Hoewel iedereen tegenwoordig met computers "stoeit", is men vaak slecht op de hoogte van de opbouw, werking en organisatie van het computergeheugen. Toch is de organisatie en de daarmee samenhangende adresdekodering heel belangrijk voor het daarbij behorende computersysteem. In dit artikel gaan we deze punten eens wat nader bekijken.

adresdekodering

de organisatie van een geheugenbereik

Om de organisatie van een computergeheugen te bekijken gaan we dit geheugen eens vergelijken met de opzet van een grote bibliotheek. De informatie, bij de computer dus de data, staat hier in de vorm van boeken met inhoud. Wat die inhoud is laten we verder buiten beschouwing. Wat ons interesseert in die bibliotheek is het systeem dat ze gebruiken om alles te rangschikken: groepen, kategorieën, onderverdelingen, enzovoorts.

In volgorde van belangrijkheid

Stel je een katalogus voor waarin enkele tienduizenden titels van boeken staan over diverse onderwerpen. Onze bibliotheek heeft bijvoorbeeld ook elektronica-boeken. Deze zijn allemaal verzameld onder de referentieletter "E". Daarbij is een onderverdeling gemaakt. De boeken over digitale elektronica worden aangeduid met "ED" en de boeken over analoge elektronica met "EA". De eerste letter (E) is het belangrijkst (meest signifikant) en de tweede letter (A of D) is het minst belangrijk (minst signifikant). Dat verschil is waarschijnlijk duidelijk: de letter "E" heeft betrekking op alle elektronica-boeken in onze denkbeeldige bibliotheek, terwijl de letters "D" en "A" maar een beperkte groep van alle elektronica-boeken omvatten. We zouden zo nog een tijd kunnen doorgaan. Zo zou een volgende letter bijvoorbeeld kunnen aangeven welke boeken in het Nederlands zijn geschreven en welke boeken in een buitenlandse taal. Een boek met de aanduiding "EDN" handelt dus over digitale elektronica in de Nederlandse taal, terwijl een boek met het opschrift "EAE" over analoge elektronica gaat in het Engels. Die laatste letter (Nederlands of niet) is minder belangrijk dan de voorgaande letter (digitaal of analoog). In de kategorie "elektronica-boeken" is de indeling tussen "digitale" en "analoge" boeken dus belangrijker dan de taal waarin de boeken geschreven zijn.
Tenslotte nog even een ander voorbeeld om het verschil in belangrijkheid te laten zien. Het voorbeeld ligt eigenlijk voor de hand, maar is toch heel illustratief. Het gaat om de prijzen van artikelen. Je zult bijna altijd bij een versterker zien staan f 999,— in plaats van f 1000,—. Het verschil tussen die twee prijzen is verwaarloosbaar klein, maar het gaat om de indruk die de getallen op ons maken. 999 *lijkt* minder dan 1000 omdat het laatste getal één meest signifikant cijfer méér heeft dan het eerste.

