

HP VASM Listings

Contained in this package are source listings for proprietary Hewlett-Packard products. This information has been released for distribution on the basis of responsible dissemination by PPC. In order to ensure the continuation of our good-faith arrangement for distribution of this type of documentation it is vital that the user understands that no manufacturer support will be provided for use of these documents.

By opening this package, the user accepts that this information is supplied as is, and understands that all documentation is strictly Not Manufacturer Supported (NOMAS). Hewlett-Packard bears no responsibility for support or correctness of this documentation, and use of the information contained within is at the users risk.

Any questions regarding the enclosed material may be addressed to:

PPC
P.O. Box 9599
Fountain Valley, CA 92728-9599 USA
(714) 754-6226

Please respect the responsibility that acceptance of this information entails. Only through the actions of each of our members can PPC continue to support our members by providing such services as NOMAS listings.

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

Copyright © PPC 1983

* HP41C MAINFRAME MICROCODE ADDRESSES 00-1777

*
4 FILE CN0B
5 ENTRY ALFDEF
6 ENTRY END2
7 ENTRY END3
8 ENTRY CLDTMG
9 ENTRY QUTCAT
10 ENTRY STMSGF
11 ENTRY XAVIEW
12 ENTRY XVIEW
13 ENTRY ADRFCH
14 ENTRY TONSTF
15 ENTRY BCDBIN
16 ENTRY BIGSRC
17 ENTRY INTINT
18 ENTRY COLDST
19 ENTRY DROWSY
20 ENTRY DRSY05
21 ENTRY DRSY25
22 ENTRY DRSY51
23 ENTRY ERRNE
24 ENTRY ERRORF
25 ENTRY GOTINT
26 ENTRY GTAINC
27 ENTRY GTAI40
28 ENTRY INCGT2
29 ENTRY MSGDLY
30 ENTRY NFRC
31 ENTRY NFRNT
32 ENTRY NFRFST
33 ENTRY NFRKB
34 ENTRY NFRKB1
35 ENTRY NFRNC
36 ENTRY NFRNIO
37 ENTRY NFRPR
38 ENTRY NFRPU
39 ENTRY NFRSIG
40 ENTRY NFRX
41 ENTRY NFRXY
42 ENTRY ROW10
43 ENTRY RUNNK
44 ENTRY WKUP70
45 ENTRY X<>ROW
46 ENTRY XCUTB1
47 ENTRY XCUTE
48 ENTRY XCUTEB
49 ENTRY XROW1

*
51 LEGAL
52 0 1 GOLNC LSWKUP
52 1 2
53 2 1 GOLONG DSWKUP
53 3 2

NOMAS
Not Manufacturer Supported
recipient agrees NOT to contact manufacturer

PPC

*THE FOLLOWING ROUTINE TAKES A BINARY REGISTER
 *NUMBER FROM STATUS AND RETURNS THAT REGISTER
 *IN C, ITS ADDRESS IN N, AND X IN M. IT SENDS
 *ILLEGAL ADDRESSES TO ERROR "NONEXISTENT."
 *IT ALSO HANDLES INDIRECT ADDRESS. CYCLE TIMES
 *AVERAGE ABOUT 30 FOR DIRECT AND 60 FOR INDIRECT
 *USES: A,B,C,M,N,ACTIVE POINTER, S9,S8,S7, DADD, + 2 SUB LEVELS
 * MAY EXIT TO ERPNE
 * IN: ADDR IN S7:0 (MAY BE INDIRECT)
 * NO PERIPHERAL ENABLED
 * OUT: C = C(EFFECTIVE ADDR)
 * N = EFFECTIVE ADDR
 * DADD = EFFECTIVE ADDR
 * M = X REGISTER CONTENTS
 * S7 = 0

71	4	ADRFCH	116	C=0	GETX
72	5		1160	DADD=C	
73	6		370	C=REGN 3	
74	7		530	M=C	X IS STORED IN M
75	10		1630	C=ST	GET ADDRESS FROM STATUS
76	11		126	C=0 XS	CLEAR 9 FROM XS
77	12		1204	S7= 0	KILL INDIRECT FLAG
78	13		1730	CST EX	PUT ORIG STATUS BACK
79	14		514	?S6=1	STACK RCL?
80	15		133	GONC	OVRSTK (30) NO
81	16		214	?S5=1	
82	17		113	GONC	OVRSTK (30) NO
83	20		114	?S4=1	
84	21		73	GONC	OVRSTK (30) NO
85	22		1434	PT= 1	SET TO CLEAR HIGH BITS
86	23		102	C=0 PT	CLEAR HIGH BITS
87	24		1160	DADD=C	PUT OUT ADDRESS
88	25		160	N=C	SAVE ADDRESS IN N
89	26		70	C=DATA	GET REGISTER
90	27		113	GOTO	FCHRTN (40) DONE
91	30	OVRSTK	256	AC EX	SAVE RELATIVE ADDRESS
92	31		1570	C=REGN 13	GET STATUS REG
93	32		74	RCR 3	MOVE REG0 TO POSITION
94	33		1006	C=C+A X	COMPUTE ADR OF REG
95	34		160	N=C	SAV ADR FOR CALNG ROUT
96	35		1	GOSUB	CHKADR
96	36		0		
97	37		316	C=B	BRING RCL VALUE TO C
98	40	FCHRTN	1214	?S7=1	INDIRECT
99	41		1640	RTN NC	DONE IF NOT INDIRECT
100	42		1	GOSUB	BCDBIN
100	43		0		DO BCD BIN
101	44		1204	S7= 0	CLEAR INDIRECT FLAG
102	45		1633	GOTO	OVRSTK (30) NEW ADR STRT OVER
103				FILLTO:045	

* THIS ROUTINE IS A SPECIAL "ROW" FOR X<>NN

107	46	X<>ROW	1166	C=C-1 XS	RESTORE BYTE 1
108	47		1166	C=C-1 XS	
109				LEGAL	
110	50		1	GOSUB	INCGT2
110	51		0		
111	52		256	AC EX	

```

112 53      1530 ST=C
113 54      1374 RCR      13
114 55      130 G=C
115 56      363 GOTO      ADRGSB ( 114)

*
117 57 XROW0      1 GOLONG ROW0
117 60           2
118 61 XROW10     1 GOLONG ROW10
118 62           2
119 63 XROW11     1 GOLONG ROW11
119 64           2
120 65 XROW12     1 GOLONG ROW12
120 66           2
121 67 XROW13     1 GOLONG XGTO
121 70           2
122 71 XROW14     1 GOLONG XXEQ
122 72           2
*****
*ONE BYTE STORE AND RECALL FUNCTIONS ENTER HERE TO
*BE TRANSMOGRAPHIED INTO TWO BYTE FUNCTIONS
*****
127      XROW2
128      XROW3
129 73      1142 C=C-1 PT      HIGH DIGIT -2
130 74      1142 C=C-1 PT
131 75      1706 C SR      X      CREATE BYTE TWO
132 76      1712 C SR      WPT      CREATE BYTE 1
133 77      1120 LC      9
134 100     34 PT=      3
135 101     43 GOTO      REGADR ( 105)
*****
*TWO BYTE RCLS, STOS, DSP FORMAT ETC COMPRISE ROW 9
*ROW9 GETS BYTE TWO, FETCHES X (IN M), FETCHES
*REGISTER NN (IN B), AND LEAVE THE ADDRESS OF NN
*IN N FOR ALL DATA RELATED FUNTIONS (0-11). IT THEN
*DOES A SIXTEEN WAY BRANCH TO SORT OUT THE ROW.
*****
143 102 XROW9      1 GOSUB      INCGT2      GET BYTE TWO
143 103           0
144 104           256 AC EX      BRING BACK TO C
145 105 REGADR 1530 ST=C      SAVE BYTE 2
146 106           1374 RCR      13      MOVE TO G POSITION
147 107           130 G=C      SAVE IN G
148 110           742 C=C+C PT      SEP OUT 0-7
149 111           33 GONC      ADRGSB ( 114) DO RCL ETC
150 112           742 C=C+C PT      SEP OUT DSP & TONE
151 113           467 GOC      TONETC ( 161)
152 114 ADRGSB      1 GOSUB      ADRFCH      GET X,RNN, AND ADR
152 115           0
153 116           356 BC EX      SAVE VALUE IN RNN
154 117 BIGBRC 1534 PT=      12      SET FOR G LOAD
155 120           230 C=G      GET BYTE ONE BACK
156 121           34 PT=      3
157 122           1 GOLONG XCUTB1      DO 256 WAY BRANCH
157 123           2
*****
*THIS SECTION SORTS OUT INDIRECT TONE, FIX, ENG, AND
*SCI FROM DIRECT. IF INDIRECT GOES THROUGH ADDRESS
*FETCH OTHERWISE GOES IMMEDIATELY TO 256 WAY BRANCH.
*****

```


163	124	TONSTF	1214	?S7=1		INDIRECT
164	125		1640	RTN NC		NO DO BRANCH
165	126		1204	S7=	0	CLEAR INDIRECT BIT
166	127		1	GOSUB	ADRFCH	GET REG
166	130		0			
167	131		1	GOSUB	BCDBIN	CONVERT TO BINARY
167	132		0			
168	133		1530	ST=C		SAVE BINARY STATUS
169	134		414	?S8=1		MORE THAN ONE DIGIT
170	135		1640	RTN NC		256 WAY BRANCH
171	136		1	GOLONG	ERRDE	YES THEN DATA ERROR
171	137		2			
172				FILLTO	@137	
173	140	ROWTBL	1173	GOTO	XROW0 (57)	ROWTBL MUST BE AT @140
174	141		233	GOTO	XROW1 (164)	OTHER LOGIC MAKES USE OF
175	142		1313	GOTO	XROW2 (73)	THE FACT THAT ROWTBL IS IN
176	143		1303	GOTO	XROW3 (73)	THE FIRST 256 WORDS OF CHIP 0
177	144		423	GOTO	XROW4 (206)	AND ON AN EVEN BOUNDARY OF
178	145		343	GOTO	XROW5 (201)	16 WORDS.
179	146		333	GOTO	XROW6 (201)	
180	147		513	GOTO	XCUTE8 (220)	ROW 7
181	150		503	GOTO	XCUTE8 (220)	ROW 8
182	151		1313	GOTO	XROW9 (102)	
183	152		1073	GOTO	XROW10 (61)	
184	153		1103	GOTO	XROW11 (63)	
185	154		1113	GOTO	XROW12 (65)	
186	155		1123	GOTO	XROW13 (67)	
187	156		1133	GOTO	XROW14 (71)	
188	157		1	GOLONG	TEXT	ROW 15
188	160		2			
189	161	TONETC	1	GOSUB	TONSTF	
189	162		0			
190	163		1343	GOTO	BIGBRC (117)	
*						
192	164	XROW1	416	A=C		SAVE FC IN A
193	165		1034	PT=	2	
194	166		1520	LC	13	SPLIT OFF DIGIT ENTRY
195	167		1426	? A<C	XS	
196	170		1	GOLC	DERUN	DIGIT ENTRY
196	171		3			
197						NOTE - WE GO TO DERUN WITH
198						THE FC IN A[3:2] AND THE
199						PTR AT 1.
200	172		1	GOSUB	GTAINC	GET ALPHA OPERAND
200	173		0			
201	174		256	AC EX		
202	175		416	A=C		
203						MUST GO TO XCUTE8 WITH
204						FC BACK IN C[3:2] AND PT=3
205	176		223	GOTO	XCUTE8 (220)	
206	177	ROW7	1704	CLR ST		FOR ISG DSE COMP MSG
207	200		203	GOTO	XCUTE8 (220)	
*						
*						
210		XROW5				
211	201	XROW6	160	N=C		
212	202		772	C=C+C	M	
213	203		560	STK=C		PUTS ADDRESS OF NFRX ON STACK
214	204		1240	SETDEC		
215	205		63	GOTO	NATH (213)	

```

216 206 XROW4      360 NC EX
217 207           270 C=REGN 2
218 210           416 A=C
219 211           1 GOSUB  CHK#S
219 212           0
220 213 MATH      370 C=REGN 3
221 214           1 GOSUB  CHK#S2
221 215           0
222 216           1704 CLR ST
223 217           360 NC EX
* (FALL INTO XCUTB HERE)
*
* XCUTB - EXECUTE, PART B
* INPUT CONDITIONS: FC IN C(3:2), PT=3, ASSUMES NONPROGRAMMABLE
* XCUTB1 ASSUMES FC IN C(13:12)
*
230 220 XCUTB     174 RCR      4
231 221 XCUTB1    460 LDI
232 222           24 CON      024          @12000\256 MAIN FCN TABLE
233 223           1174 RCR     9
234 224           1460 CXISA
235 225           120 LC       1
236 226           674 RCR     11
237 227           740 GOTOC
*
*
*
* RSTKB - RESET AND DEBOUNCE KEYBOARD
* USES C,X
* WAITS 5 MILLISEC AFTER FIRST SEEING KEY RESET BEFORE
* ALLOWING A SECOND KEY TO BE SENSED.
*
* WAIT LOOP IS 4 WORDS LONG.
* 5 MILLISEC/ 4*155 MICROSEC = 8
*
* RST05 ENTRY POINT IS FOR DEBOUNCE ONLY
*
251           ENTRY  RSTKB
252           ENTRY  RST05
253 230 RSTKB     1710 RST KB
254 231           1714 CHK KB
255 232           1767 GOC     RSTKB ( 230 )
256 233 RST05     460 LDI
257 234           10 CON      8
258 235 RST10     1710 RST KB
259 236           1714 CHK KB
260 237           1146 C=C-1  X
261 240           1753 GONC    RST10 ( 235 )
262 241           1740 RTN
263           FILLTO:0241
264 242 ERRORF    1 GOSUB  ERROR          OVERFLOW TREATED AS ERROR
264 243           0
265 244           0 XDEF    MSGOF
*
267           NFRNC
268 245           1 GOSUB  OVFL10          !! ASSUMES CHIP 0 ON
268 246           0                      FILL X AND Y FROM N AND C
269 247           360 NC EX                      GET Y. SAVE X.
270 250           324 ? PT= 10
271 251           117 GOC     XBAD ( 262 ) GO IF X OVERFLOWED

```

```

272 252          1 GOSUB OVFL10
272 253          0
273 254          324 ? PT= 10
274 255          77 GOC YBAD ( 264) GO IF Y OVERFLOWED
275 256 FILLY 250 REGN=C 2 FILL IN Y VALUE
276 257          360 CN EX GET X OUT OF STORAGE
277 260          356 BC EX PUT X IN B. GOTO FILL X AND LASTX
278 261          713 GOTO FILLXL ( 352)
279 262 XBAD 1 GOSUB OVFL10 STILL NEED TO CONVERT Y TO 9 IF NE
279 263          0
280 264 YBAD 256 AC EX SAVE Y VALUE WHILE GET FLAGS
281 265          1670 C=REGN 14
282 266          574 RCR 6 GET ERROR AND OVERFLOW FLAGS FOR S
283 267          1530 ST=C
284 270          256 AC EX PUT Y BACK INTO C
285 271          1214 ?S7=1 OVERFLOW FLAG SET?
286 272          1647 GOC FILLY ( 256) IF SO FILL Y AND GO TO FILLXL
287          ENTRY ERRIGN
288 273 ERRIGN 514 ?S6=1 ERROR FLAG SET?
289 274          1463 GONC ERROF ( 242) IF NOT GOTO ERROR: OVERFLOW
290 275          504 S6= 0 TURN OFF ERROR FLAG
291 276          1630 C=ST
292 277          474 RCR 8
293 300          1650 REGN=C 14
294 301          603 GOTO NFRC ( 361) NO PRINT, LEAVE PUSH ALONE

*
* NFRSIG IS USED BY SIGMA+, SIGMA-, CLX, AND CLST.
* NFRENT IS USED BY ENTER.
298 302 NFRSIG 1665 CON 01665 GOSUB PRT1
299 303          674 CON 0674
300 304 NFRENT 604 S11= 0 CLEAR PUSHFLAG
301 305          543 GOTO NFRC ( 361)

*
303 306 NFRKB1 1114 ?S9=1 KEY ALREADY RESET?
304 307 NFRKB 1 GSUBNC RSTKB
304 310          0
305 311          503 GOTO NFRC ( 361)

*
307          FILLTHRU 0331
308 312          23 GOTO NFRX ( 314) MUST BE 00312 FOR ROW 5
309 313          0 NOP
310          NFRX
311 314          1 GOSUB OVFL10 !! ASSUMES CHIP 0 ON
311 315          0 MUST BE 00314 FOR ROW 6
312 316          356 BC EX SAVE X IN B
313 317          324 ? PT= 10
314 320          323 GONC FILLXL ( 352)
315 321          1670 C=REGN 14
316 322          574 RCR 6
317 323          1530 ST=C
318 324          1214 ?S7=1 OVERFLOW FLAG?
319 325          1463 GONC ERRIGN ( 273)
320 326          243 GOTO FILLXL ( 352)

*
* FILL THRU 0331 - GETS SPACING RIGHT SO THAT NFRPU ENDS UP
* AT 0360 AND THERE ARE NO INLINE HOPS
* PCTOC - PROGRAM COUNTER TO C
* THIS LITTLE SUBROUTINE SIMPLY COPIES THE ADDRESS OF THE ROM WORD
*-- AFTER THE CALLING GOSUB INTO C AND RETURNS. IT IS INTENDED TO
*-- FACILITATE THE WRITING OF ROUTINES IN PLUG IN ROMS FOR SUCH THINGS

```

*- AS CALLING ANOTHER ROM CHIP OR FOR OTHER ROUTINES REQUIRING
*- KNOWLEDGE OF THE CURRENT ABSOLUTE ADDRESS OF THE ROM.

```

*
331 327 PCTOC 660 C=STK
332 330      560 STK=C
333 331      1740 RTN
334          FILLTO:0331
335      NFRXY
336 332      1 GOSUB OVFL10
336 333      0
337 334      356 BC EX          SAVE X IN B
338 335      324 ? PT= 10
339 336      63 GONC DROPST ( 344 ) IF NO OVERFLOW GO DROP STACK
340 337      1670 C=REGN 14
341 340      574 RCR 6
342 341      1530 ST=C
343 342      1214 ?S7=1          OVERFLOW FLAG?
344 343      1303 GONC ERRIGN ( 273 ) IF NOT SET GO CHECK ERROR FLAG
345          ENTRY DROPST
346 344 DROPST 170 C=REGN 1      GET Z
347 345      250 REGN=C 2      PUT INTO Y
348 346      116 C=0
349 347      1160 DADD=C
350 350      70 C=DATA          GET T
351 351      150 REGN=C 1      PUT INTOZ
352

354          ENTRY FILLXL
355 352 FILLXL 370 C=REGN 3      GET OLD X
356 353      450 REGN=C 4      FILL LASTX
357 354      356 BC EX          GET NEW X FROM B
358 355      350 REGN=C 3      FILL X
359 356 NFRPR 1665 CON 01665    GOSUB PRT1
360 357      674 CON 0674
361          FILLTO:0357
362          0360 IS PUT ON THE STK
363          AT RUNNK - NFRPU MUST BE
364          AT 0360
364 360 NFRPU 610 S11= 1      SET PUSH
365 361 NFRFC 1140 SETHEX
366 362      240 SEL P
367 363      116 C=0
368 364      1160 DADD=C
369 365      1670 C=REGN 14
370 366      1530 ST=C
371      NFRFST
372 367      540 ?LLD          TEST LOW BATTERY
373 370      103 GONC LOWBRT ( 400 )
374 371      514 ?S6=1          LOWBAT?
375 372      67 GOC LOWBRT ( 400 ) YES
376 373      510 S6= 1          SET LOWBAT
377 374      1 GOSUB STOSTO      STORE STATUS SET 0
377 375      0
378 376      1 GOSUB ANNOUT      TURN ON BAT ANNUNCIATOR
378 377      0
379      LOWBRT          LOW BATTERY LOGIC RETURN TO MAIN FL.

*
381 400      1354 ?F13=1          DOES A PERIPHERAL WANT
382          SERVICE?
383 401      37 GOC IOSERV ( 404 ) YES
384 402      1014 ?S2=1          IOFLAG?

```

```

385 403      33 GONC   NFRNIO ( 406 ) NO
386 404 IOSERV    1 GOSUB  IORUN      YES
386 405      0

*
388      NFRNIO      NORMAL FUNCTION RETURN, NO I/O
389 406      1314 ?S13=1  RUNNING?
390 407      633 GONC   DRWSYL ( 472 ) NO
391      !!! CHECK SSTFLAG HERE?

*
393      ENTRY  RUNING      NULLS RE-ENTER HERE
394      RUNING      RUNNING
395 410      1714 CHK KB
396 411      243 GONC   RUNNK ( 435 )
397 412      1040 C=KEYS
398 413      74 RCR      3
399 414      126 C=0     XS
400 415      406 A=C     X      KEYCODE TO A.X
401 416      460 LDI
402 417      30 CON      24      H18=OFF KEY
403 420      1546 ? A#C   X
404 421      1 GOLNC    OFF
404 422      2
405 423      460 LDI
406 424      207 CON      135      H87=R/S KEY
407 425      1546 ? A#C   X
408 426      57 COC      IGNKEY ( 433 ) NOT RUNSTOP
409 427      1 GOSUB    RSTSEQ      STOP THE PROGRAM
409 430      0
410 431      1 GOLONG   NFRKB
410 432      2

*
412 433 IGNKEY 1710 RST KB      TRY TO RESET KEYBOARD
413 434      1714 CHK KB

*
415      RUNNK
416 435      134 PT=      4      RUNNING, NO KEY HIT
417 436      132 C=0     M      PUT NFRPU ON THE
418 437      1720 LC      15      SUBROUTINE STACK
419 440      560 STK=C      HERE
                                NFRPU ASSUMED = 0360

*
* NXTBYT - NEXT BYTE
* - INCREMENTS PGMCTR IN PLACE
* - PLACES BYTE POINTED TO BY NEW VALUE OF PGMCTR IN C[13-12]
* - FOR RAM ONLY, S8=1 IF BYTE NUM = 0 OTHERWISE S8=0.
* - IF S8=0 THEN C[11-10] CONTAINS THE NEXT BYTE IN PROGRAM MEMORY.
* - FOR ROM, S8 IS LEFT UNDEFINED, AND ONLY THE FIRST BYTE IS
* - BROUGHT INTO C.
* - ASSUMES CHIP 0 SELECTED AND PT=3, LEAVES PT=3, USES C.
*
430 441      1470 C=REGN 12      PGMCTR TO C[3-0]
431 442      314 ?S10=1      ROMFLAG?
432 443      227 GOC      NEXROM ( 465 ) YES
433 444      404 S8=      0
434 445      1142 C=C-1     PT      DECREMENT BYTE NUMBER
435 446      107 GOC      NXTBT1 ( 456 ) BYTE 6 DESIRED
436 447      1450 REGN=C 12      REPLACE PGMCTR
437 450      1160 DADD=C      TURN ON THE RIGHT SLEEPER CHIP
438 451      174 RCR      4      BYTE NUM TO C.S
439 452      460 LDI
440 453      24 CON      024      TBLGBR\16=0500\16

```

```

441 454      374 RCR      10
442 455      740 GOTOC

*
444 456 NXTBT1 620 LC      6      DESIRED BYTE IS BYTE #6
445 457      1146 C=C-1    X
446 460      1450 REGN=C  12
447 461      34 PT=       3
448 462      1160 DADD=C
449 463      70 C=DATA
450 464      433 GOTO     NXBEND ( 527)

*
452 465 HEXROM 1056 C=C+1      INCREMENT PGMCTR
453 466      1450 REGN=C  12    PUT PGMCTR BACK
454 467      674 RCR      11
455 470      1460 CXISA     NEW BYTE TO C.X
456 471      353 GOTO     NXROM1 ( 526)

*
458 472 DRWSYL 463 GOTO     DROWSY ( 540)

*
*
* STOST0 - STORE STATUS SET 0 BACK TO REG 14
* ENTRY REQUIREMENTS: CHIP 0 ENABLED, STATUS SET 0 IN STATUS BITS
* DESTROYS C (LEAVES A COPY OF REG 14 IN C)
*
465      ENTRY STOST0
466 473 STOST0 1670 C=REGN 14
467 474      1630 C=ST
468 475      1650 REGN=C  14
469 476      1740 RTN
470      FILLTO 10477
477      0000 HOP

471
472      TBLGBR
473
474 500      243 GOTO     GBYTR0 ( 524) NEW BYTE NUM = 0
475 501      203 GOTO     GBYTR1 ( 521)
476 502      143 GOTO     GBYTR2 ( 516)
477 503      103 GOTO     GBYTR3 ( 513)
478 504      43 GOTO     GBYTR4 ( 510)
479 505 GBYTR5 70 C=DATA
480 506      1574 RCR      12
481 507      203 GOTO     NXBEND ( 527)
482 510 GBYTR4 70 C=DATA
483 511      374 RCR      10
484 512      153 GOTO     NXBEND ( 527)
485 513 GBYTR3 70 C=DATA
486 514      474 RCR      8
487 515      123 GOTO     NXBEND ( 527)
488 516 GBYTR2 70 C=DATA
489 517      574 RCR      6
490 520      73 GOTO     NXBEND ( 527)
491 521 GBYTR1 70 C=DATA
492 522      174 RCR      4
493 523      43 GOTO     NXBEND ( 527)
494 524 GBYTR0 70 C=DATA
495 525      410 SB=       1
496 526 NXROM1 1074 RCR      2
497      NXBEND
498 527      1614 ?S0=1      END OF NEXT BYTE
499 530      33 GONC     NOPRT ( 533) NO
IS A PRINTER CONNECTED?

```

500	531	1655 CON	@1655	IN TRACE MODE, PRINT
501				NEXT INSTRUCTION
502	532	674 CON	@674	GOSUB PRT2
503		ENTRY	NOPRT	FOR THE PRINTER
504		NOPRT		

10

- * XCUTE - EXECUTE
- * - DECODES AND SENDS TO EXECUTION THE BYTE FOUND IN
- * C[13-12]. IF S8=0 THEN C[11-10] CONTAINS THE
- * NEXT BYTE.
- * - ON INPUT: HEXMODE, PTR P = 3, STATUS SET 0 UP AND VALID
- * - SELECTS RAM CHIP 0.
- *

513	533	XCUTE	460	LDI		ROMTBL\$16
514	534		6	CON	@006	ROMTBL MUST BE IN 1ST 256 BYTES OF
515						SELECT RAM 0
516	535		1160	DADD=C		
517	536		374	RCR	10	
518	537		740	GOTOC		
519				EJECT		

NOMAS

Not Manufacturer Supported
recipient agrees NOT to contact manufacturer

* DROWSY - REFRESH DISPLAY AND TRY TO SLEEP

```

*
522 540 DROWSY 1110 S9= 1 KEYBOARD ALREADY RESET
523 541 DRSY05 1 GOSUB ANNOUT REFRESH ANNUNCIATORS
523 542 0
524 543 214 ?S5=1 MSGFLG?
525 544 177 GOC DRSY25 ( 563 ) YES
526 545 14 ?S3=1 PRGMMODE?
527 546 43 GONC DRSY10 ( 552 ) NO
528 547 1 GOSUB DFEST8 DFILLF WITH SCROLL & NO PROMPT
528 550 0
529 551 123 GOTO DRSY25 ( 563 )
530 552 DRSY10 1214 ?S7=1 ALPHAMODE?
531 553 53 GONC DRSY20 ( 560 ) NO
532 554 404 S8= 0 SCROLL & NO PROMPT
533 555 1 GOSUB ARGOUT
533 556 0
534 557 43 GOTO DRSY25 ( 563 )
535 560 DRSY20 370 C=REGN 13 GET X
536 561 1 GOSUB DSPCRG DISPLAY CONTENTS OF C REG
536 562 0
537 DRSY25

```

* SST (PRGMODE ONLY) AND EST (PRGMODE & NORMAL MODE) ENTER
 * AT DRSY25 TO BYPASS BOTH MAIN LCD UPDATE AND ANNUNCIATOR
 * UPDATE. ENTRY CONDITIONS ARE THE SAME AS FOR DRSY51.

```

541 563 1114 ?S9=1 KEYBOARD RESET YET?
542 564 77 GOC DRSY30 ( 573 ) YES
543 565 460 LDI DELAY 25 MILLISEC
544 566 121 CON 81 FOR DEBOUNCE
545 567 DRSY26 1146 C=C-1 X
546 570 1773 GONC DRSY26 ( 567 )
547 571 1 GOSUB RSTKB
547 572 0

```

```

*
549 573 DRSY30 104 S4= 0 CLEAR SSTFLAG
550 574 1 GOSUB STOST0
550 575 0
551 576 1414 ?S1=1 PAUSING?
552 577 307 GOC PAUSLP ( 627 ) YES

```

* LIGHT SLEEP WAKEUP LOGIC

```

*
556 ENTRY LSMKUP
557 600 LSMKUP 1 CON 00001 GOSUB DIAGNOSTIC
558 601 400 CON 00400
559 602 1 GOSUB PACH11 LEAVES SS0 UP
559 603 0
560 PACH11 GOES TO MEMCHK
561 ENTRY WKUP10 PARSE PKSEQ ENTERS HERE
562 604 WKUP10 1714 CHK KB
563 605 417 GOC WKUP20 ( 646 )
564 606 460 LDI
565 607 10 CON 8 I O SERVICE
566 610 1 GOSUB ROMCHK NEEDS CHIP 0,SS0,HEX,P SELECTED
566 611 0
567 612 1014 ?S2=1 IOFLAG?
568 613 1717 GOC WKUP10 ( 604 ) YES
569 GOING TO LIGHT SLEEP NOW
570 614 1670 C=REGN 14

```



```

571 615          1074 RCR      2
572 616          1530 ST=C
573 617          14  ?S3=1
574          ENTRY  DRSY50
575          DRSY50
576
577 620          1  GSUBNO ENLCO
577 621          0
578 622          140 POWOFF
578 623          0
*
*
581          DRSY51
582
583
584
585
586
587 624          1  GOSUB  ANNOUT
587 625          0
588 626          1353 GOTO  DRSY25 ( 563 )
*
* PAUSE LOOP
*
592 627 PAUSLP    1  GOSUB  PGMAON          TURN ON PRGM ANNUNCIATOR
592 630          0
593 631          460 LDI                    INITIALIZE PAUSETIMER
594 632          134 CON      92
595 633          406 A=C      X            A.X=PAUSETIMER
* PAUSETIMER SET EMPIRICALLY TO MATCH HP67 ON A BENCHMARK PGM
* CONSISTING OF 100 PSE'S FOLLOWED BY FIX 9, STOP.
* THIS TIMING WAS SUBSEQUENTLY SCREWED UP BY EXTENDING ROMCHK'S
* SEARCH FROM ADDRESSES C-F DOWN TO 5-F.  HP-41C'S PSE IS NOW
* .1-.2 SEC LONGER THAN HP-67'S.  DRC 10/20/79
601 634 PAUS10 1714 CHK KB          IS A KEY DOWN?
602 635          117 GDC      WKUP20 ( 646 ) YES
603 636          460 LDI
604 637          14 CON      12
605 640          1  GOSUB  RMCK05
605 641          0
606 642          646 A=A-1  X            HAS PAUSE EXPIRED?
607 643          1713 GONC  PAUS10 ( 634 ) NO, NOT YET
608 644          1  GOLONG RUN          YEP
608 645          2
*
*
611 646 WKUP20 1040 C=KEYS
612          ENTRY  WKUP21          ADD FOR ADV 10 ON 6/15/81
613 647 WKUP21    34 PT=      3
614 650          742 C=C+C  PT          OFF KEY? (OFF KC=18HEX)
615 651          1  GOLNC  PARSE          NO
615 652          2
616 653 OFFXFR    1  GOLONG OFF          YES
616 654          2
*
* DEEP SLEEP WAKEUP LOGIC
*
620          ENTRY  DSMKUP          WAKE UP FROM DEEP SLEEP
621 655 DSMKUP    1  CON      @0001          GOSUB DIAGNOSTIC
622 656          400 CON      @0400          CHIP 4

```

* ON WAKEUP FROM DEEP SLEEP, THE DISPLAY MAY BE EITHER OFF (IN THE
 * CASE WHERE THE USER OR A PROGRAM TURNED THE CALCULATOR OFF
 * EXPLICITLY) OR ON (IN THE CASE WHERE THE CALCULATOR WENT FROM
 * LIGHT SLEEP TO DEEP SLEEP AUTOMATICALLY).

627	657	1340	DISOFF		GET THE DISPLAY TO A KNOWN STATE
628					
629	660	1	GOSUB	PACH11	PACH11 GOES TO MEMCHK
629	661	0			
630	662	1714	CHK	KB	DID THE ON KEY WAKE US UP?
631	663	77	GOC	WKUP25 (672)	YES
632	664	460	LDI		NO
633	665	12	CON	10	
634	666	1	GOSUB	ROMCHK	
634	667	0			
635	670	1014	7S2=1		IOFLAG?
636	671	1273	GONC	DRSYS0 (620)	NOPE - GO BACK TO SLEEP
637			ENTRY	WKUP25	
638					
639					INITIALIZE STATUS BITS
640	672	1670	C=REGN	14	
641	673	1	GOSUB	PACH12	DECOMPILES & RTNS WITH R14 IN C,
641	674	0			
642					SS0 UP(S0-S7=0), C.X= 0
643	675	574	ROR	6	
644	676	1730	CST	EX	PUT UP SS 3
645	677	1404	S1=	0	CLEAR CATALOG FLAG
646	700	210	S5=	1	SET AUDIO ENABLE FLAG
647	701	504	S6=	0	CLEAR ERROR IGNORE FLAG
648	702	1204	S7=	0	CLEAR OUT-OF-RANGE FLAG
649	703	1730	CST	EX	
650	704	1074	ROR	2	CLEAR FLAGS 12-23
651	705	106	C=0	X	
652	706	574	ROR	6	
653	707	1650	REGN=C	14	
654	710	1304	S13=	0	CLEAR RUNNING FLAG
655					RELEASE ALL I/O BUFFERS
656	711	1570	C=REGN	13	
657	712	356	BC	EX	CHAINHEAD TO B.X
658	713	460	LDI		
659	714	277	CON	191	
660	715	416	A=C		CURRENT REG ADDR TO A.X
661	716	WKUP30	546	A=A+1	X
662	717	WKUP40	1446	? A<B	X
663	720		173	GONC	WKUP50 (737)
664	721		246	C=A	X
664	722		406		
665	723		1160	DADD=C	
666	724		70	C=DATA	
667	725		1356	? C#0	W
668	726		113	GONC	WKUP50 (737)
669	727		1076	C=C+1	S
670	730		1667	GOC	WKUP30 (716)
671	731		136	C=0	S
672	732		1360	DATA=C	
673	733		374	ROR	10
674	734		126	C=0	XS
675	735		506	A=A+C	X
676				LEGAL	
677	736		1613	GOTO	WKUP40 (717)

*

```

679 737 WKUP50 460 LDI
680 740          7 CON      7      DEEP SLEEP
681 741          1160 DADD=C    RE-ENABLE CHIP 0
682 742          1 GOSUB  ROMCHK
683 743          0
684 744          1 GOSUB  PKIOAS    GOSUB I/O AREA PACK SUBR.
685 745          0
686          RETURNS WITH CHIP 0 DISABLED
687          RE-ENABLE CHIP 0
688 746          116 C=0
689 747          1160 DADD=C
690 750          1 GOSUB  RSTKB
691 751          0
* CHECK FOR MASTER CLEAR HERE
* THE PROTOCOL FOR MASTER CLEAR IS TO PRESS AND HOLD THE
* BACKARROW KEY WHILE SIMULTANEOUSLY HITTING THE ON KEY.
692 752          1714 CHK KB      ANOTHER KEY DOWN?
693 753          113 GONC  WKUP60 ( 764 ) NO
694 754          460 LDI          YES. SEE IF IT IS BKARROW
695 755          303 CON2      12      3      KC FOR BKARROW
696 756          406 A=C      X
697 757          1040 C=KEYS
698 760          74 RCR      3
699 761          1434 PT=      1
700 762          1552 ? A#C  WPT
701 763          773 GONC  COLDST (1062) MASTER CLEAR
702          WKUP60
703 764          1440 DISTOG      TURN THE DISPLAY BACK ON
704 765 WKUP70 1670 C=REGN 14
705 766          674 RCR      11
706 767          1530 ST=C
707 770          1614 ?S0=1      FLAG 11?
708 771          1 GOLHC  NFRC      NO
709 772          2
710          GOTO NFRC TO INITIALIZE
711          LOWBAT BEFORE GOING TO
712          DROMSY
713 773          1604 S0=      0      YES. CLEAR FLAG 11
714 774          1630 C=ST
715 775          74 RCR      3
716 776          1650 REGN=C 14
717          ENTRY  WKUP80
718 777 WKUP80 1340 DISOFF      FOR CARD RDR LOAD&GO
719 1000          1 GOSUB  TONE7X    TURN OFF DISPLAY DURING BEFP
720 1001          0
721 1002          1440 DISTOG      TURN DISPLAY BACK ON
722 1003          1 GOLONG. RUN    START RUNNING THE USER'S PGM
723 1004          2
*
* MEMCHK (MEMORY CHECK) - CHECK INTEGRITY OF ROM AND RAM
*
* MEMCHK PERFORMS THREE QUICK TESTS OF RAM AND ROM IN AN
* EFFORT TO DETERMINE WHETHER ANY PLUG-IN MODULES OR THE
* BATTERIES HAVE BEEN REMOVED.
*
* 1. TEST DIGITS 0:6 OF REG 13 TO SEE WHETHER THE WARM START
* CONSTANT (0551) IS THERE. IF NOT, COLD START.
* 2. READ/WRITE/READ/RESTORE REG0-1 TO JUDGE WHETHER THE LABEL
* CHAIN IS INTACT. IF NOT, COLD START.
* 3. IF THE USER PC IS ON ROM, VERIFY THAT THE FIRST WORD OF THE
* ROM CHIP IS NON-ZERO TO JUDGE WHETHER THE ROM MODULE IS STILL

```

* PLUGGED IN. IF NOT, SET THE PC TO THE TOP OF PROGRAM MEMORY IN
 * RAM. (SEE CHKRPC COMMENTS BELOW)

11

* ON EXIT, CHIP 0 IS ENABLED, SSO IS UP, HEXMODE.
 * USES A AND C.
 * DOESN'T CALL ANY SUBROUTINES (MUST NOT, BECAUSE MEMCHK IS CALLED
 * DURING PARTIAL KEY SEQUENCES). EXITS VIA PUTPCX.
 * IF PC IS IN RAM, NORMALLY RETURNS IN 31 WORD-TIMES.
 * IF PC IS IN ROM, NORMALLY RETURNS IN 39 WORD-TIMES.
 *

743		ENTRY	MEMCHK	
744	1005	MEMCHK	1710 RST KB	THESE THREE STATES
745	1006		1714 CHK KB	NECESSARY BECAUSE OF
746	1007		1140 SETHEX	PROBLEMS WITH CPU WAKEUP
747	1010		106 C=0 X	
748	1011		1760 PFAD=C	TURN OFF PERIPHERAL CHIPS
749	1012		1160 DADD=C	TURN ON CHIP 0
750	1013		460 LDI	
751	1014		551 CON 0551	WARM START CONSTANT
752	1015		406 A=C X	
753	1016		1570 C=REGH 13	
754	1017		574 RCR 6	
755	1020		1546 ? A#C X	COLD START?
756	1021		417 GOC COLDST (1062)	YES
757				NOW HEXMODE IS ASSUMED
758	1022		674 RCR 11	REG0 TO C.X
759	1023		1146 C=C-1 X	C.X=REG0-1
760	1024		1160 DADD=C	
761	1025		70 C=DATA	GET C(REG0-1)
762	1026		416 A=C	& SAVE IN A
763	1027		1272 C=-C-1 M	

* WE INVERT THE BIT PATTERN IN DIGITS 12:13. CHARACTERISTICALLY,
 * WHEN A NON-EXISTENT DATA STORAGE REGISTER IS READ, THE DATA
 * IS EITHER ALL ONES OR ALL ZEROES. INVERTING PART OF THE REGISTER
 * GUARANTEES THAT, IF THE REGISTER EXISTS, EITHER WHAT WE READ
 * ORIGINALLY OR THE PARTIALLY INVERTED PATTERN WILL BE DIFFERENT
 * FROM ALL ZEROES AND FROM ALL ONES.

770	1030		1360 DATA=C	WRITE IT BACK
771	1031		70 C=DATA	READ IT AGAIN
772	1032		1272 C=-C-1 M	INVERT IT AGAIN
773	1033		1556 ? A#C	NON-EXISTENT REGISTER?
774	1034		267 GOC COLDST (1062)	YES
775	1035		1360 DATA=C	RESTORE THE REGISTER
776				
777	1036		106 C=0 X	RE-ENABLE CHIP 0
778	1037		1160 DADD=C	
779	1040		1670 C=REGH 14	PUT UP SSO
780	1041		1530 ST=C	
781				

* CHKRPC (CHECK ROM PC) - CONFIRMS THAT, IF ROMFLAG IS SET, THE
 * ROM CHIP POINTED TO BY THE USER PC IS ACTUALLY PLUGGED IN.

* ON ENTRY, CHIP 0 MUST BE ENABLED.
 * IF ROMFLAG IS CLEAR, RETURNS IN 2 WORD-TIMES AND USES NOTHING.
 * IF ROMFLAG IS SET, USES A[31:0] AND C AND PT AND USUALLY RETURNS
 * IN 8 WORD-TIMES.

790		ENTRY	CHKRPC	
791	1042	CHKRPC	314 ?S10=1	ROMFLAG?
792	1043		1640 RTH NC	NO, ALL FINISHED.

```

793 1044      1470 C=REGN 12
794 1045      106 C=0    X
795 1046      674 RCR    11
796 1047      1460 CX1SA
797 1050      1346 ? C#0  X
798 1051      1540 RTN C
799 1052      304 S10=   0
800 1053      1570 C=REGN 13
801 1054      74 RCR     3
802 1055      34 PT=     3
803 1056      102 C=0    PT
804 1057      412 A=C     UPT
805 1060      1 GOLONG PUTFCX
805 1061      2

```

```

GET PC
C[3:0]=ADDR OF 1ST WORD
ON CHIP

```

CHIP IS NOT THERE

C.X=REG0

* COLD START INITIALIZATION

```

*
809 1062 COLDST 1140 SETHEX
810 1063      640 CLRABC
811 1064      530 M=C
812 1065      160 N=C
813 1066      130 G=C
814 1067      1530 ST=C
815 1070      1130 F=SB
816 1071      560 STK=C
817 1072      560 STK=C
818 1073      560 STK=C
819 1074      560 STK=C
820 1075      340 SEL Q
821 1076      1334 PT=    13
822 1077      240 SEL P
823 1100      1304 S13=   0
824 1101      1504 S12=   0
825 1102      604 S11=   0
826 1103      304 S10=   0
827 1104      1104 S9=    0
828 1105      404 S8=    0
829 1106      1 GOSUB MSGA
829 1107      0
830 1110      0 XDEF MSGML "MEMORY LOST" MESSAGE
* IS THE LCD ENABLE IN THE NEXT LINE REALLY NECESSARY?
832 1111      1 GOSUB ENLCD
832 1112      0
833 1113      1 GOSUB RSTKB
833 1114      0
834 1115      460 LDI      SET UP A.X FOR ILOOP
835 1116      1777 CON    @1777
* I THINK THIS CONSTANT COULD JUST AS WELL BE @777, WHICH WOULD
* RESULT IN FASTER COLD STARTS, BUT FOR NOW I'M LEAVING WELL ENOUGH
* ALONE. DRC 3/26/79
839 1117      416 A=C
840 1120      116 C=0
841 1121      1360 WRTE
842 1122      1340 DISOFF
843 1123      1440 DISTOG
844 1124      1760 PFAD=C
845 1125 ILOOP 256 AC EX
846 1126      1160 DADD=C
847 1127      256 AC EX
848 1130      1360 DATA=C

```

CLEAR ANNUNCIATORS

849	1131	646	A=A-1	X		
850	1132	1733	GONC	IL00P	(1125)	
* 852 1133 460 LDI INITIALIZE REG0 (0EF)						
853	1134	357	CON2	14	15	
854	1135	474	RCR	8		
855	1136	460	LDI			INITIALIZE SIGNADDR (0FA)
856	1137	372	CON2	15	10	
857	1140	74	RCR	3		
858	1141	460	LDI			
859	1142	356	CON2	14	14	INITIALIZE CHAINHEAD (0EE)
860	1143	1550	REGN=C	13		
861	1144	132	C=0	N		
862	1145	1056	C=C+1			
863	1146	1450	REGN=C	12		PGMPTR (00EF)
864	1147	1160	DADD=C			PUT PERMANENT END AT CHAINHEAD
865	1150	116	C=0			LOCATION
866	1151	234	PT=	5		
867	1152	1420	LC	12		
868	1153	460	LDI			
869	1154	40	CON	32		
870	1155	1650	REGN=C	14		
871	1156	116	C=0			INITIALIZE STATUS BITS
872	1157	1160	DADD=C			
873	1160	1234	PT=	7		
874	1161	220	LC	2		TURN ON AUDIO ENABLE
875	1162	1420	LC	12		SET DIGIT GROUPING & DP FLAGS
876	1163	134	PT=	4		
877	1164	420	LC	4		#DIGITS_4
878	1165	1020	LC	8		SET FIXFLAG
879	1166	1650	REGN=C	14		STORE STATUS SETS EXCEPT SS0
880	1167	210	SS=	1		SET MSGFLG
* ROMCHK ASSUMES SS0 IS UP, CLEARS S2 (IOFLAG), AND STORES SS0						
* BACK TO REG 14						
883	1170	460	LDI			
884	1171	6	CON	6		COLD START
885	1172	1	GOSUB	ROMCHK		
885	1173	0				
886	1174	460	LDI			
887	1175	551	CON	0551		WARM START CONSTANT
888	1176	406	A=C	X		
889	1177	1570	C=REGN	13		
890	1200	574	RCR	6		
891	1201	246	AC EX	X		
892	1202	474	RCR	8		
893	1203	1550	REGN=C	13		
894	1204	1	GOLONG	WKUP70		
894	1205	2				
895			EJECT			

```

* INCGT2 -- INCREMENT PGMCTR AND VALIDATE BYTE#2
* INPUT: C AS LEFT BY ROW DECODE (FC IN DIGITS 3:2, 2ND BYTE
* MAY BE IN DIGITS 1:0). PT=3. STATUS SET 0 UP.
* USES A AND C
* RETURNS WITH VALID BYTES IN A(3:0), PT=3, STATUS SET 0 UP.
* LEAVES S8 ALONE
*
903 1206 INCGT2 416 A=C          SAVE FIRST BYTE IN A(3:2)
904 1207      1314 ?S13=1      RUNNING?
905 1210      37 GOC          INCG1 (1213) YES
906 1211      114 ?S4=1      SSTFLAG?
907 1212      1640 RTN NC      KEYBOARD - DO NOTHING.

*
909 1213 INCG1 1470 C=REGN 12    GET PGMCTR
910 1214      314 ?S10=1      ROMFLAG?
911 1215      103 GONC        INCG2 (1225) RAM
912 1216      1056 C=C+1      ROM
913 1217      1450 REGN=C 12    PUT PGMCTR BACK
914 1220      674 RCR         11
915 1221      1460 CXISA
916 1222      266 AC EX      XS    PUT THE TWO BYTES TOGETHER
917 1223      406 A=C        X
918 1224      1740 RTN

*
920 1225 INCG2 414 ?S8=1        IS BYTE 2 BAD?
921 1226      153 GONC        INCG3 (1243) BYTE 2 IS GOOD
922 1227      620 LC          6    INCREMENT PGMCTR ACROSS A
923 1230      1146 C=C-1      X    REGISTER BOUNDARY
924 1231      1450 REGN=C 12    PUT PGMCTR BACK
925 1232      1160 DADD=C      GET SECOND BYTE
926 1233      70 C=DATA
927 1234      1574 RCR         12
928 1235      1434 PT=         1
929 1236      412 A=C          WPT
930 1237      34 PT=          3    RESTORE POINTER
931 1240      116 C=0
932 1241      1160 DADD=C      RE-ENABLE STATUS CHIP
933 1242      1740 RTN

*
935 1243 INCG3 1142 C=C-1 PT    INCREMENT PGMCTR
936 1244      1450 REGN=C 12    PUT PGMCTR BACK
937 1245      1740 RTN
938      EJECT

```

*ROM10 INCLUDES FLAGS, EXEC ROM, NON-PROGRAMMABLE
 *FUNCTIONS AND EXECUTE INDIRECT. FLAGS ARE THE ONLY
 *FUNCTIONS IN ROM10 WHICH CAN BE PREPROCESSED. IN THE
 *ROM 10 ROUTINE ERROR CHECKING IS DONE AND A MASK
 *IS BUILT WITH A ONE IN THE POSITION OF THE FLAG
 *OF INTEREST.

947	1246	ROM10	1	GOSUB	INCGT2	GET BYTE 2
947	1247		0			
948	1250		256	AC	EX	
949	1251		1530	ST=C		SAVE BYTE 2
950	1252		1034	PT=	2	
951	1253		130	G=C		
952				ENTRY	P10RTN	
953	1254	P10RTN	766	C=C+C	XS	SEP XEC ROM
954	1255		1	GOLNC	XROM	
954	1256		2			
955	1257		1104	S9=	0	TEST ONLY FLAG SET
956	1260		766	C=C+C	XS	SEP SET AND CLEARS
957	1261		103	GONC	FLAGS (1271)	
958	1262		1110	S9=	1	THESE 2 TEST ONLYS
959	1263		766	C=C+C	XS	
960	1264		53	GONC	FLAGS (1271)	
961	1265		766	C=C+C	XS	SPARE FC?
962	1266		1540	RTN C		YES
963	1267		1	GOLONG	BIGBRC	XEQ/GTO INDIRECT
963	1270		2			
964	1271	FLAGS	126	C=0	XS	CLEAR FRO ERROR CHECKS
965	1272		1214	?S7=1		INDIRECT FLAG?
966	1273		63	GONC	CONFLG (1301)	NO
967	1274		1204	S7=	0	DO INDIRECT ACCESS
968	1275		1	GOSUB	ADRFCH	
968	1276		0			
969	1277		1	GOLONG	PACH10	
969	1300		2			
970	1301	CONFLG	256	AC	EX	MOVE BINARY FLAG NUMBER TO A
971	1302		216	B=A		SAVE N IN B
972	1303		460	LDI		LOAD DECIMAL 30
973	1304		36	CON	036	
974	1305		706	A=A-C	X	CHECK TO SEE IF SETCLR FLAG
975	1306		77	GOC	ALLOK (1315)	YES THEN ALL OPS OK
976	1307		1114	?S9=1		TEST ONLY FLAG?
977	1310		303	GONC	ERRNE (1340)	NO THIS ONE SET OR CLEARS
978	1311		460	LDI		SUBTRACT BALANCE OF FLAGS
979	1312		32	CON	032	
980	1313		706	A=A-C	X	IF NO NN>55
981	1314		243	GONC	ERRNE (1340)	

*

* THE ENTRY POINT "ALLOK" WAS ADDED BY STEVE CHOU ON 02-11-81

* FOR THE FUNCTION "STOFLAG" IN THE ADVANCED PROGRAMMING ROM

*

986				ENTRY	ALLOK	
987	1315	ALLOK	156	AB	EX	NO ERRORS AT THIS POINT
988	1316		460	LDI		COUNT DOWN BY 85
989	1317		10	CON	010	
990	1320		356	BC	EX	
991	1321		116	C=0		SET C=1 AND ADDRESS CHIP 0
992	1322		1160	DADD=C		


```

993 1323      1056 C=C+1
994 1324 SHF8  1074 RCR      2      SHIFT ONE RIGHT 8 AT A TIME
995 1325      606 A=A-B    X      COUNT N DOWN
996 1326      1763 GONC    SHF8    (1324)
997 1327      23  GOTO    PSTDBL (1331)
998 1330 DBL   756 C=C+C
999 1331 PSTDBL 546 A=A+1    X      SHIFT BACK BY CARRY AMOUNT
1000 1332      1763 GONC    DBL    (1330)      COUNT BACK CARRY
1001 1333      356 CB EX
1002 1334      1670 C=REGN 14      SAVE MASK
1003 1335      256 AC EX          GET STATUS SET
1004 1336      1  GOLONG BIGBRC    SAVE IN A
1004 1337      2                  DO 256 WAY BRANCH
1005 1340 ERRNE 1  GOSUB  ERROR
1005 1341      0
1006 1342      0 XDEF  MSGHE      "NONEXISTENT"

```

*THIS ROUTINE TAKES A STANDARD FLOATING POINT
 *NUMBER, STRIPS OFF AN ABSOLUTE INTEGER LESS THAN
 *1000, AND COVERTS THAT INTEGER TO BINARY.
 *IF THE FLOATING POINT INPUT IS A FRACTION ZERO
 *IS RETURNED, IF LARGER THAN 999 A NONEXISTANT
 *ERROR IS GENERATED. INPUT IS IN C, OUTPUT IS
 *IN C-X, CHARACTER DATA ALSO GENERATES ERROR.
 * USES: A,X, C, S8, AND 1 ADDITIONAL SUBROUTINE LEVEL
 * IN: C=FLOATING POINT NUMBER
 * NO PERIPHERAL ENABLED
 * OUT: C,X = BINARY NUMBER
 * CHIP 0 ENABLED
 * MAY EXIT TO ERRAD OR ERRNE

```

1022 1343 BCDBIN 1176 C=C-1  S      CHECK FOR CHARACTER
1023 1344      1176 C=C-1  S
1024 1345      1  GOLC    ERRAD
1024 1346      3
1025 1347      406 A=C    X      MOVE EXPONENT
1026 1350      136 C=0    S
1027 1351      404 S8=    0      CLEAR ZERO TO 9 FLAG
1028 1352      106 C=0    X
1029 1353      1160 DADD=C
1030 1354      1526 ? A#0  XS      NEGATIVE EXPONENT?
1031 1355      1540 RTN C      YES WE ARE DONE
1032 1356      1574 RCR    12      MOVE DIGIT 1 TO 0
1033 1357      646 A=A-1  X      DEC EXP
1034 1360      107 GOC    GOTINT (1370)  DONE IF X=0
1035 1361      410 S8=    1      SET FLAGS FOR 10 OR LARGER
1036 1362      1374 RCR    13      ROT NXT DG IN
1037 1363      646 A=A-1  X      EXP=1?
1038 1364      47  GOC    GOTINT (1370)  YES
1039 1365      1374 RCR    13      SHIFT AGAIN
1040 1366      646 A=A-1  X      X=2
1041 1367      1513 GONC  ERRNE (1340)  VALUE TOO LARGE FOR ADR

```

*THE FOLLOWING ROUTINE TAKES A BCD INTEGER IN C-X
 *(3 DIGITS) AND CONVERTS IT TO BINARY IN C-X

*
 * IN: C,X= BCD NUMBER, C[4:3]= 00
 * ASSUME: HEXMODE
 * OUT: C,X= BINARY NUMBER, HEXMODE
 * USES: A,X, C, +1 SUB LEVEL (NO ST, NO PT, NO DADD)

WOMAS
 Not Manufacturer Supported
 recipient address NOT to contact manufacturer

```

*****
1051 1370 GOTINT 1074 RCR      2      GET FIRST DIGIT
1052 1371          1 GDSUB  INTINT
1052 1372          0
1053 1373 INTINT  746 C=C+C  X      MULTIPLY BY 10
1054 1374          406 A=C    X
1055 1375          746 C=C+C  X
1056 1376          746 C=C+C  X
1057 1377          506 A=A+C  X
1058 1400          106 C=0    X
1059 1401          1374 RCR   13      SHIFT IN NEXT DIGIT
1060 1402          1066 C=C+A  X      COMBINE 10S
1061 1403          1740 RTN
1062          EJECT

```

```

*
* GTAINC - GET ALPHA LABEL AND INCREMENT PROGRAM
*          COUNTER
*-- GET AN ALPHA LABEL FROM VARIOUS LOCATIONS DEPENDING
*-- ON THE MODE OF OPERATION, AND FORMAT THE ALPHA
*-- LABEL APPROPRIATELY
*-- IN:  S9=1 IMPLIES AN ADDRESS IS RETURNED IN M
*--      A[3:2]= FUNCTION CODE
*--      CHIP 0-SELECTED
*-- OUT: M[13:0]= ALPHA LABEL (RIGHT JUSTIFIED)
*--      OR ALPHA LABEL ADDRESS
*--      PC SET AT LAST BYTE OF ALPHA LABEL
*-- USES: A[13:0], B[13:0], C[13:0], M[13:0]
*-- USES: 1 SUBROUTINE LEVEL
1077
1078
1079
1080 1404 GTAINC    34 PT=      3          -
1081 1405          216 B=A          COPY FC FROM A[3:2]
1082 1406          316 C=B          TO B[3:2] AND C[3:2]
1083 1407          1314 ?S13=1      RUNNING?
1084 1410          107 GOC          GTAI10 (1420) YES
1085 1411          114 ?S4=1      SSTFLAG?
1086 1412          67 GOC          GTAI10 (1420) YES
1087 1413          1170 C=REGN 9    M_ALPHA STRING (KYBRD)
1088 1414          1114 ?S9=1      ADDR IN M?
1089 1415          27 GOC          **2 (1417) YES
1090 1416          530 M=C          -
1091 1417          1740 RTN          -
1092 1420 GTAI10   314 ?S10=1      ROM?
1093 1421          603 GONC          GTAI40 (1501) NOPE
1094 1422          1470 C=REGN 12   B[6:3]_FGMCTR (ROM)
1095 1423          674 RCR          11
1096 1424          1072 C=C+1 M     -
1097 1425          356 BC EX          C[3:2]_F.C.
1098 1426          742 C=C+C PT    ALBL?
1099 1427          337 GOC          GTAI22 (1462) YES
1100 1430          316 C=B          XEQ/GTO F.C.
1101 1431          1460 CXISA       STRING OPERAND ADDR?
1102 1432 GTAI26   216 B=A          SAVE F.C. & K.C.
1103 1433          432 A=C M        A[6:3]_FGMCTR
1104 1434          1474 RCR          1    A[13]_#CHARS
1105 1435          1176 C=C-1 S     -
1106 1436          436 A=C S        -
1107 1437          116 C=0          -
1108 1440          1434 PT= 1       -
1109 1441 GTAI30   256 AC EX        GET A CHAR
1110 1442          1072 C=C+1 M     -
1111 1443          1460 CXISA       -
1112 1444          256 AC EX        -
1113 1445          252 AC EX WPT    -
1114 1446          1074 RCR          2    POSITION CHAR
1115 1447          676 A=A-1 S      CHARS FINISHED?
1116 1450          1713 GONC          GTAI30 (1441) NOPE
1117 1451          23 GOTO          **2 (1453) -
1118 1452          1074 RCR          2    -
1119 1453          1352 ? C#0 WPT   -
1120 1454          1763 GONC          *-2 (1452) -
1121 1455          530 M=C          M_ALPHA STRING

```

```

1122 1456          256 AC EX          A[3:0]_F.C. & K.C.
1123 1457          156 AB EX          -
1124 1460          34 PT= 3          -
1125 1461          123 GOTO GTAI20 (1473) B[3:0]_PGMCTR
1126 1462 GTAI22   316 C=B          POSITION PGMCTR
1127 1463          1072 C=C+1 M      -
1128 1464          1460 CXISA        -
1129 1465          1474 RCR 1        -
1130 1466          106 C=0 X         -
1131 1467          374 RCR 10        -
1132 1470          156 AB EX          -
1133 1471          1032 C=A+C M      -
1134 1472          1072 C=C+1 M      -
1135 1473 GTAI20   74 RCR 3          PLACE PGMCTR
1136 1474          352 BC EX WPT     -
1137 1475          1470 C=REGN 12    -
1138 1476          312 C=B WPT      -
1139 1477          1450 REGN=C 12    -
1140 1500          1740 RTN          -
1141
1142 1501 GTAI40   1104 S9= 0        -
1143 1502          742 C=C+C PT      ALBL?
1144 1503          63 GONC GTAI50 (1511) NOFE
1145 1504          1 GSBLNG GETPCA    INCREMENT
1146 1505          0
1146 1506          1 GSBLNG INCADA    -
1147 1507          0
1147 1510          33 GOTO GTAI55 (1513) -
1148 1511 GTAI50   1 GSBLNG GETPCA    A[13]_#CHARS
1148 1512          0
1149 1513 GTAI55   1 GSBLNG NXBYTA    -
1149 1514          0
1150 1515          1474 RCR 1        -
1151 1516          436 A=C S         -
1152 1517          116 C=0          -
1153 1520 GTAI60   676 A=A-1 S       CHARS FINISHED?
1154 1521          117 GOC GTAI70 (1532) YES
1155 1522          530 M=C          -
1156 1523          1 GSBLNG NXBYTA    -
1156 1524          0
1157 1525          1730 CST EX        SHIFT CHAR IN
1158 1526          630 C=M          -
1159 1527          1730 CST EX        -
1160 1530          1074 RCR 2        -
1161 1531          1673 GOTO GTAI60 (1520) -
1162 1532 GTAI70   1 GOSUB RTJLBL     RIGHT JUSTIFY
1162 1533          0
1163 1534          530 M=C          SAVE ALPHA STRING IN M
1164 1535          34 PT= 3          -
1165 1536          1 GSBLNG PUTPC     PLACE PGMCTR
1166 1537          0
1166 1540          156 AB EX          A[3:0]_F.C. & K.C.
1167 1541          1670 C=REGN 14    RESTORE SS0
1168 1542          1530 ST=C        -
1169 1543          1740 RTN          -

```

*

* VIEW ROUTINE

*

```

1175 1544 XAVIEW 1545 CON 01545
1176 1545      674 CON 0674      GOSUB PRT11
1177 1546      1214 ?S7=1      ALPHAMODE?
1178 1547      33 GONC  AVW10 (1552) NO
1179 1550      1314 ?S13=1      RUNNING?
1180 1551      1640 RTN NC      NO - KEYBOARD, ALPHAMODE
1181      DEFAULTDISPLAY IS THE SAME
1182      AS AVIEW - DON'T SET MSGFLG
1183      AVW10
1184 1552      404 S8= 0      SCROLL & NO PROMPT
1185 1553      1110 S9= 1      KEYBOARD ALREADY BEEN RESET
1186 1554      1 GOSUB ARGOUT
1186 1555      0
1187 1556      63 GOTO XVIEWA (1564)
*
1189 1557 XVIEW 1555 CON 01555      GOSUB PRT10
1190 1560      674 CON 0674
1191 1561      316 C=B
1192      ENTRY PRIORT      FOR THE PRINTER
1193      PRIORT      NOTE THE REG TO BE
* VIEWED IS EXPECTED IN C WHEN THE PRT10 LOGIC RETURNS HERE
* (IT WAS IN B WHEN WE WENT OFF TO PRT10)
1196 1562      1 GOSUB DSPCRG
1196 1563      0
1197 1564 XVIEWA 1 GOSUB STMSGF      SET MESSAGE FLAG
1197 1565      0
1198 1566      1614 ?S0=1      DOES A PRINTER EXIST?
1199 1567      57 GOC  XWV10 (1574) YES
1200 1570      474 RCR 8      NO, CK PRINTER ENABLE FLG
1201 1571      1530 ST=C
1202 1572      1014 ?S2=1      DID THE USER SET IT?
1203 1573      547 GOC  STOPS (1647) YES - STOP
1204      XWV10
1205 1574 MSGDLY 1 GOSUB BLINK
1205 1575      0
1206 1576 STMSGF 106 C=0 X
* GOSUB LOSST0 MIGHT BE USED HERE IN PLACE OF THE 4 INST SEQ
* C=0 X, DADD=C, C=REGN 14, ST=C. AN ANALYSIS OF WHO CALLS
* STMSGF AND MSGDLY MUST BE DONE TO SEE IF THEY CAN AFFORD
* ANOTHER SUBROUTINE LEVEL
1211 1577      1160 DADD=C
1212 1600      1670 C=REGN 14
1213 1601      1530 ST=C
1214 1602      210 S5= 1      SET MSGFLAG
1215 1603      323 GOTO RSTMS2 (1635)
1216
1217
1218
*
* RSTSEQ - RESET STATUS BITS AT END OF KEY SEQUENCE
* CLEARS MSGFLG, DATAENTRY, PKSEQ, CATALOGFLAG, SHIFTSET, PSEFLAG
* ALSO CLEARS RUNNING FLAG (S13)
* CHIP 0 MUST BE ENABLED ON ENTRY
* ON EXIT, S0 IS UP AND C CONTAINS A COPY OF THE STATUS REGISTER
* USES ONLY THE C REGISTER AND S0-S7
*
1227      ENTRY RSTSEQ
1228      ENTRY RSTSQ
1229 1604 RSTSEQ 1304 S13= 0      CLEAR RUNNING
1230 1605 RSTSQ 1670 C=REGN 14

```

```

1231 1606      1074 RCR      2
1232 1607      1530 ST=C
1233 1610      1404 S1=      0      LOAD SS 1
                                      CLEAR PKSEQ
1234 1611      1630 C=ST
1235 1612      1574 RCR      12
1236 1613      1530 ST=C      LOAD SS0
1237 1614      1404 S1=      0      CLEAR PAUSING
1238 1615      53 GOTO      RSTMSC (1622)
                                      THESE COMMENTS ACCURATE      RSW 6-13-80

```

*

*

*

* RSTMSC - RESET MISCELLANEOUS STATUS BITS

* RESETS CATALOGFLAG, SHIFT, DATAENTRY, AND MSGFLAG

* ON ENTRY, REG 14 IN C EXCEPT SS0 IN ST, & CHIP 0 ENABLED.

* ON EXIT, STATUS SETS HAVE BEEN STORED BACK TO CHIP 0, CHIP 0 IS ENABLED,

* SS 0 IS UP (AND C HAS A COPY OF THE STATUS SETS).

*

* RSTMS1 - SAME AS RSTMSC EXCEPT SETS UP C AND ST ON ENTRY

* DATOFF - EXACTLY THE SAME AS RSTMS1

* RSTMS0 - SAME AS RSTMS1, EXCEPT CALLS ENCP00 FIRST, THEREBY

* USING AN ADDITIONAL SUBROUTINE LEVEL

*

* USES: C, S0-S7, (NO PT, +0 SUB LEVELS[EXCEPT RSTMS0])

*

```

1255      ENTRY RSTMS0
1256      ENTRY DATOFF
1257      ENTRY RSTMS1
1258      ENTRY RSTMSC
1259 1616 RSTMS0      1 GOSUB ENCP00
1259 1617      0

```

```

1260      DATOFF
1261 1620 RSTMS1      1670 C=REGN 14
1262 1621      1530 ST=C

```

```

1263 1622 RSTMSC      574 RCR      6
1264 1623      1730 CST EX
1265 1624      1404 S1=      0

```

PUT UP SS 3
CLEAR CATALOGFLAG

```

1266 1625      1730 CST EX
1267 1626      374 RCR      10
1268 1627      1730 CST EX

```

PUT UP SS 1
CLEAR SHIFT
CLEAR DATAENTRY

```

1269 1630      1604 S0=      0
1270 1631      1004 S2=      0
1271 1632      1730 CST EX

```

```

1272 1633      1574 RCR      12
1273 1634      204 S5=      0
1274 1635 RSTMS2      1630 C=ST

```

CLEAR MSGFLAG

```

1275 1636      1650 REGN=C 14
1276 1637      1740 RTN
1277
1278
1279

```

```

1280
1281
1282

```

ENTRY XPRMPT

*

* PROMPT - THIS FUNCTION COMBINES AVIEW AND R/S

*

```

1286 1640 XPRMPT      1605 CON      01605
1287 1641      674 CON      0674
1288 1642      1 GOSUB RSTMS0
1288 1643      0

```

GOSUB PRT7

CLEAR MSGFLG (IN CASE WE'RE

```

1289                                     IN ALPHAMODE) & LEAVE
1290                                     SS0 UP
1291 1644          404 S8=      0      SET UP FOR ARGOUT
1292 1645          1214 ?S7=1      ALPHAMODE?
1293 1646          673 GONC      PATCH8 (1735) NO.
1294          P8RTN
1295          ENTRY  STOPS      ERROR CALLS STOPS
1296 1647 STOPS  1670 C=REGH 14      RETRIEVE SS 0
1297 1650          1530 ST=C
1298          ENTRY  STOPSB
1299          STOPSB
1300                                     STOP SUBROUTINE
1301                                     STOP A RUNNING OR PAUSING
1302                                     USER PROGRAM
1303                                     ON ENTRY, SS 0 UP
1304                                     USES 1 SUBROUTINE LEVEL
1305                                     AND C. LEAVES CHIP 0
1306 1651          1404 S1=      0      SELECTED.
1307 1652          1  GOSUB  STOST0      CLEAR PAUSEFLAG
1308 1653          0
1309          PSESTP          ENTRY  PSESTP
1310 1654          1304 S13=      0
1311 1655          1740 RTN
                                     ENTER FROM PAUSE FCN
                                     CLEAR RUNNING FLAG

*
*
*
1315 1656 ALPDEF  660 C=STK      GET RIGHT DEF
1316 1657          1032 C=C+A  M
1317 1660          1460 CXISA      GET LOW 10 BITS
1318 1661          1346 ? C#0  X      IF ZERO DONE CAT
1319 1662          433 GONC      QUTCAT (1725)
1320 1663          34 PT=      3
1321 1664          120 LC      1      BUILD ADR IN ROM 4
1322 1665          674 RCR      11      MOVE TO MANTISSA
1323 1666 END2    256 AC EX      SAVE IN A
1324 1667          1  GOSUB  CLLCODE  ENABLE AND CLEAR LCD
1324 1670          0
1325 1671          256 AC EX
1326 1672          1  GOSUB  PROMF2
1326 1673          0
1327 1674          1  GOSUB  LEFTJ      LEFT JUSTIFY STRING
1327 1675          0
1328 1676 END3    1  GOSUB  ENCF00      TURN OFF LCD
1328 1677          0
1329 1700          1  GOSUB  BLINK
1329 1701          0
1330 1702          1535 CON      01535      SEND DISPLAY TO PRINTER
1331 1703          674 CON      0674      GOSUB PRT12
1332 1704          1  GOSUB  RSTANN
1332 1705          0
1333 1706          114 ?S4=1
1334 1707          1  GOLNC  CNTLOP      SINGLE STEP?
1334 1710          2      IF RUNNING CAT CONTINUE
1335 1711 CLCTMG  1670 C=REGH 14      SET STATUS FOR RTN TO KBD
1336 1712          1530 ST=C
1337 1713          210 SS=      1
1338 1714          1630 C=ST
1339 1715          574 RCR      6
1340 1716          1530 ST=C

```

```

1341 1717          1410 S1=    1
1342 1720 KBD      1630 C=ST
1343 1721          474 RCR      8
1344 1722          1650 REGN=C 14
1345 1723          1 GOLONG NFRKB
1345 1724          2
1346 1725 QUTCAT 1670 C=REGN 14          CATALOG FINISH
1347 1726          1530 ST=C
1348 1727          204 SS=    0
1349 1730          1630 C=ST
1350 1731          574 RCR      6
1351 1732          1530 ST=C
1352 1733          1404 S1=    0
1353 1734          1643 GOTO  KBD    (1720)

```

*
 * PATCH8 - POST-RELEASE FIX TO AVOID PUTTING THE ALPHAREG TO THE LCD
 * AND SETTING MSGFLAG WHEN PROMPT IS EXECUTED IN ALPHAMODE. THIS IS
 * DESIRABLE BECAUSE THE ALPHAREG IS THE DEFAULTDISPLAY IN ALPHAMODE.
 *

```

1359 1735 PATCH8  1 GOSUB  ARGOUT          PUT ALPHAREG TO LCD
1359 1736          0
1360 1737          1 GOSUB  STMSGF          SET MSGFLG
1360 1740          0
1361 1741          1063 GOTO  P8RTH  (1647)

```

*
 * PATCH4 - THIS POST-RELEASE PATCH SPEEDS UP THE EXECUTION OF THE
 * RUN PORTION OF R/S
 *

```

1366          ENTRY  PACH4
1367 1742 PACH4  460 LDI          SET UP 100MS WAIT
1368 1743          247 CON      167
1369 1744 PTCH4A 1710 RST KB          IS THE KEY STILL DOWN?
1370 1745          1714 CHK KB
1371 1746          1 GOLNC  XRS45          NO, GO RUN!
1371 1747          2
1372 1750          1146 C=C-1  X          TIME OUT OVER?
1373 1751          1733 GONC  PTCH4A (1744) NO, KEEP CHECKING THE KEY
1374 1752          1 GOLONG LINNUM          DISPLAY THE STARTING STEP
1374 1753          2

```

*
 * PACH10 - POST RELEASE PATCH TO FIX A BUG IN "SF IND NN"
 *

```

1378          ENTRY  PACH10
1379 1754 PACH10  1 GOSUB  BCDBIN
1379 1755          0
1380 1756          1366 ? C#0  XS          ADDR>255?
1381 1757          1 GOLC   ERRNE          YES
1381 1760          3
1382 1761          1034 PT=    2          RESTORE 1ST BYTE
1383 1762          230 C=C          OF FC TO C3:2
1384 1763          1 GOLONG P10RTH
1384 1764          2

```

*
 * PACH11 - POST-RELEASE FIX TO DISPLAY DRIVER SYNCHRONIZATION
 * PROBLEM, 3/26/79. THE TWO DISPLAY DRIVER CHIPS RE-SYNCHRONIZE
 * EACH TIME THE CPU COMES WIDE AWAKE, NO MATTER WHETHER FROM LIGHT
 * SLEEP OR DEEP SLEEP. EACH TIME THE C REGISTER CONTAINS BOTH ONES
 * AND ZEROES, THE DISPLAY DRIVERS SORT THEMSELVES OUT. THIS PROCESS
 * CONTINUES UNTIL A DISPLAY READ INSTRUCTION IS EXECUTED. HOWEVER,
 * IF THE DATA LINE FLOATS WHILE THE DISPLAY DRIVERS ARE TRYING TO

* SORT THEMSELVES OUT, AND IF THE LEVEL ON THE DATA LINE DRIFTS,
* THE DISPLAY DRIVERS MAY GET CONFUSED AND TANK THE SYSTEM.
* THIS PATCH ENSURES THAT THE DISPLAY DRIVERS GET SYNCHRONIZED AND
* THEN DISABLE THE SYNCHRONIZATION LOGIC BEFORE ANY MICROCODE
* FLOATS THE DATA LINE (AS BY READING FROM A NON-EXISTENT DATA
* STORAGE CHIP IN CHKADR OR ENDEND).

