Når 38 er det samme som 100110

Lidt om vedtagelser, titalsystem og totalsystem – og om den meget lille tabel.

Når vi skriver »38«, mener vi »otteogtredive« som repræsentation for $3 \times 10 + 8 \times 1$ og ikke »treotte« som repræsentation for $3 + 8 = 11$.

Vi er så vant til, at det bageste ciffer i et tal repræsenterer enerne, det næstbageste tierne, det tredie (bagfra) hundrederne, det fjerde tusinderne og så videre, at vi overhovedet ikke skænker denne vedtagelse en tanke, når vi ser et tal som f. eks. 38.

Det er, som om denne tallenes positionsstatus hører urnaturen til, men som vi allerede har fortalt, er dette slet ikke tilfældet. Systemet er opfundet af mennesker, formentlig fordi også de første mennesker, der talte, havde ti fingre lige ved hånden.

Vi har tidligere set, at andre mennesker gav deres talsymboler en anden positionsstatus end ti, som er bekvemt for os, fordi vi skriver ti som »10« og ikke med et specielt symbol som f. eks. romernes X.

Vi er så vant til at arbejde med vore ti forskellige talsymboler – 0, 1, 2, 3, 4, 5, 6, 7, 8 og 9 –, at vi ikke lægger mærke til de ulemper, der kan være forbundet med at *skulle* bruge så mange forskellige.

Genkalder vi os det første skoleår og *Den lille tabel,* som skulle kunnes forfra og bagfra samt i søvne, erindrer vi også det besvær, vi havde med at indlære de ét hundrede multiplikationsresultater, tabellen indeholder.

Vi vil nu undersøge, hvad der sker, hvis vi i stedet for ti talsymboler nøjes med to. Vi er jo klar over, at vi må have to forskellige – af den grund, at man simpelthen ikke kan komme videre i en talrække, hvis man kun råder over et.

Hvis vi vælger de to første ciffre – nemlig nul og et – og stiller samme krav til en talrække med dem som grundlag, som vi stiller til vore egne ti – nemlig kravet om, at repræsentationen bliver én højere, når vi bevæger os ét skridt til højre i talrækken, får vi nogle nye værdier frem – men en stærkt forsimplet tabel.

Vi prøver os frem:

I den almindelige talrække, der ser sådan ud:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 .... o.s.v. .... 97, 98, 99, 100, 101, 102 .... o.s.v. ... 109, 110, 111, 112, 113 ... o.s.v. .... kan vi ikke bruge 2, 3, 4, 5, 6, 7, 8, 9, 12, 13, 14, 15, 16 ... o.s.v. .... 97, 98, 99, 102 .... o.s.v. .... 109, 112, 113 .... o.s.v. ...., men vi kan godt bruge resten: 0, 1, 10, 11, 100, 101, 110, 111 .... o.s.v.

Hvis kravet om en repræsentationsfølge skal opretholdes, må vi tillægge hver af de nye symboler samme repræsentation som symbolerne i den oprindelige talrække. Altså må vi vedtage, at

0 = 0
1 = 1
10 = 2
11 = 3
100 = 4
101 = 5
110 = 6
111 = 7

og så videre.

Som det vil ses, er det stadig enere, vi har i bageste ciffer, men det er ikke længere tiere, vi har i næstbageste ciffer, men derimod toere. I tredie position har vi firere i stedet for hundreder – og i fjerde har vi ottere i stedet for tusinder.

Når vi skriver 111 som repræsentation for decimaltallet 7, betyder det $\mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{1} \times 2^0$, idet $2^2 = 4$, $2^1 = 2$ og $2^0 = 1$, og fordi $4 + 2 + 1 = 7$.

Parallellen hertil i decimaltal er jo, at 38 har repræsentationen $\mathbf{3} \times 10^1 + \mathbf{8} \times 10^0 = 3 \times 10 + 8 \times 1 = 30 + 8 = 38$.

Hvis vi fjerner ti-potenserne, opstår tallet 38 på samme måde, som tallet 111 opstår, når vi fjerner to-potenserne i syvtals-eksemplet.

Omsætter vi nu tallet »38« til potenser af to i stedet for potenser af ti, får vi følgende resultat: $38 = \mathbf{1} \times 2^5 + \mathbf{0} \times 2^4 + \mathbf{0} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{0} \times 2^0$. Vi ved nemlig, at 2^5 er det samme som »to ganget med sig selv fem gange« eller 32, at $2^2 = 4$ og at $2^1 = 2$. Og $32 + 4 + 2 = 38$.

I totalsystemet – eller, som det det også kaldes: det *binære talsystem* – er det altså tallet 100110, der repræsenterer decimaltallet 38.

Den lille tabel

×	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

Den meget lille tabel

×	0	1
0	0	0
1	0	1

Sammensætningen 100110, der udelukkende består af nuller og ettaller, repræsenterer altså begrebet »otteogtredive« lige så godt som talsymbolerne »38«, og følger vi den nyligt beskrevne fremgangsmåde igennem nogle nye eksempler, vil vi opdage, at der til ethvert decimaltal svarer et ganske bestemt binært tal, der kan skrives alene ved hjælp af nuller og ettaller.

Kigger vi nu i Den lille tabel, vil vi endvidere opdage, at vi kan nøjes med *Den meget lille tabel,* idet vi ikke længere skal huske 100 multiplikationsresultater, men kun fire, og af dem er de tre oven i købet nul!

Der findes nemlig kun fire måder, man kan gange de to forskel-

Omsætningstabel

Potens:	Multiplikationsprodukt:		Decimalt:		Binært:
2^0 =	1	=	1	=	1
2^1 =	1×2	=	2	=	10
2^2 =	1×2×2	=	4	=	100
2^3 =	1×2×2×2	=	8	=	1000
2^4 =	1×2×2×2×2	=	16	=	10000
2^5 =	1×2×2×2×2×2	=	32	=	100000
2^6 =	1×2×2×2×2×2×2	=	64	=	1000000
2^7 =	1×2×2×2×2×2×2×2	=	128	=	10000000
2^8 =	1×2×2×2×2×2×2×2×2	=	256	=	100000000
2^9 =	1×2×2×2×2×2×2×2×2×2	=	512	=	1000000000
2^{10} =	1×2×2×2×2×2×2×2×2×2×2	=	1024	=	10000000000

lige talsymboler på:

$0 \times 0 = 00$
$1 \times 0 = 00$
$0 \times 1 = 00$
$1 \times 1 = 01$

- og skifter vi gangetegnet ud med et og-tegn, kan vi fastslå, at der også kun findes fire måder, vi kan lægge sammen på:

$0 + 0 = 00$
$1 + 0 = 01$
$0 + 1 = 01$
$1 + 1 = 10$

Man *siger* ikke, at »en plus en er lig med ti«, når man taler om binære tal. Man siger »en plus en er lig med en-nul«. Den binære repræsentation for decimaltallet 7 hedder altså ikke »ethundredeogelleve« men »en-en-en«.

