
1	0	0	0	Z = a
1	0	1	0	
1	1	0	1	
1	1	1	1	

tabel 5.3. Waarheidstabel van de gated latch.

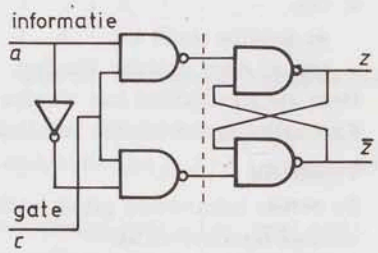


fig. 5.16. Gated latch.

Voorbeeld. *Schakeling met doorschuifwerking.*
Gevraagd wordt een schakeling te ontwerpen die drie bits informatie (a_1, a_2 en a_3) kan onthouden. Als een controlesignaal 1 wordt moet de informatie één plaats opschuiven, terwijl een nieuw bit a_0 in de eerste sectie wordt geplaatst. Is het controlesignaal 0, dan moet de informatie bewaard blijven.
Fig. 5.17 geeft een toelichting.

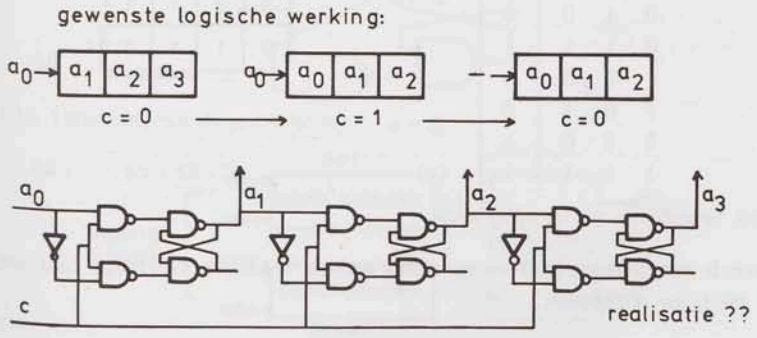


fig. 5.17. Schakeling met doorschuifwerking.

Per sectie voldoet de gated latch aan de specificatie:

C = 0 de schakeling onthoudt.

C = 1 de schakeling neemt de waarde van het aan zijn ingang aangeboden signaal aan de uitgang over.

Toch voldoet de schakeling in fig. 5.17 niet aan de specificatie.

Het wordt aan de lezer overgelaten na te gaan waarom de schakeling niet aan de gegeven specificatie voldoet. Vergelijk daartoe o.a. deze schakeling met die uit fig. 5.14, welke wel correct werkt.

Conclusie.

Voor een aantal problemen is de trekker (latch) een zeer bruikbaar geheugenelement, maar niet onder alle omstandigheden. In de volgende hoofdstukken gaan gaan we uitvoerig op deze materie in.

5.5. Het ontwerpen van schakelingen met geheugenwerking

De gewenste nieuwe stand Z van een geheugenelement hangt in het algemeen af van:

- de huidige stand z
- een of meer externe signalen.

Deze afhankelijkheid kan worden gespecificeerd in een waarheidstabel, een Karnaughdiagram of met een logische formule.

Voorbeeld

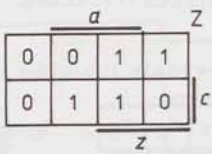
De eerder behandelde gated latch kan voor wat betreft zijn logische functie worden omschreven als:

> De schakeling heeft twee ingangssignalen a en c. Als c = 0 is onthoudt de schakeling de waarde die a had vlak voor c van 1 naar 0 ging.
> Als c = 1 is volgt de uitgang het ingangssignaal a.

Fig. 5.18 beschrijft op de drie verschillende manieren de gewenste logische functie. We zullen nu trachten hieruit tot schakelingen te komen die dit logische gedrag realiseren.

c	a	z	Z	
0	0	0	0	Z = z
0	0	1	1	
0	1	0	0	
0	1	1	1	
1	0	0	0	Z = a
1	0	1	0	
1	1	0	1	
1	1	1	1	

(a)



(b)

$Z = \bar{c}z + ca$ (c)

fig. 5.18. Specificatie van de gated-latch.

Gegeven is een $\bar{S}$-$\bar{R}$ trekker en zo nodig enkele NAND's. Gevraagd hiermee de gated latch te realiseren.

Realisatie via set en reset formules

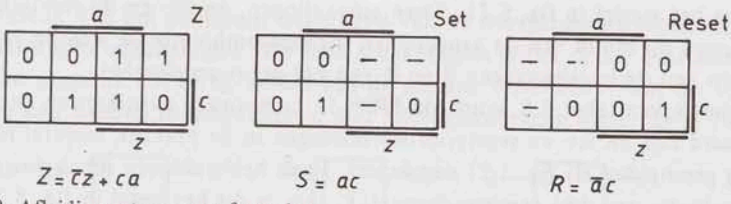


fig. 5.19. Afleiding set en reset formules.

In fig. 5.19 is de afleiding gegeven van de formules voor set en reset. De diagrammen voor set en reset volgen gemakkelijk uit het diagram voor Z met behulp van de tabellen 5.4 en 5.5.

S	R	Z
0	0	z
0	1	0
1	0	1
1	1	–

tabel 5.4. Geheugenwerking.

z	Z	S	R
0	0	0	–
0	1	1	0
1	0	0	1
1	1	–	0

tabel 5.5. Bijbehorende ingangswaarden.

Tabel 5.4 beschrijft hoe Z afhangt van S en R. Tabel 5.5 geeft aan welke set en resetsignalen nodig zijn om een bepaalde combinatie van (z,Z) = (oud,nieuw) te realiseren. Ga deze tabellen na.
Fig. 5.16 geeft de schakeling zoals deze uit de formules voor set en reset volgt. In fig. 5.20 is een variant van de schakeling vermeld, waarin de invertor vermeden wordt.
Deze schakeling berust op het feit dat:

$$c \cdot \bar{a} = c \cdot \bar{a} + c \cdot \bar{c} = c \cdot (\bar{a} + \bar{c}) \ .$$

Immers, in de schakeling is op het punt * het signaal $\bar{a} + \bar{c}$ beschikbaar, terwijl $\bar{a}$ gewenst is voor de onderste ingangspoort. De term $\bar{c}$ valt echter weg via de AND-functie met c.

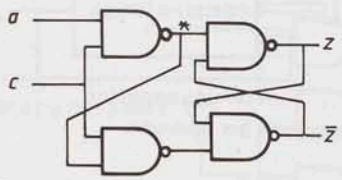


fig. 5.20. Uitvoering van de gated latch.

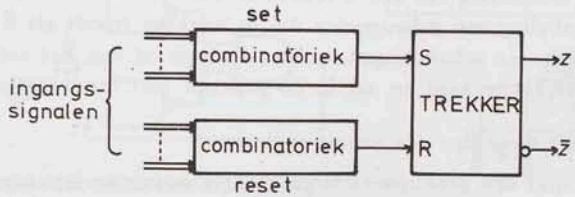


fig. 5.21. Schakeling met geheugenwerking volgens het set-reset model.

In het algemeen kan een schakeling met geheugenwerking gerealiseerd worden volgens het model in fig. 5.21. Twee schakelingen, de set- en de resetschakeling, berekenen op grond van de aangeboden ingangscombinatie de waarde van de signalen aan de instelingangen S en R van het geheugenelement.
Bij gebruik van tabel 5.5 wordt de SR = 11 combinatie automatisch vermeden! Uiteraard zijn de set- en resetpoortschakelingen in de praktijk meestal niet zo streng gescheiden als fig. 5.21 suggereert. Vaak hebben beide schakelingen in een realisatie een deel gemeenschappelijk, bijv. is dit het geval in fig. 5.20.

Realisatie via geheugenlussen

Een geheel andere vorm van de schakeling ontstaat als de formule voor Z direct wordt gerealiseerd in een poortschakeling. Het ingangssignaal z wordt dan via een terugkoppeling van de uitgang teruggevoerd. Soms denkt men in deze terugkoppeling een vertragingselement. Een discussie over de vraag of deze vertraging essentieel is kan worden kortgesloten met de constatering dat deze vertraging van nature altijd in poorten aanwezig is.
Fig. 5.22 geeft een realisatie van de gated latch met behulp van een *geheugenlus*. We onderscheiden twee situaties:

$c = 1 \leftrightarrow$ de schakeling leest in, $Z = a$
$c = 0 \leftrightarrow$ de schakeling onthoudt, $Z = z$

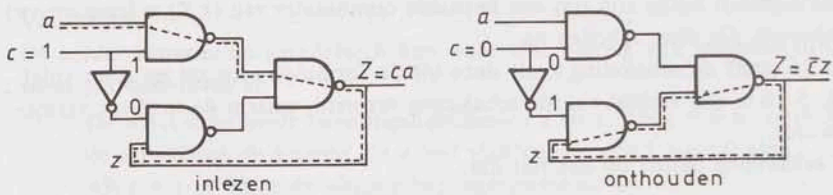


fig. 5.22. Realisatie van de gated latch via geheugenlus.

Het model van een schakeling met geheugenwerking van het type "met geheugenlus" staat in fig. 5.23.

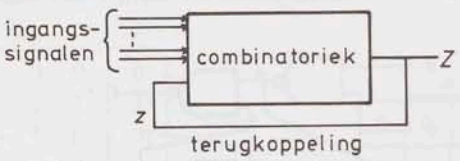


fig. 5.23. Geheugenschakeling volgens het lusmodel.

Opmerking
Bekijken we de schakeling van een $\bar{S}$-$\bar{R}$ trekker met twee NAND's, dan zien we ook in deze schakeling een geheugenlus die in werking treedt als $\bar{S}\bar{R} = 11$ is. Het onderscheid tussen schakelingen van het lustype of van het set-reset type is meer van didactische aard en zal in de praktijk niet duidelijk zijn !

Schakelingen met meer dan één geheugenelement.

In de praktijk blijkt één geheugenelement vaak te weinig te zijn om de gewenste geheugenwerking van de schakeling te realiseren. Voorbeelden hiervan volgen in hoofdstuk 6 en 7. Zie ook de pulsgever uit fig. 5.14.

5.6. Overgangsverschijnselen in schakelingen met geheugenwerking

In hoofdstuk 4 is het probleem onderkehd van de overgangsverschijnselen in combinatorische schakelingen. Bij deze schakelingen is het onder omstandigheden heel goed mogelijk dat er zg. *spikes* aan de uitgang verschijnen. Wat dit voor gevolgen kan hebben in schakelingen met geheugenwerking toont fig. 5.24.

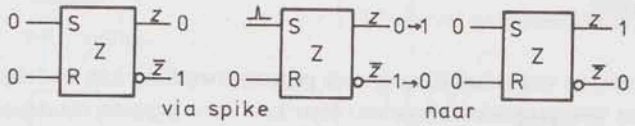


fig. 5.24. Gevolgen van spikes in setsignaal.

Ten gevolge van spikes kunnen ongewenste sets en resets optreden. Schakelingen die zijn opgebouwd uit trekkers en aparte schakelingen voor set en reset kan men over het algemeen vrij gemakkelijk controleren op het optreden van spikes. Dit dient altijd te geschieden!
Een vrij globale wijze van *ontwerpverificatie* vindt reeds plaats als de ontworpen schakeling wordt gebouwd en getest. Echter, een spike welke in een proefschakeling net niet tot ongewenste schakelhandelingen leidt, kan hiertoe wel leiden bij een iets gewijzigde opzet in een definitieve opstelling. De praktijk leert dat schakelingen, welke zijn ontworpen via het set-reset model, meestal correct werken. Een van de noodzakelijke voorzorgen is wel dat de set- en resetschakelingen in een zo gering mogelijk aantal niveaus worden opgebouwd.
Schakelingen van het lustype geven in de praktijk meer problemen. Beschouw de gated latch in fig. 5.22. Als c = 1 is, leest de schakeling in. Wordt c hierna 0, dan wordt de lus in de onderste ingangspoort gesloten via $\bar{c} = 1$ en begint de geheugenwerking. Het signaal $\bar{c}$ voor de onderste poort wordt echter door de invertor enige tijd vertraagd ten opzichte van het signaal c op de bovenste ingangspoort. In principe kan bij een voldoend grote vertragingstijd van de invertor de situatie ontstaan dat de informatie van a in de lus reeds wegvalt voordat de lus gesloten is. Men kan dit overgangsverschijnsel op twee manieren oplossen. De eerste is de *maak-voor-verbreek* methode zoals ook bij kontaktschakelingen is toegepast. Zie fig. 5.25.a.

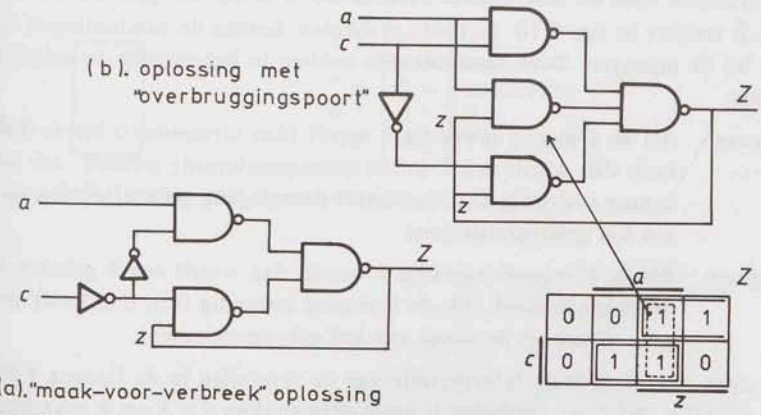


fig. 5.25. Geheugenschakelingen zonder overgangsverschijnselen.

Fig. 5.25.b geeft een andere oplossing van het omschakelprobleem. Deze oplossing berust erop dat

$$Z = ca + \bar{c}z = ca + \bar{c}z + az \ .$$

In de term az komt c niet voor! Deze term blijft dus in voorkomende gevallen 1 als c schakelt.

Conclusie

De juiste werking van schakelingen met geheugenwerking kan nadelig beïnvloed worden door overgangsverschijnselen. Men kan door gepaste maatregelen deze invloed verminderen. Hiervoor is een grondige analyse van de ontworpen schakeling vereist. Er bestaan echter situaties dat er een principieel gevaarlijke situatie kan blijven bestaan. Onder meer treedt dit op als meer dan één ingangssignaal van de schakeling tegelijk of nagenoeg tegelijk verandert.
Schakelingen volgens het set-reset model kan men wat gemakkelijker analyseren dan schakelingen van het lusmodel. Dit leidt tot enige voorkeur voor het eerste model. Vaak echter leidt het lusmodel tot schakelingen met minder poorten, een reden om dit model toch niet uit te sluiten.

In de praktijk vallen de gevolgen van overgangsverschijnselen wel mee, mits voldaan wordt aan:

- De sturende combinatorische schakelingen worden in twee niveaus gerealiseerd (maximaal in drie als de inverse signalen niet beschikbaar zijn).
- De schakelingen worden opgebouwd uit identieke bouwstenen, alleen NAND's bijvoorbeeld. Hierdoor wordt bereikt dat de vertragingstijden van poorten en geheugenlussen onderling geen grote verschillen vertonen.

Het bovenstaande beoogt niet een uitputtende behandeling te zijn van het ontwerpen van schakelingen met geheugenwerking. Slechts enkele mogelijke oorzaken van fouten zijn aangestipt. Hoe men deze kan vermijden door een andere organisatie van schakelingen is onderwerp van enkele volgende hoofdstukken.

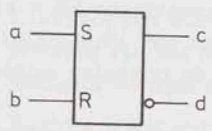
5.7. Symbolen voor trekkers

Het symbool voor de S-R trekker (S-R latch) is in fig. 5.9 getekend, dat voor de $\bar{S}$-$\bar{R}$ trekker in fig. 5.10. In beide symbolen komen de aanduidingen S en R voor bij de ingangen. Deze aanduidingen hebben in het vervolg de volgende betekenis:

S-ingang: Als de S-ingang inwendig 1 wordt (dus uitwendig 0 bij de $\bar{S}$-$\bar{R}$ trekker), dan wordt een 1 in het geheugenelement geladen. Als de S-ingang inwendig 0 is, dan heeft deze ingang geen effect op de stand van het geheugenelement.

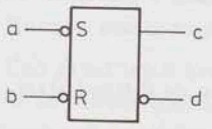
R-ingang: Als de R-ingang inwendig 1 wordt, dan wordt een 0 geladen in het geheugenelement. Als de R-ingang inwendig 0 is, dan heeft deze ingang geen effect op de stand van het geheugenelement.

Met deze afspraken is de interpretatie van de symbolen in de figuren 5.9 en 5.10 niet moeilijk. Bij deze symbolen is reeds afgesproken dat S en R niet tegelijk (inwendig) 1 zijn. Voor SR = 11 zijn de uitgangen *niet gespecificeerd (undefined).*



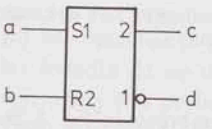
a. S-R Trekker.

a b	S R	c d
0 0	0 0	onthouden
0 1	0 1	0 1
1 0	1 0	1 0
1 1	1 1	niet gespecificeerd



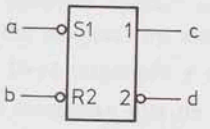
b. $\bar{S}$-$\bar{R}$ Trekker.

a b	S R	c d
1 1	0 0	onthouden
1 0	0 1	0 1
0 1	1 0	1 0
0 0	1 1	niet gespecificeerd



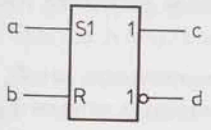
c. S-R Trekker met twee NOR's.

a b	S R	c d
0 0	0 0	onthouden
0 1	0 1	0 1
1 0	1 0	1 0
1 1	1 1	0 0



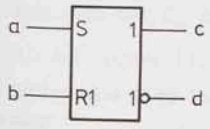
d. S-R Trekker met twee NAND's.

a b	S R	c d
1 1	0 0	onthouden
1 0	0 1	0 1
0 1	1 0	1 0
0 0	1 1	1 1



e. S-R Trekker met overheersende SET.

a b	S R	c d
0 0	0 0	onthouden
0 1	0 1	0 1
1 0	1 0	1 0
1 1	1 1	1 0



f. S-R Trekker met overheersende RESET.

a b	S R	c d
0 0	0 0	onthouden
0 1	0 1	0 1
1 0	1 0	1 0
1 1	1 1	0 1

fig. 5.26.

Zoals reeds besproken is treden problemen bij trekkers niet op omdat SR = 11 is, maar omdat SR = 11 door SR = 00 gevolgd kan worden. Het is voldoende deze overgang uit te sluiten. In dit geval kan de ingangscombinatie SR = 11 worden toegelaten. De uitgangen van een trekker zijn dan niet altijd elkaars inverse. Om dit te kunnen aangeven moet de notatie in de symbolen worden uitgebreid. Zie fig. 5.26.
Bij de uitgangsklemmen is een cijfer aangegeven, dat aangeeft dat de ermee corresponderende ingang overheerst (mits inwendig in de 1-toestand). In het symbool onder 5.26.c overheerst de resetingang R2 de met 2 gemerkte uitgang c. Deze uitgang is dus 0 als:
– de ingangscombinatie S1/R2 = –/1 is
– de ingangscombinatie S1/R2 = 0/0 is en de trekker in de 0-stand staat.
In de overige gevallen heeft de uitgang c de waarde 1.
Voor de met d gemerkte uitgang geldt:
– als S1/R2 = 1/–, dan is de d-uitgang 0 (uitwendig)
– als S1/R2 = 0/0, dan is de d-uitgang 1 (uitwendig) wanneer het geheugenelein de 0-stand staat
– als S1/R2 = 0/1, dan is de d-uitgang 1.

Op geheel overeenkomstige wijze moeten de overige symbolen in fig. 5.26 worden geïnterpreteerd. De gegeven symbolen komen overeen met de voorgestelde nieuwe internationale norm betreffende symbolen voor logische schakelingen. Vergelijk NEN 5152.

Opgaven

5.1. In een bus kan er onenigheid tussen de chauffeur en zijn passagiers ontstaan als zij beide tegelijk hun knop van de stopsignalering indrukken. De schakeling krijgt dan een Set en een Reset tegelijk. Toon aan dat dit meningsverschil niet vermeden wordt als de trekker in de schakeling voor de stopsignalering van het overheersende-set type is.

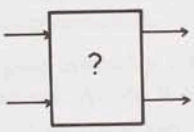
5.2. Bij de analyse van het probleem van de geheugenwerking zijn het Z_0, het Z_1 en het Z_z geheugenelement geïntroduceerd. Waarom is er geen $Z_{\bar{z}}$ geheugenelement geïntroduceerd?

5.3. De formule van het Z_1 geheugenelement is:

$$Z_1 = s + \bar{r}z_1 \ .$$

Toon door middel van formule-manipulaties aan dat deze schakeling kan worden gerealiseerd met twee NAND's.

5.4.



Van een black box is gegeven dat deze een trekker bevat. Ook is reeds bekend welke de ingangsklemmen en de uitgangsklemmen zijn. Van de ingangsklemmen is niet bekend welke S resp. R is. Hetzelfde geldt voor de volgorde van de uitgangsklemmen.

Hoe kan men bepalen welk type trekker (S-R, $\bar{S}$-$\bar{R}$ of Z_z geheugenelement) er in de black box zit?

5.5. Drukknoppen worden uitgerust met een anti-dender schakeling om de gevolgen van het nadenderen tegen te gaan. Zijn met deze schakeling alle denderproblemen de wereld uit?

5.6. Waarom is het ontdenderen van maak- resp. verbreekkontakten moeilijker dan van wisselkontakten? Welke conclusie volgt hieruit voor de keuze van drukknoppen e.d.?

5.7. Kunnen maak- resp. verbreekkontakten ontdenderd worden en zo ja, hoe?

5.8. Een gated latch heeft als functie het ingangssignaal a te volgen (c = 1) of de laatste waarde van a te onthouden (c = 0). Zie fig. 5.20. Op de uitgang van de onderste ingangspoort staat het signaal P:

$$P = (\bar{a} + \bar{c})c = \bar{a}c + \bar{c}c \ .$$

Bij het omschakelen kan er als het ingangssignaal a = 1 is een spike ($c\bar{c}$) ter breedte Δt op de resetpoort van de trekker komen. Hierbij is Δt de vertraging van de bovenste ingangspoort.
Toon aan dat de schakeling in genoemde figuur niet kritisch is.

5.9. Een schakeling bevat twee drukknoppen a en b (wisselkontakten). Deze knoppen kunnen niet tegelijk worden ingedrukt. Ontwerp een schakeling die aangeeft op welke knop gedrukt wordt, of het laatst gedrukt werd. Twee uitgangen y en z geven dit als volgt aan:

yz = 10: Knop a is of werd het laatst gedrukt.
yz = 01: Knop b is of werd het laatst gedrukt.

De schakeling dient zo snel mogelijk de nieuwe situatie aan te geven. Gebruik NAND's als bouwsteen.

5.10. Als de vorige opgave, maar nu met drie ingangssignalen a, b en c. De gewenste uitgangscombinaties zijn: xyz = 100, 010 en 001.
Ga na of uw oplossing een zodanige structuur heeft dat het principe ervan uitbreidbaar is.

5.11. Realiseer het Z_0 geheugenelement met relais en kontakten.

5.12. Als bij opgave 11, maar nu met uitsluitend maakkontakten (reedrelais).

Literatuur

1. R.M.M. Oberman, *Disciplines in Combinational and Sequential Circuit Design*, McGraw-Hill, New York, 1970.

Wegens het inleidende karakter van dit hoofdstuk wordt verwezen naar de literatuur na de hoofdstukken 6 en 7.

6. ANALYSE EN SPECIFICATIE VAN LEVEL MODE SEQUENTIELE SCHAKELINGEN

6.1. De toestandstabel en het toestandsdiagram

Het ontwerpen van digitale schakelingen begint met de *probleemspecificatie.* Bij het opstellen van de logische specificatie worden de verbaal geformuleerde werkingseisen omgezet in een zo systematisch mogelijk opgezette logische formulering. Meestal laat de gegeven formulering van het probleem de ontwerper de nodige vrijheid, niet alle situaties zijn omschreven.
Deze vrijheid dient in de logische specificatie tot uitdrukking te komen. In de realisatiefase wordt de logische specificatie omgezet in een betrouwbaar werkende schakeling. Dit deel van het ontwerpproces omvat niet alleen de componentenkeuze, montage, kostprijsaspecten, e.d. Vaak zal het voorkomen dat de logische specificatie aangepast moet worden aan de mogelijkheden die de beschikbare componenten bieden. Is dit het geval, dan moet de ontwerper weer verifiëren of nog aan de oorspronkelijke specificatie wordt voldaan. De ontwerper van digitale schakelingen dient daarom een goed overzicht te hebben van de mogelijkheden en beperkingen van zijn bouwstenen. Ook een goed inzicht in het probleem waarvoor de te ontwerpen schakeling bestemd is, is noodzakelijk.

De *probleemanalyse* en *-specificatie* is onderwerp van dit hoofdstuk. Voor een probleem van geringe complexiteit, zoals de pulsgever uit het vorige hoofdstuk, kiest men de middelen voor de specificatie van de logische werking anders dan voor de specificatie van schakelingen die uit enige duizenden componenten bestaan. In het laatste geval zullen blokschema's of andere middelen worden gekozen met een opklimmende graad van gedetailleerdheid. Een algemene methode om digitale problemen te specificeren bestaat daarom niet. In de volgende hoofdstukken wordt gebruik gemaakt van het toestandsdiagram en/of de toestandstabel. Dit zijn geschikte hulpmiddelen om schakeltechnische problemen van geringe complexiteit te analyseren en te specificeren. Daarbij zullen een aantal problemen naar voren komen die de werking van digitale schakelingen met geheugenwerking nadelig kunnen beïnvloeden.

In fig. 6.1 is het schema getekend van een schakeling met geheugenwerking. Van de twee binaire ingangssignalen is a een *informatiesignaal* en c een *controlesignaal.* Gegeven is dat a niet verandert als c verandert. De werking van de schakeling is gespecificeerd in waarheidstabel 6.1. In tabel 6.1 stelt z de huidige waarde van het uitgangssignaal van de trekker Z voor en Z de gewenste nieuwe waarde. De tabel volgt uit fig. 6.1.

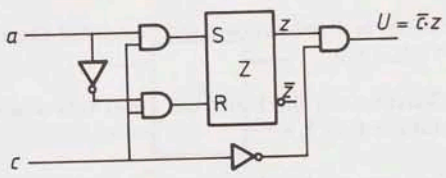


fig. 6.1. Schakeling met geheugenwerking.

	c	a	z	Z	U	
*	0	0	0	0	0	De ingangen van de trekker Z zijn geblokkeerd. Het uitgangssignaal U komt overeen met de stand van de trekker Z: U = z.
*	0	0	1	1	1	
*	0	1	0	0	0	
*	0	1	1	1	1	
	1	0	0	0	0	De uitgang U is geblokkeerd. Het geheugenelement Z volgt de ingang a.
	1	0	1	0	0	
	1	1	0	1	0	
	1	1	1	1	0	

tabel 6.1. Waarheidstabel.

De werking van de schakeling is als volgt te omschrijven:
- c = 1 : De trekker Z neemt de waarde van het informatiesignaal over.
- c → 0 : De koppeling tussen signaal a en de trekker wordt verbroken.
- c = 0 : De trekker is nu ongevoelig voor veranderingen van het ingangssignaal a. De uitgang U heeft de waarde die het ingangssignaal a had vlak voor de 1 → 0 overgang van het controlesignaal c.
- c → 1 : Omschakelen naar inlezen van het informatiesignaal; de uitgangspoort wordt geblokkeerd.

In een schakeling zoals die van fig. 6.1 kunnen drie types signalen onderscheiden worden:
- ingangssignalen : a en c
- inwendige signalen : z
- uitgangssignalen : u.

Het *inwendige signaal* z geeft de toestand aan waarin het geheugen Z van de schakeling verkeert. Uit de met * gemerkte regels van tabel 6.1 blijkt dat de schakeling bij eenzelfde combinatie van de ingangssignalen c en a in verschillende inwendige toestanden kan verkeren. Met het inwendige signaal z wordt de gewenste *geheugenwerking* vastgelegd.

In een waarheidstabel wordt geen onderscheid gemaakt tussen ingangssignalen en inwendige signalen, hoewel de functie van beide zeer verschillend is. In de praktijk wordt daarom met een gewijzigde vorm van de waarheidstabel, de *toestandstabel*, gewerkt waarin ingangssignalen en inwendige signalen gescheiden zijn. Toestandstabel 6.2 bevat dezelfde informatie als tabel 6.1, maar is in het gebruik overzichtelijker.

z \ ca	00	01	10	11	00	01	10	11
z = 0	(0)	(0)	(0)	1	0	0	0	0
z = 1	(1)	(1)	0	(1)	1	1	0	0
	nieuwe waarde Z is oude waarde		Z volgt de ingang a		Uitgangswaarde behorende bij de huidige waarde van Z en de ingangssignalen.			

tabel 6.2. Toestandstabel.

Uit de eerste regel van tabel 6.1 volgt dat als de ingangscombinatie ca = 00 is en z = 0, de nieuwe toestand Z van het geheugenelement Z gelijk is aan de oude,

nl. Z = 0. Er wordt een 0 ingevuld in de kolom onder ca = 00 op de rij achter z = 0. Op overeenkomstige wijze worden de overige regels van tabel 6.1 in tabel 6.2 ingevuld. In tabel 6.2 zijn op deze wijze alle opvolgertoestanden van Z onder de bijbehorende ingangscombinatie ingevuld. Is onder een bepaalde ingangscombinatie de nieuwe waarde Z van het geheugenelement gelijk aan de oude waarde z (0 → 0 of 1 → 1), dan heet de betreffende *inwendige toestand stabiel.* Dit wordt aangegeven met een cirkeltje om deze toestand. Betreft het een wijziging van de inwendige toestand, dan heet deze toestand onder de betreffende ingangscombinatie niet-stabiel. Een voorbeeld is de overgang van de inwendige toestand z = 1 onder ingangscombinatie ca = 10 naar de inwendige toestand z = 0.

In de uitgangskolommen van tabel 6.2 wordt het uitgangssignaal aangegeven dat behoort bij de *huidige* inwendige toestand en ingangscombinatie. In de literatuur ziet men soms dat in de uitgangskolommen het uitgangssignaal wordt vermeld dat bij de nieuwe inwendige toestand behoort. Bij de interpretatie van een gegeven toestandstabel dient men hierop te letten.
Omdat in fig. 6.1 het uitgangssignaal U behalve van de inwendige toestand (gerepresenteerd door de stand van trekker Z) ook afhangt van de ingangssignalen bevat tabel 6.2 vier uitgangskolommen. Vaak wordt het uitgangssignaal uitsluitend bepaald door de momentele inwendige toestand. In die gevallen is er slechts één uitgangskolom in de toestandstabel nodig. Voorbeelden hiervan volgen nog.

Een toestandstabel zoals tabel 6.2 wordt een *gecodeerde toestandstabel* genoemd. Deze vorm sluit direct aan bij de realisatie van de schakeling. Het is ook mogelijk de verschillende inwendige toestanden en de ingangs- en uitgangscombinaties symbolisch aan te duiden. Tabel 6.3 is hiervan een voorbeeld. (Vergelijk deze tabel met tabel 6.2.) Deze laatste vorm van de tabel sluit beter aan bij de specificatie van digitale problemen.

q \ i	i_1	i_2	i_3	i_4	i_1	i_2	i_3	i_4
q_1	((q_1))	((q_1))	((q_1))	q_2	u_1	u_1	u_1	u_1
q_2	((q_2))	((q_2))	q_1	((q_2))	u_2	u_2	u_1	u_1

tabel 6.3. Toestandstabel.

De problemen die kunnen ontstaan bij het coderen van toestandstabellen worden in het volgende hoofdstuk besproken.

Voor bepaalde toepassingen is een *toestandsdiagram* overzichtelijker dan een toestandstabel. In fig. 6.2 is het toestandsdiagram getekend dat hoort bij tabel 6.3. Elke inwendige toestand wordt gerepresenteerd door een cirkel. Een pijl naar zichzelf duidt op een stabiele toestand voor de erbij geschreven ingangscombinatie(s). Een pijl naar een andere cirkel duidt op een overgang naar een andere inwendige toestand. Deze overgang geschiedt als de bijgeschreven ingangscombinatie wordt aangeboden. Eventueel kunnen verschillende toestanden na elkaar worden doorlopen voordat er weer een stabiele toestand optreedt. In een tabel die *het uitwendig gedrag* van een schakeling beschrijft is het gebruikelijk in deze gevallen de eerstvolgende stabiele toestand te noteren.

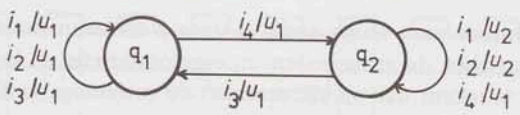


fig. 6.2. Toestandsdiagram.

De uitgangen worden bij de pijlen vermeld als de uitgangscombinatie behalve van de inwendige toestand ook van het aangeboden ingangssignaal afhangt. In het geval dat uitsluitend de inwendige toestand de uitgang bepaalt worden de uitgangswaarden als regel binnen de cirkels genoteerd.
Ook toestandsdiagrammen komen in gecodeerde en niet-gecodeerde vorm voor. Meestal ligt de codering van de in- en uitgangscombinaties vast. Alleen de codering van de diverse inwendige toestanden, waarmee de gewenste geheugenwerking wordt gespecificeerd, moet dan nog worden vastgesteld.

Het gebruik van toestandstabellen/diagrammen

Niet van alle schakelingen kan de werking met toestandstabellen resp. -diagrammen worden beschreven of gespecificeerd. Het gebruik ervan is aan bepaalde restricties gebonden, die echter ook voor praktische schakelingen van toepassing zijn. De belangrijkste zijn:

- Een aan de schakeling aangeboden ingangscombinatie mag pas veranderen als de schakeling een stabiele inwendige toestand heeft bereikt.
- Elke voor een gegeven inwendige toestand toegelaten ingangscombinatie moet tot een inwendige toestand leiden, die stabiel is onder deze ingangscombinatie.
- De gewenste inwendige toestand dient ondubbelzinnig bepaald te worden door de aangeboden ingangscombinatie en de huidige inwendige toestand. Schakelingen met stochastische elementen in welke vorm dan ook zijn taboe.
- Ook eventuele overgangsverschijnselen tussen opeenvolgende ingangscombinaties mogen geen invloed hebben op welke de stabiele toestand is die bereikt wordt.

Op de eerste beperking is een uitzondering bekend. Bij asynchrone telschakelingen (ripple counters) mag een volgende telpuls reeds gegeven worden als een deel van de schakeling nog niet tot rust is gekomen. Voorbeelden hiervan zullen nog worden behandeld.
Het is in principe mogelijk een schakeling te ontwerpen waarvan een gedeelte "genereert" onder bepaalde ingangscombinaties. Is dit gedeelte van de schakeling bij die ingangscombinaties niet mede bepalend voor het uitgangssignaal, dan zal de schakeling uitwendig correct werken. De tweede eis is dus niet absoluut. Het spreekt echter vanzelf dat dit soort situaties vermeden moeten worden. Onder de derde bepaling vallen niet alleen schakelingen waarvan de werking door het toeval bepaald wordt (marginale werking, net nog niet stuk, e.d.). Ook als de correcte werking van een schakeling afhangt van de onderlinge verhoudingen van poortvertragingen e.d. kan dit niet altijd in een toestandstabel worden beschreven. Dit wordt nog nader toegelicht.

Het coderen van de inwendige toestanden

De inwendige toestanden van de schakeling in fig. 6.1 zijn in tabel 6.2 aangeduid met 0 en 1. In de schakeling is deze codering terug te vinden als de stand van de trekker Z. Het begrip *inwendige toestand* duidt eigenlijk een zekere

hoeveelheid informatie aan, welke nodig is om op elk moment de juiste relatie te kunnen leggen tussen de aangeboden ingangscombinatie en het gewenste uitgangssignaal. Dit betekent dat op elk moment de inwendige (informatie)toestand representatief moet zijn voor het verleden van de schakeling voor zover dit verleden voor de momentele en toekomstige werking van belang is. Hoe deze informatie in de schakeling wordt gerepresenteerd kan in principe vrij gekozen worden.
Voor het vastleggen van n inwendige toestanden (coderen) zijn minimaal k geheugenelementen nodig waarbij $2^k \geqslant n$ is. Later zal blijken dat nog andere factoren een rol spelen bij het coderen, waaronder overgangsverschijnselen. Soms is het zelfs niet mogelijk een betrouwbare schakeling te ontwerpen met het minimum aantal geheugenelementen.

6.2. De analyse van schakelingen met geheugenwerking

Toestandstabellen en toestandsdiagrammen zijn nuttige hulpmiddelen om schakelingen van relatief geringe complexiteit te analyseren. Bij de analyse van een bepaalde schakeling is het meestal direct duidelijk welke de ingangs- en uitgangssignalen zijn. Het definiëren van de inwendige toestanden, waarmee de gewenste geheugenwerking wordt beschreven, geeft daarentegen vaak problemen. In veel gevallen is het niet direct duidelijk welke situaties als aparte inwendige toestand onderscheiden moeten worden. Ook kan er verschil bestaan tussen de beschrijving van de *functie* (*uitwendig gedrag*) van de schakeling enerzijds en die van de *logische werking* (*inwendig gedrag*) van de schakeling anderzijds. Met de logische werking wordt bedoeld de wijze waarop de gewenste functie gerealiseerd wordt. Het volgende voorbeeld geeft aan hoe de functionele werking van een schakeling met behulp van inwendige toestanden kan worden beschreven.

Voorbeeld

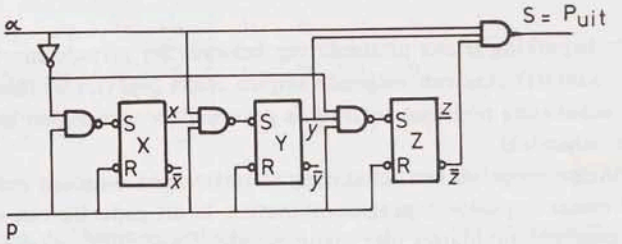


fig. 6.3. Pulsgever.

De in fig. 6.3 gegeven schakeling is gelijk aan de schakeling van fig. 5.14. De werking van deze schakeling is verduidelijkt met het tijddiagram in fig. 5.15. Dit tijddiagram is als fig. 6.4 gereproduceerd.
In het tijddiagram van fig. 6.4 zijn vier situaties te onderscheiden die essentieel zijn voor een beschrijving van de functie van de gegeven schakeling. Deze vier situaties, die elk vastgelegd zijn als een bepaalde combinatie van de standen van de trekkers X, Y en Z zijn:

q_1: (xyz = 000); de rusttoestand van de schakeling.
q_2: (xyz = 100); de schakeling stelt zich in om een puls te gaan afgeven als $\alpha = 1$ wordt.
q_3: (xyz = 110); de schakeling geeft de gewenste puls af.

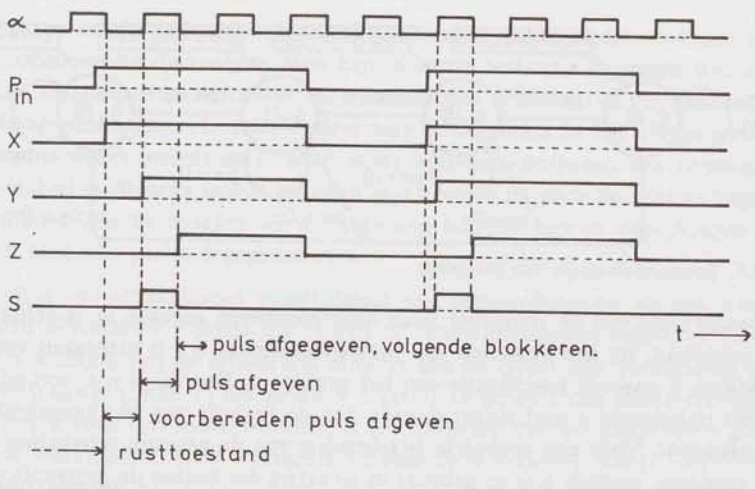


fig. 6.4. Tijddiagram van pulsgever.

q_4: (xyz = 111); er is een puls afgegeven, eventuele volgende klokpulsen moeten worden onderdrukt.

Met behulp van de vier geïntroduceerde inwendige toestanden q_1 tot en met q_4 en de overgangen ertussen kan de functionele werking van de pulsgever worden beschreven. We doen dit in de vorm van een toestandsdiagram. In het op te stellen toestandsdiagram komen twee ingangssignalen p en a voor, en één uitgangssignaal u. De constructie van het toestandsdiagram verloopt als volgt:

q_1: Zolang als p = 0 is blijft de schakeling in deze toestand staan, ongeacht de waarde van het signaal a. Dit is de rusttoestand van de schakeling. Bij de pijl van toestand q_1 naar zichzelf wordt dit aangegeven als pa = 0- (pa = 01 en pa = 00). Wanneer p = 1 wordt komt er een overgang naar de voorbereidingstoestand q_2. Deze overgang vindt plaats als a = 0 is of wordt. Voor de combinatie pa = 11 blijft de schakeling in q_1 wachten.

q_2: Zolang als a = 0 is blijft de schakeling in toestand q_2 staan. Bij het 1 worden van a gaat de schakeling naar toestand q_3 waarin de puls wordt afgegeven. Aangenomen is dat p de waarde 1 behoudt.

q_3: In deze toestand wordt het uitgangssignaal u = 1 afgegeven. Dit geschiedt door het vrijgeven van de uitgangspoort. Het weer 0 worden van a geeft het einde van de puls aan. De schakeling gaat onder pa = 10 van toestand q_3 over in toestand q_4.

q_4: De schakeling moet onthouden dat er een puls is afgegeven. Pas wanneer p = 0 geweest is mag er weer een uitgangspuls gegenereerd worden. De schakeling gaat daarom onder pa = 0- naar toestand q_1; in de overige gevallen, pa = 1-, blijft de schakeling in de stabiele toestand q_4.

Het toestandsdiagram dat met bovenstaande beschrijving overeenkomt staat in fig. 6.5.

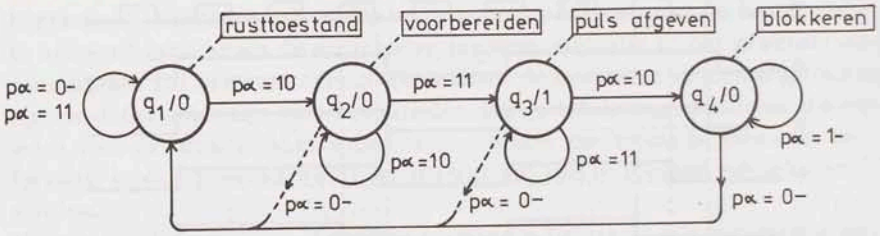


fig. 6.5. Toestandsdiagram van pulsgever.

De beschrijving van de pulsgever zoals deze hierboven gegeven is, is echter nog niet compleet. Bij het opstellen van toestandsdiagram 6.5 is uitgegaan van de in hoofdstuk 5 gegeven specificatie van het probleem. Daarbij is o.a. verondersteld dat het pulssignaal p veel langer duurde dan de periode van het aangeboden kloksignaal α. Voor een complete beschrijving van de gegeven schakeling moet ook nagegaan worden wat er gebeurt in situaties die buiten de gespecificeerde randvoorwaarden vallen. De ontworpen schakeling blijkt critisch te zijn met betrekking tot de duur van het signaal p. Zowel in toestand q_2 als in q_3 gaat de schakeling direct naar q_1 als p weer 0 wordt. In het eerste geval wordt er in het geheel geen puls afgegeven, in het tweede geval wordt de puls afgekapt.

Samenvattend:
Het toestandsdiagram is een geschikt hulpmiddel voor de beschrijving van het uitwendige gedrag van een schakeling. Aan elke voor de uitwendige werking van belang zijnde situatie wordt een inwendige toestand toegekend, aangeduid met $q_1, q_2, \ldots$. Bij iedere inwendige toestand behoort een "label" waarop zijn functie in het geheel wordt vermeld. De overgangen tussen de toestanden worden met pijlen aangegeven. De bijschriften geven aan wanneer een bepaalde overgang plaats vindt, onder welke ingangscombinatie(s), enz. Meestal laat men de labels bij de toestanden weg als uit de tekst blijkt welke situaties ermee bedoeld worden.

De keuze van de inwendige toestanden

Beperken we ons nog even tot de beschrijving van de uitwendige werking van logische schakelingen. Voor de schakeling van fig. 6.3 was het aan de hand van het tijddiagram in fig. 6.4 vrij snel duidelijk welke situaties karakteristiek waren voor de beschrijving van de logische functie.
Deze vier situaties zijn in het geheugen van de schakeling als volgt gecodeerd:

q_1: xyz = 000
q_2: xyz = 100
q_3: xyz = 110
q_4: xyz = 111.

In het algemeen kunnen vier verschillende situaties met twee geheugenelementen worden gecodeerd. Later zal blijken dat er soms dwingende redenen bestaan om van het minimum aantal af te wijken. Voor de beschrijving van de uitwendige werking zijn dit echter details die uit de beschrijving weggelaten kunnen worden.

Bij een willekeurige schakeling staat men voor het probleem te onderkennen *welke situaties* als inwendige toestand vastgelegd moeten worden. In het alge-

meen geven schakelingen met trekkers of relais als geheugenelement niet al te veel problemen bij de analyse. Men kan in eerste instantie beginnen met als inwendige toestanden te definiëren de verschillende combinaties van standen van de geheugenelementen. In feite is deze weg ook gevolgd in het vorige voorbeeld. Vervolgens wordt onderzocht onder welke ingangscombinaties een bepaalde toestand stabiel is. Daarna wordt gekeken wat er gebeurt voor de overige ingangscombinaties. Hierbij moeten we wel rekening houden met de beperkingen welke aan het eind van par. 6.1 gegeven zijn.

De analyse van schakelingen wordt direct veel gecompliceerder als ook omstandigheden beschouwd worden die buiten de randvoorwaarden vallen, of als de correcte werking van de schakeling afhangt van de onderlinge verhouding van poortvertragingen, e.d. In die gevallen waarbij de grootte van poortvertragingen essentieel is voor een juiste werking, is het mogelijk dat de opgestelde toestandstabel de werking niet juist weergeeft als men geen rekening houdt met deze poortvertragingen.

Voorbeeld

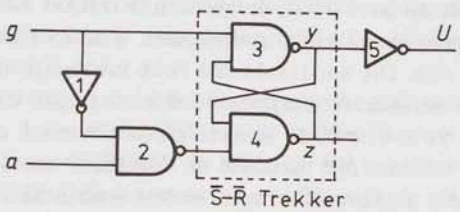


fig. 6.6. Schakeling met geheugenwerking.

De schakeling in fig. 6.6 is opgebouwd rond een $\bar{S}$-$\bar{R}$ trekker, die gevormd wordt door de poorten 3 en 4. Van de twee ingangssignalen g en a is gegeven dat zij niet tegelijk veranderen.
De uitwendige werking is als volgt:

- Als g = 0 is, dan is punt y altijd 1 en heeft U de waarde 0.
- Gaat g van 0 → 1, dan neemt op dat moment de trekker de waarde van a over. Daarna wordt poort 2 geblokkeerd. De trekker onthoudt de waarde die het signaal a had vlak voordat g van 0 → 1 ging. De uitgang U wordt gelijk aan deze waarde van a.
- Zolang als g = 1 is, is de schakeling ongevoelig voor veranderingen van a.
- Gaat g weer naar 0, dan begint het gehele proces weer van voren af aan.

We zullen nu stap voor stap een toestandstabel proberen op te stellen die de logische werking van de schakeling beschrijft.
Dit geschiedt in drie fasen:

- Bepaal de mogelijke stabiele toestanden bij elke ingangscombinatie.
- Ga na wat er gebeurt als er in een stabiele situatie een ingangsverandering optreedt.
- Ga eventueel na wat er gebeurt in omstandigheden die buiten de specificatie vallen.

Fase 1: *Het bepalen van de stabiele toestanden*

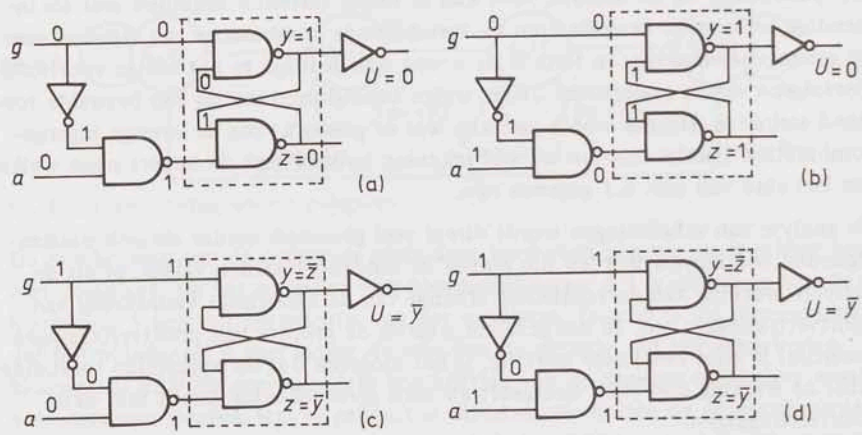


fig. 6.7. Het bepalen van de mogelijke stabiele toestanden.

In fig. 6.7.b blijkt dat als ga = 01 is, de getekende trekker zowel op de SET- als op de RESET-ingang een 0 krijgt aangeboden, waardoor beide uitgangen niet meer elkaars inverse zijn. Dit noodzaakt om twee inwendige signalen (toestandsvariabelen) y en z te definiëren, waarmee de drie mogelijke uitgangscombinaties yz = 10, yz = 11 en yz = 01 van de trekker kunnen worden onderscheiden. Het is hier niet mogelijk om met één variabele de "toestand van de trekker" vast te leggen. Ook als er geen *stabiele situatie* voorkomt waarin SET en RESET beide 1 zijn, kan het noodzakelijk zijn twee inwendige signalen te definiëren per trekker. De genoemde situatie kan nl. ook optreden als overgangsverschijnsel ten gevolge van ingangsveranderingen.
Uit fig. 6.7 volgt de gedeeltelijk ingevulde toestandstabel 6.4, waarin alle mogelijke stabiele inwendige toestanden van de schakeling zijn vastgelegd.

yz \ ga	00	01	10	11	00	01	10	11
00	.	.	.	.	.	.	.	.
01	.	.	(01)	(01)	.	.	1	1
10	(10)	.	(10)	(10)	0	.	0	0
11	.	(11)	.	.	.	0	.	.

tabel 6.4. Overzicht van de stabiele toestanden.

Uit fig. 6.7 volgt dat er geen stabiele situatie voorkomt waarbij yz = 00 is. De inwendige toestand yz = 00 kan slechts als overgangstoestand voorkomen.

Fase 2: *Het bepalen van de toestandsovergangen*

In deze stap veronderstellen we dat aan de randvoorwaarden (o.a. een ingang verandert pas als na de vorige verandering een stabiele toestand is bereikt) wordt voldaan. Stel dat de ingangscombinatie ga = 00 is. Volgens fig. 6.7 leidt dit tot de stabiele toestand yz = 10, ongeacht de initiële toestand. Hetzelfde geldt voor de kolom onder ga = 01 in tabel 6.4. Deze ingangscombinatie leidt altijd tot de

stabiele toestand yz = 11. We vullen daarom tabel 6.4 aan tot tabel 6.5 (eerste twee kolommen):

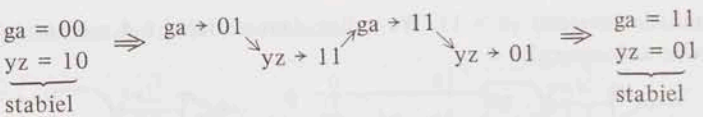
yz \ ga	00	01	10	11	00	01	10	11
00	.	.	.	.	.	.	.	.
01	10	11	(01)	(01)	–	–	1	1
10	(10)	11	(10)	(10)	0	0	0	0
11	10	(11)	.	01	0	0	.	–

tabel 6.5. Toestandstabel.

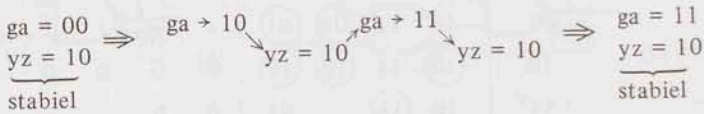
Als de stabiele situatie yz = 10 optreedt onder ga = 00, dan kan òf ingangssignaal g òf ingangssignaal a veranderen. Waartoe de verandering van ga = 00 → ga = 01 leidt is reeds vastgelegd. De verandering ga = 00 → ga = 10 is nog niet onderzocht. Uit fig. 6.7.a volgt dat als g van 0 → 1 gaat dit geen consequenties heeft voor de stand van de trekker.
De opvolgertoestand van de als yz = 10 gecodeerde inwendige toestand bij de gegeven ingangsverandering is weer de toestand yz = 10. Reeds is gevonden dat deze toestand stabiel is onder ga = 10.
Wanneer in de toestand yz = 11 het ingangssignaal g de waarde 1 aanneemt, dan volgt uit fig. 6.7.b dat eerst de SET van de trekker wegvalt, en vervolgens twee poortvertragingen later de RESET (a blijft 1). Dit is voldoende om de trekker te resetten. De toestand die ontstaat is yz = 01, met als uitgangssignaal een 1.
Wanneer het ingangssignaal a verandert in de situatie dat g = 1 is, heeft dit geen invloed op de stand van de trekker. De inwendige toestand blijft in deze gevallen altijd stabiel. (Zie fig. 6.7.d.)
De uitgangskolommen kunnen gemakkelijk worden ingevuld voor de onderzochte overgangen. Wanneer zowel voor als na een overgang y dezelfde waarde heeft, dan houdt ook U zijn waarde tijdens deze overgang. In de overige gevallen wordt een streepje genoteerd, ten teken dat de uitgang verandert gedurende de betreffende toestandsovergang.
In het bovenstaande wordt op geen enkele wijze gebruik gemaakt van de inwendige toestand yz = 00. Ook komt deze toestand niet als overgangssituatie in een of andere toestandsovergang voor. Deze toestand treedt dus in het geheel niet op. De eerste regel van tabel 6.5 kan dus weggelaten worden.

Fase 3: *Het gedrag buiten de gespecificeerde randvoorwaarden*

Als randvoorwaarde bij de gegeven schakeling is gesteld dat de ingangssignalen g en a niet tegelijk veranderen. We moeten nu onderzoeken wat de schakeling doet als zij wel tegelijk veranderen.
Een verandering van g en/of a met als eindwaarde ga = 00 of ga = 01 geeft geen problemen. In beide gevallen is er slechts één stabiele toestand mogelijk.
Anders ligt dit als ga = 00 in ga = 11 verandert. Afhankelijk van de aanwezige poortvertragingen interpreteert de schakeling deze overgang als ga = 00 → 01 → 11 of ga = 00 → 10 → 11 of ga = 00 → 11. Kijken we naar tabel 6.5, dan zou de eindtoestand in het eerste geval zijn:



In het tweede geval treedt op:



Het zal duidelijk zijn dat in dergelijke gevallen de eindtoestand niet altijd dezelfde behoeft te zijn. Voor een gegeven schakeling zal een onderzoek van de bestaande schakeling noodzakelijk zijn om te weten te komen wat er gebeurt. Een simulatie, waarbij rekening gehouden wordt met poortvertragingen, kan ook een goed inzicht geven in wat er zou kunnen gebeuren.

Blijft het gedrag in omstandigheden die niet binnen de gespecificeerde randvoorwaarden vallen buiten beschouwing, dan is tabel 6.5 om te zetten in tabel 6.6. In deze tabel komen drie inwendige toestanden voor: q_1 (01), q_2 (10) en q_3 (11). De uitwendige werking van de schakeling is met behulp van tabel 6.6 gemakkelijk te formuleren.
Uit tabel 6.6 volgt dat als g = 0 is, de schakeling de uitgang 0 heeft. Wordt g = 1 dan wordt de uitgang gelijk aan de laatste waarde van a *voor* de overgang van g van 0 → 1. De functie van de schakeling zou omschreven kunnen worden als een *binaire bemonsterpoort ("sample-and-hold").*

q \ ga	00	01	10	11	00	01	10	11
q_1	q_2	q_3	(q_1)	(q_1)	–	–	1	1
q_2	(q_2)	q_3	(q_2)	(q_2)	0	0	0	0
q_3	q_2	(q_3)	–	q_1	0	0	–	–

tabel 6.6. Toestandstabel.

Wanneer toestandstabel 6.6 gebruikt wordt om de *uitwendige functie* van de schakeling te beschrijven, dan is een eenvoudiger versie van deze tabel mogelijk. In deze tabel kan nl. de functie van toestand q_1 gecombineerd worden met die van toestand q_3. Dit leidt tot tabel 6.7. Hoe en wanneer toestanden kunnen worden gecombineerd is onderwerp van het volgende hoofdstuk.

q \ ga	00	01	10	11	00	01	10	11
$q_{1,3}$	q_2	($q_{1,3}$)	($q_{1,3}$)	($q_{1,3}$)	0	0	1	1
q_2	(q_2)	$q_{1,3}$	(q_2)	(q_2)	0	0	0	0

tabel 6.7. Gereduceerde toestandstabel.

Het is interessant de tabellen 6.2 en 6.3 te vergelijken met tabel 6.7! Hieruit blijkt dat, als het signaal c in fig. 6.1 gelijkgesteld wordt aan $\bar{g}$ in fig. 6.6, beide schakelingen dezelfde functie verrichten.

Conclusie

Hoewel de schakeling in fig. 6.6 niet beschouwd kan worden als een zeer complexe schakeling blijkt de analyse ervan een moeizaam verlopend proces te zijn. Voor schakelingen van een wat grotere omvang is een analyse met de hand weldra ondoenlijk. Dit alles wijst erop dat een "trial-and-error" methode bij het ontwerpen van schakelingen niet alleen tijdrovend, maar tengevolge van overgangsverschijnselen e.d. ook als riskant beschouwd moet worden. Er blijkt dus behoefte te bestaan aan een ontwerpmethode voor "veilige" schakelingen, zodat een analyse achteraf overbodig is.

Een tweede punt is het gebruik van de SR = 11 combinatie bij trekkers. Wordt hiervan gebruik gemaakt (zie bovenstaand voorbeeld), dan zijn er voor elke trekker in de schakeling in principe twee inwendige signalen nodig in een beschrijving. Bij drie trekkers leidt dit tot zes inwendige signalen, die in principe 64 verschillende toestanden kunnen coderen. Wanneer de ingangscombinatie SR = 11 vermeden wordt, dan is er per trekker slechts één inwendig signaal nodig voor het vastleggen van de stand ervan.

De analyse van schakelingen met geheugenlussen blijkt in de praktijk nog meer problemen te geven dan van schakelingen waarin de geheugenwerking via trekkers is gerealiseerd. In feite moet na elke poort in een lus een inwendig signaal gedefinieerd worden. Looptijdeffecten e.d. zorgen er dan nog voor dat toestandsovergangen veelal niet betrouwbaar opgespoord kunnen worden. Dit geeft aanleiding tot een sterke voorkeur voor het gebruik van trekkers als geheugenelement bij de synthese van schakelingen.

Tot slot een voorbeeld van de analyse van een schakeling die is opgebouwd uit twee relais en enkele kontakten. Aangenomen is dat in deze schakeling pas een nieuwe schakelhandeling volgt als de schakeling na de vorige schakelhandeling een stabiele inwendige toestand heeft bereikt.

Voorbeeld

De schakeling in fig. 6.8 staat onder besturing van een wisselkontakt a, dat om redenen van overgangsverschijnselen van het verbreek-voor-maak type is.

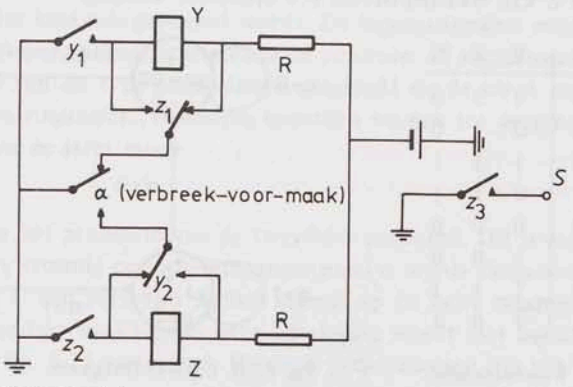


fig. 6.8. Tweedeler met relais.

De schakeling is een zg. tweedeler. Wanneer het a-kontakt met de frequentie f omgelegd wordt, dan opent en sluit het kontakt z_3 met de frequentie $\frac{1}{2}$f. De twee weerstanden R zijn in het schema opgenomen om te zorgen dat de batterij

niet kortgesloten kan worden.
De werking van de schakeling is gegeven in het tijddiagram van fig. 6.9.

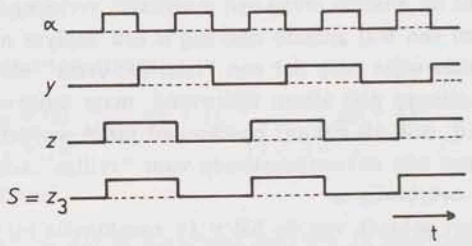


fig. 6.9. Tijddiagram van een tweedeler.

Bij de analyse van de gegeven schakeling via de introductie van inwendige toestanden ligt het voor de hand hiervoor de mogelijke combinaties van standen van de twee relais te kiezen. Het is mogelijk uitgaande van een bestaande combinatie, bijv. yz = 00, stap voor stap een toestandstabel of -diagram op te stellen. Deze aanpak vereist een nauwkeurig nagaan van alle situaties en leidt gemakkelijk tot vergissingen. Een systematische aanpak is in deze gevallen mogelijk:
Als eerste stap worden de bekrachtigingsformules voor de relais Y en Z opgesteld. In het schema vallen de relais door kortsluiting af, hetgeen correspondeert met de formules

$$Y = \overline{R}_y(S_y + y) \quad \text{en} \quad Z = \overline{R}_z(S_z + z).$$

Uit het schema volgt:

$$\begin{aligned} Y &= \bar{a}\bar{z}(\bar{a}z + y) \\ &= (a + z)(\bar{a}z + y) \\ &= \bar{a}z + ay + yz \end{aligned} \qquad \begin{aligned} Z &= \overline{ay}(a\bar{y} + z) \\ &= (\bar{a} + \bar{y})(a\bar{y} + z) \\ &= \bar{a}z + a\bar{y} + \bar{y}z. \end{aligned}$$

In deze formules corresponderen de variabelen y en z, welke bij de kontakten behoren, met de bekrachtigingstoestand van de relais. Met behulp van de formules voor Y en Z kan waarheidstabel 6.8 opgesteld worden.

y	z	a	Y	Z	S
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	1	1
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	0	1

tabel 6.8. Waarheidstabel.

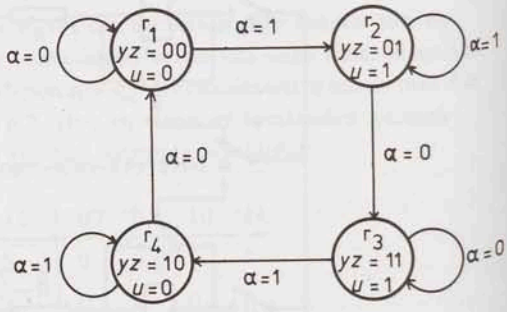


fig. 6.10. Toestandsdiagram.

Uit deze tabel wordt het toestandsdiagram van fig. 6.10 geconstrueerd. In dit diagram is de tweedelerwerking gemakkelijk te herkennen.

