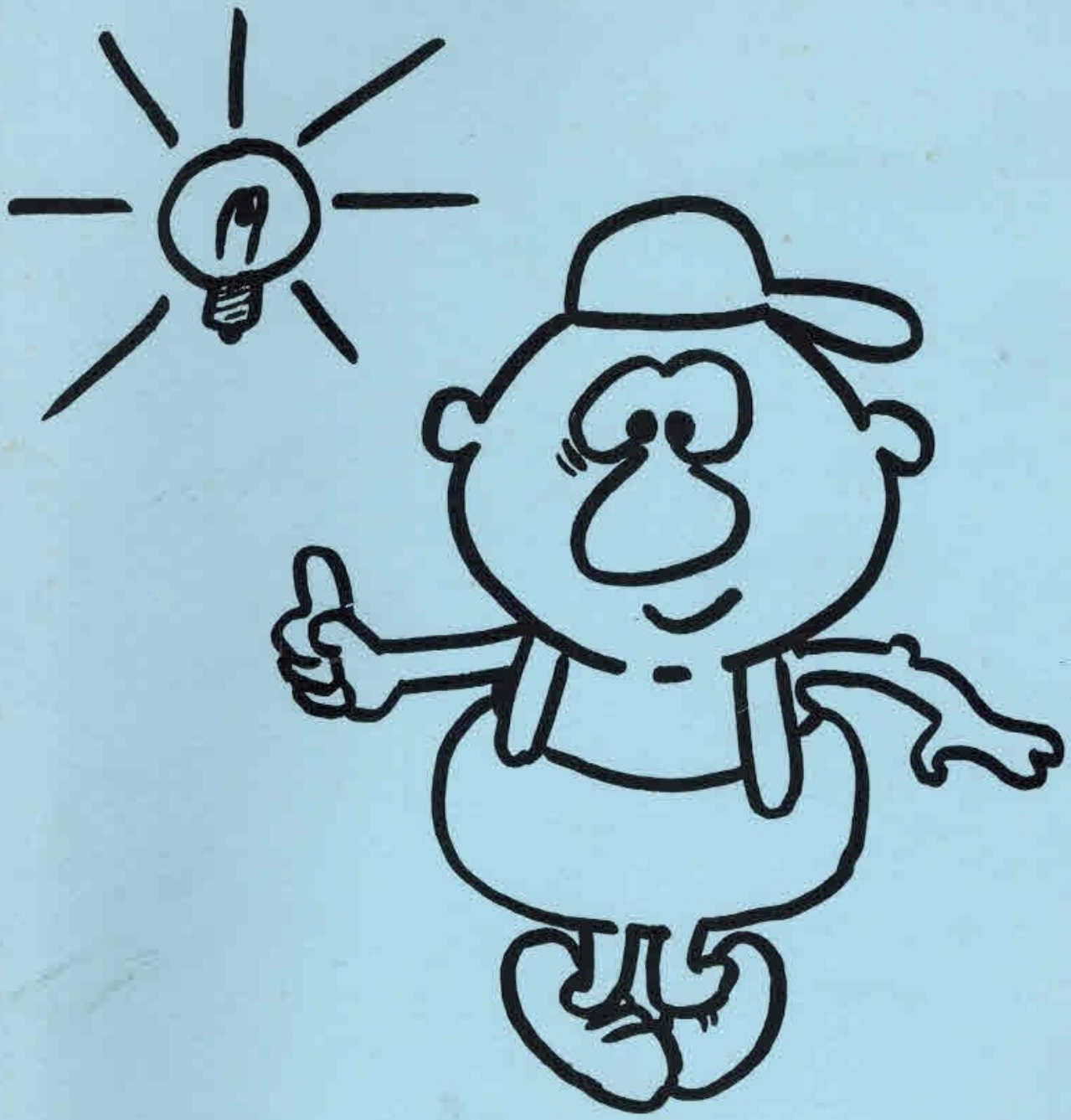
SVI Computer Club

Computer Club MSX



Club–Zeitschrift

4 - 1991

Inhalt

Hallo Herbert...

Ja,ja, ich weiß! Das Heft hätte schon etwas eher da sein müssen, aber es gab bei mir einige private Dinge, die im Vordergrund standen. Von daher also erstmal vorab:

Frohe Weihnachten nachträglich und willkommen im neuen Jahr!

Nun, anscheinend hat mein Aufruf im letzten Heft doch ein wenig genützt, den es sind doch einige Artikel mehr gekommen als davor für's Heft.

Folgender Leserbrief von Udo Bardenhagen erreichte mich:

> Hallo Herbert!
>
> Als ich vor kurzem die Club-Zeitschrift 3/91 erhalten habe, erschreckte mich dies bis ins Mark. Sage-und-Schreib 2,2 mm dick!
> So kann das nicht weitergehen und darum habe ich beschlossen auch meinen Teil zum Heft beizutragen. Ich bin nämlich der Meinung das der Witz, Comic-mäßig gesehen, zu kurz kommt.
> Dem will ich durch folgende Sachen abhelfen:
>
> Gedacht hatte ich an Comicstrips wie man sie auch ein Zeitungen findet. Also ein Kurzcomic der vielleicht aus drei oder vier Bildern besteht und eine halbe bis ganze Seite der Clubzeitschrift benötigt.
> Bei Interesse könnte man auch einen vollständigen Comic auf die Beine stellen der über mehrere Seiten geht.
>
> Gedacht hatte ich an Titel wie
> - Die Abenteuer von Billy the Chip
> - Memorien eines Megaroms
> - Chippy- der 8-Bitter
> oder ähnliches.
> (...)

Nun, warum nicht auch mal einen Comic-Strip bei uns im Heft. Kleine Karikaturen haben wir ja schon seit langem im Heft. Sehen wir mal, wie das im Club ankommt. Also, ab Heft 1/92 gibts dann einen Comic!

Wenn Ihr mal nicht wißt, worüber ihr Beiträge an das Clubheft schicken könnt, so schaut doch mal in der Ideen-Ecke nach.

In der Hoffnung auf viele Artikel verbleibe ich bis zum nächsten mal...

Herbert Kloseck

(Herbert Kloseck)

MSX1 MSX2 MSX1 MSX2 MSX1 MSX2 MSX1 MSX2 MSX1 MSX2 MSX1 MSX2

MSX—News

Genaueres zu H.S.H.:

(Zitat)
Die Firma HSH Computervertrieb GmbH hat am 19.09.1991 beim Amtsgericht in 4710 Lüdinghausen das Konkursverfahren unter der Konkursnummer 2N13/91 eingeleitet. .....

Falls Sie Forderungen an HSH Computervertrieb GmbH haben, bitten wir diese an des Amtsgericht in Lüdinghausen weiter zuleiten.
(Zitatende)

Die japanische MSX-Zeitschrift MSX-FAN ist seit Oktober mit einer Diskette ausgestattet, auf der sich die Programme zur Zeitschrift befinden. Der MSX-FAN kostet jetzt 950 Yen.

In Japan hat Panasonic jetzt seinen neusten MSX-Computer vorgestellt:
Panasonic A1GT MSX turbo R
Das besondere an diesen MSX ist:
512K RAM, MIDI-Interface eingebaut, 32K SRAM, 2032K ROM (Es gibt eine Art ROM-Disk, auf der sich MSXDOS2 und MSXViev (japanische Benutzeroberfläche) befinden als Laufwerk C:, das SRAM steht als Laufwerk D: und eine RAM-Disk als Laufwerk H: zur Verfügung.) Der Preis beträgt 99.800 Yen!

Neues von Ulrich Koppe

Genau am dritten Advent klingelte bei mir das Telefon und wer war am anderen Ende der Leitung? Ulrich Koppe! Nun also hier die aktuellsten Infos über Ulrich:

Seit im Frühjahr 1991 sein Computer ausfiel, hatte er noch keine Gelegenheit, ihn zu reparieren, da er im Studiumstress hing. Jetzt in der Weihnachtszeit wolle er ihn aber wieder in die Gänge kriegen.

Ausserdem stapeln sich bei ihm haufenweise Artikel für die Clubzeitschrift, so das ab dem Heft 1/92 endlich wieder ein SVI-Teil hier im Heft sein wird!

Also gute SVI-Aussichten für das nächste Jahr!

(Herbert)

MSX2 MSX2 MSX2 MSX2 MSX2 MSX2 MSX2 MSX2 MSX2 MSX2 MSX2 MSX2

DOSSCAN

Ein Diskettenscanner für MSXDOS2

Dieses MSXDOS2-Programm testet DOS-Volumes auf schadhafte Sektoren. Hier erst mal die Anleitung:

Aufruf: DOSSCAN [Drive:] [/T] [/W] [/?!/H]

/T schaltet die Sektorenanzeige aus, das erhöht den Datendurchsatz.
/W ermöglicht einen Diskettenwechsel. Sehr sinnvoll bei Festplatten und Ramdisks.
/? bzw. /H gibt einen kurzen Hilfstext aus.

Um den Scanvorgang überhaupt in die Gänge zu kriegen, muß der Datenträger einen lesbaren Bootsector mit einem korrekten Diskparameterblock besitzen!

Ein Protokollieren auf dem Drucker ist mittels IO-Redirection möglich (Bsp. DOSSCAN A: > PRN). Die Environmentvariable REDIR muß natürlich auf ON stehen. Nicht möglich sind alle Ausgabeumleitungen, die Dateioperationen erfordern. Das hängt mit dem DOS-Internen Errorhandling zusammen. Wer es trotzdem versucht, bekommt die Meldung "Invalid Parameter" in seine Ausgabedatei geschrieben und das Programm bricht ab.

Natürlich werden weder Bildschirmsteuerzeichen noch die laufende Sektorenanzeige umgeleitet. Diese Ausgaben laufen über den STDERR Kanal.

Wenn das Programm läuft und es treten Lesefehler auf wird die Nummer des schadhaften Sektors und die Fehlerursache ausgegeben. Falls man das Drama leid ist, mittels Druck auf die ESCAPE-Taste kann das Programm vorzeitig abgebrochen werden.

Wenn der Scanvorgang zu Ende ist, werden zum Schluß die Anzahl der Fehler und wenn keiner auftrat die Datenrate in KB/Sec ausgeben. Der letztere Wert kann bei der Suche nach dem optimalen Interleave für die Festplatte eine Hilfe sein. Als Zeitbasis dient die Uhr im Rechner, damit ist eine weitestgehende Unabhängigkeit von der Taktrate gegeben. Leider löst die rechnereigene Uhr nur im Sekundenbereich auf, somit ist ein Meßfehler von bis zu zwei Sekunden möglich. Bei dicht voneinander abweichenden Werten sind mehrere Messungen empfehlenswert falls man die "genaue" Tranferrate wissen möchte.

Festplatten und Ramdiskbesitzer (>256 KB) haben hiermit ferner die Möglichkeit den Einfluß von aktivierten FM-PAC oder 60Hz Bildwiederholfrequenz, Speichererweiterungen etc. auf den Datendurchsatz mal selbst nachzumessen, Uwe und mir glaubt ja keiner ;-).

Wer eine eigene Shell installiert hat, wird es sicher dankbar zur Kenntnis nehmen, daß DOSSCAN immer einen Exitcode liefert: 0 = Alles klar und 1 = Programm fand fehlerhafte Sektoren oder wurde abgebrochen.

Jetzt wird der wißbegierige Leser sicher das Listing (ist natürlich wieder Assembler :-(suchen, aufgrund der "starken" Beteiligung an dem Assemblerforum (in knapp einem Jahr zwei Briefe und fünf mündliche Anfragen wann der "Kurs"

denn weitergeht, obwohl niemals von einem MC-Kurs die Rede war!) konnte ich mich leider nicht durchringen dafür Platz im Heft oder auf den Clubdisketten zu verschwenden. Wer mir gegenüber genügend Sachkunde und Interesse nachweist, kann das Listing gegen Einsendung einer formatierten Leerdiskette (3½") und ausreichenden Rückporto direkt bei mir (Rolf) bekommen.

GEN80 an MSXDOS2 angepaßt

HiSoft's DEVPAC wird MSXDOS2 tauglich

Es ist schon schon etwas länger her, als ich das DEVPAC an MSXDOS angepaßt habe. Jetzt haben es auch die Holländer entdeckt (Testbericht in der MCM Heft Nr. 49 11/91). Grund genug sich mal die Lauffähigkeit unter MSXDOS2 genauer anzusehen, und prompt wurde ich fündig:
Wer seinen Assembler in einem seperaten Directory, getrennt von den Sourcen, untergebracht hat und mittels SET APPEND=C:\ASM\SOURCEN alles klar gemacht hat, wird sich beim Aufruf des GEN80 ganz schön wundern:"Wrong disk for File..." ziert den Bildschirm. Was ist passiert?

Ganz klar, ein durchtracen mit dem Debugger bringt es an das Licht, der Assembler fummelt permanent an den Laufwerkseinträgen der geöffneten FCB's herum. Das kann man dem GEN80 aber mittels einiger wohlplatzierter NOP's abgewöhnen und schon ist Ruhe. Und weil ich gerade so schön dabei war, habe ich ferner dem Wunsch eines unserer Leute erfüllt und den Dateinamen des Sourcefiles im Kopf des PRN-Files untergebracht. Das Datum wird natürlich auch noch ausgegeben.

Der Patch ist so ausgelegt, daß sowohl noch ungepatchte als auch bereits mit dem Patch aus dem Heft 2/90 bearbeitete Versionen des GEN80 bearbeitet werden können. Natürlich sollten die Patchversuche nur auf Kopien und keinesfalls auf den Orginaldisketten stattfinden!

Es existieren zwei Dateien auf der Clubdiskette: GEN80PAT.HEX ist als Eingabedatei für MLOAD vorgesehen. GEN80PAT.REL ist für Uwes RPATCH gedacht. Und wer sich mit einem Debugger versuchen will, die Source des Patches sollte auch dabei sein.

Allerdings sollte man keine Wunder von dem Patch erwarten, der Assembler macht doch einige merkwürdige Dateioperationen die ich nicht mit einem Patch geradebiegen konnte. So wird man sich an die Fehlermeldung "Wrong disk reading File" gewöhnen müssen, wenn man dem Assembler beauftragt hat ein PRN-Listing zu erzeugen und die Datei nicht zu finden war.

So ich hoffe, daß die relativ kurze Beschreibung und das Listing ausreichend sind, wenn nicht kann man mich auch fragen.

Computer Bestsellerliste

(für die Nachwelt festgehalten von Lasse Assebasse)

Erstmalig und auf Wunsch vieler Leser auch letztmalig hier der Verkaufsspiegel der Home- und Personal-Computer. Ich habe Kosten und Mühe gescheut, statistisches Material für eine solide Bilanz zusammengetragen, den ganzen Report graphisch ausgewertet und schließlich in der U-Bahn vergessen.

Gottseidank halfen mir meine excellenten Beziehungen zur Wirtschaft weiter. Dort traf ich nämlich Ernst G. Meint, seines Zeichens Aushilfskraft allererster Ordnung im Computerladen um die Ecke.