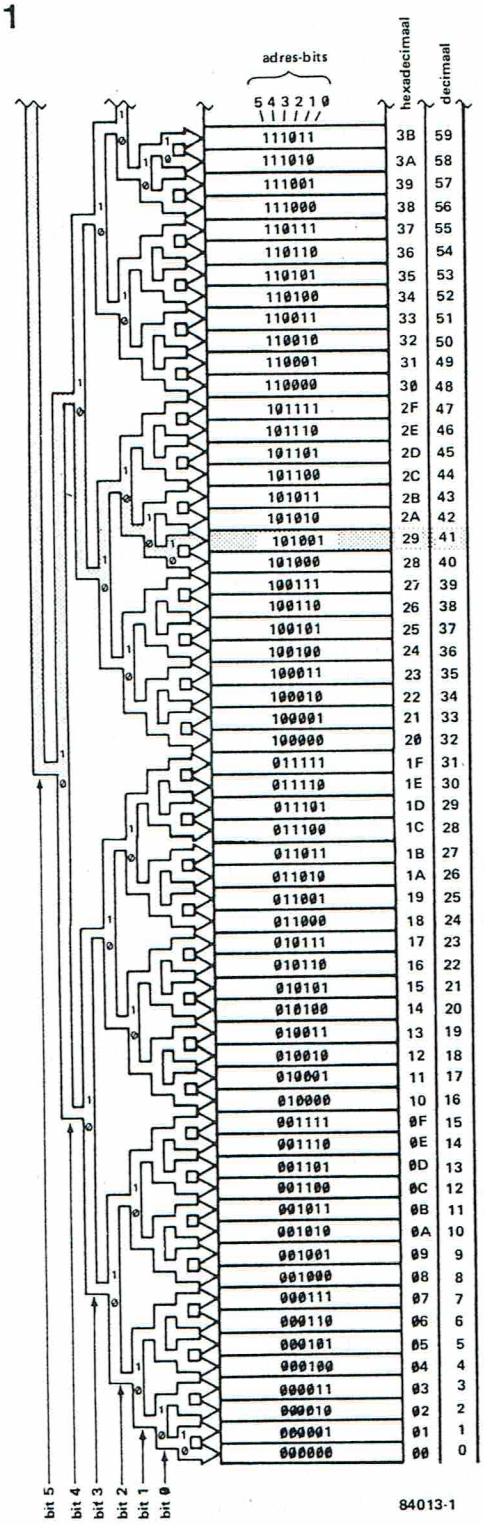
Geheugen en adressering

Na deze inleiding keren we weer terug naar het geheugen van onze computer. Dit kan worden voorgesteld als een stapel doosjes of hokjes (ook wel cellen genoemd), die bij onze normale acht-bit-computers elk weer zijn onderverdeeld in acht delen. Deze delen, bits, zijn niet apart toegankelijk. De bits worden altijd in groepen van acht stuks gebruikt en zo'n groep wordt een byte genoemd. De logische nivo's van die acht bits vormen de gegevens. De gegevens worden in het computersysteem getransporteerd via de databus, acht lijnen die genummerd zijn van D7 . . . DØ. Elke lijn transporteert dus 1 bit. Om die gegevens uit het geheugen te kunnen halen of in het geheugen op te slaan beschikt de processor over een adresbus die uit 16 lijnen bestaat: A15 . . . AØ. Met deze lijnen kunnen de verschillende hokjes van het geheugen "aangesproken" worden. De organisatie van de adreslijnen kan worden vergeleken met die van een bibliotheek.
In figuur 1 zijn een aantal kombinaties gegeven van de zes meest signifikante adresbits (A5 . . . AØ). Hierbij zijn "gangetjes" getekend zoals je zou moeten lopen om in een bibliotheek een bepaald boek te vinden. Aan het einde van elk gangetje kun je links- of rechtsaf slaan, oftewel in de tekening omhoog of omlaag. De beslissing of je omhoog of omlaag gaat wordt aangegeven met een logisch nivo (Ø naar beneden, 1 naar boven). Een andere mogelijkheid is er niet.
Hoe hoger de belangrijkheid van een bit, hoe groter het aantal bitkombinaties dat hier onder valt. Als bijvoorbeeld bit 5 en 4 in figuur 1 beide "1" zijn en bit 3 "0", dan omvat deze kombinatie de waarden ØØ . . . . Ø7. Als bit 3 "1"is, dan loopt de kombinatie van Ø8 tot ØF. Als bit 4 echter "1" is loopt de zone van 1Ø tot 17 als bit 3 nul is en van 18 tot 1F als bit 3 één is. Als het logische nivo van bit 3 niet gedefinieerd is en bit 4 en 5 beide nul zijn, dan loopt de zone van ØØ . . . ØF. Doordat bit 3 niet gedefinieerd is kan adres ØØ dezelfde adressering hebben als adres Ø8. Hetzelfde geldt voor Ø1 en Ø9, Ø2 en ØA, enzovoorts. Men spreekt in zo'n geval van dubbele adressering. Bij computers komt zo'n dubbele adressering vaker voor.

