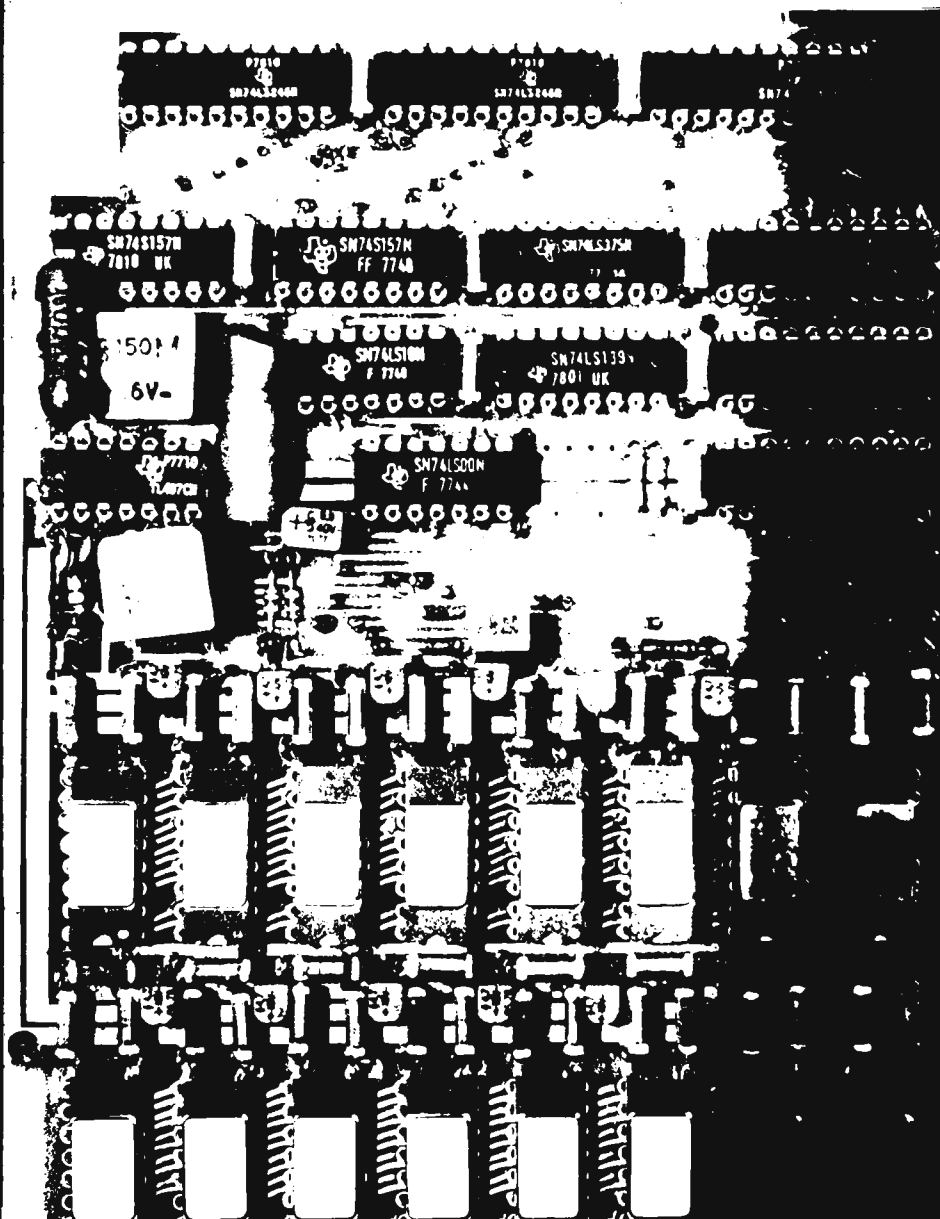
10 Håndbog for datamat-amatører

1978



INDHOLDSFORTEGNELSE

ALMENT OM PROGRAMMERBARE MASKINER

Sådan begyndte det	A	1
Den forventede udvikling	A	7
Talsystemer	A	21
Binær matematik	A	28
Logiske operationer	A	31

BIBLIOTEKET – PROGRAMMER

HP-25, Delefiltre	B	1
HP-25, Gæt et tal	B	3
HP-25, Likviditet	B	5
HP-25, Mastermind	B	11
KIM-1, Multimaze	B	13
IMSAI 8080, RAM-test	B	17
KIM-1, FUT-FUT	B	19
TI-59, Løn og skat	B	21
TI-59, Kørselsregnskab	B	27
BASIC, Lån, afdrag og renter	B	31
BASIC, Reaktionstid	B	37
TI-58/59, Omregn. mell. talsyst.	B	41
TI-58/59, Mastermind	B	43
SR-52, Sænke slagskibe	**B**	**45**
mek6800D2, MUSIC	**B**	**49**

CPU-ARKITEKTUR

CPU-Arkitektur	C	1
Motorola M6800	C	5
Intel 8080	C	7
SC/MP	C	9
Signetics 2650	C	11
Intel 8048	C	13
Zilog Z-80	C	15

DATAMAT-LITTERATUR

Elementært om Microdatorer	D	1
The first Book of KIM	D	2

KLUBINFORMATION

Datamatklubber	K	1
Datamatamatører	K	11
Seminar, april 1978	K	17

LOMMEREGNERE

TI-Programmer	L	1
HP-25/25C	L	3
TI-59/PC-100	L	5

MIKRODATAMATER

Valg af mikrodatamat	M	1
Datamatkapacitet	M	5
KIM-1	M	11
KIM-1, Kontakter og dioder	M	15
Motorola M6800	M	19
TK-80, begyndersæt	M	25
Imsai 8048 CC	M	30
Nye datamater, april 1978	M	33
PET 2001	M	37
TELMAC 1800	M	41
KIM-1, udviddet udgave	M	43

PROGRAMMERINGSTEKNIK

Lær programmering	P	1
Subrutiner	P	49
Splitning af subrutiner	P	51
BASIC	P	55
BASIC, fortsat	**P**	**69**

SELVBYGGERPROJECTER

IMSAI 8080	S	1
Z-80 mikrodatamat	S	11
77-68 selvbyggerdatamat	S	51
Z-80, fortsat	S	57
Z-80, fortsat	**S**	**69**

YDRE ENHEDER

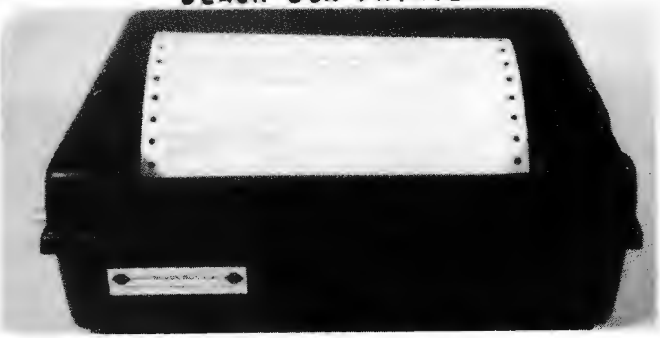
TV-skriver	Y	11
Pocket TTY	Y	13
TV-modulator	Y	16
Digital multiplexer	Y	19
ACT-IV skærmterminal	Y	25

Håndbog for datamat-amatører udgives i løsbladsformat af Telepress ApS, Greve Strandvej 42, 2670 Greve Strand. Tlf. (02) 90 86 00. Giro nr. 1 15 53 69. Tryk: Fraling Offset, Viby Sj. HFD udsendes til abonnenter som trykksag d. 1. torsdag hver måned. 1. nummer udgivet er nr. 9/1977. Et årsabonnement koster kr. 100,– incl. ringbind og porto. Ansvarshavende udgiver: H. Lind.

ISSN 0105–581X

Alment om programmerbare maskiner
Biblioteket - programmer
CPU-arkitektur
Datamat-litteratur
Interfacing
Klubinformation
Lommeregnere
Microdatamater
Programmeringsteknik
Selvbyggerprojekter
Tilbud fra læserne
Undervisningsudstyr
Ydre enheder

BLACK BOX PRINTER



SKRIVER ALLE 64 UPPERCASE ASCII TEGN
BILLIGSTE FULL-SIZE PRINTER - KR. 4700,- +MOMS

piezodan aps.

Bakkedraget 55 - DK 3480 Fredensborg - Tlf. (03) 28 37 44 - Teknisk afd. (01) 86 12 17

KOMMUNIKATION

kan være besværlig – specielt med en datamat . . .
MEN, med Micro-Term's intelligente ACT–IV terminal går det let. Den har full-size tastatur, fuld Cursor kontrol fra hardware og software, 64 ASCII karakterer og 32 printbare symboler, skærmredigering, tilslutning til ekstra skærm, RS-232 tilslutning, 300 til 19600 baud o.s.v.
ACT–IV burde koste en formue – MEN vi leverer den til

kr. 7800,– + MOMS

Og vi kan endda levere en „lillebror", ACT-1A (uden skærm) og med lidt færre „gimmics" til

kr. 3800,– + MOMS

Begge terminaler demonstreres gerne i vor tekniske afdeling.
Ring til (01) 86 12 17 og få en aftale.

piezodan aps.