*
1400 ENTRY PACH11
1401 1765 PACH11 460 LDI
1402 1766 1375 CON2 47 13
1403 1767 1160 DADD=C
1404 1770 1760 PFAD=C
1405 1771 270 FLLDC
1406 1772 1 GOLONG.MENCHK
1406 1773 2
ENABLE NON-EXISTENT DATA CHIP 2FD
ENABLE DISPLAY
NON-DESTRUCTIVE READ

*
* PACH12 - POST RELEASE FIX TO DECOMPILE ON WAKEUP WHEN MACHINE GOES
* TO SLEEP IN PROGRAM MODE. DEC 10/20/79

*
1411 ENTRY PACH12
1412 1774 PACH12 106 C=0 X
1413 1775 1650 REGN=C 14
1414 1776 1 GOLONG.DECMPL
1414 1777 2
1415 FILLTO END
1416 END

ERRORS : 0

SYMBOL TABLE

ADRFCH	4	-			
ADRGSE	114	-	111	56	
ALLOK	1315	-	1306		
ALPDEF	1656	-			
AVW10	1552	-	1547		
BCDBIN	1343	-			
BIGBRC	117	-	163		
CHKRPC	1042	-			
CLCTMG	1711	-			
COLDST	1062	-	1034	1021	763
CONFLG	1301	-	1273		
DATOFF	1620	-			
DBL	1330	-	1332		
DROPST	344	-	336		
DROWSY	540	-	472		
DRSY05	541	-			
DRSY10	552	-	546		
DRSY20	560	-	553		
DRSY25	563	-	626	557	551 544
DRSY26	567	-	570		
DRSY30	573	-	564		
DRSY50	620	-	671		
DRSY51	624	-			
DRWSYL	472	-	407		
DSWKUP	655	-			
END2	1666	-			
END3	1676	-			
ERRIGN	273	-	343	325	
ERRNE	1340	-	1367	1314	1310
ERROF	242	-	274		
FCHRTN	40	-	27		
FILLXL	352	-	326	320	261
FILLY	256	-	272		
FLAGS	1271	-	1264	1261	
GBYTR0	524	-	500		
GBYTR1	521	-	501		
GBYTR2	516	-	502		
GBYTR3	513	-	503		
GBYTR4	510	-	504		
GBYTR5	505	-			
GOTINT	1370	-	1364	1360	
GTAI10	1420	-	1412	1410	
GTAI20	1473	-	1461		
GTAI22	1462	-	1427		
GTAI26	1432	-			
GTAI30	1441	-	1450		
GTAI40	1501	-	1421		
GTAI50	1511	-	1503		
GTAI55	1513	-	1510		
GTAI60	1520	-	1531		
GTAI70	1532	-	1521		
GTAINC	1404	-			
IGNKEY	433	-	426		
ILOOP	1125	-	1132		
INCG1	1213	-	1210		
INCG2	1225	-	1215		

INCG3	1243	-	1226						
INCGT2	1206	-							
INTINT	1373	-							
IOSERV	404	-	401						
KBD	1720	-	1734						
LOWBRT	400	-	372	370					
LSWKUP	600	-							
MATH	213	-	205						
MENCHK	1005	-							
MSGDLY	1574	-							
NEXROM	465	-	443						
NFRC	361	-	311	305	301				
NFRENT	304	-							
NFRFST	367	-							
NFRKB	307	-							
NFRKB1	306	-							
NFRNC	245	-							
NFRNIO	406	-	403						
NFRPR	356	-							
NFRPU	360	-							
NFRSIG	302	-							
NFRX	314	-	312						
NFRXY	332	-							
NOPRT	533	-	530						
NXBEND	527	-	523	520	515	512	507	464	
NXROM1	526	-	471						
NXTBT1	456	-	446						
OFFXFR	653	-							
OVRSTK	30	-	45	21	17	15			
P10RTN	1254	-							
P8RTN	1647	-	1741						
PACH10	1754	-							
PACH11	1765	-							
PACH12	1774	-							
PACH4	1742	-							
PATCH8	1735	-	1646						
PAUS10	634	-	643						
PAUSLP	627	-	577						
PCTOC	327	-							
PR10RT	1562	-							
PSESTP	1654	-							
PSTDBL	1331	-	1327						
PTCH4A	1744	-	1751						
QUTCAT	1725	-	1662						
REGADR	105	-	101						
ROW10	1246	-							
ROW7	177	-							
ROUTBL	140	-							
RST05	233	-							
RST10	235	-	240						
RSTKB	230	-	232						
RSTMS0	1616	-							
RSTMS1	1620	-							
RSTMS2	1635	-	1603						
RSTMSC	1622	-	1615						
RSTSEQ	1604	-							
RSTSQ	1605	-							
RUNING	410	-							
RUHMK	435	-	411						
SHF8	1324	-	1326						

NOMAS

NOT Manufacturer Supported
recipient agrees NOT to contact manufacturer

STMSGF	1576	-				
STOPS	1647	-	1573			
STOPSB	1651	-				
STOSTO	473	-				
TBLGBR	500	-				
TONETC	161	-	113			
TONSTF	124	-				
WKUP10	604	-	613			
WKUP20	646	-	635	605		
WKUP21	647	-				
WKUP25	672	-	663			
WKUP30	716	-	730			
WKUP40	717	-	736			
WKUP50	737	-	726	720		
WKUP60	764	-	753			
WKUP70	765	-				
WKUP80	777	-				
X<>ROW	46	-				
XAVIEW	1544	-				
XBAD	262	-	251			
XCUTB1	221	-				
XCUTE	533	-				
XCUTE8	220	-	200	176	150	147
XPRMPT	1640	-				
XROW0	57	-	140			
XROW1	164	-	141			
XROW10	61	-	152			
XROW11	63	-	153			
XROW12	65	-	154			
XROW13	67	-	155			
XROW14	71	-	156			
XROW2	73	-	142			
XROW3	73	-	143			
XROW4	206	-	144			
XROW5	201	-	145			
XROW6	201	-	146			
XROW9	102	-	151			
XVIEW	1557	-				
XVIEWA	1564	-	1556			
XVM10	1574	-	1567			
YBAD	264	-	255			

ENTRY TABLE

ADRFCH	4	-
ALLOK	1315	-
ALPDEF	1656	-
BCDBIN	1343	-
BIGBRC	117	-
CHKRPC	1042	-
CLCTMG	1711	-
COLDST	1062	-
DATOFF	1620	-
DROPST	344	-
DROWSY	540	-
DRSY05	541	-
DRSY25	563	-
DRSY50	620	-
DRSY51	624	-
DSWKUP	655	-
END2	1666	-
END3	1676	-
ERRIGN	273	-
ERRNE	1340	-
ERROF	242	-
FILLXL	352	-
GOTINT	1370	-
GTAI40	1501	-
GTAINC	1404	-
INCGT2	1206	-
INTINT	1373	-
LSWKUP	600	-
MEMCHK	1005	-
MSGDLY	1574	-
NFRC	361	-
NFRENT	304	-
NFRFST	367	-
NFRKB	307	-
NFRKB1	306	-
NFRNC	245	-
NFRNIO	406	-
NFRPR	356	-
NFRPU	360	-
NFRSIG	302	-
NFRX	314	-
NFRXY	332	-
NOPRT	533	-
P10RTH	1254	-
PACH10	1754	-
PACH11	1765	-
PACH12	1774	-
PACH4	1742	-
PR10RT	1562	-
PSESTP	1654	-
OUTCAT	1725	-
ROW10	1246	-
RST05	233	-
RSTKB	230	-
RSTMS0	1616	-
RSTMS1	1620	-

RSTMSC	1622	-
RSTSEQ	1604	-
RSTSQ	1605	-
RUNING	410	-
RUNNK	435	-
STMSGF	1576	-
STOPS	1647	-
STOPSB	1651	-
STOSTO	473	-
TONSTF	124	-
WKUP10	604	-
WKUP21	647	-
WKUP25	672	-
WKUP70	765	-
WKUP80	777	-
X<>ROW	46	-
XAVIEW	1544	-
XCUTB1	221	-
XCUTE	533	-
XCUTB8	220	-
XPRMPT	1640	-
XROW1	164	-
XVIEW	1557	-

EXTERNAL REFERENCES

ADRFCH	114	127	1275
ADRFCH	115	130	1276
ANNOUT	376	541	624
ANNOUT	377	542	625
ARGOUT	555	1554	1735
ARGOUT	556	1555	1736
BCDBIN	42	131	1754
BCDBIN	43	132	1755
BIGBRC	1267	1336	
BIGBRC	1270	1337	
BLINK	1574	1700	
BLINK	1575	1701	
CHK#S	211		
CHK#S	212		
CHK#S2	214		
CHK#S2	215		
CHKADR	35		
CHKADR	36		
CLLCDE	1667		
CLLCDE	1670		
CNTLOP	1707		
CNTLOP	1710		
DECMPL	1776		
DECMPL	1777		
DERUN	170		
DERUN	171		
DFRST8	547		
DFRST8	550		
DSPCRG	561	1562	
DSPCRG	562	1563	
DSWKUP	2		
DSWKUP	3		
ENCP00	1616	1676	
ENCP00	1617	1677	
ENLCD	620	1111	
ENLCD	621	1112	
ERRAD	1345		
ERRAD	1346		
ERRDE	136		
ERRDE	137		
ERRNE	1757		
ERRNE	1760		
ERROR	242	1340	
ERROR	243	1341	
GETPCA	1504	1511	
GETPCA	1505	1512	
GTAINC	172		
GTAINC	173		
INCADA	1506		
INCADA	1507		
INCGT2	50	102	1246
INCGT2	51	103	1247
INTINT	1371		
INTINT	1372		
IORUN	404		
IORUN	405		

LEFTJ	1674				
LEFTJ	1675				
LINNUM	1752				
LINNUM	1753				
LSWKUP	0				
LSWKUP	1				
MEMCHK	1772				
MEMCHK	1773				
MSGA	1106				
MSGA	1107				
MSGNL	1110				
MSGNE	1342				
MSGOF	244				
NFRC	771				
NFRC	772				
NFRKB	431	1723			
NFRKB	432	1724			
NXBYTA	1513	1523			
NXBYTA	1514	1524			
OFF	421	653			
OFF	422	654			
OVFL10	245	252	262	314	332
OVFL10	246	253	263	315	333
P10RTN	1763				
P10RTN	1764				
PACH10	1277				
PACH10	1300				
PACH11	602	660			
PACH11	603	661			
PACH12	673				
PACH12	674				
PARSE	651				
PARSE	652				
PGMAON	627				
PGMAON	630				
PKIOAS	744				
PKIOAS	745				
PROMF2	1672				
PROMF2	1673				
PUTPC	1536				
PUTPC	1537				
PUTPCX	1060				
PUTPCX	1061				
RMCK05	640				
RMCK05	641				
ROMCHK	610	666	742	1172	
ROMCHK	611	667	743	1173	
ROW0	57				
ROW0	60				
ROW10	61				
ROW10	62				
ROW11	63				
ROW11	64				
ROW12	65				
ROW12	66				
RSTANN	1704				
RSTANN	1705				
RSTKB	307	571	750	1113	
RSTKB	310	572	751	1114	
RSTNS0	1642				

RSTMS0	1643		
RSTSEQ	427		
RSTSEQ	430		
RTJLBL	1532		
RTJLBL	1533		
RUN	644	1003	
RUN	645	1004	
STMSGF	1564	1737	
STMSGF	1565	1740	
STOST0	374	574	1652
STOST0	375	575	1653
TEXT	157		
TEXT	160		
ZONE7X	1000		
ZONE7X	1001		
TONSTF	161		
TONSTF	162		
WKUP70	1204		
WKUP70	1205		
XCUTB1	122		
XCUTB1	123		
XGTO	67		
XGTO	70		
XROM	1255		
XROM	1256		
XRS45	1746		
XRS45	1747		
XXEQ	71		
XXEQ	72		

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES 02000-3777

3	FILE	CH18
4	ENTRY	LDSSST0
5	ENTRY	OFSHFT
6	ENTRY	ANNOUT
7	ENTRY	ANN+14
8	ENTRY	RSTANN
9	ENTRY	AFORMAT
10	ENTRY	BLANK
11	ENTRY	CHKFUL
12	ENTRY	CHRLCD
13	ENTRY	CPCMHD
14	ENTRY	DEROW
15	ENTRY	DERW00
16	ENTRY	DFILLF
17	ENTRY	DF060
18	ENTRY	DF150
19	ENTRY	DF160
20	ENTRY	DF200
21	ENTRY	DFKBCK
22	ENTRY	DFRST8
23	ENTRY	DFRST9
24	ENTRY	GENNUM
25	ENTRY	MEMLFT

26	ENTRY	PROMF1
27	ENTRY	PROMF2
28	ENTRY	PROMFC
29	ENTRY	ROMHED
30	ENTRY	ROMH05
31	ENTRY	ROMH35
32	ENTRY	RW0141
33	ENTRY	ROW120
34	ENTRY	ROW933
35	ENTRY	ROW940
36	ENTRY	RW0110
37	ENTRY	TXRW10
38	ENTRY	TXTR0W
39	ENTRY	TXSTR
40	ENTRY	XMSGPR

*

* ROW JUMP TABLE

*

44	0	213 GOTO	ROW0	(	21)	
45	1	243 GOTO	ROW1	(	25)	
46	2	253 GOTO	ROW2	(	27)	
47	3	333 GOTO	ROW3	(	36)	
48	4	263 GOTO	ROW4-8	(	32)	
49	5	253 GOTO	ROW4-8	(	32)	
50	6	243 GOTO	ROW4-8	(	32)	
51	7	233 GOTO	ROW4-8	(	32)	
52	10	223 GOTO	ROW4-8	(	32)	
53	11	373 GOTO	ROW09	(	50)	
54	12	643 GOTO	ROW10	(	76)	
55	13	363 GOTO	ROW11	(	51)	
56	14	403 GOTO	ROW12	(	54)	
57	15	453 GOTO	ROW1314	(	62)	
58	16	443 GOTO	ROW1314	(	62)	
59	17	1 GOLONG	TXTR0W			
59	20	2				
60	21 ROW0	460 LDI				
61	22	317 CON2	12	15	PROMPT STRING IN C,F	
62	23 ROW010	646 A=A-1	X		OPERAND MINUS ONE	
63		LEGAL				
64	24	143 GOTO	DF120	(	40)	
65	25 ROW1	1 GOLONG	DEROW			THIS IS A DIGIT ENTRY ROW
65	26	2				
66	27 ROW2	460 LDI				
67	30	220 CON2	9	0	PROMPT STRING IN 9,0	
68	31	73 GOTO	DF120	(	40)	
69	32 ROW4-8	1 GOSUB	PROMFC			
69	33	0				
70	34	1 GOLONG	DF150			
70	35	2				
71	36 ROW3	460 LDI				
72	37	221 CON2	9	1	PROMPT STRINE IN 9,1	
73	40 DF120	2 A=0	PT		A(1) = 0	
74	41	206 B=A	X		SAVE THE OPERAND IN B	
75	42	406 A=C	X			
76	43	1 GOSUB	PROMFC		OUTPUT PROMPT STRING	
76	44	0				
77	45	146 AB EX	X		A.X _ OPERAND	
* NEXT INSTRUCTION		(S0=0) MAY NOT BE NECESSARY.				
79	46	1604 S0=	0		SAY TWO DIGITS OPERAND	
80	47	743 GOTO	ROW931	(	143)	

```

81 50 ROW09 563 GOTO ROW9 ( 126 )
82 51 ROW11 460 LDI
83 52 320 CON2 13 0 PROMPT STRING IN 13,0
84 53 1503 GOTO ROW010 ( 23 )
85 54 ROW12 460 LDI
86 55 316 CON2 12 14
87 56 1406 ? AKC X IS IF LBLNN ?OR X<>NN?
88 57 543 GONC ROW910 ( 133 ) YES
89 60 1 GOLONG ROW120
89 61 2
90 62 R01314 1634 PT= 0
91 63 2 A=0 PT
92 64 1 GOSUB PRONFC
92 65 0
93 66 1 GOSUB NBYTA0 SKIP ONE BYTE(THREE BYTE FC)
93 67 0
94 70 1 GOSUB NXTBYT
94 71 0
95 72 1730 CST EX
96 73 1204 S7= 0
97 74 1730 CST EX
98 75 433 GOTO ROW930 ( 140 )
99 76 ROW10 460 LDI
100 77 250 CON2 10 8 TEST FOT XECROM FC
101 100 1406 ? AKC X IS IT A XECROM FC ?
102 101 1 GOLC XECROM YES
102 102 3
103 103 460 LDI
104 104 256 CON2 10 14
105 105 1406 ? AKC X IS IT A XEQ/GTO IND ?
106 106 257 GOC ROW910 ( 133 ) NO
107 107 1 GOSUB NBYTAB GET OPERAND
107 110 0
108 111 1730 CST EX
109 112 1 GOSUB ENLCD
109 113 0
110 114 1214 ?S7=1 IS IT A XEQ ?
111 115 43 GONC **4 ( 121 ) NO
112 116 460 LDI
113 117 340 CON2 14 0 LOAD XEQ FC
114 120 33 GOTO **3 ( 123 )
115 121 460 LDI
116 122 320 CON2 13 0 LOAD GTO FC
117 123 1 GOSUB PROMF1
117 124 0
118 125 223 GOTO ROW933 ( 147 )

*
* NUMERICAL OPERAND
* ROW 9
*
123 126 ROW9 1610 S0= 1
124 127 460 LDI
125 130 234 CON2 9 12 TEST FOR 1 OR 2 DIGIT OPERAND
126 131 1406 ? AKC X 1 DIGIT OPERAND ?
127 132 23 GONC **2 ( 134 ) YES

*
* NUMERICAL OPERAND
* B[3:0] HAS ADDR POINT TO ONE BYTE BEFORE OPERAND
* IF S0=1 MEANS 1 DIGIT OPERAND
* IF S0=0 MEANS 2 DIGITS OPERAND

```

```

*
134 133 ROW910 1604 S0= 0
135 134 1 GOSUB PROMFC PROMPT THE FUNCTION FIRST
135 135 0
136 136 1 GOSUB NBYTA0 LOAD OPERAND
136 137 0
137 140 ROW930 406 A=C X SAVE OPERAND IN A TEMP.
138 141 1 GOSUB ENLCD ENABLE LCD CHIP
138 142 0
139 143 ROW931 246 AC EX X LOAD OPERAND BACK TO C.X
140 144 1730 CST EX MOVE OPERAND TO STATUS BITS
141 145 1214 ?S7=1 INDIRECT ?
142 146 163 GONC ROW935 ( 164 ) NO
143 147 ROW933 1204 S7= 0
144 150 1730 CST EX
145 151 1604 S0= 0 TWO DIGITS OPERAND
146 152 346 BC EX X
147 153 1 GOSUB MESSL
147 154 0
148 155 11 CON 9 I
149 156 16 CON 14 N
150 157 1004 CON @1004 D
151 160 1 GOSUB BLANK OUTPUT A BLANK
151 161 0
152 162 146 AB EX X
153 163 33 GOTO ROW936 ( 166 )
154 164 ROW935 1730 CST EX
155 165 406 A=C X A.(1:0) _ OPERAND
156 166 ROW936 26 A=0 XS
157 167 460 LDI
158 170 146 CON 102
159 171 1406 ? A<C X NUMERICAL OPERAND ?
160 172 153 GONC ROW940 ( 207 ) NO
161 173 36 A=0 S
162 174 576 A=A+1 S
163 175 1614 ?S0=1
164 176 27 GOC **2 ( 200 ) YES
165 177 576 A=A+1 S
166 LEGAL
167 200 1 GOSUB GENHUM OUTPUT OPERAND
167 201 0

```

```

*
* DFILLF EXIT POINT

```

```

*
171 202 DF150 1414 ?S1=1 DISPLAY FULL ?
172 203 1 GSUBNO.LEFTJ NO, LEFT JUSTIFY
172 204 0
173 205 DF160 1 GOLONG.LDSST0 ENABLE CHIP 0
173 206 2 & PUT UP S0
174
175 207 ROW940 460 LDI
176 210 160 CON 112
177 211 1406 ? A<C X CAPITAL A,B,C,D,E ?
178 212 307 GOC CAPABC ( 242 ) YES
179 213 1546 ? A#C X IS IT A T'S ?
180 214 213 GONC RT ( 235 ) YES
181 215 460 LDI
182 216 164 CON 116
183 217 1546 ? A#C X IS IT A LSTX ?
184 220 323 GONC RL ( 252 ) YES

```

```

185 221      1406 ? A<C  X      IS IT SMALL A,B,C,D,E ?
186 222      233 GONC  SMLABC ( 245 ) YES
187 223      460 LDI
188 224      161 CON    113      IT IS A X,Y OR Z
189 225      1106 C=A-C  X      BUT IN THE REVERSE ORDER
190 226      1474 RCR    1      C.S _ OFFSET
191 227      460 LDI
192 230      32 CON    26      LOAD A Z'S
193 231 ROW945 1176 C=C-1  S
194 232      67 GOC    ROW960 ( 240 )
195 233      1146 C=C-1  X
196      LEGAL
197 234      1753 GOTO   ROW945 ( 231 )

```

```

*
199 235 RT      460 LDI
200 236      134 CON    92
201 237 ROW950 1106 C=A-C  X
202      LEGAL
203 240 ROW960 1750 SLSABC
204 241      1413 GOTO   DF150 ( 202 )
205 242 CAPABC  460 LDI
206 243      145 CON    101
207 244      1733 GOTO   ROW950 ( 237 )
208 245 SMLABC  460 LDI
209 246      172 CON    122
210 247      1106 C=A-C  X
211 250      1066 C=C+1  XS
212      LEGAL
213 251      1673 GOTO   ROW960 ( 240 )
214 252 RL      460 LDI
215 253      150 CON    104
216 254      1633 GOTO   ROW950 ( 237 )

```

NOMAS
 Not Manufacturer Supported
 recipient agrees NOT to contact manufacturer

```

*
*
* ROW 1 - INCLUDING DIGIT ENTRY AND AGTO, AXEQ
* AI2:01 HAS THE FUNCTION CODE. BI3:01 POINTING 1ST BYTE OF
* DIGIT ENTRY STRING, IF ITS A DIGIT ENTRY FC.
*

```

```

223      DEROW
224 255      460 LDI
225 256      35 CON2    1      13
226 257      1406 ? A<C  X      IS IT A DIGIT ENTRY FC ?
227 260      713 GONC  RW0110 ( 351 ) NO, EITHER AGTO OR AXEQ
228 261      473 GOTO   DERW70 ( 330 )

```

```

*
* DIGIT ENTRY START HERE
*

```

```

232 262 DERW00  460 LDI
233 263      32 CON2    1      10
234 264      1406 ? A<C  X      IS IT A DIGIT ?
235 265      317 GOC    DERW50 ( 316 ) YES
236 266      1546 ? A#C  X      IS IT A D.P.?
237 267      147 GOC    DERW10 ( 303 ) NO
238 270      1670 FRSABC
239 271      1730 CST EX
240 272      510 S6=    1      SET D.P.
241 273      534 PT=    6      CHECK FOR EUROPEAN NOTATION
242 274      242 AC EX  PT
243 275      742 C=C+C  PT
244 276      27 GOC    DERW05 ( 300 )

```

```

245 277          1210 S7=      1          SET CONMA
246 300 DERM05 1730 CST EX
247 301          1750 SLSABC
248 302          213 GOTO     DERM60 ( 323)
249 303 DERM10 1046 C=C+1    X
250 304          1546 ? A#C   X          IS IT A EEX ?
251 305          67 GOC       DERM20 ( 313) NO
252 306          1 GOSUB     BLANK
252 307          0
253 310          460 LDI
254 311          5 CON       @05          "E"
255 312          73 GOTO     DERM55 ( 321)
256 313 DERM20 460 LDI          IT GOT BE A CHS
257 314          55 CON       @55
258 315          43 GOTO     DERM55 ( 321)
259 316 DERM50 246 AC EX     X
260 317          1434 PT=     1
261 320          320 LC       3
262 321 DERM55 1 GOSUB     CHRLCD
262 322          0
263 323 DERM60 1 GOSUB     NBYTA0          ENABLE CHIP 0
263 324          0
264          & GET NEXT BYTE
265 325          156 AB EX     PUT THE PCMPTR BACK TO B
266 326          126 C=0      XS
267 327          406 A=C       X          A.X _ NEXT BYTE
268 330 DERM70 1 GOSUB     ENLCD
268 331          0
269 332          460 LDI
270 333          35 CON2      1          13
271 334          1434 PT=     1
272 335          1542 ? A#C   PT          IS THIS BYTE A ROW 1 FC ?
273 336          37 GOC       DF190 ( 341) NO
274 337          1406 ? A<C   X          IS IT A DIGIT ENTRY FC ?
275 340          1227 GOC     DERM00 ( 262) YES
276 341 DF190 414 ?S8=1      PROMPT ?
277 342          53 GONC      DF200 ( 347) NO
278 343          460 LDI
279 344          37 CON       @37
280 345          1 GOSUB     CHRLCD
280 346          0
281 347 DF200 1 GOLONG DF150
281 350          2
282 351 RW0110 1746 A SL     X          CONVERT FC FROM 1D TO D0
283 352          26 A=0       XS          OR FROM 1E TO E0
284 353          1 GOSUB     PRGMFC
284 354          0
285 355          1 GOSUB     NBYTA0
285 356          0
286 357          156 AB EX
287 360 RW0140 406 A=C       X
288 361 RW0141 404 S8=      0

```

```

*
* TXTSTR - TEXT STRING
* A[0] HAS THE LENGTH OF THE STRING. B[3:0] POINTING ONE BYTE
* BEFORE 1ST CHAR.
* IF S2=1 ALPHA STRING IS KNOWN IN ROM
* IF S2=0, STRING IS IN RAM
*
* TXTROM - SETS S2 AND DROPS INTO TXTSTR

```

```

*
* TXRW10 - IDENTICAL TO TXTSTR
*
* TXTROW - COPIES S10 (ROMFLAG) INTO S2 AND FALLS INTO TXTSTR
*
302 362 TXTR0W 1004 S2= 0
303 363 314 ?S10=1 ROMFLAG?
304 364 23 G0NC TXTSTR ( 366 ) NO
305 ENTRY TXTR0M
306 365 TXTR0M 1010 S2= 1 YES
307 TXTSTR
308 366 TXRW10 1434 PT= 1
309 367 2 A=0 PT
310 370 246 AC EX X
311 371 132 C=0 M
312 372 126 C=0 XS
313 373 674 RCR 11
314 374 432 A=C M A.M _ CHAR COUNTER
315 375 1 GOSUB ENLCD
315 376 0
316 377 1670 FRSABC
317 400 460 LDI
318 401 407 CON @407
319 402 1750 SLSABC
320 403 TXRW30 672 A=A-1 M ALL DONE ?
321 404 1357 GOC DF190 ( 341 ) YES
322 405 156 AB EX W
323 406 1 GOSUB ENCF00
323 407 0
324 410 34 PT= 3 SET UP FOR NXBYTA
325 411 1014 ?S2=1 ROM?
326 412 1 GSUBC NXBYTO YES
326 413 1
327 414 1014 ?S2=1 SAME QUESTION
328 415 1 GSUBNC NXBYTA NO
328 416 0
329 417 156 AB EX
330 420 406 A=C X A.X _ CHAR
331 421 1 GOSUB ENLCD
331 422 0
332 423 246 AC EX X
333 424 1 GOSUB ASCLCD
333 425 0
334 426 1 GOSUB CHKFUL SEE IF LCD FULL
334 427 0
335 430 1533 GOTO TXRW30 ( 403 )
*
337 431 ROW120 156 AB EX
338 432 1 GOSUB INCAD
338 433 0
339 434 1 GOSUB NXTBYT LOAD OPERAND
339 435 0
340 436 216 B=A W SAVE PC IN B
341 437 406 A=C X
342 440 1 GOSUB ENLCD
342 441 0
343 442 1434 PT= 1
344 443 542 A=A+1 PT IS IT LBL ?
345 444 163 GONC ROW122 ( 462 ) NO, ITS A END
346 445 460 LDI

```

```

347 446      317 CON2      12      15      LOAD LBL FC
348 447      1 GOSUB      PROMF1      PROMPT THE FUNCTION
348 450      0
349 451      1 GOSUB      ENCF00      ENABLE CHIP 0
349 452      0
350 453      156 AB EX
351 454      1 GOSUB      INCAD
351 455      0
352 456      156 AB EX
353 457      656 A=A-1      CHAR COUNTER -1 (SKIP KC)
354      LEGAL
355 460      1 GOLONG:RW0141
355 461      2
356 462 ROW122 252 AC EX      WPT
357 463      1142 C=C-1      PT      RESTORE THE "END"
358 464      1530 ST=C
359 465      460 LDI
360 466      300 CON2      12      0      PROMPT "END"
361 467      1 GOSUB      PROMF1
361 470      0
362 471      314 ?S10=1      ARE WE IN ROM ?
363 472      357 GOC      ROW125 ( 527) YES, PROMPT "END" ONLY
364 473      214 ?S5=1      FINAL END ?
365 474      333 GONC      ROW125 ( 527) NO
366 475      1340 DISOFF
367 476      1670 RABCR
368 477      1670 RABCR
369 500      460 LDI
370 501      104 CON      Q104
371 502      1750 SLSABC
372 503      1 GOSUB      REGLEFT
372 504      0
373 505      406 A=C      X
374 506      36 A=0      S
375 507      1 GOSUB      ENLCD
375 510      0
376 511      1 GOSUB      GENNUM
376 512      0
377 513      460 LDI
378 514      40 CON      Q40
379 515      1750 SLSABC
380 516      1770 RABCL      READ IN LEFT MOST CHAR
381 517      1634 PT=      0
382 520      1342 ? C#0      PT      IS IT A BLANK ?
383 521      23 GONC      **2      ( 523) YES, THROW IT AWAY
384 522      1650 SRSABC      IS AN "E", PUT IT BACK
385 523      460 LDI
386 524      140 CON      Q140      LOAD A DOT
387 525      1650 SRSABC      SHIFT IN LEFT END
388 526      1440 DISTOG
389 527 ROW125 1404 S1=      0
390 530      523 GOTO      DF040 ( 602)

```

```

*
* DFILLF _ DISPLAY ONE PROGRAM STEP
*
* CALLING SEQUENCE:
*   PGMPTR _ POINT TO LAST BYTE OF PREVIOUS STEP
*   GOSUB DFILLF
*   IF "PRIVATE", DISPLAY "PRIVATE" AND RETURN
*   ELSE DISPLAY ONE LINE OF PROGRAM MEMORY
*

```


* FOUR ENTRY POINTS :
 * 1. DFILLF - NORMAL ENTRY
 * 2. DFRST9 - RESET S9 REMEMBER KEYBOARD NOT BEEN RESET YET
 * RESET S8 SAY NO-PROMPT & SCROLL
 * 3. DFRST8 - ONLY RESET S8
 * 4. DFKBCK - SPEND APPROXIMATELY 100 MILLISEC CHECKING FOR
 * KEY UP BEFORE DROPPING INTO DFRST9
 * USED S0,S1,S2, A,B,C. ASSUMED NOTHING.
 * NOT TRUE! CALLS LINNUM. SEE COMMENTS ON LINNUM.
 * USES AT LEAST TWO SUBROUTINE LEVELS
 * RETURN WITH CHIP ENABLE & STATUS SET #0 ENABLE
 * EXCEPT ON THE KEY UP PATH OUT OF DFKBCK THE CHIP ENABLE AND
 * STATUS SET ARE UNCHANGED.
 *

```

413 531 DFKBCK 460 LDI
414 532          310 CON      200
415 533          1110 S9=      1          ASSUME KB WILL BE RESET
416 534 DF010 1710 RST KB
417 535          1714 CHK KB
418 536          1640 RTN NC
419 537          1146 C=C-1 X
420 540          1743 GONC DF010 ( 534)
421 541 DFRST9 1104 S9=      0          SAY KEYBOARD NOT RESET YET
422 542 DFRST8 404 S8=      0          SAY NO PROMPT, SCROLLING
423 543 DFILLF 1404 S1=      0          SAY LCD NOT FULL YET
424 544          1604 S0=      0          ASSUME 2D OPERAND
425 545          1 GOSUB ENCP00
425 546          0
426 547          1 GOSUB LINNUM          LOAD LINE #
426 550          0
427 551          406 A=C      X          A.X _ LINE #
428 552          206 B=A      X          SAVE LINE # IN B.X
429 553          1514 ?S12=1          PRIVATE?
430 554          53 GONC DF030 ( 561) NOT PRIVATE
431 555 XMSGPR 1 GOSUB MSG
431 556          0
432 557          0 XDEF MSGPR          SAY PRIVATE
433 560          223 GOTO DF040 ( 602)
434 561 DF030 1 GOSUB CLLCDE
434 562          0
435 563          36 A=0      S
436 564          1 GOSUB GENNUM          OUTPUT LINE #
436 565          0
437 566          1306 ? B#0 X          LINE# = 0 ?
438 567          157 GOC DF050 ( 604) NO
439 570          314 ?S10=1          ARE WE IN ROM
440 571          117 GOC DF040 ( 602) YES, NO PROMPT FOR LINE#=0
441 572          1 GOSUB REGLEFT
441 573          0
442 574          406 A=C      X          A.X _ MEN LEFT
443 575          36 A=0      S
444 576          1 GOSUB ENLCD
444 577          0
445 600          1 GOSUB GENNUM
445 601          0
446 602 DF040 1 GOLONG DF150
446 603          2
447 604 DF050 460 LDI
448 605          40 CON      @40
449 606          1750 SLSABC          OUTPUT A BLANK

```

```

450 607 DF060      1 GOSUB ENCF00      ENABLE CHIP 0
450 610              0
451 611          1670 C=REGN 14      SET UP FOR D.P.(COMMA) CHECK
452 612          416 A=C
453 613              1 GOSUB GETPC      LOAD PROGRAM POINTER
453 614              0
454 615 DF100      1 GOSUB NXTBYT      NEXT BYTE
454 616              0
455 617          1434 PT= 1
456 620          1352 ? C#0 WPT      IS IT A NULL ?
457 621          1743 GONC DF100 < 615> YES, SKIP IT
458 622          216 B=A      SAVE THE PGMPTR IN B
459 623          126 C=0 XS
460 624          406 A=C X      A.X _ FUNCTION CODE
461 625          1074 RCR 2
462 626          460 LDI      JUMP TABLE START FROM
463 627          100 CON Q100      QUAD 1
464 630          374 RCR 10
465 631          740 GOTOC

```

* REGLFT - PUSHES " REG " INTO LCD FROM RIGHT END & FALLS INTO
 * MEMLFT
 * ASSUMES LCD ENABLED ON ENTRY
 * SEE MEMLFT FOR EXIT CONDITIONS
 * USES ONE ADDITIONAL SUBROUTINE LEVEL AND USES C[6:0] - SEE
 * MEMLFT FOR ADDITIONAL REGISTER USAGE

```

474              ENTRY REGLFT
475 632 REGLFT      1 GOSUB MESSL
475 633              0
476 634          40 CON 32      BLANK
477 635          22 CON Q22      R
478 636          5 CON 5      E
479 637          7 CON 7      G
480 640          1040 CON Q1040      BLANK

```

* MEMLFT - COMPUTE HOW MANY UNUSED REG LEFT IN MEM
 * ASSUME NOTHING.
 * RETURN WITH # OF REG LEFT IN C[2:0] AND CHIP 0 ENABLED.
 * USES A AND C. USES ONE ADDITIONAL SUBROUTINE LEVEL.

```

487 641 MEMLFT      1 GOSUB ENCF00
487 642              0
488 643          1570 C=REGN 13      LOAD CHAIN HEAD
489 644          132 C=0 M
490 645          416 A=C W
491 646          103 GOTO MEMLF2 < 656>
492 647 MEMLF1 1146 C=C-1 X      POINT TO NEXT REG.
493 650          416 A=C W
494 651          1160 DADD=C
495 652          70 C=DATA      LOAD THE REG.
496 653          1356 ? C#0 W      ZERO IN IT ?
497 654          107 GOC MEMLF3 < 664> NO, REACH END OF MEM
498 655          572 A=A+1 M      COUNT 1
499 656 MEMLF2 460 LDI
500 657          300 CON2 12 0
501 660          1546 ? A#C X      REACH REG.(C,0) ?
502 661          33 GONC MEMLF3 < 664> YES
503 662          256 AC EX W
504 663          1643 GOTO MEMLF1 < 647>

```

```

505 664 HENLF3 256 AC EX
506 665          74 RCR      3
507 666          1740 RTN

```

*
 * CHRLCD - OUTPUT A CHAR TO LCD AND CHECK SCROLLING
 * IF S1=1 MEANS DISPLAY ALREADY FULL, THEN AFTER SENDING THE CHAR
 * TO DISPLAY CHECK IF A DELAY IS REQUIRED BY CALLING SCROLL ROUTINE.
 * THE LCD CODE IS EXPECTED IN C[2:0]. ASSUMED LCD ENABLE.
 * USES A.X, C MAY SET S1,S9 MAY RTN VIA SCROL0
 * MAY USE A SUBROUTINE LEVEL
 *

```

516 667 BLANK      460 LDI
517 670          40 CON      @40          OUTPUT A BLANK
518 671 CHRLCD    1750 SLSABC
519 672 CHKFUL    1414 ?S1=1          LCD ALREADY FULL ?
520 673          127 GOC      CKFL10 ( 705) YES, DO DELAY BEFORE RETURN
521 674          460 LDI
522 675          40 CON      @40
523 676          406 A=C      X
524 677          1770 RABCL          READ THE LEFT MOST CHAR
525 700          1650 SRSABC        PUT IT BACK
526 701          126 C=0      XS
527 702          1546 ? A#C      X          IS IT A BLANK ?
528 703          1640 RTN NC      YES, NO NEED FOR SCROLLING YET
529 704          1410 S1=      1          REMEMBER LCD FULL
530 705 CKFL10    1 GOLONC SCROL0
530 706          2

```

*
 * PROMFC - OUTPUT A PROMPT STRING FOR A MICROCODE FUNCTION
 *
 * PROMFC ENTRY: A[1:0]=MAINFRAME FC, LCD NEED NOT BE ENABLED
 * PROMF1 ENTRY: C[1:0]=MAINFRAME FC, LCD MUST BE ENABLED
 * PROMF2 ENTRY: C[6:3]=XADR, LCD ENABLED
 * ALL ENTRY POINTS USE C AND LEAVE S8=0 AND LCD ENABLED
 * PROMFC AND PROMF1 LEAVE PT=2
 * PROMFC USES A SUBROUTINE LEVEL TO CALL ENLCD
 *

```

541 707 PROMFC    1 GOSUB ENLCD
541 710          0
542 711          246 C=A      X          C.X _ FC
542 712          406
543 713 PROMF1    1074 RCR      2
544 714          460 LDI          MAIN FUNCTION TABLE
545 715          24 CON      @24          START FROM @12000
546 716          1174 RCR      9
547 717          1460 CXISA        LOAD XADR
548 720          34 PT=      3
549 721          120 LC      1
550 722          674 RCR      11
551 723 PROMF2    410 S8=      1          INITIALIZE FINAL CHAR FLAG
552 724 PMPT20    1172 C=C-1      M
553 725          1460 CXISA        GET CHARACTER
554 726          126 C=0      XS
555 727          1730 CST EX
556 730          514 ?S6=1          SPECIAL CHARACTER?
557 731          33 GONC      PMPT30 ( 734) NO
558 732          1066 C=C+1      XS          YES. SET BIT FOR DISPLAY CREG
559 733          504 S6=      0
560 734 PMPT30    1214 ?S7=1          FINAL CHARACTER?
561 735          33 GONC      PMPT40 ( 740)

```

```

562 736          404 S8=    0          YES
563 737          1204 S7=   0
564 740 PMPT40  1730 CST EX
565 741          1750 SLSABC          PUT CHAR TO LCD
566 742          414 ?S8=1          MORE CHARS?
567 743          1617 GOC    PMPT20 ( 724 ) YES
568 744          460 LDI
569 745          40 CON    @40
570 746          1750 SLSARC          OUTPUT A BLANK
571 747          1740 RTN

```

```

*
* GENNUM - CONVERT A HEX NUMBER TO DECIMAL & OUTPUT TO LCD
* CALLING SEQUENCE:
*   A,X - HEX NUMBER
*   A,S - # OF OUTPUT DIGITS. IF A,S=0, # OF OUTPUT
*         DIGITS WILL BE EITHER 2,3 OR 4.
*   IF OUTPUT TO LCD IS DESIRED, ENTER WITH LCD CHIP ENABLED.
*   IF LCD IS TO REMAIN UNCHANGED, ENTER WITH A NON-EXISTENT
*         DATA STORAGE CHIP (I.E. CHIP 1) ENABLED.
*   GOSUB GENNUM
* LEAVES # OF DIGITS IN B,S
* LEAVES DIGIT STRING IN A,M LEFT-JUSTIFIED
* RETURNS ACTIVE POINTER=0 FOR HISTORICAL REASONS
* USED A,C,B[13]. DOESN'T CALL ANY SUBROUTINES.
* DOESN'T CHANGE WHICH CHIP (SLEEPER OR LCD) IS ENABLED.
*

```

```

588 750 GENNUM  236 B=A    S
589 751          116 C=0    W
590 752          1240 SETDEC
591 753          246 AC EX  X
592 754          416 A=C    W
593 755          1074 RCR    2          C(0) _ MOST SIGN. DIGIT
594 756          1046 C=C+1  X          CONVERT IF TO DECIMAL
595 757          1146 C=C-1  X
596 760          756 C=C+C    W          MULTIPLY IT BY 16
597 761          756 C=C+C
598 762          756 C=C+C
599 763          756 C=C+C
600 764          1746 A SL    X
601 765          246 AC EX  X
602 766          1074 RCR    2          C(0) _ SECOND DIGIT
603 767          1046 C=C+1  X          CONVERT IT TO DECIMAL
604 770          1146 C=C-1  X
605 771          1006 C=A+C  X
606 772          276 AC EX  S          A,S _ LEAST SIG. DIGIT
607 773          756 C=C+C    W          MULTIPLY 16
608 774          756 C=C+C
609 775          756 C=C+C
610 776          756 C=C+C
611 777          6 A=0      X
612 1000         256 AC EX  W
613 1001         1374 RCR    13          C(0) _ LEAST SIGN. DIGIT
614 1002         1056 C=C+1  X          CONVERT IT TO DECIMAL
615 1003         1156 C=C-1  X
616 1004         1016 C=A+C  W
617 1005         1140 SETHEX
618 1006         1336 ? B#0  S          OUTPUT DIGITS = 0 ?
619 1007         167 GOC    GENN20 (1025) NO
620 1010         1334 PT=    13
621 1011         420 LC      4

```

622	1012		436 A=C	S	DETERMINE MIN. OUTPUT DIGITS
623	1013		174 RCR	4	
624	1014		1376 ? C#0	S	NEEDS 4 DIGITS ?
625	1015		177 GOC	GENN40 (1034)	YES
626	1016		1374 RCR	13	
627	1017		676 A=A-1	S	
628	1020		1376 ? C#0	S	NEEDS THREE DIGITS ?
629	1021		137 GOC	GENN40 (1034)	YES
630	1022		1374 RCR	13	
631	1023		676 A=A-1	S	TWO DIGITS
632			LEGAL		
633	1024		103 GOTO	GENN40 (1034)	
634	1025	GENN20	176 A=B	S	
634	1026		236		
635	1027	GENN25	676 A=A-1	S	
636	1030		37 GOC	GENN30 (1033)	
637	1031		1474 RCR	1	
638	1032		1753 GOTO	GENN25 (1027)	
639	1033	GENN30	176 AB EX	S	
640	1034	GENN40	1474 RCR	1	COPY DIGIT STRING TO A.M
641	1035		432 A=C	M	
642	1036		236 B=A	S	COPY # OF DIGITS TO B.S
643	1037		1374 RCR	13	LEFT-JUSTIFY DIGIT STRING IN C
644	1040		1634 PT=	0	AS ADVERTISED FOR EXIT
645	1041	GENN55	676 A=A-1	S	
646	1042		1540 RTN C		
647	1043		460 LDI		
648	1044		3 CON	3	
649	1045		1374 RCR	13	
650	1046		1750 SLSABC		
651	1047		1723 GOTO	GENN55 (1041)	
652			EJECT		

*
 * AFORMT - FORMAT A NUMBER AND PUT IT TO ALPHA STRING
 * CALLED BY ARCL, THE NUMBER IS EXPECTED IN B
 * ASSUMED NOTHING. USED A,B,C. RETURN WITH CHIP 0 ENABLE.
 * CALLED APPEND TO PUT THE CHAR IN ALPHA REG.
 * CALLED FORMAT. 2 SUB LEVELS.
 *

```

660 1050 AFORMT 106 C=0 X
661 1051 1160 DADD=C ENABLE CHIP 0
662 1052 316 C=B W LOAD THE NUMBER
663 1053 1 GOSUB FORMAT
663 1054 0
664 1055 404 S8= 0 ASSUME FIX MODE
665 1056 24 ? PT= 3 FIX MODE ?
666 1057 23 GONC **2 (1061) YES
667 1060 410 S8= 1 SCI OR ENG MODE
668 1061 256 AC EX LOAD DISPLAY REG.A
669 1062 530 M=C SAVE IN M
670 1063 1376 ? C#0 S MANTISSA NEGATIVE ?
671 1064 1 GSUBC APND- YES
671 1065 1
672 1066 340 SEL Q
673 1067 1534 PT= 12 Q=12
674 1070 AFMT10 340 SEL Q
675 1071 AFMT11 1324 ? PT= 13 JUST DONE WITH EXP ?
676 1072 1540 RTN C YES, WE ARE ALL DONE
677 1073 AFMT12 320 LC 3
678 1074 1734 INC PT
679 1075 142 AB EX PT
680 1076 1402 ? A<C PT ENCOUNTER A BLANK ?
681 1077 327 GOC AFMT30 (1131) YES, END OF MANTISSA
682 1100 1542 ? A#C PT IS IT A DIGIT ONLY ?
683 1101 223 GONC AFMT20 (1123) YES
684 1102 142 AB EX PT
685 1103 1 GOSUB APNDG OUTPUT THR DIGIT FIRST
685 1104 0
686 1105 340 SEL Q
687 1106 302 C=B PT
688 1107 742 C=C+C PT IS A COMMA ?
689 1110 43 GONC **4 (1114) NO, IS A D.P.
690 1111 460 LDI
691 1112 54 CON 054 COMMA
692 1113 33 GOTO **3 (1116)
693 1114 460 LDI
694 1115 56 CON 056 D.P.
695 1116 1724 DEC PT
696 1117 240 SEL P
697 1120 1 GOSUB APND10
697 1121 0
698 1122 1463 GOTO AFMT10 (1070)
699 1123 AFMT20 1 GOSUB APNDG OUTPUT THE DIGIT
699 1124 0
700 1125 340 SEL Q
701 1126 1724 DEC PT
702 1127 1024 ? PT= 2 END OF MANTISSA ?
703 1130 1403 GONC AFMT10 (1070) NOT YET
704 1131 AFMT30 414 ?S8=1 FIX MODE ?
705 1132 1640 RTN NC YES, ALL DONE
706 1133 1434 PT= 1

```

```

707 1134      630 C=M
708 1135      1342 ? C#0 PT
709 1136      27 GOC      **2      (1140)
710 1137      1724 DEC PT
711 1140      240 SEL P
712 1141      460 LDI
713 1142      105 CON      @105      E
714 1143      1 GOSUB      APND10
714 1144      0
715 1145      630 C=M
716 1146      1366 ? C#0 XS      EXP NEGATIVE ?
717 1147      1 GSUBC      APND-      YES
717 1150      1
718 1151      1173 GOTO      AFMT10 (1070)

```

* APNDDG (INCLUDING APND-, APND10, APND20) HAS BEEN MOVED TO
 * QUAD 0 TO FILL UP A HOLE THERE

* ROMHED - LOCATE ROM HEAD ADDRESS
 *- RETURNS THE ADDRESS OF THE BEGIN STATEMENT AT
 *- THE START OF A PROGRAM IN ROM
 *- IN: CHIP 0 SELECTED
 *- OUT: A[3:0] = ROM HEAD ADDRESS
 *- USES: C[13:0], STATUS BIT 12, & A[3:0]
 *-
 * ENTRY POINT- ROMH05
 *- IN: PT= 3
 *- C[6:3] = ROM ADDRESS

```

733
734
735
736
737
738
739
740 1152 ROMHED  34 PT= 3      -
741 1153      1470 C=REGN 12      GET PGMCTR
742 1154 ROMH05  674 RCR  11      -
743 1155      1504 S12=  0      -
744 1156      23 GOTO  **2      (1160) -
745 1157 ROMH06 1172 C=C-1 M      FIND BEGIN STMT
746 1160      1460 CXISA      -
747 1161      1166 C=C-1 XS      -
748 1162      1757 GOC      ROMH06 (1157) -
749 1163      1166 C=C-1 XS      -
750 1164      1737 GOC      ROMH06 (1157) -
751 1165      1166 C=C-1 XS      SET PRIVACY BIT
752 1166      27 GOC      **2      (1170) -
753 1167      1510 S12=  1      -
754 1170 ROMH35  74 RCR  3      A[3:0]_HEAD ADDR
755 1171      412 A=C      WPT      -
756 1172      1740 RTN      -
757
758
759
760

```

* CPGMHD - CURRENT PROGRAM HEAD
 *- RETURNS THE ADDRESS OF THE BEGIN STATEMENT IN
 *- ROM AND THE ADDRESS OF THE FIRST STEP OF A
 *- PROGRAM IN RAM

NOMAS
 Max Manufacturer Supported
 recipient agrees NOT to contact manufacturer

```

*- IN:  A[3:0]= PROGRAM COUNTER (MUST BE THE ADDRESS OF A LINK,
*       I.E., THE FIRST BYTE OF A GLOBAL LBL OR END)
*       NO PERIPHERAL ENABLED.
*-     PT= 3
*- OUT: A[3:0]= CURRENT PROGRAM HEAD ADDRESS
*-     PT= 3
*- USES: A[3:0], C[13:0], B[4:0]
*- USES: 1 SUBROUTINE LEVEL

```

```

773
774
775
776 1173 CPGMHD 314 ?S10=1 ROMFLAG?
777 1174      33 GONC CPGM10 (1177) NOPE
778 1175      252 AC EX WPT -
779 1176      1563 GOTO ROMH05 (1154) -
780      ENTRY CPGM10 FOR CARD READER & PRINTER
781 1177 CPGM10 1 GSBLNG:GTLINK GET LINK
781 1200      0
782 1201 CPGM15 1346 ? C#0 X CHAIN END?
783 1202      1 GOLNC FSTIN YES
783 1203      2
784 1204      1 GSBLNG:UPLINK NO, TRAVERSE CHAIN
784 1205      0
785 1206      1076 C=C+1 S ALPHA LBL?
786 1207      1727 GOC CPGM15 (1201) YES
787 1210      1 GOLONG INCAD2 A[3:0]_HEAD ADDRESS
787 1211      2
788
789
790
791
792
793
794      ENTRY ALCE00
795      ENTRY KEYOP
796      ENTRY RAK60

```

```

*
* KEYOP - KEYCODE OPERAND - PARSE LOGIC FOR ASSIGN FCN
*

```

```

800      KEYOP ON ENTRY, CHIP 0 IS ON,
801      PTR=1, C.X=H01F,
802      A[1:0]=H1F
803 1212      1 GOSUB OFSHFT
803 1213      0
804 1214      1 GOSUB ENLCD
804 1215      0
805 1216      460 LDI
806 1217      40 CON 32
807 1220      1750 SLSABC INSERT BLANK
808 1221 KYOP10 1 GOSUB NEXT1
808 1222      0
809
810      ON RETURN FROM NEXT, PT=1,
811      LCD CHIP ON, SS PTEMP1 UP
812      & B.X=" "
813      & H[2:1]=LOGICAL KC(0-79)
813      ENTRY KYOPCK FOR WAND 11/26/79
814      KYOPCK
815 1223      0 NOP BKARROW IS A LEGAL OPERAND
816 1224      514 ?S6=1 SHIFT KEY?
817 1225      163 GONC KYOP40 (1243) NO

```



```

818 1226          260 C=N          RETRIEVE LOGICAL KC TO C[2:1]
819 1227          742 C=C+C PT     SHIFT ALREADY ON?
820 1230          57 GOC          KYOP11 (1235) YES
821 1231          460 LDI          NO.
822 1232          55 CON          45    "-"
823 1233          1750 SLSABC      PUT HYPHEN TO LCD
824 1234          23 GOTO         KY11A (1236)
825 1235 KYOP11 1670 RABCR        SHIFT OFF HYPHEN
826             KY11A
827 1236          1 GOSUB         TOGSHF    TOGGLE THE SHIFT FLAG
827 1237          0
828 1240          1 GOSUB         ENLCD
828 1241          0
829 1242          1573 GOTO       KYOP10 (1221)

*
831             KYOP40
832 1243          460 LDI
833 1244          303 CON2        12      3      HEX C3
834 1245          416 A=C
835 1246          1040 C=KEYS
836 1247          74 RCR          3          KC TO C[1:0]
837 1250          1412 ? A<C      WPT       KC>C3?
838 1251          43 GONC         KYOP50 (1255) NO. LEGITIMATE KEY
839 1252          1 GOSUB         BLINK     YES. USER, PRGM, OR ALPHA KEY
839 1253          0
840 1254          1453 GOTO       KYOP10 (1221)

*
842             KYOP50
843 1255          116 C=0
844 1256          1040 C=KEYS
845 1257          74 RCR          3
846 1260          320 LC          3
847 1261          1046 C=C+1      X
848 1262          1750 SLSABC      SEND ROW TO LCD
849 1263          406 A=C          X      COPY ASCII ROW TO A.X
850 1264          460 LDI
851 1265          64 CON          004      "4"
852 1266          404 SB=         0      ASSUME ROW#4
853 1267          1546 ? A#C      X      ROW#4?
854 1270          27 GOC          KYOP60 (1272) YES, NOT "ENTER" ROW
855 1271          410 SB=         1      "ENTER" ROW
856 1272 KYOP60 260 C=N          RECOVER LOG KC TO C[2:1]
857 1273          1706 C SR        X      LOG KC TO C[1:0], 0 TO C[2]
858 1274          406 A=C          X      SAVE LOG KC IN A.X
859 1275          1706 C SR        X      LOG COL TO C[0]
860 1276          1434 PT=         1
861 1277          320 LC          3
862 1300          414 ?SB=1        "ENTER" ROW
863 1301          33 GONC         KYOP70 (1304) NO
864 1302          1342 ? C#0      PT      A KEY TO THE RIGHT OF "ENTER"?
865 1303          27 GOC          KYOP80 (1305) YES - DON'T INC COLUMN #
866 1304 KYOP70 1042 C=C+1      PT      INCREMENT COLUMN #
867 1305 KYOP80 1750 SLSABC      SEND COL TO LCD
868 1306          546 A=A+1      X      KC INTERNAL FORM (1-80)
869             LEGAL
870 1307          1 GOLONG -MULT#3
870 1310          2

```

*
 * ALCL00 - LOGIC TO MAP LOCAL ALPHA OPERANDS ONTO NUMERIC OPERANDS
 *

```

874      ALCL00
875 1311      416 A=C      CHARACTER TO A.[1:0]
876      REMAINDER OF A IS ZERO
877 1312      460 LDI
878 1313      101 CON      @101      "A"
879 1314      1406 ? A<C  X      CHAR<"A"?
880 1315      1540 RTN C      NOT LOCAL
881 1316      460 LDI
882 1317      113 CON      @113      "K"
883 1320      1406 ? A<C  X      CHAR<"K"?
884 1321      363 GONC      ALCL50 (1357) NO. TEST FOR LOWER CASE
885 1322      460 LDI
886 1323      45 CON      37      MAP 65 ONTO 102
887 1324 ALCL10 1006 C=A+C  X
888 1325      1374 RCR      13
889 1326      346 BC EX  X      ARGUMENT TO B[2:1],
890      FC TO C[1:0]
891 1327      1434 PT=      1      FC TO A[1:0]
892 1330      412 A=C      WPT
893 1331      460 LDI
894 1332      36 CON2      1      14      FC FOR AXEQ
895 1333      1552 ? A#C  WPT      FC#AXEQ?
896 1334      43 GONC      ALCL20 (1340) THIS IS AXEQ
897 1335      1146 C=C-1  X      CONVERT AXEQ TO AGTO
898 1336      1552 ? A#C  WPT      FC#AGTO?
899 1337      67 GOC      ALCL30 (1345) NOT AGTO
900 1340 ALCL20 374 RCR      10      NEW FC TO C[4:3]
901 1341      416 A=C      NEW FC TO A[4:3]
902 1342      146 AB EX  X      MERGE ARGUMENT WITH
903 1343      1 GOLONG:NLT020      IN A[4:1]
903 1344      2
*
905 1345 ALCL30 460 LDI
906 1346      315 CON2      12      13      FC FOR ALBL
907 1347      1552 ? A#C  WPT      FC#ALBL?
908 1350      57 GOC      ALCL40 (1355) NOT ALBL.
909 1351      1474 RCR      1
910 1352      1076 C=C+1  S
911 1353      1076 C=C+1  S      LBL NN FC (CF) TO C[0:13]
912      LEGAL
913 1354      1643 GOTO      ALCL20 (1340)
*
915 1355 ALCL40 146 AB EX  X      FC BACK TO B.X
916 1356      1740 RTN
*
918 1357 ALCL50 460 LDI      TEST FOR LOWER CASE A...E
919 1360      141 CON      @141      SMALL A
920 1361      1406 ? A<C  X      CHAR<SMALL A?
921 1362      1540 RTN C      YES, NOT LOCAL
922 1363      460 LDI
923 1364      146 CON      @146      SMALL F
924 1365      1406 ? A<C  X      CHAR<SMALL F?
925 1366      1640 RTN NC      NO, NOT LOCAL
926 1367      460 LDI
927 1370      32 CON      26      MAP 97 TO 123
928 1371      1333 GOTO      ALCL10 (1324)
*
* ENTER HERE FROM PARSE NEWFCN LOGIC IN USER MODE WHEN THE
* BIT IN THE BIT MAP IS SET.      BIT SET IN BIT MAP
932      RAK60

```

```

933 1372      260 C=N      RECOVER KC FROM N
934 1373      1474 RCR      1      SETUP
935 1374      416 A=C      FOR
936 1375      546 A=A+1    X      -
937 1376      1404 S1=      0      GPCKC
938 1377      1 GOSUB      GCPKC      FIND REASSIGNED FCN
938 1400      0
939 1401      14 ?S3=1      RAM?
940 1402      417 GOC      RAK100 (1443) YES
941 1403      34 PT=      3
942 1404      1342 ? C#0    PT      XROM FC?
943 1405      57 GOC      RAK70 (1412) YES
944 1406      416 A=C      -
945 1407      26 A=0      XS      MUST BE MAINFRAME
946 1410      1 GOLONG NAME4A
946 1411      2

*
  948      ENTRY RAK70      FOR WAND XROM
* ENTRY POINT ADD FOR WAND ON 3-13-79
*
  951      RAK70
  952 1412      160 N=C      XROM
  953 1413      1 GOSUB      GTRNAD      SAVE FC IN N
  953 1414      0
* GTRNAD RETURNS XADR IN A[3:0]
  955 1415      123 GOTO      RAK90 (1427) MISSING ROM
  956 1416      256 AC EX
  957 1417      730 MC EX      PUT XADR TO N
  958 1420      14 ?S3=1      USER LANGUAGE?
  959 1421      47 GOC      RAK80 (1425) YES
  960 1422      260 C=N      RETRIEVE FC FROM N
  961 1423      1 GOLONG NAME4D
  961 1424      2

*
  963      RAK80
  964      ROM USER LANGUAGE
  965      XROM FC IN M[3:0]
  966 1425      1010 S2=      1      XADR IN M[3:0]
  967 1426      613 GOTO      NM440X (1507) STRING IN ROM

*
  969 1427 RAK90      1 GOSUB      CLLCDE      MISSING ROM
  969 1430      0
  970 1431      1 GOSUB      XROMNF
  970 1432      0
  971 1433      1 GOSUB      ENCF00
  971 1434      0
  972 1435      1104 S9=      0      SAY ADDR UNKNOWN
  973 1436      260 C=N      XROM TO C
  974 1437      1 GOSUB      STORFC
  974 1440      0
  975 1441      1 GOLONG NM4405
  975 1442      2

*
  977      RAK100
  978 1443      530 M=C      RAM
  979 1444      1635 CON      @1635      SAVE ADDR IN M
  980 1445      674 CON      @674      GOSUB PRT4
  981 1446      630 C=N      PRINT DATAENTRY, IF ANY
  982 1447      34 PT=      3      RECOVER ADDRESS
  983 1450      1 GOSUB      GTLNKA      GET # OF CHARS

```

```

983 1451      0
984 1452      1374 RCR      13      # OF CHARS TO
985 1453      436 A=C      S      A.S
986 1454      676 A=A-1    S      SKIP OVER KEYCODE
987 1455      676 A=A-1    S
988           LEGAL
989 1456      1 GOSUB      NXBYT3    MOVE PTR TO KEYCODE
989 1457      0
990 1460      116 C=0
991 1461 RAK110 160 N=C      INITIALIZE STRING
992 1462      1 GOSUB      NXBYTA    SAVE STRING IN N
992 1463      0
993 1464      1730 CST EX
994 1465      260 C=N
995 1466      1730 CST EX
996 1467      1074 RCR      2
997 1470      676 A=A-1    S
998 1471      1703 GONC    RAK110 (1461)
999 1472      1 GOSUB      RTJLBL
999 1473      0
1000 1474      160 N=C
1001 1475      106 C=0      X
1002 1476      1160 DADD=C
1003 1477      260 C=N
1004 1500      1150 REGN=C 9
1005 1501      460 LDI
1006 1502      36 CON2     1      14      FC FOR AXEQ
1007 1503      1574 RCR      12
1008 1504      1 GOSUB      STORFC
1008 1505      0
1009 1506      1004 S2=     0      RAM
1010 1507 NM440X 1 GO LONG :NM440
1010 1510      2
1011
1012
* OFSFT - TURN OFF SHIFTSET AND SHIFT ANNUNCIATOR
* REQUIRES CHIP 0 ENABLED ON INPUT
* DESTROYS C
* USES ONE SUBROUTINE LEVEL
* RETURNS VIA ENCP00
*
1019 1511 OFSFT 1670 C=REGN 14
1020 1512      1474 RCR      1
1021 1513      1730 CST EX
1022 1514      104 S4=     0      GET STATUS SET 1/2
1023 1515      1730 CST EX      CLEAR SHIFTSET
1024 1516      1374 RCR      13
1025 1517      1650 REGN=C 14
1026 1520      1 GOSUB      ENLCD
1026 1521      0
1027 1522      570 READEN
1028 1523      1730 CST EX
1029 1524      1204 S7=     0      RESET BIT FOR SHIFT ANNUNCIATOR
1030 1525      1730 CST EX
1031 1526      1360 WRTEH
1032 1527      1 GO LONG :ENCP00
1032 1530      2

```

*

 *THIS ROUTINE SETS ALL ANNUNCIATORS.

* ON ENTRY, ANY DATA STORAGE OR PERIPHERAL CHIP MAY BE ENABLED.
 * ON EXIT, CHIP 0 IS ENABLED, REG 14 IS IN C, AND
 * SS 0 IS IN ST
 * USES 0 SUBROUTINE LEVELS. USES A.X

```

1041 1531 RSTANN      1 GOSUB  RSTMS1
1041 1532              0
1042 1533 ANN+14 1650 REGN=C 14          SAVE REG 14
1043 1534 ANNHOUT 460 LDI              LOAD LOW BAT CONST
1044 1535          200 CON      0200
1045 1536          246 AC EX  X
1046 1537          1746 A SL  X
1047 1540          116 C=0
1048 1541          1760 PFAD=C
1049 1542          1160 DADD=C
1050 1543          1670 C=REGN 14
1051 1544          1530 ST=C          BRING UP SS0
1052 1545          1314 ?S13=1      RUNNING?
1053 1546          37 GOC      SETPGM (1551) YES. TURN ON PRGM ANHUN
1054 1547          14 ?S3=1      PGM MODE
1055 1550          33 GONC      LOWBAT (1553) NO
1056 1551 SETPGM  546 A=A+1  X
1057 1552          546 A=A+1  X
1058 1553 LOWBAT  514 ?S6=1      LOW BATTERY?
1059 1554          27 GOC      ALPHA (1556) YES
1060 1555          26 A=0      XS      CLEAR LOW BAT
1061 1556 ALPHA  1214 ?S7=1
1062 1557          23 GONC      SHIFT (1561)
1063 1560          556 A=A+1
1064 1561 SHIFT  1074 RCR      2
1065 1562          1530 ST=C
1066 1563          1614 ?S0=1
1067 1564          43 GONC      RAD (1570)
1068 1565          460 LDI
1069 1566          200 CON      0200
1070 1567          506 A=A+C  X
1071 1570 RAD      114 ?S4=1
1072 1571          57 GOC      RADSET (1576)
1073 1572 GRAD     214 ?S5=1
1074 1573          43 GONC      USER (1577)
1075 1574          566 A=A+1  XS
1076 1575          566 A=A+1  XS
1077 1576 RADSET  566 A=A+1  XS
1078 1577 USER     274 RCR      5
1079 1600          1530 ST=C
1080 1601          1614 ?S0=1
1081 1602          53 GONC      FLAGS (1607)
1082 1603          566 A=A+1  XS
1083 1604          566 A=A+1  XS
1084 1605          566 A=A+1  XS
1085 1606          566 A=A+1  XS
1086 1607 FLAGS    274 RCR      5
1087 1610          126 C=0      XS
1088 1611          756 C=C+C
1089 1612          1474 RCR      1
1090 1613          126 C=0      XS
1091 1614          756 C=C+C
1092 1615          756 C=C+C
1093 1616          516 A=A+C
1094 1617          460 LDI

```

```

1095 1620      20 CON      020
1096 1621      1160 DADD=C
1097 1622      460 LDI
1098 1623      375 CON      0375
1099 1624      1760 PFAD=C
1100 1625      256 AC EX
1101 1626      1360 WRTEN
1102 1627 LDSST0 106 C=0    X
1103 1630      1760 PFAD=C
1104 1631      1160 DADD=C
1105 1632      1670 C=REGN 14
1106 1633      1530 ST=C
1107 1634      1740 RTN

```

*

*

```

1110      ENTRY  XR/S
1111      XR/S      EXECUTE R/S
1112 1635      574 RCR      6
1113 1636      1730 CST EX
1114 1637      1414 ?S1=1
1115 1640      1 GOLC      R/SCAT
1116 1641      3
1117 1642      1530 ST=C
1118 1643      14 ?S3=1
1119 1644      53 GONC      XRS20 (1651) NO
1120 1645      460 LDI
1121 1646      204 CON2      8      4      FC FOR STOP
1122 1647      1 GOLONG PAR556
1123 1650      2
1124 1651 XRS20 1414 ?S1=1
1125 1652      1737 GOC      XRS10 (1645) YES
1126 1653      1 GOSUB      PACH4
1127 1654      0

```

*

* IF KEY IS LET UP, GO RUN. OTHERWISE PUT UP DESCRIPTION OF STEP.

