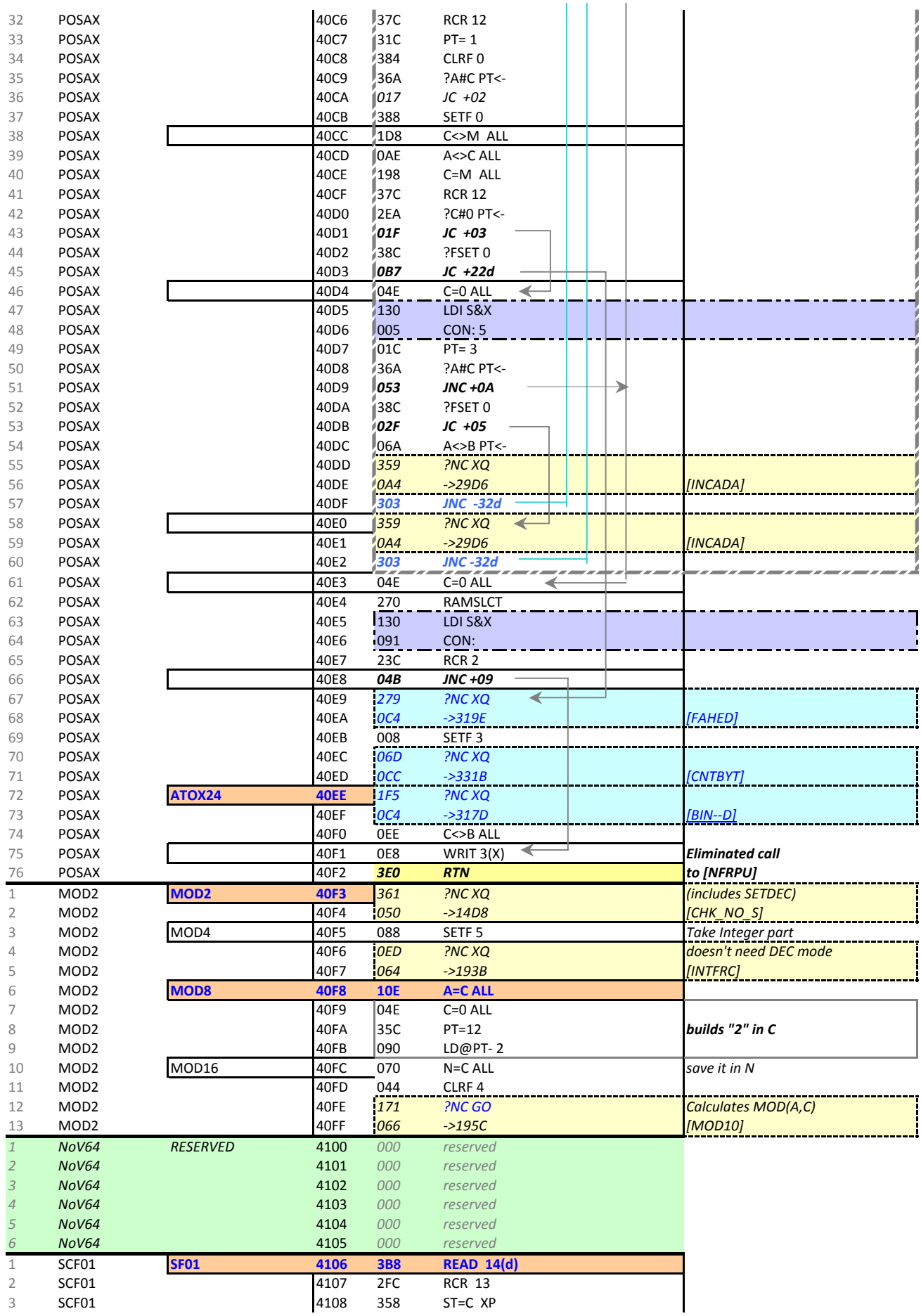
#	Function	Segment	Address	Code	Mnemonic	Comment
1	HELLO		4000	023	JNC +04	signature= 023; xrom 35,:-)
2	HELLO		4001	00B	JNC +01	DUMMY!
3	HELLO		4002	007	JNC +00	Dummy Header
4	HELLO		4003	00B	JNC +01	LIBRARY4
5	HELLO		4004	3E0	RTN	deflect the call
1	DUPRCL	DUPRCL	4005	0B8	READ 2(Y)	duplicateS Z in W
2	DUPRCL		4006	028	WRIT 0(T)	
3	DUPRCL		4007	0F8	READ 3(X)	(scratch!!)
4	DUPRCL		4008	068	WRIT 1(Z)	
5	DUPRCL		4009	1ED	?NC GO	Recall scratch (L,O) to (XY)
6	DUPRCL		400A	106	->417B	[ZRCLS4]
1	APEREX	APERX8	400B	104	CLRF 8	so it doesn't hang on!
2	APEREX		400C	013	JNC +02	
3	APEREX	APERX4	400D	108	SETF 8	so it "sticks"
4	APEREX	APERX0	400E	3DD	?NC XQ	
5	APEREX		400F	0AC	->2BF7	[LEFTJ]
6	APEREX		4010	201	?NC GO	
7	APEREX		4011	072	->1C80	[MSG105]
1	PMASK	PMASK4	4012	04C	?FSET 4	SST'ing a program?
2	PMASK		4013	01F	JC +03	yes, say no more
3	PMASK		4014	2CC	?FSET 13	RUN'ing a program?
4	PMASK		4015	02B	JNC +05	no, skip over
5	PMASK	PMASK0	4016	0F8	READ 3(X)	yes, get arg from X
6	PMASK		4017	38D	?NC XQ	Convert it to hex - uses F8
7	PMASK		4018	008	->02E3	[BCDBIN]
8	PMASK		4019	10E	A=C ALL	leaves it in C and A
9	PMASK		401A	3E0	RTN	
1	E3/E+	E3/E+	401B	0F8	READ 3(X)	
2	E3/E+		401C	361	?NC XQ	(includes SETDEC)
3	E3/E+		401D	050	->14D8	[CHK_NO_S]
4	E3/E+	E3/E+C	401E	266	C=C-1 S&X	
5	E3/E+		401F	266	C=C-1 S&X	
6	E3/E+		4020	266	C=C-1 S&X	divide by 1,000
7	E3/E+		4021	1E1	?NC XQ	Increment C
8	E3/E+		4022	100	->4078	[INCC10]
9	E3/E+		4023	0E8	WRIT 3(X)	
10	E3/E+		4024	3E0	RTN	
1	APNDBK	APNDBK	4025	130	LDI S&X	
2	APNDBK		4026	020	" "	append "blank"
3	APNDBK		4027	3D5	?NC GO	
4	APNDBK		4028	07E	->1FF5	[APND10]
1	<GAP>		4029	000	NOP	
2	<GAP>		402A	000	NOP	
1	ZOUT	ZOUT4	402B	260	SETHEX	
2	ZOUT		402C	215	?NC XQ	Build Msg - all cases
3	ZOUT		402D	0FC	->3F85	[APRMSG2]
4	ZOUT		402E	01A	"Z"	
5	ZOUT		402F	23D	"="	
8	ZOUT		4030	3DD	?NC XQ	
9	ZOUT		4031	0AC	->2BF7	[LEFTJ]
10	ZOUT		4032	399	?NC XQ	Copy Display to Alpha
11	ZOUT		4033	108	->42E6	[DTOA4]
12	ZOUT		4034	0F8	READ 3(X)	real part
13	ZOUT		4035	2EE	?C#0 ALL	is Re(z)#0?
14	ZOUT		4036	033	JNC +06	no, skip real part
15	ZOUT		4037	0EE	C<>B ALL	
16	ZOUT		4038	0A0	SLCT P	
17	ZOUT		4039	1A9	?NC XQ	AFORMT w/ integer support
18	ZOUT		403A	130	->4C6A	[AFRMT4]
19	ZOUT		403B	043	JNC +08	
20	ZOUT		403C	0B8	READ 2(Y)	imaginary part
21	ZOUT		403D	2EE	?C#0 ALL	is Im(z)#0?
22	ZOUT		403E	047	JC +08	yes, display it
23	ZOUT		403F	31C	PT= 1	

24	ZOUT		4040	3ED	?NC XQ	Append Digit
25	ZOUT		4041	07C	->1FFB	[APNDDG]+1
26	ZOUT		4042	**08B**	**JNC +17d**	[EXIT]
27	ZOUT	APPNDY	4043	**0B8**	**READ 2(Y)**	imaginary part
28	ZOUT		4044	2EE	?C#0 ALL	is Im(z)#0?
29	ZOUT		4045	**073**	**JNC +14d**	no, done! [EXIT]
30	ZOUT		4046	0EE	C<>B ALL	
31	ZOUT		4047	2DE	?B#0 MS	See if negative
32	ZOUT		4048	3CD	?C XQ	append "-"
33	ZOUT		4049	07D	->1FF5	[APND-]
34	ZOUT		404A	2DE	?B#0 MS	
35	ZOUT		404B	165	?NC XQ	Append "+" TO Alpha
36	ZOUT		404C	11C	->4759	[APND+]
37	ZOUT		404D	**03E**	**B=0 MS**	**ABS value**
38	ZOUT		404E	179	?NC XQ	Append "J" TO Alpha
39	ZOUT		404F	124	->495E	[APNDJ]
40	ZOUT	**APPND4**	**4050**	**0A0**	**SLCT P**	
41	ZOUT		4051	1A9	?NC XQ	AFORMT w/ integer support
42	ZOUT		4052	130	->4C6A	[AFRMT4]
43	ZOUT	**SHOW4**	**4053**	25D	?NC XQ	Load Status bits
44	ZOUT		4054	01C	->0797	[LDSST0]
45	ZOUT	**SHOW8**	**4055**	191	?NC XQ	
46	ZOUT		4056	00C	->0364	[XAVIEW]
47	ZOUT		4057	3C1	?NC GO	Die gently...
48	ZOUT		4058	002	->00F0	[NFRPU]
1	DUPZ	**DUPZ**	**4059**	**0B8**	**READ 2(Y)**	**duplicate Z in W**
2	DUPZ		405A	028	WRIT 0(T)	(scratch!!)
3	DUPZ		405B	0F8	READ 3(X)	
4	DUPZ		405C	068	WRIT 1(Z)	
5	DUPZ		405D	**3E0**	**RTN**	
1	CHKST	**CHKST4**	**405E**	**078**	**READ 1(Z)**	**Cheks for valid numeric entries**
2	CHKST		405F	10E	A=C ALL	in the four stack registers
3	CHKST		4060	046	C=0 S&X	
4	CHKST		4061	270	RAM SLCT	
5	CHKST		4062	038	READ DATA	
6	CHKST		4063	351	?NC XQ	(includes SETDEC)
7	CHKST		4064	050	->14D4	[CHK_NO_S1]
8	CHKST		4065	023	JNC +04	
9	CHKST	**CHKST3**	**4066**	**078**	**READ 1(Z)**	
10	CHKST		4067	361	?NC XQ	(includes SETDEC)
11	CHKST		4068	050	->14D8	[CHK_NO_S]
12	CHKST	**CHKST2**	**4069**	**0B8**	**READ 2(Y)**	
13	CHKST		406A	10E	A=C ALL	
14	CHKST		406B	0F8	READ 3(X)	
15	CHKST		406C	351	?NC GO	(includes SETDEC)
16	CHKST		406D	052	->14D4	[CHK_NO_S1]
1	UCRUN	**UCRUN**	**406E**	**260**	**SETHEX**	**Run User Code**
2	UCRUN		406F	03C	RCR 3	get adr in C(3:0)
3	UCRUN		4070	226	C=C+1 S&X	2-byte offset
4	UCRUN		4071	226	C=C+1 S&X	
5	UCRUN		4072	10E	A=C ALL	A[(3-0] = addr of first user code word
6	UCRUN		4073	2F9	?NC GO	Transfer to FOCAL
7	UCRUN		4074	0BE	->2FBE	[XRM20]
1	<GAP>		4075	000	NOP	
1	C-ONE	**INCC**	**4076**	361	?NC XQ	(this includes SETDEC)
2	C-ONE		4077	050	->14D8	[CHK_NO_S]
3	C-ONE	**INCC10**	**4078**	**10E**	**A=C ALL**	holds sign and S&X
4	C-ONE		4079	02E	B=0 ALL	clears it
5	C-ONE		407A	0FA	B<>C M	holds 13-digit mant
6	C-ONE		407B	001	?NC GO	13-bit form
7	C-ONE		407C	062	->1800	[ADDONE]
8	C-ONE	**DECC**	**407D**	361	?NC XQ	(this includes SETDEC)
9	C-ONE		407E	050	->14D8	[CHK_NO_S]
10	C-ONE	**DECC10**	**407F**	**10E**	**A=C ALL**	holds sign and S&X
11	C-ONE		4080	02E	B=0 ALL	clears it
12	C-ONE		4081	0FA	B<>C M	holds 13-digit mant
13	C-ONE		4082	009	?NC GO	13-bit form

14	C-ONE		4083	*062*	*->1802*	*[SUBONE]*
1	ISOK?	**ISOK?**	**4084**	**244**	**CLRF 9**	***Check ALPHA="OK" / "OKALL"***
2	ISOK?		4085	1A0	A=B=C=0	
3	ISOK?		4086	270	RAM SLCT	
4	ISOK?		4087	178	READ 5(M)	
5			4088	0AE	A<>C ALL	
6	*safety check for valid action*		4089	01C	PT= 3	
7	*requires key in ALPHA*		408A	110	LD@PT- 4	
8			408B	3D0	LD@PT- F	*4F = "O"*
9	ISOK?		408C	110	LD@PT- 4	
10			408D	2D0	LD@PT- B	*4B = "K"*
11	*The control flag needs to be*		408E	36E	?A#C ALL	
12	*outside from the status bits,*		408F	***063***	***JNC +12d***	
13	*or it will be altered!*		4090	110	LD@PT- 4	
14			4091	050	LD@PT- 1	*41 = "A"*
15	ISOK?		4092	110	LD@PT- 4	
16	ISOK?		4093	310	LD@PT- C	*4C = "L"*
17	ISOK?		4094	110	LD@PT- 4	
18	ISOK?		4095	310	LD@PT- C	*4C = "L"*
19	ISOK?		4096	13C	RCR 8	
20	ISOK?		4097	36E	?A#C ALL	*"OKALL"*
21	ISOK?		4098	*021*	*?C GO*	*Show "NOT OK" msg*
22	ISOK?		4099	*113*	*->4408*	*[NOTOK]*
23	ISOK?		409A	**248**	**SETF 9**	
24	ISOK?		409B	***3B8***	***READ 14(d)***	
25	ISOK?		409C	2FC	RCR 13	
26	ISOK?		409D	3D8	C<>ST XP	
27	ISOK?		409E	204	CLRF 2	*Clear UF 01*
28	ISOK?		409F	**24C**	**?FSET 9**	
29	ISOK?		40A0	*013*	*JNC +02*	
30	ISOK?		40A1	208	SETF 2	*Set UF 01*
31	ISOK?		40A2	3D8	C<>ST XP	
32	ISOK?		40A3	33C	RCR 1	
33	ISOK?		40A4	***3A8***	***WRIT 14(d)***	
34	ISOK?		40A5	*345*	*?NC GO*	*Clears Alpha*
35	ISOK?		40A6	*042*	*->10D1*	*[CLA]*
1	POSAX	**POSA4**	**40A7**	**0F8**	**READ 3(X)**	*HP Co.*
2	POSAX		40A8	27E	C=C-1 MS	
3	POSAX		40A9	2FE	?C#0 MS	*is it negative?*
4			40AA	***05F***	***JC +11d***	*NO*
5	*repetition of same CX code*		40AB	35C	PT= 12	*YES*
6	*to avoid final call to [NFRU]*		40AC	010	LD@PT- 0	
7			40AD	2EE	?C#0 ALL	
8	POSAX		40AE	***1AB***	***JNC +53d***	
9	POSAX		40AF	31C	PT= 1	
10	POSAX		40B0	23C	RCR 2	
11	POSAX		40B1	2EA	?C#0 PT<-	
12	POSAX		40B2	***3F7***	***JC -02***	
13	POSAX		40B3	23C	RCR 2	
14	POSAX		40B4	***033***	***JNC +06***	
15	POSAX		40B5	*115*	*?NC XQ*	
16	POSAX		40B6	*0C8*	*->3245*	*[X<256]*
17	POSAX		40B7	23C	RCR 2	
18	POSAX		40B8	19C	PT= 11	
19	POSAX		40B9	04A	C=0 PT<-	
20	POSAX		40BA	070	N=C ALL	
21	POSAX		40BB	*279*	*?NC XQ*	
22	POSAX		40BC	*0C4*	*->319E*	*[FAHED]*
23	POSAX		40BD	20C	?FSET 2	
24	POSAX		40BE	***12F***	***JC +37d***	
25	POSAX		40BF	08A	B=A PT<-	
26	POSAX		40C0	0B0	C=N ALL	
27	POSAX		40C1	158	M=C ALL	
28	POSAX		40C2	*2ED*	*?NC XQ*	
29	POSAX		40C3	*0A4*	*->29BB*	*[GTBYTA]*
30	POSAX		40C4	0AE	A<>C ALL	
31	POSAX		40C5	1D8	C<>M ALL	



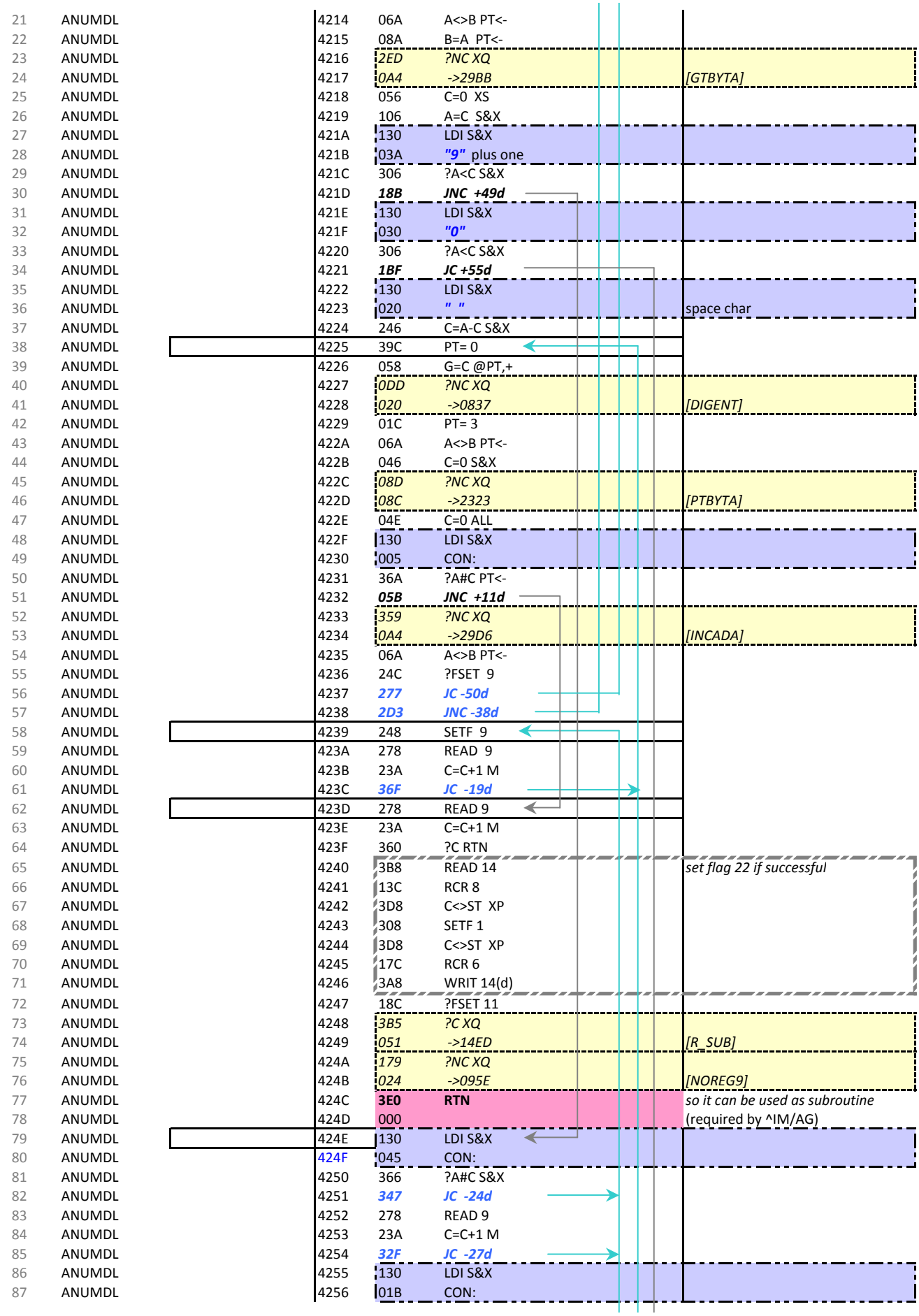
32	POSAX		40C6	37C	RCR 12	
33	POSAX		40C7	31C	PT= 1	
34	POSAX		40C8	384	CLRF 0	
35	POSAX		40C9	36A	?A#C PT<-	
36	POSAX		40CA	*017*	*JC +02*	
37	POSAX		40CB	388	SETF 0	
38	POSAX		40CC	1D8	C<>M ALL	
39	POSAX		40CD	0AE	A<>C ALL	
40	POSAX		40CE	198	C=M ALL	
41	POSAX		40CF	37C	RCR 12	
42	POSAX		40D0	2EA	?C#0 PT<-	
43	POSAX		40D1	***01F***	***JC +03***	
44	POSAX		40D2	38C	?FSET 0	
45	POSAX		40D3	***0B7***	***JC +22d***	
46	POSAX		40D4	04E	C=0 ALL	
47	POSAX		40D5	130	LDI S&X	
48	POSAX		40D6	005	CON: 5	
49	POSAX		40D7	01C	PT= 3	
50	POSAX		40D8	36A	?A#C PT<-	
51	POSAX		40D9	***053***	***JNC +0A***	
52	POSAX		40DA	38C	?FSET 0	
53	POSAX		40DB	***02F***	***JC +05***	
54	POSAX		40DC	06A	A<>B PT<-	
55	POSAX		40DD	*359*	*?NC XQ*	
56	POSAX		40DE	*0A4*	*->29D6*	*[INCADA]*
57	POSAX		40DF	*303*	*JNC -32d*	
58	POSAX		40E0	*359*	*?NC XQ*	
59	POSAX		40E1	*0A4*	*->29D6*	*[INCADA]*
60	POSAX		40E2	*303*	*JNC -32d*	
61	POSAX		40E3	04E	C=0 ALL	
62	POSAX		40E4	270	RAMSLCT	
63	POSAX		40E5	130	LDI S&X	
64	POSAX		40E6	091	CON:	
65	POSAX		40E7	23C	RCR 2	
66	POSAX		40E8	***04B***	***JNC +09***	
67	POSAX		40E9	*279*	*?NC XQ*	
68	POSAX		40EA	*0C4*	*->319E*	*[FAHED]*
69	POSAX		40EB	008	SETF 3	
70	POSAX		40EC	*06D*	*?NC XQ*	
71	POSAX		40ED	*0CC*	*->331B*	*[CNTBYT]*
72	POSAX	**ATOX24**	**40EE**	*1F5*	*?NC XQ*	
73	POSAX		40EF	*0C4*	*->317D*	*[BIN--D]*
74	POSAX		40F0	0EE	C<>B ALL	
75	POSAX		40F1	0E8	WRIT 3(X)	***Eliminated call***
76	POSAX		40F2	***3E0***	***RTN***	***to [NFRPU]***
1	MOD2	**MOD2**	**40F3**	*361*	*?NC XQ*	*(includes SETDEC)*
2	MOD2		40F4	*050*	*->14D8*	*[CHK_NO_S]*
3	MOD2	MOD4	40F5	088	SETF 5	*Take Integer part*
4	MOD2		40F6	*0ED*	*?NC XQ*	*doesn't need DEC mode*
5	MOD2		40F7	*064*	*->193B*	*[INTFRC]*
6	MOD2	**MOD8**	**40F8**	**10E**	**A=C ALL**	
7	MOD2		40F9	04E	C=0 ALL	
8	MOD2		40FA	35C	PT=12	***builds "2" in C***
9	MOD2		40FB	090	LD@PT- 2	
10	MOD2	MOD16	40FC	070	N=C ALL	*save it in N*
11	MOD2		40FD	044	CLRF 4	
12	MOD2		40FE	*171*	*?NC GO*	*Calculates MOD(A,C)*
13	MOD2		40FF	*066*	*->195C*	*[MOD10]*
1	*NoV64*	*RESERVED*	4100	*000*	*reserved*	
2	*NoV64*		4101	*000*	*reserved*	
3	*NoV64*		4102	*000*	*reserved*	
4	*NoV64*		4103	*000*	*reserved*	
5	*NoV64*		4104	*000*	*reserved*	
6	*NoV64*		4105	*000*	*reserved*	
1	SCF01	**SF01**	**4106**	**3B8**	**READ 14(d)**	
2	SCF01		4107	2FC	RCR 13	
3	SCF01		4108	358	ST=C XP	

```
4   SCF01                    4109  208  SETF 2
5   SCF01                    410A  02B  JNC +05
6   SCF01      CF01          410B  3B8  READ 14(d)
7   SCF01                    410C  2FC  RCR 13
8   SCF01                    410D  358  ST=C XP
9   SCF01                    410E  204  CLRF 2
10  SCF01                    410F  398  C=ST XP
11  SCF01                    4110  33C  RCR 1
12  SCF01                    4111  2B1  ?NC GO          update flags annunciator
13  SCF01                    4112  12A  ->4AAC          [UPDATE]
1   ONEPMT     ONEPMT        4113  130  LDI S&X
2   ONEPMT                   4114  003  CON:            pre-load numeric mask
3   ONEPMT                   4115  00C  ?FSET 3         numeric input
4   ONEPMT                   4116  05F  JC +0B          LB_A40E
5   ONEPMT                   4117  04C  ?FSET 4         rows 1 or 2
6   ONEPMT     LB_A405       4118  265  ?NC GO          Blink and return
7   ONEPMT                   4119  022  ->0899          [BLINK]
8                            411A  35E  ?A#0 MS         carry if A[MS]#0 ("J")
9   HEX digit prompt         411B  3EB  JNC -03         LB_A405
10  used during a partial entry.
                             411C  2DC  PT= 13
11  if not valid, returns to next line
                             411D  1D0  LD@PT- 7
12  if valid, skips one line and returns
                             411E  31E  ?A<C MS          carry if A[MS]<7 ("G")
13                           411F  3CB  JNC -07          LB_A405
14  ONEPMT                   4120  046  C=0 S&X         delete numeric mask
15  ONEPMT     LB_A40E       4121  0BE  A<>C MS         get A[MS] to C[MS]
16  ONEPMT                   4122  2FC  RCR 13          form the number code
17  ONEPMT                   4123  3E8  WRIT 15(e)      write chr in LCD
18  ONEPMT                   4124  14D  ?NC GO          skip one line and RTN
19  ONEPMT                   4125  032  ->0C53          [SKIP1]
1   NOROM      NOROM         4126  321  ?NC XQ          Show "NO_" msg
2   NOROM                    4127  10C  ->43C8          [NOMSG4]
3   NOROM                    4128  012  "R"
4   NOROM                    4129  00F  "O"             "ROM"
5   NOROM                    412A  20D  "M"
6   NOROM                    412B  0B3  JNC +22d
1   CHKCFG     REPORT        412C  215  ?NC XQ          Build Msg - all cases
2   CHKCFG                   412D  0FC  ->3F85          [APRMSG2]
3   CHKCFG                   412E  003  "C"
4   CHKCFG                   412F  00F  "O"
5   CHKCFG                   4130  00E  "N"
6   CHKCFG                   4131  006  "F"             "CONFIG "
7   CHKCFG                   4132  009  "I"
8   CHKCFG                   4133  007  "G"
9   CHKCFG                   4134  220  " "
10  CHKCFG                   4135  24C  ?FSET 9         were issues?
11  CHKCFG                   4136  037  JC +06
12  CHKCFG     OKDISP        4137  3BD  ?NC XQ
13  CHKCFG                   4138  01C  ->07EF          [MESSL]
14  CHKCFG                   4139  00F  "O"
15  CHKCFG                   413A  20B  "K"
16  CHKCFG                   413B  033  JNC +06
17  CHKCFG     BADONE        413C  3BD  ?NC XQ
18  CHKCFG                   413D  01C  ->07EF          [MESSL]
19  CHKCFG                   413E  002  "B"
20  CHKCFG                   413F  001  "A"             "BAD"
21  CHKCFG                   4140  204  "D"
22  CHKCFG                   4141  1F1  ?NC GO          Show and Halt
23  CHKCFG                   4142  0FE  ->3F7C          [APEREX]
24  CHKCFG     CHKCFG        4143  1A0  A=B=C=0
25  CHKCFG                   4144  244  CLRF 9
26  CHKCFG                   4145  15C  PT= 6
27  CHKCFG                   4146  090  LD@PT- 2        puts "2" in C(6)
28  CHKCFG                   4147  070  N=C ALL         now in N(6)
29  CHKCFG     NEXTJ         4148  0B0  C=N ALL
30  CHKCFG                   4149  15C  PT= 6
31  CHKCFG                   414A  222  C=C+1 @PT       increases page#
```

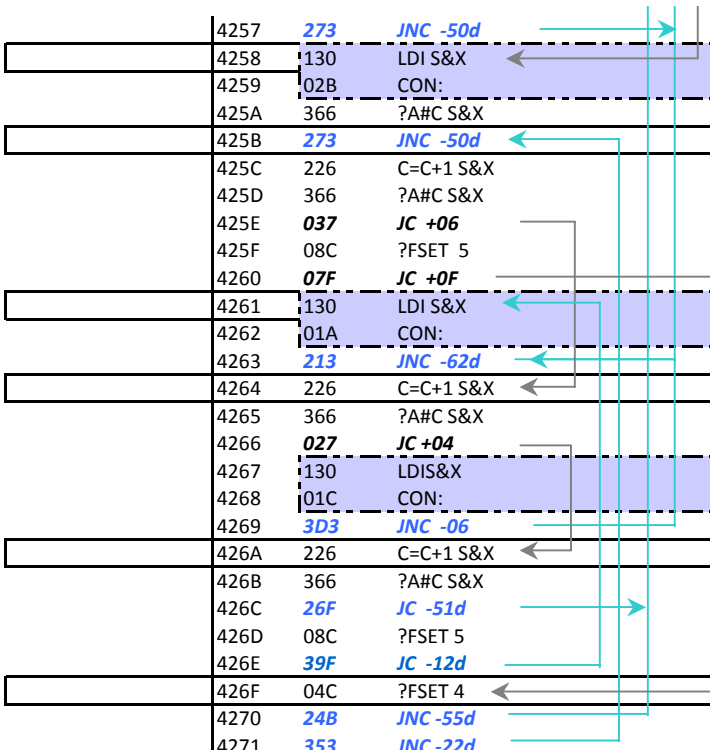
32	CHKCFG		414B	30F	JC -31d	ALLES ok!
33	CHKCFG		414C	070	N=C ALL	update 1st. Counter
34			414D	330	FETCH S&X	get 1st. Word (j)
35	FOR J= 3 TO 14		414E	106	A=C S&X	store in A[S&X]
36	W = FETCH(J)		414F	23A	C=C+1 M	increase address
37	FOR K=J+1 TO 15		4150	330	FETCH S&X	get 2nd. Word (j)
38	V = FETCH (K)		4151	2E6	?C#0 S&X	empty FAT?
39	IF V=W THEN -> BAD STUFF		4152	3B3	JNC -10d	yes, [NEXTKJ]
40	NEXT K		4153	0B0	C=N ALL	bring counter back
41	NEXT J		4154	222	C=C+1 @PT	add one: out of bounds?
42			4155	2BF	JC -41d	yes, we're done
43	CHKCFG		4156	158	M=C ALL	reset 2nd. Counter
44	CHKCFG		4157	033	JNC +06	ist. Time is different
45	CHKCFG	NEXTK	4158	198	C=M ALL	
46	CHKCFG		4159	15C	PT= 6	
47	CHKCFG		415A	222	C=C+1 @PT	increases page#
48	CHKCFG		415B	36F	JC -19d	checked all pages
49	CHKCFG		415C	158	M=C ALL	update 2nd. Counter
50	CHKCFG	1STONE	415D	330	FETCH S&X	get 1st. Word (k)
51	CHKCFG		415E	0E6	C<>B S&X	store in B[S&X]
52	CHKCFG		415F	23A	C=C+1 M	increase address
53	CHKCFG		4160	330	FETCH S&X	get 2nd. Word (k)
54	CHKCFG		4161	2E6	?C#0 S&X	empty FAT?
55	CHKCFG		4162	3B3	JNC -10d	yes, -> [NEXTK]
56	CHKCFG		4163	0C6	C=B S&X	get 1st. Word (k)
57	CHKCFG		4164	366	?A#C S&X	same xrom id#'s ?
58	CHKCFG		4165	39F	JC -13d	no, -> [NEXTK]
59	CHKCFG		4166	248	SETF 9	yes, flag it as "dirty" cfg
60	CHKCFG		4167	215	?NC XQ	Build Msg - all cases
61	CHKCFG		4168	0FC	->3F85	[APRMSG2]
62	CHKCFG		4169	004	"D"	
63	CHKCFG		416A	015	"U"	
64	CHKCFG		416B	010	"P"	
65	CHKCFG		416C	020	" "	"DUP XROM "
66	CHKCFG		416D	018	"X"	
67	CHKCFG		416E	012	"R"	
68	CHKCFG		416F	00F	"O"	
69	CHKCFG		4170	00D	"M"	
70	CHKCFG		4171	220	" "	
71	CHKCFG		4172	066	A<>B S&X	value to convert in A[S&X]
72	CHKCFG		4173	01E	A=0 MS	
73	CHKCFG		4174	3A1	?NC XQ	Generate number ->display!
74	CHKCFG		4175	014	->05E8	[GENNUM]
75	CHKCFG		4176	035	?NC XQ	Display not halting
76	CHKCFG		4177	100	->400D	[APERX4]
77	CHKCFG		4178	3A1	?NC XQ	wait a while
78	CHKCFG		4179	13C	->4FE8	[WAIT4]
79	CHKCFG		417A	2F3	JNC -34d	[NEXTK]
1	ZRCLS	ZRCLS4	417B	1F8	READ 7(O)	Recall scratch (L,O) to (XY)
2	ZRCLS		417C	0A8	WRIT 2(Y)	
3	ZRCLS		417D	138	READ 4(L)	
4	ZRCLS		417E	0E8	WRIT 3(X)	
5	ZRCLS		417F	3E0	RTN	
1	<GAP>		4180	000	NOP	
1	DTOHEX	DTOHEX	4181	0F8	READ 3(X)	From Ken Emey's book
2	DTOHEX	DTOHXC	4182	10E	A=C ALL	
3	DTOHEX		4183	1BE	A=A-1 MS	
4	DTOHEX		4184	1BE	A=A-1 MS	
5			4185	389	?C GO	
6	like the standard OS routine		4186	053	->14E2	[ERRAD]
7	[BCDBIN], but supports		4187	130	LDI S&X	
8	arguments larger than "999"		4188	004	CON: 4	
9	in the C[S&X] field		4189	306	?A<C S&X	
10	(or >4,096 in decimal)		418A	289	?NC GO	
11			418B	002	->00A2	[ERROF]
12	DTOHEX		418C	266	C=C-1 S&X	
13	DTOHEX		418D	0AE	A<>C ALL	