Für zwei Bier und einen klaren erstellte er mir eine Bestsellerliste der Home- und Personal-Computer. Für eine Currywurst sprach er sogar, leider nur mit vollem Mund, über die Hintergründe dieser Reihenfolge.

Homecomputer:	Personalcomputer:
1. Philips VG 8235	1. Commodore PC 10
2. Commodore C 64	2. IBM PC-AT
3. Sinclair ZX 81	3. Apple IIe
4. Philips VG 8235	4. Apple Macintosh
5. Atari 130 XE	5. Atari 520 ST

Hier nun die exakte Analyse von Ernst G. Meint. Sollten Sie zwischen den Zeilen Ketchup oder Wurststücke finden, so denken Sie bitte an die erschwerten Umstände unter denen diese Marktanalyse gemacht wurde. Ansonsten gibt es dem messerscharf sezierendem Verstand des stadtbekannten Wirtschaftsexperten nichts hinzuzufügen, deshalb jetzt Original- (und was für eins) zitat:

Ei bei dene Homkombjuder steht natürlich de PHILIPS gans vorne. Die Leut wolle ja gleich was sehe wenns se de Tür reinkomme, außerdem hammer hinne kaan Bladz mehr. In dem 8235 losse mer de ganse Tach die Demodiskett laafe, do hott die Kunnschaft was zum Gugge unn geht aam nitt mit bleede Frache uff de Wecker.

Der COMMODORE steht uff Bladz zwaa, weil mern do gut sehe dut unn dess muß mer nämlich, weil mer habbe uff de Monitor "Tür zu" geschribbe. Im Winter werds bei uns so forchtbar kalt und unsereens steht jo nitt im Mantel do wie die Kunnschaft. Im IBM-Laade habbe se Tebbichbode aber unser Scheff maant .... Noja, lasse mers Thema.

Der aale ZX 81 iss im Verkauf nochemaa uff Bladz drei gekomme, dess war son Werbegeck vom Scheff. Der hott drageschribbe:"JETZT IN PROFIBLACK" unn seitdem geht der widder weg wie nix. Also nitt de Scheff, sonnern de Kombjuder. Der waas scho wie mer Geld macht, de Scheff nitt de Kombjuder.

Daß der PHILIPS uff Bladz vier nochemaa in de Bestsellerlist iss, also zwaa maa, dess iss kaa Versehe, sonnern, weil den hammer zwaa maa verkaaft in der Woch jetzt.

Der ATARI hunnertdreisisch iss uff de finfte Bladz komme, weil ich den zuerst für de Finfhunnertzwansischer Estee gehalte habb. Als ich dess gemerkt habb mit dem Fehler wollt ich aach nimmer dran rumfummele.

Bei de Bersonalkombjuder sieht dess alles widder gans anners aus. De aansische der sich do richtisch auskenne tut is dem Scheff sein Kombangjong, aber der issim Urlaub unn an mir bleibt widder de gense Dreck hänge.

Also uffem erste Bladz iss de COMMODORE PC 10. Die Leut kaafe halt immer de IBM unn wenn werklich mo aaner kimmt mit Geld dann drehn mer dem de IBM PC-AT an. Mir Eingeweihte aus de Kombjuderscene sache immer, AT heest Aaler Taschenrechner, abber nitt zu de Kunnschaft, sonnern nur zum Spaß. Wie ich den geck es erste Mol gebracht habb, hott sich dess Lisbett, die wo bei uns immer putzt, nitt mer eigekrischt vor Lache.

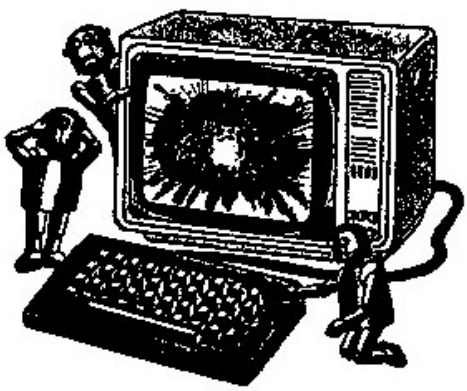
De EPPEL IIe verkaafe mer nitt, den nemme mer nur als Heizung, wenn der richtisch leeft werd der so heiß, daß ich mei Keffedippche druffstelle kann, desweche brauch ich den im Laade.

Uff Bladz vier iss der Mechintosh, der iss aach von Eppel. Als ich en zum erste Mol gesehe habb, habb ich mer gedenkt, dess is en scheene Fernseher nur die Stripp von de Fernbedienung habbe se zu korz gemacht. Abber nachher wars doch en Kombjuder. Die Kunnschaft glaabt aach des wäre en Fernseher unn mer redde des dene aach nitt aus. Wann die widderkomme unn sache se krische de Sender nit rei, dann drigge mer dene des Handbuch in die Finger unn mer habbe widder e halb Jahr Ruh. Unn wann se dann nochemaa komme is die Garantiezeit abgelaafe. Aaner hott mo umgemotzt, mer wärn en saftlaade, en richtische Epelsaftlade. Ich kann der sache, mit de Kunnschaft machste was mit.

Dess neuste was mer habbe is uff Bladz finf de ATARI Finfhunnertzwansisch. Mir persönlich gefällt jo der ZX 81 in Profiblack besser. Bei dem helle ATARI sieht mer jeden Badsche, wenn mer mo mit dreckische Finger drangeht. Außerdem hott mich mein Scheff gewaltich angeschisse, weil ich die erste Kiste fir finfhunnerzwanzisch Mark verkaaft habb, ei soller halt uffpasse, was er uff die Zettelcher schreibt.

Ansonsten iss bei uns de Kunne Könisch, jedenfalls solang mir aaner in de Kron habbe, dess war nochn Geck zum Schluss, des Lisbette lacht sich immer schlepp dabei, wenn ich dess sach.

So jetzt geh ich abber. Gebb mer noch e Pfefferminzbombom, mein Scheff glaabt ich wär beim Zahnarzt gewese.



MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$

MCBC — MSX Club Basic Compier

Hallo Leute, da wir ja einen so großen Artikelnotstand in unserem Club haben, habe ich mich auch mal wieder aufgerafft. Also dann:

Täterätätä !!

Es gibt einen neuen Basic Compiler ! Bedauerlicherweise nicht von uns, sondern von einem Holländischem Club. Ist ja auch egal, denn auf das Ergebniss kommt es an. Die neueste Ausgabe dieses Compilers trägt den Namen MCBC Version 2.1. Sie ist erst vor kurzem herausgegeben worden, und ist eine völlig überarbeitete Version. Wie Ihr sicherer schon bemerkt habt, (aufmerksam wie ihr den Artikel lest) gab es auch eine erste Version, nämlich die Version 1.1. Ärgerlicherweise konnte ich beide Versionen nicht auführlich testen, da mir das Geld fehlt um mir den Compiler zuzulegen. Doch !! Ich konnte mir die erste Version einmal bei einem Bekannten vorführen lassen. Auf der Diskette befinden sich neben dem eigentlichen Compiler noch der sogenannte Controller, ein Loader und etliche Demoprogramme. Nachdem ich mir die Demoprogramme habe zeigen lassen, war ich über die Geschwindigkeit echt beeindruckt. Aber auch das Programmieren, die Handhabung des Compilers und die Menge der compilierbaren Basicbefehle ist echt Klasse.

1.) Die Geschwindigkeit habe ich mit folgendem Programm getestet:

```
10 time = 0                                        Normal Basic    => Time= 143
20 print s                                         Compiled Basic  => Time= 9
30 for x=1 to 3000        und als 2.Vergleich der XBasic Compiler  => Time= 11
40 next
50 print E
60 Print time
```

2.) Die Programmierung ist sehr einfach: Normal in Basic programmieren. Bedauerlicherweise kenne ich nicht alle erlaubten Befehle, es sind auf jedenfall ein ganzen Haufen mehr als der XBasic Compiler kann. Bei der Version 2.1 sind die Befehle:

BEEP	BIN$(X)
CSRLIN	DATA
DEF USR x	DSKF(X)
DSKI$(X)	DSKO$X)
END	HEX$(X)
INSTR$(X,F$,H$)	LOCATE x,y,c
POS(x)	READ
RESTORE	SPACE$(n)
STRING$(c,c)	STRING$(x,a$)
SWAP a,b	VAL(t$)
WIDTH x	

außerdem noch dazu gekommen.

3.) Die Handhabung: Nachdem man sein Basicprogramm fertig geschrieben und abgespeichert hat, lädt man den Controller, der sich dann so präsentiert:

MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$ MSX$_2$

CONTROL

1 Load Compiler	(Eigentlichen Compiler laden)
2 Compile Program	(Basic-Programm laden + compilieren)
3 Report on Compilation	(Zeigt Start-,End- und Variablenadresse)
4 Save Compiled Program	(Compiliertes Programm als *.mem speichern)
5 Execute Compiled Program	(Compiliertes Program im Speicher ausführen)
6 Exit Control	(Ende Controler)

Präsentiert sich der Controler wie oben gezeigt, geht es folgendermaßen weiter:
Als erstes lädt man den Compiler. Dann drückt man auf die Taste 2,und man wird aufgefordert sein Basicprogramm zu laden. Nachdem dies geschehen ist, wird "_compile" eingegeben, und das Basicprogramm wird compiliert. Eventuelle Fehler werden sofort angezeigt. Man kann sie probemlos solange ausgebessern und neu Compilieren, bis der Compiler ein "ok" anzeigt. Ist das der Fall, dann lädt man erneut den Controler. Am sinnvollsten ist es, sein compiliertes Programm mit der Taste 4 zu speichern. Jetzt steht es einem offen, welchen der obengenannten Punkte man dann ausführt. Um sich das Ergebniss seines compilierten Programmes anzusehen, lädt man sich den Loader. Ist dieser geladen, listet man ihn auf und setzt an den dafür vorgesehenen Stellen den Namen des neuen Ladeprogramms, z.B. Test.ldr, und den Namen des abgespeicherten compilierten Programms z.B. Test.mem ein. Jetzt, den so veränderten Loader abspeichern =>save"Test.ldr", und dann laufenlassen.
Fertig !!

Wer sich jetzt für diesen Basic Compiler interessiert,kann ihn für den Preis von 85 Gulden unter folgender Adresse bestellen:

G. Willemse
Eurovisieplein 42
3402 GE IJsselstein
Tel.: Holland 03408/85634

Falls jemand zufällig schon die erste Version haben sollte: Es gibt auch ein Update für 27,50 Gulden unter der selben Adresse.

Also dann bis zum nächsten mal

Oliver Schiefke

DFU-Info Modembetrieb mit Hayes-Befehle

Der Modembetrieb ist fuer einen Anfaenger recht schwierig, vor allem wenn das Handbuch in Englisch geschrieben ist. Dazu kommt noch, dass die Software/Hardwarebeschreibungen von anderen Herstellern geschrieben sind.

Wenn das Modem richtig angeschlossen ist,wird ein Terminalprogramm benoetigt.

Fuer MSX/SVI gibt es unterschiedliche Terminalprogramme:

* z.B. im Philips-Interface ist die Sofware im RS232-Modul, das Modem kann vom BASIC angesprochen werden.
* dann gibt es eine geaenderte HiBrid-Version (nur nms1250), dort kann das Modem von der Benutzeroberflaeche aus arbeiten
* und das Programm MSXCOM fuer die simple RS232-Schnittstelle; (gibt es auch fuer PC's)
* das wichtigste Programm ist aber wohl MEX mit einem ANSI-Treiber (Alle Mailboxen bieten Ansi-Darstellung an).

Ein Modem arbeitet in 2 versch. Betriebsarten: Kommandomodus or. Datenmodus (<= DFUE-Betrieb)

Im Kommandomodus kann man das Modem konfigurieren, um den Gebrauch zu vereinfachen. Dabei wird der Hayes-Befehlssatz verwendet (Modemsprache). Die meisten Modems haben einen NVRAM, wo man seine Einstellungen ablegen kann (nicht fluechtiger Speicher).

Alle Befehle werden mit einem AT (attention=aufgepasst) eingeleitet. Um eine Aenderung vorzunehmen, muss man dem Modem erst AT&W (or AT&W2) angeben. Das Modem antwortet mit einem 'OK' (Prompt wie im Basic).

Herzlich eingeladen sind

alle MSX/SVI-Clubmitglieder,

sowie alle Intressierten User.

Wir (der MSX/SVI-Computerclub)

warten auf alle Interessierten und

stehen Euch dort Rat und Tat bereit.

INTERNATIONLE
MSX-MESSE
in
TILBURG
Samstag, den 4. April 1992
von 10.00 bis 17.00 Uhr
in der
BREMHORSTHAL
Oude Goirleseweg 167
Eintritt: fl 7,50 pro Person
(ca. 7,- DM)

HAYES Befehlssatz

kompl. mit erweiterterten Befehlsatz und NMP-Codes (bis NMP5)

+++

Einer der wenigen Befehle, der nicht mit AT beginnt.Im Datenmodus wird unterbrochen und auf Komandomodus umgeschaltet.(Manchmal der letzte Ausweg um den DFUE-Betrieb anzuhalten und mit ATH0 den Hoehrer aufzulegen.)

A/

Einer der wenigen Befehle, der nicht mit AT beginnt. Er wiederholt den letzten Befehl. Achtung, es darf kein <RETURN> gedrueckt werden.

AT

Achtung. Parameter aus NVRAM einlesen (Baudrate nachstellen).

ATA

Manuelle Rufbeanwortung. Das Modem sendet einen Antwortton aus. Das Modem nimmt den Anruf entgegen, der gerade ansteht. Sollte kein Modem anrufen, wird der Anrufer mit einem netten Pfeiffton begruesst.

ATB 0 (Die '0' kann entfallen wie das Leerzeichen ' ')

CCITT V.21 einschalten (Handshake bei 2400 bps)

ATB 1

CCITT V.22 ausschalten (Bell-Standard bei 300-1200 bps)

ATD v www x

Waehlen einer Telefonnummer.

```
v    = P or. T ==> Pulswahlverfahren or. Tonwahlverfahren
wwww =   Telefonnummer (nur Modems erlaubt!)
         folgende Zeichen werden ignoriert: -; /; ' '; ()
x    = W wartet auf Freizeichen vorm Waehlen.
x    = R Rueckwaertsmode, Sendet einen Carrier u. geht in den
         Antwortmodus nach dem Waehlvorgang.
x    = @ wartet auf Carrier (Zeit in S7)
x    = ! Wartezeit vor dem Modemkontakt. Ein '!' = 0.5 sec.
         Mit ATD 0201-283986 !!!!!!!!!!!!!!!!!!!! kann man sich noch
         10 sec unterhalten,dann erst geht das Modem ans Netz.
```

DFUe HAYES DFUe HAYES DFUe HAYES DFUe HAYES DFUe

ATE 0

Echomode ausschalten (Halbduplex)

ATE 1 (*)

Echomode einschalten (Vollduplex)

ATH 0

Verbindung trennen (wie Tel.hoehrer auflegen)

ATH 1

Verbindung vorbereiten (wie Tel.hoehrer abheben)

ATI x

Kennung:

0 gibt die Nummer des Chipsatzes 246,...,249
1 gibt eine Checksumme aus
2 gibt ein OK aus (sonst Modem defekt)

ATL x

Lautstaerke zum Mithoeren:
0 nichts mehr zu hoeren
1 leise
2 mittel
3 laut

ATM x

Akustische Kontrolle:
0 ausgeschaltet
1 eingeschaltet bis Verbindung aufgebaut
2 immer eingeschaltet
3 nach dem Waehlvorgang eingeschaltet

ATO (<== das ist ein gr. Oh!)