Når vi så omhyggeligt har gjort rede for dette system, skyldes det ikke alene hensynet til dem, der i sin tid havde besvær med Den lille tabel, men den kendsgerning, at hovedparten af de elektroniske regnemaskiner kun kan arbejde med to talsymboler.

Man kunne i og for sig godt vælge et §-tegn som det ene symbol eller et Q-tegn som det andet eller to forskellige bogstaver, f. eks. J for »ja« og N for »nej«. Det gør man vitterligt i nogle tilfælde, som vi senere skal vende tilbage til - men foreløbig vil vi holde os til nuller og ettaller - og udelukkende til addition med nuller og ettaller.

Sådan trækker man fra ved at lægge til

Lidt om, hvordan man kan udføre alle fire regnearter - alene ved addition.

Hvordan kan man bruge en maskine, der kun formår at lægge binære tal sammen to og to, til komplicerede regneopgaver, der også omfatter multiplikation, subtraktion og division?

Ja - det er ikke umiddelbart indlysende, men forståeligt, hvis man har tålmodighed nok - thi de tre andre regnearter *kan udføres som variation af addition.*

Multiplikation er jo kun gentagen addition. Eksempelvis ved vi, at 4×12 er det samme som $12 + 12 + 12 + 12$. Altså kan vi, hvis vi vil undgå at gange med fire, først sige $12 + 12 = 24$. Derefter kan vi sige $24 + 12 = 36$, og til slut, at $36 + 12 = 48$.

Den slags omveje har vi tid nok til, når vi bruger en datamaskine, der kan udføre mange hundrede tusinde af sådanne sammenlægninger i løbet af et enkelt sekund. Der er bygget datamaskiner, der kan klare cirka to millioner additioner af denne art pr. sekund!

Subtraktionsproblemet løser vi ved at *komplementere* det tal, vi skal trække fra et andet.

I titalsystemet komplementerer man et tal ved at trække det fra en række nitaller af samme længde som tallet og ved derefter at lægge 1 til resultatet.

Det lyder måske indviklet, men vi vil prøve med et eksempel:

Opgaven, vi stiller os, går ud på at trække 217 fra 349. Vi kan umiddelbart regne ud, at resultatet er 132, hvis vi bruger den almindelige fremgangsmåde. Men vi kommer til samme resultat ved addition, hvis vi først trækker 217 fra 999 og lægger resultatet - 782 - til 349. Det bliver nemlig 1131, hvoraf vi flytter menten (det første et-tal) hen under resten (131) og lægger den til:

$$\begin{array}{r} 131 \\ +\quad 1 \\ \hline 132 \end{array}$$

Resultatet bliver da, som det skal være.

Nu er det så heldigt, at man i det binære talsystem kan komplementere et tal *uden* at skulle foretage nogen subtraktion - ved simpelthen at gøre alle nuller til ettaller og alle ettaller til nuller, hvorefter en subtraktion indskrænker sig til en addition med påfølgende menteforflytning og -tillæg.

Vi prøver igen med samme tal som før, blot omskrevet til binære tal.

Tallet 349 har vi allerede lært at omskrive til $\mathbf{1} \times 2^8 + \mathbf{0} \times 2^7 + \mathbf{1} \times 2^6 + \mathbf{0} \times 2^5 + \mathbf{1} \times 2^4 + \mathbf{1} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0$ eller til **101011101**, fordi

$$\begin{array}{r} 2^8 = 256 \\ 2^6 = 64 \\ 2^4 = 16 \\ 2^3 = 8 \\ 2^2 = 4 \\ 2^0 = 1 \\ \hline 349 \end{array}$$

På samme måde kan vi se, at decimaltallet 217 er det samme som $\mathbf{1} \times 2^7 + \mathbf{1} \times 2^6 + \mathbf{0} \times 2^5 + \mathbf{1} \times 2^4 + \mathbf{1} \times 2^3 + \mathbf{0} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0$ eller **11011001**, fordi

$$\begin{array}{r} 2^7 = 128 \\ 2^6 = 64 \\ 2^4 = 16 \\ 2^3 = 8 \\ 2^0 = 1 \\ \hline 217 \end{array}$$

Regnestykket ser nu således ud:

$$\begin{array}{r} 101011101 \\ \div\ 011011001 \\ \hline ? \end{array}$$

I stedet for at gennemføre det efter de sædvanlige regneregler, omskriver vi 11011001 til 00100110 (alle nuller er blevet ettaller, alle ettaller er blevet nuller) og lægger det til i stedet for at trække det fra:

$$\begin{array}{r} 11111 \\ 101011101 \\ +\ 000100110 \\ \hline 11000011 \end{array}$$

– idet vi husker, at $1 + 1 = 10$, svarende til de menter, vi har anbragt ovenover. Hvis vi nu et sted havde fået *tre* et-taller under hinanden, ville resultatet have været *11*, altså 1 ned og 1 i mente!

Nu flytter vi det forreste ettal hen under bageste ciffer og lægger til:

$$\begin{array}{r} 11 \\ 110000011 \\ +\ \searrow\!\!\longrightarrow 1 \\ \hline 10000100 \end{array}$$

– hvorefter vi kontrollerer repræsentationen:

Det første ettal betegner 1×2^7, det andet 1×2^2. 10000100 betyder altså decimalt $128 + 4$ – og det er netop de 132, vi skulle få som resultat!

Vi fik altså – trods det indviklede forløb, som en datamaskine naturligvis klarer i løbet af nogle hundredetusinde dele af et sekund – vort regnskab til at stemme ganske præcist.

Når man vil have en datamaskine til at udføre en sådan stærkt detaljeret udregning, bruger man en af de makro-instruktioner, vi tidligere har beskæftiget os med. Man siger ikke »komplementér det og det tal og læg det til .... o.s.v. ....«. Man siger kort og præcist:

Start
$349 \div 217 = R$
Find værdien for R
Stop

Da divisionen er gentagen subtraktion på samme måde som multiplikation er gentagen addition, kan vi rigeligt klare os med en maskine, der blot kan lægge binære tal sammen to og to.

For maskinen er på forhånd programmeret med instruktioner om, hvorledes man omsætter fra decimaltal til binære tal – og omvendt, når resultatet skal skrives ud.

Men det er samtidig klart, at man ikke kan give en makro-instruktion, medmindre maskinen i forvejen er programmeret også med instruktioner for, hvad den skal *gøre* med tallene i hvert enkelt tilfælde.

En datamaskine kan udover at regne også sammenligne to tal og afgøre, om det ene er større end det andet – og så kan den instrueres om at fortsætte bearbejdningen af en datamængde på én måde, hvis et givet tal er *større* end et andet, men fortsætte på en helt anden måde, hvis tallet viser sig at være mindre, eller på en tredie måde, hvis tallene er lige store.