14	DTOHEX		418E	366	?A#C S&X	
15	DTOHEX		418F	***023***	***JNC +04***	
16	DTOHEX		4190	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
17	DTOHEX		4191	*008*	*->02E3*	*[BCDBIN]*
18	DTOHEX		4192	***07B***	***JNC +0F***	
19	DTOHEX		4193	27C	RCR 9	
20	DTOHEX		4194	11A	A=C MS	
21	DTOHEX		4195	05A	C=0 M	
22	DTOHEX		4196	*3E1*	*?NC XQ*	
23	DTOHEX		4197	*008*	*->02F8*	*[GOTINT]*
24	DTOHEX		4198	106	A=C S&X	
25	DTOHEX		4199	01C	PT= 3	
26	DTOHEX		419A	130	LDI S&X	
27	DTOHEX		419B	3E8	CON: 1000	
28	DTOHEX		419C	1A2	A=A-1 @PT	
29	DTOHEX		419D	146	A=A+C S&X	
30	DTOHEX		419E	1A2	A=A-1 @PT	
31	DTOHEX		419F	*3F3*	*JNC -02*	
32	DTOHEX		41A0	0A6	A<>C S&X	
33	DTOHEX		41A1	05E	C=0 MS	
34	DTOHEX		41A2	05A	C=0 M	
35	DTOHEX		41A3	***3E0***	***RTN***	
1	CRBF9	**CRBF9**	**41A4**	130	LDI S&X	
2	CRBF9		41A5	**009**	**CON: 9**	***Buffer id# in C(0)***
3			41A6	*2A9*	*?NC XQ*	*Check for Buffer*
4	*create buffer id#9*		41A7	*10C*	*->43AA*	*[CHKBF4] +1*
5	*used for LASTF stuff*		41A8	*043*	*JNC +08*	*Not Found - Create it !!*
6			41A9	038	READ DATA	***reload id# in MS field***
7	CRBF9		41AA	2DC	PT= 13	
8	CRBF9		41AB	250	LD@PT- 9	Buffer id#
9	CRBF9		41AC	2F0	WRITDATA	*re-brand it*
10	CRBF9		41AD	046	C=0 S&X	reset RAM settings
11	CRBF9		41AE	270	RAMSLCT	*select Chip0*
12	CRBF9		41AF	3E0	RTN	*all done.*
13	CRBF9		41B0	**066**	**A<>B S&X**	*First free reg. address (from .END.)*
14	CRBF9		41B1	046	C=0 S&X	
15	CRBF9		41B2	270	RAMSLCT	*Select Chip0*
16	CRBF9		41B3	*285*	*?NC XQ*	
17	CRBF9		41B4	*014*	*->05A1*	*[MEMLFT]*
18	CRBF9		41B5	106	A=C S&X	*number of "free regs"*
19	CRBF9		41B6	130	LDI S&X	*Must be at least 12 free regs.*
20	CRBF9		41B7	005	CON: 5	*(header + 5 complex stack levels)*
21	CRBF9		41B8	306	?A<C S&X	*Enough Memory?*
22	CRBF9		41B9	*33D*	*?C GO*	*Show "No Room" msg*
23	CRBF9		41BA	*0C3*	*->30CF*	*[NORMER]*
24	CRBF9		41BB	**0E6**	**B<>C S&X**	*First free reg. address (from .END.)*
25	CRBF9		41BC	270	RAM SLCT	select buffer header
26	CRBF9		41BD	**106**	**A=C S&X**	***buffer address in A S&X***
27	CRBF9		41BE	2DC	PT= 13	
28	CRBF9		41BF	250	LD@PT- 9	
29	CRBF9		41C0	250	LD@PT- 9	*Buffer id# to C(12)*
30	CRBF9		41C1	010	LD@PT- 0	
31	CRBF9		41C2	150	LD@PT- 5	*buffer size*
32	CRBF9		41C3	***34B***	***JNC -23d***	***Store Header Reg***
1	TOGF11	**TOGFSF4**	**41C4**	*395*	*?NC XQ*	*Toggles UF26 - enables Chip0*
2	TOGF11		41C5	*07C*	*->1FE5*	*[TOGSHF]*
3	TOGF11	TOGF11	41C6	18C	?FSET 11	*toggle FLAG*
4	TOGF11		41C7	*01B*	*JNC +03*	
5	TOGF11	CLEAR	41C8	184	CLRF 11	
6	TOGF11		41C9	3E0	RTN	
7	TOGF11	SET	41CA	188	SETF 11	
8	TOGF11		41CB	3E0	RTN	
1	RTNSST	**RTNSST**	**41CC**	**1B0**	**POPADR**	*difuse the bomb*
2	RTNSST		41CD	03C	RCR 3	*rotate into position*
3	RTNSST		41CE	358	ST=C XP	*restore flags from RTN Stack*
4	RTNSST		41CF	**3E0**	**RTN**	done.
1	*<GAP>*		41D0	*000*	*NOP*	

1	Y/N?	**Y/N?**	**41D1**	130	LDI S&X	
2	Y/N?		41D2	059	CON:	
3	Y/N?		41D3	*375*	*?NC XQ*	
4	Y/N?		41D4	*058*	*->16DD*	*[TONEB]*
5	Y/N?		41D5	*229*	*?NC XQ*	*do while not key*
6			41D6	*124*	*->498A*	*[RSTKB4] - puts KY in A/C[S&X]*
7	*YES/NO choices*		41D7	130	LDI S&X	*calc OFF*
8	*with control keys (ON, R/S, BA)*		41D8	018	*"ON"*	***"ON" key code***
9			41D9	366	?A#C S&X	
10	Y/N?		41DA	*321*	*?NC GO*	
11	Y/N?		41DB	*046*	*->11C8*	*[OFF]*
12	Y/N?		41DC	130	LDI S&X	*"yes", noskip*
13	Y/N?		41DD	016	***"Y"***	***"Y" keycode***
14	Y/N?		41DE	366	?A#C S&X	
15	Y/N?		41DF	***093***	***JNC +18d***	*[YES]*
16	Y/N?		41E0	130	LDI S&X	*"no", skip*
17	Y/N?		41E1	013	***"N"***	***"N" keycode***
18	Y/N?		41E2	366	?A#C S&X	*is it "N"?*
19	Y/N?		41E3	***083***	***JNC +16d***	*yes, [NO]*
20	Y/N?		41E4	130	LDI S&X	*stops program*
21	Y/N?		41E5	087	***"R/S"***	***"R/S" keycode***
22	Y/N?		41E6	366	?A#C S&X	*is it R/S?*
23	Y/N?		41E7	*211*	*?NC GO*	*yes, reset sequence and done*
24	Y/N?		41E8	*00E*	*->0384*	*[RSTSEQ]*
25	Y/N?		41E9	130	LDI S&X	*cancels it all*
26	Y/N?		41EA	0C3	***"<--"***	***Back-arrow***
27	Y/N?		41EB	366	?A#C S&X	
28	Y/N?		41EC	*3ED*	*?NC GO*	*yes, HALT execution*
29	Y/N?		41ED	*08A*	*->22FB*	*[ERR110]*
30	Y/N?		41EE	*261*	*?NC XQ*	*no, reset keys*
31	Y/N?		41EF	*000*	*->0098*	*[RSTKB]*
32	Y/N?		41F0	***30B***	***JNC - 31d***	*and prompt again*
33	Y/N?	YES	41F1	244	CLRF 9	
34	Y/N?		41F2	*013*	*JNC +02*	*[EXIT]*
35	Y/N?	NO	41F3	248	SETF 9	
36	Y/N?	EXIT	41F4	*239*	*?NC XQ*	
37	Y/N?		41F5	*00C*	*->038E*	*[RSTMS0]*
38	Y/N?		41F6	24C	?FSET 9	
39	Y/N?		41F7	*0B9*	*?C GO*	
40	Y/N?		41F8	*05B*	*->162E*	*[SKP]*
41	Y/N?		41F9	*065*	*?NC GO*	
42	Y/N?		41FA	*05A*	*->1619*	*[NOSKP]*
1	ZSCRTCH	**ZSTOS4**	**41FB**	**0B8**	**READ 2(Y)**	***Scratch registers 4(L) & 7(O)***
2	ZSCRTCH	***relocated in***	41FC	1E8	WRIT 7(O)	*Im(z) to Alpha (O)*
3	ZSCRTCH	***revision "K"***	41FD	0F8	READ 3(X)	
4	ZSCRTCH		41FE	128	WRIT 4(L)	*Re(z) to Alpha (N)*
5	ZSCRTCH		41FF	**3E0**	**RTN**	
1	ANUMDL	**ANUMDL**	**4200**	*279*	*?NC XQ*	
2	ANUMDL		4201	*0C4*	*->319E*	*[FAHED]*
3	ANUMDL		4202	20C	?FSET 2	
4	ANUMDL		4203	360	***?C RTN***	
5	ANUMDL		4204	08A	B=A PT<-	
6	ANUMDL		4205	04E	C=0 ALL	
7	ANUMDL		4206	26E	C=C-1 ALL	
8	ANUMDL		4207	056	C=0 XS	
9	ANUMDL		4208	2DC	PT= 13	
10	ANUMDL		4209	290	LD@PT A	
11	ANUMDL		420A	268	WRIT 9(Q)	
12	ANUMDL		420B	*28D*	*?NC XQ*	
13	ANUMDL		420C	*0B8*	*->2EA3*	*[STBT10]*
14	ANUMDL		420D	1BC	RCR 11	
15	ANUMDL		420E	008	SETF 3	
16	ANUMDL		420F	398	C=ST XP	
17	ANUMDL		4210	03C	RCR 3	
18	ANUMDL		4211	228	WRIT 8(P)	
19	ANUMDL		4212	244	CLRF 0	
20	ANUMDL		4213	01C	PT= 3	



21	ANUMDL	4214	06A	A<>B PT<-		
22	ANUMDL	4215	08A	B=A PT<-		
23	ANUMDL	4216	2ED	?NC XQ		
24	ANUMDL	4217	0A4	->29BB		[GTBYTA]
25	ANUMDL	4218	056	C=0 XS		
26	ANUMDL	4219	106	A=C S&X		
27	ANUMDL	421A	130	LDI S&X		
28	ANUMDL	421B	03A	"9" plus one		
29	ANUMDL	421C	306	?A<C S&X		
30	ANUMDL	421D	18B	JNC +49d		
31	ANUMDL	421E	130	LDI S&X		
32	ANUMDL	421F	030	"0"		
33	ANUMDL	4220	306	?A<C S&X		
34	ANUMDL	4221	1BF	JC +55d		
35	ANUMDL	4222	130	LDI S&X		
36	ANUMDL	4223	020	" "		space char
37	ANUMDL	4224	246	C=A-C S&X		
38	ANUMDL	4225	39C	PT= 0		
39	ANUMDL	4226	058	G=C @PT,+		
40	ANUMDL	4227	0DD	?NC XQ		
41	ANUMDL	4228	020	->0837		[DIGENT]
42	ANUMDL	4229	01C	PT= 3		
43	ANUMDL	422A	06A	A<>B PT<-		
44	ANUMDL	422B	046	C=0 S&X		
45	ANUMDL	422C	08D	?NC XQ		
46	ANUMDL	422D	08C	->2323		[PTBYTA]
47	ANUMDL	422E	04E	C=0 ALL		
48	ANUMDL	422F	130	LDI S&X		
49	ANUMDL	4230	005	CON:		
50	ANUMDL	4231	36A	?A#C PT<-		
51	ANUMDL	4232	05B	JNC +11d		
52	ANUMDL	4233	359	?NC XQ		
53	ANUMDL	4234	0A4	->29D6		[INCADA]
54	ANUMDL	4235	06A	A<>B PT<-		
55	ANUMDL	4236	24C	?FSET 9		
56	ANUMDL	4237	277	JC -50d		
57	ANUMDL	4238	2D3	JNC -38d		
58	ANUMDL	4239	248	SETF 9		
59	ANUMDL	423A	278	READ 9		
60	ANUMDL	423B	23A	C=C+1 M		
61	ANUMDL	423C	36F	JC -19d		
62	ANUMDL	423D	278	READ 9		
63	ANUMDL	423E	23A	C=C+1 M		
64	ANUMDL	423F	360	?C RTN		
65	ANUMDL	4240	3B8	READ 14		set flag 22 if successful
66	ANUMDL	4241	13C	RCR 8		
67	ANUMDL	4242	3D8	C<>ST XP		
68	ANUMDL	4243	308	SETF 1		
69	ANUMDL	4244	3D8	C<>ST XP		
70	ANUMDL	4245	17C	RCR 6		
71	ANUMDL	4246	3A8	WRIT 14(d)		
72	ANUMDL	4247	18C	?FSET 11		
73	ANUMDL	4248	3B5	?C XQ		
74	ANUMDL	4249	051	->14ED		[R_SUB]
75	ANUMDL	424A	179	?NC XQ		
76	ANUMDL	424B	024	->095E		[NOREG9]
77	ANUMDL	424C	3E0	RTN		so it can be used as subroutine
78	ANUMDL	424D	000			(required by ^IM/AG)
79	ANUMDL	424E	130	LDI S&X		
80	ANUMDL	424F	045	CON:		
81	ANUMDL	4250	366	?A#C S&X		
82	ANUMDL	4251	347	JC -24d		
83	ANUMDL	4252	278	READ 9		
84	ANUMDL	4253	23A	C=C+1 M		
85	ANUMDL	4254	32F	JC -27d		
86	ANUMDL	4255	130	LDI S&X		
87	ANUMDL	4256	01B	CON:		



88	ANUMDL		4257	273	JNC -50d	
89	ANUMDL		4258	130	LDI S&X	
90	ANUMDL		4259	02B	CON:	
91	ANUMDL		425A	366	?A#C S&X	
92	ANUMDL		425B	273	JNC -50d	
93	ANUMDL		425C	226	C=C+1 S&X	
94	ANUMDL		425D	366	?A#C S&X	
95	ANUMDL		425E	037	JC +06	
96	ANUMDL		425F	08C	?FSET 5	
97	ANUMDL		4260	07F	JC +0F	
98	ANUMDL		4261	130	LDI S&X	
99	ANUMDL		4262	01A	CON:	
100	ANUMDL		4263	213	JNC -62d	
101	ANUMDL		4264	226	C=C+1 S&X	
102	ANUMDL		4265	366	?A#C S&X	
103	ANUMDL		4266	027	JC +04	
104	ANUMDL		4267	130	LDIS&X	
105	ANUMDL		4268	01C	CON:	
106	ANUMDL		4269	3D3	JNC -06	
107	ANUMDL		426A	226	C=C+1 S&X	
108	ANUMDL		426B	366	?A#C S&X	
109	ANUMDL		426C	26F	JC -51d	
110	ANUMDL		426D	08C	?FSET 5	
111	ANUMDL		426E	39F	JC -12d	
112	ANUMDL		426F	04C	?FSET 4	
113	ANUMDL		4270	24B	JNC -55d	
114	ANUMDL		4271	353	JNC -22d	
1	NSWAP	NSWAP	4272	0F8	READ 3(X)	character in N
2	NSWAP		4273	128	WRIT 4(L)	preserve original X
3	NSWAP		4274	0B0	C=N ALL	
4			4275	106	A=C S&X	
5	swaps ALPHA around		4276	1F5	?NC XQ	
6	character w/ code in N[S&X]		4277	0C4	->317D	[BIN--D]
7			4278	0EE	C<>B ALL	get BCD number to C
8	NSWAP		4279	0E8	WRIT 3(X)	write it in X
9	NSWAP		427A	0B0	C=N ALL	adds target chr to ALPHA
10	NSWAP		427B	3D5	?NC XQ	
11	NSWAP		427C	07C	->1FF5	[APND10]
12	NSWAP		427D	29D	?NC XQ	Position in Alpha
13	NSWAP		427E	100	->40A7	[POSA4]
14	NSWAP		427F	2A0	SETDEC	
15	NSWAP		4280	1E1	?NC XQ	Increment C
16	NSWAP		4281	100	->4078	[INCC10]
17	NSWAP		4282	0E8	WRIT 3(X)	
18	NSWAP		4283	260	SETHEX	
19	NSWAP		4284	349	?NC XQ	
20	NSWAP		4285	0C4	->31D2	[AROT]
21	NSWAP		4286	2E5	?NC XQ	Deletes Alpha right chr
22	NSWAP		4287	110	->44B9	[ABSP4]
23	NSWAP		4288	138	READ 4(L)	restore original X
24	NSWAP		4289	331	?NC GO	
25	NSWAP		428A	002	->00CC	[NFRX]
1	<GAP>		428B	000	NOP	
2	<GAP>		428C	000	NOP	
1	PMTFNM	PMTFNL	428D	128	WRIT 4(L)	for convenience
2	PMTFNM	PMTFNM	428E	3C1	?NC XQ	Enable & Clear Display
3	PMTFNM		428F	0B0	->2CF0	[CLLCDE]
4	PMTFNM		4290	198	C=M ALL	fct id#
5	PMTFNM		4291	34D	?NC GO	
6	PMTFNM		4292	016	->05D3	[PROMF2]
1	AINT	AINT4	4293	0F8	READ 3(X)	
2	AINT		4294	361	?NC XQ	
3	AINT		4295	050	->14D8	[CHK$S]
4			4296	088	SETF 5	take integer
5	adds integer part to Alpha		4297	0ED	?NC XQ	doesn't need DEC mode
6	including sign if <0 (!)		4298	064	->193B	[INTFRC]
7			4299	0EE	C<>B ALL	needed for [AFORMT]

8	AINT	AINT8	429A	3B8	READ 14(d)	
9	AINT		429B	268	WRIT 9(Q)	save settings in Q
10	AINT		429C	05C	PT= 4	
11	AINT		429D	010	LD@PT-- 0	Sets 0 digits display
12	AINT		429E	210	LD@PT-- 8	Sets Flag 40(FIX mode)
13	AINT		429F	15C	PT= 6	
14	AINT		42A0	010	LD@PT- 0	Clear Flag 29 (digit grouping)
15	AINT		42A1	3A8	WRIT 14(d)	
16	AINT	AINT10	42A2	0A1	?NC XQ	
17	AINT		42A3	018	->0628	[AFORMT]
18	AINT		42A4	278	READ 9(Q)	restore initial settings
19	AINT		42A5	3A8	WRIT 14(d)	
20	AINT		42A6	3E0	RTN	
26	CLACHR	CLA-	42A7	130	LDI S&X	
27	CLACHR		42A8	02D	"-"	
28	CLACHR		42A9	033	JNC + 06	
29	CLACHR	CLA>	42AA	130	LDI S&X	
30	CLACHR		42AB	03E	">"	
31	CLACHR		42AC	01B	JNC +03	
32	CLACHR	CLAC	42AD	130	LDI S&X	
33	CLACHR		42AE	02C	<comma>	
34	CLACHR	CLACHR	42AF	31C	PT= 1	
35	CLACHR		42B0	10A	A=C PT<-	
36	CLACHR		42B1	178	READ 5(M)	
37	CLACHR		42B2	36A	?A#C PT<-	
38			42B3	02F	JC +05	
39	expects char code in C[S&X]		42B4	2E5	?NC XQ	
40			42B5	110	->44B9	[ABS4P]
41	CLACHR		42B6	3DB	JNC -05	
42	CLACHR		42B7	178	READ 5(M)	
43	CLACHR		42B8	2EA	?C#0 PT<-	
44	CLACHR		42B9	345	?NC GO	
45	CLACHR		42BA	042	->10D1	[CLA]
46	CLACHR		42BB	36A	?A#C PT<-	
47			42BC	027	JC +04	
48	leaves the comma in the display		42BD	23C	RCR 2	
49	(use ABSP if needs removing)		42BE	36A	?A#C PT<-	
50			42BF	360	?C RTN	ends here.
51	CLACHR		42C0	2E5	?NC XQ	
52	CLACHR		42C1	110	->44B9	[ABSP4]
53	CLACHR		42C2	3AB	JNC -11d	
1	ST<>A	ST<>A4	42C3	130	LDI S&X	
2	ST<>A		42C4	003	CON: 3	X register address
3	ST<>A		42C5	106	A=C S&X	save it in A
4	ST<>A		42C6	04E	C=0 ALL	C=A does NOT exist!
5	ST<>A		42C7	206	C=C+A S&X	C=i#
6	ST<>A		42C8	270	RAM SLCT	Select i#
7			42C9	038	READ DATA	St(i#)
8	expects chip0 enabled		42CA	0EE	C<>B ALL	Save St(i#) in B
9			42CB	130	LDI S&X	
10	ST<>A		42CC	008	CON: 8	P register address
11	ST<>A		42CD	0A6	A<>C S&X	
12	ST<>A		42CE	246	C=A-C S&X	j# = 8 - i#
13	ST<>A		42CF	270	RAM SLCT	select j#
14	ST<>A		42D0	246	C=A-C S&X	k#= 8 - (8-i#) = i#
15	ST<>A		42D1	0A6	A<>C S&X	restore i# into A
16	ST<>A		42D2	038	READ DATA	A(j#)
17	ST<>A		42D3	0EE	C<>B ALL	St(i#)
18	ST<>A		42D4	2F0	WRIT DATA	writes St(i#) into j#
19	ST<>A		42D5	04E	C=0 ALL	C=A does NOT exist!
20	ST<>A		42D6	206	C=C+A S&X	C=i#
21	ST<>A		42D7	270	RAM SLCT	select i#
22	ST<>A		42D8	0EE	C<>B ALL	
23	ST<>A		42D9	2F0	WRIT DATA	writes A(j#) into i#
24	ST<>A		42DA	1A6	A=A-1 S&X	Will set Carry when A=0
25	ST<>A		42DB	35B	JNC -21d	
26	ST<>A		42DC	3E0	RTN	

1	CLEM4	**CLEM4**	**42DD**	130	LDI S&X	
2	CLEM4		42DE	040	CON: 64	
3			42DF	270	RAM SLCT	
4	*clears XMEM master register*		42E0	04E	C=0 ALL	
5			42E1	2F0	WRIT DATA	
6	CLEM4		42E2	2CC	?FSET 13	*running a program?*
7	CLEM4		42E3	*059*	*?NC GO*	*no, show message*
8	CLEM4		42E4	*0F2*	*->3C16*	*[EMDIR]*
9	CLEM4		42E5	**3E0**	**RTN**	*yes, end here.*
1	DTOA4	**DTOA4**	**42E6**	*149*	*?NC XQ*	*Disable PER, enable RAM*
2	DTOA4		42E7	*024*	*->0952*	*[ENCP00]*
3	DTOA4	DTOA8	42E8	*345*	*?NC XQ*	*Clears Alpha*
4	DTOA4		42E9	*040*	*->10D1*	*[CLA]*
5	DTOA4	**NXTCHR**	**42EA**	*3D9*	*?NCXQ*	*Enable but not Clear LCD*
6	DTOA4		42EB	*01C*	*->07F6*	*[ENLCD]*
7	DTOA4		42EC	130	LDI S&X	
8			42ED	***020***	***" "***	*blank space*
9	*display to Alpha*		42EE	0E6	C<>B S&X	*parking lot*
10	*deals with all chrs. types*		42EF	***3F8***	***READ 15(e)***	*read from left and rotate right*
11	*assumes it's left-justified*		42F0	0A6	A<>C S&X	
12			42F1	130	LDI S&X	magic mask:
13	DTOA4		42F2	1FF	CON:	*"0001|1111|1111"*
14	DTOA4		42F3	3B0	C=C AND A	*cleanse weird bits (!)*
15	DTOA4		42F4	066	A<>B S&X	put "020" in A[S&X]
16	DTOA4		42F5	366	?A#C S&X	*a space signals the end*
17	DTOA4		42F6	***0A3***	***JNC +20d***	***wrap up***
18	DTOA4		42F7	106	A=C S&X	*save cleansed chr# in A*
19	DTOA4		42F8	276	C=C-1 XS	*carry set if STNDARD*
20	DTOA4		42F9	***02F***	***JC +05***	*[STDCHR]*
21	DTOA4	SPCLCHR	42FA	130	LDI S&X	*lower case chars here*
22	DTOA4		42FB	0A0	CON:	*need adjustment*
23	DTOA4		42FC	1C6	A=A-C S&X	*sustract A0 from LCD code*
24	DTOA4		42FD	***03B***	***JNC +07***	*[TOALPHA]*
25	DTOA4	STDCHR	42FE	130	LDI S&X	*"Z" is the last one*
26	DTOA4		42FF	***020***	***"_"***	*requiring the Alpha mask*
27	DTOA4		4300	306	?A<C S&X	is it "normal" chr (A-Z...)
28	DTOA4		4301	***01B***	***JNC +03***	*no, leave it alone*
29	DTOA4	UPPRCASE	4302	1E6	C=C+C S&X	*yes, write ALPHA mask! ("040")*
30	DTOA4		4303	146	A=A+C S&X	*add it to chr#*
31	DTOA4	TOALPHA	4304	*149*	*?NC XQ*	*Disable PER, enable RAM*
32	DTOA4		4305	*024*	*->0952*	*[ENCP00]*
33	DTOA4		4306	0A6	A<>C S&X	*get new chr value*
34	DTOA4		4307	*3D5*	*?NC XQ*	*Append to Alpha*
35	DTOA4		4308	*07C*	*->1FF5*	*[APND10]*
36	DTOA4		4309	***30B***	***JNC -31d***	*[NXTCHR]*
37	DTOA4	WRAPUP	430A	*149*	*?NC XQ*	*Disable PER, enable RAM*
38	DTOA4		430B	*024*	*->0952*	*[ENCP00]*
39	DTOA4		430C	*261*	*?NC GO*	*debounce keyboard*
40	DTOA4		430D	*002*	*->0098*	*[RSTKB]*
1	DTST4	**DTST4**	**430E**	*3C1*	*?NC XQ*	*Enable & Clear Disp*
2	DTST4		430F	*0B0*	*->2CF0*	*[CLLCDE]*
3	DTST4		4310	19C	PT= 11	
4	DTST4		4311	390	LD@PT- E	
5	DTST4		4312	010	LD@PT- 0	
6	DTST4		4313	2D4	?PT= 13	
7			4314	***3EB***	***JNC -03***	
8	*test display LCD's*		4315	0E8	WRIT 3(X)	
9			4316	0E8	WRIT 3(X)	*needed twice!*
10	DTST4		4317	046	C=0 S&X	*Delay loop*
11	DTST4		4318	2A6	C=-C-1 S&X	*counter value: FFF*
12	DTST4		4319	266	C=C-1 S&X	
13	DTST4		431A	*3FB*	*JNC -01*	
14	DTST4		431B	19C	PT= 11	
15	DTST4		431C	2D0	LD@PT- B	
16	DTST4		431D	290	LD@PT- A	
17	DTST4		431E	2D4	?PT= 13	
18	DTST4		431F	***3EB***	***JNC -03***	

19	DTST4		4320	0E8	WRIT 3(X)	
20	DTST4		4321	0E8	WRIT 3(X)	
21	DTST4		4322	046	C=0 S&X	
22	DTST4		4323	2A6	C=-C-1 S&X	
23	DTST4		4324	2F0	WRIT DATA	
24	DTST4		4325	046	C=0 S&X	
25	DTST4		4326	3F0	PRPH SLCT	
26	DTST4		4327	*1FD*	*?NC XQ*	
27	DTST4		4328	*00C*	*->037F*	*[STMSGF]_1*
28	DTST4		4329	060	POWOFF	
29	DTST4		432A	000	NOP	*Must be here!!*
30	DTST4		432B	**3E0**	**RTN**	
1	SDGT4	**SDGT4**	**432C**	**0F8**	**READ 3(X)**	
2	SDGT4		432D	00E	A=0 ALL	*initial sum =0*
3	SDGT4		432E	39C	PT= 0	
4	SDGT4	NXTDGT	432F	33C	RCR 1	
5	SDGT4		4330	3C6	RSHFC S&X	
6	*Mantissa Digit Sum*		4331	3C6	RSHFC S&X	
7			4332	146	A=A+C S&X	*add to previous sum*
8	SDGT4		4333	3DC	PT=PT+1	
9	SDGT4		4334	0D4	?PT= 10	
10	SDGT4		4335	*3D3*	*JNC -06*	*[NEXTD]*
11	SDGT4		4336	3E0	RTN	
1	RCL4	**RCL4**	**4337**	**18C**	**?FSET 11**	
2	RCL4		4338	*3B5*	*?C XQ*	
3			4339	*051*	*->14ED*	*[R^SUB]*
4	*repetition of OS code*		433A	04E	C=0 ALL	
5	*removing call to [NFRP]*		433B	270	RAMSLCT	
6			433C	0EE	B<>C ALL	
7	RCL4		433D	0E8	WRIT 3(X)	
8	RCL4		433E	3E0	RTN	
1	ROMCAT	**ROMCAT**	**433F**	**026**	**B=0 S&X**	
2	ROMCAT		4340	05A	C=0 M	
3	ROMCAT		4341	15C	PT= 6	
4			4342	110	LD@PT- 4	
5	*expects ROM id# in A[S&X]*		4343	15C	PT= 6	
6	ROMCAT		4344	222	C=C+1 @PT	
7	ROMCAT		4345	*381*	*?C GO*	
8	ROMCAT		4346	*00B*	*->02E0*	*[ERRNE]*
9	ROMCAT		4347	330	FETCH S&X	
10	ROMCAT		4348	366	?A#C S&X	
11	ROMCAT		4349	***043***	***JNC +08***	
12	ROMCAT		434A	23A	C=C+1 M	
13	ROMCAT		434B	330	FETCH S&X	
14	ROMCAT		434C	066	A<>B S&X	
15	ROMCAT		434D	146	A=A+C S&X	
16	ROMCAT		434E	066	A<>B S&X	
17	ROMCAT		434F	27A	C=C-1 M	
18	ROMCAT		4350	*3A3*	*JNC -0C*	
19	ROMCAT		4351	238	READ 8(P)	
20	ROMCAT		4352	0FC	RCR 10	
21	ROMCAT		4353	0C6	C=B S&X	
22	ROMCAT		4354	07C	RCR 4	
23	ROMCAT		4355	2DC	PT= 13	
24	ROMCAT		4356	090	LD@PT- 2	
25	ROMCAT		4357	*231*	*?NC GO*	
26	ROMCAT		4358	*02E*	*->0B8C*	*[PTCNTR]*
1	UCRUN2	**UCRUN2**	**4359**	**1B0**	**POPADR**	*calling adr in C[6:3]*
2	*expects adr in A[S&X]*		435A	03C	RCR 3	*get pg# to C(3)*
3	*calling point NOT included*		435B	11A	A=C M	*Address in A[3:0]*
4			435C	*2F9*	*?NC GO*	*Transfer to FOCAL*
5	UCRUN2		435D	*0BE*	*->2FBE*	*[XRM20]*
1	GETRG#	**GETRG#**	**435E**	**1A0**	**A=B=C=0**	*Written by W. Doug Wilder*
2	GETRG#		435F	158	M=C ALL	
3	GETRG#		4360	*141*	*?NC XQ*	*get were we're at*
4	GETRG#		4361	*0A4*	*->2950*	*[GETPC]*

5	GETRG#		4362	*01D*	*?NC XQ*	*fetch first byte after us*
6	GETRG#		4363	*0B4*	*->2D07*	*[NXBYT]*
7	GETRG#		4364	056	C=0 XS	
8	GETRG#		4365	2E6	?C#0 S&X	*?packable null*
9	GETRG#		4366	***3E3***	***JNC -04***	*get another*
10	GETRG#		4367	39C	PT= 0	
11	GETRG#		4368	06E	A<>B ALL	*save location in B*
12	GETRG#		4369	106	A=C S&X	*put fetched byte in A S&X*
13	GETRG#		436A	130	LDI S&X	
14	GETRG#		436B	01A	CON: 26	
15	GETRG#		436C	306	?A<C S&X	*?greater than a number*
16	GETRG#		436D	***06B***	***JNC +13d***	*get out out of the loop*
17	GETRG#		436E	042	C=0 @PT	
18	GETRG#		436F	306	?A<C S&X	
19	GETRG#		4370	***057***	***JC +10d***	
20	GETRG#		4371	1D8	C<>M ALL	*push nibble into the right of M*
21	GETRG#		4372	2FC	RCR 13	
22	GETRG#		4373	0A2	A<>C @PT	
23	GETRG#		4374	1D8	C<>M ALL	
24	GETRG#		4375	*019*	*?NC XQ*	*get next byte*
25	GETRG#		4376	*0B4*	*->2D06*	*[NBYTAB]*
26	GETRG#		4377	17E	A=A+1 S&X	
27	GETRG#		4378	056	C=0 XS	
28	GETRG#		4379	***373***	***JNC -12***	
29	GETRG#		437A	06E	A<>B ALL	*get back location*
30	GETRG#		437B	*31D*	*?NC XQ*	*backstep pointer*
31	GETRG#		437C	*0A4*	*->29C7*	*[DECAD]*
32			437D	346	?A#0 S&X	*did we load any digits?*
33	*skip as many bytes*		437E	*0BD*	*?C XQ*	*only update PC if number followed*
34			437F	*08D*	*->232F*	*[PUTPCX]*
35	GETRG#		4380	198	C=M ALL	
36	GETRG#		4381	05A	C=0 M	
37	GETRG#		4382	*3E1*	*?NC XQ*	*Make binary*
38	GETRG#		4383	*008*	*->02F8*	*[GOTINT]*
39	GETRG#		4384	106	A=C S&X	*Store the number in A*
40	GETRG#		4385	**3E0**	**RTN**	*also present in C[S&X]*
1	EXISTS	**EXISTS**	**4386**	130	LDI S&X	
2	EXISTS		4387	080	***<IND> mask***	
3	EXISTS		4388	306	?A<C S&X	*Carry if addr < 80*
4			4389	***057***	***JC +10d***	***NOT INDIRECT***
5	*expects reg# in A[S&X]*		438A	1C6	A=A-C S&X	*remove the IND code*
6	*and block size in B[S&X]*		438B	***255***	*?NC XQ*	*check for valid IND addr*
7			438C	*10C*	*->4395*	*[VALID]*
8	EXISTS		438D	0A6	A<>C S&X	
9	EXISTS		438E	270	RAM SLCT	*select Data Reg#*
10	EXISTS		438F	038	READATA	*READ Data Reg#*
11	EXISTS		4390	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
12	EXISTS		4391	*008*	*->02E3*	*[BCDBIN]*
13	EXISTS		4392	0A6	A<>C S&X	*Save it in A*
14	EXISTS		4393	***0C6***	***C=B S&X***	*block size to add*
15	EXISTS		4394	146	A=A+C S&X	*last reg = (RG#+bsize)*
16	EXISTS	**VALID**	**4395**	**046**	**C=0 S&X**	
17	EXISTS		4396	270	RAMSLCT	*Select Chip0*
18	EXISTS		4397	378	READ 13(c)	
19	EXISTS		4398	03C	RCR 3	*Obtain R00 address*
20	EXISTS		4399	146	A=A+C S&X	*absolute address: R00+Last#*
21	EXISTS	**VALD10**	**439A**	130	LDI S&X	
22	EXISTS		439B	200	CON: 512	*max reg# on the CX +1*
23	EXISTS		439C	306	?A<C S&X	*Carry if address<512*
24	EXISTS		439D	*381*	*?NC GO*	*Displays"NonExistent"*
25	EXISTS		439E	*00A*	*->02E0*	*[ERRNE]*
26	EXISTS		439F	**3E0**	**RTN**	***Returns with addr in A***
1	BUFFER	**CHKBFX**	**43A0**	**0F8**	**READ 3(X)**	
2	BUFFER		43A1	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
3	BUFFER		43A2	*008*	*->02E3*	*[BCDBIN]*
4	BUFFER		43A3	106	A=C S&X	*buffer id# to A*
5	BUFFER		43A4	130	LDI S&X	

```
6  BUFFER                  43A5  00F  CON:
7  BUFFER                  43A6  306  ?A<C S&X            ID# must be < 15
8  BUFFER                  43A7  0B5  ?NC GO
9  BUFFER                  43A8  0A2  ->282D             [ERRDE]
10 BUFFER  CHKBUF          43A9  0A6  A<>C S&X           recall id# to C(0)
11 BUFFER  CHKBF4          43AA  23C  RCR 2              id# to C(12)
12 BUFFER                  43AB  35C  PT= 12
13 BUFFER                  43AC  130  LDI S&X
14 BUFFER                  43AD  0BF  CON: 191           Fisrt possible reg -1
15 BUFFER                  43AE  10E  A=C ALL            store id# & addr in A
16 BUFFER  CB10            43AF  166  A=A+1 S&X          Increase reg# address
17 BUFFER  CB20            43B0  046  C=0 S&X
18 BUFFER                  43B1  270  RAMSLCT            Select Chip 0
19 BUFFER                  43B2  378  READ 13(c)         .END.
20 BUFFER                  43B3  306  ?A<C S&X           did we reach the .END. Chainhead?
21 BUFFER                  43B4  3A0  ?NC RTN            yes -> Not Found
22 BUFFER                  43B5  0A6  A<>C S&X           addr to C[S&X]
23 BUFFER                  43B6  270  RAM SLCT           Candidate address for header
24 BUFFER                  43B7  0A6  A<>C S&X           id# to A(12) & addr to A[S&X]
25 BUFFER                  43B8  038  READATA            Candidate Value for header
26 BUFFER                  43B9  2EE  ?C#0 ALL           Carry if not empty register
27 BUFFER                  43BA  3A0  ?NC RTN            empty reg -> Not Found
28 BUFFER                  43BB  23E  C=C+1 MS           Carry if  id#="F" (KAR)
29 BUFFER                  43BC  39F  JC  -13d           Key Assignment Register
30 BUFFER                  43BD  362  ?A#C @PT           is this IO Buffer?
31 BUFFER                  43BE  037  JC  +06            NO , keep searching
32 BUFFER                  43BF  1B0  POPADR             YES  !
33 BUFFER                  43C0  23A  C=C+1 M            Return to (P+2)
34 BUFFER                  43C1  170  PUSHADR
35 BUFFER                  43C2  038  READATA            Return with Header in C
36 BUFFER                  43C3  3E0  RTN                and BuffAdr in A - rg# selected
37 BUFFER  CB30            43C4  0FC  RCR 10             Skip Buffer
38 BUFFER                  43C5  056  C=0 XS
39 BUFFER                  43C6  146  A=A+C S&X          add buffer size
40 BUFFER                  43C7  34B  JNC -23d           [CB20]

1  NOMSG   NOMSG           43C8  215  ?NC XQ            Build Msg - all cases
2  NOMSG                   43C9  0FC  ->3F85            [APRMSG2]
3  NOMSG                   43CA  00E  "N"
4  NOMSG                   43CB  00F  "O"               "NO "
5  NOMSG                   43CC  220  " "
6  NOMSG                   43CD  3BD  ?NC GO
7  NOMSG                   43CE  01E   ->07EF          [MESSL]

1  BUFMSG  DUPBUF          43CF  20D  ?NC XQ            Build Msg - UF25 Clear
2  BUFMSG                  43D0  0FC  ->3F83            [APERMSG]
3  BUFMSG                  43D1  004  "D"
4  BUFMSG                  43D2  015  "U"               "DUP"
5  BUFMSG                  43D3  210  "P"
6  BUFMSG                  43D4  02B  JNC +05           complete msg
7  BUFMSG  NOBUFR          43D5  20D  ?NC XQ            Build Msg - UF25 Clear
8  BUFMSG                  43D6  0FC  ->3F83            [APERMSG]
9  BUFMSG                  43D7  00E  "N"
10 BUFMSG                  43D8  20F  "O"               "NO"
11 BUFMSG                  43D9  04B  JNC +09           complete msg
12 BUFMSG  ENDOBF          43DA  20D  ?NC XQ            Build Msg - UF25 Clear
13 BUFMSG                  43DB  0FC  ->3F83            [APERMSG]
14 BUFMSG                  43DC  005  "E"
15 BUFMSG                  43DD  00E  "N"
16 BUFMSG                  43DE  004  "D"               "END OF"
17 BUFMSG                  43DF  020  " "
18 BUFMSG                  43E0  00F  "O"
19 BUFMSG                  43E1  206  "F"
20 BUFMSG                  43E2  3BD  ?NC XQ            Message Line
21 BUFMSG                  43E3  01C  ->07EF            [MESSL]
22 BUFMSG                  43E4  020  " "
23 BUFMSG                  43E5  002  "B"               " BUF"
24 BUFMSG                  43E6  015  "U"
```