* THIS IS A PATCH TO SPEED UP EXECUTION OF R/S.

*

```

1131 1655      1514 ?S12=1
1132 1656      63 GONC      XRS25 (1664) NO
1133 1657      404 S8=      0      YES
1134 1660      1 GOSUB      MSGA
1135 1661      0
1136 1662      0 XDEF      MSGPR
1137 1663      113 GOTO      XRS40 (1674)

```

*

```

1138 1664 XRS25 1770 C=REGN 15
1139 1665      1346 ? C#0 X
1140 1666      37 GOC      XRS30 (1671) YES, NON-ZERO
1141 1667      1046 C=C+1 X
1142 1670      1750 REGN=C 15
1143 1671 XRS30 1 GOSUB      DFKECK
1144 1672      0
1145 1673      1114 ?S9=1
1146 1674 XRS40 1 GSUBNC NULTST
1147 1675      0
1148 1676 XRS45 1575 CON      01575
1149 1677      674 CON      0674

```

```

1149 1700          1 GOSUB RSTANN
1149 1701          0

*
1151              ENTRY RUN
1152 RUN
1153 1702          1310 S13= 1          GET A USER PROGRAM RUNNING
1154 1703          1770 C=REGN 15      SET RUNNING FLAG
1155 1704          106 C=0 X          SET LINE # TO FFF
1156 1705          1146 C=C-1 X
1157 1706          1750 REGN=C 15
1158 1707          1670 C=REGN 14
1159 1710          1530 ST=C          PUT UP SS0
1160 1711          766 C=C+C XS
1161 1712          766 C=C+C XS      DATA ENTRY FLAG?
1162 1713          53 GONC RUN11 (1720) NO
1163                                     YES, MUST BE PAUSE
1164                                     TERMINATION
1165 1714          1635 CON 01635      GOSUB PRT4
1166 1715          674 CON 0674      PRINT DATAENTRY STRING
1167 1716          1 GOSUB RSTMS1     CLEAR MSGFLG, DATAENTRY FLAG, ETC.
1167 1717          0
1168 1720 RUN11 1404 S1= 0          CLEAR PAUSING
1169 1721          1670 C=REGN 14      PUT
1170 1722          1630 C=ST          SS0
1171 1723          1650 REGN=C 14      BACK

*
* LOGIC FOR CLD ENTERS AT NWGOOS.
* ENTRY CONDITION: SS0 UP
*
1176              ENTRY NWGOOS
1177 1724 NWGOOS 1 GOSUB PGNAON      TURN ON PRGM ANNUNCIATOR
1177 1725          0
1178                                     & ENABLE LCD
1179 1726          214 ?S5=1          MSGFLAG?
1180                                     MSGFLG CAN ONLY BE SET
1181                                     HERE ON PAUSE TERMINATION
1182 1727          107 GOC RUN20 (1737) YES
1183 1730          1 GOSUB CLLCDE
1183 1731          0
1184 1732          460 LDI
1185 1733          56 CON2 2 14      EAST GOOSE
1186 1734          1650 SRSABC
1187 1735          1 GOSUB ENCF00
1187 1736          0
1188 1737 RUN20 1 GOLONG NFRPU      CAN'T BE A RTN BECAUSE
1188 1740          2

* XR/S IS XKD AND DOESN'T HAVE NFRPU ON THE STACK
*
* INTARG - ZERO THE ALPHA REG AND STORE THE CHAR IN G AS THE
* FIRST CHAR.
*
1194              ENTRY INTARG
1195 1741 INTARG 116 C=0 W
1196 1742          1050 REGN=C 8
1197 1743          750 REGN=C 7
1198 1744          650 REGN=C 6
1199 1745          230 C=G
1200 1746          550 REGN=C 5
1201 1747          1740 RTN
1202              ENTRY STOREC

```

*
 * STORFC - STORE FUNCTION CODE TO REG 10
 * ON ENTRY, DESIRED FC IS IN C[3:0], LEFT JUSTIFIED IN THAT FIELD.
 * ON EXIT, FC (4 DIGITS) STORED TO REG 10[4:1]
 * REG 10[0] IS SCRATCH, A[4:0] AND C ARE USED, AND PT=4.
 *

```

1209 1750 STORFC 1374 RCR      13
1210 1751          134 PT=      4
1211 1752          412 A=C      WPT
1212 1753          1270 C=REGN 10
1213 1754          252 AC EX    WPT
1214 1755          1250 REGN=C 10
1215 1756          1740 RTN

```

*
 * MESSL - LEFT SHIFT INTO LCD FROM RIGHT END
 * CALLING SEQUENCE: GOSUB MESSL
 * CON 1ST CHAR, LCD FORM
 * CON 2ND CHAR, ...
 * ...
 * CON FINAL CHAR + @1000
 * SPECIAL CHARACTERS (THOSE HAVING LCD CREG=1) CAN ONLY BE USED
 * AS THE FINAL CHARACTER OF THE MESSAGE.
 * ASSUMES LCD ENABLED ON ENTRY.
 * USES C[6:0] AND LEAVES LCD ENABLED ON EXIT
 *

```

1228          ENTRY MESSL
1229 1757 MESSL  660 C=STK
1230 1760 MESS10 1460 CXISA
1231 1761          1750 SLSABC
1232 1762          1072 C=C+1  M
1233 1763          1366 ? C#0  XS
1234 1764          1743 GONC   MESS10 (1760)
1235 1765          740 GOTOC

```

*
 * ENLCD - ENABLE LCD DRIVER CHIP
 * USES C.X ONLY
 *

```

1240          ENTRY ENLCD
1241 1766 ENLCD  460 LDI
1242 1767          20 CON2    1      0      DISABLE SLEEPER CHIPS
1243 1770          1160 DADD=C
1244 1771          460 LDI
1245 1772          375 CON2    15     13      TURN ON LCD DRIVER CHIP
1246 1773          1760 PFAD=C
1247 1774          1740 RTN
1248          UNLIST
1251          END

```

ERRORS : 0

SYMBOL TABLE

AFMT10	1070	-	1151	1130	1122
AFMT11	1071	-			
AFMT12	1073	-			
AFMT20	1123	-	1101		
AFMT30	1131	-	1077		
AFORMT	1050	-			
ALCL00	1311	-			
ALCL10	1324	-	1371		
ALCL20	1340	-	1354	1334	
ALCL30	1345	-	1337		
ALCL40	1355	-	1350		
ALCL50	1357	-	1321		
ALPHA	1556	-	1554		
ANN+14	1533	-			
ANNOUT	1534	-			
BLANK	667	-			
CAPABC	242	-	212		
CHKFUL	672	-			
CHRLCD	671	-			
CKFL10	705	-	673		
CPGM10	1177	-	1174		
CPGM15	1201	-	1207		
CPGMHD	1173	-			
DEROW	255	-			
DERW00	262	-	340		
DERW05	300	-	276		
DERW10	303	-	267		
DERW20	313	-	305		
DERW50	316	-	265		
DERW55	321	-	315	312	
DERW60	323	-	302		
DERW70	330	-	261		
DF010	534	-	540		
DF030	561	-	554		
DF040	602	-	571	560	530
DF050	604	-	567		
DF060	607	-			
DF100	615	-	621		
DF120	40	-	31	24	
DF150	202	-	241		
DF160	205	-			
DF190	341	-	404	336	
DF200	347	-	342		
DFILLF	543	-			
DFKBCK	531	-			
DFRST8	542	-			
DFRST9	541	-			
ENLCD	1766	-			
FLAGS	1607	-	1602		
GENN20	1025	-	1007		
GENN25	1027	-	1032		
GENN30	1033	-	1030		
GENN40	1034	-	1024	1021	1015
GENN55	1041	-	1047		
GENNUM	750	-			
GRAD	1572	-			

NOMAS
 NOT Manufacturer Supported
 Recipient agrees NOT to contact manufacturer

INTARG	1741	-							
KEYOP	1212	-							
KY11A	1236	-	1234						
KYOP10	1221	-	1254	1242					
KYOP11	1235	-	1230						
KYOP40	1243	-	1225						
KYOP50	1255	-	1251						
KYOP60	1272	-	1270						
KYOP70	1304	-	1301						
KYOP80	1305	-	1303						
KYOPCK	1223	-							
LDSST0	1627	-							
LOWBAT	1553	-	1550						
MEMLF1	647	-	663						
MEMLF2	656	-	646						
MEMLF3	664	-	661	654					
MEMLFT	641	-							
MESS10	1760	-	1764						
MESSL	1757	-							
NM440X	1507	-	1426						
NWGOOS	1724	-							
OFSHFT	1511	-							
PMPT20	724	-	743						
PMPT30	734	-	731						
PMPT40	740	-	735						
PROMF1	713	-							
PROMF2	723	-							
PROMFC	707	-							
RAD	1570	-	1564						
RADSET	1576	-	1571						
RAK100	1443	-	1402						
RAK110	1461	-	1471						
RAK60	1372	-							
RAK70	1412	-	1405						
RAK80	1425	-	1421						
RAK90	1427	-	1415						
REGLFT	632	-							
RL	252	-	220						
RO1314	62	-	16	15					
ROMH05	1154	-	1176						
ROMH06	1157	-	1164	1162					
ROMH35	1170	-							
ROMHED	1152	-							
ROW0	21	-	0						
ROW010	23	-	53						
ROW09	50	-	11						
ROW1	25	-	1						
ROW10	76	-	12						
ROW11	51	-	13						
ROW12	54	-	14						
ROW120	431	-							
ROW122	462	-	444						
ROW125	527	-	474	472					
ROW2	27	-	2						
ROW3	36	-	3						
ROW4-8	32	-	10	7	6	5	4		
ROW9	126	-	50						
ROW910	133	-	106	57					
ROW930	140	-	75						
ROW931	143	-	47						

ROW933	147	-	125	
ROW935	164	-	146	
ROW936	166	-	163	
ROW940	207	-	172	
ROW945	231	-	234	
ROW950	237	-	254	244
ROW960	240	-	251	232
RSTANN	1531	-		
RT	235	-	214	
RUN	1702	-		
RUN11	1720	-	1713	
RUN20	1737	-	1727	
RW0110	351	-	260	
RW0140	360	-		
RW0141	361	-		
SETPGM	1551	-	1546	
SHIFT	1561	-	1557	
SMLABC	245	-	222	
STORFC	1750	-		
TXRW10	366	-		
TXRW30	403	-	430	
TXTR0M	365	-		
TXTR0W	362	-		
TXTSTR	366	-	364	
USER	1577	-	1573	
XMSGPR	555	-		
XR/S	1635	-		
XRS10	1645	-	1652	
XRS20	1651	-	1644	
XRS25	1664	-	1656	
XRS30	1671	-	1666	
XRS40	1674	-	1663	
XRS45	1676	-		

ENTRY TABLE

AFORMT	1050	-
ALCL00	1311	-
ANN+14	1533	-
ANNOUT	1534	-
BLANK	667	-
CHKFUL	672	-
CHRLCD	671	-
CPGM10	1177	-
CPGMHD	1173	-
DEROW	255	-
DERW00	262	-
DF060	607	-
DF150	202	-
DF160	205	-
DF200	347	-
DFILLF	543	-
DFKBCK	531	-
DFRST8	542	-
DFRST9	541	-
ENLCD	1766	-
GENNUM	750	-
INTARG	1741	-
KEYOP	1212	-
KYOPCK	1223	-
LDSST0	1627	-
MEMLFT	641	-
MESSL	1757	-
NWG009	1724	-
OFSHFT	1511	-
PROMF1	713	-
PROMF2	723	-
PROMFC	707	-
RAK60	1372	-
RAK70	1412	-
REGLFT	632	-
ROMH05	1154	-
ROMH35	1170	-
ROMHED	1152	-
ROW120	431	-
ROW933	147	-
ROW940	207	-
RSTANN	1531	-
RUN	1702	-
RW0110	351	-
RW0141	361	-
STORFC	1750	-
TXRW10	366	-
TXTR0M	365	-
TXTR0W	362	-
TXSTR	366	-
XMSGPR	555	-
XR/S	1635	-
XRS45	1676	-

EXTERNAL REFERENCES

[illegible]

MSG	555				
MSG	556				
MSGA	1660				
MSGA	1661				
MSGPR	557	1662			
NAM44Q	1507				
NAM44Q	1510				
NAME4A	1410				
NAME4A	1411				
NAME4D	1423				
NAME4D	1424				
NBYTA0	66	136	323	355	
NBYTA0	67	137	324	356	
NBYTAB	107				
NBYTAB	110				
NEXT1	1221				
NEXT1	1222				
NFRPU	1737				
NFRPU	1740				
NLT020	1343				
NLT020	1344				
NM44Q5	1441				
NM44Q5	1442				
NULT#3	1307				
NULT#3	1310				
NULTST	1674				
NULTST	1675				
NXBYT3	1456				
NXBYT3	1457				
NXBYTA	415	1462			
NXBYTA	416	1463			
NXBYTO	412				
NXBYTO	413				
NXTBYT	70	434	615		
NXTBYT	71	435	616		
OFSHFT	1212				
OFSHFT	1213				
PACH4	1653				
PACH4	1654				
PARS56	1647				
PARS56	1650				
PGMAON	1724				
PGMAON	1725				
PROMF1	123	447	467		
PROMF1	124	450	470		
PROMFC	32	43	64	134	353
PROMFC	33	44	65	135	354
R/SCAT	1640				
R/SCAT	1641				
REGLFT	503	572			
REGLFT	504	573			
ROW120	60				
ROW120	61				
RSTANN	1700				
RSTANN	1701				
RSTMS1	1531	1716			
RSTMS1	1532	1717			
RTJLBL	1472				
RTJLBL	1473				
RW0141	460				

```

RW0141    461
SCROL0    705
SCROL0    706
STORFC    1437  1504
STORFC    1440  1505
TOGSHF    1236
TOGSHF    1237
TXTROW    17
TXTROW    20
UPLINK    1204
UPLINK    1205
XECROM    101
XECROM    102
XROMNF    1431
XROMNF    1432

```

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES 04000-5777

*
*

5	FILE	CN2B
6	ENTRY	CAT##2
7	ENTRY	XCAT
8	ENTRY	CNTLOP
9	ENTRY	GTCTR
10	ENTRY	SSTCAT
11	ENTRY	R/SCAT
12	ENTRY	BSTCAT
13	ENTRY	CAT##1
14	ENTRY	BAKAPH
15	ENTRY	BAKDE
16	ENTRY	BLINK
17	ENTRY	BLINK1
18	ENTRY	DEEXP
19	ENTRY	DEROVF
20	ENTRY	DERUN
21	ENTRY	DIGENT
22	ENTRY	DIGST*
23	ENTRY	DSPCA
24	ENTRY	DSPCRG
25	ENTRY	FIX57
26	ENTRY	FORMAT
27	ENTRY	GTRMAD
28	ENTRY	INPTDG
29	ENTRY	LDD.P.
30	ENTRY	HOREG9
31	ENTRY	NOTFIX
32	ENTRY	RFD855
33	ENTRY	RG9LCD
34	ENTRY	ROUND
35	ENTRY	RSTST
36	ENTRY	SETQ=P
37	EJECT	

* GTRMAD - GET XEQ ROM FUNCTION ENTRY ADDR

* CALLING SEQUENCE :

* C<2:0> = LOWER 1&HALF BYTES OF THE XEQ ROM FUNCTION CODE

* GOSUB GTRMAD

* <RETURN HERE IF ROM NOT PULGGED IN OR FC # TOO BIG>

* <RETURN HERE IF ROM PLUGGED IN AND FC # IN LIMIT>

* WHEN RETURN REG.B ALWAYS HAS:

* B<2:0> = FUNCTION NUMBER

* B<4:3> = ROM ID NUMBER

* IF THE FUNCTION FOUND IN THE ROM THEN

* A<3:0> = AUXILIARY ROM FUNCTION EXECOTION ADDRESS

* S2 = BIT 8 OF THE UPPER WORD IN FUNCTION TABLE

* S3 = BIT 9 OF THE UPPER WORD IN FUNCTION TABLE

* USED A,C,B<6:0>, STATUS SET, AND ACTIVE POINTER

```

55 0 GTRMAD 132 C=0 M
56 1 756 C=C+C W
57 2 756 C=C+C W ROM ID IN C<3:2> NOW
58 3 1374 RCR 13 C<4:3> _ ROM ID
59 4 1706 C SR X
60 5 746 C=C+C X
61 6 746 C=C+C X
62 7 1706 C SR X C.X _ FC #
63 10 74 RCR 3 C.X _ ROM ID
64 11 406 A=C X A.X _ ROM ID
65 12 674 RCR 11
66 13 534 PT= 6
67 14 352 BC EX WPT B<4:0> _ ROM ID & FC #
68 15 132 C=0 M
69 16 520 LC 5 START FROM HEX 5000 ( ASL 1 )
70 17 534 PT= 6
71 20 RMAD10 1460 CXISA READ ID FROM ONE PORT
72 21 1546 ? A#C X IS THE ID MATCH ?
73 22 43 GONC RMAD20 ( 26 ) YES
74 23 RMAD15 1042 C=C+1 PT ADDR _ ADDR + HEX 1000
75 24 1743 GONC RMAD10 ( 20 ) CHECK ANOTHER PORT
76 25 1740 RTH ROM NOT PLUGGED IN
77 26 RMAD20 1072 C=C+1 M POINT TO 2ND WORD OF ROM
78 27 306 C=B X LOAD THE FC #
79 30 416 A=C
80 31 1460 CXISA LOAD # OF FC'S IN THE ROM
81 32 1046 C=C+1 X
82 33 1406 ? A<C X IS THE FC IN THE ROM ?
83 34 303 GONC RMAD30 ( 64 ) NO, FC # TOO BIG
84 35 246 AC EX X C.X _ FC #
85 36 746 C=C+C X MULTIPLY FC # BY 2
86 37 572 A=A+1 M POINT TO BEGINNING OF FC TABLE
87 40 132 C=0 M
88 41 674 RCR 11
89 42 1032 C=A+C M C.M _ FUNCTION TABLE ENTRY
90 43 1460 CXISA
91 44 766 C=C+C XS
92 45 766 C=C+C XS
93 46 1074 RCR 2
94 47 1530 ST=C SET TOP 2 BITS TO S3,S2
95 50 374 RCR 10
96 51 416 A=C A<3:2> _ UPPER BYTE OF XADR

```



```

97 52      1074 RCR      2
98 53      1072 C=C+1    M      POINT TO LOWER BYTE OF XADR
99 54      1460 CXISA
100 55     1434 PT=      1      GET LOWER BYTE
101 56      412 A=C      WPT      A(3:0) _ XADR
102 57      74 RCR      3
103 60     532 A=A+C    M      COMPUTE CHIP ADDRESS
104 61     660 C=STK
105 62     1072 C=C+1    M      POINT TO P+2
106 63     740 GOTOC      RETURN TO P+2

*
108 64 RMAD30 1172 C=C-1 M
109      LEGAL
110 65      1363 GOTO   RMAD15 ( 23)

*
*
*
* DIGIT ENTRY
*   DIGIT ENTRY USE REG.9 & STATUS SET IN REG.10
*   TO REMEMBER ALL THE KEYS ENTERED.
*   FORMAT OF REG.9 :
*
*   DIGIT 13 : D.P. POSITION, WHICH IS THE # OF DIGITS
*               BETWEEN D.P. AND DIGIT 3. ITS INITIAL
*               VALUE IS 10.
*   DIGIT 12-3: MANTISSA DIGITS, INITIAL VALUE ARE F'S
*   DIGIT 2 : EXP. SIGN. =0 IF EXP POSITIVE
*               =D IF EXP NEGATIVE
*   DIGIT 1-0 : EXP. DIGITS, INITIAL VALUE ARE F'S
*
*   STATUS INFORMATION :
*       S0 = 1 IF D.P. HIT, OTHERWISE = 0
*       S1 = 1 IF EEX HIT, OTHERWISE = 0
*       S2 = 1 IF MANTISSA NEGATIVE, OTHERWISE = 0
*       S3 = 1 IF MANTISSA NONZERO
*   IF S9=1 MEANS THIS CALL IS FROM DERUN(DIGENT IN RUN TIME),
*   THEN CH3 SHALL NOT CHECK IF MANTISSA ZERO.
*
135      ENTRY   DGENSE8
136 66 DGENSE8 404 S8=    0      SAY NO CHS WHEN X=0
137 67 DIGENT 1070 C=REGH 8
138 70      674 RCR      11
139 71     1530 ST=C
140 72     1534 PT=      12      RESTORE DIGENT STATUS SET
141 73      230 C=G
142 74     1376 ? C#0    S      PUT THE DIGIT TO C(13)
143 75      1 GOLNC     BAKDE    IS THIS A BACK ARROW ?
144 76      2          YES
145 77     1374 RCR      13
146 100     1420 LC      12      LOAD CHS
147 101     1320 LC      11      LOAD EEX
148 102     1220 LC      10      LOAD D.P.
149 103      416 A=C      W      COPY C TO A
150 104     1756 A SL    W      GET READY FOR COMPARISON
151 105     1576 ? A#C    S      IS IT A CHS ?
152 106     623 GONC     DECHS ( 170) YES
153 107     1756 A SL
154 110     1414 ?S1=1
155 111      1 GOLC     DEEXP     YES
156 112      3

```

155	113	1576 ? A#C	S	IS IT A EEX ?
156	114	333 GONC	DEEEX (147)	YES
157	115	1756 A SL		
158	116	1576 ? A#C	S	IS IT A D.P. ?
159	117	243 GONC	DEDP (143)	YES
160	120	1376 ? C#0	S	IT A NONZERO DIGIT ?
161	121	23 GONC	DE200 (123)	NO
162	122	10 S3=	1	REMEMBER MANTISSA NONZERO
163	123	34 PT=	3	FIND A PLACE IN MANTISSA
164	124	1170 C=REGN	9	
165	125	1042 C=C+1	PT	
166	126	33 GONC	DE220 (131)	FOUND THE LAST DIGIT
167	127	1734 INC PT		POINT TO LEFT NEXT DIGIT
168		LEGAL		
169	130	1753 GOTO	DE210 (125)	
170	131	24 ? PT=	3	MANTISSA FULL ?
171	132	1 GOLC	BLINK1	YES, IGNORE THIS ENTRY
171	133	3		
172	134	1614 ?S0=1		D.P. HIT ?
173	135	47 GOC	INDGJ (141)	YES, PUT IT TO MANTISSA
174	136	1170 C=REGN	9	MOVE THE POTENTIAL D.P.
175	137	1176 C=C-1	S	TO RIGHT ONE DIGIT
176	140	1150 REGN=C	9	
177	141	1 GOLONG	INPTDG	
177	142	2		
178	143	1614 ?S0=1		D.P. HIT ALREADY ?
179	144	657 GOC	BLINK1 (231)	YES, IGNORE THIS D.P.
180	145	1610 S0=	1	SAY D.P. HIT
181	146	333 GOTO	RSTSTJ (201)	PUT THE STATUS BACK
182	147	1170 C=REGN	9	SEE DO WE ALLOW EEX NOW
183	150	416 A=C		
184	151	1334 PT=	13	
185	152	220 LC	2	
186	153	1436 ? A<C	S	HAVE WE GONE TOO FAR ?
187	154	557 GOC	BLINK1 (231)	YES, NOT EXCEPT EEX NOW
188	155	1410 S1=	1	SAY EEX HIT
189	156	14 ?S3=1		MANTISSA ZERO ?
190	157	707 GOC	RSTST (247)	NO
191	160	116 C=0		SET MANTISSA TO 1
192	161	1156 C=C-1		
193	162	126 C=0	XS	
194	163	1334 PT=	13	
195	164	1120 LC	9	D.P. POSITION = 9
196	165	120 LC	1	MANTISSA = 1
197	166	10 S3=	1	REMEMBER MANTISSA NONZERO
198	167	573 GOTO	RSTJR9 (246)	
199	170	1414 ?S1=1		EEX HIT ?
200	171	137 GOC	CHSEXP (204)	YES, CHS OF EXP.
201	172	414 ?S8=1		CHECK X=0 ?
202	173	37 GOC	DECHS1 (176)	NO
203	174	14 ?S3=1		MANTISSA NONZERO
204	175	343 GONC	BLINK1 (231)	YES, IGNORE CHS
205	176	1014 ?S2=1		CHS HIT ?
206	177	37 GOC	**+3 (202)	YES
207	200	1010 S2=	1	
208	201	463 GOTO	RSTST (247)	PUT STATUS BACK
209	202	1004 S2=	0	
210	203	443 GOTO	RSTST (247)	PUT STATUS BACK
211	204	1170 C=REGN	9	
212	205	416 A=C	W	

```

213 206          126 C=0    XS          ASSUME EXP WAS NEGATIVE
214 207          1034 PT=   2
215 210          1526 ? A#0 XS          WAS EXP NEGATIVE ?
216 211          27 GOC    **2    ( 213) YES
217 212          1520 LC    13          LOAD A "D" TO REG.9(2)
218 213 RSTOR9J  333 GOTO   RSTOR9 ( 246) RESTORE REG.9
219 214 DEEXP    1576 ? A#C S          IS IT A EEX ?
220 215          143 GONC   BLINK1 ( 231) YES, IGNORE IT
221 216          1756 A SL
222 217          1576 ? A#C S          IS IT A D.P. ?
223 220          113 GONC   BLINK1 ( 231) YES, IGNORE IT
224 221          1634 PT=   0          FIND A PLACE IN EXP
225 222          1170 C=REGN 9
226 223 EXPDG1   1042 C=C+1 PT
227 224          33 GONC    EXPDG2 ( 227) FOUND THE LAST EXP DIGIT
228 225          1734 INC PT
229          LEGAL
230 226          1753 GOTO   EXPDG1 ( 223)
231 227 EXPDG2   1624 ? PT=   0          EXP FULL ?
232 230          103 GONC    INPTDG ( 240) NOT YET

*
234          BLINK1
235          BLINK
236 231          1340 DISOFF
237 232          460 LDI
238 233          320 CON    208
239 234          1146 C=C-1 X
240 235          1773 GONC  *-1    ( 234)
241 236          1440 DISTOG
242 237          1740 RTN
243 240 INPTDG   1724 DEC PT
244 241          0 NOP
245 242          230 C=G
246 243          402 A=C    PT
247 244          1170 C=REGN 9
248 245          242 AC EX  PT
249 246 RSTOR9   1150 REGN=C 9
250 247 RSTST    1070 C=REGN 8
251 250          674 RCR    11
252 251          1630 C=ST
253 252          74 RCR     3
254 253          1050 REGN=C 8
255 254          1740 RTN

*
* DERUN - ENTRY POINT OF DIGIT ENTRY IN RUN TIME
* COME IN FROM MAINLOOP WITH THE 1ST DIGIT IN A[3:2], PC POINT
* TO 1ST BYTE OF DIGIT ENTRY IN MEM.
*
261 255 DERUN    1034 PT=   2
262 256          256 AC EX
263 257          130 G=C
264 260          410 S8=   1
265
266 261          23 GOTO   DIGST1 ( 263)

*
* DIGST* - DIGIT ENTRY INITIALIZATION
* SET UP REG.9(REFER THE FORMAT TO DIGENT)
* RESET DIGENT STATUS(CHS,D.P.,EEX) AND SAVE IT IN REG.10[1:0]
* PUSH THE STACK IF PUSHFLAG SET
* IF NOT IN PROGRAM, CLEAR X

```

DISOFF IGNORE A KEY, TURN DISPLAY

NOMAS

NOT Manufacturer Supported
recipient agrees NOT to contact manufacturer

INSERT A DIGIT TO REG.9

PUT THE STATUS BITS BACK

* CALLED BY DATAENTRY WITH DIGIT CODE IN G
 * ASSUME CHIP 0 ENABLE
 * NOT A SUBROUTINE, RETURN TO VARIES PLACE

```

*
277 262 DIGST* 404 S8= 0 REMEMBER NOT IN RUN TIME
278 263 DIGST1 116 C=0 W INITIALIZE REG.9
279 264 1156 C=C-1 W EXP POSITIVE
280 265 126 C=0 XS
281 266 1334 PT= 13 INITIAL D.P. POSITION
282 267 1220 LC 10
283 270 1150 REGN=C 9 MOVE STATUS BITS TO REG.10
284 271 1 GOSUB STBT10
284 272 0
285 273 1670 C=REGN 14 LOAD SET #0
286 274 1530 ST=C PROGMOE ?
287 275 14 ?S3=1 NO
288 276 53 GONC DIGST2 ( 303) INCREMENT LINE# BY 1
289 277 1 GOSUB INSSUB
289 300 0 RETURN TO DATAENTRY
290 301 1 GOLONG DAT320
290 302 2
291 303 DIGST2 614 ?S11=1 PUSH FLAG SET ?
292 304 1 GSUBC R^SUB YES, PUSH STACK
292 305 1
293 306 610 S11= 1
294 307 116 C=0
295 310 350 REGN=C 3 CLEAR X
296 311 414 ?S8=1 RUNNING ?
297 312 1 GOLNC DAT231 NO, RETURN TO DATAENTRY
297 313 2
298 DROP THRU TO DERUN

```

* DIGIT ENTRY DURING RUN TIME

```

*
302 314 1 GOSUB GETPC
302 315 0
303 316 212 B=A WPT
304 317 DERUN5 1 GOSUB ENCF00
304 320 0
305 321 1 GOSUB DIGENT
305 322 0
306 323 1 GOSUB NOREG9 NORMALIZE DIGIT ENTRY
306 324 0
307 325 1 GOSUB NBYTAB
307 326 0
308 327 152 AB EX WPT SAVE PC IN B
309 330 1434 PT= 1
310 331 412 A=C WPT AC[2:0] _ NEXT FC
311 332 460 LDI
312 333 35 CON2 1 13
313 334 1542 ? A#C PT IS A ROW 1 FC ?
314 335 77 GOC DERNRT ( 344) NO, EXIT
315 336 1412 ? A<C WPT IS A DIGIT FC ?
316 337 53 GONC DERNRT ( 344) NO, EXIT
317 340 256 AC EX
318 341 1634 PT= 0
319 342 130 G=C PUT THE DIGIT TO G
320 343 1543 GOTO DERUN5 ( 317)

```

* END OF DIGIT ENTRY UPDATE PC

*

```

323 344 DERRRT 156 AB EX
324 345          1 GOSUB DECAD          POINT TO LAST BYTE OF DIGITENTRY
324 346          0
325 347 DERRT1 1 GOSUB PUTPC
325 350          0
326 351          1 GOLONG=NFRPU
326 352          2

```

*
* OVERFLOW- OVERFLOW DETECT BY DIGIT ENTRY ROUTINE
*

```

330 353 DEROVF 156 AB EX
331 354          1140 SETHEX
332 355          34 PT= 3
333 356          1713 GOTO DERRT1 ( 347)

```

*
* CONSTRUCT DIGIT ENTRY DISPLAY FROM REG. 9
* (PLEASE REFER THE REG.9 FORMAT TO DIGENT)
* CALLED BY DATAENTRY. DIGENT ROUTINE ITSELF WON'T REFRESH
* THE DISPLAY, IT ONLY UPDATE THE REG.9. SO, DURING DIGIT
* ENTRY, DATAENTRY HAVE TO CALL THIS ROUTINE FOR EACH DIGIT
* TO REFRESH DISPLAY.
* STATUS BITS MEANING:
* S0 - D.P. HIT
* S2 - CHS HIT
* S5 - DECIMAL POINT FLAG
* S1 - EEX HIT
* S4 - DIGIT GROUPING FLAG

*
* RG9LCD BUILDS REGS A & C AND SETS UP P AND Q FOR A SUBSEQUENT
* CALL TO RFDS55. RFDS55 IS THE ONE THAT ACTUALLY SENDS STUFF TO
* THE LCD.
*

```

350 357 RG9LCD 1070 C=REGN 8          LOAD FLAGS - S2:CHS
351 360          674 RCR 11
352 361          1530 ST=C
353 362          1170 C=REGN 9
354 363          416 A=C W          A _ REG.9
355 364          1 GOSUB ENLCD          ENABLE LCD CHIP
355 365          0
356 366          1 GOSUB LOAD3          LOAD ALL 3'S INTO C
356 367          0
357 370          34 PT= 3          START FROM END OF MANTISSA
358 371 RFDS10 542 A=A+1 PT          FIND THE LAST DIGIT ?
359 372          53 GONC RFDS15 ( 377) YES
360 373          1142 C=C-1 PT          C(PT) _ 2
361 374          676 A=A-1 S          DECREMENT D.P. POS COUNTER
362 375          1734 INC PT          POINT TO LEFT NEXT DIGIT
363          LEGAL
364 376          1733 GOTO RFDS10 ( 371)
365 377 RFDS15 642 A=A-1 PT          RESTORE THE DIGIT
366 400          1414 ?S1=1          EEX HIT ?
367 401          107 GOC RFDS17 ( 411) YES, DON'T PROMPT MANTISSA
368 402          24 ? PT= 3          MANTISSA FULL ?
369 403          67 GOC RFDS17 ( 411) YES, DON'T PROMPT
370 404          1724 DEC PT          POINT TO PROMPT POSITION
371 405          642 A=A-1 PT          A(PT) _ 1
372 406          120 LC 1          UNDER SCORE = "1F"
373 407          1734 INC PT
374 410          1734 INC PT          RESTORE THE POINTER
375 411 RFDS17 1614 ?S0=1          D.P. HIT ?
376 412          133 GONC RFDS25 ( 425) NO, DON'T LOOK FOT D.P.
377 413 RFDS19 676 A=A-1 S          LOOK FOR D.P.

```

```

378 414          37 GOC      RFDS20 ( 417) FOUND IT !
379 415          1734 INC PT    POINT TO LEFT NEXT DIGIT
380              LEGAL
381 416          1753 GOTO      RFDS19 ( 413)
382 417 RFDS20    214 ?S5=1      LOAD THE D.P. TO C
383 420          33 GONC      **3 ( 423) LOAD A COMMA INSTEAD OF
384 421          720 LC        7
385 422          23 GOTO      **2 ( 424)
386 423          1720 LC        15
387 424          1734 INC PT    RESTORE THE POINTER
388 425 RFDS25    114 ?S4=1      GROUPING FLAG SET ?
389 426          233 GONC      RFDS35 ( 451) NO
390 427          1324 ? PT=    13
391 430          177 GOC      RFDS30 ( 447)
392 431 RFDS26    436 A=C        S      A(13) = 3
393 432 RFDS27    676 A=A-1      S      COUNT 3 FROM LEFT
394 433          57 GOC      RFDS28 ( 440) SHALL WE PUT A COMMA HERE ?
395 434          1524 ? PT=    12      REACH LEFT END OF MANTISSA ?
396 435          147 GOC      RFDS35 ( 451) YES, WE ARE DONE
397 436          1734 INC PT    POINT TO LEFT NEXT DIGIT
398              LEGAL
399 437          1733 GOTO      RFDS27 ( 432)
400 440 RFDS28    214 ?S5=1      LOAD A COMMA TO C
401 441          33 GONC      **3 ( 444)
402 442          1720 LC        15
403 443          23 GOTO      **2 ( 445)
404 444          720 LC        7      LOAD A D.P. INSTEAD OF
405 445          1734 INC PT    RESTORE POINTER
406              LEGAL
407 446          1633 GOTO      RFDS26 ( 431)
408 447 RFDS30    276 AC EX      S
409 450          23 GOTO      **2 ( 452)
410 451 RFDS35    436 A=C        S      TAKE CARE OF THE SIGN
411 452          676 A=A-1      S      A(13) = 2
412 453          136 C=0        S      ASSUME POSITIVE MANTISSA
413 454          1334 PT=      13
414 455          1014 ?S2=1      CHS HIT ?
415 456          23 GONC      **2 ( 460) NO, MANTISSA POSITIVE
416 457          1520 LC        13      "-" = 2D
417 460          276 AC EX      S
418 461          340 SEL Q
419 462          1334 PT=      13      Q= 13
420 463          240 SEL P
421 464          1166 C=C-1      XS
422 465          1414 ?S1=1      C(2) = 2
423 466          203 GONC      RFDS50 ( 506) EEX HIT ?
424 467          1434 PT=      1      NO, LET'S GOTO DISPLAY
425 470          542 A=A+1      PT      LOOK AT DIGIT 1
426 471          107 GOC      RFDS40 ( 501) IS THERE A DIGIT THERE ?
427 472          642 A=A-1      PT      NO, LET'S PROMPT AT DIGIT 1
428 473          1634 PT=      0      RESTORE DIGIT 1
429 474          542 A=A+1      PT      LOOK AT DIGIT 0
430 475          57 GOC      RFDS42 ( 502) IS THERE A DIGIT ?
431 476          642 A=A-1      PT      NO, LET'S PROMPT AT DIGIT 0
432 477          1042 C=C+1      PT      RESTORE DIGIT 0
433              LEGAL          C(0) = 3
434 500          43 GOTO      RFDS45 ( 504) WE ARE READY FOR LCD
435 501 RFDS40    12 A=0        WPT
436 502 RFDS42    642 A=A-1      PT
437 503          120 LC        1

```

```

438 504 RFD$45 34 PT= 3 SAY DISPLAY EXP
439 505 1740 RTN
440 506 RFD$50 26 A=0 XS SAY ONLY DISPLAY MANTISSA
441 507 1634 PT= 0
442 510 1740 RTN
443 511 RFD$55 1722 C SR PQ DISPLAY ONLY 12 DIGITS
444 512 1722 C SR PQ
445 513 150 SRLDB SHIFT INTO DISPLAY REG.B
446 514 256 AC EX W
447 515 1722 C SR PQ
448 516 1722 C SR PQ
449 517 50 SRLDA SHIFT INTO DISPLAY REG.A
450 520 116 C=0 W
451 521 250 SRLDC
452 522 ENCP00 106 C=0 X CLEAR DISPLAY REG.C
453 523 1760 PFAD=C ENABLE CHIP 0 & RETURN
454 524 1160 DADD=C
455 525 1740 RTN DISABLE PERIPHERALS
456 525 1740 RTN ENABLE CHIP 0

*
* PGMAON - TURN ON PROGRAM ANNUNCIATOR
* NO ENTRY REQUIREMENTS
* LEAVES CHIP 0 ENABLED ON EXIT
* USES C AND ONE SUBROUTINE LEVEL
*
463 526 PGMAON 1 ENTRY PGMAON
464 527 0 GOSUB ENLCO
465 530 570 READEN
466 531 1730 CST EX
467 532 1410 S1= 1 TURN ON PRGM ANNUNCIATOR
468 533 1730 CST EX
469 534 1360 WRTEN
470 535 1653 GOTO ENCP00 ( 522)

*
*
* NOREG9 - NORMALIZE THE DIGIT ENTRY STRIN IN REG.9 AND STORE
* IT TO X-REG
* (PLEASE REFER THE INFORMATION TO DIGIT ENTRY)
* ASSUMED CHIP 0 ENABLE. USED A.C. 1 SUB LEVEL.
* RETURN IN HEX MODE, CHIP 0 ENABLE.
* STATUS BITS MEANING :
* S1 - EEK HIT S2 - CHS HIT
* S9 - RUNNING OR SST
*
482 536 NOREG9 1170 C=REGN 9
483 537 34 PT= 3 LOOK FOR LAST DIGIT
484 540 NORGO5 1042 C=C+1 PT
485 541 33 GONC NORG10 ( 544)
486 542 1734 INC PT POINT TO LEFT NEXT DIGIT
487 543 1753 GOTO NORGO5 ( 540)
488 544 NORG10 1142 C=C-1 PT RESTORE THE DIGIT
489 545 406 A=C X NORMALIZE EXP
490 546 126 C=0 XS
491 547 1634 PT= 0
492 550 NORG20 542 A=A+1 PT SHIFT BLANK OUT OF EXP
493 551 53 GONC NORG30 ( 556)
494 552 1706 C SR X
495 553 1734 INC PT

```

```

497 554          1024 ? PT= 2
498 555          1733 GONC   NORC20 ( 550 )
499 556 NORC30  1240 SETDEC
500 557          1526 ? A#0 XS          EXP SIGN NEGATIVE ?
501 560          23 GONC   **+2    ( 562 ) NO
502 561          1206 C=-C X          COMPLIMENT EXP
503 562          416 A=C   W          COPT C TO A
504 563          1334 PT= 13
505 564          1120 LC    9
506 565          276 AC EX  S          C(13)=# OF DIGITS AFTER D.P.
507 566          736 A=A-C S          A(13)=# OF DIGITS BEFORE D.P..
508 567          167 GOC   NORC50 ( 605 )
509 570 NORC40  1502 ? A#0 PT          LEADING ZERO ?
510 571          167 GOC   NORC51 ( 607 ) NO
511 572          1772 A SL  M          SHIFT OUT LEADING ZERO
512 573          676 A=A-1 S          PASS D.P. ?
513 574          1743 GONC NORC40 ( 570 ) NOT YET
514 575 NORC42  646 A=A-1 X          ZERO FOLLOWS BY D.P.
515 576 NORC45  1502 ? A#0 PT          LEADING ZERO PASS D.P. ?
516 577          157 GOC   NORC55 ( 614 ) NO
517 600          1176 C=C-1 S          END OF MANTISSA ?
518 601          407 GOC   NORC65 ( 641 ) YES, EXIT
519 602          646 A=A-1 X          EXP - EXP-1
520 603          1772 A SL  M          SHIFT OUT LEADING ZERO
521 604          1723 GOTO  NORC45 ( 576 )
522 605 NORC50  36 A=0     S
523 606          1673 GOTO  NORC42 ( 575 )
524 607 NORC51  676 A=A-1 S          PASSED D.P. ?
525 610          47 GOC   NORC55 ( 614 ) YES
526 611          546 A=A+1 X
527 612          1753 GONC  NORC51 ( 607 ) USUALLY NO CARRY HERE
528 613          1743 GOTO  NORC51 ( 607 ) CATCHES CARRIES
529 614 NORC55  36 A=0     S          ASSUME MENTISSA POSITIVE
530 615          1014 ?S2=1          CHS HIT ?
531 616          23 GONC   **+2    ( 620 ) NO
532 617          676 A=A-1 S
533 620          256 AC EX  W          EEX HIT ?
534 621          1414 ?S1=1          NORC70 ( 642 ) NO, DON'T CHECK OVERFLOW
535 622          203 GONC
536 623          1 GOSUB   OVFL10
536 624          0
537 625          350 REGN=C 3
538 626          1524 ? PT= 12          OVERFLOW ?
539 627          147 GOC   NORC75 ( 643 ) NO
540 630          414 ?S8=1          RUNNING ?
541 631          1 GOLC   DEROVF          YES
541 632          3
542 633          1525 CON    @1525          GOSUB PRT13
543 634          674 CON    @674
544 635          1 GOSUB   DATOFF
544 636          0
545 637          1 GOLONG .NFRKB
545 640          2
546 641 NORC65  116 C=0     W
547 642 NORC70  350 REGN=C 3
548 643 NORC75  1140 SETHEX
549 644          1740 RTN

```

*
*
*

* BAKDE - BACK SPACE DURING DIGIT ENTRY
 * BAKDE LIKES DIGENT ONLY UPDATE THE DIGIT ENTRY STRING IN REG.9.
 * ASSUMED CHIP 0 ENABLE. USED A,C. RETURN WITH CHIP 0 ENABLE.

7

```

*
557 645 BAKDE 1170 C=REGN 9
558 646          340 SEL Q
559 647          1534 PT= 12
560 650          240 SEL P
561 651          436 A=C S A(13) _ D.P. POSITION
562 652          1414 ?S1=1 EEX HIT ?
563 653          33 GONC BKDE10 ( 657) NO, LOOK AT MANTISSA
564 654 BKEXP 1634 PT= 0 LAST DIGIT IN EXP
565 655          23 GOTO BKDE10 ( 657) LOOK FOR LAST DIGIT IN EXP
566 656 BKDE10 34 PT= 3 LAST DIGIT IN MANTISSA
567 657          1042 C=C+1 PT
568 660          53 GONC BKDE20 ( 665) FOUND THE LAST DIGIT !
569 661          1142 C=C-1 PT C(PT) _ F
570 662          676 A=A-1 S
571 663          1734 INC PT POINT TO LEFT NEXT DIGIT
572          LEGAL
573 664          1733 GOTO BKDE10 ( 657)
574 665 BKDE20 1414 ?S1=1 EEX HIT ?
575 666          203 GONC BKDE20 ( 706) NO
576 667 BKEX10 1024 ? PT= 2 OVER EXP ?
577 670          107 GOC BKEX20 ( 700) YES, EXP OUT
578 671 BKDG 102 C=0 PT TAKE THE DIGIT OUT
579 672          1362 ? C#0 PQ MANTISSA ZERO ?
580 673          37 GOC BKDG1 ( 676) NO
581 674          1004 S2= 0 MANT. CAN'T BE NEGATIVE ZERO
582 675          4 S3= 0 REMEMBER MANTISSA ZERO
583 676 BKDG1 1720 LC 15
584 677          413 GOTO RSTRG9 ( 740) RESTORE REG.9
585 700 BKEX20 1166 C=C-1 XS
586 701          1166 C=C-1 XS WAS EXP NEGATIVE ?
587 702          23 GONC BKEX30 ( 704) YES
588 703          1404 S1= 0 SAY EEX NOT HIT
589 704 BKEX30 126 C=0 XS
590 705          333 GOTO RSTRG9 ( 740) RESTORE REG.9
591 706 BKMN20 1324 ? PT= 13 PASSED LAST DIGIT ?
592 707          137 GOC BKMN30 ( 722) YES,DIGENT OFF
593 710          1614 ?S0=1 D.P. HIT ?
594 711          57 GOC BKMN25 ( 716) YES
595 712          1524 ? PT= 12 LAST DIGIT IN MANTISSA ?
596 713          77 GOC BKMN30 ( 722) YES DIGENT OUT
597 714          1076 C=C+1 S D.P. POS _ D.P. POS-1
598          LEGAL
599 715          1543 GOTO BKDG ( 671) BACK OUT ONE DIGIT
600 716 BKMN25 676 A=A-1 S IS THIS A D.P. ?
601 717          1523 GONC BKDG ( 671) NO, BACK OUT ONE DIGIT
602 720          1604 S0= 0
603 721          203 GOTO RSTSS ( 741)
604 722 BKMN30 1670 C=REGN 14
605 723          1530 ST=C LOAD SET #0
606 724          14 ?S3=1 PROGMODE ?
607 725          53 GONC BKDE30 ( 732) NO
608 726          1 GOSUB DATOFF
608 727          0
609 730          1 GOLONG LERR120
609 731          2
610 732 BKDE30 116 C=0 W

```

611	733	350	REGN=C 3	CLEAR X
612	734	40	SPOPND	
613	735	460	LDI	
614	736	167	CON2 7 7	LOAD THE "CLX" FC
615	737	1740	RTN	GO BACK TO PARSE
616	740	RSTRG9 1150	REGN=C 9	RESTORE REGISTER 9
617	741	RSTSS 1	GOLONG RSTST	PUT STATUS BACK TO REG.10
617	742	2		

* BAKAPH - BACK SPACE DURING ALPHA ENTRY(ONLY IN NROMAL MODE)

* ASSUMED CHIP 0 ENABLE.

* USED A,C. N[13] & B[13] USED AS LCD COUNTER.

* KEY BOARD HAS NOT BEEN RESET

623	743	BAKAPH 1104	S9= 0	KEY BOARD HAS NOT BEEN RESET
624	744	570	C=REGN 5	
625	745	1434	PT= 1	
626	746	1352	? C#0 WPT	IS ANY CHAR IN ALPHA REG. ?
627	747	1	GOLNC DAT106	NO. DO A CLA
627	750	2		
628	751	416	A=C W	SHIFT THE LAST CHAR OUT
629	752	670	C=REGN 6	
630	753	252	AC EX WPT	
631	754	256	AC EX W	
632	755	1074	RCR 2	
633	756	550	REGN=C 5	
634	757	770	C=REGN 7	
635	760	252	AC EX WPT	
636	761	256	AC EX W	
637	762	1074	RCR 2	
638	763	650	REGN=C 6	
639	764	1070	C=REGN 8	
640	765	252	AC EX WPT	
641	766	256	AC EX W	
642	767	1074	RCR 2	
643	770	750	REGN=C 7	
644	771	1070	C=REGN 8	
645	772	574	RCR 6	
646	773	112	C=0 WPT	
647	774	374	RCR 10	
648	775	1050	REGN=C 8	
649	776	1170	C=REGN 9	
650	777	1376	? C#0 S	LCD FULL ?
651	1000	473	GONC BKPH50 (1047)	YES, DO ARGOUT AGAIN
652	1001	1	GOSUB ROLBAK	
652	1002	0		
653	1003	1670	FRSABC	READ LAST CHAR FROM LCD
654	1004	1730	CST EX	TEST FOR PUNC. CHAR
655	1005	514	?S6=1	
656	1006	127	GOC BKPH20 (1020)	
657	1007	1214	?S7=1	
658	1010	107	GOC BKPH20 (1020)	
659	1011	1730	CST EX	
660	1012	BKPH10 336	C=B S	
661	1013	1076	C=C+1 S	
662	1014	376	CB EX S	
663	1015	PROMPT 1	GOSUB OPROMT	OUT PUT PROMPT CHAR
663	1016	0		
664	1017	NFRKB1 363	GOTO NFRKB0 (1055)	
665	1020	BKPH20 504	S6= 0	
666	1021	1204	S7= 0	

```

667 1022      1730 CST EX
668 1023      406 A=C      X
669 1024      460 LDI
670 1025      40 CON      @40      LOAD A BLANK
671 1026      1546 ? A#C    X      IS LAST CHAR A BLANK ?
672 1027      127 GOC      BKPH30 (1041) NO
673 1030      106 C=0      X
674 1031      1760 PFAD=C
675 1032      1160 DADD=C
676 1033      1070 C=REGN 8      LOAD LAST CHAT FROM AREG.
677 1034      126 C=0      XS
678 1035      1546 ? A#C    X      IS IT A BLANK ?
679 1036      67 GOC      BKPH40 (1044) NO
680 1037      1 GOSUB      ENLCD
680 1040      0
681 1041 BKPH30 246 AC EX    X
682 1042      1750 SLSABC
683 1043      1523 GOTO      PROMPT (1015)      PUT THE LAST CHAR BACK
684 1044 BKPH40 1 GOSUB      ENLCD
684 1045      0
685 1046      1443 GOTO      BKPH10 (1012)
686 1047 BKPH50 410 S8=      1      NO SCROLL, PROMPT
687 1050      1 GOSUB      ARGOUT
687 1051      0
688 1052      316 C=B
689 1053      1 GOSUB      STOLCC
689 1054      0
690 1055 NFRKB0 1 GOLONG    NFRKB1
690 1056      2
691          ENTRY      XRND

*
* RND FUNCTION
*
695 1057 XRND 1670 C=REGN 14      LOAD DISPLAY FORMAT
696 1060      1074 RCR      2
697 1061      1530 ST=C      LOAD STATUS SET 1
698 1062      406 A=C      X
699 1063      370 C=REGN 3      LOAD THE X
700 1064      404 S8=      0

*
* ROUNDING ROUTINE
* CALLING SEQUENCE :
* C = NORMALIZED NUMBER
* A(2) = DSP #
* S8 = 1 IF CALLED FROM "FORMAT"
*      0 IF CALLED FROM "XRND"
* GOSUB ROUND
* RETURN WITH ROUNDED NUMBER IN REG.C
* USED A,B,C
*
713 1065 ROUND 1240 SETDEC
714 1066      226 B=A      XS
715 1067 ROUNDA 1534 PT=      12      MOVE POINTER TO 12-(DSP# +1)
716 1070 RND20 1024 ? PT=      2      END OF MANTISSA ?
717 1071      447 GOC      RND90 (1135) YES, NO ROUNDING
718 1072      1724 DEC PT
719 1073      666 A=A-1      XS      STOP ?
720 1074      1743 GONC      RND20 (1070) NO, KEEP GOING
721 1075      416 A=C      W      COPY THE NUMBER TO A

```

722	1076	1366 ? C#0	XS	EXP POSITIVE ?
723	1077	453 GONC	RND30 (1144)	YES
724	1100	1214 ?S7=1		FIX MODE ?
725	1101	143 GONC	RND60 (1115)	NO, LET'S ROUND IT UP
726	1102	RND40 1734 INC PT		
727	1103	1524 ? PT=	12	PASS LEFT END OF MANTISSA?
728	1104	437 GOC	RND100 (1147)	YES, FIX MODE INFEASIBLE
729	1105	546 A=A+1	X	KEEP MOVING TO ROUNDING POINT
730	1106	1743 GONC	RND40 (1102)	
731	1107	63 GOTO	RND60 (1115)	LET'S ROUND IT NOW
732	1110	RND70 1724 DEC PT		EXP POSITIVE
733	1111	RND75 1024 ? PT=	2	PASS END OF MANTISSA ?
734	1112	517 GOC	RND120 (1163)	YES, FIX MODE INFEASIBLE
735	1113	646 A=A-1	X	
736	1114	1743 GONC	RND70 (1110)	
737	1115	RND60 56 B=0	W	HERE IS THE ROUNDING
738	1116	212 B=A	WPT	
739	1117	472 A=A+B	M	
740	1120	133 GONC	RND50 (1133)	ROUNDING OK !
741	1121	RND45 416 A=C	W	SAVE THE # IN CASE OF OVERFLOW
742	1122	12 A=0	WPT	
743	1123	406 A=C	X	
744	1124	1434 PT=	1	TEST FOR OVERFLOW NUMBER
745	1125	1052 C=C+1	WPT	
746	1126	117 GOC	RND95 (1137)	OVERFLOW
747	1127	RND47 16 A=0	W	
748	1130	576 A=A+1	S	
749	1131	1616 A SR	W	SET MANTISSA TO 1
750	1132	634 PT=	11	
751	1133	RND50 12 A=0	WPT	
752	1134	272 AC EX	M	C_ROUNDED NUMBER
753	1135	RND90 1140 SETHEX		
754	1136	1740 RTN		
755	1137	RND95 256 AC EX	W	NO ROUNDING FOR OVERFLOW #
756	1140	1526 ? A#0	XS	EXP NEGATIVE ?
757	1141	1743 GONC	RND90 (1135)	NO
758	1142	106 C=0	X	IT'S NOT REALLY A OVERFLW
759	1143	1643 GOTO	RND47 (1127)	
760	1144	RND30 1214 ?S7=1		FIX MODE ?
761	1145	1503 GONC	RND60 (1115)	NO, LET'S ROUND IT UP
762	1146	1433 GOTO	RND75 (1111)	
763	1147	RND100 546 A=A+1	X	
764	1150	43 GONC	RND105 (1154)	
765	1151	232 B=A	M	
766	1152	472 A=A+B	M	
767	1153	1467 GOC	RND45 (1121)	
768	1154	RND105 414 ?S8=1		CALLED FROM "FORMAT" ?
769	1155	37 GOC	RND110 (1160)	YES
770	1156	116 C=0	W	RETURN ZERO
771	1157	1563 GOTO	RND90 (1135)	
772	1160	RND110 1204 S7=	0	DISPLAY THE # IN SCI MODE
773	1161	166 AB EX	XS	
774	1162	1053 GOTO	ROUND4 (1067)	ROUND IT AGAIN
775	1163	RND120 414 ?S8=1		CALLED FROM "FORMAT" ?
776	1164	1513 GONC	RND90 (1135)	NO, NO ROUNDING THEN
777	1165	6 A=0	X	IS FIX MODE FEASIBLE ?
778	1166	1634 PT=	0	
779	1167	642 A=A-1	PT	A(1) _ 9
780	1170	1406 ? A<C	X	EXP < 9
781	1171	1677 GOC	RND110 (1160)	NO, FIX MODE INFEASIBLE

782 1172

1433 GOTO RND90 (1135) FIX MODE , NO ROUNDING

80

*
 *
 * FORMAT ROUTINE - FORMAT A NORMALIZED NUMBER
 * CALLING SEQUENCE :
 * C= NORMALIZED NUMBER
 * GOSUB FORMAT
 * RETURN < A : READY FOR DISPLAY REG.A >
 * < B : READY FOR DISPLAY REG.B >
 * USED A,B,C. ASSUMED CHIP 0 ENABLE
 * USED STATUS BITS 0-8
 * S4 = DIGIT GROUPING FLAG
 * S5 = DECIMAL POINT FLAG
 * S6 = ENG MODE FLAG
 * S7 = FIX MODE FLAG
 * S8 = FIX MODE FEASIBLE FLAG
 *
 * CALLS STBT10, ROUND, LOAD3, LDD.P., SETQ=P
 * PROBABLY USES ONLY ONE ADDITIONAL SUBROUTINE LEVEL
 *

NOMAS

NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

802 1173	FORMAT	356 BC EX W	SAVE THE NUMBER TO B
803 1174		1140 SETHEX	
804 1175		1 GOSUB STBT10	MOVE STATUS BITS TO REG.10
804 1176		0	
805 1177		316 C=B W	GET THE NUMBER BACK
806 1200		410 S8= 1	SIGNAL ROUNDING ROUTINE
807 1201		1 GOSUB ROUND	ROUND THE NUMBER
807 1202		0	
808 1203		356 BC EX W	MOVE THE NUMBER TO B TEMP.
809 1204		1 GOSUB LOAD3	LOAD ALL 3'S TO C
809 1205		0	
810 1206		1056 C=C+1	C(0) _ 3
811 1207		416 A=C W	A _ ALL 3'S
812 1210		1240 SETDEC	
813 1211		1670 C=REGN 14	
814 1212		274 RCR 5	C(13) _ DSP #
815 1213		356 BC EX W	B(13)_DSP# , C_ROUNDED NO.
816 1214		176 AB EX S	A(13)_DSP# , B(13)_3
817 1215		346 B=C X	COPY EXP TO B
817 1216		306	
818 1217		1376 ? C#0 S	MANTISSA POSITIVE ?
819 1220		33 GONC *+3 (1223) YES	
820 1221		1334 PT= 13	
821 1222		1520 LC 13	LOAD THE MINUS SIGN
822 1223		1214 ?S7=1	FIX MODE ?
823 1224		1 GOLNC NOTFIX	NO
823 1225		2	
824 1226	FIX00	1534 PT= 12	
825 1227		1366 ? C#0 XS	EXP POSITIVE ?
826 1230		133 GONC FIX20 (1243) YES	
827 1231		1 GOSUB LDD.P.	LOAD DECIMAL POINT
827 1232		0	
828 1233		1 GOSUB SETQ=P	
828 1234		0	
829 1235	FIX10	1724 DEC PT	
830 1236		1732 C SR M	SHIFT IN LEADING ZERO
831 1237		676 A=A-1 S	DECREMENT DSP #
832 1240		1046 C=C+1 X	UNTIL EXP = 0
833 1241		1743 GONC FIX10 (1235)	
834 1242		173 GOTO FIX40 (1261)	PUT IN THE TAIL BLANKS

835	1243	FIX20	1146	C=C-1	X	PASSING D.P. ?
836	1244		37	GOC	FIX30	(1247) YES, GOTO LOAD D.P.
837	1245		1724	DEC PT		
838				LEGAL		
839	1246		1753	GOTO	FIX20	(1243)
840	1247	FIX30	106	C=0	X	
841	1250		676	A=A-1	S	
842	1251		477	GOC	FIX60	(1320) FIX MODE, DSP# = 0
843	1252		1	GOSUB	LDD.P.	LOAD THE D.P.
843	1253		0			
844	1254		1	GOSUB	SETQ=P	SET Q=P
844	1255		0			
845	1256	FIX35	1724	DEC PT		PASSING THE DSP #
846	1257		1024	? PT=	2	END OF MANTISSA ?
847	1260		107	GOC	FIX50	(1270) YES
848	1261	FIX40	676	A=A-1	S	DSP# - DSP# -1
849	1262		1743	GONC	FIX35	(1256)
850	1263	FIX45	1724	DEC PT		
851	1264		1024	? PT=	2	END OF MANTISSA ?
852	1265		37	GOC	FIX50	(1270)
853	1266		642	A=A-1	PT	FILLING TAILING BLANK
854				LEGAL		
855	1267		1743	GOTO	FIX45	(1263)
856	1270	FIX50	1634	PT=	0	
857	1271		340	SEL Q		
858	1272		114	?S4=1		GROUPING FLAG SET ?
859	1273		103	GONC	FIX57	(1303) NO
860	1274	SETCOM	176	A=B	S	A.S _ 3(COMMA COUNTER)
860	1275		236			
861	1276	FIX55	676	A=A-1	S	COUNT 3 AND LOAD A COMMA
862	1277		107	GOC	LDCOMA	(1307)
863	1300		1734	INC PT		MOVE THE POINTER TO LEFT
864	1301		1324	? PT=	13	
865	1302		1743	GONC	FIX55	(1276)
866	1303	FIX57	1334	PT=	13	Q _ 13
867	1304		240	SEL P		
868	1305		176	AB EX	S	A(13) _ 3
869	1306		203	GOTO	FMTRTH	(1326)
870	1307	LDCOMA	242	AC EX	PT	
871	1310		214	?S5=1		LOAD A COMMA
872	1311		33	GONC	**+3	(1314)
873	1312		1720	LC	15	
874	1313		23	GOTO	**+2	(1315)
875	1314		720	LC	7	LOAD A D.P. INSTEAD OF
876	1315		1734	INC PT		
877	1316		242	AC EX	PT	
878	1317		1553	GOTO	SETCOM	(1274)
879	1320	FIX60	1	GOSUB	SETQ=P	
879	1321		0			
880	1322		114	?S4=1		
881	1323		1	GOSUB	LDD.P.	
881	1324		1			
882	1325		1363	GOTO	FIX45	(1263)
883	1326	FMTRTH	676	A=A-1	S	A(13) _ 2
884	1327		666	A=A-1	XS	A(2) _ 2
885	1330		256	AC EX	W	
886	1331		1140	SETHX		
887	1332		356	BC EX		
888	1333		1	GOLONG	LDSST0	
888	1334		2			

889	1335	NOTFIX	1534	PT=	12	
890	1336		514	?S6=1		ENG MODE ?
891	1337		623	GONC	SCI00 (1421)	NO, SCI MODE
892	1340		406	A=C	X	A.X _ EXP
893	1341		460	LDI		
894	1342		3	CON	3	
895	1343		1526	? A#0	XS	EXP NEGATIVE ?
896	1344		57	GOC	ENG10 (1351)	YES
897	1345		706	A=A-C	X	COMPUTE EXP MOD 3
898	1346		1773	GONC	*-1 (1345)	
899	1347		506	A=A+C	X	
900	1350		447	GOC	ENG60 (1414)	
901	1351	ENG10	506	A=A+C	X	ADD 3 TO NEGATIVE EXP
902	1352		1773	GONC	*-1 (1351)	
903	1353	ENG20	306	C=B	X	COPY EXP BACK TO C.X
904	1354		1206	C=-C	X	COMPLIMENT NEGATIVE EXP
905	1355		1006	C=C+A	X	
906	1356	ENG25	646	A=A-1	X	MOVE THE D.P. TO RIGHT
907	1357		67	GOC	ENG30 (1365)	
908	1360		1724	DEC PT		
909	1361		676	A=A-1	S	DECREMENT THE DSP #
910	1362		1743	GONC	ENG25 (1356)	
911	1363		36	A=0	S	DSP# _ 0
912	1364		1723	GOTO	ENG25 (1356)	
913	1365	ENG30	1	GOSUB	LDD.P.	
913	1366		0			
914	1367	ENG35	1724	DEC PT		PASSING DSP #
915	1370		1024	? PT=	2	
916	1371		77	GOC	ENG45 (1400)	
917	1372		676	A=A-1	S	
918	1373		1743	GONC	ENG35 (1367)	
919	1374	ENG40	642	A=A-1	PT	A(PT)_2, FILL TAILING BLANK
920	1375		1724	DEC PT		
921	1376		1024	? PT=	2	END OF MANTISSA ?
922	1377		1753	GONC	ENG40 (1374)	
923	1400	ENG45	1326	? B#0	XS	EXP NEGATIVE ?
924	1401		33	GONC	ENG50 (1404)	NO
925	1402		1034	PT=	2	
926	1403		1520	LC	13	LOAD THE MINUS SIGN
927	1404	ENG50	34	PT=	3	
928	1405		246	AC EX	X	
929	1406		460	LDI		
930	1407		1463	CON	01463	
931	1410		246	AC EX	X	
932	1411		340	SEL Q		
933	1412		1	GOLONG	FIX57	
933	1413		2			
934	1414	ENG60	306	C=B	X	C.X _ EXP
935	1415		246	AC EX	X	
936	1416		706	A=A-C	X	
937	1417		246	AC EX	X	
938	1420		1363	GOTO	ENG25 (1356)	
939	1421	SCI00	6	A=0	X	
940	1422		1366	? C#0	XS	EXP POSITIVE ?
941	1423		1307	GOC	ENG20 (1353)	NO
942	1424		1413	GOTO	ENG30 (1365)	
943	1425	SETQ=P	340	SEL Q		
944	1426		1334	PT=	13	
945	1427	SETQ10	440	? P=0		
946	1430		37	GOC	SETQ20 (1433)	

```

947 1431      1724 DEC PT
948           LEGAL
949 1432      1753 GOTO   SETQ10 (1427)
950 1433 SETQ20 240 SEL P
951 1434      1740 RTN
952 1435 LDD.P. 242 AC EX PT
953           ENTRY LDDP10      FOR PRINTER ROM
954 1436 LDDP10 214 ?S5=1
955 1437           33 GONC  **3   (1442)
956 1440           720 LC    7
957 1441           23 GOTO  **2   (1443)
958 1442      1720 LC    15      LOAD A COMMA INSTEAD OF
959 1443      1734 INC PT
960 1444      242 AC EX PT
961 1445      1740 RTN

```

```

*
* DSPCRG - OUTPUT REG.C TO LCD
* IF C(13) = 0 OR 9 IT MEANS A NORMALIZED NUMBER
* IF C(13) = 1 IT MEANS A ALPHA STRING
* ASSUMED CHIP 0 ENABLE.
* USED A,B,C,N, STATUS BITS 0-8. RETURN CHIP 0 ENABLE.
* 2 SUB LEVELS.
*

```

```

970 1446 DSPCRG 240 SEL P
971 1447           36 A=0   S
972 1450           576 A=A+1 S      A.S _ 1
973 1451      1576 ? A#C S      IS IT A STRING ?
974 1452           103 GONC VIEW05 (1462) YES
975 1453           1 GOSUB  FORMAT
975 1454           0
976 1455           1 GOSUB  ENLCD
976 1456           0
977 1457           356 CB EX  W
978 1460           1 GOLONG RFDS55
978 1461           2
979 1462 VIEW05 1334 PT=   13
980 1463           1420 LC   12
981 1464           376 BC EX S
982 1465 DSPCA  1340 DISOFF
983 1466           1334 PT=   13
984 1467           1720 LC   15
985 1470           1720 LC   15
986 1471           1574 RCR   12      C(1:0) _ FF:DELIMINATOR
987 1472           160 N=C      SAVE THE REG. IN N
988 1473           1 GOSUB  ENLCD
988 1474           0
989 1475 VIEW20 260 C=N
990 1476           1574 RCR   12      C(1:0) _ OUT GOING CHAR
991 1477           160 N=C
992 1500           406 A=C   X
993 1501           1434 PT=   1
994 1502           1512 ? A#0 WPT      LEADING ZERO ?
995 1503           1723 GONC VIEW20 (1475) YES, IGNORE IT
996 1504           552 A=A+1 WPT      HIT DELIMINATOR ?
997 1505           47 GOC   VIEW30 (1511) YES
998 1506           1 GOSUB  ASCLCD      SEND IT TO LCD
998 1507           0
999 1510           1653 GOTO  VIEW20 (1475)
1000 1511 VIEW30 460 LDI
1001 1512           40 CON   040

```



```

1002 1513          336 C=B      S          LEFT JUSTIFY
1003 1514 VIEW35 1176 C=C-1    S
1004 1515          43 GONC     VIEW40 (1521)
1005 1516          1440 DISTOG
1006 1517          1 GOLONG .ENCF00
1006 1520          2
1007 1521 VIEW40 1750 SLSABC
1008 1522          1723 GOTO    VIEW35 (1514)
1009
1010
1011

```

*THIS IS THE START OF THE CATALOG ROUTINE. CATALOG 2

*DISPLAYS PLUG IN ROM FUNCTIONS.

```

1016 1523 CAT##2 116 C=0
1017 1524          146 AB EX   X
1018 1525          646 A=A-1   X          GET NUMBER
1019 1526          1072 C=C+1   M          ADDR= 2ND WORD OF ROM
1020 1527          534 PT=      6          2ND WORD= # FUNCTIONS IN ROM
1021 1530          420 LC       4
1022 1531          534 PT=      6
1023 1532 NXTROM 1042 C=C+1   PT          ADDR= 2ND WORD OF NEXT ROM
1024 1533          1 GOLC      QUTCAT
1024 1534          3
1025 1535          1460 CXISA
1026 1536          706 A=A-C    X          GET 2ND WORD= # FUNCTIONS
1027 1537          1733 GONC     NXTROM (1532)
1028 1540          506 A=A+C    X          A IS NUMBER IN ROM
1029 1541          32 A=0      M          ADD A TO STRT DEF ADRS
1030 1542          256 AC EX
1031 1543          674 RCR      11
1032 1544          772 C=C+C    M          DOUBLE DISTANCE
1033 1545          1032 C=C+A   M          ADDRESS OF DEF - 1
1034 1546          1072 C=C+1   M          GET ADDRESS OF CHARACTERS
1035 1547          1460 CXISA
1036 1550          346 BC EX    X
1037 1551          1072 C=C+1   M
1038 1552          1460 CXISA
1039 1553          1074 RCR      2
1040 1554          306 C=B      X
1041 1555          1434 PT=     1          BUILD ADDRESS
1042 1556          74 RCR      3
1043 1557          242 AC EX    PT
1044 1560          674 RCR      11
1045 1561          1002 C=C+A   PT
1046 1562          766 C=C+C    XS
1047 1563          766 C=C+C    XS
1048 1564          766 C=C+C    XS
1049 1565          47 GOC       USLNG (1571) USLNG CODE
1050 1566          1174 RCR      9          MICRO DONE
1051 1567          1 GOLONG .END2          PUT OUT PROMPT
1051 1570          2
1052          USLNG
1053 1571          1574 RCR      12
1054 1572          256 AC EX
1055 1573          1010 S2=     1
1056 1574          1 GOSUB     TXTLB1
1056 1575          0
1057 1576          1 GOLONG .END3

```

1057 1577

2

 *CATALOG SUBROUTINES AND ENTRY LOGIC.

***** GET CATALOG NUMBER

1061 1600 XCAT 256 AC EX
 1062 1601 1530 ST=C
 1063 1602 1 GOSUB TONSTF
 1063 1603 0

**IN THE NEXT PART, THE CONTENTS OF THE C REG WILL BE SHOWN.
 **"C"= CATALOG #, "E"= DIGIT OF ENTRY #, "A"= ALPHA CHARACTER.

1066 1604 16 A=0 C= "*****AAA AAA"
 1067 1605 1070 C=REGN 8 *= DON'T KNOW OR DON'T CARE (OR BOTH)
 1068 1606 234 PT= 5 SAVE ALPHA IN A
 1069 1607 412 A=C WPT GET CATALOG #
 1070 1610 1630 C=ST C= "C ***** ***"
 1071 1611 1474 RCR 1 SAVE CATALOG # IN A(S)
 1072 1612 276 AC EX S C= "C 000000AAA AAA"
 1073 1613 256 AC EX
 1074 1614 1050 REGN=C 8 GET CAT # AND ENTRY #, "E"=ENTRY#
 1075 1615 GTCNTR 1070 C=REGN 8 C= "0 000AAAAAC EEE"
 1076 1616 374 RCR 10 MOVE TO NEXT ENTRY
 1077 1617 NOCHG 1046 C=C+1 X CLEAR BST FLAG
 1078 1620 1604 S0= 0 SAVE ENTRY #
 1079 1621 BSTCNT 346 B=C X
 1079 1622 306
 1080 1623 174 RCR 4 C= "C EEE0000AAA AAA"
 1081 1624 1050 REGN=C 8
 1082 1625 1176 C=C-1 S CHECK FOR CAT 0
 1083 1626 1176 C=C-1 S CHECK FOR CAT 1
 1084 1627 547 GOC CAT##1 (1703) CHECK FOR CAT 2
 1085 1630 1176 C=C-1 S
 1086 1631 1 GOLC CAT##2
 1086 1632 3
 1087 1633 1 GOLONG CAT##3
 1087 1634 2

LOAD TIME OUT CONSTANT

1089 1635 CNTLOP 460 LDI
 1090 1636 400 CON @400
 1091 1637 KPCNT 356 BC EX
 1092 1640 1714 CHK KB
 1093 1641 173 GONC DECCNT (1660)
 1094 1642 1040 C=KEYS
 1095 1643 74 RCR 3
 1096 1644 126 C=0 XS
 1097 1645 406 A=C X
 1098 1646 1634 PT= 0
 1099 1647 742 C=C+C PT CHECK FOR "ON" KEY
 1100 1650 1 GOLC OFF
 1100 1651 3
 1101 1652 460 LDI
 1102 1653 207 CON 135 R/S KEY?
 1103 1654 1546 ? A#C X
 1104 1655 XCCTMG 1 GOLNC CLCTMG CLEAR CATALOG AND MESSAGE
 1104 1656 2
 1105 1657 RSTKBD 1710 RST KB
 1106 1660 DECCNT 356 BC EX
 1107 1661 1146 C=C-1 X
 1108 1662 1553 GONC KPCNT (1637)
 1109 1663 1323 GOTO GTCNTR (1615)
 1110 1664 SSTCAT 1 GOSUB SETSST SET SST FLAG

```

1110 1665          0
1111 1666          1273 GOTO   GTCNTR (1615) INC CNT IN B
1112 1667 R/SCAT    1 GOSUB   RSTKB          CLEAR KEYBOARD
1112 1670          0
1113 1671          1243 GOTO   GTCNTR (1615)
1114 1672 BSTCAT    1 GOSUB   SETSST        SET SST FLAG
1114 1673          0
1115 1674          1610 S0=    1          SET BST FLAG
1116 1675          1070 C=REGN-8
1117 1676          374 RCR     10          BST COUNTER
1118 1677          1146 C=C-1  X
1119 1700          1346 ? C#0  X          INDEX#0?
1120 1701          1207 GOC     BSTCNT (1621)
1121 1702          1153 GOTO   NOCHG (1617)

```

*THE ROUTINES WHICH GET AND DISPLAY THE CHARACTERS

*FOR THE THREE TYPES OF CATALOGS ARE LISTED BELOW.

```

1126 1703 CAT##1    34 PT=    3
1127 1704          304 S10=    0          SET RAM FLAG
1128 1705          346 BC EX   X          CHK FOR FIST TIME
1129 1706          1146 C=C-1  X
1130 1707          1146 C=C-1  X
1131 1710          113 GONC    PC      (1721)
1132 1711          1 GOSUB    FSTIN
1132 1712          0
1133 1713          420 LC      4
1134 1714          1604 S0=    0          CLEAR BST FLAG
1135 1715          34 PT=    3
1136 1716          356 CB EX
1137 1717          1 GOSUB    CLRSB2      CLR STK,L#,SAVE NEW PC
1137 1720          0
1138 1721 PC        1 GOSUB    GETPC      FETCH PC TO A(0,3)
1138 1722          0
1139 1723          1570 C=REGN-13
1140 1724          620 LC      6
1141 1725          34 PT=    3
1142 1726          1614 ?S0=1
1143 1727          107 GOC     OVRINC (1737) DO BST?
1144 1730          1552 ? A#C    WPT
1145 1731          1 GOLNC     QUTCAT
1145 1732          2
1146 1733          1 GOSUB    INCAD2
1146 1734          0
1147 1735          1 GOSUB    INCAD      GET BY START OF LINK
1147 1736          0
1148 1737 OVRINC    1 GOSUB    FLINKA
1148 1740          0
1149 1741          1614 ?S0=1          BST
1150 1742          37 GOC     OVRROT (1745) LEAVE A(0:3) ALONE
1151 1743          174 RCR     4
1152 1744          252 AC EX    WPT
1153 1745 OVRROT    1 GOSUB    PUTPCD
1153 1746          0
1154 1747          1 GOSUB    CLLCDE
1154 1750          0
1155 1751          1404 S1=    0          NO SCROLLING
1156 1752          1 GOSUB    DF060
1156 1753          0
1157 1754          1 GOLONG .END3

```

```

1157 1755          2
*****
1159          ENTRY NFRST+
*****
*THIS CODE FINISHES REGISTER ARITHMETIC.
*****
1163 1756 NFRST+      1 GSBLNG OVFL10      CHECK OVERFLOW
1163 1757              0
1164 1760          356 BC EX
1165 1761          324 ? PT= 10          IF PT = 10 OVERFLOW
1166 1762          113 GONC NOOVF (1773)
1167 1763          106 C=0      X      RE-ENABLE CHIP 0
1168 1764          1160 DADD=C
1169 1765          1670 C=REGN 14
1170 1766          574 RCR      6
1171 1767          1530 ST=C
1172 1770          1214 ?S7=1
1173 1771          1 GOLNC ERRIGN      RANGE ERROR IGNORE?
1173 1772              2      NO. GO TEST ERROR IGNORE FLAG
1174 1773 NOOVF      260 C=N
1175 1774          1160 DADD=C
1176 1775          356 BC EX
1177 1776          1360 DATA=C
1178 1777          1740 RTN
1179
1180
1181
1182          UNLIST

ERRORS :      0

```

SYMBOL TABLE

BAKAPH	743	-					
BAKDE	645	-					
BKDE10	657	-	664	655			
BKDE20	665	-	660				
BKDE30	732	-	725				
BKDG	671	-	717	715			
BKDG1	676	-	673				
BKEX10	667	-					
BKEX20	700	-	670				
BKEX30	704	-	702				
BKEXP	654	-					
BKMANT	656	-	653				
BKMH20	706	-	666				
BKMN25	716	-	711				
BKMN30	722	-	713	707			
BKPH10	1012	-	1046				
BKPH20	1020	-	1010	1006			
BKPH30	1041	-	1027				
BKPH40	1044	-	1036				
BKPH50	1047	-	1000				
BLINK	231	-					
BLINK1	231	-	220	215	175	154	144
BSTCAT	1672	-					
BSTCNT	1621	-	1701				
CAT##1	1703	-	1627				
CAT##2	1523	-					
CHSEXP	204	-	171				
CNTLOP	1635	-					
DE200	123	-	121				
DE210	125	-	130				
DE220	131	-	126				
DECCNT	1660	-	1641				
DECHS	170	-	106				
DECHS1	176	-	173				
DEDP	143	-	117				
DEEEX	147	-	114				
DEEXP	214	-					
DERNRT	344	-	337	335			
DERQVF	353	-					
DERRT1	347	-	356				
DERUN	255	-					
DERUN5	317	-	343				
DGENS8	66	-					
DIGENT	67	-					
DIGST*	262	-					
DIGST1	263	-	261				
DIGST2	303	-	276				
DSPCA	1465	-					
DSPCRG	1446	-					
ENCP00	522	-	535				
ENG10	1351	-	1344				
ENG20	1353	-	1423				
ENG25	1356	-	1420	1364	1362		
ENG30	1365	-	1424	1357			
ENG35	1367	-	1373				
ENG40	1374	-	1377				

ENG45	1400	-	1371		
ENG50	1404	-	1401		
ENG60	1414	-	1350		
EXPDG1	223	-	226		
EXPDG2	227	-	224		
FIX00	1226	-			
FIX10	1235	-	1241		
FIX20	1243	-	1246	1230	
FIX30	1247	-	1244		
FIX35	1256	-	1262		
FIX40	1261	-	1242		
FIX45	1263	-	1325	1267	
FIX50	1270	-	1265	1260	
FIX55	1276	-	1302		
FIX57	1303	-	1273		
FIX60	1320	-	1251		
FMTTRN	1326	-	1306		
FORMAT	1173	-			
GTCNTR	1615	-	1671	1666	1663
GTRMAD	0	-			
INDGJ	141	-	135		
INPTDG	240	-	230		
KPCNT	1637	-	1662		
LDCOMA	1307	-	1277		
LDD.P.	1435	-			
LDDP10	1436	-			
NFRKB0	1055	-	1017		
NFRKB1	1017	-			
NFRST+	1756	-			
NOCHG	1617	-	1702		
NOOVF	1773	-	1762		
NOREG9	536	-			
NORG05	540	-	543		
NORG10	544	-	541		
NORG20	550	-	555		
NORG30	556	-	551		
NORG40	570	-	574		
NORG42	575	-	606		
NORG45	576	-	604		
NORG50	605	-	567		
NORG51	607	-	613	612	571
NORG55	614	-	610	577	
NORG65	641	-	601		
NORG70	642	-	622		
NORG75	643	-	627		
NOTFIX	1335	-			
NXTROM	1532	-	1537		
OVRINC	1737	-	1727		
OVRROT	1745	-	1742		
PC	1721	-	1710		
PGMAON	526	-			
PRONPT	1015	-	1043		
R/SCAT	1667	-			
RFDS10	371	-	376		
RFDS15	377	-	372		
RFDS17	411	-	403	401	
RFDS19	413	-	416		
RFDS20	417	-	414		
RFDS25	425	-	412		
RFDS26	431	-	446		

RFDS27	432	-	437	
RFDS28	440	-	433	
RFDS30	447	-	430	
RFDS35	451	-	435	426
RFDS40	501	-	471	
RFDS42	502	-	475	
RFDS45	504	-	500	
RFDS50	506	-	466	
RFDS55	511	-		
RG9LCD	357	-		
RMAD10	20	-	24	
RMAD15	23	-	65	
RMAD20	26	-	22	
RMAD30	64	-	34	
RND100	1147	-	1104	
RND105	1154	-	1150	
RND110	1160	-	1171	1155
RND120	1163	-	1112	
RND20	1070	-	1074	
RND30	1144	-	1077	
RND40	1102	-	1106	
RND45	1121	-	1153	
RND47	1127	-	1143	
RND50	1133	-	1120	
RND60	1115	-	1145	1107 1101
RND70	1110	-	1114	
RND75	1111	-	1146	
RND90	1135	-	1172	1164 1157 1141 1071
RND95	1137	-	1126	
ROUND	1065	-		
ROUND A	1067	-	1162	
RSTKBD	1657	-		
RST09J	213	-		
RST0R9	246	-	213	167
RSTRG9	740	-	705	677
RSTSS	741	-	721	
RSTST	247	-	203	201 157
RSTSTJ	201	-	146	
SCI00	1421	-	1337	
SETCOM	1274	-	1317	
SETQ10	1427	-	1432	
SETQ20	1433	-	1430	
SETQ=P	1425	-		
SSTCAT	1664	-		
USLNG	1571	-	1565	
VIEW05	1462	-	1452	
VIEW20	1475	-	1510	1503
VIEW30	1511	-	1505	
VIEW35	1514	-	1522	
VIEW40	1521	-	1515	
XCAT	1600	-		
XCCTMG	1655	-		
XRND	1057	-		

NOMAS
 Not Manufacturer Supported
 recipient agrees NOT to contact manufacturer

ENTRY TABLE

BAKAPH	743	-
BAKDE	645	-
BLINK	231	-
BLINK1	231	-
BSTCAT	1672	-
CAT##1	1703	-
CAT##2	1523	-
CNTLOP	1635	-
DEEXP	214	-
DEROVF	353	-
DERUN	255	-
DGENSB	66	-
DIGENT	67	-
DIGST*	262	-
DSPCA	1465	-
DSPCRG	1446	-
ENCP00	522	-
FIX57	1303	-
FORMAT	1173	-
GTCNTR	1615	-
GTRMAD	0	-
INPTDG	240	-
LDD.P.	1435	-
LDDP10	1436	-
NFRST+	1756	-
NOREG9	536	-
NOTFIX	1335	-
PGMA0N	526	-
R/SCAT	1667	-
RFDS55	511	-
RG9LCD	357	-
ROUND	1065	-
RSTST	247	-
SETQ=P	1425	-
SSTCAT	1664	-
XCAT	1600	-
XRND	1057	-

EXTERNAL REFERENCES

ARGOUT	1050				
ARGOUT	1051				
ASCLCD	1506				
ASCLCD	1507				
BAKDE	75				
BAKDE	76				
BLINK1	132				
BLINK1	133				
CAT##2	1631				
CAT##2	1632				
CAT##3	1633				
CAT##3	1634				
CLCTMG	1655				
CLCTMG	1656				
CLLCDE	1747				
CLLCDE	1750				
CLRSB2	1717				
CLRSB2	1720				
DAT106	747				
DAT106	750				
DAT231	312				
DAT231	313				
DAT320	301				
DAT320	302				
DATOFF	635	726			
DATOFF	636	727			
DECAD	345				
DECAD	346				
DEEXP	111				
DEEXP	112				
DEROVF	631				
DEROVF	632				
DF060	1752				
DF060	1753				
DIGENT	321				
DIGENT	322				
ENCP00	317	1517			
ENCP00	320	1520			
END2	1567				
END2	1570				
END3	1576	1754			
END3	1577	1755			
ENLCD	364	526	1037	1044	1455 1473
ENLCD	365	527	1040	1045	1456 1474
ERR120	730				
ERR120	731				
ERRIGN	1771				
ERRIGN	1772				
FIX57	1412				
FIX57	1413				
FLINKA	1737				
FLINKA	1740				
FORMAT	1453				
FORMAT	1454				
FSTIN	1711				
FSTIN	1712				

GETPC	314	1721		
GETPC	315	1722		
INCAD	1735			
INCAD	1736			
INCAD2	1733			
INCAD2	1734			
INPTDG	141			
INPTDG	142			
INSSUB	277			
INSSUB	300			
LDD.P.	1231	1252	1323	1365
LDD.P.	1232	1253	1324	1366
LDSST0	1333			
LDSST0	1334			
LOAD3	366	1204		
LOAD3	367	1205		
NBYTAB	325			
NBYTAB	326			
NFRKB	637			
NFRKB	640			
NFRKB1	1055			
NFRKB1	1056			
NFRPU	351			
NFRPU	352			
NOREG9	323			
NOREG9	324			
NOTFIX	1224			
NOTFIX	1225			
OFF	1650			
OFF	1651			
OPROMT	1015			
OPROMT	1016			
OVFL10	623	1756		
OVFL10	624	1757		
PUTPC	347			
PUTPC	350			
PUTPCD	1745			
PUTPCD	1746			
QUTCAT	1533	1731		
QUTCAT	1534	1732		
RFDS55	1460			
RFDS55	1461			
ROLBAK	1001			
ROLBAK	1002			
ROUND	1201			
ROUND	1202			
RSTKB	1667			
RSTKB	1670			
RSTST	741			
RSTST	742			
R*SUB	304			
R*SUB	305			
SETQ=P	1233	1254	1320	
SETQ=P	1234	1255	1321	
SETSST	1664	1672		
SETSST	1665	1673		
STBT10	271	1175		
STBT10	272	1176		
STOLCC	1053			
STOLCC	1054			

TONSTF 1602
 TONSTF 1603
 TXTLB1 1574
 TXTLB1 1575

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES 06000-7777

*

4	FILE	CN3B
5	ENTRY	DSPLN+
6	ENTRY	ABTSEQ
7	ENTRY	ABTS10
8	ENTRY	AJ2
9	ENTRY	AJ3
10	ENTRY	FDIGIT
11	ENTRY	FDIG20
12	ENTRY	IND
13	ENTRY	MIDDIG
14	ENTRY	NEXT
15	ENTRY	NEXT1
16	ENTRY	NEXT2
17	ENTRY	NEXT3
18	ENTRY	NLT000
19	ENTRY	NLT020
20	ENTRY	NULT#
21	ENTRY	NULT#3
22	ENTRY	NULT#5
23	ENTRY	NULTST
24	ENTRY	PAR112
25	ENTRY	PARSE
26	ENTRY	PARSEB
27	ENTRY	PARS56
28	ENTRY	PARA60
29	ENTRY	PARS75
30	ENTRY	STK

*

* CLRSB2 - CLEAR USER SUBROUTINE STACK AND CLOBBER LINE NUMBER
 * ON ENTRY - PT=3, CHIP 0 ENABLED, NEW PC IN B[3:0] IN MM FORM
 * USES B[3:0], A[3:0], C
 * EXITS VIA PUTPCX, WHICH CLOBBERS THE LINE NUMBER ONLY IF S13=0.