Hvis vi tænker os, at vi blandt 3.000 indbyggere i en by gerne vil have besked om, hvor mange drenge, der er under 14 år, kan vi bruge de almindelige hulkort, som folkeregistrene ligger med – og nøjes med at opsøge det sted, hvor fødselsdatoen er angivet.

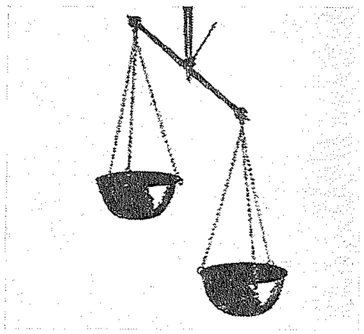
Vi kan nu starte undersøgelsen ved at bede maskinen aflæse »Kort nr. 1«. Næste spørgsmål må så blive: »Er den på kortet opgivne alder under 14 år?« »Hvis JA, læg 1 til tallet i tælleværket og 1 til tallet i første instruks og læs den. Hvis NEJ, læg 1 til tallet i første instruks og læs den«. Man kan så slutte instruktionsrækken med en besked om, at maskinen skal standse, når tallet i første instruks er 3.000. Så er man sikker på, at samtlige indbyggeres kort er blevet undersøgt.

Hvis man ønsker opgivet en procent, kan man indflette en optælling af samtlige indbyggere – og instruere maskinen om at finde procenten i stedet for det reelle antal.

Eksemplet hører til de simpleste, men viser tydeligt, at »0« og »1« kan repræsentere f. eks. »Ja« og »Nej«, hvis vi ønsker det.

Det viser sådan set også, hvordan en datamaskine kan bruges til direkte styring f. eks. af kemiske processer. Vi kan nemlig umiddelbart tænke os f. eks. en varmemåler koblet ind, så spørgsmålet til maskinen bliver: »Er vandet over 14 grader varmt? Hvis Ja, sluk for gassen, hvis Nej, lad gassen brænde«. Vi må så blot også koble en ventil til gasledningen – og lade »Nej« svare til åben og »Ja« til lukket ventil.

Vi er nu i stand til konstant at holde en temperatur i vandet på nøjagtig 14 grader – forudsat, at processen er sat i gang, når vi instruerer maskinen om at passe apparatet.



I et utal af datamatiske situationer genfinder vi de tre muligheder, som også er til stede, når vi vejer noget: En mængde kan være større end (>) en anden, lig med (=) en anden eller mindre end (<) en anden. På samme måde som det, vi anbringer i den ene vægtskål, kan være enten tungere end det, vi har anbragt i den anden – eller lettere eller veje akkurat det samme.

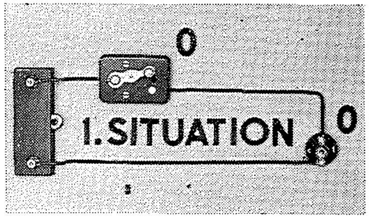
Sådan får man kredsløb til at regne

Lidt om lamper, der lyser og betyder »1« – eller er slukkede og betyder »0«.

Vi vil nu vende tilbage til og besvare spørgsmålet: Hvorfor kan de højt udviklede – og særdeles indviklede – maskiner, som al moderne databehandling er baset på, kun regne med nuller og ettaller?

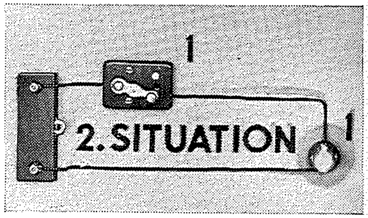
Svaret er ligetil: En datamaskine kan kun opfatte to slags symboler, fordi hele dens virksomhed er baseret på et ganske simpelt tænde- og slukkeprincip som det, vi kender fra elektriske belysninger: Enten er der blus på lampen – eller også lyser den ikke. Der gives kun disse to muligheder.

For at opnå den ene situation, må vi tænde, og for at opnå den anden må vi slukke.



Derfor har man vedtaget at lade den situation, vi opnår ved at sende strøm – f. eks. fra et batteri – gennem en tænde- og slukkekontakt til en glødelampe, repræsentere ved symbolet »1«, medens den modsatte situation, der opstår, når vi afbryder strømmen, repræsenteres ved symbolet »0«.

»0« betyder altså »Ingen strøm på kredsløbet« = slukket lampe, medens »1« betyder »Strøm på kredsløbet« = tændt lampe.



For at illustrere disse situationer, har vi konstrueret en »princip-maskine« ved hjælp af et batteri (yderst til venstre), en drejekontakt (foroven) og en lille glødelampe (i fatningen til højre). De tre ting er forbundet med hinanden ved hjælp af et stykke ledning.

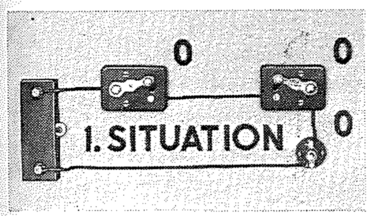
Vi vil nu se, hvorledes dette simple elektriske kredsløb kan bruges til at regne med.

Vi indretter først et kredsløb med to i stedet for med en tænde-og-slukke-kontakt – og så vi vil vi påstå, at vi har skabt verdens simpleste multiplikationsmaskine, der kan udføre samtlige de fire regnestykker, vi tidligere har gennemgået, da vi beskæftigede os med Den meget lille tabel.

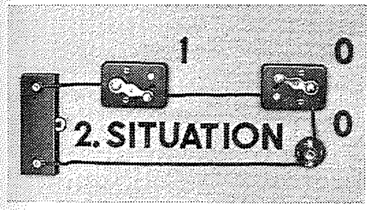
Som dengang påvist eksisterer der kun disse fire multiplikationsmuligheder i det binære system:

$$0 \times 0 = 00$$
$$1 \times 0 = 00$$
$$0 \times 1 = 00$$
$$1 \times 1 = 01$$

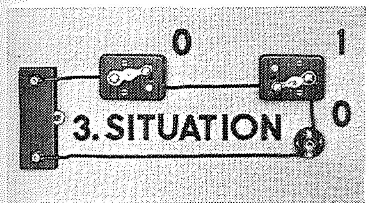
Vi ser på kredsløbet med de to kontakter, at vi har fire og kun fire muligheder for kombination af kontakt-stillinger:



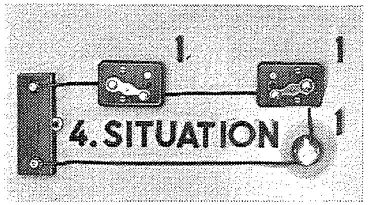
1. situation: Højre kontakt står åben som repræsentation for 0. Venstre kontakt står åben som repræsentation for 0. Lampen er slukket som repræsentation for 0. Åben × åben = slukket, $0 \times 0 = 0$.