Een vraag, waaraan in het bovenstaande stilzwijgend voorbijgegaan is, is of de schakeling de opgestelde formules voor Y en Z daadwerkelijk realiseert. Voor

de getekende schakeling is dit inderdaad het geval. In het algemeen kunnen overgangsverschijnselen, zoals nadenderen, roet in het eten gooien. Ook voor relaisschakelingen geldt dat de analyse ervan meer moeilijkheden oplevert dan men zou verwachten. In dit verband moet ook gewezen worden op het gebruik van schema's van logische schakelingen. *Als de werking van een schakeling kritisch afhangt van of berust op vertragingen, dan moet men deze ook in de schema's aangeven. Meestal is men hier erg slordig mee, zodat vele schema's in catalogi dan ook nauwelijks of in het geheel niet te begrijpen zijn.*

6.3. De specificatie van problemen met geheugenwerking

Hoewel in paragraaf 6.2 gebleken is dat de analyse van schakelingen met behulp van toestandstabellen resp. -diagrammen aan vrij sterke beperkingen is gebonden, mag hieruit niet de conclusie worden getrokken dat deze middelen ook beperkt bruikbaar zijn bij de specificatie van logische schakelingen. Bij de specificatie van schakelingen kunnen toestandstabellen en -diagrammen goede diensten bewijzen. Met behulp van enkele voorbeelden zullen de mogelijkheden worden onderzocht.

In onderstaande opzet wordt een digitale schakeling beschouwd als een black box met binaire ingangssignalen en binaire uitgangssignalen. De gewenste geheugenwerking wordt gespecificeerd met behulp van inwendige toestanden. Als ingangscombinatie op een bepaald moment wordt beschouwd de combinatie van waarden van alle binaire ingangssignalen gezamenlijk. Zodra het logisch niveau van een of meer der ingangssignalen verandert, ontstaat er dus een nieuwe ingangscombinatie waarvoor de gewenste werking weer moet worden gespecificeerd. Deze wijze van specificeren van schakelingen wordt *level mode specificatie* genoemd, ook wel aangeduid als *asynchrone specificatie* van het probleem. Bij elke mogelijke ingangsverandering moet de schakeling zich direct aanpassen aan de nieuwe situatie.
Later zal blijken dat er redenen kunnen bestaan om een schakeling niet direct op ingangsveranderingen te laten reageren, maar dit pas te doen op bepaalde (vaste) tijdstippen. Deze tijdstippen worden vastgelegd door een (meestal) periodiek signaal, dat *klokpuls* genoemd wordt. De ingangssignalen worden dan onderscheiden in *informatiesignalen* enerzijds en *controle-* of *instelsignalen* anderzijds. De specificatie van dit type schakelingen geschiedt via de *clock mode* of *synchrone beschouwingswijze*. Voorlopig beperken we ons tot de specificatie van schakelingen via de level mode.

Voorbeeld
Reeds eerder is het probleem van de tweedeler genoemd. Dit is een schakeling, waaraan een pulsvormig periodiek ingangssignaal α wordt aangeboden, en waarvan de uitgang U een periodiek signaal afgeeft op de halve ingangsfrequentie. De gewenste uitwendige werking van deze schakeling wordt met behulp van het tijddiagram in fig. 6.11 vastgelegd. Hierin is gespecificeerd dat het uitgangssig-

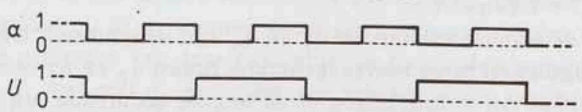


fig. 6.11. Tijddiagram van een tweedeler.

naal U uitsluitend op of direct na (i.v.m. vertragingen in poorten) de neergaande flank van het ingangssignaal a verandert.

De specificatie via het tijddiagram is niet geschikt om tot een schakeling te komen. We zullen dit diagram daarom omzetten in een toestandstabel. In het diagram kunnen vier verschillende situaties onderscheiden worden. Voor elk van deze situaties introduceren we in principe een aparte inwendige toestand, welke de gewenste geheugenwerking specificeert.

De eerste stabiele situatie die we tegenkomen is de toestand q_1 die stabiel is zolang als $a = 0$ is. Zie fig. 6.12. De bijbehorende uitgang is U = 0. Deze situatie is in tabel 6.9 vastgelegd als toestand q_1. De cirkel om q_1 onder $a = 0$ geeft aan dat zolang als $a = 0$ blijft deze toestand stabiel is.

q \ a	0	1	0	1
q_1	(q_1)	·	0	·

tabel 6.9. Specificatie van toestand q_1.

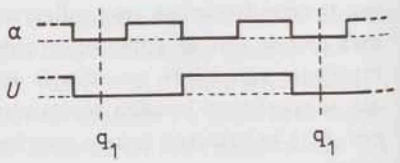


fig. 6.12. Specificatie van toestand q_1.

De ingangswaarde $a = 0$ wordt volgens fig. 6.11 in toestand q_1 gevolgd door $a = 1$. Deze verandering leidt tot een nieuwe stabiele situatie met $a = 1$ en U = 0. Deze wordt in tabel 6.10 en fig. 6.13 vastgelegd als de inwendige toestand q_2. Tevens wordt in tabel 6.10 de overgang van toestand q_1 naar toestand q_2 gespecificeerd.

q \ a	$0 \xrightarrow{*} 1$	0	1
q_1	$(q_1)\ q_2\ ^{*}_{**}$	0	0
q_2	· (q_2)	·	0

tabel 6.10. Specificatie van toestand q_2.

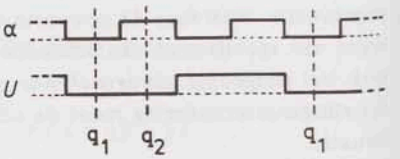


fig. 6.13. Specificatie van toestand q_2.

In tabel 6.10 kan men de oorzaak ($\xrightarrow{*}$) van de *toestandsovergang* of *transitie* aflezen, alsmede het gevolg ($\xrightarrow{**}$). Bij zowel toestand q_1 als toestand q_2 behoort het uitgangssignaal U = 0. Het is daarom zinvol om ook bij de overgangstoestand q_2 op de eerste regel de uitgangswaarde U = 0 te specificeren. Doen we dit niet, dan kan later in een realisatie van de schakeling de mogelijkheid bestaan dat tijdens de overgang van toestand q_1 naar toestand q_2 de uitgang even de waarde U = 1 aanneemt. Dit geeft aanleiding tot een kort piekje in het uitgangssignaal (spike).

Als a in toestand q_2 weer naar 0 gaat, dan moet volgens de specificatie in fig. 6.11 de uitgang U naar 1 gaan. De toestand q_2 wordt onder $a = 0$ dus gevolgd door een nieuwe toestand q_3, die stabiel is zolang $a = 0$ blijft en waarbij het uitgangssignaal U = 1 behoort.
De uitgangen in de stabiele toestanden q_2 en q_3 zijn verschillend. Om deze reden wordt het uitgangssignaal in de overgangssituatie tussen q_2 en q_3 niet gespecificeerd. Het is meestal van weinig belang of de uitgang gedurende een (zeer kort durende) overgangssituatie gelijk is aan die van de oude stabiele toestand of reeds

die van de nieuwe stabiele toestand gaat aannemen.
We zijn nu gekomen tot tabel 6.11 en fig. 6.14.

q \ a	0	1	0	1
q_1	(1)	q_2	0	0
q_2	q_3	(2)	-	0
q_3	(3)	.	1	.

tabel 6.11. Specificatie van toestand q_3.

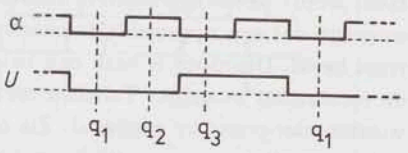


fig. 6.14. Specificatie van toestand q_3.

Als in toestand q_3, die stabiel is onder $a = 0$, het ingangssignaal weer naar 1 gaat, dan moet de uitgang U = 1 blijven. Deze nieuwe situatie leidt tot de introductie van een stabiele toestand q_4 met als uitgang U = 1 onder ingang $a = 1$. Wordt in toestand q_4 de ingang a weer 0, dan komt de schakeling in een situatie terecht die identiek is aan de als toestand q_1 omschreven situatie. Het ligt dus voor de hand toestand q_1 als opvolgertoestand van toestand q_4 onder $a = 0$ te kiezen. Het uitgangssignaal wordt in de overgangssituatie niet gespecificeerd vanwege boven vermelde redenen. Hiermede is het probleem van de tweedeler in een toestandstabel vastgelegd. Zie ook tabel 6.12 en fig. 6.15.

q \ a	0	1	0	1
q_1	(1)	q_2	0	0
q_2	q_3	(2)	-	0
q_3	(3)	q_4	1	1
q_4	q_1	(4)	-	1

tabel 6.12. Toestandstabel van de tweedeler.

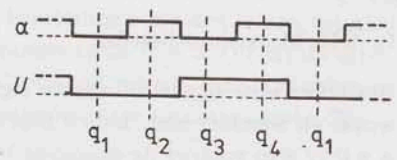


fig. 6.15. Tijddiagram van een tweedeler met inwendige toestanden.

In het algemeen is de toestandstabel, die bij de specificatie van een bepaald schakeltechnisch probleem ontstaat, niet uniek. Verschillende tabellen kunnen hetzelfde probleem beschrijven. Hierboven is het probleem van de tweedeler met vier verschillende inwendige toestanden gespecificeerd. De eerste vraag die men zich stelt is, of dit probleem niet met minder toestanden kan worden beschreven. Met andere woorden, is het mogelijk deze beschrijvingsvorm te comprimeren. Elke inwendige toestand stelt een zekere hoeveelheid informatie voor, aan de hand waarvan de schakeling op dat moment kan beslissen wat er dient te gebeuren bij de eerstvolgende ingangsverandering. Het aantal benodigde geheugenelementen om alle toestanden te coderen wordt rechtstreeks bepaald door het aantal toestanden. Het kan dus zinvol zijn een toestandstabel te minimaliseren. Het blijkt niet mogelijk te zijn in tabel 6.12 de functie van een of meer inwendige toestanden in één nieuwe toestand te combineren. Immers:

- De toestanden q_1 of q_2 kunnen nooit gecombineerd worden met q_3 of q_4 omdat de uitgangen van deze twee groepen toestanden verschillend zijn.
- Ook kan q_1 niet met q_2 gecombineerd worden in één nieuwe toestand omdat als $a = 0$ is dit paar toestanden overgaat in het paar (q_1,q_3). Voor dit paar bestaat er een uitgangsconflict.

– Een soortgelijke reden geldt ook voor het paar (q_3,q_4).
Toestandstabel 6.12 is daarom een *minimale toestandstabel.*
Elke regel van tabel 6.12 heeft één stabiele toestand. Een dergelijke toestandstabel wordt wel een *primitieve toestandstabel* genoemd. Het is echter heel goed mogelijk dat een opgestelde toestandstabel meer dan één stabiele toestand per regel bevat. Daardoor is vaak een belangrijke reductie van het aantal verschillende toestanden mogelijk. Tabellen met meer dan één stabiele toestand per regel worden *niet-primitief* genoemd. Zie ook de tabellen 6.6 en 6.7. In de praktijk is het opstellen van een primitieve toestandstabel vaak eenvoudiger dan het opstellen van een niet-primitieve tabel. Omdat er systematische methoden bestaan om toestandstabellen te reduceren zullen we ons meestal beperken tot het opstellen van een primitieve toestandstabel, om deze daarna eventueel te reduceren.

Voorbeeld. *De bemonsterpoort (sample-and-hold gate)*
Het komt in vele schakelingen voor dat een gedeelte van de schakeling een stap mag doen mits aan bepaalde voorwaarden voldaan is. Dit wordt dan vaak bereikt door een klokpulssignaal wel of niet te blokkeren. Dit klokpulssignaal bestaat uit een periodieke rij pulsen. Een mogelijke oplossing voor het probleem is geschetst in fig. 6.16.

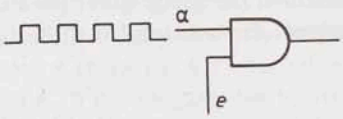


fig. 6.16. Blokkeringsschakeling.

Het voorwaardesignaal of enable signaal e beslist wanneer de rij a doorgelaten wordt en wanneer niet. Indien gegeven is dat het signaal e alleen verandert als $a = 0$ is, dan voldoet de oplossing in fig. 6.16 zonder meer. Het kan echter gebeuren dat het signaal enable op willekeurige tijdstippen verandert. In zo'n situatie kunnen ingekorte pulsen verschijnen aan de uitgang van de AND-poort. Wordt zo'n puls te smal, dan zal de achterliggende schakeling er niet meer op reageren of, wat erger is, slechts een deel van de schakeling reageert. De oplossing in fig. 6.16 voldoet dan duidelijk niet. We zullen dit probleem fundamenteel aanpakken via het opstellen van een toestandstabel.

Gespecificeerd moet worden het volgende probleem:

> Aan een schakeling worden twee binaire ingangssignalen aangeboden, waarvan a een periodiek signaal is en e een signaal dat aangeeft of de volgende a-puls moet worden doorgelaten. Als e = 1 is op het moment van de opgaande flank van a, dan moet de gehele puls van a worden doorgelaten, ook al wordt e weer 0 terwijl a nog 1 is.

Bij het oplossen van dit probleem maken we geen gebruik van het feit dat a periodiek is. De in wezen gewenste werking is de volgende:

> Als a van $0 \rightarrow 1$ gaat moet gekeken worden of e = 1 is. Zo ja, dan moet de schakeling een puls afgeven die net zo lang duurt als één periode van a. Zo nee, dan moet de uitgang 0 blijven. Dit is een *sample-and-hold* probleem.

Het probleem is zo algemeen mogelijk geformuleerd in fig. 6.17. In deze figuur is a het bemonsterende signaal (verg. enable) en g het informatiesignaal dat

wel of niet doorgelaten moet worden afhankelijk van de waarde van a bij de opgaande flank van g. Het tijddiagram legt de gewenste werking vast.

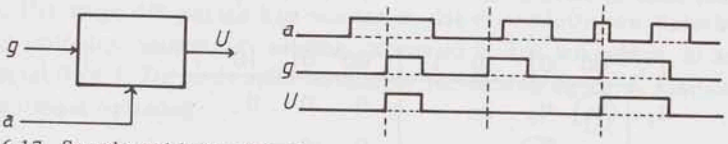


fig. 6.17. Sample and hold schakeling.

Opmerking

Bij deze omschrijving van het probleem is het niet uitgesloten dat a en g tegelijk veranderen. In deze gevallen is een ondubbelzinnige uitslag van de bemonstering niet mogelijk. We stellen een beslissing hierover uit en onderzoeken eerst het geval dat a en g niet tegelijk veranderen, althans niet op het bemonstertijdstip.

Bij het opstellen van de specificatie in een toestandstabel moet worden uitgegaan van een *bestaanbare stabiele situatie*, bijvoorbeeld de situatie dat zowel a als g de waarde 0 hebben. Ook de uitgang U heeft dan de waarde 0. Gezien de omschrijving hierboven kan deze situatie als begintoestand q_1 worden beschouwd. (In principe is elke bestaanbare stabiele toestand geschikt om mee te beginnen.) In toestand q_1 kunnen twee veranderingen in de ingangscombinatie optreden, ga van 00 → 01 of ga van 00 → 10. (De overgang van ga = 00 naar ga = 11 wordt even buiten beschouwing gelaten.) Beide gebeurtenissen moeten leiden tot nieuwe inwendige toestanden q_2 en q_3 met als uitgangswaarde U = 0. Omdat de uitgang van zowel de begintoestand als de eindtoestand van elk van deze overgangen gelijk is aan 0 krijgen ook de tussentoestanden deze uitgangswaarde. Zie tabel 6.13.

q \ ga	00	01	10	11	00	01	10	11
q_1	(q1)	q_2	q_3	.	0	0	0	.
q_2	.	(q2)	.	.	.	0	.	.
q_3	.	.	(q3)	.	.	.	0	.

tabel 6.13. Initiële toestandstabel.

De opvolgertoestand van de toestanden q_2 en q_3 onder de ingangscombinatie ga = 00 kan in beide gevallen q_1 zijn. De opvolger van toestand q_2 bij ingangscombinatie ga = 10 is voorlopig te beschouwen als don't care conditie. Dit geldt ook voor de opvolger van q_3 onder ga = 01. Beide gevallen vallen onder de beperking dat g en a niet tegelijk veranderen.
De opvolgertoestand van q_2 onder ingangscombinatie ga = 11 moet een stabiele toestand zijn met uitgang U = 1. In deze overgang is g van 0 → 1 gegaan terwijl a de waarde 1 had. De nieuwe toestand is q_4 met uitgang U = 1. Om reeds eerder vermelde redenen wordt de uitgang tijdens de overgang van toestand q_2 naar toestand q_4 niet gespecificeerd. Ook voor toestand q_3 moet een opvolger gedefinieerd worden onder ga = 11. In dit geval betreft het een overgang van het ingangssignaal a van 0 → 1 terwijl g reeds 1 was. De uitgang moet de waarde

0 behouden. Er moet een nieuwe toestand q_5 met uitgang U = 0 geïntroduceerd worden.
Het resultaat staat in tabel 6.14.

q \ ga	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	.	0	0	0	.
q_2	q_1	(2)	.	q_4	0	0	.	–
q_3	q_1	.	(3)	q_5	0	.	0	0
q_4	.	.	.	(4)	.	.	.	1
q_5	.	.	.	(5)	.	.	.	0

tabel 6.14. Initiële toestandstabel.

In de toestanden q_4 en q_5 blijft de ingangscombinatie ga = 00 buiten beschouwing. Bij de ingangscombinatie ga = 01 (het poortsignaal g is naar 0 gegaan) moet de uitgang altijd de waarde 0 aannemen. Zo'n toestand is reeds als q_2 geïntroduceerd. Verandert in een van de toestanden q_4 of q_5 het ingangssignaal a (ga = 11 → ga = 10) dan mag de uitgang van de schakeling niet veranderen. Als opvolger van toestand q_4 onder ga = 10 moet dus een nieuwe toestand q_6 geïntroduceerd worden met 1 als uitgangswaarde. Voor q_5 kan als opvolger onder ga = 10 gekozen worden de bestaande toestand q_3. Wij zijn nu gekomen tot tabel 6.15. Het zal de lezer niet veel moeite kosten de laatste regel van tabel 6.15 aan te vullen.

q \ ga	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	.	0	0	0	.
q_2	q_1	(2)	.	q_4	0	0	.	–
q_3	q_1	.	(3)	q_5	0	.	0	0
q_4	.	q_2	q_6	(4)	.	–	1	1
q_5	.	q_2	q_3	(5)	.	0	0	0
q_6	q_1	.	(6)	q_4	–	.	1	1

tabel 6.15. Initiële toestandstabel.

Het probleem van de digitale sample-and-hold schakeling is nu gedeeltelijk in een toestandstabel vastgelegd. Alleen over die situaties waarin g en a tegelijk veranderen moet nog een beslissing genomen worden.
De eerste twee kolommen van tabel 6.15 geven weinig problemen. In alle gevallen moet de schakeling naar een stabiele toestand met uitgangswaarde 0. Dit is de uitgang die behoort bij de ingangswaarde g = 0. Stabiele toestanden bestaan reeds in deze kolommen, nl. q_1 en q_2. De eerste twee kolommen van tabel 6.15 worden aldus aangevuld tot die van tabel 6.16. Dat de eerste twee kolommen zo gemakkelijk ingevuld kunnen worden is te danken aan het feit dat, hoewel beide ingangssignalen veranderen, de gewenste *eindwaarde* van de uitgang geheel vastligt omdat g naar 0 gaat.

Moeilijker is de opvolging van toestand q_1 onder ga = 11. Dit betreft een situatie dat het voorwaardesignaal a bemonsterd wordt (g van 0 → 1), terwijl dat signaal zelf verandert. In zo'n situatie ligt de uitslag van een bemonstering niet vast. Het enige dat gezegd kan worden is, dat er tenslotte een stabiele toestand moet optreden waarvan de uitgang onbepaald is. Dat wil zeggen, in het beschouwde geval 0 of 1. De reeds geïntroduceerde toestanden q_4 en q_5 kunnen dus beide als opvolger optreden.

Ook de opvolger van toestand q_2 onder ga = 10 moet nog bepaald worden. Hiervoor geldt hetzelfde als voor de opvolger van toestand q_1 in de vierde kolom: de uitslag van de bemonstering is onzeker. Naar keuze kan dan één van de toestanden q_3 of q_6 als opvolger worden aangewezen.
Hiermee is het probleem van de sample-and-hold schakeling in een toestandstabel vastgelegd. Tabel 6.16 is de primitieve toestandstabel voor dit probleem.

q \ ga	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	$q_{4/5}$	0	0	0	-/0
q_2	q_1	(2)	$q_{3/6}$	q_4	0	0	0/-	-
q_3	q_1	q_2	(3)	q_5	0	0	0	0
q_4	q_1	q_2	q_6	(4)	-	-	1	1
q_5	q_1	q_2	q_3	(5)	0	0	0	0
q_6	q_1	q_2	(6)	q_4	-	-	1	1

tabel 6.16. Primitieve toestandstabel van de "sample-and-hold" schakeling.

6.4. Een nadere beschouwing van don't care condities

Bij het ontwerpen van combinatorische schakelingen zijn we reeds voorbeelden tegengekomen waarbij een of meer der ingangscombinaties uitgesloten zijn. Deze beperking wordt dan veroorzaakt door andere schakelingen, waarvan uitgangssignalen als ingangen dienen voor de te ontwerpen schakeling.
Dergelijke ingangssignalen of combinaties ervan worden wel omschreven als *don't happen*. Het zal duidelijk zijn dat de werking van de schakeling voor deze ingangscombinaties geheel vrij te kiezen is, de schakeling komt niet in een van deze situaties terecht.
Anders ligt het voor ingangscombinaties die niet gebruikt mogen worden. Deze worden wel aangeduid als *don't use*. In principe mag verondersteld worden dat deze ingangscombinaties niet zullen worden aangeboden. Indien dit echter toch mogelijk is, bijvoorbeeld door bedieningsfouten, storingen, enz., dan moet de schakeling voor deze ingangscombinaties wel worden gespecificeerd. Alternatieven zijn het geven van een alarm (storing) of de handeling vergrendelen (bedieningsfout), enz.
In de hierboven geschetste gevallen is er sprake van niet voorkomende ingangscombinaties. Toch vergen deze bij de probleemspecificatie een verschillende interpretatie. Ook bij het specificeren van sequentiële schakelingen komen don't care condities voor. Als eerste voorbeeld het specificeren van de uitgang van een schakeling gedurende de overgangstoestanden.

Voorbeeld
Wanneer twee toestanden, waartussen een overgang plaats vindt, beide hetzelfde uitgangssignaal bezitten, dan zal als regel de uitgang tijdens de overgang deze waarde moeten behouden. Deze strategie is bij de eerder gegeven voorbeelden gevolgd. Er moet echter worden opgemerkt dat, hoewel de uitgang gedurende de overgang wordt gespecificeerd, het tengevolge van overgangsverschijnselen in de combinatorische schakeling die de uitgang realiseert heel wel mogelijk is dat er spikes optreden. Vergelijk par. 4.8.

Voorbeeld
Twee toestanden, waartussen de overgang plaats vindt, hebben verschillende uitgangswaarden.
Deze situatie is reeds ontmoet bij het opstellen van tabellen voor de tweedeler en de sample-and-hold schakeling. Het doet er in deze gevallen niet toe of de uitgang tijdens de als regel kort durende overgang nog even de uitgang van de oude toestand vasthoudt, of reeds die van de nieuwe gaat aannemen. Om deze reden is in de gegeven voorbeelden een don't care ingevuld.
Geheel anders wordt de situatie als de te ontwerpen schakeling meer dan twee uitgangscombinaties moet kunnen afgeven. Voor de uitgang gedurende de overgangstoestand kan slechts uit enkele van de mogelijke uitgangscombinaties worden gekozen. We zullen dit in het vervolg een *conditionele don't care conditie* noemen. Vergelijk bijv.: $u_1u_2u_3 = 011 \rightarrow u_1u_2u_3 = 000$ mag via $u_1u_2u_3 = 010$ maar niet via $u_1u_2u_3 = 001$ lopen.

Het zal later blijken dat de aandacht die bij level mode gespecificeerde schakelingen geschonken moet worden aan de uitgangsspecificatie, bij synchrone of klokpulsgestuurde schakelingen overbodig is. Bij level mode schakelingen is deze uitgebreide aandacht wel nodig, omdat de schakeling wordt geacht direct te (kunnen) reageren op elke ingangsverandering. Dus ook op overgangsverschijnselen!

Een ander type don't care conditie trad op bij de specificatie van de sample-and-hold schakeling. Hierbij kon de opvolgertoestand van toestand q_1 (rust) onder de ingangsverandering ga = 00 → ga = 11 niet ondubbelzinnig worden gespecificeerd. De reden was dat een veranderend signaal niet ondubbelzinnig kan worden bemonsterd. Toch is de opvolger van toestand q_1 onder ingangscombinatie ga = 11 geen don't care conditie, hoewel een keuze mogelijk is. Ook hier is weer sprake van een *conditionele don't care conditie.* (Vergelijk de tabellen 6.15 en 6.16) Hoe nu te handelen in deze situaties?

De eis die in tabel 6.16 aan de opvolger van q_1 onder ingang ga = 11 gesteld moet worden is, dat het een stabiele toestand is met uitgang 0 of 1. Er bestaan in deze tabel reeds toestanden die hieraan voldoen en wel q_4 en q_5. Hetzelfde geldt voor de opvolger van toestand q_2 onder ingangscombinatie ga = 10. Ook hiervoor bestaan twee alternatieven. In het algemeen wordt een opgestelde specificatie eerst gereduceerd voordat aan de realisatie van de schakeling wordt begonnen. Het is dan gebruikelijk bij de reductie van de tabel de conditionele don't cares voorlopig als vrije don't cares te beschouwen. Na de reductie van de tabel wordt dan onderzocht of het resultaat niet in strijd is met de gestelde voorwaarden. Meestal is dit niet het geval. Mocht dit wel zo zijn, dan moet elk

van de alternatieven worden onderzocht. Voor tabel 6.16 moeten dan vier mogelijke tabellen in een reductieproces worden onderzocht (ga dit na).

Tabel 6.17 is een gereduceerde versie van tabel 6.16. In deze tabel is de functie van de toestanden q_1, q_3 en q_5 gecombineerd in toestand r_1 en de functie van de toestanden q_2, q_4 en q_6 in toestand r_2.

	r \ ga	00	01	10	11	00	01	10	11
$\{q_1,q_3,q_5\}\rightarrow$	r_1	(r_1)	r_2	(r_1)	(r_1)	0	0	0	0
$\{q_2,q_4,q_6\}\rightarrow$	r_2	r_1	(r_2)	(r_2)	(r_2)	0	0	1	1

tabel 6.17. Gereduceerde toestandstabel.

Hoe deze tabel uit tabel 6.16 kan worden afgeleid wordt besproken in het volgende hoofdstuk. In feite is in deze tabel de toestand q_5 als opvolger van toestand q_1 op de eerste regel van tabel 6.16 gekozen. Welke mogelijkheid is er gekozen voor de opvolger van q_2 onder ga = 10?

Vindt er een bemonstering plaats (volgens de beschrijving van tabel 6.17) waarvan de uitslag dubieus is, omdat g en a nagenoeg tegelijk veranderen, dan treedt er altijd een stabiele toestand op. Gespecificeerd is:

- als ga van 00 naar 11 gaat: de opvolger van r_1 is r_1.
- als ga van 00 via 10 naar 11 gaat: de opvolger van r_1 is r_1.
- als ga van 00 via 01 naar 11 gaat: de opvolger is r_2.

Hoewel de tabel specificeert dat als ga van 00 → 11 gaat de opvolger r_1 moet zijn, zal het in een schakeling ook r_2 kunnen zijn. Tabel 6.17 beschrijft dus precies de onzekerheid die er bij een bemonstering van een veranderend signaal bestaat.

Opmerking

Tabel 6.17 is een voorbeeld van een *niet-primitieve* toestandstabel. Gereduceerde tabellen zijn in het algemeen niet-primitief. De tabellen die bij de specificatie ontstaan zijn in het algemeen wel primitief.

6.5. Een mathematisch model voor level mode sequentiële schakelingen

Tot nu toe is een te ontwerpen schakeling beschouwd als een black box met ingangssignalen en uitgangssignalen. Het begrip *inwendige toestand* is daarbij geïntroduceerd om de geheugenwerking van de black box te specificeren voor die gevallen waarbij sprake is van geheugenwerking. Bij de beschouwingen is verondersteld dat de schakeling bij elke ingangscombinatie naar een stabiele inwendige toestand gaat en daarin blijft staan totdat de volgende ingangscombinatie wordt aangeboden. De nieuwe inwendige toestand die dan ontstaat kan eventueel gelijk zijn aan de oude.

Om de optredende problemen bij de realisatie van level mode gespecificeerde schakelingen effectief te kunnen aanpakken is het nodig een mathematisch model van deze digitale schakelingen te introduceren. Dit model wordt de *sequentiële machine* genoemd.

Definitie
Een sequentiële machine M is een geordend vijftal:

$$M = (I,U,Q,\delta,\lambda)$$

waarin
I, de verzameling van de ingangscombinaties, een eindige niet-lege verzameling is;
U, de verzameling van de uitgangscombinaties, een eindige niet-lege verzameling is;
Q, de verzameling inwendige toestanden, een niet-lege aftelbare verzameling is;
δ, de toestandsfunctie, een afbeelding van $Q \times I$ *in* Q is;
λ, de uitgangsfunctie, een afbeelding van $Q \times I$ *op* U is.

De interpretatie van het model zal worden toegelicht met behulp van tabel 6.18. Deze tabel is de niet-gecodeerde versie van tabel 6.2, welke eerder als tabel 6.3 is gegeven.

q \ i	i_1	i_2	i_3	i_4	i_1	i_2	i_3	i_4
q_1	(q_1)	(q_1)	(q_1)	q_2	u_1	u_1	u_1	u_1
q_2	(q_2)	(q_2)	q_1	(q_2)	u_2	u_2	u_1	u_1

tabel 6.18. Toestandstabel.

In dit voorbeeld bestaat de ingangsverzameling I uit de vier ingangscombinaties i_1 t/m i_4:

$$I = \{i_1,i_2,i_3,i_4\}.$$

De uitgangsverzameling U bestaat uit twee uitgangscombinaties:

$$U = \{u_1,u_2\}.$$

De toestandsverzameling Q bestaat uit de inwendige toestanden q_1 en q_2:

$$Q = \{q_1,q_2\}.$$

De toestandsfunctie δ is een afbeelding van het Cartesisch produkt $Q \times I$ in Q, d.w.z. dat de functie δ vastlegt welke opvolgertoestand behoort bij elk geordend paar $\langle q,i \rangle$, waarvan $q \in Q$ en $i \in I$. De opvolgertoestand van de huidige inwendige toestand q bij de ingangscombinatie i wordt aangeduid met $\delta(q,i)$. Voor tabel 6.18 vindt men dus:

$$\delta(q_1,i_1) = q_1 \quad \delta(q_1,i_2) = q_1 \quad \delta(q_1,i_3) = q_1 \quad \delta(q_1,i_4) = q_2$$
$$\delta(q_2,i_1) = q_2 \quad \delta(q_2,i_2) = q_2 \quad \delta(q_2,i_3) = q_1 \quad \delta(q_2,i_4) = q_2.$$

Bij de analyse van schakelingen kan het voorkomen dat een toestandsovergang niet direct verloopt, maar via een of meer tussentoestanden die op zich niet stabiel zijn. Tenzij uitdrukkelijk anders vermeld wordt met $\delta(q,i)$ altijd de volgende *stabiele toestand* bedoeld, d.w.z. de toestand q_j waarvoor geldt

$$\delta(q_j,i) = q_j.$$

De uitgangsfunctie λ beeldt het geordend paar $\langle q,i \rangle$, dat gevormd wordt door

de huidige inwendige toestand q en de huidige ingangscombinatie i, af op de bij dit paar behorende *huidige uitgangswaarde.*

Voor tabel 6.18 is de λ-functie:

$$\lambda(q_1,i_1) = u_1 \quad \lambda(q_1,i_2) = u_1 \quad \lambda(q_1,i_3) = u_1 \quad \lambda(q_1,i_4) = u_1$$
$$\lambda(q_2,i_1) = u_2 \quad \lambda(q_2,i_2) = u_2 \quad \lambda(q_2,i_3) = u_1 \quad \lambda(q_2,i_4) = u_1.$$

Het komt vaak voor dat de uitgang alleen afhangt van de huidige inwendige toestand. Voorbeelden hiervan komen nog. De functie λ beeldt in dit geval de toestandsverzameling Q af op de uitgangsverzameling U.

Don't care condities zijn nog niet in het bovenstaande model opgenomen. Omdat δ en λ functies zijn, moeten voor elk geordend paar $<q,i>$ met $q \in Q$ en $i \in I$ de grootheden $\delta(q,i)$ en $\lambda(q,i)$ gespecificeerd worden. Dit zou betekenen dat don't care condities buiten het model vallen. Dit probleem kan omzeild worden door aan de toestandsverzameling Q een toestand q_d toe te voegen en aan de uitgangsverzameling een uitgangswaarde u_d. Hierbij heeft q_d de betekenis van don't care. Als opvolger van de toestand q_d wordt uiteraard in alle gevallen de toestand q_d gespecificeerd. Deze rij kan met elke andere rij van een toestandstabel worden gecombineerd en zal daarom meestal stilzwijgend worden weggelaten.

Ook combinatorische schakelingen passen in het geïntroduceerde model, zij het dat een combinatorische schakeling dan beschouwd wordt als een sequentiële schakeling met slechts één inwendige toestand. Het toestandsdiagram en de toestandstabel voor bijvoorbeeld een NAND met twee ingangen zijn hieronder gegeven.

q \ x_1x_2	00	01	10	11	00	01	10	11
q	(q)	(q)	(q)	(q)	1	1	1	0

tabel 6.19. Toestandstabel van een NAND.

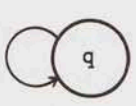


fig. 6.18. Toestandsdiagram van een NAND.

De introductie van het mathematisch model van de sequentiële machine maakt het mogelijk een efficiënte notatie in te voeren voor de specificatie van de opvolgertoestand en de uitgangswaarde, nl. $\delta(q,i)$ en $\lambda(q,i)$. Bij de behandeling van schakeltechnische algoritmen worden de eigenschappen van het model meer uitgebreid verkend.

Voor de via de level mode gespecificeerde problemen moet het model van de sequentiële machine zo geïnterpreteerd worden dat bij elke ingangsverandering de schakeling direct naar de gewenste nieuwe toestand $\delta(q,i)$ gaat. Dit in tegenstelling tot bij de nog te introduceren clock mode, waarin de toestandsverandering pas geëffectueerd wordt op door de klokpuls bepaalde tijstippen. Het zal blijken dat het model van de sequentiële machine geschikt is voor de analyse van zowel level mode als clock mode schakelingen.

In het voorafgaande zijn verschillende voorbeelden gegeven van gecodeerde en niet-gecodeerde toestandstabellen. Een niet-gecodeerde tabel is het meest geschikt voor de beschrijving van de functie van een schakeling, het input → output ge-

drag. Een gecodeerde tabel sluit het best aan bij de realisatie van de schakeling. De niet-gecodeerde toestandstabel geeft in wezen dezelfde informatie als de ermee overeenkomende sequentiële machine, er bestaat een ondubbelzinnig verband tussen beide. De gecodeerde toestandstabel geeft meer informatie, omdat tevens is vastgelegd hoe de ingangscombinaties, inwendige toestanden en uitgangscombinaties met behulp van binaire grootheden zijn vastgelegd (gecodeerd).
De binaire grootheden waarmee de diverse ingangscombinaties worden gecodeerd worden *ingangsvariabelen* genoemd. In de literatuur komt ook wel de aanduiding *primaire variabelen* voor. Ingangsvariabelen worden meestal aangeduid met de kleine letter x. Elke ingangsvariabele x_i correspondeert met een binair ingangssignaal.
Het coderen van de inwendige toestanden geschiedt met geheugenelementen. Aan elk geheugenelement wordt een *toestandsvariabele* toegekend, ook wel aangeduid als *secundaire variabele*, die de *toestand van het geheugenelement* (relais, trekker, enz.) weergeeft. Toestandsvariabelen worden meestal aangeduid met y_i, met $i = 1,2,3, \ldots$ Zijn er meer dan twee uitgangswaarden, dan zijn er voor het coderen hiervan binaire *uitgangsvariabelen* nodig. Deze worden aangeduid als z_i, $i = 1,2,3 \ldots$

Op de problemen die verband houden met het coderen van toestandstabellen en -diagrammen wordt dieper ingegaan in het volgende hoofdstuk. Reeds kan worden opgemerkt dat de codering van de ingangscombinaties en de uitgangscombinaties meestal vastligt, maar dat de toestandscodering een intern probleem voor een bepaalde schakeling is.

Het Moore en het Mealy model van schakelingen

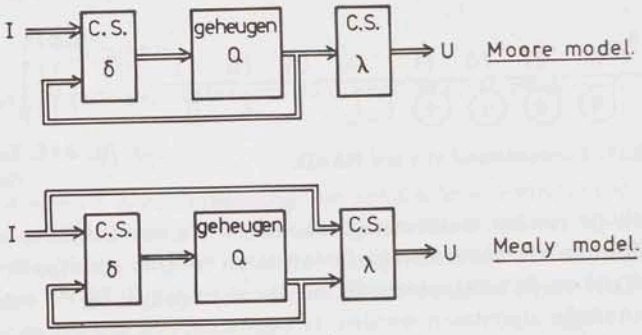


fig. 6.19. Blokschema's van schakelingen met geheugen.

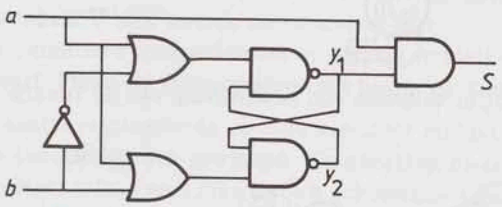
In fig. 6.19 zijn de blokschema's van het geïntroduceerde model van de sequentiële machine weergegeven. In het bovenste schema is uitsluitend de inwendige toestand bepalend voor de waarde van het uitgangssignaal, de functie λ beeldt Q op U af. Dit model wordt wel het *Moore model* genoemd. Het onderste model, ook wel het *Mealy model* genoemd, heeft een uitgangsfunctie die $Q \times I$ op U afbeeldt.
Het onderscheid tussen het Mealy model en het Moore model berust volgens deze interpretatie dus op het verschil tussen de twee verzamelingen waarop de functie λ gedefinieerd is. In de oorspronkelijke publicaties van Moore en Mealy zijn de modellen iets anders geïnterpreteerd. Mealy introduceerde het model voor

level mode schakelingen, terwijl Moore een model introduceerde voor *puls-mode* schakelingen. Wij zullen in het vervolg de interpretatie volgens fig. 6.19 volgen.

Opgaven

6.1. Onder welke voorwaarde(n) kan de logische werking van een digitale schakeling worden beschreven met een toestandstabel resp. toestandsdiagram?

6.2.



De gegeven schakeling heeft twee ingangssignalen a en b waarvan gegeven is dat zij niet tegelijk veranderen. In de schakeling zijn twee toestandsvariabelen gedefinieerd, y_1 en y_2, die de inwendige toestand vastleggen.

a. Is de uitspraak "Omdat er slechts één trekker in de schakeling zit kan eigenlijk met één toestandsvariabele worden volstaan" juist?
b. Bepaal voor elke ingangscombinatie de mogelijke stabiele toestand(en).
c. Bepaal wat er gebeurt bij elke toegestane ingangsverandering. Specificeer als opvolgertoestand de volgende stabiele toestand.
d. Wat is de functie van de schakeling?

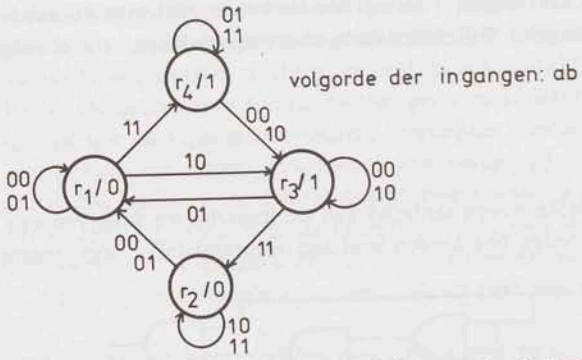
6.3. Gegeven is een S-R trekker. De ingangscombinatie SR = 11 komt niet voor.

a. Teken het toestandsdiagram voor deze trekker.
b. Als SR = 11 wel voorkomt, maar S en R veranderen niet tegelijk, hoe ziet dan het toestandsdiagram er uit?
c. En hoe als S en R wel tegelijk veranderen?

6.4. Een schakeling heeft twee bedieningsknoppen D_1 en D_2. Deze knoppen zijn mechanisch vergrendeld, zodat zij niet tegelijk kunnen worden ingedrukt.

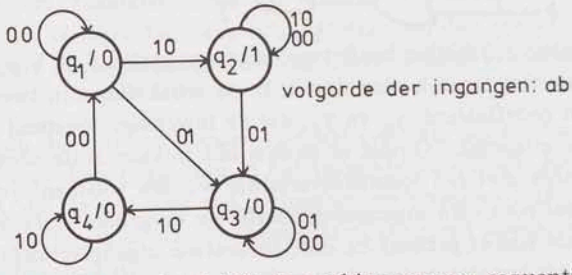
a. Specificeer via een primitieve toestandstabel in de level mode de werking van een schakeling die aangeeft welke knop het laatst werd ingedrukt of nog ingedrukt is.
b. Kent u een schakeling die dit realiseert?

6.5. In het toestandsdiagram is de werking gespecificeerd van een sequentiële schakeling, die is ontworpen volgens de level mode. De schakeling heeft twee ingangssignalen a en b, die in de aangegeven volgorde vermeld zijn. Het uitgangssignaal U kan de waarde 0 of 1 aannemen.
Gegeven is dat de 0 → 1 overgangen in de signalen a en b elkaar afwisselen, na een 0 → 1 overgang in het signaal a komt er een 0 → 1 overgang in het signaal b en omgekeerd.



Gevraagd wordt in woorden een beschrijving van de functie van deze schakeling te geven.

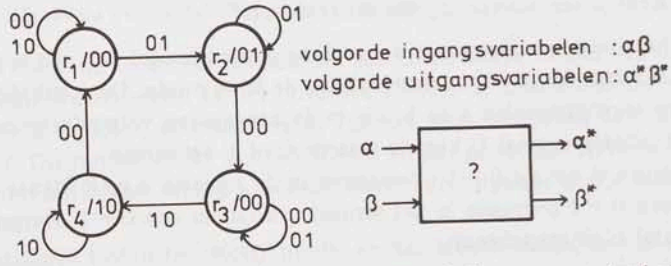
6.6.



Het toestandsdiagram beschrijft de werking van een sequentiële schakeling die is ontworpen volgens de level mode. De schakeling heeft twee ingangen a en b, waarop gelijknamige signalen worden aangeboden. Deze signalen zijn pulsvormig. Er kan geen overlapping optreden van een a-puls en een b-puls. Een volgende puls komt pas nadat de schakeling ruimschoots de tijd gehad heeft om de vorige te verwerken. Het uitgangssignaal U heeft de waarde 0 of 1.

a. Beschrijf in woorden de functie van de schakeling.
b. Geef van elke toestand apart aan wat deze bijdraagt aan het verrichten van de functie van de schakeling.

6.7. Het toestandsdiagram beschrijft de werking van een sequentiële schakeling die is ontworpen volgens de level mode. De schakeling heeft twee ingangen α en β. Op de twee uitgangen staan de signalen α^* en β^*.



a. Welke restricties zijn volgens dit toestandsdiagram opgelegd aan de onderlinge variaties van de signalen α en β.
Bedenk hierbij dat de bijbehorende toestandstabel niet primitief is.

b. Beschrijf in woorden de functie van de schakeling zoals deze in het toestandsdiagram is vastgelegd.

6.8. Aan een te ontwerpen schakeling worden twee binaire signalen a en b aangeboden. Van deze signalen is bekend dat b niet verandert als a = 1 is. Het signaal b mag wel veranderen als a = 0 of als a verandert van 0 in 1. Aangenomen wordt dat b pas weer verandert nadat a weer 0 is geworden. Het uitgangssignaal U heeft de waarde 0 of 1. Het signaal U wordt geïnverteerd telkens als a van 1 naar 0 gaat *mits* het signaal b vlak voor deze overgang van a de waarde 1 bezit. In alle overige gevallen behoudt het uitgangssignaal U zijn waarde en verandert niet.
Stel een primitieve toestandstabel op (Mealy model) die deze schakeling specificeert. Plaats de ingangsvariabelen boven de tabel in de volgorde ab.

6.9. Bij een aantal geïntegreerde circuits kan door middel van een mode control de functie worden gewijzigd. Voorbeelden hiervan zijn de omschakelbare Op/Neer tellers en Links/Rechts schuivende schuifregisters. Vaak wordt voor een correcte werking van het circuit de eis gesteld dat het "mode" signaal slechts mag veranderen als de klokingang 0 resp. 1 is (Laag resp. Hoog). Soms voldoet een signaal op de mode-ingang hier niet aan.

a. Stel een primitieve toestandstabel op die via de level mode een schakeling beschrijft met twee ingangen a en m en een uitgang m* die het volgende doet:
   - Als a = 1 is, dan is m* = m;
   - Als a = 0 is, dan heeft m* de waarde die m had toen a van 1 naar 0 ging.

   Plaats de ingangsvariabelen in de volgorde ma.

b. Geef tevens aan voor welke veranderingen van de ingangscombinatie de eerstvolgende stabiele toestand en de bijbehorende uitgangswaarde niet bepaald is op grond van bovenstaande specificatie. Motiveer het antwoord.

c. Kent u een eenvoudige schakeling waarmee dit probleem kan worden opgelost?

6.10. Een schakeling heeft twee ingangssignalen a en b die niet tegelijk veranderen. Als regel zijn deze signalen 0, maar gedurende korte tijd kunnen zij de waarde 1 aannemen. De uitgang U van de schakeling moet 1 zijn als ab = 01 is mits de ingangscombinatie vooraf gegaan is door de ingangscombinatie ab = 10 en ab = 00 (in deze volgorde). Ook moet U = 1 zijn als ab = 10 is, mits vooraf gegaan door ab = 01 en ab = 00 in deze volgorde. In alle overige gevallen heeft U de waarde 0.
Toelichting:

a	...100 ...10.0110...
b	...001 ...1100 11...
U	...001 ...000100...

Stel een primitieve toestandstabel op die de werking van de te ontwerpen schakeling specificeert. Gebruik het Mealy model van een level mode schakeling.

6.11. Een schakeling heeft twee ingangssignalen a en x en een uitgangssignaal a*.

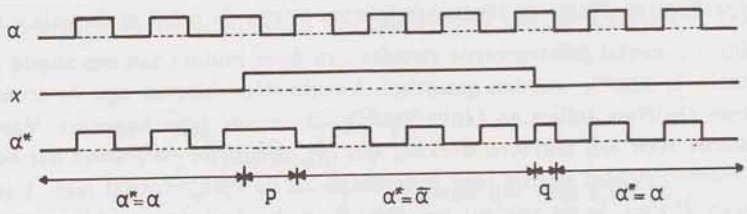
Het signaal a bestaat uit een periodieke rij pulsen en x is een omschakelsignaal. Het verband tussen a, a^* en x is als volgt:

– Als x = 0 is, dan is $a^* = a$;

– Als x = 1 is, dan is $a^* = \bar{a}$.

Dit verband geldt niet tijdens het omschakelen van x. Als x van 0 naar 1 wordt omgeschakeld, dan moet a^* voor de eerste keer van 1 naar 0 gaan als a voor de eerste keer van 0 naar 1 gaat. Hierna is $a^* = \bar{a}$.

Evenzo is vereist dat, als x van 1 naar 0 gaat, a^* voor de eerste maal van 1 naar 0 gaat als a voor de eerste keer daarna weer van 1 naar 0 gaat. Hierna is $a^* = a$. Het tijddiagram geeft een toelichting op enkele situaties.

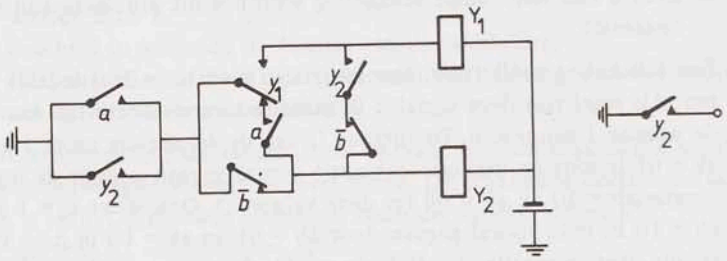


punt p: Als x naar 1 gaat als $a = 1$ is, dan moet a^* nog 1 blijven tot de eerste 0 naar 1 overgang van a.

punt q: Als x van 1 naar 0 gaat als $a = 1$ is, dan moet er een 1 naar 0 overgang in a^* komen als a van $1 \to 0$ gaat. Dit kan alleen worden bereikt als a^* direct naar 1 gaat bij de verandering van x.

Stel een primitieve toestandstabel op waarin de werking van de te ontwerpen schakeling wordt gespecificeerd. Hierbij kan worden aangenomen dat a en x niet tegelijk veranderen. Ook wordt gegeven dat veranderingen in het signaal x relatief zelden voorkomen en tenminste enkele perioden van a uit elkaar liggen. Noteer de ingangen in de volgorde xa.

6.12.



Gegeven is een relaisschakeling met twee ingangssignalen a en b en twee inwendige signalen y_1 en y_2, welke de stand van twee relais Y_1 en Y_2 vastleggen. De uitgang van de schakeling is gelijk aan $U = y_2$.

a. Stel de formules op voor de bekrachtigingsketens van de relais Y_1 en Y_2. Vereenvoudig deze formules zoveel mogelijk.

b. Specificeer in een toestandstabel welke inwendige toestanden stabiel zijn bij elke ingangscombinatie ab.

c. Ga voor elke mogelijke ingangsverandering na wat er gebeurt als de schakeling aanvankelijk in een stabiele toestand is. Een nieuwe ingangsverandering vindt pas plaats als de vorige door de schakeling geheel is verwerkt. Neem aan dat gelijknamige kontakten gelijktijdig schakelen.

6.13. In par. 3.1 is als voorbeeld gegeven de beveiliging bij de bediening van een hydraulische pers. Beide knoppen moeten worden ingedrukt voordat de pers kan worden ingeschakeld.

a. Ga na wat bij deze opzet van de schakeling de gevolgen zijn als de bedieningsman om toch een hand vrij te hebben een lucifer tussen een van de drukknoppen weet te duwen.

b. Ga na dat dit probleem kan worden opgelost door te eisen dat de pers pas kan worden ingeschakeld als na de vorige inschakeling ervan de beide drukknoppen een keer losgelaten zijn en vervolgens samen weer ingedrukt worden.

c. Stel in een primitieve toestandstabel een level mode specificatie op voor een aangepaste beveiliging.

Literatuur

1. C.H. Eversdijk, *Schakeltechniek III,* Collegediktaat T.H. Delft 1973.
2. A. Gill, *Realisation of Input-Output Relations by Sequential Machines,* J. Ass. Comput. Mach., Vol. 13, no. 1, 1966, pp. 33–42.
3. F.J. Hill and G.R. Peterson, *Introduction to Switching Theory and Logical Design,* John Wiley and Sons Inc., New York 1974.
4. D.A. Huffman, *The Synthesis of Sequential Switching Circuits,* J. Franklin Inst., Vol. 257, nos. 3 and 4, March and April 1954, pp. 161–190 and 275–303.
5. G.A. Maley and J. Earle, *The Logic of Transistor Digital Computers,* Prentice-Hall Inc., Englewood Cliffs N.J., 1963.
6. G.H. Mealy, *A Method for Synthesising Sequential Circuits,* Bell System Tech. J., Vol. 34, Sept. 1955, pp. 1045–1079.
7. E.F. Moore, *Gedanken-experiments on Sequential Machines,* Annals of Mathematical Studies, no. 34, Princeton Univ. Press; Princeton, N.J., 1956, pp. 129–153.
8. J.P. Perrin, M Denouette et E. Daclin, *Systèmes Logiques,* Dunod, Paris 1967, Tome 1 et Tome 2.
9. A.P. Thijssen, *The Assignment of State Variables in Feedback Loops of Sequential Switching Circuits,* Delft Progress Report 3, 1977, pp. 39–53

7. REALISATIE VAN LEVEL MODE SEQUENTIELE SCHAKELINGEN

7.1. Aspecten van de realisatie van level mode sequentiële schakelingen

In het vorige hoofdstuk zijn enkele aspecten besproken van de analyse en de specificatie van sequentiële schakelingen. De toestandstabel en het toestandsdiagram zijn daarbij nuttige hulpmiddelen gebleken. Het begrip *inwendige toestand* is ingevoerd om de geheugenfunctie van een schakeling te kunnen beschrijven en wel onafhankelijk van de bouwstenen waarmee de schakeling kan worden gerealiseerd. Daarna is met behulp van de toestandstabel de logische specificatie van enkele schakeltechnische problemen opgesteld. In het volgende zal blijken dat het vertalen van een toestandstabel in een correct werkende logische schakeling een proces vol haken en ogen is. Als inleidend voorbeeld behandelen we de realisatie van de tweedeler. De specificatie ervan is in het vorige hoofdstuk opgesteld.

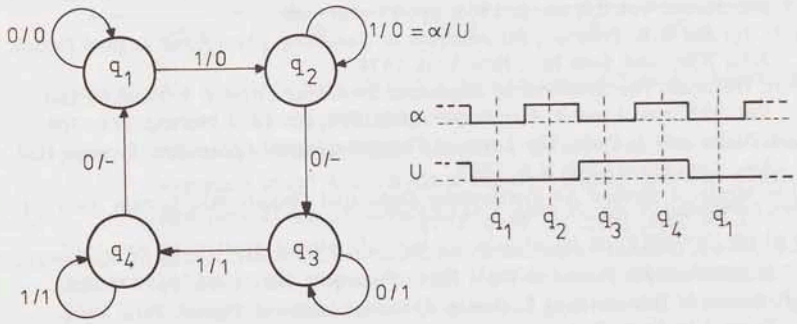


fig. 7.1. Tijddiagram en toestandsdiagram van een tweedeler.

In fig. 7.1 is de specificatie van een tweedeler met behulp van vier inwendige toestanden q_1 tot en met q_4 gegeven, zoals deze in het vorige hoofdstuk is opgesteld. In het algemeen bevat een initiële specificatie redundantie, o.a. met betrekking tot het aantal toestanden dat geïntroduceerd is. Het realisatieproces begint met een kritische beschouwing van de opgestelde specificatie. Reeds is gebleken dat voor de specificatie van de tweedeler minimaal vier toestanden nodig zijn, een reductie van de tabel of het diagram is niet mogelijk. *Fase 1 van het realisatieproces, het onderzoek naar mogelijkheden om de in de toestandstabel beschreven specificatie te reduceren,* is hiermee achter de rug.
Vervolgens komt *fase 2, het coderen van de inwendige toestanden,* aan de orde. Bij het coderen van de inwendige toestanden moet een beslissing genomen worden hoe de verschillende inwendige toestanden in de schakeling voor te stellen. In de praktijk ligt de codering van de ingangscombinaties en uitgangscombinaties meestal reeds vast, zodat alleen de codering van de inwendige toestanden moet worden toegekend. Welke aspecten van het coderingsproces van invloed zijn op de betrouwbaarheid van de schakeling zal nu worden onderzocht.

Voor het *coderen* van de vier inwendige toestanden van de tweedeler zijn twee toestandsvariabelen y_1 en y_2 nodig. De waarde van elke toestandsvariabele wordt vastgelegd in een geheugenelement, bijvoorbeeld een relais of een trekker. In

principe zijn er voor het diagram van een tweedeler drie essentieel verschillende coderingen mogelijk. Deze zijn:

q	y_1y_2	q	y_1y_2	q	y_1y_2
q_1	0 0	q_1	0 0	q_1	0 0
q_2	0 1	q_2	1 0	q_2	1 1
q_3	1 0	q_3	1 1	q_3	0 1
q_4	1 1	q_4	0 1	q_4	1 0

Alle overige coderingen zijn door verwisseling en/of inversie van de variabelen y_1 en y_2 uit een van deze coderingen af te leiden. Ga dit na.
In het algemeen kunnen bij level mode gespecificeerde schakelingen de beschikbare combinaties van de toestandsvariabelen niet willekeurig aan de inwendige toestanden worden toegekend. Als voorbeeld van de problemen die hierbij kunnen optreden worden twee van de bovenstaande toestandscoderingen van de tweedeler onderzocht. Fig. 7.2 geeft de gecodeerde toestandsdiagrammen die bij de eerste twee toekenningen (*state assignments*) behoren.

De codering volgens fig. 7.2.a leidt tot moeilijkheden en wel om de volgende reden:
Stel dat de schakeling die fig. 7.2.a gaat realiseren in de inwendige toestand q_2 is. Deze toestand is gecodeerd met $y_1y_2 = 01$ en is volgens het diagram stabiel onder $\alpha = 1$. Het uitgangssignaal U is daarbij 0. Op zeker moment gaat het ingangssignaal α van $1 \rightarrow 0$. Volgens het diagram moet de schakeling naar de nieuwe toestand $\delta(q_2,0) = q_3$ gaan.

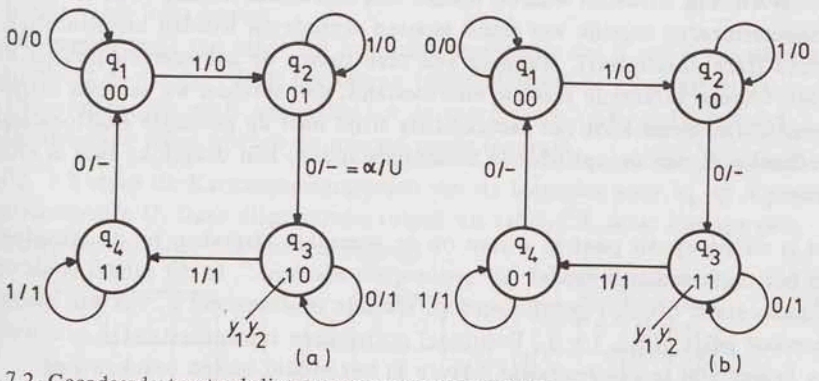


fig. 7.2. Gecodeerde toestandsdiagrammen voor de tweedeler.

Deze toestand is gecodeerd met $y_1y_2 = 10$. Beide geheugenelementen, waarin de waarde van de toestandsvariabelen y_1 en y_2 is vastgelegd, moeten dus van stand veranderen. En dit is nu een bron van moeilijkheden:
Stel dat het geheugenelement Y_1 dat de waarde van de toestandsvariabele y_1 vastlegt om een of andere reden beduidend trager is dan het geheugenelement Y_2 dat de toestandsvariabele y_2 vastlegt. We kunnen hierbij denken aan twee trekkers Y_1 en Y_2 of aan twee relais. De volgende situatie doet zich voor:
Gespecificeerd is dat $\delta(q_2,0) = q_3$ is met q_2: $y_1y_2 = 01$ en q_3: $y_1y_2 = 10$. In feite vindt ten gevolge van het verschil in reactietijd eerst de volgende overgang plaats:

$$y_1y_2 = 01 \xrightarrow{a=0} y_1y_2 = 00.$$

Maar zodra de toestand $y_1y_2 = 00$ bereikt wordt (toestand q_1) zal de schakeling er naar streven in deze toestand te blijven. Immers, er is gespecificeerd dat $\delta(q_1,0) = q_1$ is. Het hangt nu geheel van het verschil in reactietijd tussen Y_1 en Y_2 af wat er gebeurt, alsnog doorgaan naar toestand q_3 of stoppen in toestand q_1.
Verwisseling van de trekkers, zodat het geheugenelement Y_1 sneller wordt dan het geheugenelement Y_2 geeft in de praktijk wel eens een oplossing, maar moet als remedie worden verworpen. Denk bijvoorbeeld maar eens aan het vervangen van defecte onderdelen!

Conclusie
De codering volgens fig. 7.2.a is onbruikbaar om een betrouwbaar werkende schakeling te ontwerpen. Het is onder omstandigheden dus mogelijk dat een bepaalde toestandscodering aanleiding geeft tot moeilijkheden. De oorzaak kan zijn een verschil in reactietijd tussen de diverse geheugenelementen waarmee de toestanden worden gecodeerd. Het geschetste probleem treedt op als er meer dan één geheugenelement tegelijk van stand verandert gedurende een bepaalde toestandsovergang. Verandert er bij elke toestandsovergang slechts één toestandsvariabele, dan treedt het geschetste probleem niet op.

De codering van het toestandsdiagram van de tweedeler volgens fig. 7.2.b geeft geen aanleiding tot bovengenoemde verschijnselen. Deze codering is een zgn. *progressieve codering*, bij iedere toestandsovergang verandert er slechts één toestandsvariabele. Situaties waarbij tijdens een toestandsovergang twee of meer geheugenelementen tegelijk van stand moeten veranderen worden *racecondities* genoemd (*race conditions*). Wanneer een race tussen de geheugenelementen kan leiden tot een verkeerde stabiele eindtoestand, dan spreken we van een *kritische raceconditie*. Soms leidt een raceconditie altijd naar de gewenste eindtoestand, onafhankelijk van de optredende tussentoestanden. Een dergelijke race is *niet-kritisch*.

Het is nuttig op dit punt te wijzen op de gemaakte afspraken bij de introductie van het mathematisch model *"de sequentiële machine"*. In dit model is als opvolgertoestand $\delta(q_j,i_k)$ gedefinieerd de stabiele eindtoestand q_l van een overgang waarvoor geldt $\delta(q_l,i_k) = q_l$. Eventueel optredende tussentoestanden, *die wel van belang zijn in een realisatie*, blijven in het model buiten beschouwing.

q \ a	0	1	0	1
q_1	$\textcircled{q_1}$	q_2	0	0
q_2	q_3	$\textcircled{q_2}$	–	0
q_3	$\textcircled{q_3}$	q_4	1	1
q_4	q_1	$\textcircled{q_4}$	–	1

tabel 7.1. Toestandstabel

$q_1 \leftrightarrow 00$	00
$q_2 \leftrightarrow 10$	10
$q_3 \leftrightarrow 11$	11
$q_4 \leftrightarrow 01$	01

y_1y_2 \ a	0	1	0	1
00	$\textcircled{00}$	10	0	0
10	11	$\textcircled{10}$	–	0
11	$\textcircled{11}$	01	1	1
01	00	$\textcircled{01}$	–	1

tabel 7.2. Toestandstabel

De toestandstabel voor de tweedeler (tabel 7.1) wordt gecodeerd met behulp van de in fig. 7.2.b aangegeven toestandscodering. De gecodeerde versie van tabel

7.1 is tabel 7.2. Voor de codering van de inwendige toestanden zijn twee geheugenelementen Y_1 en Y_2 nodig. Uit de gecodeerde toestandstabel kunnen de werkingsvoorwaarden van de twee geheugenelementen Y_1 en Y_2 worden afgeleid. Hiertoe wordt waarheidstabel 7.3 opgesteld. We zijn nu gekomen aan *fase 3 van het ontwerpproces, het opstellen van de werkingsvoorwaarden van de geheugenelementen* (en ook de formules der uitgangssignalen).

y_1	y_2	a	Y_1	Y_2	U
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	0	0	–
0	1	1	0	1	1
1	0	0	1	1	–
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	0	1	1

tabel 7.3. Waarheidstabel voor de tweedeler.