Bakkedraget 55 - DK 3480 Fredensborg - Tlf. (03) 28 37 44 - Teknisk afd. (01) 86 12 17

STRØMFORSYNING

Desværre er det ikke kun genopfriskningen og den dobbelte brug af adressebenene, som giver problemer, når 4116 skal anvendes. Også strømforsyningen er lidt speciel. Og dog. Sagen er vel, at man efterhånden har vænnet sig til, at de fleste kredse, for at være TTL-kompatible, kun forsynes med +5V. At det kan lade sig gøre, er i virkligheden ret fantastisk, for mange af dem kræver internt flere spændinger. De folk, der konstruerer kredsene, må imidlertid kunne mere end deres fadervor, for i de fleste tilfælde er det lykkedes dem at skjule dette bag kredsenes indkapsling. Desværre hører 4116, af forskellige grunde, ikke til blandt disse og, for at det ikke skal være løgn, kræver denne kreds hele 3 forskellige spændinger.

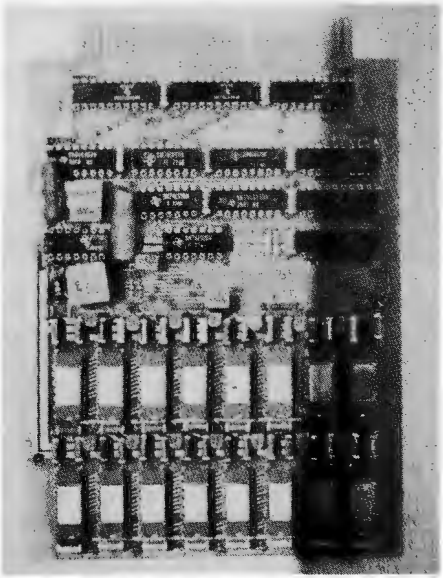
Selve den interne arbejdsspænding er +12V, og under arbejde og refresh, et ret højt strømforbrug, nemlig op til ca. 35 mA.

Kredsens ind- og udgange er TTL-kompatible. Der må altså ske en omsætning til/fra den interne spænding. Hertil bruges nogle konvertertrin, som kræver +5V, men ikke ret meget strøm. Til sidst kræver den anvendte teknologi en negativ forspænding på –5V for at arbejde. Strømforbruget er her meget lavt - 100-200 uA.

Af disse 3 spændinger er +5V jo datamatens hovedspænding og giver derfor ikke problemer. For de 2 andre spændinger var det imidlertid et spørgsmål, om de skulle dannes centralt i en stor ensretter eller direkte på kortet. Dels fordi den centrale løsning kunne betyde, at ensretteren skulle indrette for temmelig mange spændinger, dels af økonomiske grunde, valgte vi at danne spændingerne direkte på kortet. Dette skal altså kun forsynes med +5V.

Til at løse denne opgave bruger vi det på fig. 2 viste kredsløb. Det er en glimrende løsning, som vi uden at rødme, har hugget fra Zilogs MCB-board.

Hjertet er kredsen TL-497, som sammen med en spole og nogle modstande og kondensatorer danner en DC-DC konverter.



16/32K RAM-kortet. Øverst ses de 16 hukommelses kredse, derunder til højre er strømforsyningen.

Hovedkredsen til dannelse af +12V er ganske ordinær. Den virker ved, at en transistor, indbygget i kredsen og forbundet til ben 10, sender en strøm gennem spolen L1. Herved opbygges der i denne et magnetfelt. Som bekendt, har spoler den egenskab at de søger at vedligholde en konstant strøm gennem sig. Når transistoren pludselig afbryder, svarer spolen derfor ved at hæve spændingen over sig. Da den ene side er fastholdt til +5V, stiger spændingen på ben 10. Den forøgede spænding sendes gennem en ensretterdiode, forbundet til ben 6 og 7, til kondensatoren C6, som udglatter, og videre til kredsene.

Ved at variere forholdet mellem den tid, transistoren leder, og den tid, der er afbrudt, kan udgangsspændingen reguleres. Dette styrkes af ben 1, som gennem en spændingsdeler, bestående af R5 og R6, føler på spændingen.

Det smarte i koblingen er det lille sidekredsløb, som danner –5V.

Det virker ved, at når den indbyggede

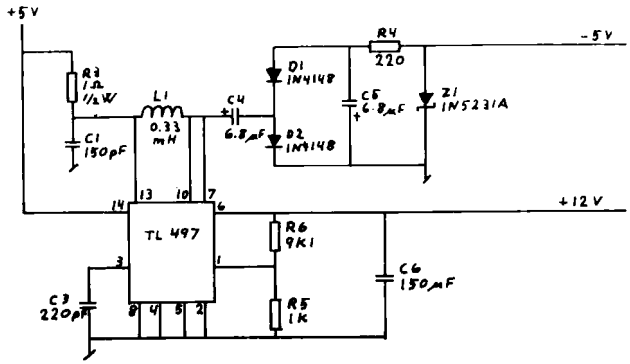


Fig 2

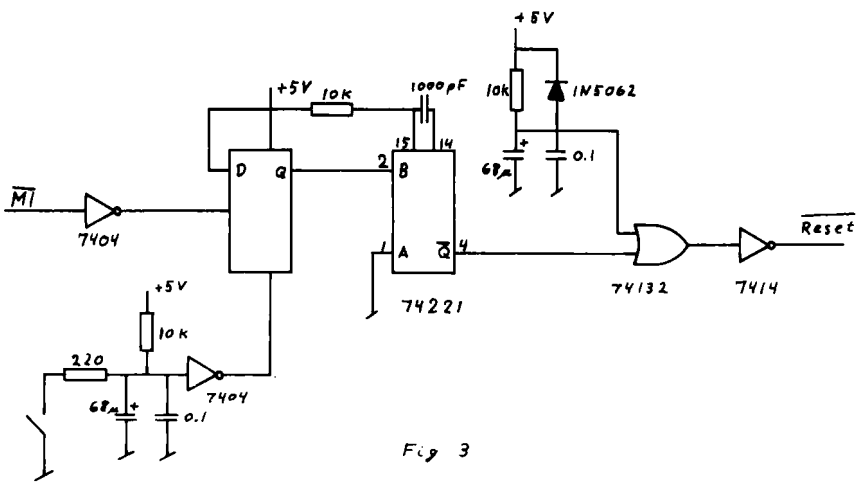


Fig 3

Forslag til Resetkredslab

transistor er afbrudt, oplades C4 til +12V. Højre side af kondensatoren er gennem D2 i forbindelse med 0V.
Når transistoren pludselig åbner for at sende ny strøm gennem L1, går fællespunktet for L1 og C4 pludselig mod 0 Volt. Nu har kondensatorer den egenskab, at de søger at fastholde en konstant spænding over sig. Det kan C4 kun gøre ved at sænke spændingen på højre side til −12V. Denne spænding føres gennem D2 og R4 ud til kredsene. C5 udglatter og Z1, som er en zenerdiode på 5,1V, regulerer udgangsspændingen.
Modstanden R3 er en strømbegrænsermodstand, og C3 stiller den interne oscillators frekvens.

KONSTRUKTIONEN

Efter at kortet, som sædvanligt, er undersøgt nøje under forstørrelsesglas, monteres først de robuste komponenter, d.v.s. IC fatninger, modstande og kondensatorer. Der er temmelig mange elektrolytkondensatorer. Pas på, at de bliver vendt rigtigt.
Om du vil anvende fatninger for de integrerede kredse, er din egen afgørelse, men de tilrådes brugt til hukommelseskredsene.
Herefter monteres de integrerede kredse - BORTSET fra hukommelseskredsene, som først monteres under afprøvningen.
Lodningerne undersøges nu nøje under forstørrelsesglas. Mange af afstandene på kortet er meget små, og en loddebro opstår let. Ja, vi ved godt, det er irriterende arbejde, men det kan spare mange timer under den senere afprøvning.