2. situation: Venstre kontakt er lukket som repræsentation for 1. Højre kontakt står åben som repræsentation for 0. Lampen er slukket som repræsentation for 0. Lukket × åben = slukket, $1 \times 0 = 0$.



3. situation: Venstre kontakt er åben som repræsentation for 0, højre kontakt er lukket som repræsentation for 1. Lampen er slukket som repræsentation for 0. Åben × lukket = slukket, $0 \times 1 = 0$.



4. situation: Venstre kontakt er lukket som repræsentation for 1. Højre kontakt er lukket som repræsentation for 1. Lampen er tændt som repræsentation for 1. Lukket × lukket = tændt, $1 \times 1 = 1$.

Et sådant multiplikationskredsløb kaldes også et »og-kredsløb«.

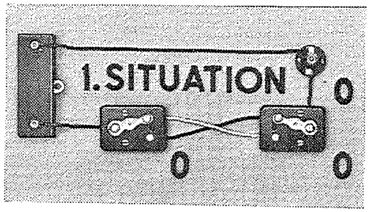
Vi vil nu prøve, om vi også kan lægge sammen ved hjælp af et kredsløb. Det viser sig muligt, hvis vi monterer en ekstra ledning mellem vore to kontakter – og lader ledningerne krydse hinanden.

Vi er nu klar til at se på additionsmulighederne. Der var fire og kun fire, har vi tidligere set:

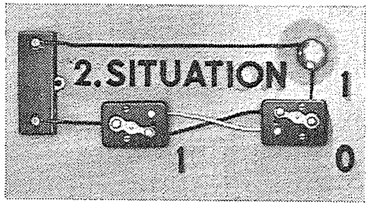
$$0 + 0 = 00$$
$$1 + 0 = 01$$
$$0 + 1 = 01$$
$$1 + 1 = 10$$

Da vi ved, at vore kontakter kan kombineres på fire måder, prøver vi os frem igen. Nu kan vi ikke længere tale om en åben og en lukket kontakt, men vi gør det nu alligevel, idet vi med »åben« i denne forbindelse mener

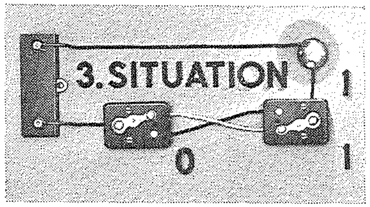
»opadvisende« og med »lukket« mener »nedadvisende« kontaktarm:



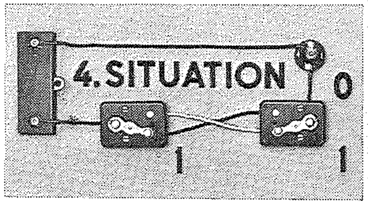
1. situation: Venstre kontakt er åben (0), højre er åben (0), lampen er slukket (0). Åben + åben = slukket, 0 + 0 = 0.



2. situation: Venstre er lukket (1), højre er åben (0), lampen er tændt (1). Lukket + åben = tændt, 0 + 1 = 1.



3. situation: Venstre er åben (0), højre er lukket (1), lampen er tændt (1). Åben + lukket = tændt, 0 + 1 = 1.



4. situation: Venstre er lukket (1), højre er lukket (1), lampen er slukket (0). Lukket + lukket = slukket, 1 + 1 = 0.

Vi ser altså, at systemet med de krydsede ledninger, som kaldes et »exclusiv-eller-kredsløb«, kan bruges i de tre første situationer, *men ikke i det fjerde,* for da skulle resultatet jo have været »10« og ikke »0«!

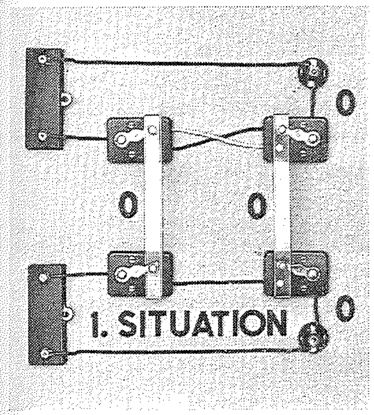
Sagt på en anden måde: Kredsløbets nul er godt nok i fjerde situation, *men menten* (ettallet foran nullet) *er gået fløjten.*

Det geniale er nu, at man ved at sætte et »-og-kredsløb« sammen med et »exclusiv-eller-kredsløb« og ved at betjene kontakterne to og to med en fælles styremekanisme kan opnå at få menten overført til en speciel »mentelampe«.

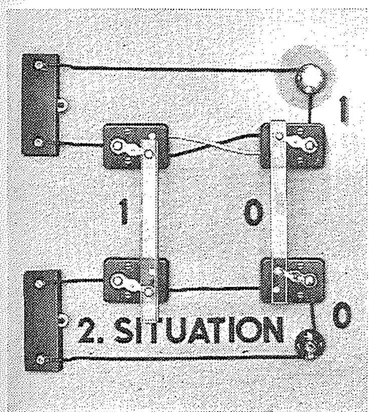
Som »styremekanisme« bruger vi en plasticstang, der lige akkurat passer ned over håndtagene på dreje-kontakterne. Hver af de to stænger bevæger altså samtidig to kontakter, sådan som vist på næste side.

Vi gentager nu de 4 additionssituationer i det nye dobbeltkredsløb — og så sker der følgende:

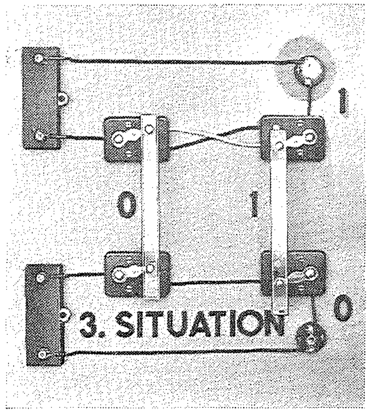
1. situation: Venstre kontakt er åben (0), højre kontakt er åben (0), nederste lampe er slukket (0),



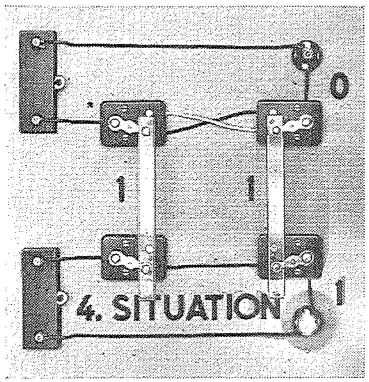
øverste lampe er slukket (0). Åben + åben = slukket/slukket, 0 + 0 = 00.