$2^{16} = 65536$

In figuur 2 is het verband gegeven tussen de hoogste adreslijnen A15 . . . A10 en de geheugenruimte die daarmee geadresseerd kan worden. De aangegeven geheugenruimte is altijd in "K", waarbij 1 K overeenkomt met 1024 bytes (niet 1000). Dat getal 1024

1



	ADRESSES DEC.	ADRESSES HEX	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
	0 ↓ 15	0000 ↓ 000F	0000	0000	0000	0000 ↓↓↓↓ 1111
	16 ↓ 31	0010 ↓ 001F	0000	0000	0001	0000 ↓↓↓↓ 1111
	32 ↓ 63	0020 ↓ 003F	0000	0000	0 001↓ 1	0000 ↓↓↓↓ 1111
	64 ↓ 127	0040 ↓ 007F	0000	0000	00 01↓↓ 11	0000 ↓↓↓↓ 1111
	128 ↓ 255	0080 ↓ 00FF	0000	0000	000 1↓↓↓ 111	0000 ↓↓↓↓ 1111
	256 ↓ 511	0100 ↓ 01FF	0000	0001	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
	512 ↓ 1023	0200 ↓ 03FF	0000	0 001↓ 1	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
1K	1024 ↓ 2047	0400 ↓ 07FF	0000	00 01↓↓ 11	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
2K	2048 ↓ 4095	0800 ↓ 0FFF	0000	000 1↓↓↓ 111	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
4K	4096 ↓ 8191	1000 ↓ 1FFF	0001	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
8K	8192 ↓ 16383	2000 ↓ 3FFF	0 001↓ 1	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
16K	16384 ↓ 32767	4000 ↓ 7FFF	00 01↓↓ 11	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
32K	32768 ↓ 65535	8000 ↓ FFFF	000 1↓↓↓ 111	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111	0000 ↓↓↓↓ 1111
64K						

Tabel 1. Met 16 adreslijnen kan een geheugenbereik van 65536 plaatsen worden bestreken. Deze tabel toont de adreslijnen die nodig zijn voor het adresseren van een bepaald bereik.

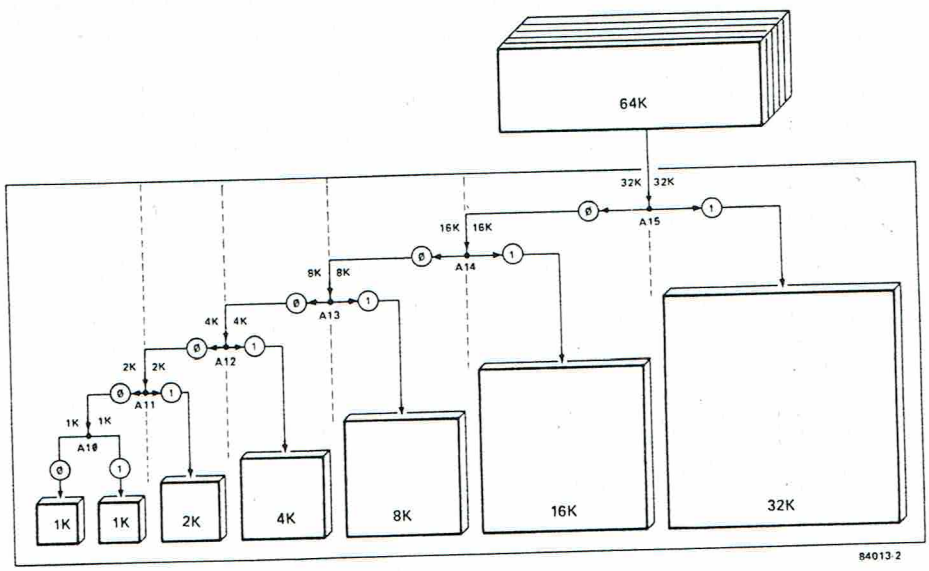
Figuur 1. Deze "binaire boom" voor de vijf hoogste adresbits laat zien hoe een adresbereik is ingedeeld.