AFPRØVNING

Kortet tilsluttes 5V, og det kontrolleres dels at der ikke kommer røg, dels at ingen af kredsene bliver alt for varme. Her skal du være opmærksom på, at IC25 og 26 ikke er LS-kredse, og derfor bliver noget varmere end de andre kredse.
Herefter indstilles DC-DC-konverteren. Et godt voltmeter forbindes til IC19 ben 6, og det kontrolleres, at spændingen er lige omkring 12V. Hukommelseskredsene har plus/minus 10 % tolerance på alle forsyningsspændingerne, så indstillingen er ikke særlig kritisk.
Hvis spændingen ikke er i orden, ændres R4 til det passer. Større R4 giver højere spænding og omvendt.
For at kompensere for spændingsfald, er det nok en god ide at søge spændingen lagt så tæt ved 12,2V, som muligt. Herefter kontrolleres over Z1, at −5V også er i orden.
Det undersøges nu, om alle hukommelseskredsene har +12V på ben 8, +5V på ben 9 og −5V på ben 16.
Med et oscilloskop kontrolleres derefter, at +12V på kredsene er "ren", d.v.s. at de spidser, som en DC-DC konverter er tilbøjelig til at lave, ikke går udenfor toleranceområdet.
Når alt er i orden, indsættes kortet i backplanen og datamaten startes i et program, hvis natur er ligegyldig, blot der hele tiden sker læsning af instruktioner, og det ikke bruger dette kort.
Med oscilloskop kontrolleres, at adressebenene A0-A6, samt RAS og CAS arbejder. Denne kontrol sker direkte på fatningerne for et par af hukommelseskredsene.
I en af de kontrollerede fatninger monteres nu en kreds. Behandl den forsigtigt, for den er temmelig sart. Vi har erfaring for, at fugtige hænder er en god forholdsregel mod statisk elektricitet, så tænk lidt på, hvad kredsen koster.
Via datamatens monitor undersøger du nu, om det er muligt at gemme et 0 eller et 1 i den pågældende kreds. Her skal du være opmærksom på, at nogle monitorer undersøger, om det man skriver ind i RAM-lageret også kommer rigtigt ind. Da kun en kreds er monteret, må denne funktion blokeres.
Er alt i orden, monteres et par hukommelseskredse mere, og undersøgelsen gentages. Således fortsættes, til alle kredse er monteret.

Under hele denne undersøgelse skal du være opmærksom på, at kortets adresse ikke overlapper med nogle af de andre kort. Som tidligere skrevet, er kortet ved levering lagt i de øverste 32K af datamatens adresseringsområde, altså fra 8000H til FFFFH.

Z-80 datamaten, som er opbygget af de beskrevne kort, indeholder 44K RAM og K ROM.

RESTART

Når du er nået hertil med et 16/32K-kort, der virker, er du jo nok i paradis, men ak, hvor længe havde Adam (m/k) udsigt til æbletræet der? - Kredsene skal jo genopfriskes, og det kan CPU'en kun gøre, når den udfører "Instruction Fetch". Det gør den altid, undtagen under Reset, Wait eller Bus Acknowledge. Ingen af disse tilstande må derfor, ifølge fabrikens specifikationer, vare længere end ca. 1 mS.

Især i forbindelse med Reset giver dette problemer, for du kan simpelthen ikke trykke så hurtigt på kontakten, at du kan være sikker på, at der ikke er forsvundet information.

Det kan derfor være nødvendigt at forsyne datamaten med det på fig. 3 viste kredsløb. Til gengæld skal så kondensatoren C fjernes fra CPU-kortet.

Koblingen er ikke indbygget på RAM-kortet, men må opbygges ved siden af, f.eks. på et stykke Vero-board.

Der findes dog også en anden løsning. De færreste amatører har brug for CPU'ens NMI-indgang. Disse kan derfor forbinde Restart-knappen mellem denne linie på backplanen og stel.

Når NMI-linien går til 0, springer CPU'en automatisk til adressen 0066H. Her kan man så anbringe en jump-instruktion til programmets rigtige startadresse.

Da NMI-indgangen er negativt flanketriget, bør Restart-knappen være bouncefri, eller man kan forbinde en kondensator på 10µF/10V tværs over knappen.

ERFARINGER

Vi har nu haft dette kort kørende i temmelig lang tid, og det har ikke på noget tidspunkt budt på problemer af nogen art. Dette til trods for, at mange professionelle betragter dynamiske RAM som noget, man kun skal anvende, hvis man er absolut tvunget dertil.

Men hvorfor i det hele taget anvende dette kort, når statiske RAM virker lige så godt og uden alle de problemer, vi har beskrevet undervejs?

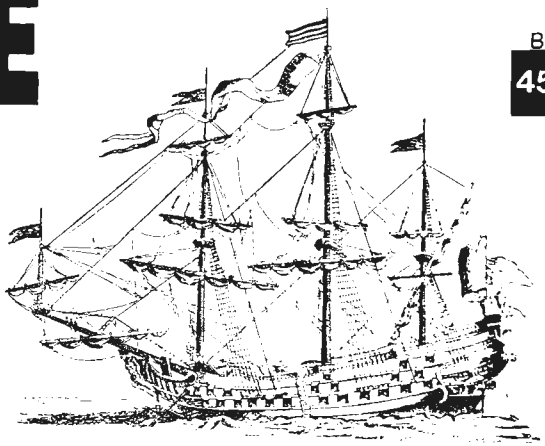
Jo, først og fremmest bliver 8 kort erstattet af eet. Der er også en vis prisforskel, især hvis vi kan få et gruppekøb af 4116 og sidst, men ikke mindst, kortet bruger kun ca. 20 % af den effekt, den tilsvarende mængde RAM-kort bruger.

Desuden er problemerne jo først og fremmest til besvær for konstruktørerne, medens brugerne ikke vil opdage noget.

STYKLISTE
for 16/32K RAM-kort

16 stk.	IC1-16	4116P-4 - ell. hurtigere
3 stk.	IC17, 20 og 23	74LS241
1 stk.	IC18	74LS00
1 stk.	IC19	TL-497
1 stk..	IC21	74LS139
1 stk.	IC222	74LS10
1 stk.	IC24	74LS375
2 stk.	IC25-26	74S157
1 stk.	IC27	74LS240
2 stk.	IC28-29	74LS245
56 stk.	Cx	0,1uF
14 stk.	Cz	2,2uF/16V Fast tantal
2 stk.	C4-5	6,8uF/40V Fast tantal
1 stk.	C6	150uF/16 V Fast tantal
1 stk.	C1	150pF
1 stk.	C3	82pF
1 stk.	L1	Drosselspole 0,33uH
2 stk.	R1-2	2K2
1 stk.	R3	1Ohm/1/2w
1 stk.	R4	220 Ohm
1 stk.	R5	IK
1 stk.	R6	9K1
1 stk.	Z1	Zenerdiode 5,1V IN5231A
2 stk.	D1-2	IN 4148
1 stk.	printkort	78073

SKYDE SLAG-SKIBE



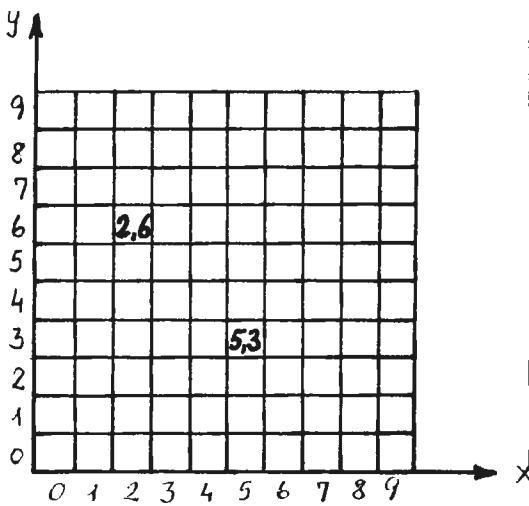
Nedenstående program, der er beregnet til en SR-52, er indsendt af Tage Bentsen, Holeby. Programmet sætter én istand til at spille det populære spil „Sænke Slagskibe" med lommeregneren.

Spillet kræver 2 magnetkort. Den sidste del af hovedprogrammet (udlægning af skibe) er indlæst på B-siden af kort 1, der iøvrigt er blank på A-siden!
A-siden af kort 2 indeholder den første halvdel af hovedprogrammet (000-111), og B-siden indeholder hele bi-programmet (sænkning af skibe).

1 Tag et tilfældigt tal $0 < x < 10$ med mindst 1 decimal og tryk STO 98.
2 Indlæs kort 1, både A- og B-side.
3 Indlæs kort 2, kun A-side.
4 Tryk E display: 10
Tryk D display: -3
Tryk D display: -2
Tryk D display: -1
Tryk D display: 0
5 Indlæs kort 2, kun B-side
6 Gæt et koordinat x,y (0,0 - 9,9) og tryk A.

Hvis forbi display: 0
Hvis ramt display: 1
Hvis sunket display: ,1111111111

Ved nyt spil, blot repetér operationerne 3-6.