In tabel 7.3 stelt y_i de oude toestand van het geheugenelement voor en Y_i de gewenste nieuwe toestand bij een bepaalde ingangscombinatie. De constructie van de tabel geschiedt als volgt:

- De eerste regel van tabel 7.3 correspondeert met de toestand q_1 ($y_1y_2 = 00$) en het ingangssignaal $a = 0$. Volgens tabel 7.2 is deze toestand stabiel, hetgeen leidt tot $Y_1Y_2 = 00$. Deze waarde wordt in tabel 7.3 ingevuld. Dezelfde informatie staat in de eerste twee Karnaughdiagrammen van fig. 7.3 in het vakje links boven.
- De tweede regel van tabel 7.3 beschrijft de situatie dat a van $0 \rightarrow 1$ gaat in toestand q_1 ($y_1y_2 = 00$). De nieuwe toestand moet de toestand q_2 ($y_1y_2 = 10$) worden. Dit geeft aanleiding tot de specificatie $Y_1Y_2 = 10$ in tabel 7.3.
- Op dezelfde wijze worden de overige regels ingevuld.

In fig. 7.3 staan de Karnaughdiagrammen van de formules voor Y_1 en Y_2 en de uitgangsfunctie U. Deze diagrammen volgen uit tabel 7.3, maar kunnen ook rechtstreeks uit tabel 7.2 worden bepaald.

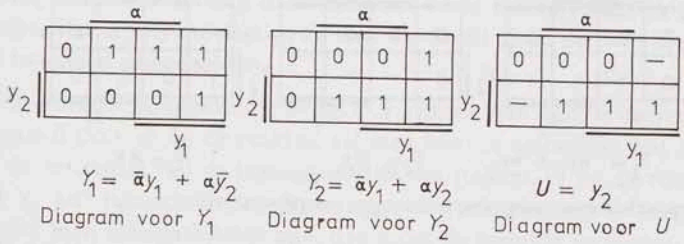


fig. 7.3. Karnaughdiagrammen voor de tweedeler.

Met behulp van de diagrammen in fig. 7.3 kunnen formules worden bepaald die de werkingsvoorwaarden van de geheugenelementen Y_1 en Y_2 vastleggen, alsmede de gewenste uitgangsfunctie. Deze zijn:

$$Y_1 = \bar{\alpha} y_1 + \alpha \bar{y}_2$$
$$Y_2 = \bar{\alpha} y_1 + \alpha y_2$$
$$U = y_2.$$

We zijn nu gekomen aan de *vierde fase van het ontwerpproces, het realiseren van de schakeling.* Als voorbeeld zal een realisatie met behulp van $\bar{S}$-$\bar{R}$ trekkers als geheugenelementen worden ontworpen.

Realisatie met $\bar{S}$-$\bar{R}$ trekkers

De werking van een trekker wordt beschreven in de tabellen 7.4 en 7.5. Tabel 7.5 volgt gemakkelijk uit tabel 7.4. Wanneer $z_{oud} = 0$ is en ook $Z_{nieuw} = 0$ is, dan kan dit worden bereikt door een RESET te geven (SR = 01), of door niets te doen (SR = 00). In feite is de RESET in dit geval facultatief. Evenzo volgen de andere regels van tabel 7.5 uit tabel 7.4.

	S	R	Z
onthouden	0	0	z
resetten	0	1	0
setten	1	0	1
verboden	1	1	–

tabel 7.4. Waarheidstabel van een trekker.

$\Longleftrightarrow$

z	Z	S	R
0	0	0	–
0	1	1	0
1	0	0	1
1	1	–	0

tabel 7.5. Ingangscondities voor een trekker.

Met behulp van tabel 7.5 kunnen de formules voor de ingangen S en R van de trekkers Y_1 en Y_2 uit de diagrammen in fig. 7.3 worden afgeleid. Het resultaat staat in fig. 7.4.

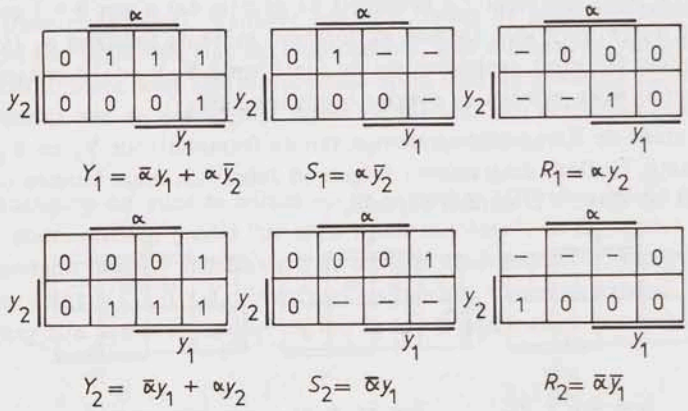


fig. 7.4. Karnaughdiagrammen voor het set- en resetsignaal.

Wanneer $\bar{S}$-$\bar{R}$ trekkers gebruikt worden in een realisatie, dan moeten de inverse waarden van S en R als ingangssignaal worden aangeboden. De in fig.7.4 gevonden formules moeten daartoe worden geïnverteerd. Bij gebruik van NAND's wordt deze inversie automatisch uitgevoerd. De schakeling voor de tweedeler, zoals deze met behulp van fig. 7.4 wordt gevonden staat in fig. 7.5.
Fase 4 van het ontwerpproces, *de realisatiefase,* is hiermee in eerste instantie voltooid. De schakeling van fig. 7.5 werkt als tweedeler:

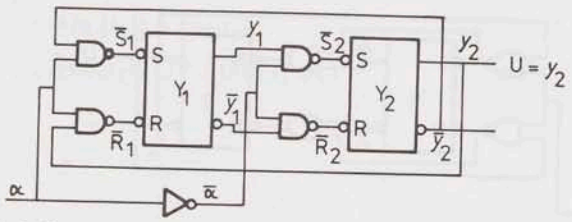


fig. 7.5. Schakeling voor de tweedeler.

- Als $a = 0$ is, dan neemt trekker Y_2 de waarde van trekker Y_1 over. De ingangspoorten van Y_1 zijn geblokkeerd.
- Gaat a vervolgens naar 1, dan worden de ingangspoorten van Y_2 geblokkeerd en neemt Y_1 de inverse stand van Y_2 over. Zolang als $a = 1$ blijft, treden er geen veranderingen meer op.
- Gaat a daarna weer naar 0, dan wordt trekker Y_1 geblokkeerd. De stand van deze trekker is dus precies het omgekeerde van de oude stand van trekker Y_2. Trekker Y_2 neemt als $a = 0$ is als nieuwe stand de stand van trekker Y_1 weer over.
- Tot de eerstvolgende verandering van a van $0 \rightarrow 1$ blijft deze situatie bestaan.

We zien dus als resultaat na één overgang van a van $0 \rightarrow 1 \rightarrow 0$ dat de uitgang U geïnverteerd is. En dat is precies wat de tweedeler moet doen!

Het zou ideaal zijn als hiermee het ontwerpproces ten einde was. Dit is echter niet zo. De bouwstenen waarover de ontwerper beschikt zijn niet-ideale bouwstenen. Zo kent iedere bouwsteen een zekere vertragingstijd tussen een ingangsverandering en de uitgangsreactie hierop. Er moet onderzocht worden of de schakeling, die uit niet-ideale bouwstenen is opgebouwd, het gewenste logisch gedrag ook inderdaad realiseert. Daarbij rekening houdend met o.a. deze vertragingen.
De ontworpen schakeling in fig. 7.5 blijkt in principe kritisch te zijn als de vertraging van de toegepaste invertor in de a-draad extreem groot zou zijn. Dit is als volgt in te zien:
Zij $a = 0$. De eerste trekker Y_1 is geblokkeerd, terwijl de tweede trekker Y_2 de stand van de eerste overneemt. Stel nu dat a van $0 \rightarrow 1$ gaat, maar dat $\bar{a}$ nog even 1 blijft. Er ontstaat dan de situatie dat beide trekkers tijdelijk gedeblokkeerd zijn. Het is niet moeilijk in te zien dat in dit geval de definitieve stand van de tweedeler onbepaald is.

De reden waarom de schakeling in fig. 7.5 kritisch kan zijn, is de vertraging van het signaal $\bar{a}$ t.o.v. a. In de praktijk zal men hiervan nauwelijks last ondervinden omdat de vertraging van de ingangspoorten van trekker Y_1 en de vertraging van trekker Y_1 zelf ruimschoots opwegen tegen de vertraging van de toegepaste invertor. Wil men helemaal zeker zijn, dan moet de invertor vermeden worden. Voor deze schakeling kan dit toevallig. Een fraaie schakeltechnische oplossing staat in fig. 7.6. Het wordt aan de lezer overgelaten na te gaan of en zo ja hoe de oplossing van fig. 7.6 in de diagrammen van fig. 7.4 begrepen is.
Met deze stap is de *laatste fase, de ontwerpverificatie*, afgesloten en is een betrouwbaar werkende schakeling ontstaan.

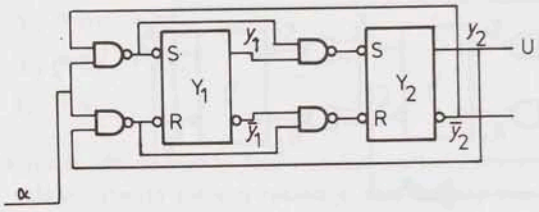


fig. 7.6. Tweedeler.

In het voorafgaande zijn enkele facetten van het ontwerpen van level mode gespecificeerde digitale schakelingen belicht, te weten:

- de specificatie (toestandstabel, toestandsdiagram).
- de reductie van de specificatie.
- het coderen van de inwendige toestanden (racecondities).
- het opstellen van de werkingsvoorwaarden.
- het kiezen van de bouwstenen en het opstellen van de formules.
- de ontwerpverificatie, waarbij onderzocht wordt of het veronderstelde gedrag inderdaad wordt gerealiseerd.

In de praktijk is het meestal niet mogelijk elke stap onafhankelijk van de voorafgaande stappen te optimaliseren.

7.2. Het reduceren van toestandstabellen

Een toestandstabel, zoals deze bij het specificeren van level mode sequentiële schakelingen ontstaat, kan meestal worden vereenvoudigd. Hetzelfde probleem kan ook beschreven worden in een tabel met minder inwendige toestanden. Wanneer dit het geval is, dan moet men voor elke toestand in de oorspronkelijke tabel een toestand in de nieuwe tabel kunnen aanwijzen, die de functie van de betreffende toestand kan overnemen. Om de eisen te kunnen formuleren, die aan een tabel gesteld moeten worden opdat deze een andere tabel kan vervangen, moet het begrip ingangswoord worden geïntroduceerd. Ook de toestandsfunctie δ en de uitgangsfunctie λ moeten worden uitgebreid voor ingangswoorden.

Ingangswoord

Onder een *ingangswoord J*

$$J = [i_1 i_2 \ldots i_{n-1} i_n]$$

met een lengte $L(J) = n$ wordt verstaan een rij ingangscombinaties $i_1, i_2, \ldots, i_{n-1}, i_n$ welke opeenvolgend aan de schakeling worden aangeboden. Dat wil zeggen eerst de combinatie i_1. Daarna, als de schakeling tot rust is gekomen, de combinatie i_2. Enz. Met behulp van ingangswoorden is het mogelijk de toestandsfunctie δ en de uitgangsfunctie λ uit te breiden voor rijen ingangscombinaties ter lengte $L(J) \geqslant 1$.

De uitgebreide toestandsfunctie $\underline{\delta}$

Onder de toestand $\underline{\delta}(q,J)$ wordt verstaan de stabiele inwendige toestand waarin de schakeling terecht komt als deze in de toestand q begint en achtereenvolgens alle ingangscombinaties van de rij J worden aangeboden. In formule:

$$\underline{\delta}(q,[i_1]) = \delta(q,i_1)$$
$$\underline{\delta}(q,[i_1 i_2]) = \delta(\delta(q,i_1),i_2)$$

.

.

.

$$\underline{\delta}(q,[i_1 i_2 \ldots i_{n-1} i_n]) = \delta(\underline{\delta}(q,[i_1 i_2 \ldots i_{n-1}]),i_n).$$

Een probleem vormen hierbij toestanden waarvan de opvolger onder een bepaalde ingangscombinatie niet gespecificeerd is. Rijen ingangscombinaties waarin voor een toestand q op zeker moment dit effect optreedt, kunnen voor die toestand buiten beschouwing blijven. Toon dit aan.

De uitgebreide uitgangsfunctie $\underline{\lambda}$

Evenals de toestandsfunctie kan ook de λ-functie worden gedefinieerd voor ingangswoorden. De functie λ wordt als volgt uitgebreid:

$$\underline{\lambda}(q,[i_1]) = \lambda(q,i_1)$$
$$\underline{\lambda}(q,[i_1 i_2]) = \lambda(\delta(q,i_1),i_2)$$

.

.

.

$$\underline{\lambda}(q,[i_1 i_2 \ldots i_{n-1} i_n]) = \lambda(\underline{\delta}(q,[i_1 i_2 \ldots i_{n-1}]),i_n).$$

Deze definitie is alleen zinvol als alle tussenliggende toestanden gedefinieerd zijn, zodat de aanvullende eis gesteld wordt dat $\underline{\delta}(q,[i_1 i_2 \ldots i_{n-1}])$ gedefinieerd moet zijn.

Met behulp van de uitgebreide functies $\underline{\delta}$ en $\underline{\lambda}$ kan worden omschreven welke ingangswoorden zinvol zijn voor elke inwendige toestand van de toestandstabel.

Gespecificeerd ingangswoord

Een ingangswoord $J = [i_1 i_2 \ldots i_{n-1} i_n]$ is een voor een inwendige toestand q *gespecificeerd ingangswoord* als voor deze toestand de functies

$$\underline{\delta}(q,[i_1 i_2 \ldots i_{n-1}])$$

en

$$\underline{\lambda}(q,[i_1 i_2 \ldots i_{n-1} i_n])$$

gedefinieerd zijn. De verzameling van alle voor een toestand q gespecificeerde ingangswoorden wordt aangeduid met Γ_q ("gamma q").
Uitsluitend de ingangswoorden die behoren tot Γ_q behoeven voor de toestand q in beschouwing genomen te worden om te onderzoeken of de functie van deze toestand kan worden overgenomen (gedekt) door een toestand in een andere tabel (of van dezelfde tabel). Als nu geldt dat voor elke toestand q van een toestandstabel er een toestand q_r van een gereduceerde tabel kan worden aangewezen zodat geldt

$$\underline{\lambda}_r(q_r,J) = \underline{\lambda}(q,J)$$

voor alle ingangswoorden $J \in \Gamma_q$, dan beschrijft ook deze gereduceerde tabel het probleem.

Voorbeeld

q \ i	i_1	i_2	i_3	i_4	i_1	i_2	i_3	i_4
q_1	(1)	q_2	q_3	–	0	0	0	–
q_2	q_1	(2)	–	q_4	0	0	–	–
q_3	q_1	–	(3)	q_5	0	0	0	0
q_4	–	q_2	q_6	(4)	0	0	1	1
q_5	–	q_2	q_3	(5)	0	0	0	0
q_6	q_1	–	(6)	q_4	0	0	1	1

tabel 7.6. Toestandstabel.

Voor toestandstabel 7.6 geldt dat

$$\begin{aligned}\underline{\delta}(q_1,[i_1i_3i_4i_2]) &= \underline{\delta}(\delta(q_1,i_1),[i_3i_4i_2]) \\ &= \underline{\delta}(\delta(q_1,i_3),[i_4i_2]) \\ &= \underline{\delta}(\delta(q_3,i_4),[i_2]) \\ &= \underline{\delta}(q_5,i_2) \\ &= q_2.\end{aligned}$$

De bijbehorende uitgangsfunctie $\underline{\lambda}$ voor deze ingangsrij is

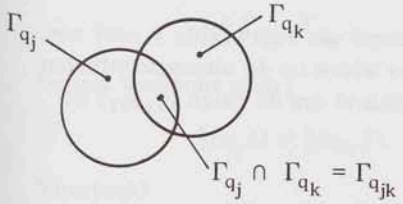
$$\begin{aligned}\underline{\lambda}(q_1,[i_1i_3i_4i_2]) &= \lambda(\underline{\delta}(q_1,[i_1i_3i_4]),i_2) \\ &= \lambda(q_5,i_2) \\ &= 0.\end{aligned}$$

Wordt echter de ingangsrij $[i_3i_4i_2i_4]$ in toestand q_1 aangeboden dan volgt dat

$$\begin{aligned}\underline{\delta}(q_1,[i_3i_4i_2i_4]) &= q_4 \\ \underline{\lambda}(q_1,[i_3i_4i_2i_4]) &= \lambda(q_2,i_4) = - \text{ (don't care).}\end{aligned}$$

De ingangsrij $[i_3i_4i_2i_4]$ leidt wel tot een gedefinieerde inwendige toestand, maar wordt *niet* als een voor toestand q_1 gespecificeerde ingangsrij beschouwd. Daarentegen is de ingangsrij $[i_3i_4i_2i_4]i_2 = [i_3i_4i_2i_4i_2]$ wel te beschouwen als een voor toestand q_1 gespecificeerde ingangsrij. Uitsluitend die rijtjes worden als gespecificeerd ingangswoord beschouwd die leiden tot een gespecificeerd uitgangssignaal. Hoe een schakeling inwendig werkt, is hierbij niet van belang. Alleen het uitwendig gedrag wordt onderzocht. Het reduceren van toestandstabellen van sequentiële schakelingen komt er dus op neer een tabel met minder toestanden op te stellen die het gewenste uitwendige gedrag volgens de oorspronkelijke specificatie specificeert. Als regel specificeert een gereduceerde tabel meer, maar dat is niet belangrijk.

De vraag die nu gesteld kan worden is: Onder welke voorwaarden kan de functie van twee verschillende toestanden in één nieuwe toestand worden gecombineerd? Fig. 7.7 geeft een illustratie van de voorwaarden waaronder dit kan.



eis 1: $\underline{\lambda}(q_j,J) = \underline{\lambda}(q_k,J)$ voor alle J in $\Gamma_{q_j} \cap \Gamma_{q_k}$

eis 2: $\underline{\lambda}_r(q_r,J) = \underline{\lambda}(q_j,J)$ voor alle J in Γ_{q_j}

$\underline{\lambda}_r(q_r,J) = \underline{\lambda}(q_k,J)$ voor alle J in Γ_{q_k}

Fig. 7.7. Voorwaarden waaronder de functies van twee toestanden q_j en q_k in één toestand q_r kunnen worden gecombineerd.

In feite betekenen de voorwaarden in fig. 7.7 dat als van twee toestanden q_j en q_k geldt dat:

$$\underline{\lambda}(q_j,J) = \underline{\lambda}(q_k,J) \text{ voor alle ingangswoorden J in } \Gamma_{q_j} \cap \Gamma_{q_k},$$

er altijd een nieuwe toestand q_r gedefinieerd kan worden die de functie van beide toestanden in zich verenigt!

Definitie
Twee toestanden q_j en q_k van een toestandstabel waarvoor geldt dat

$$\underline{\lambda}(q_j,J) = \underline{\lambda}(q_k,J) \text{ voor alle J in } \Gamma_{q_j} \cap \Gamma_{q_k}$$

worden *compatibel,* d.w.z. verenigbaar, genoemd.

Het reductieprobleem van toestandstabellen valt uiteen in twee deelproblemen:
1. Het bepalen van een overzicht welke paren toestanden compatibel zijn.
2. Het selecteren van een geschikte combinatie uit alle mogelijkheden.
Beide deelproblemen worden verder uitgewerkt in de paragrafen 7.3 en 7.4.

Opmerking
Voor volledig gespecificeerde toestandstabellen geldt dat *ieder* ingangswoord voor elke toestand een gespecificeerd ingangswoord is. Het reductieprobleem voor volledig gespecificeerde tabellen is in principe dus eenvoudiger oplosbaar. De benadering voor onvolledig gespecificeerde tabellen is echter ook toepasbaar. Om deze reden wordt van een aparte behandeling afgezien.

7.3. Compatibiliteit van toestanden

De logische functie van twee inwendige toestanden kan niet in één toestand worden gecombineerd als voor tenminste één ingangscombinatie de bijbehorende uitgangscombinaties van de twee toestanden verschillend zijn. Een don't care uitgangscombinatie wordt in deze niet als conflict beschouwd. Hieruit volgt voorwaarde 1.

Voorwaarde 1
Twee toestanden q_j en q_k kunnen slechts tot één nieuwe toestand q_r worden samengenomen als er geen ingangscombinatie i bestaat waarvoor $\lambda(q_j,i)$ verschillend is van $\lambda(q_k,i)$.

Deze voorwaarde is een noodzakelijke, maar geen voldoende voorwaarde. In een driehoekvormige tabel wordt voor elk paar toestanden aangegeven of zij wel of niet te combineren zijn. Een dergelijke tabel wordt *compatibiliteitsdriehoek* genoemd.
Tabel 7.7 geeft de *uitgangsconflicten* die op grond van voorwaarde 1 voor een tweedeler aangegeven kunnen worden. Wanneer alleen op de uitgangsconflicten wordt gelet, dan bestaat in principe de mogelijkheid om de paren (q_1,q_2) en (q_3,q_4) te combineren.

q \ x	0	1	0	1
q_1	q_1	q_2	0	0
q_2	q_3	q_2	–	0
q_3	q_3	q_4	1	1
q_4	q_1	q_4	–	1

	q_1	q_2	q_3	q_4
q_1	·	×	×	
q_2		×	×	
q_3			·	
q_4				

tabel 7.7. Toestandstabel van een tweedeler met uitgangsconflicten.

Notatie
In de compatibiliteitsdriehoek betekent:
× het betreffende paar toestanden kan niet worden gecombineerd (is incompatibel);
– het betreffende paar toestanden kan tot één toestand worden gecombineerd, d.w.z. de toestanden zijn compatibel;
· van het betreffende paar toestanden is nog niet bekend of zij compatibel zijn.

Zouden we nu de beslissing nemen om de toestanden q_1 en q_2 samen te voegen, dan houdt dit automatisch in dat ook de opvolgers van toestand q_1, de toestanden

$$\underline{\delta}(q_1,J) \text{ met } J \in \Gamma_{q_1},$$

niet meer onderscheiden kunnen worden van de opvolgers van q_2, de toestanden

$$\underline{\delta}(q_2,J) \text{ met } J \in \Gamma_{q_2}.$$

Komen er onder deze opvolgertoestanden paren voor (bij dezelfde ingangsrij J) met een verschillend uitgangssignaal, dan leidt dit tot een conflict.
Voor de opvolgers van q_1 en q_2 onder $x = 0$ geldt:

$$\delta(q_1,0) = q_1 \quad \text{en} \quad \delta(q_2,0) = q_3$$

maar

$$\lambda(q_1,0) \neq \lambda(q_3,0).$$

De toestanden q_1 en q_2 zijn dus incompatibel. Hetzelfde blijkt te gelden voor de toestanden q_3 en q_4.
Uit het bovenstaande volgt een tweede voorwaarde waaraan voldaan moet zijn opdat twee toestanden kunnen worden samengevoegd:

Voorwaarde 2

Het is niet mogelijk twee toestanden q_j en q_k samen te nemen als er een ingangsrij J:

$$J \in \Gamma_{q_j} \cap \Gamma_{q_k}$$

bestaat waarvoor geldt:

$$\underline{\lambda}(q_j,J) \neq \underline{\lambda}(q_k,J).$$

Voorbeeld

q \ x	0	1	0	1
q_1	$\textcircled{q_1}$	q_2	0	0
q_2	q_3	$\textcircled{q_2}$	0	0
q_3	$\textcircled{q_3}$	q_4	0	–
q_4	q_5	$\textcircled{q_4}$	1	1
q_5	$\textcircled{q_5}$	q_6	1	1
q_6	q_1	$\textcircled{q_6}$	–	1

	q_1	q_2	q_3	q_4	q_5	q_6
q_1	·	·	×	×	×	
q_2		·	×	×	×	
q_3			·	×	×	
q_4				·	·	
q_5					·	
q_6						

tabel 7.8. Toestandstabel met uitgangsconflicten.

Op grond van het niet aanwezig zijn van uitgangsconflicten is het in principe mogelijk de toestanden q_1 en q_2 van tabel 7.8 samen te voegen. Onderzocht moet worden of dit ook geldt voor de opvolgerparen onder alle mogelijke rijen ingangscombinaties. De opvolgers van het paar (q_1,q_2) onder $x = 0$ zijn:

$$(q_1,q_2) \xrightarrow{x = 0} (\delta(q_1,0),\delta(q_2,0)) = (q_1,q_3).$$

Ook dit paar bezit geen uitgangsconflict.
Wordt vervolgens $x = 1$ als ingangscombinatie aangeboden, dan vindt men

$$(q_1,q_3) \xrightarrow{x = 1} (\delta(q_1,1),\delta(q_3,1)) = (q_2,q_4).$$

Er geldt echter

$$\lambda(q_2,1) \neq \lambda(q_4,1).$$

Uit het feit dat de opvolgers van het paar (q_1,q_2) onder de ingangsrij [01] een verschillend uitgangssymbool bezitten, volgt dat q_1 en q_2 niet gecombineerd kunnen worden, q_1 en q_2 zijn *incompatibel.*

Twee vragen dringen zich naar aanleiding van het bovenstaande op:

1. Is er een verkorte procedure mogelijk voor het bepalen van de compatibiliteit van paren toestanden, zodat niet voor alle ingangsrijen de uitgangen van de opvolgers behoeven te worden onderzocht?
2. Kan men ooit beslissen dat twee toestanden compatibel zijn? Het aantal mogelijke ingangsrijen is nl. oneindig groot.

Er bestaat een vrij eenvoudige procedure die het antwoord levert op bovenstaande

vragen. Als een paar toestanden (q_j,q_k) onder ingangsrij J afgebeeld wordt op het paar $(\underline{\delta}(q_j,J),\underline{\delta}(q_k,J)) = (q_m,q_n)$, en er is reeds bekend dat het paar (q_m,q_n) incompatibel is, dan is ook het paar (q_j,q_k) incompatibel. Dit leidt tot de suggestie om, uitgaande van een incompatibel paar, terug te werken en na te gaan welke paren erop worden afgebeeld. Deze paren zijn dan incompatibel. Voor de uitleg van de procedure gebruiken we het voorbeeld in tabel 7.8.

	q_1	q_2	q_3	q_4	q_5	q_6
q_1	·	·	$\times^1$	$\times^1$	$\times^1$	
q_2		·	$\times^1$	$\times^1$	$\times^1$	
q_3			$\times^1$	$\times^1$	·	
q_4				·	·	
q_5					·	
q_6						

tabel 7.9.

	q_1	q_2	q_3	q_4	q_5	q_6
q_1	·	$\times^2$	$\times^1$	$\times^1$	$\times^1$	
q_2		$\times^2$	$\times^1$	$\times^1$	$\times^1$	
q_3			$\times^1$	$\times^1$	·	
q_4				·	$\times^2$	
q_5					$\times^2$	
q_6						

tabel 7.10.

	q_1	q_2	q_3	q_4	q_5	q_6
q_1	$\times^3$	$\times^2$	$\times^1$	$\times^1$	$\times^1$	
q_2		$\times^2$	$\times^1$	$\times^1$	$\times^1$	
q_3			$\times^1$	$\times^1$	$\times^3$	
q_4				$\times^3$	$\times^2$	
q_5					$\times^2$	
q_6						

tabel 7.11.

Procedure

1. Ga van elk paar toestanden na of er sprake is van een uitgangsconflict. Zo ja, geef dit voor het betreffende paar aan met $\times^1$.
2. Ga van elk paar toestanden onder elke ingangscombinatie na of het paar afgebeeld wordt op een paar met een uitgangsconflict. Paren, waarvan reeds gevonden is dat zij incompatibel zijn, blijven hierbij buiten beschouwing. Geef de nieuwe conflicten aan met $\times^2$.
3. Onderzoek vervolgens alle resterende paren, daarbij gebruik makend van alle in vorige ronden gevonden incompatibiliteiten. Doe dit totdat in een bepaalde ronde geen nieuwe kruisjes worden toegevoegd. Het proces is nu ten einde. Noteer de conflicten als $\times^3$,$\times^4$, enz, afhankelijk van de ronde waarin zij gevonden zijn.

Tabel 7.9 bevat de uitgangsconflicten van toestandstabel 7.8. De kruisjes $\times^2$ welke in stap 2 van de procedure worden toegevoegd behoren bij paren toestanden waarvan een opvolgerpaar een uitgangsconflict bezit. Zie tabel 7.10. In tabel 7.10 blijven nog twee paren over, het paar (q_1,q_2) en het paar (q_4,q_5), waarvoor het onderzoek moet worden voortgezet. In de volgende (derde) ronde vallen ook deze paren af, zoals uit tabel 7.11 blijkt.

Opmerking

In het bovenstaand proces wordt het onderzoek of een paar toestanden compatibel (verenigbaar) is van achter naar voor aangepakt. Daarmee wordt bereikt dat niet alle 2^k ingangswoorden van lengte k moeten worden onderzocht wanneer een uitgangsconflict pas optreedt bij het k-de opvolgerpaar voor een bepaalde ingangsrij. Ook worden gedeeltelijk samenvallende rijen opvolgerparen van verschillende beginparen niet elk apart maar gecombineerd onderzocht.

Voorbeeld

q \ x_1x_2	00	01	10	11	U
q_1	(1)	(1)	2	3	0
q_2	1	–	(2)	–	0
q_3	–	4	–	(3)	0
q_4	(4)	(4)	(4)	5	0
q_5	–	6	–	(5)	0
q_6	(6)	(6)	(6)	7	1
q_7	–	1	–	(7)	1

q_1	q_2	q_3	q_4	q_5	q_6	q_7
q_1	$-^2$	$\times^4$	$\times^3$	$\times^2$	$\times^1$	$\times^1$
	q_2	$-^2$	$\times^4$	$-^2$	$\times^1$	$\times^1$
		q_3	$\times^3$	$\times^2$	$\times^1$	$\times^1$
			q_4	$\times^2$	$\times^1$	$\times^1$
				q_5	$\times^1$	$\times^1$
					q_6	$\times^2$
						q_7

tabel 7.12. Toestandstabel met incompatibiliteiten.

In de driehoek naast tabel 7.12 staat aangegeven in hoeveel ronden gevonden is welke toestanden met elkaar gecombineerd kunnen worden. In elke nieuwe rondgang is gebruik gemaakt van alle streepjes en kruisjes die in vorige ronden gevonden zijn. Enkele bijzonderheden van tabel 7.12 zijn:
Voor het paar (q_1,q_2) geldt dat $\lambda(q_1,i) = \lambda(q_2,i)$ voor alle ingangscombinaties $i \in \{ 00,01,10,11 \}$. De opvolgers van q_1 en q_2 zijn òf dezelfde toestand òf een der opvolgers is niet gespecificeerd. Er kan dus nooit een uitgangsconflict optreden voor het paar (q_1,q_2).
Reeds in de tweede ronde kan daarom bij tabel 7.12 voor enkele paren toestanden aangegeven worden dat zij altijd compatibel, d.w.z. combineerbaar zijn. In het algemeen kan gesteld worden dat twee toestanden zeker compatibel zijn als geldt

- $\lambda(q_j,i)$ en $\lambda(q_k,i)$ zijn niet verschillend voor alle $i \in I$.
- $\delta(q_j,i)$ en $\delta(q_k,i)$ zijn niet verschillend, of indien zij wel verschillend zijn dan is reeds gevonden dat $\delta(q_j,i)$ en $\delta(q_k,i)$ compatibel zijn. Dit moet gelden voor alle ingangscombinaties $i \in I$.

Wordt hiervan tijdens een rondgang door de driehoek gebruik gemaakt, dan kan men vaak in een vroegtijdig stadium beslissen dat toestanden compatibel zijn.

Opmerking
Bij een bepaalde rondgang mag natuurlijk reeds gebruik gemaakt worden van kruisjes of streepjes die in dezelfde ronde gevonden zijn. Voor tabel 7.12 bijvoorbeeld zou het kruisje $\times^4$ voor het paar (q_2,q_4) dan reeds in de derde ronde gevonden zijn. De aanduiding 1,2, . . . bij de kruisjes dient slechts om aan te geven in welke ronde het kruisje gezet is, en heeft verder geen enkele betekenis.

Voorbeeld

q \ x_1x_2	00	01	10	11	00	01	10	11
q_1	(1)	q_2	(1)	–	0	–	1	–
q_2	q_6	(2)	q_3	(2)	–	0	–	0
q_3	(3)	q_5	(3)	q_4	0	–	1	–
q_4	(4)	q_2	q_1	(4)	1	–	–	1
q_5	q_3	(5)	(5)	q_6	–	0	1	–
q_6	(6)	q_2	q_1	(6)	1	–	–	1

q_1	q_2	q_3	q_4	q_5	q_6
q_1	$\times^2$	$\times^3$	$\times^1$	$\times^3$	$\times^1$
	q_2	$\times^2$	$\times^1$	$\times^2$	$\times^1$
		q_3	$\times^1$	·	$\times^1$
			q_4	$\times^2$	·
				q_5	$\times^2$
					q_6

tabel 7.13. Toestandstabel met incompatibiliteiten.

In de driehoek naast tabel 7.13 is de situatie na drie rondgangen beschreven. In een vierde ronde komen er geen nieuwe kruisjes meer bij. Op grond hiervan mag besloten worden dat de paren (q_3,q_5) en (q_4,q_6) compatibel zijn.
Zodra in een bepaalde ronde geen kruisjes meer toegevoegd worden, mag besloten worden dat de overige paren compatibel zijn. Hiervoor kan een formeel bewijs geleverd worden. Gezien het doel van deze tekst zullen we ons beperken tot een korte argumentatie van bovenstaande bewering.

Als in een bepaalde ronde geen kruisjes meer toegevoegd zijn, dan is de situatie ontstaan dat een paar toestanden compatibel is mits een ander paar (eventueel hetzelfde paar) dit ook is. Voor het andere paar geldt het omgekeerde. Van alle openstaande paren geldt nu dat er geen conflicten meer onder de opvolgerparen kunnen schuilen, elk opvolgerpaar waar zich dit ook in een opvolgerrij bevindt, is onderzocht.

Toelichting
Voor tabel 7.13 geldt:
De opvolgers van het paar (q_4,q_6) zijn:

$$(q_4,q_6) \Longrightarrow \begin{cases} (q_4,q_6) \\ q_2 \\ q_1 \\ (q_4,q_6). \end{cases}$$

Dit kan nooit leiden tot een uitgangsconflict, q_4 en q_6 zijn dus compatibel.
De opvolgers van het paar (q_3,q_5) zijn:

$$(q_3,q_5) \Longrightarrow \begin{cases} q_3 \\ q_5 \\ (q_3,q_5) \\ (q_4,q_6). \end{cases}$$

Ook dit paar is nu compatibel.
Het onderzoek naar het compatibel zijn van paren toestanden loopt vrij snel af, zoals onderstaande stelling aangeeft.

Stelling
Voor een toestandstabel met k toestanden is na maximaal $\frac{1}{2}k(k-1)$ rondes door de driehoek bekend welke toestandsparen compatibel zijn.

Bewijs
In elke ronde wordt tenminste één kruisje toegevoegd. Is dit in een bepaalde ronde niet meer het geval, dan kan het proces worden gestopt omdat dan de compatibiliteiten bekend zijn. Uit het feit dat er maximaal $\frac{1}{2}k(k-1)$ kruisjes gezet kunnen worden volgt dat maximaal $\frac{1}{2}k(k-1)$ ronden nodig zijn. Hiermee is een bovengrens voor het aantal ronden afgeleid.

Notatie
Om aan te geven dat twee toestanden compatibel (is combineerbaar) zijn, wordt de volgende notatie ingevoerd:

$q_j \sim q_k$ compatibel
$q_j \not\sim q_k$ incompatibel.

7.4. Het opstellen van een gereduceerde toestandstabel

Voor toestandstabel 7.13 is in de driehoek aangegeven dat de toestandsparen (q_3,q_5) en (q_4,q_6) elk tot één nieuwe toestand kunnen worden gecombineerd. Voor de overige toestanden, te weten q_1 en q_2 bestaan geen mogelijkheden om te combineren. Met behulp van dit overzicht kan nu een gereduceerde toestandstabel worden opgesteld. Deze gereduceerde tabel moet voor elke toestand van de gegeven tabel een toestand bezitten die de functie ervan over kan nemen. In het beschreven geval bezit een minimale tabel tenminste vier toestanden:

$q_1 \longrightarrow r_1$
$q_2 \longrightarrow r_2$
$(q_3,q_5) \longrightarrow r_3$
$(q_4,q_6) \longrightarrow r_4$

Toestand q_1 in tabel 7.13 wordt afgebeeld op toestand r_1 van de gereduceerde tabel. Onder $x_1x_2 = 00$ is $\delta(q_1,00) = q_1$. Hieruit volgt dat $\delta_r(r_1,00) = r_1$ moet zijn. Ook is $\delta_r(r_1,01) = r_2$ omdat $\delta(q_1,01) = q_2$ is en toestand r_2 in de gereduceerde tabel de functie van toestand q_2 overneemt.
Als op dezelfde wijze wordt voortgegaan met het bepalen van de opvolgers van de toestanden r_1 tot en met r_4, dan wordt als *gereduceerde tabel* voor tabel 7.13 gevonden de toestandstabel 7.14.

r \ x_1x_2	00	01	10	11	00	01	10	11
r_1	(1)	2	(1)	–	0	–	1	–
r_2	4	(2)	3	(2)	–	0	–	0
r_3	(3)	(3)	(3)	4	0	0	1	–
r_4	(4)	2	1	(4)	1	–	–	1

tabel 7.14. Gereduceerde toestandstabel.

De uitgangssignalen behorend bij de toestanden r_1 tot en met r_4 in tabel 7.14 worden bepaald door na te gaan welke de uitgangssignalen zijn van de overeenkomstige toestanden in tabel 7.13. Deze uitgangswaarden worden dan overgenomen. Een uitgangswaarde 0 of 1 overheerst een don't care!

Soms is het mogelijk de functie van meer dan twee toestanden in één toestand van een gereduceerde tabel te combineren. Om de voorwaarden waaronder dit toegestaan is te kunnen bepalen, moet het begrip *gespecificeerd ingangswoord* weer worden gebruikt. Twee toestanden zijn compatibel, $q_i \sim q_j$, als voor alle ingangswoorden J:

$$J \in \Gamma_{q_i} \cap \Gamma_{q_j}$$

geldt dat

$$\underline{\lambda}(q_i,J) = \underline{\lambda}(q_j,J).$$

Hierbij zijn Γ_{q_i} en Γ_{q_j} de verzamelingen van alle ingangswoorden die voor het ingangs-uitgangsgedrag van de schakeling van belang zijn als gestart wordt in q_i resp. q_j (zie de definitie van gespecificeerd ingangswoord in par. 7.2).

Stelling
Een nodige en voldoende voorwaarde opdat drie toestanden van een toestandstabel vervangen kunnen worden door één toestand in een gereduceerde tabel is dat deze toestanden paarsgewijs compatibel zijn:

$$q_i \sim q_j\,, \quad q_j \sim q_k\,, \quad q_i \sim q_k.$$

Bewijs:
Uit het gegeven

$$q_i \sim q_j\,, \quad q_j \sim q_k \quad \text{en} \quad q_i \sim q_k$$

volgt dat het mogelijk is een nieuwe toestand q_r te definiëren met als uitgangsfunctie

$$\underline{\lambda}_r(q_r,J) = \underline{\lambda}(q_i,J) \text{ voor alle } J \in \Gamma_{q_i}$$

$$\underline{\lambda}_r(q_r,J) = \underline{\lambda}(q_j,J) \text{ voor alle } J \in \Gamma_{q_j}$$

$$\underline{\lambda}_r(q_r,J) = \underline{\lambda}(q_k,J) \text{ voor alle } J \in \Gamma_{q_k}.$$

Immers omdat $q_i \sim q_j$ is, is het uitgesloten dat er een ingangsrij $J \in \Gamma_{q_i} \cap \Gamma_{q_j}$ bestaat waarvoor $\underline{\lambda}(q_i,J) \neq \underline{\lambda}(q_j,J)$. Hetzelfde geldt voor de overige paren. De voorwaarde is dus voldoende.
Nu het nodig zijn van de voorwaarde. Dit kan het best met behulp van een voorbeeld worden aangetoond.

Voorbeeld

q \ x_1x_2	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	–	1	–	–	–
q_2	q_1	(2)	–	q_4	–	0	–	0
q_3	q_1	–	(3)	q_4	–	–	0	0
q_4	–	q_5	q_6	(4)	–	–	–	0
q_5	q_1	(5)	–	q_4	1	1	–	–
q_6	q_1	–	(6)	q_4	1	–	1	–

	q_1	q_2	q_3	q_4	q_5	q_6
q_1	–	$-^2$	$-^2$	$\times^2$	$\times^2$	$\times^2$
q_2		–	$-^2$	$\times^2$	$\times^1$	$-^2$
q_3			–	$\times^2$	$-^2$	$\times^1$
q_4				–	$-^2$	$-^2$
q_5					–	$-^2$
q_6						–

tabel 7.15. Toestandstabel met compatibiliteiten.

Voor tabel 7.15 is gegeven dat:

$$q_2 \sim q_3 \quad \text{en} \quad q_3 \sim q_5.$$

Tracht men de toestanden $\{q_2,q_3,q_5\}$ op één nieuwe toestand af te beelden, dan lukt dit niet omdat $\underline{\lambda}(q_2,01) \neq \underline{\lambda}(q_5,01)$ is.

In het algemeen:
Een groep toestanden van een toestandstabel kan op één toestand van een (gereduceerde) tabel worden afgebeeld als de toestanden van deze groep twee aan twee compatibel zijn. Een verzameling paarsgewijs compatibele toestanden, waaraan geen toestand kan worden toegevoegd die paarsgewijs compatibel is met alle reeds in de verzameling opgenomen toestanden, wordt een *maximale compatibele klasse* van de toestandsverzameling genoemd. Voor tabel 7.15 is de verzameling ζ, bestaande uit alle maximale compatibele klassen, gelijk aan:

$$\zeta = \{\{q_1,q_2,q_3\},\{q_2,q_6\},\{q_3,q_5\},\{q_4,q_5,q_6\}\}.$$

Nu bekend is onder welke voorwaarden toestanden kunnen worden gecombineerd, rijst de vraag of het ook altijd toegestaan is zoveel mogelijk toestanden te combineren. Voor tabel 7.15 kan een minimale tabel worden gedefinieerd met behulp van twee maximale compatibele klassen $\{q_1,q_2,q_3\}$ en $\{q_4,q_5,q_6\}$. Ga dit na!

In het algemeen is het probleem wat lastiger. In feite is hierboven slechts één van de twee voorwaarden onderzocht, waaronder een toestandstabel een tweede tabel kan vervangen, nl. voor elke toestand in de gegeven tabel moet een toestand in de andere tabel kunnen worden aangewezen, die de functie ervan kan vervangen. *We drukken dit uit door te zeggen dat de toestandsverzameling van een (gereduceerde) tabel een dekking moet zijn van de toestandsverzameling van de gegeven tabel.*
Er is echter een tweede eis, die meer problemen geeft. Als de functies van een aantal toestanden (compatibele klassen) van een toestandstabel in één toestand van een (gereduceerde) tabel worden gecombineerd, dan houdt dit automatisch in dat hetzelfde moet gebeuren voor alle opvolgerverzamelingen van deze compatibele klasse. *De dekking moet gesloten zijn.*

Voorbeeld

q \ i	i_1	i_2	i_3	i_1	i_2	i_3
q_1	(1)	q_3	q_5	0	0	–
q_2	q_1	q_4	(2)	0	0	1
q_3	q_5	(3)	q_2	1	1	0
q_4	q_6	(4)	q_2	1	1	0
q_5	(5)	q_4	(5)	0	–	1
q_6	(6)	–	(6)	0	1	1

	q_1	q_2	q_3	q_4	q_5	q_6
q_1		$-^2$	$\times^1$	$\times^1$	$-^2$	$\times^1$
q_2			$\times^1$	$\times^1$	$-^2$	$\times^1$
q_3				$-^2$	$\times^1$	$\times^1$
q_4					$\times^1$	$\times^1$
q_5						$-^2$
q_6						

tabel 7.16. Toestandstabel met incompatibiliteiten.

Voor tabel 7.16 is de verzameling der maximale compatibele klassen

$$\zeta = \{\{q_1,q_2,q_5\},\{q_3,q_4\},\{q_5,q_6\}\}.$$

Een dekking zou zijn:

$$\{\{q_1,q_2\},\{q_3,q_4\},\{q_5,q_6\}\}.$$

Het definiëren van de opvolgers van de toestanden $r_2 = \{q_3,q_4\}$ en $r_3 = \{q_5,q_6\}$ geeft geen problemen. De opvolgerverzameling van de verzameling $\{q_1,q_2\}$ onder ingang i_3 is de verzameling $\{q_2,q_5\}$. Deze verzameling is op geen enkele andere toestand van de gereduceerde tabel afgebeeld.

Conclusie

De verzameling $\{q_1,q_2\}$ moet uitgebreid worden tot de verzameling $\{q_1,q_2,q_5\}$. De opvolgers hiervan zijn wel te definiëren. Een minimale gesloten dekking voor tabel 7.16 is:

$$C = \{\{q_1,q_2,q_5\},\{q_3,q_4\},\{q_5,q_6\}\}.$$

Hieruit is een minimale tabel met drie toestanden af te leiden.

Er bestaan systematische methoden om toestandstabellen te reduceren. Deze worden behandeld in de theorie der sequentiële machines. Aan het eind van dit hoofdstuk volgt een literatuuropgave. Uit nog te bespreken voorbeelden zal blijken dat er in de regel voor een bepaalde toestandstabel meer dan één minimale toestandstabel kan worden opgesteld.

7.5. Het coderen van de inwendige toestanden

De inwendige toestanden, zoals deze tot nu toe ter sprake zijn gekomen, representeren elk een zekere hoeveelheid informatie die het de schakeling mogelijk maakt te beslissen welke uitgang gegenereerd moet worden na een ingangsverandering. Deze informatie moet in de schakeling vastgelegd worden. Dit kan geschieden door af te spreken welke combinatie van standen van de geheugenelementen correspondeert met iedere inwendige toestand. Voor een toestandstabel met slechts twee inwendige toestanden is dit geen probleem. Een van de toestanden correspondeert met de 0-stand van het geheugenelement, de andere toestand met de 1-stand.

In het algemeen is het niet van belang welke van de twee toestanden als 0 en welke als 1 gecodeerd wordt. Bij trekkers heeft men, mits men afziet van het gebruik van de ingangscombinatie SR = 11, de inverse uitgang altijd beschikbaar. Bij relais betekent het verwisselen van de toestandscode van $0 \rightarrow 1$ en andersom veelal niet meer dan het vervangen van maakkontakten door verbreekkontakten en omgekeerd.
Komen in de toestandstabel meer dan twee toestanden voor, dan kunnen er voor sommige coderingen problemen ontstaan. Met deze problemen is reeds kennis gemaakt in par. 7.1 bij het coderen van de toestandstabel van de tweedeler. We zullen het coderingsprobleem wat nader onderzoeken.

Wat is de oorzaak van het coderingsprobleem?

Deze moet worden gezocht in het feit dat bij bepaalde toestandsovergangen tussentoestanden kunnen optreden die bij de gegeven ingangscombinatie naar verschillende stabiele eindtoestanden (kunnen) leiden. Specificeert de toestandstabel voor alle tussentoestanden echter dezelfde eindtoestand (de gewenste toestand), dan is het duidelijk dat er geen problemen bestaan.
Fig. 7.8 geeft een toelichting. In deze figuur zijn enkele fragmenten van gecodeerde toestandsdiagrammen getekend.

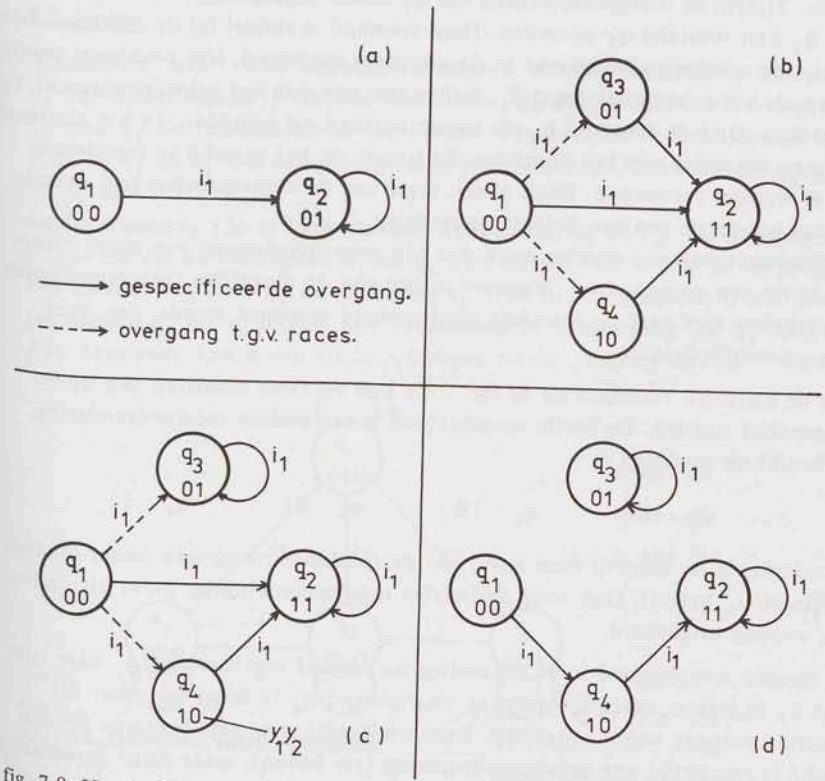


fig. 7.8. Voorbeelden van toestandsovergangen.

De toestandsovergang in fig. 7.8.a verloopt zonder problemen. Alleen het geheugenelement Y_2 verandert van $0 \rightarrow 1$, de overgang is een zg. *directe toestandsovergang* waarbij één toestandsvariabele verandert. In het algemeen:

> Iedere codering van de toestanden in een toestandsdiagram, waarvoor de codering van elk tweetal toestanden waartussen een overgang kan plaatsvinden in slechts één toestandsvariabele verschilt, is geschikt en geeft geen aanleiding tot moeilijkheden.

Een *toestandscodering (state assignment)* waarbij in elke toestandsovergang slechts één van de toestandsvariabelen verandert, wordt een *progressieve codering* genoemd. Voor een aantal diagrammen is een progressieve codering direct op te stellen. Een voorbeeld is de codering van het tweedelerdiagram in fig. 7.2.b. Voor andere diagrammen, bijvoorbeeld sommige met drie toestanden, is geen progressieve codering mogelijk. We onderzoeken nu eerst wanneer een toestandscodering aanleiding kan geven tot moeilijkheden.
Bij de toestandscodering in fig. 7.8.b verloopt de overgang van toestand q_1 naar toestand q_2 òf direct, òf gaat via de tussentoestanden q_3 of q_4. Welke weg de schakeling kiest, hangt af van het verschil in reactiesnelheid van de geheugenelementen Y_1 en Y_2. Volgens de specificatie in fig. 7.8.b gaat elk van de toestanden q_1, q_3 en q_4 onder ingangscombinatie i_1 naar toestand q_2. De gewenste eindtoestand wordt altijd bereikt.
De toestandscodering in fig. 7.8.c geeft aanleiding tot een potentieel gevaarlijke situatie. Tijdens de toestandsovergang van q_1 onder ingangscombinatie i_1 naar toestand q_2 kan toestand q_3 optreden. Deze toestand is stabiel bij de ingangscombinatie i_1, de schakeling komt niet in de gewenste eindstand. Het probleem treedt niet op als het geheugenelement Y_1 sneller zou zijn dan het geheugenelement Y_2 omdat dan altijd de toestand q_4 als tussentoestand zal optreden. In het algemeen dient een oplossing van het probleem die berust op het verschil in reactiesnelheid te worden verworpen. Denk alleen maar aan de consequenties met betrekking tot het vervangen van defecte onderdelen.
Een toestandsovergang waarbij meer dan één geheugenelement van stand verandert, bevat een *raceconditie.* Wanneer in een van de mogelijke tussentoestanden de schakeling niet naar de gewenste eindtoestand gestuurd wordt, dan heet de raceconditie *kritisch.*

Voor de kritische raceconditie in fig. 7.8.c kan op twee manieren een oplossing gezocht worden. De eerste mogelijkheid is een andere toestandscodering. Een bruikbare codering is

$$q_1 : 00 \qquad q_2 : 10 \qquad q_3 : 01 \qquad q_4 : 11$$

De codering is nu racevrij voor zover het de toestandsovergangen onder ingangscombinatie i_1 betreft. Ook voor de overige ingangscombinaties moet dit onderzoek worden uitgevoerd.

Een tweede mogelijkheid is de schakeling niet direct van toestand q_1 naar toestand q_2 te leiden, maar de overgang via toestand q_4 te laten verlopen. De toestandsovergang wordt *omgelegd.* Deze omlegging mag niet berusten op een verschil in reactietijd van geheugenelementen (zie boven), maar dient gerealiseerd te worden via de logische functie van de sturende poortschakelingen. Hoe dit uitwerkt in de toestandsvergelijkingen zien we hierna.

Voorbeeld

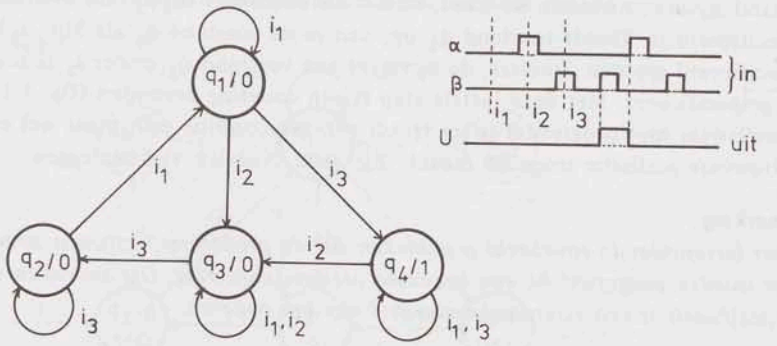


fig. 7.9. Level mode gespecificeerde schakeling.

De schakeling, waarvan fig. 7.9 de werking specificeert, heeft twee ingangssignalen α en β die niet tegelijk 1 zijn. De schakeling moet een alarm (uitgang is 1) afgeven indien twee of meer keren achtereen een β-puls wordt ontvangen zonder dat tussen deze twee pulsen een α-puls is ontvangen. De α-puls heeft de functie van een resetsignaal. Ga na hoe fig. 7.9 de gewenste werking specificeert.

Codering

Toestand q_2 gaat onder ingangscombinatie i_1 over in toestand q_1. Zowel q_3 als q_4 zijn onder ingang i_1 stabiele toestanden. Zij mogen niet in de overgang van q_2 naar q_1 als tussentoestand voorkomen. Dit is alleen te vermijden als de toestanden q_2 en q_1 onderling progressief gecodeerd zijn, bijvoorbeeld q_2 als 00 en q_1 als 01.
Onder ingang i_3 zijn er twee stabiele toestanden, q_2 en q_4. Dit gegeven leidt tot de eis dat de toestanden q_1 en q_4 niet mogen voorkomen in de overgang van toestand q_3 naar q_2 onder ingang i_3. Ook het omgekeerde is het geval, de toestanden q_2 en q_3 mogen niet voorkomen in de overgang van q_1 naar q_4. Aan deze eisen kan alleen voldaan worden bij de codering van fig. 7.10.

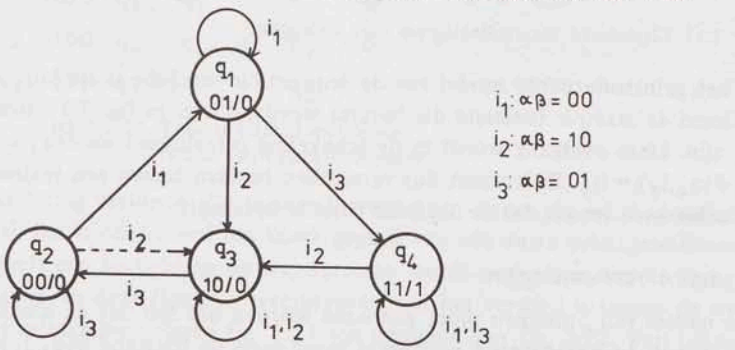


fig. 7.10. Gecodeerd toestandsdiagram.

Bij de codering in fig. 7.10 is rekening gehouden met de eisen die voor de ingangscombinatie i_1 reeds gesteld zijn. Tot slot moet nu onderzocht worden of er onder ingangscombinatie i_2 problemen optreden. Deze kunnen inderdaad optreden omdat de overgang van toestand q_1 naar toestand q_3 een raceconditie is. Als tussen-

toestanden kunnen de toestanden q_4 en q_2 optreden. Bij het optreden van tussentoestand q_4 is er niets aan de hand, omdat als opvolger $\delta(q_4,i_2)$ de toestand q_3 gespecificeerd is. Treedt toestand q_2 op, dan moet toestand q_3 als $\delta(q_2,i_2)$ gespecificeerd worden. Immers, de opvolger van toestand q_2 onder i_2 is nog niet gespecificeerd. Met deze laatste stap is een codering gevonden (fig. 7.10) die weliswaar niet progressief is (er treedt een raceconditie op), maar wel een betrouwbare realisatie mogelijk maakt. Zie ook: Nadelen van omleggen.

Opmerking

In het bovenstaande voorbeeld is gebleken dat de probleemspecificatie soms moet worden aangevuld bij een bepaalde toestandscodering. Dit aanvullen van de specificatie is een essentieel onderdeel van het coderen.

Voorbeeld

Neem aan dat voor $\delta(q_2,i_2)$ in fig. 7.9 toestand q_2 zelf gespecificeerd is. De extra specificatie $\delta(q_2,i_2) = q_3$, welke nodig is voor de raceconditie tussen q_1 en q_3, is dan niet mogelijk. Bij het gegeven diagram is er ook geen andere codering mogelijk die voldoet aan de eisen welke onder de ingangscombinaties i_1 en i_3 gesteld worden. Tenslotte kan men nog trachten het probleem door *omleggen* op te lossen. Het is een gelukkig toeval dat dit hier kan. Het resultaat staat in fig. 7.11.

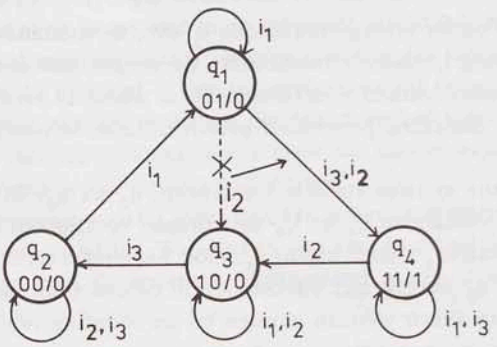


fig. 7.11. Gecodeerd toestandsdiagram met omlegging.

In het geïntroduceerde model van de sequentiële machine is als $\delta(q_1,i_2)$ gedefinieerd de *stabiele* toestand die bereikt wordt, d.w.z. in fig. 7.11 zou $\delta(q_1,i_2) = q_3$ zijn. Deze overgang wordt in de schakeling gerealiseerd als $\delta(q_1,i_2) = q_4$ en $\delta(q_4,i_2) = q_3$. Er kunnen dus verschillen bestaan tussen een realisatie en het bijbehorende model dat de logische functie beschrijft.

Nadelen van omleggen

Als nadeel van omleggen moet genoemd worden het feit dat de schakeling trager wordt. In bovenstaand voorbeeld moet eerst de toestand $\delta(q_1,i_2) = q_4$ bereikt worden, voordat de overgang van q_4 naar q_3 plaats vindt. Globaal is de reactietijd twee maal zo groot als bij een directe toestandsovergang. Een ander nadeel (dat overigens ook kan gelden voor elke niet-progressieve codering waarbij tussentoestanden optreden) is dat er ongewenste spanningspieken in de uitgang kunnen optreden. Voor fig. 7.11 is $\lambda(q_1) = 0$ en $\lambda(q_3) = 0$ (zie fig. 7.9), maar $\lambda(q_4) = 1$. Tijdens de overgang van q_1 naar q_3 via q_4 kan er dus een *uitgangs-*

spike ($0 \to 1 \to 0$) optreden. Soms is een andere omlegging mogelijk via een toestand met dezelfde uitgangswaarde. Deze verdient dan de voorkeur.

Voorbeeld

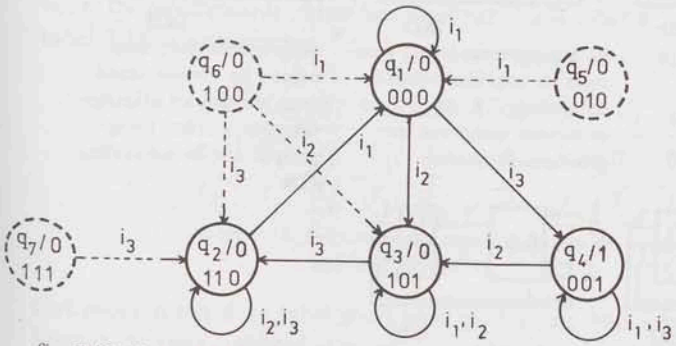


fig. 7.12. Toestandsdiagram met Tracey-codering.

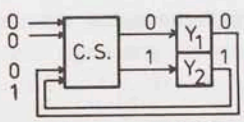
In de literatuur zijn verschillende methoden bekend om tot een codering te komen, zonder kritische racecondities, waarbij alle overgangen direct zijn. Een ervan is van Tracey afkomstig. Fig. 7.12 is volgens deze methode gecodeerd. Voor de methode zelf wordt verwezen naar hoofdstuk 20. Tijdens een toestandsovergang tussen twee toestanden kunnen tussentoestanden ontstaan die extra gespecificeerd moeten worden. Zie tabel 7.17. De Tracey-methode waarborgt dat dit altijd kan zonder dat kritische races optreden.

	q \ i	i_1	i_2	i_3	u
000	q_1	(q1)	q_3	q_4	0
110	q_2	q_1	(q2)	(q2)	0
101	q_3	(q3)	(q3)	q_2	0
001	q_4	(q4)	q_3	(q4)	1
010	q_5	q_1	–	–	0
100	q_6	q_1	q_3	q_2	0
111	q_7	–	–	q_2	0

tabel 7.17. Toestandstabel met specificatie van tussentoestanden.

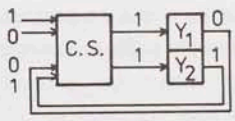
Met deze codering verlopen alle toestandsovergangen direct, de schakelsnelheid is maximaal. Dit is echter wel ten koste gegaan van een extra geheugenelement. Fig. 7.13 geeft een toelichting op wat verstaan wordt onder een directe toestandsovergang. In deze figuur is weergegeven wat het verschil is tussen de werking van de schakeling volgens fig. 7.11 ten opzichte van fig. 7.12. Van beide diagrammen is de overgang van q_1 naar q_3 beschreven bij de ingangsverandering $i_1 \to i_2$.

De eerste vijf situaties beschrijven de toestandsovergang via de omlegging in fig. 7.11. Kenmerkend voor een omlegging is dat de combinatorische schakeling C.S. meer dan één maal een nieuw stel uitgangswaarden genereert, terwijl ook het geheugendeel verschillende keren verandert.