25	BUFMSG		43E7	*206*	***"F"***	
26	BUFMSG		43E8	*1F1*	*?NC GO*	*LeftJ, Show and Halt*
27	BUFMSG		43E9	*0FE*	*->3F7C*	*[APEREX]*
28	BUFMSG	**NOBFFS**	**43EA**	*321*	*?NC XQ*	*Show "NO_" msg*
29	BUFMSG		43EB	*10C*	*->43C8*	*[NOMSG4]*
30	BUFMSG		43EC	*002*	***"B"***	
31	BUFMSG		43ED	*015*	***"U"***	
32	BUFMSG		43EE	*006*	***"F"***	
33	BUFMSG		43EF	*006*	***"F"***	*"NO BUFFERS"*
34	BUFMSG		43F0	*005*	***"E"***	
35	BUFMSG		43F1	*012*	***"R"***	
36	BUFMSG		43F2	*213*	***"S"***	
40	BUFMSG		43F3	***3AB***	***JNC -11d***	
1	NOXMEM	**CHKEM**	**43F4**	130	LDI S&X	
2	NOXMEM		43F5	040	CON: 64	
3	NOXMEM		43F6	270	RAM SLCT	
4	NOXMEM		43F7	038	READATA	
5	NOXMEM		43F8	10E	A=C ALL	
6	NOXMEM		43F9	2BA	C=-C-1 M	
7	NOXMEM		43FA	2F0	WRITDATA	
8	NOXMEM		43FB	038	READATA	
9	NOXMEM		43FC	2BA	C=-C-1 M	
10	NOXMEM		43FD	2F0	WRITDATA	
11	NOXMEM		43FE	36E	?A#C ALL	
12	NOXMEM		43FF	***3A0***	***?NC RTN***	
13	NOXMEM	**NOXMEM**	**4400**	*321*	*?NC XQ*	*Show "NO_" msg*
14	NOXMEM		4401	*10C*	*->43C8*	*[NOMSG4]*
15	NOXMEM		4402	*018*	***"X"***	
16	NOXMEM		4403	*02D*	***"-"***	
17	NOXMEM		4404	*00D*	***"M"***	*"NO X-MEM"*
18	NOXMEM		4405	*005*	***"E"***	
19	NOXMEM		4406	*20D*	***"M"***	
20	NOXMEM		4407	***363***	***JNC -20d***	
1	NOTOK	**NOTOK**	**4408**	*20D*	*?NC XQ*	*Build Msg - UF25 Clear*
2	NOTOK		4409	*0FC*	*->3F83*	*[APERMSG]*
3	NOTOK		440A	*00E*	***"N"***	
4	NOTOK		440B	*00F*	***"O"***	
5	NOTOK		440C	*014*	***"T"***	*"NOT OK"*
6	NOTOK		440D	*020*	***" "***	
7	NOTOK		440E	*00F*	***"O"***	
8	NOTOK		440F	*20B*	***"K"***	
9	NOTOK		4410	***3BB***	***JNC -09***	
1	CHKKEYS	**CHKKEYS**	**4411**	**3F8**	**READ 15(e)**	
2	CHKKEYS		4412	10E	A=C ALL	
3	CHKKEYS		4413	2B8	READ 10(+)	
4	CHKKEYS		4414	370	C=C OR A	
5	CHKKEYS		4415	10E	A=C ALL	
6	CHKKEYS		4416	04E	C=0 ALL	
7	CHKKEYS		4417	29C	PT= 7	
8	CHKKEYS		4418	210	LD@PT- 8	
9	CHKKEYS		4419	2AE	C=-C-1 ALL	
10	CHKKEYS		441A	05C	PT= 4	
11	CHKKEYS		441B	04A	C=0 PT<-	
12	CHKKEYS		441C	3B0	C=C AND A	
13	CHKKEYS		441D	2EE	?C#0 ALL	
14	CHKKEYS		441E	360	?C RTN	
15	CHKKEYS	**NOKEYS**	**441F**	*321*	*?NC XQ*	*Show "NO_" msg*
16	CHKKEYS		4420	*10C*	*->43C8*	*[NOMSG4]*
17	CHKKEYS		4421	*00B*	***"K"***	
18	CHKKEYS		4422	*005*	***"E"***	*"NO KEYS"*
19	CHKKEYS		4423	*019*	***"Y"***	
20	CHKKEYS		4424	*213*	***"S"***	
21	CHKKEYS		4425	***35B***	***JNC -21d***	
1	NOHPIL	**NOHPIL**	**4426**	**15C**	**PT= 6**	
2	NOHPIL		4427	1D0	LD@PT- 7	
3	NOHPIL		4428	04A	C=0 PT<-	

4			4429	130	LDI S&X	
5	*this method is compatible*		442A	01C	CON:	
6	*with other ROMS in pg#7 :-)*		442B	106	A=C S&X	
7			442C	330	FETCH S&X	
8	NOHPIL		442D	366	?A#C S&X	
9	NOHPIL		442E	***3A0***	***?NC RTN***	
10	NOHPIL	NOIL10	442F	*321*	*?NC XQ*	*Show "NO_" msg*
11	NOHPIL		4430	*10C*	*->43C8*	*[NOMSG4]*
12	NOHPIL		4431	*008*	***"H"***	
13	NOHPIL		4432	*010*	***"P"***	*"NO HPIL"*
14	NOHPIL		4433	*009*	***"I"***	
15	NOHPIL		4434	*20C*	***"L"***	
16	NOHPIL		4435	*383*	*JNC -16d*	*LeftJ, Show and Halt*
1	NOCX	**NOCX?**	**4436**	**04E**	**C=0 ALL**	
2	NOCX		4437	15C	PT= 6	
3	NOCX		4438	0D0	LD@PT- 3	*addr: 3000*
4			4439	**330**	**FETCH S&X**	*read word*
5	*this must happen after*		443A	2E6	?C#0 S&X	*carry set if CX*
6	*checking for Library4*		443B	***360***	***?C RTN***	*return quietely*
7			443C	2E0	DSPOFF	*reset the baseline*
8	NOCX		443D	320	DSPTOG	*display ON*
9			443E	*3C1*	*?NC XQ*	*Enable and Clear Display*
10	*by definition there's no CX,*		443F	*0B0*	*->2CF0*	*[CLLCDE]*
11	*so no calls to port#3, please :(*		4440	*3BD*	*?NC XQ*	*Message Line*
12			4441	*01C*	*->07EF*	*[MESSL]*
13	NOCX		4442	*00E*	***"N"***	
14	NOCX		4443	*00F*	***"O"***	
15	NOCX		4444	*020*	***" "***	
16	NOCX		4445	*003*	***"C"***	*"NO CX/OS"*
17	NOCX		4446	*018*	***"X"***	
18	NOCX		4447	*02F*	***"/"***	
19	NOCX		4448	*00F*	***"O"***	
20	NOCX		4449	*213*	***"S"***	
21	NOCX		444A	*035*	*?NC XQ*	*Display not halting*
22	NOCX		444B	*100*	*->400D*	*[APERX4]*
23	NOCX		444C	*3ED*	*?NC GO*	*HALT execution*
24	NOCX		444D	*08A*	*->22FB*	*[ERR110]*
1	TOGF4	**TOGF4A**	**444E**	**0AE**	**A<>C ALL**	
2	TOGF4	**TOGF4**	**444F**	130	LDI S&X	
3	TOGF4		4450	037	CON: 55	
4	TOGF4		4451	0AE	A<>C ALL	
5	TOGF4		4452	1C6	A=A-C S&X	
6	TOGF4		4453	*381*	*?C GO*	
7	TOGF4		4454	*00B*	*->02E0*	*[ERRNE]*
8	TOGF4		4455	04E	C=0 ALL	
9	TOGF4		4456	226	C=C+1 S&X	
10	TOGF4		4457	1A6	A=A-1 S&X	
11	TOGF4		4458	***01F***	***JC +03***	
12	TOGF4		4459	1EE	C=C+C ALL	
13	TOGF4		445A	*3EB*	*JNC -03*	
14	TOGF4		445B	0EE	C<>B ALL	
15	TOGF4		445C	3B8	READ 14(d)	
16	TOGF4		445D	10E	A=C ALL	
17	TOGF4		445E	0CE	C=B ALL	
18	TOGF4		445F	3B0	C=C AND A	
19	TOGF4		4460	2EE	?C#0 ALL	
20	TOGF4		4461	*135*	*?C GO*	
21	TOGF4		4462	*05B*	*->164D*	*[XCF]*
22	TOGF4		4463	*129*	*?NC GO*	
23	TOGF4		4464	*05A*	*->164A*	*[XSF]*
1	RAND4	**RAND4**	**4465**	**2A0**	**SETDEC**	
2	RAND4		4466	05E	C=0 MS	*absolute value*
3	RAND4		4467	10E	A=C ALL	
4	RAND4		4468	04E	C=0 ALL	
5	RAND4		4469	35C	PT= 12	
6	RAND4		446A	250	LD@PT- 9	

7	RAND4		446B	210	LD@PT- 8	
8	RAND4		446C	090	LD@PT- 2	***9.821***
9	RAND4		446D	050	LD@PT- 1	
10	RAND4		446E	130	LDI S&X	
11	RAND4		446F	003	CON: 003	
12	RAND4		4470	*135*	*?NC XQ*	
13	RAND4		4471	*060*	*->184D*	*[MP2-10]*
14	RAND4		4472	04E	C=0 ALL	
15	RAND4		4473	266	C=C-1 S&X	
16	RAND4		4474	35C	PT= 12	
17	RAND4		4475	090	LD@PT- 2	
18	RAND4		4476	050	LD@PT- 1	***0,211327***
19	RAND4		4477	050	LD@PT- 1	
20	RAND4		4478	0D0	LD@PT- 3	
21	RAND4		4479	090	LD@PT- 2	
22	RAND4		447A	1D0	LD@PT- 7	
23	RAND4		447B	*025*	*?NC XQ*	
24	RAND4		447C	*060*	*->1809*	*[AD1-10]*
25	RAND4		447D	084	CLRF 5	*Take Fractional part*
26	RAND4		447E	*0ED*	*?NC GO*	
27	RAND4		447F	*066*	*->193B*	*[INTFRC]*
1	CDE	**CDE4**	**4480**	**248**	**SETF 9**	*expects Chip0 enabled*
2	CDE		4481	2DC	PT= 13	
3	CDE		4482	250	LD@PT- 9	
4	CDE		4483	0BE	A<>C MS	
5	CDE		4484	178	READ 5(M)	
6	CDE		4485	33C	RCR 1	
7	CDE		4486	006	A=0 S&X	
8	CDE		4487	29C	PT= 7	
9	CDE		4488	358	ST=C XP	
10	CDE		4489	20C	?FSET 2	
11	CDE		448A	*013*	*JNC +02*	
12	CDE		448B	21E	C=C+A MS	
13	CDE		448C	3AE	RSHFB ALL	
14	CDE		448D	0FE	B<>C MS	*leaves result in B[ALL]*
15	CDE		448E	23C	RCR 2	
16	CDE		448F	3D4	PT=PT-1	
17	CDE		4490	394	?PT= 0	
18	CDE		4491	***3BB***	***JNC -09***	
19	CDE		4492	24C	?FSET 9	
20	CDE		4493	***3A0***	***?NC RTN***	
21	CDE		4494	244	CLRF 9	
22	CDE		4495	1B8	READ 6(N)	
23	CDE		4496	***37B***	***JNC -17***	
1	DCD	**DCD4**	**4497**	**2A0**	**SETDEC**	*expects Chip0 enabled*
2	DCD		4498	248	SETF 9	
3	DCD		4499	39C	PT= 0	
4	DCD		449A	04E	C=0 ALL	
5	DCD		449B	228	WRIT 8(P)	
6	DCD		449C	1E8	WRIT 7(O)	
7	DCD		449D	130	LDI S&X	
8	DCD		449E	006	CON:	
9	DCD		449F	1BC	RCR 11	
10	DCD		44A0	130	LDI S&X	
11	DCD		44A1	030	CON:	
12	DCD		44A2	0AE	A<>C ALL	
13	DCD		44A3	04E	C=0 ALL	
14	DCD		44A4	062	A<>B @PT	*expects data in B[ALL]*
15	DCD		44A5	3AE	RSHFB ALL	
16	DCD		44A6	206	C=C+A S&X	
17	DCD		44A7	362	?A#C @PT	
18	DCD		44A8	*013*	*JNC +02*	
19	DCD		44A9	222	C=C+1 @PT	
20	DCD		44AA	23C	RCR 2	
21	DCD		44AB	1BA	A=A-1 M	
22	DCD		44AC	***3C3***	***JNC -08***	
23	DCD		44AD	24C	?FSET 9	

24	DCD		44AE	023	JNC +04	
25	DCD		44AF	244	CLRF 9	
26	DCD		44B0	168	WRIT 5(M)	
27	DCD		44B1	34B	JNC -23	
28	DCD		44B2	1A8	WRIT 6(N)	
29	DCD		44B3	260	SETHEX	
30	DCD		44B4	3E0	RTN	
1	XAVEW?	XAVEW?	44B5	2CC	?FSET 13	running a PRGM?
2	XAVEW?		44B6	360	?C RTN	yes, return (do nothing)
3	XAVEW?		44B7	191	?NC GO	no, show ALPHA
4	XAVEW?		44B8	00E	->0364	[XAVIEW]
1	ABSP	ABSP4	44B9	31C	PT= 1	
2	ABSP		44BA	238	READ 8(P)	
3	ABSP		44BB	0EA	B<>C PT<-	
4	ABSP		44BC	17C	RCR 6	
5			44BD	04A	C=0 PT<-	
6	uses B for scrath, why?		44BE	0FC	RCR 10	
7	I guess it could also use A		44BF	228	WRIT 8(P)	
8	unless this is called from		44C0	1F8	READ 7(O)	
9	a location using it already		44C1	0EA	B<>C PT<-	
10			44C2	23C	RCR 2	
11	ABSP		44C3	1E8	WRIT 7(O)	
12	ABSP		44C4	1B8	READ 6(N_	
13	ABSP		44C5	0EA	B<>C PT<-	
14	ABSP		44C6	23C	RCR 2	
15	ABSP		44C7	1A8	WRIT 6(N)	
16	ABSP		44C8	178	READ 5(M)	
17	ABSP		44C9	0EA	B<>C PT<-	
18	ABSP		44CA	23C	RCR 2	
19	ABSP		44CB	168	WRIT 5(M)	
20	ABSP		44CC	3E0	RTN	
1	ALIST	BCKARW	44CD	055	?NC GO	Delete char plus logic
2	ALIST		44CE	116	->4515	[DELCHR]
3	ALIST	ALIST	44CF	115	?NC XQ	Partial Data Entry!
4	ALIST		44D0	038	->0E45	[NEXT1]
5			44D1	3E3	JNC -04	[BCKARW]
6	Used by the matrix and		44D2	00C	?FSET 3	numeric input?
7	polynomial input functions,		44D3	093	JNC +18d	NO, KEEP LOOKING
8	in SandMatrix and Algebra		44D4	0BE	A<>C MS	recall LS digit from A[13]
9	modules.		44D5	130	LDI S&X	
10			44D6	003	CON:	pre-load the numeric mask
11	ALIST		44D7	2FC	RCR 13	move it to C[S&X]
12	ALIST		44D8	3E8	WRIT 15(e)	write it in display (9-bit)
13	ALIST	fixed logic	44D9	344	CLRF 12	enable SPACE
14	ALIST	TOALPH	44DA	39C	PT= 0	
15	ALIST		44DB	058	G=C @PT,+	
16	ALIST		44DC	149	?NC XQ	Disable PER, enable RAM
17	ALIST		44DD	024	->0952	[ENCP00]
18	ALIST		44DE	051	?NC XQ	
19	ALIST		44DF	0B4	->2D14	[APNDNW]
20	ALIST	GOBACK	44E0	042	C=0 @PT	
21	ALIST		44E1	058	G=C @PT,+	reset PTEMP bits
22	ALIST		44E2	3D9	?NC XQ	Enable Display (not cleared)
23	ALIST		44E3	01C	->07F6	[ENLCD]
24	ALIST	ANCHOR1	44E4	35B	JNC -21d	ONE PROMPT
25	ALIST	RDXTST	44E5	28C	?FSET 7	decimal key pressed?
26	ALIST		44E6	03B	JNC +07	NO, KEEP LOOKING
27	ALIST		44E7	18C	?FSET 11	been used already?
28	ALIST		44E8	3E7	JC -04	ONE PROMPT
29	ALIST		44E9	188	SETF 11	no more radix (unless deletion)
30	ALIST		44EA	10D	?NC XQ	adds proper radix sign
31	ALIST		44EB	114	->4543	[RADIX4]
32	ALIST	ANCHOR2	44EC	373	JNC -18d	[TOALPH]
33	ALIST		44ED	0B0	C=N ALL	PRESSED KEY CODE
34	ALIST		44EE	106	A=C S&X	
35	ALIST		44EF	130	LDI S&X	
36	ALIST		44F0	030	CON:	ENTER^ keycode [030]

37	ALIST			44F1	*366*	*?A#C S&X*		
38	ALIST			44F2	***04F***	***JC +09***		
39	ALIST			44F3	34C	?FSET 12		*blank space used?*
40	ALIST		*fixed logic*	44F4	***387***	***JC -16d***		***yes, ONE PROMPT***
41	ALIST			44F5	0C4	CLRF 10		allows CHS
42	ALIST			44F6	184	CLRF 11		allows RADIX
43	ALIST		*fixed logic*	44F7	348	SETF 12		no SPC allowed
44	ALIST			44F8	*355*	*?NC XQ*		*add space to LCD*
45	ALIST			44F9	*03C*	*->0FD5*		*[DSPL20]*
46	ALIST			44FA	***393***	***JNC -14d***		add to Alpha
47	ALIST			44FB	130	LDI S&X		
48	ALIST			44FC	370	CON:		***R/S keycode [370]***
49	ALIST			44FD	*366*	*?A#C S&X*		*terminate digit entry*
50	ALIST			44FE	***07B***	***JNC +15d***		*[WAYOUT]*
51	ALIST			44FF	130	LDI S&X		
52	ALIST			4500	230	CON:		***CHS keycode [230]***
53	ALIST			4501	*366*	*?A#C S&X*		
54	ALIST			4502	*023*	*JNC +04*		
55	ALIST			4503	*265*	*?NC XQ*		*Blink Display*
56	ALIST			4504	*020*	*->0899*		*[BLINK1]*
57	ALIST			4505	***2FB***	***JNC -33d***		***ONE PROMPT***
58	ALIST			4506	0CC	?FSET 10		*been used already?*
59	ALIST			4507	***3F7***	***JC -02***		***ONE PROMPT***
60	ALIST			4508	0C8	SETF 10		*first time*
61	ALIST			4509	130	LDI S&X		
62	ALIST			450A	02D	*"-"*		*appends "-"*
63	ALIST			450B	***3E8***	***WRIT 15(e)***		*9-bit LCD write*
64	ALIST			450C	***303***	***JNC -32d***		*[TOALPH]*
65	ALIST	**WAYOUT**		**450D**	*000*			
66	ALIST			450E	*000*			
67	ALIST			450F	*161*	*?NC XQ*		*Clear LCD and reset things*
68	ALIST			4510	*124*	*->4958*		*[EXIT3]*
69	ALIST			4511	*175*	*?NC XQ*		*Adjust F10/11/12 Status*
70	ALIST			4512	*114*	*->455D*		*[ADJF10]*
71	ALIST			4513	*31D*	*?NC GO*		*Normal Function ReturnKB*
72	ALIST			4514	*002*	*->00C7*		*[NFRKB]*
73	ALIST	**DELCHR**		**4515**	**3B8**	**READ 14(d)**		*to delete rightmost chr*
74	ALIST			4516	158	M=C ALL		*save it for later*
75	ALIST			4517	*149*	*?NC XQ*		*Disable PER, enable RAM*
76	ALIST			4518	*024*	*->0952*		*[ENCP00]*
77	ALIST			4519	178	READ 5(M)		
78	ALIST			451A	2EE	?C#0 ALL		*anything in Alpha?*
79	ALIST			451B	***037***	***JC +06***		*yes, go on*
80	ALIST			451C	104	CLRF 8		*no, abort if empty*
81	ALIST			451D	*1B1*	*?NC XQ*		*Mainframe Message*
82	ALIST			451E	*070*	*->1C6C*		*[MSGA]*
83	ALIST			451F	03C	*"NULL"*		*from table*
84	ALIST		*fixed bug*	**4520**	***37B***	***JNC -17d***		*Reset everything and leave*
85	ALIST			4521	*2E5*	*?NC XQ*		*remove last Alpha char*
86	ALIST			4522	*110*	*->44B9*		*[ABSP4]*
87	ALIST			4523	198	C=M ALL		*recall deleted char value*
88	ALIST			4524	106	A=C S&X		*store in A for comparisons*
89	ALIST			4525	130	LDI S&X		*check for SPACE*
90	ALIST			4526	020	***"space"***		*<space>*
91	ALIST			4527	*0AD*	*?NC XQ*		*complete the logic*
92	ALIST			4528	*114*	*->452B*		*[CHUNK4]*
93	ALIST			4529	*381*	***?NC GO***		*repeat the prompt*
94	ALIST			452A	*112*	*->44E0*		*[GOBACK]*
95	ALIST	**CHUNK4**		**452B**	*366*	*?A#C S&X*		*carry if different*
96	ALIST			452C	*01F*	*JC + 03*		
97	ALIST		*fixed bug*	452D	**344**	**CLRF 12**		*allow new space entry*
98	ALIST			452E	***0A3***	***JNC +20d***		BAIL OUT
99	ALIST	NOSPC		452F	130	LDI S&X		*check for "-" chr*
100	ALIST			4530	02D	*"-"*		*"-" char value*
101				4531	*366*	*?A#C S&X*		*carry if not "-"*
102	*Executed within [DELCHR]*			4532	*02F*	*JC + 05*		
103	*an opportunistic routine*			4533	***34C***	***?FSET 12***		*is there SPACE chr?*

104	*just grouping common code*		4534	***017***	***JC +02***	yes, skip
105			4535	0C4	CLRF 10	*allow new "-" entry*
106	ALIST		4536	***063***	***JNC +12d***	BAIL OUT
107	ALIST	NOCHS	4537	198	C=M ALL	*recall deleted char value*
108	ALIST		4538	3D8	C<>ST XP	*Got a radix? If so, we neet to*
109	ALIST		4539	14C	?FSET 6	*replace it without comma*
110	ALIST		453A	***043***	***JNC +08***	no radix, skip
111	ALIST		453B	*3D9*	*?NC XQ*	*Enable Display (not cleared)*
112	ALIST		453C	*01C*	*->07F6*	*[ENLCD]*
113	ALIST		453D	144	CLRF 6	*remove the radix value*
114	ALIST		453E	284	CLRF 7	*(both if need be)*
115	ALIST		453F	3D8	C<>ST XP	*recall modified char value*
116	ALIST		4540	***3E8***	***WRIT 15(e)***	write back in display
117	ALIST		4541	***184***	***CLRF 11***	*Re-allow RADIX writing*
118	ALIST	NORDX	4542	**3E0**	**RTN**	
119	ALIST	**RADIX4**	**4543**	*149*	*?NC XQ*	*Disable PER, enable RAM*
120	ALIST		4544	*024*	*->0952*	*[ENCP00]*
121	ALIST		4545	3B8	READ 14(d)	*put F28 to F9*
122			4546	2BC	RCR 7	
123	*transfer staus of UF28 to F9,*		4547	**248**	**SETF 9**	
124	*adds the converted crh code*		4548	1EE	C=C+C ALL	*comma or period ?*
125	*to the LCD and prepares ALPHA*		4549	*013*	*JNC +02*	*overflows if COMMA (cf28)*
126			454A	**244**	**CLRF 9**	*comma = CF 28*
127	ALIST		454B	*3D9*	*?NC XQ*	*Enable Display (not cleared)*
128	ALIST		454C	*01C*	*->07F6*	*[ENLCD]*
129	ALIST		454D	**3B8**	**READ 14(d)**	*read right*
130	ALIST		454E	3D8	C<>ST XP	
131	ALIST		454F	148	SETF 6	
132	ALIST		4550	**24C**	**?FSET 9**	*comma or period ?*
133	ALIST		4551	*013*	*JNC +02*	
134	ALIST		4552	288	SETF 7	*should replace the last chr*
135	ALIST		4553	3D8	C<>ST XP	*with the same one w/ radix*
136	ALIST		4554	***3E8***	***WRIT 15(e)***	*9-bit LCD write*
137	ALIST		4555	130	LDI S&X	
138	ALIST		4556	02C	*","*	*appends "," [02C]*
139	ALIST		4557	**24C**	**?FSET 9**	
140	ALIST		4558	***360***	***?C RTN***	*no need, return*
141	ALIST		4559	226	C=C+1 S&X	
142	ALIST		455A	226	C=C+1 S&X	*appends "." [02E]*
143	ALIST		455B	***3E0***	***RTN***	
1	*<GAP>*		455C	*000*	*NOP*	
1	ADJF10	**ADJFGS**	**455D**	**344**	**CLRF 12**	*program not PRIVATE*
2	ADJF10	**ADJF10**	**455E**	**188**	**SETF 11**	*stack-lift enabled*
3	ADJF10		455F	*149*	*?NC XQ*	*Disable PER, enable RAM*
4	ADJF10		4560	*024*	*->0952*	*[ENCP00]*
5			4561	**338**	**READ 12(b)**	*read program pointer*
6	*adjusts F10 status RAM/ROM*		4562	11A	A=C M	
7	*depending on where's called from.*		4563	01C	PT=3	*fourth digit*
8	*Used by SandMath and PowerCL.*		4564	190	LD@PT- 6	*lowest possible page#*
9	*Will *fail* if plugged in pg#6*		4565	3DC	PT=PT+1	*reset PT = 3*
10			4566	**0C4**	**CLRF 10**	*pointer in RAM (default)*
11	ADJF10		4567	302	?A<C @PT	*is PC<6?*
12	ADJF10		4568	*017*	*JC +02*	*yes, RAM*
13	ADJF10		4569	**0C8**	**SETF 10**	*no, pointer in ROM*
14	ADJF10		456A	3E0	RTN	
1	*<GAP>*		456B	*000*	*NOP*	
2	*<GAP>*		456C	*000*	*NOP*	
1	READNM	**READNM**	**456D**	**31C**	**PT= 1**	*used by FDATA*
2	READNM		456E	02E	B=0 ALL	
3			456F	130	LDI S&X	
4	*Read Alpha, expected to have*		4570	006	CON: 6	*maximum number of bytes*
5	*the function name written.*		4571	0A6	A<>C S&X	
6			4572	178	READ 5(M)	
7	READNM	NXTBYT	4573	2E6	?C#0 S&X	*byte zero?*
8	READNM		4574	***04B***	***JNC +09***	*yes, exit loop*
9	READNM		4575	0EA	C<>B PT<-	
10	READNM		4576	3CE	RSHFC ALL	

11	READNM		4577	3CE	RSHFC ALL	
12	READNM		4578	0EE	C<>B ALL	
13	READNM		4579	37C	RCR 12	
14	READNM		457A	0EE	C<>B ALL	
15	READNM		457B	1A6	A=A-1 S&X	*reach the limit?*
16	READNM		457C	*3BB*	*JNC -09*	*no, next byte*
17	READNM	EXTLOP	457D	046	C=0 S&X	
18	READNM		457E	270	RAM SLCT	
19	READNM		457F	0EE	C<>B ALL	
20	READNM		4580	23C	RCR 2	
21	READNM		4581	268	WRIT 9(Q)	*write "EMANF" in Q :)*
22	READNM		4582	3E0	RTN	
1	*<GAP>*		4583	*000*	*NOP*	
1	SAVRM4	**SAVRM4**	**4584**	*099*	*?NC XQ*	
2	SAVRM4		4585	*110*	*->4426*	*[NOHPIL]*
3	SAVRM4		4586	*035*	*?NC XQ*	
4	SAVRM4		4587	*118*	*->460D*	*[SVGT4]*
5	SAVRM4		4588	05E	C=0 MS	
6	SAVRM4		4589	*02D*	*?NC XQ*	
7	SAVRM4		458A	*1E0*	*->780B*	*[FLSCH]*
8	SAVRM4		458B	198	C=M ALL	
9	SAVRM4		458C	2EE	?C#0 ALL	
10	SAVRM4		458D	*249*	*?C GO*	
11	SAVRM4		458E	*1DB*	*->7692*	*[DUPFL]*
12	SAVRM4		458F	04E	C=0 ALL	
13	SAVRM4		4590	130	LDI S&X	
14	SAVRM4		4591	078	CON: 120	
15	SAVRM4		4592	23C	RCR 2	
16	SAVRM4		4593	130	LDI S&X	
17	SAVRM4		4594	280	CON: 640	
18	SAVRM4		4595	07C	RCR 4	
19	SAVRM4		4596	130	LDI S&X	
20	SAVRM4		4597	014	CON: 20	
21	SAVRM4		4598	07C	RCR 4	
22	SAVRM4		4599	*2F1*	*?NC XQ*	
23	SAVRM4		459A	*1D8*	*->76BC*	*[CRF14]*
24	SAVRM4		459B	*1C9*	*?NC XQ*	
25	SAVRM4		459C	*1FC*	*->7F72*	*[SEEKN]*
26	SAVRM4		459D	130	LDI S&X	
27	SAVRM4		459E	0A2	CON: 162	
28	SAVRM4		459F	*2E9*	*?NC XQ*	
29	SAVRM4		45A0	*1C0*	*->70BA*	*[SCMD]*
30	SAVRM4		45A1	*369*	*?NC XQ*	
31	SAVRM4		45A2	*1C0*	*->70DA*	*[DDL0]*
32	SAVRM4		45A3	*39D*	*?NC XQ*	
33	SAVRM4		45A4	*1DC*	*->77E7*	*[PLERK]*
34	SAVRM4		45A5	124	SELP 4	
35	SAVRM4		45A6	005	PLDI: 01	
36	SAVRM4		45A7	*035*	*?NC XQ*	
37	SAVRM4		45A8	*118*	*->460D*	*[SVGT4]*
38	SAVRM4		45A9	266	C=C-1 S&X	
39	SAVRM4		45AA	***12F***	***JC +25***	
40	SAVRM4		45AB	05C	PT= 4	
41	SAVRM4		45AC	006	A=0 S&X	
42	SAVRM4		45AD	0E6	C<>B S&X	
43	SAVRM4		45AE	**330**	**FETCH S&X**	
44	SAVRM4		45AF	23A	C=C+1 M	
45	SAVRM4		45B0	0B6	A<>C XS	
46	SAVRM4		45B1	0A6	A<>C S&X	
47	SAVRM4		45B2	3E6	LSHFA S&X	
48	SAVRM4		45B3	3EE	LSHFA ALL	
49	SAVRM4		45B4	3EE	LSHFA ALL	
50	SAVRM4		45B5	1E6	C=C+C S&X	
51	SAVRM4		45B6	1E6	C=C+C S&X	
52	SAVRM4		45B7	3C6	RSHFC S&X	
53	SAVRM4		45B8	0A6	A<>C S&X	
54	SAVRM4		45B9	3D4	PT=PT-1	

55	SAVRM4		45BA	394	?PT= 0	
56	SAVRM4		45BB	39B	JNC -13d	
57	SAVRM4		A5BC	0E6	C<>B S&X	
58	SAVRM4		45BD	158	M=C ALL	
59	SAVRM4		45BE	3E6	LSHFA S&X	
60	SAVRM4		45BF	3EE	LSHFA ALL	
61	SAVRM4		45C0	0AE	A<>C ALL	
62	SAVRM4		45C1	09C	PT= 5	
63	SAVRM4		45C2	268	WRIT 9(Q)	
64	SAVRM4		45C3	278	READ 9(Q)	
65	SAVRM4		45C4	23C	RCR 2	
66	SAVRM4		45C5	268	WRIT 9(Q)	
67	SAVRM4		45C6	099	?NC XQ	
68	SAVRM4		45C7	1C4	->7126	[SDATA0]
69	SAVRM4		45C8	39D	?NC XQ	
70	SAVRM4		45C9	1DC	->77E7	[PLERK]
71	SAVRM4		45CA	3D4	PT=PT-1	
72	SAVRM4		45CB	394	?PT= 0	
73	SAVRM4		45CC	3BB	JNC -09	
74	SAVRM4		45CD	198	C=M ALL	
75	SAVRM4		45CE	2DB	JNC -37d	
76	SAVRM4		45CF	130	LDI S&X	
77	SAVRM4		45D0	0A8	CON: 168	
78	SAVRM4		45D1	2E9	?NC XQ	
79	SAVRM4		45D2	1C0	->70BA	[SCMD]
80	SAVRM4		45D3	2BD	?NC GO	
81	SAVRM4		45D4	1C2	->70AF	[UNL]
1	GETRM4	GETRM4	45D5	099	?NC XQ	
2	GETRM4		45D6	110	->4426	[NOHPIL]
3	GETRM4		45D7	035	?NC XQ	
4	GETRM4		45D8	118	->460D	[SVGT4]
5	GETRM4		45D9	130	LDI S&X	
6	GETRM4		45DA	007	CON: 7	
7	GETRM4		45DB	029	?NC XQ	
8	GETRM4		45DC	1E0	->780A	[FLSCH0]
9	GETRM4		45DD	1DD	?NC XQ	
10	GETRM4		45DE	1FC	->7F77	[SEEKRN]
11	GETRM4		45DF	399	?NC XQ	
12	GETRM4		45E0	1C0	->70E6	[SNDATA]
13	GETRM4		45E1	035	?NC XQ	
14	GETRM4		45E2	118	->460D	[SVGT4]
15	GETRM4		45E3	266	C=C-1 S&X	
16	GETRM4		45E4	2B1	?C GO	receives EOT
17	GETRM4		45E5	1C3	->70AC	[UNT]
18	GETRM4		45E6	158	M=C ALL	
19	GETRM4		45E7	09C	PT= 5	
20	GETRM4		45E8	041	?NC XQ	
21	GETRM4		45E9	1C4	->7110	[RDDFRM]
22	GETRM4		45EA	24C	?FSET 9	
23	GETRM4		45EB	0D1	?C GO	
24	GETRM4		45EC	1DB	->7634	[CSERK]
25	GETRM4		45ED	280	HPIL=C 2	
26	GETRM4		45EE	0A6	A<>C S&X	
27	GETRM4		45EF	278	READ 9(Q)	
28	GETRM4		45F0	0A6	A<>C S&X	
29	GETRM4		45F1	23C	RCR 2	
30	GETRM4		45F2	268	WRIT 9(Q)	
31	GETRM4		45F3	3D4	PT=PT-1	
32	GETRM4		45F4	394	?PT= 0	
33	GETRM4		45F5	39B	JNC -13d	
34	GETRM4		45F6	10E	A=C ALL	
35	GETRM4		45F7	23C	RCR 2	
36	GETRM4		45F8	0EE	C<>B ALL	
37	GETRM4		45F9	198	C=M ALL	
38	GETRM4		45FA	05C	PT= 4	
39	GETRM4		45FB	0EE	C<>B ALL	
40	GETRM4		45FC	0AE	A<>C ALL	