SÆNKE SLAGSKIBE (udlægning af skibe)

000	LBL	045	1	090	=	135	RCL	180	SUM
	E′		7		E′		1		1
	INV		1		÷		6		7
	EE		0		1		=		RCL
	INV		IND		0		INV		1
005	DMS	050	STO	095	=	140	if zro	185	7
	INV		0		STO		2		-
	DMS		0		1		0		5
	fix		dsz		4		8		=
	0		0		E′		1		rset
010	DMS	055	4	100	STO	145	SUM	190	1
	INV		7		1		1		STO
	fix		rset		6		9		1
	rtn		LBL		RCL		RCL		5
	LBL		D		1		1		RCL
015	D′	060	RCL	105	4	150	4	195	1
	RCL		1		-		IND		4
	9		7		IND		STO		-
	8		SUM		RCL		1		9
	+		1		0		9		,
020	2	065	8	110	0	155	RCL	200	9
	$=x^y$		INV		=		1		5
	9		st flg		if zro		5		=
	-		0		2		SUM		INV
025	E′	070	-	115	0	160	1	205	if pos
	=		4		8		4		1
	STO		,		dsz		1		4
	9		5		1		0		4
	8		=		0		STO		RCL
030	×	075	if pos	120	if flg	165	0	210	1
	1		0		0		RCL		8
	0		8		1		1		-
	=		1		9		9		RCL
	rtn		st flg		0		-		1
035	LBL	080	0	125	,	170	RCL	215	7
	E		1		1		1		=
	CMs		0		STO		8		STO
	1		STO		1		=		1
	0		0		5		INV		8
040	STO	085	0	130	RCL	175	if zro	220	STO
	0		D′		1		1		1
	0		×		4		0		9
	1		1		E′		3		GTO
	STO		0		-		1		D

SÆNKE SLAGSKIBE (sænkning af skibe)

000	LBL	020	1	040	0	060	1	085	1
	A		9		-		1		0
	STO		-		7		RCL		6
	1		IND		=		1		1
	9		RCL		if pos		1		HLT
005	-	025	0	045	0	065	-	090	1
	RCL		0		9		2		SUM
	0		=		0		=		1
	1		if zro		RCL		if zro		3
	=		0		0		1		RCL
010	if zro	030	3	050	0	070	0	095	1
	1		8		-		6		3
	0		dsz		4		1		-
	6		0		=		HLT		4
	1		1		if pos		1		=
015	0	035	9	055	0	075	SUM	100	if zro
	STO		0		7		1		1
	0		HLT		4		2		0
	0		RCL		1		RCL		6
	RCL		0		SUM		1		1
						080	2	105	HLT
							-		9
							3		$1/x$
							=		HLT
							if zro		

ORIENTERING OM FUNKTIONERNE

UDLÆGNING AF SKIBE

000-013 Integer

014-034 Tilfældige tal, register 98 anvendes, fordi det ikke påvirkes af CMs.

035-057 Klargøring, herunder en indlagring af 10 i registerne 01-10.

058-080 Skibets retning fastlægges af et tilfældigt tal.

081-102 Koordinatgenerator (x,y).

103-119 Dubletundersøgelse af koordinater, hvis to ens springes til 208, og nyt skib udlægges.

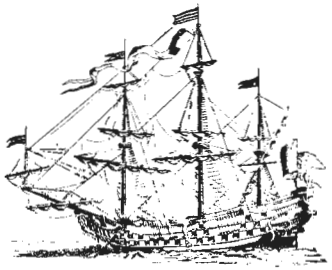
120-124 Hvis flag 0 (skibet har vandret orintering) springes til 190.

125-143 Bruges ved lodret orientering. 0,1 lagres i reg. 15 til senere opsummering som fortsættelse af grundkoordinaten. Ligeledes føres der kontrol med, om hver skibssektion - da det ligger lodret - har samme x-værdi, hvis integerskift springes til 208, og nyt skib udlægges.

144-189 Lagring af koordinater. For hver indlagring føres dubletkontrol (afsnit 103-119). Forskellen mellem reg. 19 og reg. 18 angiver hvor mange led, skibet skal bestå af - først 1, så 2 og 3, og sluttelig 4 led.

190-207 Anvendes ved vandret orientering. 1 lagres i reg. 15 til senere opsummering som fortsættelse af grundkoordinaten. Her føres selvsagt også kontrol m.h.t. om skibet begynder at vokse ud over værdien x = 9,y. Hvis det er tilfældet, fortsætter programmet ned til 208, og nyt skib konstrueres.

208-223 I tilfælde af at to skibe „kolliderer" under udlægningen, eller et skib vokser ud over koordinatsystemet, rekonstrueres de værdier, der før „uheldet" eksisterede i reg. 18 og reg. 19. Derpå bygges et nyt ved ordren GTO D.



SÆNKNING AF SKIBE

Jeg vil først oplyse, at skibenes koordinater ligger i reg. 01-10, fordelt således:

01	02	03	04	05	06	07	08	09	10

000-013 Det valgte koordinat lagres i reg. 19 og sammenlignes med reg. 00, hvis ens springes til 106.

014-018 10 lagres i reg. 00.

019-037 Det valgte koordinat sammenlignes med samtlige 10 koordinatregistre, der springes dog til 038, når der er lighed mellem indholdet af et koordinatregister og det valgte koordinat. Ellers 0 HLT.

038-047 Hvis indholdet af reg. 00 er 7≤ reg. 00. hvis det er tilfældet springes ned til 074, ellers fortsættes.

048-057 Sammenligning igen med reg. 00: 4 ≤ reg. 00, hvis det er tilfældet, springes til 074, ellers fortsættes.

058-073 1 summeres i reg. 11, derpå trækkes reg. 11 frem og subtraheres med 2, hvis 0 (2 træffere) springes til 106, ellers 1 HLT.

074-089 Princippet identisk med 058-073, blot har vi her fat i det 3-ledede skib.

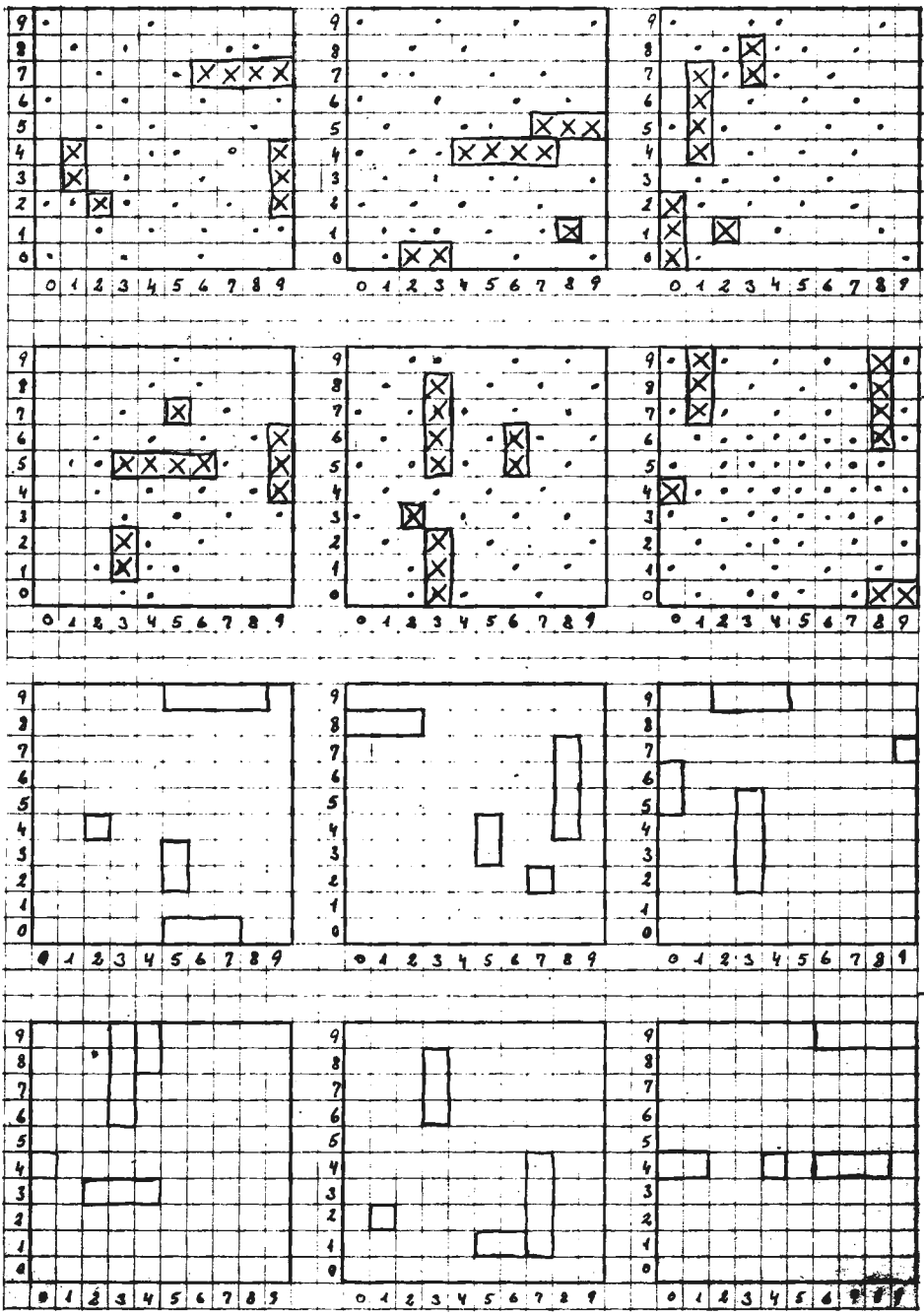
090-105 Princippet identisk med 058-073, blot har vi her fat i det 4-ledede skib.

106-108 Hvis et skib er sunket, fremtræder der en række 1-taller på displayet.