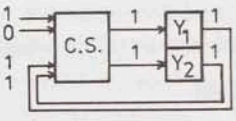
(a1)

stabiele situatie in toestand q_1 onder i_1



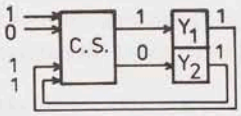
(a2)

na ingangsverandering genereert de combinatorische schakeling C.S. de gewenste nieuwe stand van de geheugenelementen



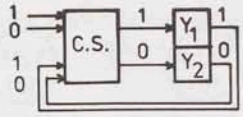
(a3)

de geheugenelementen nemen de nieuwe stand over, de nieuwe uitgangswaarden worden teruggemeld aan de schakeling C.S.



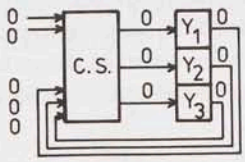
(a4)

de C.S. genereert de gewenste nieuwe stand van de geheugenelementen



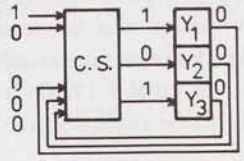
(a5)

de nieuwe stand wordt overgenomen en er ontstaat een nieuwe stabiele situatie



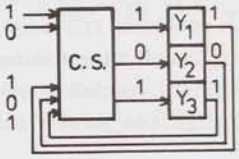
(b1)

stabiele situatie in toestand q_1 (000) onder ingang i_1.



(b2)

na de ingangsverandering genereert de C.S. de gewenste nieuwe stand van de geheugenelementen



(b3)

de geheugenelementen nemen de nieuwe stand over, de C.S. krijgt deze nieuwe stand aangeboden maar genereert *geen* nieuwe uitgangswaarden.

fig. 7.13. Directe en indirecte toestandsovergang.

De laatste drie situaties beschrijven de directe overgang volgens de *Tracey-codering*. De combinatorische schakeling C.S. en het geheugendeel veranderen slechts één maal, ongeacht het feit dat tijdens de toestandsovergang de ingangen van de combinatorische schakeling C.S. kunnen veranderen. Het spreekt vanzelf dat de schakeling C.S. hierbij geen spikes aan de uitgang mag hebben.

7.6. Keuze der bouwstenen en realisatie van het ontwerp

Nadat een toestandscodering is toegewezen kan men overgaan tot de realisatie van de schakeling. Een eerste aspect van de realisatiefase is de keuze der bouwstenen. Globaal beperkt de keuze zich tot pneumatisch, elektromechanisch of elektronisch werkende bouwstenen. Vooral in de huidige series van geïntegreerde circuits is een grote verscheidenheid van bouwstenen verkrijgbaar. In de voorbeelden zullen we ons beperken tot relais en geïntegreerde poortschakelingen.

Voorbeeld

In hoofdstuk 6 is het probleem van de binaire *sample-and-hold* schakeling in een toestandstabel geformuleerd. Tabel 6.16 is de primitieve toestandstabel hiervoor. De gereduceerde versie van deze tabel was tabel 6.17, die hieronder als tabel 7.18 is opgenomen.

r \ ga	00	01	10	11	00	01	10	11
r_1	(1)	r_2	(1)	(1)	0	0	0	0
r_2	r_1	(2)	(2)	(2)	0	0	1	1

tabel 7.18. Gereduceerde specificatie van de sample-and-hold schakeling.

Het coderen van deze tabel geeft geen problemen. Met één geheugenelement Y kunnen de twee toestanden r_1 en r_2 worden gecodeerd. Zie tabel 7.19.

y \ ga	00	01	10	11	00	01	10	11
0	(0)	1	(0)	(0)	0	0	0	0
1	0	(1)	(1)	(1)	0	0	1	1

tabel 7.19. Gecodeerde toestandstabel.

Uit de gecodeerde toestandstabel volgen de Karnaughdiagrammen in fig. 7.14. Het eerste diagram legt de gewenste nieuwe toestand Y van het geheugenelement vast, die volgt op de huidige toestand y en de aangeboden ingangscombinatie. Het tweede diagram specificeert de uitgangsfunctie.

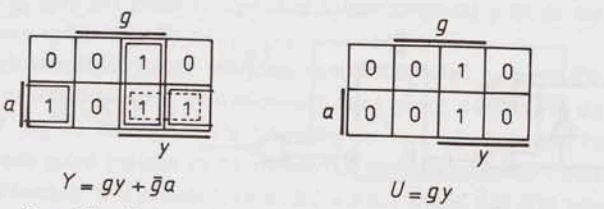


fig. 7.14. Karnaughdiagrammen.

Realisatie met relais

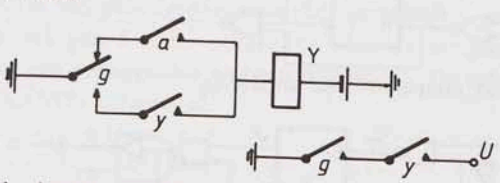


fig. 7.15. Relaisoplossing voor sample-and-hold schakeling.

In fig. 7.15 is de schakeling getekend zoals deze volgt uit de formules in fig. 7.14. Men komt in de verleiding de term gy in de formule voor Y ook te benutten voor het uitgangssignaal U. Dit blijkt niet zonder meer mogelijk te zijn, tenzij een relais met twee wikkelingen wordt gebruikt. Ga dit na!
Ook via de toepassing van een diode kan dit worden toegestaan.

Ontwerpverificatie

De schakeling in fig. 7.15 is tot op zekere hoogte kritisch. Als de omslagtijd van het kontakt g in de buurt ligt van de opkom- en afvalvertraging van het relais Y, dan is het mogelijk dat dit relais afvalt tijdens het omschakelen van g. Zie ook par. 3.5. Er zijn twee oplossingen mogelijk:

1. Het wisselkontakt g is van het maak-voor-verbreektype.
2. Het omschakelen van g overbruggen door een extra term ay in de formule voor Y. Zie fig. 7.14.

Uit dit voorbeeld blijkt wel dat de ontwerpverificatie een essentieel onderdeel is van het ontwerpproces. De reden is het niet-ideaal gedrag van de bouwstenen.

Realisatie met trekkers

Met trekkers als geheugenelement moeten formules voor de ingangssignalen S en R worden opgesteld. Deze volgen uit fig. 7.16.

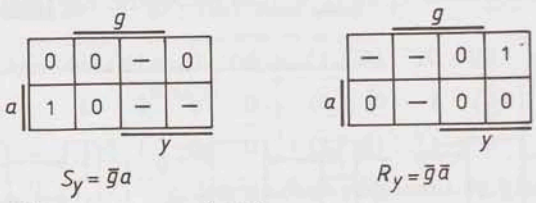


fig. 7.16. Karnaughdiagrammen voor S en R.

De diagrammen in fig. 7.16 volgen uit fig. 7.14 met behulp van tabellen 7.4 en 7.5. In fig. 7.17 staan twee realisaties met NOR's als bouwstenen. Een oplossing met NAND's staat in fig. 7.18. Ook de trekkers zijn opgebouwd uit NOR's resp. NAND's.

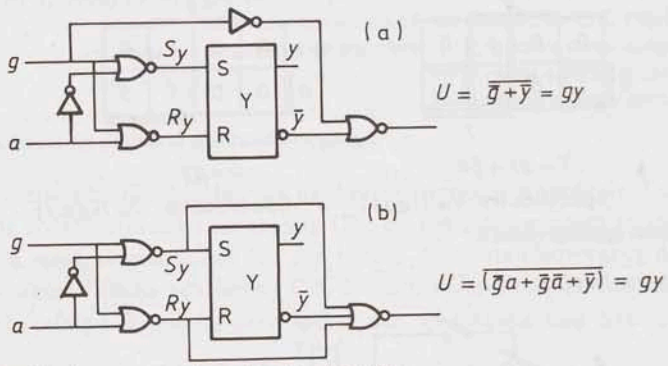


fig. 7.17. Realisaties van sample-and-hold schakeling.

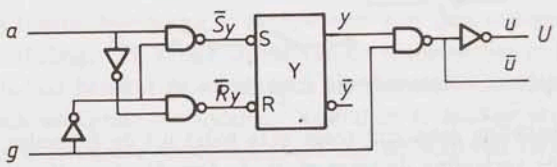


fig. 7.18. Realisatie van sample-and-hold schakeling.

In de schakeling volgens fig. 7.17.b is gebruik gemaakt van het feit dat de schakelwet $(a + \bar{a}) = 1$

$$\bar{g}a + \bar{g}\bar{a} = \bar{g}(a + \bar{a}) = \bar{g}$$

oplevert. Hiermee kan één invertor bespaard worden, de uitgangspoort heeft dan één ingang meer.
Wederom komt nu de *ontwerpverificatie* aan bod. Stilzwijgend is bij het sample-and-hold probleem verondersteld dat g en a niet tegelijk veranderen. De schakelingen in fig. 7.17.a en 7.18 zijn betrouwbaar als poortvertragingen in rekening worden gebracht. De verificatie van deze bewering wordt aan de lezer overgelaten. De oplossing in fig. 7.17.b verdient een nadere beschouwing. In deze schakeling is gebruik gemaakt van het feit dat $a + \bar{a} = 1$ wordt verondersteld. Nu is dit in de gegeven schakeling niet altijd waar. Bij een vaste poortvertraging Δt verlopen de signalen S_y en R_y bij verandering van a zoals in fig. 7.19 is aangegeven.

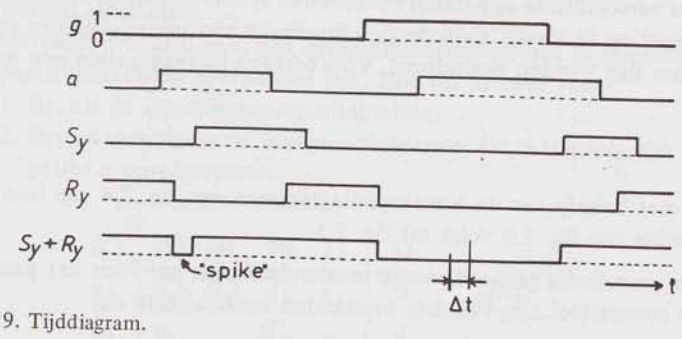


fig. 7.19. Tijddiagram.

Het blijkt dat als $g = 0$ is er een ongewenste spike zit in het signaal $S_y + R_y = \bar{g}(a + \bar{a}) = \bar{g}$. De vraag is nu of deze spike ook in de uitgang U doorwerkt. Dit blijkt gelukkig niet het geval te zijn, het uitgangssignaal $\bar{y}$ in de uitgangspoort overbrugt deze spike. Ga dit na!
In de praktijk wordt de juiste werking van een schakeling geverifieerd door middel van een proefschakeling. Functioneert deze goed, dan wordt de schakeling geaccepteerd. Bij de huidige snelle bouwstenen met reactietijden van enkele nanoseconden is een goed inzicht in de mogelijke storingsbronnen essentieel om de schakeling afdoende te kunnen testen. Bij schakelingen die zijn opgebouwd uit trekkers en combinatorische schakelingen die de signalen S en R realiseren, is een grondige analyse met betrekking tot spikes en andere gevolgen van vertragingen als regel gemakkelijker uitvoerbaar. Schakelingen waarin de geheugenfunctie in de vorm van geheugenlussen wordt gerealiseerd geven meestal meer problemen. Zie ook par. 5.6. De schakeling voor de sample-and-hold functie kan ook met een geheugenlus worden uitgevoerd. De schakeling staat in fig. 7.20, met NAND's uitgevoerd.

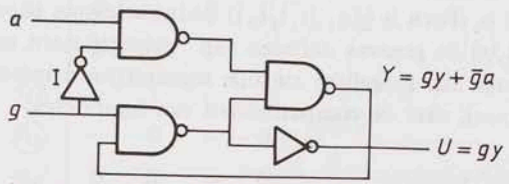


fig. 7.20. Realisatie van sample-and-hold schakeling.

Deze schakeling is niet kritisch. Zou de schakeling echter als ingangssignaal het signaal $\bar{g}$ aangeboden krijgen, dan moet de invertor worden omgekeerd. Het sig-

naal g wordt dan vertraagd t.o.v. het signaal $\bar{g}$. Hierdoor wordt de schakeling kritisch. Een oplossing wordt gevonden door op het eerste niveau een NAND met ingangen a en y toe te voegen. Vergelijk de vorenstaande relaisoplossingen van hetzelfde probleem en ook par. 5.6.

Met dit voorbeeld is een eind gekomen aan de behandeling van via de level mode gespecificeerde schakelingen. Dit onderwerp is daarmee niet uitputtend behandeld, maar de gegeven methodes en oplossingsschema's zijn voor de ontwerper meestal voldoende om de optredende problemen bij relatief kleine level mode gespecificeerde schakelingen te kunnen oplossen. Deze problemen treden enerzijds op bij het ontwerpen van bouwstenen voor een relatief ingewikkelde logische functie (meester-en-slaaf geheugenelement en andere bouwstenen), maar ook de bij het koppelen van verschillende apparaten optredende synchronisatieproblemen zijn er effectief mee op te lossen. Ook de synchronisatie van ingangssignalen en interruptsignalen kan worden bestudeerd. Voorbeelden hiervan zullen nog worden behandeld.

Opgaven

7.1. Toon met behulp van de Karnaughdiagrammen van fig. 7.4 aan hoe de schakeling van fig. 7.6 volgt uit fig. 7.5.

7.2. Bij een onvolledig gespecificeerde toestandstabel is het voor het paarsgewijs compatibel zijn van drie toestanden noodzakelijk dat

$$q_a \sim q_b,\ q_b \sim q_c \text{ en } q_a \sim q_c \implies \{q_a,q_b,q_c\} \text{ is een compa-}$$

tibele klasse.
Bij een volledig gespecificeerde toestandstabel is het voldoende dat

$$q_a \sim q_b \text{ en } q_b \sim q_c \implies \{q_a,q_b,q_c\} \text{ is een compatibele klasse.}$$

Verklaar dit verschil in eisen tussen een volledig en een niet-volledig gespecificeerde toestandstabel.

7.3. In een toestandstabel zijn onder de ingangscombinatie i_3 alle opvolgertoestanden gespecificeerd en bovendien aan elkaar gelijk, nl. de toestand q_k.
Bovendien geldt dat:

$$\delta(q_i,i_1) = q_j \quad \text{en} \quad \delta(q_j,i_2) = -.$$

Volgens de in par. 7.2 gegeven definitie is de ingangsrij $[i_1 i_2 i_3]$ geen gespecificeerd ingangswoord voor de toestand q_i omdat $\underline{\delta}(q_i,[i_1 i_2])$ niet gespecificeerd is. Toch is $\underline{\delta}(q_1,[i_1 i_2 i_3])$ ondubbelzinnig bepaald.
Toon aan dat bij de gegeven definitie van "gespecificeerd ingangswoord" desondanks toch alle mogelijke zinvolle ingangsrijen worden beschouwd bij het onderzoek naar de compatibiliteit van toestanden.

7.4.

q \ i	i_1	i_2	i_3	i_1	i_2	i_3
q_1	(1)	q_3	–	0	–	–
q_2	(2)	q_5	(2)	0	–	0
q_3	q_1	(3)	q_4	–	1	–
q_4	q_2	q_3	(4)	0	–	0
q_5	q_1	(5)	(5)	–	1	1
q_6	(6)	q_5	(6)	1	1	1

In de toestandstabel is de level mode specificatie gegeven van een te ontwerpen schakeling.
Gevraagd wordt d.m.v. een compatibiliteitsdriehoek te bepalen welke paren toestanden compatibel zijn. Doe dit in twee fasen:
1. Bepaal de uitgangsincompatibiliteiten.
2. Bepaal vervolgens de incompatibiliteiten die het gevolg zijn van incompatibele opvolgerparen.

7.5.

q \ ab	00	01	10	11	U
q_1	(1)	(1)	q_2	q_3	0
q_2	q_1	–	(2)	–	0
q_3	–	q_4	–	(3)	0
q_4	(4)	(4)	(4)	q_5	0
q_5	–	q_6	–	(5)	0
q_6	(6)	(6)	(6)	q_7	1
q_7	–	q_1	–	(7)	1

Geef in een compatibiliteitsdriehoek weer welke paren toestanden compatibel zijn.

7.6.

q \ ab	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	–	0	–	0	–
q_2	q_4	(2)	–	q_5	1	1	–	1
q_3	q_1	–	(3)	q_6	0	–	0	0
q_4	(4)	q_7	q_8	–	1	–	1	–
q_5	–	q_2	q_3	(5)	–	1	–	1
q_6	–	q_7	q_8	(6)	–	0	–	0
q_7	q_1	(7)	–	q_6	0	0	–	0
q_8	q_4	–	(8)	q_5	1	–	1	1

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
q_1	×	–	×	–	×	×	×	
q_2		×	×	–	×	×	–	
q_3			×	×	×	–	×	
q_4				×	–	×	–	
q_5					×	×	×	
q_6						×	–	×
q_7							×	
q_8								

Van de gegeven toestandstabel is bekend welke paren toestanden compatibel zijn.
Bepaal een gereduceerde versie van deze tabel met zo weinig mogelijk toestanden.

7.7.

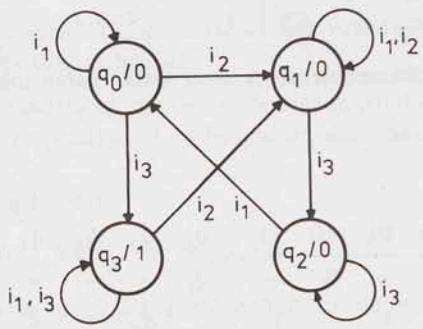
q \ ab	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	–	0	0	0	–
q_2	q_1	(2)	–	q_4	0	0	–	0
q_3	q_1	–	(3)	–	0	–	0	–
q_4	–	q_5	–	(4)	–	–	–	0
q_5	q_6	(5)	–	q_7	1	1	–	1
q_6	(6)	q_5	q_8	–	1	1	1	–
q_7	–	q_5	–	(7)	–	1	–	1
q_8	q_1	–	(8)	–	–	–	1	–

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8
q_1	–	–	×	×	×	×	×	
q_2		–	×	×	×	×	–	
q_3			–	×	×	–	×	
q_4				–	×	–	×	
q_5					–	–	×	
q_6						–	×	
q_7							–	
q_8								

Van de gegeven toestandstabel is bekend welke paren toestanden compatibel zijn.

a. Bepaal uit de driehoek de verzameling der maximale compatibele klassen.
b. Bepaal hieruit een of meer gereduceerde versies van de gegeven toestandstabel.

7.8.



Het gegeven toestandsdiagram specificeert de werking van een te ontwerpen schakeling via de level mode.
Toon aan dat een toestandscodering met twee toestandsvariabelen mogelijk is, waarbij er geen kritische racecondities optreden.

7.9.

q \ ab	00	01	10	11	U
q_1	(q_1)	q_4	(q_1)	q_3	0
q_2	q_1	q_3	(q_2)	–	0
q_3	q_1	(q_3)	q_2	(q_3)	1
q_4	(q_4)	(q_4)	q_1	(q_4)	1

De gegeven toestandstabel beschrijft de werking van een via de level mode gespecificeerde sequentiële schakeling. De uitgang wordt direct bepaald door de huidige inwendige toestand.

a. Toon aan dat een toestandscodering met twee toestandsvariabelen voor een ongewijzigde tabel niet mogelijk is.

b. Is het mogelijk via een modificatie van de tabel een geldige toestandscodering met twee toestandsvariabelen te vinden? Zo ja, welke? Uiteraard moet hierbij dezelfde uitwendige werking worden gerealiseerd.

7.10.

y_1 y_2 y_3		q \ ab	00	01	10	11	U
0 0 0	→	q_1	(q_1)	q_2	(q_1)	q_3	0
0 1 1	→	q_2	q_1	(q_2)	q_3	(q_2)	0
1 0 1	→	q_3	q_4	q_2	(q_3)	(q_3)	1
1 1 0	→	q_4	(q_4)	(q_4)	q_3	q_2	1

In de toestandstabel is de specificatie gegeven van een schakeling, die werkt in de level mode. Een toestandscodering met drie toestandsvariabelen is gegeven.

a. Vul de specificatie aan voor de bij deze codering mogelijk optredende tussentoestanden. Bij de gegeven codering is dit mogelijk.

b. Is, eventueel via omlegging, een codering met twee toestandsvariabelen mogelijk?

7.11.

y_1y_2	r \ ab	00	01	10	11	U
0 0	r_1	(r_1)	r_2	(r_1)	r_3	1
0 1	r_2	(r_2)	(r_2)	r_1	(r_2)	1
1 0	r_3	(r_3)	(r_3)	r_1	(r_3)	0

Van de toestanden in de gegeven tabel is de toestandscodering met twee toestandsvariabelen gegeven.

Bepaal zo eenvoudig mogelijke formules van de signalen voor de Set- en Resetingang van twee trekkers Y_1 en Y_2, waarmee de schakeling wordt gerealiseerd.

7.12. In tabel 7.8 is via de level mode de werking gespecificeerd van een schakeling. Veronderstel dat het ingangssignaal x een periodiek pulsvormig signaal is.

a. Teken een tijddiagram met het ingangssignaal x en het bijbehorende

uitgangssignaal U.

b. Omschrijf de functie van de schakeling.

c. Codeer het toestandsdiagram met een progressieve toestandscodering. Baseer deze codering op de gespiegeld binaire code en houd hierbij rekening met het gewenste uitgangssignaal.

d. Bepaal de ontwerpvergelijkingen voor de Sets en Resets. Voor de toestandscodering worden trekkers gebruikt.

e. Hoeveel circuits kost de schakeling indien deze wordt gerealiseerd in NAND's?
(6 × 1, 4 × 2, 3 × 3, 2 × 4 en 1 × 8 inputs per circuit).

7.13. Als de ingangen van een schakeling dienen drie drukknoppen a, b en c, welke reeds ontdenderd zijn. Een mechanische vergrendeling zorgt ervoor dat slechts één knop tegelijk kan worden ingedrukt. Ontwerp een schakeling die aangeeft welke knop het laatst is of werd ingedrukt. (Drie-standen trekker).

a. Stel een primitieve toestandstabel op voor deze schakeling.
Codeer de uitgangswaarde als volgt:

Knop a is of werd het laatst ingedrukt ↔ U = 100
" b " " " " " ↔ U = 010
" c " " " " " ↔ U = 001.

b. Reduceer deze tabel.

c. Codeer de toestanden. Houd hierbij rekening met:
1. Racecondities.
2. De gewenste uitgangscodering.

d. Realiseer de schakeling met NAND's.

e. Is het, gezien uw oplossing, noodzakelijk de drukknoppen te ontdenderen?

7.14. Ontwerp via de level mode specificatie een schakeling met de volgende functie:
De schakeling heeft twee ingangssignalen a en b, waarvan de 0 → 1 overgangen elkaar afwisselen en niet tegelijk kunnen plaatsvinden. De 1 → 0 overgang van beide signalen kan op willekeurige tijdstippen optreden. De uitgang U moet de waarde 1 aannemen na een 0 → 1 overgang van het signaal a. U behoudt deze waarde tot de eerstvolgende 0 → 1 overgang van het signaal b. Dan wordt U weer 0. In de overige gevallen behoudt U zijn waarde.

a. Teken in een tijddiagram enkele situaties.

b. Stel een primitieve toestandstabel op die dit probleem beschrijft.

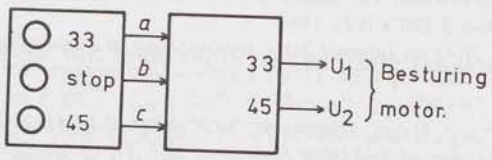
c. Geef in een driehoek aan welke paren toestanden compatibel zijn.

d. Bepaal een gereduceerde toestandstabel.

e. Bepaal een toestandscodering zonder kritische races.

f. Realiseer de schakeling met $\bar{S}$-$\bar{R}$ trekkers, aangevuld met NAND's.

7.15.

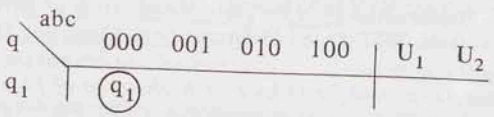


Op een pick-up zitten drie drukknoppen voor instelling van het toerental (33/45 t/m resp. Stop). Een mechanische vergrendeling zorgt ervoor dat slechts één knop tegelijk kan worden ingedrukt. Om technische redenen is het ongewenst van het ene toerental naar het andere om te schakelen anders dan via de stopstand. Omdat men bij een verkeerd ingesteld toerental de neiging heeft direct de knop voor het andere toerental in te drukken, is een vergrendeling tegen deze bedieningsfout nodig. Men besluit dit te doen via een digitale schakeling. Deze heeft drie ingangen a(33), b(stop) en c(45) en twee uitgangen U_1 en U_2, die het volgende aangeven:

$U_1U_2 = 10 \rightarrow 33$ t/m
$U_1U_2 = 00 \rightarrow$ Stop
$U_1U_2 = 01 \rightarrow 45$ t/m.

Gevraagd wordt deze schakeling te ontwerpen. Bij een bedieningsfout (a gedrukt (=1) gevolgd door c gedrukt (=1) enz.) wordt gekozen voor een continuering van de bestaande situatie.

a. Stel een primitieve toestandstabel op voor dit probleem.
Plaats de variabelen als volgt:



b. Reduceer deze tabel.
c. Codeer de toestanden. Houd hierbij rekening met de gewenste uitgangscodering.
d. Ontwerp de schakeling met NAND's.

Literatuur

1. D.B. Armstrong, e.a., *Realization of Asynchronous Sequential Circuits without Inserted Delay Elements*, IEEE Tr. on Computers, Vol. C–17, Febr. 1968, pp. 129–134.
2. S. Ginsburg, *On the Reduction of Superfluous States in a Sequential Machine*, J. ACM, Vol. 6, 1959, pp. 252–282.
3. A Grasselli and F. Luccio, *A Method for Minimizing the Number of Internal States of Incompletely Specified Sequential Networks*, IEEE Tr. on Electronic Computers, Vol. EC–14, June 1965, pp. 350–359.
4. F.J. Hill and G.R. Peterson, *Introduction to Switching Theory and Logical Design*, John Wiley and Sons Inc., New York, 1974.
5. J. Kella, *State Minimization of Incompletely Specified Sequential Machines*, IEEE Tr. on Computers, Vol. C–19, April 1970, pp. 342–348.
6. C.N. Liu, *A State Variabele Assignment Method for Asynchronous Sequential Switching Circuits*, J. ACM, Vol. 10, 1963, pp. 209–216.

7. G.H. Maley and J. Earle, *The Logic of Transistor Digital Computers,* Prentice-Hall Inc., Englewood Cliffs N.J., 1963.
8. W.S. Meisel, *A Note on Internal State Minimization in Incompletely Specified Sequential Networks*, IEEE Tr. on Electronic Computers, Vol. EC–16, Aug. 1967, pp. 508–509.
9. M.C. Paull and S.H. Unger, *Minimizing the Number of States in Incompletely Specified Sequential Switching Functions,* IRE Tr. on Electronic Computers, Vol. EC–8, Sept. 1959, pp. 356–367.
10. P.K. Sinha Roy and C.L. Sheng, *A Decomposition Method of Determining Maximum Compatibles*, IEEE Tr. on Computers, Vol. C–21, March 1972, pp. 309–312.
11. K.L. Stoffers, *Sequential Algorithm for Determining of Maximum Compatibles,* IEEE Tr. on Computers, Vol. C–23, Jan. 1974, pp. 95–98.
12. J.H. Tracey, *Internal State Assignments for Asynchronous Sequential Machines,* IEEE Tr. on Electronic Computers, Vol. EC–15, no. 4, Aug. 1966, pp. 551–560.
13. S.H. Unger, *Hazards and Delays in Asynchronous Sequential Switching Circuits,* IRE Tr. on Circuit Theory, Vol. CT–6, March 1959, pp. 12–25.

Voor een verdere bestudering van de vele aspecten van de realisatie van via de level mode gespecificeerde sequentiële schakelingen wordt verwezen naar de volgende artikelen:

14. D.D. Aufenkamp and F.E. Hohn, *Analysis of Sequential Machines,* IRE Tr. on Electronic Computers, Vol. EC–6, Dec. 1957, pp. 276–285.
15. D.D. Aufenkamp, *Analysis of Sequential Machines II,* IRE Tr. on Electronic Computers, Vol. EC–7, Dec. 1958, pp. 299–306.
16. A.D. Friedman and P.R. Menon, *Synthesis of Asynchronous Sequential Circuits with Multiple-Input Changes,* IEEE Tr. on Computers, Vol. C–17, June 1968, pp. 559–566.
17. F. Handoko, *A Discussion on Two Algorithms for Determining Maximum Compatibles,* IEEE Tr. on Computers, Vol. C–24, Aug. 1975, pp. 838–840.
18. D.E. Muller, *A Generalization of the Theory of Incompletely Specified Machines,* J. Computer and System Sciences 6, 1972, pp. 419–447.
19. M.C. Paull and G. Waldbaum, *A Note on State Minimization of Asynchronous Sequential Functions,* IEEE Tr. on Electronic Computers, Vol. EC–16, Febr. 1967, pp. 94–97
20. S.H. Unger, *A Row Assignment for Delay-Free Realizations of Flow Tables without Essential Hazards,* IEEE Tr. on Computers, Vol. C–17, Febr. 1968, pp. 146–151.
21. S.H. Unger, *Asynchronous Sequential Switching Circuits with Unrestricted Input Changes,* IEEE Tr. on Computers, Vol. C–20, Dec. 1971, pp. 1437–1444.
22. C.C. Yang, *Closure Partition Method for Minimizing Incomplete Sequential Machines,* IEEE Tr. on Computers, Vol. C–22, Dec. 1973, pp. 1109–1122.

8. FLIP-FLOP GEHEUGENELEMENTEN

8.1. Risico-analyse van level mode sequentiële schakelingen

In de hoofdstukken 6 en 7 zijn enkele aspecten behandeld van het ontwerpen van sequentiële schakelingen, waarvan de probleemspecificatie in de *level mode* is uitgevoerd. Tijdens deze behandeling zijn enkele potentiële storingsbronnen naar voren gekomen, o.a. de kritische racecondities, die het ontwerpproces en de vrijheid van de ontwerper sterk beïnvloeden. Het geïntroduceerde model van de *sequentiële machine* splitst een schakeling op in twee combinatorische schakelingen δ en λ en een geheugen Q waarin de inwendige toestand van de schakeling wordt vastgelegd. De *combinatorische schakeling* δ kent twee types ingangssignalen, nl. de signalen die de ingangscombinatie coderen (ingangsvariabelen) en de signalen die de inwendige toestand vastleggen (toestandsvariabelen). Deze laatste signalen worden gegenereerd door het geheugen Q. Fig. 8.1 geeft dit in een blokschema weer.

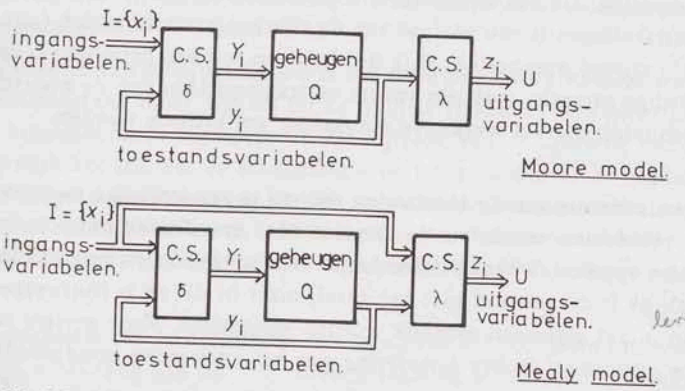


fig. 8.1. Modellen van sequentiële schakelingen.

De *uitgangsschakeling* λ is een combinatorische schakeling met één of twee groepen ingangssignalen, afhankelijk van het gebruik van het Moore of het Mealy model. We zullen nagaan in hoeverre de modellen van fig. 8.1 representatief zijn voor praktische schakelingen. Onmiddellijk moet worden opgemerkt dat het in de praktijk vaak zeer moeilijk is de schakeling precies in drie blokken δ, Q en λ op te splitsen. Een poort die deel uitmaakt van een trekker behoort volgens deze beschouwingswijze tot Q. Stuurt deze poort met zijn uitgang ook een ander geheugenelement, dan zou deze poort tot de schakeling δ behoren. Zien we nog even af van deze problemen, dan komen we nu toe aan een analyse van de risico's van een schakeling die volgens een van de in fig. 8.1 gegeven modellen is gerealiseerd.

Risico's van de schakeling δ

De combinatorische schakeling δ legt de *werkingsvoorwaarden* vast van de geheugenelementen. Door de schakeling δ worden de noodzakelijke set- en resetsignalen opgewekt welke de geheugenelementen instellen op de gewenste waarden. Zoals bij iedere combinatorische schakeling kunnen ook bij deze schakeling aan de uitgangen *spikes* optreden. Deze overgangsverschijnselen worden veroor-

zaakt door de fysische traagheid van de poorten en kunnen ontstaan als bijvoorbeeld

– een ingangssignaal de schakeling op verschillende niveaus binnentreedt,
– verschillende ingangssignalen tegelijk of nagenoeg tegelijk veranderen.

Juist het laatste punt, het veranderen van verschillende ingangssignalen tegelijk, maakt het moeilijk alle overgangsverschijnselen die tot spikes aanleiding geven te onderdrukken. Een extra complicatie van de schakeling δ is het feit dat deze schakeling ingangssignalen krijgt aangeboden uit verschillende bronnen, nl. de uitwendige ingangssignalen en de inwendige toestandssignalen.
Een gevolg van de overgangsverschijnselen aan de uitgang van de schakeling δ kan zijn dat deze schakeling onbedoelde en dus ongewenste set of reset commando's voor de geheugenelementen afgeeft. Vergelijk fig. 5.24 in par. 5.6.
Het hier kort geschetste probleem staat los van het probleem der racecondities!
Hieruit blijkt wel dat met een codering die vrij is van kritische races niet alle problemen zijn opgelost. Voor problemen van geringe complexiteit is het probleem van de mogelijkheid van overgangsverschijnselen nog wel te overzien. Voor grotere schakelingen is een analyse van de overgangsverschijnselen (ontwerpverificatie!) vrijwel onmogelijk. Ook het bouwen van een proefschakeling geeft geen volledige garantie, met iets andere vertragingstijden van de poorten kan een ogenschijnlijk correct werkende schakeling wel kritisch worden.

Conclusie

Hoewel met een codering van de toestanden die vrij is van kritische racecondities reeds vele problemen van de via de level mode specificatie ontworpen schakelingen zijn opgelost, kunnen desondanks deze schakelingen nog kritisch zijn. In de praktijk is dit *meestal* niet het geval, maar of dit zo is *(ontwerpverificatie)* kan in het algemeen moeilijk worden vastgesteld. *Voor grotere schakelingen is een principieel andere benadering van het ontwerpen noodzakelijk.*

De eerste aanzet tot een andere benadering kan worden gegeven via het model van de combinatorische schakeling, zoals dit in par. 4.8 is geïntroduceerd. In dit model heeft elke combinatorische schakeling een zekere ingebouwde vertragingstijd Δt. De uitgang reageert op een ingangsverandering binnen deze tijdsduur Δt. Voor een schakeling, die uit een aantal poorten in verschillende niveaus is opgebouwd, kan een maximale vertragingstijd Δt_{max} worden bepaald, waarbinnen na een ingangsverandering de uitgangssignalen weer stabiel zijn. Een tijd Δt_{max} na een ingangsverandering zijn alle uitgangssignalen dan stabiel, d.w.z. de overgangsverschijnselen zijn afgelopen. Als op dat moment pas de ingangen van de geheugenelementen worden vrijgegeven, dan krijgen de geheugenelementen de juiste set- en resetsignalen weliswaar vertraagd maar zonder overgangsverschijnselen aangeboden.

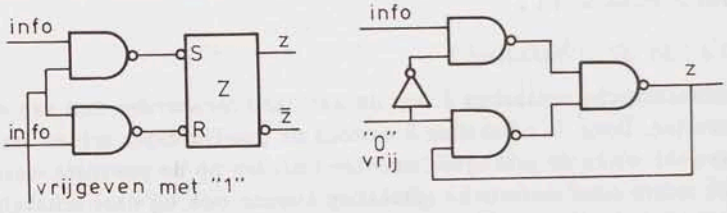


fig. 8.2. Geheugenelementen met enable ingang.

Fig. 8.2 geeft twee voorbeelden van geheugenelementen die passen in deze filosofie. Elk geheugenelement heeft één of meer informatie-ingangen en een apart ingangssignaal *enable*. Dit signaal is een stuursignaal dat aangeeft of het geheugenelement moet onthouden of inlezen. Overgangsverschijnselen in de sturende schakeling δ worden bij deze opzet niet onderdrukt, maar met het enable - signaal kan worden bereikt dat de doorwerking ervan in de er achter liggende schakeling is verhinderd. Hoe dit enable-signaal gegenereerd moet worden blijft voorlopig buiten beschouwing.

Met de introductie van een signaal enable zijn niet alle problemen opgelost. Zodra het signaal enable na een ingangsverandering de geheugenelementen vrijgeeft vertoont de schakeling weer alle kenmerken van een schakeling met de opbouw van fig. 8.1. Alleen overgangsverschijnselen ten gevolge van veranderingen in de ingangssignalen zijn omzeild. Met name kritische racecondities kunnen nog steeds optreden. Dit komt omdat de combinatorische schakeling δ nu ingangsveranderingen krijgt aangeboden via de toestandsvariabelen. Alle gevolgen daarvan worden via de schakeling δ weer rechtstreeks doorgegeven aan de geheugenelementen.

De functie van de schakeling δ is dat deze schakeling vastlegt welke de nieuwe geheugenstand moet zijn als bij de huidige (stabiele) geheugenstand een bepaalde ingangsverandering optreedt. De oorzaak van de kritische racecondities is eigenlijk het feit dat de schakeling δ de informatie over de huidige (oude) stand van de geheugenschakeling verliest *voordat* de geheugenschakeling de gewenste nieuwe stand heeft bereikt. *Er moet dus voorkomen worden dat de schakeling δ de noodzakelijke ingangsinformatie verliest vóór de nieuwe geheugenstand is bereikt.*

De suggestie tot de definitieve oplossing van dit probleem ligt reeds besloten in de schakeling van fig. 7.5. Deze schakeling is een oplossing voor het probleem van de tweedeler en is met het bijbehorende tijddiagram in fig. 8.3 gereproduceerd.
In deze schakeling wordt de gewenste nieuwe geheugenstand in twee stappen ingesteld:

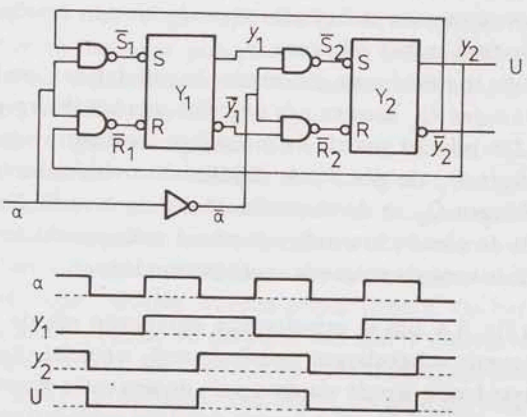


fig. 8.3. Tweedeler met tijddiagram.

$\alpha = 1$: Trekker Y_1 leest de gewenste nieuwe stand van het geheugenelement in. (Het is toeval dat dit de inverse stand van Y_2 is, in het algemeen is dit een willekeurig signaal.) Trekker Y_2 is geblokkeerd.

$\alpha = 0$: Trekker Y_1 is geblokkeerd, d.w.z. geeft nu de gewenste nieuwe geheugenstand aan. Ingangsveranderingen hebben hierop geen invloed meer. Trekker Y_2 neemt de gewenste nieuwe stand over en stelt het uitgangssignaal U in.

In deze schakeling voor de tweedeler is bereikt dat de informatie over de oude geheugenstand niet verloren gaat voordat de nieuwe stand is ingelezen. Het bereiken van de nieuwe stand geschiedt in twee fasen. Uitgaande van dit idee kan men een model van een sequentiële schakeling opstellen zoals fig. 8.4 schematisch weergeeft:

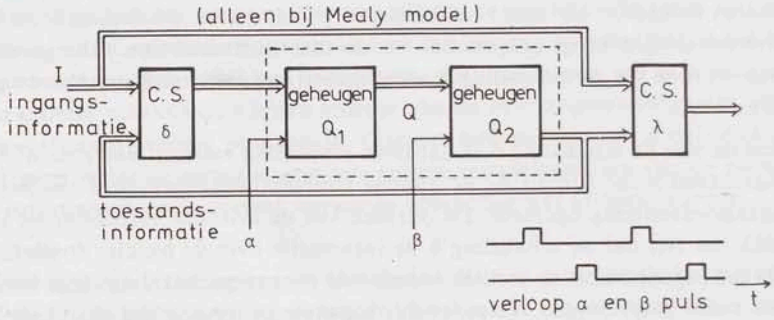


fig. 8.4. Model van een schakeling.

De veronderstelde werking van de schakeling is als volgt:

- Na een ingangsverandering "berekent" de schakeling δ de gewenste nieuwe inwendige toestand. Voor deze berekening heeft de schakeling δ informatie nodig over de ingangscombinatie (beschikbaar) en over de huidige inwendige toestand. Deze laatste informatie wordt door het geheugen Q_2 via de terugkoppeling aangeboden.
- Wanneer na enige tijd de gewenste nieuwe inwendige toestand bekend is kan deze worden overgenomen in het geheugen Q_1 via een α-puls. Daarna staat de nieuwe toestand in het geheugen Q_1.
- De nieuwe inwendige toestand mag gedurende de tijd dat het enable-signaal α voor het geheugen Q_1 aanwezig is nog niet worden doorgegeven aan het geheugen Q_2. Dit gebeurt pas als de inwendige toestand overgenomen wordt in het geheugen Q_2 via een β-puls. Tijdens dit overnemen is de koppeling tussen het geheugen Q_1 en de sturende schakeling δ verbroken!
- Na de β-puls wordt de nieuwe inwendige toestand teruggemeld en staat de schakeling weer klaar voor de volgende ingangsverandering.

In het model volgens fig. 8.4 zijn in principe alle problemen van de via de level mode gespecificeerde schakelingen opgelost, resp. omzeild. Een zekere tijd na een ingangsverandering wordt via de α-puls de gewenste nieuwe inwendige toestand ingelezen. De β-puls zorgt daarna dat deze verandering van de inwendige toestand pas teruggemeld wordt als de gewenste nieuwe inwendige toestand niet meer kan worden beïnvloed. *Hiermee wordt o.a. bereikt dat geen*

enkele race in een toestandsovergang kritisch is. Dit houdt o.a. in dat men geheel vrij is in het kiezen van de toestandscodering. Dit laatste is een groot voordeel, men kan nu bij de toestandscodering o.a. rekening houden met de gewenste codering van de uitgangssignalen.

Opmerking

In de praktijk is het niet noodzakelijk om met twee verschillende pulsen α en β te werken in het *dubbel-puls systeem*. Men kan ook volstaan met twee fasen α en $\bar{\alpha}$ van dezelfde puls zoals fig. 8.3 laat zien. Tot welke voorwaarden dit leidt met betrekking tot de maximale vertraging van de toegepaste invertor is reeds in hoofdstuk 7 bij de realisatie van de tweedeler besproken.
Ook is het mogelijk de twee geheugendelen vlak na elkaar te laten reageren op dezelfde flank van de klokpuls. Dergelijke systemen worden nog besproken.

Meester-en-slaaf principe

Het idee om in twee stappen de gewenste nieuwe inwendige toestand in te lezen met alle voordelen van dien, wordt het meester-en-slaaf principe genoemd. De puls die gebruikt wordt om de diverse handelingen in de tijd te synchroniseren heet gewoonlijk de klokpuls (clock pulse). Schakelingen die volgens dit principe ingericht zijn heten clock mode of synchrone schakelingen. De geheugenelementen in clock mode schakelingen kunnen worden opgebouwd uit twee trekkers (latches), elk voorafgegaan door een poortschakeling met enable-ingang en een of meer informatie-ingangen. Voorbeelden van deze *geklokte trekkers (clocked latches)* zijn gegeven in fig. 8.2. In de volgende paragraaf van dit hoofdstuk zullen enkele gebruikelijke types geheugenelementen voor clock mode of synchrone schakelingen nader worden onderzocht.
Synchrone schakelingen kunnen worden ontworpen met een zeer hoge graad van betrouwbaarheid omdat de invloed van allerlei overgangsverschijnselen geheel kan worden geëlimineerd. Een consequentie van deze opzet van digitale schakelingen is wel dat alle veranderingen in een schakeling gerelateerd moeten zijn aan de klokpuls. Binnen één schakeling kan hieraan gemakkelijk worden voldaan als alle geheugenelementen van hetzelfde type zijn en op hetzelfde tijdstip de klokpuls krijgen aangeboden. *Een apart probleem vormen de ingangssignalen van buiten, welke als regel onafhankelijk van de systeemklok veranderen.* Deze signalen moeten dan eerst gesynchroniseerd worden. Hetzelfde geldt voor signalen tussen twee schakelingen met verschillende klokpulsfrequenties. Ook hierbij treedt een *synchronisatieprobleem* op.
De vraag rijst of de voorafgaande behandeling van de realisatie van via de level mode gespecificeerde schakelingen wel nodig was, nu gebleken is dat bij synchrone schakelingen vele problemen kunnen worden vermeden. Het antwoord hierop kan kort zijn. Ten eerste zijn de meeste bouwstenen van de synchrone techniek (bijvoorbeeld meester-en-slaaf geheugenelementen) schakelingen die in de level mode moeten worden gespecificeerd. Op het laagste niveau is alles level mode omdat de gevolgen van *iedere* ingangsverandering moeten worden gespecificeerd. Ten tweede de synchronisatieproblemen. Dit zijn in principe level mode problemen. Ten derde, de prijs die voor de introductie van een klokpuls moet worden betaald. Niet alleen het feit dat de geheugenelementen complexer zijn speelt een rol. Ook het feit dat de schakeling in twee stappen schakelt en dus trager is kan onder omstandigheden belangrijk zijn. Dit

alles maakt dat level mode sequentiële schakelingen alleen voorkomen bij relatief kleine problemen of als bouwsteen gebruikt worden in clock mode schakelingen. Voor wat complexere problemen vormen op de clock mode gebaseerde schakelingen het enige reële alternatief. Deze schakelingen vragen wel een zorgvuldige organisatie, waarbij de *timing* van de schakelhandelingen een essentiële rol speelt.

Opmerking
In sommige boeken worden nog *pulse mode* schakelingen onderscheiden. Dit zijn schakelingen met pulsen als ingangssignalen op de informatie-ingangen in plaats van niveaus (levels). Omdat iedere puls beschouwd kan worden als twee opeenvolgende niveauveranderingen beperken we ons tot de "clock mode" en de "level mode" schakelingen, zoals deze hierboven zijn geïntroduceerd.

8.2. Geheugenelementen gebaseerd op het Meester-en-Slaaf principe

Uit een risico-analyse van level mode gespecificeerde schakelingen is het *meester-en-slaaf principe* naar voren gekomen als oplossing voor de problemen die samenhangen met overgangsverschijnselen. Geheugenelementen die volgens dit principe werken zijn de basisbouwstenen voor de synchrone schakeltechniek. Een principe-opzet voor meester-en-slaaf geheugenelementen is gegeven in fig. 8.5. We zijn dit element reeds tegengekomen als onderdeel van de tweedeler.

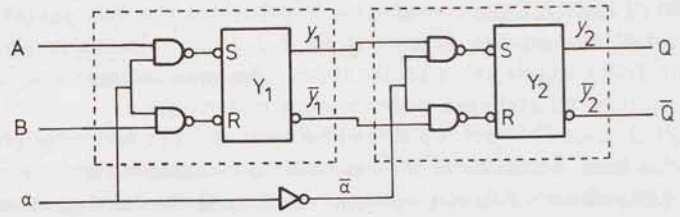


fig. 8.5. Basisprincipe van meester-en-slaaf geheugenelement.

Het meester-en-slaaf geheugenelement volgens fig. 8.5 wordt gedacht te zijn opgebouwd uit twee geklokte trekkers (clocked latches), die elk op een *verschillend* tijdstip hun informatie opnemen. De besturing geschiedt door een klokpuls α. De gevolgen van de vertragingstijd van de toegepaste invertor in de klokpulslijn zijn reeds geanalyseerd bij het tweedelerontwerp. Uiteraard is de gegeven schakeling in fig. 8.5 niet uniek, ook met vele andere schakelingen kan dezelfde uitwendige werking gerealiseerd worden.

De signalen A en B in fig. 8.5 bieden de te registreren informatie aan. Via deze ingangen wordt de gewenste nieuwe stand van het geheugenelement aangegeven, terwijl door de klokpuls α beslist wordt wanneer het geheugenelement wordt ingesteld en de nieuwe informatie wordt ingelezen. Het afgebeelde geheugenelement heeft één onprettige eigenschap:
Als de klokpuls α van $1 \rightarrow 0$ gaat en de signalen A en B zijn op dat moment beide 1, dan is de stand van de ingangstrekker Y_1 na deze overgang niet bepaald. Dit probleem is identiek aan het ingangsprobleem van de trekker zoals dit in hoofdstuk 5 reeds is behandeld.
In principe is het mogelijk in de sturende schakelingen vergrendelingen in te

bouwen zodat de signalen A en B niet tegelijk 1 kunnen zijn. Het ligt echter meer voor de hand deze vergrendeling in het geheugenelement zelf aan te brengen, dus ervoor te zorgen dat aan de ingangstrekker Y_1 de combinatie SR = 11 niet aangeboden wordt. De diverse mogelijkheden leiden tot verschillende types flip-flops zoals hieronder uiteengezet zal worden.

De groep S-R flip-flops

Het ingangsprobleem voor de schakeling in fig. 8.5 kan op dezelfde wijze worden benaderd als bij de trekker is gebeurd. Tabel 8.1 geeft de alternatieven.

	S-R flip-flop	S-R-Q flip-flop	S-R-S flip-flop	S-R-R flip-flop
A B	S R	S R	S R	S R
0 0	0 0	0 0	0 0	0 0
0 1	0 1	0 1	0 1	0 1
1 0	1 0	1 0	1 0	1 0
1 1	verbieden	0 0	1 0	0 1

tabel 8.1. Verschillende waarheidstabellen voor S-R flip-flops.

Voor geheugenelementen volgens het meester-en-slaaf principe gebruiken we de naam *flip-flop*. Volgens tabel 8.1 bestaan er vier types S-R flip-flops:

S-R flip-flop : Heeft geen vergrendeling, het probleem wordt doorgeschoven naar de sturende schakeling. (zie fig. 8.5)
S-R-Q flip-flop: In dit geval overheerst de oude stand van het geheugenelement.
S-R-S flip-flop : De setingang domineert.
S-R-R flip-flop: De resetingang domineert.

Wordt de eerste mogelijkheid buiten beschouwing gelaten, dan resteren drie geheugenelementen. Het verband tussen de signalen A en B enerzijds en de S en R ingangssignalen anderzijds is:

S-R-Q flip-flop: $S = a\bar{b}$ $R = \bar{a}b$
S-R-S flip-flop: $S = a$ $R = \bar{a}b$
S-R-R flip-flop: $S = a\bar{b}$ $R = b$

Fig. 8.6 geeft een mogelijke realisatie van de S-R-Q flip-flop.

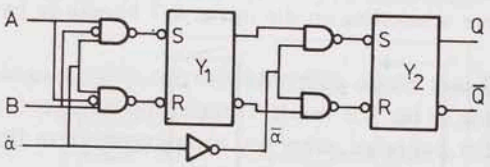


fig. 8.6. Realisatie van de S-R-Q flip-flop.

De schakeling voor de andere types S-R flip-flops wijkt niet veel af van fig. 8.6. Het ontwerp ervan wordt aan de lezer overgelaten.

Opmerking
Invertoren in ingangslijnen zullen in het vervolg vaak als cirkeltje aan een poortingang worden weergegeven wanneer dit de duidelijkheid van de tekening niet schaadt.

De D flip-flop (schuifsectie)

Het ingangsprobleem rond trekker Y_1 kan ook vermeden worden als de beide signalen A en B gekoppeld worden, waarbij het ene signaal het inverse signaal is van het andere. Een mogelijke oplossing is geschetst in fig. 8.7.

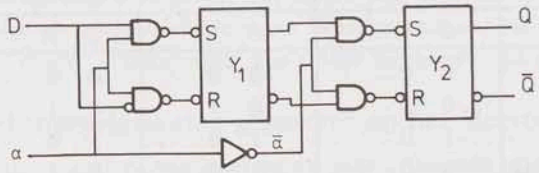


fig. 8.7. D flip-flop of schuifsectie.

Een analyse van de schakeling in fig. 8.7 leert dat de waarde van het aangeboden ingangssignaal na de klokpuls aan de uitgang verschijnt. De schakeling is daarom zeer geschikt om als sectie te dienen in een schuifregister. Vandaar de naam D(elay) flip-flop voor deze schakeling.
Het is op dit punt nuttig een parallel te trekken met het schuifprobleem zoals dit aan het eind van par. 5.4 in fig. 5.17 is geïntroduceerd.

Flip-flops met inverse terugkoppeling

Een geheel andere oplossing voor het ingangsprobleem is reeds toegepast bij de realisatie van de tweedeler (fig. 8.3). De beide uitgangssignalen zijn in deze schakeling invers teruggekoppeld. Daarmee is bereikt dat als het geheugenelement in de 1-stand staat (trekker Y_2 dus) de setpoort van de ingangstrekker is geblokkeerd. In het andere geval is de resetpoort door het uitgangssignaal geblokkeerd.
Met andere woorden, de ingangscombinatie SR = 11 is vergrendeld. Door de inverse terugkoppeling van beide uitgangen zijn we weer geheel vrij in de combinatiemogelijkheden van de ingangssignalen. De mogelijkheden zijn:

- Weglaten van de ingangssignalen. Deze mogelijkheid leidt tot de tweedeler in fig. 8.3.
- Koppelen via een invertor. Deze schakeling werkt als D flip-flop. Zie fig. 8.8. (Verschillen tussen deze schakeling en die in fig. 8.7 komen in paragraaf 8.6 nog aan bod.)
- Beide signalen A en B met elkaar doorverbinden tot één ingangssignaal. De bijbehorende schakeling in fig. 8.9 wordt T(oggle) flip-flop genoemd.
- De signalen A en B niet koppelen, maar wel aanbrengen. Deze flip-flop wordt J-K flip-flop genoemd en staat in fig. 8.10.

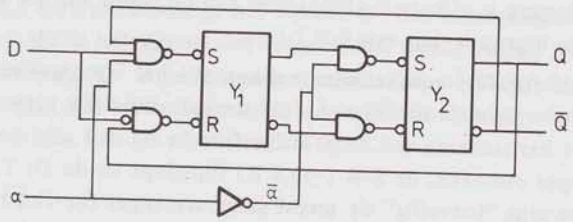


fig. 8.8. D flip-flop met inverse terugkoppeling.

De T flip-flop zoals deze in fig. 8.9 is beschreven heeft de eigenschap dat als het signaal op de T-ingang de waarde 1 heeft de schakeling als tweedeler werkt. Is het ingangssignaal echter 0, dan zijn beide ingangspoorten geblokkeerd en onthoudt het geheugenelement zijn laatste stand. Men zou de werking van de T flip-flop aldus kunnen samenvatten:

- Door middel van het signaal op de T-ingang kan het tweedelen worden onderbroken. De schakeling is een bestuurbare tweedeler.

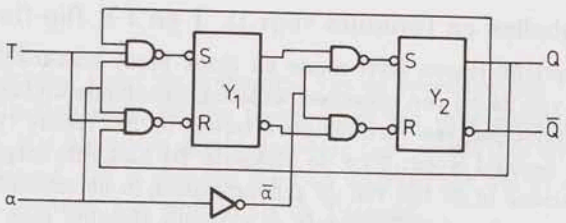


fig. 8.9. T flip-flop.

De werking van de J-K flip-flop in fig. 8.10 kan als volgt worden geformuleerd:

- Het geheugenelement staat in de stand Q = 1. Door de terugkoppeling wordt de bovenste ingangspoort geblokkeerd. Alleen het signaal op de K-ingang is dus effectief!

  K = 0: Het geheugenelement blijft in de stand Q = 1 staan
  K = 1: De ingangstrekker krijgt een reset aangeboden. Na de klokpuls staat het geheugenelement in de stand Q = 0.

- Het geheugenelement staat in de stand Q = 0.
  Nu is de resetpoort geblokkeerd en is het signaal op de J-ingang effectief. Door middel van het ingangssignaal op de J-ingang kan men het geheugenelement op de klokpuls setten (J = 1) of laten staan (J = 0).

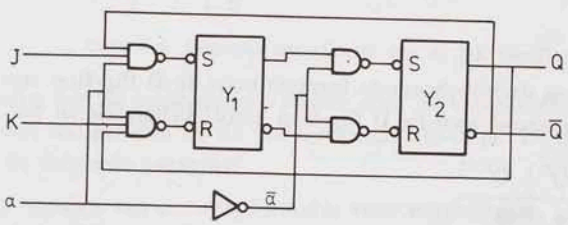


fig. 8.10. J-K flip-flop.

Bij het ontwerpen van synchrone schakelingen zal blijken dat het een groot voordeel is dat J en K onafhankelijk van elkaar gekozen kunnen worden. Òf de

J-ingang òf de K-ingang is effectief, afhankelijk van de stand van de uitgangstrekker. De andere ingang is dan een don't care.

De geïntroduceerde flip-flop schakelingen hebben alle het voordeel van een overzichtelijke opbouw waarin de meester-en-slaaf werking duidelijk zichtbaar is. Uitgaande van het basisschema van de S-R flip-flop in fig. 8.5 zijn achtereenvolgens enkele types ontstaan, de S-R-(-,Q,S,R) flip-flops en de D, T en J-K flip-flops. Hiermee zijn "toevallig" de meest gangbare types flip-flops gevonden zoals zij in de praktijk gebruikt worden. Als gebruiker van deze geheugenelementen zijn we uitsluitend geïnteresseerd in het uitwendig logisch gedrag van de bouwstenen. De inwendige opbouw behoeft niet overzichtelijk te zijn. In dat geval bestaan er schakelingen met bijvoorbeeld minder poorten die uitwendig hetzelfde logisch gedrag realiseren als de gegeven schakelingen. In de volgende paragraaf zullen we voor het uitwendig gedrag van meester-en-slaaf geheugenelementen de waarheidstabellen en de bijbehorende werkingsformules opstellen. Daarna komen diverse details ter sprake, zoals klokpulssystemen, andere types flip-flops, enz.

8.3. Waarheidstabellen en formules voor D, T en J-K flip-flops

Een kenmerkend verschil tussen level mode en clock mode schakelingen is dat bij het eerste type de bij een ingangsverandering behorende uitgangsreactie direct plaatsvindt (afgezien van vertragingen), terwijl bij het laatste type het tijdstip der reactie bepaald wordt door de klokpuls. De klokpuls zorgt voor de gewenste synchronisatie in de tijd van de gebeurtenissen in de schakeling. Op de logische functie van de schakeling heeft de klokpuls eigenlijk geen invloed. Het is daarom zinvol een onderscheid te maken tussen signaalwaarden voor en na de klokpuls. Signaalwaarden tijdens een klokpulsovergang blijven daarbij buiten beschouwing. Hoe een bepaald geheugenelement op de klokpuls reageert (op de voor/achterflank e.d.) is van essentieel belang voor de specificatie van de timing van de schakeling, maar niet voor de logische functie ervan. De beschrijving van de werking van digitale schakelingen kan in twee delen gesplitst worden:

- de specificatie van de logische functie
- de specificatie van de timing e.d.

We houden ons eerst bezig met de specificatie van de logische functie. Indien dit noodzakelijk is zal van de waarde van een signaal door middel van een index n of n + 1 worden aangegeven of het de waarde ervan na de n^e of $(n+1)^e$ klokpuls betreft.

D flip-flop

De waarheidstabel en de bijbehorende formule voor de D flip-flop zijn gegeven in tabel 8.2. Het symbool voor de D flip-flop wordt behandeld in par. 8.8.

D^n	Q^n	Q^{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

$Q^{n+1} = D^n$ formule.

tabel 8.2. Logische specificatie van de D flip-flop.

In tabel 8.2 vertegenwoordigt het signaal D^n het ingangssignaal na de n^e klokpuls, d.w.z. dit is het signaal dat aangeboden wordt op het moment van de $(n+1)^e$ klokpuls. Q^n is het uitgangssignaal na de n^e klokpuls.
Waarheidstabel 8.2 en de formule $Q^{n+1} = D^n$ die hieruit volgt specificeren de logische werking van de schakeling in fig. 8.7.

T flip-flop

Voor dit geheugenelement zijn de logische functie en de formule gegeven in tabel 8.3.

T^n	Q^n	Q^{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

$$Q^{n+1} = [\, T\overline{Q} + \overline{T}Q \,]^n = [\, T \oplus Q \,]^n$$

tabel 8.3. Logische specificatie van de T flip-flop.

Opmerking
In de literatuur komt als aanduiding van de tweedeler (zie fig. 8.3) ook de naam Toggle flip-flop voor. Bij het tegenkomen van deze naam dient men hier op te letten. Het hierboven geïntroduceerde type wordt dan wel Clocked T flip-flop genoemd. Dit is dus een flip-flop waarvan het wel of niet tweedelen op de klokpuls bestuurd kan worden via de T-ingang.

J-K flip-flop

De waarheidstabel hiervoor is tabel 8.4. Voor het symbool zie wederom par. 8.8. Tabel 8.4 volgt direct uit de schakeling in fig. 8.10.

J^n	K^n	Q^n	Q^{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

$$Q^{n+1} = [\, J\overline{Q} + \overline{K}Q \,]^n$$

tabel 8.4. Logische specificatie van de J-K flip flop.

Het bepalen van de ingangsvoorwaarden voor de verschillende types S-R flip-flops is wat lastiger dan bij de bovenstaande flip-flops. De behandeling ervan volgt in de volgende paragraaf.

Voor het bepalen van ontwerpformules voor schakelingen is de informatie via de formules en de tabellen 8.2 t/m 8.4 in een wat onhandige vorm gegeven. Tijdens het ontwerpen wordt meestal de vraag gesteld: "Welke ingangscombinatie(s) kan ik aanbieden om het geheugenelement op de volgende klokpuls in de stand 0 resp. 1 te brengen? ". In een aantal gevallen zal het antwoord op deze

vraag afhangen van de huidige stand Q^n van het geheugenelement.
De tabellen 8.5 t/m 8.7 geven de gewenste informatie in een beter hanteerbare vorm. Deze tabellen zijn gemakkelijk uit de vorige af te leiden door voor elke combinatie (Q^n,Q^{n+1}) na te gaan welke ingangscombinatie(s) tot de gewenste toestandsovergang $Q^n \rightarrow Q^{n+1}$ leiden.

Q^n	Q^{n+1}	D^n
0	0	0
0	1	1
1	0	0
1	1	1

tabel 8.5. Ingangsvoorwaarden D flip-flop.

Q^n	Q^{n+1}	T^n
0	0	0
0	1	1
1	0	1
1	1	0

tabel 8.6. Ingangsvoorwaarden T flip-flop.

Q^n	Q^{n+1}	J^n	K^n
0	0	0	0
		0	1
0	1	1	0
		1	1
1	0	0	1
		1	1
1	1	0	0
		1	0

$\Rightarrow$

Q^n	Q^{n+1}	J^n	K^n
0	0	0	–
0	1	1	–
1	0	–	1
1	1	–	0

tabel 8.7. Ingangsvoorwaarden J-K flip-flop.