*

* CLRSB3 - ENTRY POINT TO FINISH PUSHING THE SUBROUTINE STACK
 * ON ENTRY - C[13:4] HAS WHAT SHOULD GO INTO REG 12[13:4],
 * PT=3, CHIP 0 ENABLED, NEW PC IN B[3:0] IN MM FORM
 * OTHERWISE THE SAME AS CLRSB2

*

42	ENTRY	CLRSB2
43	ENTRY	CLRSB3
44	0	CLRSB2 116 C=0
45	1	1350 REGN=C 11
46	2	CLRSB3 1450 REGN=C 12
47	3	152 AB EX WPT
48	4	263 GOTO CLRSBX (32)
49		FILLTO 04

*

* PARSE - KEY SEQUENCE PARSER
 * ENTRY CONDITIONS: CHIP 0 SELECTED, HEX, P SEL

```

*
54      PARSE
55      5      1670 C=REGN 14      LOAD STATUS SET 1/2
56      6      1474 RCR      1
57      7      1530 ST=C
58      10     640 CLRABC
59      11     1040 C=KEYS
60      12     274 RCR      5      KC TO C[13:12]
61      13     460 LDI
62      14     301 CON      @301    @6020\16
63      15     374 RCR      10
64      16     34 PT=      3
65      17     740 GOTOC
66          FILLTO:017
67      20     0 NOP      CAUSES COL 0 TO MAP
68          ONTO COLUMN 1
69      21     20 LC      0      1
70      22     133 GOTO    PAR003 ( 35) 2
71      23     120 LC      1      3
72      24     113 GOTO    PAR003 ( 35) 4
73      25 PAR001 220 LC      2      5
74      26     73 GOTO    PAR003 ( 35) 6
75      27     1763 GOTO   PAR001 ( 25) 7
76      30     320 LC      3      8
77      31     43 GOTO    PAR003 ( 35) 9
78      32 CLRSBX 1 GOLONG-PUTPCX
78      33     2
79          FILLTO:033
80      34     420 LC      4      C
81      35 PAR003 1474 RCR      1      C[2:1]=LOGCOL, ROW
82      36     406 A=C      X      A[2:1]=LOGCOL, ROW
83      37     114 ?S4=1
84      40     43 GONC     PAR005 ( 44) NO      SHIFTSET?
85      41     460 LDI
86      42     200 CON2     8      0      ADJ ROW FOR SHIFT
87      43     1006 C=A+C   X
88      44 PAR005 160 N=C
89      45     214 ?S5=1
90      46     603 GONC     NEWFCN ( 126) NO      NC[2:1]=LOG KC
91          PKSEQ?
92          CONTINUING KEY SEQUENCE
93          A[2:1]=LOGCOL, ROW
94          A[0]=0, A.M=0
95          ROW DOES NOT HAVE SHIFT
96          ADJUSTMENT IN IT
97      47     256 AC EX
98      50     1434 PT=      1
99      51     742 C=C+C   PT
100     52     756 C=C+C
101     53     756 C=C+C
102     54     756 C=C+C
103     55     1374 RCR      13
104     56     534 PT=      6
105     57     120 LC      1      PKTTAB IS AT @10000
106     60     1460 CXISA
107     61     1474 RCR      1      CONSTRUCT PTENP1
108     62     746 C=C+C   X
109     63     746 C=C+C   X

```

```

110          ENTRY PARS05
* ENTRY POINT FOR WAND ON 3-13-79
*
113          PARS05
114 64          416 A=C
115 65          1634 PT= 0
116 66          1770 C=REGN 15
117 67          74 RCR 3
118 70          130 G=C
119 71          582 A=A+C PT
120 72          246 AC EX X
121 73          1530 ST=C
122 74          1104 S9= 0
123 75          1 GOSUB ENLCD
123 76          0
124 77          460 LDI
125 100         40 CON 32
126 101         406 A=C X
127 102         206 B=A X
128 103         1434 PT= 1
129 104 PARS10 1670 RABCR
130 105         1552 ? A#C WPT
131 106         1763 GONC PARS10 ( 104 )
132 107         652 A=A-1 WPT
133 110 PARS20 1552 ? A#C WPT
134 111         67 GOC PARS30 ( 117 )
135 112         306 C=B X
136 113         1750 SLSABC
137 114         1650 SRSABC
138 115         1670 RABCR
139 116         1723 GOTO PARS20 ( 110 )
140 117 PARS30 1770 RABCL
*
142 120         576 A=A+1 S
143 121         1540 RTN C
144 122         676 A=A-1 S
145 123         660 C=STK
146 124         1072 C=C+1 M
147 125         740 GOTOC
148
*
* NEWFCN - NEW FUNCTION
* FIRST KEY OF A NEW KEY SEQUENCE
152          NEWFCN
153
154 126         406 A=C X
155 127         1670 C=REGN 14
156 130         1530 ST=C
157 131         1214 ?S7=1
158 132         43 GONC PARS50 ( 136 ) NO
159 133         460 LDI
160 134         525 CON 0525
161 135         563 GOTO PARS55 ( 213 )
*
163 136 PARS50 574 RCR 6
164 137         1530 ST=C
165 140         114 ?S4=1
166 141         463 GONC PARS52 ( 207 ) NO
167 142         1 GOSUB TBITHP
167 143         0

```

RESTORE PTEMP2 TO G
MERGE OPERAND TYPE INFO

PUT UP PTEMP1
SAY ADDRESS NOT FOUND YET
TURN ON LCD CHIP

BLANK
BLANK TO A.X
AND B.X

RIGHT JUSTIFY LCD

TURN BLANK INTO PROMPT
NOT A PROMPT?
NOT A PROMPT
RETRIEVE BLANK
GET RID OF PROMPT

SHIFT OFF SOMETHING

CHECK FOR BACKARROW

INCREMENT RETURN ADDRESS
ON EXIT, PT=1, LCD CHIP ON,
SS PTEMP1 UP, B.X=BLANK

ON ENTRY, SS1/2 UP, CHIP 0
ON, KC IN C[2:1], B=0
A[2:1]=LOG KC

PUT UP SS0
ALPHAMODE?

H1550\16=0525

PUT UP SS3
USERMODE?

YES. TEST BIT MAP

```

168 144      1356 ? C#0      KEY REASSIGNED?
169 145      1 GOLC   RAK60   YES
169 146      3
170 147      1670 C=REGN 14
171 150      1530 ST=C      PUT UP SS0 AGAIN
172 151      14 ?S3=1      PRGNMODE?
173 152      337 GOC   RAK10  ( 205) YES - SKIP AUTO-ASSIGN TESTS
174 153      260 C=N
175 154      132 C=0   M
176 155      1074 RCR   2      LOG ROW TO C.S
177 156      406 A=C   X      LOG COL TO A.X
178 157      460 LDI
179 160      146 CON2   6      6      ROW 0 OFFSET
180 161      1176 C=C-1 S      ROW 0?
181 162      147 GOC   RAK05  ( 176) YES
182 163      1634 PT=   0
183 164      1320 LC    11      SET UP ROW 1 OFFSET
184 165      1376 ? C#0 S      ROW#1?
185 166      103 GONC   RAK05  ( 176) ROW 1
186 167      1434 PT=   1
187 170      720 LC     7      SET UP SHIFTED ROW 0 OFFSET
188 171      1076 C=C+1 S
189 172      776 C=C+C S      SHIFTED?
190 173      123 GONC   RAK10  ( 205) NO
191 174      1376 ? C#0 S      NOT SHIFTED ROW 0?
192 175      107 GOC   RAK10  ( 205) NOT AUTO-ASSIGNED
193 176 RAK05 1006 C=A+C X      C.X=IMPLIED LOCAL LABEL

*
195      ENTRY RAK06
* ENTRY POINT ADD FOR WAND ON 3-13-79
*
198      RAK06
199 177      530 M=C      SAVE OPERAND IN M
200 200      416 A=C      SET UP A[1:0] FOR SEARCH
201 201      1 GOSUB SEARCH
201 202      0
202 203      1356 ? C#0      FOUND?
203 204      607 GOC   PARS60 ( 264) YES
204      RAK10      KEY IS NOT REASSIGNED
205 205      260 C=N      RETRIEVE LOGICAL KC
206 206      416 A=C      RESTORE LOG KC TO A[2:1]
207 207 PARS52 1670 C=REGN 14
208 210      1530 ST=C      PUT UP SS0
209 211      460 LDI
210 212      520 CON     0520      H1500\16=0520
211      PARS55      NORMAL MODE DEFAULT TABLE
212      ST HAS SS0
213      LOG KC IN A[2:1]
214      DEFAULT TABLE ADDR\16 IN C.X
215 213      136 C=0   S
216 214      1574 RCR   12
217 215      1006 C=A+C X
218 216      1574 RCR   12      C[6:3]=TABLE ADDRESS
219 217      1460 CXISA

*
221      ENTRY PARSDE
* ENTRY POINT FOR WAND TO EXECUTE DATA ENTRY KEY (3-15-79)
*
224      PARSDE
225 220      1166 C=C-1 XS      DATA ENTRY KEY?

```

226	221	1	GSUBC	DATENT	GOES TO DATENT W/ ASCII
226	222	1			
227					OR DE FC OR 0 FOR BKARROW
228					IN C[1:0] & W/ SS0 UP
229					CHS, CLX, DELETE,
230	PARS56				STORAGE ARITHMETIC,
231					AND STOP RE-ENTER HERE.
232					ENTRY REQUIREMENTS:
233					FC IN C.X, CHIP 0 ON,
234					SS 0 UP
235	223	674	RCR	11	FC TO DIGITS 4,3
236	224	356	BC	EX	SAVE IN B
237	225	1270	C=REGN	10	
238	226	134	PT=	4	
239	227	312	C=B	WPT	MERGE FC TO DIGITS 4,3 OF REG 10
240	230	106	C=0	X	
241	231	1250	REGN=C	10	
242	232	14	?S3=1		PROGRAM MODE?
243	233	53	GONC	PARS57 (240)	
244	234	1342	? C#0	PT	PROGRAMMABLE?
245	235	33	GONC	PARS57 (240)	NOT PROGRAMMABLE
246	236	1434	PT=	1	
247	237	1042	C=C+1	PT	SET INSERT BIT
248	240	PARS57 1634	PT=	0	SAVE PTEMP2 IN G
249	241	130	G=C		
250	242	1530	ST=C		PUT UP PTEMP2
251	243	316	C=B		RECOVER FC TO C[4:3]
252	244	274	RCR	5	FC TO DIGITS 13:12
253	245	460	LDI		
254	246	24	DEF	024 (24)	024=012000\256 MAIN FCN TABLE
255	247	1174	RCR	9	
256	250	1460	CXISA		GET XADR
257	251	34	PT=	3	
258	252	120	LC	1	
259	253	674	RCR	11	FULL XADR IN C[6:3]
260	254	530	M=C		SAVE XADR IN M
261	255	560	STK=C		AND ON SUBR STACK
262	256	1460	CXISA		GET C(XADR)
263	257	1346	? C#0	X	NOT XKD?
264	260	347	GOC	PARS70 (314)	NOT XKD.
265	261	1670	C=REGN	14	
266	262	1530	ST=C		PUT UP SS 0
267	263	1740	RTN		GO EXECUTE IMMEDIATELY
*					
269		PARS60			
270	264	730	CM	EX	SAVE ADR IN M,
271					RETRIEVE ARGUMENT TO C
272	265	246	AC	EX X	PUT ARG TO A.X FOR ROW940
273	266	206	B=A	X	& SAVE ARG IN B.X
274	267	1	GOSUB	OFSHFT	
274	270	0			
275	271	1	GOSUB	CLLCODE	
275	272	0			
276	273	460	LDI		
277	274	340	CON2	14 0	FC FOR XEQ
278	275	1	GOSUB	PROMF1	PROMPT "XEQ "
278	276	0			
279	277	1404	S1=	0	SET UP FOR ROW940
280	300	1	GOSUB	ROW940	PROMPT ARGUMENT
280	301	0			

281	302	460 LDI			SET UP FOR NLT020
282	303	340 CON2	14	0	FC FOR XEQNN
283	304	1634 PT=	0		CLEAR INSERT BIT IN G
284	305	130 G=C			FOR NLT020
285	306	674 RCR	11		
286	307	416 A=C			
287	310	146 AB EX	X		NOW FC IN A[4:3],
288					ARG IN A[1:0]
* ARG IS PRESERVED HERE FOR THE BENEFIT OF THE PRINTER. SINCE S9 IS					
* SET WHEN WE GET TO XEQ NN, XEQNN NEVER LOOKS AT THE ARGUMENT AT ALL.					
291	311	1110 S9=	1		TELL XEQNN THAT ADDRESS
292					IS ALREADY KNOWN IN M
293	312	1 GOLONG..NULT#5			
293	313	2			
* INITIALIZE S9.					
295	314	PARS70 1104 S9=	0		S9=1 TELLS AXEQ & XEQNN
296					THAT THEIR ADDRESS HAS
297					ALREADY BEEN FOUND AND IS
298					IN M.
299					RETURN FROM AXEQ & RASNKY
300	PARS75				FOR MICROCODED XROM FCNS
301					* ENTRY REQ FOR PARS75:
302					* PTEMP2 IN STATUS BITS
303					* & M=XADR
304					TURN OFF SHIFT
305	315	1 GOSUB	OF SHFT		
305	316	0			
306	317	1 GOSUB	DSPLN+		ENABLE AND CLEAR LCD
306	320	0			
307					IF INSERT THEN INC & DSP LINE#
308	321	630 C=M			RETRIEVE XADR
309	322	1 GOSUB	PROMF2		PROMF2 RETURNS S8=0
309	323	0			
310					XROM MICROCODE FCNS RELY ON
311					S8=0 HERE
312					(WHEN S9 IS SET, S8 TELLS
313					XROM WHETHER THE FCN IS
314					MICROCODE OR USER LANG)
315	324	630 C=M			RETRIEVE XADR AGAIN
316	325	1172 C=C-1	M		POINT TO XADR-1
317	326	1460 CXISA			OP1 TO C.XS
318	327	1366 ? C#0	XS		OP1#0?
319	330	1 GOLNC	NLT000		NO OPERAND
319	331	2			
320	332	766 C=C+C	XS		
321	333	766 C=C+C	XS		
322	334	426 A=C	XS		A.XS=4*OP1
323	335	1172 C=C-1	M		POINT TO XADR-2
324	336	1460 CXISA			C.XS=OP2
325	337	1026 C=A+C	XS		
326	340	1074 RCR	2		
327	341	1530 ST=C			
328	342	4 S3=	0		CLEAR OP1 BIT 1
329	343	1730 CST EX			OP1 BIT 1 STILL EXISTS
330					IN ST, BUT IS CLEAR IN C
331	344	1634 PT=	0		
332	345	242 AC EX	PT		
333	346	230 C=G			
334	347	242 AC EX	PT		MERGE OPTYPE INTO PTEMP2
335	350	130 G=C			PUT PTEMP2 BACK TO G


```

336 351      14 ?S3=1      OP1 BIT 1?
337 352      623 GONC      PARSEA ( 434 ) NO
*
*
340 353 PAR110      1 GOSUB NEXT2
340 354      0
341      ENTRY PAR111      ADDED FOR WAND 11/5/79
342      PAR111
343 355      453 GOTO      ABTSEQ ( 422 ) MUST BE SHORT GTO!!!
344 356      114 ?S4=1      A...J?
345 357      1 GOLC      AJ2      YES
345 360      3
346 361      14 ?S3=1      DIGIT?
347 362      63 GONC      PAR115 ( 370 ) NO
348 363      1 GOSUB      FDIGIT
348 364      0
349 365 PAR112      1 GOSUB      BLINK
349 366      0
350 367      1643 GOTO      PAR110 ( 353 )
351 370 PAR115      1014 ?S2=1      OP1 BIT 0?
352 371      1 GOLC      PARSEB      YES
352 372      3
353 373      514 ?S6=1      SHIFT?
354 374      1 GOLC      IND      YES
354 375      3
355 376      1414 ?S1=1      OP2 BIT 1?
356 377      1667 GOC      PAR112 ( 365 ) YES
357 400 PAR130      1214 ?S7=1      DP?
358 401      1 GOLC      STK      YES
358 402      3
359 403      1614 ?S0=1      OP2 BIT 0?
360 404      1617 GOC      PAR112 ( 365 )
361      MUST BE STO
362 405      1536 ? A#0      S      +-*/ ?
363 406      1573 GONC      PAR112 ( 365 ) NO
364 407      460 LDI      YES
365 410      221 CON      145
366 411      256 AC EX
367 412      1374 RCR      13
368 413      1634 PT=      0
369 414      502 A=A+C      PT
370      LEGAL
371 415      1 GOSUB      LDSST0
371 416      0
372 417      256 AC EX
373 420      1 GOLONG PAR556      START OVER WITH NEW FC
373 421      2

```

* ABTSEQ - ABORT PARTIAL KEY SEQUENCE

* NOTE THAT ABTSEQ DOESN'T CLEAR ALPHAMODE, WHICH MAY BE SET IF WE'RE
 * IN THE MIDDLE OF KEYING IN AN ALPHA OPERAND. IF IT IS DESIRED TO
 * ENSURE THAT THE ALPHAMODE FLAG AND ANNUNCIATOR ARE CLEARED, THE
 * DO A GOLONG TO NAME33, WHICH CLEARS ALPHAMODE AND THEN JUMPS TO
 * ABTSEQ.

```

*
383 422 ABTSEQ      1 GOSUB      CLLCDE      CLEAR DISPLAY
383 423      0
384 424      1 GOSUB      ANNOUT
384 425      0

```

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

```

385 426 ABTS10 1635 CON 01635
386 427 674 CON 0674
387 430 1 GOSUB RSTSEQ
387 431 0
388
389
390 432 1 GOLONG NFRKB
390 433 2
*
392 PARSEA
393
394
395 434 1614 ?S0=1
396 435 33 GONC PARA05 ( 440)
397
398 436 1414 ?S1=1
399 437 263 GONC PARA50 ( 465)
*
*
402 1 DIGIT NUMERIC
403 ALPHA NAME ALPHA
404 440 PARA05 1 GOSUB NEXT1
404 441 0
405 ENTRY PARA06
406 PARA06
407 442 1603 GOTO ABTSEQ ( 422) MUST BE SHORT GTO!!
408 443 1614 ?S0=1 OP2 BIT 0?
409 444 153 GONC PARA45 ( 461)
410 445 114 ?S4=1
411 446 1 GOLC FDIG20
411 447 3
412 450 14 ?S3=1
413 451 1 GOLC FDIG20
413 452 3
414 453 514 ?S6=1
415 454 1 GOLC IND
415 455 3
416 456 PARA10 1 GOSUB BLINK
416 457 0
417 460 1603 GOTO PARA05 ( 440)
*
419 PARA45
420 461 214 ?S5=1
421 462 1 GOLC NAMEA
421 463 3
422 464 1723 GOTO PARA10 ( 456)
*
424 PARA50
425 465 PARA60 1 GOSUB NEXT3
425 466 0
426 ENTRY PARA61
427 PARA61
428 467 1333 GOTO ABTSEQ ( 422)
429 470 114 ?S4=1
430 471 1 GOLC AJ3
430 472 3
431 473 14 ?S3=1
432 474 63 GONC PARA70 ( 502) NO
433 475 1 GOSUB MIDDIG
433 476 0

```

GOSUB PRT4
CLEAR SHIFTSET, PKSEQ,

MSGFLG, DATAENTRY,
CATALOGFLAG, & PAUSING

ALPHA NAME ALPHA
1 DIGIT NUMERIC
3 DIGIT NUMERIC
OP2 BIT 0?

1 DIG OR 3 DIG NUMERIC
OP2 BIT 1?

1 DIGIT NUMERIC
ALPHA NAME ALPHA

ADDED FOR WAND 11/5/79

MUST BE SHORT GTO!!
OP2 BIT 0?

A...J?
YES

DIGIT?
YES

SHIFT?
YES

ALPHA NAME ALPHA
ALPHA KEY?
YES

3 DIGIT NUMERIC

FOR THE WAND

A...J?
YES

DIGIT?
NO

```

434 477 PARA65      1 GOSUB BLINK
434 500              0
435 501            1643 GOTO  PARA60 ( 465)

*
*
438 502 PARA70 1040 C=KEYS          CHECK FOR EEK
439 503          74 RCR      3
440 504          412 A=C      WPT
441 505          460 LDI
442 506          203 CON2      8      3      KC FOR EEK
443 507          1552 ? A#C    WPT
444 510          177 GOC      PRA110 ( 527)

*
446              ENTRY  PARA75
* ENTRY POINT ADD FOR WAND ON '3-13-79
*
449          PARA75
450 511          460 LDI
451 512          61 CON      061
452 513          1750 SLSABC
453 514 PARA80      1 GOSUB  NEXT3
453 515          0
454 516          73 GOTO     PRA100 ( 525)
455 517          14 ?S3=1      DIGIT?
456 520          1 GSUBC  MIDDIG
456 521          1
457 522 PARA90      1 GOSUB  BLINK
457 523          0
458 524          1703 GOTO     PARA80 ( 514)

*
460 525 PRA100 1670 RABCR
461 526          1373 GOTO     PARA60 ( 465)

*
463 527 PRA110 1414 ?S1=1      OP2 BIT 1? (GTO.?)
464 530          1473 GONC      PARA65 ( 477) NO
465 531          1214 ?S7=1      DP KEY?
466 532          147 GOC        PRA115 ( 546) YES
467 533          214 ?S5=1      ALPHA KEY?
468 534          1433 GONC      PARA65 ( 477) NO
469 535          1 GOSUB      ENCP00      YES
469 536          0
470 537          6 A=0      X
471 540          646 A=A-1    X
472 541          646 A=A-1    X
473 542          1270 C=REGN 10
474 543          246 AC EX    X
475 544          1250 REGN=C 10
476 545          403 GOTO     XFRNMA ( 605)
477          PRA115
478 546          1670 RABCR
479 547          1770 RABCL
480 550          1750 SLSABC
481 551          6 A=0      X
482 552          646 A=A-1    X
483 553          1 GOLC      NULT#3
483 554          3

*
485          PARSEB
486
487
INPUT: SS=PTEMP1, LCD ON,
      HEX, PSEL, P=1
      GTO,LBL, AND XEQ

```

```

488 555          1 GOSUB ENCF00      RE-ENABLE CHIP 0
488 556          0
489 557          1270 C=REGN 10      GET PARSE TEMPS
490 560          74 RCR 3           FC NOW IN DIGITS 1:0
491 561          416 A=C           SAVE REG 10 IN A
492
493 562          1614 ?S0=1         OP2 BIT 0?
494 563          247 GOC PARB20 ( 607) YES
495          LBL
496 564          460 LDI           LOAD FC FOR ALBL
497 565          315 CON2 12 13
498 566 PARB10 214 ?S5=1         ALPHA KEY?
499 567          57 GOC PARB15 ( 574)
500 570          1 GOSUB ENLCD
500 571          0
501 572          1 GOLONG PAR112
501 573          2

*
503 574 PARB15 252 AC EX WPT
504 575          256 AC EX
505 576          674 RCR 11
506 577          1250 REGN=C.10
507 600          230 C=G
508 601          1530 ST=C
509 602          210 S5= 1
510 603          1630 C=ST
511 604          130 G=C
512 605 XFRNMA 1 GOLONG NAMEA
512 606          2

*
514 607 PARB20 514 ?S6=1         SHIFT?
515 610          123 GONC PARB30 ( 622)
516 611          460 LDI           LOAD FC FOR GTO/IND
517 612          256 CON2 10 14
518 613          252 AC EX WPT
519 614          256 AC EX
520 615          674 RCR 11
521 616          1250 REGN=C 10
522 617          1414 ?S1=1         OP2 BIT 1?
523 620          527 GOC INDGTO ( 672)
524 621          433 GOTO INDXEQ ( 664)

*
526 622 PARB30 460 LDI
527 623          36 CON2 1 14 FC FOR AXEQ
528 624          1414 ?S1=1         OP2 BIT 1?
529 625          1413 GONC PARB10 ( 566) XEQ
530 626          1146 C=C-1 X      CONVERT TO FC FOR AGTO
531 627          1214 ?S7=1         GTO DP KEY?
532 630          1363 GONC PARB10 ( 566)

*
534          ENTRY PARB40
* ENTRY POINT FOR WAND (3-15-79)
*
537
538          PARB40
539 631          1634 PT= 0
540 632          230 C=G
541 633          1730 CST EX
542 634          104 S4= 0
543 635          1730 CST EX

```

544	636	130	G=C		
545	637	1434	PT=	1	
546	640	460	LDI		FC FOR GTOL
547	641	1	CON	1	
548	642	252	AC EX	WPT	
549	643	256	AC EX		
550	644	674	RCR	11	
551	645	1250	REGN=C	10	
552	646	1	GOSUB	CLLCDE	
552	647	0			
553	650	460	LDI		
554	651	320	CON2	13 0	FC FOR GTO
555	652	1	GOSUB	PROMF1	
555	653	0			
556	654	1650	SRSABC		
557	655	460	LDI		
558	656	140	CON	@140	" , "
559	657	1750	SLSABC		
560	660	1	GOLONG	PARA60	
560	661	2			
561			EJECT		

* IND - TAKES CARE OF INDIRECT OPERANDS

```

*
564 662 IND      1 GOSUB ENCF00
564 663          0
565 664 INDEXEQ 1634 PT=    0
566 665          230 C=G
567 666          1730 CST EX
568 667          510 S6=    1          INDIRECT
569 670          1730 CST EX
570 671          130 G=C
571      INDGTO
572 672          1 GOSUB ENLCD
572 673          0
573 674          1 GOSUB NESSL
573 675          0
574 676          11 CON      9          I
575 677          16 CON     14          N
576 700          4 CON      4          D
577 701          1040 CON    01040      BLANK

```

```

*
580 702 IND20      1 GOSUB NEXT2
580 703          0
581      ENTRY    IND21          ADDED FOR WAND 11/5/79
582      IND21
583 704          553 GOTO    ABTXF3 ( 761 ) MUST BE SHORT GTO!!
584 705          114 ?S4=1    A...J?
585 706          167 GOC      AJ2   ( 724 )
586 707          1214 ?S7=1    DP?
587 710          537 GOC      STK    ( 763 )
588 711          14 ?S3=1      DIGIT?
589 712          33 GONC      IND30 ( 715 )
590 713          1 GOSUB      FDIGIT
590 714          0
591 715 IND30      1 GOSUB      BLINK
591 716          0
592 717          1633 GOTO     IND20 ( 702 )
593      EJECT

```

* AJ3 AND AJ2 - TAKE CARE OF A...J KEYS FOR 3 AND 2 DIGIT OPERANDS

*
 596 AJ3 GTO.--- OR FC---
 597 720 460 LDI
 598 721 60 CON 060 ZERO
 599 722 1750 SLSABC
 600 723 33 GOTO AJ210 (726)

* FC IND-- OR FC--
 602 AJ2
 603 724 460 LDI
 604 725 60 CON 060 ZERO
 605 726 AJ210 1536 ? A#0 S
 606 727 27 GOC AJ220 (731)
 607 730 1056 C=C+1
 608 731 AJ220 1750 SLSABC
 609 732 1474 RCR 1
 610 733 276 AC EX S
 611 734 1374 RCR 13
 612 735 1750 SLSABC
 613 736 1 GOLONG NULT#
 613 737 2

* MIDDIG
 615
 616 740 276 AC EX S
 617 741 1374 RCR 13
 618 742 320 LC 3
 619 743 126 C=0 XS
 620 744 1750 SLSABC
 621 745 MID10 1 GOSUB NEXT2
 621 746 0
 622 747 103 GOTO MID20 (757)
 623 750 14 ?S3=1 DIGIT?
 624 751 33 GONC MID15 (754)
 625 752 1 GOSUB FDIGIT
 625 753 0
 626 754 MID15 1 GOSUB BLINK
 626 755 0
 627 756 1673 GOTO MID10 (745)

* MID20
 629
 630 757 1670 RABCR
 631 760 1740 RTN

*
 633 761 ABTXF3 1 GOLONG ABTSEQ
 633 762 2

* STK - HANDLES STACK REGISTER OPERANDS X,Y,Z,T,L

*
 637 STK
 638 763 STK03 1 GOSUB MESSL
 638 764 0
 639 765 23 CON 023 S
 640 766 24 CON 024 T
 641 767 1040 CON 01040 BLANK
 642 770 1 GOSUB NEXT1
 642 771 0
 643 ENTRY STK00 FOR THE WAND
 644 STK00
 645 772 1673 GOTO ABTXF3 (761) MUST BE SHORT GOTO!!

646	773	460 LDI		
647	774	40 CON	32	BLANK
648	775	1650 SRSABC		
649	776	1650 SRSABC		
650	777	1650 SRSABC		

* 652 ENTRY STK04
 * ENTRY POINT FOR WAND (3-16-79)
 *

655	STK04	1 GOSUB	GTACOD	GET ALPHACODE[KEYCODE]
656	1000	0		
656	1001	1334 PT=	13	
657	1002	420 LC	4	SET FOR LASTX
658	1003	416 A=C		REG INDEX IN A.S. CHAR IN A.X
659	1004	460 LDI		
660	1005	114 CON	0114	"L"
661	1006	1546 ? A#C	X	
662	1007	317 GOC	STK20 (1041)	
663	1010	1 GOSUB	MASK	PUT CHAR OUT TO LCD
664	1011	0		
664	1012	1 GOSUB	LEFTJ	
665	1013	0		
665	1014	1 GOSUB	ENCF00	
666	1015	0		
666	1016	1634 PT=	0	GET PTEMP2
667	1017	230 C=G		
668	1020	1530 ST=C		
669	1021	1270 C=REGN	10	
670	1022	1034 PT=	2	
671	1023	720 LC	7	
672	1024	514 ?S6=1		INDIRECT?
673	1025	33 GONC	STK10 (1031)	
674	1026	1034 PT=	2	
675	1027	1720 LC	15	
676	1030			
677	STK10	256 AC EX		
678	1031	1574 RCR	12	
679	1032	242 AC EX	PT	
680	1033	1 GOLONG	NLT020	
681	1034	2		
681	1035			

* 683 1036 STK15 1 GOSUB BLINK
 683 1037 0
 684 1040 1233 GOTO STK03 (763)

* 686 1041 STK20 460 LDI "W"
 687 1042 127 CON 87
 688 1043 STK30 676 A=A-1 S
 689 1044 1536 ? A#0 S
 690 1045 53 GONC STK40 (1052)
 691 1046 1046 C=C+1 X
 692 1047 1546 ? A#C X
 693 1050 1737 GOC STK30 (1043)
 694 1051 1403 GOTO STK05 (1011)

* 696 1052 STK40 460 LDI "T"
 697 1053 124 CON 0124
 698 1054 1546 ? A#C X
 699 1055 1343 GONC STK05 (1011)

700 1056
701

1603 GOTO STR15 (1036)
EJECT

* FDIGIT - FINAL DIGIT

* ENTRY CONDITIONS: A.S=SECOND TO LAST DIGIT, HEX, P SEL,

* LCD CHIP ON, STATUS SET PTEMP1 UP

SHIFT PROMPTS OFF RIGHT END OF LCD

```

705      FDIGIT
706 1057      116 C=0
707 1060      276 AC EX S
708 1061      1374 RCR 13
709 1062      1434 PT= 1
710 1063      320 LC 3
711 1064      1750 SLSABC
712      FDIG10
713 1065      1 GOSUB NEXT1
713 1066      0
714 1067      143 GOTO FDIG30 (1103) BACKARROW RETURN (SHORT GOTO!!!)
715 1070      14 ?S3=1 DIGIT?
716 1071      47 GOC FDIG20 (1075) YES
717 1072      1 GOSUB BLINK
717 1073      0
718 1074      1713 GOTO FDIG10 (1065)
719      FDIG20
720 1075      116 C=0
721 1076      276 AC EX S
722 1077      1374 RCR 13
723 1100      320 LC 3
724 1101      1750 SLSABC
725 1102      433 GOTO NULT# (1145)

```

SEND DIGIT TO DISPLAY

```

* 727      FDIG30
728 1103      1670 RABCR
729 1104      1740 RTN

```

SHIFT DIGIT OFF

*
*

```

732 1105 NEXT1 460 LDI
733 1106      37 CON 31
734 1107      103 GOTO NXT1E (1117)
735 1110 NEXT2 460 LDI
736 1111      37 CON 31
737 1112      43 GOTO NXT2E (1116)
738 1113 NEXT3 460 LDI
739 1114      37 CON 31
740 1115      1750 SLSABC
741 1116 NXT2E 1750 SLSABC
742 1117 NXT1E 1750 SLSABC
743 1120 NEXT 1 GOSUB LEFTJ
743 1121      0
744 1122      1 GOSUB ENCP00
744 1123      0
745 1124      1770 C=REGN 15
746 1125      34 PT= 3
747 1126      230 C=G
748 1127      1750 REGN=C 15
749 1130      1670 C=REGN 14
750 1131      1474 RCR 1
751 1132      1530 ST=C
752 1133      210 S5= 1
753 1134      1410 S1= 1
754 1135      1630 C=ST
755 1136      1374 RCR 13
756 1137      1 GOSUB ANN+14

```

SAVE PTEMP2 IN
REG 15[4:3]

SET PKSEQ
SET MSGFLG

```

756 1140      0
757 1141      1 GOSUB RSTKB
757 1142      0
758 1143      1 GOLONG WKUP10
758 1144      2

```

* NULT# - NULL TEST FOLLOWING NUMERIC OPERAND

* ENTRY CONDITIONS: P SEL, LCD ON

```

762      NULT#
763 1145      1334 PT= 13      INITIALIZE # OF DIGITS COUNTER
764 1146      1720 LC 15
765 1147      1434 PT= 1
766 1150      416 A=C      # OF DIGITS COUNTER IN A.S
767 1151 NULT#1 1670 RABCR
768 1152      576 A=A+1 S
769 1153      1530 ST=C
770 1154      114 ?S4=1
771 1155      1747 GOC NULT#1 (1151) SHIFT UNTIL " " OR " ."
772 1156      6 A=0 X      INITIALIZE SUM
773 1157 NULT#2 1770 RABCL
774 1160      102 C=0 PT
775 1161      126 C=0 XS
776 1162      506 A=A+C X
777 1163      676 A=A-1 S      DECREMENT # OF DIGITS
778 1164      107 GOC NULT#3 (1174)
779 1165      246 AC EX X      MULTIPLY BY 10
780 1166      746 C=C+C X
781 1167      406 A=C X
782 1170      746 C=C+C X
783 1171      746 C=C+C X
784 1172      506 A=A+C X
785      LEGAL
786 1173      1643 GOTO NULT#2 (1157)

```

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

```

*
788      NULT#3
789 1174      1634 PT= 0
790 1175      230 C=G
791 1176      1530 ST=C      PUT UP PTEMP2
792 1177      514 ?S6=1      INDIRECT?
793 1200      43 GONC NULT#4 (1204) NO
794 1201      460 LDI
795 1202      200 CON 128
796 1203      506 A=A+C X      SET INDIRECT BIT
797 1204 NULT#4 206 B=A X      SAVE ARG IN B.X
798 1205      1 GOSUB LEFTJ
798 1206      0
799 1207      1 GOSUB ENCF00
799 1210      0
800 1211      1270 C=REGN 10      GET FC
801 1212      416 A=C      FC TO A[4...]
802 1213      214 ?S5=1      XROM?
803 1214      247 GOC NLT020 (1240) YES, ARG IN B.X ONLY
804 1215      146 A=B X      COPY ARG TO A.X
804 1216      206
805 1217 NULT#5 1746 A SL X      & COZY UP TO FC
806 1220      203 GOTO NLT020 (1240) FC, ARG IN A[4:1]
807      EJECT

```

```

808      NLT000
809 1221      1 GOSUB LEFTJ
809 1222      0
810 1223      1 GOSUB ENCF00
810 1224      0
811 1225      1270 C=REGH 10
812 1226      416 A=C
813 1227      460 LDI
814 1230      310 CON 200
815 1231      1340 DISOFF
816 1232 NLT010 1710 RST KB
817 1233      1714 CHK KB
818 1234      73 GONC NLT030 (1243)
819 1235      1146 C=C-1 X
820 1236      1743 GONC NLT010 (1232)
821 1237      1440 DISTOG
822      NLT020
823
824 1240      1 GOSUB NULTST
824 1241      0
825 1242      103 GOTO NLT040 (1252)
826 1243 NLT030 1 GOSUB RST05
826 1244      0
827 1245      1 GOSUB CLLCDE
827 1246      0
828 1247      1440 DISTOG
829 1250      1 GOSUB ENCF00
829 1251      0
830      ENTRY NLT040
831      NLT040
832 1252      1625 CON 01625
833 1253      674 CON 0674
834 1254      1 GOSUB RSTSEQ
834 1255      0
835
836
837
838 1256      1634 PT= 0
839 1257      230 C=G
840 1260      1730 CST EX
841 1261      114 ?S4=1
842 1262      73 GONC NLT050 (1271)
843 1263      1514 ?S12=1
844 1264      437 GOC AB10XF (1327)
845 1265      256 AC EX
846 1266      274 RCR 5
847 1267      1 GOLONG INSLIN
847 1270      2

*
849 1271 NLT050 1730 CST EX
850 1272      116 C=0
851 1273      134 PT= 4
852 1274      1720 LC 15
853 1275      560 STK=C
854 1276      256 AC EX
855 1277      274 RCR 5
856 1300      1376 ? C#0 S
857 1301      1 GOLC XGUTE
857 1302      3

```

SAVE FC IN A

FOR ENTRY HERE,
FC, ARG IN A[4:1]

DEBOUNCE KEY UP

KEY IS UP. GO EXECUTE FCN
FIRST GIVE PRINTER A CHANCE
GOSUB PRT5
CLEAR SHIFTSET, PKSEQ.

MSGFLAG, DATAENTRY,
CATALOGFLAG, & PAUSING
LEAVES SS0 UP

GET PTENP2
INSERT?
NO
PRIVATE?
YES

BRING BACK SS 0

PUT NFRPU (0360)
ON THE SUBROUTINE STACK

```

858 1303      146 AB EX X
859 1304      1 GOLONG XCUTB1
859 1305      2

```

RETRIEVE 3 DIGIT ARGUMENT
FROM B.X TO A.X

11

*
* NULTST - NULL TEST
*

```

863      NULTST      NULL TEST
864 1306      460 LDI
865 1307      1100 CON      576      INITIALIZE NULL TIMER
866 1310      746 C=C+C X
867 1311 NULT10 1710 RST KB
868 1312      1714 CHK KB
869 1313      1 GOLNC RST05      KEY UP YET?
869 1314      2      GO DEBOUNCE
870 1315      1146 C=C-1 X      NO. DECREMENT COUNTER
871 1316      1733 GONC NULT10 (1311)
872 1317      404 S8= 0      DON'T PRINT MESSAGE
873 1320      1 GOSUB MSGA      KEY DOWN TOO LONG
873 1321      0
874 1322      0 XDEF MSGNL
875 1323      460 LDI
876 1324      1750 CON      1000
877 1325 NULT20 1146 C=C-1 X      310 MILLISEC DELAY
878 1326      1773 GONC NULT20 (1325) SO "NULL" CAN BE SEEN
879 1327 AB10XF 1 GOLONG ABTS10
879 1330      2
880      ENTRY NAME20
881      ENTRY NAMEA
882      ENTRY NAME21
883      ENTRY NAM40
884      ENTRY NAM440
885      ENTRY NM4405
886      ENTRY NAME4A
887      ENTRY NAME4D

```

*
* PARSE LOGIC FOR ALPHA OPERANDS STARTS HERE
*

```

891      NAMEA      ON ENTRY, SS IS SCRATCH
892 1331      1 GOSUB ENCF00
892 1332      0
893 1333      1645 CON      01645      GOSUB PRT3
894 1334      674 CON      0674
895      ENTRY PR3RT      FOR PRINTER
896      PR3RT
897 1335      1670 C=REGN 14
898 1336      1730 CST EX
899 1337      1210 S7= 1      SET ALPHAMODE
900 1340      1730 CST EX
901 1341      1650 REGN=C 14
902 1342      116 C=0      INITIALIZE ALPHA OPERAND
903 1343      1150 REGN=C 9
904 1344 NAME10 1 GOSUB ENLCD
904 1345      0
905 1346 NAME20 1 GOSUB NEXT1
905 1347      0
906 1350      163 GOTO NAME30 (1366) BACKARROW
907      NAME21      ON ENTRY HERE, PT=1,
908      LCD CHIP ON, SS PTEMP1 UP
909 1351      514 ?S6=1      SHIFT KEY?
910 1352      1 GOLNC NAM40

```

```

910 1353      2
911 1354      1 GOSUB TOGSHF      TOGGLE SHIFT KEY
911 1355      0
912 1356      1663 GOTO NAME10 (1344)

*
914          ENTRY NAME33      USED BY CARD RDR LOGIC
915          TO ABORT A PARTIAL
916          KEY SEQUENCE
917          MAY ALSO BE USED BY
918          PRINTER LOGIC
919 1357 NAME33      1 GOSUB LDSST0
919 1360      0
920 1361      1204 S7= 0      CLEAR ALPHAMODE
921 1362      1630 C=ST
922 1363      1650 REGN=C 14
923 1364      1 GOLONG ABTSEQ
923 1365      2

*
925          NAME30
926 1366      1 GOSUB ENCP00      ON ENTRY, PT=1, LCD CHIP ON
926 1367      0      BKARROW HIT
927 1370      1170 C=REGN 9
928 1371      1356 ? C#0      ANY CHARS TO DELETE?
929 1372      1653 GONC NAME33 (1357) NO
930 1373      1574 RCR 12      YES, DELETE ONE CHAR
931 1374      112 C=0 WPT
932 1375      1150 REGN=C 9
933 1376      1 GOSUB OFSRFT
933 1377      0
934 1400      1 GOSUB ENLCD
934 1401      0
935 1402      1670 RABCR
936 1403 NAME31 1433 GOTO NAME20 (1346)      SHIFT OFF ONE CHARACTER

*
938 1404 NAME34 1414 7S1=1      OP2 BIT 1?
939          (IS EMPTY OPERAND AN ERROR?)
940 1405      713 GONC NAME42 (1476) NO

*
942 1406 NAME35      1 GOSUB BLINK
942 1407      0
943 1410      1343 GOTO NAME10 (1344)

*
945          ENTRY NAME37
* ENTRY POINT ADD FOR WAND ON 3-13-79

*
948 1411 NAME37      1 GOSUB GTACOD
948 1412      0
949 1413      406 A=C X      COPY CHARACTER TO A.X
950 1414      1 GOSUB OFSHFT
950 1415      0
951 1416      666 A=A-1 XS      IS IT A CHARACTER?
952 1417      1673 GONC NAME35 (1406) NO
953 1420      1434 PT= 1
954 1421      460 LDI
955 1422      177 CON 127      LAZY T
956 1423      1552 ? A#C WPT
957 1424      1623 GONC NAME35 (1406)
958 1425      460 LDI
959 1426      72 CON 58      COLON
960 1427      1552 ? A#C WPT

```

961 1430	1563 GONC	NAME35 (1406)	
962 1431	460 LDI		
963 1432	56 CON	46	D.P.
964 1433	1552 ? A#C	WPT	
965 1434	1523 GONC	NAME35 (1406)	
966 1435	460 LDI		
967 1436	54 CON	44	COMMA
968 1437	1552 ? A#C	WPT	
969 1440	1463 GONC	NAME35 (1406)	
970 1441	1170 C=REGN	9	
971 1442	1352 ? C#0	WPT	FULL ALREADY?
972 1443	1437 GOC	NAME35 (1406)	FULL
973 1444	252 AC EX	WPT	ADD CHARACTER TO REG 9
974 1445	412 A=C	WPT	RESTORE CHARACTER TO A.X
975 1446	1074 RCR	2	-
976 1447	1150 REGN=C	9	-
977			ADD CHAR TO DISPLAY
978 1450	356 BC EX		SAVE OPERAND IN B
979 1451	1 GOSUB	ENLCD	
979 1452	0		
980 1453	1 GOSUB	MASK	TRANSLITERATE CHAR AND
980 1454	0		
981			SEND TO DISPLAY
982			NOTE MASK DECREMENTS B.S
983 1455	1263 GOTO	NAME31 (1403)	
984	FILLTO	01463	PRESERVE ENTRY TABLE
1456	0000 NOP		
1457	0000 NOP		
1460	0000 NOP		
1461	0000 NOP		
1462	0000 NOP		
1463	0000 NOP		
*			
986 1464 NAM40	1 GOSUB	ENCP00	
986 1465	0		
987 1466	214 ?S5=1		ALPHA KEY?
988 1467	1223 GONC	NAME37 (1411)	
989 1470	1434 PT=	1	
990 1471	1170 C=REGN	9	
991 1472	1356 ? C#0		ANY CHARS IN OPERAND?
992 1473	1113 GONC	NAME34 (1404)	NO
993 1474	1 GOSUB	RTJLBL	RIGHT JUSTIFY OPERAND
993 1475	0		
994 1476 NAME42	1150 REGN=C	9	PUT BACK RIGHT-JUSTIFIED
995			OPERAND
996 1477	1670 C=REGN	14	PUT UP SS 0
997 1500	1530 ST=C		
998 1501	1204 S7=	0	CLEAR ALPHAMODE
999 1502	1630 C=ST		
1000 1503	1 GOSUB	ANN+14	STORE STATUS SETS AND
1000 1504	0		
1001			UPDATE ALPHA ANNUNCIATOR
1002 1505	1270 C=REGN	10	
1003 1506	74 RCR	3	
1004 1507	406 A=C	X	FC TO A.X
1005 1510	460 LDI		
1006 1511	17 CON2	0	15 FC FOR ASN
1007 1512	1552 ? A#C	WPT	FC#ASN?
1008 1513	1 GOLNC	KEYOP	THIS IS ASN
1008 1514	2		

```

1009 1515      206 B=A      X      SAVE FC IN B.X
1010 1516      1 GOSUB ENLCD      THIS IS NOT ASN
1010 1517      0
1011 1520      1 GOSUB LEFTJ
1011 1521      0
1012 1522      1 GOSUB ENCF00
1012 1523      0
1013 1524      1170 C=REGN 9
1014 1525      340 SEL Q
1015 1526      1334 PT= 13
1016 1527      240 SEL P
1017 1530      1034 PT= 2
1018 1531      1362 ? C#0 PQ      MORE THAN 1 CHAR IN LABEL?
1019 1532      1 GSUBNC ALCL00      NO. TEST FOR LOCAL ALPHA LBL
1019 1533      0
1020 1534      146 AB EX X      RETRIEVE FC FROM B
1021 1535      460 LDI
1022 1536      36 CON2 1 14      FC FOR AXEQ
1023 1537      1434 PT= 1
1024 1540      1552 ? A#C WPT      FC#AXEQ?
1025 1541      717 GOC NAME46 (1632) NOT AXEQ
1026 1542      1170 C=REGN 9
1027 1543      530 M=C
1028 1544      1 GOSUB ASRCH
1028 1545      0
1029 1546      1356 ? C#0      FOUND?
1030 1547      217 GOC NAME44 (1570) YES
1031 1550      1670 C=REGN 14      RESTORE SS 0
1032 1551      1530 ST=C
1033 1552      1770 C=REGN 15      RESTORE PTEMP2 TO G
1034 1553      74 RCR 3
1035 1554      1634 PT= 0
1036 1555      130 G=C
1037 1556      14 ?S3=1      PROGRAM MODE?
1038 1557      537 GOC NAME46 (1632) YES
1039 1560      1270 C=REGN 10      RESTORE FC TO A
1040 1561      416 A=C      FOR PRT5
1041 1562      1625 CON 01625      GOSUB PRT5
1042 1563      674 CON 0674
1043 1564      1 GOSUB RSTSEQ
1043 1565      0
1044 1566      1 GOLONG LPRNE
1044 1567      2

*
1046 1570 NAME44 1114 ?S9=1      MICROCODE FCN?
1047 1571      467 GOC NAME48 (1637) YES
* USER PROGRAM. PC IN C[3:0].
* IF IN ROM THEN S2=1 AND XROM IN C[7:4]
1050 1572      530 M=C      SAVE PC IN M
1051 1573      174 RCR 4
1052 1574      160 N=C      PUT XROM TO N[3:0]
1053      NAM440
* ENTRY CONDITIONS FOR NAM440:
* S2=1 FOR ROM, S2=0 FOR RAM
* PC IN N[3:0]
* IF ROM, THEN XROM IN N[3:0]
* IF RAM, THEN AXEQ ALREADY IN PLACE IN REG 10
1059 1575      1110 S9= 1      SAY ADDRESS ALREADY KNOWN
1060      NM4405
* INSTRUCTIONS BELOW TO CLEAR AND SET SS MAY NOT BE NECESSARY

```


* BECAUSE NLT020 DOESN'T LOOK AT S5.

1063	1576	204	S5=	0	CLEAR ROM BIT FOR NLT020
1064	1577	1014	?S2=1		ROM?
1065	1600	53	GONC	NAM44A (1605)	NO
1066	1601	210	S5=	1	SET ROM BIT FOR NLT020
1067	1602	260	C=N		GET XROM TO C[3:0]
1068	1603	1	GOSUB	STORFC	
1068	1604	0			
1069	1605	NAM44A	104	S4=	0
1070	1606		1670	C=REGN	14
1071	1607		1730	CST EX	
1072	1610		14	?S3=1	
1073	1611		43	GONC	NAM44B (1615)
1074	1612		1530	ST=C	
1075	1613		110	S4=	1
1076	1614		1630	C=ST	
1077	1615	NAM44B	1530	ST=C	
1078	1616		1634	PT=	0
1079	1617		130	G=C	
1080	1620		1114	?S9=1	
1081	1621		113	GONC	NAME46 (1632)
1082	1622		1	GOSUB	DSPLN+
1082	1623		0		
1083					IF S4 THEN INC & DSP LINE#
1084	1624		1	GOSUB	ENCP00
1084	1625		0		
1085	1626		630	C=M	
1086	1627		416	A=C	
1087	1630		1	GOSUB	TXTLBL
1087	1631		0		
1088	1632	NAME46	1270	C=REGN	10
1089	1633		416	A=C	
1090	1634		206	B=A	X
1091	1635		1	GOLONG	NLT020
1091	1636		2		

*

*

1094		NAME48			MICROCODE FCN
1095	1637		214	?S5=1	MAINFRAME?
1096	1640		123	GONC	NAME4C (1652)
1097					NO
1098	1641		174	ROR	4
1099	1642		126	C=0	XS
1100	1643		416	A=C	
1101	1644	NAME4A	1670	C=REGN	14
1102	1645		1530	ST=C	
1103	1646		256	AC EX	
1104	1647		1104	S9=	0
1105					PUT UP SS 0
1106	1650		1	GOLONG	PARS56
1106	1651		2		

*

1108 NAME4C
* WE COME TO NAME4C FROM ASRCH IN THE AXEQ LOGIC

* XADR IS IN C[3:0] AND XROM IS IN C[7:4]

1111	1652		530	M=C	SAVE XADR IN M[3:0]
1112	1653		174	ROR	4
1113		NAME4D			MOVE XROM TO C[3:0]

* REPARSE LOGIC FOR MICROCODED XROM FUNCTIONS

* ON ENTRY, XADR IS IN M[3:0] AND XROM IS IN C[3:0]

```

1116 1654          1 GOSUB STORFC          PUT XROM TO REG 10
1116 1655          0
1117 1656          1670 C=REGN 14          GET SS 0
1118 1657          1530 ST=C
1119 1660          1434 PT= 1
1120 1661          630 C=M                GET XADR
1121 1662          674 RCR 11             PUT XADR TO C.M
1122 1663          530 M=C                PUT XADR TO M[6:3]
1123 1664          14 ?S3=1              PROGRAM MODE?
1124 1665          133 GONC NAME4F (1700) NO
1125 1666          1460 CXISA
1126 1667          1346 ? C#0 X          PROGRAMMABLE?
1127 1670          33 GONC NAME4E (1673) NO
1128 1671          320 LC 3              SET XROM BIT(5)
1129                                     & INSERT BIT(4)
1130 1672          73 GOTO NAME4G (1701)
* FOR MICROCODE FCNS IN PLUG-IN ROMS, IF C(XADR)=0 THEN WE LOOK
* AT C(XADR+1) TO DETERMINE WHETHER THE FCN SHOULD BE EXECUTED ON
* KEY DOWN. IF C(XADR+1)=0 THEN THE FCN IS XKD ELSE THE FCN IS
* A NORMAL NON-PROGRAMMABLE FUNCTION.
1135 1673 NAME4E 1072 C=C+1 M
1136 1674          1460 CXISA
1137 1675          1346 ? C#0 X          IS C(XADR+1) NON-ZERO?
1138 1676          27 GOC NAME4F (1700)   XKD FCN - GO DO IT
1139 1677          740 GOTOC
*
1141 1700 NAME4F 220 LC 2                SET XROM BIT(5) ONLY
1142 1701 NAME4G 20 LC 0
1143 1702          1530 ST=C              INITIALIZE PTEMP2
1144 1703          1634 PT= 0
1145 1704          130 G=C                & SAVE IN G
1146 1705          1 GOLONG PARS75
1146 1706          2
1147
*
* DSPLN+ - DISPLAY (LINE#+1)
* ON ENTRY, LINE NUMBER MUST BE VALID IN REG 15, AND CHIP 0 MUST
* BE ENABLED.
* 1. GETS LINE NUMBER FROM REG 15
* 2. CLEARS LCD
* IF S4 IS CLEAR, THEN RETURNS IMMEDIATELY
* 3. INCREMENTS LINE NUMBER (BUT DOESN'T STORE BACK TO REG 15)
* 4. IF PRIVATE, REPLACES LINE NUMBER WITH 0
* 5. CALLS GENNUM TO PUT LINE NUMBER TO LCD
* 6. SHIFTS ON A BLANK FOLLOWING THE LINE NUMBER
* ON EXIT, THE DISPLAY CHIP IS ENABLED AND THE PT=0
* USES A, B.X, B.S, C, & ONE SUBROUTINE LEVEL
*
1162 1707 DSPLN+ 1770 C=REGN 15          GET LINE NUMBER
1163 1710          346 BC EX X
1164 1711          1 GOSUB CLLCDE
1164 1712          0
1165 1713          114 ?S4=1
1166 1714          1640 RTN NO
1167 1715          146 AB EX X          BRING LINE # TO A.X
1168 1716          546 A=A+1 X          INCREMENT IT
1169 1717          1514 ?S12=1          PRIVATE?
1170 1720          23 GONC DSPL10 (1722) NO
1171 1721          6 A=0 X              YES - ZERO OUT LINE#
1172 1722 DSPL10 36 A=0 S              SET UP FOR GENNUM

```

```

1173 1723          1 GOSUB  GENNUM
1173 1724          0
1174 1725          460 LDI
1175 1726          40 CON      32
1176 1727          1750 SLSABC      SHIFT IN A BLANK
1177 1730          1740 RTN

*
* GOLONG - LONG BRANCH ROUTINE FOR PLUG IN ROMS
* SAME AS GOSUB EXCEPT USES 1 SUBROUTINE LEVEL TEMPORARILY.
*
* GOSUB - SUBROUTINE ROUTINE FOR PORT ADDRESSED PLUG IN ROMS
* THIS SUBROUTINE ALLOWS SUBROUTINE CALLS IN PORT ADDRESSED
*- PLUG IN ROMS.
* THE CALLING SEQUENCE IS:
*   GOSUB  GOSUB      MUST BE IN HEX MODE ON ENTRY!!
*   DEF    <NAME>
* WHERE NAME IS IN THE SAME 1024 WORD ROM AS THE CALLING ROUTINE.
*
* WARNING!!! - CALLING A SUBROUTINE IN ANOTHER 1024 ROM FROM THE
* CURRENT ONE WILL NOT WORK. USE GOSUB[0-3].
*
* USES ONLY C, NO ADDITIONAL SUBROUTINE LEVELS
*
* GOLNGH - SAME AS GOLONG EXCEPT SETS HEX MODE ON ENTRY.
* GOSUBH - SAME AS GOSUB EXCEPT SETS HEX MODE ON ENTRY.
*
1198          ENTRY  GOLNGH
1199          ENTRY  GOLONG
1200          ENTRY  GOSUBH
1201          ENTRY  GOSUB
1202 1731 GOLNGH 1140 SETHEX
1203 1732 GOLONG 660 C=STK      GET ADDRESS OF CALLING ROUTINE
1204 1733          1460 CXISA      GET DESTINATION ADDRESS
1205 1734          63 GOTO  GOSUBA (1742) GO CREATE THE CORRECT 16 BIT ADDRESS
1206
1207 1735 GOSUBH 1140 SETHEX
1208 1736 GOSUB  660 C=STK      GET ADDRESS OF CALLING ROUTINE
1209 1737          1460 CXISA      GET THE DESTINATION ADDRESS
1210 1740          1072 C=C+1  M  ADVANCE ADDRESS BEYOND ARGUMENT FOR
1211 1741          560 STK=C    PUT RETURN ADDRESS BACK
1212 1742 GOSUBA 756 C=C+C      MOVE OVER BOTH ADDRESSES TWO BITS
1213 1743          756 C=C+C    SO THAT THE DESIRED 10 BIT BOUNDARY
1214 1744          1732 C SR   M  FALLS ON A DIGIT BOUNDARY C[2-3]
1215 1745          1732 C SR   M  COMBINE 10 BITS FROM ARGUMENT
1216 1746          1732 C SR   M  WITH 6 BITS FROM SUBROUTINE STACK
1217 1747          756 C=C+C    TO FORM A 16 BIT ADDRESS
1218 1750          756 C=C+C    AND POSITION PROPERLY FOR GOTOC
1219 1751          1574 RCR     12
1220 1752          740 GOTOC      GO TO THE DESIRED ADDRESS.

*
1222
1223          ENTRY  GT3DBT
1224 1753 GT3DBT  1 GSBLNG GETPC      STATUS_3RD BYTE
1224 1754          0
1225 1755          530 M=C          -
1226 1756          1 GSBLNG INCAD2   -
1226 1757          0
1227 1760          1 GSBLNG GTBYTA   -
1227 1761          0
1228 1762          1730 CST EX      -

```

1229 1763 1740 RTN

1230

1231

1232

ENTRY XSIGN

* THE SIGN FUNCTION RETURNS ONE FOR POSITIVE
 * NUMBERS AND -1 FOR NEGATIVE NUMBERS AND ZERO FOR
 * ALPHA DATA.

1238

1239 1764 XSIGN 1534 PT= 12

1240 1765 370 C=REGN 3

1241 1766 112 C=0 WPT

1242 1767 416 A=C

1243 1770 676 A=A-1 S

1244 1771 676 A=A-1 S

1245 1772 27 GOC DONSGN (1774) MAKES USE OF OVFL10

1246 AT NFRX TO ZERO OUT

1247 WHOLE WORD BECAUSE

1248 MANTISSA IS ZERO

1249 1773 120 LC 1

1250 1774 DONSGN 1 GOLONG NFRX

1250 1775 2

1251

*
 * MUST HAVE AT LEAST 2 WORDS AT THE END OF CN3 FOR CHECKSUM AND
 * TRAILER.