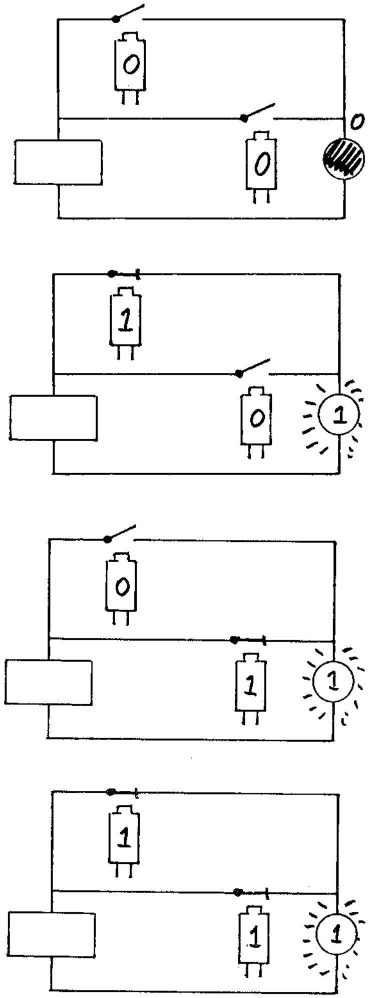
2. situation: Venstre kontakt er lukket (1), højre er åben (0), nederste lampe er slukket, øverste lampe er tændt. Lukket + åben = slukket/åben, 1 + 0 = 01.



3. situation: Venstre kontakt er åben (0), højre er lukket (1), nederste lampe er slukket, øverste lampe er tændt. Åben + lukket = slukket/åben, 0 + 1 = 01.



4. situation: Venstre kontakt er lukket (1), højre er lukket (1), nederste lampe lyser (1), øverste lampe er slukket. Lukket + lukket = tændt/slukket, 1 + 1 = 10.



»Eller-kredsløb« i 4 situationer.

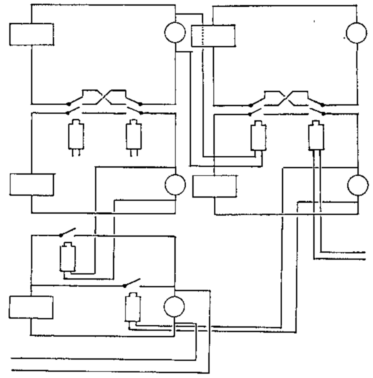
Vi har altså vitterligt konstrueret en maskine, der kan lægge to encifrede tal sammen fuldt ud korrekt i enhver henseende.

Det er blot en begrænsning ved en sådan regneenhed, at den kun kan regne med encifrede tal – men så er det heldigt, at vi ved at kombinere to sådanne dobbeltkredsløb og sætte dem i forbindelse med en tredie slags kredsløb, et såkaldt »eller-kredsløb«, kan frembringe en regneenhed, der ikke alene kan lade menten gå videre til næste ciffer i et flercifret tal, men også modtage en eventuel mente fra et forudgående ciffer (på næste plads til højre), hvis et sådant findes.

Også et »eller-kredsløb« har to kontakter, men det er indrettet, så lampen lyser, når én af kontakterne *eller de begge* er lukkede.

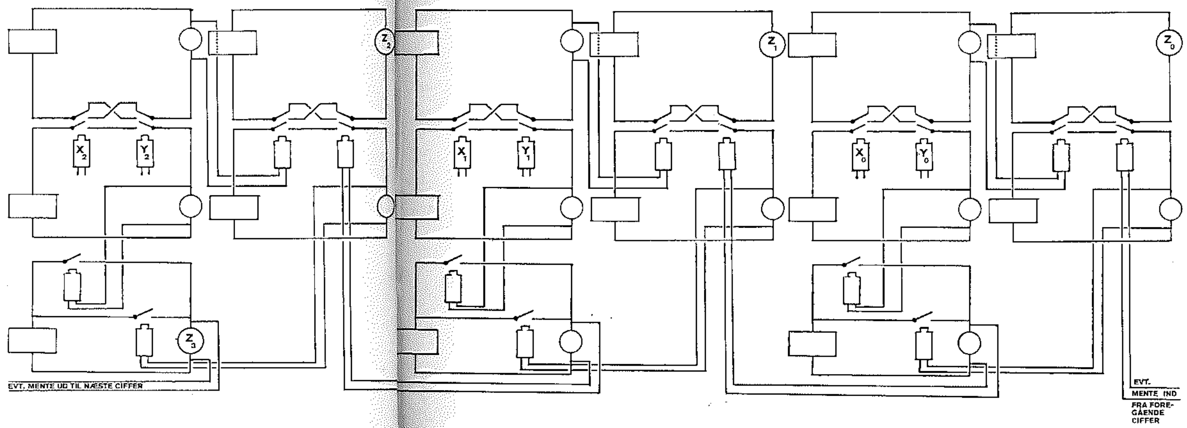
Hvordan de fem kredsløb skal forbindes, for at man kan frembringe en »heladder«, kan man se hosstående, hvor tænde-og-slukkekontakterne er erstattet med simple, elektromagnetiske relæer sådan som i de allerførste datamaskiner.

Vi ser, at strømmen fra sumlampen i venstre halvadder foroven styrer den venstre elektromagnet i den højre halvadder. Vi ser, at strømmen fra mentelampen i højre halvadder styrer højre elektromagnet i eller-eller-kredsløbet forneden. Vi ser, at strømmen fra venstre halvadders mentelampe forneden styrer den venstre elektromagnet i eller-eller-kredsløbet. Vi ser, at strømmen fra eller-eller-kredsløbets lampe er ført ud til venstre, og at der tilsvarende kommer to ledninger ind fra højre med strøm

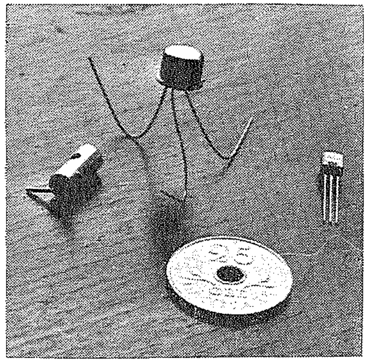


Universel regneenhed, bestående af 2 »halv-adders« og et »eller-eller-kredsløb« kaldes for en »heladder«.

til højre halvadders højre elektromagnet. Det er mente-strømmen, der kommer henholdsvis ud



Additionsmaskine, bestående af tre »hel-adders« i serie. Kan lægge 2 trecifrede tal sammen ($X_2X_1X_0 + Y_2Y_1Y_0 = Z_2Z_1Z_0$) og modtage en eventuel mente fra et foregående ciffer og give en mente videre til følgende.



Tre generationer transistorer, den ældste til venstre.

af og ind i vor helt selvstændige regneenhed.

Den er nu klar til at blive bygget sammen med en hel række ligestillede - lige så mange, vi ønsker, der skal være mulighed for at skaffe cifferplads til. Der skal bruges en heladder for hvert ciffer.

Den række, der er arrangeret side 59, kan eksempelvis gengive et trecifret resultat - efter at have modtaget en eventuel mente fra højre. Og den kan herudover videregive en eventuel mente fra tredie ciffer ude til venstre.