Online schalten, nach einer Unterbrechung mit +++ in den Datenmodus zurueckkehren.

ATQ x

Antworten auf Befehle:
0 ein
1 aus (Quiet Command «=» ohne Antwort)

ATS x ?

Registerwert von x (0-255) ausgeben

DFUe HAYES DFUe HAYES DFUe HAYES DFUe HAYES DFUe

ATS x = y

Register x mit dem Wert y (0-255) laden (nur wenn vorher AT&W bzw ATW0 gegeben wurde).

ATV x

Modemmeldungen: 0 = numerisch; 1 = in Textform

0	OK
1	CONNECT
2	RING
3	NO CARRIER
4	ERROR
5	CONNECT 1200
6	NO DIAL TONE
7	BUSY
8	NO ANSWER
9	CONNECT 600
10	CONNECT 2400
85	NO DATA COMPRESSION
86	CLASS 5 COMPRESSION
97	SELECTED SPEED UNAVAILABLE
98	NO ERROR CONTROL (kein NMP)
99	ERROR CONTROL (NMP4,3,2,1)

ATX x

Verbindungsaufbau:
- 0 simple Meldung (CONNECT)
- 1 ausfuehrl. Meldung (CONNECT 1200 bps)
- 2 ausfuehrl. Meldung und warten auf ein Freizeichen vor dem Waehlen.
- 3 ausfuehrl. Meldung und bei besetztem Anschluss wird abgebrochen.
- 4 Kombination von 2 und 3

ATZ x

Reset Modem. Die Parameter werden aus dem NVRAM ausgelesen.
(x kann 0-3 sein, geht aber nicht bei allen Modems .)

AT&C x

DCD Traegererkennung:
- 0 ist immer aktiv (Carrier-Detect)
- 1 ist nur aktiv, wenn eine Verbindung besteht

AT&D x

DTR Endgeraet betriebsbereit:
- 0 DTR vom Computer ignoriert das Modem
- 1 bei DTR 0 geht das Modem in den Komandomodus, Verbindung bleibt.
- 2 bei DTR 0 geht das Modem in den Komandomodus und trennt die Verb.
- 3 wie 2 jedoch mit RESET Modem.

DFUe HAYES DFUe HAYES DFUe HAYES DFUe HAYES DFUe

AT&F

Alle Parameter werden geloescht und die Herstellereinstellung zurueckgesetzt. NVRAM-Daten bleiben erhalten. Mit ATZ werden sie wieder aktiviert.

AT&G x

Antworton : x = 0 abgeschaltet (bei ATB0 !)
x = 1 550 Hz (nur bei ATB1 Echo-Sperre
x = 2 1800 Hz der Post aufheben ????)

AT&J x

Externes Busy-Signal: 0 abgeschaltet (Standart RJ-12_Stecker)
1 Busy-Signal aktiv (zusaetzliche Leitung??)

AT&L x

Telefonverbindung: 0 normale Telefonverbindung
1 priv. 2-adrige Leitung (Nebenstellanlage)

AT&M x

Synchron- Asynchronbetrieb 0 Asynchronbetrieb (Standard)
1 Synchronbetrieb bis DTR low
2 Synchronbetrieb bis DTR low wird mit Rueckruf der &Z0 Telefonnr.
3 Asynchronbetrieb beim Anwaehlen dann wird zum Synchronbetrieb umgeschaltet. (was weiss ich ??)

AT&P x

Impulszeitverhaeltnis beim ATDP: x=0 39% Impulszeit (US-Standart)
x=1 33% Impulszeit (GB-Standart)

AT&R x

RTS/CTS Option (Sendebereitschaft). Der Befehl steuert das CTS-Signal im synchronen Betrieb. CTS wird im asynchronen Betrieb erzwungen:

0 CTS folgt nach der Zeit von S26 wenn Modem angeschaltet.
1 Modem ignoriert RTS

(Ich tappe noch im Dustern??)

DFUe HAYES DFUe HAYES DFUe HAYES DFUe HAYES DFUe

AT&S x

DSR Control (Betriebsbereitschaft) 0 DSR immer ein
1 DSR normal

(wer kennt jemanden, der weiss was das soll ??)

AT&T x

Der Testmodus (S18=TT) 0 Ende des Testes
1 Alle Daten die zum Modem (analog) kommen, gehen wieder zurueck zum Computer.
Beisp. ATS18=0&T1 (Start test)
Das ist ein Test (Testdaten)
+++ (stopp)
OK (Modem Stoppt)
AT&T0 (Testende)
OK (Modem im Komandomodus)
2 nicht benutzt
3 Loopback (digital) eines andern Modems
4/5 Select Local or ext. Remote Modem
6 Testet Telefonnetz. zwischen 2 Modems eins steht auf AT&4 eines auf AT&5
7 kombiniert Modus 1 mit 6
8 Analoge Schleife mit Selbsttest

AT&V

Anzeige der aktuellen Konfiguration incl. des Inhalts verschiedener S-Register (bei mir ist der Befehl aber AT\S)

AT&W x

Speichern einer Konfiguration in Speicher (0-3). Das wird aktiviert mit ATZ x (Reset auf eine best. Konfiguration).

AT&X x

Taktquelle bestimmen: 0 interne Taktangabe
1 externe Taktangabe
2 Slave/Recive
(Sklaven/Empfangen, was ist das denn ??)

AT&Z x

TelefonNr. (0-3) Abspeichern (max 16 Ziffern)
Anrufen mit der abgelegten Nr. ATDS=x

AT\A x

Datenblockgroesse: 0/1/2/3 ==> 64/128/192/256 Bytes je Block

DFUe HAYES DFUe HAYES DFUe HAYES DFUe HAYES DFUe

AT\B x

Sende ein Break zum Empfaenger ?? Zeit entweder immer 0,3sec oder aber auch x/10sec, je nach Modem.

AT\C x

Puffer (200Bytes) setzen: x=0 Schaltet Puffer bei NMP-Betrieb ein
bei Normalbetrieb aus.
x=1 Puffer ein (immer).
x=2 Puffer aus.

AT\G x

Datenflusskontrolle, Computer # Modem

x = 0 ohne Xon Xoff bei Nicht-MNP-Verbindung (RTS/CTS)
x = 1 mit Xon Xoff bei Nicht-MNP-Verbindung

AT\I x

Interface-Protokoll ??

AT\J x

Baudratenwandler: x=0 Baudrate Modem/Comp = Modem/Mailbox
x=1 Baudrate Modem/Com >= Modem/Mailbox
(ist wichtig bei MNP5-Modus)

DFUe HAYES DFUe HAYES DFUe HAYES DFUe HAYES DFUe

AT\K x

Setzen der Break-Kontrolle:

x	Break vom Comp. Online NMP or Normal-Verbndg.	Break vom Comp. Online im Direktmodus	Break empfangen vom Gegenmodem Online/Normalv.	AT\B empfangen vom Computer Online/NMP-Ver.
0	in den Komandomodus, ohne Break	Kommandomodus, Break zum Gegenmodus	Puffer loeschen Break zum Computer.	Puffer loeschen Break zum Gegenmodem.
1	Puffer loeschen Break zum Gegenmodem	Break zum Gegenmodus		
2	in den Komandomodus ohne Break	Kommandomodus, Break zum Gegenmodus	Break zum Computer.	Break zum gegenmodem
3	Break zum Gegenmodem	Break zum Gegenmodem		
4	in den Komandomodus, ohne Break	Kommandomodus, Break zum Gegenmodus	Break und die Daten zum Computer	Break und die Daten zum Gegenmodem
5	Break und die Daten zum Gegenmodem	Break zum Gegenmodem		

AT\L x

Selektiert den Datenstrom x = 0 normaler Datenfluss (NMP normal)
x = 1 Daten in Bloecken (max 260??)
(ist das die Groesse von AT\Ax ??)

AT\N x

Betriebsart: x = 0 Normalmodus, Abschalten aller NMP-Modi
x = 1 ohne Flusskontrolle, ohne MNP-Funktion
x = 2 nur MNP-Protokoll zulaessig
x = 3 Automatische Erkennung, ob mit or ohne MNP.

AT\O (<== das ist keine Null)

Fordert das andere Modem, auf in MNP-Modus zu gehen, wenn AT\N3 ansteht.

AT\Q x

Datenflusskontrolle: x = 0 ohne Datenflusskontrolle
x = 1 Xon/Xoff-Flusskontrolle
(Software-Handshake)
x = 2 CTS/RTS HardwareHandshake-nur Modem
x = 3 CTS/RTS Modem und Computer

AT\S

Anzeige der aktuellen Konfiguration mit den wichtigsten S-Registern.

AT\T x

Setzt die Zeitbegrenzung in Minuten. Wenn in dieser Zeit kein Datentransfer stattfindet, schaltet das Modem ab. x = 0 ohne Zeitlimit. Wird im Direktmodus ignoriert.

AT\U

Schaltet nach Aufforderung eines fremden Modems den MNP-Modus.

AT\V x

MNP-Meldungstext: x = 0 Standart-Text Connect 1200
x = 1 NMP-Meldung ausgeben: Connect 1200/REL

AT\X x

Filtert die Blockmarker Xon/Xoff aus: 1 ein
2 aus

AT\Y

Schaltet waehrend einer Verbindung auf MNP-Protokoll um.

AT\Z

Schaltet waehrend einer Verbindung MNP-Protokoll aus.

AT%A x

Setzt den Fallbackcharacter x = 0-127. Das Modem versucht eine MNP-Verbindung aufzubauen, wenn AT\C2 gesetzt war. Typisch ist der Code 13 [CR]. Damit wird von MNP auf Normal-Modus gesetzt. Es kann jeder beliebige Code verwendet werden.

AT%C *x*

MNP5 Datenkompression 0 aus
1 an (Das arbeitet nur effektiv, wenn die Datenverbindung >Modem#Rechner< mit einer groesseren Baudrate laeuft, als die Datenverbindung zum Gegenmodem.)

AT%U

Reset des SerialportSpeed, wenn die Kommandos langsamer als 4800 bps zum Modem kommen, wenn normalerweise mit 4800 or 9600 bps zwischen Modem und Rechner benutzt werden. (Manche Modems machen das auch ohne diesen Befehl.) Alle MSX-ler koennen das wieder vergessen, da bei uns bei 2400 bps das Ende erreicht ist.

******** Die wichtigsten S-Register *********

In der Regel 8-Bit gross und koennen mit AT&W und ATSxx=yyy beschrieben und mit ATSxx? ausgelesen werden.

S0 = Bei S0 > 0 schaltet das Modem auf Automatische_Anrufbeantwortung [AA]. Die Groesse von S0 bestimmt, wie oft das Modem laeuten laesst, bis es ans Netz geht.

S1 = Zaehler des Telefonlaeuten, nach 8 sec ohne Laeuten wird der Zaehler geloescht.

S2 = ASCII-Zeichen, womit man das Modem in den Komandomodus umschaltet. In der Regel S2=43 [+].

S3 = Das ASCII-Zeichen fuer Return S3=13 [CR]

S4 = Das ASCII-Zeichen fuer Linefeed S4=10 [LF]

S5 = Das ASCII-Zeichen fuer Back Space S5=8 [BS]

S6 = Wartet auf Freizeichen (bei ATX2 or ATX4). Die Wartezeit wird in sec angegeben. Dann wird der Waehlvorgang abgebrochen. S6=2 [in sec].

S7 = Wartet auf einen Traegerton nach dem Waehlen (bei ATX3 or ATX4), im Mittel wird 30 sec empfohlen.

S8 = Die Zeidauer fuer ein Komma waehrend des Waehlens. Beim Waehlen kann man eine Pause mit eingeben (ATDP0231.353060). Sie sollte 2 sec lang sein.

S9 = Pruefzeit fuer Carrier_Detected in Zehntelsekunden beim Verbindungsaufbau. Bei mir S9=6 [1/10 sec]

S10 = Max. Unterbrechungszeit des Carriers in Zentelsekunden, danach wird die Verbindung abgebrochen.

S11 = Waehlgeschwindigkeit beim Tonwahlverfahren in msec.
Die Einstellung sollte S11=100 [msec] sein.

S12 = Unterbrechungszeit in msec nach einer ESC-Sequenz [+++],
bevor das Modem zur Dateneingabe wieder freigegeben wird.

Dieser Text ist ohne die dtsch. Umlaute / Sonderzeichen. Keine Gewaehr fur Richtigkeit und Vollstaendigkeit. Bei Fehler, Aenderungen und Ergaenzungen bitte bei mir melden.

Ludger Honnacker
Auf der Donau 16
4300 Essen 1

Tel 0201/283986

Ruf mich in der COMSTAR mit LUKAS.

Hurra,
wir sind noch da zur diesjährigen HOBBY-tronic. Ich hoffe auf zahlreiche Besucher.

Diesmal brauche ich dringend ein paar Standhelfer, die nicht müde werden und gute Laune mitbringen.

Damit das Chaos vollkommen wird, wollen wir die Messearbeiten auch planen.

Es wäre auch ganz gut wenn einige SVI-Clubmitglieder zu uns kommen.

Westfalenhallen Dortmund

Zwei Themen – ein Ereignis!
Hobby-tronic
15. Ausstellung für Funk- und Hobby-Elektronik
COMPUTERSCHAU
8. Ausstellung für Computer, Software und Zubehör
25.–29. März 1992
Messezentrum Westfalenhallen Dortmund

Call me ...

(siehe oben)

Das'n Ding, der DASM !

(Teil 1:)

Moin, moin!
Muß mich auch mal wieder zu Wort melden, habe es schließlich versprochen (hallo Rolf, hallo Uwe). Leider war die Pause viel größer als geplant, aber mein Chef hat nun mal das Sagen (er überredet mich immer mit Geld...)

Ihr erinnert Euch, ich möchte auf ROWIS "Assembler-Kurs" reagieren... Clubheft und Club gehört doch uns ALLEN.

Beim letzten Mal gab es was zum Basteln und den Anstoß, mal in den Z8E hineinzuschnuppern. Auch die folgenden Artikel werden sich mit erschwinglicher Public-Domain-Software bzw. Clubware beschäftigen.