Met behulp van de tabellen 8.5 t/m 8.7 is het bepalen van de mogelijke ingangswaarde(n) voor de realisatie van de gewenste toestandsovergang zeer gemakkelijk. Voor de J-K flip-flop blijkt uit tabel 8.7 dat òf J òf K in een bepaalde situatie een don't care is. Een blik op het schema in fig. 8.10 geeft de verklaring. Door de inverse terugkoppeling is altijd één van de twee ingangspoorten geblokkeerd.

Voorbeeld
In tabel 8.8 is gespecificeerd hoe de stand van een geheugenelement Q afhangt van twee ingangssignalen A en B en de stand van het geheugenelement zelf. In deze tabel zijn A^n, B^n en Q^n de signaalwaarden vóór de klokpuls en is Q^{n+1} de gewenste nieuwe stand na de klokpuls. De in tabel 8.8 gegeven specificatie kan ook via een Karnaughdiagram of een formule (fig. 8.11) worden gegeven.

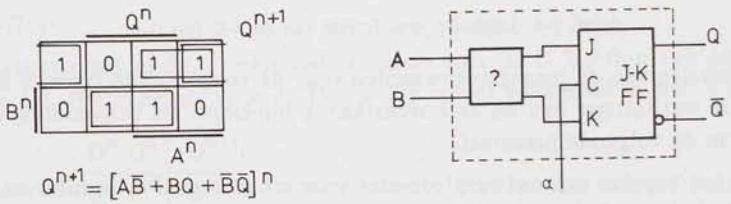


fig. 8.11. Specificatie via Karnaughdiagram.

A^n	B^n	Q^n	Q^{n+1}	D^n	T^n	J^n	K^n
0	0	0	1	1	1	1	–
0	0	1	0	0	1	–	1
0	1	0	0	0	0	0	–
0	1	1	1	1	0	–	0
1	0	0	1	1	1	1	–
1	0	1	1	1	0	–	0
1	1	0	0	0	0	0	–
1	1	1	1	1	0	–	0

tabel 8.8. Specificatie nieuwe toestand van een geheugenelement met bijbehorende ingangscondities.

De rechter kolommen van tabel 8.8 kunnen met behulp van de tabellen 8.5 t/m 8.7 worden ingevuld. Voor elke toestandsovergang moeten de bijbehorende ingangswaarden in deze tabellen worden opgezocht. De formules voor de ingangssignalen zijn:

$$D = A\bar{B} + BQ + \bar{B}\bar{Q} \qquad \text{D flip-flop}$$

$$T = \bar{A}\bar{B} + \bar{B}\bar{Q} \qquad \text{T flip-flop}$$

$$\left.\begin{aligned} J &= \bar{B} \\ K &= \bar{A}\bar{B} \end{aligned}\right\} \qquad \text{J-K flip-flop}$$

Het blijkt dat de formules voor de J-K flip-flop tot de eenvoudigste schakeling leiden, hoewel voor deze flip-flop twee ingangssignalen in plaats van één ingangssignaal gegenereerd moeten worden. In het algemeen geven sequentiële schakelingen met J-K flip-flops de beste resultaten met betrekking tot de economie van de schakeling. De reden is dat voor iedere ingangscombinatie òf J òf K vrij te kiezen is.

Voorbeeld

Gegeven is een J-K flip-flop. Ontwerp met deze flip-flop een sectie van een schuifregister (D flip-flop). Voor een schuifsectie met ingangssignaal D geldt:

$$Q^{n+1} = D^n \ .$$

Voor de gegeven J-K flip-flop is de formule

$$Q^{n+1} = [\, J\bar{Q} + \bar{K}Q \,]^n \ .$$

Uit beide formules volgt dat als

$$J = D \qquad K = \bar{D}$$

is de J-K flip-flop als schuifsectie werkt. Met J-K flip-flops is een tweedraadsverbinding nodig om ze als schuifregistersectie te kunnen schakelen.
Het gevonden resultaat kan ook via tabel 8.7 worden afgeleid (tabel 8.9 en fig. 8.12).

D^n	Q^n	Q^{n+1}	J^n	K^n
0	0	0	0	–
0	1	0	–	1
1	0	1	1	–
1	1	1	–	0

tabel 8.9. Waarheidstabel.

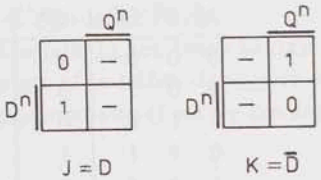


fig. 8.12. Karnaughdiagrammen.

Voorbeeld
Gegeven is een D flip-flop. Maak hiervan een J-K flip-flop.

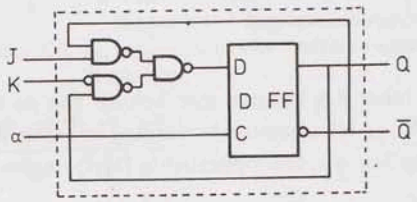


fig. 8.13. Transformatie van D in J-K flip-flop.

De formules moeten aan elkaar gelijk zijn, dus:

$$Q^{n+1} = [J\overline{Q} + \overline{K}Q]^n = D^n \ .$$

De oplossing staat in fig. 8.13. Zonder de ingangsinvertor op de onderste poort is de flip-flop te omschrijven als een J-$\overline{K}$ flip-flop. De J-$\overline{K}$ ingangscombinatie komt bij geïntegreerde circuits vaker voor. Voor de ingangsschakeling kan uiteraard ook een AND-OR-INVERT combinatie worden gebruikt.

8.4. Waarheidstabellen en formules voor S-R flip-flops

De waarheidstabel en de formule voor de S-R-Q flip-flop zijn gegeven in tabel 8.10. Vergelijk ook fig. 8.6.

S^n	R^n	Q^n	Q^{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$$Q^{n+1} = [S\overline{R} + SQ + \overline{R}Q]^n$$

tabel 8.10. Specificatie S-R-Q flip-flop.

De presentatie in tabel 8.10 is onhandig als de ingangscombinaties bepaald moeten worden waarbij de gewenste overgang $Q^n \rightarrow Q^{n+1}$ plaatsvindt. Tabel 8.11 is voor dit doel geschikter.

Q^n	Q^{n+1}	S^n	R^n	
		0	0	
0	0	0	1	$\Rightarrow S \leqslant R$
		1	1	
0	1	1	0	$\Rightarrow S = 1,\ R = 0$
1	0	0	1	$\Rightarrow S = 0,\ R = 1$
		0	0	
1	1	1	0	$\Rightarrow S \geqslant R$
		1	1	

tabel 8.11. Ingangscombinaties S-R-Q flip-flop.

Ook bij de J-K flip-flop komt het voor dat bij een toestandsovergang van $Q^n \rightarrow Q^{n+1}$ er meer dan één ingangscombinatie mogelijk is. Voor deze flip-flop geldt echter dat òf J òf K geheel vrij te kiezen is. Bij de S-R flip-flops is dit niet mogelijk, S en R *kunnen niet onafhankelijk van elkaar* worden gekozen. Deze afhankelijkheid is lastig bij het bepalen van eenvoudige ingangsformules, zoals het volgende voorbeeld aantoont.

Voorbeeld

In fig. 8.14 is gespecificeerd hoe Q^{n+1} van een geheugenelement Q afhangt van de stand van het element zelf en van twee ingangssignalen A en B.

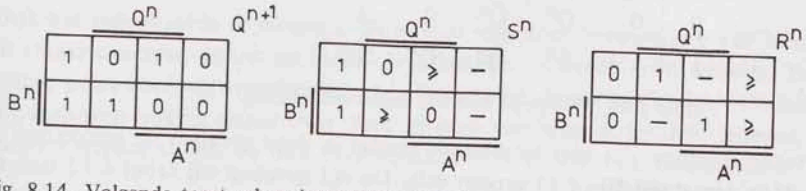


fig. 8.14. Volgende toestand en ingangsvoorwaarden voor een S-R-Q flip-flop.

Uit het eerste diagram kunnen met behulp van tabel 8.11 de mogelijkheden worden bepaald voor de S en R ingangen. Het is niet eenvoudig uit de diagrammen voor S en R de eenvoudigste formules te bepalen omdat beide signalen afhankelijk zijn. Mogelijke oplossingen zijn bijvoorbeeld:

$$S = \bar{A}\bar{Q} \qquad \text{en} \qquad R = \bar{A}\bar{B}Q + AB$$

of:

$$S = \bar{Q} + A\bar{B} \qquad \text{en} \qquad R = A + \bar{B}Q\ .$$

Het feit dat de ingangssignalen niet onafhankelijk gekozen kunnen worden is er de oorzaak van dat de S-R flip-flops niet veel gebruikt worden. In de praktijk komt men ze dan ook zelden tegen. Hetzelfde geldt voor de S-R flip-flops met overheersende set resp. reset.

8.5. Overige types flip-flops

Via de problematiek van de ingangstrekker in het meester-en-slaaf geheugenelement in par. 8.2 zijn enkele types flip-flops geïntroduceerd. Een intrigerende vraag is of er nog meer types gedefinieerd kunnen worden. Dit kan inderdaad.

Daarbij zullen we ons beperken tot geheugenelementen met een zo klein mogelijk aantal informatie-ingangen.
Een flip-flop die de formule

$$Q^{n+1} = [J_1 J_2 \cdot \overline{Q} + \overline{K_1 K_2} \cdot Q]^n$$

realiseert, zal bijvoorbeeld buiten beschouwing gelaten worden. De termen $J_1 J_2$ en $K_1 K_2$ kunnen ook met losse AND's gerealiseerd worden.

Flip-flops zonder informatie-ingang

Van de vier mogelijke uitgangswaarden 0, 1, Q^n en $\overline{Q}^n$ is alleen de schakeling die

$$Q^{n+1} = \overline{Q}^n$$

realiseert interessant. Dit is de tweedelerschakeling.

Flip-flops met één informatie-ingang

De volgende vijf gevallen zijn zinvol voor geheugenelementen met één informatie-ingang I:

I^n	Q_1^{n+1}	Q_2^{n+1}	Q_3^{n+1}	Q_4^{n+1}	Q_5^{n+1}
0	0	Q^n	$\overline{Q}^n$	0	1
1	1	$\overline{Q}^n$	$\overline{Q}^n$	$\overline{Q}^n$	$\overline{Q}^n$

tabel 8.12. Flip-flops met één informatie-ingang.

De overige gevallen zijn niet interessant omdat in deze gevallen de schakeling zichzelf in één stand (0 of 1) vergrendelt. De vijf gevallen uit tabel 8.12 leiden tot:

1. $Q^{n+1} = I^n$ D flip-flop
2. $Q^{n+1} = [I \oplus Q]^n$ T flip-flop
3. $Q^{n+1} = \overline{Q}^n$ Tweedeler. De ingang is overbodig.
4. $Q^{n+1} = [I\overline{Q}]^n$ Te beschouwen als tweedeler met overheersende (geklokte) reset.
5. $Q^{n+1} = [\overline{I} + \overline{Q}]^n$ Te beschouwen als tweedeler met overheersende (geklokte) set.

Fig. 8.15 geeft een schema voor een tweedeler met overheersende reset. Hoewel het op zichzelf een aardig geheugenelement is, heeft het geen zin om deze als apart type in te voeren. Een standaard J-K flip-flop met J = I en K = 1 doet hetzelfde.

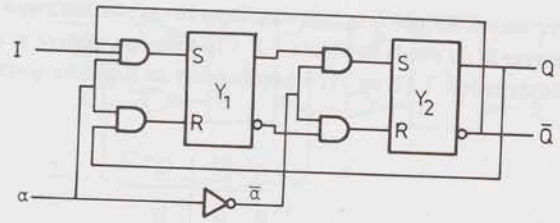


fig. 8.15. Tweedeler met overheersende reset.

Opmerking

De onder 5. genoemde tweedeler is als identiek te beschouwen met de onder 4. genoemde. Door verwisseling van het uitgangssignaal gaan zij in elkaar over. Zie fig. 8.16.

fig. 8.16. Omzetting van tweedeler met overheersende reset in een met overheersende set.

Ook het omgekeerde is mogelijk. Flip-flops, die door verwisseling van uitgangen in elkaar overgaan zullen als identieke schakelingen beschouwd worden. Hetzelfde geldt voor verwisseling van ingangssignalen. Soms gaan twee verschillende schakelingen ook in elkaar over door inversie van ingangssignalen. Hoewel iets minder triviaal zullen we ook deze schakelingen als identiek beschouwen.

Flip-flops met twee informatie-ingangen

Schakelingen met verboden ingangscombinaties blijven buiten beschouwing (vergelijk S-R flip-flop). Ook de schakelingen met slechts twee verschillende uitgangswaarden uit $\{0,1,Q^n,\bar{Q}^n\}$ blijven buiten beschouwing omdat zij met één ingang kunnen worden gerealiseerd en dus tot de vorige groep behoren. Met deze beperkingen kan men twee groepen flip-flops definiëren:

- flip-flops waarbij drie van de vier mogelijke uitgangswaarden optreden (één uitgangswaarde treedt dus twee maal op).
- flip-flops waarbij de vier uitgangswaarden $0,1,Q^n$ en $\bar{Q}^n$ elk één maal voorkomen.

De eerste groep bevat 14 niet-identieke flip-flops. De overige zijn door verwisseling en/of inversie van ingangen resp. verwisseling van uitgangen tot een van deze 14 terug te brengen. Een aardige variant is bijvoorbeeld de flip-flop die de formule

$$Q^{n+1} = [I_1 + I_2\bar{Q}]^n$$

realiseert. Is $I_1 = 0$ en $I_2 = 1$, dan is de schakeling een tweedeler. Voor $I_1 = D$ en $I_2 = 0$ vormt de schakeling een D flip-flop.

Ook de verschillende S-R flip-flops behoren tot deze groep van 14 flip-flops.

De tweede groep, met vier verschillende uitgangswaarden, bevat slechts twee niet-identieke flip-flops. De eerste is de reeds bekende J-K flip-flop, de andere is de A-S flip-flop. De waarheidstabellen 7.13 en 7.14 beschrijven de logische werking ervan.

J^n	K^n	Q^{n+1}
0	0	Q^n
0	1	0
1	0	1
1	1	$\overline{Q}^n$

$$Q^{n+1} = [J\overline{Q} + \overline{K}Q]^n$$

tabel 8.13. J-K flip-flop.

A^n	S^n	Q^{n+1}
0	0	0
0	1	Q^n
1	0	1
1	1	$\overline{Q}^n$

$$Q^{n+1} = [A \oplus SQ]^n$$

tabel 8.14. A-S flip-flop.

In het gebruik bestaan er belangrijke verschillen tussen de J-K en de A-S flip-flop. Tabel 8.15 geeft een overzicht van de ingangsvoorwaarden bij een gewenste $Q^n \rightarrow Q^{n+1}$ overgang.

Q^n	Q^{n+1}	J^n	K^n	A^n	S^n
0	0	0	–	0	–
0	1	1	–	1	–
1	0	–	1	A = S	
1	1	–	0	$A \neq S$	

tabel 8.15. Ingangsvoorwaarden van J-K en A-S flip-flop.

Bij de J-K flip-flop zijn de twee ingangen J en K altijd onafhankelijk van elkaar, terwijl de keuze voor A en S in slechts twee gevallen onafhankelijk van de ander is. In de praktijk betekent dit een sterke voorkeur voor de J-K flip-flop.

8.6. Timing van flip-flop geheugenelementen

In het voorafgaande is gesproken over de stand Q^n van een geheugenelement vóór de klokpuls en de stand Q^{n+1} er na. Daarbij is in het midden gelaten hoe het element op de klokpuls reageert. Zo eenvoudig als deze formulering doet vermoeden is dit echter niet.

De uitgang van de tot nu toe behandelde schakelingen verandert op de neergaande flank van de klokpuls (de *negative going edge*). Duidelijk is dat dit geen algemeen kenmerk is van flip-flop geheugenelementen.

De schakeling in fig. 8.17, een D flip-flop met NOR's, reageert met zijn Q-uitgang bijvoorbeeld op de opgaande flank (de *positive going edge*) van de klokpuls.

In eenzelfde schakeling mag men geen flip-flops door elkaar gebruiken die op verschillende flanken van de klokpuls reageren. Het zo zorgvuldig opgezette systeem van scheiding tussen de ingangsreactie en uitgangsreactie valt dan in het water. Behalve met dit gegeven moet de ontwerper met nog enkele andere eigenschappen van flip-flops rekening houden. Ter introductie vergelijken we

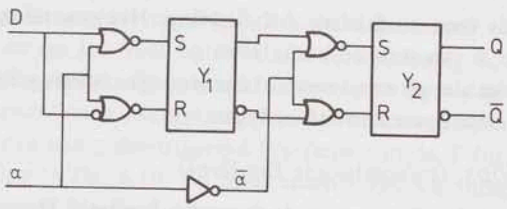


fig. 8.17. D flip-flop met uitgangsreactie op de opgaande of positieve flank van de klokpuls.

de twee uitvoeringen van de D flip-flop uit de figuren 8.7 en 8.8. Voor beide flip-flops geldt de formule:

$$Q^{n+1} = D^n \ .$$

Beide flip-flops passen hun uitgangen aan op de neergaande flank van de klokpuls α, zodat zij ook hierin niet verschillen. Het verschil tussen beide uitvoeringen wordt veroorzaakt door het tijdstip waarop de ingangsinformatie stabiel moet zijn teneinde te kunnen worden ingelezen. We beschouwen beide schakelingen op dit punt wat nauwkeuriger.

D flip-flop zonder terugkoppeling. (fig. 8.7)

Na de uitgangsreactie op de neergaande flank van α worden nieuwe ingangswaarden gegenereerd door de combinatorische schakelingen. Deze worden ingelezen als de klokpuls α weer 1 wordt. Zolang als $\alpha = 1$ is volgt de ingangstrekker Y_1 het signaal op de D-ingang. De enige eis die we aan het signaal op de D-ingang moeten stellen is, dat het zijn *eindwaarde* heeft bereikt vlak voor het weer 0 worden van α. Dus vlak voor het moment dat de uitgangstrekker dichtgaat en zijn laatste ingangswaarde onthoudt.
In feite betekent dit dat nagenoeg de gehele klokpulsperiode ter beschikking staat als *insteltijd* t_i voor de combinatorische schakelingen die de D-ingang sturen. Afhankelijk van de pulsbreedte van α mogen overgangsverschijnselen nog enige tijd optreden als α reeds 1 is.

D flip-flop met inverse terugkoppeling. (fig. 8.8)

Door de inverse terugkoppeling is steeds één van de twee ingangspoorten van trekker Y_1 geblokkeerd. In de stand $Q = 1$ bijvoorbeeld is de bovenste poort, de setpoort, geblokkeerd. Is $\bar{Q} = 1$ dan is de resetpoort geblokkeerd. Deze ingebouwde blokkade heeft consequenties!
Stel dat $Q^n = 1$ is en dat ook de nieuwe stand $Q^{n+1} = 1$ moet zijn. Het informatiesignaal op de D-ingang moet dan 1 zijn. Zou nu ten gevolge van een storing resp. een overgangsverschijnsel het signaal op de D-ingang gedurende de tijd dat $\alpha = 1$ is even 0 zijn, dan kan de ingangstrekker Y_1 gereset worden en naar de nulstand gaan. Wordt daarna D weer 1, dan is herstel niet meer mogelijk omdat het terugkoppelsignaal de setpoort blokkeert!

Conclusie
Bij een flip-flop met inverse terugkoppeling moet het informatiesignaal op de D-ingang reeds vóór de klokpuls stabiel zijn.
Bij de D flip-flop in fig. 8.8 zijn de terugkoppelingen overbodig. Dit geldt niet

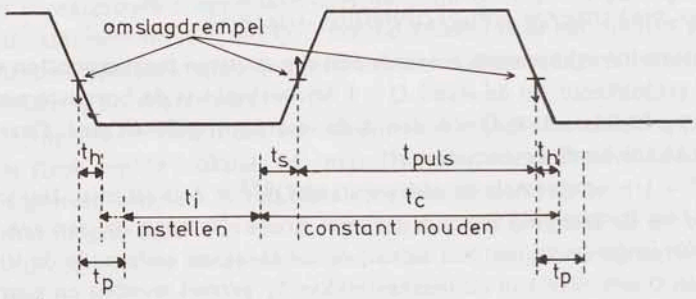
bij de getekende schema's voor de T of de J-K flip-flop. Het gevonden verschil in reactie op de klokpuls is van wezenlijke betekenis.
Uit het bovenstaande volgt dat er een verschil is in de wijze waarop flip-flops op de klokpuls reageren. We komen tot twee types:

Pulse-triggered flip-flops (pulsgestuurde flip-flops)

Bij de zgn. *pulse-triggered flip-flops* moet de ingangsinformatie klaar staan vóórdat de klokpuls komt (d.w.z. vóór de opgaande flank wanneer de uitgang van de flip-flop op de neergaande flank reageert). De informatie moet bovendien minimaal een tijd t_s voor de betreffende flank klaar staan om de ingangstrekker betrouwbaar te kunnen inlezen. De tijd t_s wordt de *setup time* genoemd. De setup time ligt bij TTL-geïntegreerde circuits meestal tussen $0 \leqslant t_s \leqslant 50$ ns. Vaak moet de ingangsinformatie ook gedurende enkele nanoseconden na het instellen van de uitgangstrekker worden aangeboden. Deze tijd heet de *hold time* t_h. Gebruikelijke waarden voor de hold time liggen tussen $0 \leqslant t_h \leqslant 20$ ns.
Na het omschakelen duurt het enige tijd voordat de uitgang van de flip-flop reageert. Deze tijd wordt de *propagation delay time* t_p genoemd. Ook de aanduiding t_d komt voor. De tijd t_p is te beschouwen als de insteltijd van de uitgangstrekker. Meestal treft men in catalogi t_{pHL} en t_{pLH} aan, waarmee de reactietijd van Hoog naar Laag resp. van Laag naar Hoog wordt aangeduid. Tussen t_{pHL} en t_{pLH} kan enkele ns verschil zitten. Van beide tijden wordt het minimum en het maximum opgegeven. Binnen $t_{p(min)}$ reageert de flip-flop nog niet, na $t_{p(max)}$ is de uitgang ervan zeker weer stabiel. Een en ander is in fig. 8.18 voor een pulse-triggered flip-flop in een tijdschema aangegeven.
Uit fig. 8.18 volgt dat voor het instellen van de combinatoriek die de instelsignalen voor de D en andere data-ingangen produceert een insteltijd t_i beschikbaar is. Vervolgens moeten de instelsignalen een tijd t_c constant blijven:

$$t_c = t_s + t_{puls} + t_h \ .$$

De tijd t_c hangt mede af van de duur van de klokpuls.



t_p : propagation delay time
t_s : setup time
t_h : hold time
t_{puls} : pulsduur
t_i : insteltijd combinatoriek
t_c : signalen constant houden.

fig. 8.18. Specificatie pulse-triggered timing.

Bij TTL-geïntegreerde circuits ligt de omslagdrempel van poorten op ca. 1,4 Volt bij 20°C en op 1,2 Volt bij 80°C. Ook de tijden t_s, t_h en t_p zijn enigszins temperatuurafhankelijk. De invloed hiervan is geringer dan de normale spreiding tussen verschillende exemplaren flip-flops en wordt daarom meestal niet vermeld. Voorbeelden van pulse-triggered flip-flops zijn de T flip-flop in fig. 8.9 en de J-K flip-flop in fig. 8.10. Bij meer recente types geïntegreerde circuits komen pulse-triggered flip-flops weinig meer voor.

Edge-triggered flip-flops (flankgestuurde flip-flops)

Bij edge-triggered flip-flops speelt het gehele omschakelen zich af rond één flank van de klokpuls. Zie fig. 8.19 voor het tijdschema. Vlak voor de *actieve klokflank* moet de nieuwe informatie klaar staan. Deze tijd is aangegeven als t_s. Direct na het passeren van het omslagniveau, d.w.z. een tijd t_h later, mogen de signalen weer veranderen. Ten behoeve van het inlezen moeten de signalen een tijd t_c:

$$t_c = t_s + t_h$$

constant blijven. Bij deze flip-flops is t_c onafhankelijk van de pulsduur. De D flip-flop in fig. 8.7 is een voorbeeld van een flip-flop met een edge-triggered timing. De werking ervan is reeds besproken.

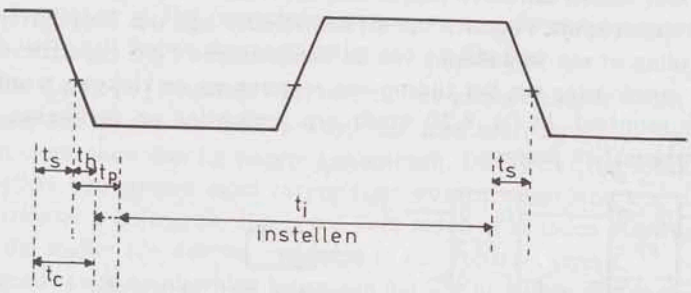


fig. 8.19. Specificatie edge-triggered timing.

Men hoort wel eens de opmerking dat "schakelingen met edge-triggered flip-flops sneller zijn dan schakelingen met pulse-triggered flip-flops". Bij een gegeven *puls-pauze verhouding* van de klokpuls kan men bij edge-triggered flip-flops meer poortniveaus in de sturende schakelingen toestaan. Of, hetgeen hetzelfde betekent, bij een gelijk aantal poortniveaus de klokpuls eerder aanbieden omdat als α hoog is nog kan worden ingesteld. Bij een puls-pauze verhouding van 50% zijn schakelingen met edge-triggered flip-flops in principe sneller. Gaan we echter uit van een klokpuls met een minimale pulsbreedte waarop de flip-flop nog juist reageert, dan is er nagenoeg geen verschil in de maximale klokfrequentie voor beide types.

Sommige uitvoeringen van edge-triggered flip-flops werken intern pulse-triggered. Zij leiden van de aangeboden klokpuls een veel smallere puls af waarop de flip-flop dan reageert. Uitwendig hebben die flip-flops een edge-triggered timing. Voor de gebruiker is uiteraard het uitwendig gedrag van belang.

Opmerking

De werking van flip-flops berust soms kritisch op de aanwezigheid van poortvertragingen. Zie als voorbeeld de D flip-flop in fig. 8.7. De uitgangstrekker Y_2

hiervan reageert m.b.t. het dichtzetten van zijn ingangspoorten later op de klokpuls dan de ingangstrekker, die open gezet wordt. Toch functioneert deze flip-flop correct omdat de vertraging van de ingangstrekker het tijdsverschil opvangt. Voor een inzicht in de werking van de flip-flopschakelingen valt het te betreuren dat men in de catalogi de voor de correcte werking belangrijke poortvertragingen niet in de schema's opneemt.

Edge-triggered flip-flops komen voor als:

- *positive edge-triggered flip-flops.*
  Dit type reageert op de Laag-naar-Hoog overgang van de klokpuls.
- *negative edge-triggered flip-flops.*
  Dit type reageert op de Hoog-naar-Laag overgang van de klokpuls.

Ook wordt onderscheid gemaakt in *positive* en *negative pulse-triggered* flip-flops. Kenmerkend voor de indeling hierbij is de flank voor de welke de *ingangs-informatie* klaar moet staan. De opgaande flank wordt "positief" gerekend.

Clock skew

Een voorwaarde voor de betrouwbare werking van een schakeling met flip-flops als geheugenelementen is dat alle flip-flops tegelijk op dezelfde klokpulsflank reageren. Door allerlei oorzaken kan er enig tijdsverschil optreden in het aanbieden van de klokpuls. Factoren die dit beïnvloeden zijn o.a. looptijdverschillen in de bedrading of een verandering van de flanksteilheid t.g.v. capacitieve belasting. Deze verschuiving van het tijdstip van reageren op de klokpuls wordt *clock skew* genoemd. In fig. 8.20 wordt een toelichting op de hieraan verbonden problematiek gegeven.

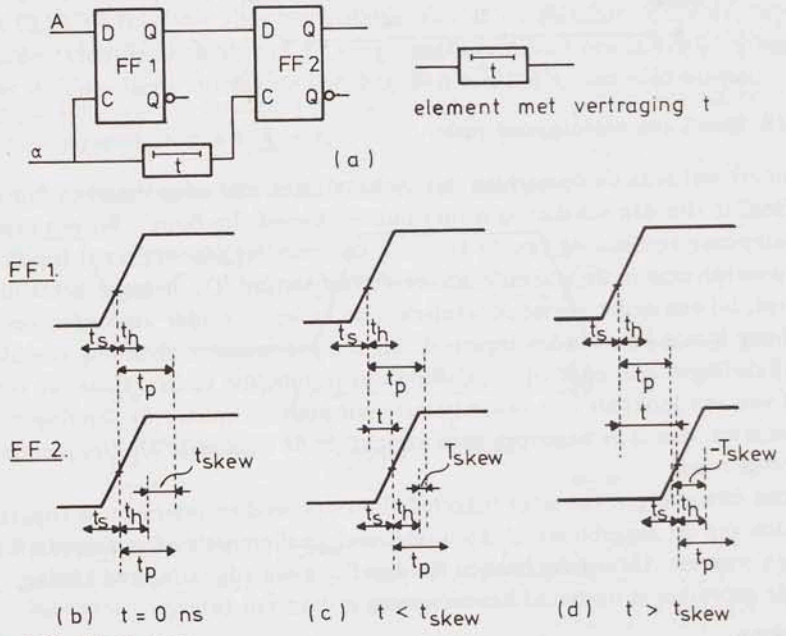


fig. 8.20. Clock skew.

In fig. 8.20.a is een schakeling met twee D flip-flops van het type positive edge-triggered getekend waarvan $D_2 = Q_1$ is. In de kloklijn is een vertraging t opge-

nomen. In fig. 8.20.b is de timing geschetst van beide flip-flops onder de conditie dat t = 0 ns.
Na het passeren van de omslagdrempel reageert FF1 op de klokpuls. Dit zien we aan de uitgang Q_1 een tijd t_p later. Flip-flop FF2 stelt als eis dat het signaal op de D ingang na het passeren van de omslagdrempel nog een tijd t_h constant blijft. Uit het diagram b. zien we dat hieraan ruim voldaan wordt. De klokpuls van FF2 mag zelfs een tijd t_{skew} verschuiven:

$$t_{skew} = t_{p(min)} - t_{h(max)} .$$

Deze tijd t_{skew} is te beschouwen als een parameter van het type flip-flop!

In fig. 8.20.c is de situatie geschetst bij een geringe verschuiving $t < t_{skew}$ van de klokpuls voor FF2. Ook hier geeft het doorgeven van de informatie geen problemen. De marge die overblijft

$$T_{skew} = t_{skew} - \text{vertraging} = t_{skew} - t$$

is kleiner dan in het vorige geval. T_{skew} geeft de marge aan die in een concrete situatie effectief beschikbaar is. In fig. 8.20.d is T_{skew} negatief geworden. FF1 verandert zijn uitgang reeds voordat de noodzakelijke hold time voor het signaal op D_2 verstreken is. Het correct doorgeven van de informatie is problematisch. Deze situatie valt buiten de specificaties van de flip-flop.

Bij snelle flip-flops (Schottky TTL ed.) zijn de gespecificeerde tijden t_s, t_p en t_h kleiner dan bij gewone TTL bouwstenen. Men heeft bij snelle types eerder last van clock skew dan bij tragere bouwstenen. De lay-out van schakelingen met snellere bouwstenen moet zorgvuldiger worden uitgevoerd, o.a. de *klokpulsdistributie* is belangrijk. Het is om deze reden af te raden bouwstenen te kiezen die sneller zijn dan de toepassing in een apparaat vereist.
De volgende twee voorbeelden tonen aan dat het probleem van de clock skew in een ruimer kader onderzocht moet worden dan alleen de flip-flop parameter t_{skew} suggereert.

Voorbeeld

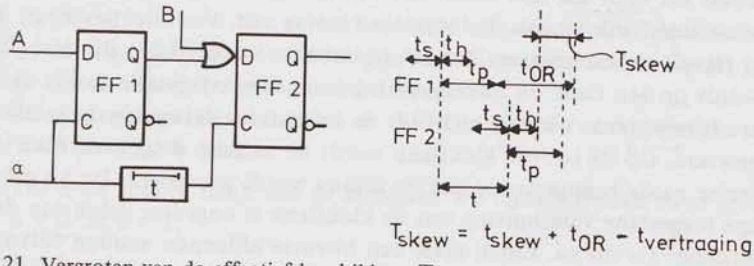


fig. 8.21. Vergroten van de effectief beschikbare T_{skew}.

In fig. 8.21 is beschreven hoe het tijdschema verandert als in de datalijn $Q_1 \rightarrow D_2$ een OR-poort geplaatst wordt. De ingang D_2 ziet een verandering van Q_1 pas na een tijd $T_{p(effectief)}$:

$$T_p = t_p + t_{OR} .$$

De toegestane verschuiving van de klokpuls voor FF2 is nu groter. De marge voor t, de vertragingstijd, is:

$$0 \leqslant t \leqslant t_{skew} + t_{OR} \cdot$$

Onder omstandigheden kan deze wetenschap dienen om de gevolgen van een soms niet te vermijden klokverschuiving te compenseren!

Voorbeeld

In fig. 8.22 zijn twee manieren aangegeven om flip-flops van een *enable-ingang* te voorzien. In fig. 8.22.a geschiedt het "vasthouden" van de D flip-flop via een onderbreking van de klokpuls door een AND-poort. De gevolgen van het schakelen in kloklijnen voor de T_{skew} zijn hiervoor reeds besproken. Oplossing 8.22.b is wat dit punt betreft aanzienlijk beter. Wel moet de informatie voor deze flip-flop eerder worden aangeboden. Bij de setup time moeten twee poortvertragingen worden opgeteld.

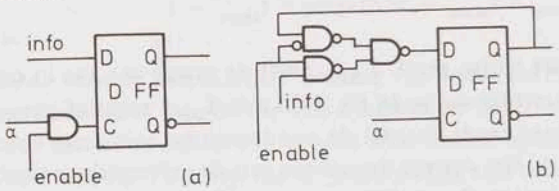


fig. 8.22. Flip-flops met enable-ingang.

Het schakelen in kloklijnen treft men bij vele oudere types circuits aan, o.a. bij tellers en schuifregisters. Het inzicht dat hierdoor de betrouwbare werking van geklokte schakelingen nadelig wordt beïnvloed heeft ertoe geleid dat bij recent uitgekomen types selectoren worden toegepast. De aanwezigheid van de marge t_{skew} van de flip-flops betekent niet dat deze tijd voor schakeldoeleinden bestemd is. Deze marge is nodig om vertragingen in de klokpulsdistributie en verschillen in flanksteilheden e.d. te compenseren.

Data lockout flip-flops

Soms komt het voor dat niet onder alle omstandigheden gegarandeerd kan worden dat de klokflank binnen de toegestane marge valt. Voor dat geval zijn *data lockout flip-flops* beschikbaar. Zie het tijdschema in fig. 8.23. Bij deze flip-flops wordt op één flank de informatie ingelezen. De informatie wordt dan als het ware bemonsterd. Vervolgens wordt de informatie gedurende de gehele pulsduur bewaard. Op de tweede klokflank wordt de uitgang aangepast. Elke ingangsverandering na de bemonstering van de ingang wordt genegeerd. De bij dit type flip-flops toegestane verschuiving van de klokflank is ongeveer gelijk aan de gehele pulsduur. Ga dit na. Clock skew kan hiermee afdoende worden bestreden. Data lockout flip-flops kan men opgebouwd denken uit een edge-triggered D flip-flop gevolgd door een zg. clocked latch. Zie fig. 5.16.

In principe kunnen de drie genoemde flip-flops in een schakeling door elkaar gebruikt worden. De uitgangen moeten alle op dezelfde flank van de klok reageren. Indien nodig of gewenst kan men van deze regel afwijken. Men dient in dat geval te verifiëren of aan de vereisten voor t_s en t_h voldaan wordt. Bij schakelen op dezelfde flank wordt hieraan vrijwel automatisch voldaan.

Nemen we de *gevoeligheid voor storingen* of *overspraak* op de datalijnen mede

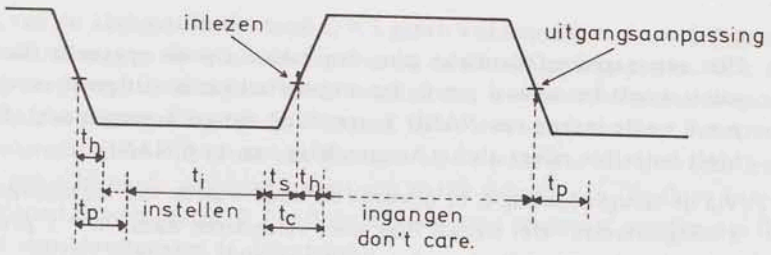


fig. 8.23. Tijdschema data lockout flip-flops.

in beschouwing, dan gaat de voorkeur uit naar edge-triggered of data lockout flip-flops. Alleen de storingen op het bemonstertijdstip zijn hinderlijk. Bij pulse-triggered flip-flops kunnen in principe storingen gedurende de gehele tijd dat $a = 1$ is tot een verkeerde stand leiden.

8.7. Voorbeelden van flip-flop schakelingen

We bespreken in deze paragraaf enkele flip-flop schakelingen zoals men die in de praktijk kan aantreffen. Deze schakelingen hebben niet meer de overzichtelijke interne opbouw van de reeds besproken schakelingen.

Voorbeeld

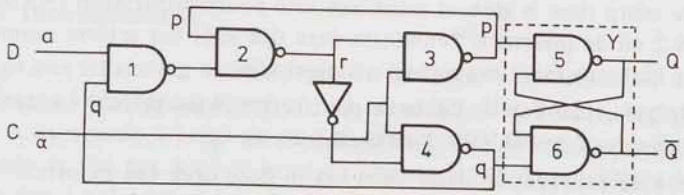


fig. 8.24. Schema edge-triggered D flip-flop.

De schakeling in fig. 8.24 is een voorbeeld van een edge-triggered D flip-flop, welke op de positieve flank (L → H) van de klokpuls reageert. Om de werking te kunnen beschrijven worden drie hulpsignalen p, q en r geïntroduceerd. De schakeling werkt als volgt:

$\alpha = 0$: De NAND's 3 en 4 zijn geblokkeerd. De signalen p en q hebben de waarde 1; de uitgangstrekker onthoudt zijn laatst ingelezen stand. Omdat $p = q = 1$ is zijn de ingangspoorten 1 en 2 vrijgegeven.
Het ingangssignaal a dringt via de NAND's 1 en 2 tot de ingangen van de NAND's 3 en 4 door.

$\alpha \rightarrow 1$: Dit is de actieve flank van de klokpuls. We onderscheiden twee gevallen, $a = 0$ of $a = 1$.

$\underline{a = 0}$
Zolang $\alpha = 0$ is, krijgt NAND 3 $\alpha a = 00$ aangeboden en NAND 4 $\alpha\bar{a} = 01$ (via de invertor). Gaat α van $0 \rightarrow 1$, dan krijgt op dat moment NAND 4 twee enen aangeboden waardoor de uitgang q ervan naar 0 gaat. Dit heeft twee gevolgen: de uitgangstrekker Y krijgt een resetsignaal en NAND 1 krijgt een 0 aangeboden via signaal q. Deze 0 vergrendelt als het ware de momentele waarde van a, zodat de flip-flop ongevoelig wordt voor verdere veranderingen van het ingangssignaal a.

a = 1

Hier een nagenoeg identieke gang van zaken. Op de opgaande flank van a wordt het signaal p = 0. De uitgangstrekker wordt geset en via p = 0 op de ingang van NAND 2 wordt het signaal $\bar{a}$ vergrendeld. Dit heeft hetzelfde effect als het vergrendelen van a op NAND 1.

$a = 1$: Via de terugkoppeling p of q wordt de oude waarde van ingangssignaal a vastgehouden. Het signaal mag weer veranderen nadat a = 1 geworden is, dit heeft geen invloed op de uitgangswaarde.

$a \rightarrow 0$: De NAND's 3 en 4 worden geblokkeerd, zodat de uitgangen p en q ervan weer naar 1 gaan. Hierdoor worden de ingangspoorten weer vrijgegeven en kan de ingangsschakeling zich instellen op de nieuwe waarde van het ingangssignaal a.

Samenvattend:

Als tijdens de overgang van $a = 0 \rightarrow a = 1$ de klokpuls het *aanspreekniveau* van de circuits 3 en 4 passeert, dan gaat òf p òf q naar 0. Daarmee wordt de waarde die a op dat moment heeft vergrendeld. Tevens wordt de uitgangstrekker ingesteld. De uitgang van de flip-flop kan nu veranderen maar door inwendige vertragingen gebeurt dit vlak na het vergrendelen van de ingangen.
Voor deze flip-flop geldt:

t_s: De setup time is globaal gelijk aan drie poortvertragingen (NAND's 1 en 2 en de invertor). Tenminste deze tijd vóór het actieve moment van de klokpuls moet het signaal a stabiel zijn.

t_h: De hold time kan op ca. twee poortvertragingen gesteld worden (de vertraging van de NAND's 3 en 2 of 1 en 4).

t_p: Ook de propagation delay time ligt in deze orde van grootte.

De in fig. 8.24 beschreven edge-triggered D flip-flop werkt op één flank van de klokpuls. Dit geheugenelement is geconstrueerd op maximale schakelsnelheid. Ook de D flip-flop in fig. 8.7 kan beschouwd worden als een edge-triggered D flip-flop. De reactie van deze schakeling op de klokpuls is echter beduidend trager. Ga dit na!

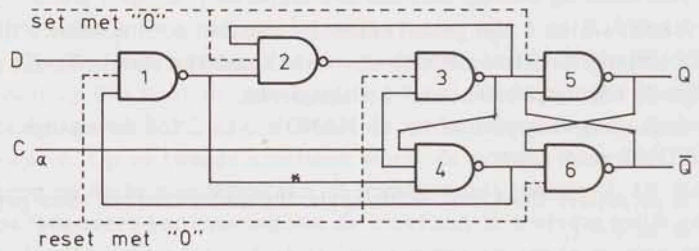


fig. 8.25. D flip-flop met preset en clear.

In fig. 8.25 is dezelfde D flip-flop nogmaals getekend. De invertor is vervangen door de uitgang van NAND 3 terug te voeren naar de ingang van NAND 4. Om de setup time te reduceren wordt ook de uitgang van NAND 1 naar NAND 4 doorverbonden. Ga het nuttig effect hiervan na.
Tevens is de flip-flop in fig. 8.25 uitgerust met een *preset-* en *clear-ingang.* Deze zijn als stippellijnen aangegeven. Met de bovenste lijn kan de flip-flop onafhan-

kelijk van de klokpuls in de stand Q = 1 gezet worden. Dit is de preset-ingang. De onderste ingang dient om de flip-flop in de stand Q = 0 te zetten. Ook deze ingang overheerst de klokpuls. Beide instelgangen worden *directe instelingangen* genoemd, in tegenstelling tot de D ingang welke een *voorbereidend karakter* heeft. Het volgende voorbeeld toont aan dat men bij de realisatie van in wezen level mode gespecificeerde problemen soms een nuttig gebruik van flip-flops kan maken. Vooral edge-triggered D flip-flops kunnen vaak toegepast worden om flanken of signaalovergangen te detecteren.

Voorbeeld

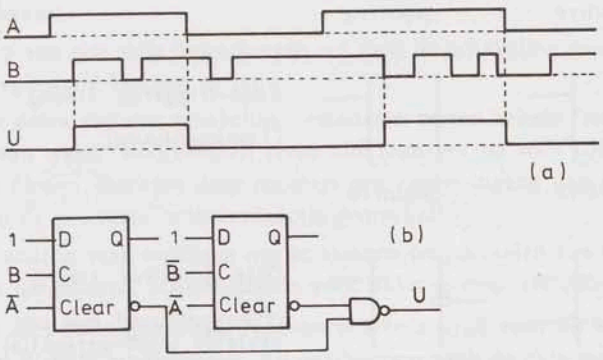


fig. 8.26. Detectieschakeling.

Gevraagd een schakeling te ontwerpen die gedurende een instelbaar tijdsinterval controleert of een signaal B wel of niet verandert. Het interval begint zodra een signaal A de waarde 1 krijgt en eindigt wanneer A = 0 wordt. Als B verandert gedurende de tijd dat A = 1 is moet de uitgang U van de schakeling 1 worden. U blijft dan 1 zolang A = 1 is. In de overige gevallen is U = 0. Zie als toelichting het tijddiagram in fig. 8.26.a.
Men kan voor dit probleem een level mode specificatie opstellen. Met enige ervaring vindt men echter ook de oplossing volgens fig. 8.26.b wel. Het signaal B wordt hierin als klokpuls aangeboden aan twee edge-triggered D flip-flops. Als B van 0 → 1 verandert, dan reageert de linker flip-flop. De 1 → 0 overgang wordt in de rechter D flip-flop gedetecteerd. Het interval wordt ingesteld via de clear-ingang. Hiervan is aangenomen dat het een directe instelingang is en de klokpuls en D ingang overheerst.

8.8. Symbolen voor flip-flops

Bij het level mode model van schakelingen hebben we geen onderscheid gemaakt tussen de functie van de verschillende ingangssignalen. Dit in tegenstelling tot bij het clock mode model, waarin o.a. het klokpulssignaal een andere functie heeft dan de datasignalen op de D resp. J en K ingangen. Wel staan het klokpulssignaal en de datasignalen in een onderlinge relatie t.o.v. elkaar. Het timingsysteem van de flip-flop specificeert wanneer de datasignalen stabiel moeten zijn. Gewoonlijk geeft men het timingsysteem slechts aan bij de beschrijving in de catalogus. In het symbool is slechts het typenummer vermeld. Informatie over de timing, de indeling in:

edge-triggered (flankgestuurd)
pulse-triggered (pulsgestuurd)
data-lockout

kan men gemakkelijk in de symbolen opnemen. In deze tekst geven we de informatie over de timing aan volgens fig. 8.27. Een driehoekje aan de ingang *(dyna-*

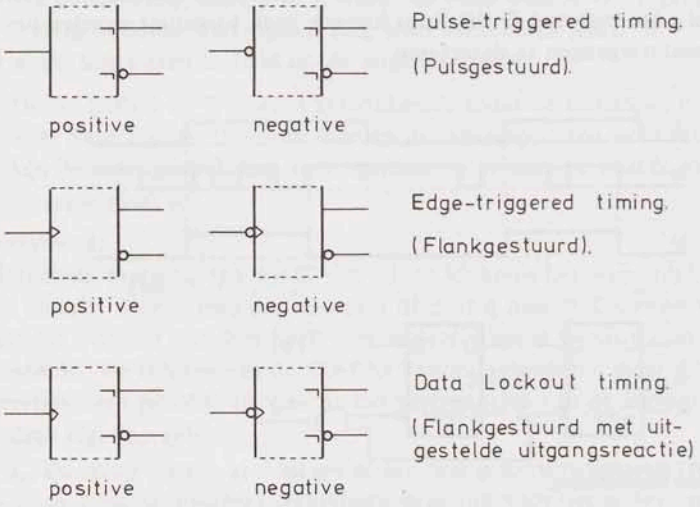
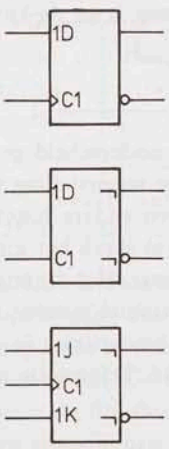


fig. 8.27. Het aangeven van de timing in symbolen.

mic input) moet men zo interpreteren dat de inwendige klokpuls aanwezig is ("1" is) als uitwendig de actieve klokflank wordt aangeboden. De klokflank wordt als het ware omgezet in een smal pulsje. De data behoeven slechts stabiel te zijn rond deze overgang. De *uitsteloperator* ¬ geeft aan dat de uitgang pas op de andere klokflank wordt aangepast.
Wordt niets bij de klokingang C *(Command input)* aangegeven, dan komt het uitwendig 1 zijn van de klokpuls overeen met het inwendig 1 zijn van de klokpuls. Een inversiecirkel aan de klokingang geeft aan dat het uitwendige 1-niveau correspondeert met het inwendige 0-niveau. We nemen aan dat de flip-flop slechts reageert op het inwendige 1-niveau (afspraak!).



D flip-flop met edge-triggered timing.
Ingangen stabiel rond opgaande flank. Uitgang reageert na opgaande flank. Met gewone en geïnverteerde uitgang.

D flip-flop met pulse triggered timing. Ingangen stabiel van voor opgaande flank tot na neergaande flank. Uigang reageert na neergaande flank. Met gewone en geïnverteerde uitgang.

Data lockout flip-flop. Bemonstert op opgaande flank. Ingangen stabiel rond opgaande flank. Uitgang reageert na neergaande flank. Met gewone en geïnverteerde uitgang.

fig. 8.28. Het aangeven van de data-ingangen.

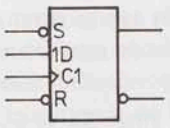
Om de afhankelijkheid van de data-ingangen en de klokpuls aan te geven noteren we een cijfer na het symbool C bij de klokpulsingang. Vóór de symbolen bij de data-ingangen plaatsen we hetzelfde cijfer als bij de corresponderende klokingang. De symbolen D, J en K hebben de gebruikelijke betekenis en mogen niet voor andere doeleinden worden gebruikt. Fig. 8.28 geeft enkele voorbeelden.
In principe kan men met flip-flops met uitsluitend klok- en data-ingangen volstaan. Er zijn redenen om signalen met andere functies te introduceren:

- Bij het aanzetten van een schakeling is de beginstand van de geheugenelementen meestal onbepaald. Een snelle mogelijkheid om alle flip-flops te resetten is dan gewenst.
- Na afloop van een serie bewerkingen wil men de schakeling snel in een bepaalde beginstand kunnen zetten.
- Bepaalde delen van een schakeling veranderen in een zekere fase van het proces niet van stand. Voorbeelden ervan zijn registers die voor tijdelijke opslag van data dienen. Bezitten deze registers een enable-ingang dan is het realiseren van de gewenste onthoudfunctie gemakkelijk.
- Tellers bezitten vaak ingangen om te kunnen omschakelen van Op- naar Neertellen en omgekeerd. Schuifregisters voor linksom resp. rechtsom schuiven, enz.

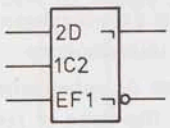
De signalen met bovengenoemde functies zijn belangrijk voor de besturing van processen in digitale schakelingen. Zij overheersen vaak de data-ingangen en maken het mogelijk de volgorde van een serie bewerkingen te wijzigen. Ingangen van circuits met een van de genoemde functies worden *besturings-* of *instelingangen* genoemd. Zij kunnen worden onderscheiden in twee categoriëen:

- *Rechtstreeks werkende instelingangen (direct acting inputs)*
  De flip-flop of het circuit reageert direct als een van deze ingangen zijn actieve niveau bereikt. De flip-flop stelt zich onafhankelijk van de klokpuls in. Bij afspraak is inwendig het actieve niveau gelijk aan het "1"-niveau. Voor deze ingangen wordt in catalogi een minimale tijdsduur voorgeschreven.
- *Voorbereidende instelingangen (preparatory inputs)*
  Deze staan onder besturing van de klokpuls. Om deze reden zijn ook voor deze ingangen setup en hold time gespecificeerd.

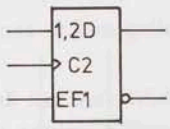
De afhankelijkheid van de klok- en data-ingangen van de instelingangen, alsmede de onderlinge prioriteit wanneer verschillende instelingangen aanwezig zijn, kan gemakkelijk in de symbolen worden aangegeven. Fig. 8.29 geeft een toelichting. Staan twee of meer cijfers bij een bepaalde ingang voor de letteraanduiding, dan moet aan elk van de voorwaarden worden voldaan. Geen cijfer of letteraanduiding betekent dat de ingang rechtstreeks werkend is.



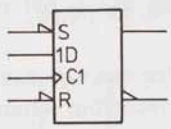
(a) D flip-flop, edge-triggered, met directe Set en Reset ingang (Preset en Clear) waarvan het uitwendig 0-niveau het actieve niveau is. Bezit gewone en geïnverteerde uitgang.



(b) D flip-flop, pulse-triggered met enable ingang die de klok onderbreekt. Zie fig. 8.22.a.



(c) D flip-flop, edge-triggered. Enable beïnvloedt D-ingang. Zie fig. 8.22.b.



(d) Als (a), Hoog-Laag beschrijving, met polariteitsindicator.

fig. 8.29. Voorbeelden van flip-flops met instelingangen.

De symbolen (a) t/m (c) van fig. 8.29 sluiten aan bij een beschrijving in "nullen en enen". Bij toepassing van *positieve logica* geldt dan: uitwendig L $\Leftrightarrow$ 0 en H $\Leftrightarrow$ 1. Het is ook mogelijk om direct in het symbool aan te geven welk uitwendig niveau met de inwendige 1 correspondeert. Hiertoe is de *polariteitsindicator* ingevoerd. Is deze aan een ingang og uitgang aangegeven, dan correspondeert uitwendig L met inwendig 1 (en uitwendig H met inwendig 0). Draagt een in- of uitgang geen polariteitsindicator, dan correspondeert uitwendig H met inwendig 1. Zie het symbool in fig. 8.29.d als voorbeeld.

Voor de poortsymbolen hebben we in deze tekst gekozen voor de symbolen uit o.a. de Texas Instruments catalogus. In de vorm van het poortsymbool wordt reeds veel informatie gegeven. Dit vergroot de leesbaarheid van schema's. Voor de flip-flops e.d. zijn afwijkende symbolen geïntroduceerd. De redenen hiervoor zijn:

1. In de meeste catalogi komt geen aanduiding van de onderlinge prioriteit van de ingangen voor. Vooral bij het opstellen van besturingsspecificaties is dit zeer hinderlijk.
2. De in de catalogi gegeven bouwsteenbeschrijvingen zijn verre van consistent opgebouwd. Wat onder andere te denken van de vermelding "clear" binnen en buiten een symbool, terwijl op het aansluitpunt een negatiecirkeltje is getekend.

De symbolen voor de nog te introduceren circuits zullen ter plaatse worden gegeven.

Opgaven

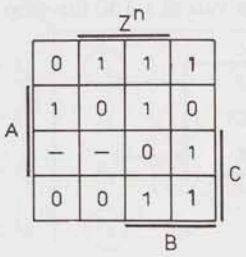
8.1. Bij trekkers kon het SR = 11 ingangsprobleem niet worden opgelost door trekkers met overheersende set/reset te gebruiken. Zie par. 5.3. Is het dan wel zinvol om de S-R-Q, de S-R-S en de S-R-R flip-flop te introduceren?

8.2. Gegeven is een J-K flip-flop Y. Is de bewering:

"De formules voor de J en K ingangssignalen kunnen altijd zo opgesteld worden dat noch Q_y, noch $\bar{Q}_y$ er in voorkomt."

juist of niet juist? Motiveer uw antwoord.

8.3.

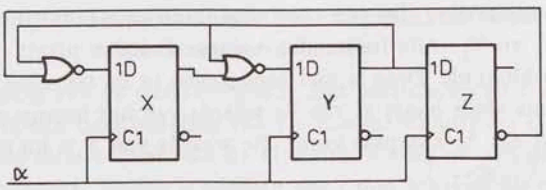


Z^{n+1} = functie (Z^n, A, B, C)

In het Karnaughdiagram is vastgelegd hoe van een flip-flop Z de gewenste stand na de klokpuls (Z^{n+1}) afhangt van de stand voor de klokpuls (Z^n) en de ingangscombinatie $[ABC]^n$.
Bepaal zo eenvoudig mogelijke formules voor de ingangssignalen als de gegeven flip-flop een:

a. D flip-flop is,
b. T flip-flop is,
c. J-K flip-flop is.

8.4.



Gegeven is een schakeling die bestaat uit drie D flip-flops X, Y en Z en twee NOR's. Geef in een waarheidstabel aan hoe de nieuwe stand $[XYZ]^{n+1}$ afhangt van de huidige stand $[XYZ]^n$ van de flip-flops.

8.5. Geef aan hoe een D flip-flop als tweedeler gebruikt kan worden.
Idem voor een J-K flip-flop.

8.6. Gegeven is een flip-flop waarvan de logische werking is beschreven met de formule

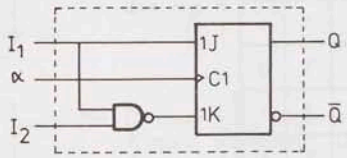
$$Q^{n+1} = [I_1 + I_2 \bar{Q}]^n .$$

a. Welk type flip-flop is deze schakeling als
   – $I_2 = 0$ is,
   – $I_2 = 1$ is.
b. Vul de volgende tabel aan:

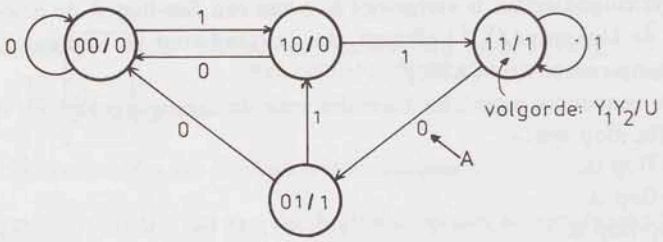
Q^n	Q^{n+1}	I_1	I_2
0	0	·	·
0	1	·	·
1	0	·	·
1	1	·	·

c. Modificeer deze flip-flop tot een J-K flip-flop.
d. Maak deze flip-flop uitgaande van een J-K flip-flop.

8.7. Een flip-flop is opgebouwd uit een J-K flip-flop en een NAND.
a. Bepaal de formule voor Q^{n+1}.
b. Toon ook het omgekeerde aan, d.w.z. geef aan hoe deze formule in een schakeling met behulp van een J-K flip-flop gerealiseerd kan worden.

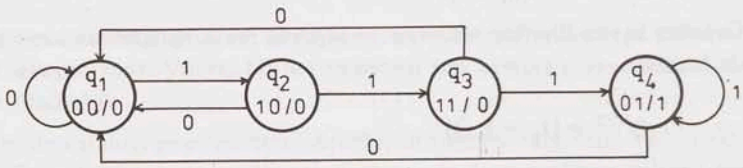


8.8.



Het gegeven toestandsdiagram beschrijft de werking van een clock mode sequentiële schakeling. De vier toestanden zijn gecodeerd met twee J-K flip-flops Y_1 en Y_2. Alle toestandsovergangen vinden plaats onder besturing van de klokpuls. Deze is niet aangegeven in de tekening. Welke overgang er plaats vindt hangt af van de waarde van het ingangssignaal A op het moment dat de klokpuls komt. De waarde van A is bij de pijlen geschreven.
a. Plaats in een waarheidstabel alle combinaties van waarden van $[Y_1\ Y_2]^n$ en A. Vul deze tabel aan met de bijbehorende waarden $[Y_1\ Y_2]^{n+1}$ en de uitgangswaarde U.
b. Bepaal zo eenvoudig mogelijke formules voor de J en K ingangssignalen.

8.9.



In het toestandsdiagram is de werking beschreven van een sequentiële schakeling in de clock mode. Alle toestandsovergangen vinden plaats onder commando van de klokpuls. De toestandscodering met twee J-K flip-flops Y_1 en Y_2 is reeds gegeven, alsmede de gewenste uitgangscodering.
a. Omschrijf in woorden de werking van deze schakeling. Het ingangssignaal A heeft de waarde 0 of 1, maar is stabiel op de actieve momenten van de klokpuls.
b. Ontwerp een eenvoudige realisatie van de schakeling. Gegeven is dat van A ook de inverse waarde beschikbaar is.

8.10. In de onderstaande toestandstabel is de werking gespecificeerd van een D flip-flop van het type edge triggered. De flip-flop reageert op de opgaande flank van de klokpuls. Voor de toestandscodering worden twee trekkers Y_1 en Y_2 gebruikt. De toestandscodering is reeds gegeven.

$Y_1 Y_2$ r \ aD	00	01	10	11	U
0 0 ↔ r_1	(**r_1**)	r_2	(**r_1**)	(**r_1**)	0
1 0 ↔ r_2	r_1	(**r_2**)	r_3	r_3	0
1 1 ↔ r_3	r_4	(**r_3**)	(**r_3**)	(**r_3**)	1
0 1 ↔ r_4	(**r_4**)	r_3	r_1	r_1	1

a. Leid de formules af voor S en R van de trekkers Y_1 en Y_2.

b. Ga na of en zo ja hoe de schakeling van fig. 8.24 uit de bovenstaande specificatie volgt. Ga vervolgens de overgang na van fig. 8.24 naar fig. 8.25.

8.11. Een schakeling heeft twee ingangssignalen a en t, waarvan a de klokpuls is. Het signaal t is bedoeld als een omschakelsignaal:

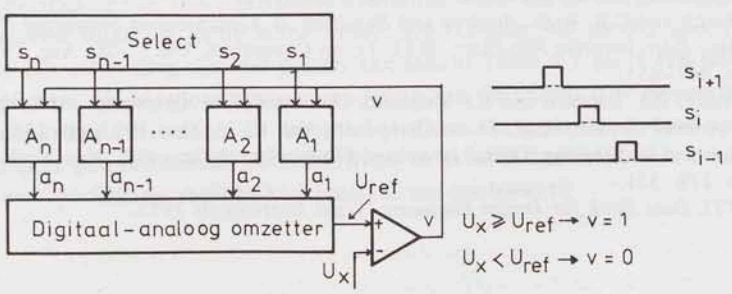
t = 0: De schakeling is een tweedeler, die reageert op de opgaande flank van a.

t = 1: De schakeling is een tweedeler, die reageert op de neergaande flank van a.

De uitgang van de schakeling mag niet veranderen als t wordt omgeschakeld. Na een omschakeling van t verandert de uitgang U weer op de eerstvolgende actieve flank van de klokpuls, die bij de dan geldende waarde van t behoort. Verder is gegeven dat t niet schakelt als a verandert.
Deze schakeling kan op twee manieren worden ontworpen:
- Ga uit van een bestaande tweedeler en schakel in de kloklijn.
- Stel een volledige level mode specificatie op van de schakeling en realiseer deze volgens de reeds gegeven methode.

Ga beide mogelijkheden na. (Zie ook opgave 6.11)

8.12. Een digitaal-analoog omzetter zet een binair gecodeerd getal $(a_n a_{n-1} \ldots a_2 a_1)$ om in een analoge spanning U_{ref}. Deze spanning wordt in een verschilversterker vergeleken met een onbekende spanning U_x. Van het binaire getal is a_n het hoogstwaardige bit.



Uitgaande van een digitaal-analoog omzetter en een verschilversterker kan een analoog-digitaal omzetter worden ontworpen. In de getekende figuur is een van de mogelijke uitvoeringsvormen geschetst.

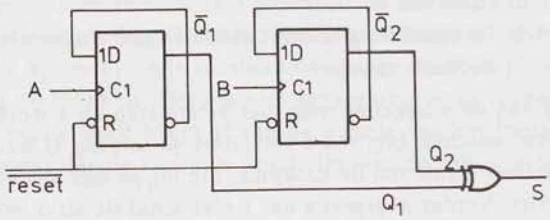
Een schakeling "select" selecteert achtereenvolgens de schakelingen A_n tot en met A_1. Zie ook het tijddiagram. De uitgang a_i van schakeling A_i voldoet aan:

- $a_i = 0$ als het blokje A_i nog niet geselecteerd is.
- $a_i = 1$ als $s_i = 1$ is.
- na selectie van het blokje is $a_i = 1$ dan en slechts dan als $U_x \geqslant U_{ref}$ was tijdens de selectie.

Aangenomen kan worden dat de schakeling select schakelt wanneer de omzetter weer tot rust is gekomen.

a. Specificeer de werking van de schakeling A_i met ingangen s_i en v en uitgang a_i via de level mode in een toestandsdiagram. Voorzieningen voor een reset van het systeem mogen buiten beschouwing blijven.
b. Realiseer de schakeling. Overweeg of het gebruik van flip-flops voordelen kan bieden.