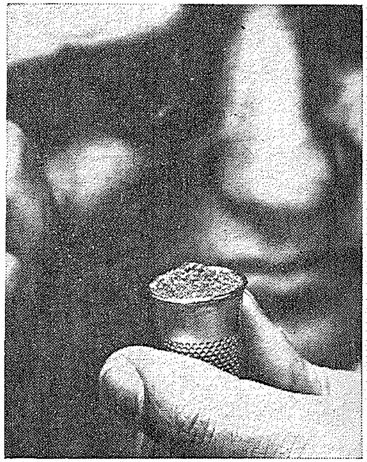
Hvis vi betegner det ene trecifrede tal, som vi vil regne med, ved koden $X_2X_1X_0$ og det andet tal ved koden $Y_2Y_1Y_0$, medens resultatet af additionen kaldes $Z_2Z_1Z_0$, kan man på tegningen se, hvor X- og Y-cifrene skal anbringes i maskinen, og hvor additionsresultatet kommer til syne, uden at nogen har så meget som rørt en finger for at regne det ud.

Det går helt af sig selv!

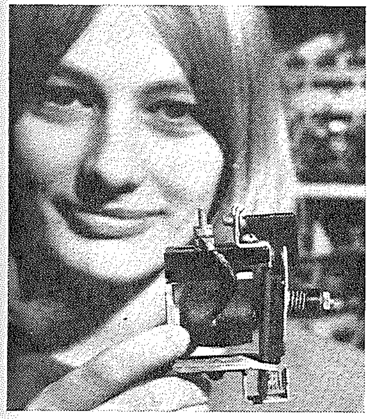
Da vi ved, at mange unge er meget interesserede i disse ting, vil vi ikke undlade at gøre opmærksom på, at man kan købe elektromagnetiske relæer flere steder - og at tegningen direkte kan bruges til at bygge en miniature »elektronregnemaskine« efter.

Som allerede omtalt i afsnittet om datamaskinernes historie, benytter man ikke længere elektromagnetiske relæer, men transistorer og integrerede kredsløb, det vil sige meget små, meget tæt sammenbyggede og »altkunnende« kredsløb, som masseproduceres.

Nogle af dem er så små, at der kan være 60.000 af dem i et fingerbøl. Det betyder, at man nu kan bygge datamaskiner, der er



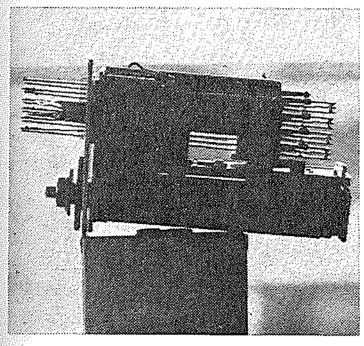
60.000 kredsløb kan der være i et fingerbøl — og så er der ovenikøbet ikke engang trængsel . . .



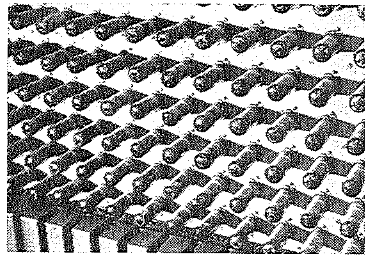
Tidligt elektromagnetisk relæ (enkelt) fra en IBM-hulkortsorterer.

betydeligt mindre end de første, der kom på markedet, og som samtidig er betydelig sikrere at regne med.

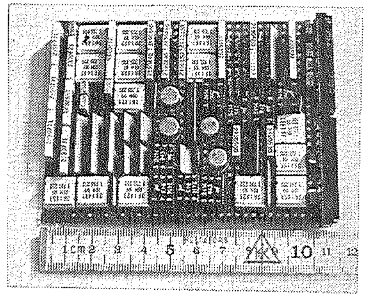
Det betyder også en væsentlig billiggørelse af datamaskiner.



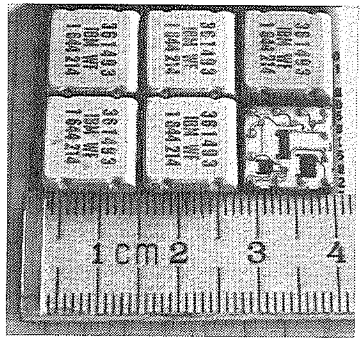
Elektromagnetisk relæ med mange »klapper«.



Panel med radiorør - til Regnecentralens første-generations-maskine DASK.



Tredie generations kredsløb til datamaskine.



Tredie generations kredsløb til datamaskine. Låget løftet viser de enkelte minikomponenters placering.

Sådan får man jernringe til at huske

Lidt om ferritkerner, der kan gemme på en oplysning – og give den fra sig igen.

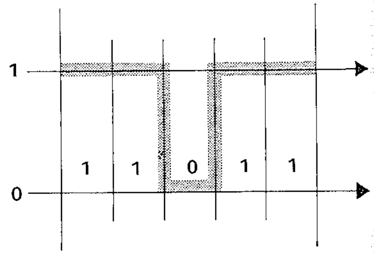
Vi har nu gjort rede for princippet i den del af en datamaskine, der kaldes regneenheden, og som Babbage kaldte »Møllen«.

Vi har fortalt, at de elektromagnetiske relæer er erstattet af transistorer og integrerede kredsløb – og naturligvis bruger man heller ikke batterier som strømkilde og glødelamper til at vise resultater med.

Inden vi derefter går over til »Depotet«, må det lige nævnes, at *regneenheden* i ethvert moderne databehandlingsanlæg har sin egen *strømforsyning*, som i sig selv er en kompliceret – og transistoriseret – affære.

For at få et ensartet »grundmateriale« at arbejde med, hakker en speciel mekanisme den frembragte, elektriske strøm i småstykker, så man får nøjagtigt lige lange perioder af »ingen-strøm-situationer«, der repræsenterer »0«, og af »strøm-situationer«, der repræsenterer »1«.

Når først disse perioder eller *sekvenser* er skabt, kan man tappe dem, så meget man vil, og bruge dem til at repræsentere forskellige binære tal med, efterhånden som disse indføres i maskinen.

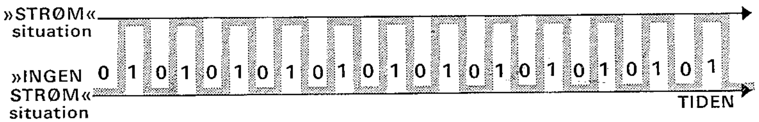


Her er den binære repræsentation for decimaltallet 27 udtrykt i elektriske impulser.

I de første datamaskiner skabte man de ønskede perioder af strøm og ikke-strøm ved hjælp af en roterende valse, på hvis side der var indgraveret en serie tidsmarkeringer, som henholdsvis sluttede og afbrød strømmen, når valsen roterede. I vore dage udvikles den i en fritløbende elektronisk oscillator.