Ähnliches gilt für die kleinen Prögämmchen, die wir uns pöapö zu Gemüte führen. Es läuft also alles ganz leicht und locker ab, für jeden verständlich und lesbar.

So, nun aber zur Überschrift - dem DASM.

DASM ist zunächst einmal ein DisASSeMbler, der als Public-Domain durch die Diskettenboxen geistert.

Noch so'n Ding - haben wir doch schon so viele ...

Weit gefehlt - DASM ist kein gewöhnlicher Disassembler, der aus Maschinencode einen lesbaren Z80-Code erstellt. Dann hättet Ihr Recht, so was hat man tatsächlich in allen Hosentaschen. DASM ist so etwas wie "intelligent" und "interaktiv".

Bevor wir uns das näher ansehen, gönnt Euch erstmal "n'Tass Kaff". (ich natürlich "Flasch Bier" die erste)

Wer von Euch schon mit einem "Hosentaschendisassembler" sein Glück versucht hat, dem ging es bestimmt genauso wie mir:
Das Listing ergab irgendwie keinen Sinn, da stimmte doch was nicht!
"Hineinsehen" ins Programm per Hexmonitor zeigte im ASCII-Teil "echte" Wörter.
Und unser Disassembler ignoriert das ?!?

Schlimmer noch, er erzeugt aus diesen "Wörtern" auch noch Quellcode! Igitt!
Und da wundert sich nun alle Welt, warum Assemblern so wenig Spaß macht.

So, und dieser Punkt wird jetzt von der "Intelligenz" des DASM in Angriff genommen. DASM ist in der Lage, Datenbereiche von mindestens 8 Byte Länge automatisch von Programmcode zu unterscheiden und selbsttätig Labels für Sprungziele zu generieren. Wir werden es später ausprobieren, schluckt es erst mal hinunter.

Und warum ist er jetzt "interaktiv"? Na ganz einfach, unser DASM erarbeitet ZUSAMMEN mit seinem Bediener ein (hoffentlich) fehlerfreies Quellcodelisting. Letzteres könnte dann wieder einem Assembler vorgesetzt werden usw. usw... Daran gemessen wäre DASM sogar ein Reassembler!

Ok, fassen wir kurz zusammen:
DASM ist kein Debugger wie Z8E. Er ist auch kein Monitorprogramm. - Seine Welt ist das Auflisten von Speicherbereichen und das Disassemblieren von Programmcode!

Nächstes Problem unserer Hosentaschendisassembler ist das Übel mit dem Speicherbereich:
Man lädt seinen Disassembler und danach das zu disassemblierende Programm - und Ende der Veranstaltung! Disassembler und Programm liegen im gleichen bzw. in gleichen Speicherbereichen. Der Klügere gibt nach, also DASM muß her!

DASM arbeitet intern mit einem sogenannten Offset. Zu Deutsch: Das zu disassemblierende Programm wird still und heimlich dahin verschoben, wo es "keinem" weh tut. Wir als Bediener merken davon gar nichts, weil alle DASM-Kommandos bereits darauf abgestimmt sind. Also lassen wir den Offset so wie er ist, es sieht ja so aus als würde unser Programm an der richtigen Stelle liegen (Adresse 0100H). So weit, so gut!

Pause! "Tass' Kaff", Flasch Bier, die zweite.

Nun aber 'ran an den Feind! Arbeitsdiskette mit DASM.COM erstellen, DOS sollte auch nicht fehlen und das eine oder andere COM-File kann auch noch auf die Scheibe. Fürs Erste empfehlen sich kleine Com-Files, wir wollen es ja sachte angehen lassen...
Gestartet wird schlicht und einfach mit DASM ...

```
A>DASM

DASM, Version 1.5
DASM for Zilog-Standard Mnemonics
Derived from ZZSOURCE/RESOURCE
Type H for Help, ? for Stats

*
```

Der Cursor steht hinter dem Bereitschaftszeichen, dem Stern. Wenn wir "H" eingeben erhalten wir die Befehlsliste, die wir uns erst mal ausdrucken lassen (Ctrl- P, wie bei DOS).

Das "?" zeigt uns Statistiken über die Speicheraufteilung, doch wir interessieren uns vorerst nur für die Befehle.

Da dieses Command Summery in Neu-Deutsch verfaßt ist, will ich mal nicht so sein und den Krempl übersetzen: (in japanisch mach ich so was aber nicht...)

```
H - Kommandoübersicht von DASM anzeigen
? - Informationen über die Speicheraufteilung anzeigen
;adr,Kommentar - Kommentar für die Adresse <adr> einfügen
;adr - Kommentar für die Adresse <adr> auflisten
; - gesamte Kommentartabelle anzeigen
;adr, - Kommentar für die Adresse <adr> löschen
A - nach DEFBs im Programmcode suchen
B - Symboltabelle automatisch erstellen mit Labels Lxxxx
C - Kontrolltabelle anzeigen
Cadr - Kontrolltabelle ab Adresse <adr> anzeigen
Cadr,kenner - Kontrolleintrag für Adresse <adr> setzen
              B=DEFB,     H=Hex-DEFB,     W=DEFW,     S=DEFS,
              I=Instruktion, E=End
Cadr, - Kontrolleintrag für Adresse <adr> löschen
D - Speicherauszug anzeigen (Hexdump+ASCII)
D=nn - Anzahl der Bytes für D-Befehl festlegen
DS - Symboltabelle anzeigen
DS.smbol - Symboltabelle ab .symbol anzeigen
Eadr,.symbol - Symbol zur Adresse <adr> in der Symboltabelle
               speichern
Fwort,adr - 16-Bit-Wort ab Adresse <adr> suchen
K.symbol - Symbol aus der Symboltabelle löschen
L - Disassemblieren (20 Zeilen)
L=nn - Zeilenanzahl für L-Befehl festlegen
L,nnnn - Disassembliert bis nnnn
Lnnnn - Disassembliert ab nnnn
Lnnnn,nnnn - Disassembliert von .... bis
O - zeigt den aktuellen Offset (sprachen wir schon von)
Onnnn - neuen Offset festlegen (eine Adresse)
Pstart,ende - Programmvorspann von Adresse <start> bis <ende>
              erzeugen
Rdateiname.COM - .COM-Programa an Offset+0100H einlesen
Rdateiname.CTL - Kontrolltabelle einlesen
Rdateiname.SYM - Symboltabelle einlesen
Rdateiname.DOC - Kommentartabelle einlesen
Rdateiname.ALL - .COM, .CTL, .SYM, .DOC einlesen
Uadr - Kommentartabelle an die reale Adresse <adr> legen
Sdateiname.ASM -.ASM-Datei speichern
Sdateiname.CTL - Kontrolltabelle speichern
Sdateiname.SYM - Symboltabelle speichern
Sdateiname.DOC - Kommentartabelle speichern
Sdateiname.ALL - na was wohl
X - "Restartet" DASM (!!!)
Z - END-Befehl in die geöffnete .ASM-Datei schreiben

Return-Taste - bricht laufenden Befehl ab
Control-C - "Feierabendtaste"

Stichwort "Feierabend", Pause! Tass'Kaff, Flasch Bier, die dritte.
```

Langsam aber sicher werden wir blau, - Unsinn - neugierig natürlich. Laßt uns zum Abschluß des ersten Teils doch eine kleine Übung durchziehen...

Wir "untersuchen" einfach mal den DASM.COM mit sich selbst!

DASM ist geladen und meldet sich mit Sternchen, ok?

*RDASM.COM wird jetzt eingegeben. Was das bedeutet ist klar.

Sobald der Einlesevorgang beendet ist erscheinen zwei Info-Zeilen am Bildschirm:
"Der letzte Block wurde in den Speicher gelesen bei 7200"
"Der letzte Block endet an der relativen Adresse 2600"

Im Klartext heißt das, daß unser geladenes Programm (in dem Fall DASM selbst) in der für uns sichtbaren Form von 0100H bis 2600H reicht. Tatsächlich das Programmende aber bei 7200H. Das interessiert uns aber zur Zeit nicht (Offset...).

Mit dem Befehl L (mit und ohne Parameter) können wir nun munter Drauflosdisassemblieren. Die Eingabe *L0100 bringt also erste Ergebnisse!

Probiert es auch mal mit dem D - Befehl. Ihr seht hier noch bei den Ergebnissen von "L" und "D" die konventionellen "Fehler" eines Hosentaschendisassemblers.

Als nächstes gebt bitte *A0100 ein. Befehl "A" sucht automatisch nach Bytelisten. Der Speicher wird dabei nach Bytes durchsucht, die wie "Buchstaben" aussehen. Texte größer als sieben Zeichen sollten immer erkannt werden. Im dargestellten Listing vom "A"-Befehl erkennt man an den Lücken zwischen den Adressen, daß "A" etwas gefunden hat.
Wenn wir jetzt wieder mit "L" disassemblieren zeigt DASM die richtigen DB bzw. DEFB Befehle an.

Ich glaube das macht einen besseren Eindruck, als das Absuchen per Monitor, es ist auch komfortabeler.

Ok, da bei mir kein Bier mehr im Kühlschrank ist, machen wir Schluß für heute. Nächstes Mal schauen wir uns den "Rest" von DASM an und Üben uns a bisserl am Z8E ...

Bis dahin viele Grüße an Euch alle !!!

Helmut Leschik

Pascal-Kurs

PASCALKURS VI

Typen und ihre Operationen

Einfache Typen

Ein einfacher Typ ist einer der Standart-Typen "boolean", "real", "integer", "char" oder ein Aufzählungstyp. Sein WERTEBEREICH ist nicht-leer, endliche Folgen von Werten.

(w0, wi, ...,wn)

(n>=0) mit den folgenden Eigenschaften:

Eindeutigkeit:	wi <> wj	für i <> j
Ordnung:	wi < wj	für i < j
Nachfolger:	succ(wi) = wi+1	für 0 <= i < n
Vorgänger:	pred(wi) = wi-1	für 0 < i <= n
Ordinalzahl:	ord(wi) = i	für 0 <= i <= n

Die syntaktische Darstellung eines Wertes w als Zeichenfolge in einem rograms ist typabhängig:

- Bei den Standart-Typen "boolean" und "char" und bei den Aufzählungstypen hat ein Wert w genau eine syntaktische Darstellung.
- Bei den Standart-Typen "integer" und "real" gibt es für einen Wert w mehrere Darstellungen, z.B. geben die ganzen Zahlen 11, 011, 0011 den gleichen Wert an.

Jeder Wertebereich eines einfachen Typs ist geordnet. Die Operatoren

a = b	Gleich	a <> b	Ungleich
a < b	Kleiner	a > b	Größer
a <= b	Kleiner-Gleich	a >= b	Größer-Gleich

sind definiert für Operanden a und b vom gleichen einfachen Typ und liefern ein Ergebnis des Typs "boolean". Ist wi der Wert des Operanden a und wj der des Operanden b und ist OP ein Vergleichsoperator, so hat die Operation a OP b den Wert "true", wenn i OP j gilt, andernfalls den Wert false".

Die Nachfolgerfunktion "succ" und die Vorgängerfunktion "pred" sind für alle einfachen Typen außer dem Typ "real" definiert. Man beachte jedoch, daß pred(w0) und succ(wn) undefiniert sind. Ist zudem i ein Wert vom Typ "integer", so ergeben succ(i) und der Ausdruck i+1 (bzw. pred(i) und der

Ausdruck i-1) den gleichen Wert, sofern er existiert.

Die Funktion "ord" ist für alle einfachen Typen außer dem Typ "real" definiert. Ist wi der Wert des Ausdrucks a, gilt ord(a)=i; das Ergebnis st vom Typ "integer".

Der Typ "boolean"

Der Wertebereich des Typs "boolean" enthält die logischen Elemente true" (wahr) und "false" (falsch):

```
boolean = (false, true)
```

Für diesen Typ gibt es die Operationen

AND	log. Und	(Konjunktion)
OR	log. Oder	(Disjunktion)
NOT	log. Nicht	(Negation)
=	Gleich	(Äquivalenz)
<>	Ungleich	(Antivalenz)
<=	Kleiner-Gleich	(Implikation)
<	Kleiner	(Inhibition)
>	Größer	(Inhibition)
>=	Größer-Gleich	(Implikation)

deren Definition die folgende Tabelle angibt:

a	b	a AND b	a OR b	NOT a	a = b	a < b
true	true	true	true	false	true	false
true	false	false	true	false	false	false
false	true	false	true	true	false	true
false	false	false	false	true	true	false

Die restlichen Operationen definieren die Beziehungen

```
a <> b :   NOT (a = b)
a <= b :   (a < b) OR (a = b)
a >  b :   NOT (a < b) AND NOT (a = b)
a >= b :   NOT (a < b)
```

Für die logischen Operationen AND, OR und NOT gelten die folgenden Eigenschaften:

a. Kommutativität

```
a OR b  = b OR a
a AND b = b AND a
```

b. Assoziativität

```
(a AND b) AND c = a AND (b AND c)
(a OR b) OR c   = a OR (b OR c)
```

c. Distributivität

```
(a AND b) OR c = (a OR c) AND (b OR c)
(a OR b) AND c = (a AND c) OR (b AND c)
```

d. De Morgan'sche Gesetze

```
NOT (a OR b)  = (NOT a) AND (NOT b)
NOT (a AND b) = (Not a) OR (NOT b)
```

Bemerkung zu Ausführung einer Operation:
Beide Operanden einer binären Operation werden ausgewertet, auch wenn schon der erste Operand das Gesamtergebnis festlegt. Die muß man z.B. in den folgenden Fällen beachten.

Es sei a eine Reihung des Typs "ARRAY[1..5] OF boolean", Der logische Ausdruck in der Zuweisung

```
b:= (i <= 5) AND a[i]
```

ist fehlerhaft für i>5: Der linke Operand hat den Wert false, der rechte ist undefiniert, da eine Komponente a[i] für i>5 nicht existiert. Man muß daher diese Zuweisung folgendermaßen schreiben:

```
b:= i <= 5; IF b THEN b:= a[i]
```

Diese Technik ist auch bei einer Wiederholanweisung erforderlich:

```
WHILE (i <= 5) AND a[i] DO BEGIN i:= i 1; ... END
```

Bemerkung zu den Vergleichsoperatoren:
Es ist ungewöhnlich, daß der Bereich der logischen Werte geordnet ist und somit die sechs Vergleichsoperatoren Operanden vom Typ "boolean" haben können. Jedoch sind dadurch 8 der 16 möglichen (binären) logischen Operatoren direkt verfügbar, während die restlichen durch Komposition entstehen.

Der Typ "integer"

Der Wertebereich des Typs "integer" ist die endliche und geordnete Menge der ganzen Zahlen im (abgeschlossenen) Intervall

```
[ - maxint : maxint ].
```

Die Standart-Konstante "maxint" gibt also die größte verwendbare ganze Zahl an; ihr Wert ist implementierungsabhängig.