8.13. Zie voor deze opgave ook de probleemstelling van opgave 7.14.
Een ontwerp voor deze probleemstelling ziet er uit als in de getekende figuur gegeven is. De D flip-flops in de schakeling zijn van het type positive edge-triggered.



De schakeling berust op het principe dat beide flip-flops tweedelen op de opgaande flank van A resp. B.
Gevraagd wordt na te gaan of deze schakeling het gestelde probleem wel/niet realiseert.

Literatuur

1. T.A. Dolotta and E.J. McCluskey, *The Coding of Internal States of Sequential Circuits,* IEEE Tr. on Electronic Computers, Vol. EC–13, Oct. 1964, pp. 549–562.
2. P.S. Noe and V.T. Rhyne, *Optimum State Assignment for the D flip-flop,* IEEE Tr. on Computers, Vol. C–25, March 1976, pp. 306–311.
3. J.R. Smith and C.H. Roth, *Analysis and Synthesis of Asynchronous Sequential Networks Using Edge-Sensitive Flip-Flops,* IEEE Tr. on Computers, Vol. C–20, Aug. 1971, pp. 847–855.
4. J.R. Story, H.J. Harrison and E.J. Reinhard, *Optimum State Assignment for Synchronous Sequential Circuits,* IEEE Tr. on Computers, Vol. C–21, Dec. 1972, pp. 1365–1373.
5. H. Taub and D. Schilling, *Digital Integrated Electronics,* McGraw-Hill, New York 1977, pp. 278–321.
6. *The TTL Data Book for Design Engineers,* Texas Instruments 1977.

9. TELLEN

9.1. Specificatie van telschakelingen

Een belangrijk aspect van het ontwerpen van complexe digitale schakelingen is de overzichtelijkheid van de opbouw van de schakeling. Een systematisch opgebouwde schakeling kan gemakkelijk getest worden en ook het opsporen van fouten kost niet veel tijd. Het streven naar een overzichtelijke opbouw houdt o.a. in dat men elke bewerking op data tracht terug te brengen tot één of een opeenvolging van enkele standaardhandelingen. Tevens kan men dan zo veel mogelijk van gestandaardiseerde bouwstenen gebruik maken. Tot de bewerkingen waarvoor circuits in vele uitvoeringen beschikbaar zijn behoren optellen en aftrekken, vermenigvuldigen en delen, vergelijken, schuiven, enz. Sommige van deze bewerkingen, bijvoorbeeld vermenigvuldigen, zijn terug te brengen tot herhaald optellen en schuiven. In dit en enkele volgende hoofdstukken worden enkele standaardbewerkingen nader onderzocht, waarna de opbouw van grotere synchrone systemen kan worden behandeld.

Een van de meest elementaire bewerkingen uit de schakeltechniek is het (af)tellen van pulsen. Behalve het tellen van aantallen is een belangrijk toepassingsgebied van tellers het vastleggen van de momentele stand van een proces. Het bijhouden van hoe ver de afwikkeling van een proces gevorderd is komt zeer veel voor, vooral daar waar een gewenste logische bewerking op data in een aantal opeenvolgende stappen wordt uitgevoerd.
Een bekend voorbeeld is het gebruik van een programmateller in een computer. Ook serie-optellers en -vermenigvuldigers bevatten een dergelijke teller.
Naar hun opbouw kunnen telschakelingen onderscheiden worden in asynchrone en synchrone tellers. In een *synchrone telschakeling* staan alle geheugenelementen onder directe besturing van de telpuls, welke puls dan de rol van klokpuls voor alle telsecties krijgt toebedeeld. Bij de *asynchrone telschakelingen* staan niet alle geheugenelementen onder directe besturing van de telpuls. Soms staat slechts één geheugenelement onder besturing van de telpuls zoals bij de nog te behandelen tweedelerteller. De keuze tussen een asynchrone en een synchrone opzet van een teller hangt o.a. af van de beschikbare bouwstenen (relais, trekkers, flip-flops), maar ook het toepassingsgebied is van invloed. In een volledig synchroon werkende schakeling kan men meestal geen asynchrone tellers gebruiken. Van de huidige als geïntegreerd circuit beschikbare tellers is soms de inwendige opbouw niet geheel synchroon, hoewel deze circuits voor wat betreft hun uitwendig gedrag aan de pennen meestal goed in geklokte synchrone schakelingen passen.

De specificatie van telschakelingen kan afhankelijk van de toepassing geschieden in de *level mode* of in de *clock mode*. De tabellen 9.1 en 9.2 specificeren beide een schakeling die vier pulsen kan tellen. Tabel 9.1 geeft een level mode specificatie. Voor iedere niveauverandering van de telpuls a is gespecificeerd wat de schakeling moet doen. In tabel 9.2 is alleen het gewenste gedrag voor en na de telpuls gespecificeerd. De beschrijving in tabel 9.2 is in zoverre niet compleet dat een specificatie van het klokpulssysteem ontbreekt.

q \ a	0	1	0	1
q_0	(q_0)	q_1	u_0	u_0
q_1	q_2	(q_1)	–	u_0
q_2	(q_2)	q_3	u_1	u_1
q_3	q_4	(q_3)	–	u_1
q_4	(q_4)	q_5	u_2	u_2
q_5	q_6	(q_5)	–	u_2
q_6	(q_6)	q_7	u_3	u_3
q_7	q_0	(q_7)	–	u_3

tabel 9.1. Level mode specificatie van telschakeling.

r^n	r^{n+1}	U
r_0	r_1	u_0
r_1	r_2	u_1
r_2	r_3	u_2
r_3	r_0	u_3

tabel 9.2. Clock mode specificatie van telschakeling.

De codering van de toestanden in tabel 9.1 moet vrij zijn van kritische raceconditities. Een realisatie met een minimum aantal geheugenelementen (3) leidt daarom tot een progressieve codering van de acht toestanden. De codering van de toestanden in tabel 9.2 daarentegen is geheel vrij. Vaak ziet men dan dat de toestandscodering aangepast wordt aan de gewenste uitgangscodering, waardoor een aparte schakeling voor de uitgangssignalen vermeden wordt. Een bijkomend voordeel is dat alle uitgangen tegelijk veranderen en geheel vrij van spikes zijn. Onder omstandigheden kan dit belangrijk zijn.
De interpretatie van de tabellen 9.1 en 9.2 is verschillend. Globaal kan gezegd worden dat toestand q_0 overeenkomt met toestand r_0, toestand q_2 met toestand r_1, toestand q_4 met toestand r_2 en tenslotte toestand q_6 met toestand r_3. De overige toestanden in tabel 9.1 beschrijven het gedrag tijdens de ontvangst van een telpuls. Zoals reeds gezegd, deze informatie ontbreekt in tabel 9.2 en wordt voor deze tabel geacht beschreven te zijn via de specificatie van het klokpulssysteem. Uit de voorafgaande hoofdstukken zal duidelijk zijn dat overgangsverschijnselen in een geklokte schakeling eigenlijk niet interessant zijn.
Tabel 9.1 geeft duidelijk aan dat de uitgangsreactie van de geheugenelementen in de schakeling plaats heeft als de telpuls a van $1 \rightarrow 0$ gaat. Als a van $0 \rightarrow 1$ gaat, behoudt de uitgang U zijn oude waarde. Het klokpulssysteem behorend bij tabel 9.2 moet dus de uitgangsreactie op de neergaande flank presenteren.
In de volgende paragrafen wordt de algemene opzet van telschakelingen behandeld. In tegenstelling tot hetgeen meestal geschiedt, wordt een bespreking van als geïntegreerd circuit verkrijgbare tellers grotendeels vermeden. Inzicht in de factoren die een tellerontwerp beïnvloeden moet het mogelijk maken verkrijgbare schakelingen te analyseren en op een juiste wijze toe te passen.

9.2. Synchrone binaire telschakelingen tot 2^n

Tabel 9.3 specificeert de werking van een synchrone binaire teller, welke in de gewone binaire code acht pulsen kan tellen. De teller bestaat o.a. uit drie geheugenelementen X, Y en Z, waarvan de inhoud het binaire gewicht 2^2, 2^1 en 2^0 toegekend krijgt.

Stap	$[X\,Y\,Z]^n$	$[X\,Y\,Z]^{n+1}$
0	0 0 0	0 0 1
1	0 0 1	0 1 0
2	0 1 0	0 1 1
3	0 1 1	1 0 0
4	1 0 0	1 0 1
5	1 0 1	1 1 0
6	1 1 0	1 1 1
7	1 1 1	0 0 0

tabel 9.3. Synchrone teller tot acht pulsen.

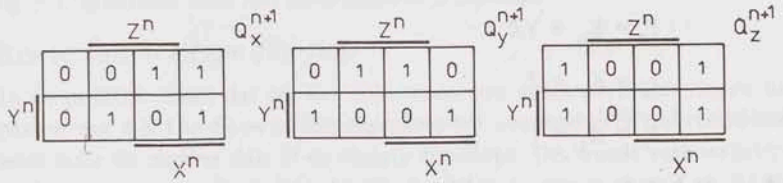


fig. 9.1. Karnaughdiagrammen.

De werking van de teller kan ook in drie Karnaughdiagrammen worden gespecificeerd, zoals fig. 9.1 laat zien. In elk diagram is voor één sectie de gewenste waarde Q^{n+1} gespecificeerd behorend bij de huidige stand van de teller, aangeduid met $[XYZ]^n$. De huidige stand is in fig. 9.1 via de randschriften van de diagrammen aangeduid. We onderzoeken de realisatie van de schakeling met enkele types flip-flops, te beginnen met de J-K flip-flop. De werking van deze flip-flop is gegeven via de tabellen 9.4.a en 9.4.b.

J^n	K^n	Q^{n+1}
0	0	Q^n
0	1	0
1	0	1
1	1	$\overline{Q}^n$ (a)

Q^n	Q^{n+1}	J^n	K^n
0	0	0	–
0	1	1	–
1	0	–	1
1	1	–	0 (b)

tabel 9.4. Werking J-K flip-flop.

Met behulp van deze tabellen wordt tabel 9.3 aangevuld tot tabel 9.5. In deze tabel is voor elke gewenste toestandsovergang $Q^n \rightarrow Q^{n+1}$ van de geheugenelementen X, Y en Z de bijbehorende waarde voor de J en K ingangssignalen gegeven. De bijbehorende formules kunnen weer met Karnaughdiagrammen worden bepaald. Deze Karnaughdiagrammen kunnen ook rechtstreeks uit fig. 9.1 worden afgeleid met behulp van tabel 9.4.

Stap	$[X\,Y\,Z]^n$	$[X\,Y\,Z]^{n+1}$	$J_x^n\ K_x^n$	$J_y^n\ K_y^n$	$J_z^n\ K_z^n$
0	0 0 0	0 0 1	0 –	0 –	1 –
1	0 0 1	0 1 0	0 –	1 –	– 1
2	0 1 0	0 1 1	0 –	– 0	1 –
3	0 1 1	1 0 0	1 –	– 1	– 1
4	1 0 0	1 0 1	– 0	0 –	1 –
5	1 0 1	1 1 0	– 0	1 –	– 1
6	1 1 0	1 1 1	– 0	– 0	1 –
7	1 1 1	0 0 0	– 1	– 1	– 1

tabel 9.5. Ingangsvoorwaarden voor telschakeling.

De eenvoudigste formules zijn:

$$J_x = K_x = YZ$$

$$J_y = K_y = Z$$

$$J_z = K_z = 1\ .$$

Fig. 9.2 geeft de bijbehorende schakeling, welke is uitgevoerd met pulsgestuurde (pulse-triggered) J-K flip-flops. Op de neergaande klokflank vindt de uitgangsaanpassing plaats.

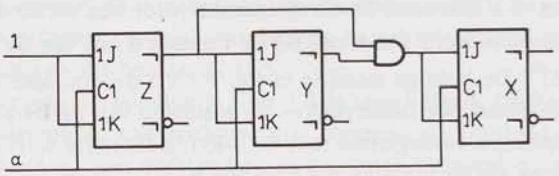


fig. 9.2. Synchrone teller met pulse-triggered J-K flip-flops.

Opmerking 1
Sommige als geïntegreerd circuit verkrijgbare J-K flip-flops hebben aan de informatie-ingangen J en K reeds een AND-ingangspoort. De extra AND-poort in fig. 9.2 is dan overbodig.

Opmerking 2
In tekeningen wordt bij voorkeur links-naar-rechts als richting van de informatiestroom aangehouden. Men let er dus op dat van telschakelingen het laagstwaardige bit links staat.

Dezelfde teller kan ook met D flip-flops worden gerealiseerd. In dit geval moet de gewenste nieuwe uitgangswaarde Q^{n+1} als informatie op de D-ingang worden aangeboden. De formules voor de D-ingangssignalen kunnen rechtstreeks uit fig. 9.1 worden bepaald. (Ga dit na.) Deze formules zijn:

$$D_x = X\bar{Z} + X\bar{Y} + \bar{X}YZ$$

$$D_y = \bar{Y}Z + Y\bar{Z}$$

$$D_z = \bar{Z}$$

De resulterende schakeling staat in fig. 9.3. In de praktijk zal men voor de combinatoriek in dit geval AND-OR-INVERT poorten nemen. De formules van D_x t/m D_z moeten dan geïnverteerd worden. De flip-flops zijn pulsgestuurd (pulse-triggered).

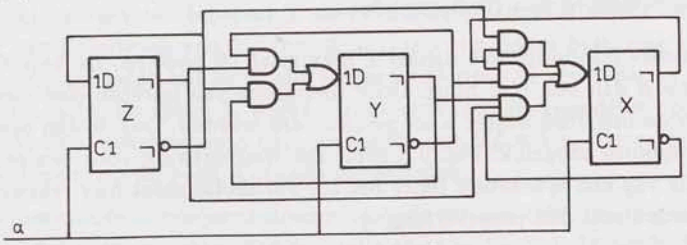


fig. 9.3. Synchrone teller met pulse-triggered D flip-flops.

Keuze van het type flip-flop

In de praktijk blijkt dat bij het ontwerpen van *volgorde schakelingen* het toepassen van J-K flip-flops in het algemeen tot eenvoudigere poortschakelingen leidt voor de sturing dan D en andere flip-flops. Dit wordt veroorzaakt door het grote aantal don't cares in de Karnaughdiagrammen voor J en K. J òf K is geheel vrij te kiezen voor elke toestandsovergang. Dit is in de praktijk aanleiding tot een vrijwel automatische voorkeur voor het gebruik van J-K flip-flops. Een illustratie van dit ervaringsfeit vormen de schakelingen in fig. 9.2 en 9.3. Deze schakelingen zijn uitwendig geheel identiek in hun gedrag, de inwendige opbouw echter niet!
De gevraagde telschakeling kan ook met behulp van T flip-flops worden gerealiseerd. Dit ontwerp wordt aan de lezer overgelaten.

Uitbreiding van synchrone telschakelingen voor grote n

Het ontwerp van telschakelingen tot $2^n > 8$ pulsen verschilt niet wezenlijk van de hierboven beschreven tellers. Voor grote n worden de formules voor J en K:

$$J_z = K_z = 1$$

$$J_y = K_y = Z$$

$$J_x = K_x = YZ$$

$$J_w = K_w = XYZ$$

$$J_v = K_v = WXYZ$$

enz.

De formules breiden zich voor iedere volgende sectie op regelmatige wijze uit. Voor elke volgende sectie is een sturende poort nodig die één ingang meer bezit dan die van de vorige sectie. Bij een teller tot 2^{20} (ca. 10^6 pulsen) houdt dit in dat de sturende poort voor de hoogstwaardige sectie 19 ingangen moet bezitten, een onpraktisch groot aantal.
Een oplossing voor dit probleem vormt de schakeling waarvan de principe-opbouw in fig. 9.4 is getekend. Een nadeel van deze oplossing is dat de opeenvolgende poortvertragingen in de besturing accumuleren.

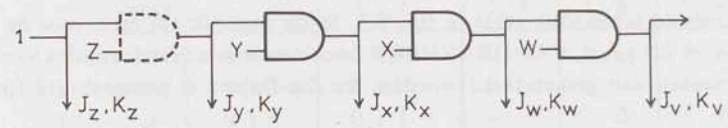


fig. 9.4. Stuurschakeling voor synchrone teller.

Voor 20 secties doorloopt het signaal Z maximaal 18 poorten. De bereikbare telfrequentie is dan ook niet hoog. Door toepassing van poorten met meer ingangen kan de insteltijd echter weer gereduceerd worden. Ook is dan een sectiegewijze opbouw mogelijk. Fig. 9.5 geeft het stuurnetwerk voor één sectie van vier bits van een synchrone teller tot 2^n. Per sectie moet hier rekening gehouden worden met één poortvertraging.

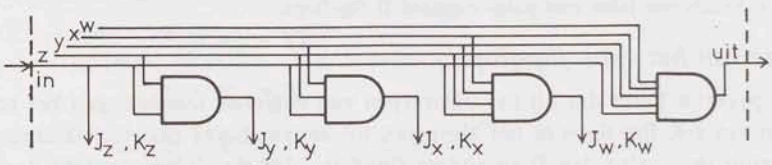


fig. 9.5. Stuurschakeling voor één sectie van vier bits.

Het boven geschetste probleem komt vrij algemeen voor bij het ontwerpen van grote telschakelingen. Geïntegreerde circuits bevatten meestal secties van vier bits (0–15 teller) of één decade van een teller in de BCD-code. Bij het doorverbinden van transportsignalen kan er een optelling van vertragingstijden optreden, waardoor de totale teller een lagere maximale telfrequentie kan bezitten dan de afzonderlijke secties. De opgegeven maximale telfrequentie in catalogi geldt per sectie, niet voor een teller die uit een aantal secties is opgebouwd.

9.3. Asynchrone binaire telschakelingen tot 2^n

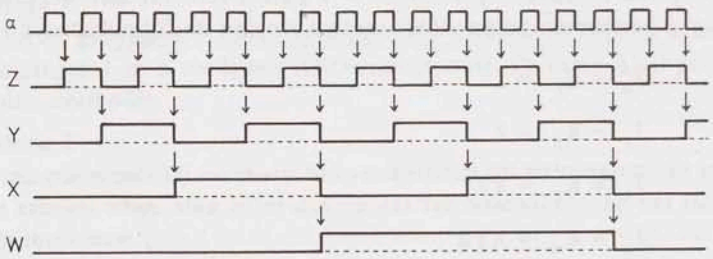


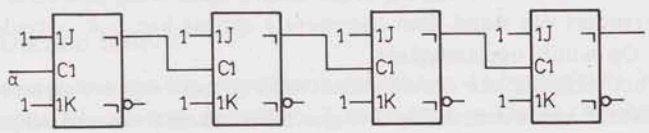
fig. 9.6. Tijddiagram van een synchrone teller.

Het tijddiagram van een synchrone binaire teller, zoals deze in de vorige paragraaf is ontworpen, is getekend in fig. 9.6.
Aangenomen is dat de flip-flops pulse-triggered zijn en hun uitgangswaarde aan de nieuwe situatie aanpassen op de neergaande klokpulsflank. Uit het diagram zien we:

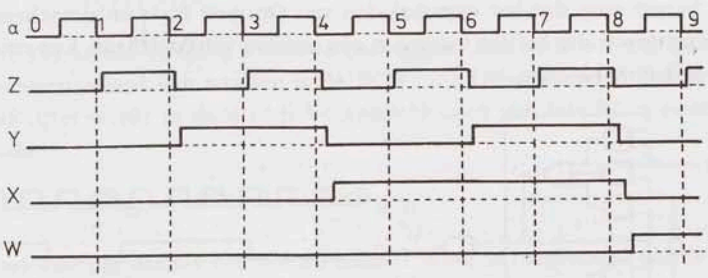
– De sectie Z (het laagstwaardige bit) reageert als tweedeler op de telpuls α. Bij een uitvoering met J-K flip-flops moet deze sectie dus als tweedeler geschakeld worden, m.a.w. $J_z = K_z = 1$. ($Q^{n+1} = [J\bar{Q} + \bar{K}Q]^n$).

- Sectie Y tweedeelt op a mits Z = 1. In de schakeling van fig. 9.2 is dit gerealiseerd door $J_y = K_y = Z$ te maken. Als Z = 0 is blijft Y staan, als Z = 1 is wordt de stand van flip-flop Y geïnverteerd als a van 1 → 0 gaat. Uit het tijddiagram volgt dat (afgezien van vertragingen) een juiste werking van Y ook bereikt kan worden wanneer Y als tweedeler geschakeld wordt onder besturing van de uitgang van flip-flop Z in plaats van onder besturing van a.
- Dit principe geldt ook voor alle volgende secties van een teller in de gewone binaire code. De sectie X werkt òf iets vertraagd als tweedeler op de uitgang van Y, òf synchroon op a op voorwaarde dat YZ = 1 is. Deze voorwaarde wordt weer op de J- en K-ingang aangeboden.

Het bovenstaande consequent doorgezet leidt tot een teller in de gewone binaire code zonder extra poorten. Een realisatie staat in fig. 9.7. In het bijbehorende tijddiagram is de prijs vermeld die voor deze oplossing betaald moet worden: een toenemende vertraging t.o.v. de klokpuls voor iedere sectie die aan de teller wordt toegevoegd.



(a) Asynchrone teller met tweedelers (Ripple counter).



(b) Tijddiagram.

fig. 9.7. Tweedelerteller.

De steeds toenemende vertraging t.o.v. de klokpulsflank houdt in dat de toepassing van de *asynchrone tweedelerteller* in geklokte schakelingen beperkt is. Voor die toepassingen, waarbij slechts het eindresultaat van belang is, kunnen zij gebruikt worden. Processen waarin op vaste tijdstippen het telresultaat van belang is, of waarbij een bepaalde stand tijdens het tellen gedetecteerd moet worden, maken meestal gebruik van volledig synchroon uitgevoerde telschakelingen.

Op-Neer tellen met tweedelers

De telschakeling in fig. 9.7 telt in de gewone binaire code van 0 (00 . . . 00) tot $2^n - 1$ (11 . . . 11) en keert op de 2^n-de klokpuls automatisch in de nulstand terug. Wordt echter niet de Q-uitgang met de klokingang van de volgende sectie verbonden maar de $\overline{Q}$-uitgang, dan telt een tweedelerteller terug. De laagstwaardige sectie echter blijft als tweedeler op a geschakeld en niet op $\bar{a}$! Met behulp van een tijddiagram kan het terugtellen zoals hierboven aangegeven is gemakkelijk worden nagegaan. Fig. 9.8 geeft het schema van een omschakelbare

tweedelerteller, die naar keuze op of neer kan tellen. De gewenste telmode kan met het signaal U/$\overline{D}$ (Up/$\overline{Down}$) worden ingesteld (Up ≡ 1 en Down ≡ 0).

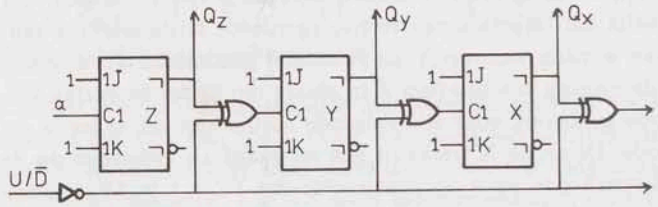


fig. 9.8. Op/Neer teller met omschakeldraad.

Een nadeel van de schakeling in fig. 9.8 is dat bij het omschakelen van Op naar Neer en omgekeerd de teller in een aantal gevallen een niet gewenste sprong maakt. Staat een uitgang van een sectie in de stand Q = 1, dan heeft het omschakelen van Op naar Neer (U/$\overline{D}$ = 1 → 0) tot gevolg dat de klokpulsingang van de volgende sectie een 1 → 0 overgang krijgt aangeboden. Deze sectie zal hierop reageren en verandert van stand. Een ongewenste sprong kan ook optreden als van Neer naar Op wordt omgeschakeld.
In principe is het mogelijk een omschakelnetwerk te ontwerpen waarmee "sprongloos" omgeschakeld kan worden. Fig. 9.9 geeft het schema van een schakeling waarmee dit kan. De werking van deze schakeling, die ontworpen is met de in de hoofdstukken 6 en 7 behandelde ontwerpmethode voor level mode schakelingen, berust erop dat het omschakelen van Op naar Neer en omgekeerd uitgesteld wordt voor die secties waarvoor een actieve klokpulsflank kan optreden tijdens het omschakelen.

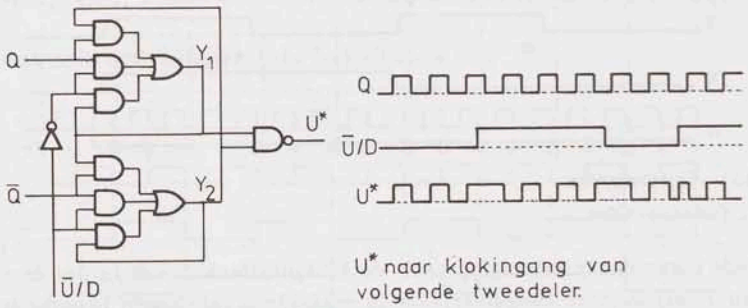


fig. 9.9. Omschakelnetwerk voor Op-Neer teller.

Opmerking
De flip-flops in de getekende tweedelertellers reageren op de neergaande klokpulsflank. Het is een aardig probleem na te gaan hoe deze tellers er uit komen te zien als de flip-flops op de opgaande flank reageren.

Op-Neer tellen met volledig synchrone tellers

Het omschakelen van tellers, waarvan elke flip-flop werkt onder directe besturing van de klokpuls *a* is gemakkelijker. Het omschakelen van deze tellers geschiedt nl. via de informatie-ingangen en niet via de klokingang. Om deze reden worden voor omschakelbare tellers bij voorkeur geheel synchrone tellers gebruikt. Een waarschuwing is op zijn plaats. Bij gebruik van omschakelbare synchrone

telsecties van vier bits, welke als geïntegreerd circuit verkrijgbaar zijn, moet men ervoor zorgen dat ook de koppeling van deze circuits tot een grote teller van k maal vier bits geheel synchroon verloopt. Zie ook de volgende paragraaf.

Toepassing

Tweedelertellers munten uit door hun eenvoud en hoge telfrequentie, maar passen slecht in geklokte schakelingen vanwege hun asynchrone karakter. Soms kan men dit probleem omzeilen:
Het komt veelvuldig voor dat in een bepaald proces een aantal pulsen moet worden afgeteld. Als het gewenste aantal bereikt is, moet zo snel mogelijk een melding gegeven worden. Tussenstanden van de teller zijn in dit geval niet van belang. Een eenvoudige schakeling voor dit probleem is een asynchrone tweedelerteller die, ingesteld op het gewenste aantal, van deze waarde naar de nulstand terugtelt. Immers, de overgang van de stand 1 (00 . . . 01) naar de nulstand geschiedt altijd onder directe besturing van de klokpuls! Op deze wijze vindt een volledig synchrone detectie van de eindstand plaats.

9.4. Decade tellers

Ten gevolge van het tweewaardige karakter van de bouwstenen ligt het voor de hand voor telprocessen de gewone binaire code te kiezen. Menigeen heeft er echter moeite mee om in de binaire rij

101110011

het getal 371 in het 10-tallig stelsel te herkennen. Een code die wat gemakkelijker geïnterpreteerd kan worden is de BCD-code (Binary Coded Decimal). Elk decimaal cijfer wordt in deze code voorgesteld door vier bits in de gewone binaire code

$$371_{10} \rightarrow 0011.0111.0001_{BCD}$$

Eventueel kan per decade een code-omzetter worden toegevoegd die de BCD representatie van elke decimaal in een 1-uit-10 code omzet.
Een decimale representatie van het resultaat van een telling kan op verschillende manieren worden bereikt. Ten eerste kan men uitgaan van een gewone binaire teller, waarna met een code-omzetter de decimale representatie wordt bepaald. Deze code-omzetters zijn als geïntegreerd circuit beschikbaar. Bij grote tellers vereist deze methode veel materiaal en/of tijd voor de omzetting. Bij voorkeur vermijdt men daarom deze code-omzetter. Als alternatief kan men de telschakeling opdelen in een aantal decadesecties. Elke decade telt binair van 0 (0000) tot 9 (1001). Bij elke tiende telpuls geeft een decade een transport door aan de volgende decade. Op deze wijze wordt direct in de BCD-code geteld. Eventueel kan nu per sectie van vier bits het resultaat via een code-omzetter in een andere code worden omgezet.
Tabel 9.6 beschrijft het ontwerp van een BCD-telsectie met vier J-K flip-flops W, X, Y en Z. Per regel is de huidige stand $[WXYZ]^n$ van de teller gespecificeerd, de gewenste nieuwe stand $[WXYZ]^{n+1}$ en de J- en K-waarden waarbij de gewenste overgang kan worden bereikt. De standen 10 t/m 15 en hun opvolgers zijn niet gespecificeerd. Onder normale bedrijfsomstandigheden komen zij niet voor. In principe volgt hieruit een aantal don't care condities.

Stand	$[W X Y Z]^n$	$[W X Y Z]^{n+1}$	J_w^n	K_w^n	J_x^n	K_x^n	J_y^n	K_y^n	J_z^n	K_z^n
0	0 0 0 0	0 0 0 1	0	–	0	–	0	–	1	–
1	0 0 0 1	0 0 1 0	0	–	0	–	1	–	–	1
2	0 0 1 0	0 0 1 1	0	–	0	–	–	0	1	–
3	0 0 1 1	0 1 0 0	0	–	1	–	–	1	–	1
4	0 1 0 0	0 1 0 1	0	–	–	0	0	–	1	–
5	0 1 0 1	0 1 1 0	0	–	–	0	1	–	–	1
6	0 1 1 0	0 1 1 1	0	–	–	0	–	0	1	–
7	0 1 1 1	1 0 0 0	1	–	–	1	–	1	–	1
8	1 0 0 0	1 0 0 1	–	0	0	–	0	–	1	–
9	1 0 0 1	0 0 0 0	–	1	0	–	0	–	–	1
10/15	don't care condities		–	–	–	–	–	–	–	–

tabel 9.6. Specificatie synchrone decadesectie.

De formules voor de signalen op de diverse J- en K-ingangen zijn:

$$J_w = XYZ \qquad K_w = Z$$

$$J_x = YZ \qquad K_x = YZ$$

$$J_y = \overline{W}Z \qquad K_y = Z$$

$$J_z = 1 \qquad K_z = 1 \,.$$

De schakeling staat in fig. 9.10. Via een AND-poort wordt de stand 9 ten behoeve van de volgende decade gedetecteerd.

In het bovenstaande voorbeeld van een decade telsectie zijn de standen 10 t/m 15 niet gespecificeerd. Van deze don't cares is een dankbaar gebruik gemaakt bij het vereenvoudigen van de ingangsformules. In het algemeen is dit toegestaan. Door allerlei oorzaken, o.a. bij het inschakelen is het mogelijk dat de teller in een stand komt die niet tot de normale cyclus behoort. Meestal stelt men de eis dat de schakeling in zo'n geval na een of enkele pulsen in een stand moet terugkeren die tot de normale cyclus behoort.

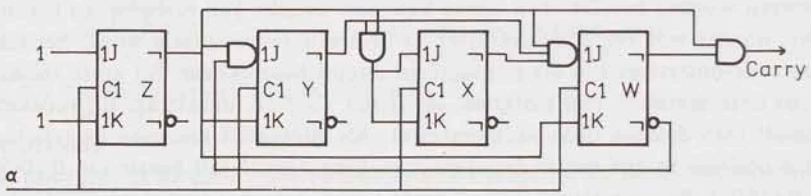


fig. 9.10. Decade telsectie.

In andere gevallen wordt een detectie gevraagd van een onjuiste stand. Deze aanvullende eisen betekenen dat men in het algemeen een onderzoek moet instellen naar het gedrag van een volgordeschakeling in de zgn. *verboden toestanden.* Tabel 9.7 geeft aanvullende informatie over de zojuist ontworpen decade telsectie van fig. 9.10. Bij elk van de standen 10–15 wordt de waarde van de J- en K-signalen uit de gegeven formules bepaald. Daarmee ligt de volgende stand $[WXYZ]^{n+1}$ in al deze gevallen vast. Het toestandsdiagram in fig. 9.11 geeft op een overzichtelijke wijze een beeld van het gedrag van de telsectie.

Stand	$[WXYZ]^n$	J_w^n	K_w^n	J_x^n	K_x^n	J_y^n	K_y^n	J_z^n	K_z^n	$[WXYZ]^{n+1}$
10	1010	0	0	0	0	0	0	1	1	1011
11	1011	0	1	1	1	0	1	1	1	0100
12	1100	0	0	0	0	0	0	1	1	1101
13	1101	0	1	0	0	0	1	1	1	0100
14	1110	0	0	0	0	0	0	1	1	1111
15	1111	1	1	1	1	0	1	1	1	0000

tabel 9.7. Specificatie van het gedrag in verboden toestanden.

Ten hoogste twee pulsen na een storing keert de teller weer in zijn normale cyclus terug.

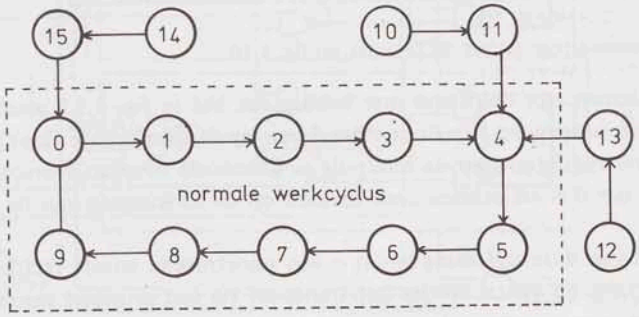


fig. 9.11. Toestandsdiagram decade telsectie (clock mode).

Wanneer een volgordeschakeling niet het gewenste gedrag bij een storing vertoont moet de specificatie worden aangepast. Stelt men de eis dat de schakeling weer tot zijn cyclus terugkeert, dan zal de ontworpen schakeling meestal voldoen. Een controle achteraf is echter wel nodig. Wordt de eis gesteld dat de schakeling na een storing naar een vaste stand gaat (de nulstand), dan dient men deze eis direct in de specificatie te verwerken. Het is echter niet noodzakelijk dat een teller door een storing in een verboden stand komt. Een sprong naar een tot de cyclus behorende toestand blijft onopgemerkt.

Transportpropagatie in synchrone tellers

Behalve de klokingang voor de telpuls bezitten tellers meestal een aantal instelingangen zoals *Preset*, *Reset* (Clear) en *Enable*. De instelingang *Load* dient om de teller in de gewenste beginstand te brengen. Deze instelingangen kunnen zowel een voorbereidend als een direct karakter bezitten. De hiermee verbonden problemen m.b.t. de interne organisatie van schakelingen worden in deel II uitvoerig toegelicht.

Een apart probleem vormt de koppeling van 4-bit tellersecties tot langere tellers. We lichten dit toe voor 4-bit BCD-tellers. Voor 4-bit binaire tellers gelden overeenkomstige beschouwingen. Als voorbeeld de BCD-tellersectie van fig. 9.10. Staat de sectie in stand 9 (1001), dan gaat de sectie op de eerstvolgende telpuls naar de stand 0 (0000). Tegelijk moet een *transport* worden doorgegeven naar de naastgelegen (hoger gewaardeerde) sectie. Het transportsignaal (carry) uit de in de schakeling getekende AND-poort kan daarbij eventueel worden benut als

klokpuls voor de naastgelegen sectie. Diens transport bedient dan weer de daarnaast gelegen sectie, enz.. Op deze wijze ontstaat een asynchroon opgebouwde teller, waarvan de bouwstenen bestaan uit (intern synchrone) 4-bit secties. Vergelijk ook de structuur van de teller in fig. 9.7.
Deze oplossing bezit de reeds genoemde nadelen van asynchrone tellers, zoals o.a. het niet gelijktijdig reageren van alle secties. Bovendien is het in fig. 9.10 niet uitgesloten dat er een spike op de transportuitgang staat bij de overgang van stand 7 (0111) naar stand 8 (1000). Zowel W als Z veranderen dan immers. Zie fig. 9.12.

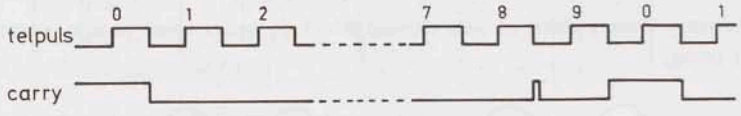


fig. 9.12. Transportsignaal van BCD-sectie uit fig. 9.10.

Deze problemen zijn oplosbaar met behulp van het in fig. 9.13 beschreven transportnetwerk (uitgangen flip-flops veranderen op de neergaande flank). Elke sectie werkt nu synchroon op de klokpuls a. Eventuele overgangsverschijnselen treden op als $a = 0$ is en hebben geen invloed op de klokingang van de volgende sectie.
Een nadeel van deze oplossing is dat a één poortniveau wordt vertraagd, terwijl bij een toenemend aantal secties het transport via een groeiend aantal niveaus van poorten wordt vertraagd.

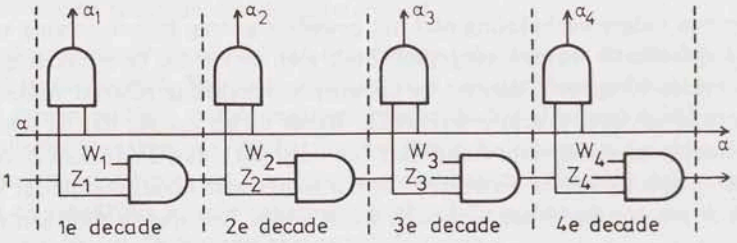


fig. 9.13. Klokpulsdistributie synchrone BCD-teller.

Fig. 9.14 beschrijft een verkrijgbare BCD-tellersectie (vergelijk SN 74160). Alle flip-flops hiervan staan onder directe besturing van de klokpuls a. Twee *enable-signalen* P en T en de *ripple carry* uitgang van elke sectie verzorgen het transport. Zowel P als T kunnen een sectie blokkeren resp. vrijgeven. P en T staan daartoe in een EN-relatie. Tevens schakelt T de ripple carry van elke sectie. We zullen aantonen dat hiermee het transportprobleem afdoende kan worden opgelost.
Vanwaar twee enable-ingangen? Zie daartoe fig. 9.15. Het bovenste deel van deze figuur beschrijft een transportnetwerk voor een uit vier BCD-secties bestaande teller. Elke sectie bezit één enable-ingang. Het transportsignaal van de eerste sectie doorloopt alle andere secties. Per sectie is het doorgeven van het transport afhankelijk van de stand. Is deze WXYZ = 1001, dan wordt een inkomend transport doorgegeven. Tevens stapt de sectie zelf ook op de eerstvolgende klokpuls. Bij langere tellers is de transportpropagatietijd ook hier

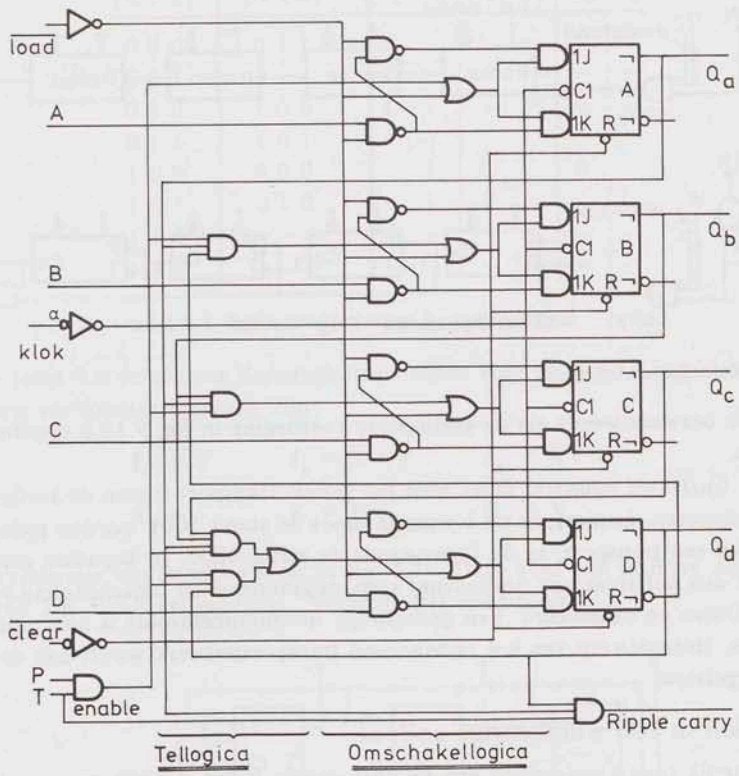


fig. 9.14. Instelbare BCD-teller.

een serieus probleem.
Merken we echter op dat:
- de eerste decade altijd telt,
- de tweede decade telt als de eerste in de stand 1001 staat en onthoudt in de overige gevallen enz.,

dan zien we dat de tweede decade tien klokpulsen eerder in de stand 1001 staat dan de eerste (laagstwaardige) sectie. Het transport naar de derde en hogere secties wordt nu gesplitst in twee bronnen,
- één vanuit de tweede sectie op doorloopbasis: enable T,
- één vanuit de eerste sectie op parallelbasis: enable P.

Het transport uit de eerste sectie moet in één klokperiode worden doorgegeven. Dat vanuit de tweede sectie dient, zoals reeds gezegd, binnen ca. tien perioden verwerkt te worden. Hetgeen ook bij een groeiend aantal secties gemakkelijk kan.
Via enable P en T ingangen is de bouw van snelle tellers mogelijk. Vaak echter wil men een teller ook via een uitwendig instelsignaal kunnen stoppen resp. vrijgeven. Zie datapad/besturing model in deel II. In principe kan dit geschieden via de enable T ingang van de laagstwaardige sectie. Zie fig. 9.15.b.
Ook hier weer nadelen:
- het vrijgeven van de teller geschiedt niet volledig parallel.
- de carry-uitgang van de hoogstwaardige sectie geeft niet aan of de teller overloopt.

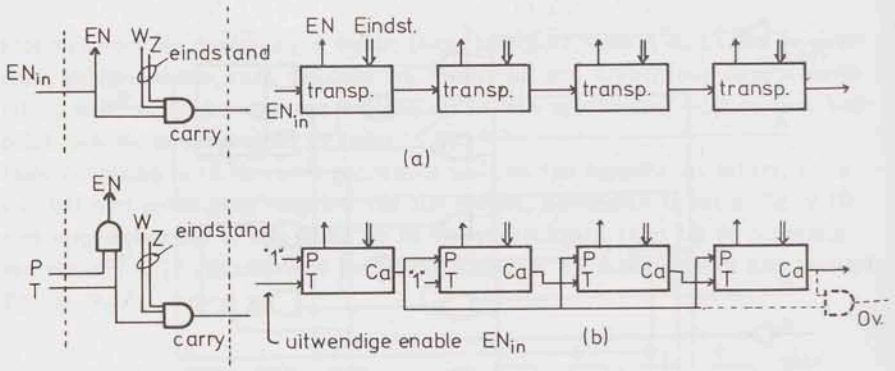


fig. 9.15. Transportdistributie.

Dit laatste bezwaar wordt via de gestippelde toevoeging in fig. 9.15.b opgeheven. Ga dit na.

Bij *snelle Up/Down counters* moet men het gehele transport tussen de secties parallel uitvoeren. Immers, in de Up-mode moet de stand 1001 worden gedetecteerd bij een transport, in de Down-mode de stand 0000. In bepaalde gevallen wordt dus het transport onderdrukt resp. gegenereerd bij omschakeling van Up naar Down en omgekeerd. Een gedeeltelijk doorlooptransport is hier niet toegestaan. Het ontwerp van het bijbehorend transportnetwerk wordt aan de lezer overgelaten.

9.5. Tellen in een willekeurige code

Een belangrijk toepassingsgebied van telschakelingen is het gebruik ervan in besturingsschakelingen. Hoewel het hier niet de plaats is om uitgebreid in te gaan op de organisatie van digitale schakelingen, moet deze toepassing toch reeds genoemd worden. Als regel bestaat een digitale schakeling uit een aantal deelschakelingen met een duidelijke organisatie (tellers, optellers, schuifregisters, enz.). De communicatie tussen deze deelschakelingen wordt verzorgd door een besturingsschakeling. Deze schakeling zorgt voor het op de juiste tijd vrijgeven van datapaden, genereert de enable-signalen, enz. Soms werkt een besturing autonoom, d.w.z. werkt een vast programma af op commando van de klokpuls. Meestal heeft een besturing ook instelingangen waarmee het ingestelde programma kan worden gewijzigd resp. onderbroken. Fig. 9.16 beschrijft in een tijddiagram hoe in een besturingsschakeling voor een zeker doel drie signalen X, Y en Z in de tijd verlopen ten opzichte van de klokpuls α van het systeem. Na vijf stappen herhaalt zich het gewenste besturingspatroon.

Het ontwerp van een dergelijke besturingsschakeling volgt in grote lijnen het ontwerp van tellers. Het tijddiagram wordt omgezet in een volgordetabel. Zie tabel 9.8.

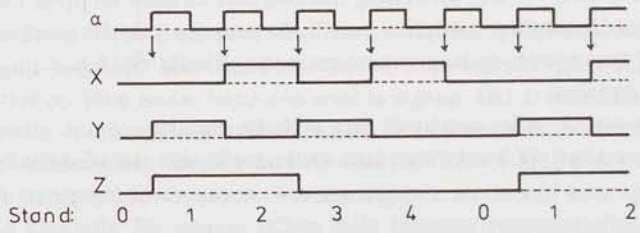


fig. 9.16. Specificatie besturingsschakeling.

$[X\,Y\,Z]^n$	$[X\,Y\,Z]^{n+1}$	J_x^n	K_x^n	J_y^n	K_y^n	J_z^n	K_z^n
0 0 0	0 1 1	0	–	1	–	1	–
0 0 1	– – –	–	–	–	–	–	–
0 1 0	1 0 0	1	–	–	1	0	–
0 1 1	1 0 1	1	–	–	1	–	0
1 0 0	0 0 0	–	1	0	–	0	–
1 0 1	0 1 0	–	1	1	–	–	1
1 1 0	– – –	–	–	–	–	–	–
1 1 1	– – –	–	–	–	–	–	–

tabel 9.8. Ingangsvoorwaarden besturingsteller.

Uit tabel 9.8 volgen zes Karnaughdiagrammen voor de J- en K-signalen. De meest eenvoudige formules zijn:

$$J_x = Y \qquad J_y = \overline{X} + Z \qquad J_z = \overline{X}\overline{Y}$$

$$K_x = 1 \qquad K_y = 1 \qquad K_z = X$$

De realisatie van de teller staat in fig. 9.17. Deze keer zijn positive edge-triggered J-K flip-flops gebruikt. Vergelijk het tijddiagram!

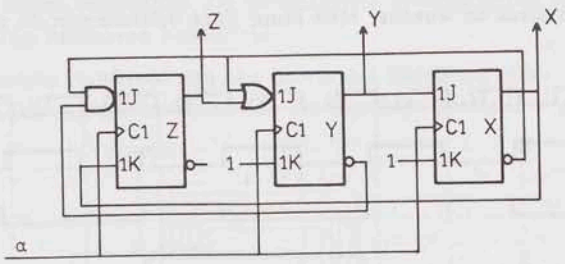


fig. 9.17. Besturingsteller.

Een andere benaderingswijze voor het ontwerpen van een besturingsteller is, dat men uitgaat van een verkrijgbaar circuit. Bijvoorbeeld een binaire teller of een BCD-teller. Met behulp van een combinatorische *code-omzetter* wordt de gewenste uitgangscode gerealiseerd. Dit type besturingstellers kan als nadeel bezitten dat niet alle uitgangen tegelijk veranderen. Ook uitgangsspikes kunnen optreden als gevolg van looptijdverschillen in de code-omzetter. Dit houdt o.a. in dat niet elke uitgang van zo'n schakeling met de klokingang van een flip-flop verbonden mag worden. De uitgangen van een besturingsteller volgens fig. 9.17 bezitten deze nadelen niet, deze signalen veranderen synchroon en zijn vrij van spikes. Bij het ontwerpen van besturingen komen we uitgebreid op deze problematiek terug.

Het inkorten van de cycluslengte

Bij vele processen moet een aantal pulsen afgeteld worden. De schakeling dient een detectiesignaal af te geven als het gewenste aantal pulsen bereikt is. Voor deze toepassingen is de telcode niet van belang. Het ligt daarom voor de hand zoveel mogelijk gebruik te maken van verkrijgbare circuits. De gewenste cycluslengte kan men via de preset- en clear-ingang van de geheugenelementen gemak-

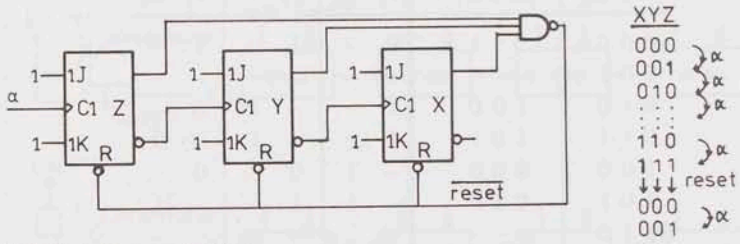


fig. 9.18. Gedeeld-door-7 teller.

kelijk instellen. Fig. 9.18 geeft een voorbeeld van een in de praktijk vaak toegepaste oplossing, zowel bij asynchrone als synchrone tellers. De schakeling bestaat uit een asynchrone tweedelerteller en een detectiepoort die aangeeft dat de stand 7 is bereikt. Het signaal uit deze AND-poort wordt gebruikt om de drie flip-flops terug op 0 te stellen via een Reset op de directe clear-ingang. Hiermee is de achtteller gedresseerd tot zeventeller. Deze oplossing kent enkele nadelen. Ten eerste het optreden van spikes aan de uitgangen van de flip-flops, welke inherent zijn aan deze wijze van schakelen. Zie fig. 9.19. Ten tweede bestaat de mogelijkheid dat de resetpuls te kort is voor een of meer van de geheugenelementen. Immers, zodra één van de geheugenelementen gereset is, valt de detectie weg! Deze nadelen veroorzaken dat deze oplossing van het probleem sterk ontraden dient te worden. Men komt deze oplossing in de praktijk helaas vaak tegen.

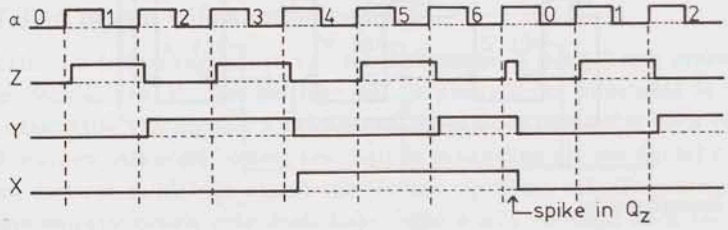


fig. 9.19. Tijddiagram van gedeeld-door-7 teller.

Aan een van de bezwaren tegen deze oplossing van het terugstelprobleem van tellers, de korte duur van de resetpuls, kan men tegemoet komen. Men gebruikt dan een extra trekker die geset wordt door het signaal dat detecteert dat de eindstand bereikt is. De trekker wordt weer gereset op de volgende fase van de klokpuls α. Het uitgangssignaal van de aldus bestuurde trekker stuurt de instelingangen van de telschakeling. Op deze wijze is een reset- of instelpuls gecreëerd ter breedte van de klokpuls.
Toch blijft deze oplossing riskant. Eventuele spikes uit de detectieschakeling voor de eindstand kunnen de trekker op een ongewenst tijdstip setten! Bovendien kan het resetten van een bepaalde flip-flop in een asynchrone teller leiden tot een klokpuls voor een volgende flip-flop.

Sommige als geïntegreerd circuit uitgevoerde tellers bezitten een synchrone loadingang. Afhankelijk van de waarde van het load-signaal telt de schakeling of neemt een beginstand over van inkomende datalijnen. Deze circuits maken het mogelijk volledig synchroon met de klokpuls een telcyclus te beëindigen en te springen naar een ingestelde waarde. Fig. 9.20 geeft een voorbeeld van een teller in de BCD-code die van de stand 23 (0010.0011) telt tot 78 (0111.1000)

en daarna geheel synchroon in de stand 23 terugkeert. Het toegepaste circuit is de SN 74160 van Texas Instruments. De uitgangen van de teller zijn geheel vrij van spikes.

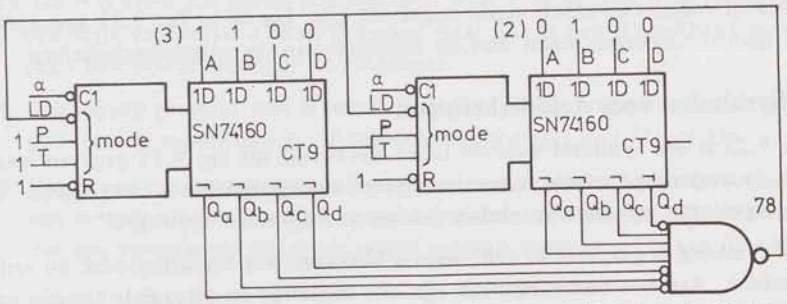


fig. 9.20. Instelbare teller, welke is geprogrammeerd van stand 23–78 in de BCD-code.

Behalve de hier behandelde schema's bestaan er nog vele mogelijkheden om tellers voor velerlei doeleinden te realiseren. Zo kunnen o.a. schuifregisters toegepast worden (zie volgend hoofdstuk), terwijl ook het gebruik van een accumulator (zie hoofdstuk over optellen) soms voordelen biedt. Een logisch ontwerper zal in zijn ontwerp bij voorkeur verkrijgbare circuits toepassen en zo weinig mogelijk speciale tellers ontwerpen. De voornaamste moeilijkheden die hierbij kunnen optreden zijn hierboven behandeld.

Tot slot een laatste voorbeeld van een universeel instelbare teller, bestaande uit een PROM (zie hoofdstuk 4) en een uit flip-flops bestaand register.

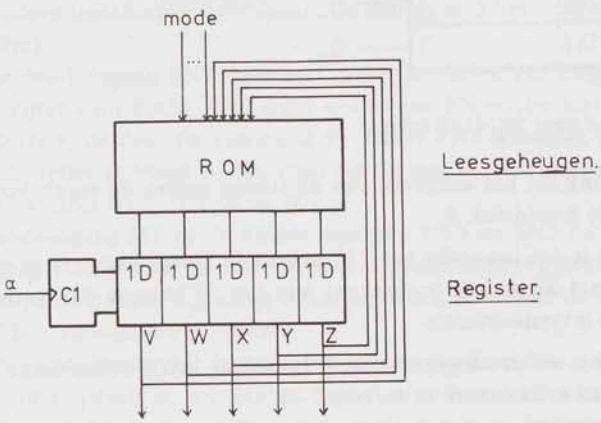


fig. 9.21. Programmeerbare teller.

De bij elke huidige tellerstand behorende volgende stand wordt op de bij deze stand behorende geheugenplaats in de PROM opgeslagen. Na de klokpuls wordt de nieuwe tellerstand als adres aangeboden. De PROM genereert dan de gewenste volgende stand, waarna de teller gereed is voor de volgende stap. Door de relatieve traagheid van PROM's is deze wijze van tellen niet snel. In de toepassing zijn deze tellers echter zeer flexibel.

Opmerking

Indien de PROM meer ingangen bezit dan het aantal geheugenelementen van de teller bestaat de mogelijkheid meer dan één cyclus te programmeren. De gewenste telcyclus kan via deze extra ingangen worden ingesteld. De toepassing van een

ROM is niet beperkt tot de realisatie van telschakelingen. Ook algemeen geformuleerde volgordeschakelingen kunnen met een register/ROM combinatie worden gerealiseerd. Een voorwaarde is wel dat het toestandsdiagram resp. de toestandstabel volgens de clock mode opgesteld zijn. Op de in fig. 9.21 aangegeven mode control ingangen staan dan de ingangen van de volgordeschakeling.

9.6. Symbolen voor telschakelingen

In fig. 9.22 is een symbool voor de teller SN74160 uit fig. 9.15 gegeven waarin ook de onderlinge relatie tussen de diverse ingangssignalen is aangegeven. Voor het interpreteren van deze symbolen gelden de volgende "spelregels".

1. Het symbool is gesplitst in een tweetal blokken, een *besturingsblok* en een *datablok*. Aan het besturingsblok zijn alle ingaande en uitgaande timing- en instelsignalen getekend, aan het datablok de datasignalen.

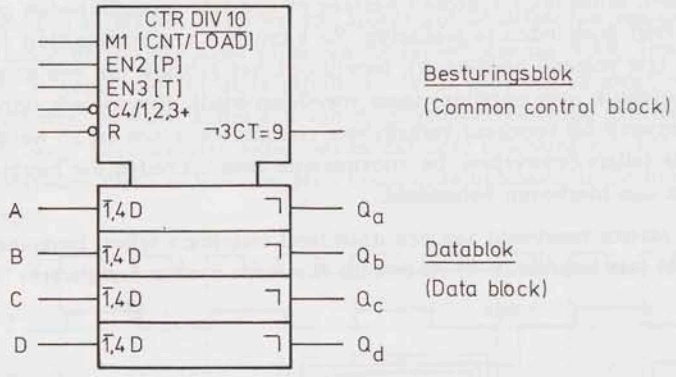


fig. 9.22. Symbool voor SN74160 teller.

2. Met betrekking tot het aangeven van de timing gelden de regels voor de flip-flops. Zie hoofdstuk 8.
3. Afgesproken is dat inwendig het "1"-niveau (= Hoog) het *actieve niveau* is. Correspondeert dit aan de buitenkant met het "0"-niveau dan geven we dit aan met een inversiecirkeltje.
4. Omhoog tellen wordt aangegeven met +, omlaag met −. Geschiedt het tellen in stappen van n dan staat er +n resp. −n.
5. CTn (Contents = n) betekent dat de betreffende uitgang inwendig 1 wordt als de teller in stand n staat.
6. De telcyclus wordt boven in het symbool aangegeven. Daarbij betekent DIVn een cyclus van n stappen. Een BCD-teller wordt dus aangegeven door CTRDIV10. Is n echter een macht van 2, dan wordt $DIV2^n$ vervangen door n. Een 4-bit binaire telsectie wordt aangegeven met CTR4.
7. Er zijn twee mogelijkheden om een ingang te blokkeren als aan bepaalde voorwaarden niet voldaan is, nl. via een zg. *gate input* G of een *mode input* M. Voor beide is het actieve niveau "1". Zij deblokkeren dan hetgeen erachter staat. Het verschil treedt op voor het "0"-niveau.

7a. Mn = 0 heeft als gevolg dat alle ingangen waar het cijfer n voor staat niet actief zijn, d.w.z. geen effect hebben.

7b. Gn = 0 heeft tot gevolg dat ingangen waar n voor staat 0 worden. Dit kan een actie veroorzaken! Een G-ingang staat dus in een EN-verband met de hierdoor beïnvloede ingangen en uitgangen.

8. Een ingang gemerkt met $\bar{n}$ wordt beïnvloed door de inverse van de corresponderende mode-ingang (of gate-ingang) gemerkt met Mn of Gn.

9. Hangt een ingang van verschillende mode-ingangen e.d. af, dan geven we dit aan met verschillende cijfers, gescheiden door een komma. Zo duidt m,$\bar{n}$ aan dat aan voorwaarde m (mode, gate) voldaan moet zijn EN aan de voorwaarde $\bar{n}$.

10. Een ingang kan verschillende functies hebben. We scheiden de aanduidingen hiervan door een schuine streep.

11. Bij elke ingang mogen we een willekeurige tekst opnemen, mits tussen rechte haken geplaatst. Bij voorbeeld vermeldt men de functie van zo'n ingang.

We zullen nu aan de hand van deze spelregels proberen het symbool van fig. 9.22 te interpreteren. Vergelijk hiermee het schema van de teller in fig. 9.15. Uit het symbool volgt met behulp van bovenstaande spelregels dat de gegeven teller uitwendig gezien de volgende eigenschappen heeft:

a. De clear- of resetingang staat niet onder besturing van enige andere ingang (geen cijfer voor R). De teller bezit een *directe reset* R welke, indien actief, alle andere instellingen overheerst. Uitwendig is 0 het actieve niveau (negatiecirkeltje).
b. De Enable-T ingang EN3 staat niet onder besturing van enige andere ingang (geen cijfer voor EN3). EN3 staat wel in een EN-relatie met uitgang CT=9 (*contents* 9, de detectie van stand 9). Heeft EN3 inwendig de waarde 1 en staat de teller in stand 9, dan staat op de met 3CT=9 gemerkte carry-uitgang een 1. Anders staat 3CT=9 op 0.
c. De mode-ingang M1 en de enable-ingangen EN2 en EN3 besturen tezamen de telingang 1,2,3+, m.a.w. de teller wordt opgehoogd (+) in stappen van één eenheid als de mode-instelling *count* aanwezig is en aan de door enable P/T gestelde voorwaarden is voldaan.
d. Als de mode-ingang M1 inwendig en uitwendig 0 is, d.w.z. als de mode-instelling *load* aanwezig is, worden de data A t/m D onder besturing van de *command input* C4 ingelezen. De teller wordt dan in de beginstand DCBA gezet.
e. De timing van zowel de instelsignalen als de datasignalen is negative pulse-triggered. Dit houdt in: Pulsgestuurd, data en instellingen stabiel bij de neergaande klokflank en vasthouden tot na de volgende opgaande klokflank. We halen dit gegeven uit:
   1. Bij de klokingang C4/1,2,3+ staat geen driehoekje, dus geen *dynamic input.*
   2. Bij de uitgangen van het datablok is de uitsteloperator aangegeven.
   3. Het cirkeltje voor de klokingang.
f. Uit de vermelding CTRDIV10 boven in het symbool volgt dat de teller cyclisch de standen 0 – 9 doorloopt.

Het symbool uit fig. 9.22 geeft de eigenschappen van de teller uit fig. 9.14 cor-

Voorbeeld

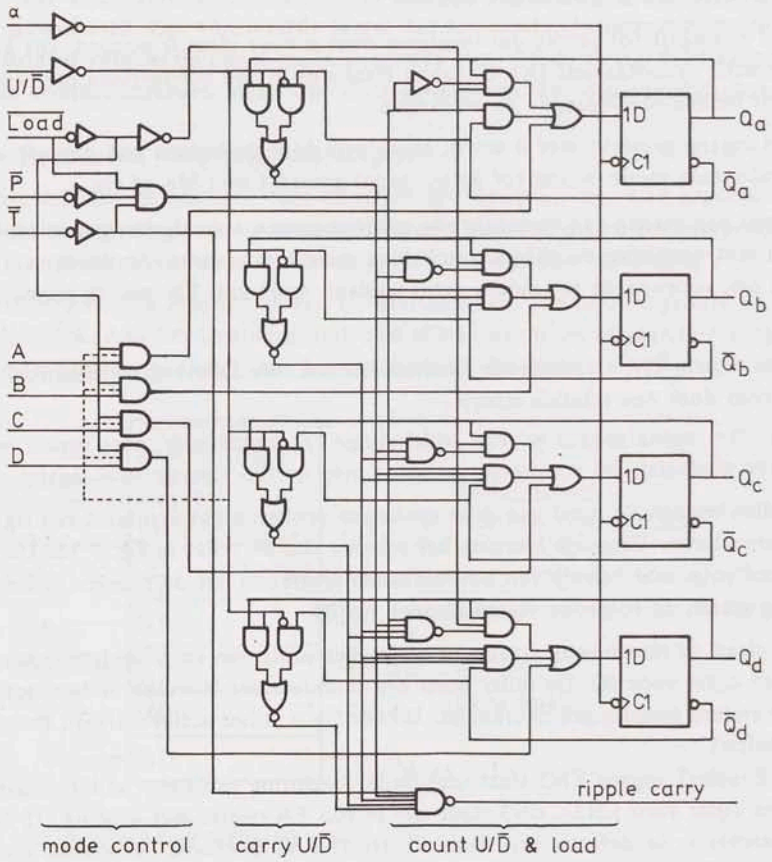


fig. 9.23. Omschakelbare Up/Down binaire teller.