41	GETRM4		45FD	37C	RCR 12	
42	GETRM4		45FE	0AE	A<>C ALL	
43	GETRM4		45FF	0B6	A<>C XS	
44	GETRM4		4600	0A6	A<>C S&X	
45	GETRM4		4601	0FA	C<>B M	
46	GETRM4		4602	040	WRIT S&X	
47	GETRM4		4603	23A	C=C+1 M	
48	GETRM4		4604	0FA	C<>B M	
49	GETRM4		4605	1EE	C=C+C ALL	
50	GETRM4		4606	1EE	C=C+C ALL	
51	GETRM4		4607	33C	RCR 1	
52	GETRM4		4608	3D4	PT=PT-1	
53	GETRM4		4609	394	?PT= 0	
54	GETRM4		460A	*393*	*JNC -14d*	
55	GETRM4		460B	0EE	C<>B ALL	
56	GETRM4		460C	*2BB*	*JNC -41d*	
57	GETRM4	**SVGT4**	**460D**	**0F8**	**READ 3(X)**	
58	GETRM4		460E	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
59	GETRM4		460F	*008*	*->02E3*	*[BCDBIN]*
60	GETRM4		4610	106	A=C S&X	
61	GETRM4	**SVGTA**	**4611**	130	LDI S&X	
62	GETRM4		4612	010	CON: 16	*upper limit: pg# < 16*
63	GETRM4		4613	306	?A<C S&X	
64	GETRM4		4614	*289*	*?NC GO*	*"Out of Range"*
65	GETRM4		4615	*002*	*->00A2*	*[ERROF]*
66	GETRM4	noidea	4616	0A6	A<>C S&X	x x x x x x x x x x x 0 0 p
67	GETRM4		4617	13C	RCR 8	x x x x x 0 0 p x x x x x x
68	GETRM4		4618	21C	PT= 2	
69	GETRM4		4619	110	LD@PT- 4	x x x x x 0 0 p x x x 4 x x
70	GETRM4		461A	**3E0**	**RTN**	
1	NAMEA4	**NAMEA4**	**461B**	**158**	**M=C ALL**	*expects fcn address*
2	NAMEA4		461C	*239*	*?NC XQ*	*Display Function Name*
3	NAMEA4		461D	*108*	*->428E*	*[PMTFNM]*
4	NAMEA4	NAMEAX	461E	*149*	*?NC XQ*	*Enables Chip0*
5	NAMEA4		461F	*024*	*->0952*	*[ENCP00]*
6			4620	3B8	READ 14(d)	
7	*prompts for sub-fnc name !, hehe*		4621	3D8	C<>ST XP	*sets UF48*
8	*places it in REG 9 (Q)*		4622	288	SETF 7	*will set ALPHA mode*
9			4623	3D8	C<>ST XP	*at the [NEXT1] call*
10	NAMEA4		4624	3A8	WRIT 14(d)	
11	NAMEA4		4625	04E	C=0 ALL	*initialize alpha operand*
12	NAMEA4		4626	268	WRIT 9(Q)	
13	NAMEA4	NAME10	4627	*3D9*	*?NC XQ*	*Enables LCD*
14	NAMEA4		4628	*01C*	*->07F6*	*[ENLCD]*
15	NAMEA4	NAME20	4629	*115*	*?NC XQ*	
16	NAMEA4		462A	*038*	*->0E45*	*[NEXT1]*
17	NAMEA4		462B	*033*	*JNC +06*	*backarrow, NAME30*
18	NAMEA4		462C	*14C*	*?FSET 6*	*SHIFT key?*
19	NAMEA4		462D	***1DB***	***JNC +59d***	*[NAME40]*
20	NAMEA4		462E	*395*	*?NC XQ*	*yes, toggle SHIFT key*
21	NAMEA4		462F	*07C*	*->1FE5*	*[TOGSHF]*
22	NAMEA4		4630	*3BB*	*JNC -09*	*[NAME10]*
23	NAMEA4	NAME30	4631	*149*	*?NC XQ*	*Enables Chip0*
24	NAMEA4		4632	*024*	*->0952*	*[ENCP00]*
25	NAMEA4		4633	278	READ 9(Q)	*read alpha argument*
26	NAMEA4		4634	2EE	?C#0 ALL	*any chars to delete?*
27	NAMEA4		4635	*3BD*	*?NC GO*	*no, ALPHA off and bail out*
28	NAMEA4	*bug fixed*	4636	*03A*	*->0EEF*	*[NAME33]*
29	NAMEA4		4637	37C	RCR 12	*yes,. Delete one char*
30	NAMEA4		4638	04A	C=0 PT<-	
31	NAMEA4		4639	268	WRIT 9(Q)	
32	NAMEA4		463A	*125*	*?NC XQ*	*turn SHIFT off*
33	NAMEA4		463B	*01C*	*->0749*	*[OFSHFT]*
34	NAMEA4		463C	*3D9*	*?NC XQ*	*Enables LCD*
35	NAMEA4		463D	*01C*	*->07F6*	*[ENLCD]*
36	NAMEA4		463E	3B8	READ 14(d)	*shift off one character*
37	NAMEA4	NAME31	463F	*353*	*JNC -22d*	*[NAME20]*

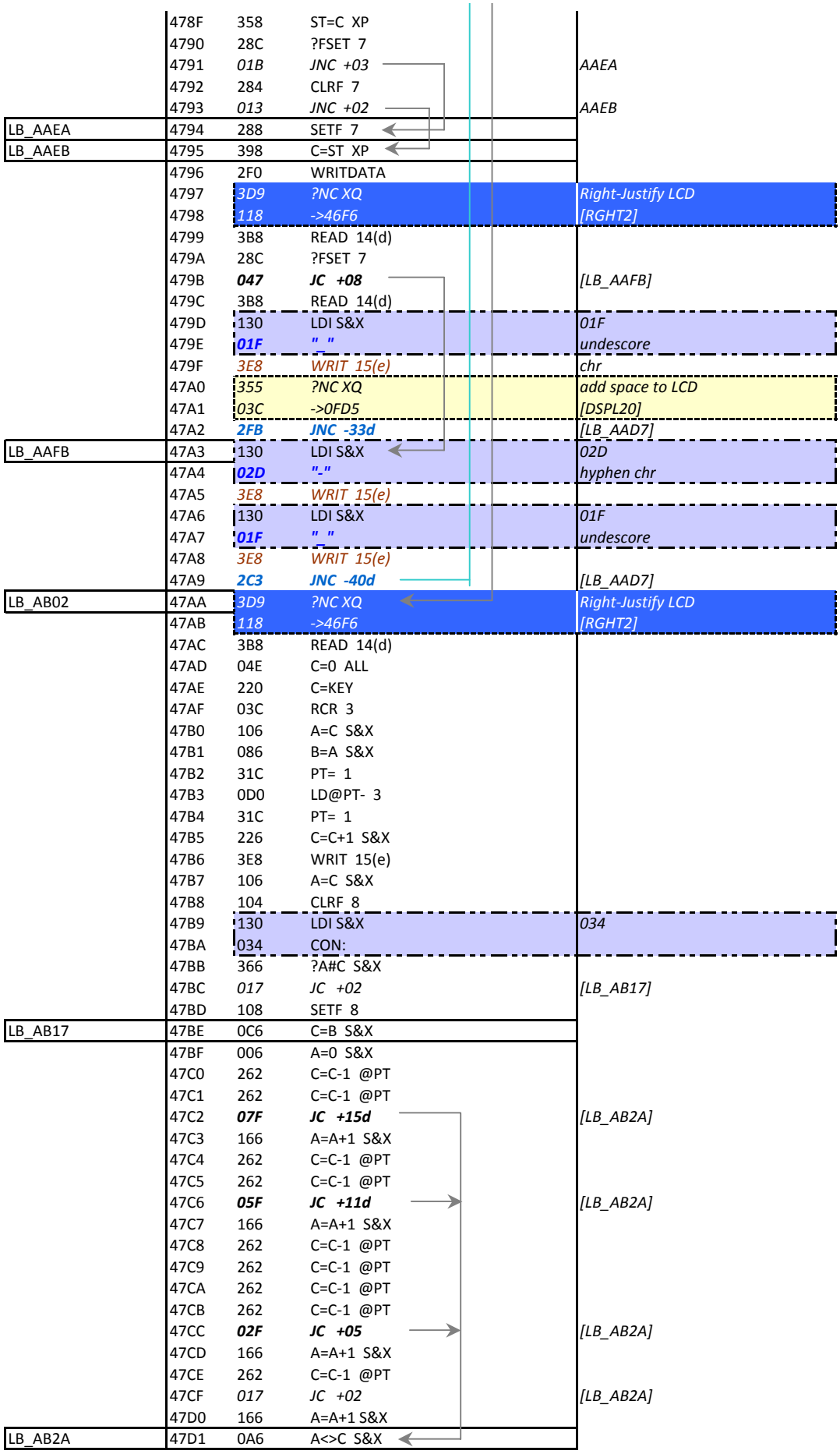
```
38  NAMEA4                  4640  265  ?NC XQ          yes, blink and prompt back
39  NAMEA4                  4641  020  ->0899          [BLINK]
40  NAMEA4                  4642  32B  JNC -27d       [NAME10]
41  NAMEA4  NAME37          4643  36D  ?NC XQ          get alpha code to C[S&X]
42  NAMEA4                  4644  07C  ->1FDB         [GTACOD]
43  NAMEA4                  4645  106  A=C S&X         copy character to A[S&X]
44  NAMEA4                  4646  125  ?NC XQ
45  NAMEA4                  4647  01C  ->0749          [OFSHFT]
46  NAMEA4                  4648  1B6  A=A-1 XS       is it a character?
47  NAMEA4                  4649  3BB  JNC -09        no, NAME35
48  NAMEA4                  464A  31C  PT= 1
49  NAMEA4                  464B  130  LDI S&X
50  NAMEA4                  464C  07F  "|-"           lazy "T"
51  NAMEA4                  464D  36A  ?A#C PT<-
52  NAMEA4                  464E  393  JNC -14d       [NAME35]
53  NAMEA4                  464F  130  LDI S&X
54  NAMEA4                  4650  03A  ":"           colon
55  NAMEA4                  4651  36A  ?A#C PT<-
56  NAMEA4                  4652  373  JNC -18d       [NAME35]
57  NAMEA4                  4653  130  LDI S&X
58  NAMEA4                  4654  02E  ". "          decimal point
59  NAMEA4                  4655  36A  ?A#C PT<-
60  NAMEA4                  4656  353  JNC -22d       [NAME35]
61  NAMEA4                  4657  130  LDI S&X
62  NAMEA4                  4658  02C  ","           comma
63  NAMEA4                  4659  36A  ?A#C PT<-
64  NAMEA4                  465A  333  JNC -26d       [NAME35]
65  NAMEA4                  465B  278  READ 9(Q)
66  NAMEA4                  465C  2EA  ?C#0 PT<-      full already?
67  NAMEA4                  465D  31F  JC -29d        full, [NAME35]
68  NAMEA4                  465E  0AA  A<>C PT<-      add character to REG 9
69  NAMEA4                  465F  10A  A=C PT<-       restore character to A[S&X]
70  NAMEA4                  4660  23C  RCR 2
71  NAMEA4                  4661  268  WRIT 9(Q)      add character to display
72  NAMEA4                  4662  0EE  B<>C ALL       save operand in B
73  NAMEA4                  4663  3D9  ?NC XQ          Enables LCD
74  NAMEA4                  4664  01C  ->07F6         [ENLCD]
75  NAMEA4                  4665  221  ?NC XQ          transliterate char and send to dislp.
76  NAMEA4                  4666  0B0  ->2C88         [MASK]
77  NAMEA4                  4667  2C3  JNC -40d       [NAME31] -> [NAME20]
78  NAMEA4  NAME40          4668  149  ?NC XQ          Enables Chip0
79  NAMEA4                  4669  024  ->0952          [ENCP00]
80  NAMEA4                  466A  08C  ?FSET 5        ALPHA key>
81                          466B  2C3  JNC -40d       no, [NAME37]
82  this is where it gets to be  466C  31C  PT= 1      yes
83  very different from the OS   466D  278  READ 9(Q)
84  which unfortunately ends with  466E  2EE  ?C#0 ALL  any chars in operand?
85  a call to [NFRPU] even for the  466F  28B  JNC -47  no, [NAME35]
86  LBL case.               4670  325  ?NC XQ          Right Justify Operand
87                          4671  050  ->14C9          [RTJLBL]
88  NAMEA4                  4672  268  WRIT 9(Q)       put back right-justified
89  NAMEA4                  4673  349  ?NC XQ          AOFF w/ LCD selected!
90  NAMEA4                  4674  130  ->4CD2          [AOFF4]
91  NAMEA4  CLNUP4          4675  3DD  ?NC XQ          Left Justified format
92  NAMEA4                  4676  0AC  ->2BF7         [LEFTJ]
93  NAMEA4                  4677  319  ?NC XQ          last chance to cancel out
94  NAMEA4                  4678  038  ->0EC6          [NULTST]
95  NAMEA4                  4679  169  ?NC GO          Enable Ram & Reset Seq
96  NAMEA4                  467A  126  ->495A          [EXIT4]
1   <GAP>                   467B  000  NOP
1   ASN4    ASN4            467C  149  ?NC XQ
2   ASN4                    467D  024  ->0952          [ENCP00]
3   ASN4                    467E  01C  PT= 3           Store fcn in P<13:10>
4   ASN4                    467F  238  READ 8(P)
5   ASN4                    4680  0FC  RCR 10          position fnc code at C<3:0>
                            4681  0CA  C=B PT<--       write FNC code in P<3:0>
; This routine will assign any 2 byte  4682  07C  RCR 4  rotate back
```

; function to a given key.		4683	228	WRIT 8(P)	*restore it*
; IN: A<1:0>=logical keycode		4684	2B8	READ 10(+)	
; B<3:0>=2-byte fcn to be assigned		4685	0AA	A<>C PT<--	put the key code in C
; Error exit is: Packing, Try Again		4686	2A8	WRIT 10(+)	*Store keycode*
; Uses: A,B,C,N,M,S0-S9(most of them)		4687	10A	A=C PT<--	restore it
; P<13:10>,LAZY T<3:0> (but NOT the Q register!!		4688	*1FD*	*?NC XQ*	*Test if key already assigned,*
		4689	*0BC*	*->2F7F*	*[TBITMA]*
ASN4		468A	2EE	?C#0 ALL	*Key taken?*
ASN4		468B	**03B**	***JNC +07***	*No -> ASN02*
ASN4		468C	2B8	READ 10(+)	*Yes, Clear keycode*
ASN4		468D	10E	A=C ALL	
ASN4		468E	308	SETF 1	
ASN4		468F	*201*	*?NC XQ*	*Clear the key*
ASN4		4690	*0AC*	*->2B80*	*[GCPKC]*
ASN4		4691	*01B*	*JNC +03*	
ASN4	ASN02	4692	*295*	*?NC XQ*	*Set bit*
ASN4		4693	*0BC*	*->2FA5*	*[SRBMAP]*
ASN4		4694	238	READ 8(P)	*Get the fcn again*
ASN4		4695	0FC	RCR 10	*puts it in B<3:0>*
ASN4		4696	0EE	C<>B ALL	
ASN4		4697	2B8	READ 10(+)	*And the keycode*
ASN4		4698	10E	A=C ALL	*back into A<1:0>*
ASN4		4699	3EE	LSHFA ALL	
ASN4		469A	3EE	LSHFA ALL	*shifted into A<3:2>*
ASN4		469B	*2B5*	*?NC GO*	*Assign it*
ASN4		469C	*09E*	*->27AD*	*[XASN05]*
CCDC4	**CCDC4**	**469D**	**278**	**READ 9(Q)**	*LBL Name*
CCDC4		469E	158	M=C ALL	*required for [ASRCH]*
CCDC4		469F	2EE	?C#0 ALL	*is it null?*
CCDC4		46A0	**05B**	***JNC +11d***	*yes, jump over -> EBC6*
CCDC4		46A1	*315*	*?NC XQ*	*no, look it up*
CCDC4		46A2	*098*	*->26C5*	*[ASRCH]*
CCDC4		46A3	2EE	?C#0 ALL	*found?*
CCDC4		46A4	*381*	*?NC GO*	*no, show "NONEXISTENT" msg*
CCDC4		46A5	*00A*	*->02E0*	*[ERRNE]*
CCDC4		46A6	20C	?FSET 2	*yes, is it ROM?*
CCDC4		46A7	**04B**	***JNC +09***	*no, skip -> EBCB*
CCDC4		46A8	*3D5*	*?NC XQ*	*yes, show error message*
CCDC4		46A9	*088*	*->22F5*	*[ERROR]*
CCDC4		46AA	06A	*"ROM"*	*MSG: ROM*
CCDC4		46AB	0CC	?FSET 10	*is in ROM?*
CCDC4		46AC	*3E7*	*JC -04*	*yes, -> EBC3*
CCDC4		46AD	*141*	*?NC XQ*	*no, go on*
CCDC4		46AE	*0A4*	*->2950*	*[GETPC]*
CCDC4		46AF	0AA	A<>C PT<-	*???*
CCDC4		46B0	*0A1*	*?NC XQ*	
CCDC4		46B1	*0A4*	*->2928*	*[FLINK]*
CCDC4		46B2	13C	RCR 8	
CCDC4		46B3	158	M=C ALL	
CCDC4		46B4	10E	A=C ALL	
CCDC4		46B5	*1FD*	*?NC GO*	
CCDC4		46B6	*01A*	*->067F*	*[CPGM10]*
ASG1	**ASG#1**	**46B7**	130	LDI S&X	
ASG1		46B8	046	CON:	
ASG1		46B9	302	?A<C @PT	
ASG1		46BA	*017*	*JC +02*	*[LB_AA30]*
ASG1		46BB	266	C=C-1 S&X	
ASG1	LB_AA30	46BC	39C	PT= 0	
ASG1		46BD	050	LD@PT- 1	
ASG1		46BE	31C	PT= 1	
ASG1		46BF	362	?A#C @PT	
ASG1		46C0	***027***	***JC +04***	*[LB_AA38]*
ASG1		46C1	36A	?A#C PT<-	
ASG1		46C2	*013*	*JNC +02*	*[LB_AA38]*
ASG1		46C3	16A	A=A+1 PT<	
ASG1	LB_AA38	46C4	1A6	A=A-1 S&X	
ASG1		46C5	0A6	A<>C S&X	

16	ASG1		46C6	106	A=C S&X	
17	ASG1		46C7	3C6	RSHFC S&X	
18	ASG1		46C8	3E6	LSHFA S&X	
19	ASG1		46C9	0A2	A<>C @PT	
20	ASG1		46CA	106	A=C S&X	
21	ASG1		46CB	35E	?A#0 MS	
22	ASG1		46CC	3A0	?NC RTN	
23	ASG1		46CD	130	LDI S&X	
24	ASG1		46CE	008	CON:	
25	ASG1		46CF	146	A=A+C S&X	
26	ASG1		46D0	***3E0***	***RTN***	
27	ASG2	**ASG#2**	**46D1**	**1B0**	**POPADR**	
28	ASG2		46D2	170	PUSHADR	
29	ASG2		46D3	03C	RCR 3	*page# in C(3)*
30	ASG2		46D4	0E6	B<>C S&X	*table start to C[S&X]*
31	ASG2		46D5	1BC	RCR 11	*put in ADR field*
32	ASG2		46D6	026	B=0 S&X	*clear B[S&X]*
33	ASG2		46D7	31C	PT= 1	
34	ASG2	LB_AA4C	46D8	00E	A=0 ALL	
35	ASG2	LB_AA4D	46D9	330	FETCH S&X	*read table value*
36	ASG2		46DA	3EE	LSHFA ALL	
37	ASG2		46DB	3EE	LSHFA ALL	
38	ASG2		46DC	0AA	A<>C PT<-	
39	ASG2		46DD	23A	C=C+1 M	
40	ASG2		46DE	2F6	?C#0 XS	
41	ASG2		46DF	*3D3*	*JNC -06*	*[LB_AA4D]*
42	ASG2		46E0	000	NOP	
43	ASG2		46E1	1D8	C<>M ALL	
44	ASG2		46E2	36E	?A#C ALL	
45	ASG2		46E3	***063***	***JNC +12***	*[LB_AA63]*
46	ASG2		46E4	1D8	C<>M ALL	
47	ASG2		46E5	06E	A<>B ALL	
48	ASG2		46E6	166	A=A+1 S&X	
49	ASG2		46E7	06E	A<>B ALL	
50	ASG2		46E8	24C	?FSET 9	
51	ASG2		46E9	*013*	*JNC +02*	*[LB_AA5F]*
52	ASG2		46EA	23A	C=C+1 M	
53	ASG2	LB_AA5F	46EB	330	FETCH S&X	
54	ASG2		46EC	2E6	?C#0 S&X	*end of table?*
55	ASG2		46ED	3A0	?NC RTN	*yes, end here.*
56	ASG2		46EE	*353*	*JNC -22d*	*[LB_AA4C]*
57	ASG2	LB_AA63	46EF	1D8	C<>M ALL	
58	ASG2		46F0	0AE	A<>C ALL	
59	ASG2		46F1	1B0	POPADR	
60	ASG2		46F2	23A	C=C+1 M	*skip one line*
61	ASG2		46F3	***1E0***	***GOTO ADR***	
1	RIGHTJ	**RIGHTJ**	**46F4**	*3D9*	*?NC XQ*	*Enable but not Clear LCD*
2	RIGHTJ		46F5	*01C*	*->07F6*	*[ENLCD]*
3	RIGHTJ	**RGTHJ2**	**46F6**	130	LDI S&X	
4	RIGHTJ		46F7	***020***	***" "***	*space chr code*
5	RIGHTJ		46F8	106	A=C S&X	
6	RIGHTJ		46F9	31C	PT= 1	
7	RIGHTJ	LB_AA6C	46FA	3B8	READ 14(d)	*rotate righttmost chr to left*
8	RIGHTJ		46FB	36A	?A#C PT<-	*is it space?*
9	RIGHTJ		46FC	*3F3*	*JNC -02*	*yes, get next chr*
10	RIGHTJ		46FD	***3F8***	***READ 15(e)***	*no, put it back where it was*
11	RIGHTJ		46FE	3E0	RTN	
1	*<GAP>*		46FF	*000*	*NOP*	
2	*<GAP>*		4700	*000*	*NOP*	
1	SPLASH	*Header*	4701	*0B4*	***"4"***	
2	SPLASH	*Header*	4702	*023*	***"#"***	
3	SPLASH	*Header*	4703	*019*	***"Y"***	
4	SPLASH	*Header*	4704	*012*	***"R"***	
5	SPLASH	*Header*	4705	*001*	***"A"***	*function name is shown*
6	SPLASH	*Header*	4706	*012*	***"R"***	*in the Page_Catalog*
7	SPLASH	*Header*	4707	*002*	***"B"***	
8	SPLASH	*Header*	4708	*009*	***"I"***	

9	SPLASH	*Header*	4709	*00C*	*"L"*	
10	SPLASH	*Header*	470A	*02D*	*"-"*	*Nelson C. Crowle*
11	SPLASH	**LIBRARY4**	**470B**	***063***	***JNC +12d***	
12	SPLASH	MSGTXT	470C	***1E2***	*"**"*	*1C7*
13	SPLASH		470D	*005*	*"E"*	*"M"*
14	SPLASH		470E	*00E*	*"N"*	*" "*
15	SPLASH		470F	*009*	*"I"*	*"V"*
16			4710	*00C*	*"L"*	*"E"*
17	*Splash screen, scrolling*		4711	*02D*	*"-"*	*"R"*
18	*by Nelson C. Crowle*		4712	*00E*	*"N"*	*"-"*
19			4713	*00F*	*"O"*	*"4"*
20	SPLASH		4714	*022*	*""*	*"#"*
21	SPLASH		4715	*034*	*"4"*	*"B"*
22	SPLASH		4716	*023*	*"#"*	*"L"*
23	SPLASH	**SPLASH**	**4717**	*35D*	*?NC XQ*	
24	SPLASH		4718	*000*	*->00D7*	*[PCTOC]*
25	SPLASH		4719	27A	C=C-1 M	*backup two bytes*
26	SPLASH		471A	27A	C=C-1 M	*to beginning of string*
27	SPLASH		471B	0EE	B<>C ALL	*save it in B[M]*
28	SPLASH	SPLSH4	471C	*37D*	*?NC XQ*	*Cancels TURBO mode*
29	SPLASH		471D	*13C*	*->4FDF*	*[TURBO0]*
30	SPLASH		471E	*3C1*	*?NC XQ*	*Enable and Clear Display*
31	SPLASH		471F	*0B0*	*->2CF0*	*[CLLCDE]*
32	SPLASH		4720	130	LDI S&X	
33	SPLASH		4721	***107***	"**"	***little "T"***
34	SPLASH		4722	31C	PT= 1	
35	SPLASH		4723	3A8	WRIT 14(d)	*Places it (Long) into the LEFT*
36	SPLASH		4724	3B8	READ 14(d)	*Fetch Right Long: moves right to left*
37	SPLASH	LB_A09D	4725	0CE	C=B ALL	
38	SPLASH		4726	27A	C=C-1 M	previous address
39	SPLASH		4727	0EE	B<>C ALL	update repository
40	SPLASH		4728	0CE	C=B ALL	keep it in C
41	SPLASH		4729	330	FETCH S&X	read byte value
42	SPLASH		472A	358	ST=C XP	place it in ST
43	SPLASH		472B	28C	?FSET 7	*is it > "0FF"?*
44	SPLASH		472C	*023*	*JNC +04*	*no, skip to A0A8*
45	SPLASH		472D	284	CLRF 7	*clear it*
46	SPLASH		472E	398	C=ST XP	*restore byte to C[S&X]*
47	SPLASH		472F	288	SETF 7	*flag this fact*
48	SPLASH	[LB_A0A8	4730	3E8	WRIT 15(e)	*write it to RIGHT end*
49	SPLASH		4731	28C	?FSET 7	*was it last one?*
50	SPLASH		4732	***10F***	***JC +33***	*yes, we're done -> A0CB*
51	SPLASH	LB_A0AB	4733	130	LDI S&X	
52	SPLASH		4734	***100***	CON:	
53	SPLASH		4735	266	C=C-1 S&X	*count until 100(hex)*
54	SPLASH		4736	*3FB*	*JNC -01*	*A0AD*
55	SPLASH		4737	130	LDI S&X	
56	SPLASH		4738	***020***	" "	***blank space***
57	SPLASH		4739	106	A=C S&X	store in A[S&X]
58	SPLASH	LB_A0B2	473A	3B8	READ 14(d)	*read right chr and move it to left*
59	SPLASH		473B	366	?A#C S&X	is it a space?
60	SPLASH		473C	*3F3*	*JNC -02*	*yes, read next chr. -> A0B2*
61	SPLASH		473D	158	M=C ALL	*no, save chr1 in M*
62	SPLASH		473E	3B8	READ 14(d)	*read chr2 and move to left*
63	SPLASH		473F	366	?A#C S&X	*is it a space?*
64	SPLASH		4740	***05F***	***JC +11***	*yes, jump over to A0C3*
65	SPLASH		4741	198	C=M ALL	*no, recall chr1*
66	SPLASH		4742	3E8	WRIT 15(e)	*write it to RIGHT end*
67	SPLASH		4743	130	LDI S&X	
68	SPLASH		4744	***020***	" "	***blank space***
69	SPLASH		4745	3E8	WRIT 15(e)	*write it to RIGHT end*
70	SPLASH	LB_A0BE	4746	3F8	READ 15(e)	*Fetch Left Long*
71	SPLASH		4747	2E2	?C#0 @PT	
72	SPLASH		4748	*3F7*	*JC -02*	*A0BE*
73	SPLASH		4749	3A8	WRIT 14(e)	
74	SPLASH		474A	***34B***	***JNC -23 d***	*A0AB*
75	SPLASH	LB_A0C3	474B	3E8	WRIT 15(e)	*write it to RIGHT end*

76	SPLASH		474C	198	C=M ALL		*recall chr1 for real*
77	SPLASH		474D	3E8	WRIT 15(e)		*write it to RIGHT end*
78	SPLASH	LB_A0C6	474E	3F8	READ 15(e)		*Fetch Left Long*
79	SPLASH		474F	366	?A#C S&X		
80	SPLASH		4750	*3F3*	*JNC -02*		*A0C6*
81	SPLASH		4751	3A8	WRIT 14(d)		
82	SPLASH		4752	***29B***	***JNC -45d***		*next character, A09D*
83	SPLASH	LB_A0CB	4753	*149*	*?NC XQ*		*Enables Chip0*
84	SPLASH		4754	*024*	*->0952*		*[ENCP00]*
85	SPLASH		4755	*38D*	*?NC XQ*		*Sets TURBO mode*
86	SPLASH		4756	*13C*	*->4FE3*		*[TURB50]*
87	SPLASH		4757	*1F9*	*?NC GO*		*Set MSG Flag*
88	SPLASH		4758	*00E*	*->-037E*		*[STMSGF]*
1	APMD+	**APND+**	**4759**	130	LDI S&X		
2	APND+		475A	02B	*"+"*		*append "+"*
3	APND+		475B	*3D5*	*?NC GO*		
4	APND+		475C	*07E*	*->1FF5*		*[APND10]*
1	ASG	**ASG4**	**475D**	*125*	*?NC XQ*		
2	ASG		475E	*01C*	*->0749*		*[OFSHFT]*
3	ASG		475F	278	READ 9(Q)		
4	ASG		4760	244	CLRF 9		
5	ASG		4761	2EE	?C#0 ALL		*de-assigning?*
6	ASG		4762	*017*	*JC +02*		*AAB6*
7	ASG		4763	248	SETF 9		
8	ASG	LB_AAB6	4764	*3D1*	*?NC XQ*		*Right Justify Display*
9	ASG		4765	*118*	*->46F4*		*[RIGHTJ]*
14	ASG		4766	*355*	*?NC XQ*		*add space to LCD*
15	ASG		4767	*03C*	*->0FD5*		*[DSPL20]*
16	ASG		4768	24C	?FSET 9		
17	ASG		4769	*013*	*JNC +02*		*[LB_AAC1]*
18	ASG		476A	*3E8*	*WRIT 15(e)*		
19	ASG	LB_AAC1	476B	130	LDI S&X		
20	ASG		476C	***01F***	*" "*		*undescore*
21	ASG		476D	*3E8*	*WRIT 15(e)*		
22	ASG		476E	*3DD*	*?NC XQ*		
23	ASG		476F	*0AC*	*->2BF7*		*[LEFTJ]*
24	ASG	LB_AAC6	4770	3CC	?KEY		
25	ASG		4771	*3FB*	*JNC -01*		*AAC6*
26	ASG		4772	220	C=KEY		
27	ASG		4773	03C	RCR 3		
28	ASG		4774	056	C=0 XS		
29	ASG		4775	106	A=C S&X		
30	ASG		4776	130	LDI S&X		
31	ASG		4777	018	CON:		
32	ASG		4778	366	?A#C S&X		
33	ASG		4779	*321*	*?NC GO*		
34	ASG		477A	*046*	*->11C8*		*[OFF]*
35	ASG		477B	130	LDI S&X		
36	ASG		477C	0C4	CON:		
37	ASG		477D	306	?A<C S&X		
38	ASG		477E	***047***	***JC +08***		*AADC*
39	ASG		477F	*265*	*?NC XQ*		
40	ASG		4780	*020*	*->0899*		*[BLINK]*
41	ASG	LB_AAD7	4781	*3DD*	*?NC XQ*		
42	ASG		4782	*0AC*	*->2BF7*		*[LEFTJ]*
43	ASG		4783	*261*	*?NC XQ*		
44	ASG		4784	*000*	*->0098*		*[RSTKB]*
45	ASG		4785	***35B***	***JNC -21d***		*AAC6*
46	ASG	LB_AADC	4786	130	LDI S&X		
47	ASG		4787	012	CON:		
48	ASG		4788	366	?A#C S&X		
49	ASG		4789	***10F***	***JC +33d***		*AB02*
50	ASG		478A	*395*	*?NC XQ*		
51	ASG		478B	*07C*	*->1FE5*		*[TOGSHF]*
52	ASG		478C	*3D9*	*?NC XQ*		
53	ASG		478D	*01C*	*->07F6*		*[ENLCD]*
54	ASG		478E	178	READ 5(M)		



55	ASG		478F	358	ST=C XP	
56	ASG		4790	28C	?FSET 7	
57	ASG		4791	*01B*	*JNC +03*	*AAEA*
58	ASG		4792	284	CLRF 7	
59	ASG		4793	*013*	*JNC +02*	*AAEB*
60	ASG	LB_AAEA	4794	288	SETF 7	
61	ASG	LB_AAEB	4795	398	C=ST XP	
62	ASG		4796	2F0	WRITDATA	
63	ASG		4797	*3D9*	*?NC XQ*	*Right-Justify LCD*
64	ASG		4798	*118*	*->46F6*	*[RGHT2]*
65	ASG		4799	3B8	READ 14(d)	
66	ASG		479A	28C	?FSET 7	
67	ASG		479B	***047***	***JC +08***	*[LB_AAFB]*
68	ASG		479C	3B8	READ 14(d)	
69	ASG		479D	130	LDI S&X	*01F*
70	ASG		479E	***01F***	*" "*	*undescore*
71	ASG		479F	*3E8*	*WRIT 15(e)*	*chr*
72	ASG		47A0	*355*	*?NC XQ*	*add space to LCD*
73	ASG		47A1	*03C*	*->0FD5*	*[DSPL20]*
74	ASG		47A2	***2FB***	***JNC -33d***	*[LB_AAD7]*
75	ASG	LB_AAFB	47A3	130	LDI S&X	*02D*
76	ASG		47A4	***02D***	*"-"*	*hyphen chr*
77	ASG		47A5	*3E8*	*WRIT 15(e)*	
78	ASG		47A6	130	LDI S&X	*01F*
79	ASG		47A7	***01F***	*" "*	*undescore*
80	ASG		47A8	*3E8*	*WRIT 15(e)*	
81	ASG		47A9	***2C3***	***JNC -40d***	*[LB_AAD7]*
82	ASG	LB_AB02	47AA	*3D9*	*?NC XQ*	*Right-Justify LCD*
83	ASG		47AB	*118*	*->46F6*	*[RGHT2]*
84	ASG		47AC	3B8	READ 14(d)	
85	ASG		47AD	04E	C=0 ALL	
86	ASG		47AE	220	C=KEY	
87	ASG		47AF	03C	RCR 3	
88	ASG		47B0	106	A=C S&X	
89	ASG		47B1	086	B=A S&X	
90	ASG		47B2	31C	PT= 1	
91	ASG		47B3	0D0	LD@PT- 3	
92	ASG		47B4	31C	PT= 1	
93	ASG		47B5	226	C=C+1 S&X	
94	ASG		47B6	3E8	WRIT 15(e)	
95	ASG		47B7	106	A=C S&X	
96	ASG		47B8	104	CLRF 8	
97	ASG		47B9	130	LDI S&X	*034*
98	ASG		47BA	034	CON:	
99	ASG		47BB	366	?A#C S&X	
100	ASG		47BC	*017*	*JC +02*	*[LB_AB17]*
101	ASG		47BD	108	SETF 8	
102	ASG	LB_AB17	47BE	0C6	C=B S&X	
103	ASG		47BF	006	A=0 S&X	
104	ASG		47C0	262	C=C-1 @PT	
105	ASG		47C1	262	C=C-1 @PT	
106	ASG		47C2	***07F***	***JC +15d***	*[LB_AB2A]*
107	ASG		47C3	166	A=A+1 S&X	
108	ASG		47C4	262	C=C-1 @PT	
109	ASG		47C5	262	C=C-1 @PT	
110	ASG		47C6	***05F***	***JC +11d***	*[LB_AB2A]*
111	ASG		47C7	166	A=A+1 S&X	
112	ASG		47C8	262	C=C-1 @PT	
113	ASG		47C9	262	C=C-1 @PT	
114	ASG		47CA	262	C=C-1 @PT	
115	ASG		47CB	262	C=C-1 @PT	
116	ASG		47CC	***02F***	***JC +05***	*[LB_AB2A]*
117	ASG		47CD	166	A=A+1 S&X	
118	ASG		47CE	262	C=C-1 @PT	
119	ASG		47CF	*017*	*JC +02*	*[LB_AB2A]*
120	ASG		47D0	166	A=A+1 S&X	
121	ASG	LB_AB2A	47D1	0A6	A<>C S&X	

```
122 ASG                  47D2  0D0   LD@PT- 3
123 ASG                  47D3  10C   ?FSET 8
124 ASG                  47D4  01B   JNC +03          [LB_AB30]
125 ASG                  47D5  2E2   ?C#0 @PT
126 ASG                  47D6  017   JC +02           [LB_AB31]
127 ASG  LB_AB30         47D7  222   C=C+1 @PT
128 ASG  LB_AB31         47D8  3E8   WRIT 15(e)
129 ASG                  47D9  066   A<>B S&X
130 ASG                  47DA  3E6   LSHFA S&X
131 ASG                  47DB  31C   PT= 1
132 ASG                  47DC  0A2   A<>C @PT
133 ASG                  47DD  222   C=C+1 @PT
134 ASG                  47DE  0E6   B<>C S&X
135 ASG                  47DF  3DD   ?NC XQ
136 ASG                  47E0  0AC   ->2BF7            [LEFTJ]
137 ASG                  47E1  149   ?NC XQ            Disable PER, enable RAM
138 ASG                  47E2  024   ->0952            [ENCP00]
139 ASG                  47E3  319   ?NC XQ
140 ASG                  47E4  038   ->0EC6            [NULTST]
141 ASG                  47E5  066   A<>B S&X
142 ASG                  47E6  01E   A=0 MS
143 ASG                  47E7  28C   ?FSET 7
144 ASG                  47E8  013   JNC +02           [LB_AB43]
145 ASG                  47E9  17E   A=A+1 MS
146 ASG  LB_AB43         47EA  2DD   ?NC XQ
147 ASG                  47EB  118   ->46B7            [ASG#1]
148 ASG                  47EC  2B8   READ 10(+)
149 ASG                  47ED  0A6   A<>C S&X
150 ASG                  47EE  2A8   WRIT 10(+)
151 ASG                  47EF  125   ?NC XQ
152 ASG                  47F0  01C   ->0749            [OFSHFT]
153 ASG                  47F1  0A0   SLCTP
154 ASG                  47F2  130   LDI S&X           020
155 ASG                  47F3  020   " "              space chr
156 ASG                  47F4  0A6   A<>C S&X
157 ASG                  47F5  278   READ 9(Q)
158 ASG                  47F6  158   M=C ALL
159 ASG                  47F7  384   CLRF 0
160 ASG                  47F8  31C   PT= 1
161 ASG  LB_AB53         47F9  254   ?PT= 9
162 ASG                  47FA  03B   JNC +07           [LB_AB5B]
163 ASG                  47FB  31C   PT= 1
164 ASG                  47FC  2B8   READ 10(+)
165 ASG                  47FD  106   A=C S&X
166 ASG                  47FE  278   READ 9(Q)
167 ASG                  47FF  1B1   ?NC GO
168 ASG                  4800  09E   ->276C            [XASN] +2
169 ASG  LB_AB5B         4801  3EE   LSHFA ALL
170 ASG                  4802  3EE   LSHFA ALL
171 ASG                  4803  10A   A=C PT<-
172 ASG                  4804  3DC   PT=PT+1
173 ASG                  4805  3DC   PT=PT+1
174 ASG                  4806  36A   ?A#C PT<-
175 ASG                  4807  397   JC -14d           [LB_AB53]
176 ASG                  4808  0E0   SLCTQ
177 ASG                  4809  2DC   PT= 13
178 ASG                  480A  052   C=0 P-Q
179 ASG                  480B  070   N=C ALL
180 ASG                  480C  130   LDI S&X           049
181 ASG                  480D  049   "I"              IND symbol
182 ASG                  480E  39C   PT= 0
183 ASG                  480F  058   G=C @PT,+
184 ASG                  4810  278   READ 9(Q)
185 ASG                  4811  10E   A=C ALL
186 ASG                  4812  0A0   SLCTP
187 ASG                  4813  3DC   PT=PT+1
188 ASG                  4814  098   C=G @PT,+
```