Die Werte des Typs werden (syntaktisch gesehen) durch Zifernfolgen angegeben:

```
11   011   234   2340
```

Wenn man von führenden Nullen absieht, gehört zu jeder Zifernfolge ein Wert.

Die Operationen

a + b	ganzzahlige Addition
a - b	ganzzahlige Subtraktion
a * b	ganzzahlige Multiplikation
a DIV b	ganzzahlige Division
a MOD b	Rest der ganzzahligen Division

sind definiert für Operanden a und b vom Typ "integer" und haben ein Ergebnis vom Typ "integer", wenn es in dem obigen Intervall liegt. (Andernfalls sollte eine Programmausführung mit der Fehlermeldung "ganzzahliger Überlauf/Unterlauf" enden.)

Bemerkung zu arithmetischen Operationen:

Im Wertebereich des Typs "integer" gelten die meisten arithmetischen Gesetze für ganze Zahlen nicht uneingeschränkt, z.B. das assoziative Gesetz der Addition. Der Ausdruck

(maxint + 1) - 1 bzw. maxint + (1 - 1)

ist undefiniert bzw. definiert. Man beachte zudem, daß aus |a| <= maxint und |b| <= maxint nicht |a*b| <= maxint folgt.

Für den Typ integer gibt es drei Standartfunktionen mit Parametern dieses Typs:

abs(i)	Absolutbetrag mit Ergebnistyp integer.
odd(i)	Ist gleichwertig mit "abs(i) MOD 2 = 1", d.h. odd(i) liefert den Wert true, wenn i ungerade ist, andernfalls den Wert false.
sqr(i)	Quadrat (i^2) mit Ergebnistyp integer, sofern der Wert im zulässigen Wertebereich liegt.

(Manche Implementierungen haben weitere Standartfunktionen.)

Der Typ "real"

Der Wertebereich des Typs "real" ist eine (implementierungsabhängige) endliche und geordnete Teilmenge der reellen Zahlen.

Werte dieses Typs gibt man durch Ziffernfolgen mit Dezimalpunkt und/oder Exponent an:

1.0 1.02E+03 34E-10

Die Operationen

a + b	Gleitkomma-Addition
a - b	Gleitkomma-Subtraktion
a * b	Gleitkomma-Multiplikation
a / b	Gleitkomma-Division

sind definiert für Operanden a und b vom Typ "real" und haben ein Ergebnis vom Typ "real", wenn es in dem Wertebereich des Typs "real" liegt. (Andernfalls

sollte eine Programmausführung mit der Fehlermeldung "Gleitkomma-Überlauf/Unterlauf" enden.)

Bemerkung zum Rechnen mit reellen Zahlen:
Jeder Wert des Typs "real" kann in normalisierter Gleitkomma-Darstellung

$$x = m * b \hat{} e \quad \text{mit} \quad 1/B <= m < 1$$

angegeben werden, wobei der Exponent e eine ganze Zahl und die Basis B im allgemeinen die Zahl 2, 8, 10 oder 16 ist. x ist Repräsentant einer (unendlichen) Menge reeller Zahlen, zu der auch x gehört. Das Rechnen mit Werten des Typs "real" ist daher "nicht exakt" und erfordert Kenntnisse der Numerischen Mathematik, die in diesem Kurs nicht vermittelt werden können.

Für den Typ real gibt es die folgende Standartfunktion mit Parametern diese Typs. Der Ergebnistyp dieser Standartfunktion ist der Typ real.

abs(x)	Absolutbetrag
sqr(x)	Quadrat (x^2)
sin(x)	Sinusfunktion
cos(x)	Cosinusfunktion
exp(x)	Expotentialfunktion
ln(x)	Natürlicher Logarithmus
sqrt(x)	Quadratwurzel
arctan(x)	Arcustangens

(Manche Implementierungen haben weitere Standartfunktionen.)

Der Typ "char"

Der Wertebereich des Typs "char" ist eine endliche und geordnete Menge von Zeichen (engl. character) und stimmt im allgemeinen mit dem Zeichensatz einer Rechenanlage überein. Dieser Zeichensatz ist oftmals der ASCII-Zeichensatz mit 128 Zeichen oder der EBCDIC-Zeichensatz mit 256 Zeichen. Normalerweise enthält der Wertebereich des Typs "char" mindestens 26 Großbuchstaben, die 10 Ziffern, das Zeichen ' ' (Zwischenraum) und einige Sonderzeichen wie '-', '+', ';'. Jedoch muß nicht jedes Zeichen explizit darstellbar sein.

Bemerkung zu Implementierungsabhängigkeit:
Wegen der häufigen Abhängigkeit des Typs char vom Zeichensatz einer Rechenanlage ist es oftmals problematisch, ein Pascal-Programm auf Rechenanlagen unterschiedlichen Typs auszuführen. Es ist daher ratsam, solche implementierungsabhängigen Programmteile gut zu dokumentieren und im Programm sichtbar zu machen. Vielfach kann man die Abhängigkeit durch Mengen des Typs "SET OF char" verringern:

```
VAR grossbuchstaben, kleinbuchstaben, ziffern: SET OF char;
kleinbuchstaben:= ['a'..'z']
grossbuchstaben:= ['A'..'Z']
ziffern        := ['0'..'9']
```

Diese Mengenangaben sind nur dann sinnvoll, wenn die drei Mengen kohärent sin, und wenn sie und das Komplement ihrer Vereinigung paarweise disjunkt sind. Ist das nicht der Fall, müssen die Mengenelemente einzeln aufgeführt werden.

Bemerkung zu Konvertierung von Ziffern:
Enthält die Menge "ziffern" 10 Elemente (d.h. sie ist kohärent) und gilt "z IN ziffern", dann ergibt "ord(z)-ord(0)" den Zahlenwert des Zeichens z.

Aufzählungstypen

Der Wertebereich eines Aufzählungstyps wird durch die Aufzählung seiner Werte

(w0, w1, ..., wn)

mit n>=0 definiert. Die Werte wi werden durch Bezeichner angegeben; auch der Aufzählungstyp ist mit einem Bezeichner benennbar:

```
farbe    = (rot, gelb, gruen, blau)
tag      = (so, mo, di , mi, do, fr, sa)
richtung = (sued, nord, west, ost)
```

Bemerkung:
Andere Programmiersprachen wie ALGOL 60, ALGOL 68, PL/I oder FORTRAN kennen keine Aufzählungstypen, und der Programmierer dieser Sprachen muß einen solchen Wertebereich durch ganze Zahlen codieren, während bei der Sprache Pascal dem Übersetzer diese Aufgabe zufällt. Zudem kann dieser statisch (während der Übersetzung eines Programms) oder dynamisch (während der Ausführung eines Programms) prüfen, ob ein "Unterlauf" oder ein "Überlauf" des Wertebereichs vorkommt.

Zudem erhöhen Aufzählungstypen die Lesbarkeit und den Dokumentationswert eines Programms, und man sollte daher nicht auf ihren Einsatz verzichten.

Ausschnittstypen

Es sei T ein einfacher Typ, aber nicht der Standart-Typ "real", und {w0, ..., wn} sein Wertebereich. Ein Ausschnittstyp AT definiert eine Einschränkung des Wertebereichs auf eine Teilfolge {wi, ..., wj} mit w0 <= wi <= wj <= wn. wi heißt die "untere" Grenze und wj die "obere" Grenze von AT. Der einfache Typ T heißt sein "Grundtyp". Untere und obere Grenzen werden durch Konstanten angegeben, und der dadurch definierte Ausschnittstyp kann mit einem Bezeichner benannt werden:

```
wochentag     = mo .. fr
buchstaben    = 'A' .. 'Z'
plusminuszehn = -10 .. 10
```

Ist w ein Wert von AT und tritt w als Operand in einem Ausdruck auf, erfolgt eine Anpassung von AT an seinen Grundtyp T. Daher hat AT keine eigens für ihn definierten Operationen. Ist aber aber v eine Variable vom Typ AT, so muß bei einer Zuweisung

v:= Ausdruck

der Ausdruck einen Wert vom Grundtyp T als Ergebnis haben, der im Wertebereich von AT liegt. Andernfalls liegt ein Fehler vor. (Ausschnittstypen sind also keine Typen im bisherigen Sinne, da sie lediglich den Wertebereich eines

Grundtyps einschränken.)

Beispiel:
Es sei v eine Variable vom Typ "plusminuszehn", die den Zuweisungen

```
v:= v + 10; v:= 0
```

auftritt.t Die Richtigkeit der ersten Zuweisung muß während ihrer Ausgeprüft werden. Ist w der Wert des Ausdrucks "v + 10" und gilt -10 <= w <= 10, so ist eine Zuweisung an v erlaubt. Die Richtigkeit der zweiten Zuweisung kann jedoch schon während ihrer Übersetzung festgestellt werden; ein Test während ihrer Ausführung ist unnötig.

Bemerkung zur Verwendung:
Für Ausschnittstypen gilt die gleiche Aussage über Lesbarkeit und Dokumentationswert wie für Aufzählungstypen. Die Beschränkung auf einen problem-orientierten Teil des Wertebereichs erhöht zudem die Programmsicherheit, die stärker wiegen sollte als der vermehrte Testaufwand während der Programmausführung. Ein Übersetzer sollte zudem keinen Test, den er selber durchführen kann, auslassen.

Ausschnittstypen werden auch für die Angabe von Indexbereichen für Reihungstypen benötigt, insbesondere solche mit Grundtyp "integer".

Anpassungsoperationen

Anpassungsoperationen dienen zur Anpassung eines Wertes des Typs T1 an einen Wert eines andern Typs T2. Die wichtigste Operation ist die Anpassung der Typen "integer" und "real". Ist w der ganzzahlige Wert eines Operanden in einem Ausdruck mit Ergebnistyp "real", so wird für w der korrespondierende reelle Wert genommen:

```
        1  -->  1.0
31 + 1.1  -->  31.0 + 1.1
```

Alle anderen Anpassungen erfordern einen Funktionsaufruf.

1. Funktion trunc
Ist x ein Wert vom Typ "real", so ist das Ergebnis von trunc(x) ein ert i vom Typ "integer" mit der folgenden Eigenschaft:
- Ist x >= 0, so gilt x-1 < i <= x.
- Ist x < 0, so gilt x <= i < x+1.

Beispiel: trunc(2.7) = 2, trunc(-2.7) = -2.

2. Funktion round
Ist x ein Wert vom Typ "real", so ist das Ergebnis von round(x) ein ert i vom Typ "integer" mit der folgenden Eigenschaft:
- Ist x >= 0, so gilt x-0.5 < i <= x+0.5.
- Ist x < 0, so gilt x-0.5 <= i < x+0.5.

Beispiel: round(2.7) = 3, round(-2.7) = -3.

3. Funktion chr
Ist w0, ..., wn} er Wertebereich des Typs "char" und ist i ein Wert vom Typ "integer" mit 0<=i<=n, so ist chr(i) ein Wert vom Typ "char" it chr(i)=wi.

Beispiel: chr(ord('A')) = 'A'.

Anwendung einfacher Typen

Wir wollen in diesem Abschnitt einige typische Anwendungen für einfache Typen, vor allem für Aufzählungstypen vorstellen. Obwohl wir dazu noch nicht vorgestellte Spracheigenschaften heranziehen, ist dieser Abschnitt auch ohne ihre genaue Kenntnis verständlich.

Für unsere Beispiele vereinbaren wir den Typ

```
TYPE monate = (januar, februar, maerz, april, mai, juni, juli,
               august, september, oktober, november, dezember)
```

dessen Wertebereich also 12 Werte umfaßt.

Durchlaufen eines Wertebereichs

Die lineare Anordnung des Wertebereichs gestattet uns nun, ihn in aufsteigender (Operation succ) oder fallender (Operation pred) Richtung zu durchlaufen. Für diesen Zweck bietet Pascal LAUFANWEISUNGEN an:

```
aufsteigend: FOR m:= januar TO dezember DO Anweisung

fallend    : FOR m:= dezember DOWNTO januar DO Anweisung
```

"m" ist eine Variable des Typs "monate". Aus Gründen, die bei der Diskussion von Laufanweisungen ersichtlich werden, gehen wir davon aus, daß der Wert der dem Symbol FOR folgende Variablen, die wir auch LAUFVARIABLE nennen, durch die Ausführung der dem Symbol DO folgenden Anweisung nicht verändert wird. Nur in diesem Fall sind wir sicher, daß der Wertebereich streng monoton durchlaufen wird.

Wenn wir nun einen Ausschnitt des Wertebereichs durchlaufen wollen, so können wir ebenfalls eine Laufanweisung benutzen:

```
aufsteigend: FOR m:= A1 TO A2 DO Anweisung

Fallend    : FOR m:= A1 DOWNTO A2 Anweisung
```

Zunächst wird der Wert w1 des Ausdrucks A1 und der Wert w2 des Ausdrucks A2 berechnet. Gilt im aufsteigenden Falle w1 <= w2, so durchläuft die Laufvariable m die Werte w1, succ(w1), ..., w2 und die Anweisung wird (ord(w2) - ord(w1) + 1)-mal, sonst keinmal ausgeführt. Gilt im anderen Fall w1 >= w2, so durchläuft m die Werte w1, pred(w1), ..., w2 und die Anweisung wird (ord(w1) - ord(w2) + 1)-mal, sonst keinmal ausgeführt.

Wenn wir z.B. zwei Teilbereiche durchlaufen wollen, so müssen wir auch zwei Laufanweisungen schreiben:

```
FOR m:= april TO september DO Anweisung;
FOR m:= august DOWNTO maerz DO Anweisung
```

Eine Laufanweisung gestattet also das "zusammenhängende" Durchlaufen eines Wertebereichs. Wenn wir hingegen nur an einem Durchlauf mit Lücken interessiert sind, so müssen wir weitere Pascal-Anweisungen hinzunehmen.

Beispiele:
1. Durchlaufen eines ganzzahligen Intervalls mit Schrittweite 2:

a)
```
FOR i:= a TO b DO
  IF (i - a) MOD 2 = 0 THEN Anweisung
```

b)
```
i:= a;
WHILE i <= b DO
BEGIN Anweisung; i:= i +2 END
```

2. Durchlaufen des Typs "monate" mit Schrittweite 2:

a)
```
FOR m:= januar TO dezember DO
  IF m IN [januar, maerz, mai, juli, september, november]
  THEN Anweisung
```

b)
```
FOR m:= januar TO dezember DO
  IF (ord(m) - ord(januar)) MOD 2 = 0 THEN Anweisung
```

c)
```
m:= januar;
REPEAT Anweisung; m:= succ(m);
  IF m <> dezember THEN m:= succ(m)
UNTIL m = dezember
```

Fallunterscheidungen

Das Beispiel 2a) zeigt als weitere Anwendung die FALLUNTERSCHEIDUNG. Wenn wir z.B. zu jedem Monat die Zahl seiner Tage wissen wollen, so können wir dies durch bedingte Anweisungen formulieren:

```
IF m IN [januar, maerz, ,mai, juli, august, oktober, dezember]
THEN zahl:= 31 ELSE
IF m IN [april, juni, september, november]
THEN zahl:= 30 ELSE
IF schaltjahr THEN zahl:= 29 ELSE zahl:= 28
```

(Die Variable schaltjahr hat den Typ boolean.)