wordt bepaald door de adresruimte die met de lijnen A9 . . . AØ kan worden bestreken: dat is namelijk 2^{10} = 1024. Bij geheugens is 1 K dus altijd 1024.
Terug naar figuur 2. Adreslijn 15 kan nul of één zijn, zodat deze lijn de totale beschikbare geheugenruimte (16 adreslijnen geven 2^{16} = 65536 mogelijkheden) verdeelt in twee gelijke blokken van 32768 bytes. In elk van deze twee blokken zorgt lijn A14 voor een verdere verdeling in twee helften van ieder 16384 bytes . . . en zo door, zodat we tenslotte met lijn A10 twee blokken van 1024 bytes kunnen kiezen. Ook hier kunnen we zien wat er gebeurt bij een onvolledige adressering. Als het nivo van A15 bijvoorbeeld niet gedefinieerd is, dan komt adres Ø overeen met adres 32768, adres 1 met adres 32769, enzovoorts. Vergeet trouwens niet bij al deze adrestoestanden dat men altijd begint te tellen bij Ø, zowel in het binaire, het decimale als het hexadecimale stelsel.
In tabel 1 is aangegeven welke adressen veranderen in welk adresbereik. Voor byte Ø . . . 15 zijn dat de adreslijnen A3 . . . AØ en steeds als het aantal adressen verdubbelt komt er een adreslijn bij: voor de plaatsen 16 . . . 31 zijn A4 . . . AØ nodig, voor 32 . . . 63 A5 . . . AØ, enzovoorts.

Het maken van adresseringssignalen

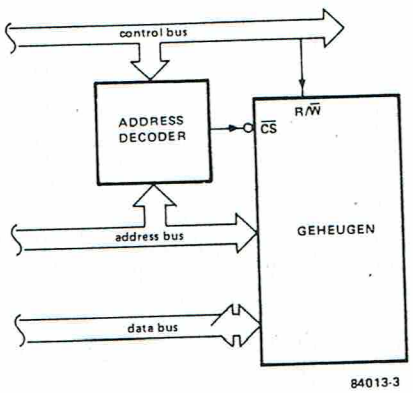
Tot nu toe hebben we alleen maar gekeken naar de adressering vanuit een algemene hoek. Als we nou gaan kijken naar de IC's die in werkelijkheid moeten worden geadresseerd, dan blijkt dat de meeste van die IC's geen 16 adres-ingangen bezitten, maar een veel kleiner aantal dat afhangt van de geheugenkapaciteit van het betreffende IC. Voor een IC dat 4 Kbyte bevat (bijvoorbeeld een 2732) zijn volgens figuur 2 12 adreslijnen nodig (A11 . . . AØ). De adressering van elk van de 4096 bytes in het IC wordt verzorgd door een interne adresdekoder. Voor een 2 Kbyte-geheugen (bijvoorbeeld een 6116-RAM) geeft figuur 2 een benodigd aantal adreslijnen van 11 stuks, A1Ø . . . . AØ. De in het IC aanwezige adresdekoder kan hiermee kiezen uit 2048 geheugenplaatsen. Gewoonlijk bedoelt men met adresdekodering niet de in een IC aanwezige adresdekoder, maar de wijze waarop een geheugenblok in het adresseerbare geheugenbereik van de processor wordt opgenomen. In de volgende voorbeelden hebben we

2



Figuur 2. Hier is te zien hoe groot het geheugenblok is dat met een bepaalde adreslijn kan worden geadresseerd. A15 is goed voor een blok van 32 K, A14 voor 16 K, enzovoorts.

3



Figuur 3. Voor het adresseren van het geheugen zijn buiten de adreslijnen en datalijnen ook nog verschillende kontrolesignalen nodig voor het verzorgen van de juiste chronologische volgorde bij lees- en schrijfoperaties.

ons beperkt tot de 6502 en de Z 80, twee processoren die beide 16 adreslijnen hebben en daarmee dus maximaal 64 Kbyte geheugen kunnen adresseren.
Elke geïntegreerde geheugenschakeling bevat naast de reeds genoemde adres-ingangen één of meerdere select- of enable-ingangen. Meestal reageren deze op een logische nul (te zien aan het inverteringsstreepje boven de aanduiding bij de aansluiting). Pas als dit enable-signaal is gegeven werkt de interne adresdekoder van het IC en wordt de data op de uitgangen gezet of data ingelezen. Zo'n enable-signaal wordt gewoonlijk gemaakt uit een kombinatie van hogere (niet op het IC aangesloten) adreslijnen en kontrolesignalen van de processor (zie figuur 3).
De kontrolesignalen zijn bij elk systeem weer anders. Bij de 6502 zijn dat:

- Het kloksignaal Φ2 dat er voor zorgt dat de lees- en schrijf-operaties alleen kunnen plaatsvinden tijdens de tweede helft van elke klokperiode van de processor.
- Het R/$\overline{W}$-signaal dat de lees (read)- en schrijf ($\overline{write}$)-operaties uit elkaar houdt.