189	ASG		4815	3DC	PT=PT+1	
190	ASG		4816	36A	?A#C PT<-	
191	ASG		4817	***027***	***JC +04***	*[LB_AB75]*
192	ASG		4818	388	SETF 0	
193	ASG		4819	3DC	PT=PT+1	
194	ASG		481A	3DC	PT=PT+1	
195	ASG	LB_AB75	481B	0D0	LD@PT- 3	
196	ASG		481C	3DC	PT=PT+1	
197	ASG		481D	362	?A#C @PT	
198	ASG		481E	***08F***	***JC +17d***	*[LB_AB89]*
199	ASG		481F	3EA	LSHFA PT<-	
200	ASG		4820	0AE	A<>C ALL	
201	ASG		4821	058	G=C @PT,+	
202	ASG		4822	1A0	A=B=C=0	
203	ASG		4823	39C	PT= 0	
204	ASG		4824	098	C=G @PT,+	
205	ASG		4825	0A2	A<>C @PT	
206	ASG		4826	3C6	RSHFC S&X	
207	ASG		4827	3E6	LSHFA S&X	
208	ASG		4828	0A2	A<>C @PT	
209	ASG		4829	0A6	A<>C S&X	
210	ASG		482A	03C	RCR 3	
211	ASG		482B	226	C=C+1 S&X	
212	ASG		482C	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
213	ASG		482D	*008*	*->02E3*	*[BCDBIN]*
214	ASG		482E	***0AB***	***JNC +21d***	*[LB_AB9E]*
215	ASG	LB_AB89	482F	0AE	A<>C ALL	
216	ASG		4830	3D4	PT=PT-1	
217	ASG		4831	058	G=C @PT,+	
218	ASG		4832	04E	C=0 ALL	
219	ASG		4833	39C	PT= 0	
220	ASG		4834	098	C=G @PT,+	
221	ASG		4835	158	M=C ALL	
222	ASG		4836	21C	PT= 2	
223	ASG		4837	210	LD@PT- 8	*beginning of table*
224	ASG		4838	1D0	LD@PT- 7	
225	ASG		4839	110	LD@PT- 4	
226	ASG		483A	0E6	B<>C S&X	*saved in B*
227	ASG		483B	*345*	*?NC XQ*	*Fetch Argument*
228	ASG		483C	*118*	*->46D1*	*[ASG#2]*
229	ASG	NOTFND	483D	*013*	*JNC +02*	*not found, show error*
230	ASG	FOUND	483E	*01B*	*JNC +03*	*found, go on*
231	ASG	LB_AB9A	483F	*381*	*?NC GO*	
232	ASG		4840	*00A*	*->02E0*	*[ERRNE]*
233	ASG	LB_AB9C	4841	0EE	B<>C ALL	
234	ASG		4842	1D0	LD@PT- 7	
235	ASG	LB_AB9E	4843	38C	?FSET 0	
236	ASG		4844	*023*	*JNC +04*	*[LB_ABA3]*
237	ASG		4845	358	ST=C XP	
238	ASG		4846	288	SETF 7	
239	ASG		4847	398	C=ST XP	
240	ASG	LB_ABA3	4848	268	WRIT 9(Q)	
241	ASG		4849	0B0	C=N ALL	
242	ASG		484A	158	M=C ALL	
243	ASG		484B	21C	PT= 2	
244	ASG		484C	190	LD@PT- 6	
245	ASG		484D	03C	RCR 3	
246	ASG		484E	11E	A=C MS	
247	ASG		484F	198	C=M ALL	
248	ASG	LB_ABAB	4850	0EE	B<>C ALL	
249	ASG		4851	06A	A<>B PT<-	
250	ASG		4852	*221*	*?NC XQ*	
251	ASG		4853	*0B0*	*->2C88*	*[MASK]*
252	ASG		4854	000	NOP	
253	ASG		4855	31C	PT= 1	
254	ASG		4856	2F6	?C#0 XS	
255	ASG		4857	***01B***	***JNC +03***	*[LB_ABB5]*

256	ASG		4858	110	LD@PT- 4	
257	ASG		4859	31C	PT= 1	
258	ASG	LB_ABB5	485A	0EE	B<>C ALL	
259	ASG		485B	0EA	B<>C PT<-	
260	ASG		485C	23C	RCR 2	
261	ASG		485D	2EA	?C#0 PT<-	
262	ASG		485E	***023***	***JNC +04***	*[LB_ABBD]*
263	ASG		485F	1BE	A=A-1 MS	
264	ASG		4860	***02F***	***JC +05***	*[LB_ABC0]*
265	ASG		4861	*37B*	*JNC -17*	*[LB_ABAB]*
266	ASG	LB_ABBD	4862	23C	RCR 2	
267	ASG		4863	2EA	?C#0 PT<-	
268	ASG		4864	*3F3*	*JNC -02*	*[LB_ABBD]*
269	ASG	LB_ABC0	4865	158	M=C ALL	
270	ASG		4866	15C	PT= 6	
271	ASG		4867	*0E9*	*?NC XQ*	*263A*
272	ASG		4868	*098*	*->263A*	*[unlabeled]*
273	ASG		4869	2EE	?C#0 ALL	
274	ASG		486A	***2AB***	***JNC -43***	*[LB_AB9A]*
275	ASG		486B	05C	PT= 4	
276	ASG		486C	058	G=C @PT,+	
277	ASG		486D	278	READ 9(Q)	
278	ASG		486E	21C	PT= 2	
279	ASG		486F	098	C=G @PT,+	
280	ASG		4870	208	SETF 2	
281	ASG		4871	0FC	RCR 10	
282	ASG		4872	*1DD*	*?NC GO*	
283	ASG		4873	*09E*	*->2777*	*[XASNRC]*
284	ASG	**ASGTBL**	**4874**	154	***"T"***	
285	ASG		4875	15A	***"Z"***	
286	ASG		4876	159	***"Y"***	
287	ASG		4877	158	***"X"***	
288	ASG		4878	14C	***"L"***	
289	ASG		4879	14D	***"M"***	
290	ASG		487A	14E	***"N"***	
291	ASG		487B	14F	***"O"***	
292	ASG		487C	150	***"P"***	
293	ASG		487D	151	***"Q"***	
294	ASG		487E	12B	***"|-"***	
295	ASG		487F	161	***"a"***	
296	ASG		4880	162	***"b"***	
297	ASG		4881	163	***"c"***	
298	ASG		4882	164	***"d"***	
299	ASG		4883	165	***"e"***	
300	ASG		4884	*200*	*end of table*	
1	RAMEDIT	**HEXUT0**	**4885**	*3D9*	*?NC XQ*	
2	RAMEDIT		4886	*01C*	*->07F6*	*[ENLCD]*
3	RAMEDIT	**HEXUT4**	4887	**1A6**	**A=A-1 S&X**	*Outputs HEX Digits*
4			4888	360	?C RTN	
5	*output hex digits to LCD*		4889	08E	B=A ALL	*convert to LCD code*
6	*Hex digits left justified in A*		488A	2DC	PT= 13	
7	*A<2:0> -> digits*		488B	046	C=0 S&X	
8	*assumes LCD enable*		488C	250	LD@PT- 9	
9	*uses A,B,C, pt*		488D	0BE	A<>C MS	
10			488E	31E	?A<C MS	
11	RAMEDIT		488F	*027*	*JC +04*	
12	RAMEDIT		4890	130	LDI S&X	
13	RAMEDIT		4891	003	CON: 3	
14	RAMEDIT		4892	*01B*	*JNC +03*	
15	RAMEDIT		4893	0BE	A<>C MS	*subtract 9*
16	RAMEDIT		4894	25E	C=A-C MS	
17	RAMEDIT		4895	2FC	RCR 13	*char in C<2:0>*
18	RAMEDIT		4896	3E8	WRIT 15(e)	*write digit to LCD*
19	RAMEDIT		4897	0CE	C=B ALL	
20	RAMEDIT		4898	2FC	RCR 13	
21	RAMEDIT		4899	10E	A=C ALL	
22	RAMEDIT		489A	066	A<>B S&X	

23	RAMEDIT		489B	363	JNC -20d	
1	LASTFN	LASTF2	489C	070	N=C ALL	safeguard data
2			489D	130	LDI S&X	buffer offset
3	expects fcn id# in C[ADR]		489E	002	CON:	PowerCL
4	and XROM.FNC# in B[4:0]		489F	0E6	C<>B S&X	offset to B[S&X]
5	typically as first line in FNC code		48A0	21C	PT= 2	FNC# to C<1:0>
6			48A1	0D0	LD@PT- 3	complete the code!
7	LASTFN		48A2	158	M=C ALL	save parameter in M
8	LASTFN		48A3	3D1	?NC XQ	Store fcn. Id# in LASTF buffer
9	LASTFN		48A4	120	->48F4	[LASTF0] - inputs in B and M
10	LASTFN		48A5	0F0	C<>N ALL	restore initial data
11	LASTFN		48A6	3E0	RTN	
12	LASTFN	LASTF3	48A7	130	LDI S&X	buffer offset
13	LASTFN		48A8	003	CON:	SandMath
14	LASTFN		48A9	0E6	C<>B S&X	offset to B[S&X]
15	LASTFN		48AA	310	LD@PT- C	complete the code!
16	LASTFN		48AB	3BB	JNC -09	
1	YFNZ4?	YFNZ4?	48AC	130	LDI S&X	depends on YFNS version!
2	YFNZ4?		48AD	3FB	CON:	YFNS? Function code
3			48AE	001	?NC XQ	Get ROM address to A[3:0]
4	assumes YFNX case!		48AF	020	->0800	[GTRMAD]
5			48B0	013	JNC +02	not there! - reset
6	YFNZ4?		48B1	3E0	RTN	return if found
7	YFNZ4?		48B2	020	XQ>GO	defuse the bomb...
8	YFNZ4?	YRESET4	48B3	215	?NC XQ	Build Msg - all cases
9			48B4	0FC	->3F85	[APRMSG2]
10	by showing the message first		48B5	099	"Y:"	
11	we make sure this is always		48B6	012	"R"	
12	available before disabling		48B7	005	"E"	announce the intention
13	the MMU		48B8	013	"S"	while we're still here...
14			48B9	005	"E"	
15	YFNZ4?		48BA	214	"T"	
16	YFNZ4?		48BB	035	?NC XQ	Display msg - not halting
17	YFNZ4?		48BC	100	->400D	[APERX4]
18	YFNZ4?		48BD	04E	C=0 ALL	disable the MMU if not there!
19	YFNZ4?		48BE	05C	PT= 4	to fix disengaged scenarios
20	YFNZ4?		48BF	190	LD@PT- 6	after plugging something In
21	YFNZ4?		48C0	1FC	WCMD	the location of YFNZ
22	YFNZ4?		48C1	3E0	RTN	
1	BAILOUT	ABORT	48C2	020	XQ>GO	defuse the bomb...
2	BAILOUT	BAILOUT	48C3	1B1	?NC XQ	Mainframe Message
3	BAILOUT		48C4	070	->1C6C	[MSGA]
4	BAILOUT		48C5	03C	"NULL"	from table!
5	BAILOUT		48C6	061	?NC GO	abort bit sequence - clears F13
6	BAILOUT		48C7	036	->0D18	[ABTS20]
1	HEXKEY		48C8	265	?NC XQ	Blink Display
2	HEXKEY		48C9	020	->0899	[BLINK1]
3		HEXKEY	48CA	115	?NC XQ	Partial Key Sequence!
4	accepts A-F, all numbers		48CB	038	->0E45	[NEXT1]
5	and SHIFT - but no other keys		48CC	3B3	JNC -10d	back arrow pressed
6			48CD	14C	?FSET 6	SHIFT key pressed
7	HEXKEY		48CE	01B	JNC +03	no, keep looking
8	HEXKEY		48CF	026	B=0 S&X	yes, flag it so
9	HEXKEY		48D0	3E0	RTN	and return!
10	HEXKEY		48D1	00C	?FSET 3	numeric input?
11	HEXKEY		48D2	033	JNC +06	NO, KEEP LOOKING
12	HEXKEY		48D3	0BE	A<>C MS	recall LS digit from A[13]
13	HEXKEY		48D4	130	LDI S&X	
14	HEXKEY		48D5	003	CON:	pre-load numeric mask
15	HEXKEY		48D6	2FC	RCR 13	move it over digit 0
16	HEXKEY		48D7	063	JNC +12d	
17	HEXKEY		48D8	04C	?FSET 4	rows 1 & 2?
18	HEXKEY		48D9	37B	JNC -17d	ONE PROMPT
19	HEXKEY		48DA	35E	?A#0 MS	was "J" pressed?
20	HEXKEY		48DB	3F3	JNC -02	yes, blink & go back
21	HEXKEY		48DC	130	LDI S&X	

22	HEXKEY		48DD	007	"G"	
23	HEXKEY		48DE	33C	RCR 1	put in MS field
24	HEXKEY		48DF	31E	?A<C MS	
25	HEXKEY		48E0	3CB	JNC -07	blink & ONE PROMPT
26	HEXKEY		48E1	0BE	A<>C MS	
27	HEXKEY		48E2	2FC	RCR 13	
28	HEXKEY		48E3	0E6	C<>B S&X	save chr# in B[S&X]
29	HEXKEY		48E4	3E0	RTN	
1	READID	READID	48E5	141	?NC XQ	get were we're at
2	READID		48E6	0A4	->2950	[GETPC]
3			48E7	0AE	A<>C ALL	result to C<3:0>
4	Must be run in bank-1		48E8	070	N=C ALL	safekeep in N
5	in order to read the arguments		48E9	179	?NC XQ	Get Parameter from NextLine
6	of FOCAL programs that use it		48EA	10C	->435E	[GETRG#] - uses A,B,C,M
7			48EB	236	C=C+1 XS	mask as sub-fcn index
8	READID		48EC	158	M=C ALL	this is the index#
9	READID		48ED	0F0	C<>N ALL	initial PC to C<3:0>
10	READID		48EE	10E	A=C ALL	initial counter to A<3:0>
11	READID		48EF	0DD	?NC GO	restore initial counter
12	READID		48F0	08E	->2337	[PUTPC]
1	LASTF#	LASTF#	48F1	0E6	C<>B S&X	offset to B[S&X]
2	LASTF#		48F2	149	?NC XQ	Disable PER, enable RAM
3	LASTF#		48F3	024	->0952	[ENCP00]
4		LASTF0	48F4	130	LDI S&X	
5	write the function id# in M[S&X]		48F5	009	CON: 9	buffer id#
6	into the proper buffer register		48F6	2A9	?NC XQ	Check for Buffer
7	for the LASTF functionallity		48F7	10C	->43AA	[CHKBF4] +1
8			48F8	043	JNC +08	Not Found - be quiet!
9	SandMatrix 4		48F9	126	A=A+B S&X	variable offset: 2,3,4
10	SandMath 3		48FA	0A6	A<>C S&X	put adr in C[S&X]
11	PowerCL 2		48FB	270	RAMSLCT	select register
12	TimeSeed 1		48FC	198	C=M ALL	get fnc id#
13	Header 0		48FD	05E	C=0 MS	mark the MS with "F"
14			48FE	27E	C=C-1 MS	to tell cases apart
15	LASTF#		48FF	2F0	WRITDATA	re-write it
16	LASTF#		4900	073	JNC +14d	[SELRAM]
17	LASTF#	LASTF$	4901	0E6	C<>B S&X	offset to B[S&X]
18	LASTF#		4902	278	READ 9(Q)	retrieves "EMANF"
19	LASTF#		4903	0F0	C<>N ALL	can't use chip0 later
20	LASTF#		4904	130	LDI S&X	
21			4905	009	CON: 9	buffer id#
22	support code for LASTF:		4906	2A9	?NC XQ	Check for Buffer
23	need to store the FNAME		4907	10C	->43AA	[CHKBF4] +1
24	in buffer #9		4908	033	JNC +06	Not Found - be quiet!
25			4909	126	A=A+B S&X	variable offset: 2,3,4
26	LASTF#		490A	0A6	A<>C S&X	put adr in C[S&X]
27	LASTF#		490B	270	RAMSLCT	select register
28	LASTF#		490C	0F0	C<>N ALL	get "EMANF" to C
29	LASTF#		490D	2F0	WRITDATA	store in buffer
30	LASTF#	SELRAM	490E	046	C=0 S&X	
31	LASTF#		490F	270	RAMSLCT	select Chip0
32	LASTF#		4910	3E0	RTN	done.
1	MIRROR	MIROR4	4911	0E0	SLCT Q	Q is the counter
2	MIRROR		4912	39C	PT= 0	reset it
3	MIRROR	NXTNYB	4913	3DC	PT=PT+1	increase counter
4			4914	3EE	LSHFA ALL	shift destination
5	changes contents in C		4915	3EE	LSHFA ALL	shift it again
6	into their mirror-image,		4916	0A0	SLCT P	
7	grouped by two nybbles		4917	31C	PT= 1	P delimits the two-digit pack
8			4918	10A	A=C PT<-	copy them to A
9	MIRROR		4919	23C	RCR 2	rotate source
10	MIRROR		491A	0E0	SLCT Q	
11	MIRROR		491B	294	?PT= 7	done 7 times?
12	MIRROR		491C	3BB	JNC -09	no, repeat
13	MIRROR		491D	0AE	A<>C ALL	retrieve value
14	MIRROR		491E	325	?NC GO	Right Justify Operand
15	MIRROR		491F	052	->14C9	[RTJLBL]

1	OSREV	**OSREV**	**4920**	*215*	*?NC XQ*	*Build Msg - all cases*
2	OSREV		4921	*0FC*	*->3F85*	*[APRMSG2]*
3	OSREV		4922	*012*	***"R"***	
4	OSREV		4923	*00F*	***"O"***	
5	OSREV		4924	*00D*	***"M"***	*"ROM 0:"*
6	OSREV		4925	*020*	***" "***	
7	OSREV		4926	*2B0*	***"0:"***	
8	OSREV		4927	15C	PT= 6	
9	OSREV		4928	010	LD@R 0	*position ADR to ROM-0*
10	OSREV		4929	3D0	LD@R F	
11	OSREV		492A	3D0	LD@R F	
12	OSREV		492B	390	LD@R E	
13	OSREV		492C	10E	A=C ALL	
14	OSREV		492D	330	FETCH S&X	*read ROM id# value*
15	OSREV		492E	3E8	WRIT 15(e)	
16	OSREV		492F	15C	PT= 6	
17	OSREV	ONENOP	4930	162	A=A+1 @PT	*position ADR to ROM-1*
18	OSREV		4931	*3BD*	*?NCXQ*	
19	OSREV		4932	*01C*	*->07EF*	*[MESSL]*
20	OSREV		4933	*020*	***" "***	
21	OSREV		4934	*2B1*	***"1:"***	
22	OSREV		4935	0AE	A<>C ALL	
23	OSREV		4936	10E	A=C ALL	
24	OSREV		4937	162	A=A+1 @PT	*position ADR to ROM-2*
25	OSREV		4938	330	FETCH S&X	*read ROM id# value*
26	OSREV		4939	3E8	WRIT 15(e)	
27	OSREV		493A	*3BD*	*?NCXQ*	
28	OSREV		493B	*01C*	*->07EF*	*[MESSL]*
29	OSREV		493C	*020*	***" "***	
30	OSREV		493D	*2B2*	***"2:"***	
31	OSREV		493E	0AE	A<>C ALL	
32	OSREV		493F	330	FETCH S&X	*read ROM id# value*
33	OSREV		4940	3E8	WRIT 15(e)	
34	OSREV		4941	*1F9*	*?NC GO*	*Set MSG Flag*
35	OSREV		4942	*00E*	*->-037E*	*[STMSGF]*
1	NODVC4	**NODVC4**	**4943**	*321*	*?NC XQ*	*Show "NO_" msg*
2	NODVC4		4944	*10C*	*->43C8*	*[NOMSG4]*
3	NODVC4		4945	*004*	***"D"***	
4	NODVC4		4946	*005*	***"E"***	
5	NODVC4		4947	*016*	***"V"***	*"NO DEVICE"*
6	NODVC4		4948	*009*	***"I"***	
7	NODVC4		4949	*003*	***"C"***	
8	NODVC4		494A	*205*	***"E"***	
9	NODVC4		494B	*1F1*	*?NC GO*	*Show and Halt*
10	NODVC4		494C	*0FE*	*->3F7C*	*[APEREX]*
1	VRG4	**VRG4**	**494D**	**0A6**	**A<>C S&X**	***View (decoded) Register***
2	VRG4		494E	270	RAMSLCT	
3	VRG4		494F	038	READATA	
4			4950	0EE	B<>C ALL	*save rg value in B*
5	*expects absolute addr in A[S&X]*		4951	046	C=0 S&X	
6			4952	270	RAMSLCT	*Select Chip0*
7	VRG4		4953	*25D*	*?NC XQ*	
8	VRG4		4954	*110*	*->4497*	*[DCD4]*
9	VRG4		4955	*2D5*	***?NC GO***	
10	VRG4		4956	*112*	*->44B5*	*[XAVEW?]*
1	*<GAP>*		4957	*000*	*NOP*	
1	EXITS	**EXIT3**	**4958**	*3D9*	*?NC XQ*	*clear the LCD*
2	EXITS		4959	*0B0*	*->2CF6*	*[CLRLCD]*
3	EXITS	**EXIT4**	**495A**	*149*	*?NC XQ*	*Disable PER, enable RAM*
4	EXITS		495B	*024*	*->0952*	*[ENCP00]*
5	EXITS		495C	*215*	*?NC GO*	*Reset BIT sequence*
6	EXITS		495D	*00E*	*->0385*	*[RSTSQ] - leaves F13 alone (!)*
1	APPEND	**APNDJ**	**495E**	130	LDI S&X	
2	APPEND		495F	04A	***"J"***	*append "J"*
3	APPEND		4960	*01B*	*JNC +03*	
4	APPEND	**APND1**	**4961**	130	LDI S&X	

5	APPEND		4962	031	"1"	append "1"
6	APPEND		4963	01B	JNC +03	
7	APPEND	APND|	4964	130	LDI S&X	
8	APPEND		4965	021	"|"	append "!"
9	APPEND		4966	01B	JNC +03	
10	APPEND	APNDZ	4967	130	LDI S&X	
11	APPEND		4968	05A	"Z"	append "Z"
12	APPEND		4969	01B	JNC +03	
13	APPEND	APNDW	496A	130	LDI S&X	
14	APPEND		496B	057	"W"	append "W"
15	APPEND		496C	3D5	?NC GO	
16	APPEND		496D	07E	->1FF5	[APND10]
1	SURE?	SURE?	496E	215	?NC XQ	Build Msg - all cases
2	SURE?		496F	0FC	->3F85	[APRMSG2]
3	SURE?		4970	00F	"O"	
4	SURE?		4971	00B	"K"	"OK?"
5	SURE?		4972	23F	"?"	
6	SURE?	YESNO	4973	3BD	?NC XQ	
7	SURE?		4974	01C	->07EF	[MESSL]
8	SURE?		4975	020	" "	
9	SURE?		4976	019	"Y"	" Y/N"
10			4977	02F	"/"	
11	Yes/No confirmation check		4978	20E	"N"	
12	returs 1/0 in A[MS]		4979	3DD	?NC XQ	
13			497A	0AC	->2BF7	[LEFTJ]
14	SURE?		497B	01E	A=0 MS	NO result (default)
15	SURE?	LOOP	497C	229	?NC XQ	do while not key
16	SURE?		497D	124	->498A	[RSTKB4] - puts KY in A/C[S&X]
17	SURE?		497E	130	LDI	
18	SURE?		497F	013	"N"	keycode
19	SURE?		4980	366	?A#C S&X	
20	SURE?		4981	033	JNC +06	LB_AE1A
21	SURE?		4982	130	LDI	
22	SURE?		4983	016	"Y"	keycode
23	SURE?		4984	366	?A#C S&X	
24	SURE?		4985	3BF	JC -09	LB_AE03
25	SURE?	YES	4986	17E	A=A+1 MS	YES result
26	SURE?	NO	4987	149	?NC GO	
27	SURE?		4988	026	->0952	[ENCP00]
28			4989	000		
29	CLV4 1x	RSTKB4	498A	3C8	CLRKEY	reset key - but not on CLV4!!!!
30	CLV4 1x		498B	3CC	?KEY	do while not key
31			498C	3F3	JNC -02	[DONOT]
32	SURE?		498D	220	C=KEY KY	
33	SURE?		498E	261	?NC XQ	Reset keyboard
34	SURE?		498F	000	->0098	[RSTKB]
35	SURE?		4990	03C	RCR 3	move to S&X field
36	SURE?		4991	056	C=0 XS	clear C(2)
37	SURE?		4992	106	A=C S&X	save in A
38	SURE?		4993	3E0	RTN	
1	SPEED	SPEED	4994	1A0	A=B=C=0	
2	SPEED		4995	130	LDI S&X	
3	SPEED		4996	07A	CON:	
4			4997	1BC	RCR 11	
5	only used in PowerCL		4998	2DC	PT= 13	
6	as a sub-function		4999	1D0	LD@PT- 7	
7			499A	130	LDI S&X	
8	SPEED		499B	01A	CON: 26	Time Module ID#
9	SPEED		499C	0AE	A<>C ALL	
10	SPEED		499D	15C	PT= 6	
11	SPEED		499E	150	LD@PT- 5	
12	SPEED		499F	330	FETCH S&X	
13	SPEED		49A0	366	?A#C S&X	timer present?
14	SPEED		49A1	087	JC +16d	no, -> [LB_AFA5]
15	SPEED		49A2	01A	A=0 M	
16	SPEED		49A3	389	?NC XQ	Enables Timer

17	SPEED		49A4	*140*	*->50E2*	*[ENTMR]*
18	SPEED		49A5	038	READATA	
19	SPEED		49A6	106	A=C S&X	
20	SPEED	LB_AF9B	49A7	038	READATA	
21	SPEED		49A8	366	?A#C S&X	
22	SPEED		49A9	*3F3*	*JNC -02*	*[LB_AF9B]*
23	SPEED	LB_AF9E	49AA	106	A=C S&X	
24	SPEED	LB_AF9F	49AB	17A	A=A+1 M	
25	SPEED		49AC	038	READATA	
26	SPEED		49AD	366	?A#C S&X	
27	SPEED		49AE	***3EB***	***JNC -03***	*[LB_AF9F]*
28	SPEED		49AF	1BE	A=A-1 MS	
29	SPEED		49B0	***3D3***	***JNC -06***	*[LB_AF9E]*
30	SPEED	LB_AFA5	49B1	07A	A<>B M	
31	SPEED		49B2	0CE	C=B ALL	
32	SPEED		49B3	3F0	PRPHSLCT	
33	SPEED		49B4	270	RAMSLCT	
34	SPEED		49B5	03C	RCR 3	
35	SPEED		49B6	106	A=C S&X	
36	SPEED		49B7	27C	RCR 9	
37	SPEED		49B8	268	WRIT 9(Q)	
38	SPEED		49B9	2B8	READ 10(+)	
39	SPEED		49BA	0A6	A<>C S&X	
40	SPEED		49BB	2A8	WRIT 10(+)	
41	SPEED		49BC	106	A=C S&X	
42	SPEED		49BD	*1F5*	*?NC XQ*	
43	SPEED		49BE	*0C4*	*->317D*	*[BIN--D]*
44	SPEED		49BF	2A0	SETDEC	
45	SPEED		49C0	04E	C=0 ALL	
46	SPEED		49C1	35C	PT= 12	
47	SPEED		49C2	050	LD@PT- 1	
48	SPEED		49C3	250	LD@PT- 9	
49	SPEED		49C4	090	LD@PT- 2	***1926 E5***
50	SPEED		49C5	190	LD@PT- 6	
51	SPEED		49C6	130	LDI S&X	
52	SPEED		49C7	005	CON: 5	
53	SPEED		49C8	0AE	A<>C ALL	
54	SPEED		49C9	0EE	C<>B ALL	
55	SPEED		49CA	*261*	*?NC XQ*	
56	SPEED		49CB	*060*	*->1898*	*[DV2_10]*
57	SPEED		49CC	0E8	WRIT 3(X)	
58	SPEED		49CD	3E0	RTN	
1	MSG4	**MSG4**	**49CE**	**0F8**	**READ 3(X)**	
2	MSG4		49CF	2FE	?C#0 MS	
3	MSG4		49D0	*02F*	*JC +05*	
4	MSG4		49D1	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
5	MSG4		49D2	*008*	*->02E3*	*[BCDBIN]*
6	MSG4		49D3	*1C5*	*?NC GO*	
7	MSG4		49D4	*072*	*->1C71*	*[MSGE]*
8	MSG4		49D5	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
9	MSG4		49D6	*008*	*->02E3*	*[BCDBIN]*
10	MSG4		49D7	05E	C=0 MS	
11	MSG4		49D8	05A	C=0 M	
12	MSG4		49D9	1BC	RCR 11	
13	MSG4		49DA	11A	A=C M	
14	MSG4		49DB	*35D*	*?NC XQ*	
15	MSG4		49DC	*000*	*->00D7*	
16	MSG4		49DD	21A	C=C+A M	
17	MSG4		49DE	108	SETF 8	
18	MSG4		49DF	*1D5*	*?NC GO*	
19	MSG4		49E0	*072*	*->1C75*	*[MSGX]*
1	RMLST4	**RMLST4**	**49E1**	*345*	*?NC XQ*	*Clears Alpha*
2	RMLST4		49E2	*040*	*->10D1*	*[CLA]*
3			49E3	1A0	A=B=C=0	
4	*used in ROMLST function*		49E4	15C	PT= 6	
5			49E5	090	LD@PT- 2	*puts "2" in C(6)*
6	RMLST4		49E6	0F0	C<>N ALL	*now in B(6)*

7	RMLST4		49E7	0B0	C=N ALL	*page counter in C(6)*
8	RMLST4		49E8	15C	PT= 6	
9	RMLST4		49E9	222	C=C+1 @PT	*increases page#*
10	RMLST4		49EA	***10F***	***JC +33d***	*exit if last page*
11	RMLST4		49EB	070	N=C ALL	*save updated counter*
12	RMLST4		49EC	330	FETCH S&X	
13	RMLST4		49ED	2E6	?C#0 S&X	
14	RMLST4		49EE	***3CB***	***JNC -07***	
15	RMLST4		49EF	106	A=C S&X	*value to convert in A[S&X]*
16	RMLST4		49F0	23A	C=C+1 M	*next address*
17	RMLST4		49F1	330	FETCH S&X	*read 2nd. Byte*
18	RMLST4		49F2	2E6	?C#0 S&X	*is there any FAT?*
19	RMLST4		49F3	***3DB***	***JNC -05***	*no, next please*
20	RMLST4		49F4	130	LDI S&X	*yes, go ahead*
21	RMLST4		49F5	010	CON: 16	
22	RMLST4		49F6	270	RAM SLCT	*select "sleeper" chip*
23	RMLST4		49F7	01E	A=0 MS	
24	RMLST4		49F8	*3A1*	*?NC XQ*	*Generate number ->display!*
25	RMLST4		49F9	*014*	*->05E8*	*[GENNUM]*
26	RMLST4		49FA	*149*	*?NC XQ*	*Enable Chip0*
27	RMLST4		49FB	*024*	*->0952*	*[ENCP00]*
28	RMLST4		49FC	0AE	A<>C ALL	*put result into C[M]*
29	RMLST4		49FD	1BC	RCR 11	put it in C(1:0)
30	RMLST4		49FE	0E6	C<>B S&X	
31	RMLST4		49FF	0C6	C=B S&X	
32	RMLST4		4A00	31C	PT= 1	
33	RMLST4		4A01	*3ED*	*?NC XQ*	*Append Digit*
34	RMLST4		4A02	*07C*	*->1FFB*	*[APNDDG]+*
35	RMLST4		4A03	0C6	C=B S&X	
36	RMLST4		4A04	*3ED*	*?NC XQ*	*Append Digit*
37	RMLST4		4A05	*07C*	*->1FFB*	*[APNDDG]+*
38	RMLST4		4A06	*29D*	*?NC XQ*	*append Colon ":"*
39	RMLST4		4A07	*2BC*	*->AFA7*	*[APNDCL]*
40	RMLST4		4A08	000		
41	RMLST4		4A09	000		
42	RMLST4		4A0A	***34B***	***JNC -23d***	
43	RMLST4		4A0B	*2E5*	*?NC XQ*	*Delete Alpha last chr*
44	RMLST4		4A0C	*110*	*->44B9*	*[ABSP4]*
45	RMLST4		4A0D	*2D5*	***?NC GO***	*Shows ALPHA if not in PRGM*
46	RMLST4		4A0E	*112*	*->44B5*	*[XAVEW?]*
1	T>BS4	BASE36	4A0F	*20D*	*?NC XQ*	*Build Msg - UF25 Clear*
2	T>BS4		4A10	*0FC*	*->3F83*	*[APERMSG]*
3	T>BS4		4A11	*002*	***"B"***	
4	T>BS4		4A12	*001*	***"A"***	
5	T>BS4		4A13	*013*	***"S"***	*"BASE>36"*
6	T>BS4		4A14	*005*	***"E"***	
7	T>BS4		4A15	*03E*	***">"***	
8	T>BS4		4A16	*033*	***"3"***	
9	T>BS4		4A17	*236*	***"6"***	
10	T>BS4		4A18	*1F1*	*?NC GO*	*LeftJ, Show and Halt*
11	T>BS4		4A19	*0FE*	*->3F7C*	*[APEREX]*
12	T>BS4	**T>BS4**	**4A1A**	***04C***	***?FSET 4***	***Decimal to Base***
13	T>BS4		4A1B	***01F***	***JC +03***	*SST'ing a program*
14	T>BS4		4A1C	2CC	?FSET 13	
15	T>BS4		4A1D	***043***	***JNC +08***	*NOT RUN'ing a program*
16	T>BS4		4A1E	*1A5*	*?NC XQ*	*Check for valid entries*
17	T>BS4		4A1F	*100*	*->4069*	*[CHKST2]*
18	T>BS4		4A20	0AE	A<>C ALL	*Y has base*
19	T>BS4		4A21	260	SETHEX	
20	T>BS4		4A22	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
21	T>BS4		4A23	*008*	*->02E3*	*[BCDBIN]*
22	T>BS4		4A24	10E	A=C ALL	
23	T>BS4		4A25	1A6	A=A-1 S&X	
24	T>BS4		4A26	346	?A#0 S&X	
25	T>BS4		4A27	*0B5*	*?NC GO*	*"Data Error"*
26	T>BS4		4A28	*0A2*	*->282D*	*[ERRDE]*