Diese Berechnung läßt sich eleganter und effizienter durch eine Pascal-Fallunterscheidung ausdrücken:

```
CASE m OF
januar, maerz, mai, juli, august, oktober, dezember: zahl:= 31;
april, juni, september, november: zahl:= 30;
februar: IF schaltjahr THEN zahl:= 29 ELSE zahl:= 28
END
```

In ähnlicher Weise können wir auch Aktionen für andere Zeitabschnitte angeben:

```
CASE m OF
januar, februar, maerz, april          : Anweisung1;
```

```
mai, juni, juli , august                : Anweisung2;
september, oktober, november, dezember : Anweisung3
END
```

Abbildungen

Das vorletzte Beispiel zeigt in Wirklichkeit eine Abbildung zwischen den Wertebereichen des Typs "monate" und des ganzzahligen Ausschnittstyps "28..31". Diese Abbildung können wir auch durch eine Pascal-Reihung

```
VAR m: ARRAY[monate] OF 28..31
```

ausdrücken, die wir durch eine Folge von Zuweisungen initialisieren:

```
FOR m:= januar TO dezember DO zahl[m]:= 31;
m[april]:= 30; m[juni]:= 30;
m[september]:= 30; m[november]:= 30;
IF schaltjahr THEN m[februar]:= 29 ELSE m[februar]:= 28
```

Man beachte, daß die Wahl einer Variablen

```
VAR m1: ARRAY[monate] OF integer
```

oder gar

```
VAR m2: ARRAY[1..12] OF integer
```

einen Verlust an Information bedeutet und daher nicht empfehlenswert ist.

Schluß für heute, liebe Leute.

ANWEISUNGEN

```
Anweisung::= Unmarkierte-Anweisung |
             Marke: Unmarkierte-Anweisung

Unmarkierte-Anweisung::= Einfache-Anweisung |
                         Zusammengesetzte-Anweisung

Einfache-Anweisung::= Zuweisung | Prozedurkopf | Sprung |
                      Leeranweisung

Zuweisung::= Variable:= Ausdruck |
             Funktions-Bezeichner:= Ausdruck

Prozeduraufruf::= Prozedur-Bezeichner |
                  Prozedur-Bezeichner ( Aktueller-
                  Parameter-Teil )

Aktueller-Parameter-Teil::= Aktueller-Parameter { , Aktueller-
                            Parameter }

Aktueller-Parameter::= Ausdruck | Variable | Prozedur-
                       Bezeichner | Funktions-Bezeichner
```

```
                Sprung::= GOTO Marke

          Leeranweisung::= leer

Zusammengesetzte-Anweisung::= Anweisungssequenz | Bedingte-Anweisung |
                              Wiederholungsanweisung | Laufanweisung |
                              Fallunterscheidung | Inspektionsanweisung

     Anweisungsequenz::= BEGIN Anweisung { ; Anweisung } END

   Bedingte-Anweisung::= IF Ausdruck THEN Anweisung |
                         IF Ausdruck THEN Anweisung ELSE Anweisung

     Fallunterscheidung::= CASE Ausdruck OF Alternative
                          { ; Alternative } END

       Alternative::= Fallmarken : Anweisung | leer

        Fallmarken::= Konstante { , Konstante }

  Wiederholanweisung::= WHILE Ausdruck DO Anweisung |
                        REPEAT Anweisung { ; Anweisung } UNTIL
                        Ausdruck

     Laufanweisung::= FOR Lauf-Bezeichner := Z{hlung DO
                      Anweisung

          Z{hlung::= Ausdruck TO Ausdruck |
                     Ausdruck DOWNTO Ausdruck

Inspektionsanweisung::= WITH Variable { , Variable } DO Anweisung


                     AUSDRUCKE

         Variable::= Einfache-Variable | Indizierte-Variable |
                     Selektierte-Variable | Zeiger-Variable |
                     Sequenz-Variable

  Einfache-Variable::= Bezeichner

 Indizierte-Variable::= Variable [ Ausdruck { , Ausdruck } ]

Selektierte-Variable::= Variable . Komponenten-Bezeichner

   Zeiger-Variable::= Variable ^

  Sequenz-Variable::= Variable ^

        Ausdruck::= Einfacher-Ausdruck |
                    Einfacher-Ausdruck Vergleichsoperator
                    Einfacher-Ausdruck

Vergleichsoperator::= = | <> | < | <= | > | >= | IN
```

```
Einfacher-Ausdruck::= Term | Vorzeichen Term | Einfacher-
                      Ausdruck Additionsoperator Term

Additionsoperator::= + | - | OR

Term::= Faktor | Term Multiplikationsoperator
        Faktor

Multiplikationsoperator::= * | / | DIV | MOD | AND

Faktor::= Variable | Konstante | ( Ausdruck ) |
          Funktionsaufruf | Menge | NOT Faktor |
          NIL

Funktionsaufruf::= Funktions-Bezeichner | Funktions-
                   Bezeichner ( Aktueller-Parameter-Teil )

Menge::= [ Mengenkomponentenfolge ]

Mengenkomponentenfolge::= leer | Mengenkomponenten
                          { , Mengenkomponenten }

Mengenkomponenten::= Ausdruck | Ausdruck .. Ausdruck
```

Das nächste mal werden wir uns mit Zusammengesetzten Typen befassen.

BIS BALD!

C – Kurs

Lösungen Lektion 5

Oh, dieses Mal wars ja schon wieder mal ein bißchen schwieriger, aber hier wird Eure Neugier befriedigt:

1. Programm zum Vergleichen zweier Strings:

```
#include <stdio.h>
#include <string.h>

main()
{
  char text1[81], text2[81];

  puts("Gib den ersten text ein:");
  gets(text1);
  puts("Gib den zweiten Text ein:");
  gets(text2);

  if (strcmp(text1, text2) == 0)
    puts("Die Texte sind gleich");
  else if (strcmp(text1, text2) < 0)
    puts("Der erste Text kommt alphabetisch vor dem zweiten");
  else
    puts("Der zweite Text kommt alphabetisch vor dem ersten");
}
```

2. MiniQuiz

```
#include <stdio.h>
#include <string.h>       /* Für String-Funktionen notwendig (strcmp) */

main()
{
  char frage[81], richtig[81], antwort[81];

  puts("          MiniQuiz");
  puts("          ========\n\n");
  puts("Wie hei~t die Frage?");
  gets(frage);
  puts("Wie lautet die richtige Antwort?");
  gets(richtig);

  puts("\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
  puts(frage);
  gets(antwort);
```

```
 if (strcmp(antwort, richtig) == 0)
  {
    puts("Oh wow! Du kennst Dich ja echt aus!");
    puts("Weiter so!");
  }
  else
  {
    puts("Oh no! Die richtige Antwort lautet:");
    puts(richtig);
  }
}
```

Alles richtig? Na, nur nicht übermutig werden. Das kommt noch besser. Da kommt dann die Sternstunde der anderen!

Orca

C-Kurs Teil 6

Her mit den Integern

Integer haben wir schon kennengelernt, ebenso die Möglichkeit einfacher Entscheidungen. Mit Strings haben wir auch schon ein bißchen herumgespielt. Jedoch mit Integern (der einzige Datentyp, den wir schon richtig eingeführt haben) können wir noch nicht richtig arbeiten. Die ersten nötigen Dinge dazu lernen wir jetzt kennen!

Zuweisungen

Einer Variable kann man einen Wert zuweisen. Ginge dies nicht, so wäre die Verwendbarkeit von Variablen ja auch ziemlich eingeschränkt. Der Zuweisungs-Operator ist das '=' - Zeichen. Das sollte nicht verwechselt werden, mit dem Boolschen Operator '==', der auf Gleichheit testet. Dies ist ein bei C-Programmierern sehr beliebter Fehler. Manche Compiler weisen auch darauf hin, wenn die Chance einer Verwechslung ziemlich hoch ist.
Die Syntax sieht so aus:

<L-Value> = <Ausdruck>

Puuh, L-Value, was ist das schon wieder?? Nun, bei einem L-Value handelt es sich einfach um etwas, dem man etwas zuweisen kann. Als einziges, was wir bis jetzt kennen, kommt dafür eine Variable in Frage. Jedoch wäre z.B. auch eine absolut adressierte Speicherstelle denkbar, solche L-Values sind jedoch nicht so häufig.

C-Kurs C-Kurs C-Kurs C-Kurs C-Kurs C-Kurs C-Kurs C-Kurs

Was ein Ausdruck ist, haben die meisten sicher eine wage Vorstellung. Hier ein paar Beispiele:

```
3
a + b
4 * c
(3+a) / (b-2)
```

Eine gültige Zuweisung ist z.B.:

```
a = 3
a = a + 5
```

Bei der Zuweisung wird zuerst der rechte Teil der Zuweisung ausgewertet und dann das Ergebnis dem linken Teil zugewiesen. Daher ist auch das zweite Beispiel korrekt. Der Wert von a wird um 5 erhöht. Aber auch die Zuweisung selber stellt wieder einen ausdruck dar, daher ist auch die Zuweisung

```
a = (b = 5)
```

völlig korrekt. Hierbei wird b der Wert 5 zugewiesen und a der Wert der Zuweisung (b = 5). Der Wert einer Zuweisung ist aber gleich dem zugewiesenen Wert, so daß auch a den Wert 5 erhält. Betrachten wir dazu noch ein Beispiel, daß gleich einen ganzen Block umfaßt:

```
{
  int a, b, c;

  a = b = 5;     /* Die Klammern dürfen also auch weggelassen werden */
  c = (a == b);
}
```

Welchen Wert bekam c zugewiesen? Die richtige Antwort lautet 1. Hier kommt der Unterschied zwischen '=' und '==' zum Tragen. Der Wert des Tests auf Gleichheit ist 1, wenn die zu vergleichenden Ausdrücke gleich sind, sonst 0. Hier waren a und b beide gleich (nämlich 5), so hat der Ausdruck (a == b) den Wert 1, den c dann zugewiesen bekommt.

Nach soviel Theorie jetzt wieder ein Stückchen Programmierpraxis; jetzt folgt ein Programm, das zwei Integer einliest und die Summe dazu ausgibt:

```
#include <stdio.h>

main()
{
  char eingabe[81];
  int summand1, summand2, summe;

  puts("Gib den ersten Summanden ein!");
  gets(eingabe);
  summand1 = atoi(eingabe);
  puts("Gib den zweiten Summanden ein!");
  gets(eingabe);
  summand2 = atoi(eingabe);
  summe = summand1 + summand2;
  printf("Die Summe von %d und %d ist %d.\n", summand1, summand2, summe);
}
```

Da sehen wir schon wieder eine neue Funktion (seid getröstet, da gibts noch Hunderte :-)), werden einige jetzt stöhnen, aber es geht nicht ohne diese. Die Funktion atoi (für ASCII to Integer) wandelt den als Parameter angegebenen String in einen integer um, der dann zurückgegeben wird. Das ist für uns im Moment die einfachste Methode zur Eingabe von Integern. Hier taucht auch schon ein Problem auf, das bei der Verwendung von Integern sehr häufig anzutreffen ist: Überschretet eine der drei int-Variablen ihren Wertebereich, so erscheint eine unsinnige Ausgabe, jedoch nichts sonst weist auf die Überschreitung des Wertebereichs hin. Versuchts alle einmal. In der Integer-Lektion haben wir ein wenig mit dem Wertebereich experimentiert, ich hoffe Ihr wißt alle, wie groß dieser bei Euch für ein int ist.

Operatoren

Wir haben schon drei Operatoren kennengelernt:
'==' Gleichheitstest,
'=' Zuweisung und
'+' Addition.

Langsam wollt Ihr wahrscheinlich auch selbst ein bißchen ausprobieren und nicht immer nur auf das nächste Beispielprogramm warten wollt. Deshalb werde ich hier eine kurze Aufstellung einiger wichtiger Operatoren angeben. Beginnen werde ich mit den Boolschen Operatoren (zur Manipulation von Wahrheitswerten):

==	gleich
!=	ungleich
>	größer als
<	kleiner als
>=	größer oder gleich
<=	kleiner oder gleich

!	nicht (Umkehrung des Wahrheitswertes)
&&	und (wahr, wenn beide Operanden wahr sind)
||	oder (wahr, wenn mindestens einer der Operanden wahr ist)

Dazu jetzt noch ein paar arithmetische Operatoren (genau, so wie '+'):

+	Addition
-	Subtraktion
*	Multiplikation
/	Division
%	modulo (Rest einer Division)

Viel Spaß beim Ausprobieren.

Übungsaufgabe

Schreibe ein Programm, welches für zwei eingelesene Integer alle fünf vorgestellten arithmetischen Operationen ausführt und zusätzlich testet, welche der beiden Zahlen größer ist (bzw. Gleichheit feststellt). Die Ergebnisse sollen natürlich auch in einer möglichst übersichtlichen Form ausgegeben werden.
ACHTUNG! Da wir ja benutzerfreundlich sind, vergeßt nicht sicherzustellen, daß nicht durch 0 geteilt wird (wichtig bei Division und modulo)! Das wollen wir nicht dem Laufzeitsystem überlassen, daß sich dann mit der schlichten Meldung "Division by Zero" verabschiedet.

Ideenecke

Die Ideenecke ist als lockerer Nachfolger der Projekte gedacht. Nachteil der Projekte war, daß Einzelne an festgelegten, komplexen Dingen arbeiteten und der Rest nur gierig auf Resultate wartete! Oft kommen aber Mitglieder mit kleineren oder größeren Problemen oder auch einer guten Idee zu mir, die ich aus Zeitmangel oder mangels Wissen nicht lösen kann, die sich aber auch nicht für ein eigenes Projekt eignen.

Also Ideen und Probleme zu mir; ich prüfe, ob es grundsätzlich realisierbar ist und schreib's dann in die Ideenecke. Wenn jetzt jemand glaubt, daß irgendeine Sache genau seine Kragenweite hat, kann er sich dransetzen. Manchmal fehlt ja nur genau dieser Anstoß!

1. In eigener Sache: es werden laufend Leute gesucht, die Aufgaben im Club übernehmen wollen. Was genau, bleibt jedem selbst überlassen - das kann z.B. Beratung oder irgendein Bastelvorhaben sein. Zur Realisierung gibt's auch finanzielle Unterstützung vom Club (Unkostenerstattung).

2. Unser Ladenhüter: Seit den ersten Clubtagen gibt es eine Diskette mit BASIC-Programmen vom SVI-328, die auf eine Umsetzung nach MSX warten (MSX-CLUB.002)! Es müssen eigentlich nur eine handvoll Befehle geändert werden (z.B. aus SCREEN 1 überall SCREEN 2 machen oder PRINT im Grafik-Modus durch OPEN "GRP:" und PRINT#1 ersetzen). Ein paar Leute haben schon angefangen, aber es sind genug Programme drauf für alle.