Bij de Z 80 zijn dat:

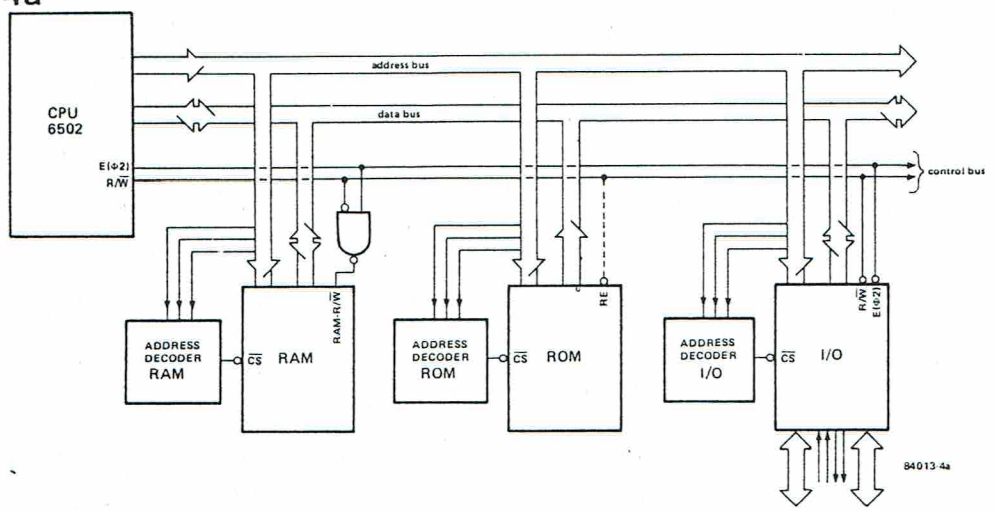
- $\overline{WE}$ en $\overline{RE}$ voor het onderscheiden van lezen ($\overline{read\ enable}$) en schrijven ($\overline{write\ enable}$).
- $\overline{MREQ}$ en $\overline{IOREQ}$ voor het onderscheid tussen operaties die betrekking hebben op het geheugen en operaties die betrekking hebben op de in- en uitgangsmodules waarvoor de Z 80 aparte instrukties heeft.

Deze verschillen zijn te zien in figuur 4a en 4b. De enable-signalen die van de hoogste adreslijnen en de kontrolesignalen worden afgeleid, worden hier allemaal $\overline{CS}$ ($\overline{chip\ select}$) genoemd. Voor het gemak nemen we tevens aan dat al deze signalen aktief laag zijn. De diverse fabrikanten gebruiken vaak andere namen voor de enable-signalen en bovendien kunnen ze in de praktijk ook aktief hoog zijn.
Voordat we gaan kijken naar schakelingen voor het "maken" van enable-signalen willen we eerst nog even wijzen op het belang van het hexadecimale stelsel. Er zijn in totaal 16 adreslijnen die worden onderverdeeld in vier groepen van elk vier lijnen. Elke groep korrespondeert met een hexadecimaal cijfer (Ø . . . F, oftewel 0 . . . 15 in het decimale stelsel). Zo geeft de 4 bij adres 4A2F de logische nivo's voor de adreslijnen A15, A14, A13 en A12 (ØlØØ), de A geeft de nivo's voor de lijnen A11, A10, A9 en A8 (1Ø1Ø), de 2 voor A7, A6, A5 en A4 (ØØ1Ø) en de F voor A3, A2, A1 en AØ (1111). Op die manier kan men gemakkelijk de logische nivo's van de 16 adreslijnen afleiden uit een adres dat in hexadecimale vorm is gegeven.