27	T>BS4		4A29	166	A=A+1 S&X	
28	T>BS4		4A2A	130	LDI S&X	
29	T>BS4		4A2B	025	CON: 37	
30	T>BS4		4A2C	306	?A<C S&X	*is < 37?*
31	T>BS4		4A2D	*313*	*JNC -30d*	*Error msg.*
32	T>BS4		4A2E	*1F5*	*?NC XQ*	
33	T>BS4		4A2F	*0C4*	*->317D*	*[BIN--D]*
34	T>BS4		4A30	0CE	C=B ALL	
35	T>BS4		4A31	***128***	***WRIT 4(L)***	*repository for base*
36	T>BS4		4A32	130	LDI S&X	
37	T>BS4		4A33	00C	CON: 12	*number of mantissa digits*
38	T>BS4		4A34	***268***	***WRIT 9(Q)***	*counter value = 12*
39	T>BS4		4A35	0F8	READ 3(X)	
40	T>BS4		4A36	10E	A=C ALL	
41	T>BS4		4A37	*3C1*	*?NC XQ*	*Enable & Clear Disp*
42	T>BS4		4A38	*0B0*	*->2CF0*	*[CLLCDE]*
43	T>BS4		4A39	34E	?A#0 ALL	
44	T>BS4		4A3A	*027*	*JC +04*	
45	T>BS4		4A3B	130	LDI S&X	
46	T>BS4		4A3C	030	***"0"***	
47	T>BS4		4A3D	***328***	***WRIT 12(b)***	*write to Display*
48	T>BS4		4A3E	*149*	*?NC XQ*	*Disable PER, enable RAM*
49	T>BS4		4A3F	*024*	*->0952*	*[ENCP00]*
50	T>BS4		4A40	0F8	READ 3(X)	*decimal number*
51	T>BS4		4A41	*361*	*?NC XQ*	*(this includes SETDEC)*
52	T>BS4		4A42	*050*	*->14D8*	*[CHK_NO_S]*
53	T>BS4		4A43	2E0	DSPOFF	*eliminates visual noise*
54	T>BS4		4A44	05E	C=0 MS	*absolute value*
55	T>BS4		4A45	088	SETF 5	*Take Integer*
56	T>BS4		4A46	*0ED*	*?NC XQ*	*Doesn't need DEC mode*
57	T>BS4		4A47	*064*	*->193B*	*[INTFRC]*
58	T>BS4		4A48	2FA	?C#0 M	*non-zero mantissa?*
59	T>BS4		4A49	***03F***	***JC +07***	*YES, not done yet*
60	T>BS4		4A4A	260	SETHEX	*no, we're done*
61	T>BS4		4A4B	*399*	*?NC XQ*	*Copy Display to Alpha*
62	T>BS4		4A4C	*108*	*->42E6*	*[DTOA4]*
63	T>BS4		4A4D	320	DSPTOG	*turn display back on*
64	T>BS4		4A4E	*2D5*	***?NC GO***	*Shows ALPHA if not in PRGM*
65	T>BS4		4A4F	*112*	*->44B5*	*[XAVEW?]*
66	T>BS4		4A50	***158***	***M=C ALL***	*int(x)*
67	T>BS4		4A51	10E	A=C ALL	
68	T>BS4		4A52	138	READ 4(L)	*Base*
69	T>BS4		4A53	070	N=C ALL	
70	T>BS4		4A54	*171*	*?NC XQ*	
71	T>BS4		4A55	*064*	*->195C*	*[MOD10]*
72	T>BS4		4A56	260	SETHEX	
73	T>BS4		4A57	*38D*	*?NC XQ*	*Convert it to hex - uses F8*
74	T>BS4		4A58	*008*	*->02E3*	*[BCDBIN]*
75	T>BS4		4A59	106	A=C S&X	
76	T>BS4		4A5A	130	LDI S&X	
77	T>BS4		4A5B	030	***"0"***	
78	T>BS4		4A5C	146	A=A+C S&X	
79	T>BS4		4A5D	130	LDI S&X	
80	T>BS4		4A5E	03A	***"starburst"***	
81	T>BS4		4A5F	306	?A<C S&X	*is (chr+30)<"9+1"?*
82	T>BS4		4A60	*01F*	*JC +03*	
83	T>BS4		4A61	1C6	A=A-C S&X	
84	T>BS4		4A62	166	A=A+1 S&X	
85	T>BS4		4A63	*3D9*	*?NC XQ*	*Enable LDC but not clear*
86	T>BS4		4A64	*01C*	*->07F6*	*[ENLCD]*
87	T>BS4		4A65	0A6	A<>C S&X	
88	T>BS4		4A66	***328***	***WRIT 12(b)***	*write to Display*
89	T>BS4		4A67	*149*	*?NC XQ*	*Disable PER, enable RAM*
90	T>BS4		4A68	*024*	*->0952*	*[ENCP00]*
91	T>BS4		4A69	***278***	***READ 9(Q)***	*counter value*
92	T>BS4		4A6A	266	C=C-1 S&X	*See if we overflow?*
93	T>BS4		4A6B	*289*	*?C GO*	*"Out of Range"*

```
94  T>BS4                    4A6C  003   ->00A2          [ERROF]
95  T>BS4                    4A6D  268   WRIT 9(Q)       new counter value
96  T>BS4                    4A6E  2A0   SETDEC
97  T>BS4                    4A6F  198   C=M ALL         int(x)
98  T>BS4                    4A70  10E   A=C ALL
99  T>BS4                    4A71  138   READ 4(L)       Base
100 T>BS4                    4A72  261   ?NC XQ
101 T>BS4                    4A73  060   ->1898          [DV2_10]
102 T>BS4                    4A74  28B   JNC -47d
1   ROMOK?   ROMOK?          4A75  0F8   READ 3(X)
2   ROMOK?                   4A76  361   ?NC XQ          (includes SETDEC)
3   ROMOK?                   4A77  050   ->14D8          [CHK_NO_S]
4   ROMOK?                   4A78  260   SETHEX
5   ROMOK?                   4A79  38D   ?NC XQ          Convert it to hex - uses F8
6   ROMOK?                   4A7A  008   ->02E3          [BCDBIN]
7   ROMOK?                   4A7B  2E6   ?C#0 S&X         yes, is it zero?
8   ROMOK?   BADROM          4A7C  0B5   ?NC GO          yes, "DATA ERROR"
9   ROMOK?                   4A7D  0A2   ->282D          [ERRDE]
10  ROMOK?                   4A7E  106   A=C S&X         no, go on
11  ROMOK?                   4A7F  130   LDI S&X
12  ROMOK?                   4A80  021   CON:            limit chr >=ROM id# 33
13  ROMOK?                   4A81  306   ?A<C S&X        lower than limit?
14  ROMOK?                   4A82  3D3   JNC -06         no, bail out
15  ROMOK?                   4A83  3E0   RTN
1   PGOK?    PGOK?           4A84  0F8   READ 3(X)       checks that 3 < pg# < F
2   PGOK?                    4A85  38D   ?NC XQ          Convert it to hex - uses F8
3   PGOK?                    4A86  008   ->02E3          [BCDBIN]
4   PGOK?    PGOK4           4A87  33C   RCR 1           put pg# in C[MS]
5   PGOK?                    4A88  2E6   ?C#0 S&X        larger than "F"?
6   PGOK?                    4A89  381   ?C GO           yes, show "NONEXISTENT"
7   PGOK?                    4A8A  00B   ->02E0          [ERRNE]
8   PGOK?    PGOK8           4A8B  35C   PT= 12
9   PGOK?                    4A8C  04A   C=0 PT<-        clear all C save MS
10  PGOK?                    4A8D  2BC   RCR 7           move pg# to C(6)
11  PGOK?                    4A8E  10E   A=C ALL         copy in A(7)
12  PGOK?                    4A8F  15C   PT= 6
13  PGOK?                    4A90  0D0   LD@PT- 3        load lower limit in C(6)
14  PGOK?                    4A91  15C   PT= 6
15  PGOK?                    4A92  302   ?A<C @PT        is p<limit?
16  PGOK?                    4A93  34F   JC -23d         yes, LB_AF8D
17  PGOK?                    4A94  0AE   A<>C ALL        get PG# in C<6>
18  PGOK?                    4A95  10E   A=C ALL         keep it in A<6>
19  PGOK?                    4A96  3E0   RTN
1   PLNG     CCDD            4A97  04E   C=0 ALL
2   PLNG                     4A98  0A2   A<>C @PT
3   PLNG                     4A99  345   ?NC XQ          Halve Mantissa
4   PLNG                     4A9A  13C   ->4FD1          [M=M/2]
5   PLNG                     4A9B  0A2   A<>C @PT
6   PLNG                     4A9C  206   C=C+A S&X
7   PLNG                     4A9D  1E6   C=C+C S&X
8   PLNG                     4A9E  146   A=A+C S&X
9   PLNG                     4A9F  1E6   C=C+C S&X
10  PLNG                     4AA0  1A2   A=A-1 @PT
11  PLNG                     4AA1  01F   JC +03
12  PLNG                     4AA2  166   A=A+1 S&X
13  PLNG                     4AA3  3EB   JNC -03
14  PLNG                     4AA4  002   A=0 @PT
15  PLNG                     4AA5  14A   A=A+C PT<-
16  PLNG                     4AA6  3E0   RTN
1   <GAP>                    4AA7  000   NOP
2   <GAP>                    4AA8  000   NOP
1   RSTFLG   RSTFLG          4AA9  149   ?NC XQ          Disable PER, enable RAM
2   Resets flags status      4AAA  024   ->0952          [ENCP00]
3   saved in L previously    4AAB  138   READ 4(L)       recover flags
4            UPDATE          4AAC  260   SETHEX          restore status
5   RSTFLG                   4AAD  16D   ?NC GO          writes d & refresh annunciators
6   RSTFLG                   4AAE  01E   ->075B          [ANN+14]
```

```
1  UNTERR  NOSUCH  4AAF  321  ?NC XQ          Show "NO_" msg
2  UNTERR          4AB0  10C  ->43C8          [NOMSG4]
3  UNTERR          4AB1  013  "S"
4  UNTERR          4AB2  015  "U"
5  UNTERR          4AB3  003  "C"
6  UNTERR          4AB4  008  "H"
7  UNTERR          4AB5  020  " "             "NO SUCH UNIT"
8  UNTERR          4AB6  015  "U"
9  UNTERR          4AB7  00E  "N"
10 UNTERR          4AB8  009  "I"
11 UNTERR          4AB9  214  "T"
12 UNTERR          4ABA  07B  JNC +15d
13 UNTERR  INVALID 4ABB  20D  ?NC XQ          Build Msg - UF25 Clear
14 UNTERR          4ABC  0FC  ->3F83          [APERMSG]
15 UNTERR          4ABD  009  "I"
16 UNTERR          4ABE  00E  "N"
17 UNTERR          4ABF  016  "V"
18 UNTERR          4AC0  001  "A"
19 UNTERR          4AC1  00C  "L"
20 UNTERR          4AC2  009  "I"             "INVALID CONV"
21 UNTERR          4AC3  004  "D"
22 UNTERR          4AC4  020  " "
23 UNTERR          4AC5  003  "C"
24 UNTERR          4AC6  00F  "O"
25 UNTERR          4AC7  00E  "N"
26 UNTERR          4AC8  216  "V"
27 UNTERR          4AC9  06B  JNC +13d
28 UNTERR  SYXERR  4ACA  20D  ?NC XQ          Build Msg - UF25 Clear
29 UNTERR          4ACB  0FC  ->3F83          [APERMSG]
30 UNTERR          4ACC  013  "S"
31 UNTERR          4ACD  019  "Y"
32 UNTERR          4ACE  00E  "N"
33 UNTERR          4ACF  014  "T"
34 UNTERR          4AD0  001  "A"             "SYNTAX ERR"
35 UNTERR          4AD1  018  "X"
36 UNTERR          4AD2  020  " "
37 UNTERR          4AD3  005  "E"
38 UNTERR          4AD4  012  "R"
39 UNTERR          4AD5  212  "R"
40 UNTERR          4AD6  1F1  ?NC GO          LeftJ, Show and Halt
41 UNTERR          4AD7  0FE  ->3F7C          [APEREX]
1  TOLER4  TOLER4  4AD8  01E  A=0  MS         absolute value
2                  4AD9  04E  C=0  ALL
3  expects error value stored
4  in {A,B} in 13-digit form
                   4ADA  2BE  C=-C-1 MS
                   4ADB  35C  PT=12
5                  4ADC  050  LD@PT- 1        C= -1 E-9
6  TOLER4          4ADD  266  C=C-1 S&X
7  TOLER4          4ADE  39C  PT= 0
8  TOLER4          4ADF  050  LD@PT- 1
9  TOLER4          4AE0  025  ?NC GO
10 TOLER4          4AE1  062  ->1809          [AD1_10]
1  <GAP>           4AE2  000  NOP
1  PPLNG4  PPLNG4  4AE3  275  ?NC XQ          Get LBL address
2  PPLNG4          4AE4  118  ->469D          [CCDC4]
3  PPLNG4          4AE5  198  C=M ALL
4  PPLNG4          4AE6  1C2  A=A-C @PT
5  PPLNG4          4AE7  023  JNC  +04        E4E0
6  PPLNG4          4AE8  1A2  A=A-1 @PT
7  PPLNG4          4AE9  1A2  A=A-1 @PT
8  PPLNG4          4AEA  1A6  A=A-1  S&X
9  PPLNG4          4AEB  1C6  A=A-C  S&X
10 PPLNG4          4AEC  25D  ?NC XQ          ????
11 PPLNG4          4AED  128  ->4A97          [CCDD]
12 PPLNG4          4AEE  166  A=A+1  S&X
13 PPLNG4          4AEF  166  A=A+1  S&X
14 PPLNG4          4AF0  01E  A=0  MS
```

```
15  PPLNG4                  4AF1  3C1  ?NC XQ          Enable and Clear Display
16  PPLNG4                  4AF2  0B0  ->2CF0         [CLLCDE]
17  PPLNG4                  4AF3  3A1  ?NC XQ
18  PPLNG4                  4AF4  014  ->05E8         [GENNUM]
19  PPLNG4  DSPBYTS         4AF5  3BD  ?NC XQ
20  PPLNG4                  4AF6  01C  ->07EF         [MESSL]
21  PPLNG4                  4AF7  020  " "
22  PPLNG4                  4AF8  002  "B"
23  PPLNG4                  4AF9  019  "Y"            " BYTES"
24  PPLNG4                  4AFA  014  "T"
25  PPLNG4                  4AFB  005  "E"
26  PPLNG4                  4AFC  213  "S"
27  PPLNG4                  4AFD  201  ?NC GO
28  PPLNG4                  4AFE  072   ->1C80        [MSG105]
1   NORMS   NRM3D           4AFF  199  ?NC XQ          Checks XYZ - sets DEC mode
2   NORMS                   4B00  100  ->4066         [CHKST3]
3   NORMS                   4B01  10E  A=C ALL        x value is in C
4                           4B02  135  ?NC XQ          x^2
5   final result is placed in {A,B}
                            4B03  060  ->184D         [MP2_10]
6   in 13-digit form, and   4B04  089  ?NC XQ          x^2
7   also in C in 10-digit form.
                            4B05  064  ->1922         [STSCR]
8                           4B06  078  READ 1(Z)
9   NORMS                   4B07  10E  A=C ALL
10  NORMS                   4B08  135  ?NC XQ          z^2
11  NORMS                   4B09  060  ->184D         [MP2_10]
12  NORMS                   4B0A  0D1  ?NC XQ          x^2
13  NORMS                   4B0B  064  ->1934         [RCSCR]
14  NORMS                   4B0C  031  ?NC XQ          x^2+z^2
15  NORMS                   4B0D  060  ->180C         [AD2-13]
16  NORMS                   4B0E  089  ?NC XQ
17  NORMS                   4B0F  064  ->1922         [STSCR]
18  NORMS                   4B10  043  JNC +08        [NRM20]
19  NORMS   NRM2D           4B11  1A5  ?NC XQ          Checks XY - sets DEC mode
20  NORMS                   4B12  100  ->4069         [CHKST2]
21  NORMS   NRM10           4B13  10E  A=C ALL        x value is in C
22  NORMS                   4B14  135  ?NC XQ
23  NORMS                   4B15  060  ->184D         [MP2_10]
24  NORMS                   4B16  089  ?NC XQ          x^2
25  NORMS                   4B17  064  ->1922         [STSCR]
26  NORMS   NRM20           4B18  0B8  READ 2(Y)
27  NORMS                   4B19  10E  A=C ALL
28  NORMS                   4B1A  135  ?NC XQ          y^2
29  NORMS                   4B1B  060  ->184D         [MP2_10]
30  NORMS                   4B1C  0D1  ?NC XQ          x^2
31  NORMS                   4B1D  064  ->1934         [RCSCR]
32  NORMS                   4B1E  031  ?NC XQ          |z|^2
33  NORMS                   4B1F  060  ->180C         [AD2-13]
34  NORMS                   4B20  305  ?NC GO
35  NORMS                   4B21  062  ->18C3         [SQR13]
1   ST<>Σ   ST<>Σ           4B22  130  LDI S&X
2   ST<>Σ                   4B23  004  CON: 4         L register address
3   ST<>Σ                   4B24  10E  A=C ALL        save it in A
4   ST<>Σ                   4B25  04E  C=0 ALL
5   ST<>Σ                   4B26  206  C=C+A S&X      absolute address
6   ST<>Σ                   4B27  270  RAMSLCT        Select "From-To" block
7   ST<>Σ                   4B28  038  READATA
8   ST<>Σ                   4B29  0EE  B<>C ALL       Save it in B
9   ST<>Σ                   4B2A  378  READ  13( c)
10  ST<>Σ                   4B2B  1BC  RCR 11         location of ΣREG
11  ST<>Σ                   4B2C  206  C=C+A S&X
12  ST<>Σ                   4B2D  270  RAMSLCT
13  ST<>Σ                   4B2E  038  READATA
14  ST<>Σ                   4B2F  0EE  B<>C ALL
15  ST<>Σ                   4B30  2F0  WRITDATA
16  ST<>Σ                   4B31  04E  C=0 ALL
17  ST<>Σ                   4B32  206  C=C+A S&X
18  ST<>Σ                   4B33  270  RAMSLCT
```

19	ST<>Σ		4B34	0EE	B<>C ALL	
20	ST<>Σ		4B35	2F0	WRITDATA	
21	ST<>Σ		4B36	1A6	A=A-1 S&X	*Will set Carry when A=0*
22	ST<>Σ		4B37	*373*	*JNC -18d*	
23	ST<>Σ		4B38	**3E0**	**RTN**	
1	NDF4	**NDF4**	**4B39**	**2A0**	**SETDEC**	
2	NDF4		4B3A	**078**	**READ 1(Z)**	*μ*
3	NDF4		4B3B	2BE	C=-C-1 MS	
4			4B3C	10E	A=C ALL	
5	*Probability (Normal)*		4B3D	**0F8**	**READ 3(X)**	*x*
6	*Density Function*		4B3E	*01D*	*?NC XQ*	*x- μ*
7			4B3F	*060*	*->1807*	*[AD2_10]*
8	*expects CPU in DEC mode*		4B40	**0B8**	**READ 2(Y)**	*σ*
9			4B41	*269*	*?NC XQ*	
10	NDF4		4B42	*060*	*->189A*	*[DV1-10]*
11	NDF4		4B43	*13D*	*?NC XQ*	*[(x- μ)/σ]^2*
12	NDF4		4B44	*060*	*->184F*	*[MP1_10]*
13	NDF4		4B45	*3D9*	*?NC XQ*	*{A,B} = {A,B} /2*
14	NDF4		4B46	*13C*	*->4FF6*	*[DIVBY2]*
15	NDF4		4B47	2BE	C=-C-1 MS	*Sign change*
16	NDF4		4B48	11E	A=C MS	*ditto in 13-digit form*
17	NDF4		4B49	044	CLRF 4	*don't subtract 1*
18	NDF4		4B4A	*035*	*?NC XQ*	
19	NDF4		4B4B	*068*	*->1A0D*	*[EXP13]*
20	NDF4		4B4C	**0B8**	**READ 2(Y)**	*σ*
21	NDF4		4B4D	*269*	*?NC XQ*	
22	NDF4		4B4E	*060*	*->189A*	*[DV1-10]*
23	NDF4		4B4F	*089*	*?NC XQ*	*partial result*
24	NDF4		4B50	*064*	*->1922*	*[STSCR]*
25	NDF4		4B51	00E	A=0 ALL	
26	NDF4		4B52	*269*	*?NC XQ*	
27	NDF4		4B53	*064*	*->199A*	*[PI/2]*
28	NDF4		4B54	1EE	C=C+C ALL	
29	NDF4		4B55	1EE	C=C+C ALL	*2 π*
30	NDF4		4B56	0EE	B<>C ALL	
31	NDF4		4B57	*305*	*?NC XQ*	
32	NDF4		4B58	*060*	*->18C1*	*[SQR13]*
33	NDF4		4B59	*0D1*	*?NC XQ*	*partial result*
34	NDF4		4B5A	*064*	*->1934*	*[RCSCR]*
35	NDF4		4B5B	*24D*	*?NC XQ*	*{M,C} / {A,B}*
36	NDF4		4B5C	*060*	*->1893*	*[X/Y13]*
37	NDF4		4B5D	*331*	*?NC GO*	*Overflow, DropST, FillXL & Exit*
38	NDF4		4B5E	*002*	*->00CC*	*[NFRX]*
1	*<GAP>*		4B5F	*000*	*NOP*	
2	*<GAP>*		4B60	*000*	*NOP*	
3	*<GAP>*		4B61	*000*	*NOP*	
4	*<GAP>*		4B62	*000*	*NOP*	
1	OFSHF4	**OFSHF4**	**4B63**	*149*	*?NC XQ*	*Disable PER, enable RAM*
2	OFSHF4		4B64	*024*	*->0952*	*[ENCP00]*
3	OFSHF4		4B65	*125*	*?NC GO*	*Switch off SHIFT & F47*
4	OFSHF4		4B66	*01E*	*->0749*	*[OFSHFT]*
1	HARMN	**HRMN**	**4B67**	**070**	**N=C ALL**	***argument***
2	HARMN		4B68	1A0	A=B=C=0	*zero trinity - initial sum!*
3			4B69	*089*	*?NC XQ*	*uses {Q,+} (!)*
4	*expects CPU in DEC mode*		4B6A	*064*	*->1922*	*[STSCR]*
5			4B6B	*0B0*	*C=N ALL*	*argument to C*
6	HARMN		4B6C	2EE	?C#0 ALL	*Carry set if NOT Zero*
7	HARMN		4B6D	*0A9*	*?NC GO*	*leave result in {A,B}*
8	HARMN		4B6E	*066*	*->192A*	*[EXSCR]*
9	HARMN		4B6F	***128***	***WRIT 4(L)***	*k as Counter to L*
10	HARMN		4B70	3CC	?KEY	
11	HARMN		4B71	360	?C RTN	*bail out upon key depressed*
12	HARMN		4B72	*22D*	*?NC XQ*	*1/k*
13	HARMN		4B73	*060*	*->188B*	*[1/X_10]*
14	HARMN		4B74	*0D1*	*?NC XQ*	*Σ(k)*
15	HARMN		4B75	*064*	*->1934*	*[RCSCR]*
16	HARMN		4B76	*031*	*?NC XQ*	*Σ(k)+1/k*

$$f(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}}$$

17	HARMN		4B77	060	->180C	[AD2-13]
18	HARMN		4B78	070	N=C ALL	new Partial Sum in N
19	HARMN		4B79	089	?NC XQ	Σ(k) + 1/k
20	HARMN		4B7A	064	->1922	[STSCR]
21	HARMN		4B7B	138	READ 4(L)	Recall n
22	HARMN		4B7C	1FD	?NC XQ	{A,B} = C-1
23	HARMN		4B7D	100	->407F	[DECC10]
24	HARMN		4B7E	373	JNC -18d	
1	TSVRES	WAIT4L	4B7F	271	?NC XQ	Save current Turbo
2	TSVRES		4B80	12C	->4B9C	[TSAVE]
3			4B81	37D	?NC XQ	Slow things down
4	waits for a while...		4B82	13C	->4FDF	[TURBO0]
5	compatible with current		4B83	130	LDI S&X	
6	turbo settings		4B84	1AF	CON:	initial delay value
7			4B85	3CC	?KEY	Key pressed?
8	TSVRES		4B86	017	JC +02	yes, skip addition
9	TSVRES		4B87	1E6	C=C+C S&X	no, delay value is doubled
10	TSVRES		4B88	3C8	CLRKEY	Small delay
11	TSVRES		4B89	266	C=C-1 S&X	loop
12	TSVRES		4B8A	3FB	JNC -01	
13	TSVRES	TREST	4B8B	130	LDI S&X	point at on-chip peripheral
14	TSVRES		4B8C	3F0	CON:	3F0
15	TSVRES		4B8D	270	RAMSLCT	
16			4B8E	3F0	PRPHSLCT	
17	restore previous TURBO		4B8F	2A1	?NC XQ	Load RG address
18	previously saved in sRAM		4B90	12C	->4BA8	[804018]
19			4B91	150	LD@PT- 5	"read" command code
20	TSVRES		4B92	1FC	WCMD	read stored turbo at 0x804018
21	TSVRES		4B93	0B8	READ 2(Y)	get the value
22	TSVRES		4B94	0A6	A<>C S&X	put it aside
23	TSVRES		4B95	130	LDI S&X	
24	TSVRES		4B96	008	CON:	008
25	TSVRES		4B97	206	C=C+A S&X	and add 0x8 to form turbo command
26	TSVRES		4B98	0FC	RCR 10	align the command
27	TSVRES	EXIT	4B99	1FC	WCMD	and issue it
28	TSVRES		4B9A	149	?NC GO	Enables Chp0
29	TSVRES		4B9B	026	->0952	[ENCP00]
30	TSVRES	TSAVE	4B9C	130	LDI S&X	point at on-chip peripheral
31			4B9D	3F0	CON:	3F0
32	saves current TURBO setting		4B9E	270	RAMSLCT	
33	in RAM register located at		4B9F	3F0	PRPHSLCT	
34	address 0x804018		4BA0	138	READ 4(L)	read turbo mode
35			4BA1	2A1	?NC XQ	Load RG address
36	TSVRES		4BA2	12C	->4BA8	[804018]
37			4BA3	110	LD@PT- 4	"write" command code
38	CL chip remains selected!		4BA4	1FC	WCMD	store turbo mode at 0x804018
39			4BA5	05C	PT= 4	
40	TSVRES		4BA6	210	LD@PT- 8	
41	TSVRES		4BA7	1FC	WCMD	CL chip remains selected
42	TSVRES	804018	4BA8	19C	PT= 11	(warning!)
43	TSVRES		4BA9	210	LD@PT- 8	
44	TSVRES		4BAA	010	LD@PT- 0	
45	TSVRES		4BAB	110	LD@PT- 4	
46	TSVRES		4BAC	010	LD@PT- 0	
47	TSVRES		4BAD	050	LD@PT- 1	
48	TSVRES		4BAE	210	LD@PT- 8	
49	TSVRES		4BAF	010	LD@PT- 0	
50	TSVRES		4BB0	3E0	RTN	
1	LASTF	NOBUFR	4BB1	355	?NC GO	Show "NO BUFF" msg
2	LASTF		4BB2	10E	->43D5	[NOBUFR] - Halts execution
3	LASTF	LASTF4	4BB3	0E6	C<>B &X	offset to B[S&X]
4	LASTF		4BB4	215	?NC XQ	Build Msg - all cases
5	LASTF		4BB5	0FC	->3F85	[APRMSG2]
6	LASTF		4BB6	00C	"L"	
7	LASTF		4BB7	001	"A"	"LASTF"
8	LASTF		4BB8	013	"S"	
9	LASTF		4BB9	014	"T"	

10	LASTF		4BBA	206	"F"	
11	LASTF		4BBB	3DD	?NCXQ	make it eye-pleasant
12	LASTF		4BBC	0AC	->2BF7	[LEFTJ]
13	LASTF		4BBD	1FD	?NC XQ	wait a little - CL compatible
14	LASTF		4BBE	12C	->4B7F	[WAIT4L] - Enables RAM
15	LASTF		4BBF	130	LDI S&X	
16	LASTF		4BC0	009	CON: 9	buffer id#
17	LASTF		4BC1	2A9	?NC XQ	Check for Buffer
18	LASTF		4BC2	10C	->43AA	[CHKBF4] +1
19	LASTF	NTFND	4BC3	373	JNC -18d	Not Found
20	LASTF	FOUND	4BC4	126	A=A+B S&X	variable offset: 2,3,4
21			4BC5	0A6	A<>C S&X	put adr in C[S&X]
22	SandMatrix 4		4BC6	270	RAMSLCT	Select register
23	SandMath 3		4BC7	038	READATA	Get FNAME
24	PowerCL 2		4BC8	0AE	A<>C ALL	preserve it in A
25	TimeSeed 1		4BC9	046	C=0 S&X	
26	Header		4BCA	270	RAMSLCT	select Chip0
27			4BCB	0AE	A<>C ALL	retrieve value
28	LASTF		4BCC	2E6	?C#0 S&X	valid value?
29	LASTF		4BCD	360	?C RTN	yes, done
30	LASTF		4BCE	321	?NC XQ	Show "NO_" msg
31	LASTF		4BCF	10C	->43C8	[NOMSG4]
32	LASTF		4BD0	00C	"L"	
33	LASTF		4BD1	001	"A"	
34	LASTF		4BD2	013	"S"	"NO LASTF"
35	LASTF		4BD3	014	"T"	
36	LASTF		4BD4	206	"F"	
37	LASTF		4BD5	1F1	?NC GO	LeftJ, Show and Halt
38	LASTF		4BD6	0FE	->3F7C	[APEREX]
1	<GAP>		4BD7	000	NOP	
2	<GAP>		4BD8	000	NOP	
1	CBRSUB	CBRSUB	4BD9	2EE	?C#0 ALL	expects DEC mode
2	CBRSUB		4BDA	3A0	?NC RTN	end here if zero
3			4BDB	104	CLRF 8	
4	cubic root routine		4BDC	2FE	?C#0 MS	is it negative?
5	expects CPU in DEC mode		4BDD	01B	JNC +03	
6			4BDE	05E	C=0 MS	Sign change (ABS)
7	CBRSUB		4BDF	108	SETF 8	remember this fact
8	CBRSUB		4BE0	3C4	ST=0	
9	CBRSUB		4BE1	115	?NC XQ	returns 13-digit form too
10	CBRSUB		4BE2	06C	->1B45	[LN10]
11	CBRSUB		4BE3	3C1	?NC XQ	Divide by 3 - 13digit
12	CBRSUB		4BE4	12C	->4BF0	[DVTRE2]
13	CBRSUB		4BE5	044	CLRF 4	
14	CBRSUB		4BE6	035	?NC XQ	uses scratch {Q,+}
15	CBRSUB		4BE7	068	->1A0D	[EXP13]
16	CBRSUB		4BE8	10C	?FSET 8	negative argument?
17	CBRSUB		4BE9	3A0	?NC RTN	no, done
18	CBRSUB		4BEA	2BE	C=-C-1 MS	yes, change sign
19	CBRSUB		4BEB	11E	A=C MS	in13-digit form too
20	CBRSUB		4BEC	3E0	RTN	done
1	DVTREE	DVTREE	4BED	02E	B=0 ALL	clears B
2	DVTREE		4BEE	0FA	B<>C M	B holds 13-digit mant
3	DVTREE		4BEF	0AE	A<>C ALL	A holds sign and S&X
4	DVTREE	DVTRE2	4BF0	04E	C=0 ALL	
5	DVTREE		4BF1	35C	PT=12	build "3" in C
6	DVTREE		4BF2	0D0	LD@PT-3	
7	DVTREE		4BF3	269	?NC GO	
8	DVTREE		4BF4	062	->189A	[DV1-10]
1	NXTRT3	NXTRT3	4BF5	138	READ 4(L)	next root
2	NXTRT3		4BF6	10E	A=C ALL	
3	NXTRT3		4BF7	04E	C=0 ALL	
4			4BF8	35C	PT = 12	
5	adds 120 deg to L		4BF9	050	LD@PT- 1	
6	expects CPU in DEC mode		4BFA	090	LD@PT- 2	Buids 120 in C
7			4BFB	130	LDI S&X	