EKSEMPEL, x = π

B

48



Konklusionen må blive, at programmet med de to READ-sætninger er både hurtigere og sikrere, selv om det fylder en linje mere. Hvis man har en RC COMAL eller en ID COMAL til sin rådighed, skal man naturligvis benytte versionen med REPEAT..UNTIL. Dels undgår man helt at bruge GOTO, og dels får man den automatiske linjeindrykning mellem REPEAT og UNTIL, der giver en meget tydelig afgrænsning af løkken. I den nye ID COMAL (findes på ID 7000 og MPS 2000) har man endog den mulighed at bruge en helt tredje metode, som er antydet her:

```
30 LOOP
40    READ T
50    IF T=0 THEN EXIT
60    LET SUM=SUM+T
70 ENDLOOP
```

Ordet LOOP betyder "løkke" og ENDLOOP betyder "slutløkke". De tre linjer inden i løkken gentages, indtil T har fået værdien 0. Når det Boolske udtryk efter IF er sand, udføres ordren EXIT (udgang), med det resultat, at udførelsen fortsættes med linjen efter ENDLOOP. Ved at se nærmere på løkken vil læseren sikkert indse, at den har samme struktur som programmet med de to hopsætninger. Forfatteren har endnu ikke haft lejlighed til at afprøve ID COMAL, der først bliver frigivet i dette efterår, men det kan udmærket vise sig, at LOOP.. ENDLOOP er lige så hurtig, som REPEAT..UNTIL (eller hurtigere), og man sparer naturligvis en READ-sætning ved at bruge den.

Man kan have flere DATA-sætninger i et program, og de kan anbringes hvor som helst i programmet. De bliver nemlig under alle omstændigheder udført først af fortolkeren, idet den simpelthen begynder med at søge hele programmet igennem for at finde eventuelle DATA-sætninger. Hvis den finder sådanne sætninger, stiller den samtlige data i dem op i een lang liste. Når den første READ-sætning udføres, læses der forfra i denne liste, som vi har set i eksemplet ovenfor.

Vi har også set, at der kan være flere READ-sætninger, som læser i den samme liste. En sådan liste kaldes også en **data-kø**. Betegnelsen er meget rammende, idet de enkelte data netop behandles, som man plejer at ekspedere en kø. Hvis data-køen tømmes under en læsning, fremkommer der er fejlmelding fra systemet. Derimod gør det naturligvis ikke noget, at der evt. er data tilovers, når man er færdig med at læse. I så henseende er en data-kø mere tålmodig end en sædvanlig kø af personer, hvorfra der sikkert vil lyde vrede råb, hvis lugen lukkes for næsen af de tre sidste i køen. Hvis der er data tilovers, når den sidste READ-sætning er udført, tyder det dog på, at der er fejl i programmet.

Selv om en data-kø er læst til ende, står tallene stadig i programmet, og man har en mulighed for at læse hele køen engang til. Man anvender da blot en sætning, der kun består at ordet

RESTORE

Hvis der efter denne sætning optræder flere READ-sætninger, begynder læsningen forfra i køen.

En READ-sætning kan indeholde flere variable. I de fleste Basic-versioner kan man have lige så mange, som linjens længde tillader. Lad os slutte dette afsnit med et eksempel:

```
0010 READ A,B,C
0020 READ X
0030 REPEAT
0040   PRINT X,(A*X+B)*X+C
0050   READ X
0060 UNTIL X=1E+30
0070 DATA 2,-1,3
0080 DATA -3,-2,-1,0,1,2,3
0090 DATA 1E+30
```

Øvelse

Hvilke værdier har A, B og C, når linje 10 er udførte? Hvilken værdi har X, når linje 20 er udført? Udregn ved almindelig "blyantsregning" værdien af udtrykket i linje 40, når X har denne værdi. Hvilken værdi har X, når REPEAT..

UNTIL-løkken standser? bliver denne værdi brugt i udregningen i linje 40?
Skriv programmet om til standard Basic med brug af een betinget hopsætning. Skriv det derpå med brug af een betinget og een ubetinget hopsætning, således at der kun er een linje, som indeholder sætningen READ X.

8. INDDATA FRA TERMINALEN

Vi vil endnu engang se på det program, der bruges til at finde gennemsnittet af et observationssæt. I de versioner af programmet, vi hidtil har skrevet, er observationssættet anført i en DATA-sætning, og de enkelte observationer bliver læst i en READ-sætning. I stedet for på denne måde at stille tallene op i en datakø kan man vælge at indtaste dem fra terminalen. Hvis vi vælger at bruge denne metode, kan programmet, der beregner gennemsnit, skrives således:

```
0010 LET SUM=0
0020 INPUT T
0030 REPEAT
0040   LET SUM=SUM+T
0050   INPUT T
0060 UNTIL T=0
0070 LET GNS=SUM/10
0080 PRINT "GENNEMSNITTET ER:";GNS
```


Man bemærker, at observationssættet ikke længere er opført i programmet, og at vi i stedet for sætningen READ T har indført sætningen:

INPUT T

Når denne sætning udføres, sker der følgende: Systemet skriver et bestemt tegn på terminalen, fx. et spørgsmålstegn (?) eller et kolon (:), hvorpå **det standser og venter** på at operatøren skal **indtaste** et tal. Lad os tænke os, at operatøren sidder med den liste af tal, vi brugte i forrige afsnit. I så fald skal tallet 67 indtastes. Når dette er sket og operatøren trykker på RETURN-tasten, bliver det indtastede tal **tildelt den variable T som værdi.** T får altså værdien 67 i vort eksempel. INPUT T virker helt analog med READ T, idet T blot får sin værdi fra terminalen i stedet for at læse den i en data-kø. Operatøren fortsætter med at indtaste tallene i observationssættet og slutter med at indtaste et 0. Derpå har det Boolske udtryk efter UNTIL i linje 60 værdien sand (T er lige med 0), og udførelsen af løkken standser, hvorpå gennemsnittet udregnes og udskrives.
I de fleste tilfælde er INPUT-sætninger mere bekvemme at bruge end READ-DATA-sætninger, idet man umiddelbart kan ændre værdierne til de variable uden at man skal til at skrive DATA-sætninger om. Derfor bruges DATA-READ-sætninger som regel kun, når der er tale om sæt af **faste værdier**, der skal bruges ved alle kørsler af programmet. Vi skal senere se eksempler på dette. DATA-READ-sætninger har dog een fordel fremfor INPUT-sætninger. Hvis man ikke har nogen mulighed for at lagre sine data på kassettebånd eller diskette men kun kan skrive programlister på hullestrimler, bliver evt. data i DATA-sætninger naturligvis hullet ud sammen med programmet og går ikke tabt, hvorimod tal, der er indlæst ved hjælp af INPUT-sætninger, i givet fald må indtastes igen. I standard BASIC ser programmet med INPUT-sætningerne således ud:

```
0010 LET S=0
0020 INPUT T
0030 LET S=S+T
0040 INPUT T
0050 IF T<>0 THEN GOTO 0030
0060 LET G=S/10
0070 PRINT "GENNEMSNITTET ER:";G
```

ØVELSE

Skriv programmet om, så der bruges to hopsætninger i det, og til gengæld kun een linje med INPUT-sætningen. Jvf. programmet fra forrige afsnit.
Der er endnu en fordel forbundet med at bruge INPUT-sætninger i stedet for READ-sætninger. Lad os tænke os, at det observationssæt, for hvilket vi vil udregne gennemsnittet, består af et stort antal observationer, og at vi ikke

ved helt nøjagtigt, hvor mange der er. Vi kan naturligvis give os til at tælle elementerne i det, men vi kan også benytte et program som følgende:

```
0010 LET SUM=0; ANTAL=0
0020 INPUT T
0030 REPEAT
0040   LET SUM=SUM+T
0050   LET ANTAL=ANTAL+1
0060   INPUT T
0070 UNTIL T=0
0080 PRINT "GENNEMSNITTET ER:";SUM/ANTAL
```

I det program har vi indført en ny tæller ANTAL, som fra begyndelsen sættes lig med 0. Hver gang, der indtastes en ny talværdi, øges ANTAL med 1 (linje 50), idet det dog er så snedigt indrettet, at denne optælling ikke sker, hvis operatøren indtaster et 0. I dette sidste tilfælde standser løkken, og sætning 50 udføres ikke mere. Programmet tillader altså, at et vilkårligt observationssæt indtastes, og når indtastningen er afsluttet med et 0, bliver gennemsnittet udregnet som SUM/ANTAL (bemærk, at udregningen er anbragt i selve PRINT-sætningen). Hvis man ønsker det, kan man naturligvis også få udskrevet værdien af ANTAL ved blot at anbringe en PRINT-sætning mere i programmet:

```
0010 LET SUM=0; ANTAL=0
0020 INPUT T
0030 REPEAT
0040   LET SUM=SUM+T
0050   LET ANTAL=ANTAL+1
0060   INPUT T
0070 UNTIL T=0
0075 PRINT "DER ER";ANTAL;"OBSERVATIONER
     I SAETTET, OG"
0080 PRINT "GENNEMSNITTET ER:";SUM/ANTAL
```