Vaste logische schakelingen

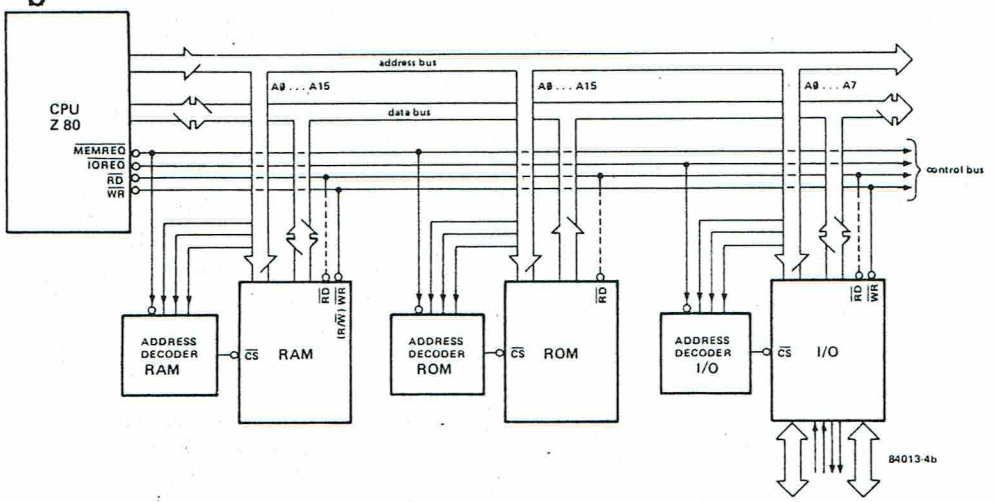
Nu gaan we de schakeling voor de adresdekodering bekijken. Meestal bestaat zo'n schakeling uit enkele digitale poorten of IC's die op de een of andere manier aan elkaar zijn geknoopt. Als voorbeeld nemen we eerst een schakeling die een enable-signaal moet leveren bij de adressen 2ØØØ . . . 2FFF. Met de adreslijnen A11 . . . AØ kunnen 4098 plaatsen worden geadresseerd tussen XØØØ en XFFF. Een kombinatie van de lijnen A15 . . . A12, zoals in figuur 5a is getekend, levert een $\overline{CS}$-signaal dat aktief laag is als de adressen A15 . . . A12 "ØØ1Ø" zijn. Dat is dus het cijfer 2 (voor het bereik 2ØØØ . . . 2FFF). Een ander voorbeeld is te zien in figuur 5b. Hier wordt alleen $\overline{CS}$ gegeven bij

4a



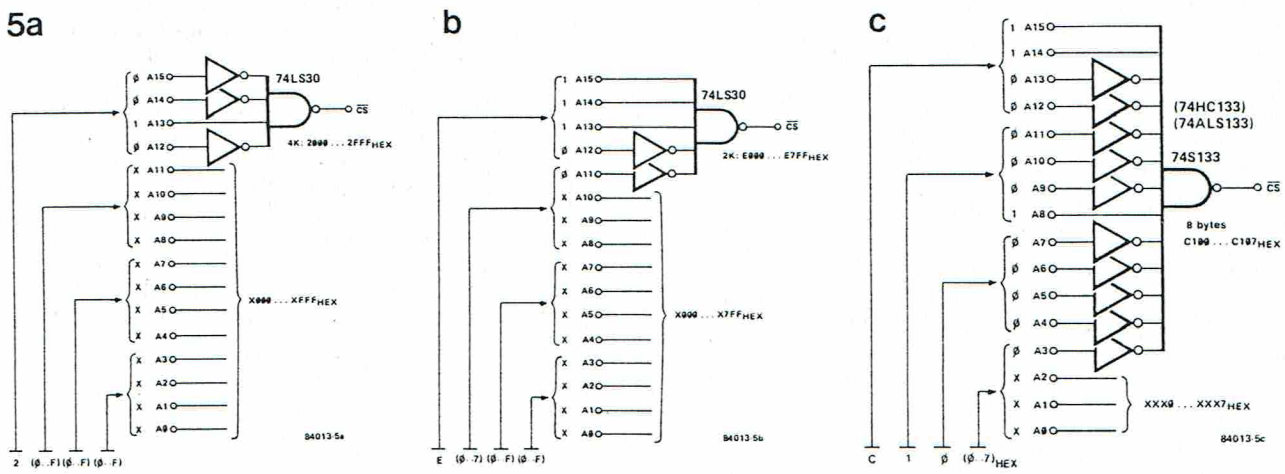
Figuur 4a. De systeemopzet bij de 6502. De kontrolesignalen Φ2 (klok) en R/$\overline{W}$ (lezen/schrijven) worden hier gebruikt om alles in de goede volgorde te laten verlopen.