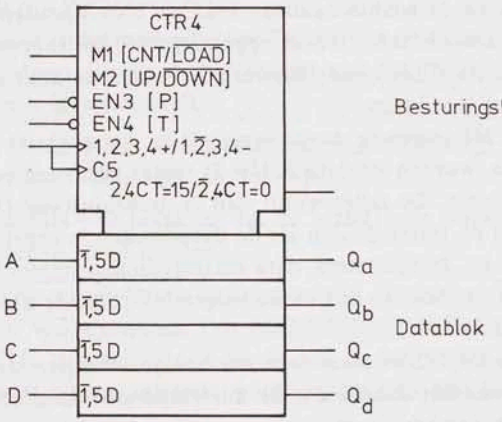


fig. 9.24. Symbool.

rect en overzichtelijk weer. De classificatie "negative pulse-triggered" van de timing betekent dat de schakeling correct werkt als de instelsignalen voor de neergaande klokflank klaar staan. Sommige van de signalen mogen onder om-

standigheden later worden ingesteld. Ook komt het voor dat een Hoog-Laag overgang nog wel toegestaan is maar een Laag-Hoog overgang niet. Dergelijke details, die de uitzonderingen op de algemene regel betreffen, worden niet in het symbool opgenomen. Bij het normale gebruik van geïntegreerde circuits zijn deze details onbelangrijk. Volledige informatie hierover staat in de catalogi of moet, bij het ontbreken hiervan, door metingen worden vastgesteld.

In fig. 9.23 is het schema van een binaire teller (vergelijk SN74LS169A en SN74LS669) gegeven. Deze teller telt cyclisch van 0 – 15 of omgekeerd. Deze teller is een "Up/Down Counter". De combinatoriek van deze teller is in drie niveaus opgebouwd:

- Het ingangsniveau voor de instelling van de "telmode" resp. het laden.
- Vier selectoren voor het omschakelen van "Up" naar "Down", zowel voor de telrichting als voor het transport.
- Poorten voor de sturing van de vier flip-flops.

Het symbool voor deze teller staat in fig. 9.24. De interpretatie van het symbool wordt aan de lezer als oefening aanbevolen.

Opmerking

Indien de fabrikant de in fig. 9.23 gestippelde verbindingen had aangebracht, dan was het mogelijk geweest de teller op synchrone wijze te resetten met de instelling:

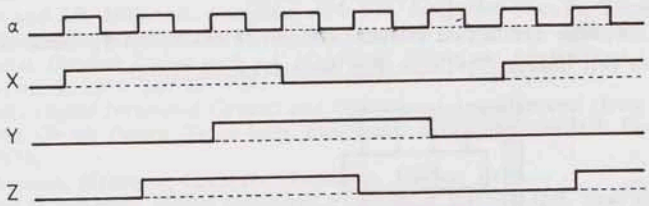
$$\overline{\text{Load}},U/\overline{D} = 0,1$$

Nu is slechts synchroon laden met DCBA mogelijk. Tijdens de instelling "Load" is de $U/\overline{D}$ ingang een don't care.
Ga na hoe het symbool voor het aangepaste circuit wordt.

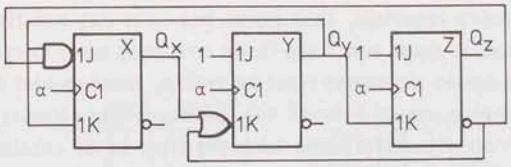
Opgaven

9.1. Bij sommige toepassingen van tellers moet synchroon gedetecteerd worden dat een bepaald aantal pulsen is ontvangen. Toon aan dat dit behalve met een geheel synchroon werkende teller ook kan worden uitgevoerd met een asynchrone down-counter. Kan dit ook met een asynchrone up-counter.

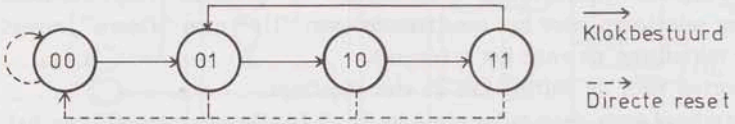
9.2. Ontwerp een synchrone telschakeling die cyclisch de onderstaande zes standen doorloopt. Gebruik J-K flip-flops (pos. edge-triggered).



9.3. Gegeven is een sequentiële schakeling die bestaat uit drie flip-flops X, Y en Z. Stel een toestandsdiagram op voor deze schakeling, waarin alle mogelijke toestanden en hun overgangen voorkomen.



9.4. Ontwerp een telschakeling die de in het diagram getekende cyclus doorloopt. Via een clear-commando gaat de schakeling altijd naar de stand YZ = 00. Vervolgens doorloopt de schakeling bij ontvangst van telpulsen de cyclus YZ = 00, 01, 10, 11, 01, 10, 11, 01, . . .
Kies zelf welk type flip-flop het gunstigste resultaat geeft.



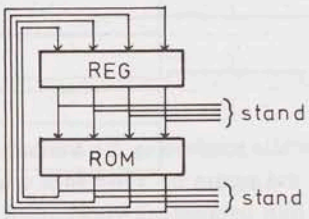
9.5. Ontwerp een binaire telschakeling tot 2^n. De teller moet worden voorzien van een mode-signaal, waarmee ingesteld wordt of de schakeling telt resp. zijn laatste stand onthoudt. In de onthoudstand worden wel pulsen aangeboden. De keuze van het type te gebruiken flip-flop is geheel vrij. Ook mag de werking geheel of gedeeltelijk asynchroon zijn.

9.6. De schakeling in fig. 9.15 heeft twee enable-ingangen P en T. Zijn deze ingangen verwisselbaar, of hebben zij een gedeeltelijk verschillende functie?

9.7. In fig. 9.14 is het transportsysteem voor een snelle BCD-teller gegeven. In deze opzet zijn de flip-flops van het type negative edge-triggered. Ontwerp het transportsysteem van een snelle BCD-teller waarvan de flip-flops van het type positive edge-triggered zijn.

9.8. In fig. 9.9 zijn de schakeling en het bijbehorende tijddiagram gegeven die nodig zijn om een asynchrone tweedelerteller zonder sprongen in de telcyclus te kunnen omschakelen van up naar down en omgekeerd.
a. Specificeer de gewenste werking van het koppelnetwerk in een toestandstabel. Gebruik hiervoor het Moore type.
b. Reduceer deze tabel.
c. Onderzoek of gebruik van het Mealy model leidt tot een tabel met minder toestanden in de gereduceerde versie.

9.9. Het is mogelijk synchrone telschakelingen te realiseren op basis van een ROM en een blokje flip-flops. Zie onderstaande figuur.



De huidige stand van de flip-flops dient als adres voor de ROM. Op het betreffende adres is de gewenste nieuwe stand van de telschakeling opgeslagen.

a. Het is op de aangegeven twee plaatsen mogelijk de tellerstand uit te lezen. Welke verdient de voorkeur?
b. Wat denkt u van de snelheid van een dergelijke teller t.o.v. een teller die bestaat uit losse componenten?

9.10. Gegeven is een omschakelbare tweedeler:

m = 0: Tweedeler op de voorflank van de klokpuls.
m = 1: Tweedeler op de achterflank van de klokpuls.

Het mode-signaal m wordt niet geschakeld tijdens de actieve momenten van de klokpuls. De uitgang Q van de flip-flop verandert niet als m wordt omgeschakeld.
Toon aan dat met deze tweedeler als bouwsteen een omschakelbare tweedelerteller (asynchrone opzet) kan worden ontworpen, die geen sprong maakt als van up naar down wordt omgeschakeld en omgekeerd.

9.11. Beschikbaar zijn J-K flip-flops van het type positive edge-triggered. Behalve de Q-uitgang van deze flip-flops is ook de $\overline{Q}$-uitgang beschikbaar. Met deze flip-flops moet een teller worden ontworpen die in de gewone binaire code cyclisch de standen 000 t/m 101 doorloopt. Een niet geheel synchrone werking is toegestaan.
a. Teken een tijddiagram met hierin tenminste twee cycli (zonder tijdsvertragingen).
b. Ga voor elke flip-flop na welke signalen eventueel als klokpuls kunnen dienen. Een signaal is hiertoe geschikt als het tenminste op alle momenten dat de flip-flop van stand moet veranderen een opgaande flank bezit.
c. Bepaal voor elk alternatief de formules voor de J en K ingangssignalen.
d. Kies de meest eenvoudige oplossing.
e. Welke risico's zijn aan deze wijze van ontwerpen van tellers verbonden?

Literatuur

1. W.N. Carr and J.P. Mize, *MOS/LSI Design and Application,* McGraw-Hill, New York 1972, pp. 229–258.
2. F.B. Manning and R.R. Fenichel, *Synchronous Counters Constructed Entirely of J-K Flip-Flops,* IEEE Tr. on Computers, Vol. C–25, March 1976, pp. 300–306.
3. R.L. Morris and J.R. Miller ed., *Designing with TTL Integrated Circuits,* Texas Instr. Electronics Series, McGraw-Hill, New York 1971, pp. 243–284.
4. P.A. Neeteson, *Gateless Scalers with J-K Flip-Flops,* Electronic Applications, Vol. 28, no. 3, 1968, pp. 99–109.
5. B. Norris ed., *Digital Integrated Circuits and Operational-Amplifier and Opto-Electronic Circuit Design,* Texas Instr. Electronics Series, McGraw-Hill, New York 1976.
6. R.M.M. Oberman, *Electronic Counters,* MacMillan, London 1973.
7. H. Taub and D. Schilling, *Digital Integrated Electronics,* McGraw-Hill, New York 1977, pp. 322–355.
8. *The TTL Data Book for Design Engineers,* Texas Instruments, 1977.

De volgende literatuur wordt aanbevolen voor de studie van de realisatie van volgordeschakelingen met register/ROM combinaties:

9. H.A. Sholl, *Direct Transition Memory and its Application in Computer Design,* IEEE Tr. on Computers, Vol. C–23, Oct. 1974, pp. 1048–1061.
10. H.A. Sholl and S.C. Yang, *Design of Asynchronous Sequential Networks Using Read-Only Memories,* IEEE Tr. on Computers, Vol. C–24, Febr. 1975, pp. 195–206.

10. SCHUIFREGISTERS

10.1. Enkele toepassingen van schuifregisters

De bewerking "schuiven" komt in vele digitale apparaten als onderdeel van het uit te voeren proces voor. De schuifoperatie is een bewerking waarbij op commando van de klokpuls a de inhoud van een register één of meer plaatsen naar links resp. naar rechts schuift. Een schuifregister bestaat uit een rij flip-flops die zodanig gekoppeld zijn dat geldt (sectie no. 1 rechts):

$$Q_i^{n+1} = Q_{i+1}^n \text{ voor een rechtsschuivend } (\rightarrow) \text{ schuifregister}$$
$$Q_i^{n+1} = Q_{i-1}^n \text{ voor een linksschuivend } (\leftarrow) \text{ schuifregister.}$$

Schuifregisters zijn in velerlei uitvoeringen verkrijgbaar. Voor sommige toepassingen is een eenvoudige uitvoering, waarbij alleen kan worden geschoven en tevens parallel laden mogelijk is, voldoende. Andere toepassingen vragen meer faciliteiten. Fig. 10.1 geeft het schema van een universeel toepasbaar schuifregister met de volgende faciliteiten:

Rechts schuiven ($\rightarrow$).

Links schuiven ($\leftarrow$).

Parallel laden.

Onthouden (do nothing, hold).

De eerste drie bewerkingen worden synchroon op de klokpuls uitgevoerd. De aanwezige reset is direct en overheerst wanneer hij gegeven wordt de andere instellingen. Zoals uit het schema blijkt kan men de gewenste functie instellen met twee insteldraden s_0 en s_1. Ook de clear-ingang kan men als een instelingang beschouwen. Deze functie overheerst echter de instelling via s_0 en s_1. Tabel 10.1 geeft een overzicht hoe de gewenste functie ingesteld kan worden.

clear	s_1	s_0	Ingestelde functie
0	–	–	Reset, asynchroon
1	0	0	Onthouden (do nothing, inhibit, hold)
1	0	1	Rechts schuiven ($\rightarrow$)
1	1	0	Links schuiven ($\leftarrow$)
1	1	1	Parallel laden.

tabel 10.1. Functietabel van een schuifregister.

Opmerking

In het algemeen mag men instelsignalen slechts op bepaalde tijdstippen omschakelen. Bij sommige circuits mag dit alleen als de klokpuls hoog resp. laag is. Bepalend is hierbij de wijze waarop de flip-flops van het register reageren op de klokpuls (edge-triggered, enz.), alsmede het feit of er wel of niet in de klokpuls geschakeld wordt. (Vergelijk: Texas Instruments type SN74194 en SN74S194.) Nemen we aan dat de flip-flops in fig. 10.1 edge-triggered zijn op de opgaande flank van de klokpuls a, dan is voor deze uitvoering de enige eis dat niet geschakeld wordt tijdens deze opgaande flank.

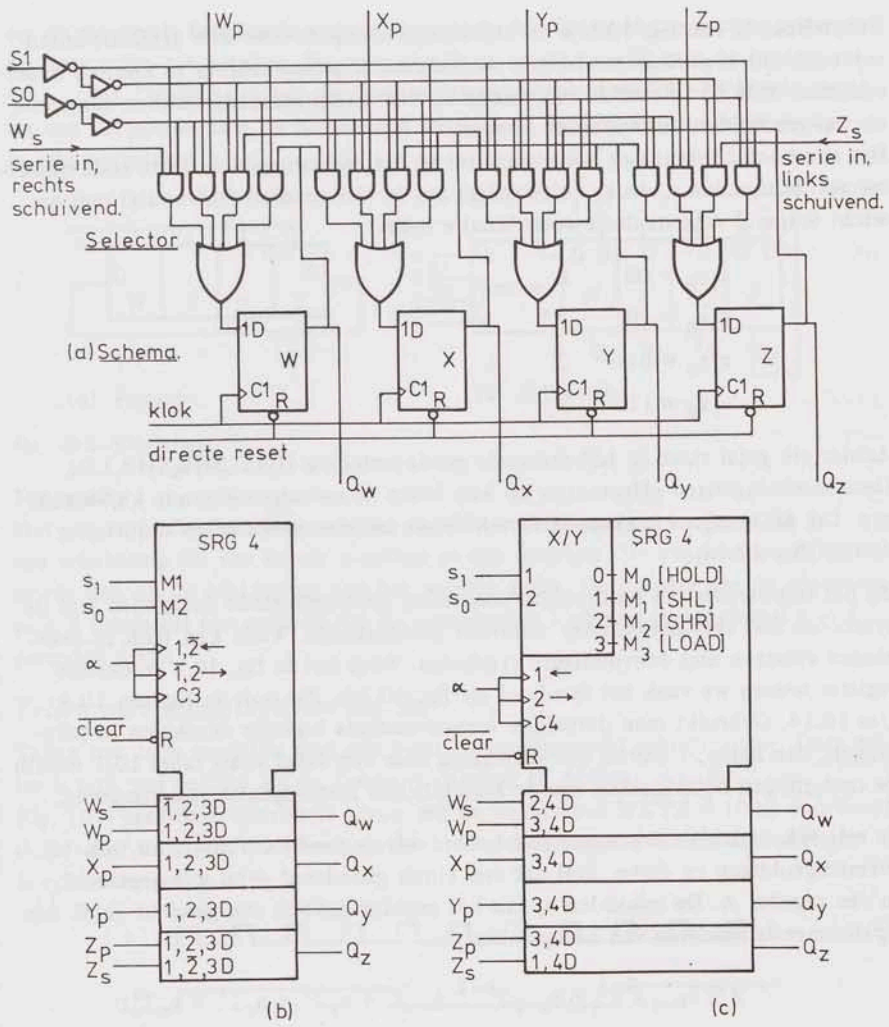


fig. 10.1. Instelbaar schuifregister.

Symbolen

In fig. 10.1 zijn behalve het logisch schema twee symbolen getekend. Voor de interpretatie van deze symbolen gelden behalve de bij flip-flops en tellers gemaakte afspraken ook de volgende afspraken:

1. Het teken → geeft de schuifrichting van-links-naar-rechts aan of de richting van-boven-naar-beneden.
2. Het teken ← geeft de schuifrichting van-rechts-naar-links aan of de richting van-beneden-naar-boven.

Voorbeeld

1,2 ← betekent dat als M1,M2 = 1,0 (hetgeen uitwendig $s_1 s_0 = 10$ is), het schuifregister naar links schuift op zijn interne klokpuls en tegelijkertijd onder besturing van klokpuls C3 het signaal Z_s via de met 1,2,3D gemerkte serie-ingang inleest.

In symbool b. van fig. 10.1 is de uitwendige klokpuls over drie aansluitpunten verdeeld. Dit is gedaan om de drie werkingsmodi gemakkelijker te kunnen onderscheiden. Men had dezelfde informatie bij één klokpulsingang kunnen aangeven, en wel gescheiden door schuine strepen.
Het symbool c. bevat een code-omzetter in het besturingsblok. Deze code-omzetter zet de signalen s_1 en s_0 (uitwendig) om in vier getallen (inwendig) met gewicht 0 t/m 3 volgens de gewone binaire code:

$s_1 s_0 = 00 \quad 0$
$s_1 s_0 = 10 \quad 1$
$s_1 s_0 = 01 \quad 2$
$s_1 s_0 = 11 \quad 3$

Achter elk getal staat de bijbehorende mode-instelling (HLD,SHL,SHR,LD). Deze mode-ingangen effectueren op hun beurt de corresponderende klokingangen. Uit dit symbool blijken de verschillende instelmogelijkheden duidelijker dan uit fig. 10.1.b.

Bij het ontwerpen van besturingen voor meer gecompliceerde datapaden zijn de symbolen met een *dependency notation* noodzakelijk. Vaak kan men in tekeningen volstaan met eenvoudigere symbolen. Voor het in fig. 10.1 getekende register nemen we vaak het symbool uit fig. 10.2.b. Zie ook de figuren 10.4 t/m 10.14. Gebruikt men dergelijke vereenvoudigde logische symbolen in tekeningen, dan behoort hierbij een verwijzing naar een tabel zoals tabel 10.1 waarin de onderlingen beinvloeding van de verschillende instelsignalen is aangegeven.

In vele *rekenkundige processen* komt schuiven als deelbewerking voor, o.a. bij vermenigvuldigen en delen. Stel dat een binair gecodeerd getal g is geplaatst in een register A. De inhouden a_i van het register hebben een gewicht gelijk aan opklimmende machten van het grondtal 2:

$$g = a_{k-1}2^{k-1} + a_{k-2}2^{k-2} + \ldots + a_2 2^2 + a_1 2^1 + a_0 2^0.$$

Vermenigvuldigen van g met een factor 2 geeft

$$2g = a_{k-1}2^k + a_{k-2}2^{k-1} + \ldots + a_2 2^3 + a_1 2^2 + a_0 2^1.$$

Deze vermenigvuldiging kan worden uitgevoerd door de inhoud van het register A één plaats naar links te schuiven en rechts een 0 in te schuiven. Ook delen door 2 kan op deze wijze worden uitgevoerd door de inhoud één plaats naar rechts te schuiven. Het laagstwaardige bit valt daarbij uit het register, hetgeen rekenkundig een afronding naar beneden inhoudt.
Hiermee is de betekenis van schuiven voor rekenkundige bewerkingen aangetoond. Er zijn nog vele andere toepassingen:

Tellen met schuifregisters tot n

Met schuifregisters kan men eenvoudige tellers ontwerpen. De meest bekende is de *ringteller (ring counter)*, die bestaat uit een *rondgekoppeld schuifregister.* Zie fig. 10.2. Deze schakeling bezit geen extra poorten, zodat men kan tellen

op de maximale frequentie van de toegepaste flip-flops. Met n secties kan men op deze wijze tot n tellen. De cycluslengte n wordt bereikt met de beginstand 10 . . . 00. Degeneratie van de cycluslengte is echter mogelijk. Zo is een register van zes secties met de beginstand 100100 na drie klokpulsen weer terug in zijn beginstand.

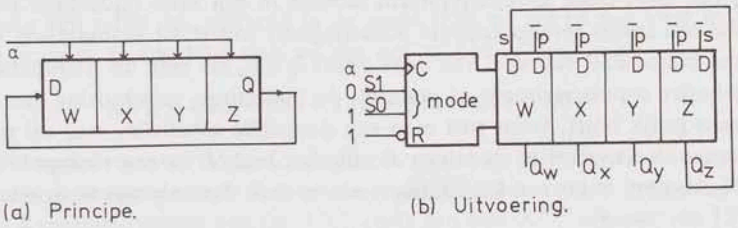


fig. 10.2. Ringteller.

Toepassing

Met een ringteller kan men een zg. *divide-by-n counter* ontwerpen. Dit is een schakeling die van iedere n pulsen er één doorlaat. De aangeboden rij pulsen wordt dan op de klokingang van het register gezet. De breedte van de afgegeven puls is minimaal één periode van de aangeboden klokpuls en maximaal n − 1 perioden ervan.

Tellen met schuifregisters tot 2n

Tellen tot 2n is mogelijk met een invers teruggekoppeld schuifregister. Deze teller is ook wel bekend als de *Johnson counter* of *Möbius counter*.
Fig. 10.4 geeft een voorbeeld ervan. Bij de beginstand WXYZ = 1000 doorloopt deze teller de in fig. 10.3 aangegeven cyclus. Degeneratie van de cycluslengte is echter mogelijk.

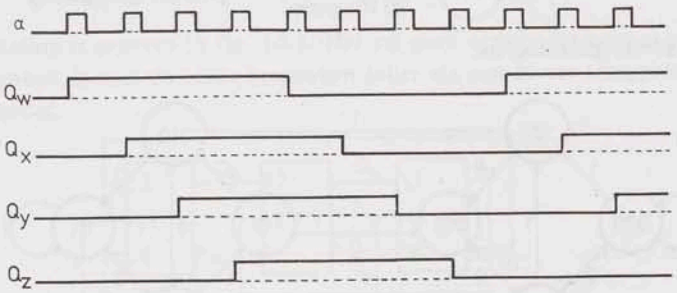


fig. 10.3. Meerfasen klokpulssysteem.

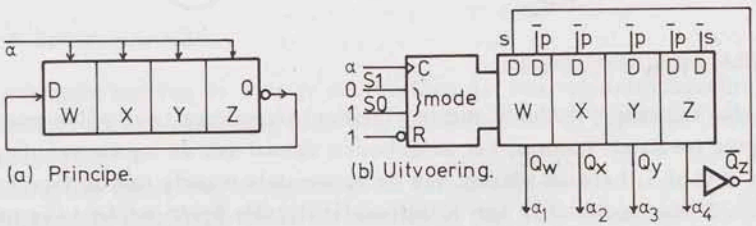


fig. 10.4. Twisted ring counter als meerfasen klokpulsgenerator.

Een belangrijke toepassing van deze zg. *twisted ring counter* is het gebruik ervan als meerfasige klokpulsgenerator. De tot nu toe behandelde clock mode schakelingen kennen één klokpuls die alle schakelhandelingen synchroniseert. In grotere digitale schakelingen past men vaak een meer-fasen klokpulssysteem toe. Bij het ontwerpen van besturingen komen we hier op terug. De diverse fasen in een meer-fasen klokpulssysteem moeten in een vaste onderlinge relatie staan. Via een invers teruggekoppeld schuifregister wordt dit automatisch bereikt. Een bijkomend voordeel van deze opzet is dat, als men de frequentie van de aangeboden α-puls verhoogt of verlaagt, de onderlinge verschuiving van de (vier) fasen gelijk blijft. Soms rust men een dergelijke schakeling nog uit met een detectie die aangeeft of door een storing het register in een verkeerde stand is gekomen. Voor een klokpulssysteem is deze detectie aan te bevelen!

Tellen met schuifregisters in een willekeurige cyclus

In schuifregisters kunnen cycli worden opgewekt met lengten tot 2^n. De terugkoppeling is dan in het algemeen een functie van verschillende uitgangssignalen. Fig. 10.5 en 10.6 geven twee voorbeelden van *schuifregisterdiagrammen*, die door een rechtsschuivend schuifregister van lengte twee resp. drie kunnen worden doorlopen bij een geschikte keuze van de terugkoppelfunctie.

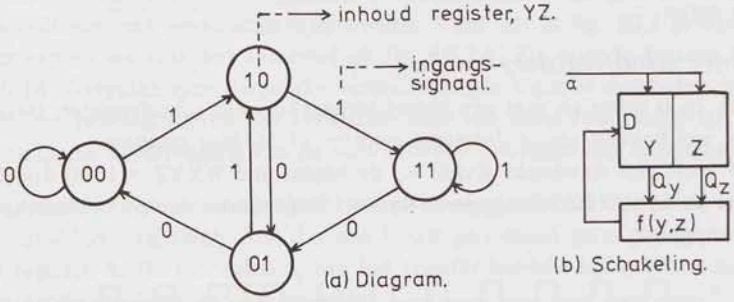


fig. 10.5. Schuifregisterdiagram.

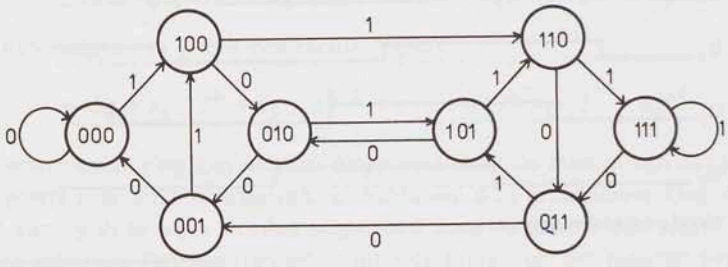


fig. 10.6. Schuifregisterdiagram.

Van elke toestand (= inhoud van het register) vertrekken twee pijlen naar een volgende toestand. Immers, het aangeboden signaal aan de ingang van de eerste sectie is 0 of 1, hetgeen afhangt van de momentele waarde van de terugkoppelfunctie. Iedere toestand in een schuifregisterdiagram heeft precies twee mogelijke opvolgertoestanden. Evenzo kent iedere toestand precies twee voorgangers. Diagrammen voor schuifregisters van lengte groter dan vier secties worden wel-

dra onoverzichtelijk. Men kan met behulp van deze diagrammen eenvoudige telschakelingen ontwerpen voor het aftellen van een bepaalde cyclus. Alleen de ingangsfunctie van de eerste sectie moet worden bepaald. De overige secties schuiven hun inhoud op de klokpuls door.

Voorbeeld
Ontwerp een teller die, startend in de stand 000, na zes pulsen weer in deze stand terug is. In fig. 10.6 zien we dat een mogelijke cyclus is:

$$XYZ = 000 \rightarrow 100 \rightarrow 110 \rightarrow 111 \rightarrow 011 \rightarrow 001.$$

De geheugenelementen Y en Z zijn flip-flops die als schuifsectie zijn geschakeld. Voor het geheugenelement X moet de ingangsfunctie worden bepaald. Het eerste Karnaughdiagram van fig. 10.7 geeft aan hoe X^{n+1} afhangt van $[XYZ]^n$, de huidige inhoud van het register. Uit dit diagram worden de diagrammen voor J_x en K_x op de gebruikelijke wijze bepaald.

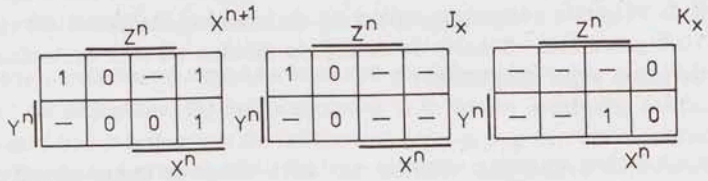


fig. 10.7. Het ontwerp van een schuifregisterteller.

We zien dat

$$J_x = \overline{Z} \quad \text{en} \quad K_x = Z$$
$$J_y = X \quad \text{en} \quad K_y = \overline{X}$$
$$J_z = Y \quad \text{en} \quad K_z = \overline{Y}.$$

De schakeling is gegeven in fig. 10.8. Het zal geen verwondering wekken dat deze teller identiek is met de reeds besproken teller via een invers teruggekoppeld schuifregister.

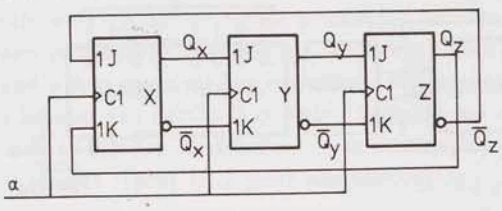


fig. 10.8. Schuifregisterteller.

Dit is overigens een van de weinige voorbeelden dat een volgordeschakeling met J-K flip-flops niet zo eenvoudig is als een schakeling met D flip-flops (enkeldraadskoppeling!).
Een belangrijke klasse van teruggekoppelde schuifregisters zijn de *modulo-twee teruggekoppelde schuifregisters.* Deze worden in par. 10.4 behandeld. We onderzoeken nu eerst de toepassing van schuifregisters in parallel-serie en serie-paral-

lel omzetters. Bij deze omzetters treedt een besturingsprobleem op: op gezette tijden moeten de instelingangen van het schuifregister worden omgeschakeld.

10.2. Parallel-serie en serie-parallel omzetters

Bij het koppelen van digitale apparaten komt het vaak voor dat de databreedte van de apparatuur niet overeenstemt met de databreedte van het kanaal waarover het datatransport moet plaatsvinden. Soms hebben ook de diverse deelsystemen een verschillende databreedte. Vergelijk: een 16-bit computer en een 8-sporen ponsbandlezer. Men staat dus vaak voor de noodzaak de breedte van de aangeboden data aan te passen. Sommige kanalen staan slechts bit-voor-bit transport toe, o.a. een telexverbinding.
Schuifregisters met *parallel load* faciliteiten bieden een eenvoudige oplossing voor het probleem hoe parallel aangeboden data in serie te verzenden. Met een schuifregister van k bits lang kunnen woorden tot een breedte van k bits in serie worden uitgezonden. Aan de ontvangzijde van het kanaal vindt het omgekeerde plaats. Bij de volgende ontwerpen zullen we als basis het instelbare schuifregister van fig. 10.1 gebruiken. Behalve de datalijnen moeten we ook de besturingssignalen op de juiste wijze aansluiten.

Voorbeeld
Ontwerp een schakeling waarmee woorden van acht bits in serie kunnen worden uitgezonden. Na het verzenden van acht bits moet de schakeling automatisch een nieuw woord laden. Dit woord staat altijd klaar, er behoeft niet getest te worden of dit zo is.
De te ontwerpen schakeling bestaat uit twee schuifsecties van elk vier bits waarin het 8-bit datawoord kan worden geplaatst. Er is bovendien een *besturingsschakeling* nodig die ervoor zorgt dat op iedere achtste klokpuls een nieuw woord van acht bits wordt geladen. Daartoe moeten dan de beide instelingangen s_0 en s_1 op 1 gezet worden. Gedurende de overige klokpulsen schuift het register naar rechts ($s_1 s_0 = 01$). We komen in eerste instantie tot een opzet zoals die is geschetst in fig. 10.9.

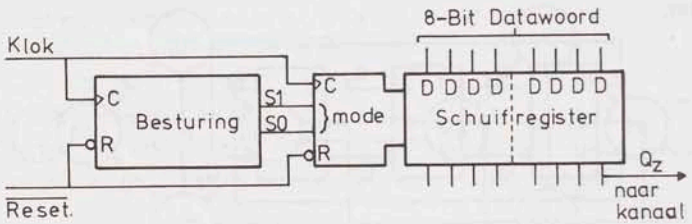


fig. 10.9. Opzet van een parallel-serie omzetter.

De besturingsschakeling, zoals deze in fig. 10.9 globaal is aangeduid, kan men realiseren op basis van een teller die cyclisch acht standen doorloopt. In de nulstand [000] worden de twee registers ingesteld op parallel load. Na de eerste klokpuls is de data geladen en moet worden omgeschakeld naar schuiven. Hiervoor dient de getekende NOR-poort. Zie fig. 10.10. Na acht klokpulsen, of eventueel na een reset, keert de teller in de nulstand terug en wordt het volgende datawoord op de eerstvolgende klokpuls binnengehaald. In fig. 10.10 is het

resultaat gegeven. In dit schema is ook de teller in een besturingsblok en een datablok gesplitst.

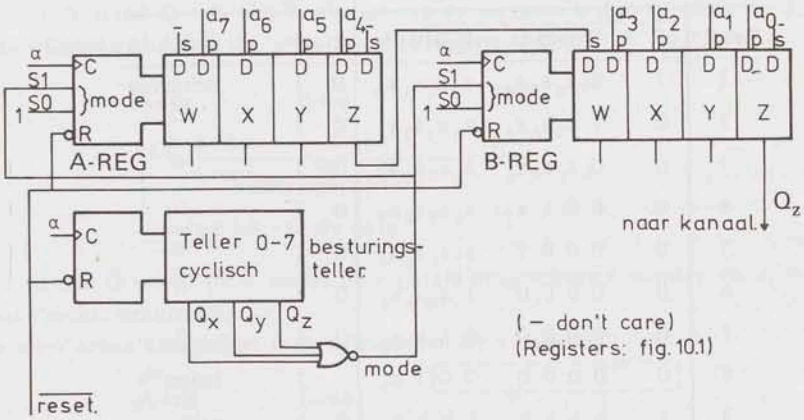


fig. 10.10. Parallel-serie omzetter met besturingsteller.

Deze parallel-serie omzetter kan ook gerealiseerd worden met weinig meer dan het schuifregister. De besturingsteller is in wezen overbodig omdat gebruik gemaakt kan worden van de tellende eigenschappen van het schuifregister.

- Op de eerste klokpuls wordt het parallel aangeboden datawoord geladen. Aan de uitgang Q_z van het B-register staat het eerste bit van de data voor transport gereed.
- Na de volgende klokpuls staat het volgende bit gereed voor transport. De sectie W van het A-register is dan een don't care geworden. Immers, de inhoud ervan is in sectie X van het A-register geschoven. Sectie W van het A-register mag nu eventueel voor besturingsdoeleinden benut worden.
- Na de k-de klokpuls zijn de eerste $k-1$ secties leeg, zodat $k-1$ secties voor besturingsdoeleinden beschikbaar zijn.

Men kan nu van links af het register gedurende opeenvolgende klokpulsen met nullen (of enen) vullen. Een blok van $k-1$ aaneengesloten nullen betekent dan dat er reeds even zoveel bits zijn getransporteerd. Een voorwaarde opdat deze detectie werkt is dat de telbits gescheiden worden van de informatiebits. Deze kunnen immers ook alle nul zijn. Hiervoor is een extra flip-flop nodig die in de 1-stand wordt gezet als het register wordt geladen met een nieuw datawoord. De inhoud van de flip-flop (was 1) schuift het register in. Gedurende de klokpulsen na het laden wordt een 0 in de flip-flop gelezen, welke weer in het register schuift. Tabel 10.2 geeft een toelichting.

Opmerking

De besturingsbits in tabel 10.2 mogen enen zijn. Bij reset moet dan de besturingsflip-flop in de stand 1 gezet worden; en de NOR wordt een AND.

Na een reset van het systeem moet het register worden geladen met de nieuwe data. Daartoe moet $s_1 s_0 = 11$ zijn. Zowel in stand 0 (na reset) als in stand 8 (opnieuw laden) kan dit gedetecteerd worden met het in tabel 10.2 getekende venster. In alle overige standen moet het register schuiven, d.w.z. $s_1 s_0 = 01$. We kunnen de besturingsingang s_0 dus 1 houden, terwijl s_1 gelijk kan zijn aan het signaal dat aangeeft dat er allemaal nullen in het venster staan. Deze detec-

Stap	Resultaat FF	Reg. A	Reg. B	In te stellen functie s_1	s_0	
Reset	0	0 0 0 0	0 0 0	1	1	laden (na reset)
1	1	$a_7a_6a_5a_4$	$a_3a_2a_1a_0$	0	1	schuiven
2	0	1 $a_7a_6a_5$	$a_4a_3a_2a_1$	0	1	”
3	0	0 1 a_7a_6	$a_5a_4a_3a_2$	0	1	”
4	0	0 0 1 a_7	$a_6a_5a_4a_3$	0	1	”
5	0	0 0 0 1	$a_7a_6a_5a_4$	0	1	”
6	0	0 0 0 0	1 $a_7a_6a_5$	0	1	”
7	0	0 0 0 0	0 1 a_7a_6	0	1	”
8	0	0 0 0 0	0 0 1 a_7	1	1	laden
9	1	$b_7b_6b_5b_4$	$b_3b_2b_1b_0$	0	1	schuiven
10	0	1 $b_7b_6b_5$	$b_4b_3b_2b_1$	0	1	”
11	0	0 1 b_7b_6	$b_5b_4b_3b_2$	0	1	”

tabel 10.2. Datatransport in een parallel-serie omzetter.

tie kan in principe gebeuren met een NOR-poort met zeven ingangen. Fig. 10.11 geeft de totale schakeling.

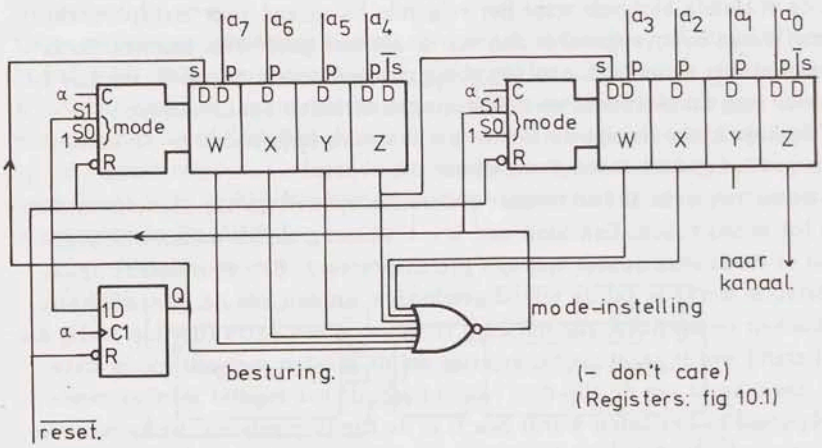
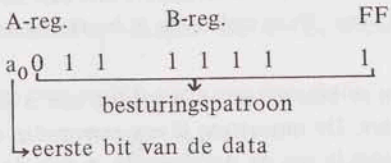


fig. 10.11. Parallel-serie omzetter via tellend schuifregister.

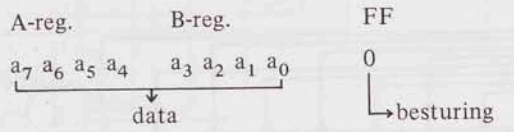
Vergelijken we de besparing in componenten van de schakeling in fig. 10.11 met die uit fig. 10.10, dan is deze niet spectaculair te noemen. Het principe van het *tellend schuifregister* kan echter vaak worden toegepast bij omzettingsproblemen.

Het omgekeerde, de omzetting van acht in serie ontvangen bits naar een parallel aangeboden 8-bit woord verloopt vrijwel op dezelfde wijze. Fig. 10.12 beschrijft een oplossing waarbij weer gebruik wordt gemaakt van het tellend schuifregister. De schakeling spreekt voor zich. Via een tabel van de vorm als tabel 10.2 kan men de opzet van de besturing gemakkelijk nagaan.

Voor de besturing van de serie-parallel omzetter van fig. 10.12 is slechts één D flip-flop en één poort nodig. Na een reset van de schakeling staat de flip-flop in de stand Q = 0. Het s_1-signaal van de registers is dan 1, zodat op de eerstvolgende klokpuls de inhoud van de registers wordt:



Het signaal $\overline{Q}$ wordt nu 0, zodat de registers omgeschakeld worden via $s_1 = 0$ naar "rechts schuiven".
Na weer zeven klokpulsen is de inhoud van de schakeling:



De eerste acht bits zijn nu parallel beschikbaar. Bij de volgende klokpuls start een nieuwe cyclus.

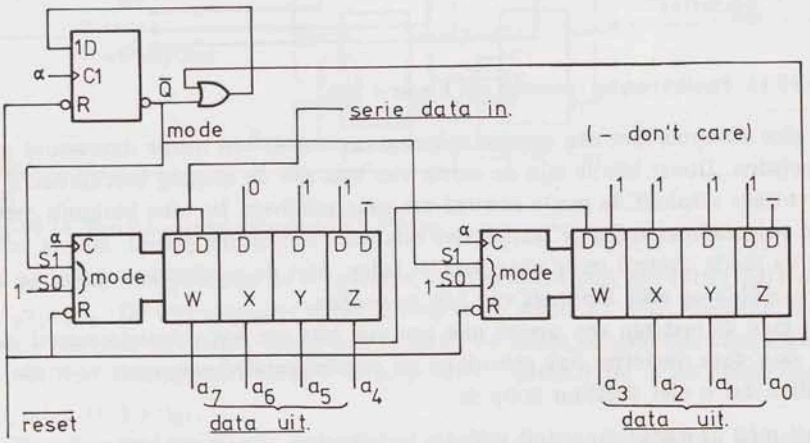


fig. 10.12. Serie-parallel omzetter.

10.3. Parallel-parallel omzetters

Een besturingstechnisch gezien zeer interessant probleem is het omzetten van de databreedte van n bits naar m bits. Het geval dat n of m gelijk aan 1 is, werd reeds in de vorige sectie besproken.
Stel dat de breedte van de aangeboden data m bits is en die van het kanaal n bits. Is $m < n$, dan bestaan er weinig problemen als men bereid is een gedeelte van de kanaalcapaciteit ongebruikt te laten. Het komt in de praktijk vaak voor dat men voor deze oplossing kiest. Is $m = n$, dan is in het geheel geen omzetter nodig. In het geval dat $m > n$, kan men kiezen uit de volgende mogelijkheden:

- $m = qn + r$ met $0 \leqslant r < n$.
  Zend q woorden van n bits en vervolgens de rest r. Is $r = 0$, dan is er geen verlies aan kanaalcapaciteit.
- $m = qn + r$ met $0 \leqslant r < n$.
  Zend q woorden uit van n bits. Vul de resterende r bits aan met $(n - r)$ bits van het volgende woord, enz. Deze oplossing is besturingstechnisch gezien niet eenvoudig.

Een schakeling die woorden van m bits omzet in woorden van n bits noemt men een *parallel-parallel* omzetter. De omzetting is erg eenvoudig als de databreedte n van het kanaal een deler is van de databreedte m van de aangeboden data. Fig. 10.13 geeft het principeschema van een omzetter die een woord van acht bits omzet in twee woorden van vier bits. De besturing van het register, welke niet is aangegeven, kan men gemakkelijk realiseren op basis van een tellend schuifregister.

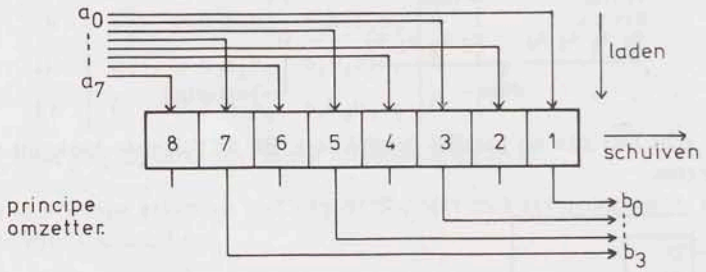


fig. 10.13. Parallel-parallel omzetter van 8 naar 4 bits.

Op elke klokpuls met een oneven volgnummer wordt een nieuw datawoord parallel geladen. Direct hierna zijn de eerste vier bits aan de uitgang beschikbaar. Vervolgens schakelt de mode control om naar schuiven. Na elke klokpuls met even volgnummer staan de laatste vier bits voor verzending gereed. Hierna schakelt de mode control terug naar parallel laden. Met de geschetste schakeling is het mogelijk op elke klokpuls vier bits te zenden.
Vult men de rest van een woord niet aan met bits van het volgende woord, dan kan men deze omzetter ook gebruiken als parallel-parallel aanpasser voor die gevallen dat n niet deelbaar is op m.

Wenst men de kanaalcapaciteit volledig te benutten, dan is een vrij gecompliceerde omzetter nodig. De principe-opzet van zo'n schakeling wordt nu behandeld m.b.v. een omzetter die 5-bit woorden omzet in 3-bit woorden, waarbij de kanaalcapaciteit volledig wordt benut. Als toelichting: het overzenden als 3 + 2 bits leidt tot ca. 17 % verlies aan kanaalcapaciteit.
De te ontwerpen schakeling dient twee telfuncties te realiseren. Een teller houdt bij wanneer de volgende drie bits beschikbaar zijn aan de uitgang. De andere teller signaleert wanneer de volgende vijf bits moeten worden ingelezen. Ook hier is voor een of beide telfuncties gebruik van de tellende eigenschap van een schuifregister mogelijk. We kiezen voor een aparte teller als uitgangsteller. Dit blijkt nl. een heel eenvoudige teller te worden. De teller die aangeeft dat de volgende vijf bits kunnen worden ingelezen wordt als *tellend schuifregister* gerealiseerd. Fig. 10.14 geeft de schakeling, bestaande uit twee instelbare schuif-

registersecties van vier bits, twee flip-flops en enkele NAND's.
Voor een inzicht in de logische werking van de schakeling gaan we uit van een reset via de clear-ingangen van de registers en de uitgangsteller. Na een reset staan de registers A en B en de uitgangsteller in de nulstand. Dit heeft twee gevolgen:

- de mode control s_1 schakelt van 0 naar 1 via poort 3, dus parallel load instelling.
- De besturingsuitgang S geeft aan "wachten".

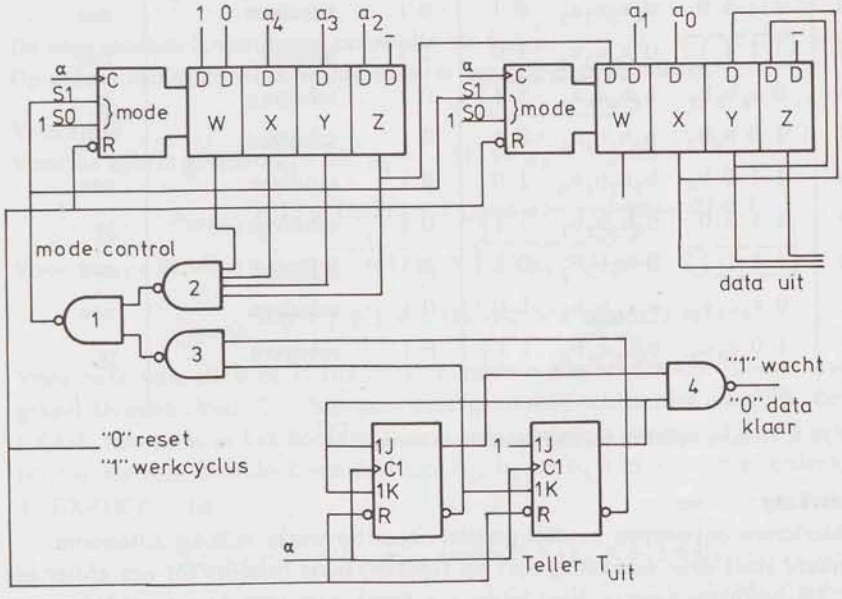


fig. 10.14. Parallel-parallel omzetter (5 → 3 bits).

Na de eerste klokpuls zijn de registers A en B geladen met het eerste datawoord $a_4a_3a_2a_1a_0$. De uitgangsteller komt hierbij in de stand YZ = 01, hetgeen nog steeds wachten signaleert. Inmiddels schakelt de mode control om naar schuiven. Tabel 10.3 beschrijft het verdere verloop van de datastroom in de schakeling.

Uit tabel 10.3 volgt:

- De teller die bijhoudt of een volgend datawoord van vijf bits kan worden geladen is d.m.v. het schuifregister gerealiseerd. Poort 2 detecteert de eindstand waarop geladen moet worden.
- De teller die aangeeft wanneer er drie bits voor verzending gereed staan is apart gerealiseerd. Mede hierdoor is het mogelijk de schakeling te resetten in de stand T_{uit} = 00, welke stand de uitgangspoort 4 (wachten/nieuwe data) bij een reset blokkeert. Gedurende de normale werkcyclus komt de uitgangsteller niet in de stand T_{uit} = 00.

Opmerking

De uitgangsteller in fig. 10.14 is een voorbeeld van een toepassing van een teller, waarin een bewust gebruik is gemaakt van een toestand die niet tot de normale cyclus behoort.

Stap	Inhoud A-register	B-register	Teller	Programma $s_1 s_0$	Inlezen/schuiven	Data gereed
Reset	0 0 0 0	0 0 0 0	0 0	1 1	Inlezen	nee
1	0 $a_4a_3a_2$	$a_1a_0$0 0	0 1	0 1	schuiven	nee
2	1 0 a_4a_3	$a_2a_1a_0$0	1 0	0 1	schuiven	nee
3	1 1 0 a_4	$a_3a_2a_1a_0$	1 1	0 1	schuiven	ja
4	1 1 1 0	$a_4a_3a_2a_1$	0 1	0 1	schuiven	nee
5	1 1 1 1	0 $a_4a_3a_2$	1 0	1 1	inlezen	nee
6	0 $b_4b_3b_2$	$b_1b_0a_4a_3$	1 1	0 1	schuiven	ja
7	1 0 b_4b_3	$b_2b_1b_0a_4$	0 1	0 1	schuiven	nee
8	1 1 0 b_4	$b_3b_2b_1b_0$	1 0	0 1	schuiven	nee
9	1 1 1 0	$b_4b_3b_2b_1$	1 1	0 1	schuiven	ja
10	1 1 1 1	0 $b_4b_3b_2$	0 1	1 1	inlezen	nee
11	0 $c_4c_3c_2$	$c_1c_0b_4b_3$	1 0	0 1	schuiven	nee
12	1 0 c_4c_3	$c_2c_1c_0b_4$	1 1	0 1	schuiven	ja
13	enz.					

tabel 10.3. Datatransport in parallel-parallel omzetter.

Opmerking

De hierboven ontworpen parallel-parallel omzetter werkt volledig autonoom. Uiteraard staat deze schakeling niet op zichzelf, maar behoort tot een groter geheel. De besturing hiervan dient er op zijn beurt weer voor te zorgen dat er tijdig een volgend 5-bit datawoord beschikbaar is om te worden geconverteerd. Ook moet er voor worden gezorgd dat de omzetter geen stap doet zolang de aan het kanaal aangeboden data niet zijn ontvangen resp. zijn overgenomen.

In bovenstaande voorbeelden hebben we voor het eerst kennis gemaakt met het probleem van het ontwerpen van een besturing van een digitale schakeling. In de voorafgaande hoofdstukken doorliepen de schakelingen vaste cycli en behoefde er geen instelingang te worden omgeschakeld. Vergelijkt men de complexiteit van fig. 10.14 met die van een recht-toe-recht-aan teller uit hoofdstuk 9, dan blijkt wel dat het opzetten van een besturingsschema voor digitale schakelingen tot de moeilijkere onderwerpen van de digitale schakeltechniek behoort. Ook de verwerking van de klaarmeldingen van zender en ontvanger levert interessante besturingsproblemen op.
Om een inzicht te krijgen in wat er voor nodig is om tot een schakeling als in fig. 10.14 te komen kan de lezer proberen om een omzetter te ontwerpen die woorden van drie bits omzet in woorden van vijf bits.

10.4. Modulo-2 teruggekoppelde schuifregisters

Van de door schuifregisters opgewekte reeksen worden de *maximum-lengte reeksen* het meest toegepast. Dit komt doordat deze reeksen een aantal prettige

eigenschappen bezitten, terwijl ook de mathematische formulering ervan bekend is. Een afzonderlijke behandeling van dit type reeksen is daarom op zijn plaats.

De optelling modulo-2

Definitie
Onder de *som modulo-2* van een aantal gehele getallen verstaan we de niet-negatieve rest r van de som van de getallen na deling door twee:

$$S = q \cdot 2 + r \quad \text{met} \quad 0 \leqslant r < 2.$$

De som modulo-2 wordt wel aangeduid als $S_{\text{mod-2}}$.
Op geheel identieke wijze wordt de *som modulo-n* gedefinieerd.

Voorbeeld
Voor de gehele getallen $g_1 = 13$, $g_2 = -21$, en $g_3 = 5$ geldt:

$$S_{\text{mod-2}} = 13 + (-21) + 5 \ (\text{mod-2}) = -3 \ (\text{mod-2}) = 1.$$

Voor binaire getallen $g_1 = 0$, $g_2 = 1$, $g_3 = 1$ en $g_4 = 1$ geldt:

$$S_{\text{mod-2}} = 0 + 1 + 1 + 1 \ (\text{mod-2}) = 3 \ (\text{mod-2}) = 1.$$

Voor twee bits, elk 0 of 1, zijn de som modulo-2 en de EX-OR functie ervan geheel identiek. Voor drie bits gaat deze gelijkheid echter niet meer op. De EX-OR functie is in het hoofdstuk over schakelalgebra aangeduid met het symbool $\oplus$. De som modulo-2 van drie bits b_1, b_2 en b_3 kan worden uitgedrukt in de EX-OR functie:

$$S_{\text{mod-2}} = b_1 + b_2 + b_3 \ (\text{mod-2}) = (b_1 \oplus b_2) \oplus b_3.$$

Vaak laat men de haken in uitdrukkingen als boven weg en schrijft men:

$$S_{\text{mod-2}} = b_1 \oplus b_2 \oplus b_3$$

hoewel dit eigenlijk onjuist is. We laten dit slordig gebruik van haken echter toe omdat er geen verwarring door ontstaat als EX-OR poorten met twee ingangen worden gebruikt.

Restklassen modulo-2

De gehele getallen

$$\{\ldots, -3, -2, -1, 0, +1, +2, +3, \ldots\}$$

worden modulo-2 ingedeeld in *restklassen*, de klasse [0] en de klasse [1]. De klasse [0] bevat de even getallen, terwijl de klasse [1] alle oneven getallen bevat:

$$[0] = \{\ldots, -4, -2, 0, +2, +4, \ldots\}$$
$$[1] = \{\ldots, -3, -1, +1, +3, +5, \ldots\}.$$

In een getallensysteem waarin de optelling modulo-2 wordt uitgevoerd, zijn de getallen $+1$ en -1 niet verschillend omdat zij tot dezelfde restklasse behoren. Twee gehele getallen zijn modulo-2 gelijk als

$$a \oplus b \in [0]$$

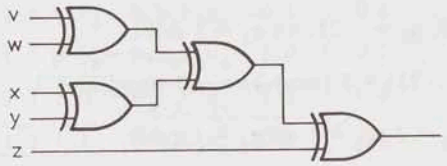
en ongelijk als

$$a \oplus b \in [1].$$

Het minteken komt in een systeem met de optelling modulo-2 derhalve niet voor.

De realisatie van een modulo-2 opteller voor bits

De EX-OR poort is een geschikte bouwsteen om binaire modulo-2 optellers te maken. Voor vijf binaire variabelen v, w, x, y en z geeft fig. 10.15 de schakeling. Ernaast is met haakjes aangegeven hoe deze schakeling uit de formule volgt.



$$S = v + w + x + y + z \pmod 2$$
$$= ((v \oplus w) \oplus (x \oplus y)) \oplus z.$$

fig. 10.15. Modulo-2 opteller voor vijf bits.

De in fig. 10.15 getekende schakeling is niet de enige mogelijkheid om deze opteller te realiseren. Ook de formules

$$S_1 = (((v \oplus w) \oplus x) \oplus y) \oplus z$$

$$S_2 = ((v \oplus w) \oplus x) \oplus (y \oplus z)$$

leiden tot een realisatie. De realisatie van de formule S_1 met EX-OR poorten leidt tot een schakeling met een ongelijke verdeling van de propagatietijden voor de diverse ingangssignalen. De formule S_2 leidt tot een zo gelijk mogelijke verdeling van de propagatietijden.

Modulo-2 teruggekoppelde schuifregisters

Een modulo-2 teruggekoppeld schuifregister is een schuifregister, waarvan de terugkoppelfunctie gelijk is aan de modulo-2 som van een aantal uitgangen van secties van het register. Een modulo-2 teruggekoppeld schuifregister kan verschillende cycli doorlopen. Welke cyclus wordt doorlopen hangt af van
– de gekozen terugkoppelfunctie
– de begininhoud.
Een cyclus, de *nulcyclus*, treedt in elk register dat een terugkoppeling modulo-2 heeft op. Immers, welke uitgangen ook modulo-2 gesommeerd worden, bij de begininhoud (00 . . . 00) is deze som altijd 0. Er schuift dus steeds een 0 in het register.
We bekijken nu eerst enkele voorbeelden van teruggekoppelde schuifregisters.

Voorbeeld
In fig. 10.16 is een schuifregister getekend, waarvan de terugkoppelfunctie f(w,x,y,z) gelijk is aan

$$f(w,x,y,z) = w \oplus x \oplus y \oplus z.$$

Dit register kan verschillende cycli doorlopen:

	W X Y Z
S_0	0 0 0 0

nulcyclus

	W X Y Z
S_1	0 0 0 1
	1 0 0 0
	1 1 0 0
	0 1 1 0
	0 0 1 1

	W X Y Z
S_2	0 0 1 0
	1 0 0 1
	0 1 0 0
	1 0 1 0
	0 1 0 1

	W X Y Z
S_3	0 1 1 1
	1 0 1 1
	1 1 0 1
	1 1 1 0
	1 1 1 1

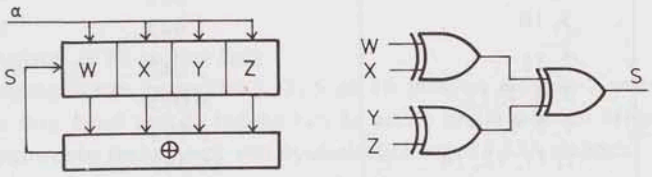


fig. 10.16. Modulo-2 teruggekoppeld schuifregister.

Voorbeeld

	W X Y Z
S_0	0 0 0 0

nulcyclus

	W X Y Z
S_1	0 0 0 1
	1 0 0 0
	0 1 0 0
	0 0 1 0
	1 0 0 1
	1 1 0 0
	0 1 1 0
	1 0 1 1
	0 1 0 1
	1 0 1 0
	1 1 0 1
	1 1 1 0
	1 1 1 1
	0 1 1 1
	0 0 1 1

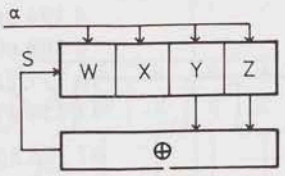


fig. 10.17. Maximum-lengte teruggekoppeld schuifregister.

Het register in fig. 10.17 doorloopt een maximum-lengte reeks ter lengte $2^4 - 1 = 15$. Uitgaande van de beginstand 0001 is de reeks S aan de ingang:

. . . 0 0 0 | 1 0 0 0 1 1 0 1 0 1 1 1 1 0 0 0 | 1 0 0 . . .

één periode van de reeks

Aantal secties	Terugkoppeling van de secties	Lengte van de opgewekte reeks
1	1	1
2	1, 2	3
3	1, 3	7
4	1, 4	15
5	2, 5	31
6	1, 6	63
7	1, 7	127
8	1, 5, 6, 8	255
9	4, 9	511
10	3, 10	1 023
11	2, 11	2 047
12	3, 4, 7, 12	4 095
13	1, 3, 4, 13	8 191
14	1, 11, 12, 14	16 383
15	1, 15	32 767
16	2, 3, 5, 16	65 535
17	3, 17	131 071
18	7, 18	262 143
19	1, 5, 6, 19	524 287
20	3, 20	1 048 575
21	2, 21	2 097 151
22	1, 22	4 194 303
23	5, 23	8 388 607
24	1, 3, 4, 24	16 777 215
25	3, 25	33 554 431
26	1, 7, 8, 26	67 108 863
27	1, 7, 8, 27	134 217 727
28	3, 28	268 435 455
29	2, 29	536 870 911
30	1, 15, 16, 30	1 073 741 823
31	3, 31	2 147 483 647
32	1, 27, 28, 32	4 294 967 295
33	13, 33	8 589 934 591
34	1, 14, 15, 34	17 179 869 183

Tabel 10.4. Terugkoppelingen die leiden tot een maximum-lengte reeks.

De reeks zelf loopt oneindig lang door, maar de lengte van één *cyclus* is $2^4 - 1 = 15$. Zodra een inhoud voor de tweede keer optreedt herhaalt de cyclus zich. De inhoud 0000 kan niet tot een cyclus van lengte groter dan 1 behoren om reeds vermelde redenen. Voor een register van k secties is de maximum lengte van de opgewekte reeks dus $L_{max} = 2^k - 1$.
Theoretisch kan worden aangetoond, dat er voor elke lengte n van een teruggekoppeld schuifregister tenminste één modulo-2 terugkoppelfunctie bestaat waarmee een maximum-lengte reeks kan worden opgewekt. Tabel 10.4 geeft voor schuifregisters van lengte $\leqslant 34$ zo'n terugkoppelfunctie.
Lees deze tabel als volgt:

16 2, 3, 5, 16 65.535

betekent
- het register is 16 secties lang
- de uitgangen van de secties 2, 3, 5 en 16 moeten modulo-2 worden opgeteld en de som moet aan de ingang van de eerste sectie worden aangeboden
- de opgewekte reeks heeft een cycluslengte van 65.535 stappen.

Reeksen van lengte 2^n

Uit een maximum-lengte reeks van lengte $2^n - 1$ kan een reeks van lengte 2^n worden afgeleid. Hiertoe moet de inhoud 00 . . . 00 aan de cyclus worden toegevoegd. De plaats in de cyclus waar dit dient te geschieden is daar waar de inhoud 00 . . . 01 overgaat in de inhoud 10 . . . 00. Fig. 10.18 geeft aan hoe dit eenvoudig kan worden gerealiseerd. Deze oplossing is universeel toepasbaar bij elk maximum-lengte teruggekoppeld schuifregister.

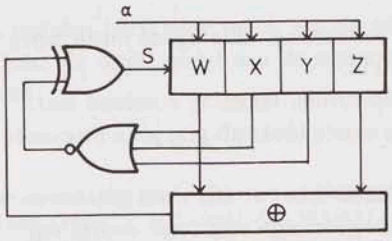


fig. 10.18. Schuifregister met een cycluslengte van 2^n.

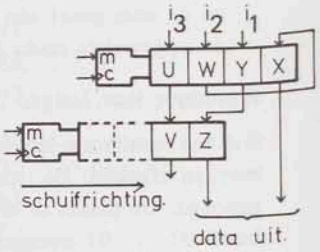
Opgaven

10.1. Ontwerp een driefasen klokpulssysteem op basis van een schuifregister, waarvan de fasen precies een-derde periodeduur t.o.v. elkaar verschoven zijn. De opgewekte pulsen dienen symmetrisch te zijn.

10.2. Is het mogelijk met een schuifregister van drie secties en een geschikt gekozen terugkoppeling alle reekslengten van 1 tot en met 8 te doorlopen? Zo nee, welke reekslengten zijn dan wel en welke niet mogelijk?