8	NXTRT3		4BFC	002	CON: 2	
9	NXTRT3		4BFD	01D	?NC XQ	
10	NXTRT3		4BFE	060	->1807	[AD2_10]
11	NXTRT3		4BFF	128	WRIT 4(L)	
12	NXTRT3	ROOT3	4C00	138	READ 4(L)	first root
13	NXTRT3		4C01	070	N=C ALL	
14			4C02	1F1	?NC XQ	
15	parameters in {T,L}		4C03	048	->127C	[COS]
16	NXTRT3		4C04	11E	A=C MS	bug or what??
17	NXTRT3		4C05	046	C=0 S&X	expects data in T(0)
18	NXTRT3		4C06	270	RAMSLCT	
19	NXTRT3		4C07	038	READATA	2*sqr[xxxxxxx]
20	NXTRT3		4C08	13D	?NC XQ	
21	NXTRT3		4C09	060	->184F	[MP1_10]
22	NXTRT3		4C0A	078	READ 1(Z)	a2/3
23	NXTRT3		4C0B	2BE	C=-C-1 MS	
24	NXTRT3		4C0C	000	NOP	
25	NXTRT3		4C0D	025	?NC GO	
26	NXTRT3		4C0E	062	->1809	[AD1_10]
1	CUBE10	CUBE10	4C0F	070	N=C ALL	store argument
2	CUBE10		4C10	02E	B=0 ALL	clears B
3			4C11	0FA	B<>C M	B holds 13-digit mant
4	cube power routine		4C12	0AE	A<>C ALL	A holds sign and S&X
5	CUBE10		4C13	0B0	C=N ALL	
6	CUBE13	CUBE13	4C14	070	N=C ALL	store argument
7	CUBE13		4C15	13D	?NC XQ	
8	CUBE13		4C16	060	->184F	[MP1_10]
9	CUBE13		4C17	0B0	C=N ALL	
10	CUBE13		4C18	13D	?NC GO	
11	CUBE13		4C19	062	->184F	[MP1_10]
1	LINEUP	LINEUP	4C1A	260	SETHEX	more than needed!
2	LINEUP		4C1B	215	?NC XQ	Build Msg - all cases
3	LINEUP		4C1C	0FC	->3F85	[APRMSG2]
4	LINEUP		4C1D	019	"Y"	
5	LINEUP		4C1E	23D	"="	
6			4C1F	3DD	?NC XQ	
7	displays line equation		4C20	0AC	->2BF7	[LEFTJ]
8			4C21	399	?NC XQ	Copy Display to Alpha
9	LINEUP		4C22	108	->42E6	[DTOA4]
10	LINEUP		4C23	0F8	READ 3(X)	a
11	LINEUP		4C24	0EE	C<>B ALL	required by [AFORMT}
12	LINEUP		4C25	0A0	SLCT P	
13	LINEUP		4C26	1A9	?NC XQ	AFORMT w/ integer support
14	LINEUP		4C27	130	->4C6A	[AFRMT4]
15	LINEUP		4C28	215	?NC XQ	Build Msg - all cases
16	LINEUP		4C29	0FC	->3F85	[APRMSG2]
17	LINEUP		4C2A	02A	"*"	
18	LINEUP		4C2B	218	"X"	
19	LINEUP		4C2C	3DD	?NC XQ	
20	LINEUP		4C2D	0AC	->2BF7	[LEFTJ]
21	LINEUP		4C2E	3A9	?NC XQ	Append Display to Alpha
22	LINEUP		4C2F	108	->42EA	[NXTCHR]
23	LINEUP		4C30	0B8	READ 2(Y)	b
24	LINEUP		4C31	0EE	C<>B ALL	
25	LINEUP		4C32	2DE	?B#0 MS	See if negative
26	LINEUP		4C33	165	?NC XQ	Append "+" TO Alpha
27	LINEUP		4C34	11C	->4759	[APND+]
28	LINEUP		4C35	141	?NC GO	Append B and end Display
29	LINEUP		4C36	102	->4050	[APPND4]
1	TOPOL4	TOPOL4	4C37	141	?NC XQ	Target Set / XY
2	TOPOL4		4C38	130	->4C50	[XYTGST]
3	TOPOL4		4C39	125	?NC XQ	
4			4C3A	074	->1D49	[TOPOL]
5	uses F8 to return (set) or		4C3B	05E	C=0 MS	This can't be negative
6	to terminate w/ [NFRPU]		4C3C	0E8	WRIT 3(X)	write module in X
7			4C3D	0F0	C<>N ALL	bring argument to C

```
8  TOPOL4                  4C3E  06B   JNC +13d
9  TOPOL4   TOREC4         4C3F  141   ?NC XQ          Target Set / XY
10 TOPOL4                  4C40  130   ->4C50           [XYTGST]
11 TOPOL4                  4C41  1D5   ?NC XQ
12 TOPOL4                  4C42  078   ->1E75           [TOREC]
13 TOREC4                  4C43  2FA   ?C#0 M
14 TOREC4                  4C44  017   JC +02
15 TOREC4                  4C45  05E   C=0 ALL          anihilate the S.O.B.
16 TOREC4                  4C46  0E8   WRIT 3(X)
17 TOREC4                  4C47  0F0   C<>N ALL         Echange Mod<>Arg
18 TOREC4                  4C48  2FA   ?C#0 M
19 TOREC4                  4C49  017   JC +02
20 TOREC4                  4C4A  05E   C=0 ALL          anihilate the S.O.B.
21 TOREC4                  4C4B  0A8   WRIT 2(Y)
22 TOREC4                  4C4C  10C   ?FSET 8          first pass?
23 TOREC4                  4C4D  360   ?C RTN           yes, return
24 TOREC4                  4C4E  3C1   ?NC GO           no, terminate
25 TOREC4                  4C4F  002   ->00F0           [NFRPU]
26 TOREC4   XYTGSET        4C50  0B8   READ 2(Y)
27 TOREC4                  4C51  10E   A=C ALL          Y in A
28 TOREC4                  4C52  0F8   READ 3(X)        X in C
29 TOREC4                  4C53  070   N=C ALL          Mod in both C and N
30 TOREC4                  4C54  351   ?NC GO
31 TOREC4                  4C55  086   ->21D4           [TRGSET]
1  DISPXY   DISPYX         4C56  345   ?NC XQ          Clears Alpha
2  DISPXY                  4C57  040   ->10D1          [CLA]
3  DISPXY                  4C58  095   ?NC XQ          append "blank"
4                          4C59  100   ->4025          [APNDBK]
5  used in STLINE, LR, and XFCT  4C5A  0F8   READ 3(X)
6  all in the SandMath     4C5B  0EE   C<>B ALL        store value in B
7                          4C5C  1A9   ?NC XQ          AFORMT w/ integer support
8  DISPXY                  4C5D  130   ->4C6A          [AFRMT4]
9  DISPXY                  4C5E  095   ?NC XQ          append "blank"
10 DISPXY                  4C5F  100   ->4025          [APNDBK]
11 DISPXY                  4C60  130   LDI S&X
12 DISPXY                  4C61  045   "E"             append "E"
13 DISPXY                  4C62  3D5   ?NC XQ
14 DISPXY                  4C63  07C   ->1FF5          [APND10]
15 DISPXY                  4C64  0B8   READ 2(Y)
16 DISPXY                  4C65  0EE   C<>B ALL        store value in B
17 DISPXY                  4C66  269   ?NC XQ          Append INT(B) to Alpha
18 DISPXY                  4C67  108   ->429A          [AINT8]
19 DISPXY                  4C68  14D   ?NC GO          Final Show w/ [LDSST0]
20 DISPXY                  4C69  102   ->4053          [SHOW4]
1  AFRMT4   AFRMT4         4C6A  3B8   READ 14(d)      Save FIX settings
2  AFRMT4                  4C6B  268   WRIT 9(Q)       temporary save
3  AFRMT4                  4C6C  0EE   C<>B ALL        EXPECTS VALUE IN B
4                          4C6D  128   WRIT 4(L)       preserve it in L
5  AFRMT with integer support (!)  4C6E  084   CLRF 5   FRC part
6  uses L(4) and Q(9)      4C6F  0ED   ?NC XQ          Doesn't need DEC mode
7                          4C70  064   ->193B          [INTFRC]
8  AFRMT4                  4C71  0EE   C<>B ALL        save frc(B) in B
9  AFRMT4                  4C72  138   READ 4(L)       recall Value from "safe box"
10 AFRMT4                  4C73  0EE   C<>B ALL        restore value to B
11 AFRMT4                  4C74  2EE   ?C#0 ALL        was it integer?
12 AFRMT4                  4C75  269   ?C GO           yes, adjust and append
13 AFRMT4                  4C76  10A   ->429A          [AINT8]
14 AFRMT4                  4C77  289   ?NC GO          no, just append it
15 AFRMT4                  4C78  10A   ->42A2          [AINT10]
1  STSORT   SRTSUB         4C79  070   N=C ALL
2  STSORT                  4C7A  270   RAM SLCT
3  STSORT                  4C7B  038   READ DATA       read current rg#1
4                          4C7C  10E   A=C ALL
5  sorts registers which addresses  4C7D  0B0   C=N ALL
6  are in C[M] and C[S&X}  4C7E  03C   RCR 3
7                          4C7F  270   RAM SLCT
8  STSORT                  4C80  038   READATA        read current rg#2
```

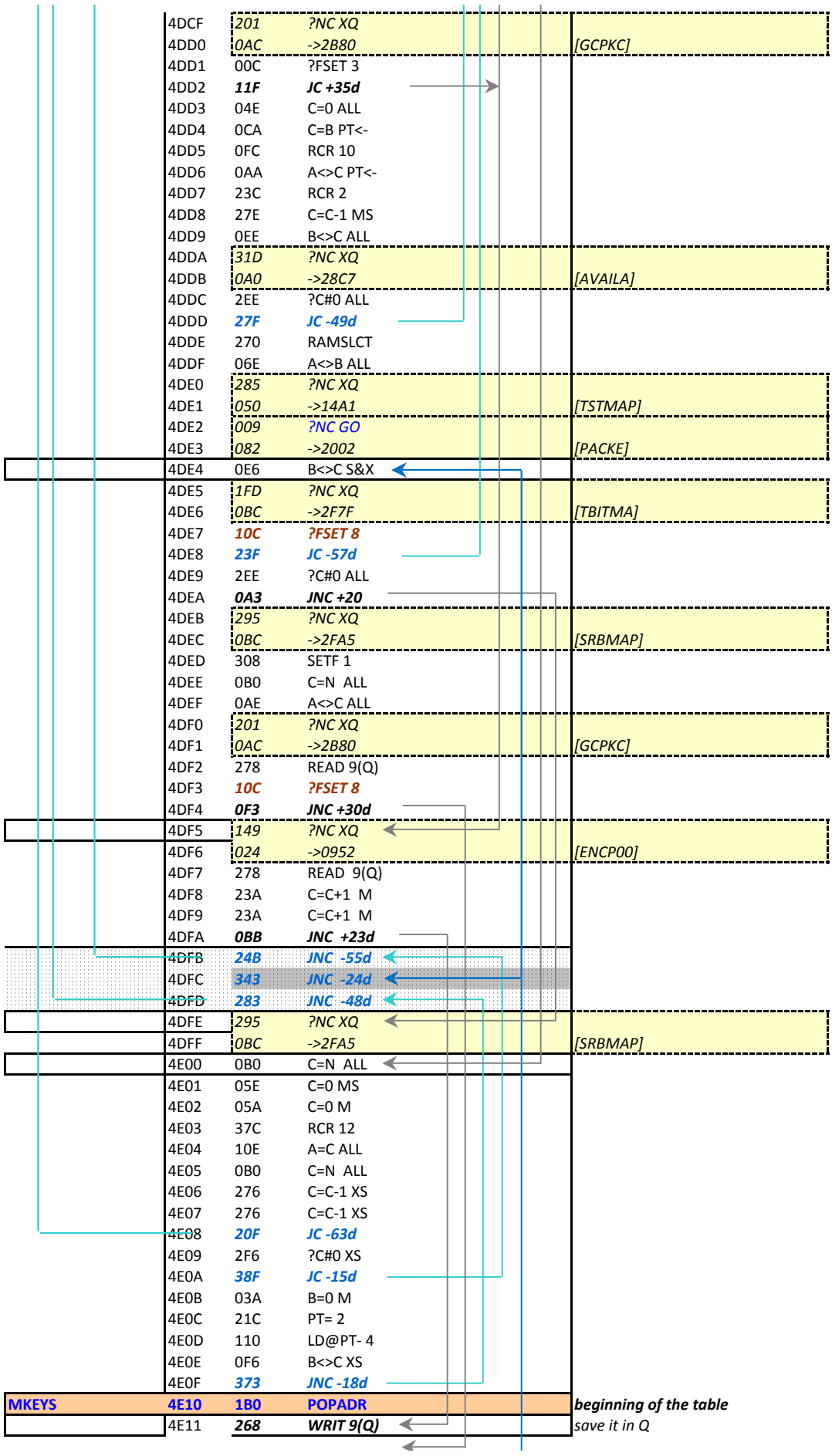
9	STSORT		4C81	*351*	*?NC XQ*	*(includes SETDEC)*
10	STSORT		4C82	*050*	*->14D4*	*[CHK_NO_S1]*
11	STSORT		4C83	2BE	C=-C-1 MS	*-rg#2*
12	STSORT		4C84	000	NOP	*let Carry settle*
13	STSORT		4C85	*01D*	*?NC XQ*	*rg#1-rg#2*
14	STSORT		4C86	*060*	*->1807*	*[AD2_10]*
15	STSORT		4C87	01C	PT= 3	
16	STSORT		4C88	2EE	?C#0 ALL	*were they different?*
17	STSORT		4C89	3A0	?NC RTN	*if the same, return.*
18	STSORT		4C8A	05A	C=0 M	
19	STSORT		4C8B	046	C=0 S&X	
20	STSORT		4C8C	23E	C=C+1 MS	
21	STSORT		4C8D	360	?C RTN	*carry set if <0*
22	STSORT		4C8E	*0B0*	*C=N ALL*	*chkst4*
23	STSORT		4C8F	270	RAM SLCT	
24	STSORT		4C90	038	READ DATA	
25	STSORT		4C91	10E	A=C ALL	
26	STSORT		4C92	*0B0*	*C=N ALL*	
27	STSORT		4C93	03C	RCR 3	
28	STSORT		4C94	270	RAM SLCT	
29	STSORT		4C95	038	READ DATA	
30	STSORT		4C96	0AE	A<>C ALL	
31	STSORT		4C97	2F0	WRIT DATA	
32	STSORT		4C98	*0B0*	*C=N ALL*	
33	STSORT		4C99	270	RAM SLCT	
34	STSORT		4C9A	0AE	A<>C ALL	
35	STSORT		4C9B	2F0	WRIT DATA	
36	STSORT		4C9C	04E	C=0 ALL	
37	STSORT		4C9D	**3E0**	**RTN**	
1	WSIZE4	**WSIZE4**	**4C9E**	130	LDI S&X	
2	WSIZE4		4C9F	005	CON:	buffer id#
3	WSIZE4		4CA0	106	A=C S&X	
4	WSIZE4		4CA1	*2A5*	*?NC XQ*	*Locate Buff (id# in A[S&X])*
5	WSIZE4		4CA2	*10C*	*->43A9*	*[CHKBUF]*
6	WSIZE4		4CA3	*02B*	*JNC +05*	*[LB_ADD0]*
7	WSIZE4		4CA4	0A6	A<>C S&X	*header adr to C[S&X]*
8	WSIZE4		4CA5	226	C=C+1 S&X	*first buffer reg*
9	WSIZE4		4CA6	270	RAMSLCT	*select it*
10	WSIZE4		4CA7	038	READATA	*read content*
11	WSIZE4		4CA8	10E	A=C ALL	*copy in A*
12	WSIZE4		4CA9	046	C=0 S&X	
13	WSIZE4		4CAA	270	RAMSLCT	*select Chip0*
14	WSIZE4		4CAB	378	READ 13(c)	
15	WSIZE4		4CAC	27C	RCR 9	*rotate CCD custom bits to XP*
16	WSIZE4		4CAD	358	ST=C XP	*copy in Status*
17	WSIZE4		4CAE	046	C=0 S&X	*reset C[S&X]*
18	WSIZE4		4CAF	31C	PT= 1	
19	WSIZE4		4CB0	08C	?FSET 5	*is 1CMP on?*
20	WSIZE4		4CB1	*01B*	*JNC +03*	*no, skip -> [LB_ADDC]*
21	WSIZE4		4CB2	110	LD@PT- 4	*yes, mark it as "4"*
22	WSIZE4		4CB3	*023*	*JNC +04*	*[LB_ADDF]*
23	WSIZE4		4CB4	04C	?FSET 4	*is 2CMP on>*
24	WSIZE4		4CB5	*013*	*JNC +02*	*no, skip -> [LB_ADDF]*
25	WSIZE4		4CB6	210	LD@PT- 8	yes, mark it as "8"
26	WSIZE4		4CB7	146	A=A+C S&X	
27	WSIZE4		4CB8	*17D*	*?NC GO*	*[BIN-BCD] plus [RCL]*
28	WSIZE4		4CB9	*0C6*	*->315F*	*[ATOX20]*
1	WAYOUT	**EXECFN**	**4CBA**	*3C1*	*?NC XQ*	*Enable & Clear Disp*
2	WAYOUT		4CBB	*0B0*	*->2CF0*	*[CLLCDE]*
3	WAYOUT		4CBC	198	C=M ALL	*recall fnc id#*
4	WAYOUT	**EXECF4**	**4CBD**	*149*	*?NC XQ*	*Display Function Name*
5	WAYOUT		4CBE	*0BC*	*->2F52*	*[unlabeled]*
6	WAYOUT	**WAYOUT**	**4CBF**	*3DD*	*?NC XQ*	*left justify LCD*
7	WAYOUT		4CC0	*0AC*	*->2BF7*	*[LEFTJ]*
8	WAYOUT	**RESET2**	**4CC1**	*319*	*?NC XQ*	*last chance to cancel out*
9	WAYOUT		4CC2	*038*	*->0EC6*	*[NULTST]*
10	WAYOUT	**RESET4**	**4CC3**	*261*	*?NC XQ*	*Reset keyboard*

11	WAYOUT		4CC4	000	->0098	[RSTKB]
12	WAYOUT		4CC5	169	?NC XQ	Enable Chip0 and reset Seq
13	WAYOUT		4CC6	124	->495A	[EXIT4]
14	WAYOUT	RAK702	4CC7	198	C=M ALL	function id#
15	WAYOUT		4CC8	2E6	?C#0 S&X	Check for valid entries
16	WAYOUT		4CC9	3A0	?NC RTN	**bail out if id#=0**
17	WAYOUT	RAK704	4CCA	01C	PT= 3	complete the fnc. "code"
18	WAYOUT		4CCB	290	LD@PT- A	(not needed for GTRMAD)
19	WAYOUT		4CCC	029	?NC GO	**Execute function!**
20	WAYOUT		4CCD	01E	->070A	[RAK70]
1	AON4	AON4	4CCE	25D	?NC XQ	Load Status bits from d->C
2	AON4		4CCF	01C	->0797	[LDSST0]
3	AON4		4CD0	0F1	?NC GO	set UF48, write in d, update ANNs
4	AON4		4CD1	04E	->133C	[AON]
1	ALPHA4	AOFF4	4CD2	25D	?NC XQ	Load Status bits from d->C
2	ALPHA4		4CD3	01C	->0797	[LDSST0]
3	ALPHA4		4CD4	284	CLRF 7	clears flag 49
4	ALPHA4		4CD5	0ED	?NC XQ	store status bits in d
5	ALPHA4		4CD6	004	->013B	[STOST0]
6	ALPHA4		4CD7	3D9	?NC GO	Enable but not clear LCD
7	ALPHA4		4CD8	01E	->07F6	[ENLCD]
1	FOG	FOG4	4CD9	215	?NC XQ	Build Msg - all cases
2	FOG		4CDA	0FC	->3F85	[APRMSG2]
3			4CDB	003	"C"	
4	ALPHA has the password		4CDC	00F	"O"	
5	so can't use AVIEW in FOCAL		4CDD	004	"D"	"CODING..."
6			4CDE	009	"I"	
7	FOG		4CDF	00E	"N"	
8	FOG		4CE0	207	"G"	
9	FOG		4CE1	02D	?NC XQ	Display not halting - Clears F8
10	FOG		4CE2	100	->400B	[APERX8]
11	FOG	FOG10	4CE3	149	?NC XQ	Select Chip0
12	FOG		4CE4	024	->0952	[ENCP00]
13	FOG		4CE5	178	READ 5(M)	read CODE
14	FOG		4CE6	23E	C=C+1 MS	
15	FOG		4CE7	0A8	WRIT 2(Y)	
16	FOG	ND2D	4CE8	130	LDI S&X	
17	FOG		4CE9	004	CON:	
18	FOG		4CEA	158	M=C ALL	
19	FOG		4CEB	31C	PT= 1	
20	FOG		4CEC	130	LDI S&X	
21	FOG		4CED	080	CON:	
22	FOG		4CEE	106	A=C S&X	
23	FOG		4CEF	178	READ 5(M)	
24	FOG		4CF0	1EE	C=C+C ALL	
25	FOG		4CF1	1EE	C=C+C ALL	divide it by 2
26	FOG		4CF2	1EE	C=C+C ALL	
27	FOG		4CF3	3CE	RSHFC ALL	
28	FOG		4CF4	168	WRIT 5(M)	
29	FOG		4CF5	358	ST=C XP	
30	FOG		4CF6	056	C=0 XS	
31	FOG		4CF7	070	N=C ALL	
32	FOG		4CF8	0DD	?NC XQ	
33	FOG		4CF9	134	->4D37	[SUB1]
34	FOG		4CFA	358	ST=C XP	AE34
35	FOG		4CFB	0DD	?NC XQ	
36	FOG		4CFC	134	->4D37	[SUB1]
37	FOG		4CFD	086	B=A S&X	
38	FOG		4CFE	0F0	C<>N ALL	
39	FOG		4CFF	106	A=C S&X	
40	FOG		4D00	0B0	C=N ALL	
41	FOG		4D01	0FD	?NC XQ	
42	FOG		4D02	134	->4D3F	[SUB2]
43	FOG		4D03	066	A<>B S&X	
44	FOG		4D04	358	ST=C XP	
45	FOG		4D05	0DD	?NC XQ	
46	FOG		4D06	134	->4D37	[SUB1]

47	FOG		4D07	3B0	C=C AND A	
48	FOG		4D08	10A	A=C PT<-	
49	FOG		4D09	178	READ 5(M)	
50	FOG		4D0A	0FC	RCR 10	
51	FOG		4D0B	0EE	C<>B ALL	
52	FOG		4D0C	0CA	C=N PT<-	
53	FOG		4D0D	370	C=C OR A	
54	FOG		4D0E	0EA	C<>B PT<-	
55	FOG		4D0F	0CE	C=B ALL	
56	FOG		4D10	07C	RCR 4	
57	FOG		4D11	168	WRIT 5(M)	
58	FOG		4D12	198	C=M ALL	
59	FOG		4D13	266	C=C=1 S&X	
60	FOG		4D14	158	M=C ALL	
61	FOG		4D15	2E6	?C#0 S&X	
62	FOG		4D16	*2B7*	*JC -42d*	
63	FOG		4D17	0F8	READ 3(X)	
64	FOG		4D18	1BC	RCR 11	
65	FOG		4D19	**330**	**FETCH S&X**	
66	FOG		4D1A	016	A=0 XS	
67	FOG		4D1B	10A	A=C PT<-	
68	FOG		4D1C	0F6	C<>B XS	
69	FOG		4D1D	178	READ 5(M)	
70	FOG		4D1E	0FC	RCR 10	
71	FOG		4D1F	056	C=0 XS	
72	FOG		4D20	0A6	A<>C S&X	
73	FOG		4D21	070	N=C ALL	
74	FOG		4D22	*0FD*	*?NC XQ*	
75	FOG		4D23	*134*	*->4D3F*	*[SUB2]*
76	FOG		4D24	10A	A=C PT<-	
77	FOG		4D25	076	A<>B S&X	
78	FOG		4D26	0F8	READ 3(X)	
79	FOG		4D27	1BC	RCR 11	
80	FOG		4D28	0A6	A<>C S&X	
81	FOG		4D29	**040**	**WROM**	
82	FOG		4D2A	0F8	READ 3(X)	
83	FOG		4D2B	226	C=C+1 S&X	
84	FOG		4D2C	0E8	WRIT 3(X)	
85	FOG		4D2D	106	A=C S&X	
86	FOG		4D2E	21C	PT= 2	*make sure it doesn't mess*
87	FOG		4D2F	3D0	LD@PT- F	*with the polling points,*
88	FOG		4D30	3D0	LD@PT- F	*starting at xFF4*
89	FOG		4D31	110	LD@PT- 4	
90	FOG		4D32	366	?A#C S&X	*reached that point yet?*
91	FOG		4D33	*3C1*	*?NC GO*	*Normal Function Return*
92	FOG		4D34	*002*	*->00F0*	*[NFRPU]*
93	FOG		4D35	*3A1*	***?NC GO***	*get next word*
94	FOG		4D36	*132*	*->4CE8*	*[ND2D]*
95	FOG	**SUB1**	**4D37**	**1E6**	**C=C+C S&X**	
96	FOG		4D38	1E6	C=C+C S&X	*divide it by 2*
97	FOG		4D39	1E6	C=C+C S&X	
98	FOG		4D3A	3C6	RSHFC S&X	
99	FOG		4D3B	38C	?FSET 0	
100	FOG		4D3C	*013*	*JNC +02*	
101	FOG		4D3D	370	C=C OR A	
102	FOG		4D3E	3E0	RTN	
103	FOG	**SUB2**	**4D3F**	**370**	**C=C OR A**	
104	FOG		4D40	0F0	C<>N ALL	
105	FOG		4D41	3B0	C=C AND A	
106	FOG		4D42	2AA	C=-C-1 PT<-	
107	FOG		4D43	10A	A=C PT<-	
108	FOG		4D44	0B0	C=N ALL	
109	FOG		4D45	3B0	C=C AND A	
110	FOG		4D46	3E0	RTN	
1	NATXY	**NATXY**	**4D47**	*1A5*	*?NC XQ*	*Check for valid entries*
2	NATXY		4D48	*100*	*->4069*	*[CHKST2]*
3	NATXY		4D49	0AE	A<>C ALL	

4	NATXY		4D4A	088	SETF 5	*Take Integer part*
5	NATXY		4D4B	*0ED*	*?NC XQ*	*doesn't need DEC mode*
6	NATXY		4D4C	*064*	*->193B*	*[INTFRC]*
7	NATXY		4D4D	05E	C=0 MS	*make it positive >0*
8	NATXY		4D4E	2EE	?C#0 ALL	
9	NATXY		4D4F	*03B*	*JNC +07*	*[ZERO]*
10	NATXY		4D50	***0A8***	***WRIT 2(Y)***	*n (integer, >0)*
11	NATXY	NATX4	4D51	0F8	READ 3(X)	***r***
12	NATXY		4D52	*0ED*	*?NC XQ*	*Take Integer part*
13	NATXY		4D53	*064*	*->193B*	*[INTFRC]*
14	NATXY		4D54	05E	C=0 MS	*make it positive >0*
15	NATXY		4D55	2EE	?C#0 ALL	*Carry set if NOT Zero*
16	NATXY	ZERO	4D56	*30D*	*?NC GO*	
17	NATXY		4D57	*062*	*->18C3*	*[ERR0]*
18	NATXY		4D58	***0E8***	***WRIT 3(X)***	*r (integer, >0)*
19	NATXY		4D59	3E0	RTN	
1	D<>H	**SUBHD**	**4D5A**	**248**	**SETF 9**	
2	D<>H		4D5B	1B8	READ 6(N)	
3	D<>H		4D5C	0EE	C<>B ALL	
4	D<>H		4D5D	130	LDI S&X	
5	D<>H		4D5E	006	CON: 6	
6	D<>H		4D5F	0EE	C<>B ALL	
7	D<>H		4D60	37C	RCR 12	
8	D<>H		4D61	3EE	LSHFA ALL	
9	D<>H		4D62	358	ST=C XP	
10	D<>H		4D63	39C	PT= 0	
11	D<>H		4D64	102	A=C @PT	
12	D<>H		4D65	14C	?FSET 6	
13	D<>H		4D66	*023*	*JNC +04*	
14	D<>H		4D67	250	LD@PT- 9	
15	D<>H		4D68	39C	PT= 0	
16	D<>H		4D69	142	A=A+C @PT	
17	D<>H		4D6A	0EE	C<>B ALL	
18	D<>H		4D6B	266	C=C-1 S&X	
19	D<>H		4D6C	***39B***	***JNC -0D***	
20	D<>H		4D6D	24C	?FSET 9	
21	D<>H		4D6E	***023***	***JNC +04***	
22	D<>H		4D6F	244	CLRF 9	
23	D<>H		4D70	178	READ 5(M)	
24	D<>H		4D71	***35B***	***JNC -15***	
25	D<>H		4D72	0AE	A<>C ALL	
26	D<>H		4D73	**3E0**	**RTN**	
27	D<>H	**SUBDH**	**4D74**	**05A**	**C=M ALL**	
28	D<>H		4D75	0AE	A<>C ALL	
29	D<>H		4D76	2FC	RCR 13	
30	D<>H		4D77	10E	A=C ALL	
31	D<>H		4D78	08E	B=A ALL	
32	D<>H		4D79	130	LDI S&X	
33	D<>H		4D7A	03A	CON: 58	
34	D<>H		4D7B	39C	PT= 0	
35	D<>H		4D7C	302	?A<C @PT	
36	D<>H		4D7D	***02F***	***JC +05***	
37	D<>H		4D7E	31C	PT= 1	
38	D<>H		4D7F	110	LD@PT- 4	
39	D<>H		4D80	1C2	A=A-C @PT	
40	D<>H		4D81	162	A=A+1 @PT	
41	D<>H		4D82	0A2	A<>C @PT	
42	D<>H		4D83	*3D9*	*?NC XQ*	
43	D<>H		4D84	*07C*	*->1FF6*	*[APND10] + 1*
44	D<>H		4D85	198	C=M ALL	
45	D<>H		4D86	266	C=C-1 S&X	
46	D<>H		4D87	360	?C RTN	
47	D<>H		4D88	158	M=C ALL	
48	D<>H		4D89	0CE	C=B ALL	
49	D<>H		4D8A	***363***	***JNC -14***	
1	MKEYS	**MKEYS_**	**4D8B**	*3BD*	*?NC XQ*	*Message Line*
2	MKEYS		4D8C	*01C*	*->07EF*	*[MESSL]*

```
3   MKEYS                                4D8D  02D  "-"
4   MKEYS                                4D8E  00B  "K"
5   Expects Make/Delete in A (1/0)       4D8F  005  "E"          "-KEYS 0"
6   Leading chr expected in LCD          4D90  019  "Y"
7   and LCD enabled.  X has "trigger".   4D91  013  "S"
8   MKEYS                                4D92  020  " "
9   MKEYS                                4D93  20F  "O"
10  MKEYS                                4D94  34E  ?A#0 ALL
11  MKEYS                                4D95  033  JNC  +06
12  MKEYS                                4D96  104  CLRF 8        make assignment
13  MKEYS                                4D97  3BD  ?NC XQ        Message Line
14  MKEYS                                4D98  01C  ->07EF        [MESSL]
15  MKEYS                                4D99  20E  "N"
16  MKEYS                                4D9A  033  JNC +06
17  MKEYS                                4D9B  108  SETF 8        remove assignments
18  MKEYS                                4D9C  3BD  ?NC XQ        Message Line
19  MKEYS                                4D9D  01C  ->07EF        [MESSL]
20  MKEYS                                4D9E  006  "F"
21  MKEYS                                4D9F  206  "F"
22  MKEYS                                4DA0  039  ?NC GO        Display not halting
23  MKEYS                                4DA1  102  ->400E        [APERX0]
24  MKEYS                                4DA2  06E  A<>B ALL
25  MKEYS                                4DA3  198  C=M  ALL
26  MKEYS                                4DA4  01C  PT= 3
27  MKEYS                                4DA5  36A  ?A#C PT<-
28  MKEYS                                4DA6  0E7  JC +28d
29  MKEYS                                4DA7  0B0  C=N  ALL
30  MKEYS                                4DA8  0AE  A<>C ALL
31  MKEYS                                4DA9  1FD  ?NC XQ
32  MKEYS                                4DAA  0BC  ->2F7F        [TBITMA]
33  MKEYS                                4DAB  0C3  JNC +24d
34  MKEYS                                4DAC  309  ?NC XQ
35  MKEYS                                4DAD  09C  ->27C2        [ASN15]
36  MKEYS                                4DAE  0A3  JNC +20d
37  MKEYS                                4DAF  2EE  ?C#0 ALL
38  MKEYS                                4DB0  093  JNC +18d
39  MKEYS                                4DB1  0B0  C=N  ALL
40  MKEYS                                4DB2  0AE  A<>C ALL
41  MKEYS                                4DB3  304  CLRF 1
42  MKEYS                                4DB4  201  ?NC XQ
43  MKEYS                                4DB5  0AC  ->2B80        [GCPKC]
44  MKEYS                                4DB6  00C  ?FSET 3
45  MKEYS                                4DB7  05F  JC  +11d
46  MKEYS                                4DB8  158  M=C  ALL
47  MKEYS                                4DB9  149  ?NC XQ        Enable Chip0
48  MKEYS                                4DBA  024  ->0952        [ENCP00]
49  MKEYS                                4DBB  278  READ 9(Q)
50  MKEYS                                4DBC  330  FETCH S&X
51  MKEYS                                4DBD  070  N=C  ALL
52  MKEYS                                4DBE  23A  C=C+1 M
53  MKEYS                                4DBF  330  FETCH  S&X
54  MKEYS                                4DC0  0EE  B<>C ALL
55  MKEYS                                4DC1  1FB  JNC  +63
56  MKEYS                                4DC2  19B  JNC  +51
57  MKEYS                                4DC3  143  JNC +40
58  MKEYS                                4DC4  21C  PT= 2
59  MKEYS                                4DC5  110  LD@PT-  4
60  MKEYS                                4DC6  06E  A<>B  ALL
61  MKEYS                                4DC7  156  A=A+C  XS
62  MKEYS                                4DC8  06E  A<>B  ALL
63  MKEYS                                4DC9  130  LDI S&X
64  MKEYS                                4DCA  00A  CON: 10
65  MKEYS                                4DCB  1BC  RCR  11
66  MKEYS                                4DCC  0FA  B<>C  M
67  MKEYS                                4DCD  10C  ?FSET 8
68  MKEYS                                4DCE  2A7  JC -44
```



69	MKEYS		4DCF	*201*	*?NC XQ*	
70	MKEYS		4DD0	*0AC*	*->2B80*	*[GCPKC]*
71	MKEYS		4DD1	00C	?FSET 3	
72	MKEYS		4DD2	***11F***	***JC +35d***	
73	MKEYS		4DD3	04E	C=0 ALL	
74	MKEYS		4DD4	0CA	C=B PT<-	
75	MKEYS		4DD5	0FC	RCR 10	
76	MKEYS		4DD6	0AA	A<>C PT<-	
77	MKEYS		4DD7	23C	RCR 2	
78	MKEYS		4DD8	27E	C=C-1 MS	
79	MKEYS		4DD9	0EE	B<>C ALL	
80	MKEYS		4DDA	*31D*	*?NC XQ*	
81	MKEYS		4DDB	*0A0*	*->28C7*	*[AVAILA]*
82	MKEYS		4DDC	2EE	?C#0 ALL	
83	MKEYS		4DDD	***27F***	***JC -49d***	
84	MKEYS		4DDE	270	RAMSLCT	
85	MKEYS		4DDF	06E	A<>B ALL	
86	MKEYS		4DE0	*285*	*?NC XQ*	
87	MKEYS		4DE1	*050*	*->14A1*	*[TSTMAP]*
88	MKEYS		4DE2	*009*	*?NC GO*	
89	MKEYS		4DE3	*082*	*->2002*	*[PACKE]*
90	MKEYS		4DE4	0E6	B<>C S&X	
91	MKEYS		4DE5	*1FD*	*?NC XQ*	
92	MKEYS		4DE6	*0BC*	*->2F7F*	*[TBITMA]*
93	MKEYS		4DE7	***10C***	***?FSET 8***	
94	MKEYS		4DE8	***23F***	***JC -57d***	
95	MKEYS		4DE9	2EE	?C#0 ALL	
96	MKEYS		4DEA	***0A3***	***JNC +20***	
97	MKEYS		4DEB	*295*	*?NC XQ*	
98	MKEYS		4DEC	*0BC*	*->2FA5*	*[SRBMAP]*
99	MKEYS		4DED	308	SETF 1	
100	MKEYS		4DEE	0B0	C=N ALL	
101	MKEYS		4DEF	0AE	A<>C ALL	
102	MKEYS		4DF0	*201*	*?NC XQ*	
103	MKEYS		4DF1	*0AC*	*->2B80*	*[GCPKC]*
104	MKEYS		4DF2	278	READ 9(Q)	
105	MKEYS		4DF3	***10C***	***?FSET 8***	
106	MKEYS		4DF4	***0F3***	***JNC +30d***	
107	MKEYS		4DF5	*149*	*?NC XQ*	
108	MKEYS		4DF6	*024*	*->0952*	*[ENCP00]*
109	MKEYS		4DF7	278	READ 9(Q)	
110	MKEYS		4DF8	23A	C=C+1 M	
111	MKEYS		4DF9	23A	C=C+1 M	
112	MKEYS		4DFA	***0BB***	***JNC +23d***	
113	MKEYS		4DFB	***24B***	***JNC -55d***	
114	MKEYS		4DFC	***343***	***JNC -24d***	
115	MKEYS		4DFD	***283***	***JNC -48d***	
116	MKEYS		4DFE	*295*	*?NC XQ*	
117	MKEYS		4DFF	*0BC*	*->2FA5*	*[SRBMAP]*
118	MKEYS		4E00	0B0	C=N ALL	
119	MKEYS		4E01	05E	C=0 MS	
120	MKEYS		4E02	05A	C=0 M	
121	MKEYS		4E03	37C	RCR 12	
122	MKEYS		4E04	10E	A=C ALL	
123	MKEYS		4E05	0B0	C=N ALL	
124	MKEYS		4E06	276	C=C-1 XS	
125	MKEYS		4E07	276	C=C-1 XS	
126	MKEYS		4E08	***20F***	***JC -63d***	
127	MKEYS		4E09	2F6	?C#0 XS	
128	MKEYS		4E0A	***38F***	***JC -15d***	
129	MKEYS		4E0B	03A	B=0 M	
130	MKEYS		4E0C	21C	PT= 2	
131	MKEYS		4E0D	110	LD@PT- 4	
132	MKEYS		4E0E	0F6	B<>C XS	
133	MKEYS		4E0F	***373***	***JNC -18d***	
134	MKEYS	**MKEYS**	**4E10**	**1B0**	**POPADR**	***beginning of the table***
135	MKEYS		4E11	***268***	***WRIT 9(Q)***	*save it in Q*