1255

1256

1257

*

1259

1260

1261

1262 1776 REVLEV 7 CON 7 FILLTO 01775

1263 1777 CKSUM0 0 CON 00000 REV LEVEL= G

1264 END

ERRORS 1 0

SYMBOL TABLE

AB10XF	1327	-	1264		
ABTS10	426	-			
ABTSEQ	422	-	467	442	355
ABTXF3	761	-	772	704	
AJ2	724	-	706		
AJ210	726	-	723		
AJ220	731	-	727		
AJ3	720	-			
CKSUM0	1777	-			
CLRSB2	0	-			
CLRSB3	2	-			
CLRSB8	32	-	4		
DOHSGH	1774	-	1772		
DSPL10	1722	-	1720		
DSPLN+	1707	-			
FDIG10	1065	-	1074		
FDIG20	1075	-	1071		
FDIG30	1103	-	1067		
FDIGIT	1057	-			
GOLNGH	1731	-			
GOLONG	1732	-			
GOSUB	1736	-			
GOSUBA	1742	-	1734		
GOSUBH	1735	-			
GT3DBT	1753	-			
IND	662	-			
IND20	702	-	717		
IND21	704	-			
IND30	715	-	712		
INDGTO	672	-	620		
INDEXEQ	664	-	621		
MID10	745	-	756		
MID15	754	-	751		
MID20	757	-	747		
MIDDIG	740	-			
NAM40	1464	-			
NAM440	1575	-			
NAM44A	1605	-	1600		
NAM44B	1615	-	1611		
NAME10	1344	-	1410	1356	
NAME20	1346	-	1403		
NAME21	1351	-			
NAME30	1366	-	1350		
NAME31	1403	-	1455		
NAME33	1357	-	1372		
NAME34	1404	-	1473		
NAME35	1406	-	1443	1440 1434 1430 1424 1417	
NAME37	1411	-	1467		
NAME42	1476	-	1405		
NAME44	1570	-	1547		
NAME46	1632	-	1621	1557 1541	
NAME48	1637	-	1571		
NAME4A	1644	-			
NAME4C	1652	-	1640		
NAME4D	1654	-			
NAME4E	1673	-	1670		

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

NAME4F	1700	-	1676	1665			
NAME4G	1701	-	1672				
NAMEA	1331	-					
NEWFCN	126	-	46				
NEXT	1120	-					
NEXT1	1105	-					
NEXT2	1110	-					
NEXT3	1113	-					
NLT000	1221	-					
NLT010	1232	-	1236				
NLT020	1240	-	1220	1214			
NLT030	1243	-	1234				
NLT040	1252	-	1242				
NLT050	1271	-	1262				
NM4405	1576	-					
NULT#	1145	-	1102				
NULT#1	1151	-	1155				
NULT#2	1157	-	1173				
NULT#3	1174	-	1164				
NULT#4	1204	-	1200				
NULT#5	1217	-					
NULT10	1311	-	1316				
NULT20	1325	-	1326				
NULTST	1306	-					
NXT1E	1117	-	1107				
NXT2E	1116	-	1112				
PAR001	25	-	27				
PAR003	35	-	31	26	24	22	
PAR005	44	-	40				
PAR110	353	-	367				
PAR111	355	-					
PAR112	365	-	406	404	377		
PAR115	370	-	362				
PAR130	400	-					
PARA05	440	-	460	435			
PARA06	442	-					
PARA10	456	-	464				
PARA45	461	-	444				
PARA50	465	-	437				
PARA60	465	-	526	501			
PARA61	467	-					
PARA65	477	-	534	530			
PARA70	502	-	474				
PARA75	511	-					
PARA80	514	-	524				
PARA90	522	-					
PARB10	566	-	630	625			
PARB15	574	-	567				
PARB20	607	-	563				
PARB30	622	-	610				
PARB40	631	-					
PARS05	64	-					
PARS10	104	-	106				
PARS20	110	-	116				
PARS30	117	-	111				
PARS50	136	-	132				
PARS52	207	-	141				
PARS55	213	-	135				
PARS56	223	-					
PARS57	240	-	235	233			

PARS60	264	-	204		
PARS70	314	-	260		
PARS75	315	-			
PARSDE	220	-			
PARSE	5	-			
PARSEA	434	-	352		
PARSEB	555	-			
PR3RT	1335	-			
PRA100	525	-	516		
PRA110	527	-	510		
PRA115	546	-	532		
RAK05	176	-	166	162	
RAK06	177	-			
RAK10	205	-	175	173	152
REVLEV	1776	-			
STK	763	-	710		
STK00	772	-			
STK03	763	-	1040		
STK04	1000	-			
STK05	1011	-	1055	1051	
STK10	1031	-	1026		
STK15	1036	-	1056		
STK20	1041	-	1010		
STK30	1043	-	1050		
STK40	1052	-	1045		
XFRNMA	605	-	545		
XSIGN	1764	-			

ENTRY TABLE

ABTS10	426	-
ABTSEQ	422	-
AJ2	724	-
AJ3	720	-
CLRSB2	0	-
CLRSB3	2	-
DSPLN+	1707	-
FDIG20	1075	-
FDIGIT	1057	-
GOLNGH	1731	-
GOLONG	1732	-
GOSUB	1736	-
GOSUBH	1735	-
GT3DBT	1753	-
IND	662	-
IND21	704	-
MIDDIG	740	-
NAM40	1464	-
NAM440	1575	-
NAME20	1346	-
NAME21	1351	-
NAME33	1357	-
NAME37	1411	-
NAME4A	1644	-
NAME4D	1654	-
NAMEA	1331	-
NEXT	1120	-
NEXT1	1105	-
NEXT2	1110	-
NEXT3	1113	-
NLT000	1221	-
NLT020	1240	-
NLT040	1252	-
NM4405	1576	-
NULT#	1145	-
NULT#3	1174	-
NULT#5	1217	-
NULTST	1306	-
PAR111	355	-
PAR112	365	-
PARA06	442	-
PARA60	465	-
PARA61	467	-
PARA75	511	-
PARB40	631	-
PARS05	64	-
PARS56	223	-
PARS75	315	-
PARSDE	220	-
PARSE	5	-
PARSEB	555	-
PR3RT	1335	-
RAK06	177	-
STK	763	-
STK00	772	-
STK04	1000	-

XSIGN 1764 -

124

MASK	1011	1453		
MASK	1012	1454		
MESSL	674	763		
MESSL	675	764		
MIDDIG	475	520		
MIDDIG	476	521		
MSGA	1320			
MSGA	1321			
MSGNL	1322			
NAM40	1352			
NAM40	1353			
NAMEA	462	605		
NAMEA	463	606		
NEXT1	440	770	1065	1346
NEXT1	441	771	1066	1347
NEXT2	353	702	745	
NEXT2	354	703	746	
NEXT3	465	514		
NEXT3	466	515		
NFRKB	432			
NFRKB	433			
NFRX	1774			
NFRX	1775			
NLT000	330			
NLT000	331			
NLT020	1034	1635		
NLT020	1035	1636		
NULT#	736			
NULT#	737			
NULT#3	553			
NULT#3	554			
NULT#5	312			
NULT#5	313			
NULTST	1240			
NULTST	1241			
OFSHFT	267	315	1376	1414
OFSHFT	270	316	1377	1415
PAR112	572			
PAR112	573			
PARA60	660			
PARA60	661			
PARS56	420	1650		
PARS56	421	1651		
PARS75	1705			
PARS75	1706			
PARSEB	371			
PARSEB	372			
PROMF1	275	652		
PROMF1	276	653		
PROMF2	322			
PROMF2	323			
PUTPCX	32			
PUTPCX	33			
RAK60	145			
RAK60	146			
ROW940	300			
ROW940	301			
RST05	1243	1313		
RST05	1244	1314		
RSTKB	1141			

```

RSTKB 1142
RSTSEQ 430 1254 1564
RSTSEQ 431 1255 1565
RTJLBL 1474
RTJLBL 1475
SEARCH 201
SEARCH 202
STK 401
STK 402
STORFC 1603 1654
STORFC 1604 1655
TBITMP 142
TBITMP 143
TOGSHF 1354
TOGSHF 1355
TXTLBL 1630
TXTLBL 1631
WKUP10 1143
WKUP10 1144
XCUTB1 1304
XCUTB1 1305
XCUTE 1301
XCUTE 1302

```

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES @10000-11777
 * CONTENTS:
 * 1. EXECUTION POINTS FOR MAINFRAME FUNCTIONS (MUST BE IN
 * @10000-11736)
 *

7	FILE	CN4B
8	ENTRY	CAT##3
9	ENTRY	+
10	ENTRY	-
11	ENTRY	-DEC
12	ENTRY	-OCT
13	ENTRY	(<*>)
14	ENTRY	ADVANCE
15	ENTRY	/
16	ENTRY	(<10>^X
17	ENTRY	ABS
18	ENTRY	ACOS
19	ENTRY	AGTO
20	ENTRY	AOFF
21	ENTRY	AON
22	ENTRY	ARCL
23	ENTRY	ASHF
24	ENTRY	ASIN
25	ENTRY	ASN
26	ENTRY	ASTO
27	ENTRY	ATAN
28	ENTRY	AVIEW
29	ENTRY	AXEQ
30	ENTRY	BEEP
31	ENTRY	BST

32	ENTRY	CAT
33	ENTRY	CF
34	ENTRY	CHS
35	ENTRY	CLa
36	ENTRY	CLDSP
37	ENTRY	CLP
38	ENTRY	CLREG
39	ENTRY	CLSIG
40	ENTRY	CLST
41	ENTRY	CLX
42	ENTRY	COPY
43	ENTRY	COS
44	ENTRY	D-R
45	ENTRY	DEG
46	ENTRY	DEL
47	ENTRY	DELETE
48	ENTRY	DSE
49	ENTRY	END
50	ENTRY	ENG
51	ENTRY	ENTER^
52	ENTRY	E^X
53	ENTRY	E^X-1
54	ENTRY	FACT
55	ENTRY	FC?
56	ENTRY	FC?C
57	ENTRY	FIX
58	ENTRY	FRAC
59	ENTRY	FS?
60	ENTRY	FS?C
61	ENTRY	GRAD
62	ENTRY	GTO
63	ENTRY	GTOL
64	ENTRY	HMS+
65	ENTRY	HMS-
66	ENTRY	HMS-H
67	ENTRY	H-HMS
68	ENTRY	INT
69	ENTRY	ISG
70	ENTRY	LASTX
71	ENTRY	LBL
72	ENTRY	LN
73	ENTRY	LN1+X
74	ENTRY	LOG
75	ENTRY	MEAN
76	ENTRY	MOD
77	ENTRY	MODE
78	ENTRY	OFF
79	ENTRY	ONE/X
80	ENTRY	P-R
81	ENTRY	PACK
82	ENTRY	PCT
83	ENTRY	PCTCH
84	ENTRY	PI
85	ENTRY	PROMPT
86	ENTRY	PSE
87	ENTRY	R-D
88	ENTRY	R-P
89	ENTRY	R/S
90	ENTRY	RAD
91	ENTRY	RCL

92	ENTRY	RDN
93	ENTRY	RND
94	ENTRY	RTN
95	ENTRY	R ⁻
96	ENTRY	SCI
97	ENTRY	SF
98	ENTRY	SHIFT
99	ENTRY	SIGMA+
100	ENTRY	SIGMA-
101	ENTRY	SIGN
102	ENTRY	SIGREG
103	ENTRY	SIN
104	ENTRY	SIZE
105	ENTRY	SQRT
106	ENTRY	SST
107	ENTRY	STAYON
108	ENTRY	STDEV
109	ENTRY	STO
110	ENTRY	STO+
111	ENTRY	STO-
112	ENTRY	STO*
113	ENTRY	STO/
114	ENTRY	STOP
115	ENTRY	TAN
116	ENTRY	TONE
117	ENTRY	VIEW
118	ENTRY	X#0?
119	ENTRY	X#Y?
120	ENTRY	X<0?
121	ENTRY	X<=0?
122	ENTRY	X<=Y?
123	ENTRY	X<>
124	ENTRY	X<>Y
125	ENTRY	X<Y?
126	ENTRY	X=0?
127	ENTRY	X=Y?
128	ENTRY	X>0?
129	ENTRY	X>Y?
130	ENTRY	XEQ
131	ENTRY	XGOIND
132	ENTRY	X^2
133	ENTRY	Y^X
134		

*

* PKTTAB - PARSE KEY TYPE TABLE

* MUST START AT 0 IN QUAD 4 (1000-@10000)

* LOGICAL COLUMN 0

139	0	101 CON	65	A
140	1	106 CON	70	F
141	2	400 CON	256	SHIFT
142	3	0 CON	0	
143	4	2 CON	2	-
144	5	1 CON	1	+
145	6	3 CON	3	*
146	7	4 CON	4	/

* LOGICAL COLUMN 1

148	10	102 CON	66	B
149	11	107 CON	71	G
150	12	0 CON	0	
151	13	0 CON	0	

152	14	47 CON	39	7
153	15	44 CON	36	4
154	16	41 CON	33	1
155	17	40 CON	32	0
* LOGICAL COLUMN 2				
157	20	103 CON	67	C
158	21	110 CON	72	H
159	22	0 CON	0	
160	23	0 CON	0	
161	24	50 CON	40	8
162	25	45 CON	37	5
163	26	42 CON	34	2
164	27	1000 CON	512	DP
* LOGICAL COLUMN 3				
166	30	104 CON	68	D
167	31	111 CON	73	I
168	32	0 CON	0	
169	33	0 CON	0	
170	34	51 CON	41	9
171	35	46 CON	38	6
172	36	43 CON	35	3
173	37	0 CON	0	
* LOGICAL COLUMN 4				
175	40	105 CON	69	E
176	41	100 CON	64	J
177	42	0 CON	0	
178	43	17 CON	15	BACKARROW
179	44	200 CON	128	ALPHA
180	45	0 CON	0	
181	46	0 CON	0	
182				OFF KEY IS SPECIAL
183	47	230 CON	0230	
184	50	36 CON	036	
185	51	31 CON	031	
186	52 Y^X	260 C=N		
187	53	1 GSBLGX XY^X		
187	54	0		
188	55	543 GOTO	NFRXY* (131)	
189	56	253 CON	0253	
190	57	23 CON	023	
191	60	15 CON	015	
192	61	10 CON	010	
193	62 HMS+	260 C=N		
194	63	1 GSBLGX XTOHRS		
194	64	0		
195	65	360 HC EX		
196	66	270 C=REGN .2		
197	67	1 GSBLGX XTOHRS		
197	70	0		
198	71	416 A=C	W	
199	72	260 C=N		
200	73	1 GSBLGX AD2-10		
200	74	0		
201	75	210 S5=	1	
202	76	1 GSBLGX XTOHRS		
202	77	0		
203	100	313 GOTO	NFRXY* (131)	
204	101	255 CON	0255	
205	102	23 CON	023	
206	103	15 CON	015	

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

```

207 104          10 CON      @10
208 105 HMS-    260 C=N
209 106          1276 C=-C-1 S
210 107          360 CN EX
211 110          1523 GOTO    HMS+   ( 62 )
212 111          253 CON      @253
213 112 +        260 C=N
214 113          143 GOTO    ADD210 ( 127 )
215 114          204 CON      @204
216 115          17 CON      @17
217 116          15 CON      @15
218 117 MOD      260 C=N
219 120          1 GSBLGX MOD10
219 121          0
220 122          73 GOTO    NFRXY* ( 131 )
221 123          255 CON      @255
222 124 -        260 C=N
223 125          1276 C=-C-1 S
224 126          0 NOP
225 127 ADD210    1 GSBLGX AD2-10
225 130          0
226 131 NFRXY*    1 GOLNGX NFRXY
226 132          2
227 133          252 CON      @252
228 134 (*)      260 C=N
229 135          1 GSBLGX MP2-10
229 136          0
230 137          1723 GOTO    NFRXY* ( 131 )
231 140          245 CON      @245
232 141 PCT      646 A=A-1 X
233 142          646 A=A-1 X
234 143          260 C=N
235 144 TIMES    1 GSBLGX MP2-10
235 145          0
236 146          1 GOLNGX NFRX
236 147          2
237 150          262 CON      @262
238 151          36 CON      @36
239 152          30 CON      @30
240 153 X^2      260 C=N
241 154          416 A=C
242 155          1673 GOTO    TIMES ( 144 )
243 156          257 CON      @257
244 157 /        260 C=N
245 160          1 GSBLGX DY2-10
245 161          0
246 162          1473 GOTO    NFRXY* ( 131 )
247 163          223 CON      @223
248 164          2 CON      @2
249 165          1 CON      @1
250 166 ABS      260 C=N
251 167          136 C=0 S
252 170          1740 RTN
253 171          223 CON      @223
254 172          17 CON      @17
255 173          3 CON      @3
256 174          1 CON      @1
257 175 ACOS     1 GSBLGX TRGSET
257 176          0
258 177          1610 S0= 1

```



```

259 200 ASIN1 1410 S1= 1
260 201 ATAN1 1010 S2= 1
261 202 1 GOLNGX BRT100
261 203 2
262 204 0 CON 00
263 205 AGTO 1204 S7= 0
264 206 XGAXFR 1 GOLONG XGA00
264 207 2
265 210 214 CON 0214
266 211 3 CON 03
267 212 422 CON 0422
268 213 1001 CON 01001
269 214 ARCL 1 GOLNGX XARCL
269 215 2
270 216 206 CON 0206
271 217 10 CON 010
272 220 23 CON 023
273 221 1 CON 01
274 222 ASHF 1 GOLNGX XASHF
274 223 2
275 224 216 CON 0216
276 225 11 CON 011
277 226 23 CON 023
278 227 1 CON 01
279 230 ASIN 1 GSBLGX TRGSET
279 231 0
280 232 1463 GOTO ASIN1 ( 200 )
281 233 216 CON 0216
282 234 23 CON 023
283 235 401 CON 0401
284 236 ASN 1 GOLNGX XASN
284 237 2
285 240 217 CON 0217
286 241 24 CON 024
287 242 423 CON 0423
288 243 1001 CON 01001
289 244 ASTO 1 GOLNGX XASTO
289 245 2
290 246 216 CON 0216
291 247 1 CON 01
292 250 24 CON 024
293 251 1 CON 01
294 252 ATAN 1 GSBLGX TRGSET
294 253 0
295 254 1253 GOTO ATAN1 ( 201 )
296 255 227 CON 0227
297 256 5 CON 05
298 257 11 CON 011
299 260 26 CON 026
300 261 1 CON 01
301 262 AVIEW 1 GOLNGX XAVIEW
301 263 2
302 264 0 CON 00
303 265 AXEQ 1210 S7= 1
304 266 1203 GOTO XGAXFR ( 206 )
305 267 220 CON 0220
306 270 5 CON 05
307 271 5 CON 05
308 272 2 CON 02
309 273 BEEP 460 LDI

```

```

310 274          7 CON      7
311 275          1 GOLONG XBEEP
311 276          2
312 277          224 CON    @224
313 300          23 CON    @23
314 301          2 CON    @2
315 302 BST      0 NOP
316 303          1 GOLONG XBST
316 304          2
317 305          224 CON    @224
318 306          1401 CON   @1401
319 307          403 CON    @403
320 310 CAT      1 GOLNGX XCAT
320 311          2
321 312          1206 CON   @1206
322 313          1003 CON   @1003
323 314 CF       1 GOLNGX XCF
323 315          2
324 316          201 CON    @201
325 317          14 CON    @14
326 320          3 CON    @3
327 321 CLA      116 C=0
328 322          550 REGN=C 5
329 323          650 REGN=C 6
330 324          750 REGN=C 7
331 325          1050 REGN=C 8
332 326 NFRPUL   1740 RTN
333 327          217 CON    @217
334 330          24 CON    @24
335 331          1023 CON   @1023
336 332 STO      630 C=M
337 333          1360 DATA=C
338 334          1740 RTN
339 335          204 CON    @204
340 336          14 CON    @14
341 337          3 CON    @3
342 340 CLDSP    1 GOSUB DATOFF
342 341          0
* DATOFF ALSO CLEARS DATAENTRY FLAG, BUT THERE'S NO HARM DONE.
344 342          1 GOLONG NWGOOS
344 343          2
* IN KEYBOARD MODE, PUTTING UP A NEW GOOSE ISN'T VERY USEFUL,
* BUT SINCE THE DEFAULT DISPLAY LOGIC WRITES OVER IT, NO HARM
* IS DONE.
348 344          220 CON    @220
349 345          14 CON    @14
350 346          403 CON    @403
351 347 CLP      1 GOLONG CLRPGM
351 350          2
352 351          207 CON    @207
353 352          22 CON    @22
354 353          14 CON    @14
355 354          3 CON    @3
356 355 CLREG    410 SS= 1
357 356          1 GOLNGX CLR
357 357          2
358 360          316 CON    @316
359 361          14 CON    @14
360 362          3 CON    @3
361 363 CLSIG    1 GOLNGX XCLSIG

```

XKD

CLEAR MSGFLG

361	364	2		
362	365	224	CON	@224
363	366	23	CON	@23
364	367	14	CON	@14
365	370	3	CON	@3
366	371	116	C=0	
367	372	50	REGN=C	0
368	373	150	REGN=C	1
369	374	250	REGN=C	2
370	375	53	GOTO	XCLX1 (402)
371	376	230	CON	@230
372	377	14	CON	@14
373	400	3	CON	@3
374	401	116	C=0	
375			ENTRY	XCLX1
376	402	350	REGN=C	3
377	403	1	GOLONG	NFRSIG
377	404	2		
378	405	231	CON	@231
379	406	20	CON	@20
380	407	17	CON	@17
381	410	403	CON	@403
382	411	1	GOLNGX	XCOPY
382	412	2		
383	413	222	CON	@222
384	414	55	CON	@55
385	415	4	CON	@4
386	416	260	C=N	
387	417	1	GOLNGX	DTOR
387	420	2		
388	421	207	CON	@207
389	422	5	CON	@5
390	423	4	CON	@4
391	424	1	GOLNGX	XDEG
391	425	2		
392	426	204	CON	@204
393	427	1	CON	@1
394	430	22	CON	@22
395	431	7	CON	@7
396	432	1	GOLONG	XGRAD
396	433	2		
397	434	204	CON	@204
398	435	1	CON	@1
399	436	22	CON	@22
400	437	1	GOLONG	XRAD
400	440	2		
401	441	214	CON	@214
402	442	405	CON	@405
403	443	404	CON	@404
404	444	1	GOLNGX	DELHNN
404	445	2		
405	446	0	CON	@0
406	447	0	NOP	
407	450	1	GOLONG	XDELET
407	451	2		
408	452	205	CON	@205
409	453	423	CON	@423
410	454	1004	CON	@1004
411	455	1	GOLNGX	XDSE
411	456	2		

USED BY SIGMA+ AND SIGMA-
STORE NEW X

```

412 457          204 CON      @204
413 460          16 CON      @16
414 461          5 CON      @5
415          END
416 462          207 CON      @207
417 463          1416 CON     @1416
418 464          405 CON      @405
419 465 ENG      510 S6=      1
420 466          1 GOLNGX XSCI
420 467          2
421 470          236 CON      @236
422 471          22 CON      @22
423 472          5 CON      @05
424 473          24 CON      @24
425 474          16 CON      @16
426 475          5 CON      @5
427 476 ENTER^   1 GOSUB     R^SUB
427 477          0
428 500          270 C=REGN 2
429 501          350 REGN=C 3
430 502          1 GOLONG NFRENT
430 503          2
431 504          230 CON      @230
432 505          36 CON      @36
433 506          5 CON      @5
434 507 E^X      260 C=N
435 510          1 GOLNGX EXP10
435 511          2
436 512          226 CON      @226
437 513          4 CON      @4
438 514          1 CON      @1
439 515 ADVNCE   1565 CON      @1565
440 516          674 CON      @674
441 517          1740 RTN
442 520          224 CON      @224
443 521          3 CON      @3
444 522          1 CON      @1
445 523          6 CON      @6
446 524 FACT     260 C=N
447 525          1 GOLNGX XFT100
447 526          2
448 527          277 CON      @277
449 530          1003 CON      @1003
450 531          1006 CON      @1006
451 532 FC?      256 AC EX
452 533          1256 C=-C-1
453 534          256 AC EX
454 535          453 GOTO     FS?   ( 602 )
455 536          261 CON      @261
456 537          55 CON      @55
457 540          30 CON      @30
458 541          36 CON      @36
459 542          5 CON      @5
460 543 E^X-1    110 S4=      1
461 544          260 C=N
462 545          1 GOLONG EXP10
462 546          2
463 547          203 CON      @203
464 550          77 CON      @77
465 551          1003 CON      @1003

```

GOSUB PRT9

466	552	1006 CON	Q1006	
467	553 FC?C	1 GOSUB	XCF	
467	554	0		
468	555	1553 GOTO	FC?	(532)
469	556	230 CON	Q230	
470	557	1411 CON	Q1411	
471	560	406 CON	Q406	
472	561 FIX	1210 S7=	1	
473	562	1 GOLNGX	XSCI	
473	563	2		
474	564	224 CON	Q224	
475	565	16 CON	Q16	
476	566	11 CON	Q11	
477	567 INT	210 S5=	1	
478	570	43 GOTO	FRAC	(574)
479	571	203 CON	Q203	
480	572	22 CON	Q22	
481	573	6 CON	Q6	
482	574 FRAC	260 C=N		
483	575	1 GOLNGX	INTERC	
483	576	2		
484	577	277 CON	Q277	
485	600	1023 CON	Q1023	
486	601	1006 CON	Q1006	
487	602 FS?	1 GOLNGX	XFS?	
487	603	2		
488	604	203 CON	Q203	
489	605	77 CON	Q77	
490	606	1023 CON	Q1023	
491	607	1006 CON	Q1006	
492	610 FS?C	1 GOSUB	XCF	
492	611	0		
493	612	1703 GOTO	FS?	(602)
494	613	0 CON	0	
495	614 GTOL	1 GOLONG	GT0NN	
495	615	2		
496	616	217 CON	Q217	
497	617	1424 CON	Q1424	
498	620	1407 CON	Q1407	
499	621 GTO			CAN'T BE FOLLOWED BY 0
500	621	222 CON	Q222	
501	622	10 CON	Q10	
502	623 HMS-H	260 C=N		
503	624	1 GOLNGX	XTOHRS	
503	625	2		
504	626	223 CON	Q223	
505	627	15 CON	Q15	
506	630	10 CON	Q10	
507	631 H-HMS	210 S5=	1	
508	632	1713 GOTO	HMS-H	(623) SAVE CODE HERE
509	633	207 CON	Q207	
510	634	423 CON	Q423	
511	635	1011 CON	Q1011	
512	636 ISG	1604 S0=	0	
513	637	1 GOLNGX	XISG	
513	640	2		
514	641	214 CON	Q214	
515	642	2 CON	Q2	
516	643	1414 CON	Q1414	
517	643 LBL			CAN'T BE FOLLOWED BY 0

```

518 644      216 CON      @216
519 645      14 CON      @14
520 646 LN    260 C=N
521 647      1 GOLNGX LN10
521 650      2
522 651      207 CON      @207
523 652      17 CON      @17
524 653      14 CON      @14
525 654 LOG   210 S5=      1
526 655      1713 GOTO    LN      ( 646 ) SAVE SPACE HERE
527 656      226 CON      @226
528 657      5 CON      @5
529 660      4 CON      @4
530 661      23 CON      @023
531 662 STDEV 1 GSBLGX SD
531 663      0
532 664      73 GOTO     NFRNC* ( 673 )
533 665      216 CON      @216
534 666      1 CON      @1
535 667      5 CON      @5
536 670      15 CON      @15
537 671 MEAN  1 GSBLGX XBAR
537 672      0
538 673 NFRNC* 1 GOLNGX NFRNC
538 674      2
539 675      220 CON      @220
540 676      55 CON      @55
541 677      22 CON      @22
542 700 R-P   1 GSBLGX TRGSET
542 701      0
543 702      1 GSBLGX TOPQL
543 703      0
544 704      1673 GOTO    NFRNC* ( 673 )
545 705      206 CON      @206
546 706      6 CON      @6
547 707      17 CON      @17
548 710 OFF  1340 DISOFF
549 711      460 LDI
550 712      11 CON      9
551 713      1 GOSUB    ROMCHK
551 714      0
552 715      1 GOSUB    MEMCHK
552 716      0
553 717      1 GOSUB    RSTKB
553 720      0
554 721      1 GOLONG   DRSY50
554 722      2
555 723      230 CON      @230
556 724      57 CON      @57
557 725      61 CON      @61
558 726 ONE/X 260 C=N
559 727      1 GOLNGX 1/X10
559 730      2
560 731      222 CON      @222
561 732      55 CON      @55
562 733      20 CON      @20
563 734 P-R   1 GSBLGX TRGSET
563 735      0
564 736      1410 S1=      1
565 737      1010 S2=      1

```

```

566 740          1 GSBLGX TOREC
566 741          0
567 742        1313 GOTO  NFRNC* ( 673)
568 743        213 CON    @213
569 744          3 CON    @3
570 745          1 CON    @1
571 746          20 CON    @20
572 747 PACK      1 GOLONG XPACK
572 750          2
573 751        210 CON    @210
574 752          3 CON    @3
575 753          45 CON    @45
576 754 PCTCH     276 AC EX  S
577 755        1276 C=-C-1 S
578 756          276 AC EX  S
579 757          260 C=N
580 760          1 GSBLGX AD2-10
580 761          0
581 762        546 A=A+1  X
582 763        546 A=A+1  X
583 764        270 C=REGH 2
584 765          1 GSBLGX DV1-10
584 766          0
585 767          1 GOLNGX NFRX
585 770          2
586 771        205 CON    @205
587 772          23 CON    @23
588 773          20 CON    @20
589 774 PSE      1314 ?S13=1          RUNNING?
590 775          43 GONC   PSE10 (1001) NO
591 776        1410 S1=    1          SET PAUSEFLAG
592 777          1 GOSUB   STOST0      PUT SS 0 BACK
592 1000         0
593 1001 PSE10    1 GOLONG PSESTP
593 1002         2
594 1003        224 CON    @224
595 1004        20 CON    @20
596 1005        15 CON    @15
597 1006        17 CON    @17
598 1007        22 CON    @22
599 1010        20 CON    @20
600 1011 PROMPT  1 GOLNGX XPRMPT
600 1012         2
601 1013        204 CON    @204
602 1014        55 CON    @55
603 1015        22 CON    @22
604 1016 R-D     260 C=N
605 1017          1 GOLNGX RTOD
605 1020         2
606 1021        220 CON    @220
607 1022        17 CON    @17
608 1023        24 CON    @24
609 1024        23 CON    @23
610 1025 STOP    1 GOLONG STOPSB
610 1026         2
611 1027          0 CON    @0          NO PROMPTING
612 1030 R/S      0 NOP              XKD
613 1031          1 GOLNGX XR/S
613 1032         2
614 1033        230 CON    @230

```

615	1034	53	CON	053	
616	1035	61	CON	061	
617	1036	16	CON	016	
618	1037	14	CON	014	
619	1040	260	C=N		
620	1041	1	GOLONG	.XLN1+X	
620	1042	2			
621	1043	230	CON	0230	
622	1044	24	CON	024	
623	1045	23	CON	023	
624	1046	1	CON	01	
625	1047	14	CON	014	
626	1050	470	C=REGN	4	
627	1051	356	CB EX		
628	1052	43	GOTO	RCL	(1056)
629	1053	214	CON	0214	
630	1054	403	CON	0403	
631	1055	1022	CON	01022	
632	1056	614	?S11=1		
633	1057	1	GSUBCX	R*SUB	
633	1060	1			
634	1061	116	C=0		
635	1062	1160	DADD=C		
636	1063	356	CB EX		
637	1064	350	REGN=C	3	
638	1065	1	GOLNGX	NFRPRL	
638	1066	2			
639	1067	223	CON	0223	
640	1070	10	CON	8	
641	1071	3	CON	3	
642	1072	260	C=N		GET X
643	1073	1372	? C#0	M	
644	1074	33	GONC	DONCHS (1077)	X IS ZERO DO NOTHING
645	1075	1276	C=-C-1	S	DO CHS
646	1076	350	REGN=C	3	
647	1077	1663	GOTO	NFRPRL (1065)	
648	1100	211	CON	0211	
649	1101	20	CON	020	
650	1102	1240	SETDEC		
651	1103	1	GSLGX	PI/2	
651	1104	0			
652	1105	756	C=C+C	W	
653	1106	1072	C=C+1	M	
654	1107	106	C=0	X	
655	1110	1413	GOTO	LXEX (1051)	
656	1111	276	CON	0276	
657	1112	474	CON	0474	
658	1113	1030	CON	01030	
659	1114	630	C=N		
660	1115	1360	DATA=C		
661	1116	1433	GOTO	NFRCL (1061)	
662	1117	216	CON	0216	
663	1120	4	CON	04	
664	1121	22	CON	022	
665	1122	1	GOLNGX	XRDN	
665	1123	2			
666	1124	204	CON	0204	
667	1125	16	CON	016	
668	1126	22	CON	022	
669	1127	1	GOLNGX	XRND	

669	1130	2		
670	1131	216	CON	@216
671	1132	24	CON	@24
672	1133	22	CON	@22
673	1134	1	GOLNGX	XRTN
673	1135	2		
674	1136	236	CON	@236
675	1137	22	CON	@22
676	1140	1	GOLNGX	XR^
676	1141	2		
677	1142	211	CON	@211
678	1143	1403	CON	@1403
679	1144	423	CON	@423
680	1145	1	GOLNGX	XSCI
680	1146	2		
681	1147	1206	CON	@1206
682	1150	1023	CON	@1023
683	1151	1	GOLNGX	XSF
683	1152	2		
684	1153	253	CON	@253
685	1154	116	CON	@116
686	1155	1	GOLNGX	SIGMA
686	1156	2		
687	1157	255	CON	@255
688	1160	116	CON	@116
689	1161	210	SS=	1
690	1162	1733	GOTO	SIGMA+ (1155) SAVE CODE HERE
691	1163	207	CON	@207
692	1164	5	CON	@5
693	1165	1022	CON	@1022
694	1166	1116	CON	@1116
695	1167	1	GOLNGX	XSGREG
695	1170	2		
696	1171	223	CON	@223
697	1172	17	CON	@17
698	1173	3	CON	@3
699	1174	1	GSBLGX	TRGSET
699	1175	0		
700	1176	153	GOTO	COS1 (1213)
701	1177	216	CON	@216
702	1200	1	CON	@1
703	1201	24	CON	@24
704	1202	1	GSBLGX	TRGSET
704	1203	0		
705	1204	103	GOTO	XTRIG (1214)
706	1205	216	CON	@216
707	1206	11	CON	@11
708	1207	23	CON	@23
709	1210	1	GSBLGX	TRGSET
709	1211	0		
710	1212	1610	S0=	1
711	1213	1410	S1=	1
712	1214	1	GOLNGX	TRG100
712	1215	2		
713	1216	205	CON	@205
714	1217	32	CON	@32
715	1220	411	CON	@411
716	1221	423	CON	@423
717	1222	1	GOLNGX	XSIZE
717	1223	2		

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

718 1224	224 CON	@224	
719 1225	22 CON	@22	
720 1226	21 CON	@21	
721 1227	23 CON	@23	
722 1230 SQRT	260 C=N		
723 1231	1 GOLNGX	SQR10	
723 1232	2		
724 1233	224 CON	@224	
725 1234	23 CON	@23	
726 1235	23 CON	@23	
727 1236 SST	0 NOP		KKD
728 1237	1 GOLNGX	XSST	
728 1240	2		
729 1241	216 CON	@216	
730 1242	17 CON	@17	
731 1243 STAYON	1 GOLONG	XSTYON	
731 1244	2		
732 1245	252 CON	@252	
733 1246	424 CON	@424	
734 1247	1023 CON	@1023	
735 1250 STO*	1 GSBLGX	SEPMY	
735 1251	0		
736 1252	1 GSBLGX	MP2-10	
736 1253	0		
737 1254	103 GOTO	NFRST (1264)	
738 1255	253 CON	@253	
739 1256	424 CON	@424	
740 1257	1023 CON	@1023	
741 1260 STO+	1 GSBLGX	SEPMY	
741 1261	0		
742 1262	1 GSBLGX	AD2-10	
742 1263	0		
743 1264 NFRST	1 GOLONG	NFRST+	
743 1265	2		
744 1266	255 CON	@255	
745 1267	424 CON	@424	
746 1270	1023 CON	@1023	
747 1271 STO-	630 C=M		
748 1272	1240 SETDEC		
749 1273	1276 C=-C-1 S		
750 1274	530 M=C		
751 1275	1633 GOTO	STO+ (1260)	
752 1276	257 CON	@257	
753 1277	424 CON	@424	
754 1300	1023 CON	@1023	
755 1301 STO/	1 GSBLGX	SEPMY	
755 1302	0		
756 1303	1 GSBLGX	DV2-10	
756 1304	0		
757 1305	1573 GOTO	NFRST (1264)	
758 1306	230 CON	@230	
759 1307	36 CON	@36	
760 1310	60 CON	@60	
761 1311	61 CON	@61	
762 1312 (10)*X	1 GOLNGX	10TOX	
762 1313	2		
763 1314	205 CON	@205	
764 1315	16 CON	@16	
765 1316	1417 CON	@1417	
766 1317	424 CON	@424	

767 1320 TONE	1 GOLNGX XTONE	
767 1321	2	
768 1322	227 CON	@227
769 1323	5 CON	@5
770 1324	411 CON	@411
771 1325	1026 CON	@1026
772 1326 VIEW	1 GOLNGX XVIEW	
772 1327	2	
773 1330	277 CON	@277
774 1331	60 CON	@60
775 1332	115 CON	@115
776 1333	30 CON	@30
777 1334 X#0?	1 GOLNGX XX#0?	
777 1335	2	
778 1336	277 CON	@277
779 1337	31 CON	@31
780 1340	115 CON	@115
781 1341	30 CON	@30
782 1342 X#Y?	1 GOLNGX XX#Y?	
782 1343	2	
783 1344	277 CON	@277
784 1345	60 CON	@60
785 1346	74 CON	@74
786 1347	30 CON	@30
787 1350 X<0?	1 GOLNGX XX<0?	
787 1351	2	
788 1352	277 CON	@277
789 1353	60 CON	@60
790 1354	75 CON	@75
791 1355	74 CON	@74
792 1356	30 CON	@30
793 1357 X<=0?	1 GOLONG XX<=0A	
793 1360	2	
794 1361	277 CON	@277
795 1362	31 CON	@31
796 1363	75 CON	@75
797 1364	74 CON	@74
798 1365	30 CON	@30
799 1366 X<=Y?	1 GOLNGX XX<=Y?	
799 1367	2	
800 1370	231 CON	@231
801 1371	76 CON	@76
802 1372	74 CON	@74
803 1373	30 CON	@30
804 1374 X<>Y	370 C=REGN 3	
805 1375	256 AC EX	
806 1376	270 C=REGN 2	
807 1377	350 REGN=C 3	
808 1400	256 AC EX	
809 1401	250 REGN=C 2	
810 1402	1 GOLNGX HFRPR	
810 1403	2	
811 1404	277 CON	@277
812 1405	31 CON	@31
813 1406	74 CON	@74
814 1407	30 CON	@30
815 1410 X<Y?	1 GOLONG XX<Y?	
815 1411	2	
816 1412	277 CON	@277
817 1413	60 CON	@60

818 1414	75 CON	@75	
819 1415	30 CON	@30	
820 1416 X=0?	1 GOLNGX	XX=0?	
820 1417	2		
821 1420	277 CON	@277	
822 1421	31 CON	@31	
823 1422	75 CON	@75	
824 1423	30 CON	@30	
825 1424 X=Y?	1 GOLNGX	XX=Y?	
825 1425	2		
826 1426	277 CON	@277	
827 1427	60 CON	@60	
828 1430	76 CON	@76	
829 1431	30 CON	@30	
830 1432 X>0?	1 GOLNGX	XX>0?	
830 1433	2		
831 1434	277 CON	@277	
832 1435	31 CON	@31	
833 1436	76 CON	@76	
834 1437	30 CON	@30	
835 1440 X>Y?	1 GOLNGX	XX>Y?	
835 1441	2		
836 1442	0 CON	0	
837 1443 XGOIND	1 GOLONG	XGI	
837 1444	2		
838 1445	221 CON	@221	
839 1446	405 CON	@0405	
840 1447	1430 CON	@1430	
841 XEQ			CAN'T BE FOLLOWED BY 0
842 1450	203 CON	@203	
843 1451	5 CON	5	
844 1452	4 CON	4	
845 1453 -DEC	110 S4=	1	
846 1454	43 GOTO	-OCT (1460)	
847 1455	224 CON	@224	
848 1456	3 CON	3	
849 1457	17 CON	15	
850 1460 -OCT	260 C=N		
851 1461	1 GOLNGX	TOOCT	
851 1462	2		
852 1463	216 CON	@216	
853 1464	7 CON	7	
854 1465	11 CON	9	
855 1466	23 CON	19	
856 1467 SIGN	1 GOLONG	XSIGN	
856 1470	2		
* 859 1471	216 CON	@216	N
* 860 1472	17 CON	15	O
861 1473	1 CON	1	A
862 1474 AON	1210 S7=	1	SET ALPHAMODE
863 1475 AON10	1 GOSUB	STOSTO	
863 1476	0		
864 1477	1 GOLONG	ANNOUT	
864 1500	2		
* 867 1501	206 CON	@206	F
868 1502	6 CON	6	F

```

869 1503          17 CON      15          0
870 1504          1 CON      1          A
871 1505 AOFF     1204 S7=    0          CLEAR ALPHAMODE
872 1506          1673 GOTO   AON10 (1475)

*
*
875 1507          0 CON      00          NO PROMPTING
876 1510 SHIFT    0 NOP          XKD
877 1511          1 GOSUB   TGSHF1     TOGGLE SHIFT FLAG
877 1512          0
878 1513          253 GOTO   USCOM1 (1540)

*
*
881 1514          0 CON      00          NO PROMPTING
882 1515 MODE     0 NOP          XKD
883 1516          1040 C=KEYS

*
885                      ENTRY MODE1     FOR WAND ALPHA,PRGM,USER
* ENTRY POINT ADD FOR WAND ON 3-13-79
*
888      MODE1
889 1517          34 PT=      3
890 1520          742 C=C+C  PT
891 1521          742 C=C+C  PT
892 1522          742 C=C+C  PT
893 1523          203 GONC    ALFPRG (1543)
894                      USER KEY
895 1524          1670 C=REGH 14
896 1525          574 RCR      6
897 1526          1730 CST EX
898 1527          114 ?S4=1
899 1530          37 GOC      USEROF (1533) YES
900 1531          110 S4=      1          SET USERMODE
901 1532          23 GOTO     USERC (1534)
902 1533 USEROF    104 S4=      0          CLEAR USERMODE
903 1534 USERC    1730 CST EX
904 1535          474 RCR      8
905 1536 USCOM     1630 C=ST
906 1537          1650 REGN=C 14          MERGE SS 0 WITH OTHER SETS
907 1540 USCOM1   1104 S9=      0          KEYBOARD NOT RESET YET
908 1541          1 GOLONG    DRSY51     REFRESH ANNUNCIATORS ONLY
908 1542          2

*
*
911 1543 ALFPRG    1515 CON      01515     GOSUB PRT14
912 1544          674 CON      0674
913                      ENTRY   PR14RT     FOR PRINTER
914      PR14RT
915 1545          1342 ? C#0  PT          PRGM KEY?
916 1546          177 GOC      PRGM (1565) YES
917 1547          1 GOSUB     RSTMS1     ALPHA KEY
917 1550          0
918 1551          1214 ?S7=1
919 1552          73 GONC      ALPHON (1561) NO
920 1553          1204 S7=      0          CLEAR ALPHAMODE
921 1554 APCOM     1630 C=ST          MERGE SS0 W/ OTHER SETS
922 1555          1650 REGN=C 14          PUT STATUS SETS BACK
923 1556 D05XFR   1104 S9=      0          KEYBOARD NOT RESET YET
924 1557          1 GOLONG    DRSY05     REFRESH MAIN DISPLAY
924 1560          2

```

```

925 1561 ALPHON 1210 S7=      1      SET ALPHAMODE
                                14 ?S3=1  PROGRAM MODE?
926 1562                                1713 GONC  APCOM  (1554) NO
927 1563                                1523 GOTC  USCOM  (1536) YES
928 1564

```

*

*

```

931 1565 PRGM      1 GOSUB  RSTMS1
931 1566            0
932 1567            1404 S1=      0      CLEAR PAUSEFLAG
933 1570            1204 S7=      0      CLEAR ALPHAMODE
934 1571            14 ?S3=1      PRGMOF (1575) YES
935 1572            37 GOC        NO. SET PRGMODE
936 1573            10 S3=      1
937 1574            1603 GOTO     APCOM  (1554)
938 1575 PRGMOF      4 S3=      0      CLEAR PRGMODE
939 1576            1630 C=ST
940 1577            1650 REGN=C  14      PUT STATUS SETS BACK
941 1600            1 GOSUB  DECMPL  DECOMPILE
941 1601            0
942 1602            1543 GOTO  D05XFR (1556)
943
944
945

```

```

947 1603 CAT##3    116 C=0      MOVE # TO A MANT
948 1604            346 BC EX   X
949 1605            674 RCR     11
950 1606            256 AC EX
951 1607            1 GOSUB  ALPDEF  SEL. CORECT DEF
951 1610            0
952 1611            0 NOP
953 1612            112 DEF      +      ( 112)
954 1613            124 DEF      -      ( 124)
955 1614            134 DEF      (*)     ( 134)
956 1615            157 DEF      /      ( 157)
957 1616            726 DEF      ONE/X   ( 726)
958 1617            1312 DEF     (10)*X  (1312)
959 1620            166 DEF      ABS      ( 166)
960 1621            175 DEF      ACOS     ( 175)
961 1622            515 DEF      ADVNCE   ( 515)
962 1623            1505 DEF      AOFF    (1505)
963 1624            1474 DEF      AON     (1474)
964 1625            214 DEF      ARCL     ( 214)
965 1626            222 DEF      ASHF     ( 222)
966 1627            230 DEF      ASIN     ( 230)
967 1630            236 DEF      ASH      ( 236)
968 1631            244 DEF      ASTO     ( 244)
969 1632            252 DEF      ATAN     ( 252)
970 1633            262 DEF      AVIEW    ( 262)
971 1634            273 DEF      BEEP     ( 273)
972 1635            302 DEF      BST      ( 302)
973 1636            310 DEF      CAT      ( 310)
974 1637            314 DEF      CF       ( 314)
975 1640            1072 DEF      CHS     (1072)
976 1641            321 DEF      CLA      ( 321)
977 1642            340 DEF      CLDSP    ( 340)
978 1643            347 DEF      CLP      ( 347)
979 1644            355 DEF      CLREG    ( 355)
980 1645            363 DEF      CLSIG    ( 363)
981 1646            371 DEF      CLST     ( 371)

```

982 1647	401 DEF	CLX	(401)
983 1650	411 DEF	COPY	(411)
984 1651	1174 DEF	COS	(1174)
985 1652	416 DEF	D-R	(416)
986 1653	1453 DEF	-DEC	(1453)
987 1654	424 DEF	DEG	(424)
988 1655	444 DEF	DEL	(444)
989 1656	455 DEF	DSE	(455)
990 1657	462 DEF	END	(462)
991 1660	465 DEF	ENG	(465)
992 1661	476 DEF	ENTER*	(476)
993 1662	507 DEF	E*X	(507)
994 1663	543 DEF	E*X-1	(543)
995 1664	524 DEF	FACT	(524)
996 1665	532 DEF	FC?	(532)
997 1666	553 DEF	FC?C	(553)
998 1667	561 DEF	FIX	(561)
999 1670	574 DEF	FRAC	(574)
1000 1671	602 DEF	FS?	(602)
1001 1672	610 DEF	FS?C	(610)
1002 1673	432 DEF	GRAD	(432)
1003 1674	621 DEF	GTO	(621)
1004 1675	631 DEF	H-HMS	(631)
1005 1676	62 DEF	HMS+	(62)
1006 1677	105 DEF	HMS-	(105)
1007 1700	623 DEF	HMS-H	(623)
1008 1701	567 DEF	INT	(567)
1009 1702	636 DEF	ISG	(636)
1010 1703	1050 DEF	LASTX	(1050)
1011 1704	644 DEF	LBL	(644)
1012 1705	646 DEF	LN	(646)
1013 1706	1040 DEF	LN1+X	(1040)
1014 1707	654 DEF	LOG	(654)
1015 1710	671 DEF	MEAN	(671)
1016 1711	117 DEF	MOD	(117)
1017 1712	1460 DEF	-OCT	(1460)
1018 1713	710 DEF	OFF	(710)
1019 1714	1243 DEF	STAYON	(1243)
1020 1715	734 DEF	P-R	(734)
1021 1716	747 DEF	PACK	(747)
1022 1717	141 DEF	PCT	(141)
1023 1720	754 DEF	PCTCH	(754)
1024 1721	1102 DEF	PI	(1102)
1025 1722	1011 DEF	PROMPT	(1011)
1026 1723	774 DEF	PSE	(774)
1027 1724	1140 DEF	R*	(1140)
1028 1725	1016 DEF	R-D	(1016)
1029 1726	700 DEF	R-P	(700)
1030 1727	437 DEF	RAD	(437)
1031 1730	1056 DEF	RCL	(1056)
1032 1731	1122 DEF	RDN	(1122)
1033 1732	1127 DEF	RND	(1127)
1034 1733	1134 DEF	RTN	(1134)
1035 1734	662 DEF	STDEV	(662)
1036 1735	1145 DEF	SCI	(1145)
1037 1736	1151 DEF	SF	(1151)
1038 1737	1155 DEF	SIGNA+	(1155)
1039 1740	1161 DEF	SIGNA-	(1161)
1040 1741	1167 DEF	SIGREG	(1167)
1041 1742	1210 DEF	SIN	(1210)

1042	1743	1467	DEF	SIGN	(1467)
1043	1744	1222	DEF	SIZE	(1222)
1044	1745	1230	DEF	SQRT	(1230)
1045	1746	1236	DEF	SST	(1236)
1046	1747	1260	DEF	STO+	(1260)
1047	1750	1271	DEF	STO-	(1271)
1048	1751	1250	DEF	STO*	(1250)
1049	1752	1301	DEF	STO/	(1301)
1050	1753	332	DEF	STO	(332)
1051	1754	1025	DEF	STOP	(1025)
1052	1755	1202	DEF	TAN	(1202)
1053	1756	1320	DEF	TOHE	(1320)
1054	1757	1326	DEF	VIEW	(1326)
1055	1760	1416	DEF	X=0?	(1416)
1056	1761	1334	DEF	X#0?	(1334)
1057	1762	1350	DEF	X<0?	(1350)
1058	1763	1357	DEF	X<=0?	(1357)
1059	1764	1432	DEF	X>0?	(1432)
1060	1765	1424	DEF	X=Y?	(1424)
1061	1766	1342	DEF	X#Y?	(1342)
1062	1767	1410	DEF	X<Y?	(1410)
1063	1770	1366	DEF	X<=Y?	(1366)
1064	1771	1440	DEF	X>Y?	(1440)
1065	1772	1114	DEF	X<>	(1114)
1066	1773	1374	DEF	X<>Y	(1374)
1067	1774	1450	DEF	XEQ	(1450)
1068	1775	153	DEF	X^2	(153)
1069	1776	52	DEF	Y^X	(52)
1070	1777	0	CON	0	
1071					
1072					
1073					

*
*
*

* NOTE: NO EXECUTION POINT MAY BE LOCATED AFTER @1736 IN THIS
* QUAD. THE SEARCH ALGORITHM USES ENTRIES > @1736 IN THE
* MAIN FUNCTION TABLE AS DELIMITERS OF HOLES IN THE TABLE.

1082 UNLIST

ERRORS : 0

SYMBOL TABLE

(<*)	134	-	1614
(<10)*X	1312	-	1617
+	112	-	1612
-	124	-	1613
-DEC	1453	-	1653
-OCT	1460	-	1712 1454
/	157	-	1615
ABS	166	-	1620
ACOS	175	-	1621
ADD210	127	-	113
ADVANCE	515	-	1622
AGTO	205	-	
ALFPRG	1543	-	1523
ALFHON	1561	-	1552
AOFF	1505	-	1623
AON	1474	-	1624
AON10	1475	-	1506
APCOM	1554	-	1574 1563
ARCL	214	-	1625
ASHF	222	-	1626
ASIN	230	-	1627
ASIN1	200	-	232
ASN	236	-	1630
ASTO	244	-	1631
ATAN	252	-	1632
ATAN1	201	-	254
AVIEW	262	-	1633
AXEQ	265	-	
BEEP	273	-	1634
BST	302	-	1635
CAT	310	-	1636
CAT##3	1603	-	
CF	314	-	1637
CHS	1072	-	1640
CLA	321	-	1641
CLDSP	340	-	1642
CLP	347	-	1643
CLREG	355	-	1644
CLSIG	363	-	1645
CLST	371	-	1646
CLX	401	-	1647
COPY	411	-	1650
COS	1174	-	1651
COS1	1213	-	1176
D-R	416	-	1652
D05XFR	1556	-	1602
DEG	424	-	1654
DEL	444	-	1655
DELETE	447	-	
DONCHS	1077	-	1074
DSE	455	-	1656
END	462	-	1657
ENG	465	-	1660
ENTER^	476	-	1661
E^X	507	-	1662
E^X-1	543	-	1663

FACT	524	-	1664				
FC?	532	-	1665	555			
FC?C	553	-	1666				
FIX	561	-	1667				
FRAC	574	-	1670	570			
FS?	602	-	1671	612	535		
FS?C	610	-	1672				
GRAD	432	-	1673				
GTO	621	-	1674				
GTOL	614	-					
H-HMS	631	-	1675				
HMS+	62	-	1676	110			
HMS-	105	-	1677				
HMS-H	623	-	1700	632			
INT	567	-	1701				
ISG	636	-	1702				
LASTX	1050	-	1703				
LBL	644	-	1704				
LN	646	-	1705	655			
LN1+X	1040	-	1706				
LOG	654	-	1707				
LXEX	1051	-	1110				
MEAN	671	-	1710				
MOD	117	-	1711				
MODE	1515	-					
MODE1	1517	-					
NFRNC*	673	-	742	704	664		
NFRPRL	1065	-	1077				
NFRPUL	326	-					
NFRST	1264	-	1305	1254			
NFRXY*	131	-	162	137	122	100	55
NPRCL	1061	-	1116				
OFF	710	-	1713				
ONE/X	726	-	1616				
P-R	734	-	1715				
PACK	747	-	1716				
PCT	141	-	1717				
PCTCH	754	-	1720				
PI	1102	-	1721				
PR14RT	1545	-					
PRGM	1565	-	1546				
PRGNOF	1575	-	1572				
PROMPT	1011	-	1722				
PSE	774	-	1723				
PSE10	1001	-	775				
R-D	1016	-	1725				
R-P	700	-	1726				
R/S	1030	-					
RAD	437	-	1727				
RCL	1056	-	1730	1052			
RDN	1122	-	1731				
RND	1127	-	1732				
RTN	1134	-	1733				
R*	1140	-	1724				
SCI	1145	-	1735				
SF	1151	-	1736				
SHIFT	1510	-					
SIGMA+	1155	-	1737	1162			
SIGMA-	1161	-	1740				
SIGN	1467	-	1743				

SIGREG	1167	-	1741
SIN	1210	-	1742
SIZE	1222	-	1744
SQRT	1230	-	1745
SST	1236	-	1746
STAYON	1243	-	1714
STDEV	662	-	1734
STO	332	-	1753
STO*	1250	-	1751
STO+	1260	-	1747 1275
STO-	1271	-	1750
STO/	1301	-	1752
STOP	1025	-	1754
TAN	1202	-	1755
TIMES	144	-	155
TOHE	1320	-	1756
USCOM	1536	-	1564
USCOM1	1540	-	1513
USERC	1534	-	1532
USEROF	1533	-	1530
VIEW	1326	-	1757
X#0?	1334	-	1761
X#Y?	1342	-	1766
X<0?	1350	-	1762
X<=0?	1357	-	1763
X<=Y?	1366	-	1770
X<>	1114	-	1772
X<>Y	1374	-	1773
X<Y?	1410	-	1767
X=0?	1416	-	1760
X=Y?	1424	-	1765
X>0?	1432	-	1764
X>Y?	1440	-	1771
XCLX1	402	-	375
XEQ	1450	-	1774
XGAXFR	206	-	266
XGOIND	1443	-	
XTRIG	1214	-	1204
X^2	153	-	1775
Y^X	52	-	1776

NOMAS
 Not Manufacturer Supported
 recipient agrees NOT to contact manufacturer

ENTRY TABLE

(*)	134	-
(10)^X	1312	-
+	112	-
-	124	-
-DEC	1453	-
-OCT	1460	-
/	157	-
ABS	166	-
ACOS	175	-
ADVANCE	515	-
AGTO	205	-
AOFF	1505	-
AON	1474	-
ARCL	214	-
ASHF	222	-
ASIN	230	-
ASN	236	-
ASTO	244	-
ATAN	252	-
AVIEW	262	-
AXEQ	265	-
BEEP	273	-
BST	302	-
CAT	310	-
CAT##3	1603	-
CF	314	-
CHS	1072	-
CLA	321	-
CLDSP	340	-
CLP	347	-
CLREG	355	-
CLSIG	363	-
CLST	371	-
CLX	401	-
COPY	411	-
COS	1174	-
D-R	416	-
DEG	424	-
DEL	444	-
DELETE	447	-
DSE	455	-
END	462	-
ENG	465	-
ENTER^	476	-
E^X	507	-
E^X-1	543	-
FACT	524	-
FC?	532	-
FC?C	553	-
FIX	561	-
FRAC	574	-
FS?	602	-
FS?C	610	-
GRAD	432	-
GTO	621	-
GTOL	614	-

H-HMS	631	-
HMS+	62	-
HMS-	105	-
HMS-H	623	-
INT	567	-
ISG	636	-
LASTX	1050	-
LBL	644	-
LN	646	-
LN1+X	1040	-
LOG	654	-
MEAN	671	-
MOD	117	-
MODE	1515	-
MODE1	1517	-
OFF	710	-
ONE/X	726	-
P-R	734	-
PACK	747	-
PCT	141	-
PCTCH	754	-
PI	1102	-
FR14RT	1545	-
PROMPT	1011	-
PSE	774	-
R-D	1016	-
R-P	700	-
R/S	1030	-
RAD	437	-
RCL	1056	-
RDN	1122	-
RND	1127	-
RTN	1134	-
R^	1140	-
SCI	1145	-
SF	1151	-
SHIFT	1510	-
SIGMA+	1155	-
SIGMA-	1161	-
SIGN	1467	-
SIGREG	1167	-
SIN	1210	-
SIZE	1222	-
SQRT	1230	-
SST	1236	-
STAYON	1243	-
STDEV	662	-
STO	332	-
STO*	1250	-
STO+	1260	-
STO-	1271	-
STO/	1301	-
STOP	1025	-
TAN	1202	-
TONE	1320	-
VIEW	1326	-
X#0?	1334	-
X#Y?	1342	-
X<0?	1350	-
X<=0?	1357	-

X<=Y?	1366	-
X<>	1114	-
X<>Y	1374	-
X<Y?	1410	-
X=0?	1416	-
X=Y?	1424	-
X>0?	1432	-
X>Y?	1440	-
XCLX1	402	-
XEQ	1450	-
XGOIND	1443	-
X^2	153	-
Y^X	52	-

EXTERNAL REFERENCES

1/X10	727			
1/X10	730			
10TOX	1312			
10TOX	1313			
AD2-10	73	127	760	1262
AD2-10	74	130	761	1263
ALPDEF	1607			
ALPDEF	1610			
ANNOUT	1477			
ANNOUT	1500			
BRT100	202			
BRT100	203			
CLR	356			
CLR	357			
CLRPGM	347			
CLRPGM	350			
DATOFF	340			
DATOFF	341			
DECMPL	1600			
DECMPL	1601			
DELNNN	444			
DELNNN	445			
DRSY05	1557			
DRSY05	1560			
DRSY50	721			
DRSY50	722			
DRSY51	1541			
DRSY51	1542			
DTOR	417			
DTOR	420			
DV1-10	765			
DV1-10	766			
DV2-10	160	1303		
DV2-10	161	1304		
EXP10	510	545		
EXP10	511	546		
GTONN	614			
GTONN	615			
INTFRC	575			
INTFRC	576			
LN10	647			
LN10	650			
MEMCHK	715			
MEMCHK	716			
MOD10	120			
MOD10	121			
MP2-10	135	144	1252	
MP2-10	136	145	1253	
NFRENT	502			
NFRENT	503			
NFRNC	673			
NFRNC	674			
NFRPR	1065	1402		
NFRPR	1066	1403		
NFRSIG	403			
NFRSIG	404			

NFRST+	1264							
NFRST-	1265							
NFRX	146	767						
NFRX	147	770						
NFRXY	131							
NFRXY	132							
NWGOOS	342							
NWGOOS	343							
PI/2	1103							
PI/2	1104							
PSESTP	1001							
PSESTP	1002							
ROMCHK	713							
ROMCHK	714							
RSTKB	717							
RSTKB	720							
RSTMS1	1547	1565						
RSTMS1	1550	1566						
RTOD	1017							
RTOD	1020							
R*SUB	476	1057						
R*SUB	477	1060						
SD	662							
SD	663							
SEPMY	1250	1260	1301					
SEPMY	1251	1261	1302					
SIGMA	1155							
SIGMA	1156							
SQR10	1231							
SQR10	1232							
STOPSB	1025							
STOPSB	1026							
STOST0	777	1475						
STOST0	1000	1476						
TGSHF1	1511							
TGSHF1	1512							
TOOCT	1461							
TOOCT	1462							
TOPOL	702							
TOPOL	703							
TORC	740							
TORC	741							
TRG100	1214							
TRG100	1215							
TRGSET	175	230	252	700	734	1174	1202	1210
TRGSET	176	231	253	701	735	1175	1203	1211
XARCL	214							
XARCL	215							
XASHF	222							
XASHF	223							
XASN	236							
XASN	237							
XASTO	244							
XASTO	245							
XAVIEW	262							
XAVIEW	263							
XBAR	671							
XBAR	672							
XBEEP	275							
XBEEP	276							

XBST	303		
XBST	304		
XCAT	310		
XCAT	311		
XCF	314	553	610
XCF	315	554	611
XCLSIG	363		
XCLSIG	364		
XCOPY	411		
XCOPY	412		
XDEG	424		
XDEG	425		
XDELET	450		
XDELET	451		
XDSE	455		
XDSE	456		
XFS?	602		
XFS?	603		
XFT100	525		
XFT100	526		
XGA00	206		
XGA00	207		
XGI	1443		
XGI	1444		
XGRAD	432		
XGRAD	433		
XISG	637		
XISG	640		
XLN1+X	1041		
XLN1+X	1042		
XPACK	747		
XPACK	750		
XPRMPT	1011		
XPRMPT	1012		
XR/S	1031		
XR/S	1032		
XRAD	437		
XRAD	440		
XRDN	1122		
XRDN	1123		
XRND	1127		
XRND	1130		
XRTN	1134		
XRTN	1135		
XR^	1140		
XR^	1141		
XSCI	466	562	1145
XSCI	467	563	1146
XSF	1151		
XSF	1152		
XSGREG	1167		
XSGREG	1170		
XSIGN	1467		
XSIGN	1470		
XSIZE	1222		
XSIZE	1223		
XSST	1237		
XSST	1240		
XSTYON	1243		
XSTYON	1244		

XTOHRS	63	67	76	624
XTOHRS	64	70	77	625
XTONE	1320			
XTONE	1321			
XVIEW	1326			
XVIEW	1327			
XX#0?	1334			
XX#0?	1335			
XX#Y?	1342			
XX#Y?	1343			
XX<0?	1350			
XX<0?	1351			
XX<=0A	1357			
XX<=0A	1360			
XX<=Y?	1366			
XX<=Y?	1367			
XX<Y?	1410			
XX<Y?	1411			
XX=0?	1416			
XX=0?	1417			
XX=Y?	1424			
XX=Y?	1425			
XX>0?	1432			
XX>0?	1433			
XX>Y?	1440			
XX>Y?	1441			
XY^X	53			
XY^X	54			

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

[Missing Pages From 339
Page VASM ROM ASSEMBLY for
addresses shown on this
page. (32 pages total)]

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES @12000-13777

*

4	FILE	CNSB
5	ENTRY	FCNTBL
6	ENTRY	ERRAD
7	ENTRY	PCKDUR
8	ENTRY	STFLGS
9	ENTRY	XSTYCN
10	ENTRY	FIND#1
11	ENTRY	CLF
12	ENTRY	FNDEND
13	ENTRY	CHKADR
14	ENTRY	LOAD3
15	ENTRY	XASHF
16	ENTRY	XBEEP
17	ENTRY	XCLSIG
18	ENTRY	XRAD
19	ENTRY	XGRAD
20	ENTRY	DEGDO
21	ENTRY	XDEG
22	ENTRY	XTONE
23	ENTRY	TONE7
24	ENTRY	TOHEB
25	ENTRY	GETLIN
26	ENTRY	FLGANN
27	ENTRY	XSCI
28	ENTRY	XARCL
29	ENTRY	XASTO
30	ENTRY	XDSE
31	ENTRY	XISG
32	ENTRY	XSIZE
33	ENTRY	XSGREG
34	ENTRY	XR*
35	ENTRY	XRDN
36	ENTRY	RDN SUB
37	ENTRY	R* SUB
38	ENTRY	XX=0?
39	ENTRY	XX#0?
40	ENTRY	XX<0?
41	ENTRY	XX>0?
42	ENTRY	XX<=0?
43	ENTRY	XX<=0A
44	ENTRY	XX=Y?
45	ENTRY	XX#Y?
46	ENTRY	XX>Y?
47	ENTRY	XX<Y?
48	ENTRY	XX<=Y?
49	ENTRY	XCF
50	ENTRY	XSF
51	ENTRY	XFS?
52	ENTRY	Y-X
53	ENTRY	SEPMY
54	ENTRY	CHK#S
55	ENTRY	CHK#S2
56	ENTRY	SUNCHK

SET MSGFLG & DATAENTRY FLAG

```

57          ENTRY  SUNCK2

*
59          FCNTBL                                MAIN FUNCTION TABLE
60          0          0 XDEF  CAT                00
61          1          0 XDEF  GTOL              01
62          2          0 XDEF  DEL                02
63          3          0 XDEF  COPY              03
64          4          0 XDEF  CLP                04
65          5          0 XDEF  K/S              05
66          6          0 XDEF  SIZE              06
67          7          0 XDEF  BST                07
68          10         0 XDEF  SST                08
69          11         0 XDEF  STAYON            09
70          12         0 XDEF  PACK             0A
71          13         0 XDEF  DELETE           0B
72          14         0 XDEF  MODE             0C
73          15         1740 CON  @1740
74          16         0 XDEF  SHIFT            0E
75          17         0 XDEF  ASK              0F
76          20         1754 CON  @1754          DIGIT ENTRY (13)
77          21 XSTYON  1670 C=REGN 14          SET THE STAYON FLAG
78          22         1074 RCR  2
79          23         1530 ST=C
80          24         10 S3= 1                SET STAYON
81          25         1630 C=ST
82          26         1574 RCR  12
83          27         1650 REGN=C 14
84          30         1740 RTN
85          31 GETLIN  116 C=0
86          32         1160 DADD=C
87          33         1770 C=REGN 15
88          34         1740 RTN
89          35         FILLTO @34
90          35         0 XDEF  AGTO
91          36         0 XDEF  AXEQ
92          37         1740 CON  @1740
93          40         1777 CON  @1777          ROWS 2 AND 3

*****
*THIS ROUTINE DOES SUBTRACT FOR COMPARES.
*****
97          41 Y-X    1240 SETDEC
98          42         260 C=N
99          43         1276 C=-C-1 S
100         44         0 NOP
101         45         1 GOSUB  AD2-10
102         46         0
103         47         160 N=C
104         50         1740 RTN
105         53         ENTRY  OVFL10

*
* CHECK OVERFLOW/UNDERFLOW
* C HAS THE NUMBER
* RETURN WITH :
*          PT= 12  OK
*          PT= 10  OVERFLOW
*          PT= 11  UNDERFLOW
* RETURN IN DEC MODE
113         51 OVFL10 1372 ? C#0  M
114         52         27 GOC   OVFL15 ( 54)
115         53         116 C=0  W

```

116	54	OVFL15	1240	SETDEC	
117	55		1534	PT=	12
118	56		1366	? C#0	XS
119	57		1640	R1N NC	
120	60		1146	C=C-1	X
121	61		1066	C=C+1	XS
122	62		1166	C=C-1	XS
123	63		33	GONC	OVFL30 (66)
124	64		1046	C=C+1	X
125				LEGAL	
126	65	OVFL20	1740	R1N	
127	66	OVFL30	766	C=C+C	XS
128	67		33	GONC	OVFL40 (72)
129	70		116	C=0	W
130	71		53	GOTO	OVFL50 (76)
131	72	OVFL40	112	C=0	WPT
132	73		1152	C=C-1	WPT
133	74		126	C=0	XS
134	75		1724	DEC PT	
135	76	OVFL50	1724	DEC PT	
136	77		1740	R1N	
137				FILLTO 077	
138	100		0	XDEF	+
139	101		0	XDEF	-
140	102		0	XDEF	(*)
141	103		0	XDEF	/
142	104		0	XDEF	X<Y?
143	105		0	XDEF	X>Y?
144	106		0	XDEF	X<=Y?
145	107		0	XDEF	SIGMA+
146	110		0	XDEF	SIGMA-
147	111		0	XDEF	HMS+
148	112		0	XDEF	HMS-
149	113		0	XDEF	MOD
150	114		0	XDEF	PCT
151	115		0	XDEF	PCTCH
152	116		0	XDEF	P-R
153	117		0	XDEF	R-P
154	120		0	XDEF	LN
155	121		0	XDEF	X^2
156	122		0	XDEF	SQRT
157	123		0	XDEF	Y*X
158	124		0	XDEF	CHS
159	125		0	XDEF	E*X
160	126		0	XDEF	/
161	127		0	XDEF	(10)^X
162	130		0	XDEF	E*X-1
163	131		0	XDEF	SIN
164	132		0	XDEF	COS
165	133		0	XDEF	TAN
166	134		0	XDEF	ASIN
167	135		0	XDEF	ACOS
168	136		0	XDEF	ATAN
169	137		0	XDEF	-DEC
170	140		0	XDEF	ONE/X
171	141		0	XDEF	ABS
172	142		0	XDEF	FACT
173	143		0	XDEF	X#0?
174	144		0	XDEF	X>0?
175	145		0	XDEF	LN1+X

ROW 4

ROW 5

ROW 6

176	146	0 XDEF	X<0?	
177	147	0 XDEF	X=0?	
178	150	0 XDEF	INT	
179	151	0 XDEF	FRAC	
180	152	0 XDEF	D-R	
181	153	0 XDEF	R-D	
182	154	0 XDEF	H-HMS	
183	155	0 XDEF	HMS-H	
184	156	0 XDEF	RND	
185	157	0 XDEF	-OCT	
186	160	0 XDEF	CLSIG	ROW 7
187	161	0 XDEF	X<>Y	
188	162	0 XDEF	PI	
189	163	0 XDEF	CLST	
190	164	0 XDEF	R^	
191	165	0 XDEF	RDN	
192	166	0 XDEF	LASTX	
193	167	0 XDEF	CLX	
194	170	0 XDEF	X=Y?	
195	171	0 XDEF	X#Y?	
196	172	0 XDEF	SIGN	
197	173	0 XDEF	X<=0?	
198	174	0 XDEF	MEAN	
199	175	0 XDEF	STDEV	
200	176	0 XDEF	AVIEW	
201	177	0 XDEF	CLDSP	
202	200	0 XDEF	DEG	ROW 8
203	201	0 XDEF	RAD	
204	202	0 XDEF	GRAD	
205	203	0 XDEF	ENTER^	
206	204	0 XDEF	STOP	
207	205	0 XDEF	RTN	
208	206	0 XDEF	BEEP	
209	207	0 XDEF	CLA	
210	210	0 XDEF	ASHF	
211	211	0 XDEF	PSE	
212	212	0 XDEF	CLREG	
213	213	0 XDEF	AOFF	
214	214	0 XDEF	ACH	
215	215	0 XDEF	OFF	
216	216	0 XDEF	PROMPT	
217	217	0 XDEF	ADVANCE	
218	220	0 XDEF	RCL	ROW 9
219	221	0 XDEF	STO	
220	222	0 XDEF	STO+	
221	223	0 XDEF	STO-	
222	224	0 XDEF	STO*	
223	225	0 XDEF	STO/	
224	226	0 XDEF	ISG	
225	227	0 XDEF	DSE	
226	230	0 XDEF	VIEW	
227	231	0 XDEF	SIGREG	
228	232	0 XDEF	ASTO	
229	233	0 XDEF	ARCL	
230	234	0 XDEF	FIX	
231	235	0 XDEF	SCI	
232	236	0 XDEF	ENG	
233	237	0 XDEF	TONE	
234	240	1747 CON	@1747	XTOM (8)
235		ENTRY	TSTMAP	

```

* TSTMAP - TEST BIT MAP
*- A SUBROUTINE USED TO ELIMINATE DUPLICATE CODE.
*- TEST BIT MAP FOR A SPECIFIED KEYCODE AND CLEAR
*- ITS CORRESPONDING BIT IF SET.
*- IN:  A[13:0]= LOGICAL KEYCODE + 1
*-      CHIP 0 SELECTED
*- OUT: CHIP 0 SELECTED
*- USES: A[13:0], C[13:0], M[13:0]
*- USES: 1 SUBROUTINE LEVEL
*
246 241 TSTMAP      1 GSBLNG TBITMA      TEST BIT MAP
246 242              0
247 243          1356 ? C#0              BIT SET?
248 244              1 GOLNC  NFRPU      NOPE
248 245              2
249 246              1 GOLONG SRBMAP      RESET BIT MAP
249 247              2
250              FILLTO @247
251 250              0 XDEF  SF
252 251              0 XDEF  CF
253 252              0 XDEF  FS?C
254 253              0 XDEF  FC?C
255 254              0 XDEF  FS?
256 255              0 XDEF  FC?
257 256              0 XDEF  XGOIND
258 257          1760 CON  @1760
259
ROW 11
*****
*THIS CODE CLEARS THE SIGMA REGISTERS
*****
263 260 XCLSIG      1 GOSUB  SUMCHK      LEGAL AND GET ADDRESS
263 261              0
264
SUMCHK RETURNS ADDRESS OF
265
LAST SIGMA REG IN C.X
266 262          534 PT=      6
267 263          16 A=0
268 264 CLR NXT  256 AC EX
269 265          1360 DATA=C
270 266          256 AC EX
271 267          1156 C=C-1
272 270          1160 DADD=C
273 271          1724 DEC PT
274 272          1624 ? PT=  0
275 273          1713 GONC  CLR NXT ( 264 )
276 274          1740 RTN
*****
*THE ROLE UP FUNCTION HAPPENS HERE.
*****
280 275 XRDN      1 GOSUB  RDNSUB
280 276              0
281 277          503 GOTO  NFRPRL ( 347 )
282              FILLTO @277
283 300              0 XDEF  END
284 301          1754 CON  @1754
285              ENTRY  FSTIN
* FSTIN - FIRST INSTRUCTION
* SETS A[0-3] TO THE ADDRESS IN MM FORMAT OF THE FIRST LOCATION
*-IN PROGRAM MEMORY MINUS 1 BYTE. (IN PACKED MEMORY THIS IS THE
*-ADDRESS OF THE FIRST INSTRUCTION.)
* USES A[0-3],AND C

```

* EXPECTS PT=3 IN AND OUT

*

```

293 302 FSTIN 116 C=0 SET A TO REGO ADDRESS.
294 303 1160 DADD=C
295 304 1570 C=REGH 13
296 305 74 RCR 3
297 306 406 A=C X
298 307 2 A=0 PT
299 310 1740 RTN

```

*

```

301 ENTRY RTJLBL

```

*

```

* RTJLBL - RIGHT JUSTIFY ALPHA OPERAND
* ON ENTRY, C HAS A NON-ZERO ALPHA STRING IN THE FORM
* "CBA0000" WHERE THE STRING IS "ABC"
* RTJLBL MOVES ZERGES TO THE LEFT SIDE: "0000CBA"
* USES THE PTR AND C ONLY.
* ON EXIT, PT=1.