En datamaskines *lagerenhed* kan sammenlignes med de celler i den menneskelige hjerne, der bruges til opbevaring af erfaret viden. Begge »depoter« er indrettet, så de kan *modtage* data, *gemme* data og *afgive* data.

I en datamaskine består cellerne af ganske små ringe af jernilte og andre stoffer, som er presset sammen og bagt i en ovn ligesom keramik. Jernindholdet i disse ferritkerner muliggør magnetisering, og det er dét, man benytter sig af:



Sekvensforløb.

Man trækker en isoleret kobbertråd gennem en ring og sender en elektrisk strøm gennem tråden. Så opstår der magnetisme i ringen, og *den bliver der*, selv om man afbryder den elektriske strøm.

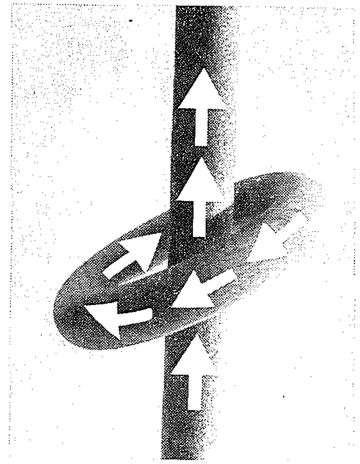
Herved indfører man tiden som faktor. Og tiden er jo en faktor i enhver hukommelsesproces. Ligesom på en båndoptager kan man indspille data, gemme dem så længe man ønsker – og genskabe dem i deres oprindelige skikkelse med deres repræsentation ubeskadiget.

Man gør det ved at måle magnetismen i ringen ved hjælp af en anden tråd, der også er stukket gennem ringen.

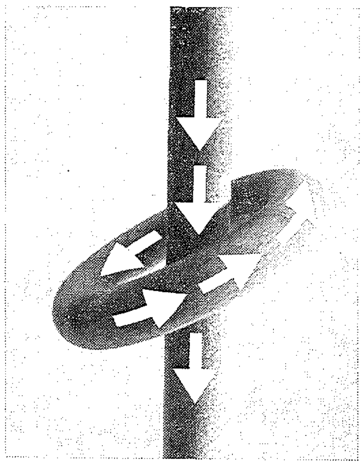
Nu kan man i én ring kun lagre ét binært ciffer – et ettal, hvis strømmen kommer oppefra, hvorved magnetismen i ringen får en retning den ene vej rundt, eller et nul, hvis strømmen kommer nedefra og giver magnetismen i ringen en retning den modsatte vej rundt.

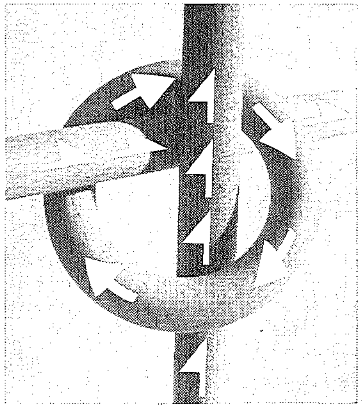
Derfor må man have flere ringe, *mange* ringe, i et datalager – og derfor har man fundet på at trække dem på samme tråd som perler på en snor og sætte trådene op side om side som på en kugleramme.

De lodrette tråde har man derefter suppleret med vandrette tråde, der også er ført gennem ferritringene, og ved at sende halvdelen af den strøm, der er



I en ferritkerne kan man skifte magnetisk »omløbsretning« ved at ændre den elektriske strøms retning i den tråd, der går gennem »gardinringen«.





Ved at sende kun halvdelen af den nødvendige elektriske effekt ind i datalageret ad 2 ledninger, som står vinkelret på hinanden, kan man tilføre én og kun én ring i systemet nok til en magnetisering.

nødvendig for en magnetisering igennem den lodrette tråd og halvdelen gennem den vandrette har man skabt mulighed for at tilføre én – og kun én – ring i systemet nok elektrisk strøm til en magnetisering. Alle de andre ringe på samme lodrette og samme vandrette tråd får hver især kun halvdelen af den strøm, der er nødvendig – og de forbliver upåvirkede.

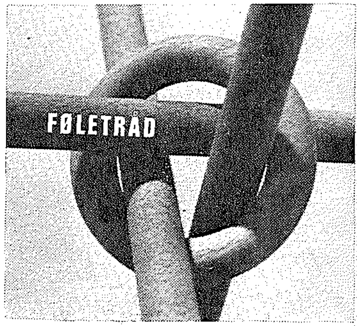
Endelig har man sat alle rammerne op ved siden af hinanden til en kasse – og syet »føletråden« igennem samtlige ringe i den derved opståede tredie dimension.

Der går altså tre tråde gennem hver ferritkerne, men så er systemet også umiddelbart brugbart til sit formål: at modtage data, at gemme data og at afgive data.

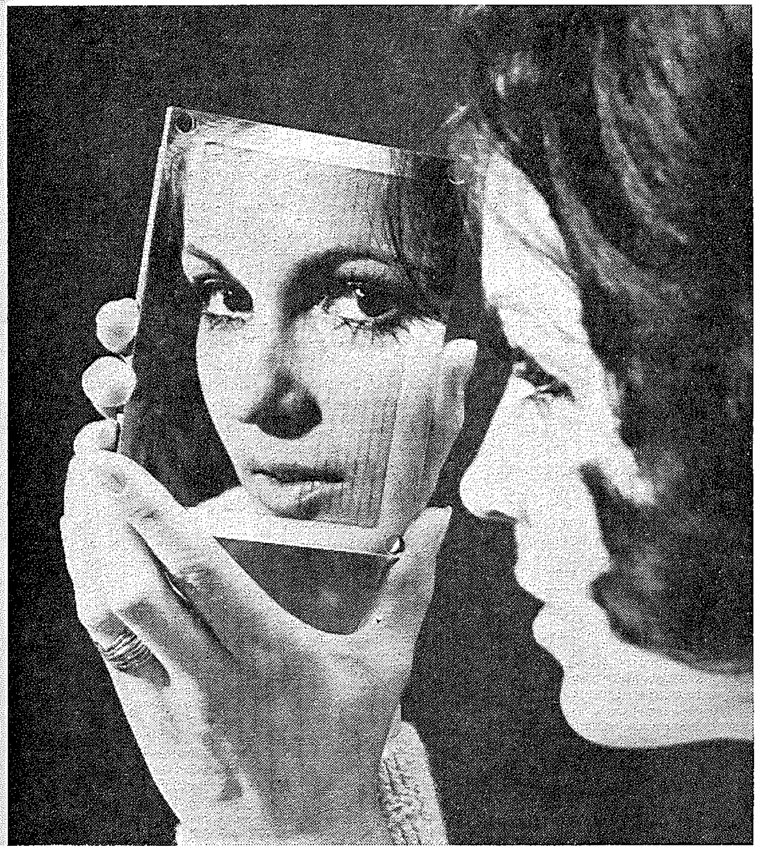
I stedet for glødelamperne i vor simple princip-maskine fører man nu ledninger fra regneenheden til ferritlageret, men indskyder konktakter – »porte« eller »ventiler« – der betjenes pr. instruktion til maskinen.