136	MKEYS		4E12	330	FETCH S&X	*get key code*
137	MKEYS		4E13	10E	A=C ALL	
138	MKEYS		4E14	070	N=C ALL	
139	MKEYS		4E15	23A	C=C+1 M	
140	MKEYS		4E16	330	FETCH S&X	*fnc cde*
141	MKEYS		4E17	356	?A#0 XS	*end of table?*
142	MKEYS		4E18	327	JC -28d	*no, loop back*
143	MKEYS		4E19	10C	?FSET 8	*removing assignments?*
144	MKEYS		4E1A	04B	JNC +09	*no, skip packing*
145	MKEYS		4E1B	001	?NC XQ	*yes ,go ahead and pack*
146	MKEYS		4E1C	080	->2000	*[PACKN]*
147	MKEYS		4E1D	239	?NC XQ	*Reset Status Bits*
148	MKEYS		4E1E	00C	->038E	*[RSTMS0]*
149	MKEYS		4E1F	3C1	?NC XQ	*Enable & Clear Disp*
150	MKEYS		4E20	0B0	->2CF0	*[CLLCDE]*
151	MKEYS		4E21	149	?NC XQ	*Enable Chip 0*
152	MKEYS		4E22	024	->0952	*[ENCP00]*
153	MKEYS		4E23	3C1	?NC GO	*Normal Function Return*
154	MKEYS		4E24	002	->00F0	*[NFRPU]*
1	HEPAXA4	HEPAXA4	4E25	184	CLRF 11	*clrf 11 - PowerCL*
2	HEPAXA4		4E26	09C	PT= 5	
3	HEPAXA4		4E27	023	JNC +04	
4	HEPAXA4	SPFCA4	4E28	188	SETF 11	*setf 11 - all others*
5	HEPAXA4		4E29	09C	PT= 5	
6	HEPAXA4		4E2A	210	LD@PT- 8	*point at 3rd. Quad*
7	HEPAXA4		4E2B	04A	C=0 PT<-	*clear C[5:0] / C[4:0]*
8			4E2C	070	N=C ALL	*save pg# in N*
9	*this will happen regardless of*		4E2D	04C	?FSET 4	*SSTing a program?*
10	*if using the Alpha or the Numeric*		4E2E	01F	JC +03	*yes, [SSTPRG]*
11	*function (!!) - flag 8 status*		4E2F	2CC	?FSET 13	*RUNNING a program?*
12			4E30	193	JNC +50	*no, skip over [NOPRGM]*
13	HEPAXA4	SSTPRG	4E31	108	SETF 8	*yes, flag "first byte" condition*
14	HEPAXA4		4E32	03A	B=0 M	*reset mantissa in B*
15	HEPAXA4		4E33	149	?NC XQ	*Get PC adr to A<3:0>*
16	HEPAXA4		4E34	0A4	->2952	*[GETPCA]*
17	HEPAXA4	NXTLINE	4E35	01D	?NC XQ	*get next byte to C(1:0)*
18			4E36	0B4	->2D07	*[NXTBYT]*
19	*so effectively there's no need*		4E37	056	C=0 XS	*clear XS nybble*
20	*for a "numeric input" function,*		4E38	2E6	?C#0 S&X	*is it all zeros? (NULL)*
21	*unless such is a desired way*		4E39	0A7	JC +20d	*no,-> LB_A062*
22	*to use it in RUN mode.*		4E3A	10C	?FSET 8	*first byte?*
23			4E3B	3D7	JC -06	*yes, LB_A048*
24	HEPAXA4	GETVAL	4E3C	31D	?NC XQ	*Decrease Address RAM/ROM*
25	HEPAXA4		4E3D	0A4	->29C7	*[DECAD]*
26	HEPAXA4		4E3E	0DD	?NC XQ	*Put away PRGM counter*
27	HEPAXA4		4E3F	08C	->2337	*[PUTPC]*
28	HEPAXA4		4E40	0DA	C=B M	*get PC to C[M]*
29	HEPAXA4		4E41	03C	RCR 3	*rotate to S&X field*
30	HEPAXA4		4E42	3E1	?NC XQ	*get parameter value*
31	HEPAXA4		4E43	008	->02F8	*[GOTINT]*
32	HEPAXA4		4E44	106	A=C S&X	*save it in A[S&X]*
33	HEPAXA4		4E45	3F8	READ 15(e)	*get prgm line number*
34	HEPAXA4		4E46	226	C=C+1 S&X	*increase program line*
35	HEPAXA4		4E47	013	JNC +02	*LB_A05C*
36	HEPAXA4		4E48	266	C=C-1 S&X	*was the last, undo!*
37	HEPAXA4	LB_A05C	4E49	3E8	WRIT 15(e)	*set new line number*
38	HEPAXA4		4E4A	3B8	READ 14(d)	*read flags*
39	HEPAXA4		4E4B	358	ST=C XP	*get UF(55-48) in ST*
40	HEPAXA4		4E4C	0B3	JNC +22d	*[NOPRGM]*
41	HEPAXA4	LB_A062	4E4D	06E	A<>B ALL	*save adr in B*
42			4E4E	106	A=C S&X	*get byte value to A[S&X]*
43	*for a byte to be a valid number*		4E4F	130	LDI S&X	*010*
44	*the second digit must be a "1"*		4E50	010	CON:	*ten's mask*
45	*and the first digit < A*		4E51	1C6	A=A-C S&X	*remove mask*
46			4E52	02F	JC +05	*if A(1)=0 -> LB_A06C*
47	HEPAXA4		4E53	130	LDI S&X	*00A*
48	HEPAXA4		4E54	00A	CON:	*units limit*

49	HEPAXA4		4E55	306	?A<C S&X	*is it a number?*
50	HEPAXA4		4E56	**037**	**JC +06**	*yes! -> LB_A071*
51	HEPAXA4	LB_A06C	4E57	06E	A<>B ALL	*restore MM adr in A<1:3>*
52	HEPAXA4		4E58	**10C**	**?FSET 8**	*first digit?*
53	HEPAXA4		4E59	**31B**	**JNC -29d**	*no -> [GETVAL]*
54	HEPAXA4		4E5A	0B5	?NC GO	
55	HEPAXA4		4E5B	0A2	->282D	*[ERRDE]*
56	HEPAXA4	LB_A071	4E5C	**104**	**CLRF 8**	*flags "last digit" condition*
57	HEPAXA4		4E5D	3E6	LSHFA S&X	
58	HEPAXA4		4E5E	3E6	LSHFA S&X	
59	HEPAXA4		4E5F	3EE	LSHFA ALL	
60	HEPAXA4		4E60	06E	A<>B ALL	
61	HEPAXA4		4E61	**2A3**	**JNC -44d**	*LB_A048*
62	HEPAXA4	NOPRGM	4E62	**0B0**	**C=N ALL**	***get pg# back (!)***
63	HEPAXA4		4E63	23A	C=C+1 M	***"p001000" / "p801000"***
64	HEPAXA4	***"p8001fns"***	4E64	330	FETCH S&X	*get # of functions as counter*
65	HEPAXA4		4E65	**10C**	**?FSET 8**	*ALPHA input?*
66	HEPAXA4		4E66	**1A3**	**JNC +52d**	*no, numeric instead -> LB_A0B9*
67	HEPAXA4		4E67	10E	A=C ALL	*yes, prepare to read FNAME*
68	HEPAXA4		4E68	**278**	**READ 9(Q)**	*placed there in the prompt*
69	HEPAXA4		4E69	**158**	**M=C ALL**	*copy FNAME to M*
70	HEPAXA4		4E6A	0AE	A<>C ALL	***"p8001fns"***
71	HEPAXA4		4E6B	10E	A=C ALL	*C=A ALL*
72	HEPAXA4		4E6C	1A6	A=A-1 S&X	*last FNAME character? (!)*
73	HEPAXA4		4E6D	**147**	**JC +40d**	*yes, exit loop -> [NOTFND]*
74	HEPAXA4	***"p8002(fns-1)"***	4E6E	23A	C=C+1 M	*no, increase adr*
75	HEPAXA4		4E6F	315	?NC XQ	*Get adr from FAT entry*
76	HEPAXA4	***"p8003(fns-1)"***	4E70	13C	->4FC5	*[GETADR5] - leaves F11 alone*
77	HEPAXA4		4E71	0AE	A<>C ALL	*target address in A[ADR]*
78	HEPAXA4		4E72	**070**	**N=C ALL**	*save it in N*
79	HEPAXA4		4E73	**198**	**C=M ALL**	*recall FNAME*
80	HEPAXA4		4E74	0AE	A<>C ALL	*store FNAME in A*
81	HEPAXA4		4E75	31C	PT= 1	*"pcba" is in C[M]*
82	HEPAXA4		4E76	27A	C=C-1 M	*one byte upper*
83	HEPAXA4		4E77	330	FETCH S&X	*read chr$*
84	HEPAXA4		4E78	358	ST=C XP	*put chr to sts*
85	HEPAXA4		4E79	284	CLRF 7	*unmask the "080"*
86	HEPAXA4		4E7A	3D8	C<>ST XP	*put unmsk'd back in C*
87	HEPAXA4		4E7B	36A	?A#C PT<-	*chr$ different from FNAME?*
88	HEPAXA4		4E7C	**03F**	**JC +07**	*yes, not the one -> LB_A0A5*
89	HEPAXA4		4E7D	38E	RSHFA ALL	*no, shift A right one nibble*
90	HEPAXA4		4E7E	38E	RSHFA ALL	*and shitf again, two nibbles*
91	HEPAXA4		4E7F	28C	?FSET 7	*was it last character? (!)*
92	HEPAXA4		4E80	**3B3**	**JNC -10d**	*no, get next -> LB_A098*
93	HEPAXA4		4E81	34E	?A#0 ALL	*yes, were we in the last one?*
94	HEPAXA4		4E82	**02B**	**JNC +05**	*yes, jump to LB_A0A9*
95	HEPAXA4	LB_A0A5	4E83	0B0	C=N ALL	*no, restore N*
96	HEPAXA4		4E84	0AE	A<>C ALL	*save it in A*
97	HEPAXA4		4E85	0DA	C=B M	*recall B*
98	HEPAXA4		4E86	**333**	**JNC -26d**	*next function -> LB_A085*
99	HEPAXA4	LB_A0A9	4E87	09C	PT= 5	
100	HEPAXA4		4E88	**18C**	**?FSET 11**	*is it Smath44?*
101	HEPAXA4		4E89	013	JNC +02	*no, skip offset*
102	HEPAXA4		4E8A	210	LD@PT- 8	*yes, point at 3rd. Quad*
103	HEPAXA4		4E8B	04A	C=0 PT<-	*clear C[5:0] / C[4:0]*
104	HEPAXA4		4E8C	23A	C=C+1 M	***"p001" / "p801"***
105	HEPAXA4		4E8D	330	FETCH S&X	*get # of functions as counter*
106	HEPAXA4		4E8E	10E	A=C ALL	
107	HEPAXA4		4E8F	3B8	READ 14(d)	*read status bits*
108	HEPAXA4		4E90	358	ST=C XP	*will not work w/ sub-fncts*
109	HEPAXA4		4E91	0B0	C=N ALL	
110	HEPAXA4		4E92	1C6	A=A-C S&X	
111	HEPAXA4		4E93	1A6	A=A-1 S&X	
112	HEPAXA4		4E94	**04B**	**JNC +09**	*LB_A0BC*
113	HEPAXA4	NOTFND	4E95	044	CLRF 4	*reset SST' flag for repeat XQ2*
114	HEPAXA4		4E96	**3A0**	**?NC RTN**	***no, return to B3 / B4 (Not found)***
115	HEPAXA4		4E97	188	SETF 11	*restore defaults*

116	HEPAXA4		4E98	*381*	*?NC GO*	*yes, show "NONEXISTENT"*
117	HEPAXA4		4E99	*00A*	*->02E0*	*[ERRNE]*
118	HEPAXA4	LB_A0B9	4E9A	306	?A<C S&X	*did we exceed # of fns?*
119	HEPAXA4		4E9B	***3E3***	***JNC -04***	
120	HEPAXA4		4E9C	11A	A=C M	
121	HEPAXA4	LB_A0BC	4E9D	***020***	***XQ>GO***	***defuse the return (!)***
122	HEPAXA4		4E9E	***00C***	***?FSET 3***	***is PRGM ON?***
123	HEPAXA4		4E9F	***0A7***	***JC +20d***	*yes, divert to LB_A0D6*
124	HEPAXA4		4EA0	0A6	A<>C S&X	*no, calculate fnc adr*
125	HEPAXA4		4EA1	106	A=C S&X	
126	HEPAXA4		4EA2	226	C=C+1 S&X	*index+1*
127	HEPAXA4		4EA3	1E6	C=C+C S&X	*2 x (index+1)*
128	HEPAXA4		4EA4	1BC	RCR 11	
129	HEPAXA4		4EA5	15C	PT= 6	
130	HEPAXA4		4EA6	0A2	A<>C @PT	*copy pg# to C(6)*
131	HEPAXA4		4EA7	102	A=C @PT	*keep it in A*
132	HEPAXA4		4EA8	***18C***	***?FSET 11***	*is it Smath44?*
133	HEPAXA4		4EA9	*023*	*JNC +04*	*no, skip offset*
134	HEPAXA4		4EAA	09C	PT= 5	*yes, set pointer*
135	HEPAXA4		4EAB	210	LD@PT- 8	*point at 3rd. Quad*
136	HEPAXA4		4EAC	15C	PT= 6	*reset pointer*
137	HEPAXA4		4EAD	188	SETF 11	*restore defaults (!)*
138			4EAE	*315*	*?NC XQ*	*Get adr from FAT entry*
139	*work out the location for*		4EAF	*13C*	*->4FC5*	*[GETADR5] - leaves F11 alone*
140	*the function code. Requires*		4EB0	0A2	A<>C @PT	*copy pg# to C(6)*
141	*the bank to be active, as it*		4EB1	102	A=C @PT	*keep it in A*
142	*just goes there...*		4EB2	***1E0***	***GOTOADR***	***go to function (!)***
143			4EB3	05E	C=0 MS	
144	HEPAXA4		4EB4	130	LDI S&X	*00A*
145	HEPAXA4		4EB5	00A	CON:	
146	HEPAXA4		4EB6	306	?A<C S&X	
147	HEPAXA4		4EB7	***027***	***JC +04***	*LB_A0DE*
148	HEPAXA4		4EB8	1C6	A=A-C S&X	
149	HEPAXA4		4EB9	23E	C=C+1 MS	
150	HEPAXA4		4EBA	***3E3***	***JNC -04***	*LB_A0D9*
151	HEPAXA4		4EBB	05A	C=0 M	
152	HEPAXA4		4EBC	130	LDI S&X	*101*
153	HEPAXA4		4EBD	101	CON:	
154	HEPAXA4		4EBE	33C	RCR 1	
155	HEPAXA4		4EBF	39C	PT= 0	
156	HEPAXA4		4EC0	0A2	A<>C @PT	
157	HEPAXA4		4EC1	00E	A=0 ALL	*reset A*
158	HEPAXA4		4EC2	35C	PT= 12	
159	HEPAXA4		4EC3	2E2	?C#0 @PT	
160	HEPAXA4		4EC4	*013*	*JNC +02*	*LB_A0E9*
161	HEPAXA4		4EC5	37C	RCR 12	
162	HEPAXA4		4EC6	01C	PT= 3	
163			4EC7	10A	A=C PT<--	*A(3:0) = C(3:0)*
164	*As an alternative to make it*		4EC8	0AE	A<>C ALL	*put it in C*
165	*more general-purpose:*		4EC9	0FC	RCR 10	*move it to C(7:4")*
166	*Store the function code in P (!)*		4ECA	***18C***	***?FSET 11***	*is it PowerCL?*
167	*and recall it right at this point*		4ECB	***03B***	***JNC +07***	yes, skip over
168			4ECC	10E	A=C ALL	*save in A again*
169	HEPAXA4	SMTH / VECT	4ECD	***238***	***READ 8(P)***	*has XROM id# in C(13:11)*
170	HEPAXA4		4ECE	0FC	RCR 10	*move to C<3:0>*
171	HEPAXA4		4ECF	10A	A=C PT<--	*A(3:0) = C(3:0)*
172	HEPAXA4		4ED0	0AE	A<>C ALL	*put it in C*
173	HEPAXA4		4ED1	***053***	***JNC +10d***	
174	HEPAXA4	PWRCL	4ED2	0D0	LD@PT- 3	
175	HEPAXA4		4ED3	190	LD@PT- 6	***XROM A3:36 = "XQ1" function***
176	HEPAXA4		4ED4	***24C***	***?FSET 9***	*is it SECOND pass (XEQ1)?*
177	HEPAXA4		4ED5	***01F***	***JC +03***	yes, go on
178	HEPAXA4		4ED6	21C	PT= 2	*no, change to XEQ2*
179	HEPAXA4		4ED7	1D0	LD@PT- 7	***XROM A3:37 = "XQ2" function***
180	HEPAXA4		4ED8	290	LD@PT- A	*part of Power_CL*
181	HEPAXA4		4ED9	0D0	LD@PT- 3	
182	HEPAXA4		4EDA	188	SETF 11	*restore defaults*

183	HEPAXA4	XEQSPF	4EDB	268	WRIT 9(Q)	writes fnc code in Q
184	HEPAXA4		4EDC	2C9	?NC XQ	
185	HEPAXA4		4EDD	08C	->23B2	[INSSUB]
186	HEPAXA4		4EDE	1CD	?NC XQ	
187	HEPAXA4		4EDF	0A8	->2A73	[INSTR]
188	HEPAXA4		4EE0	38D	?NC XQ	
189	HEPAXA4		4EE1	0A4	->29E3	[INBYT0]
190	HEPAXA4		4EE2	181	?NC GO	execution terminates
191	HEPAXA4		4EE3	08A	->2260	[XSST]
1	FCAT	FCAT4	4EE4	09C	PT= 5	Sub-Function CATalog
2	FCAT	FCAT8	4EE5	04A	C=0 PT<-	clear C[5:0]
3			4EE6	23A	C=C+1 M	"p001"
4	The same routine is used for		4EE7	158	M=C ALL	save FAT adr in M
5	FATS in Banks 3 and 4 (!)		4EE8	188	SETF 11	all functions, not just headers
6	Full-featured implementation !		4EE9	244	CLRF 9	clear single step flag
7			4EEA	108	SETF 8	going forwards
8	FCAT	ENTRY	4EEB	3C1	?NC XQ	clear & enable LCD
9			4EEC	0B0	->2CF0	[CLLCDE]
10	obviously the corresponding		4EED	198	C=M ALL	current FAT entry
11	block has been switched		4EEE	10C	?FSET 8	going forwards?
12			4EEF	09F	JC +19d	Yes, ignore this
13	FCAT	PREVNTR	4EF0	27A	C=C-1 M	no, backup one entry
14	FCAT		4EF1	27A	C=C-1 M	decrease adr to w2
15	FCAT		4EF2	27A	C=C-1 M	decrease adr to w1
16	FCAT		4EF3	330	FETCH S&X	read word1: "00a"
17	FCAT		4EF4	2F6	?C#0 XS	reached the top end?
18	FCAT		4EF5	07B	JNC +15d	no, merge cases
19	FCAT	BAILOUT	4EF6	24C	?FSET 9	SST mode?
20	FCAT		4EF7	027	JC +04	yes,do not abort
21			4EF8	188	SETF 11	no, restore defaults
22	caveat: doesn't re-enable		4EF9	161	?NC GO	Exit w/out NFRTN
23	the main blck, B1		4EFA	126	->4958	[EXIT3]
24			4EFB	18C	?FSET 11	headers only?
25	FCAT		4EFC	01F	JC +03	no, skip
26	FCAT		4EFD	0B0	C=N ALL	yes, recall last valid entry
27	FCAT		4EFE	02B	JNC +05	continue
28	FCAT	HEADS	4EFF	10C	?FSET 8	going forwards?
29	FCAT		4F00	387	JC -16d	yes, previous entry (-3 lines)
30	FCAT		4F01	23A	C=C+1 M	
31	FCAT	FORTH	4F02	23A	C=C+1 M	point at w1 in FAT
32	FCAT	VALID	4F03	330	FETCH S&X	read w1: "00a"
33	FCAT	MERGE	4F04	37C	RCR 12	
34	FCAT		4F05	0F6	B<>C XS	puts "a" in B[XS]
35	FCAT		4F06	23C	RCR 2	rotate back in place
36	FCAT		4F07	23A	C=C+1 M	increase adr to w2
37	FCAT		4F08	330	FETCH S&X	read w2: "0bc"
38	FCAT		4F09	2F6	?C#0 XS	reached the bottom end?
39	FCAT		4F0A	367	JC -20d	yes, bail out!
40	FCAT		4F0B	158	M=C ALL	no, update FAT adr
41	FCAT		4F0C	32D	?NC XQ	Format adr
42	FCAT		4F0D	13C	->4FCB	[FMTADR]
43	FCAT		4F0E	0EE	C<>B ALL	save it in B for either case
44	FCAT		4F0F	059	?NC XQ	Add Fname to display
45	FCAT		4F10	13C	->4F16	[FNAME]
46	FCAT		4F11	201	?NC XQ	Display and Print in Norm/Trace
47	FCAT		4F12	070	->1C80	[MSG105]
48	FCAT		4F13	0F1	?NC XQ	offer CAT choices
49	FCAT		4F14	13C	->4F3C	[CATKYS]
50	FCAT		4F15	2B3	JNC -42d	
51	FCAT	FNAME	4F16	130	LDI S&X	
52	FCAT		4F17	02D	"-"	group lead chr#
53	FCAT		4F18	0A6	A<>C S&X	pre-load for comparisons
54	FCAT	FNAME4	4F19	0CE	C=B ALL	Get FNC address in C[ADR]
55			4F1A	27A	C=C-1 M	one byte upper
56	expects fnc adr in B[ADR]		4F1B	330	FETCH S&X	read chr#
57			4F1C	18C	?FSET 11	headers only?

58	FCAT		4F1D	05F	JC +11d	no, skip
59	FCAT		4F1E	366	?A#C S&X	yes, header chr#?
60			4F1F	0BF	JC +23d	no, ignore entry!!
61	save this valid adr in N		4F20	1D8	C<>M ALL	last shown FAT entry
62	it'll be needed in SST mode		4F21	27A	C=C-1 M	w1 pointer
63	when going forwards		4F22	070	N=C ALL	save for later use
64	and headers-only is active		4F23	23A	C=C+1 M	w2 pointer
65			4F24	1D8	C<>M ALL	restore chr# adr
66	FCAT		4F25	01B	JNC +03	
67	FCAT		4F26	27A	C=C-1 M	
68	FCAT		4F27	330	FETCH S&X	read chr#
69	FCAT		4F28	358	ST=C XP	yes, valid entry follows
70	FCAT		4F29	284	CLRF 7	unmask the "0C0"
71	FCAT		4F2A	144	CLRF 6	unmask the "040"
72	FCAT		4F2B	3D8	C<>ST XP	
73	FCAT		4F2C	3E8	WRIT 15(e)	write to display
74	FCAT		4F2D	28C	?FSET 7	was it last character? (!)
75	FCAT		4F2E	3C3	JNC -08	no, get next chr#
76	FCAT		4F2F	3DD	?NC GO	Left Justified format
77	FCAT		4F30	0AE	->2BF7	[LEFTJ]
78	FCAT	ENTER^	4F31	24C	?FSET 9	SST mode?
79	FCAT		4F32	3A0	?NC RTN	no, ignore
80	FCAT		4F33	18C	?FSET 11	headers only?
81	FCAT		4F34	02B	JNC +05	yes, cancel it!
82	FCAT		4F35	184	CLRF 11	no, set headers-only flag!
83	FCAT	IGNORE	4F36	1B0	POPADR	defuse the return, and
84	FCAT		4F37	2F3	JNC -34d	short-cut to next entry
85	FCAT	ADIOS	4F38	1B0	POPADR	don't you come back :)
86	FCAT	SOLONG	4F39	188	SETF 11	restore defaults
87	FCAT		4F3A	244	CLRF 9	clear single step flag
88	FCAT		4F3B	3E0	RTN	return to main flow
89	FCAT	CATKYS	4F3C	261	?NC XQ	debounce and reset keyboard
90	FCAT		4F3D	000	->0098	[RSTKB]
91	FCAT		4F3E	130	LDI S&X	
92			4F3F	2FF	CON:	wait & show for about 140ms
93	CLV4 1x	CLV4 removed	4F40	3C8	CLRKEY	
94	CLV4 1x		4F41	3CC	?KEY	key pressed?
95			4F42	037	JC +06	yes, process it
96	FCAT		4F43	266	C=C-1 S&X	no, keep the countdown
97	FCAT		4F44	3E3	JNC -04	(WaitK)
98	FCAT		4F45	24C	?FSET 9	SST mode?
99	FCAT		4F46	3C7	JC -08	yes, reset counter (WaitSST)
100	FCAT		4F47	3A0	?NC RTN	no, return if no key was pressed
101	FCAT		4F48	130	LDI S&X	
102	FCAT		4F49	007	CON:	# of entries
103	FCAT		4F4A	001	?NC XQ	
104	FCAT		4F4B	0EC	->3B00	[KEYFNC]
105	FCAT		4F4C	0C3	KeyCode: C3	"BackArrow"
106	FCAT		4F4D	0C2	KeyCode: C2	"SST"
107	FCAT		4F4E	032	KeyCode: 32	"XEQ" (Execute)
108	FCAT		4F4F	013	Keycode: 13	"ENTER^" (Headers only)
109	FCAT		4F50	083	KeyCode: 83	"EEX" (Print)
110	FCAT		4F51	012	KeyCode: 12	"SHIFT"
111	FCAT		4F52	087	KeyCode: 87	"R/S"
112	FCAT		4F53	000	End of Table	
113	FCAT	BCKARW	4F54	323	JNC -28d	abandon ship
114	FCAT	SST	4F55	3E0	RTN	return to enumeration
115			4F56	123	JNC +36d	[Execute]
116	dispatcher section		4F57	2D3	JNC -38d	Headers only
117	all structured nice & tidy		4F58	0D3	JNC +26d	print display
118			4F59	05B	JNC +11d	[KeySHIFT]
119	FCAT		4F5A	013	JNC +02	[keyRS]
120	FCAT		4F5B	30B	JNC -31d	any other key
121	FCAT	R/S	4F5C	261	?NC XQ	debounce keyboard
122	FCAT		4F5D	000	->0098	[RSTKB]
123	FCAT		4F5E	24C	?FSET 9	SST mode?

124	FCAT		4F5F	01B	JNC +03	
125	FCAT		4F60	244	CLRF 9	yes -> clear it and return
126	FCAT		4F61	3D3	JNC -06	STEP
127	FCAT		4F62	248	SETF 9	set SST flag
128	FCAT		4F63	3F3	JNC -02	STEP
129	FCAT	SHIFT	4F64	24C	?FSET 9	SST mode?
130	FCAT		4F65	3A0	?NC RTN	no, ignore
131	FCAT		4F66	395	?NCXQ	Toggles UF26 - enables Chip0
132	FCAT		4F67	07C	->1FE5	[TOGSHF]
133	FCAT		4F68	171	?NC XQ	Update Ann's
134	FCAT		4F69	01C	->075C	[ANNOUT]
135	FCAT		4F6A	3D9	?NC XQ	Enable but not Clear LCD
136	FCAT		4F6B	01C	->07F6	[ENLCD]
137	FCAT		4F6C	10C	?FSET 8	SHIFT on?
138	FCAT		4F6D	01B	JNC +03	no, skip over
139	FCAT		4F6E	104	CLRF 8	yes, toggle it off
140	FCAT		4F6F	3A3	JNC -12d	
141	FCAT		4F70	108	SETF 8	toggle it ON
142	FCAT		4F71	3F3	JNC -02	
143	FCAT	PRT	4F72	24C	?FSET 9	SST mode?
144	FCAT		4F73	3A0	?NC RTN	no, ignore
145	FCAT		4F74	1B0	POPADR	don't you come back :)
146	FCAT		4F75	261	?NC XQ	debounce keyboard
147	FCAT		4F76	000	->0098	[RSTKB]
148	FCAT		4F77	108	SETF 8	
149	FCAT		4F78	201	?NC GO	
150	FCAT		4F79	072	->1C80	[MSG105]
151	FCAT	XEQ	4F7A	24C	?FSET 9	SST mode?
152	FCAT		4F7B	3A0	?NC RTN	no, ignore
153	FCAT		4F7C	215	?NC XQ	Build Msg - all cases
154			4F7D	0FC	->3F85	[APRMSG2]
155	back door execution,		4F7E	018	"X"	
156	will transfer the PC to the		4F7F	005	"E"	"XEQ ' "
157	function address in page		4F80	011	"Q"	
158			4F81	227	" ' "	single quote
159	FCAT		4F82	188	SETF 11	override headers-only
160	FCAT		4F83	059	?NC XQ	Add Fname to display
161	FCAT		4F84	13C	->4F16	[FNAME]
162	FCAT		4F85	319	?NC XQ	last chance to cancel out
163	FCAT		4F86	038	->0EC6	[NULTST]
164			4F87	1B0	POPADR	don't you come back :)
165	the calling module has the		4F88	261	?NC XQ	Reset keyboard
166	appropriate bank active,		4F89	000	->0098	[RSTKB]
167	but we don't know where		4F8A	169	?NC XQ	Select Chip0 & Reset SEQ
168			4F8B	124	->495A	[EXIT4]
169	FCAT		4F8C	138	READ 4(L)	recall "carrier" (S&X has offset rg.)
170	FCAT		4F8D	0E6	C<>B S&X	get index code and offset to B
171			4F8E	198	C=M ALL	last FAT adr is still here
172	the last challenge is solved!		4F8F	09C	PT= 5	for the SandMath/Matrix
173	this code will get the Index		4F90	042	C=0 @PT	need to clear "8" in ADR
174	of the selected XEQ'd in		4F91	27A	C=C-1 M	back up to 1st. word
175	the sub-fnc. Catalog, then		4F92	345	?NC XQ	Halve Mantissa
176	will brand it in the LASTF		4F93	13C	->4FD1	[M=M/2]
177	buffer according to the data		4F94	27A	C=C-1 M	minus one = INDEX!
178	in the "carrier" L		4F95	03C	RCR 3	put index# in C<1:0>
179			4F96	0D6	C=B XS	use index code!
180	FCAT		4F97	158	M=C ALL	fnc. Index in S&X
181	FCAT		4F98	036	B=0 XS	clear [XS] nybble
182	FCAT		4F99	3D1	?NC XQ	Store fcn. Id# in LASTF buffer
183	FCAT		4F9A	120	->48F4	[LASTF0] - inputs in B and M
184	FCAT		4F9B	0FA	C<>B M	get go-to ADR to C[M]
185	FCAT		4F9C	1E0	GOTO ADR	go for it!
1		XEQFNC	4F9D	315	?NC XQ	MAIN Function Search
2	There are three chances:		4F9E	098	->26C5	[ASRCH]
3	1. found wirhin FAT2, or		4F9F	2EE	?C#0 ALL	was it found?
4	2. found within FAT1, or		4FA0	381	?NC GO	no, show error
5	3. found as normal function		4FA1	00A	->02E0	[ERRNE]

6	*FNAme is still in Q & M*		4FA2	08C	?FSET 5	mainframe FNC?
7	*bank3 was still active*		4FA3	***3EF***	***JC -03***	yes, give it up
8	*needed to switch to B1 again*		4FA4	07C	RCR 4	*put FNC code in C(3:0)*
9	*upon the 2nd. Return*		4FA5	*029*	*?NC GO*	***Execute function!***
10			4FA6	*01E*	*->070A*	*[RAK70]*
1	APNDCL	**APNDCL**	**4FA7**	130	LDI S&X	
2	APNDCL		4FA8	***03A***	***":"***	*append ":"*
3	APNDCL		4FA9	*3D5*	*?NC GO*	*append to Alpha*
4	APNDCL		4FAA	*07E*	*->1FF5*	*[APND10]*
1	RUNMSG	**RUNMS0**	**4FAB**	**108**	**SETF 8**	*to display the text!*
2	RUNMSG	**RUNMSG**	**4FAC**	**260**	**SETHEX**	
3	RUNMSG		4FAD	*215*	*?NC XQ*	*Build Msg - all cases*
4	RUNMSG		4FAE	*0FC*	*->3F85*	*[APRMSG2]*
5			4FAF	*012*	***"R"***	
6	*CPU must be in HEX mode*		4FB0	*015*	***"U"***	
7			4FB1	*00E*	***"N"***	
8	RUNMSG		4FB2	*00E*	***"N"***	*"RUNNING..."*
9	RUNMSG		4FB3	*009*	***"I"***	
10	RUNMSG		4FB4	*00E*	***"N"***	
11	RUNMSG		4FB5	*047*	***"G."***	
12	RUNMSG		4FB6	*060*	***"."***	
13	RUNMSG		4FB7	*260*	***"."***	
14	RUNMSG		4FB8	*039*	***?NC GO***	*Display not halting - leaves F8 as-is*
15	RUNMSG		4FB9	*102*	*->400E*	*[APERX4]*
1	APND#	**APND#**	**4FBA**	130	LDI S&X	
2	APND#		4FBB	***023***	***"#"***	*append "#"*
3	APND#		4FBC	***36B***	***JNC -19d***	
1	GETADR	**GETADR**	**4FBD**	**3DA**	**RSHFC M**	*move pg# to C<5>*
2			4FBE	3DA	RSHFC M	*move pg# to C<4>*
3	*addr bytes: "A800" + 2x(1+indx#)*		4FBF	3DA	RSHFC M	*move pg# to C<3>*
4	*then read two bytes and*		4FC0	23A	C=C+1 M	*upper page (SandMath)*
5	*generate the address -*		4FC1	1BC	RCR 11	
6	*then read the FNAME from there*		4FC2	09C	PT= 5	
7			4FC3	210	LD@PT- 8	*third quad*
8	GETADR	**GETADR4**	**4FC4**	**188**	**SETF 11**	*default - not "Headers Only"*
9	GETADR	**GETADR5**	**4FC5**	**330**	**FETCH S&X**	***read w1: "00a"***
10			4FC6	37C	RCR 12	*rotate two left*
11	*Gets fnc. adr from FAT entry*		4FC7	0F6	B<>C XS	*puts "a" in B[XS]*
12	*expects "pAD1" in C[ADR]*		4FC8	23C	RCR 2	*rotate back in place*
13	*for the first byte (two in total)*		4FC9	23A	C=C+1 M	*increase adr to w2*
14			4FCA	**330**	**FETCH S&X**	***read w2: "0bc"***
15	GETADR	**FMTADR**	**4FCB**	**0D6**	**C=B XS**	*puts "abc" in C[S&X]*
16	GETADR		4FCC	0FA	B<>C M	*saves ADR2 in B[M]*
17	GETADR		4FCD	1BC	RCR 11	*puts "abc" in C(5:3)*
18	GETADR		4FCE	15C	PT= 6	*put page# in C(6)*
19	GETADR		4FCF	0C2	C=B @PT	*"pabc" is in C[M]*
20	GETADR		4FD0	3E0	RTN	done.
1	M=M/2	**M=M/2**	**4FD1**	**1FA**	**C=C+C M**	
2	M=M/2		4FD2	1FA	C=C+C M	*divide it by 2*
3	M=M/2		4FD3	1FA	C=C+C M	*(oh boy... had I known!)*
4	M=M/2		4FD4	3DA	RSHFC M	
5	M=M/2		4FD5	3E0	RTN	done.
1	KEYPG#	**KEYPG#**	**4FD6**	**066**	**A<>B S&X**	*put page# in A[S&X]*
2	KEYPG#		4FD7	31C	PT= 1	*clean up parameter:*
3			4FD8	342	?A#0 @PT	*from chr# to page#*
4	*converts keycode into*		4FD9	*027*	*JC +04*	
5	*hex value for page#*		4FDA	130	LDI S&X	*A[S&X] goes from 1 to 6*
6			4FDB	009	CON:	*need to add 9 to chr#*
7	KEYPG#		4FDC	146	A=A+C S&X	*it now ranges from A to F*
8	KEYPG#		4FDD	002	A=0 @PT	*clear the "3" digit!*
9	KEYPG#		4FDE	3E0	RTN	
1	TURBO	**TURBO0**	**4FDF**	**04E**	**C=0 ALL**	
2	TURBO		4FE0	05C	PT= 4	*cancels TURBO mode*
3	TURBO		4FE1	210	LD@PT- 8	*(so we see it well)*
4	TURBO		4FE2	*023*	*JNC +04*	
5	TURBO	**TURB50**	**4FE3**	**04E**	**C=0 ALL**	

6	TURBO	TURB51	4FE4	05C	PT= 4	*sets TURBO50*
7	TURBO		4FE5	350	LD@PT- D	
8	TURBO		4FE6	**1FC**	**WCMD**	
9	TURBO		4FE7	3E0	RTN	
1	WAIT4	**WAIT4**	**4FE8**	*37D*	*?NC XQ*	*Slow things down*
2	WAIT4		4FE9	*13C*	*->4FDF*	*[TURBO0]*
3	WAIT4		4FEA	130	LDI S&X	
4	WAIT4		4FEB	200	CON:	initial delay value
5	WAIT4		4FEC	3CC	?KEY	*Key pressed?*
6	WAIT4		4FED	*017*	*JC +02*	*yes, skip addition*
7	WAIT4		4FEE	1E6	C=C+C S&X	*no, delay value is doubled*
8	WAIT4		4FEF	3C8	CLRKEY	*Small delay*
9	WAIT4		4FF0	266	C=C-1 S&X	*loop*
10	WAIT4		4FF1	*3FB*	*JNC -01*	
11	WAIT4		4FF2	***38B***	***JNC -15d***	*run baby, run…*
1	DIVTWO	**DIVTWO**	**4FF3**	**02E**	**B=0 ALL**	*clears B*
2	DIVTWO		4FF4	0FA	B<>C M	*B holds 13-digit mant*
3	DIVTWO		4FF5	0AE	A<>C ALL	*A holds sign and S&X*
4	DIVTWO	**DIVBY2**	**4FF6**	04E	C=0 ALL	
5	DIVTWO		4FF7	35C	PT=12	***build "2" in C***
6	DIVTWO		4FF8	090	LD@PT- 2	
7	DIVTWO		4FF9	*269*	*?NC GO*	
8	DIVTWO		4FFA	*062*	*->189A*	*[DV1-10]*
1	ROMID		4FFB	*034*	***"4"***	
2	ROMID		4FFC	*002*	***"B"***	
3	ROMID		4FFD	*009*	***"I"***	
4	ROMID		4FFE	*00C*	***"L"***	
5	ROMID		4FFF	044	CHKSUM	