```

*

```

310 311 RTJLBL 1434 PT= 1
311 312 RTJ10 1352 ? C#0 WPT
312 313 1540 RTN C
313 314 1074 RCR 2
314 315 1753 GOTO RTJ10 ( 312)
315 FILLTO @315
316 316 0 XDEF X<>
317 317 0 XDEF LBL
318 320 0 XDEF GT0 ROW 13
319 321 1756 CON @1756

```

*THIS ROUTINE CHECKS FOR CHARACTER DATA.

```

323 322 SEPXY 156 AB EX
324 323 630 C=M
325 ENTRY CHK#S1
326 324 CHK#S1 256 AC EX
327 325 1 GOSUB CHK#S
327 326 0
328 327 256 AC EX
329 330 CHK#S 1240 SETDEC
330 331 CHK#S2 1376 ? C#0 S
331 332 1640 RTN NC
332 333 1076 C=C+1 S
333 334 63 GONC ERRAD ( 342)
334 335 1176 C=C-1 S
335 336 1740 RTN
336 FILLTO @337
337 337 0000 NOP
337 340 0 XDEF XEQ ROW 14
338 341 0 CON 0 END OF MAIN FUNCTION TABLE
339 342 ERRAD 1 GOSUB ERROR
339 343 0
340 344 0 XDEF MSGAD
341 EJECT

```

 *THIS ROUTINE DOES A ROLL DOWN,

```

345 345 XR^      1 GOSUB R^SUB
345 346          0
346 347 NFRPRL   1 GOLONG NFRPR
346 350          2
347 351 RDNSUB   1 GOSUB R^SUB
347 352          0
348 353          1 GOSUB R^SUB
348 354          0
349 355 R^SUB    116 C=0
350 356          1160 DADD=C
351 357          70 C=DATA
352 360          256 AC EX
353 361          170 C=REGN 1
354 362          50 REGN=C 0
355 363          270 C=REGN 2
356 364          150 REGN=C 1
357 365          370 C=REGN 3
358 366          250 REGN=C 2
359 367          256 AC EX
360 370          350 REGN=C 3
361 371          1740 RTN
362
363

```

*
 * LOAD3 - SET REG.C = 3333333333332
 *

```

367 372 LOAD3   1240 SETDEC          SET TO DECIMAL MODE
368 373          116 C=0      W
369 374          1156 C=C-1  W      GET ALL 9'S IN C
370 375          1140 SETHEX          PUT BACK TO HEX MODE
371 376          756 C=C+C  W      C _ 3333333333332
372 377          1740 RTN
373
374          FILLTO @377

```

*
 * DCTAB - DEFAULTCODE TABLE
 * THERE ARE TWO TYPES OF ENTRIES. ENTRY TYPE IS ENCODED IN
 * BITS 8 AND 9:
 * BITS 9,8 = 00 DATA ENTRY (DIGIT ENTRY AND ALPHA ENTRY KEYS)
 * BITS 9,8 = 01 FUNCTION IN MAIN FCN TABLE
 * FOR FCN ENTRIES, THE INDEX TO THE TABLE IS ENCODED IN BITS 7-0.
 * FOR DATA ENTRY ENTRIES, BITS 7-0 CONTAIN EITHER THE ASCII
 * CHARACTER (ALPHA ENTRY), OR THE FCN CODE FOR THE DIGIT ENTRY FCN.
 * DCTAB MUST START AT @400 IN QUAD 5 (H1500=@12400)
 *

```

* LOGICAL COL 0, UNSHIFTED, NORMAL
387 400          507 CON   327          SIGMA+
388 401          561 CON   369          X<>Y
389 402          416 CON   @0416        SHIFT
390 403          603 CON   387          ENTER
391 404          501 CON   321          -
392 405          500 CON   320          +
393 406          502 CON   322          *
394 407          503 CON   323          /

* LOGICAL COL 0, SHIFTED, NORMAL
396 410          510 CON   328          SIGMA-

```

397	411	560 CON	368	CLSIGMA
398	412	416 CON	@0416	SHIFT
399	413	400 CON	256	CATALOG
400	414	570 CON	376	X=Y?
401	415	506 CON	326	X<=Y?
402	416	505 CON	325	X>Y?
403	417	547 CON	359	X=0?
* LOGICAL COL 1, UNSHIFTED, NORMAL				
405	420	540 CON	352	1/X
406	421	565 CON	373	RDN
407	422	740 CON	480	XEQ
408	423	0 CON	0	RIGHT HALF OF ENTER KEY
409	424	27 CON	23	7
410	425	24 CON	20	4
411	426	21 CON	17	1
412	427	20 CON	16	0
* LOGICAL COL 1, SHIFTED, NORMAL				
414	430	523 CON	339	Y^X
415	431	514 CON	332	%
416	432	417 CON	@417	ASN
417	433	0 CON	0	RIGHT HALF OF ENTER KEY
418	434	650 CON	424	SF
419	435	606 CON	390	BEEP
420	436	634 CON	412	FIX
421	437	562 CON	370	PI
* LOGICAL COL 2, UNSHIFTED, NORMAL				
423	440	522 CON	338	SQRT
424	441	531 CON	345	SIN
425	442	621 CON	401	STD
426	443	34 CON	28	CHS
427	444	30 CON	24	8
428	445	25 CON	21	5
429	446	22 CON	18	2
430	447	32 CON	26	DECIMAL POINT
* LOGICAL COL 2, SHIFTED, NORMAL				
432	450	521 CON	337	X^2
433	451	534 CON	348	ASIN
434	452	717 CON	463	LBL
435	453	626 CON	406	ISC
436	454	651 CON	425	CF
437	455	516 CON	334	P-R
438	456	635 CON	413	SCI
439	457	566 CON	374	LASTX
* LOGICAL COL 3, UNSHIFTED, NORMAL				
441	460	526 CON	342	LOG
442	461	532 CON	346	COS
443	462	620 CON	400	RCL
444	463	33 CON	27	EEX
445	464	31 CON	25	9
446	465	26 CON	22	6
447	466	23 CON	19	3
448	467	405 CON	@405	R/S
* LOGICAL COL 3, SHIFTED, NORMAL				
450	470	527 CON	343	10^X
451	471	535 CON	349	ACOS
452	472	720 CON	464	GTO
453	473	605 CON	389	RTN
454	474	654 CON	428	FS?
455	475	517 CON	335	R-P
456	476	636 CON	414	ENG

457	477	630 CON	408	VIEW
* LOGICAL COL 4, UNSHIFTED, NORMAL MODE				LN
459	500	520 CON	336	TAN
460	501	533 CON	347	SST
461	502	410 CON	@410	BACKARROW
462	503	0 CON	0	MODE (ALPHA)
463	504	414 CON	@414	MODE (PRGM)
464	505	414 CON	@414	MODE (USER)
465	506	414 CON	@414	OFF KEY IS SPECIAL
466	507	0 NOP		
* LOGICAL COL 4, SHIFTED, NORMAL MODE				E^X
468	510	525 CON	341	ARCTAN
469	511	536 CON	350	BST
470	512	407 CON	@407	CLX
471	513	567 CON	375	MODE (ALPHA)
472	514	414 CON	@414	MODE (PRGM)
473	515	414 CON	@414	MODE (USER)
474	516	414 CON	@414	OFF KEY IS SPECIAL
475	517	0 NOP		
* LOGICAL COL 0, UNSHIFTED, ALPHA MODE				A
478	520	101 CON	65	F
479	521	106 CON	70	SHIFT
480	522	416 CON	@0416	N
481	523	116 CON	78	Q
482	524	121 CON	81	U
483	525	125 CON	85	Y
484	526	131 CON	89	:
485	527	72 CON	58	
* LOGICAL COL 0, SHIFTED, ALPHA MODE				LOWER CASE A
487	530	141 CON	97	SIGMA
488	531	176 CON	126	SHIFT
489	532	416 CON	@0416	^
490	533	136 CON	94	-
491	534	55 CON	45	+
492	535	53 CON	43	*
493	536	52 CON	42	/
494	537	57 CON	47	
* LOGICAL COL 1, UNSHIFTED, ALPHA MODE				B
496	540	102 CON	66	G
497	541	107 CON	71	K
498	542	113 CON	75	RIGHT HALF OF ENTER KEY
499	543	0 CON	0	R
500	544	122 CON	82	V
501	545	126 CON	86	Z
502	546	132 CON	90	SPACE
503	547	40 CON	32	
* LOGICAL COL 1, SHIFTED, ALPHA MODE				LOWER CASE B
505	550	142 CON	98	%
506	551	45 CON	37	LAZY T (APPEND)
507	552	177 CON	127	RIGHT HALF OF ENTER KEY
508	553	0 CON	0	7
509	554	67 CON	55	4
510	555	64 CON	52	1
511	556	61 CON	49	0
512	557	60 CON	48	
* LOGICAL COL 2, UNSHIFTED, ALPHA MODE				C
514	560	103 CON	67	H
515	561	110 CON	72	L
516	562	114 CON	76	

517	563	117 CON	79	0
518	564	123 CON	83	S
519	565	127 CON	87	W
520	566	75 CON	61	=
521	567	54 CON	44	COMMA
* LOGICAL COL 2, SHIFTED, ALPHA MODE				
523	570	143 CON	99	LOWER CASE C
524	571	35 CON	29	NOT EQUAL SIGN
525	572	632 CON	410	ASTO
526	573	15 CON	13	ANGLE SIGN
527	574	70 CON	56	8
528	575	65 CON	53	5
529	576	62 CON	50	2
530	577	56 CON	46	DECIMAL POINT
* LOGICAL COL 3, UNSHIFTED, ALPHA MODE				
532	600	104 CON	68	D
533	601	111 CON	73	I
534	602	115 CON	77	M
535	603	120 CON	80	P
536	604	124 CON	84	T
537	605	130 CON	88	X
538	606	77 CON	63	?
539	607	405 CON	0405	R/S
* LOGICAL COL 3, SHIFTED, ALPHA MODE				
541	610	144 CON	100	LOWER CASE D
542	611	74 CON	60	<
543	612	633 CON	411	ARCL
544	613	44 CON	36	\$
545	614	71 CON	57	9
546	615	66 CON	54	6
547	616	63 CON	51	3
548	617	576 CON	382	AVIEW
* LOGICAL COL 4, UNSHIFTED, ALPHA MODE				
550	620	105 CON	69	E
551	621	112 CON	74	J
552	622	410 CON	0410	SST
553	623	0 CON	000	BACKARROW
554	624	414 CON	0414	MODE (ALPHA)
555	625	414 CON	0414	MODE (PRGM)
556	626	414 CON	0414	MODE (USER)
557	627	0 NGP		OFF KEY IS SPECIAL
* LOGICAL COL 4, SHIFTED, ALPHA MODE				
559	630	145 CON	101	LOWER CASE E
560	631	76 CON	62	>
561	632	1007 CON	519	BSY
562	633	607 CON	391	CLA
563	634	414 CON	0414	MODE (ALPHA)
564	635	414 CON	0414	MODE (PRGM)
565	636	414 CON	0414	MODE (USER)
566				OFF KEY IS SPECIAL

*ISG AND DSE ARE DONE HERE. THE ROUTINE BREAKS				
*THE INPUT IN B INTO ITS THREE PARTS - INT,				
*COMPARE, AND INC. IT ADDS THE INC, STORES THE				
*RESULT AND BRANCHES TO THE COMPARE ROUTINE				
*TO DECIDE WHETHER TO SKIP OR NOT.				

574	637 XDSE	1610 S0=	1	SET DSE FLAG
575	640 XISG	356 EC EX		GET CONTENT OF RNN
576	641	1 GOSUB	CHK#S	CHECK FOR ALPHA

```

576 642          0
577 643          33 GOTO NOFRAC ( 646 )
578 644 ELMFRC 1732 C SR M CHANGE TO FIX NOTATION
579 645          1046 C=C+1 X
580 646 NOFRAC 1366 ? C#0 XS
581 647          1757 GOC ELMFRC ( 644 )
582 650          1 GSBLNG SINFR DO INT AND FRC
582 651          0
583 652          630 C=M GET INT PART
* NEXT TWO STATES (?PT=0, GOC OVRDEC) ARE VESTIGIAL FROM WHEN
* SINFR WORKED FOR 13 DIGIT MANTISSAS IN ANOTHER MACHINE. THESE
* TWO STATES CAN BE REMOVED WITHOUT HARM.
587 653          1624 ? PT= 0 PREVNT CLEAR FOR LARGE VALUES
588 654          27 GOC OVRDEC ( 656 )
589 655          1724 DEC PT
590 656 OVRDEC 112 C=0 WPT CLEAR FRACTIONAL PART
591 657          372 BC EX M JOIN MANTISSA WITH EXP AND SIGN
592 660          356 BC EX W BRING COMPLETE INT TO C
593 661          530 M=C SAVE FOR LATER
594 662          256 C=A PUT FRAC IN C
594 663          416
595 664          1360 DATA=C
596 665          1234 PT= 7 SET TO CLR TRAIL DIGITS
597 666          112 C=0 WPT
598 667          1134 PT= 9 PICK OFF INC
599 670          1352 ? C#0 WPT DUMMY ONE FOR ZERO
600 671          47 GOC SEPA ( 675 )
601 672          434 PT= 8
602 673          120 LC 1
603 674          1134 PT= 9
604 675 SEPA 16 A=0
605 676          412 A=C WPT PICK OFF INC
606 677          112 C=0 WPT LEAVE COMPARE VAL
607 700          1756 A SL
608 701          1756 A SL
609 702          1756 A SL INCREMENT LEFT JUST
610 703          1534 PT= 12 POINT TO DIGIT ONE
611 704          1046 C=C+1 X EXP COMPARE
612 705          1046 C=C+1 X
613 706          546 A=A+1 X EXP INC
614 707          1502 ? A#0 PT IS INC EXP OK?
615 710          37 GOC TSTEXP ( 713 ) YES
616 711          1772 A SL M
617 712          6 A=0 X
618 713 TSTEXP 1342 ? C#0 PT EXP 2 TOO LARGE?
619 714          107 GOC ADDIT ( 724 ) NO, JUST RIGHT
620 715          1374 RCR 13 SHIFT COMPARE LEFT
621 716          106 C=0 X EXP=1?
622 717          1046 C=C+1 X
623 720          1342 ? C#0 PT EXP COMP OK NOW?
624 721          37 GOC ADDIT ( 724 ) YES
625 722          1374 RCR 13 SHIFT LEFT
626 723          106 C=0 X EXP MUST BE ZERO
627 724 ADDIT 1614 ?S0=1 DSE OR ISG?
628 725          23 GONC ADDEN ( 727 ) ISG
629 726          676 A=A-1 S MAKE DEC OUT OF INC
630 727 ADDEN 160 N=C SAVE FLOATING POINT COMPARE
631 730          630 C=M GET INTEGER PART BACK
632 731          1 GOSUB AD2-10
632 732          0

```

```

633 733          530 M=C          SAVE INT PART OF RESULT
634 734          416 A=C          DUP RESULT IN A
635 735          70 C=DATA        GET FRAC PART BACK
636 736          246 C=A          DUP EXP
636 737          406
637 740 MRSHT 1732 C SR M          SHIFT FRAC INTO POSITION
638 741          1146 C=C-1 X      IN POSITION YET?
639 742          37 GOC COMBIN ( 745) YES
640 743          1372 ? C#0 M      FRACTION ZERO
641 744          1747 GOC MRSHT ( 740) NO NOT YET
642 745 COMBIN 106 C=0 X          SIGN AND EXP C=0
643 746          1032 C=C+A M
644          LEGAL
645 747          1 GOSUB SHF40
645 750          0
646 751          1360 DATA=C      STORE UPDATED COUNTER
647 752          1210 S7= 1
648 753          630 C=M
649 754          256 AC EX
650 755          1614 ?S0=1      DSE??
651 756          123 GONC XX>Y? ( 770)
*****
*THE COMPARISONS FOLLOW. X VALUES ENTER IN N
*WHILE Y VALUES ARE IN A.
*****
656 757 XX<Y? 1 GOSUB Y-X DO SUBTRACT
656 760          0
657 761 XX>0? 1240 SETDEC
658 762          260 C=N
659 763          1356 ? C#0
660 764          723 GONC SKP (1056)
661 765          1076 C=C+1 S
662 766          433 GONC NOSKP (1031)
663 767          673 GOTO SKP (1056)
664 770 XX>Y? 1 GOSUB Y-X
664 771          0
665 772 XX<0? 1214 ?S7=1
666 773          227 GOC XX<=0? (1015)
667 774 XX<0 1240 SETDEC
668 775          260 C=N
669 776          1076 C=C+1 S
670 777          573 GONC SKP (1056)
671 1000          313 GOTO NOSKP (1031)
672 1001 XX<=Y? 1 GOSUB Y-X
672 1002          0
673 1003          1356 ? C#0
674 1004          253 GONC NOSKP (1031)
675 1005          1543 GOTO XX>0? ( 761)
676 1006 XX=0? 16 A=0
677 1007          260 C=N
678 1010          173 GOTO XYZ (1027)
679 1011 XX<=0A 370 C=REGN 3 ROW LOGIC DOESN'T CHECK
680 1012          1 GOSUB CHK#S FOR ALPHA DATA ON X<=0?
680 1013          0
681 1014          160 N=C
682 1015 XX<=0? 260 C=N
683 1016          1356 ? C#0
684 1017          123 GONC NOSKP (1031)
685 1020          1543 GOTO XX<0 ( 774)
686 1021 XX#0? 16 A=0

```

```

687 1022          260 C=N
688 1023          313 GOTO   XYN      (1054)
689 1024 XX=Y?    270 C=REGN 2
690 1025          256 AC EX
691 1026          370 C=REGN 3
692 1027 XYY      1556 ? A#C
693 1030          267 GOC     SKP      (1056)
694          ENTRY  NOSKP
695 1031 NOSKP    1314 ?S13=1
696 1032          157 GOC     NOSKPO (1047)
697 1033          116 C=0
698 1034          1160 DADD=C
699 1035          1670 C=REGN 14
700 1036          1730 CST EX
701 1037          114 ?S4=1
702 1040          77 GOC     NOSKPO (1047)
703 1041          1730 CST EX
704 1042          1214 ?S7=1          IS THIS ISG OR DSG
705 1043          47 GOC     NOSKPO (1047)
706 1044          1 GOSUB   MSG
706 1045          0
707 1046          0 XDEF    MSGYES
708 1047 NOSKPO   1 GOLONG  NFRPU          CAN'T DO A RTN HERE
709 1050          2
* BECAUSE SOME COMPARISONS HAVE NFRX ON THE STACK INSTEAD OF NFRPU
710 1051 XX#Y?    270 C=REGN 2
711 1052          256 AC EX
712 1053          370 C=REGN 3
713 1054 XYN      1556 ? A#C
714 1055          1547 GOC    NOSKP   (1031)
715          ENTRY  SKP
716 1056 SKP      1140 SETHEX
717 1057          1314 ?S13=1
718 1060          103 GONC    SST?    (1070)
719 1061 DOSKP    1 GOSUB   GETPC
719 1062          0
720          ENTRY  DOSKP
721 1063          1 GOSUB   SKPLIN
721 1064          0
722 1065          1 GOSUB   PUTFCX          FORCE RECALC OF LINE NUMBER
722 1066          0
723 1067          1603 GOTO   NOSKPO (1047)
724 1070 SST?     116 C=0
725 1071          1160 DADD=C
726 1072          1670 C=REGN 14
727 1073          1730 CST EX
728 1074          114 ?S4=1
729 1075          1647 GOC    DOSKP   (1061)
730 1076          1730 CST EX
731 1077          1214 ?S7=1          ISG DSE?
732 1100          1477 GOC    NOSKPO (1047)
733 1101          1 GOSUB   MSG          NO COMP OR FLAGS
733 1102          0
734 1103          0 XDEF    MSGNO
735 1104          1433 GOTO   NOSKPO (1047)

```

```

*****
*THE FLAG CONDITIONALS FOLLOW. ENTRY IS WITH R14 IN A AND A
*MASK IN B. THE MASK CONSISTS OF ALL ZEROS EXCEPT FOR A
*ONE AT THE LOCATION OF THE SELECTED FLAG
*****

```

```

741 1105 XFS?      316 C=B                      GET MASK
742 1106          1660 C=C.A                     AND R14 WITH MASK
743 1107          1356 ? C#0                     IS ANYTHING LEFT
744 1110          1217 GOC      NOSKP (1031) YES NO SKP
745 1111 SKPIT     1453 GOTO      SKP (1056)
746 1112 XSF       356 BC EX                      MOVE MASK TO C
747 1113          1560 C-CORA                     SET MASKED BIT
748 1114          53 GOTO      FLGANN (1121)
749 1115 XCF       316 C=B
750 1116          1256 C=-C-1
751 1117          0 NOP
752 1120          1660 C=C.A
753 1121 FLGANN    1650 REGN=C 14
754 1122          256 AC EX
755 1123          530 M=C
756 1124          1 GOSUB      ANNOUT
756 1125          0
757 1126          630 C=M
758 1127          256 AC EX
759 1130          1740 RTN
*****
*SUM+NN SETS THE ADDRESS OF THE REGISTER USED
*FOR SIGMA PLUS. IT CHECKS TO SEE IF THE ADDRESS
*IS IN FACT A LEGAL ADDRESS. THE SUBROUTINE
*SUMCHK IS CALLED BY THE SIGMA PLUS FUNCTION
*FOR THIS CHECK.
*****
767 1131 XSGREG    260 C=N                      ADDRESS OR SUM 1
768 1132          1 GOSUB      SUMCK2             LEGAL?
768 1133          0
769 1134          260 C=N                      YES
770 1135          256 CA EX                      PUT ADDRESS IN SCRATCH
771 1136          116 C=0
772 1137          1160 DADD=C
773 1140          1570 C=REGN 13
774 1141          674 RCR      11
775 1142          106 C=0      X
776 1143          1006 C=C+A  X
777 1144          74 RCR      3
778 1145          1550 REGN=C 13
779 1146          1740 RTN
780 1147 SUMCHK    1570 C=REGN 13             GET ADDRESS
781 1150          674 RCR      11
782 1151 SUMCK2    256 CA EX                      ADD5
783 1152          460 LDI
784 1153          5 CON      5
785 1154          1140 SETHEX
786 1155          1006 C=C+A  X
787          LEGAL
*****
*CHKADR-CHECKS FOR VALID DATA ADDRESSES.
* IN: ADDR IN C,X
* OUT: DATA IN B, ADDR IN C,X, HEXMODE,
* DADD=C,X (EXCEPT SOME ERROR EXITS)
* USES: ACTIVE POINTER, A, S9, DADD, C, B
* MAY EXIT TO ERRNE
*
* CHKAD4 - CHECKS TO SEE IF THE REGISTER IS THERE.
* ON ENTRY, DADD=B=ADDRESS OF REGISTER AND C=CONTENTS OF REG
* STATUS OF S9 ON ENTRY CONTROLS EXIT IF THE REGISTER ISN'T

```


* THERE: EXITS TO ERRNE IF S9=1 ELSE GOES TO COLD START!
 * ON EXIT, ADDRESS OF REGISTER IS IN C,X AND CONTENTS ARE IN B
 * AND HEXMODE AND USES ACTIVE POINTER AND A
 * NOTE - CHKAD4 IS PROBABLY OBSOLETE NOW. IT USED TO BE CALLED BY
 * MEMCHK IN CNO. DRC 10/20/79

```

805 1156 CHKADR      1 GOLONG PATCH6
805 1157              2
806                  ENTRY  P6RTN
807 1160 P6RTN      356 BC EX          SAVE ADDRESS
808 1161              70 C=DATA        GET CURRENT CONTENT
809 1162              1240 SETDEC
810 1163              1056 C=C+1  W      LOGIC IN HERE ASSURES
811 1164              1156 C=C-1  W      DATA IS IN A CANONICAL
812                  FORM
813 1165              1376 ? C#0  S      NON-POSITIVE?
814 1166              103 GONC  CKAD3  (1176) POSITIVE NUMBER
815 1167              1076 C=C+1  S      NEGATIVE NUMBER?
816 1170              57 GOC   CKAD2  (1175) YES
817 1171              70 C=DATA        ASSUME AN ALPHA STRING
818 1172              136 C=0   S      ASSURE A 1 IN DIGIT 13
819 1173              1076 C=C+1  S
820                  LEGAL
821 1174              123 GOTO  CKAD4  (1206)
822 1175 CKAD2      1176 C=C-1  S      RESTORE 9 IN SIGN DIGIT
823 1176 CKAD3      1366 ? C#0  XS      NEGATIVE EXPONENT?
824 1177              33 GONC   CKAD3J (1202) NO.
825 1200              126 C=0   XS      ASSURE EXP SIGN = 9
826 1201              1166 C=C-1  XS
827 1202 CKAD3J     1534 PT=      12
828 1203              1342 ? C#0  PT      IS MANTISSA NORMALIZED?
829 1204              27 GOC   CKAD4  (1206) YES
830 1205              116 C=0   W      FORCE WHOLE WORD TO ZERO
831                  ENTRY  CHKAD4
832                  CHKAD4
833 1206 CKAD4      1140 SETHX
834 1207              356 BC EX
835 1210              416 A=C          GET ADR BACK
836 1211              1360 DATA=C    WRITE ADR OUT
837 1212              70 C=DATA        BRING ADR BACK IN
838 1213              1556 ? A#C      GET ADR BACK?
839 1214              63 GONC   CKAD10 (1222) YES
840                  REG ISN'T THERE
841 1215              1114 ?S9=1
842 1216              1 GOLC   ERRNE
843 1217              3
844 1220              1 GOLONG COLDST
845 1221              2
846                  CKAD10
847 1222              316 C=B          PUT DATA BACK
848 1223              1360 DATA=C
849 1224              256 AC EX        ADR BACK TO CX
850 1225              1740 RTN

```

```

850 1226 XARCL      316 C=B
851 1227              1176 C=C-1  S
852 1230              1376 ? C#0  S      NUMERIC DATA?
853 1231              273 GONC   REGALP (1260) NO, ALPHA DATA
854 1232              1 GOSUB  AFORNT    NUMERIC DATA
855 1233              0

```

```

855 1234 ARCL10 1314 ?S13=1          RUNNING?
856 1235          1540 RTN C
857 1236          1670 C=REGN 14
858 1237          1530 ST=C
859 1240          114 ?S4=1          SSTFLG?
860 1241          1540 RTN C
861 1242          1214 ?S7=1          ALPHA MODE?
862 1243          1640 RTN NC        NO
863 1244          410 S8= 1          SAY PROMPT & NO SCROLL
864 1245          1 GOSUB ARGOUT
864 1246          0

*
* STFLGS - SET MSGFLG & DATAENTRY FLAG
* ASSUMES CHIP 0 ENABLED. LEAVES SS 1/2 UP AND REG 14 IN C
*
869 1247 STFLGS 1670 C=REGN 14
870 1250          1474 RCR 1
871 1251          1530 ST=C
872 1252          1410 S1= 1          SET MSGFLG
873 1253          510 S6= 1          SET DATAENTRY FLAG
874 1254          1630 C=ST
875 1255          1374 RCR 13
876 1256 STFL10 1650 REGN=C 14
877 1257          1740 RTN

*
879 1260 REGALP 106 C=0 X          RE-ENABLE CHIP 0
880 1261          1160 DADD=C
881 1262          1534 PT= 12          RELIES ON P ACTIVE
882 1263 ARCL20 240 SEL P
883 1264          1724 DEC PT
884 1265          1724 DEC PT          MOVE P RIGHT 1 BYTE
885 1266          1524 ? PT= 12      WRAPAROUND?
886 1267          1457 GOC ARCL10 (1234) YES, DONE
887 1270          316 C=B          PUT CHAR TO G FOR APNDNW
888 1271          130 G=C
889 1272          340 SEL Q
890 1273          634 PT= 11
891 1274          1322 ? B#0 PQ      ANY CHARS FOUND YET?
892 1275          1 GSUBC APNDNW     APPEND TO ALPHA REG
892 1276          1
* APNDNW CLOBBERS A, C, AND THE ACTIVE POINTER, WHICH IS Q HERE.
894 1277          1643 GOTO ARCL20 (1263)
*****
*THIS ROUTINE SETS THE INTERNAL DISPLAY FORMAT
*STATUS.
*****
899 1300 XSCI 1630 C=ST          FE00NNNN
900 1301          1074 RCR 2          XXXXXXXX,FE00NNNN
901 1302          1630 C=ST          FE00NNNN,FE00NNNN
902 1303          1374 RCR 13          FE00NNNNFE00,NNNN
903 1304          1530 ST=C          SAVE NNNNFE00
904 1305          1670 C=REGN 14      GET STATUS
905 1306          74 RCR 3          MOVE DSP TO POSITION
906 1307          1730 CST EX        MOVE GRAD RAD TO STATUS
907 1310          1614 ?S0=1          IS GRAD SET
908 1311          23 GONC S1? (1313)HEN RAD
909 1312          1056 C=C+1          SET LOW BIT
910 1313 S1? 1414 ?S1=1          RAD?
911 1314          33 GONC DSPDN (1317)
912 1315          1056 C=C+1          SET BIT TWO

```

```

913 1316          1056 C=C+1
914 1317 DSPDN    674 RCR      11
915 1320          1363 GOTO    STFL10 (1256)
*****
*XBEEP-COCOONUT BEEP
*SETS UP STATUS WITH A TONE NUMBER THEN CALLS TONE AS IF
*FROM KEYBOARD.
*****
921          XBEEP
922 1321          1 GOSUB  TONEB
922 1322          0
923 1323          460 LDI                      LOAD A 5
924 1324          5 CON      5
925 1325          1 GOSUB  TONEB
925 1326          0
926 1327          460 LDI                      LOAD A 8
927 1330          10 CON      8
928 1331          1 GOSUB  TONEB
928 1332          0
929          ENTRY  TONE7X
930 1333 TONE7X  460 LDI
931 1334          7 CON      7
932 1335 TONEB  1530 ST=C                      FALL INTO TONE
*****
*XTONE-EXECUTE 41-C TONE FUNCTION
*
* ONE DIGIT(0-9) OPERAND FUNCTION, 0=LOW, 9=HIGH. TONES ARE NOT
* MUSICAL NOTES. THE FREQUENCY DIFFERENCE ARE RATHER ARBITRARY.
*   TONE N      WORD TIMES/CYCLE
*   9          3
*   8          4
*   7          5
*   6          6
*   5          8
*   4          10
*   3          12
*   2          14
*   1          16
*   0          18
* THE DURATION OF EACH TONE IS EQUALLY .25 SECONDS
* IF THE AUDIO ENABLE FLAG IS NOT SET, SILENT RETURN.
*
* INPUT:  1. S[0:7] = OPERAND (0-9)
*         2. CHIP 0 ENABLE
*
* USES : A, C, S[0:7], FO[0:7], NO PT.  + 1 SUB LEVEL
*
* OUTPUT: 1. HEXMODE
*         2. FO[0:7] = 0
*         3. A.X = FFF
*         4. CHIP 0 ENABLE
* SPECIAL ENTRIES :
*
* TONE7X - GENERATES TONE 7
* SAME AS XTONE EXCEPT NO OPERAND IS REQUIRED.
*
* TONEB - SAME AS XTONE EXCEPT THE OPERAND IS IN C[0:1]
*
*****
969 1336 XTONE  1670 C-REGN 14                IS BEEP ENABLED

```

970	1337	1274	ROR	7	
971	1340	1730	CST EX		
972	1341	1414	?SI=1		
973	1342	1640	RTN NC		NO NOT ENABLED
974	1343	1704	CLR ST		CLEAR FLAG OUT REG
975	1344	1330	FEXSB		
976	1345	1530	ST=C		SAVE FREQUENCY INPUT IN ST
977	1346	116	C=0		
978	1347	460	LDI		PUT ALL 1S IN ST
979	1350	377	CON2	15	15
980	1351	1730	CST EX		& GET COUNT BACK
981	1352	674	ROR	11	PLACE COUNT FOR LOOK UP
982	1353	416	A=C		
983	1354	174	ROR	4	SHIFT FREQ INTO DIGIT 13
984	1355	1240	SETDEC		
985	1356	1276	C=-C-1	S	
986	1357	1140	SETHX		
987	1360	1	GOSUB	PCKDUR	PICK .25 S DURATION
987	1361	0			
988	1362	133	CON	@133	CONSTANTS MUST BE ODD
989	1363	145	CON	@145	
990	1364	163	CON	@163	
991	1365	207	CON	@207	
992	1366	241	CON	@241	
993	1367	311	CON	@311	
994	1370	413	CON	@413	
995	1371	477	CON	@477	
996	1372	617	CON	@617	
997	1373	1025	CON	@1025	
998	1374	PCKDUR	660	C=STK	
999	1375	1032	C=C+A	M	
1000	1376	1460	CXISA		
1001	1377	246	AC EX	X	DURATION NOW IN A X TEST FOR TONE 7,8,9
1002					
1003	1400	1176	C=C-1	S	
1004	1401	147	GOC	TONE9 (1415)	
1005	1402	1176	C=C-1	S	
1006	1403	167	GOC	TONE8 (1421)	
1007	1404	1176	C=C-1	S	
1008	1405	217	GOC	TONE7 (1426)	
1009	1406	DELOOP	436	A=C	GET FREQ CNTR
1010	1407	1330	FEXSB		TURN ON TONE
1011	1410	676	A=A-1	S	FREQ COUNT
1012	1411	1773	GONC	*-1 (1410)	
1013	1412	646	A=A-1	X	COUNT DOWN DURTATION
1014	1413	1733	GONC	DELOOP (1406)	
1015	1414	1740	RTN		
*					
1017	1415	TONE9	1330	FEXSB	
1018	1416		646	A=A-1	X
1019	1417		1763	GONC	*-2 (1415)
1020	1420		1740	RTN	
1021	1421	TONES	1330	FEXSB	
1022	1422		0	NOP	
1023	1423		646	A=A-1	X
1024	1424		1753	GONC	*-3 (1421)
1025	1425		1740	RTN	
1026	1426	TONE7	1330	FEXSB	
1027	1427		0	NOP	
1028	1430		0	NOP	

```

1029 1431          646 A=A-1  X
1030 1432          1743 GONC  *-4   (1426)
1031 1433          1740 RTN
*****
*THIS ROUTINE SETS DEGREES RADIAN OR GRAD.
*****
1035 1434 XDEG      1 GOSUB  DEGDO
1035 1435          0
1036 1436 XDEG2    1630 C=ST
1037 1437          674 RCR      11
1038 1440          1 GOLONG ANN+14
1038 1441          2
1039 1442 XRAD      1 GOSUB  DEGDO
1039 1443          0
1040 1444          1610 S0=     1
1041 1445          1713 GOTO   XDEG2 (1436)
1042 1446 XGRAD     1 GOSUB  DEGDO
1042 1447          0
1043 1450          1410 S1=     1
1044 1451          1653 GOTO   XDEG2 (1436)
1045 1452 DEGDO     1670 C=REGN 14
1046 1453          74 RCR      3
1047 1454          1530 ST=C
1048 1455          1604 S0=     0
1049 1456          1404 S1=     0
1050 1457          1740 RTN
*****
*FNDEND-FIND THE HIGH END OF RAM
* ROUTINE STARTS AT REGO STORING AND RETRIEVING
* THE REGISTERS ADDRESS UNTIL THE
* RETRIVED VALUE DOEN NOT MATCH THE STORED VALUE.
* UPON RETURN A(0:2) CONTAINS THE ADDRESS OF THE
* FIRST NON-EXISTANT REGISTER.
*IF FLAG 8 IS SET AND ENTRY IS AT CLEAR THE ROUTINE
* ALSO CLEARS ALL DATA REGISTERS.
* NOTE: FNDEND RELIES ON THE TESTED REGISTER BEING DIFFERENT FROM
* WHAT YOU GET WHEN YOU READ A NONEXISTENT REGISTER - PROBABLY AN
* OK ASSUMPTION IF NONEXISTENT REGISTERS GIVE ALL ZERGES OR ALL ONES
*****
1064 1460 FNDEND    404 S8=     0          CLEAR CLEAR FLAG
1065 1461          116 C=0          ADDRESS SHIP ZERO
1066 1462          1160 DADD=C
1067 1463 CLR       1570 C=REGN 13      GET REG 0
1068 1464          74 RCR      3
1069 1465          132 C=0    M      MUST HAVE ZEROS IN TEST
1070 1466 CLEN      1160 DADD=C      ADDRESS REGISTER
1071 1467          256 AC EX      SAVE ADR
1072 1470          70 C=DATA      SAVE VALUE
1073 1471          356 BC EX
1074 1472          256 C=A          DUP ADR
1074 1473          416
1075 1474          1360 DATA=C      SEND ADR OUT
1076 1475          70 C=DATA      BRING IT BACK
1077 1476          1556 ? A#C      HAS IT CHANGED?
1078 1477          1540 RTN C      YES SO RTN
1079 1500          356 BC EX      PUT ORIGNAL VAL BACK
1080 1501          414 ?S8=1      CLEAR REGISTER?
1081 1502          23 GONC   OVR0   (1504) NO
1082 1503          116 C=0
1083 1504 OVR0      1360 DATA=C      PUT VAL BACK

```

```

1084 1505          556 A=A+1          INC ADR
1085 1506          256 AC EX          GET ADR
1086 1507          1573 GOTO CLEM (1466)
*****
*THIS ROUTINE      KILLS THE FIRST SIX CHARACTERS
*IN THE ALPHA REGISTER.
*****
1091 1510 XASHF      1 GOSUB FIND#1    FIND FIRST CHARACTER
1091 1511          0
1092 1512          1524 ? PT= 12      7 IN THIS REG
1093 1513          53 GONC REGG (1520) NO REGULAR 6 OR LESS
1094 1514          1034 PT= 2        CLEAR TOP SIX AND DONE
1095 1515 INSHFT    122 C=0          PQ
1096 1516 DONSHF    1360 DATA=C
1097 1517          1740 RTN
1098 1520 REGG      116 C=0
1099 1521          1360 DATA=C      CLEAR FIRST REG
1100 1522          324 ? PT= 10      DONE SIX
1101 1523          1540 RTN C
1102 1524          256 AC EX
1103 1525          1160 DADD=C      ADDRESS NEXT REG
1104 1526          70 C=DATA
1105 1527          1734 INC PT      CLEAR REMAINING CHARACTERS
1106 1530          1734 INC PT
1107 1531          1734 INC PT
1108 1532          1734 INC PT
1109 1533          1623 GOTO INSHFT (1515)
*****
*THIS FUNCTION TAKES THE FIRST SIX NON NULLS
*IN THE ALPHA REGISTER AND STORES THEM.
*****
1114 1534 XASTO      1 GOSUB FIND#1
1114 1535          0
1115 1536          1524 ? PT= 12      ALL IN THIS REG?
1116 1537          113 GONC REG (1550) NO
1117 1540          1074 RCR 2        SHIFT RIGHT 2
1118 1541 DONSTO    1334 PT= 13
1119 1542          120 LC 1        SET ONE IN 13
1120 1543          20 LC 0        AND CLEAR 12
1121 1544          360 NC EX      GET DATA ADDRESS BACK
1122 1545          1160 DADD=C
1123 1546          360 NC EX      GET REGISTER CONTENT BACK
1124 1547          1473 GOTO DONSHF (1516)
1125 1550 REG      324 ? PT= 10      ALL IN THIS REG
1126 1551          1707 GOC DONSTO (1541) DONE
1127 1552          256 AC EX      GET ADR OF NEXT REG
1128 1553          1160 DADD=C
1129 1554          70 C=DATA      GET NEXT REG
1130 1555          1734 INC PT
1131 1556          252 AC EX WPT    COMBINE TWO REG
1132 1557 SHFLFT    1734 INC PT    SHIFT THE
1133 1560          1734 INC PT
1134 1561          1574 RCR 12
1135 1562          624 ? PT= 11      IS THE POINTER IN POSITION
1136 1563          1743 GONC SHFLFT (1557)
1137 1564          1553 GOTO DONSTO (1541)
*****
*THIS SUBROUTINE FINDS THE FIRST NON NULL IN
*THE ALPHA REGISTER.
*****

```

```

*****
1143 1565 FIND#1 460 LDI LOAD ADDRESS
1144 1566 10 CON 010
1145 1567 1160 DADD=C
1146 1570 256 CA EX PUT ADDRESS IN A
1147 1571 340 SEL Q SET Q AT 13 FOR TEST AND CLEARS
1148 1572 1334 PT= 13
1149 1573 240 SEL P SET P AT SIX TO CLEAR GARB
1150 1574 534 PT= 6
1151 1575 70 C=DATA
1152 1576 122 C=0 PQ
1153 1577 34 PT= 3
1154 1600 THSIT? 656 A=A-1 DEC ADDRESS
1155 1601 1356 ? C#0 ANYTHING IN THIS REG
1156 1602 157 GOC FOUND1 (1617) IF YES THEN GUT
1157 1603 256 C=A OTHERWISE GET NEXT REG
1158 1604 416
1159 1605 1160 DADD=C
1160 1606 70 C=DATA
1161 1607 1724 DEC PT COUNT DOWN ON LOOP
1162 1610 1624 ? PT= 0 DONE YET
1163 1611 1673 GONC THSIT? (1600)
1164 1612 1534 PT= 12
1165 1613 1362 ? C#0 PQ IF 7 CHARACTERS IN LAST RTN
1166 1614 1540 RTN C OTHERWISE DONE WITH 10
1167 1615 334 PT= 10
1168 1616 1740 RTN
1169 1617 FOUND1 1634 PT= 0
1170 1620 FNDPT 1724 DEC PT
1171 1621 1724 DEC PT
1172 1622 1362 ? C#0 PQ
1173 1623 1753 GONC FNDPT (1620)
1174 1624 1740 RTN RTN WITH POINTER AT FIRST BYTE
*****
*THE SIZE FUNCTION PLACES R0 SUCH THAT COCONUT CONTAINS
*THE CORRECT NUMBER OF REGISTERS.
*THE NUMBER OF REGISTERS IS DELIVERED IN HEX IN AX
* ENTRY SIZSUB USES S9 TO TELL WHETHER TO GO TO PACKE OR RETURN
* IF THERE ISN'T ENOUGH ROOM - S9=1 GOES TO PACKE, S9=0 RETURNS
* WITH S9=1 IF NOT SUCCESSFUL.
*****
1182 1625 XSIZE 1110 S9= 1 EXIT VIA PACKE IF
1183 UNSUCCESSFUL
1184 ENTRY SIZSUB
1185 1626 246 AC EX X GET USER SPEC NUM OF REGS
1186 1627 SIZSUB 160 N=C N CONTAINS THE NUMBER NEEDED
1187 1630 1 GOSUB MEMLFT CALCULATE THE MEM UNUSED
1188 1631 0
1189 1632 530 M=C M=UNUSED REGISTERS
1190 1633 1 GOSUB FNDEND FIND THE END OF MEM
1191 1634 0
*****
*MORE OR LESS REGISTERS THAN WE HAVE NOW?
*****
1193 1635 116 C=0
1194 1636 1160 DADD=C
1195 1637 216 B=A MEM END IN B
1196 1640 1570 C=REGN 13
1197 1641 74 RCR 3
1198 1642 706 A=A-C X

```

```

1199 1643          260 C=N          C=REGISTERS WE NEED
1200 1644          706 A=A-C  X      COMP # 2 SHIFT LEFT POISTIVE
1201 1645          177 GOC    LARGER (1664) USER WANTS MORE REGISTERS
*****
*CODE BELOW SETS AX=CHAIN HEAD-WHERE WE STOP
*          AM=-1 DEC
*          BX=FROM ADDRESS-WHERE WE GET DATA
*          BM=TO ADDRESS-WHERE WE PUT DATA
*****
1208 1646          410 S9=      1      SET CLEAN UP FLAG0
1209 1647          316 C=B      CX=TO
1210 1650          256 AC EX
1211 1651          530 M=C      SAVE SHIFT DISTANCE
1212 1652          1116 C=A-C    CX=FROM
1213 1653          74 RCR      3
1214 1654          346 BC EX  X      C=FRMXXXXXXXXXTO
1215 1655          674 RCR      11
1216 1656          356 BC EX      B=TOFRM
1217 1657          1570 C=REGN 13    GET TEST ADDRESS
1218 1660          132 C=0  M
1219 1661          1172 C=C-1  M
1220 1662          256 AC EX      A=INCTST
1221 1663          303 GOTO    STRTMV (1713)
*****
*SETS UP AS FOR SMALLER EXCEPT WE STOP AT NOTHINGNESS
*AND START AT CHAIN END USING 1 FOR AN INCREMENT.
*****
1226 1664 LARGER  630 C=N          GET UNUSED REG
1227 1665          1006 C=C+A  X      ADD NEG SHIFT
1228 1666          67 GOC    LARG10 (1674) MADE IT
1229 1667          1114 ?S9=1      EXIT MODE?
1230 1670          1 GOLL    PACKE
1230 1671          3
1231 1672          1110 S9=      1      SAY DIDN'T MAKE IT
1232 1673          1740 RTN
*
1234 1674 LARG10 1570 C=REGN 13      A=NEGSHT C=FROM
1235 1675          1006 C=C+A  X      C=TO
1236 1676          674 RCR      11
1237 1677          356 BC EX      C=TST B=TOXXX
1238 1700          132 C=0  M
1239 1701          1072 C=C+1  M      C=INCTST
1240 1702          256 AC EX
1241 1703          530 M=C      SAVE NEG SHIFT
1242 1704          1570 C=REGN 13    C=FROM
1243 1705          346 BC EX  X      B=TOFRM
1244 1706          53 GOTO    STRTMV (1713)
*****
*THIS ROUTINE SHIFTS MEMORY EITHER LEFT OR RIGHT
*ACCORDING TO THE INPUTS IN A AND B.
*****
1249 1707 KPMVN  674 RCR      11      C=TOFRM
1250 1710          1032 C=C+A  M      INC TO
1251 1711          356 BC EX      C=DATA B=TOFRM
1252 1712          1360 DATA=C    DATA MOVED
1253 1713 STRTMV  356 BC EX      A=INCTST C=TOFRM
1254 1714          256 AC EX      C=TOFRM A=INCTST
1255 1715          414 ?S8=1      SMALLER MOVE?
1256 1716          63 GONC    CHKTOP (1724) NO
1257 1717          1406 ? A<C  X      IS FRM<TST

```



```

1258 1720          63 GONC   GETREG (1726) IF NO CONTINUE
1259 1721 B=0      256 AC EX          GET 0 FOR TO
1260 1722          56 B=0
1261 1723          143 GOTO   DOTO    (1737)
1262 1724 CHKTOP  1546 ? A#C   X          ZEROS AFTER MEM END
1263 1725          1743 GONC   B=0    (1721)
1264 1726 GETREG  256 AC EX          A=INCTST C=TOFROM
1265 1727          1160 DADD=C      DADD=FROM
1266 1730          372 BC EX   M          SAVE TO PART
1267 1731          674 RCR     11          INC DEC FROM
1268 1732          1032 C=C+A   M
1269 1733          74 RCR     3
1270 1734          346 BC EX   X          A=INCTST B=TOFROM
1271 1735          70 C=DATA
1272 1736          356 BC EX
1273 1737 DOTO    74 RCR     3          C=FRMXX...XX0TO
1274 1740          1160 DADD=C      DADD=TO
1275 1741          1546 ? A#C   X          DONE IF TO=TST
1276 1742          1457 GOC     KPMVN (1707) NO
1277 1743          356 BC EX          CLEAR LAST REGISTER
1278 1744          1360 DATA=C
*****
*FIX POINTERS AFTER MOVE.
*****
1282 1745          630 C=M          GET MOVE DISTANCE
1283 1746          356 CB EX          SAVE
1284 1747          1 GOSUB   GETPC    ENABLE CHIP0
1284 1750          0
1285 1751          314 ?S10=1        ROM RAM?
1286 1752          27 GOC     NOTRAM (1754)
1287 1753          446 A=A+B   X          SHIFT PC
1288 1754 NOTRAM  156 AB EX          B=PC FOR CLRSB2
1289 1755          1570 C=REGN 13
1290 1756          1006 C=C+A   X
1291 1757          74 RCR     3
1292 1760          1006 C=C+A   X
1293 1761          474 RCR     8
1294 1762          1006 C=C+A   X
1295 1763          74 RCR     3
1296 1764          1550 REGN=C 13
1297 1765          156 AB EX          BRING PC BACK TO A[3:0]
1298 1766          1104 S9=     0
1299 1767          1 GOLONG DCRT10
1299 1770          2
1300          ENTRY SETSST
*
* SETSST - SET SINGLE STEP BIT
* REQUIRES CHIP 0 ENABLED ON INPUT
* DESTROYS C
*
1306 1771 SETSST  1670 C=REGN 14
1307 1772          1530 ST=C
1308 1773          110 S4=     1          SET SST BIT
1309 1774          1630 C=ST
1310 1775          1650 REGN=C 14
1311 1776          1740 RTN
*
1313          UNLIST

ERRORS :      0

```

SYMBOL TABLE

ADDEM	727	-	725	
ADDIT	724	-	721	714
ARCL10	1234	-	1267	
ARCL20	1263	-	1277	
B=0	1721	-	1725	
CHK#S	330	-		
CHK#S1	324	-		
CHK#S2	331	-		
CHKAD4	1206	-		
CHKADR	1156	-		
CHKTOP	1724	-	1716	
CKAD10	1222	-	1214	
CKAD2	1175	-	1170	
CKAD3	1176	-	1166	
CKAD3J	1202	-	1177	
CKAD4	1206	-	1204	1174
CLEN	1466	-	1507	
CLR	1463	-		
CLRNXT	264	-	273	
COMBIN	745	-	742	
DEGDO	1452	-		
DELOOP	1406	-	1413	
DONSHF	1516	-	1547	
DONS10	1341	-	1564	1551
DOSKP	1061	-	1075	
DOTO	1737	-	1723	
DSPDN	1317	-	1314	
ELMFRC	644	-	647	
ERRAD	342	-	334	
FCNTBL	0	-		
FIND#1	1565	-		
FLGANN	1121	-	1114	
FNDEND	1460	-		
FNDPT	1620	-	1623	
FOUND1	1617	-	1602	
FSTIN	302	-		
GETLIN	31	-		
GETREG	1726	-	1720	
INSHFT	1515	-	1533	
KPMVN	1707	-	1742	
LARG10	1674	-	1666	
LARGER	1664	-	1645	
LOAD3	372	-		
MRSHT	740	-	744	
NFRPRL	347	-	277	
NOFRAC	646	-	643	
NOSKP	1031	-	1110	1055 1017 1004 1000 766
NOSKPD	1047	-	1104	1100 1067 1043 1040 1032
NOTRAM	1754	-	1752	
OVFL10	51	-		
OVFL15	54	-	52	
OVFL20	65	-		
OVFL30	66	-	63	
OVFL40	72	-	67	
OVFL50	76	-	71	
OVR0	1504	-	1502	

OVRDEC	656	-	654				
P6RTN	1160	-					
PCKDUR	1374	-					
RDNSUB	351	-					
REG	1550	-	1537				
REGALP	1260	-	1231				
REGG	1520	-	1513				
RTJ10	312	-	315				
RTJLBL	311	-					
R*SUB	355	-					
S1?	1313	-	1311				
SEPA	675	-	671				
SEFXY	322	-					
SETSST	1771	-					
SHFLFT	1557	-	1563				
SIZSUB	1627	-					
SKP	1056	-	1111	1030	777	767	764
SKPIT	1111	-					
SST?	1070	-	1060				
STFL10	1256	-	1320				
STFLGS	1247	-					
STRTMV	1713	-	1706	1663			
SUMCHK	1147	-					
SUMCK2	1151	-					
THSIT?	1600	-	1611				
TONE7	1426	-	1405				
TONE7X	1333	-					
TONE8	1421	-	1403				
TONE9	1415	-	1401				
TONER	1335	-					
TSTEXP	713	-	710				
TSTMAP	241	-					
XARCL	1226	-					
XASHF	1510	-					
XASTO	1534	-					
XBEEP	1321	-					
XCF	1115	-					
XCLSIG	260	-					
XDEG	1434	-					
XDEG2	1436	-	1451	1445			
XDSE	637	-					
XFS?	1105	-					
XGRAD	1446	-					
XISG	640	-					
XRAD	1442	-					
XRDN	275	-					
XR*	345	-					
XSCI	1300	-					
XSF	1112	-					
XSGREG	1131	-					
XSIZE	1625	-					
XSTYON	21	-					
XTONE	1336	-					
XX#0?	1021	-					
XX#Y?	1051	-					
XX<0	774	-	1020				
XX<0?	772	-					
XX<=0?	1015	-	773				
XX<=0A	1011	-					
XX<=Y?	1001	-					

XX<Y?	757	-	
XX=0?	1006	-	
XX=Y?	1024	-	
XX>0?	761	-	1005
XX>Y?	770	-	756
XYN	1054	-	1023
XYX	1027	-	1010
Y-X	41	-	

ENTRY TABLE

CHK#S	330	-
CHK#S1	324	-
CHK#S2	331	-
CHKAD4	1206	-
CHKADR	1156	-
CLR	1463	-
DEGDO	1452	-
DOSKF	1061	-
ERRAD	342	-
FCNTBL	0	-
FIND#1	1565	-
FLGANN	1121	-
FNDEND	1460	-
FSTIN	302	-
GETLIN	31	-
LOAD3	372	-
NOSKF	1031	-
OVFL10	51	-
P6RTN	1160	-
PCKDUR	1374	-
RDNUS8	351	-
RTJLBL	311	-
R*SUB	355	-
SEPMY	322	-
SETSST	1771	-
SIZSUB	1627	-
SKP	1056	-
STFLGS	1247	-
SUMCHK	1147	-
SUMCK2	1151	-
TONE7	1426	-
TONE7X	1333	-
TONEB	1335	-
TSTMAP	241	-
XARCL	1226	-
XASHF	1510	-
XASTO	1534	-
XBEEP	1321	-
XCF	1115	-
XCLSIG	260	-
XDEG	1434	-
XDSE	637	-
XFS?	1105	-
XGRAD	1446	-
XISG	640	-
XRAD	1442	-
XRDN	275	-
XR*	345	-
XSCI	1300	-
XSF	1112	-
XSGREG	1131	-
XSIZE	1625	-
XSTYON	21	-
XTONE	1336	-
XX#0?	1021	-
XX#Y?	1051	-

XX<0?	772	-
XX<=0?	1015	-
XX<=0A	1011	-
XX<=Y?	1001	-
XX<Y?	757	-
XX=0?	1006	-
XX=Y?	1024	-
XX>0?	761	-
XX>Y?	770	-
Y-X	41	-

EXTERNAL REFERENCES

(*)	102		
(10)*X	127		
+	100		
-	101		
-DEC	137		
-OCT	157		
/	103		
ABS	141		
ACOS	135		
AD2-10	45	731	
AD2-10	46	732	
ADVANCE	217		
AFORMT	1232		
AFORMT	1233		
AGTO	35		
ANH+14	1440		
ANH+14	1441		
ANNOUT	1124		
ANNOUT	1125		
AOFF	213		
AON	214		
APNDHW	1275		
APNDHW	1276		
ARCL	233		
ARGOUT	1245		
ARGOUT	1246		
ASHF	210		
ASIN	134		
ASN	17		
ASTO	232		
ATAN	136		
AVIEW	176		
AXEQ	36		
BEEP	206		
BST	7		
CAT	0		
CF	251		
CHK#S	325	641	1012
CHK#S	326	642	1013
CHS	124		
CLA	207		
CLDSP	177		
CLP	4		
CLREG	212		
CLSIG	160		
CLST	163		
CLX	167		
COLDST	1220		
COLDST	1221		
COPY	3		
COS	132		
D-R	152		
DCRT10	1767		
DCRT10	1770		
DEG	200		
DEGDO	1434	1442	1446

DEGDO	1435	1443	1447
DEL	2		
DELETE	13		
DSE	227		
END	300		
ENG	236		
ENTER^	203		
ERRNE	1216		
ERRNE	1217		
ERROR	342		
ERROR	343		
E^X	125		
E^X-1	130		
FACT	142		
FC?	255		
FC?C	253		
FIND#1	1510	1534	
FIND#1	1511	1535	
FIX	234		
FNDEND	1633		
FNDEND	1634		
FRAC	151		
FS?	254		
FS?C	252		
GETPC	1061	1747	
GETPC	1062	1750	
GRAD	202		
GTO	320		
GTOL	1		
H-HMS	154		
HMS+	111		
HMS-	112		
HMS-H	155		
INT	150		
ISG	226		
LASTX	166		
LBL	317		
LN	120		
LN1+X	145		
LOG	126		
MEAN	174		
MEMLFT	1630		
MEMLFT	1631		
MOD	113		
MODE	14		
MSG	1044	1101	
MSG	1045	1102	
MSGAD	344		
MSGNO	1103		
MSGYES	1046		
NFRPR	347		
NFRPR	350		
NFRPU	244	1047	
NFRPU	245	1050	
OFF	215		
ONE/X	140		
P-R	116		
PACK	12		
PACKE	1670		
PACKE	1671		

PATCH6	1156		
PATCH6	1157		
PCKDUR	1360		
PCKDUR	1361		
PCT	114		
PCTCH	115		
PI	162		
PROMPT	216		
PSE	211		
PUTPCX	1065		
PUTPCX	1066		
R-D	153		
R-P	117		
R/S	5		
RAD	201		
RCL	220		
RDN	165		
RDNSUB	275		
RDNSUB	276		
RND	156		
RTN	205		
R^	164		
R^SUB	345	351	353
R^SUB	346	352	354
SCI	235		
SF	250		
SHF40	747		
SHF40	750		
SHIFT	16		
SIGMA+	107		
SIGMA-	110		
SIGN	172		
SIGREG	231		
SIN	131		
SINFR	650		
SINFR	651		
SIZE	6		
SKPLIN	1063		
SKPLIN	1064		
SQRT	122		
SRBMAP	246		
SRBMAP	247		
SST	10		
STAYON	11		
STDEV	175		
STO	221		
STO*	224		
STO+	222		
STO-	223		
STO/	225		
STOP	204		
SUNCHK	260		
SUNCHK	261		
SUMCK2	1132		
SUMCK2	1133		
TAN	133		
TBITNA	241		
TBITNA	242		
TONE	237		
TONEB	1321	1325	1331

TONEB	1322	1326	1332
VIEW	230		
X#0?	143		
X#Y?	171		
X<0?	146		
X<=0?	173		
X<=Y?	106		
X<>	316		
X<>Y	161		
X<Y?	104		
X=0?	147		
X=Y?	170		
X>0?	144		
X>Y?	105		
XEQ	340		
XGOIND	256		
X^2	121		
Y-X	757	770	1001
Y-X	760	771	1002
Y^X	123		

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

2

FILE CN6B

```
*****
*   NUT MATH ROM 1
*   STARTING ADDRESS=014000
*****
*   COMMON MATH ENTRIES
*       IF NUMBER IS 2-10,
*           THEN FORM IS:
*               A HAS 10 DIGIT FORM
*               C HAS 10 DIGIT FORM
*       IF NUMBER IS 1-10,
*           THEN FORM IS:
*               A HAS SIGN AND EXP
*               B HAS 13 DIGIT MANTISSA
*               C HAS 10 DIGIT FORM
*****
```

```
*      IF NUMBER IS 2-13,      ***
*      THEN FORM IS:          ***
*      A AND B AS IN 1-10      ***
*      M HAS SIGN AND EXP      ***
*      C HAS 13 DIGIT MANTISSA ***
*
*      ON EXIT, C HAS 10 DIGIT FORM ***
*      A AND B HAVE 13 DIGIT FORM ***
*
```

```
28      ENTRY  AD2-10
29      ENTRY  AD1-10
30      ENTRY  AD2-13
31      ENTRY  MP2-10
32      ENTRY  MP1-10
33      ENTRY  MP2-13
34      ENTRY  MPY150
35      ENTRY  DV2-10
36      ENTRY  DV1-10
37      ENTRY  DV2-13
38      ENTRY  1/X10
39      ENTRY  1/X13
40      ENTRY  X/Y13
41      ENTRY  DIV120
42      ENTRY  DIV110
43      ENTRY  DIV15
44      ENTRY  ADDONE
45      ENTRY  SUBONE
46      ENTRY  SQR10
47      ENTRY  SQR13
48      ENTRY  ERR0
49      ENTRY  NRM10
50      ENTRY  NRM11
51      ENTRY  NRM12
52      ENTRY  NRM13
53      ENTRY  LN10
54      ENTRY  LNC10*
55      ENTRY  XLN1+X
56      ENTRY  XY*X
57      ENTRY  SHF10
58      ENTRY  SHF40
59      0 ADDONE  116 C=0  W
60      1          33 GOTO  SUBON1 ( 4 )
61      2 SUBONE  116 C=0  W
62      3          1276 C=-C-1 S
63      4 SUBON1  1534 PT=  12
64      5          120 LC  1
65      6          33 GOTO  AD1-10 ( 11 )
66      7 AD2-10  56 B=0  W
67      10        172 AB EX M
68      11 AD1-10  730 MC EX
69      12        630 C=M
70      13        106 C=0  X
71      14 AD2-13  76 B=0  S
72      15        136 C=0  S
73      16        730 MC EX
74      17        1434 PT=  1
75      20 ADD10  1724 DEC PT
76      21        566 A=A+1 XS
77      22        1066 C=C+1 XS
```

THESE 2 STATES COULD
BE JUST "M=C"

78	23	1524 ? PT=	12	
79	24	1743 GONC	ADD10	(20)
80	25	356 BC EX	W	
81	26	1356 ? C#0	W	
82	27	553 GONC	ADD60	(104)
83	30	730 MC EX		
84	31	156 AB EX	W	
85	32	1356 ? C#0	W	
86	33	513 GONC	ADD60	(104)
87	34	ADD30 1446 ? A<B	X	
88	35	207 GOC	ADD65	(55)
89	36	ADD90 730 MC EX		
90	37	156 AB EX	W	
91	40	1446 ? A<B	X	
92	41	1737 GOC	ADD30	(34)
93	42	ADD45 256 AC EX	W	
94	43	730 MC EX		
95	44	1416 ? A<C	W	
96	45	43 GONC	ADD55	(51)
97	46	730 MC EX		
98	47	256 AC EX	W	
99	50	53 GOTO	ADD65	(55)
100	51	ADD55 730 MC EX		
101	52	256 AC EX	W	
102	53	730 MC EX		
103	54	156 AB EX	W	
104	55	ADD65 636 A=A-B	S	
105	56	1536 ? A#0	S	
106	57	133 GONC	ADD50	(72)
107	60	ADD40 1216 C=-C	W	
108	61	436 A=C	S	
109	62	1446 ? A<B	X	
110	63	73 GONC	ADD50	(72)
111	64	730 MC EX		
112	65	1374 RCR	13	
113	66	730 MC EX		
114	67	156 AB EX	W	
115	70	646 A=A-1	X	
116	71	156 AB EX	W	
117	72	ADD50 1446 ? A<B	X	
118	73	113 GONC	ADD60	(104)
119	74	546 A=A+1	X	
120	75	1716 C SR	W	
121	76	276 AC EX	S	
122	77	436 A=C	S	
123	100	1724 DEC PT		
124	101	1324 ? PT=	13	
125	102	1703 GONC	ADD50	(72)
126	103	116 C=0	W	
127	104	ADD60 256 AC EX	W	
128	105	630 C=M		
129	106	356 BC EX	W	
130	107	456 A=A+B	W	
131	110	1166 C=C-1	XS	
132	111	1166 C=C-1	XS	
133	112	1166 C=C-1	XS	
134	113	0 NOP		
135	114	313 GOTO	MPY150	(145)
136	115	MP2-10 56 B=0	W	
137	116	172 AB EX	M	

138	117	MP1-10	730	MC EX	
139	120		630	C=M	
140	121		106	C=0	X
141	122	MP2-13	76	B=0	S
142	123		136	C=0	S
143	124		730	MC EX	
144	125		1006	C=A+C	X
145	126		1136	G=A-C	S
146	127		23	GONC	MPY110 (131)
147	130		1236	C=-C	S
148	131	MPY110	16	A=0	W
149	132		730	MC EX	
150	133		1334	PT=	13
151	134	MPY120	1734	INC PT	
152	135		1616	A SR	W
153	136		23	GOTO	MPY140 (140)
154	137	MPY130	456	A=A+B	W
155	140	MPY140	1142	C=C-1	PT
156	141		1763	GONC	MPY130 (137)
157	142		1524	? PT=	12
158	143		1713	GONC	MPY120 (134)
159	144		630	C=M	
160	145	MPY150	256	AC EX	W
161	146		730	MC EX	
162	147		630	C=M	
163	150	MPY160	1376	? C#0	S
164	151		33	GONC	SHF40 (154)
165	152		546	A=A+1	X
166	153		1716	C SR	W
167	154	SHF40	256	AC EX	W
168	155	SHF10	1534	PT=	12
169	156		1512	? A#0	WPT
170	157		277	GOC	SHF20 (206)
171	160	HRM10	356	BC EX	W
172	161	HRM11	256	AC EX	W
173	162	HRM12	1534	PT=	12
174	163		416	A=C	W
175	164		746	C=C+C	X
176	165		153	GONC	HRM20 (202)
177	166		1072	C=C+1	M
178	167		133	GONC	HRM20 (202)
179	170		306	C=B	X
180	171		1046	C=C+1	X
181	172		1042	C=C+1	PT
182	173	HRM30	336	C=B	S
183	174		156	AB EX	W
184	175		1372	? C#0	M
185	176		1540	RTN C	
186	177		116	C=0	W
187	200		36	A=0	S
188	201	HRM40	1740	RTN	
189	202	HRM20	306	C=B	X
190	203		1703	GOTO	HRM30 (173)
191	204	HRM13	156	AB EX	W
192	205		1543	GOTO	HRM11 (161)
193	206	SHF20	1502	? A#0	PT
194	207		1517	GOC	HRM10 (160)
195	210		1146	C=C-1	X
196	211		1752	A SL	WPT
197	212		1743	GOTO	SHF20 (206)

***ROUND,SHIFT AND NORMALIZE

```

198 213 1/X10      56 B=0      W
199 214           372 BC EX    M
200 215           256 AC EX    W
201 216 1/X13      116 C=0     W
202 217           730 MC EX
203 220           116 C=0     W
204 221           1534 PT=     12
205 222           120 LC      1
206 223 X/Y13      356 BC EX    W
207 224           256 AC EX    W
208 225           730 MC EX
209 226           256 AC EX    W
210 227           63 GOTO     DY2-13 ( 235)
211 230 DV2-10     56 B=0     W
212 231           172 AB EX    M
213 232 DV1-10     730 MC EX
214 233           630 C=M
215 234           106 C=0     X
216 235 DV2-13     76 B=0     S
217 236           1372 ? C#0   M
218 237           443 GONC     ERR0 ( 303)
219 240 DIV100     730 MC EX
220 241           1106 C=A-C   X
221 242           1136 C=A-C   S
222 243           23 GONC     DIV110 ( 245)
223 244           1236 C=-C    S
224 245 DIV110     730 MC EX
225 246           136 C=0     S
226 247           256 AC EX    W
227 250           156 AB EX    W
228 251 DIV15      1456 ? A<B   W
229 252           53 GONC     DIV120 ( 257)
230 253           730 MC EX
231 254           1756 A SL     W
232 255           1146 C=C-1   X
233 256           730 MC EX
234 257 DIV120     1534 PT=     12
235 260           116 C=0     W
236 261           23 GOTO     DIV140 ( 263)
237 262 DIV130     1042 C=C+1   PT
238 263 DIV140     616 A=A-B    W
239 264           1763 GONC     DIV130 ( 262)
240 265           456 A=A+B    W
241 266           1756 A SL     W
242 267           1724 DEC PT
243 270           1324 ? PT=    13
244 271           1723 GONC     DIV140 ( 263)
245 272           256 AC EX    W
246 273           630 C=M
247 274           1 GOLONG     NRM10
247 275           2
248 276 SQR10      56 B=0     W          ***SQUARE ROOT
249 277           372 BC EX    M
250 300           256 AC EX    W
251 301 SQR13      1536 ? A#0   S
252 302           33 GONC     SQR20 ( 305)
253 303 ERR0       1 GOLONG     ERRDE    **ERROR EXIT
253 304           2
254 305 SQR20      76 B=0     S
255 306           316 C=B     W

```

256	307	156 AB EX	W
257	310	756 C=C+C	W
258	311	756 C=C+C	W
259	312	1016 C=A+C	W
260	313	356 BC EX	W
261	314	132 C=0	M
262	315	416 A=C	W
263	316	756 C=C+C	W
264	317	746 C=C+C	X
265	320	23 GONC	SQR30 (322)
266	321	1272 C=-C-1	M
267	322 SQR30	516 A=A+C	W
268	323	1634 PT=	0
269	324	1502 ? A#0	PT
270	325	27 GOC	SQR50 (327)
271	326	1656 B SR	W
272	327 SQR50	1616 A SR	W
273	330	116 C=0	W
274	331	156 AB EX	W
275	332	1334 PT=	13
276	333	520 LC	5
277	334	1716 C SR	W
278	335	123 GOTO	SQR100 (347)
279	336 SQR60	1042 C=C+1	PT
280	337 SQR70	716 A=A-C	W
281	340	1763 GONC	SQR60 (336)
282	341	516 A=A+C	W
283	342	1756 A SL	W
284	343	1624 ? PT=	0
285	344	1 GOLC	NRM12
285	345	3	
286	346	1724 DEC PT	
287	347 SQR100	1712 C SR	WPT
288	350	1673 GOTO	SQR70 (337)
289		ENTRY	XFT100
290	351 XFT120	646 A=A-1	X
291	352	73 GONC	XFT110 (361)
292	353	1303 GOTO	ERR0 (303)
293	354 XFT100	1376 ? C#0	S
294	355	1267 GOC	ERR0 (303)
295	356	1366 ? C#0	XS
296	357	1247 GOC	ERR0 (303)
297	360	416 A=C	W
298	361 XFT110	216 B=A	W
299	362	34 PT=	3
300	363	1756 A SL	W
301	364	1612 A SR	WPT
302	365	1534 PT=	12
303	366	1512 ? A#0	WPT
304	367	1627 GOC	XFT120 (351)
305	370	546 A=A+1	X
306		LEGAL	
307	371	1406 ? A<C	X
308	372	33 GONC	XFT130 (375)
309	373	1066 C=C+1	XS
310	374	1740 RTN	
311	375 XFT130	116 C=0	W
312	376	1042 C=C+1	PT
313	377	1716 C SR	W
314	400	1076 C=C+1	S


```

315 401          356 BC EX W
316 402 XFT140 1302 ? B#0 PT
317 403          33 GONC XFT150 ( 406 )
318 404          1652 B SR WPT
319 405          1046 C=C+1 X
320 406 XFT150   16 A=0 W
321 407          702 A=A-C PT
322 410          43 GONC XFT170 ( 414 )
323 411          1756 A SL W
324 412 XFT160   456 A=A+B W
325 413          1773 GONC XFT160 ( 412 )
326 414 XFT170   736 A=A-C S
327 415          63 GONC XFT190 ( 423 )
328 416          1612 A SR WPT
329 417          556 A=A+1 W
330 420          1046 C=C+1 X
331 421 XFT180   456 A=A+B W
332 422          1773 GONC XFT190 ( 421 )
333 423 XFT190   152 AB EX WPT
334 424          1142 C=C-1 PT
335 425          1553 GONC XFT140 ( 402 )
336 426          1176 C=C-1 S
337 427          1533 GONC XFT140 ( 402 )
338 430          1756 A SL W
339 431          206 B=A X
340 432          132 C=0 M
341 433          136 C=0 S
342 434          452 A=A+B WPT
343 435          516 A=A+C W
344 436          272 AC EX M
345 437          1740 RTN

```

```

*****
* MATH SCRATCH ROUTINES
*
* STSCR STORES S AND 13 DIGIT MANTISSA IN
* REGN 9 AND EXP IN-REGN 10, LEAVING
* A AND B ALONE
* RCSCR RECALLS MATH SCRATCH INTO C AND M,
* LEAVING A AND B ALONE
* EXSCR EXCHANGES A AND B WITH THE MATH
* SCRATCH REGISTERS, DESTROYING C
*****

```

```

357          ENTRY STSCR
358          ENTRY RCSCR
359          ENTRY EXSCR
360          ENTRY RCSCR*
361          ENTRY STSCR*
362 440 STSCR* 116 C=0 W
363 441          1160 DADD=C
364 442 STSCR 236 B=A S
365 443          316 C=B W
366 444          1150 REGN=C 9
367 445          1270 C=REGN 10
368 446          246 AC EX X
369 447          406 A=C X
370 450 STSCR1 1250 REGN=C 10
371 451          1740 RTN
372 452 EXSCR 176 AB EX S
373 453          1170 C=REGN 9
374 454          356 BC EX W

```

```

375 455          1150 REGN=C 9
376 456          1270 C=REGN 10
377 457          246 AC EX X
378 460          176 AB EX S
379 461          1673 GOTO STSCR1 ( 450 )
380 462 RCSCR*   116 C=0
381 463          1160 DADD=C
382 464 RCSCR    1170 C=REGN 9
383 465          376 BC EX S
384 466          730 MC EX
385 467          1270 C=REGN 10
386 470          336 C=B S
387 471          730 CM EX
388 472          1740 RTN
389          ENTRY INTFR0
390          ENTRY SINFR
391          ENTRY SINFRA
392          ENTRY MOD10
393          ENTRY DTOR
394          ENTRY RTOD
395          ENTRY LD90
396          ENTRY PI/2
397          ENTRY TRC10
*****
* IF S5=1, THEN ROUTINE INTFR0 FINDS INT *
* IF S5=0, INTFR0 FINDS FRACTIONAL PART *
*****
402 473 INTFR0   1 GOSUB SINFR
402 474          0
403 475          214 ?S5=1
404 476          1 GOLNC SHF10
404 477          2
405 500          630 C=M
* NEXT TWO STATES ARE A HOLDOVER FROM A VERSION OF INTFR0 WHICH
* WORKED FOR 13 DIGIT ARITHMETIC. NOT NECESSARY HERE.
408 501          1624 ? PT= 0
409 502          37 GOC INT30 ( 505 )
410 503          1724 DEC PT
411 504          112 C=0 WPT
412 505 INT30    1 GOLONG NRM12
412 506          2
413 507 SINFR    56 B=0 W
414 510          372 BC EX M
415 511          256 AC EX W
416 512 SINFRA 1334 PT= 13
417 513          316 C=B W
418 514          730 CM EX
419 515          156 AB EX W
420 516          316 C=B W
421 517          1366 ? C#0 XS
422 520          1540 RTN C
423 521          1046 C=C+1 X
424 522 SINFR1 1346 ? C#0 X
425 523          73 GONC SINFR2 ( 532 )
426 524          1146 C=C-1 X
427 525          1756 A SL W
428 526          36 A=0 S
429 527          1724 DEC PT
430 530          1516 ? A#0 W
431 531          1717 GOC SINFR1 ( 522 )

```

```

432 532 SINFR2 1146 C=C-1 X
433 533 1740 RTN
434 534 MOD10 566 A=A+1 XS
435 535 1372 ? C#0 M
436 536 73 GONC MOD5 ( 545 )
437 537 1066 C=C+1 XS
438 540 1576 ? A#0 S
439 541 23 GONC MOD1 ( 543 )
440 542 110 S4= 1
441 MOD1
442 543 1106 C=A-C X
443 544 43 GONC MOD2 ( 550 )
444 545 MOD5 666 A=A-1 XS
445 546 256 AC EX W
446 547 223 GOTO MOD4 ( 571 )
447 550 MOD2 56 B=0 W
448 551 360 CN EX
449 552 276 AC EX S
450 553 360 CN EX
451 554 36 A=0 S
452 555 6 A=0 X
453 556 372 BC EX M
454 557 MOD3 616 A=A-B W
455 560 1773 GONC MOD3 ( 557 )
456 561 456 A=A+B W
457 562 1756 A SL W
458 563 1146 C=C-1 X
459 564 1733 GONC MOD3 ( 557 )
460 565 1616 A SR W
461 566 260 C=N
462 567 1 GSBLGX SHF10
462 570 0
463 571 MOD4 114 ?S4=1
464 572 1640 RTN NC
465 573 1372 ? C#0 M
466 574 1640 RTN NC
467 575 416 A=C W
468 576 370 C=REGN 3
469 577 1 GOLONG AD2-10
469 600 2
470 601 DTOR 16 A=0
471 602 1 GOSUB PI/2
471 603 0
472 604 356 BC EX W
473 605 260 C=N
474 606 1 GSBLNG MP1-10
474 607 0
475 610 1 GOSUB LD90
475 611 0
476 612 1 GOLONG DV1-10
476 613 2
477 614 RTOD 416 A=C
478 615 1 GOSUB LD90
478 616 0
479 617 1 GSBLNG MP2-10
479 620 0
480 621 1 GOSUB PI/2
480 622 0
481 623 1 GOLONG DV2-13
481 624 2

```

```

482 625 LD90      116 C=0      W
483 626          1534 PT=     12
484 627          1046 C=C+1   X
485 630          1120 LC      9
486 631          1740 RTN
487 632 PI/2      116 C=0      W
488 633          730 CM EX
489 634          1 GOSUB     TRC10
489 635          0
490 636          756 C=C+C     W
491 637          1716 C SR     W
492 640          1740 RTN
493 641 TRC10     1534 PT=     12
494 642          116 C=0      W
495 643          720 LC      7
496 644          1020 LC      8
497 645          520 LC      5
498 646          320 LC      3
499 647          1120 LC      9
500 650          1020 LC      8
501 651          120 LC      1
502 652          620 LC      6
503 653          320 LC      3
504 654          320 LC      3
505 655          1120 LC      9
506 656          720 LC      7
507 657          520 LC      5
508 660          1534 PT=     12
509 661          1740 RTN
510          ENTRY   XTOHRS
511          ENTRY   HMSMP
512          ENTRY   HMSDV

```

```

*****
*      IF TO H.MMSS, THEN S5=1      *
*      IF TO H.DDDD, THEN S5=0      *
*****

```

```

517 662 XTOHRS 1372 ? C#0 M
518 663          1640 RTN NC
519 664          416 A=C      W
520 665          216 B=A      W
521 666          1046 C=C+1   X
522 667          1046 C=C+1   X
523 670          406 A=C      X
524 671          1534 PT=     12
525 672          506 A=A+C     X
526 673          107 GOC      HMS140 ( 703 )
527 674 HMS110 1724 DEC PT
528 675          1624 ? PT=     0
529 676          33 GONC      HMS130 ( 701 )
530 677 HMS120 316 C=B      W
531 700          1740 RTN
532 701 HMS130 1146 C=C-1   X
533 702          1723 GONC      HMS110 ( 674 )
534 703 HMS140 116 C=0      W
535 704          332 C=B      M
536 705          214 ?S5=1
537 706          223 GONC      HRS100 ( 730 )
538 707          1734 INC PT
539 710          1324 ? PT=     13
540 711          43 GONC      HMS150 ( 715 )