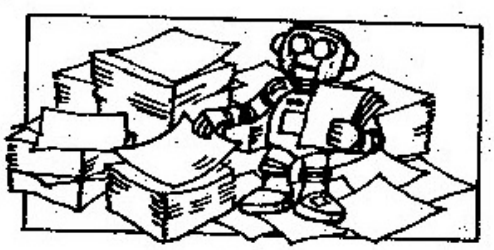
3. Unsere Public-Domain-Programmdisketten enthalten größtenteils sehr umfangreiche, aber leider englische Beschreibungen. Wer übersetzt den Krempel?

4. Lothar Breitner plant den Selbstbauplotter MONDRIAN, vorgestellt in der Elektor 1/88, an MSX anzupassen. Wer hilft ihm?
   Anschrift: Lothar Breitner
   Winkelsfelderstr. 24
   4000 Düsseldorf 30

5. Wer holt die Utilities (Uhr, Taschenrechner, Kalender) aus HiBrid heraus und macht daraus ein residentes TSR-Programm á la Sidekick?

6. Wer schreibt S-RAM Routinen unter Turbo-Pascal für's FM-PAC?

7. Wer weiß, wie man die einzelnen Farben auf den Screens 10, 11 und 12 errechnet? Wie wird das innerhalb eines 4-Pixelblocks richtig ermittelt?



Die Aktiven

Sehr oft bekomme ich Fragen, bei denen ich nicht weiterhelfen kann. Ich habe dann bisher auf die Projektleiter verwiesen, was dem Anrufer nur zusätzliche Telefonkosten einbrachte. Wer Fragen zu Hard- oder Software hat, sollte sich direkt an die entsprechenden Leute wenden - es sei denn, das Thema ist so exotisch, daß sich keiner in der folgenden Liste findet:

Wolfgang Kuhlmann
Marcq-En-Baroeul-Str.17
4390 Gladbeck
Tel.: (02043) 44869

Software: MSX-Emulator
Hardware: Programm-Module (Cartridges) / 40/80-Zeichen-Umschaltung/RS-232/Technik SVI allgemein / Eventuell Einbau C.U.C.-Projekte (6 MHz, etc.) nach Absprache und gegen geringe Aufwandsentschädigung

Thomas Wahle
Lütgdtmd. Hellweg 99
4600 Dortmund 72
Tel.: (0231) 603007

Software: TurboPascal / CP/M
Hardware: 6 MHz-Projekt / BIOS-Platine (eventuell Einbau der Projekte nach Absprache und gegen Aufwandsentschädigung) / Fremdtastaturen am SVI / 80-Track Laufwerke (gute Einkaufsadresse!!) / allgemeine Hardware-Tips!

Wolfgang Bertleff
Am Vorderberg 11
8802 Sachsen
Tel.: (09827) 1783

Software: Ein/Ausgabe-Steuerung für den SV.318/328

Hardware: I/O-Module (PIO-Interface etc.) / Modellbahnsteuerung / Interrupts / Anwendungen für I/O-Module

Rainer Kulhanek
Hilblestr. 63
8000 München
Tel.: ( )

Software: CP/M allgemein / CB-Utility (C.U.C.-BIOS) / CP/M-Assembler

Hardware: -

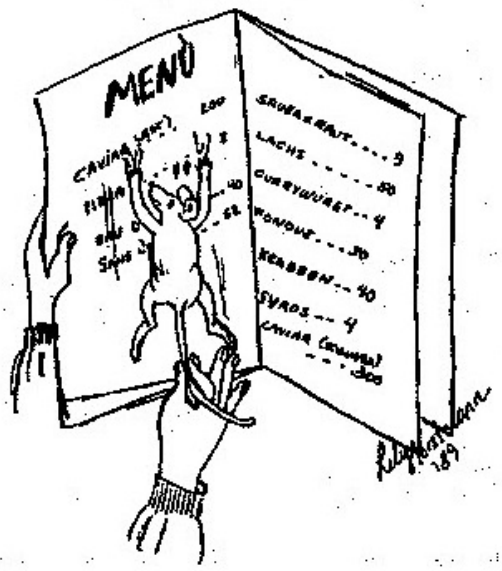
Bernd Fitzke
(nicht: Pfitzke, Fitzge oder Ähnliches!!!!!!!!)
Dachstr. 27
4300 Essen 11
Tel.: (0201) 682104

Special Task: Er sammelt immer noch, auch wenn nur wenige Ihm etwas schicken. Gesucht sind Programme und Artikel für SVI und MSX, die er archivieren kann. So ein Archiv ist natürlich auch dafür da, daß jeder Anfragen zu einem Thema stellen kann, das ihn interessiert. Vielleicht liegt ja im Archiv die Lösung...

Ulrich Koppe c/o Koke
Hatzenbecker Str. 85
5600 Wuppertal 1
Tel.: (0202) 427160

Software: C.U.C.-BIOS / C.U.C. Disk-BASIC / WordStar / Anwenderprogramme unter CP/M / Datenfernübertragungsprogramme SVI
Hardware: Weiterleiten spezieller Fragen an den C.U.C. Holland

Udo Willigmann Thuiner Str. 13 4450 Lingen Tel.: (0591) 2393	*Software*: Datenfernübertragungsprogramme MSX *Hardware*: Alles über Datenfern-Übertragung --- MODEMS, Akustikkoppler / was ist erlaubt? / wie steige ich in die DFÜ ein? / wo bekomme ich was?
Willy Czysch Ginsterweg 29 4200 Oberhausen 12 Tel.: (0208) 603409	*Hardware*: Reparatur von SVI-Hardware (!) / Einbau von Hardware-Projekten / Bau von Gehäusen. Herr Czysch hört sofort wieder auf, wenn die Leute ihm jetzt die Bude einrennen! Eigeninitiative (z.B. besorgen der Ersatzteile) ist unerläßlich, aber ich finde es gut, daß überhaupt jemand da ist!
Rainer Lüdicke Brackeler Hellweg 109 4600 Dortmund 12 Tel.: (0231) 255823	*Hard- und Software:* Ansprechpartner für alle "Anfängerfragen" (insbesondere MSX). Wer sich nicht traut, Fragen zu stellen, weil er Angst hat, sich zu blamieren, kann sich vertrauensvoll an Rainer wenden.
Rolf Witt Herderstr. 38 . 4600 Dortmund Tel.: (0231) 824976	*Hardware:* MSX_2, Drucker *Software:* MSX-Systemroutinen und MSX-DOS, Einhaltung des MSX-Standards (was muß ich bei meinen Programmen beachten, damit sie auf allen MSX-Rechnern laufen?).
Ralf Siedek Am Ringwall 10 4224 Hünxe Tel.: (02858) 6147	*Software:* Spieletips für MSX



... die Maus über ein Menue ziehen ...

Kleinanzeigen

Ich biete euch folgende Sachen zum Kauf an:
- VG 8010 o. Handbuch, sonst komplett 60,- DM
- VG 8020 komplett 80,- DM
- Printer Interface für VG 8010 40,- DM
- Slotverteiler (kein Expander!) f. VG 8010 40,- DM
- Philips Datenrecorder in Pultform 40,- DM
- zwei Monocassettenrecorder je 30,- DM
- vier 3,5" Laufwerke, 360KB formatiert je 50,- DM

Und wer verkauft mir die folgenden Sachen ???
- Hardware: Sony Plotter PRN C41, Drucker NMS 1431, FM PAC Stereo, Sony Joysticks JS 55
- Module : Konamis Ping Pong, Roadfighter, Hyper Rally, Nemesis, Penguin Adventure, Taitos Bubble Bobble, Hals Rollerball, Eggerland Mistery, Sonys 4 KB Data Cartridges und das D.M.S.1 Modul

Schickt mir bitte eure Angebote oder Kaufgesuche oder ruft mich nach 20 Uhr einfach mal an.
Lars Aschenbach, Stückenstr. 57, W 2000 Hamburg 76,
Tel.: 040 / 299 30 31 nach 20 Uhr

Suche unbedingt NEMESIS-3 von Konami, ebenfalls suche ich KINGS VALLEY2 MSX2-Version! Angebote an Herbert Kloseck. Adresse siehe Impressum.

MSX- und SVI-Computer gesucht! Angebote an Ulrich Koppe. Adresse siehe Impressum.

Software für Sammler zu verkaufen :

Programmodule

SONY:	MUSIKSTUDIO G7	33.-DM	KONAMI:	METAL GEAR (MSX2)	25.-DM
	CHESS	10.-DM		KNITHMARE	17.-DM
	SUPER SOCCER	10.-DM		HYPER RALLEY	18.-DM
	ALIBABA AND 40 THIEVES	10.-DM		CIRCUS CHARLIE	12.-DM
	MOUSER	12.-DM		HYPERSPORTS 1&2 JE	12.-DM
HAL:	MUE (Musikeditor)	17.-DM		YIE AR KUNG FU 2	15.-DM
XAIN:	DRACHENKAMPFER	32.-DM		GOLF	13.-DM
				COMIC BAKERY	11.-DM
Original_Diskkette			RAX:	ALIEN SLIME	15.-DM

Wer verkauft mir den seinen Gammaster 2 (Konami). Ich zahle Höchstpreise. Dann suche ich folgende Module Nemesis III, USAS, Space Mainbow, SD-Snatcher und Quart.
Wer leiht mir seinen Gamemaster 2 für 3 Wochen. Unkosten werden erstattet.

Bitte melden bei Ludger Honnacker 0201-283986

Impressum

Die "Clubzeitschrift" wird herausgegeben vom SVI/MSX-Club Deutschland und erscheint viermal jährlich. Mitglieder erhalten sie kostenlos.

Redaktion:
MSX: Herbert Kloseck, Yorckstr. 7, 4670 Lünen - 6, (0231) 87 01 06
SVI-328: Ulrich Koppe c/o Tobias Koke, Hatzenbecker Str. 85, 5600 Wuppertal 1

E-Mail:

MSX:	(0231) 35 60 30 HERBY;FORUM	(Parameter 1200/2400 8M1XNNNN)
	(0231) 35 60 30 ROWI;FORUM	(Parameter 1200/2400 8N1XNNNN)
	(0231) 82 80 279 ROWI$DOO.ZER	(Parameter 1200/2400 8N1XNNNN)
SVI-328:	(0202) 46 42 59 ATREJU;WODS	(Parameter 300/1200 8N1XNNNN)

Unix-Mail:
BITNET: usch$aea.e-technik.uni-bochum.de
UUCP: ... mcsun!unido!rubmez!rubaea!usch

Fax: (0209) 20 98 89

Kommerzielle Anzeigen:
M.S. Data, Am Lenningskamp 17, 5840 Schwerte, (02304) 8 62 37
Es gilt Anzeigenpreisiste 1/88.

Druck: Kopie + Druck, Unicenter 253-255, 4630 Bochum-Querenburg

Für die Richtigkeit der in den Artikeln gemachten Angaben sowie für die veröffentlichten Programme keine Haftung. Sofern die veröffentlichten Informationen dem Patent-, Gebrauchsmuster- oder Urheberrechtsschutz unterliegen, ist gewerblicher Gebrauch nur mit Zustimmung des Schutzrechtinhabers zulässig; nur der private Gebrauch ist frei.

Verantwortlich sind jeweils die im Inhaltsverzeichnis angegebenen Autoren.

Mitgliedsbeitrag des SVI/MSX-Club Deutschland:
60.- DM / Jahr für Erwachsene,
30.- DM / Jahr für Jugendliche unter 18 Jahren.

Konto: Uwe Schröder, Postgiroamt Essen, Kto.-Nr. 3744 82-437

Mitglieder-Verwaltung: Marion Willigmann, Thuiner Str. 13, 4450 Lingen

Redaktionsschluß für Heft 1/92 ist der 29. Februar 1991!

MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2

Hardware-Service MSX

MSXHARD 1: MC-EPROM-Programmiergerät
Aus der Zeitschrift MC, angepasst an MSX. Fertiggerät incl. Beschreibung, Gehäuse, Software (Für die Typen 2716 bis 27256, bei MSXDOS2 bis 27512). **Preis: 230,- DM**

MSXHARD 2: Platine, Beschreibung, Software zu MSXHARD 1 **Preis: 25,- DM**

MSXHARD 3: B-Laufwerk
3.5 Zoll, 80 Spuren, Doppelseitig, Netzteil, Anschlußkabel, Gehäuse, Höhe 1 Zoll, getestet an Philips und Sony, komplett.
Preis: 300,- DM

MSXHARD 3a: B-Laufwerk für SVI-738, ohne Netzteil **Preis: 260,- DM**

MSXHARD 4: SpeedUp-IIa Beschleuniger-Satz
Diese Karte zum Einbauen versorgt alle MSX-Computer mit einer höheren Taktfrequenz. Diese ist fest einstellbar bis max. 8 MHz, damit alle Systeme mit ihrer höchsten Geschwindigkeit arbeiten können. Alle Programme laufen bedeutend schneller, solange sie nicht das Laufwerk benutzen und komplexe Videozugriffe beinhalten. Weiterhin gibt es eine frei regelbare Frequenz von 1,5 MHz bis ca. 5/6 der max. Frequenz. **Preis: 55,- DM**

MSXHARD 5: Slotadapter
Dieser Adapter, der in den Slot gesteckt wird, hat als Ausgang einen Stecker. Für Zusatzkarten, die über ein Flachbandkabel angeschlossen werden.
Diesen Adapter gibt es auch zum Anschluß eines SVI-707-Laufwerks!
(bei Bestellung vermerken) **Preis ohne Gehäuse: 20,- DM**
Preis mit Gehäuse: 35,- DM

MSXHARD 6: Verlängerungskabel für Tastaturanschluß
Tastaturverlängerung für HB-F700, HB-G900 etc. Länge 2 Meter.
Preis: 20,- DM

MSXHARD 7: Clubcartridge Version 1.10
Diese Cartridge enthält neue BASIC-Befehle zu Bildschirmbehandlung, Zeichensatz, Diskettenbetrieb und Hardcopy unter BASIC und MSX-DOS, sowie neue Tastenkombinationen, um solche Funktionen auszulösen. Läuft jetzt auch unter MSX-DOS-2. **Preis: 40,- DM**

UPDATE zur Clubcartridge
Ein Update erfolgt gegen Einsendung der Originalcartridge.
Preis: 10,- DM

MSXHARD 8: Cartridge-Leergehäuse, klein (wie SONY-Spiele) **Preis: 15,- DM**

MSXHARD 10: Cartridge-Leergehäuse, groß (wie SONY-RS232) **Preis: 20,- DM**

MSX1 MSX2 MSX1 MSX2 MSX1 MSX2 MSX1 MSX2 MSX1 MSX2 MSX1 MSX2

MSXHARD 12: SCSI-Interface
Die Schnittstelle zur Welt der großen Computer. Vorzugsweise zum Anschluß von SCSI-Festplatten, aber auch Computer und Disk-Laufwerke, insgesamt max. 8 Geräte, können angeschlossen sein. Im Gehäuse. **Preis: nur noch 200,- DM**