Programmet giver nu udskrifter som fx. denne:

DER ER 22 OBSERVATIONER I SÆTTET, OG GENNEMSNITTET ER 157.5

I standard BASIC ser programmet således ud:

```
0010 LET S=0
0020 LET A=0
0025 INPUT T
0040 LET S=S+T
0050 LET A=A+1
0060 INPUT T
0070 IF T<>0 THEN GOTO 0040
0075 PRINT "DER VAR";A;"OBSERVATIONER
     I SAETTET, OG"
0080 PRINT "GENNEMSNITTET ER:";S/A
```

Man bør bemærke, at de variable i BASIC-programmet betegnes ved enkelte bogstaver, og at vi ikke bruger flere tildelinger på samme linje. Begge disse omstændigheder er desværre med til at gøre BASIC-programmet væsentligt mere uoverskuelig og vanskelig at læse. I mange nye versioner af almindelig BASIC tillades lange variabelnavne, og man kan have flere tildelinger på samme linje. I nogle versioner kan man fx. skrive således:

LET S=0 : LET A=0

og i andre således:

S=0 : A=0

I fx. den BASIC, som findes på PET-datamaten kan man endog skrive:

SUM=0 : ANTAL=0

Læseren bør se omhyggeligt efter i manualen for at finde ud af, om BASIC-versionen tillader sådanne skrivemåder. Desværre findes den fuldt strukturerede BASIC (COMAL) endnu kun på de tidligere nævnte mini- og mikrodatamater, der dog alle er velkendte her i landet.
Som tidligere nævnt sætter systemet et bestemt tegn, når en INPUT-sætning udføres, inden det standser og venter på indtastningen. Dette tegn kan være forskellig fra system til system, og læseren må konsultere sin BASIC-manual for at se efter, hvilket tegn der skrives som INPUT-signal. Man kan have flere variable angivet efter ordet INPUT, og man må foretage lige så mange indtast-

ninger, som der er variable til. Som eksempel kan vi se på følgende omskrivning af det program, der afsluttede forrige afsnit:

```
0010 INPUT A,B,C
0020 INPUT X
0030 REPEAT
0040   PRINT "POLYNOMIETS VAERDI:";(A*X+B)*X+C
0050   INPUT X
0060 UNTIL X=1E+50
```

Når sætningen i linje 10 udføres, skriver systemet det tegn, der angiver, at det er klar til at læse et tal fra terminalen. Når det første tal er indtastet, bliver det tildelt A som værdi, hvorpå systemet igen angiver, at det er klart til at læse et tal. Når dette er indtastet, bliver det tildelt B som værdi, hvorpå systemet igen skriver INPUT-tegnet. Det tredje tal, der indstastes, bliver tildelt C som værdi. Derpå udføres linje 20, og nu kan der indtastes en værdi for X. Når værdien af polynomiet er udskrevet (linje 40), kan der indtastes en ny værdi for X, og polynomiets værdi for dette X udskrives, såfremt det ikke er lig med 10^{50} (1E+50). Hvis det er tilfældet, standser programmet. I standard BASIC kan programmet fx. se således ud:

```
0010 INPUT A,B,C
0020 INPUT X
0040 PRINT "POLYNOMIETS VAERDI:";(A*X+B)*X+C
0050 INPUT X
0060 IF X<>1E+50 THEN GOTO 0040
```

eller således:

```
0010 INPUT A,B,C
0020 INPUT X
0030 IF X=1E+50 THEN GOTO 0060
0040 PRINT "POLYNOMIETS VAERDI:";(A*X+B)*X+C
0050 GOTO 0020
0060 END
```

I linje 60 finder man sætningen, der kun består af ordet END. Når denne sætning udføres, standser programmet. I dette tilfælde er det nødvendigt at have denne sætning med som **hopadresse** for den betingede hopordre i linje 30.

I de umiddelbart foregående programmer skal der i linje 10 indtastes værdier for A, B og C. Af det tegn, systemet sætter, når INPUT-sætningen udføres, kan man **ikke** se, hvilken betydning de talværdier, der indtastes, har. Hvis der er tale om store programmer, som ikke er skrevet af den, der bruger dem, kan det hurtigt blive meget uoverskueligt, hvad de forskellige størrelser betyder. I sådanne tilfælde bør man ubetinget udstyre programmet med **operatørvejledninger**, dvs. udskrifter der fortæller brugeren, hvilken betydning de indtastede tal har. Som eksempel vil vi atter se på det program, der udregner og udskriver værdien af et polynomium:

```
0005 PRINT "INDTAST KOEFFICIENTERNE A, B OG C:";
0010 INPUT A,B,C
0012 PRINT
0015 PRINT "INDTAST EN VAERDI FOR DEN VARIABLE X:";
0020 INPUT X
0030 REPEAT
0035   PRINT
0040   PRINT "POLYNOMIETS VAERDI:";(A*X+B)*X+C
0043   PRINT
0045   PRINT "NY VAERDI FOR X:";
0050   INPUT X
0060 UNTIL X=1E+50
```

Når linje 5 og 10 udføres, skriver systemet:

INDTAST KOEFFICIENTERNE A, B OG C:?

og operatøren er klar over, at de indtastede talværdier vil blive tildelt koefficienterne A, B og C. Når de tre værdier er indtastet, skriver systemet:

INDTAST EN VAERDI FOR DEN VARIABLE X:?

og brugeren ved straks, hvad programmet ønsker at vide. I standard BASIC ser programmet fx. således ud:

```
0010 PRINT "INDTAST KOEFFICIENTERNE A, B OG C:";
0020 INPUT A,B,C
0030 PRINT
0040 PRINT "INDTAST EN VAERDI FOR DEN VARIABLE X:";
0050 INPUT X
0060 PRINT
0070 PRINT "POLYNOMIETS VAERDI:";(A*X+B)*X+C
0080 PRINT
0090 PRINT "NY VAERDI FOR X:";
0100 INPUT X
0110 IF X<>1E+50 THEN GOTO 0060
```

De mange PRINT-sætninger uden indhold er anbragt for at "give luft" mellem udskrifterne.

Som afslutning på dette afsnit vil vi skrive et program, der kan udføre nogle simple statistiske beregninger.

Som grundlag for beregningerne har vi et observationssæt, som vi ikke på forhånd behøver vide noget særligt om. Programmet skal med andre ord, kunne bruges på vilkårlige observationssæt. Vi ønsker at beregne følgende størrelser: **middelværdi, varians** og **spredning**. Middelværdien er det samme som gennemsnittet, til hvis beregning vi allerede har opstillet adskillige programmer. Variansen får man på følgende måde: man danner **summen af kvadraterne** på samtlige observationer og finder gennemsnittet af denne sum ved at dividere med antallet af observationer. Herfra trækker vi kvadratet på middelværdien. Hvis vi betegner de enkelte observationer x_1, x_2, x_3, . . ., x_n, har vi i sædvanlig matematisk notation:

middelværdien $mX = (X_1 + X_2 + X_3 + \ldots + X_n)/n$

variansen $vX = (X_1^2 + X_2^2 + X_3^2 + \ldots + X_n^2)/n - mX^2$

Spredning er **kvadratroden** af variansen. Vort statistiske program ser således ud i struktureret BASIC (COMAL):

```
0010 LET SUM=0; ANTAL=0
0020 INPUT T
0030 REPEAT
0040   LET SUM=SUM+T
0050   LET KVSUM=KVSUM+T*T
0060   LET ANTAL=ANTAL+1
0070   INPUT T
0080 UNTIL T=0
0090 LET MX=SUM/ANTAL
0100 PRINT "DER ER";ANTAL;"OBSERVATIONER
     I SAETTET, OG"
0110 PRINT "MIDDELVAERDIEN ER:";MX
0120 LET VX=KVSUM/ANTAL-MX*MX
0130 PRINT "VARIANSEN ER:    ";VX
0140 PRINT "SPREDNINGEN ER:  ";SQR(VX)
```

Vi beregner som før ANTAL og SUM, og desuden har vi nu en sætning, der beregner KVSUM (kvadratsummen) af observationerne (linje 50). I linje 90 beregnes middelværdien MX, og i linje 120 beregnes variansen VX. I linje 140 beregnes spredningen, og hertil benyttes den **indbyggede funktion** SQR (SQuare Root: kvadratrod), idet SQR(VX) netop er **kvadratroden af variansen.** Denne funktion findes i enhver BASIC-version. I standard BASIC kan man programmere opgaven således:

```
0010 LET S=0
0020 LET A=0
0030 INPUT T
0040 LET S=S+T
0050 LET K=K+T*T
0060 LET A=A+1
0070 INPUT T
0080 IF T<>0 THEN GOTO 0040
0090 LET M=S/A
0100 PRINT "DER ER";A;"OBSERVATIONER
     I SAETTET, OG"
0110 PRINT "MIDDELVAERDIEN ER:";M
0120 LET V=K/A-M*M
0130 PRINT "VARIANSEN ER:    ";V
0140 PRINT "SPREDNINGEN ER:  ";SQR(V)
```

Læseren vil muligvis bemærke, at det nu begynder at spille en rolle, at almindelig BASIC kun tillader meget korte betegnelser for variable.