```

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

```

541 712          1 GOSUB HMSMP
541 713          0
542 714          53 GOTO HMS160 ( 721 )
543 715 HMS150 1734 INC PT
544 716          1 GOSUB HMSMP
544 717          0
545 720          1724 DEC PT
546 721 HMS160 1724 DEC PT
547          LEGAL
548 722          1 GOSUB HMSMP
548 723          0
549 724          416 A=C      W
550 725          316 C=B      W
551 726 HMS170    1 GOLONG: MPYT50
551 727          2
552 730 HRS100    16 A=0      W
553 731          1 GOSUB HMSDV
553 732          0
554 733          1734 INC PT
555 734          1324 ? PT= 13
556 735          27 GOC      HRS120 ( 737 )
557 736          1734 INC PT
558 737 HRS120    1 GOSUB HMSDV
558 740          0
559 741          1756 A SL     W
560 742          516 A=A+C    W
561 743          356 BC EX    W
562 744          1623 GOTO HMS170 ( 726 )
563 745 HMSDV     1712 C SR    WPT
564 746          1012 C=A+C    WPT
565 747 HMSMP     412 A=C      WPT
566 750          1712 C SR    WPT
567 751          752 C=C+C    WPT
568 752          752 C=C+C    WPT
569 753          1112 C=A-C    WPT
570 754          214 ?S5=1
571 755          63 GONC      HMSM20 ( 763 )
572 756          16 A=0      W
573 757          406 A=C      X
574 760          1016 C=A+C    W
575 761          106 C=0      X
576 762          1740 RTN
577 763 HMSM20    512 A=A+C    WPT
578 764          1712 C SR    WPT
579 765          1352 ? C#0    WPT
580 766          1757 GOC      HMSM20 ( 763 )
581 767          1740 RTN
582          ENTRY EXP710
583          ENTRY LN560
584          ENTRY PMUL
585          ENTRY LNSUB
586          ENTRY LNSUB-
587          ENTRY LNC20
588          ENTRY LNAP
589          ENTRY EXP10
590          ENTRY EXP13
591          ENTRY 10TOX
592          ENTRY LNC10
593          ENTRY EXP500
594          ENTRY EXP400

```

```

595                                     ENTRY EXP720
596 770 LNSUB- 1276 C=-C-1 S
597 771 LNSUB 156 AB EX W
598 772       216 B=A W
599 773       730 MC EX
600 774       630 C=M
601 775 LNSUB1 1616 A SR W
602 776       1516 ? A#0 W
603 777       47 GOC LNSUB2 (1003)
604 1000       630 C=M
605 1001       416 A=C W
606 1002       1740 RTN
607 1003 LNSUB2 1046 C=C+1 X
608 1004       1713 GONC LNSUB1 ( 775)
609 1005       1534 PT= 12
610 1006       542 A=A+1 PT
611 1007       156 AB EX W
612 1010       1 GOLONG DIV15
612 1011       2
613 1012 EXP10 56 B=0 W ***EXP(X)
614 1013       372 BC EX M
615 1014       256 AC EX W
616 1015 EXP13 4 S3= 0
617 1016       1536 ? A#0 S
618 1017       23 GONC EXP110 (1021)
619 1020       10 S3= 1
620 1021 EXP110 36 A=0 S
621 1022       32 A=0 M
622 1023       156 AB EX W
623 1024       316 C=B W
624 1025       746 C=C+C X
625 1026       163 GONC EXP200 (1044)
626 1027       316 C=B W
627 1030       1536 ? A#0 S
628 1031       177 GOC EXP120 (1050)
629 1032       1334 PT= 13
630 1033 EXP130 1724 DEC PT
631 1034       224 ? PT= 5
632 1035       1 GOLC EXP500
632 1036       3
633 1037       1046 C=C+1 X
634 1040       1733 GONC EXP130 (1033)
635 1041 EXP400 1 GSBLNG LNC20
635 1042       0
636 1043       643 GOTO EXP420 (1127)
637 1044 EXP200 1 GOSUB LNC10
637 1045       0
638 1046       534 PT= 6
639 1047       63 GOTO EXP220 (1055)
640 1050 EXP120 156 AB EX W
641 1051       546 A=A+1 X
642 1052       1656 B SR W
643 1053       1463 GOTO EXP110 (1021)
644 1054 EXP210 1072 C=C+1 M
645 1055 EXP220 616 A=A-B W
646 1056       1763 GONC EXP210 (1054)
647 1057       456 A=A+B W
648 1060       1756 A SL W
649 1061       1146 C=C-1 X
650 1062       123 GONC EXP230 (1074)

```

651	1063	234	PT=	5
652	1064	1342	? C#0	PT
653	1065	53	GONC	EXP240 (1072)
654	1066	1142	C=C-1	PT
655	1067	1342	? C#0	PT
656	1070	117	GOC	EXP300 (1101)
657	1071	1042	C=C+1	PT
658	1072	EXP240	1534	PT= 12
659	1073	443	GOTO	EXP430 (1137)
660	1074	EXP230	256	AC EX W
661	1075	1772	A SL	M
662	1076	256	AC EX	W
663	1077	1342	? C#0	PT
664	1100	1553	GONC	EXP220 (1055)
665	1101	EXP300	116	C=0 W
666	1102	1534	PT=	12
667	1103	1152	C=C-1	WPT
668	1104	416	A=C	W
669	1105	1034	PT=	2
670	1106	120	LC	I
671	1107	14	?S3=1	
672	1110	23	GONC	EXP700 (1112)
673	1111	1246	C=-C-1	X
674	1112	EXP700	156	AB EX W
675	1113	416	A=C	W
676	1114	EXP710	114	?S4=1
677	1115	33	GONC	EXP720 (1120)
678	1116	1	GSBLNG	SUBONE
678	1117	0		
679	1120	EXP720	1214	?S7=1
680	1121	33	GONC	EXP730 (1124)
681	1122	1276	C=-C-1	S
682	1123	436	A=C	S
683	1124	EXP730	1	GOLONG
683	1125	2		
684	1126	EXP410	1042	C=C+1 PT
685	1127	EXP420	616	A=A-B W
686	1130	1763	GONC	EXP410 (1126)
687	1131	456	A=A+B	W
688	1132	524	? PT=	6
689	1133	77	GOC	EXP510 (1142)
690	1134	1756	A SL	W
691	1135	1146	C=C-1	X
692	1136	1724	DEC	PT
693	1137	EXP430	356	BC EX W
694	1140	1013	GOTO	EXP400 (1041)
695	1141	EXP500	356	BC EX W
696	1142	EXP510	114	?S4=1
697	1143	53	GONC	EXP570 (1150)
698	1144	1	GOSUB	LNAP
698	1145	0		
699	1146	256	AC EX	W
700	1147	156	AB EX	W
701	1150	EXP570	1334	PT= 13
702	1151	620	LC	6
703	1152	234	PT=	5
704	1153	EXP550	1372	? C#0 M
705	1154	633	GONC	EXP600 (1237)
706	1155	1734	INC	PT
707	1156	EXP560	1342	? C#0 PT

708	1157		63 GONC	EXP520 (1165)
709	1160		1142 C=C-1	PT
710	1161		216 B=A	W
711	1162		730 MC EX	
712	1163		630 C=M	
713	1164		203 GOTO	EXP530 (1204)
714	1165	EXP520	1046 C=C+1	X
715	1166		1616 A SR	W
716	1167		1176 C=C-1	S
717	1170		1633 GONC	EXP550 (1153)
718	1171		1716 C SR	W
719	1172		1716 C SR	W
720	1173		1716 C SR	W
721	1174		542 A=A+1	PT
722	1175		156 AB EX	W
723	1176		416 A=C	W
724	1177		14 ?S3=1	
725	1200		1 GSUBCX	1/X13
725	1201		1	
726	1202		1123 GOTO	EXP710 (1114)
727	1203	EXP540	1656 B SR	W
728	1204	EXP530	1176 C=C-1	S
729	1205		1763 GONC	EXP540 (1203)
730	1206		456 A=A+B	W
731	1207		576 A=A+1	S
732	1210		630 C=M	
733	1211		1453 GOTO	EXP560 (1156)
734	1212	LNAP	730 MC EX	
735	1213		630 C=M	
736	1214		216 B=A	W
737	1215		356 BC EX	W
738	1216		756 C=C+C	W
739	1217		756 C=C+C	W
740	1220		1016 C=A+C	W
741	1221		156 AB EX	W
742	1222	LNAP1	1716 C SR	W
743	1223		1356 ? C#0	W
744	1224		47 GOC	LNAP2 (1230)
745	1225		630 C=M	
746	1226		256 AC EX	W
747	1227		1740 RTN	
748	1230	LNAP2	546 A=A+1	X
749	1231		1713 GONC	LNAP1 (1222)
750	1232		1216 C=-C	W
751	1233		730 MC EX	
752	1234		1046 C=C+1	X
753			LEGAL	
754	1235		1 GOLONG	DIV110
754	1236		2	
755	1237	EXP600	156 AB EX	W
756	1240		132 C=0	M
757	1241		136 C=0	S
758	1242		416 A=C	W
759	1243		14 ?S3=1	
760	1244		33 GONC	EXP740 (1247)
761	1245		1 GSBLNG	LN SUB-
761	1246		0	
762	1247	EXP740	114 ?S4=1	
763	1250		37 GOC	EXP750 (1253)
764	1251		1 GSBLNG	ADDONE

764	1252		0	
765	1253	EXP750	1	GOLONG EXP720
765	1254		2	
766	1255	LNC10*	356	BC EX W
767	1256	LNC10	1534	PT= 12
768	1257		220	LC 2
769	1260		320	LC 3
770	1261		20	LC 0
771	1262		220	LC 2
772	1263		520	LC 5
773	1264		1020	LC 8
774	1265		520	LC 5
775	1266		20	LC 0
776	1267		1120	LC 9
777	1270		220	LC 2
778	1271		1120	LC 9
779	1272		1120	LC 9
780	1273		420	LC 4
781	1274		413	GOTO LNCEND (1335)
782	1275	LNC20	116	C=0 W
783	1276		1524	? PT= 12
784	1277		227	GOC LNC30 (1321)
785	1300		1172	C=C-1 M
786	1301		420	LC 4
787	1302		1072	C=C+1 M
788	1303		324	? PT= 10
789	1304		347	GOC LNC40 (1340)
790	1305		1124	? PT= 9
791	1306		477	GOC LNC50 (1355)
792	1307		424	? PT= 8
793	1310		617	GOC LNC60 (1371)
794	1311		1224	? PT= 7
795	1312		717	GOC LNC70 (1403)
796	1313		524	? PT= 6
797	1314		777	GOC LNC80 (1413)
798	1315		1634	PT= 0
799	1316		320	LC 3
800	1317		534	PT= 6
801	1320		153	GOTO LNCEND (1335)
802	1321	LNC30	620	LC 6
803	1322		1120	LC 9
804	1323		320	LC 3
805	1324		120	LC 1
806	1325		420	LC 4
807	1326		720	LC 7
808	1327		120	LC 1
809	1330		1020	LC 8
810	1331		20	LC 0
811	1332		520	LC 5
812	1333		620	LC 6
813	1334		1534	PT= 12
814	1335	LNCEND	356	BC EX W
815	1336		1740	RTN
816	1337		0	NOP
817	1340	LNC40	320	LC 3
818	1341		120	LC 1
819	1342		20	LC 0
820	1343		120	LC 1
821	1344		720	LC 7
822	1345		1120	LC 9

PRESERVE ENTRY POINT ADDRESSES

823	1346	1020	LC	8	
824	1347	20	LC	0	
825	1350	420	LC	4	
826	1351	320	LC	3	
827	1352	320	LC	3	
828	1353	634	PT=	11	
829	1354	1613	GOTO	LNCEND	(1335)
830	1355	LNC50	434	PT=	8
831	1356		320	LC	3
832	1357		320	LC	3
833	1360		20	LC	0
834	1361	1020	LC	8	
835	1362		520	LC	5
836	1363		320	LC	3
837	1364		120	LC	1
838	1365		620	LC	6
839	1366	1020	LC	8	
840	1367		334	PT=	10
841	1370	1453	GOTO	LNCEND	(1335)
842	1371	LNC60	534	PT=	6
843	1372		320	LC	3
844	1373		320	LC	3
845	1374		320	LC	3
846	1375		20	LC	0
847	1376	1020	LC	8	
848	1377		320	LC	3
849	1400		520	LC	5
850	1401	1134	PT=	9	
851	1402	1333	GOTO	LNCEND	(1335)
852	1403	LNC70	134	PT=	4
853	1404		320	LC	3
854	1405		320	LC	3
855	1406		320	LC	3
856	1407		320	LC	3
857	1410		120	LC	1
858	1411		434	PT=	8
859	1412	1233	GOTO	LNCEND	(1335)
860	1413	LNC80	1034	PT=	2
861	1414		320	LC	3
862	1415		320	LC	3
863	1416		320	LC	3
864	1417		1234	PT=	7
865	1420	1153	GOTO	LNCEND	(1335)
866	1421	XY^X	56	B=0	W
867	1422		372	BC EX	M
868	1423		256	AC EX	W
869	1424		1	GOSUB	STSCR
869	1425		0		
870	1426		270	C=REGN	2
871	1427		1	GOSUB	CHK#S
871	1430		0		
872	1431		416	A=C	W
873	1432		370	C=REGN	3
874	1433		1536	? A#0	S
875	1434		273	GONC	YX13 (1463)
876	1435		6	A=0	X
877	1436		272	AC EX	M
878	1437		1366	? C#0	XS
879	1440		63	GONC	YX11 (1446)
880	1441	ERR0*	1	GOLONG	ERR0

880	1442	2			
881	1443	YX12	1346 ? C#0	X	
882	1444		1753 GONC	ERR0*	(1441)
883	1445		1146 C=C-1	X	
884	1446	YX11	1756 A SL	W	
885	1447		1532 ? A#0	M	
886	1450		1737 GOC	YX12	(1443)
887	1451		1346 ? C#0	X	
888	1452		117 GOC	YX13	(1463)
889	1453		276 AC EX	S	
890	1454		436 A=C	S	
891	1455		776 C=C+C	S	
892	1456		776 C=C+C	S	
893	1457		1036 C=A+C	S	
894	1460		1376 ? C#0	S	
895	1461		23 GONC	YX13	(1463)
896	1462		1210 S7=	1	
897	1463	YX13	270 C=REGN	2	
898	1464		136 C=0	S	
899	1465	YXTEN	56 B=0	W	
900	1466		372 BC EX	M	
901	1467		256 AC EX	W	
902	1470	YX31	1410 S1=	1	
903	1471		1332 ? B#0	M	
904	1472		167 GOC	LN13	(1510)
905	1473		1 GOSUB	RCSCR	
905	1474		0		
906	1475		1372 ? C#0	M	
907	1476		1433 GONC	ERR0*	(1441)
908	1477		630 C=M		
909	1500		1376 ? C#0	S	
910	1501		1407 GOC	ERR0*	(1441)
911	1502		56 B=0	W	
912	1503		1 COLONG	NRM13	
912	1504		2		
913	1505	LN10	56 B=0	W	
914	1506		372 BC EX	M	
915	1507		256 AC EX	W	
916	1510	LN13	76 B=0	S	
917	1511		1536 ? A#0	S	
918	1512		1277 GOC	ERR0*	(1441)
919	1513		1332 ? B#0	M	
920	1514		1253 GONC	ERR0*	(1441)
921	1515		246 AC EX	X	
922	1516		406 A=C	X	
923	1517		1346 ? C#0	X	
924	1520		203 GONC	LN220	(1540)
925	1521		506 A=A+C	X	
926	1522		33 GONC	LN140	(1525)
927	1523		1246 C=-C-1	X	
928	1524		10 S3=	1	
929	1525	LN140	16 A=0	W	
930	1526		136 C=0	S	
931	1527		132 C=0	M	
932	1530		1534 PT=	12	
933	1531		612 A=A-B	WPT	
934	1532		1142 C=C-1	PT	
935	1533	LN310	1042 C=C+1	PT	
936	1534	LN300	216 B=A	W	
937	1535		730 MC EX		

938	1536	630	C=M	
939	1537	613	GOTO	LN330 (1620)
940	1540 LN220	1534	PT=	12
941	1541	316	C=8	W
942	1542	1142	C=C-1	PT
943	1543	256	AC EX	W
944	1544	1516	? A#0	W
945	1545	1	GOLNC	LN560
945	1546	2		
946	1547 LN200	1502	? A#0	PT
947	1550	47	GOC	LN210 (1554)
948	1551	1146	C=C-1	X
949	1552	1756	A SL	W
950	1553	1743	GOTO	LN200 (1547)
951	1554 LN210	730	MC EX	
952	1555	1	GOSUB	DIV15
952	1556	0		
953	1557	213	GOTO	LN1+X6 (1600)
954	1560 LN1+X2	1	GOSUB	ADDONE
954	1561	0		
955	1562	1263	GOTO	LN13 (1510)
956	1563 XLN1+X	56	B=0	W
957	1564	372	BC EX	M
958	1565	416	A=C	W
959	1566	256	AC EX	W
960	1567	416	A=C	W
961	1570	1046	C=C+1	X
962	1571	746	C=C+C	X
963	1572	1663	GONC	LN1+X2 (1560)
964	1573	1536	? A#0	S
965	1574	127	GOC	LN1+X3 (1606)
966	1575	256	AC EX	W
967	1576	1	GOSUB	LNSUB
967	1577	0		
968	1600 LN1+X6	1534	PT=	12
969	1601	116	C=0	W
970	1602	246	AC EX	X
971	1603	416	A=C	W
972	1604	156	AB EX	W
973	1605	73	GOTO	LN1+X7 (1614)
974	1606 LN1+X3	10	S3=	1
975	1607	1713	GOTO	LN1+X6 (1600)
976	1610 LN1+X8	524	? PT=	6
977	1611	767	GOC	LN410 (1707)
978	1612	1724	DEC PT	
979	1613	1076	C=C+1	S
980	1614 LN1+X7	1046	C=C+1	X
981	1615	1733	GONC	LN1+X8 (1610)
982	1616	1163	GOTO	LN300 (1534)
983	1617 LN320	1616	A SR	W
984	1620 LN330	1176	C=C-1	S
985	1621	1763	GONC	LN320 (1617)
986	1622	630	C=M	
987	1623	456	A=A+B	W
988	1624	676	A=A-1	S
989	1625	1063	GONC	LN310 (1533)
990	1626	1076	C=C+1	S
991	1627	156	AB EX	W
992	1630	1756	A SL	W
993	1631	1724	DEC PT	

994 1632	224 ? PT=	5	
995 1633	1013 GONC	LN300	(1534)
996 1634	256 AC EX	W	
997 1635	216 B=A	W	
998 1636	1752 A SL	WPT	
999 1637	1752 A SL	WPT	
1000 1640	1752 A SL	WPT	
1001 1641	256 AC EX	W	
1002 1642	1634 PT=	0	
1003 1643	720 LC	?	
1004 1644	1206 C=-C	X	
1005 1645	1306 ? B#0	X	
1006 1646	743 GONC	LN420	(1742)
1007 1647	534 PT=	6	
1008 1650 LN460	1616 A SR	W	
1009 1651 LN430	356 BC EX	W	
1010 1652 LN431	1 GOSUB	LNC20	
1010 1653	0		
1011 1654	1 GOSUB	PMUL	
1011 1655	0		
1012 1656	1372 ? C#0	M	
1013 1657	343 GONC	LN530	(1713)
1014 1660	1324 ? PT=	13	
1015 1661	1673 GONC	LN460	(1650)
1016 1662	56 B=0	W	
1017 1663	1634 PT=	0	
1018 1664	202 B=A	PT	
1019 1665	456 A=A+B	W	
1020 1666	1616 A SR	W	
1021 1667 LN500	1 GOSUB	LNC10*	
1021 1670	0		
1022 1671	14 ?S3=1		
1023 1672	67 GOC	LN570	(1700)
1024 1673	156 AB EX	W	
1025 1674	616 A=A-B	W	
1026 1675	156 AB EX	W	
1027 1676	456 A=A+B	W	
1028 1677	156 AB EX	W	
1029 1700 LN570	34 PT=	3	
1030 1701 LN520	1 GOSUB	PMUL	
1030 1702	0		
1031 1703	1372 ? C#0	M	
1032 1704	73 GONC	LN530	(1713)
1033 1705	1616 A SR	W	
1034 1706	1733 GOTO	LN520	(1701)
1035 1707 LN410	356 BC EX	W	
1036 1710	333 GOTO	LN400	(1743)
1037 1711 LN540	1616 A SR	W	
1038 1712	43 GOTO	LN550	(1716)
1039 1713 LN530	1536 ? A#0	S	
1040 1714	1757 GOC	LN540	(1711)
1041 1715	1146 C=C-1	X	
1042 1716 LN550	136 C=0	S	
1043 1717	14 ?S3=1		
1044 1720	33 GONC	LN560	(1723)
1045 1721	1276 C=-C-1	S	
1046 1722	0 NOP		
1047 1723 LN560	1 GOSUB	SHF10	
1047 1724	0		
1048 1725	1414 ?S1=1		

```

1049 1726      313 GONC   LN580 (1757)
1050 1727      1 GOSUB   RCSCR
1050 1730      0
1051 1731      1 GOSUB   MP2-13
1051 1732      0
1052 1733 YTOX50 630 C=M
1053 1734 YTOX60 1376 ? C#0 S
1054 1735      1 GOLNC   EXP13
1054 1736      2
1055 1737      646 A=A-1 X
1056 1740      356 BC EX W
1057 1741      1733 GOTO   YTOX60 (1734)
1058 1742 LN420 1616 A SR W
1059 1743 LN400 1 GOSUB   LNAP
1059 1744      0
1060 1745      156 AB EX W
1061 1746      534 PT= 6
1062 1747      1033 GOTO   LN431 (1652)
1063 1750 PMUL1 456 A=A+B W
1064 1751 PMUL 1142 C=C-1 PT
1065 1752      1763 GONC   PMUL1 (1750)
1066 1753      102 C=0 PT
1067 1754      1046 C=C+1 X
1068 1755      1734 INC PT
1069 1756      1740 RTN
1070 1757 LN580 214 ?S5=1
1071 1760      1640 RTN NC
1072 1761      116 C=0 W
1073 1762      730 MC EX
1074 1763      1 GOSUB   LNC10
1074 1764      0
1075 1765      356 BC EX W
1076 1766      1 GOLONG DV2-13
1076 1767      2
1077 1770 10TOX 16 A=0 W
1078 1771      1 GSBLNG LNC10
1078 1772      0
1079 1773      260 C=N
1080 1774      1 GSBLNG MP1-10
1080 1775      0
1081 1776      1353 GOTO   YTOX50 (1733)
1082      FILLTO END
1082 1777      0000 NOP
1083      END

```

ERRORS : 0

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

SYMBOL TABLE

1/X10	213	-				
1/X13	216	-				
10TOX	1770	-				
AD1-10	11	-	6			
AD2-10	7	-				
AD2-13	14	-				
ADD10	20	-	24			
ADD30	34	-	41			
ADD40	60	-				
ADD45	42	-				
ADD50	72	-	102	63	57	
ADD55	51	-	45			
ADD60	104	-	73	33	27	
ADD65	55	-	50	35		
ADD90	36	-				
ADDONE	0	-				
DIV100	240	-				
DIV110	245	-	243			
DIV120	257	-	252			
DIV130	262	-	264			
DIV140	263	-	271	261		
DIV15	251	-				
DTOR	601	-				
DV1-10	232	-				
DV2-10	230	-				
DV2-13	235	-	227			
ERR0	303	-	357	355	353	237
ERR0*	1441	-	1514	1512	1501	1476 1444
EXP10	1012	-				
EXP110	1021	-	1053	1017		
EXP120	1050	-	1031			
EXP13	1015	-				
EXP130	1033	-	1040			
EXP200	1044	-	1026			
EXP210	1054	-	1056			
EXP220	1055	-	1100	1047		
EXP230	1074	-	1062			
EXP240	1072	-	1065			
EXP300	1101	-	1070			
EXP400	1041	-	1140			
EXP410	1126	-	1130			
EXP420	1127	-	1043			
EXP430	1137	-	1073			
EXP500	1141	-				
EXP510	1142	-	1133			
EXP520	1165	-	1157			
EXP530	1204	-	1164			
EXP540	1203	-	1205			
EXP550	1153	-	1170			
EXP560	1156	-	1211			
EXP570	1150	-	1143			
EXP600	1237	-	1154			
EXP700	1112	-	1110			
EXP710	1114	-	1202			
EXP720	1120	-	1115			
EXP730	1124	-	1121			

EXP740	1247	-	1244	
EXP750	1253	-	1250	
EXSCR	452	-		
HMS110	674	-	702	
HMS120	677	-		
HMS130	701	-	676	
HMS140	703	-	673	
HMS150	715	-	711	
HMS160	721	-	714	
HMS170	726	-	744	
HMSDV	745	-		
HMSM20	763	-	766	755
HMSMP	747	-		
HRS100	730	-	706	
HRS120	737	-	735	
INT30	505	-	502	
INTFRC	473	-		
LD90	625	-		
LN1+X2	1560	-	1572	
LN1+X3	1606	-	1574	
LN1+X6	1600	-	1607	1557
LN1+X7	1614	-	1605	
LN1+X8	1610	-	1615	
LN10	1505	-		
LN13	1510	-	1562	1472
LN140	1525	-	1522	
LN200	1547	-	1553	
LN210	1554	-	1550	
LN220	1540	-	1520	
LN300	1534	-	1633	1616
LN310	1533	-	1625	
LN320	1617	-	1621	
LN330	1620	-	1537	
LN400	1743	-	1710	
LN410	1707	-	1611	
LN420	1742	-	1646	
LN430	1651	-		
LN431	1652	-	1747	
LN460	1650	-	1661	
LN500	1667	-		
LN520	1701	-	1706	
LN530	1713	-	1704	1657
LN540	1711	-	1714	
LN550	1716	-	1712	
LN560	1723	-	1720	
LN570	1700	-	1672	
LN580	1757	-	1726	
LNAP	1212	-		
LNAP1	1222	-	1231	
LNAP2	1230	-	1224	
LNC10	1256	-		
LNC10*	1255	-		
LNC20	1275	-		
LNC30	1321	-	1277	
LNC40	1340	-	1304	
LNC50	1355	-	1306	
LNC60	1371	-	1310	
LNC70	1403	-	1312	
LNC80	1413	-	1314	
LNCEND	1335	-	1420	1412 1402 1370 1354 1320 1274

LNSUB	771	-		
LNSUB-	770	-		
LNSUB1	775	-	1004	
LNSUB2	1003	-	777	
MOD1	543	-	541	
MOD10	534	-		
MOD2	550	-	544	
MOD3	557	-	564	560
MOD4	571	-	547	
MOD5	545	-	536	
MP1-10	117	-		
MP2-10	115	-		
MP2-13	122	-		
MPY110	131	-	127	
MPY120	134	-	143	
MPY130	137	-	141	
MPY140	149	-	136	
MPY150	145	-	114	
MPY160	150	-		
NRM10	160	-	207	
NRM11	161	-	205	
NRM12	162	-		
NRM13	204	-		
NRM20	202	-	167	165
NRM30	173	-	203	
NRM40	201	-		
PI/2	632	-		
PMUL	1751	-		
PMUL1	1750	-	1752	
RCSCR	464	-		
RCSCR*	462	-		
RTOD	614	-		
SHF10	155	-		
SHF20	206	-	212	157
SHF40	154	-	151	
SINFR	507	-		
SINFR1	522	-	531	
SINFR2	532	-	523	
SINFRA	512	-		
SQR10	276	-		
SQR100	347	-	335	
SQR13	301	-		
SQR20	305	-	302	
SQR30	322	-	320	
SQR50	327	-	325	
SQR60	336	-	340	
SQR70	337	-	350	
STSCR	442	-		
STSCR*	440	-		
STSCR1	450	-	461	
SUBON1	4	-	1	
SUBONE	2	-		
TRC10	641	-		
X/Y13	223	-		
XFT100	354	-		
XFT110	361	-	352	
XFT120	351	-	367	
XFT130	375	-	372	
XFT140	402	-	427	425
XFT150	406	-	403	

XFT160	412	-	413
XFT170	414	-	410
XFT180	421	-	422
XFT190	423	-	415
XLN1+X	1563	-	
XTOHRS	662	-	
XY^X	1421	-	
YTOX50	1733	-	1776
YTOX60	1734	-	1741
YX11	1446	-	1440
YX12	1443	-	1450
YX13	1463	-	1461 1452 1434
YX31	1470	-	
YXTEN	1465	-	

ENTRY TABLE

1/X10	213	-
1/X13	216	-
10TOX	1770	-
AD1-10	11	-
AD2-10	7	-
AD2-13	14	-
ADDONE	0	-
DIV110	245	-
DIV120	257	-
DIV15	251	-
DTOR	601	-
DV1-10	232	-
DV2-10	230	-
DV2-13	235	-
ERR0	303	-
EXP10	1012	-
EXP13	1015	-
EXP400	1041	-
EXP500	1141	-
EXP710	1114	-
EXP720	1120	-
EXSCR	452	-
HMSDV	745	-
HMSMP	747	-
INTERC	473	-
LD90	625	-
LN10	1505	-
LN560	1723	-
LNAP	1212	-
LNC10	1256	-
LNC10*	1255	-
LNC20	1275	-
LNSUB	771	-
LNSUB-	770	-
MOD10	534	-
MP1-10	117	-
MP2-10	115	-
MP2-13	122	-
MPY150	145	-
NRM10	160	-
NRM11	161	-
NRM12	162	-
NRM13	204	-
PI/2	632	-
PNUL	1751	-
RCSCR	464	-
RCSCR*	462	-
RTOD	614	-
SHF10	155	-
SHF40	154	-
SINFR	507	-
SINFRA	512	-
SQR10	276	-
SQR13	301	-
STSCR	442	-
STSCR*	440	-

SUBONE	2	-
TRC10	641	-
X/Y13	223	-
XFT100	354	-
XLN1+X	1563	-
XTOHRS	662	-
XY^X	1421	-

EXTERNAL REFERENCES

1/X13	1200		
1/X13	1201		
AD2-10	577		
AD2-10	600		
ADDONE	1251	1560	
ADDONE	1252	1561	
CHK#S	1427		
CHK#S	1430		
DIV110	1235		
DIV110	1236		
DIV15	1010	1555	
DIV15	1011	1556	
DV1-10	612		
DV1-10	613		
DV2-13	623	1766	
DV2-13	624	1767	
ERR0	1441		
ERR0	1442		
ERRDE	303		
ERRDE	304		
EXP13	1735		
EXP13	1736		
EXP500	1035		
EXP500	1036		
EXP720	1253		
EXP720	1254		
HMSDV	731	737	
HMSDV	732	740	
HMSMP	712	716	722
HMSMP	713	717	723
LD90	610	615	
LD90	611	616	
LN560	1545		
LN560	1546		
LNAP	1144	1743	
LNAP	1145	1744	
LNC10	1044	1763	1771
LNC10	1045	1764	1772
LNC10*	1667		
LNC10*	1670		
LNC20	1041	1652	
LNC20	1042	1653	
LNSUB	1576		
LNSUB	1577		
LNSUB-	1245		
LNSUB-	1246		
MP1-10	606	1774	
MP1-10	607	1775	
MP2-10	617		
MP2-10	620		
MP2-13	1731		
MP2-13	1732		
MPY150	726		
MPY150	727		
NRM10	274		
NRM10	275		

NRM12	344	505	
NRM12	345	506	
NRM13	1124	1503	
NRM13	1125	1504	
PI/2	602	621	
PI/2	603	622	
PMUL	1654	1701	
PMUL	1655	1702	
RCSCR	1473	1727	
RCSCR	1474	1730	
SHF10	476	567	1723
SHF10	477	570	1724
SINFR	473		
SINFR	474		
STSCR	1424		
STSCR	1425		
SUBONE	1116		
SUBONE	1117		
TRC10	634		
TRC10	635		

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

2	FILE	CN7B
* ROM ADDRESSES @16000-17777		
4	ENTRY	GTACOD
5	ENTRY	TOGSHF
6	ENTRY	TGSHF1

* NUT MESSAGE TABLE & MESSAGE ROUTINE *		

10	ENTRY	MSG
11	ENTRY	MSGA
12	ENTRY	MSGE
13	ENTRY	MSGX
14	ENTRY	MSGAD
15	ENTRY	MSGDE
16	ENTRY	MSGML
17	ENTRY	MSGNE
18	ENTRY	MSGNL
19	ENTRY	MSGNO
20	ENTRY	MSGOF
21	ENTRY	MSGPR
22	ENTRY	MSGRAM
23	ENTRY	MSGROM
24	ENTRY	MSGTA
25	ENTRY	MSGYES
26	ENTRY	MSGWR
*		
28	FILLTO:02	PUT SPARE AT BEGINNING
0	0000	NOP
1	0000	NOP
2	0000	NOP
*		
* PATCH9 - POST-RELEASE FIX TO SIGMA+ AND SIGMA- TO GET OLD X		
* PRESERVED IN LASTX.		

```

*
33          ENTRY  PATCH9
34      3 PATCH9 260 C=N          GET NEW X
35      4          1 GOLONG.XCLXI  GO UPDATE X
35      5          2

```

```

*
* PATCH6 - POST-RELEASE FIX TO CHKADR TO PREVENT WRAPAROUND WHEN
* PHYSICAL REGISTER ADDRESS CARRIES INTO THE 10TH OR 11TH BITS.
* WITH THIS PATCH, CHKADR WILL ACCEPT PHYSICAL REGISTER ADDRESSES
* UP THRU 511 ONLY (9 BITS ONLY).

```

```

*
42          ENTRY  PATCH6
43      6 PATCH6 1110 S9= 1          REMEMBER ERROR EXIT TO ERRENE
44      7          26 A=0  XS
45     10          566 A=A+1 XS
46     11          1426 ? A<C XS      511<REG ADDRESS?
47     12          1 GOLC  ERRENE    YES - NO SUCH REG
47     13          3
48     14          1160 DADD=C        ADDRESS THE REGISTER
49     15          1 GOLONG.PERTN
49     16          2

```

```

*
* MESSAGE TABLE

```

```

*
53     17          401 CON  0401      A
54     20          14 CON  014        L
55     21          20 CON  020        P
56     22          10 CON  010        H
57     23          1 CON  001        A
58     24          40 CON  040
59     25          4 CON  004          D
60     26          1 CON  001          A
61     27          24 CON  024          T
62     30 MSGAD    1 CON  001          A
63     31          404 CON  0404      D
64     32          1 CON  001          A
65     33          24 CON  024          T
66     34          1 CON  001          A
67     35          40 CON  040
68     36          5 CON  005          E
69     37          22 CON  022          R
70     40          22 CON  022          R
71     41          17 CON  017          O
72     42 MSGDE    22 CON  022          R
73     43          415 CON  0415      M
74     44          5 CON  005          E
75     45          15 CON  015          M
76     46          17 CON  017          O
77     47          22 CON  022          R
78     50          31 CON  031          Y
79     51          40 CON  040
80     52          14 CON  014          L
81     53          17 CON  017          O
82     54          23 CON  023          S
83     55 MSGML    24 CON  024          T
84     56          416 CON  0416      N
85     57          17 CON  017          O
86     60          16 CON  016          N
87     61          5 CON  005          E
88     62          30 CON  030          X

```

89	63	11	CON	011	I
90	64	23	CON	023	S
91	65	24	CON	024	T
92	66	5	CON	005	E
93	67	16	CON	016	N
94	70	24	CON	024	T
95	71	416	CON	0416	N
96	72	25	CON	025	U
97	73	14	CON	014	L
98	74	14	CON	014	L
99	75	420	CON	0420	P
100	76	22	CON	022	R
101	77	11	CON	011	I
102	100	26	CON	026	V
103	101	1	CON	001	A
104	102	24	CON	024	T
105	103	5	CON	005	E
106	104	417	CON	0417	O
107	105	25	CON	025	U
108	106	24	CON	024	T
109	107	40	CON	040	
110	110	17	CON	017	O
111	111	6	CON	006	F
112	112	40	CON	040	
113	113	22	CON	022	R
114	114	1	CON	001	A
115	115	16	CON	016	N
116	116	7	CON	007	G
117	117	5	CON	005	E
118	120	420	CON	0420	P
119	121	1	CON	001	A
120	122	3	CON	003	C
121	123	13	CON	013	K
122	124	11	CON	011	I
123	125	16	CON	016	N
124	126	7	CON	007	G
125	127	424	CON	0424	T
126	130	22	CON	022	R
127	131	31	CON	031	Y
128	132	40	CON	040	
129	133	1	CON	001	A
130	134	7	CON	007	G
131	135	1	CON	001	A
132	136	11	CON	011	I
133	137	16	CON	016	N
134	140	431	CON	0431	Y
135	141	5	CON	005	E
136	142	23	CON	023	S
137	143	416	CON	0416	N
138	144	17	CON	017	O
139	145	422	CON	0422	R
140	146	1	CON	001	A
141	147	15	CON	015	M
142	150	422	CON	0422	R
143	151	17	CON	017	O
144	152	15	CON	015	M

- *
 * MSG - SEND A MESSAGE TO LCD DISPLAY
 * CALLING MSG WITH S8 SET, MSGFLAG
 * WILL BE SET SO THE DISPLAY WON'T BE REFRESHED BY DISPLAY REFRESH

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer


```

* LOGIC. OTHERWISE, THE DISPLAY WILL BE REFRESHED.
* CALLING SEQUENCE:
*   GOSUB MSGA
*   XDEF <MSGXXX>
* MSG - SET S8 AUTOMATICALLY, THEN DROP TO MSGA
* MSGX - PLUG-IN ROM CAN CALL MSGX TO DISPLAY MESSAGE IN ROM.
*         IF S8= 1, GOSUB PRT6, BLINK LCD, SET MESSAGE FLAG
*         IF S8= 0, DON'T PRINT OR SET MESSAGE FLAG
*         IN: C[6:3]= ADDRESS OF FIRST CHARACTER OF MESSAGE
*         OUT: IF S8= 1: SST 0 UP, MSG FLAG SET, CHIP 0 ENABLED, C= REG 14
*              IF S8= 0: CHIP 0 ENABLED
*         USES: IF S8= 1: A,C,G,N, ST[7:0], ACTIVE PT, 2 ADDITIONAL SUB LEVELS
*              IF S8= 0: A,C, ACTIVE PT, 1 ADDITIONAL SUB LEVEL
*         ASSUME: HEXMODE
*
* MESSAGE TABLE FORMAT:
* EVERY CHAR IN THE MESSAGE COST A 10 BITS WORD TO STORE IT.
* ENTRY OF EACH MESSAGE POINTING LAST CHAR OF THE MESSAGE. THE
* MSG ROUTINE WORKS BACKWARD, IT PICK UP LAST CHAR FIRST AND SHIFT
* IT FROM RIGHT END TO THE DISPLAY, THEN PICK UP NEXT LAST ONE UNTIL
* DONE WITH THE 1ST CHAR WHICH HAS BIT 8 SET.
* CHAR IN THE MESSAGE TABLE IS IN LCD FORM.
*
172 153 MSG      410 S8=      1
173 154 MSGA     660 C=STK
174 155          1140 SETHK
175 156          1460 CXISA
176 157          1072 C=C+1  M
177 160          560 STK=C    POINT TO P+2
178 161 MSGE     674 RCR      11
179 162          534 PT=      6   POINT TO MSG ENTRY
180 163          120 LC        1   IN QUAR 7
181 164          1420 LC       12
182 165 MSGX     416 A=C      W
183 166          1 GOSUB      CLLCDE
183 167          0
184 170          256 AC EX     W
185 171 MSG100   1460 CXISA
186 172          1172 C=C-1  M   LOAD A CHAR
187 173          406 A=C      X   POINT TO NEXT CHAR
188 174          126 C=0     XS
189 175          1650 SRGABC
190 176          1526 ? A#0  XS   IS THIS THE LAST CHAR ?
191 177          1723 GOMC    MSG100 ( 171 ) NO
192
193          ENTRY MSG105      CALLED FROM TIMER ROM
194 200 MSG105    1 GOSUB      ENCF00  ENABLE CHIP 0
194 201          0
195 202          414 ?SRAI
196 203          1640 RTN NO
197 204          1615 CON      01615
198 205          674 CON      0674   GOSUB PRT6
* TO CONSERVE SUBROUTINE LEVELS, THE PRINTER POPS ITS RETURN OFF
* THE STACK AND DOES A GOLONC BACK TO MSG110
201          ENTRY MSG110      FOR THE PRINTER
202          MSG110
203 206          1 GOLONC MSGDLY   DELAY FOR VIEWING MSG
203 207          2
204          AND SET MSGFLG
* STATUS SET 0 IS UP FROM MSGDLY

```

206

EJECT

192

```

207          ENTRY  SIGMA
208          ENTRY  STATCK
209          ENTRY  CHSA
210          ENTRY  BRT140
211          ENTRY  BRT200
212          ENTRY  CHSA1
213          ENTRY  BRT160
214          ENTRY  BRT290
215          ENTRY  TRCS10
216          ENTRY  TOREC
217          ENTRY  ADD1
218          ENTRY  ADD2
219          ENTRY  TRG240
220 210 SIGMA      1 GOSUB  STATCK
220 211              0
221 212              270 C=REGN .2
222 213 SIGMA1     360 CH EX
223 214              260 C=N
224 215              416 A=C      W
225 216              114 ?S4=1
226 217              43 GONC      SIGMA2 ( 223 )
227 220              1 GOSUB  GETX
227 221              0
228 222              33 GOTO    SIGMA3 ( 225 )
229 223 SIGMA2     1 GOSUB  GETY
229 224              0
230 225 SIGMA3     1 GOSUB  CHSA
230 226              0
231 227              1 GOSUB  ADD2
231 230              0
232 231              260 C=N
233 232              416 A=C
234 233              1 GSBLGX MP2-10
234 234              0
235 235              1 GOSUB  CHSA
235 236              0
236 237              114 ?S4=1
237 240              43 GONC      SIGMA4 ( 244 )
238 241              1 GOSUB  GETXSQ
238 242              0
239 243              33 GOTO    SIGMA5 ( 246 )
240 244 SIGMA4     1 GOSUB  GETYSQ
240 245              0
241 246 SIGMA5     1 GOSUB  ADD1
241 247              0
242 250              106 C=0      X
243 251              1160 DADD=C
244 252              370 C=REGN .3
245 253              114 ?S4=1
246 254              37 GOC      SIGMA6 ( 257 )
247 255              110 S4=      1
248 256              1353 GOTO    SIGMA1 ( 213 )
249 257 SIGMA6     416 A=C      W
250 260              270 C=REGN .2
251 261              1 GSBLGX MP2-10
251 262              0
252 263              1 GOSUB  CHSA
252 264              0
253 265              1 GOSUB  GETXY

```

```

253 266      0
254 267      1 GOSUB  ADD1
254 270      0
255 271      16 A=0      W
256 272      542 A=A+1  PT
257 273      1 GSUBNC CHSA
257 274      0
258 275      1 GOSUB  GETN
258 276      0
259 277      1 GOSUB  ADD2
259 300      0
260 301      360 NC EX
261 302      116 C=0      W
262 303      1160 DADD=C
263 304      370 C=REGN 3
264 305      450 REGN=C 4
265 306      1 GOLONG PATCH9
265 307      2

```

GET OLD X
UPDATE LASTX

*THIS SUBROUTINE CHECKS ALL STAT REGISTERS FOR
*ALPHA DATA. IT STARTS AT THE HIGHEST ADDRESS
*AND WORKS DOWN THROUGH THE OTHER FIVE REGISTERS.

```

271 310 STATCK 1 GOSUB  SUMCHK      CX=ADR N B=N
271 311      0
272 312      534 PT=      6
273 313      356 BC EX
274 314 DOCHK  1 GOSUB  CHK#S      IS THIS NUMBER?
274 315      0
275 316      1140 SETHEX      YES
276 317      356 BC EX      GET ADR
277 320      1146 C=C-1  X
278 321      1160 DADD=C      ADR REGISTER
279 322      356 BC EX      SAVE ADR
280 323      70 C=DATA      GET NXT REG
281 324      1724 DEC PT      COUNT DOWN
282 325      1624 ? PT= 0
283 326      1663 GONC  DOCHK  ( 314 ) NO
284 327      116 C=0
285 330      1160 DADD=C
286 331      423 GOTO  GET1  ( 373 )

```

```

288 332 CHSA  214 ?S5=1
289 333      1640 RTN NC
290 334 CHSA1 276 AC EX  S
291 335      1276 C=-C-1 S
292 336      276 AC EX  S
293 337      1740 RTN
294 340 ADD1  1 GSBLGX AD1-10
294 341      0
295 342      33 GOTO  STOVF  ( 345 )
296 343 ADD2  1 GSBLGX AD2-10
296 344      0
297 345 STOVF 1 GSBLGX OVFL10
297 346      0
298 347      1360 DATA=C
299 350      1534 PT= 12
300 351      1740 RTN
301      ENTRY  GETN
302      ENTRY  GETX

```

303			ENTRY	GETXSQ
304			ENTRY	GETY
305			ENTRY	GETYSQ
306			ENTRY	GETXY
307			ENTRY	XBAR
308	352	GETN	1724	DEC PT
309	353	GETXY	1724	DEC PT
310	354	GETYSQ	1724	DEC PT
311	355	GETY	1724	DEC PT
312	356	GETXSQ	1724	DEC PT
313	357	GETX	1140	SETHX
314	360		116	C=0
315	361		1160	DADD=C
316	362		1570	C=REGN 13
317	363		674	RCR 11
318	364	GETADD	1046	C=C+1 X
319	365		1734	INC PT
320	366		1324	? PT= 13
321	367		1753	GONC GETADD (364)
322	370		1146	C=C-1 X
323	371		1160	DADD=C
324	372		70	C=DATA
325	373	GET1	1240	SETDEC
326	374		1534	PT= 12
327	375		1740	RTN
328	376	XBAR	1	GOSUB STATCK
328	377		0	
329	400		1	GOSUB GETY
329	401		0	
330	402		1	GOSUB XBAR*
330	403		0	
331	404		360	CN EX
332	405		1	GOSUB GETX
332	406		0	
333	407	XBAR*	416	A=C W
334	410		1	GOSUB GETN
334	411		0	
335	412		356	BC EX W
336	413		116	C=0 W
337	414		1160	DADD=C
338	415		356	BC EX W
339	416		1	GOLNGX DV2-10
339	417		2	
340			ENTRY	XBAR*
341			ENTRY	SD
342	420	SD	204	S5= 0
343	421		1	GOSUB STATCK
343	422		0	
344	423		1	GOSUB GETYSQ
344	424		0	
345	425	STDEV1	416	A=C W
346	426		1	GOSUB GETN
346	427		0	
347	430		1	GSBLGX MP2-10
347	431		0	
348	432		1	GSBLGX STSCR*
348	433		0	
349	434		214	?S5=1
350	435		47	GOC STDEV4 (441)
351	436		1	GOSUB GETY

```

351 437      0
352 440      33 GOTO STDEV5 ( 443 )
353 441 STDEV4 1 GOSUB GETX
353 442      0
354 443 STDEV5 416 A=C
355 444      1 GSBLGX MP2-10
355 445      0
356 446      1276 C=-C-1 S
357 447      276 AC EX S
358 450      1 GSBLGX RCSCR*
358 451      0
359 452      1 GSBLGX AD2-13
359 453      0
360 454      1 GOSUB GETN
360 455      0
361 456      1 GSBLGX DV1-10
361 457      0
362 460      1 GSBLGX STSCR*
362 461      0
363 462      1 GOSUB GETH
363 463      0
364 464      56 B=0 W
365 465      372 BC EX M
366 466      416 A=C W
367 467      1 GSBLGX SUBONE
367 470      0
368 471      1 GSBLGX RCSCR*
368 472      0
369 473      1 GSBLGX X/Y13
369 474      0
370 475      1376 ? C#0 S
371 476      1 GOLC ERROF
371 477      3
372 500      1 GSBLGX SQR13
372 501      0
373 502      214 ?S5=1
374 503      1540 RTN C
375 504      360 CH EX
376 505      210 S5= 1
377 506      1 GOSUB GETXSO
377 507      0
378 510      1153 GOTO STDEV1 ( 425 )
379      ENTRY BRT100
380      ENTRY TOPOL
381      ENTRY TRC30
382      ENTRY BRTS10
383      ENTRY TRG430
384      ENTRY TRG100
385 511 TOPOL 260 C=N
386 512      1372 ? C#0 M
387 513      463 GONC TOPOL2 ( 561 )
388 514      1376 ? C#0 S
389 515      43 GONC TOPOL1 ( 521 )
390 516      1210 S7= 1
391 517      1610 S0= 1
392 520      136 C=0 S
393 521 TOPOL1 360 NC EX
394 522      416 A=C W
395 523      1 GOSUB MP2-10 CALC X^2
395 524      0

```

```

396 525      1 GOSUB STSCR
396 526      0
397 527      270 C=REGN .2
398 530      416 A=C      W
399 531      1 GOSUB MP2-10      CALC Y^2
399 532      0
400 533      1 GOSUB RCSCR
400 534      0
401 535      1 GOSUB AD2-13      CALC X^2+Y^2
401 536      0
402 537      1 GOSUB SQR13      CALC SQR(X^2+Y^2)
402 540      0
403 541      360 NC EX
404 542      416 A=C      W
405 543      270 C=REGN .2
406 544      256 AC EX      W
407 545      1 GOSUB DV2-10      CALC Y/ABS(X)
407 546      0
408 547      1372 ? C#0      M
409 550      337 GOC      BRT110 ( 603 )
410 551      16 A=0      W
411 552      313 GOTO      BRT110 ( 603 )
412 553 BRTS10 1614 ?S0=1
413 554      37 GOC      BRTS20 ( 557 )
414 555      1610 S0=      1
415 556      1740 RTN
416 557 BRTS20 1604 S0=      0
417 560      1740 RTN
418 561 TOPOL2 256 AC EX      W
419 562      1210 S7=      1
420 563      1376 ? C#0      S
421 564      23 GONC      TOPOL4 ( 566 )
422 565      510 S6=      1
423 566 TOPOL4 136 C=0      S
424 567      360 NC EX
425 570      260 C=N
426 571      1372 ? C#0      M
427 572      1640 RTN NC
428 573 TOPOL3 16 A=0      W
429 574      753 GOTO      BRT301 ( 671 )
430 575 BRT120 1 GOSUB      BRTS10
430 576      0
431 577      1743 GOTO      TOPOL3 ( 573 )
432 600 BRT100 56 B=0      W
433 601      416 A=C      W
434 602      172 AB EX      M
435 603 BRT110 1376 ? C#0      S
436 604      113 GONC      BRT130 ( 615 )
437 605      510 S6=      1
438 606      1014 ?S2=1
439 607      63 GONC      BRT130 ( 615 )
440 610      1614 ?S0=1
441 611      43 GONC      BRT130 ( 615 )
442 612      1604 S0=      0
443 613      1210 S7=      1
444 614      504 S6=      0
445 615 BRT130 1534 PT=      12
446 616      246 AC EX      X
447 617      406 A=C      X
448 620      746 C=C+C      X

```

```

449 621          1 GOLC  BRT140
449 622          3
450 623 BRT150 1506 ? A#0  X
451 624          217 GOC  BRT170 ( 645)
452 625          316 C=B  W
453 626          1316 ? B#0  W
454 627          1443 GONC  TOPOL3 ( 573)
455 630          1534 PT=   12
456 631          1142 C=C-1 PT
457 632          1356 ? C#0  W
458 633          127 GOC  BRT170 ( 645)
459 634          1414 ?S1=1
460 635          1407 GOC  BRT120 ( 575)
461 636          1 GSBLGX TRC10
461 637          0
462 640          16 A=0    W
463 641          646 A=A-1  X
464 642          256 AC EX  W
465 643          1 GOLONG BRT200
465 644          2
466 645 BRT170 1414 ?S1=1
467 646          1 GOLCX  ERR0
467 647          3
468 650 BRT160 1 GSBLGX 1/X13
468 651          0
469 652          1 GOSUB  BRTS10
469 653          0
470 654 BRT290 156 AB EX  W
471 655          316 C=B    W
472 656          1534 PT=   12
473 657          132 C=0    M
474 660          136 C=0    S
475 661 BRT300 1046 C=C+1  X
476 662          1346 ? C#0  X
477 663          103 GONC  BRT310 ( 673)
478 664          1076 C=C+1  S
479 665          1724 DEC PT
480 666          524 ? PT=   6
481 667          1723 GONC  BRT300 ( 661)
482 670          356 BC EX  W
483 671 BRT301 1 GOLONG BRT200
483 672          2
484 673 BRT310 730 MC EX
485 674          116 C=0    W
486 675          1076 C=C+1  S
487 676          1716 C SR  W
488 677          153 GOTO  BRT340 ( 714)
489 700 BRT320 256 AC EX  W
490 701          730 MC EX
491 702          1042 C=C+1  PT
492 703          436 A=C    S
493 704          730 MC EX
494 705 BRT330 1656 B SR  W
495 706          1656 B SR  W
496 707          676 A=A-1  S
497 710          1753 GONC  BRT330 ( 705)
498 711          36 A=0    S
499 712          456 A=A+B  W
500 713          256 AC EX  W
501 714 BRT340 216 B=A    W

```


502	715	716	A=A-C	W
503	716	1623	GONC	BRT320 (700)
504	717	730	MC EX	
505	720	1076	C=C+1	S
506	721	730	MC EX	
507	722	156	AB EX	W
508	723	1756	A SL	W
509	724	1724	DEC PT	
510	725	524	? PT=	6
511	726	1663	GONC	BRT340 (714)
512	727	356	BC EX	W
513	730	1	GSBLGX	DIV120
513	731	0		
514	732	156	AB EX	W
515	733	730	MC EX	
516	734	106	C=0	X
517	735	1234	PT=	7
518	736	356	BC EX	W
519	737	1	GOSUB	TRC30
519	740	0		
520	741	356	BC EX	W
521	742	23	GOTO	BRT370 (744)
522	743	456	A=A+B	W
523	744	1142	C=C-1	PT
524	745	1763	GONC	BRT360 (743)
525	746	1616	A SR	W
526	747	102	C=0	PT
527	750	1372	? C#0	N
528	751	433	GONC	BRT190 (1014)
529	752	1734	INC PT	
530	753	1633	GOTO	BRT350 (736)
531	754	1414	?S1=1	
532	755	343	GONC	BRT141 (1011)
533	756	1	GSBLGX	STSCR
533	757	0		
534	760	1	GSBLGX	ADDONE
534	761	0		
535	762	1	GSBLGX	EXSCR
535	763	0		
536	764	1	GSBLGX	SUBONE
536	765	0		
537	766	1	GSBLGX	RCSCR
537	767	0		
538	770	1	GSBLGX	MP2-13
538	771	0		
539	772	36	A=0	S
540	773	1	GSBLGX	SQR13
540	774	0		
541	775	260	C=N	
542	776	136	C=0	S
543	777	730	MC EX	
544	1000	630	C=N	
545	1001	106	C=0	X
546	1002	1	GSBLGX	X/Y13
546	1003	0		
547	1004	246	AC EX	X
548	1005	406	A=C	X
549	1006	746	C=C+C	X
550	1007	1	GOLMC	BRT160
550	1010	2		

```

551 1011 BRT141      1 GOLONG BRT290
551 1012              2
552 1013 BRT180 1734 INC PT
553 1014 BRT190 1146 C=C-1 X
554 1015          1524 ? PT= 12
555 1016          1753 GONC   BRT180 (1013)
556 1017 BRT200  136 C=0    S
557 1020          1 GSBLGX SHF10
557 1021          0
558 1022          1614 ?S0=1
559 1023          73 GONC    BRT220 (1032)
560 1024          1276 C=-C-1 S
561 1025          276 AC EX  S
562 1026          1 GSBLGX PI/2
562 1027          0
563 1030          1 GSBLGX AD2-13
563 1031          0
564 1032 BRT220 1214 ?S7=1
565 1033          53 GONC    BRT240 (1040)
566 1034          1 GSBLGX PI/2
566 1035          0
567 1036          1 GSBLGX AD2-13
567 1037          0
568 1040 BRT240  114 ?S4=1
569 1041          207 GOC     BRT250 (1061)
570 1042          1 GSBLGX PI/2
570 1043          0
571 1044          546 A=A+1 X
572 1045          546 A=A+1 X
573 1046          0 NOP
574 1047          1 GSBLGX DV2-13
574 1050          0
575 1051          214 ?S5=1
576 1052          77 GOC     BRT250 (1061)
577 1053          116 C=0    W
578 1054          1534 PT=   12
579 1055          1146 C=C-1 X
580 1056          1120 LC     9
581 1057          1 GSBLGX MP1-10
581 1060          0
582 1061 BRT250  514 ?S6=1
583 1062          23 GONC    BRT260 (1064)
584 1063          1276 C=-C-1 S
585 1064 BRT260 1014 ?S2=1
586 1065          1540 RTN C
587 1066          360 NC EX
588 1067          1740 RTN
589 1070 TRC30   116 C=0    W
590 1071          1156 C=C-1 W
591 1072          136 C=0    S
592 1073          1524 ? PT= 12
593 1074          157 GOC    TRC90 (1111)
594 1075          624 ? PT= 11
595 1076          357 GOC    TRC50 (1133)
596 1077          324 ? PT= 10
597 1100          447 GOC    TRC60 (1144)
598 1101          1124 ? PT= 9
599 1102          517 GOC    TRC70 (1153)
600 1103          424 ? PT= 8
601 1104          547 GOC    TRC80 (1160)

```

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

602	1105		1634 PT=	0
603	1106	TRC35	720 LC	7
604	1107	TRC40	1234 PT=	7
605	1110		1740 RTN	
606	1111	TRC90	334 PT=	10
607	1112		620 LC	6
608	1113		620 LC	6
609	1114		1020 LC	8
610	1115		620 LC	6
611	1116		520 LC	5
612	1117		220 LC	2
613	1120		420 LC	4
614	1121		1120 LC	9
615	1122		120 LC	1
616	1123		120 LC	1
617	1124		620 LC	6
618	1125	TRC91	1534 PT=	12
619	1126		1740 RTN	
620	1127	TRCS10	620 LC	6
621	1130		1624 ? PT=	0
622	1131		1763 GONC	TRCS10 (1127)
623	1132		1543 GOTO	TRC35 (1106)
624	1133	TRC50	434 PT=	8
625	1134		1 GOSUB	TRCS10
625	1135		0	
626	1136		1634 PT=	0
627	1137		520 LC	5
628	1140		134 PT=	4
629	1141		1020 LC	8
630	1142		634 PT=	11
631	1143		1740 RTN	
632	1144	TRC60	534 PT=	6
633	1145		1 GOSUB	TRCS10
633	1146		0	
634	1147		1634 PT=	0
635	1150		1120 LC	9
636	1151		334 PT=	10
637	1152		1740 RTN	
638	1153	TRC70	134 PT=	4
639	1154		1 GOSUB	TRCS10
639	1155		0	
640	1156		1134 PT=	9
641	1157		1740 RTN	
642	1160	TRC80	1034 PT=	2
643	1161		1 GOSUB	TRCS10
643	1162		0	
644	1163		434 PT=	8
645	1164		1740 RTN	
646	1165	TOREC	1010 S2=	1
647	1166		1410 S1=	1
648	1167		256 AC EX	W
649	1170	TRG100	16 A=0	W
650	1171		56 B=0	W
651	1172		272 AC EX	M
652	1173		1376 ? C#0	S
653	1174		103 GONC	TRG130 (1204)
654	1175		1210 S7=	1
655	1176		1414 ?S1=1	
656	1177		33 GONC	TRG110 (1202)
657	1200		1614 ?S0=1	

658	1201		23 GONC	TRG120 (1203)
659	1202	TRG110	510 S6=	1
660	1203	TRG120	136 C=0	S
661	1204	TRG130	356 BC EX	W
662	1205		114 ?S4=1	
663	1206		1 GOLC	TRG240
663	1207		3	
664	1210		214 ?S5=1	
665	1211		53 GONC	TRG135 (1216)
666	1212		256 AC EX	W
667	1213		416 A=C	W
668	1214		1716 C SR	W
669	1215		716 A=A-C	W
670	1216	TRG135	116 C=0	W
671	1217		1534 PT=	12
672	1220		420 LC	4
673	1221		520 LC	5
674	1222		356 BC EX	W
675	1223		1146 C=C-1	X
676	1224		1366 ? C#0	XS
677	1225		57 GOC	TRG140 (1232)
678	1226		1146 C=C-1	X
679	1227		33 GONC	TRG140 (1232)
680	1230		1046 C=C+1	X
681	1231		1616 A SR	W
682	1232	TRG140	356 BC EX	W
683	1233	TRG150	730 MC EX	
684	1234		630 C=M	
685	1235		756 C=C+C	W
686	1236		756 C=C+C	W
687	1237		756 C=C+C	W
688	1240		1716 C SR	W
689	1241		356 BC EX	W
690	1242		1366 ? C#0	XS
691	1243		237 GOC	TRG190 (1266)
692	1244	TRG155	616 A=A-B	W
693	1245		1773 GONC	TRG155 (1244)
694	1246		456 A=A+B	W
695	1247		1756 A SL	W
696	1250		1146 C=C-1	X
697	1251		1733 GONC	TRG155 (1244)
698	1252		116 C=0	W
699	1253		356 BC EX	W
700	1254		630 C=M	
701	1255		756 C=C+C	W
702	1256		114 ?S4=1	
703	1257		33 GONC	TRG160 (1262)
704	1260		1616 A SR	W
705	1261		1716 C SR	W
706	1262	TRG160	356 BC EX	W
707	1263	TRG170	616 A=A-B	W
708	1264		133 GONC	TRG190 (1277)
709	1265		456 A=A+B	W
710	1266	TRG180	356 BC EX	W
711	1267		630 C=M	
712	1270		356 BC EX	W
713	1271		114 ?S4=1	
714	1272		413 GONC	TRG270 (1333)
715	1273		1346 ? C#0	X
716	1274		367 GOC	TRG260 (1332)

717	1275		1756 A SL	W
718	1276		353 GOTO	TRG270 (1333)
719	1277	TRG190	1614 ?S0=1	
720	1300		107 GOC	TRG220 (1310)
721	1301		1610 S0=	1
722	1302	TRG200	514 ?S6=1	
723	1303		37 GOC	TRG210 (1306)
724	1304		510 S6=	1
725	1305		1563 GOTO	TRG170 (1263)
726	1306	TRG210	504 S6=	0
727	1307		1543 GOTO	TRG170 (1263)
728	1310	TRG220	1604 S0=	0
729	1311		1414 ?S1=1	
730	1312		1703 GONC	TRG200 (1302)
731	1313		1214 ?S7=1	
732	1314		33 GONC	TRG230 (1317)
733	1315		1204 S7=	0
734	1316		1453 GOTO	TRG170 (1263)
735	1317	TRG230	1210 S7=	1
736	1320		1433 GOTO	TRG170 (1263)
737	1321	TRG240	1 GSBLGX	TRC10
737	1322		0	
738	1323		1103 GOTO	TRG150 (1233)
739	1324	TRG250	156 AB EX	W
740	1325		616 A=A-B	W
741	1326		0 NOP	
742	1327		1 GOSUB	BRTS10
742	1330		0	
743	1331		73 GOTO	TRG280 (1340)
744	1332	TRG260	1046 C=C+1	X
745	1333	TRG270	1366 ? C#0	XS
746	1334		47 GOC	TRG280 (1340)
747	1335		616 A=A-B	W
748	1336		1663 GONC	TRG250 (1324)
749	1337		456 A=A+B	W
750	1340	TRG280	1146 C=C-1	X
751	1341		0 NOP	
752	1342		1 GSBLGX	SHF10
752	1343		0	
753	1344		114 ?S4=1	
754	1345		137 GOC	TRG300 (1360)
755	1346		630 C=M	
756	1347		756 C=C+C	W
757	1350		1146 C=C-1	X
758	1351		0 NOP	
759	1352		1 GSBLGX	DV1-10
759	1353		0	
760	1354		1 GSBLGX	PI/2
760	1355		0	
761	1356		1 GSBLGX	NP2-13
761	1357		0	
762	1360	TRG300	730 MC EX	
763	1361		256 AC EX	W
764	1362		416 A=C	W
765	1363		1046 C=C+1	X
766	1364		73 GONC	TRG310 (1373)
767	1365		156 AB EX	W
768	1366		1756 A SL	W
769	1367		73 GOTO	TRG330 (1376)
770	1370	TRG305	1724 DEC PT	

771	1371		524 ? PT=	6	
772	1372		667 GOC		TRG315 (1460)
773	1373	TRG310	1046 C=C+1	X	
774	1374		1743 GONC		TRG305 (1370)
775	1375		156 AB EX	W	
776	1376	TRG330	116 C=0	W	
777	1377	TRG340	356 BC EX	W	
778	1400		1 GOSUB		TRC30
778	1401		0		
779	1402		356 BC EX	W	
780	1403		23 GOTO		TRG800 (1405)
781	1404	TRG810	1076 C=C+1	S	
782	1405	TRG800	616 A=A-B	W	
783	1406		1763 GONC		TRG810 (1404)
784	1407		456 A=A+B	W	
785	1410		1724 DEC PT		
786	1411		1716 C SR	W	
787	1412		1756 A SL	W	
788	1413		524 ? PT=	6	
789	1414		1633 GONC		TRG340 (1377)
790	1415		730 MC EX		
791	1416		1616 A SR	W	
792	1417		1616 A SR	W	
793	1420		116 C=0	W	
794	1421		1534 PT=	12	
795	1422		120 LC	1	
796	1423		730 MC EX		
797	1424		1634 PT=	0	
798	1425		620 LC	6	
799	1426		620 LC	6	
800	1427		133 GOTO		TRG370 (1442)
801	1430	TRG350	1612 A SR	WPT	
802	1431		1612 A SR	WPT	
803	1432	TRG360	676 A=A-1	S	
804	1433		1753 GONC		TRG350 (1430)
805	1434		36 A=0	S	
806	1435		730 MC EX		
807	1436		256 AC EX	W	
808	1437		1116 C=A-C	W	
809	1440		456 A=A+B	W	
810	1441		730 MC EX		
811	1442	TRG370	216 B=A	W	
812	1443		436 A=C	S	
813	1444		1142 C=C-1	PT	
814	1445		1653 GONC		TRG360 (1432)
815	1446		256 AC EX	W	
816	1447		1772 A SL	M	
817	1450		256 AC EX	W	
818	1451		1372 ? C#0	M	
819	1452		413 GONC		TRG400 (1513)
820	1453		1176 C=C-1	S	
821	1454		1146 C=C-1	X	
822	1455		36 A=0	S	
823	1456		1616 A SR	W	
824	1457		1633 GOTO		TRG370 (1442)
825	1460	TRG315	630 C=M		
826	1461		1014 ?S2=1		
827	1462		517 GOC		TRG430 (1533)
828	1463		1614 ?S0=1		
829	1464		473 GONC		TRG430 (1533)

```

830 1465      1 GSBLGX 1/X13
830 1466      0
831 1467      443 GOTO   TRG430 (1533)
832 1470 TOREC1 260 C=N
833 1471      730 MC EX
834 1472      630 C=M
835 1473      106 C=0   X
836 1474      1 GSBLGX X/Y13
836 1475      0
837 1476      360 NC EX
838 1477      1 GSBLGX RCSCR
838 1500      0
839 1501      1 GSBLGX MP2-13
839 1502      0
840 1503      1614 ?S0=1
841 1504      23 GONC   TOREC2 (1506)
842 1505      360 NC EX
842 1506 TOREC2 1214 ?S7=1
844 1507      23 GONC   TOREC3 (1511)
845 1510      1276 C=-C-1 S
846 1511 TOREC3 360 NC EX
847 1512      413 GOTO   TRG500 (1553)
848 1513 TRG400 136 C=0   S
849 1514      730 MC EX
850 1515      256 AC EX   W
851 1516      630 C=M
852 1517      656 A=A-1   W
853 1520      1014 ?S2=1
854 1521      37 GOC     TRG415 (1524)
855 1522      1614 ?S0=1
856 1523      37 GOC     TRG420 (1526)
857 1524 TRG415 1206 C=-C   X
858 1525      156 AB EX   W
859 1526 TRG420 1332 ? B#0 M
860 1527      303 GONC   TRG440 (1557)
861 1530      730 MC EX
862 1531      1 GSBLGX DIV15
862 1532      0
863 1533 TRG430 1414 ?S1=1
864 1534      173 GONC   TRG500 (1553)
865 1535      1 GSBLGX STSCR
865 1536      0
866 1537      1 GSBLGX RCSCR
866 1540      0
867 1541      1 GSBLGX MP2-13
867 1542      0
868 1543      1 GSBLGX ADDONE
868 1544      0
869 1545      1 GSBLGX SOR13
869 1546      0
870 1547      1014 ?S2=1
871 1550      1207 GOC   TOREC1 (1470)
872 1551      1 GSBLGX 1/X13
872 1552      0
873 1553 TRG500 514 ?S6=1
874 1554      1640 RTN NC
875 1555      1276 C=-C-1 S
876 1556      1740 RTN
877 1557 TRG440 116 C=0   W
878 1560      1534 PT=    12

```

```

879 1561      1152 C=C-1  WPT
880 1562      126 C=0   XS
881 1563      416 A=C   W
882 1564      216 B=A   W
883 1565      1414 ?S1=1
884 1566      1640 RTN NC
885 1567      1443 GOTO   TRG430 (1533)
886 1570      0 NOP
887           ENTRY  TODEC
888           ENTRY  TOOCT

```

PRESERVE ENTRY POINT ADDRESSES

```

*****
*   IF S4=1, THEN DOING TO DECIMAL      *
*   IF S4=0, THEN DOING TO OCTAL      *
*****

```

```

893 1571 TOOCT      1 GSBLGX INTERC
893 1572           0
894 1573           1372 ? C#0  M
895 1574           1 GOLCX  ERR0
895 1575           3
896 1576           260 C=N
897 1577           114 ?S4=1
898 1600           637 GOC    TODEC (1663)
899 1601           416 A=C    W
900 1602           36 A=0    S
901 1603           116 C=0    W
902 1604           1534 PT=   12
903 1605           1146 C=C-1 X
904 1606           320 LC     3
905 1607           1 GSBLGX AD2-10
905 1610           0
906 1611           116 C=0    W
907 1612           120 LC     1
908 1613           20 LC     0
909 1614           720 LC     7
910 1615           320 LC     3
911 1616           720 LC     7
912 1617           420 LC     4
913 1620           120 LC     1
914 1621           1020 LC    8
915 1622           220 LC     2
916 1623           420 LC     4
917 1624           1634 PT=   0
918 1625           1120 LC    9
919 1626           1 GSBLGX DV1-10
919 1627           0
920 1630           1366 ? C#0  XS
921 1631           1 GOLNCX ERR0
921 1632           2
922 1633           340 SEL Q
923 1634           434 PT=   8
924 1635           240 SEL P
925 1636           1634 PT=   0
926 1637           156 AB EX  W
927 1640           23 GOTO    TOOCT2 (1642)
928 1641 TOOCT1     1616 A SR  W
929 1642 TOOCT2     1046 C=C+1 X
930 1643           1763 GONC  TOOCT1 (1641)
931 1644           116 C=0    W
932 1645           256 AC EX  W
933 1646 TOOCT3     756 C=C+C  W

```


934	1647		756	C=C+C	W
935	1650		756	C=C+C	W
936	1651		1756	A SL	W
937	1652		1374	RCR	13
938	1653		240	SEL P	
939	1654		242	AC EX	PT
940	1655		340	SEL Q	
941	1656		1716	C SR	W
942	1657		1724	DEC PT	
943	1660		1524	? PT=	12
944	1661		1653	GONC	TOOCT3 (1646)
945	1662		423	GOTO	TODEC6 (1724)
946	1663	TODEC	1634	PT=	0
947	1664		102	C=0	PT
948	1665		1346	? C#0	X
949	1666		1	GOLCX	ERR0
949	1667		3		
950	1670		260	C=N	
951	1671	TODEC1	1042	C=C+1	PT
952	1672		37	GOC	TODEC2 (1675)
953	1673		1732	C SR	M
954	1674		1753	GOTO	TODEC1 (1671)
955	1675	TODEC2	256	AC EX	W
956	1676		642	A=A-1	PT
957	1677		116	C=0	W
958	1700		56	B=0	W
959	1701		1534	PT=	12
960	1702		1020	LC	8
961	1703		1534	PT=	12
962	1704		356	BC EX	W
963	1705	TODEC7	1442	? A<B	PT
964	1706		47	GOC	TODEC4 (1712)
965	1707		1	GOLONG	ERR0
965	1710		2		
966	1711	TODEC3	1056	C=C+1	W
967	1712	TODEC4	642	A=A-1	PT
968	1713		1763	GONC	TODEC3 (1711)
969	1714		646	A=A-1	X
970	1715		67	GOC	TODEC5 (1723)
971	1716		756	C=C+C	W
972	1717		756	C=C+C	W
973	1720		756	C=C+C	W
974	1721		1772	A SL	M
975	1722		1633	GOTO	TODEC7 (1705)
976	1723	TODEC5	256	AC EX	W
977	1724	TODEC6	260	C=N	
978	1725		112	C=0	WPT
979	1726		1434	PT=	1
980	1727		120	LC	1
981	1730		220	LC	2
982	1731		1	GOLNGX	SHF10
982	1732		2		

*
 * GTACOD - GET ALPHACODE[KEYCODE]
 * GETS THE ALPHAMODE DEFAULT FUNCTION TABLE ENTRY FOR THE
 * CURRENT KEY. USED BY NAMEA AND STK SECTIONS OF PARSE.
 * ENTRY CONDITIONS: CHIP 0 ON, LOGICAL KEYCODE IN N[2:1]
 * USES A,X & C.
 * RETURNS ALPHACODE[KEYCODE] IN C,X
 *

```

991 1733 GTACOD 260 C=N
992 1734 406 A=C X
993 1735 116 C=0
994 1736 460 LDI
995 1737 525 CON 0525 H1550\16=0525
996 1740 1574 RCR 12
997 1741 1006 C=A+C X
998 1742 1574 RCR 12
999 1743 1460 CXISA
1000 1744 1740 RTN
1001
1002
1003

```

```

*
* TOGSHF - TOGGLE SHIFT FLAG
*
* USES C AND 1 SUBROUTINE LEVEL. LEAVES CHIP 0 ENABLED.
*
* TGSHF1 - SAME AS TOGSHF EXCEPT REQUIRES CHIP 0 ENABLED ON ENTRY.
*

```

```

1011 1745 TOGSHF 1 GOSUB ENCR00
1011 1746 0
1012 1747 TGSHF1 1670 C=REGN 14
1013 1750 1074 RCR 2
1014 1751 1730 CST EX PUT UP SS 1
1015 1752 1614 ?S0=1 SHIFT?
1016 1753 37 GOC TOG10 (1756) YES
1017 1754 1610 S0= 1 NO, SET SHIFT.
1018 1755 23 GOTO TOG20 (1757)
1019 1756 TOG10 1604 S0= 0 CLEAR SHIFT
1020 1757 TOG20 1730 CST EX
1021 1760 1574 RCR 12
1022 1761 1650 REGN=C 14
1023 1762 1740 RTN
1024 ENTRY APND-
1025 ENTRY APND10
1026 ENTRY APNDDG
1027 1763 APND- 460 LDI
1028 1764 55 CON 055
1029 1765 APND10 1634 PT= 0
1030 1766 APND15 130 G=C
1031 1767 240 SEL P
1032 1770 1 GO LONG APNDNM
1032 1771 2
1033 1772 APNDDG 630 C=M
1034 1773 1734 INC PT
1035 1774 320 LC 3
1036 1775 1713 GOTO APND15 (1766)
1037
1038

```

```

* RESERVE 2 WORDS AT THE END OF CN7 FOR CHIP 1 CHECKSUM AND
* TRAILER.