10.3. Een schakeling moet groepjes van drie bits omzetten in groepjes van twee bits. Men tracht dit te realiseren met schuifsecties van elk vier bits. Elk blokje van vier bits kan d.m.v. een mode-signaal worden omgeschakeld van parallel laden (m = 0) naar schuiven (m = 1). De omzetting van drie naar twee bits geschiedt zoals is aangegeven:

Reset	0	0	0	0	0	0			
Laden	a_3	·	a_2	·	a_1	·			
Laden	b_3	a_3	b_2	a_2	b_1	a_1	→	$a_2 a_1$	uit
Schuiven	·	·	b_3	b_1	b_2	a_3	→	$b_1 a_3$	uit
Laden	c_3	·	c_2	b_3	c_1	b_2	→	$b_3 b_2$	uit
Laden	d_3	c_3	d_2	c_2	d_1	c_1	→	$c_2 c_1$	uit
Schuiven	·	·	d_3	d_1	d_2	c_3	→	$d_1 c_3$	uit
Laden	e_3	·	e_2	d_3	e_1	d_2	→	$d_3 d_2$	uit
enz.									



Ga na dat de getekende schakeling, mits op de juiste wijze bestuurd, de gewenste omzetting verricht.
Ontwerp vervolgens de besturing, rekening houdend met:
- na een reset zijn na de eerste klokpuls nog geen uitgaande data beschikbaar.
- na iedere twee maal laden moet er één maal geschoven worden.
- van het tweede schuifregister zijn nog twee secties vrij.

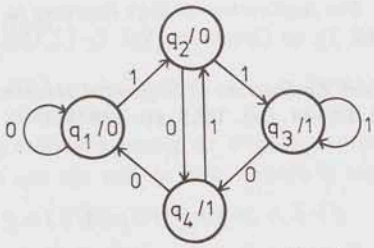
Aangenomen mag worden dat een ander deel van de schakeling ervoor zorgt dat de te converteren data beschikbaar staan.

10.4. Van een schuifregister van vier secties bestaat de terugkoppeling uit de modulo-2 som van de vier uitgangen van de secties.
Welke reeksen worden aan de ingang van de eerste sectie gegenereerd voor de diverse begininhouden?

10.5. Toon aan dat een modulo-2 teruggekoppeld schuifregister slechts een maximum-lengte reeks kan genereren als het aantal teruggekoppelde secties even is.

10.6. Toon aan dat in een maximum-lengte reeks, die door een schuifregister van k secties wordt gegenereerd, $2^{k-1} - 1$ nullen en 2^{k-1} enen voor moeten komen in één periode van de reeks.

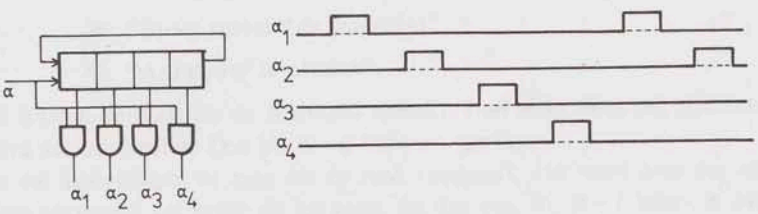
10.7.



In het toestandsdiagram is de werking beschreven van een clock mode sequentiële schakeling. Alle toestandsovergangen vinden plaats op commando van de (niet aangegeven) klokpuls. De schakeling heeft één ingangssignaal dat 0 of 1 kan zijn. Ook het uitgangssignaal U is tweewaardig. De waarde van U is bij elke toestand vermeld.

a. Beschrijf in woorden de functie van deze schakeling.
b. Geef een eenvoudige schakeling die dit gedrag realiseert.

10.8.



In het tijddiagram is aangegeven hoe vier signalen a_1 t/m a_4 van elkaar afhangen. De gegeven schakeling zou een generator zijn voor deze vier signalen. De flip-flops van het register zijn van het type positive edge-triggered.

a. Welke nadelen bezit deze opzet van de schakeling?
b. Modificeer de opzet zodat aan de genoemde bezwaren tegemoet wordt gekomen.

Literatuur

1. W.D.T. Davies, *Generation and properties of Maximal Length Sequences,* Control, Vol. 10, June 1966, pp. 302–304/364–365/431–433.
2. S.W. Golomb, *Shift Register Sequences*, Holden-Day, San Francisco 1967.
3. W.N. Ninke and G.R. Ritchie, *Shift Register Binary Rate Multiplier*, IEEE Tr. on Computers, Vol. C–26, March 1977, pp. 276–278.
4. R.M.M. Oberman, *Disciplines in Combinational and Sequential Circuit Design*, McGraw-Hill, New York 1970.
5. W. Stahnke, *Primitive Binary Polynomials*, Mathematics of Computation, Vol. 27, Oct. 1973, pp. 977–980.
6. *The TTL Data Book for Design Engineers*, Texas Instr., 1977.

Voor een verdere studie van de toepassingen van schuifregisters wordt verwezen naar de volgende bronnen:

7. E.R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, New York 1968.
8. W.D.T. Davies, *System Identification for Self-Adaptive Control*, Wiley-Interscience, London 1970.
9. W.A. Davis, *On Shift Register Realizations for Sequential Machines*, IEEE Conf. Rec., 6th Ann. Symp. on Switching Theory and Logical Design, Ann Arbor, Mich., Oct. 1965, pp. 71–83.

10. D.L. Johnson and K.H. O'Keefe, *The Application of Shift Registers to Secondary State Assignment: Part I and II*, IEEE Tr. on Computers, Vol. C–17, Oct. 1968, pp. 954–977.
11. A.J. Nichols, *Minimal Shift Register Realizations of Sequential Machines*, IEEE Tr. on Electronic Computers, Vol. EC–14, Oct. 1965, pp. 688–700.

Antwoorden deel 1

Hoofdstuk 1

1.1.a. Zij S de uitspraak
"Het bestuur is overeenkomstig de wensen samengesteld".
Dan is S = 1 als aan elk van de drie wensen is tegemoet gekomen. Dus:

$$S = (\overline{A \cdot B}) \cdot (C \oplus D) \cdot (A \cdot C + \overline{A} \cdot \overline{C})$$
$$= \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot \overline{C} \cdot D + A \cdot \overline{B} \cdot C \cdot \overline{D}$$

waarin A staat voor "persoon A is in het bestuur opgenomen", enz..

1.1.b. Maximaal twee, nl. A en C resp. B en D.

1.2. De inclusieve OF. Denk aan de regenboog.

1.3.a. De vraag is: "Welke kant zal uw broer aanwijzen als ik hem de weg naar A vraag?"

1.3.b. We doen dit met de proposities W en C:

W: "Broer spreekt de waarheid",

C: "Antwoord is correct".

Stel we treffen de man die de waarheid spreekt. Dan weet deze dat zijn broer het antwoord zal omdraaien. Dus bij W = 0 behoort C = 0.
Treffen we daarentegen de man die de zaak omdraait, dan weet deze dat zijn broer het juiste antwoord zal geven en hij keert dit dus om. Bij W = 1 behoort dus ook C = 0.
Met andere woorden, het antwoord is altijd fout. De andere weg is dan goed.

1.4. Noem de variabelen D, K en A_1 t/m A_4. Dan is S, de formule voor het slot:

$$S = D + K(A_1 + A_2 + A_3 + A_4) + A_1A_2A_3 + A_1A_2A_4 + A_1A_3A_4 + A_2A_3A_4 .$$

1.5.a.
$$(x + y)(x + z)(y + z) = (x + yz)(y + z)$$
$$= xy + xz + yz \cdot y + yz \cdot z$$
$$= xy + xz + yz + yz = xy + xz + yz .$$

1.5.b. Nee.

1.6.
$$S_1 = \bar{x}\bar{y} + x\bar{z} + yz = \bar{x}\bar{y}(\bar{z} + z) + x(\bar{y} + y)\bar{z} + (x + \bar{x})yz$$
$$= \bar{x}\bar{y}\bar{z} + \bar{x}\bar{y}z + x\bar{y}\bar{z} + xy\bar{z} + xyz + \bar{x}yz$$
$$= \bar{x}(\bar{y} + y)z + xy(\bar{z} + z) + (x + \bar{x})\bar{y}\bar{z}$$
$$= \bar{x}z + xy + \bar{y}\bar{z} = S_2 .$$

1.7. $S_1 = x$ $\qquad$ $S_2 = \bar{w}\bar{x} + wy\bar{z}$.

1.8. $xy + \bar{x}z + yz = (x + z)(\bar{x} + y)$.

1.9. Minterm vorm : $S = \bar{x}y\bar{z} + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z}$
Maxterm vorm : $S = (\bar{x} + \bar{y} + \bar{z})(x + \bar{y} + \bar{z})(x + y + \bar{z})(x + y + z)$
Som-van-produkten : $S = x\bar{y} + y\bar{z}$
Produkt-van-sommen : $S = (\bar{y} + \bar{z})(x + y)$.

1.10. $f_0 = f_1 = f_2 = f_3 = f_4 = f_6 = 0$, $f_5 = f_7 = 1$.

1.11.a. $S = \overline{\bar{x}\bar{y} + \bar{z}} = (x + y)\bar{z}$.

1.11.b. De duale vorm van de oorspronkelijke formule. Zie theorie.

1.12.a. $S = m_1, m_2, m_4, m_7 = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

1.12.b. Het bewijs berust op de regel dat $a \oplus 1 = \bar{a}$.

1.13.a. $\bar{S} = \overline{xy + xz + yz} = (\bar{x} + \bar{y})(\bar{x} + \bar{z})(\bar{y} + \bar{z}) = \bar{x}\bar{y} + \bar{x}\bar{z} + \bar{y}\bar{z}$.

1.13.b. Nee.

Hoofdstuk 2

De volgorde van de variabelen in de Karnaughdiagrammen is zoals in de figuren 2.3. resp. 2.10. We geven alleen de inhoud aan.

2.1.

1	2	3	2
0	1	2	1

2.2.

0	1	2	1	2	3	2	1
1	2	3	2	3	4	3	2
2	3	4	3	4	5	4	3
1	2	3	2	3	4	3	2

2.3.

1	0	1	0	0	1	0	1	0	0	0	0
0	0	1	1	1	0	1	0	0	0	1	0
1	1	1	0	0	1	0	1	0	1	0	1
0	0	1	0	1	0	1	0	0	0	1	0

Opmerking: Schakelfuncties met de exclusieve OF in plaats van de inclusieve OF kunnen we gemakkelijk in een Karnaughdiagram plaatsen. We plaatsen dan alle termen afzonderlijk, waarbij dus meer dan één 1 in een hokje kan komen te staan als dit hokje door verschillende termen wordt bedekt. Vervolgens wordt het aantal enen in elk hokje geteld. Is dit even, dan wordt een 0 ingevuld, anders een 1.

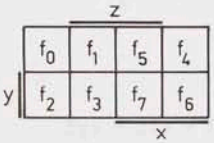
2.4.

0	0	0	1
1	1	1	1
1	0	1	1
0	0	1	1

2.5. $\bar{z} + z = 1$

$z + z = z$.

2.6. Ga uit van een Karnaughdiagram in drie variabelen, waarin de functiewaarden f_0 t/m f_7 geplaatst zijn.



Teken hierin de gebieden die bedekt worden door elk van de termen indien de bijbehorende factor $c_i = 1$ is. Dan blijkt dat de minterm behorend bij f_0 uitsluitend bedekt wordt door $c_0 \cdot 1$. Dus geldt dat $c_0 = f_0$ is.
Evenzo vindt men dat de minterm behorende bij f_1 bedekt wordt door $c_0 \cdot 1$ en $c_1 \cdot z$. Hiervan heeft c_0 reeds een waarde, nl. f_0. Dus moet gelden:

$$c_0 \oplus c_1 = f_0 \oplus c_1 = f_1 .$$

Hieruit volgt de waarde van c_1.
Op overeenkomstige wijze volgt de waarde van de termen f_2 en f_4; waarna f_3, f_5 en f_6 en tenslotte f_7.
Het tweede deel volgt gemakkelijk uit een tegenvoorbeeld.

2.7. $c_0 = c_2 = c_4 = c_5 = c_6 = 1 \qquad c_1 = c_3 = c_7 = 0 .$

2.8. $S_1 = w\bar{x}y + xy\bar{z} + \bar{y}z = (y \div z)(\bar{x} + \bar{y} + \bar{z})(w + x + \bar{y}/z)$

$S_2 = \bar{w}x\bar{z} + w\bar{y}z + w\bar{x} + y\bar{z} = (\bar{w} + \bar{x} + z)(w + x + y)(w + \bar{z})(\bar{x} + \bar{y})$

$S_3 = wy\bar{z} + xz + \bar{x}\bar{y} = (x + \bar{y} + \bar{z})(\bar{x} + y + z)(w + \bar{x}/\bar{y} + z)$

2.9.a. Priemimplicanten S_1 : $\{w\bar{x}y, wy\bar{z}, xy\bar{z}, \bar{x}z, \bar{y}z\}$,

S_2 : $\{\bar{w}x\bar{z}, w\bar{y}z, w\bar{x}, y\bar{z}\}$,

S_3 : $\{wxy, wy\bar{z}, \bar{w}\bar{x}, \bar{w}\bar{y}, \bar{w}z, \bar{x}\bar{y}, \bar{x}\bar{z}, xz, \bar{y}z\}$.

2.9.b. Als voorbeeld de dekkingstabel voor S_3.

		m_0	m_1	m_7	m_8	m_9	m_{10}	m_{13}	m_{14}	m_{15}	
wxy	A								×	×	
wyz	B						⊗		⊗		
wx	C	⊗	⊗								vervalt
wy	D	⊗	⊗								vervalt
wz	E		×	×							
xy	F	⊗	⊗		⊗	⊗					dominant over C en D.
xz	G	×			×		×				
xz	H			⊗				⊗		⊗	
yz	I		×			×		×			

Petrickfunctie :

$$S = (F + G)(E + F + I)(E + H)(F + G)(F + I)(B + G)(H + I)(A + B)(A + H)$$

$$= \underline{BFH} + AEGI + AFGH + AGHI + BGHI + ABEFI .$$

2.10.a. Nee. De combinaties $S_1S_2S_3 = 100$ en 101 komen niet voor. De oorzaak is dat $S_1S_2 = 10$ niet kan optreden. S_1 en S_2 zijn afhankelijke variabelen! Dit komt vaak voor!

2.10.b. $\bar{P} = \bar{S}_2 + \bar{S}_1\bar{S}_3 .$

Zou men geen gebruik maken van het onder a. gevondene, dan is $\bar{P} = \bar{S}_1\bar{S}_2 + \bar{S}_1\bar{S}_3 .$

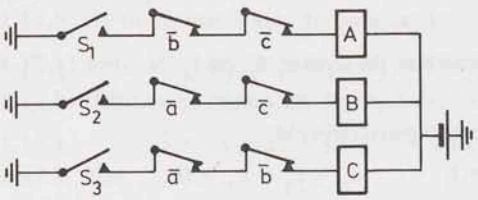
Hoofdstuk 3

3.1.a. De schakeling detecteert of w èn x alsmede y èn z paarsgewijs dezelfde waarde (stand) hebben.

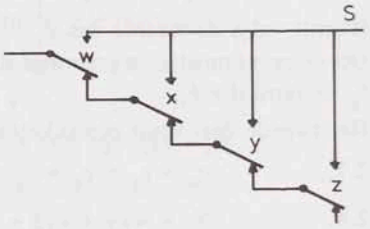
3.1.b. $S = \bar{w}\bar{x}(\bar{y}\bar{z} + yz) + wx(\bar{y}\bar{z} + yz) = (\bar{w}\bar{x} + wx)(\bar{y}\bar{z} + yz) .$

Zie verder fig. 3.21.

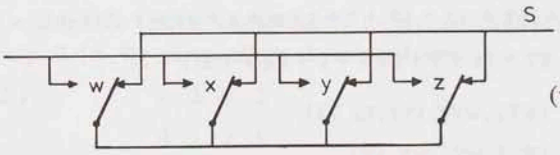
3.2.



3.3.



3.4.



(via inverse formule)

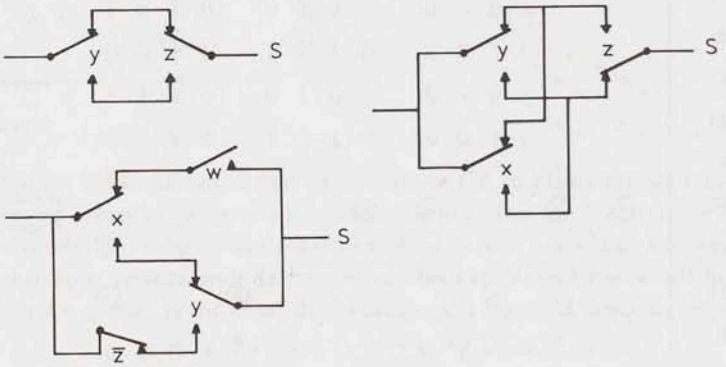
Men vindt een variant, ook met vier wisselkontakten, door de formule naar de variabele w te ontbinden en vervolgens de twee residuen te vereenvoudigen:

$$S = w(\bar{x} + \bar{y} + \bar{z}) + \bar{w}(x + y + z) .$$

3.5. De schakeling correspondeert direct met de formule

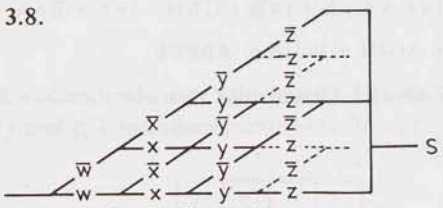
$$S = uv + (w + y)(x + z)$$ (6 maakkontakten) .

3.6.



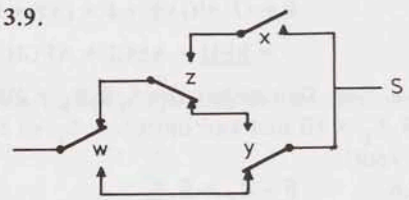
3.7. $S = \bar{w} + \bar{x}\bar{y} + xy$.

3.8.



8 wisselkontakten

3.9.

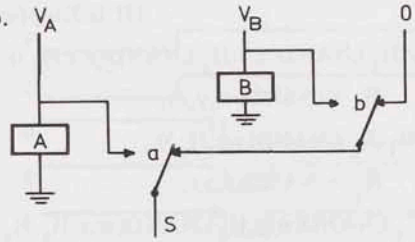


3.10. $S = \bar{w}\bar{x}y + w\bar{x}\bar{y}$.

3.11.a. Stel A de uitspraak "$V_A \neq 0$" en B de uitspraak "$V_B \neq 0$".

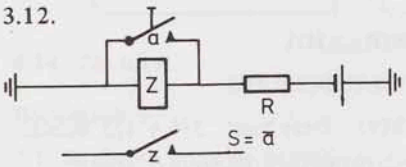
$$S = A \cdot V_A + \bar{A}B \cdot V_B + \bar{A}\bar{B} \cdot V_0 .$$

3.11.b.

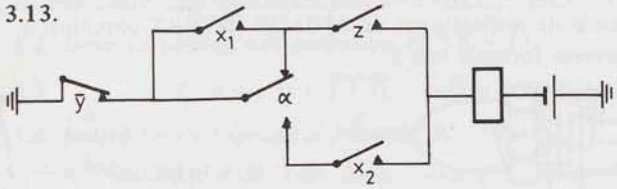


De schakeling is uitbreidbaar.

3.12.



3.13.



Aangenomen is dat de overige kontakten wel snel schakelen. Dan overbrugt x_2 de omschakeling van *a*.

Hoofdstuk 4

4.1.a. $S_1 = \bar{w}\bar{y}z + \bar{w}x + w\bar{x}y + w\bar{z}/x\bar{z}/y\bar{z}$.

$S_1 = (\bar{w} + \bar{x} + \bar{z})(\bar{w} + y)(w + x + \bar{y})(y + z)$.

$S_2 = \bar{w}\bar{x}\bar{y} + \bar{w}\bar{x}\bar{z} + wxy + \bar{y}\bar{z}$.

$S_2 = (\bar{w} + x)(w + \bar{x})(x + \bar{y} + \bar{z}/w + \bar{y} + \bar{z})(\bar{w} + y + \bar{z}/\bar{x} + y + \bar{z})$.

4.1.b. 1. Deze volgen direct uit de eenvoudigste som/produkt vorm.

2. $S_1 = \text{AND}(\text{OR}(w,x,\text{AND}(\bar{y},z)),\text{OR}(\bar{w},\bar{z},\text{AND}(\bar{x},y)))$.

$S_2 = \text{OR}(\text{AND}(w,x,\text{OR}(y,\bar{z})),\text{AND}(\bar{w},\bar{x},\text{OR}(\bar{y},\bar{z})))$.

3. Oplossingen identiek met die onder 1.

4.2. $S = \bar{w}\bar{x}y + wx\bar{y}z$.

4.3. $S = \underline{\underline{\bar{w}y(x + \bar{z})}} + \underline{\underline{\bar{x}\bar{y}(w + z)}} + \underline{x\bar{z}}$ 9 poorten,

$S = x(\bar{w} + \bar{z}) + \bar{x}\bar{y}(w + z) + \bar{w}y\bar{z}$ 9 poorten.

4.4.a. $S_i = x_{i-1}x_{i-2} \ldots x_2 x_1$.

4.4.b. $F_i = x_i S_i + x_{i+1}$.

4.5. $S = \text{NAND}(\text{NAND}(w,x),\text{NAND}(w,y))$.

4.6.a. Nee.

4.6.b. $S_i = x_i x_{i-1} + \bar{x}_i x_{i+1} = (x_i + x_{i+1})(\bar{x}_i + x_{i-1})$.

4.6.c. $\text{NAND}(\text{NAND}(x_{i-1},x_i),\text{NAND}(\text{NAND}(x_i,x_i),x_{i+1}))$.

$\text{NOR}(\text{NOR}(x_{i+1},x_i),\text{NOR}(\text{NOR}(x_i,x_i),x_{i-1}))$.

4.7. $H_1 = NAND(x,y,z)$. (H is Hulpfunctie!).

$S_1 = NAND(NAND(x,y,H_1),NAND(x,z,H_1),NAND(y,z,H_1))$.

$H_2 = NAND(x,x)$. $H_3 = NAND(w,y,z)$.

$S_2 = NAND(NAND(w,H_1,H_2),NAND(y,z,H_3))$.

$H_4 = NAND(w,z)$. $H_5 = NAND(x,y)$.

$S_3 = NAND(NAND(z,H_4),NAND(w,y,H_5),NAND(w,x,H_4,H_5))$.

4.8. $H = NAND(x,y,z)$.

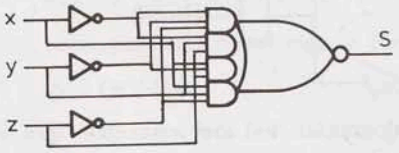
$S = NAND(NAND(w,x,H),NAND(y,z,H))$.

4.9. $S = NAND(NAND(NAND(\overline{v},\overline{w}),NAND(\overline{x},\overline{y},\overline{z})))$.

4.10. Oplossing berust op $S = (x + \overline{x}\overline{y}\overline{z})(y + \overline{x}\overline{y}\overline{z})$. Deze kost 3/4 + 1/3 huisje.

$H = NOR(x,y,z)$; $S = NOR(NOR(x,H),NOR(y,H))$.

4.11.a. Een geschikte keuze is de realisatie via de AND-OR-INVERT combinatie en wel uitgaande van de inverse formule van S. .



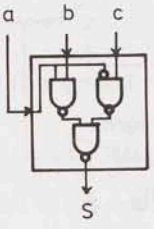
1 + ½ huisje.

4.11.b. Schakeling berust op

$S = \overline{p}x + \overline{p}y + \overline{p}z + xyz$. 1 1/6 huisje.

$S = INV(OR(AND(\overline{p},x),AND(\overline{p},y),AND(\overline{p},z),AND(x,y,z)))$.

4.12.



Bouwsteen

S_1 : (a,b,c = y,z,0) ,

S_2 : (a,b,c = y,1,z) ,

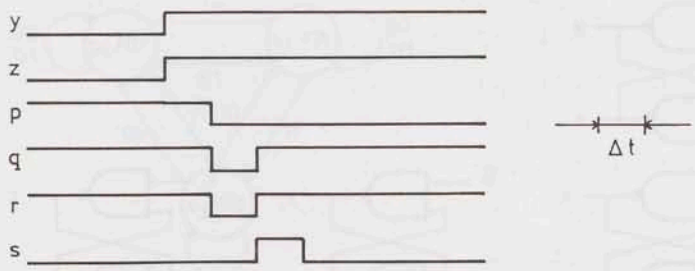
S_3 : (a,b,c = y,x,z) ,

H_1 : (a,b,c = y,1,z) , S_4 : (a,b,c = x,H_1,0) ,

H_2 : (a,b,c = w,x,0) , H_3 : (a,b,c = y,z,0) S_5 : (a,b,c = H_2,1,H_3)

H_4 : (a,b,c = w,y,0) , H_5 : (a,b,c = x,H_4,$\overline{y}$) S_6 : (a,b,c = z,x,H_5) .

4.13.



4.14. Zie tekst.

Hoofdstuk 5

5.1. Bepalend voor de definitieve stand is degene die het *laatst* zijn drukknop loslaat. Hierop heeft het "overheersende set karakter" geen invloed.

5.2. Deze schakeling zou genereren bij SR = 11.

5.3. $Z_1 = s + \bar{r}z = \overline{\overline{s + \bar{r}z}} = \overline{\bar{s} \cdot \overline{\bar{r}z}} = \text{NAND}(\bar{s},\text{NAND}(\bar{r},z))$.

5.4. Indien bij de ingangscombinatie SR = 00 :

SR = 00 $\Rightarrow$ Beide uitgangen 1 $\Rightarrow$ $\bar{S}$-$\bar{R}$ trekker.

SR = 00 $\Rightarrow$ Uitgangen elkaars inverse, dan SR = 11 aanbieden.

SR = 11 $\Rightarrow$ Beide uitgangen 0 $\Rightarrow$ S-R trekker.

SR = 11 $\Rightarrow$ Uitgangen elkaars inverse $\Rightarrow$ Z_z trekker.

5.5. Nee. Bij kontakten die geheel terugveren helpt deze oplossing niet. In dat geval kan men beter andere relais aanschaffen.

5.6. Met het wisselkontakt ziet de anti-dender schakeling drie ingangstoestanden:

kontakt aan maakzijde,

kontakt zwevend,

kontakt aan verbreekzijde.

Veert tijdens het denderen het kontakt niet geheel terug, dan kan de schakeling zijn taak verrichten door het laatste commando (maak resp. verbreek) aan te houden tijdens het zweven.

Bij maakkontakten kan de schakeling geen onderscheid maken tussen

zweven (dynamisch),

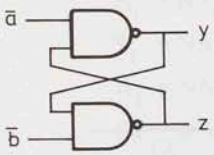
verbroken (statisch).

Het verdient daarom aanbeveling voor drukschakelaars wisselkontakten te kiezen.

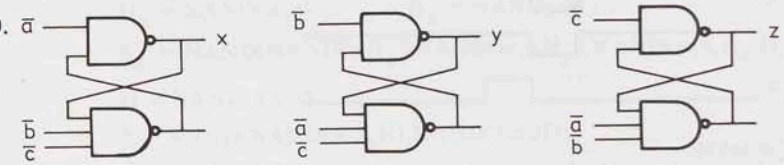
5.7. Dit is mogelijk. Men dient dan een zgn. "one shot" of " monostabiele multivibrator" toe te passen, die de dendertijd moet overbruggen.

5.8. Bepalend voor de nieuwe (onthoud)stand van trekker Z is de waarde van a aan 't eind van de periode dat c = 1 is. De spike valt in het begin van deze periode.

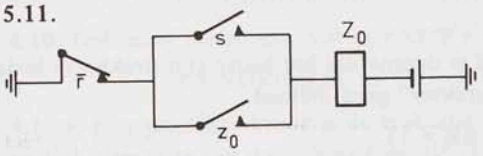
5.9.



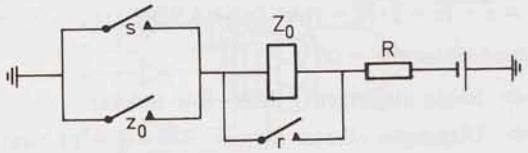
5.10.



5.11.



5.12.



Hoofdstuk 6

6.1. Zie tekst.

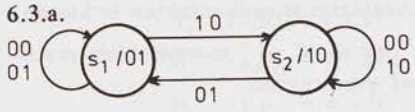
6.2.a. Nee, in het algemeen niet.

6.2.b/c.

a b / y_1 y_2	00	01	10	11	00	01	10	11
00	–	–	–	–	–	–	–	–
01	(01)	10	(01)	(01)	0	0	0	0
10	01	(10)	(10)	(10)	0	0	1	1
11	–	–	–	–	–	–	–	–

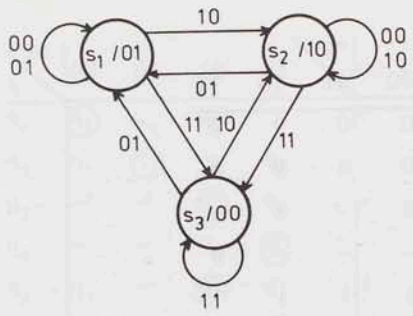
6.2.d. Is a = 0, dan is S = 0.
Gaat a van 0→1, dan wordt b op dat moment bemonsterd en de uitgang S krijgt deze waarde gedurende de gehele tijd dat a = 1 is.

6.3.a.



Volgorde: SR resp U_1U_2

6.3.b.



6.3.c. Dit valt buiten het toegestane gebruik van toestandsdiagrammen.

6.4.a.

q \ $d_1 d_2$	00	01	10	11	00	01	10	11
q_1	(1) q_1	q_2	q_3	–	0	–	0	–
q_2	q_4	(2) q_2	–	–	1	1	–	–
q_3	q_1	–	(3) q_3	–	0	–	0	–
q_4	(4) q_4	q_2	q_3	–	1	1	–	–

primitieve, niet gereduceerde tabel.

6.4.b. De trekker.

6.5. De uitgang U wordt 1 bij een 0 → 1 overgang in het signaal a en wordt 0 bij een 0 → 1 overgang in het signaal b. In de overige gevallen blijft $U_{nieuw} = U_{oud}$.

6.6.a. De schakeling geeft een alarm (U = 1) wanneer er twee of meer opeenvolgende a-pulsen worden ontvangen.

6.6.b. q_1 : de laatste puls was een a-puls.
q_2 : alarm, er zijn twee of meer a-pulsen ontvangen.
q_3 : de laatste puls was een b-puls; dan nooit alarm.
q_4 : er wordt een a-puls aangeboden.

6.7.a. α en β zijn nooit tegelijk 1. Bovendien: α en β veranderen waarschijnlijk niet tegelijk. Omdat de toestandstabel niet primitief is, kan men dit niet met zekerheid vaststellen.

6.7.b. Uitgaande van de veronderstelling dat α en β niet tegelijk 1 zijn, en α en β pulsreeksen zijn, dan laat de schakeling van elke cyclus α-pulsen resp. cyclus β-pulsen alleen de eerste puls door. De schakeling geeft dus om en om een α-puls resp. β-puls door.

6.8.

q \ ab	00	01	10	11	U: 00	01	10	11
q_1	$\textcircled{q_1}$	q_2	q_3	q_4	0	0	0	0
q_2	q_1	$\textcircled{q_2}$	q_3	q_4	0	0	0	0
q_3	q_1	–	$\textcircled{q_3}$	–	0	–	0	–
q_4	–	q_6	–	$\textcircled{q_4}$	–	–	–	0
q_5	$\textcircled{q_5}$	q_6	q_7	q_8	1	1	1	1
q_6	q_5	$\textcircled{q_6}$	q_7	q_8	1	1	1	1
q_7	q_5	–	$\textcircled{q_7}$	–	1	–	1	–
q_8	–	q_2	–	$\textcircled{q_8}$	–	–	–	1

6.9.a.

ma \ m*	00	01	10	11	00	01	10	11
q_1	$\textcircled{q_1}$	q_2	q_3	q_4	0	0	0	–
q_2	q_1	$\textcircled{q_2}$	*	q_4	0	0	*	–
q_3	q_1	q_2	$\textcircled{q_3}$	q_4	0	0	0	–
q_4	*	q_2	q_5	$\textcircled{q_4}$	*	–	1	1
q_5	q_6	q_2	$\textcircled{q_5}$	q_4	1	–	1	1
q_6	$\textcircled{q_6}$	q_2	q_5	q_4	1	–	1	1

6.9.b. Wanneer a van $1 \to 0$ gaat terwijl m verandert, dan kan de laatste waarde van m niet ondubbelzinnig worden vastgelegd. Deze situaties zijn met * gemerkt.
6.9.c. De "gated latch".

6.10. De tabel kan gemakkelijk gevonden worden via de tussenstap van een toestandsdiagram.

ab \ U	00	01	10	11	00	01	10	11
q_1	q_2	$\textcircled{q_1}$	–	q_7	0	0	–	0
q_2	$\textcircled{q_2}$	q_1	q_3	–	0	0	–	–
q_3	q_5	–	$\textcircled{q_3}$	–	–	–	1	–
q_4	q_5	–	$\textcircled{q_4}$	q_7		–	0	0
q_5	$\textcircled{q_5}$	q_6	q_4	–	0	–	0	–
q_6	q_2	$\textcircled{q_6}$	–	q_7	–	1	–	–
q_7	–	q_1	q_4	$\textcircled{q_7}$	–	0	0	0

6.11.

q \ xa	00	01	10	11	a* 00	01	10	11
q_1	(1)	q_2	q_3	–	0	–	–	–
q_2	q_1	(2)	–	q_4	–	1	–	1
q_3	–	–	(3)	q_6	–	–	1	–
q_4	–	–	q_5	(4)	–	–	1	1
q_5	q_7	–	(5)	q_6	1	–	1	–
q_6	–	q_8	q_5	(6)	–	–	–	0
q_7	(7)	q_2	–	–	1	1	–	–
q_8	q_1	(8)	–	–	–	1	–	–

Opmerking. De toestanden $q_1 - q_2$ vormen de hoofdcyclus als $x = 0$ is. Voor $x = 1$ zijn dit de toestanden $q_5 - q_6$. De overige toestanden dienen om het omschakelen in goede banen te leiden. De functie van de gespecificeerde schakeling zou men als volgt kunnen samenvatten:

"Van klokpuls a^* is de neergaande flank de actieve flank. Deze moet als $x = 0$ is corresponderen met de neergaande flank van a. Is $x = 1$, dan is de opgaande flank van a de corresponderende flank".

6.12.a. $Y_1 = ay_1 + \bar{b}y_2 + y_1y_2$.

$Y_2 = a\bar{b} + a\bar{y}_1 + \bar{b}y_2$.

6.12.b.

y_1y_2 \ ab	00	01	10	11	U
00	(00)	(00)	.	.	0
01	.	.	.	(01)	1
10	.	.	.	(10)	0
11	(11)	.	(11)	.	1

6.12.c.

y_1y_2 \ ab	00	01	10	11	U
00	(00)	(00)	11*	01	0
01	11	00	11	(01)	1
10	00	00	11	(10)	0
11	(11)	00*	(11)	10	1

Opmerking. De overgangen 11* en 00* verlopen via tussentoestanden $y_1y_2 = 01$ resp. $y_1y_2 = 10$.

6.13.a. In dat geval werkt de beveiliging niet. De schakeling degradeert tot een één-knops bediening.

6.13.c.

q \ LR	00	01	10	11	Z
q_1	(q_1)	q_2	q_3	q_4	0
q_2	q_1	(q_2)	q_3	q_4	0
q_3	q_1	q_2	(q_3)	q_4	0
q_4	q_1	q_5	q_6	(q_4)	1
q_5	q_1	(q_5)	q_6	q_7	0
q_6	q_1	q_5	(q_6)	q_7	0
q_7	q_1	q_7	q_7	(q_7)	0

Opmerking. Na iedere bekrachtiging ($Z = 1$) moeten beide knoppen tenminste één maal losgelaten worden eer er weer een nieuwe bekrachtiging kan volgen.

Hoofdstuk 7

7.1. In fig. 7.5. wordt $\overline{a}$ apart gemaakt. In plaats hiervan zijn wel de signalen

$$\overline{S}_1 = \overline{a} + \overline{y}_2 \qquad \text{en} \qquad \overline{R}_1 = \overline{a} + \overline{y}_2$$

beschikbaar. Gebruik van deze signalen levert

$$S_2 = y_1(\overline{a} + y_2) \qquad \text{en} \qquad R_2 = \overline{y}_1(\overline{a} + \overline{y}_2).$$

Toevallig behoren de termen $y_1 y_2$ van S_2 en $\overline{y}_1 \overline{y}_2$ van R_2 tot de don't care gebieden van S_2 en R_2. Zie fig. 7.4.

7.2. De sleutel tot het bewijs ligt bij de constatering dat voor een volledig gespecificeerde tabel geldt:

$$\Gamma_{q_a} = \Gamma_{q_b} = \Gamma_{q_c} = \Gamma,$$

de verzameling van gespecificeerde ingangswoorden.

7.3. De rij $[i_1 i_2 i_3]$ is bij het onderzoek naar de compatibiliteit van toestanden identiek met de rij $[i_3]$.

7.4.a.

	q_1	q_2	q_3	q_4	q_5	q_6
q_1		.	.	.	.	×
q_2			.	.	×	×
q_3				.	.	.
q_4					×	×
q_5						.
q_6						

7.4.b.

	q_1	q_2	q_3	q_4	q_5	q_6
q_1		×	—	×	×	×
q_2			×	×	×	×
q_3				×	×	×
q_4					×	×
q_5						×
q_6						

7.5. $q_1 \sim q_2$; $q_2 \sim q_3$; $q_2 \sim q_5$.

Op de overige plaatsen staan kruisjes.

7.6.

r \ ab	00	01	10	11	00	01	10	11
$\{q_1,q_3\} \to r_1$	(1)	r_2	(1)	r_4	0	–	0	0
$\{q_2,q_5\} \to r_2$	r_3	(2)	r_1	(2)	1	1	–	1
$\{q_4,q_8\} \to r_3$	(3)	r_4	(3)	r_2	1	–	1	1
$\{q_6,q_7\} \to r_4$	r_1	(4)	r_3	(4)	0	0	–	0

Een andere minimale oplossing berust op:

$$C = \{\{q_1,q_5\},\{q_2,q_8\},\{q_3,q_7\},\{q_4,q_6\}\}.$$

7.7.a. $\xi = \{\{q_1,q_2,q_3\},\{q_2,q_8\},\{q_3,q_4\},\{q_3,q_7\},\{q_4,q_6\},\{q_4,q_8\},\{q_5,q_6,q_7\},\{q_7,q_8\}\}$.

7.7.b. Uit a. volgt dat er tenminste drie toestanden nodig zijn, waaronder twee die gebaseerd zijn op $\{q_1,q_2,q_3\}$ en $\{q_5,q_6,q_7\}$. Dit laatste vanwege het eenmalig voorkomen van q_1 en q_5. De overige twee toestanden, $\{q_4,q_8\}$, liggen ook in één klasse. Met deze drie klassen vinden we een dekking, welke gesloten blijkt te zijn.

r \ ab	00	01	10	11	00	01	10	11
$\{q_1,q_2,q_3\} \to r_1$	(1)	(1)	(1)	r_2	0	0	0	0
$\{q_4,q_8\} \to r_2$	r_1	r_3	(2)	(2)	–	–	1	0
$\{q_5,q_6,q_7\} \to r_3$	(3)	(3)	r_2	(3)	1	1	1	1

7.8. Met $\delta(q_2,i_2) = q_1$ is zo'n codering mogelijk:

$$q_0: y_1y_2 = 00 \quad q_1: y_1y_2 = 11 \quad q_2: y_1y_2 = 01 \quad q_3: y_1y_2 = 10.$$

7.9.a. Elk van de drie coderingen van par. 7.1. geeft aanleiding tot kritische race-condities.

7.9.b. Ja, met $\delta(q_2,00) = q_1$, $\delta(q_0,11) = q_1$ en $\delta(q_1,11) = q_2$.
Een mogelijke codering is dan:

$$q_1: y_1y_2 = 00 \quad q_2: y_1y_2 = 01 \quad q_3: y_1y_2 = 11 \quad q_4: y_1y_2 = 10.$$

Er treedt geen uitgangsspike op.
Ook met $\delta(q_1,00) = q_2$ in plaats van q_0 is een codering met twee variabelen mogelijk:

$$q_1: y_1y_2 = 00 \quad q_2: y_1y_2 = 11 \quad q_3: y_1y_2 = 10 \quad q_4: y_1y_2 = 01.$$

Hierbij treedt een uitgangsspike op.
Een voordeel van de eerste codering is nog dat de uitgangswaarde U direct bepaald wordt door één toestandsvariabele.

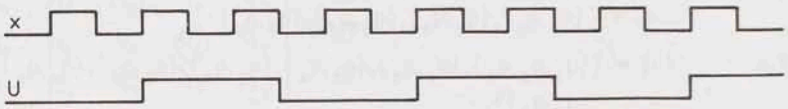
7.10.a.

$y_1\ y_2\ y_3$	q	00	01	10	11
0 0 1	q_5	q_1	q_2	q_3	q_3
0 1 0	q_6	q_1	q_2	–	q_2
1 0 0	q_7	q_4	–	q_3	q_3
1 1 1	q_8	q_4	q_2	q_3	q_2

7.10.b. Ja, met $\delta(q_3,01) = q_1$ en $\delta(q_2,10) = q_4$.

7.11. $S_1 = ab\overline{y}_2$, $S_2 = \bar{a}b\overline{y}_1$, $R_1 = a\overline{b}$, $R_2 = a\overline{b}$.

7.12.a.



7.12.b. De werking kan omschreven worden als "symmetrische driedeler".

7.12.c. q_1 t/m q_6: $y_1y_2y_3$ = 000/001/011/111/101/100/ .

7.12.d. $S_{y_1} = xy_2$ $\quad S_{y_2} = \bar{x}\overline{y}_1y_3$ $\quad S_{y_3} = x\overline{y}_1$

$R_{y_1} = \bar{x}\overline{y}_3$ $\quad R_{y_2} = \bar{x}y_1$ $\quad R_{y_3} = xy_1\overline{y}_2$

7.12.e. 3,5 inclusief inversie van x.

7.13.a.

q \ abc	000	001	010	100	000	001	010	100	
q_1	(1)	q_4	q_5	q_6	001	001	0--	-0-	c werd gedrukt
q_2	(2)	q_4	q_5	q_6	010	0--	01-	--0	b werd gedrukt
q_3	(3)	q_4	q_5	q_6	100	-0-	--0	100	a werd gedrukt
q_4	q_1	(4)	–	–	001	001	–	–	c wordt gedrukt
q_5	q_2	–	(5)	–	010	–	010	–	b wordt gedrukt
q_6	q_3	–	–	(6)	100	–	–	100	a wordt gedrukt

7.13.b.

r \ abc	000	001	010	100	
r_1	(1)	(1)	r_2	r_3	001
r_2	(2)	r_1	(2)	r_3	010
r_3	(3)	r_1	r_2	(3)	100

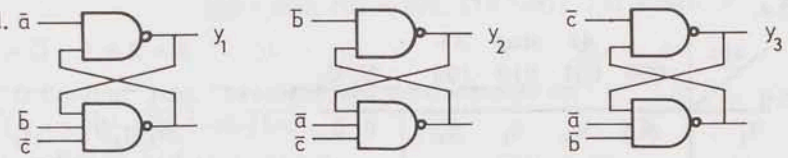
7.13.c. Alle coderingen met twee variabelen zijn geschikt. Hierbij moeten wel tussentoestanden worden gespecificeerd.
Houden we rekening met de 3-bit uitgangscode, dan is een geschikte codering:

$$r_1/r_2/r_3: \quad y_1y_2y_3 = 001/010/100 .$$

De toestandstabel moet dan worden aangevuld met:

$y_1 y_2 y_3$	r	000	001	010	100	
0 1 1	r_4	–	r_1	r_2	–	–
1 0 1	r_5	–	r_1	–	r_3	–
1 1 0	r_6	–	–	r_2	r_3	–
0 0 0	r_7	–	r_1	r_2	r_3	–

7.13.d.



Schakeling is uitbreidbaar.

7.13.e. Nee.

7.14.b.

q \ ab	00	01	10	11	00	01	10	11
q_1	(1)	–	q_7	–	0	–	–	–
q_2	q_1	(2)	q_7	q_8	0	0	–	–
q_3	q_1	–	(3)	–	0	–	0	–
q_4	q_1	q_2	q_3	(4)	0	0	0	0
q_5	(5)	q_2	–	–	1	–	–	–
q_6	q_5	(6)	–	–	1	1	–	–
q_7	q_5	q_2	(7)	q_4	1	–	1	–
q_8	q_5	q_6	q_7	(8)	1	1	1	1

7.14.c.

$q_1 \sim q_2$

$q_3 \sim q_4$

$q_5 \sim q_7$

$q_6 \sim q_8$

7.14.d.

r \ ab	00	01	10	11	U
r_1	(r_1)	(r_1)	r_3	r_4	0
r_2	r_1	r_1	(r_2)	(r_2)	0
r_3	(r_3)	r_1	(r_3)	r_2	1
r_4	r_3	(r_4)	r_3	(r_4)	1

7.14.e. Codering:

$r_1/r_2/r_3/r_4$: $y_1 y_2$ = 00/01/11/10

met omlegging van

$\delta(r_1,10) = r_4$ in plaats van r_3 ,

$\delta(r_3,01) = r_2$ in plaats van r_1 .

7.14.f. $S_{y_1} = a\overline{y}_2 \qquad R_{y_1} = y_2 b \qquad U = y_1$

$S_{y_2} = \overline{b}y_1 \qquad R_{y_2} = \overline{a}\overline{y}_1$

7.15.a.

q \ abc	000	001 (45)	010 (stop)	100 (33)	U_1 U_2
q_1	$\textcircled{q_1}$	q_3	q_2	q_4	0 0
q_2	q_1	–	$\textcircled{q_2}$	–	0 0
q_3	q_5	$\textcircled{q_3}$	–	–	0 1
q_4	q_6	–	–	$\textcircled{q_4}$	1 0
q_5	$\textcircled{q_5}$	q_3	q_2	q_7	0 1
q_6	$\textcircled{q_6}$	q_8	q_2	q_4	1 0
q_7	q_5	–	–	$\textcircled{q_7}$	0 1
q_8	q_6	$\textcircled{q_8}$	–	–	1 0

$\{q_1,q_2\} \rightarrow r_1$

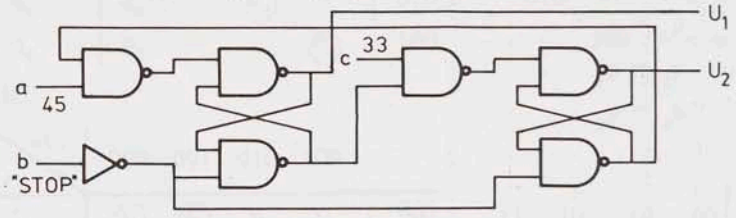
$\{q_3,q_5,q_7\} \rightarrow r_2$

$\{q_4,q_6,q_8\} \rightarrow r_3$

7.15.b.

r \ abc	000	001	010	100	U_1 U_2	
r_1	$\textcircled{r_1}$	r_2	$\textcircled{r_1}$	r_3	0 0	stop.
r_2	$\textcircled{r_2}$	$\textcircled{r_2}$	r_1	$\textcircled{r_2}$	0 1	45 toeren/minuut.
r_3	$\textcircled{r_3}$	$\textcircled{r_3}$	r_1	$\textcircled{r_3}$	1 0	33 toeren/minuut.

7.15.d. Toestandscodering = uitgangscodering.



Hoofdstuk 8

8.1. Bij trekkers bepalen de signalen S en R direct de nieuwe stand van het geheugenelement. Daardoor geeft de SR = 11 → SR = 00 overgang problemen.
Bij flip-flops bepaalt de klokpuls wanneer de informatie wordt ingelezen. S en R worden op dat moment stabiel verondersteld. In principe zijn flip-flops uit de S-R groep bruikbare geheugenelementen.

8.2. Juist. Op de J-ingang zit reeds $\overline{Q}$ en heeft Q geen invloed. Idem op de K-ingang met betrekking tot Q.

8.3. $Z^{n+1} = [\bar{A}B + A\bar{B}\bar{Z} + \bar{A}\bar{C}Z + B\bar{C}Z + AC\bar{Z} + BC\bar{Z}]^n$.

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

$T = AC + A\bar{B} + \bar{B}CZ + \bar{A}B\bar{Z}$.

$J = A\bar{B} + \bar{A}B + AC$, $K = A\bar{B} + \bar{B}C + AC$.

8.4. $[xyz]^n \rightarrow [xyz]^{n+1}$: 000 → 010, 001 → 110, 010 → 001, 011 → 001,
100 → 000, 101 → 100, 110 → 001, 111 → 001.

8.5. $D = \overline{Q}$; $J = K = 1$.

8.6.a. "D flip-flop" resp. "tweedeler met overheersende set".

8.6.b. $I_1 I_2 = 00/-1$ of $1-/0-/1-$.

8.6.c. $I_1 = \overline{K}Q$, $I_2 = J$.

8.6.d. $J = I_1 + I_2$, $K = \overline{I}_1$.

8.7.a. $Q^{n+1} = [I_1 (I_2 + \overline{Q})]^n$.

8.7.b. $J = I_1$, $K = \overline{I}_1 + \overline{I}_2$ en wel via een tabel met de volgende indeling:

$I_1 I_2 Q^n$: Q^{n+1}	J	K
0 0 0 → 0	0	–
0 0 1 → 0	–	1
enz.		

Ook het omgekeerde is mogelijk, nl. met een I_1–I_2 flip-flop een J-K flip-flop maken via $I_1 = J + Q$, $I_2 = \overline{K}$. Ga dit na.

8.8.a.

$A[Y_1 Y_2]^n$			$[Y_1 Y_2]^{n+1}$		U	J_1	K_1	J_2	K_2
0	0	0	0	0	0	0	–	0	–
0	0	1	0	0	1	0	–	–	1
0	1	0	0	0	0	–	1	0	–
0	1	1	0	1	1	–	1	–	0
1	0	0	1	0	0	1	–	0	–
1	0	1	1	0	1	1	–	–	1
1	1	0	1	1	0	–	0	1	–
1	1	1	1	1	1	–	0	–	0

8.8.b. $J_1 = A$, $K_1 = \overline{A}$, $J_2 = Y_1 A$, $K_2 = \overline{Y}_1$.

8.9.a. De schakeling geeft een signaal U = 1 af dan en slechts dan als het ingangssignaal A gedurende tenminste drie klokpulsperioden de waarde 1 bezit.

8.9.b. $J_1 = A\overline{Y}_2$, $K_1 = \overline{A} + Y_2$, $J_2 = AY_1$, $K_2 = \overline{A}$.

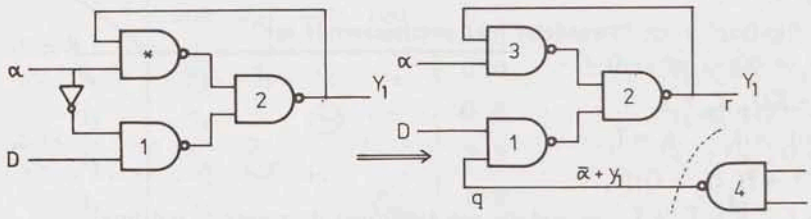
8.10.a. $Y_1 = \bar{a}D + ay_1 \rightarrow S_{y_1} = \bar{a}D \quad R_{y_1} = \bar{a}\overline{D}$

$Y_2 = ay_1 + \bar{a}y_2 \rightarrow S_{y_2} = ay_1 \quad R_{y_2} = a\overline{y}_1$

8.10.b. Uit de formules voor S_{y_2} en R_{y_2} volgt het achterste deel van fig. 8.24. Op het punt r moet dan Y_1 aangeboden worden.
Deze maken we als volgt:

$$Y_1 = \alpha y_1 + \bar{\alpha} D = \alpha y_1 + D(\bar{\alpha} + y_1) \qquad \text{(teken Karnaughdiagram).}$$

Het signaal αy_1 wordt reeds gemaakt in NAND 3 en kost dus geen extra poort.
Idem het signaal $\bar{\alpha} + y_1$ in NAND 4.
M.a.w.: voor de eerste sectie volgt:



Zie verder de tekst.

8.11. In opgave 6.11. is een schakeling gespecificeerd die de actieve flank van de klokpuls verschuift.
Een (reeds gereduceerde) level mode specificatie voor de gehele schakeling geeft de volgende tabel:

r \ tα	00	01	10	11	U
r_1	(1)	r_2	(1)	r_4	0
r_2	r_3	(2)	r_1	(2)	1
r_3	(3)	r_4	(3)	r_2	1
r_4	r_1	(4)	r_3	(4)	0

Bij codering
$r_1/r_2/r_3/r_4$: $y_1 y_2$ = 00/01/11/10

$$S_{y_1} = t\alpha\bar{y}_2 + \bar{t}\bar{\alpha}y_2 ,$$

$$S_{y_2} = t\bar{\alpha}y_1 + \bar{t}\alpha\bar{y}_1 ,$$

$$R_{y_1} = t\alpha y_2 + \bar{t}\bar{\alpha}\bar{y}_2 ,$$

$$R_{y_2} = t\bar{\alpha}\bar{y}_1 + \bar{t}\alpha y_1 ,$$

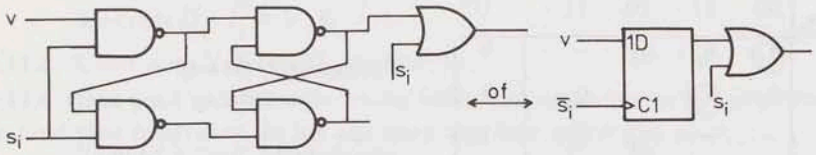
$$U = y_2 .$$

8.12.a.

q \ sv	00	01	10	11	00	01	10	11
q_1	(1)	q_2	q_3	–	0	0	–	–
q_2	q_1	(2)	–	q_4	0	0	–	–
q_3	q_1	–	(3)	q_4	–	–	1	1
q_4	–	q_5	q_3	(4)	–	1	1	1
q_5	q_6	(5)	–	–	1	1	–	–
q_6	(6)	q_5	–	–	1	1	–	–

Na reductie twee toestanden.

8.12.b.



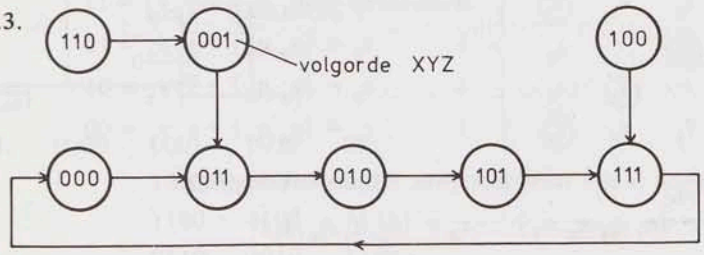
8.13. De schakeling voldoet niet. Ga o.a. na wat er gebeurt als B het eerst een $0 \rightarrow 1$ overgang laat zien!

Hoofdstuk 9

9.1. Bij een asynchrone down-counter geschiedt de laatste stap synchroon met de klokpuls. Namelijk: de laagstwaardige sectie gaat dan van $1 \rightarrow 0$, maar deze sectie staat onder directe besturing van de klok- of telpuls.
Bij een up-counter, die asynchroon is doorgekoppeld, kan dit i.h.a. niet, tenzij de telfrequentie zo laag is dat het transport door de secties heen geheel is verwerkt vóór de volgende puls.

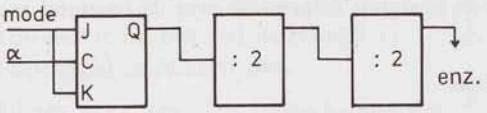
9.2. $J_x = \overline{Y}$, $K_x = Y$, $J_y = Z$, $K_y = \overline{Z}$, $J_z = X$, $K_z = \overline{X}$.

9.3.



9.4. J-K flip-flops: $J_y = K_y = Z$; $J_z = \overline{Z}$, $K_z = 1$.

9.5.



mode = 0 : onthouden,
mode = 1 : tellen.

9.6. Niet verwisselbaar. Vergelijk o.a. de invloed op de transportuitgang. Zie verder de tekst.

9.7. Men kan dit doen door in de blokkeertoestand de klokpulsingang 1 te houden, d.w.z. hoog bij positieve logica.
Schakeling: vervang in fig. 9.14 alle AND's door OR's en bied $\overline{E}_1$, $\overline{E}_2$ en $\overline{W}_i$ en $\overline{Z}_i$ aan voor i = 1,2, . . . ,n.

9.8.a. Afspraak: U = 0 : count UP.
U = 1 : count DOWN.

q \ UQ	00	01	10	11	U*	
q_1	(1)	q_2	q_3	–	0	"count up", U* = Q.
q_2	q_1	(2)	–	q_4	1	
q_3	–	–	(3)	q_6	1	omschakelen "up" → "down".
q_4	–	–	q_5	(4)	1	
q_5	q_7	–	(5)	q_6	1	"count down", U* = $\overline{Q}$.
q_6	–	q_8	q_5	(6)	0	
q_7	(7)	q_2	–	–	1	omschakelen "down" → "up".
q_8	q_1	(8)	–	–	1	

Het is zeer illustratief enkele tijddiagrammen te tekenen!

9.8.b.

r \ UQ	00	01	10	11	U*	
r_1	(r_1)	r_2	r_3	(r_1)	0	$r_1 = \{q_1, q_6\} \rightarrow y_1 y_2 = 11$
r_2	r_1	(r_2)	–	r_4	1	$r_2 = \{q_2, q_8\} \rightarrow y_1 y_2 = 10$
r_3	r_4	–	(r_3)	r_1	1	$r_3 = \{q_3, q_5\} \rightarrow y_1 y_2 = 01$
r_4	(r_4)	r_2	r_3	(r_4)	1	$r_4 = \{q_4, q_7\} \rightarrow y_1 y_2 = 00$

9.8.c. Ja, in dat geval is een tabel met twee toestanden mogelijk:

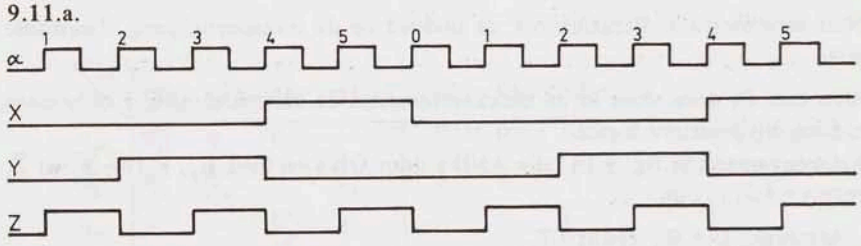
$$r_1 = \{q_1, q_2, q_4, q_8\} \qquad r_2 = \{q_3, q_5, q_6, q_7\}.$$

9.9.a. Direct na het register. Ten eerste treffen we daar "schone" uitgangen aan, d.w.z. spike-vrij. Ten tweede is na de klokpuls informatie over de toestand sneller beschikbaar.

9.9.b. In het algemeen wat langzamer.

9.10. Het gevraagde kan worden aangetoond door enkele tijddiagrammen te tekenen en hierin alle gevallen na te gaan.

9.11.a.



9.11.b. Z: uitsluitend a Y: a of $\overline{Q}_z$ X: a of $\overline{Q}_z$

Een signaal is als klokpuls bruikbaar wanneer op alle plaatsen waar de betreffende flip-flop moet veranderen er 0 → 1 overgangen optreden.

9.11.c. $J_z = K_z = 1$

Y: klokpuls a : $J_y = \overline{X}Z$, $K_y = Z$,

klokpuls $\overline{Q}_z$: $J_y = K_y = \overline{X}$,

X: klokpuls a : $J_x = YZ$, $K_x = Z$,
klokpuls $\overline{Q}_z$: $J_x = Y$, $K_x = 1$.

9.11.d. X met a en Y/Z met $\overline{Q}_z$ als klokpuls.

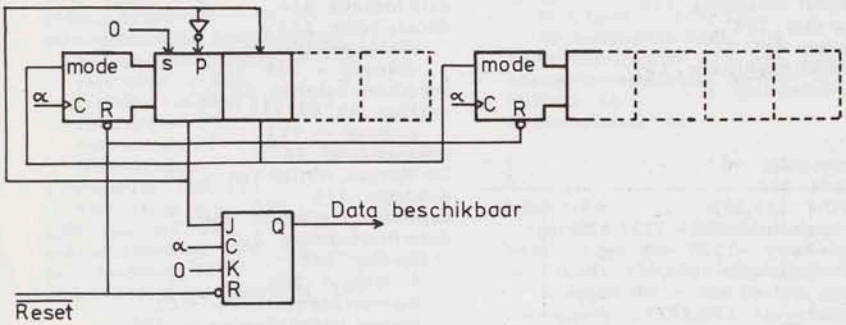
9.11.e. Geen goed gedefinieerde timing basis, met name niet m.b.t. referentie setup en hold time intervallen. In het algemeen zijn deze tellers ook trager.

Hoofdstuk 10

10.1. Zie fig. 10.4.

10.2. Alle lengtes zijn mogelijk. Zie de wegen in fig. 10.6.

10.3.



10.4.

0000	0001	0010	0111
	1000	1001	1011
	1100	0100	1101
	0110	1010	1110
	0011	0101	1111

10.5. Tot elke m.l.r. behoort de inhoud $11 \cdots 11$. Deze gaat bij een even aantal terugkoppelingen in zichzelf over.

10.6. Bij een m.l.r. van $2^k - 1$ lang komen alle $2^k - 1$ registerinhouden voor (00 . . . 00 niet). Immers, zou dit niet zo zijn, dan is de reeks eerder dan in $2^k - 1$ stappen cyclisch. Het te bewijzene volgt nu direct uit een beschouwing over het aantal enen en nullen in de verschillende inhouden.

10.7. De schakeling is een schuifregister van twee secties, waarbij het ingangssignaal a aangeboden wordt aan de eerste sectie en U de uitgang van de tweede sectie is. De schakeling vertraagt a dus twee klokpulsperioden.

10.8.a. Er kunnen spikes optreden in de uitgangen en wel als a van $0 \rightarrow 1$ gaat en Q van de betreffende sectie van $1 \rightarrow 0$.

10.8.b. Bied $\overline{a}$ aan het register aan.

REGISTER

t

u

v

w

z

Deze studiehandleiding wordt ten behoeve van de Vereniging voor Studie- en Studentenbelangen te Delft (VSSD) uitgegeven door de Delftse Uitgevers Maatschappij.

De VSSD is een vereniging van studenten aan de Technische Hogeschool Delft, die zich ten doel stelt de belangen van de studenten te behartigen.

Deze belangenbehartiging heeft vele, overigens samenhangende, kanten. De verdediging van de kwaliteit van het onderwijs, bezinning op de beroepspraktijk en het bevorderen van de toegankelijkheid van het wetenschappelijk onderwijs voor alle lagen van de bevolking zijn de hoofdzaken van wat de "ideële" kant van de belangenbehartiging genoemd zou kunnen worden.

De "materiële" kant betreft de strijd voor een aanvaardbaar inkomen voor de student en voor goede leefomstandigheden (huisvesting, voedsel), goedkoop studiemateriaal e.d.

Bij het verzorgen en doen uitgeven van studiehandleidingen zoals deze zijn de beide aspecten vertegenwoordigd: de beschikbaarheid van goede en handzame dictaten vergroot de kwaliteit van het onderwijs en verbetert de studieresultaten, anderzijds worden de boekwerkjes tegen kostprijs (dus zo goedkoop mogelijk; het zetwerk geschiedt in eigen beheer) aan de leden van de VSSD beschikbaar gesteld. Daarbij kunnen ook anderen tegen een, zij het (van wege de verkoopkosten) hogere, doch zeer acceptabele prijs deze werkjes kopen.

ISBN 90-6562-007-9