ØVELSE

Skriv programmet om, så der benyttes to hopsætninger og kun een INPUT-sætning. Prøvekør programmet på en datamat og benyt følgende observationssæt som inddata:
Tæl det antal bogstaver, der forekommer i de enkelte ord i denne øvelse. Hver gang et ords bogstaver er talt, indtastes dette antal. Når hele øvelsen er "læst" på denne måde, indtastes tallet 0.

9. FORGRENINGER

Vi vil fortsætte med at udbygge statistikprogrammet fra forrige afsnit. Foruden de hidtil beregnede størrelser kan man også være interesseret i at kende de såkaldte **ekstremalværdier** for sættet, dvs. de to observationer der har hhv. den mindste og den største talværdi.
Lad os tænke os, at vort observationssæt består af tal, der er fremkommet ved at måle legemshøjderne på de unge mænd, der mødte på session en bestemt dag i Aarhus. Sættet begynder således:

172 184 168 176 170 180 178 173
185 163 . . .

Programmet, der skal bruges til de statistiske beregninger, kan se således ud:

```
0010 LET MAX=0; MIN=300
0020 LET SUM=0; ANTAL=0
0030 INPUT T
0040 REPEAT
0050   IF T>MAX THEN LET MAX=T
0060   IF T<MIN THEN LET MIN=T
0070   LET SUM=SUM+T
0080   LET KVSUM=KVSUM+T*T
0090   LET ANTAL=ANTAL+1
0100   INPUT T
0110 UNTIL T=0
0120 LET MX=SUM/ANTAL
```

```
013Ø PRINT "DER ER";ANTAL;"OBSERVATIONER
     I SAETTET, OG"
014Ø PRINT "MIDDELVAERDIEN ER:";MX
015Ø LET VX=KVSUM/ANTAL-MX*MX
016Ø PRINT "VARIANSEN ER:    ";VX
017Ø PRINT "SPREDNINGEN ER:  ";SQR(VX)
018Ø PRINT "STØRSTE VAERDI I SAETTET ER:";MAX
019Ø PRINT "MINDSTE VAERDI I SAETTET ER:";MIN
```

T	172	184	168	176	170	180	178	173	185	163
MAX	172	184	184	184						
MIN	172	172	168	168						

Det mest af programmet ser ud, som det vi tidligere har skrevet, men der er tilføjet nogle linjer med det formål at bestemme ekstremalværdierne. Den variabel, der skal indeholde værdien af mindste observation, kalder vi MIN (minimum), og den, der skal indeholde værdien af den største observation, kalder vi MAX (maximum). I linje 10 sættes MAX til 0 og MIN til 300. Disse begyndelsesværdier virker måske noget løjerlige, når man betænker, at MAX skal betyde den største, og MIN den mindste, men det følgende vil vise, at man gør klogt i at starte med sådanne værdier. I linje 50 og 60 finder vi sætningerne:

IF T>MAX THEN LET MAX=T

IF T<MIN THEN LET MIN=T

Sådanne sætninger kaldes **sammensatte sætninger**. Hver for sig består de af en IF . . THEN-sætning, der indeholder et Boolsk udtryk, efterfulgt af en LET-sætning (en tildeling). Når en sådan sammensat sætning udføres, undersøger systemet, om det Boolske udtryk har værdien sand eller falsk. Hvis det har værdien **sand, udføres sætningen efter** THEN; hvis det derimod har værdien **falsk**, går programmet **videre med næste sætning**. Det svarer helt nøje til udførelsen af en betinget hopsætning. Hvis betingelsen er opfyldt, udføres hoppet, og ellers fortsættes med næste linje. For at finde ud af, om de to sætninger virker efter hensigten, vil vi udføre en **manual prøvekørsel (sporing)** med den række observationer, vi indledte med at angive. Til det formål indretter vi en **sportabel**:

Den første værdi for T, som indtastes, er 172. Da MAX har værdien 0, er det Boolske udtryk i linje 50 sandt (172 er større end 0), og følgelig sættes MAX lig med 172. Det noterer vi i tabellen. Dernæst betragter vi sætningen i linje 60. Det Boolske udtryk i denne sætning er **også sandt**, idet MIN var sat til 300 (172 er mindre end 300). Følgelig sættes MIN lig med 172. Også dette resultat noterer vi i tabellen. Allerede nu aner læseren måske, at begyndelsesværdierne var valgt med omtanke. Når linje 100 udføres, indtastes tallet 184, og da det er forskellig fra 0, udføres løkken atter. Det Boolske udtryk T>MAX er atter sandt (184 er større end 172), så derfor sættes MAX lig med T (184), hvilket vi noterer. Derimod er udtrykket T<MIN falsk (184 er ikke mindre end 172), og tildelingen MIN=T udføres følgelig **ikke**. Vi noterer, at MIN stadig har værdien 172. Næste gang, løkken udføres, har T værdien 168. Da dette **ikke** er større end 184, ændres værdien af MAX ikke. Derimod er 168 mindre end 172, derfor sættes MIN lig med 168, hvilket vi noterer. Læseren har nu sikkert opdaget, hvor simpel den fremgangsmåde, vi benytter, i grunden er. Hver gang, der indtastes et tal, som er **større end det hidtil største** (MAX), sætter vi MAX lig med denne større værdi, og hver gang, der indtastes et tal, som er **mindre end det hidtil mindste** (MIN), sætter vi MIN lig med denne mindre værdi. Når vi er færdig med at indtaste hele observationssættet, har MAX altså netop en værdi, der svarer til det største tal i sættet, mens MIN har den mindste værdi.

ØVELSE
Gør sportabellen ovenfor færdig.

Mange almindelige BASIC-versioner tillader sådanne sammensatte sætninger, og herunder er anført et BASIC-program, der udfører det samme som det strukturerede COMAL-program ovenfor:

```
0010 LET M1=300
0020 LET M2=0
0030 LET S=0
0040 LET A=0
0050 INPUT T
0060 IF T>M2 THEN LET M2=T
0070 IF T<M1 THEN LET M1=T
0080 LET S=S+T
0090 LET K=K+T*T
0100 LET A=A+1
0110 INPUT T
0120 IF T<>0 THEN GOTO 0060
0130 LET M=S/A
0140 PRINT "DER ER";A;"OBSERVATIONER I SAETTET, OG"
0150 PRINT "MIDDELVAERDIEN ER:";M
0160 LET V=K/A-M*M
0170 PRINT "VARIANSEN ER:    ";V
0180 PRINT "SPREDNINGEN ER:  ";SQR(V)
0190 PRINT "MINDSTE OBSERVATION ER:";M1
0200 PRINT "STØRSTE OBSERVATION ER:";M2
```

De variable MAX og MIN betegnes her hhv. M2 og M1. I fx. PET-BASIC kan man imidlertid bruge variabelnavnene MAX og MIN, og det gør naturligvis programmet betydeligt lettere at læse og forstå. Lange variabelnavne gør det også meget lettere at ændre programmer, idet man hurtigt kan finde de steder, der har betydning for rettelsen, når man uden videre kan læse, hvad de variable betyder. I begge programmer ovenfor skrives værdierne af MAX og MIN ud sammen med de øvrige statistiske størrelser.
Hvis den BASIC, man har til rådighed, ikke tillader brugen af sammensatte sætninger, må man klare sig med betingede hopsætninger. Det gør programmeringen betydeligt vanskeligere, men hvis man går systematisk til værks, kan dette naturligvis også klares. Herunder er vist et udsnit af et BASIC-program, som udfører helt de samme udregninger, som det ovenfor. Linjerne 60 og 70 er imidlertid ændrede og linjerne 65 og 75 er tilføjet:

```
0060 IF T<=M2 THEN GOTO 0070
0065 LET M2=T
0070 IF T>=M1 THEN GOTO 0080
0075 LET M1=T
0080 LET S=S+T
```

Lad os se nærmere på linjerne 60 og 65: Hvis den indtastede observation T er **mindre end eller lig med** M2 (MAX), udføres hopordren GOTO 70, med det resultat, at sætningen LET M2=T **ikke** udføres. Hvis derimod T er **større end** M2, udføres hopordren ikke, men udførelsen fortsætter med linje 65, hvor tildelingen LET M2=T udføres. Tildelingen LET M2=T udføres altså netop **når** T er større end M2, hvilket også er korrekt. Læg godt mærke til, at det Boolske udtryk, der anvendes i denne forbindelse, er **negationen** (det modsatte) af det, der anvendes i de foregående programmer. Hvis man er nødt til at programmere på denne måde, er det klogt først at udtrykke sætningen, som det er gjort i første tilfælde og så **bagefter omskrive** ved hjælp af hopsætninger. Skematisk kan forholdet udtrykkes således:

IF p THEN "sætning"

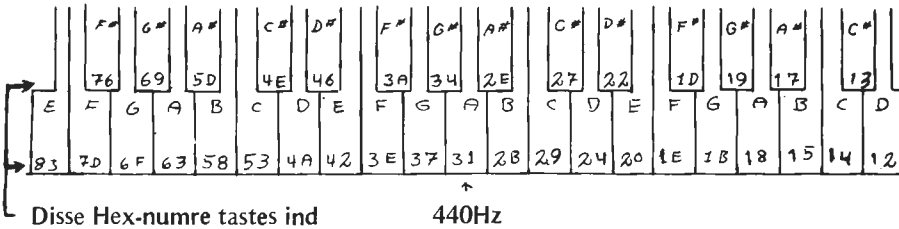
omskrives til (linjenumrene er tilfældigt valgt):

50 IF not p THEN GOTO 70
60 "sætning"
70 "næste sætning"

I begge tilfælde skal p betyde et Boolsk udtryk og "sætning " er naturligvis een **eller** anden BASIC-sætning. Udtrykket "not p" betyder **negationen af p**, altså den modsatte betingelse til p. Sammenhængen kan også udtrykkes således: Hvis **ikke betingelsen er opfyldt**, hopper vi **udenom den sætning**, hvis udførelse er underkastet betingelsen.