b



Figuur 4b. De systeemopzet van een Z 80 is praktisch gelijk aan die van een 6502, alleen zijn hier meer kontrolesignalen aanwezig.

5a

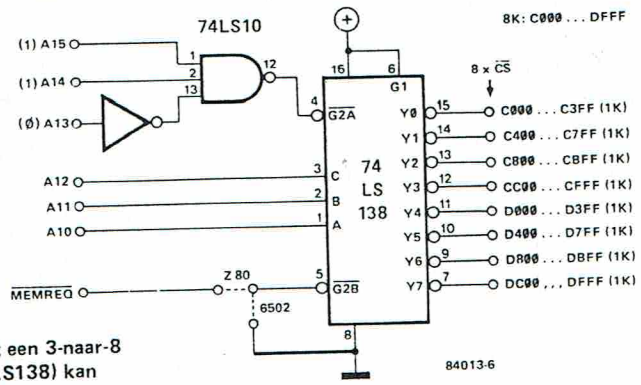


Figuur 5. Enkele voorbeelden van een "vaste" adresdekodering voor respektievelijk 4 K, 2 K en 8 bytes. Hoe kleiner het te adresseren geheugenblok, hoe meer adreslijnen worden gebruikt voor het maken van het $\overline{CS}$-signaal.

de kombinatie "11100" van de adreslijnen A15 . . . A11. Met de lijnen A10 . . . A0 en het $\overline{CS}$-signaal kunnen dus 2048 plaatsen worden geadresseerd in het bereik E000 . . . E7FF. In het voorbeeld van figuur 5c is het te adresseren bereik nog kleiner gekozen. Alleen als het hexadecimale getal C10 op de adreslijnen A15 . . . A3 staat wordt een $\overline{CS}$-signaal gegeven. Met de overgebleven drie adreslijnen kunnen de acht plaatsen tussen C100 en C107 worden geadresseerd.

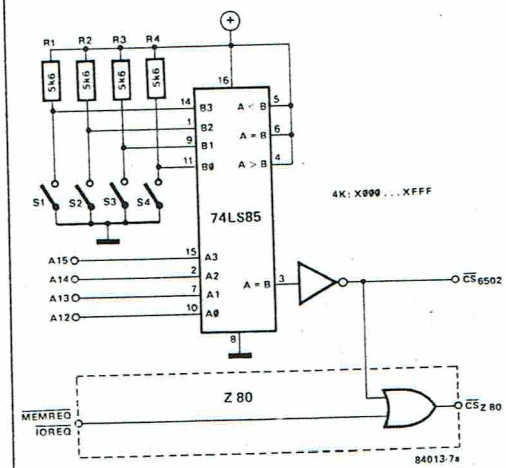
Deze drie voorbeelden laten zien hoe het te adresseren bereik kleiner kan worden gemaakt door meer adreslijnen te gebruiken voor het maken van het $\overline{CS}$-signaal. Voor de eenvoud hebben we allerlei signalen die verder nodig zijn voor de goede gang van (adresdekoderings)zaken maar weggelaten. In figuur 6 staat een schakeling die meerdere $\overline{CS}$-signalen levert voor diverse bereiken. Hier

6

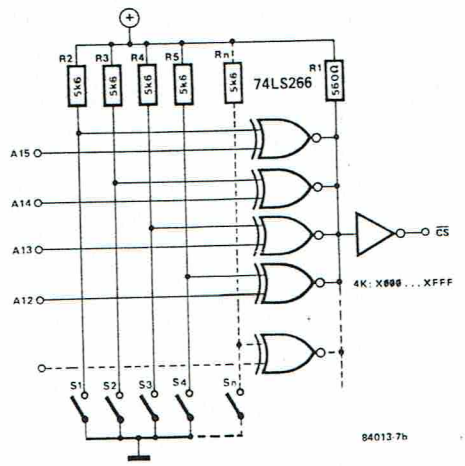


Figuur 6. Met een 3-naar-8 dekoder (74LS138) kan een blok van 8 K heel eenvoudig worden opgedeeld in acht blokken van 1 K.