Da enhver lagercelle har sit ganske bestemte nummer – eller *adresse,* som EDB-folkene siger –, er det muligt at instruere maskinen om at sende et resultat til midlertidig opbevaring – eller komplicere fremgangsmåden ved at gøre en adressering betinget – »Hvis Ja, celle 47586 –, hvis Nej, celle 46572«.

Når en regneopgave er løst, kan man instruere maskinen om at hente indholdet af en bestemt celle og føre det til en skrivemaskine (»Skriveren« i Babbages regi) – og trykke det på papir som resultat af beregningen. Man kan også – som tidligere påvist – føre resultatet til en ventil eller anden regulator, der kan virke



Føletråden i en enkelt ferritkerne er her vist i model — det er den, der gør det muligt at »aflæse« indholdet af hver enkelt ferritkerne.



I de seneste år er flere og flere datamaskine-fabrikker gået over til at bruge tyndfilm — spindelvævstynde jern-nikkel-flager, beklædt med et isolerende materiale — i stedet for ferritkerner. Det »spejl«, den unge dame holder i sine hænder, er en ramme med sådanne tyndfilm-flager. Der er plads til 3150 lagerceller i en sådan ramme, og hver celle kan rumme en information. Ved hjælp af en sådan ramme — den er pillet ud af en SIEMENS datamaskine — er det muligt at indføre eller slette cirka ti millioner informationer pr. sekund.

tilbage på den indførte datamængde og ændre denne – i et fuldautomatisk styringssystem, der også er åbent over for ændring af andre data, f. eks. på grund af uforudsete begivenheder andre steder i processen.

Det er umiddelbart forståeligt, at et sådant anlæg er særdeles velegnet til overvågning af komplicerede og langvarige videnskabelige laboratorieforsøg – og kemisk storindustri.

Foruden ferritlageret, der også kaldes »hurtiglageret« eller »arbejdslageret«, og som oftest er koblet direkte til regneenheden, har man et eller flere *ydre lagre*, som især bruges til opbevaring af særligt store datamængder.

Det kan have forskellige former. Det kan være en eller flere *magnetbåndstationer,* som i princippet er båndoptagere. Det kan være et *pladelager,* bygget efter grammofonpladeprincippet med mange plader stablet oven på hinanden og en pick-up til ind- og udlæsning for hver plade.

Det kan også være et *tromlelager* med magnetisk materiale om en cylinder.

Der er flere andre former – men fælles for dem er, at de ligesom det indre lager kan modtage data, gemme data og afgive data i ubeskadiget tilstand.

Af Babbages fire enheder har vi herefter kun »Kontrollen« tilbage at beskrive:

Kontrolenheden – eller *Styreenheden* – i en moderne datamaskine sørger for den interne trafik på samme måde som færdselspolitiet og trafiksignalerne i en storby modvirker færdselskaos og sørger for, at alle vejfarende kommer hjem i god behold (= med ubeskadiget datarepræsentation).

Derimod fører den ingen udadvendt kontrol med dataindholdet. Det gør et antal faste programmer, som er lagrede konstant i maskinen. Disse programmer fører bl. a. regnskab med start, arbejdsmængde, stop, maskinfejl, skrivefejl, fejldata o.s.v., og i større EDB-anlæg fører de endvidere automatisk en selvstændig »logbog« på en speciel *konsolskriver* – en elektrisk skrivemaskine, der også bruges til igangsætning af processerne.

Hvorledes de forskellige enheder er forbundet med hinanden, og hvordan arbejdsgangen mellem dem er, fremgår af principskitsen side 31.

Hvorledes det er i virkeligheden, afhænger meget af, hvilket fabrikat man arbejder med, men det har vi ikke betragtet som vor opgave at gøre rede for i dette kapitel, som kun skal danne indledningen til den egentlige beskrivelse af EDB.

De kan læse mere om

Indre lager side 167.
Ydre lager side 167.
Adresser side 168.

Datamaskinen

Materiel, virkemåde og mikroprogrammering

Af professor Per Gert Jensen

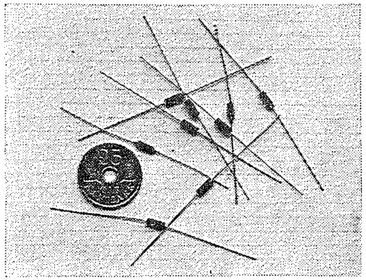
Datamaskiner kan programmeres til at løse enhver opgave, blot der kan angives en fremgangsmåde til bestemmelse af løsningen. Man kunne heraf forledes til at tro, at en datamaskine skal kunne udføre et stort antal forskellige operationer, og at datamaskinen tilsvarende må indeholde et stort antal forskellige elektroniske komponenter. Dette er imidlertid *ikke* tilfældet. Enhver kompliceret beregningsproces kan opløses i en række simple processer, og det elektroniske kredsløb, som udfører disse simple processer, kan igen opløses i nogle meget simple grundkredsløb. I indledningen så vi at elektroniske kontakter kunne benyttes som grundkredsløb i et additionsnetværk, idet værdien af et binært ciffer kunne repræsenteres ved en sluttet – henholdsvis en åben – kontakt. I elektroniske datamaskiner benyttes ikke kontakter som grundkredsløb, men *portkredse,* og værdien af et binært ciffer repræsenteres ved en elektrisk spænding.

I det følgende skal vi indledningsvis se, hvorledes portkredse er opbygget, og hvorledes de virker. Derefter vises som eksempel *sammenkobling af portkredse til en aritmetisk enhed.* Endvidere behandles kort *aktive registre,* der er specielle hukommelseselementer til opbevaring af mellemresultater, og endelig omtales *den interne styring af aritmetiske operationer.*

Til slut vises på blokskemaform en komplet datamaskine med sine principielle dataføringsveje – *databusser* – og styringen af datatransporter på busserne.

PORTKREDSE

I moderne datamaskiner udføres de logiske grundoperationer af portkredse opbygget af dioder og transistorer, og de øvrige operationer er sammensat af disse logiske grundoperationer. Rent teknologisk kan grundkredsene fremstilles af enkeltkomponenter – diskrete komponenter – såsom dioder, transistorer, modstande etc., der fabrikeres hver for sig og derefter loddes sammen til det ønskede kredsløb. Eller man kan fremstille hele kredsløbet med alle dets enkeltkomponenter og interne forbindelser i en enkelt arbejdsgang af samme art som bruges ved



Figur 1. Typiske dioder til brug i datamaskiner. Mønten tjener som størrelsesmål.