Program: MUSIC

NODE-KODE KORT



Hermed sender jeg jer en lille program for mek6800D2, som jeg har tyvstjålet fra Motorolas M6800 User Group Library. Programtitlen er "Music" og det er nr. 73 i M6800 UGL.
Programmet kan få D2 kittet til at spille sang sekvenser som man loader efter et node-kodekort.
Outputfrekvenserne er "stemt" i forhold til et almindeligt klaver, så den lille hjemmemusiker kan klimpre med.
Programmet starter ved at man blot taster G. Melodien spiller da 7 gange, men for hver gang ændres den harmoniske struktur, indtil 7'ende gang hvor tre harmoniske på en gang gør det til en lyst at lytte til. Herefter går der en tid svarende til sangens varighed, hvorefter det hele begynder forfra.
NB. Sangen kan stoppes, men KUN ved at benytte Reset knappen på MPU-Modulet og IKKE "E" – G starter den igen.
Outputtet fås ved at forbinde 3 stk. 10 K modstande til de tre mindst betydende bits af PIA port A (kantkonnektor J1, ben H, J og K). De tre modsatte ender klaskes sammen og herfra tages signalet til en LF-forstærker. Stel kan både være + eller –.

```
0000  7F 8005  CLR   CLEAR CRA-2 BIT
0003  7C 8004  INC   INCR DATA DIRECTION REG
0006  73 8005  COM   SET CRA-2 BIT
0009  8E 0025  LDS   POINT TO FIRST NOTE -1
000C  CE 08FF  LDX   TIME PER NOTE
000F  33       PULB  PULL NEXT NOTE FROM STACK
0010  5D       TSTB  HAS LAST NOTE BEEN PLAYED?
0011  27 ED    BEQ   IFNOT, CONTINUE
0013  F7 0025  STAB  STORE NOTE-CODE
0016  4C       INCA  TOGGLES OUTPUT WHEN STORED
0017  F6 0025  LDAB  GET NOTE CODE
001A  09       DEX   HOLD TONE FOR A WHILE
001B  27 EF    BEQ   LONG ENOUGH YET?
001D  5A       DECB
001E  26 FA    BNE
0020  B7 8004  STAA
0023  20 F1    BRA
0025  01       RMB
0026
```

Her kommer alle tonerne som kan fylde resten af hukommelsen.

KIM-1 mikrodatamat

Rockwell har nu gjort det muligt for dataamatører verden over, at erhverve sig en komplet, færdigsamlet og afprøvet KIM-1 mikrodatamat til kun

kr. 2150,- incl moms

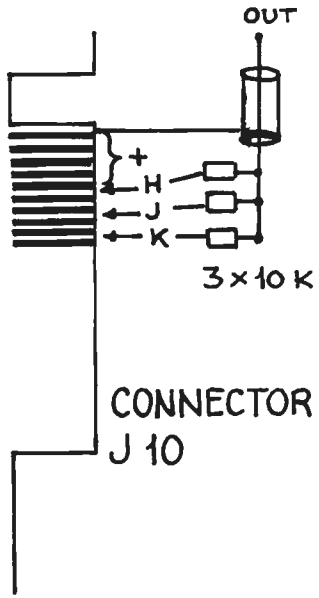
Rockwell International

I denne pris er indeholdt:

- 1 stk KIM-1 Monteret i fiks PVC-BOKS
- 1 stk KIM-1 User Manual
- 1 stk KIM-1 Referencekort
- 1 stk KIM-1 Diagram
- 1 stk 44 Pins kant connector

Rekvirer yderligere information hos:

MICROMOR ApS
Tindbjergvej 16 · DK-8600 Silkeborg
Tlf. (06) - 83 60 08



Hastigheden kan ændres ved at ændre indholdet af adresse 000D—000E. Mindre data giver større hastighed og vice versa.
Her er to programeksempler, det først er temaet fra Dr. Zhivago, det andet er Fur Elise.
Zhivago temaet viser hvordan der kan opnås en slags akkordeffekt ved at skifte mellem to harmonerende toner.

```
0026  53,42,53,42,53,42,53,42,53,42,53,42,37,42,37,42,37,42,37,42,37,42,
003A  22,2B,22,2B,20,29,20,29,20,29,20,29,20,29,20,29,20,29,20,29,20,29,
004E  20,29,20,29,20,29,20,29,2B,37,2B,37,24,2B,24,2B,29,37,29,37,
0062  42,37,42,37,42,37,42,37,3A,46,3A,46,3E,4A,3E,4A,3E,4A,3E,4A,3E,4A,
0076  3E,4A,3E,4A,3E,4A,3E,4A,3E,4A,3E,4A,3E,4A,3E,4A,3E,58,3E,58,
008A  3E,58,3E,58,3E,58,3E,58,37,4A,37,4A,37,4A,37,4A,31,3E,31,3E,
009E  2B,37,2B,37,2B,37,2B,37,2B,37,2B,37,2B,37,2B,37,2B,37,2B,37,
00B2  2B,37,2B,37,31,3E,31,3E,37,42,37,42,3A,46,3A,46,3E,4A,3E,4A,
00C6  3E,4A,3E,4A,24,3E,24,3E,29,42,29,42,29,42,29,42,29,42,29,42,
00DA  29,42,29,42,2B,31,37,3E,42,4A,00.
```

Fur Elise. OBS. 000D ændres til 2000.

```
0026  20,22,20,22,20,2B,24,29,31,63,53,42,31,2B,83,69,58,42,2B,29,
003A  83,63,53,42,31,29,20,22,20,22,20,2B,24,29,31,63,53,42,31,2B,
004L  83,69,58,42,2B,29,2B,31,63,53,42,31,2B,29,24,20,6F,53,42,37,
0062  2C,1E,20,24,6F,58,4A,37,24,20,24,29,83,63,53,42,29,24,29,2B,
0076  83,83,69,69,58,58,42,42,34,34,2B,2B,20,20,20,20,20,00.
```

Som det ses skal man altid slutte med 00 efter en melodi. JH

DET KUNNE JEG TÆNKE MIG AT LÆSE MERE OM:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

KLIP LANGS DE FULDT OPTRUKNE STREGER
SENDES SOM BREVKORT, HUSK PORTO

Giv dette brevkort til en ven, som gerne selv vil have sit eget eksemplar af Håndbog for Datamat-amatører, eller brug det til meddelelser angående flytning.

HFD udkommer med 11 numre om året, og et abonnement kan tegnes når som helst og med start fra valgfri måned. Det anbefales dog, at der tegnes abonnement fra bladets start, da den specielle opbygning bedst udnyttes, hvis alle numre haves. 1. nummer udsendt er nr. 9/1977. Abonnement koster idag kr. 100,– for en årgang og inkluderer 11 numre af HFD, 1 praktisk og solidt ringbind til en hel årgang, porto og moms.

Undertegnede bestiller herved et abonnement på HFD for 1 år i henhold til ovenstående for kr. 100,–.
Jeg ønsker at abonnementet starter med nr. . . . . . .

☐ Beløbet, kr. 100,–, vedlægges i check.

☐ I bedes fremsende girokort.

Undertegnede ønsker at meddele adresseforandring på mit abonnement på HFD.

Navn: . . . . . .

Gl. gadeadresse: . . . . . .

Gl. postnr. og by: . . . . . .

Ny gadeadresse: . . . . . .

Nyt postnr. og by: . . . . . .

Ovenstående adresseændring træder i kraft d. . . . . .

Andet: . . . . . .

. . . . . .

BREV

Porto
120
øre

Husk afsender

Til:

Telepress ApS

Greve Strandvej 42
2670 Greve Strand

KLIP LANGS STREGERNE HELT TIL BLADETS KANT

BREV

Porto
120
øre

Husk afsender

Til:

Telepress ApS

Greve Strandvej 42
2670 Greve Strand