7a



b



Figuur 7. Bij sommige toepassingen kan het handig zijn als men het adresbereik met schakelaars kan instellen. Dat is mogelijk door gebruik te maken van een komparator-IC of EXNOR-poorten.

is gebruik gemaakt van een 74LS138, een 3-naar-8-dekoder die vaak voor dit doel wordt gebruikt. Dit IC heeft drie ingangen A, B en C en drie enable-ingangen, G1, $\overline{G2A}$ en $\overline{G2B}$. Het signaal voor $\overline{G2A}$ wordt verkregen uit een kombinatie van A15, A14 en A13, zodat op deze wijze het adresbereik al wordt beperkt tot C000 . . . DFFF, een blok van 8 K. Ingang $\overline{G2B}$ is verbonden met de $\overline{MREQ}$-lijn bij een Z 80; bij een 6502 wordt deze ingang aan massa gelegd. De ingangen A, B en C zijn verbonden met A12, A11 en A10, zodat het mogelijk is het 8 K-bereik op te splitsen in acht delen van elk 1 K. De acht $\overline{CS}$-signalen die nu beschikbaar zijn kunnen verder worden gebruikt voor het geheugen, in kombinatie met de kontrolesignalen $\overline{WE}$, $\overline{RD}$ of R/$\overline{W}$.

Universele dekodeerschakelingen

De tot nog toe behandelde dekodeerschakelingen hebben allemaal één nadeel: ze kunnen niet eenvoudig worden veranderd voor een ander adresbereik. Er zijn echter ook dekodeerschakelingen die vrij universeel van opzet zijn. In figuur 7 zijn er twee getekend. Het schema van figuur 7a maakt gebruik van een 4 bit komparator. Het binaire woord A3 . . . AØ wordt geleverd door de adreslijnen A15 . . . A12. Dit woord wordt vergeleken met de logische nivo's op de ingangen B3 . . . BØ. Deze nivo's kunnen worden ingesteld met de schakelaars. Als een schakelaar gesloten is, dan staat op de bijbehorende ingang een logische nul. Is de schakelaar open, dan zorgt de weerstand aan de ingang voor een logische één. Als de data op A3 . . . AØ gelijk is aan de data op B3 . . . BØ wordt de uitgang A=B (pen 3) logisch één. Dit signaal wordt geïnverteerd en kan dan worden gebruikt als $\overline{CS}$ voor een blok van 4 K (XØØØ . . . XFFF, waarbij X het hexadecimale getal is dat men heeft ingesteld met de vier schakelaars). Een zelfde soort programmeerbare adresdekoder kan worden gemaakt met EXNOR-poorten, zoals in figuur 7b is getekend. De open-kollektor-uitgangen van de 74LS266-poorten zijn alleen allemaal "1" als de twee ingangen van elke poort hetzelfde logische nivo hebben. Elke EXNOR vergelijkt een adreslijn met een nivo dat met een schakelaar kan worden ingesteld. Zulke schakelingen hebben het voordeel dat ze erg flexibel zijn; men hoeft geen soldeerbout te pakken of de dekodeerschakeling te wijzigen om een geheugenblok in een ander bereik te leggen. Bovendien kan de schakeling van figuur 7b heel eenvoudig worden uitgebreid: voor elke adreslijn is slechts één enkele EXNOR nodig. We hopen dat men een indruk gekregen heeft van de wijze waarop een geheugenbereik is ingedeeld, hoe de dekodeersignalen voor een bepaald bereik moeten worden gemaakt en waarvoor deze dienen. Natuurlijk valt er over dit onderwerp nog veel meer te vertellen, maar dat zou voor één artikel te veel van het goede worden.