MSXHARD 12A:SCSI-Interface ohne Gehäuse **Preis: nur noch 160,- DM**

MSXHARD 13: RS232-Schnittstelle
Die Standard-Schnittstelle zum Anschluß fremder Geräte, wie Modem, Akustikkoppler, Plotter usw. Aus urheberrechtlichen Gründen ohne programmiertes Eprom, läuft aber z.B. am DFÜ-Progamm MEX. Ohne Gehäuse. **Preis: 190,- DM**

MSXHARD 13A:Schnittstelle im Gehäuse mit Modem/TerminalUmschalter **Preis: 240,- DM**

MSXHARD 13B:Platine und Bauanleitung **Preis: 80,- DM**

MSXHARD 14: 1MB-Speichererweiterung
Speichererweiterung mit Memory-Mapper, ohne Speicherbausteine, bis 1MegaByte Speicher aufrüstbar in Schritten von 256kB. Läuft auch mit hoher Taktrate bis ca. 7,5 MHz! Wird geliefert ohne Gehäuse, zum Anschluß wird MSXHARD 5 benötigt. **Preis: 160,- DM**

MSXHARD 14A:Speichererweiterung mit Gehäuse und Slotadapter **Preis: 230,- DM**

MSXHARD 14B:Platine und Bauanleitung **Preis: 90,- DM**

MSXHARD 14C:Speicherchips für je 256kB **Preis: 30,- DM**

MSXHARD 15: Universal-EPROM-Platine
Für 16k/4000-7FFF, 16k/8000-BFFF, 32k, 64k, mit Bankumschalter. Doppelseitige Platine ohne Durchkontakt für kleines Cartridge. **Preis: 15,- DM**

MSXHARD 15A:EPROM-Platine, durchgenietet **Preis: 20,- DM**

MSXHARD 15B:EPROM-Platine, durchgenietet, Bankumschalter bestückt **Preis: 30,- DM**

MSXHARD 16: Universal-EPROM-Platine, wie HARD12, zum Einbauen in NMS-8280, benutzt den unbestückten Slot 3-1. **Preis: 15,- DM**

MSXHARD 16A:EPROM-Platine für NMS8280, durchgenietet **Preis: 20,- DM**

MSXHARD 16B:EPROM-Platine für NMS8280, durchgenietet, Bankumschalter bestückt **Preis: 30,- DM**

EPROM-Service bei MSXHARD 15A/15B/16A/16B:
Bei Einsendung des EPROM-Inhalts auf Diskette wird das EPROM gebrannt, bestückt und getestet. **Preis: 15,- DM**

MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2

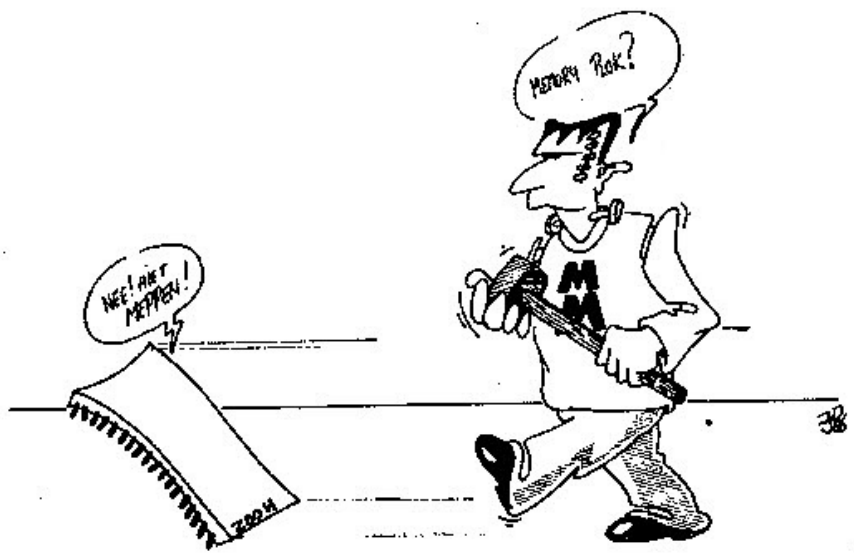
MSXHARD 17: 128k-Speichererweiterung für Philips
Einbausatz für Philips-MSX_2-Computer. Erweitert den eingebauten Speicherplatz auf 256 kByte.
Achtung: Beim 8235/22/20/29/36/39 wird ein Memory-Mapper benötigt!
Preis: 50,- DM

Neu
MSXHARD 18: Mathematik-Coprozessor-Karte
Diese Karte bearbeitet mathematische Funktionen, die ihr über eine I/O-Adresse übergeben werden. Dies geht schneller als wenn die CPU alles alleine macht. Benötigt spezielle Software. Im Gehäuse.
Preis: 175,- DM

MSXHARD 18A:Coprozessor-Karte ohne Gehäuse **Preis: 160,- DM**

Versand erfolgt nur gegen Vorkasse, Verrechnungsscheck oder Nachnahme (+ 2,50 DM). Zu den angegebenen Preisen kommt eine Versandpauschale von 3,00 DM.
Ausland: Die Mehrwertsteuer von 14% wird abgezogen, Versandpauschale hier 6,00 DM.

Das hier angegebene Angebot gilt ausschließlich für Mitglieder des SVI/MSX-Clubs Deutschland!



MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2

Software-Service MSX-BASIC und MSX-DOS

MSXCLUB.000: Ausführlicher Gesamtkatalog unserer MSX-Disketten

MSXCLUB.001: Alle MSX-Programme aus den Clubzeitschriften bis 4/86 plus ein paar Spielchen.

MSXCLUB.002: Für Programmbastler. Keins der Programme auf dieser Diskette läuft unmittelbar auf MSX; es sind BASIC-Programme vom SVI-328, die mehr oder weniger angepaßt werden müssen.

MSXCLUB.003: Mini-LISP. Für Anhänger der intelligenten Programmiersprachen. Nach wie vor Experten gesucht, die eine Dokumentation dazu anfertigen.

MSXCLUB.004: FORTH-83 Teil 1.

MSXCLUB.005: FORTH-83 Teil 2. Weitere Sources und die englische Systemdokumentation.

MSXCLUB.006: Alle MSX-Programme aus den Clubzeitschriften von 1987.

MSXCLUB.007-
MSXCLUB.009: Digitalisierte Bilder für MSX_2.

MSXCLUB.010: Atari-ST-Bilder für MSX_2.

MSXCLUB.011: Amiga-Bilder für MSX_2.

MSXCLUB.012: ADVENTURE - Das klassische Textadventure in englischer Sprache mit komplettem Quelltext.

MSXCLUB.013: DISKDOC 4.5 - Der endgültige (?) MSX-Disk-Editor.
SHOOT-IT - Der Hopsball-Killer.

MSXCLUB.014: Noch'n paar digitalisierte Bilder (einmal quer durch die Hitparade).

MSXCLUB.015: Garfield-Bilder. Wer's mag...

MSXCLUB.016: Alle MSX-Programme aus den Clubzeitschriften von 1988.

MSXCLUB.017: SONY GRAPHICS & Demonstration
Nur für MSX_2 und doppelseitige Laufwerke (720K)!

MSXCLUB.018: MEX-DFÜ-Programm

MSXCLUB.019: MasterDraw (MSX_1) und GraphicMaster (MSX_2)

MSXCLUB.020: Small-C Compiler, Assembler, Linker und Editor.

MSXCLUB.021: ALF-Demo (nur für MSX_2)

MSXCLUB.022: Z80E, Bildschirm-orientierter Z80-Debugger

MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2 MSX_1 MSX_2

MSXCLUB.023: MSX-Programmsammlung - Bisher unveröffentlichte Programme, die zum Abdrucken in der Clubzeitschrift nicht geeignet, aber zu schade zum Wegwerfen sind.

MSXCLUB.024: SCREEN-8-Picturesqueezer, kompatibel zum Sony-Grafikprogramm. Mit diesem Programm können SCREEN-8-Dateien um 10..80 % verkleinert werden. Die komprimierten Dateien können vom Sony-Programm eingelesen werden! Außerdem dabei: eine Maschinensprache-Routine, um die komprimierten Bilder unter BASIC zu nutzen.

MSXCLUB.025: Modem-Programme IMP, MDM709, MDM727 u.a. für den Fall, daß jemand mit MEX nicht glücklich ist. Die Programme sind allesamt *nicht* für MSX angepaßt, ausführliche Anleitungen sind dabei. **Achtung:** Doppelseitige Diskette (720K)!

MSXCLUB.026: Alle Programme aus den Clubzeitschriften von 1989.

MSXCLUB.027: Nützliches für Assembler- und Pascal-Programmierer.

MSXCLUB.028: Kompletter APL/Z-Interpreter (Beschreibung siehe Clubzeitschrift 1/90).

MSXCLUB.029: Alle Programme aus den Clubzeitschriften von 1990 plus Herberts Jubiläums-Demo.

MSXCLUB.030: BGM-Compiler für das FM-PAC. Setzt Score-Files vom Synth Saurus in BGM-Dateien um, die unter BASIC im Timer-Interrupt abgespielt werden können.

MSXCLUB.031: Für Nena- und MSX_{2+}-Fans: Nena live auf Screen 12. Für alle MSX-Rechner mit dem neuen VDP V9958, auch ohne MSX-BASIC 3.0! **Achtung:** Doppelseitige Diskette (720K)!

MSXCLUB.032: Deutsche Übersetzungen zu den englischen Anleitungen auf unseren anderen Clubdisketten.

MSXCLUB.033: Compy-Graf - Der Fraktalgenerator und Funktionsplotter, jetzt mit Quelltext in Turbo-Pascal.

MSXCLUB.034: MAD Interlace Demo. Die höchste Grafik-Auflösung, die MSX zu bieten hat: 512 * 424 Punkte!

MSXCLUB.035: Alle Programme aus den Clubzeitschriften von 1991. Diese Diskette "wächst" noch!

MSXCLUB.036: Noch einmal unser allseits beliebtes ALF-Demo (MSXCLUB.021), diesmal als *MSX turbo R*-Version mit PCM-Sound!

MSXCLUB.037: VGA-Bilder auf SCREEN 12 (nur für MSX_{2+} und turbo R!)

Preise, Versandbedingungen, Adressen

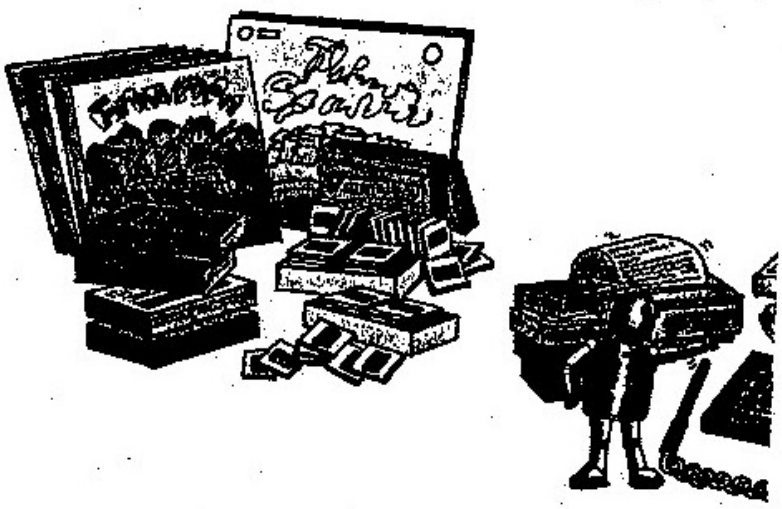
Alle Bestellungen bitte per Brief, Telefon oder Mailbox an den zuständigen Service richten. Das erspart uns unnötigen Zeit- und Geldaufwand. Wir sind sowieso zeitlich schon sehr eingespannt und über hohe Telefonkosten brauchen wir uns auch nicht zu wundern.

Alle Preise sind bei den Produkten aufgeführt, außer bei der Software. Hier gelten, wenn nicht anders angegeben, folgende Preise:

SVI:	Diskette, 5¼"	6.- DM
	Diskette, 3½"	12.- DM
	Kassette	3.- DM
MSX:	Diskette, 5¼"	4.- DM
	Diskette, 3½"	4.- DM

Die in diesem Heft genannten Preise gelten ausschließlich für Clubmitglieder. Nicht-Mitglieder zahlen bei Club-Software das Doppelte, bei anderer Software und Hardware nach Vereinbarung.

Den Betrag bitte als Scheck, Schein oder bei kleinen Beträgen auch als (deutsche!) Briefmarken der Bestellung beilegen, bei MSX-Hardware-Bestellungen ist auch Versand per Nachnahme möglich.



- ACHTUNG - ACHTUNG - ACHTUNG - ACHTUNG - ACHTUNG - ACHTUNG -

Zu jeder Bestellung, egal ob Hardware, Software, Paperware oder EPROM, kommt eine Versandpauschale von 3.- DM für Porto und Verpackung hinzu! Falls dieser Betrag nicht mitbezahlt wurde, wird er auch nachträglich in Rechnung gestellt!!

Bestelladressen

Software-Service SVI

Stefan Ludewig
Wilhelm-Firl-Straße 1
O-9047 Karl-Marx Stadt
Tel.: (???) 22 59 59

Paperware SVI & MSX

Wolfgang Maschke
Ringstr. 12
3119 Weste (Bhf.)
Tel.: (05828) 12 29
Kto. 36460-107, Blz. 10010010
Postgiroamt Berlin

Hardware SVI

Klaus Dörmann
Heinrich-Imbusch-Str. 13
4708 Kamen
Tel.: (02307) 13936
Kto. 351021-462, Blz.44010046
Postgiroamt Dortmund

Hardware MSX (Nord)

Klaus Kruse
Siedlungsstr. 6
2732 Hamersen
Kto. 30 4720-202, Blz. ???
Postgiroamt Hamburg

EPROM (SVI) und C.U.C.-System Zeichensätze 80-Zeichen Karte

Ulrich Koppe c/o Tobias Koke
Hatzenbecker Str. 85
5600 Wuppertal 1
Tel.: (0202) 427160
Mailbox: ATREJU (WODS)
Kto. 73733172, Blz. 37050198
Stadtsparkasse Köln

Software MSX

Hans-Dieter Schneider
Postfach 1346
2943 Esens

Hardware MSX (West)

M.S.DATA - Martin Schumann
Am Lenningskamp 17
5840 Schwerte
Tel.: (02304) 86237
Mailbox: M.S.DATA;PORUM
Kto. 353248-462, Blz. 44010046
Postgiroamt Dortmund

Hallo Ihr da, im Ferbruar oder März gibt's ein Clubtreffen in Schwerte. Termin wird noch öffentl. Ausgeschrieben

------

[MARTIN]