```

```

1041 FILLTO @1775
1042 1776 REVLV1 6 CON 6 REV LEVEL= F
1043 1777 CKSUM1 0 CON @0000
1044 END

```

```

ERRORS : 0

```

SYMBOL TABLE

ADD1	340	-			
ADD2	343	-			
APND-	1763	-			
APND10	1765	-			
APND15	1766	-	1775		
APNDDG	1772	-			
BRT100	600	-			
BRT110	603	-	552	550	
BRT120	575	-	635		
BRT130	615	-	611	607	604
BRT140	754	-			
BRT141	1011	-	755		
BRT150	623	-			
BRT160	650	-			
BRT170	645	-	633	624	
BRT180	1013	-	1016		
BRT190	1014	-	751		
BRT200	1017	-			
BRT220	1032	-	1023		
BRT240	1040	-	1033		
BRT250	1061	-	1052	1041	
BRT260	1064	-	1062		
BRT290	654	-			
BRT300	661	-	667		
BRT301	671	-	574		
BRT310	673	-	663		
BRT320	700	-	716		
BRT330	705	-	710		
BRT340	714	-	726	677	
BRT350	736	-	753		
BRT360	743	-	745		
BRT370	744	-	742		
BRTS10	553	-			
BRTS20	557	-	554		
CHSA	332	-			
CHSA1	334	-			
CKSUM1	1777	-			
DOCHK	314	-	326		
GET1	373	-	331		
GETADD	364	-	367		
GETN	352	-			
GETX	357	-			
GETXSQ	356	-			
GETXY	353	-			
GETY	355	-			
GETYSQ	354	-			
GTACOD	1733	-			
MSG	153	-			
MSG100	171	-	177		
MSG105	200	-			
MSG110	206	-			
MSGA	154	-			
MSGAD	30	-			
MSGDE	42	-			
MSGE	161	-			
MSGML	55	-			

MSGNE	70	-	
MSGNL	74	-	
MSGNO	144	-	
MSGOF	117	-	
MSGPR	103	-	
MSGRAM	147	-	
MSGROM	152	-	
MSGTA	137	-	
MSGWR	126	-	
MSGX	165	-	
MSGYES	142	-	
PATCH6	6	-	
PATCH9	3	-	
REVLV1	1776	-	
SD	420	-	
SIGMA	210	-	
SIGMA1	213	-	256
SIGMA2	223	-	217
SIGMA3	225	-	222
SIGMA4	244	-	240
SIGMA5	246	-	243
SIGMA6	257	-	254
STATCK	310	-	
STDEV1	425	-	510
STDEV4	441	-	435
STDEV5	443	-	440
STOVF	345	-	342
TGSHF1	1747	-	
TODEC	1663	-	1600
TODEC1	1671	-	1674
TODEC2	1675	-	1672
TODEC3	1711	-	1713
TODEC4	1712	-	1706
TODEC5	1723	-	1715
TODEC6	1724	-	1662
TODEC7	1705	-	1722
TOG10	1756	-	1753
TOG20	1757	-	1755
TOGSHF	1745	-	
TOOCT	1571	-	
TOOCT1	1641	-	1643
TOOCT2	1642	-	1640
TOOCT3	1646	-	1661
TOPOL	511	-	
TOPOL1	521	-	515
TOPOL2	561	-	513
TOPOL3	573	-	627
TOPOL4	566	-	564
TOREC	1165	-	
TOREC1	1470	-	1550
TOREC2	1506	-	1504
TOREC3	1511	-	1507
TRC30	1070	-	
TRC35	1106	-	1132
TRC40	1107	-	
TRC50	1133	-	1076
TRC60	1144	-	1100
TRC70	1153	-	1102
TRC90	1160	-	1104
TRC90	1111	-	1074

577

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

TRC91	1125	-	
TRCS10	1127	-	1131
TRG100	1170	-	
TRG110	1202	-	1177
TRG120	1203	-	1201
TRG130	1204	-	1174
TRG135	1216	-	1211
TRG140	1232	-	1227 1225
TRG150	1233	-	1323
TRG155	1244	-	1251 1245
TRG160	1262	-	1257
TRG170	1263	-	1320 1316 1307 1305
TRG180	1266	-	1243
TRG190	1277	-	1264
TRG200	1302	-	1312
TRG210	1306	-	1303
TRG220	1310	-	1300
TRG230	1317	-	1314
TRG240	1321	-	
TRG250	1324	-	1336
TRG260	1332	-	1274
TRG270	1333	-	1276 1272
TRG280	1340	-	1334 1331
TRG300	1360	-	1345
TRG305	1370	-	1374
TRG310	1373	-	1364
TRG315	1460	-	1372
TRG330	1376	-	1367
TRG340	1377	-	1414
TRG350	1430	-	1433
TRG360	1432	-	1445
TRG370	1442	-	1457 1427
TRG400	1513	-	1452
TRG415	1524	-	1521
TRG420	1526	-	1523
TRG430	1533	-	1567 1467 1464 1462
TRG440	1557	-	1527
TRG500	1553	-	1534 1512
TRG800	1405	-	1403
TRG810	1404	-	1406
XBAR	376	-	
XBAR*	407	-	

ENTRY TABLE

ADD1	340	-
ADD2	343	-
APND-	1763	-
APND10	1765	-
APNDDG	1772	-
BRT100	600	-
BRT140	754	-
BRT160	650	-
BRT200	1017	-
BRT290	654	-
BRTS10	553	-
CHSA	332	-
CHSA1	334	-
GETN	352	-
GETX	357	-
GETXSQ	356	-
GETXY	353	-
GETY	355	-
GETYSQ	354	-
GTACOD	1733	-
MSG	153	-
MSG105	200	-
MSG110	206	-
MSGA	154	-
MSGAD	30	-
MSGDE	42	-
MSGE	161	-
MSGML	55	-
MSGNE	70	-
MSGNL	74	-
MSGNO	144	-
MSGOF	117	-
MSGPR	103	-
MSGRAM	147	-
MSGROM	152	-
MSGTA	137	-
MSGWR	126	-
MSGX	165	-
MSGYES	142	-
PATCH6	6	-
PATCH9	3	-
SD	420	-
SIGMA	210	-
STATCK	310	-
TGSHF1	1747	-
TODEC	1663	-
TOGSHF	1745	-
TOOCT	1571	-
TOPOL	511	-
TOREC	1165	-
TRC30	1070	-
TRCS10	1127	-
TRG100	1170	-
TRG240	1321	-
TRG430	1533	-
XBAR	376	-

XBAR* 407 -

EXTERNAL REFERENCES

1/X13	650	1465	1551		
1/X13	651	1466	1552		
AD1-10	340				
AD1-10	341				
AD2-10	343	1607			
AD2-10	344	1610			
AD2-13	452	535	1030	1036	
AD2-13	453	536	1031	1037	
ADD1	246	267			
ADD1	247	270			
ADD2	227	277			
ADD2	230	300			
ADDONE	760	1543			
ADDONE	761	1544			
APNDNW	1770				
APNDNW	1771				
BRT140	621				
BRT140	622				
BRT160	1007				
BRT160	1010				
BRT200	643	671			
BRT200	644	672			
BRT290	1011				
BRT290	1012				
BRTS10	575	652	1327		
BRTS10	576	653	1330		
CHK#S	314				
CHK#S	315				
CHSA	225	235	263	273	
CHSA	226	236	264	274	
CLLCDE	166				
CLLCDE	167				
DIV120	730				
DIV120	731				
DIV15	1531				
DIV15	1532				
DV1-10	456	1352	1626		
DV1-10	457	1353	1627		
DV2-10	416	545			
DV2-10	417	546			
DV2-13	1047				
DV2-13	1050				
ENCP00	200	1745			
ENCP00	201	1746			
ERR0	646	1574	1631	1666	1707
ERR0	647	1575	1632	1667	1710
ERRNE	12				
ERRNE	13				
ERROF	476				
ERROF	477				
EXSCR	762				
EXSCR	763				
GETN	275	410	426	454	462
GETN	276	411	427	455	463
GETX	220	405	441		
GETX	221	406	442		

GETXSQ	241	506				
GETXSQ	242	507				
GETXY	265					
GETXY	266					
GETY	223	400	436			
GETY	224	401	437			
GETYSQ	244	423				
GETYSQ	245	424				
INTFRC	1571					
INTFRC	1572					
MP1-10	1057					
MP1-10	1060					
MP2-10	233	261	430	444	523	531
MP2-10	234	262	431	445	524	532
MP2-13	770	1356	1501	1541		
MP2-13	771	1357	1502	1542		
MSGDLY	206					
MSGDLY	207					
OVFL10	345					
OVFL10	346					
P6RTN	15					
P6RTN	16					
PATCH9	306					
PATCH9	307					
PI/2	1026	1034	1042	1354		
PI/2	1027	1035	1043	1355		
RCSCR	533	766	1477	1537		
RCSCR	534	767	1500	1540		
RCSCR*	450	471				
RCSCR*	451	472				
SHF10	1020	1342	1731			
SHF10	1021	1343	1732			
SQR13	500	537	773	1545		
SQR13	501	540	774	1546		
STATCK	210	376	421			
STATCK	211	377	422			
STSCR	525	756	1535			
STSCR	526	757	1536			
STSCR*	432	460				
STSCR*	433	461				
SUBONE	467	764				
SUBONE	470	765				
SUMCHK	310					
SUMCHK	311					
TRC10	636	1321				
TRC10	637	1322				
TRC30	737	1400				
TRC30	740	1401				
TRCS10	1134	1145	1154	1161		
TRCS10	1135	1146	1155	1162		
TRG240	1206					
TRG240	1207					
X/Y13	473	1002	1474			
X/Y13	474	1003	1475			
XBAR*	402					
XBAR*	403					
XCLX1	4					
XCLX1	5					

End of VASM assembly

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES @20000-21777

*

4	FILE	CN88
5	ENTRY	GSUBS1
6	ENTRY	XBST
7	ENTRY	XSST
8	ENTRY	XDELET
9	ENTRY	CLRPGM
10	ENTRY	UPLINK
11	ENTRY	GTLINK
12	ENTRY	GEHLNK
13	ENTRY	GTLNKA
14	ENTRY	INSSUB
15	ENTRY	DELNNH
16	ENTRY	DELLIN
17	ENTRY	PTLINK
18	ENTRY	PTLNKA
19	ENTRY	PTLNKB
20	ENTRY	PTBYTA
21	ENTRY	PTBYTP
22	ENTRY	PUTPC
23	ENTRY	PUTPCF
24	ENTRY	PUTPCX
25	ENTRY	PUTPCA
26	ENTRY	SKPDEL
27		
28		
29		
30		
31	ENTRY	CLRREG
32	ENTRY	ERR110
33	ENTRY	ERROR
34	ENTRY	ERR120
35	ENTRY	GTFEN1
36	ENTRY	GTFEND
37	ENTRY	MOVREG
38	ENTRY	PACKE
39	ENTRY	PACKN
40	ENTRY	PAK200
41	ENTRY	PAKEND
42	ENTRY	PAK6PC
43	ENTRY	PKIOAS
44	ENTRY	XCOPY
45	ENTRY	XPACK

*

* PACK - PACK MEMORY INCLUDING I/O AREA AND KEY ASSIGNMENT AREA

*

* PACKN - NORMAL PACKING SUBROUTINE

* PACKE - FATAL PACKING, AFTER PACKING WOULD SAY "TRY AGAIN"

* EXITS TO ERROR INSTEAD OF RETURNING

* XPACK - SET PUSHFLAG THEN NORMAL PACKING

*

* USED A,B,C,M,N,G STATUS 0-9, THREE LEVEL SUB.

* DURING PACKING USED M,N AS COUNTER :

* M(3:0) - NEXT PACKED BYTE ADDR

* M(7:4) - LAST PACKED END OR ALBL ADDR
 * N(3:0) - LAST PICKING UP BYTE ADDR
 *
 * EXITS VIA DECOMPILE ENTRIES DCPL00 OR DCPLRT.
 * EVENTUALLY RETURNS (EXCEPT PACKE) WITH CHIP 0 ENABLED AND
 * STATUS SET 0 UP.
 *

64		XPACK				
65	0	PACKN	1104	S9=	0	NORMAL PACKING ENTRY
66	1		23	GOTO	PACK (3)	
67	2	PACKE	1110	S9=	1	FATAL PACKING ENTRY
68	3	PACK	1	GOSUB	MSG	SAY "PACKING"
68	4		0			
69	5		0	XDEF	MSGWR	
70	6		1	GOSUB	RSTMS0	ENABLE CHIP 0 AND CLR MSGFLG
70	7		0			
71	10		1	GOSUB	PK10AS	PACK 10 BUFFER AREA
71	11		0			
72	12		116	C=0	W	
73	13		530	M=C		INDICATE START FROM CHAIN END
74	14		104	S4=	0	
75	15		1210	S7=	1	
76	16		1	GOSUB	GTFEND	GET FINAL END
76	17		0			
77	20		1	GOSUB	STBT31	RESET PACK BIT
77	21		0			
78	22		1360	DATA=C		
79	23		263	GOTO	PAK108 (51)	
80	24	PAK100	1	GOSUB	UPLINK	
80	25		0			
81	26		212	B=A	WPT	
82	27		1076	C=C+1	S	IS THIS AN "END" ?
83	30		43	GONC	PAK105 (34)	YES
84	31	PAK102	1346	? C#0	X	REACH CHAIN END ?
85	32		1727	GOC	PAK100 (24)	NOT YET
86	33		253	GOTO	PAK110 (60)	
87	34	PAK105	252	C=A	WPT	SAVE 1ST BYTE ADDR OF END IN N
87	35		412			
88	36		160	N=C		
89	37		1	GOSUB	INCA2	BY PASS THE LINK
89	40		0			
90	41		1176	C=C-1	S	
91	42		1574	RCR	12	C(0:1) _ THIRD BYTE
92	43		1	GOSUB	STBT30	RESET THE PACK BIT
92	44		0			
93	45		1	GOSUB	PTBYTA	
93	46		0			
94	47		260	C=N		GET ADDR OF END
95	50		252	AC EX	WPT	
96	51	PAK108	1	GOSUB	GTLINK	
96	52		0			
97	53		1563	GOTO	PAK102 (31)	
98	54	PAK120	1	GOSUB	FSTIN	GET TOP MEM ADDR -1
98	55		0			
99	56		116	C=0	W	PACKING START FROM TOP
100	57		133	GOTO	PAK115 (72)	
101	60	PAK110	414	?S8=1		DOES 1ST PGM NEEDS PACKING ?
102	61		1737	GOC	PAK120 (54)	YES
103	62		630	C=N		
104	63		1352	? C#0	WPT	ANY PGM NEEDS PACKING ?

```

105 64      1 GOLNC DCPLRT      NONE OF THE PGM NEEDS PACKING
105 65      2
106
107 66      412 A=C      WPT      EXIT VIA DCPLRT IN DECOMPILE
108 67      374 RCR      10      AC(3:0) _ STARTING ADDR
109 70      1 GOSUB      INCAD2    CK(7:4) _ STARTING ADDR
109 71      0      POINT TO THIRD BYTE OF END
110 72 PAK115 1 GOSUB      INCADA    POINT TO NEXT PACKING ADDR
110 73      0
111 74      252 C=A      WPT
111 75      412
112 76      530 M=C      SET THE TWO ADDRS IN M
113 77      1 GOSUB      DECADA    BACKUP ONE BYTE
113 100     0
114 101     212 B=A      WPT
* DECIDE WHETHER WE NEED TO ADJUST THE PC WHILE PACKING
116 102     1 GOSUB      GETPC
116 103     0
117 104     314 ?S10=1      ARE WE IN ROM ?
118 105     127 GOC      PAK117 ( 117) YES, DON'T TOUCH PC AT ALL
119 106     630 C=M      C_ADDR TO START PACKING
120 107     1406 ? A<C      X      PC<START ADDR?
121 110     107 GOC      PAK118 ( 120) YES, NEED TO ADJUST PC
122 111     1546 ? A#C      X      PC>START ADDR?
123 112     57 GOC      PAK117 ( 117) YES, DON'T TOUCH PC
124 113     1402 ? A<C      PT      PC<START ADDR?
125 114     47 GOC      PAK118 ( 120) YES, NEED TO ADJUST PC
126 115     1542 ? A#C      PT      PC>START ADDR?
127 116     23 GONC      PAK118 ( 120) NO, MUST ADJUST PC
128 117 PAK117 12 A=0      WPT      LEAVE THE PGMPC ALONE
129 120 PAK118 256 AC EX
130 121     374 RCR      10      CK(7:4) _ OLD PC
131 122     312 C=B      WPT
132 123     160 N=C
133 124 PAK130 404 S8=      0      REMEMBER LAST LINE NOT A D.E.
*
135 125 PAK200 260 C=N      CK(3:0) _ STARTING PICK UP ADDR
136 126     412 A=C      WPT
137 127 PAK210 212 B=A      WPT
138 130     260 C=N
139 131     312 C=B      WPT
140 132     160 N=C
141 133     1 GOSUB      NXBYTA    PICK UP NEXT BYTE
141 134     0
142 135     1574 RCR      12      CHECK IF IT IS A NULL ?
143 136     1342 ? C#0      PT
144 137     37 GOC      PAK220 ( 142) NOT A NULL
145 140     1366 ? C#0      XS
146 141     1663 GONC      PAK210 ( 127) SKIP A NULL
147 142 PAK220 1142 C=C-1      PT      IS IT A ROW 0 FC ?
148 143     277 GOC      PAK250 ( 172) YES
149 144     1142 C=C-1      PT      IS IT A ROW 1 FC ?
150 145     253 GONC      PAK250 ( 172) NO
151 146     766 C=C+C      XS      IS COLUMN # <= 7 ?
152 147     53 GONC      PAK230 ( 154) YES IT IS A D.E.
153 150     766 C=C+C      XS      IS COLUMN # <= 11 ?
154 151     33 GONC      PAK230 ( 154) YES, IT IS A D.E.
155 152     1366 ? C#0      XS      IS IT A CHS ?
156 153     177 GOC      PAK250 ( 172) NO
157 154 PAK230 414 ?S8=1      PREVIOUS LINE A D.E. ?

```

158	155	133 GONC	PAK240 (170)	NO	
159	156	630 C=M			LOAD NEXT PACKED ADDR
160	157	412 A=C	WPT		
161	160	106 C=0	X		
162	161	1 GOSUB	PTBYTA		STORE A NULL BETWEEN D.E.
162	162	0			
163	163	1 GOSUB	INCADA		POINT TO NEXT PACKED ADDR
163	164	0			
164	165	630 C=M			
165	166	252 AC EX	WPT		
166	167	530 M=C			
167	170	PAK240 410 S8=	1		REMEMBER THIS LINE IS D.E.
168	171	23 GOTO	PAK260 (173)		REMEMBER THIS LINE NOT A D.E.
169	172	PAK250 404 S8=	0		
170	173	PAK260 260 C=N			
171	174	412 A=C	WPT		A(3:0) _ POSSIBLE NEW PC
172	175	174 RCR	4		C(3:0) _ OLD PC
173	176	1406 ? A<C	X		PASS OLD PC ALREADY?
174	177	417 GOC	PAK280 (240)		YES, SET NEW PC
175	200	1546 ? A#C	X		SAME REG ?
176	201	57 GOC	PAK265 (206)		NO, NOT CLOSE TO PC YET
177	202	1402 ? A<C	PT		PASS PC BYTE ?
178	203	357 GOC	PAK280 (240)		YES, SET NEW PC
179	204	1542 ? A#C	PT		SAME BYTE ?
180	205	333 GONC	PAK280 (240)		YES, SET NEW PC
181	206	PAK265 1 GOSUB	NXLDEL		GET NEXT LINE
181	207	0			
182	210	260 C=N			C(3:0) _ LAST PICK UP ADDR
183	211	252 AC EX	WPT		C(3:0) _ END PICK UP ADDR
184	212	160 N=C			N(3:0) _ END PICK UP ADDR
185	213	PAK270 1 GOSUB	NXBYTA		PICK UP NEXT BYTE
185	214	0			
186	215	352 BC EX	WPT		SAVE THE BYTE IN B
187	216	630 C=M			C(3:0) _ NEXT PACKED ADDR
188	217	1160 DADD=C			
189	220	1434 PT=	1		
190	221	1 GOSUB	PTBYTP		PACK ONE BYTE
190	222	0			
191	223	630 C=M			
192	224	1142 C=C-1	PT		
193	225	43 GONC	PAK275 (231)		
194	226	1142 C=C-1	PT		
195	227	1142 C=C-1	PT		
196	230	1156 C=C-1			
197	231	PAK275 1142 C=C-1	PT		
198	232	530 M=C			
199	233	260 C=N			C(3:0) _ END PICKING ADDR
200	234	1552 ? A#C	WPT		PICKED UP LAST BYTE ALREADY ?
201	235	1567 GOC	PAK270 (213)		NO
202	236	1 GOLONG	PAK200		
202	237	2			
203	240	PAK280 106 C=0	X		
204	241	374 RCR	10		SET OLD PC TO VERY SMALL
205	242	160 N=C			
206	243	212 B=A	WPT		
207	244	630 C=M			
208	245	412 A=C	WPT		GET NEXT PACKING ADDR
209	246	1 GOSUB	DECADA		
209	247	0			
210	250	1 GOSUB	PUTPCX		SET NEW PC

```

210 251      0
211 252      152 AB EX WPT
212 253      1333 GOTO PAK265 ( 206 )

```

*
* PAKEND - THIS IS FINAL PART OF THE PACK ROUTINE. WHEN THE
* PACKING REACHED THE FINAL END, IT WOULD BRANCH TO HERE TO
* GENERATE A NEW FINAL END INSTEAD OF PACKING IT. THE REASON
* IS THAT THE FINAL END HAS TO BE RIGHT JUSTIFIED IN THE REG.
*

```

219 254 PAKEND 40 SPOPHD
220 255      260 C=N          LOAD THE .END. ADDR
221 256      1160 DADD=C
222 257      70 C=DATA      LOAD THE .END.
223 260      1634 PT= 0
224 261      130 G=C
225 262      630 C=M
226 263      1160 DADD=C
227 264      416 A=C
228 265      70 C=DATA
229 266      216 B=A
230 267      1616 A SR
231 270      1434 PT= 1
232 271 PKEND1 666 A=A-1 XS
233 272      77 GOC PKEND2 ( 301 ) YES
234 273      666 A=A-1 XS
235 274      1734 INC PT
236 275      1734 INC PT
237 276      1733 GOTO PKEND1 ( 271 )
238 277 PKEND4 402 A=C PT
239 300      173 GOTO PKEND3 ( 317 )
240 301 PKEND2 112 C=0 WPT
241 302      1360 DATA=C
242 303      34 PT= 3
243 304      152 AB EX WPT
244 305      420 LC 4
245 306      34 PT= 3
246 307      1402 ? A<C PT
247 310      1673 GONC PKEND4 ( 277 )
248 311      246 AC EX X
249 312      1146 C=C-1 X
250 313      1160 DADD=C
251 314      412 A=C WPT
252 315      116 C=0 W
253 316      1360 DATA=C
254 317 PKEND3 630 C=M
255 320      174 RCR 4
256 321      252 AC EX WPT
257 322      1 GOSUB GENLNK
258 323      0
259 324 PKEND5 1160 DADD=C
260 325      70 C=DATA
261 326      1634 PT= 0
262 327      230 C=G
263 330      1360 DATA=C

```

*
* NOW CLEAR ALL REG'S BETWEEN NEW .END. AND OLD .END.
*

```

266 331      106 C=0 X
267 332      1160 DADD=C
268 333      1570 C=REGN 13

```

ENABLE CHIP 0
C.X _ OLD CHAIN HEAD

NOMAS

NOT Manufacturer Supported
recipient agrees NOT to contact manufacturer

```

269 334          246 AC EX X          C.X _ NEW CHAIN HEAD
270 335          1550 REGN=C 13
271 336          56 B=0 W
272 337 PKEND6 1546 ? A#C X          ALL DONE ?
273 340          1 GOLNC DCPL00      YES - DECOMPILE AND EXIT
273 341          2
274 342          1146 C=C-1 X
275 343          1160 DADD=C
276 344          356 BC EX W
277 345          1360 DATA=C
278 346          356 BC EX W
279 347          1703 GOTO PKEND6 ( 337 )

```

*
* GTFEND - LOAD FINAL END

```

283 350 GTFEND 106 C=0 X
284 351          1160 DADD=C
285 352          1570 C=REGN 13      LOAD CHAIN HEAD
286 353 GTFENI 34 PT= 3
287 354          420 LC 4
288 355          1160 DADD=C
289 356          34 PT= 3
290 357          412 A=C WPT
291 360          70 C=DATA
292 361          1740 RTN

```

*
* PAKSPC

*
* SPECIAL PACK LOGIC - A SUBBRANCH FROM NEXT LINE ROUTINE
* PACK CALLS NXLDEL TO FIGURE OUT HOW MANY BYTES IN NEXT LINE,
* BUT NXLDEL WOULD SEND IT BACK TO HERE IF IT ENCOUNTER AN
* ALBL OR AN END. THE SPECIAL PACK LOGIC HERE WILL GENERATE A
* NEW LINK FOR EACH ALBL OR END, BUT NOT FOR THE FINAL END. IT
* WILL BRANCH AGAIN TO PAKEND, LET HIM TO TAKE CARE OF THE
* FINAL END.

```

304 362 PAKSPC 1 GOSUB GTLINK
304 363          0
305 364          1076 C=C+1 S          IS IT AN END
306 365          67 GOC PKSPC1 ( 373 ) NO, IS AN ALBL
307 366          776 C=C+C S          CHECK FOR FINAL END
308 367          776 C=C+C S
309 370          776 C=C+C S
310 371          1 GOLC PAKEND          IS A FINAL END
310 372          3
311 373 PKSPC1 630 C=M          C(3:0) _ NEXT PACKING ADDR
312 374          412 A=C WPT
313 375          174 RCR 4          C(3:0) _ PREVIOUS END OR ALBL ADDR
314 376          252 AC EX WPT
315 377          1 GOSUB GENLNK
315 400          0
316 401          374 RCR 10
317 402          1 GOSUB INCAD2
317 403          0
318 404          252 AC EX WPT
319 405          530 M=C
320 406 PKSPC3 260 C=N
321 407          412 A=C WPT
322 410          1 GOSUB INCAD2
322 411          0

```

```

323 412      252 C=A      WPT
323 413      412
324 414      160 N=C
325 415      1 GOSUB     NXBYTA
325 416      0
326 417      1574 RCR     12
327 420      1042 C=C+1   PT
328 421      1 GOLC      NXLTX
328 422      3
329 423      1740 RTN

```

- * PKIOAS - PACK I/O BUFFER AREA & KEY ASSIGNMENT AREA AT BOTTOM
- * OF PROGRAM MEMORY.
- * ASSUME CHIP 0 ENABLE
- * USED A,B,C,M,N. RETURN WITH CHIP 0 DISABLE
- * 2 SUB LEVELS DEEP

* FOR FLOWCHART OF PKIOAS, SEE DRC'S LAB NOTEBOOK #00004 P.26

```

*
339 424 PKIOAS 1570 C=REGN 13      LOAD CHAIN HEAD
340 425      346 BC EX X          SAVE CHAIN HEAD IN B.X
341 426      116 C=0 W
342 427      460 LDI
343 430      300 CON2 12 0        ADDR OF CHIP 12 REG.0
344 431      530 M=C              MC(2:0) - CURRENT STACK ADDR
345 432      160 N=C              NC(2:0) - CURRENT CHECKING ADDR
346 433      203 GOTO PKASN3 ( 453)
347 434 PKM10 1160 DADD=C          ENABLE CHECKING REG.
348 435      70 C=DATA            LOAD THE CHECKING REG.
349 436      1076 C=C+1 S          IS A KEY ASSIGNMENT REG. ?
350 437      223 GONC PKI010 ( 461) NO, TRY TO PACK I/O AREA
351 440      1434 PT= 1
352 441      1352 ? C#0 WPT       KEYCODE IN REG.[1:0] ?
353 442      47 GOC PKASN1 ( 446) YES
354 443      574 RCR 6
355 444      1352 ? C#0 WPT       KEYCODE IN REG.[6:7] ?
356 445      43 GONC PKASN2 ( 451) NO, PACK THIS REG.
357 446 PKASN1 1 GOSUB MOVREG
357 447      0
358 450      33 GOTO PKASN3 ( 453)
359 451 PKASN2 1 GOSUB CLRREG
359 452      0
360 453 PKASN3 306 C=B X          LOAD CHAIN HEAD
361 454      406 A=C X
362 455      260 C=N
363 456      1546 ? A#C X          REACHED CHAIN HEAD ?
364 457      1557 GOC PKM10 ( 434) NOT YET
365 460      1740 RTN

```

- * NOTE THERE IS AT LEAST ONE STATE TO BE HAD BY MOVING THE LOGIC
- * AT PKASN3 AHEAD OF PKM10. THE USE OF B.M INSTEAD OF N11:10 TO
- * STORE THE BUFFER LENGTH IN THE PKI015 PATH MIGHT RESULT IN
- * SAVING ANOTHER STATE OR TWO.

```

*
371 461 PKI010 1176 C=C-1 S      RESTORE THE REG.
372 462      1356 ? C#0 W          REACHED EMPTY AREA ?
373 463      1640 RTN NC          YES WE ARE DONE
374 464      376 BC EX S          CHECK I O BUFFER
375 465      334 PT= 10
376 466      130 G=C              G _ BUFFER LENGTH
377 467      260 C=N

```



```

378 470      230 C=G      NK(11:10) _ BUFFER LENGTH
379 471      160 N=C
380 472      1434 PT= 1
381 473 PKI020 260 C=N      C.(11:0) _ BUFFER LENGTH
382 474      374 RCR 10      DONE WITH THIS BUFFER ?
383 475      1152 C=C-1 WPT
384 476      1557 GOC PKASH3 ( 453 ) YES
385 477      174 RCR 4
386 500      160 N=C      PUT THE UPDATED LENGTH BACK
387 501      1 GOSUB CLRREG
387 502      0
388 503      1336 ? B#0 S      IS THIS BUFFER BEING USED?
389 504      1 GSUBC PUTREG      YES. PUT REG BACK.
389 505      1
390 506      1653 GOTO PKI020 ( 473 )
391      FILLTO:0524      PRESERVE ENTRY TABLE
      507      0000 NOP
      510      0000 NOP
      511      0000 NOP
      512      0000 NOP
      513      0000 NOP
      514      0000 NOP
      515      0000 NOP
      516      0000 NOP
      517      0000 NOP
      520      0000 NOP
      521      0000 NOP
      522      0000 NOP
      523      0000 NOP
      524      0000 NOP

```

```

*
* CLRREG - CLEAR AN UNUSED REG. IN PROGRAM MEMORY
* THIS ROUTINE CALLED BY PKIOAS WHEN CLEAR AN UNUSED REG
* IN I/O OR KEY ASSIGNMENT AREA.
* NI(3:0) = CURRENT CHECKING REG'S ADDR ( THE REG TO BE CLEARED )
*
*      398 525 CLRREG 260 C=N      C.X _ CURRENT CHECKING ADDR
*      399 526      1160 DADD=C      POINT TO NEXT CHECKING REG.
*      400 527      1056 C=C+1
*      401 530      160 N=C
*      402 531      70 C=DATA
*      403 532      16 A=0
*      404 533      73 GOTO MOVREG ( 542 )
*
* MOVREG - MOVE A REG. TO OTHER ADDR
* THIS ROUTINE CALLED BY PKIOAS FOR PACKING THOSE UNUSED REG
* NI(3:0) = ADDR OF THE REG TO BE MOVED
* MI(3:0) = DESTINATION ADDR
*      410
* PUTREG - PUT A REGISTER BACK TO THE DESTINATION ADDRESS, AND
* INCREMENT THE DESTINATION ADDRESS.
* USES: A,C
* IN: A = REG TO BE PUT BACK
*      M.X = DESTINATION ADDRESS
* OUT: (A) IS STORED TO DATA REG (M.X)
*      DADD = M.X
*      M.X IS INCREMENTED BY 1.
*      A IS CLOBBERED.
* ASSUMES: HEXMODE, NO PERIPHERAL ENABLED.
*

```

```

422 534 MOVREG      1 GOSUB CLRREG      CLEAR THE CHECKING REG FIRST
422 535              0
423              ENTRY PUTREG
424 536 PUTREG      630 C=M
425 537              1160 DADD=C          ENABLE THE DESTINATION REG.
426 540              1056 C=C+1          POINT TO NEXT DESTINATION REG
427 541              530 M=C
428 542 MOVR10      256 AC EX  W
429 543              1360 DATA=C
430 544              1740 RTN

*
* COPY - COPY A PROGRAM FROM ROM TO RAM
* ALPHA STRING IS IN REG.9 RIGHT JUSTIFIED. THE ROUTINE WILL
* SEARCH THE STRING IN ROM AND START COPYING FROM TOP OF
* THAT PROGRAM. IF REG.9 HAS NULL STRING AND IN ROM MODE, COPY
* THE CURRENT PGM.
* USED A,B,C CALL ASRCH, MEMLFT.
* NORMAL RETURN TO HFRKB. IF NOT ENOUGH MEM TO STORE THE PGM,
* DO NOTHING, SIMPLY GOTO PACK THE MEM AND SAY TRY AGAIN.
* DURING THE COPYING:
* C[6:3] HAS ROM ADDR. B[5:3] HAS RAM ADDR. B[2:0] HAS THE
* REMAINING # OF REG.'S TO BE COPIED.
443 545 XCOPY      1170 C=REGN 9          GET ALPHA STRING
444 546              530 M=C
445 547              1356 ? C#0  W          NULLSTRING ?
446 550              213 GONC  CPY120 ( 571) YES, START FROM PGM HEAD
447 551              1 GOSUB  ASRCH          DO THE ALPHA SEARCH
447 552              0
448 553              1356 ? C#0  W          FIND IT ?
449 554 CPYNE      1 GOLNC  ERRNE
449 555              2
450 556 CPY100     1114 ?S9=1          USER CODE ?
451 557              1757 GOC  CPYNE ( 554) NO, MICRO CODE
452 560              1014 ?S2=1          ROM ?
453 561              47 GOC  CPY110 ( 565) YES
454              ENTRY  ERRRAM
455 562 ERRRAM      1 GOSUB  ERROR
455 563              0
456 564              0 XDEF  MSGRAM          SAY RAM
457 565 CPY110      34 PT= 3          GET THE PRM HEAD
458 566              1 GOSUB  ROMH05
458 567              0
459 570              53 GOTO  CPY130 ( 575)
460 571 CPY120      314 ?S10=1          ARE WE IN ROM ?
461 572              1703 GONC  ERRRAM ( 562) NO, WE ARE IN RAM
462 573              1 GOSUB  ROMHED          GET PGM HEAD
462 574              0

*
464 575 CPY130      256 AC EX  W
465 576              674 RCR  11          C.M _ PGM HEAD ADDR
466 577              1460 CXISA          GET HEADED WORD
467 600              1166 C=C-1  XS          CHECK PRIVATE
468 601              1166 C=C-1  XS
469 602              1166 C=C-1  XS
470 603              47 GOC  CPY140 ( 607) NOT PRIVATE
471              ENTRY  ERRPR
472 604 ERRPR      1 GOSUB  ERROR
472 605              0
473 606              0 XDEF  MSGPR
474 607 CPY140      1172 C=C-1  M          POINT TO P-1

```

475	610	372 BC EX	M	SAVE ADDR IN B.M
476	611	1 GOSUB	MEMLEFT	HOW MANY REG LEFT ?
476	612	0		
477	613	406 A=C	X	A.X_ # OF REG LEFT IN MEM
478	614	332 C=B	M	SEE HOW MANY REG REQUIRED
479	615	432 A=C	M	
480	616	1460 CXISA		LOAD # OF REG REQUIRED
481	617	1406 ? AKC	X	ENOUGH ROOM FOR THIS PRM ?
482	620	1 GOLC	PACKE	NO, TRY TO PACK MEM
482	621	3		
483	622	1146 C=C-1	X	
484	623	346 BC EX	X	SAVE # OF REG IN B.X
485	624	106 C=0	X	
486	625	1160 DADD=C		
487	626	1570 C=REGN	13	LOAD CHAIN HEAD
488	627	1160 DADD=C		
489	630	1146 C=C-1	X	POINT TO STARTING RAM ADDR
490	631	674 RCR	11	
491	632	372 BC EX	M	SAVE RAM ADDR IN B.M
492	633	70 C=DATA		LOAD FINAL END
493	634	1730 CST EX		
494	635	204 SS=	0	MAKE IT BECOME LOCAT END
495	636	1730 CST EX		
496	637	1360 DATA=C		
497	640	272 AC EX	M	GET ROM ADDR
498	641	1072 C=C+1	M	POINT TO HEADER WORD
499	642	1460 CXISA		LOAD # OF BYTES IN 1ST REG
500	643	1434 PT=	1	
501	644	1142 C=C-1	PT	
502	645	402 A=C	PT	MOVE COUNTER TO C.S
503	646	1574 RCR	12	
504	647	242 AC EX	PT	
505	650	1074 RCR	2	
506	651	16 A=C	W	PACK 7 ROM WORDS TO 1 REG
507	652	1072 C=C+1	M	
508	653	1460 CXISA		LOAD A BYTE
509	654	412 A=C	WPT	
510	655	1176 C=C-1	S	
511	656	47 GOC	CPY160 (662)	DONE WITH ONE REG.
512	657	1756 A SL	W	
513	660	1756 A SL	W	
514	661	1713 GOTO	CPY150 (652)	
515	662	356 BC EX	W	
516	663	74 RCR	3	C.X _ RAM ADDR
517	664	1160 DADD=C		
518	665	1146 C=C-1	X	
519	666	356 BC EX	W	
520	667	256 AC EX	W	
521	670	1360 DATA=C		
522	671	356 BC EX	W	
523	672	674 RCR	11	C.X _ # OF REG REMAINED
524	673	1146 C=C-1	X	
525	674	77 GOC	CPY170 (703)	ALL DONE
526	675	356 BC EX	W	
527	676	256 AC EX	W	
528	677	1334 PT=	13	
529	700	620 LC	6	
530	701	1434 PT=	1	
531	702	1473 GOTO	CPY145 (651)	

*

```

533 703 CPY170      74 RCR      3
534 704             1046 C=C+1  X
535 705             406 A=C     X
536 706             106 C=0     X
537 707             1160 DADD=C
538 710             1570 C=REGN 13
539 711             246 AC EX   X
540 712             1550 REGN=C 13
541 713             34 PT=     3
542 714             2 A=0      PT
543
544 715             304 S10=    0
545 716             1 GOSUB   PUTPCX
546 717             0
547 720             1 GOSUB   DECMPL
548 721             0
549 722 NFRKBX      1 GOLONG  NFRKB
550 723             2
551 724             ENTRY   TRGSET
552 725             1670 C=REGN 14
553 726             106 C=0     X
554 727             1074 RCR     2
555 730             1530 ST=C
556 731             504 S6=     0
557 732             1204 S7=     0
558 733             260 C=N
559 734             1740 RTN

```

A.X _ NEW CHAIN HEAD ADDR

PUT NEW CHAINHEAD AWAY

SET PC TO BYTE 0
OF OLD CHAINHEAD REGISTER
CLEAR ROMFLAG

DECOMPILE

*
*
* PATCH1 - POST-RELEASE FIX TO DECAD & DECADA 9/21/78
* DECAD IS IN QUAD 10.
*

```

562             ENTRY   PATCH1
563 734 PATCH1      542 A=A+1  PT
564 735             1 GOLONG  INCADA
565 736             2
566 737             556 A=A+1
567 740             1740 RTN

```

*
* PATCH2 - POST-RELEASE FIX TO CLRPGM FOUND LATER IN QUAD 8.
* THIS PATCH ALLOWS CLEARING OF PRIVATE PROGRAMS AT THE END OF
* PROGRAM MEMORY.
* THE S10=0 IS ANOTHER FIX TO ALLOW CLEARING OF RAM PROGRAMS
* WHEN THE PROGRAM COUNTER IS POINTING TO ROM.
*

```

574             ENTRY   PATCH2
575 741 PATCH2      304 S10=    0
576 742             1 GOSUB   FIXEND
577 743             0
578 744             312 C=B     WPT
579 745             1730 CST EX
580 746             504 S6=     0
581 747             1730 CST EX
582 750             1 GOSUB   PTBYTM
583 751             0
584 752             474 RCR     8
585 753             412 A=C     WPT
586 754             1 GOLONG  CPGMHD
587 755             2

```

CLEAR ROM FLAG
GET THE 3RD BYTE OF THE CURRENT END

TURN OFF THE PRIVATE BIT.

PUT THE 3RD BYTE BACK

SET UP FOR CPGMHD

GO TO THE TOP OF THE PROGRAM AND RE

* PATCH3 - POST-RELEASE FIX TO INSSUB FOUND AT THE END OF QUAD 8
 * THIS FIX PREVENTS DATA ENTRY INTO PRIVATE PROGRAMS.

```

*
589                      ENTRY PATCH3
590 756 PATCH3 314 ?S10=1          IS THIS A ROM PROGRAM?
591 757          1640 RTN NC        NO, CONTINUE
592 760          1 GOSUB ERROR      YES, ERROR OUT
592 761          0
593 762          0 XDEF MSGROM

```

* PATCH5 - POST-RELEASE FIX TO DEL NNN TO MAKE IT WORK WHEN LINE#=000

```

*
597                      ENTRY PATCH5
598 763 PATCH5 1146 C=C-1 X        DEC LINE# & TEST FOR 000
599 764          1640 RTN NC        NOT ZERO - OK
600 765          214 ?S5=1          IS THIS BACKARROW?
601 766          1347 GOC NFRKBX ( 722) YES, DO NOTHING.
602 767          106 C=0 X          MUST BE DEL NNN.
603                                PUT LINE# BACK TO 000
604 770          1740 RTN
605                                FILLTO 0777
606 771          0000 NOP
607 772          0000 NOP
608 773          0000 NOP
609 774          0000 NOP
610 775          0000 NOP
611 776          0000 NOP
612 777          0000 NOP

```

*
 * UPLINK JUMP TABLE HERE

```

*
609 1000 TABUPL 70 C=DATA          GET THE FIRST BYTE
610 1001          553 GOTO UPLB0 (1056) SPECIAL CASE
611 1002          70 C=DATA
612 1003          443 GOTO UPLB1 (1047) ANOTHER SPECIAL CASE
613 1004          70 C=DATA
614 1005          403 GOTO UPLB2 (1045)
615 1006          70 C=DATA
616 1007          343 GOTO UPLB3 (1043)
617 1010          70 C=DATA
618 1011          303 GOTO UPLB4 (1041)
619 1012          70 C=DATA
620 1013          243 GOTO UPLB5 (1037)
621 1014 UPLB6 70 C=DATA
622 1015 GBAS 374 RCR 10           ROTATE LINK INTO PLACE
623 1016          1740 RTN
624                                FILLTO 01017
625 1017          0000 NOP

```

* GET BYTE JUMP TABLE HERE

```

626 1020 TBLGBA 70 C=DATA          7 ENTRY POINTS(0,2,4,...,12)
627 1021          1740 RTN
628 1022          70 C=DATA
629 1023          223 GOTO GBA1 (1045)
630 1024          70 C=DATA
631 1025          163 GOTO GBA2 (1043)
632 1026          70 C=DATA
633 1027          123 GOTO GBA3 (1041)
634 1030          70 C=DATA
635 1031          63 GOTO GBA4 (1037)
636 1032          70 C=DATA

```

```

637 1033      1623 GOTO  GBA5  (1015)
638 1034 GBA6      70 C=DATA
639 1035 GBA6A     1574 RCR   12
640 1036      1740 RTN
641      UPLB5
642 1037 GBA4      474 RCR   8
643 1040      1740 RTN
644      UPLB4
645 1041 GBA3      574 RCR   6
646 1042      1740 RTN
647      UPLB3
648 1043 GBA2      174 RCR   4
649 1044      1740 RTN
650      UPLB2
651 1045 GBA1      1074 RCR   2
652 1046      1740 RTN
653 1047 UPLB1      206 B=A    X
654 1050      352 BC EX  WPT
655 1051      1146 C=C-1  X
656 1052      1160 DADD=C
657 1053      70 C=DATA
658 1054      312 C=B    WPT
659 1055      1740 RTN
660 1056 UPLB0      206 B=A    X
661 1057      346 BC EX  X
662 1060      1146 C=C-1  X
663 1061      1160 DADD=C
664 1062      70 C=DATA
665 1063      306 C=B    X
666 1064      1513 GOTO  GBA6A (1035) PUT ALL 3 BYTES IN PLACE
* UPLINK - MOVE UP ONE LINK OF THE LABEL CHAIN
* GIVEN AN ADDRESS OF THE FIRST BYTE OF A LINK IN A[0-3] IN
*- MM FORMAT, AND THE LINK AT THAT ADDRESS IN C[0-2], RETURNS
*- THE ADDRESS OF THE NEXT LINK IN A AND THE NEXT LINK IN C IN
*- THE SAME FORMAT AS INPUT. IN ADDITION, THE BYTE FOLLOWING THE
*- NEXT LINK IS FOUND IN C[12-13].
* EXPECTS AND RETURNS PT=3
* USES A[0-3], R[0-3], AND C.
* LEAVES DADD#0.
*
* GTLINK - GET A LINK. GIVEN THE ADDRESS OF A LINK IN A[0-3]
*- RETURNS THE LINK IN THE SAME FORM AS UPLINK.
*
* GTLNKA - SAME AS GTLINK, BUT EXPECT ADDRESS IN C[0-3]
*
682 1065 UPLINK     102 C=0    PT      CREATE THE NEW ADDRESS
683 1066      752 C=C+C    WPT      EXPAND THE LINK
684 1067      752 C=C+C    WPT
685 1070      752 C=C+C    WPT
686 1071      1042 C=C+1    PT      ADD 2 TO BYTE NUMBER WHEN DOUBLED
687 1072      742 C=C+C    PT      PREPARE FOR BASE 14 ADD
688 1073      746 C=C+C    X
689 1074      37 GOC      ULINK1 (1077)
690 1075      1706 C SR    X
691 1076      33 GOTO     ULINK2 (1101)
692 1077 ULINK1     1706 C SR    X
693 1100      1066 C=C+1    XS
694 1101 ULINK2     1012 C=A+C    WPT      FORM NEW ADDRESS
695 1102      33 GONC     ULINK3 (1105)
696 1103      1046 C=C+1    X

```

```

697                                LEGAL
698 1104                        33 GOTO   GTLNKA (1107) ADDRESS READY
699 1105 ULINK3 1142 C=C-1 PT
700 1106                        1142 C=C-1 PT          THE ADDRESS IS READY
701 1107 GTLNKA 412 A=C WPT          SAVE THE ADDRESS
702 1110                        1160 DADD=C          SELECT THE CORRECT REGISTER
703 1111                        174 RCR 4             PREPARE FOR 7 WAY TABLE
704 1112                        460 LDI
* TABLE JUMP
706 1113                        1040 CON 01040          UPLINK TABLE ADDRESS
707 1114                        374 RCR 10
708 1115                        740 GOTOC          GO GET LINK
* GTLINK HERE
710 1116 GTLINK 252 AC EX WPT          PUT ADDRESS IN PLACE
711 1117                        1703 GOTO GTLNKA (1107) GO GET LINK
*
* XBST - EXECUTE BACKSTEP
* MAINLINE CODE TO EXECUTE BACKSTEP FUNCTION
* ASSUMES STATUS SET 0 UP, PRGM MODE BIT USED
* USES 3 SUB LEVELS.
*
718 1120 XBST 574 RCR 6          CATALOG SET
719 1121      1730 CST EX
720 1122      1414 ?S1=1
721 1123      1 GOLC BSTCAT
721 1124      3
722 1125      1 GOSUB SSTBST
722 1126      0
723 1127      1 GOSUB BSTEP          BACK UP ONE LINE
723 1130      0
724 1131 XBST1 1 GOSUB DFRST9          DISPLAY STEP NUMBER UNTIL KEY UP
724 1132      0
725 1133      14 ?S3=1          PROG MODE?
726 1134      1 GOLC DRSY25          YES, DON'T PUT UP NEW DISPLAY
726 1135      3
727 1136      1 GOLONG NFRKB1          DONE!
727 1137      2
*
* XSST - EXECUTE SINGLE STEP
* ASSUMES STATUS SET 0 UP, PRGM MODE BIT USED.
*
732 1140 XSST 574 RCR 6          CATALOG MODE
733 1141      1730 CST EX
734 1142      1414 ?S1=1
735 1143      1 GOLC SSTCAT
735 1144      3
736 1145      1 GOSUB SSTBST
736 1146      0
737 1147      1 GOSUB GETPC
737 1150      0
738 1151      1770 C=REGN 15          ALSO GET THE LINE NUMBER
739 1152      14 ?S3=1          PRGM MODE?
740 1153      203 GONC XSSTR (1173) NO, RUN MODE
741 1154      1346 ? C#0 X          IF THE LINE NUMBER #0
742 1155      1 GSUBC NXLSST          GO TO THE NEXT LINE
742 1156      1
743 1157      1 GOSUB GETLIN          FIX THE LINE NUMBER
743 1160      0
744 1161      514 ?S6=1          TOP OF PROG?
745 1162      23 GONC *+2 (1164) NO, DO A SIMPLE INCREMENT

```

746 1163	106 C=0	X	YES, SET LINE # TO 1
747 1164	1046 C=C+1	X	
748 1165	23 GONC	*+2	(1167) IF NOT VALID, LEAVE ALONE
749 1166	1146 C=C-1	X	
750 1167	1750 REGN=C	15	
751 1170	1 GOSUB	PUTPC	FIX UP PC
751 1171	0		
752 1172	1373 GOTO	XBST1	(1131) DONE!
* RUN MODE SINGLE STEP			
754 1173 XSSTR	1346 ? C#0	X	LINE 0?
755 1174	37 GOC	XSSTR1	(1177) NO, DO NOTHING
756 1175	1046 C=C+1	X	YES, MOVE TO LINE 1
757 1176	1750 REGN=C	15	
758 1177 XSSTR1	1 GOSUB	DFKBC	DISPLAY LINE NUMBER
758 1200	0		
759 1201	1114 ?S9=1		KEYBD RESET YET?
760 1202	1 GSUBNC	NULLST	NO
760 1203	0		
761 1204	1 GOSUB	GETLIN	INCREMENT THE LINE NUMBER
761 1205	0		
762 1206	1046 C=C+1	X	
763 1207	1750 REGN=C	15	
764 1210	1 GOSUB	SETSST	SET SST BIT
764 1211	0		
765 1212	1 GOLONG	RUNNK	GO DO 1 INSTRUCTION
765 1213	2		

*

* CLRPGM - CLEAR PROGRAM

* THIS ROUTINE CLEARS THE PROGRAM WHOSE NAME IS FOUND IN REG 9.

* IF REG 9 = NULL, THEN CLEAR THE PROGRAM WHERE THE PC IS

*- CURRENTLY POINTING

* USES A, B[0-3], C, M, N, PT, S[0-7, 9], 4 SUB LEVELS.

*

* SEE DRC'S LAB NOTEBOOK #10422X P.106 FOR FLOWCHART OF CLRPGM

*

775 1214 CLRPGM	1170 C=REGN	9	RETRIEVE THE NAME
776 1215	530 M=C		SAVE FOR ASCH
777 1216	1356 ? C#0		LABEL PRESENT?
778 1217	207 GOC	CLRP1	(1237) YES, GO FIND IT
779 1220	314 ?S10=1		ROM FLAG?
780 1221	437 GOC	XCLPX1	(1264) YES, DO NOTHING
781 1222	1 GOSUB	GETPC	NULL STRING HERE, GET CURRENT
781 1223	0		
782 1224 CLRPG2	1 GOSUB	FLINK	FIND THE END OF THE PROGRAM
782 1225	0		
783 1226	1 GOSUB	PATCH2	THIS PATCH CLEARS THE PRIVATE
783 1227	0		
784 1230	1 GOSUB	PUTPCL	STORE IN PC & GET LINE #
784 1231	0		
785 1232	210 S5=	1	DELETE THE PROGRAM END
786 1233	116 C=0		SET # LINES TO DELETE = FFF
787 1234	1146 C=C-1	X	
788 1235	1110 S9=	1	SET UP FOR PACH
789 1236	413 GOTO	CLRP3	(1277) GO DELETE THE PROGRAM
790 1237 CLRPG1	1 GOSUB	ASCH	GO DO ALPHA SEARCH
790 1240	0		
791 1241	34 PT=	3	
792 1242	1356 ? C#0		SUCCESS?
793 1243	33 GONC	CLPEER	(1246) NO, ERROR EXIT
794 1244	1014 ?S2=1		LABEL FOUND IN ROM?

NOMAS
 NOT A MACHINE LANGUAGE
 Register Address Ref. to Global memory


```

795 1245          1573 GONC   CLRP2 (1224) NO, FOUND IN RAM.
796 1246 CLPERR    1 GOLONG ERRNE      ERROR EXIT
796 1247          2

```

```

*
* DELNNN - DELETE NNN INSTRUCTIONS
* THIS ROUTINE DELETES NNN LINES OF PROGRAM STARTING WITH
*- THE ONE POINTED TO BY THE PC.
* THE NNN ARGUMENT IS FOUND IN A[X].
* DELNNN WILL NOT DELETE A PROGRAM END STATEMENT.
* IN ALL OTHER WAYS THIS FUNCTION IS LIKE XDELET FOUND BELOW.
*
*
* XDELET - EXECUTE DELETE LINE
* DELETES 1 LINE FROM PROGRAM MEMORY STARTING WITH
*- THE BYTE POINTED TO BY THE PC +1. THE PC IS SET
*- TO PROPERLY POINT TO THE PRECEEDING LINE.
* UPDATES LINKS IF A CHAIN ELEMENT IS DELETED.
* ALSO SETS PACK AND DECOMPILE BITS OF THE
*- FOLLOWING END.
* S6 SET TO 1 WHEN AN END IS DELETED
* NOTE, WILL NOT DELETE THE FINAL END!
* USES A,B,C,M,N,S[0-9],PT
*
* SEE DRC'S LAB NOTEBOOK #10422X P.107 FOR FLOWCHART OF DELNNN AND
* XDELET.
*

```

```

820 1250 DELNNN 256 AC EX          STORE # TO DELETE -1 IN N
821 1251          1146 C=C-1 X
822 1252          617 GOC   XDELEX (1333) ZERO TO DELETE, DO NOTHING.
823 1253          204 S5=   0      DON'T DELETE AN END.
824 1254          14 ?S3=1      PRGM MODE?
825 1255          563 GONC   XDELEX (1333) NO, DON'T DO IT.
826 1256          33 GOTO   XDELA (1261) GO DELETE.
827 1257 XDELET 210 S5=   1      DELETE ENDS.
828 1260          116 C=0      DELETE 1 LINE
829 1261 XDELA  160 N=C      STORE # OF LINES TO DELETE -1
830 1262          1514 ?S12=1    PRIVATE?
831 1263          507 GOC   XDELEX (1333) YES - DO NOTHING
832 1264 XCLPX1 314 ?S10=1      ROM FLAG?
833 1265          1 GOLC   INSSUB  YES, DO NOTHING. DISPLAY [ROM].
833 1266          3
834 1267          1 GOSUB  GETPC   GET STARTING ADDRESS OF DELETE
834 1270          0
835 1271          1770 C=REGN 15   DECREMENT LINE NUMBER IF NON ZERO
836 1272          1 GOSUB  PATCH5
836 1273          0
837 1274          1750 REGN=C 15
838 1275          1104 S9=   0      CLEAR PACK FLAG
839 1276          260 C=N
840 1277 CLRP3   504 S6=   0      CLEAR END FLAG
841 1300 XDELM1 160 N=C      STORE # OF LINES LEFT TO DELETE -1
842 1301          1 GOSUB  DELLIN  DELETE 1 LINE
842 1302          0
843 1303          514 ?S6=1      TRAVERSED AN END?
844 1304          47 GOC   XDELM2 (1310) YES. QUIT.
845 1305          260 C=N      C[X]-1 = # LEFT TO DELETE
846 1306          1146 C=C-1 X    DONE?
847 1307          1713 GONC   XDELM1 (1300) NO, GO AROUND AGAIN
848 1310 XDELM2  1 GOSUB  FLINKP  FIND THE CURRENT END
848 1311          0

```

849	1312	174	RCR	4	SAVE PREVIOUS LINK ADDRESS IN C13-6
850	1313	252	AC EX	WPT	
851	1314	374	RCR	10	PUT DECOMPILE BITS IN END
852	1315	1	GOSUB	FIXEND	
852	1316	0			
853	1317	174	RCR	4	PUT A[0-3] BACK
854	1320	412	A=C	WPT	
855	1321	1114	?S9=1		GO PACK?
856	1322	1	GOLC	GT0.5	YES.
856	1323	3			
857	1324	1	GOSUB	GETLIN	BACK STEP IF NEW LINE NUM. # 0.
857	1325	0			
858	1326	1346	? C#0	X	BACK STEP?
859	1327	43	GONC	XDELEX (1333)	NO, LINE 0.
860	1330	630	C=M		RETRIEVE THE CURRENT ADDRESS
861	1331	1	GOSUB	BSTEPA	BACK STEP.
861	1332	0			
862	1333	1	GOLONG	NFRKB	ALL DONE!
862	1334	2			

*
* SSTBST - LOGIC COMMON TO SST AND BST
*

866		ENTRY	SSTBST	
867	1335	SSTBST	1505	CON @1505 GOSUB PRT15
868	1336	674	CON @674	
869		ENTRY	PR15RT	FOR THE PRINTER
870	PR15RT			
871	1337	1	GOSUB	RSTSEQ CLEAR 6 FLAGS
871	1340	0		
872	1341	1	GOSUB	ANNOUT UPDATE ANNUNCIATORS
872	1342	0		
873	1343	1	GOSUB	LINNUM RECONSTRUCT PRIVACY FLAG
873	1344	0		
874	1345	1514	?S12=1	PRIVACY?
875	1346	1657	GOC	XDELEX (1333) YES. GOLONG NFRKB
876	1347	1740	RTN	
877				

* ERROR - ERROR EXIT
* CALLING SEQUENCE:
* GOSUB ERROR
* XDEF <MSGXXX>
* ERROR ROUTINE PERFORMS :
* 1. IF ERROR FLAG ALREADY SET, RESET IT, RTN TO NFRKB
* 2. UNCONDITIONAL RESET DATAENTRY FLAG, AND OTHERS
* 3. DISPLAY ERROR MESSAGE
* 4. IF PROGRAM RUNNING, STOP-RUNNING AND DO A BACK STEP
* 5. ALWAYS RETURN TO NFRKB, WON'T RETURN TO CALLING PROGRAM

*
*
* ERR110 - ERROR EXIT SIMPLY DECIDE TO DO A BACK STEP OR NOT BEFORE
* RETURNING TO NFR. REQUIRED STATUS SET 0 LOADING.
*

893		ENTRY	ERRSUB	
894	1350	ERRSUB	1	GOSUB RSTMS0 ENABLE CHIP 0 AND
894	1351	0		
895				CLEAR DATAENTRY FLAG
896	1352	410	S8=	1 TELL MSG TO SET MSGFLAG
897	1353	1140	SETHX	
898	1354	1274	RCR	7
899	1355	1530	ST=C	

```

900 1356      1014 ?S2=1      ERROR FLAG ?
901 1357      1640 RTN NC      NO
902 1360      1004 S2= 0      RESET ERROR FLAG
903 1361      1630 C=ST
904 1362      1274 RCR 7
905 1363      1650 REGN=C 14
906 1364 ERRTN 1473 GOTO XDELEX (1333)

*
908 1365 ERROR 1 GOSUB ERRSUB
908 1366      0
909 1367      660 C=STK
910 1370      1460 CXISA
911 1371      1 GOSUB MSGE
911 1372      0
912 1373 ERR110 1314 ?S13=1      RUNNING ?
913 1374      37 GOC ERR120 (1377) YES
914 1375      114 ?S4=1      SST ?
915 1376      33 GONC ERR130 (1401) NO
916 1377 ERR120 1 GOSUB BSTEP
916 1400      0
917      ERR130
918 1401      1 GOSUB STOPS      CLEAR PAUSEFLAG & RUNNING
918 1402      0
919 1403      1 GOSUB LINNUM      GUARANTEE VALID LINE NUMBER
919 1404      0
920      FOR PARSE IN PRGM MODE
921 1405      1573 GOTO ERRTN (1364)

*
*
* DELLIN - DELETE LINE FROM PROGRAM MEMORY
* THIS ROUTINE DELETES A LINE OF CODE STARTING WITH THE
*-- NEXT BYTE AFTER THE ONE SPECIFIED BY A10-31 IN MM FORMAT.
* WILL NOT DELETE THE FINAL END.
* RETURNS S6=1 IF END DELETED.
* IF A CHAIN ELEMENT IS DELETED, THE PREVIOUS LINK IS UPDATED
*-- TO INCLUDE THE DELETED LINK.
* USES A10-31,B,C,M,3 SUBROUTINE LEVELS
*
* NOTE !!! THIS ROUTINE CANNOT BE CALLED FROM A SUBROUTINE
*
935 1406 DELLIN 252 AC EX WPT      SAVE STARTING ADDRESS
936 1407      412 A=C WPT
937 1410      530 M=C
938 1411      110 S4= 1
939 1412      1210 S7= 1
940 1413      1 GOSUB HXLDEL      FIND THE ENDING ADDRESS
940 1414      0
941 1415      630 C=M      RETRIEVE THE STARTING ADDRESS
942 1416      252 AC EX WPT      AND SAVE THE ENDING ADDRESS
943 1417      530 M=C
944 1420      63 GOTO DELLN1 (1426) ZERO OUT APPROPRIATE BYTES
945 1421 DELLN2 116 C=0      ZERO 1 BYTE
946 1422      1 GOSUB INCADA      MOVE THERE
946 1423      0
947 1424      1 GOSUB PTBYTA      PUT ZERO'S IN
947 1425      0
948 1426 DELLN1 630 C=M      RETRIEVE THE ENDING ADDRESS
949 1427      1552 ? A#C WPT      DONE?
950 1430      1717 GOC DELLN2 (1421) NO, DELETE SOME MORE
951 1431      1740 RTN      ALL DONE

```

952

```

*
* PTLINK - PUT LINK
* PUTS C[0-3] INTO PROGRAM MEMORY AT THE ADDRESS POINTED TO
*-- A[0-3] IN MM FORMAT.
* PT=3 EXPECTED AND RETURNED
* USES B[0-3]
* THIS ROUTINE MIXED IN WITH PTBYTA
*
  961 1432 PTLINK 252 AC EX WPT      SAVE BYTES TO STORE
  962 1433 PTLNKA 212 B=A    WPT      IN B
  963 1434          1160 DADD=C      WAKE UP THE RIGHT REGISTER
  964 1435          412 A=C    WPT      RESTORE A
  965 1436          174 RCR      4      PREPARE FOR BRANCH TABLE (7)
  966 1437          460 LDI
* TABLE JUMP
  968 1440          1203 CON      Q1203      PUT LINK TABLE ADDRESS
  969 1441 PTLNKB 374 RCR      10      PUT ADDRESS IN-POSITION
  970 1442          740 GOTOC      7 WAY BRANCH
*
* PTBYTA - PUT BYTE
* PUT THE BYTE IN C[0-1] INTO RAM AT THE ADDRESS
*-- POINTED TO BY A[0-3] IN MM-FORMAT.
* PT=3 OUT.
* USES B[0-1]
*
  978 1443 PTBYTA 256 AC EX      SAVE BYTE TO STORE IN B
  979 1444          1434 PT=      1      SET UP FOR 1 BYTE STORE
  980 1445          212 B=A    WPT      SAVE BYTE
  981 1446          1160 DADD=C      WAKE UP THE RIGHT REG.
  982 1447          416 A=C      RESTORE A
  983 1450 PTBYTP 174 RCR      4      PREPARE FOR TABLE JUMP
  984 1451          460 LDI
* TABLE JUMP
  986 1452          1200 CON      Q1200      PUT BYTE TABLE ADDRESS
  987 1453          1663 GOTO      PTLNKB (1441)
*
* PUTPC - PUT AWAY THE PROGRAM COUNTER
* PLACES A[0-3] IN MM FORMAT INTO THE PC AFTER CONVERTING
*-- TO PC FORMAT BY SHIFTING A[3] RIGHT 1 BIT IF S10=0
* PT=3 ASSUMED AND RETURNED
*
* PUTPCF - SAME AS PUTPC, BUT SETS LINE# TO FFF
*
* PUTPCX - SAME AS PUTPCF EXCEPT IF RUNNING LINE# NOT SET TO FFF
*
* PUTPCD - SAME AS PUTPC EXCEPT CALLS DECAD BEFORE GOING TO PUTPC
* CONSEQUENTLY USES 1 SUBROUTINE LEVEL
*
  1001          ENTRY PUTPCD
  1002 1454 PUTPCD 1 GOSUB DECAD
  1002 1455          0
  1003 1456          113 GOTO PUTPC (1467)
*
  1005 1457 PUTPCX 1314 ?S13=1      RUNNING?
  1006 1460          77 GOC PUTPC (1467) YES, DON'T SET LINE# TO FFF
  1007 1461 PUTPCF 116 C=0      SET LINE# TO FFF
  1008 1462          1160 DADD=C
  1009 1463          1770 C=REGN 15
  1010 1464          106 C=0      X

```

```

1011 1465      1146 C=C-1  X
1012 1466      1750 REGN=C 15
1013 1467 PUTPC 106 C=0   X
1014 1470      1160 DADD=C
1015 1471 PUTPC 1470 C=REGN 12      GET PC
1016 1472      252 AC EX  WPT
1017 1473      412 A=C    WPT      NEW PC IN PLACE
1018 1474      314 7S10=1      ROM ADDRESS?
1019 1475      127 GOC   PUTPC3 (1507) YES, NO SHIFT
1020 1476      742 C=C+C  PT      SHIFT 1 BIT RIGHT
1021 1477      23  GONC   PUTPC1 (1501)
1022 1500      1042 C=C+1  PT
1023 1501 PUTPC1 742 C=C+C  PT
1024 1502      23  GONC   PUTPC2 (1504)
1025 1503      1042 C=C+1  PT
1026 1504 PUTPC2 742 C=C+C  PT
1027 1505      23  GONC   PUTPC3 (1507)
1028 1506      1042 C=C+1  PT
1029 1507 PUTPC3 1450 REGN=C 12      PUT PC BACK
1030 1510      1740 RTN
* SPECIAL DELETE AND PACK LOGIC HERE
1032 1511 SKPDEL 114 7S4=1      DELETE LOGIC?
1032 1512      1  GOLNC  PAKSPC   PACK LOGIC GOES HERE
1032 1513      2
1034 1514 SKPDL  1  GOSUB  INCADA   MOVE INSIDE LINK
1034 1515      0
1035 1516      1  GOSUB  FLINKA    FIND VARIOUS LINKS
1035 1517      0
1036 1520      252 AC EX  WPT
1037 1521      412 A=C    WPT      RESTORE ADDRESS BEFORE LINK
1038 1522      1  GOSUB  DECADA
1038 1523      0
1039 1524      252 AC EX  WPT
1040 1525      530 M=C
1041 1526      1  GOSUB  GTLINK    SAVE CURRENT AND PREVIOUS LINK ADDRESS
1041 1527      0                  GET THE CURRENT LINK
1042 1530      1076 C=C+1  S      END?
1043 1531      137 GOC   SKPD7 (1544) NO, ALPHA LABEL,
1044 1532      510 S6=    1      END, REMEMBER IT.
1045 1533      214 7S5=1      TRAVERSE THE END? (DELETE)
1046 1534      63  GONC   SKPD4 (1542) NO, (DEL NNN)
1047 1535      1176 C=C-1  S      FINAL END?
1048 1536      776 C=C+C  S
1049 1537      776 C=C+C  S
1050 1540      776 C=C+C  S
1051 1541      343 GONC   SKPD1 (1575) NO, TRAVERSE THE END
1052 1542 SKPD4  660 C=STK      THIS IS A SPECIAL CASE EXT OUT
1053 1543      1740 RTN      OF DELLIN!!! WATCH OUT!!!
1054 1544 SKPD7  630 C=M      SAVE M IN B
1055 1545      356 BC EX
1056 1546      1  GOSUB  INCAD2    CHECK BYTE AFTER TEXT CHAR.
1056 1547      0
1057 1550      1  GOSUB  HXBYTA    GET KEYCODE
1057 1551      0
1058 1552      1434 PT=    1
1059 1553      1152 C=C-1  WPT      SUBTRACT 1
1060 1554      117 GOC   SKPD6 (1565) DO NOTHING IF 0 KEYCODE
1061 1555      406 A=C    X      POSITION FOR BITMAP SUBS.
1062 1556      1746 A SL  X
1063 1557      116 C=0

```

1064	1560	1160	DADD=C		
1065	1561	1	GOSUB	TBITMP	CLEAR BIT
1065	1562	0			
1066	1563	1	GOSUB	SRENAP	
1066	1564	0			
1067	1565	316	C=B		RESTORE B TO M
1068	1566	530	M=C		
1069	1567	34	PT=	3	
1070	1570	412	A=C	WPT	
1071	1571	1	GOSUB	INCADA	GET ADDRESS OF LINK IN A[0-3]
1071	1572	0			
1072	1573	1	GOSUB	GTLINK	GET THE LINK
1072	1574	0			
1073	1575	374	RCR	10	SAVE THE CURRENT LINK
1074	1576	356	BC EX		FIX PREVIOUS LINK
1075	1577	630	C=M		GET ADDRESS OF PREVIOUS LINK
1076	1600	174	RCR	4	POSITION FOR GTLINK
1077	1601	1	GOSUB	GTLINKA	GET IT
1077	1602	0			
1078	1603	356	BC EX		PLACE THE FIRST LINK IN A[0-3]
1079	1604	174	RCR	4	
1080	1605	356	BC EX		
1081	1606	152	AB EX	WPT	
1082	1607	1506	? A#0	X	END OF CHAIN SPECIAL CASE
1083	1610	27	GOC	++2 (1612)	NOT HERE
1084	1611	106	C=0	X	PUT END OF CHAIN 1 LINK DOWN
1085	1612	1066	C=C+1	XS	BASE 14 ADD
1086	1613	1066	C=C+1	XS	
1087	1614	1006	C=A+C	X	CREATE NEW LONGER LINK
1088	1615	47	GOC	SKPD2 (1621)	
1089	1616	1166	C=C-1	XS	
1090	1617	1166	C=C-1	XS	
1091	1620	23	GONC	SKPD3 (1622)	
1092	1621	1046	C=C+1	X	NEW LINK READY
1093	1622	152	AB EX	WPT	PUT THE NEW LINK BACK
1094	1623	1	GOSUB	PTLINK	
1094	1624	0			
1095	1625	630	C=M		
1096	1626	412	A=C	WPT	GET ADDRESS OF BYTE BEFORE LINK
1097	1627	1204	S7=	0	TRAVERSE THE LINK THIS TIME
1098	1630	1	GOLONG	NXLDEL	AROUND WE GO AGAIN.
1098	1631	2			
1099					

*
 * GENLNK - GENERATE LINK
 * GIVEN 2 ADDRESSES IN A[0-3](LARGER) AND C[0-3](SMALLER),
 *- CREATES THE NECESSARY LINK TO GO UP THE CHAIN FROM C TO A.
 *- THIS LINK IS STORED IN THE C ADDRESS IN MEMORY.
 * ASSUMES AND RETURNS PT=3.
 * USES A[0-3], B[0-3], M, AND 1 SUB LEVEL. C SAVED.
 * FOR THE SPECIAL CASE OF A=C[0-3], 0 IS STORED AS THE LINK.
 *

1109	1632	GENLNK	530	M=C	
1110	1633	1512	? A#0	WPT	TOP OF CHAIN?
1111	1634	27	GOC	++2 (1636)	NO
1112	1635	112	C=0	WPT	YES, CREATE A ZERO LINK.
1113	1636	1112	C=A-C	WPT	CREATE LINK
1114	1637	43	GONC	GENLNK1 (1643)	
1115	1640	1146	C=C-1	X	FIX UP ADDRESS
1116	1641	1142	C=C-1	PT	

```

1117 1642          1142 C=C-1 PT          IN THE BORROW CASE
1118 1643 GENLK1  746 C=C+C X          MAKE COMPACT LINK
1119 1644          746 C=C+C X
1120 1645          746 C=C+C X
1121 1646          746 C=C+C X
1122 1647          23 GONC  **2    (1651)
1123 1650          1042 C=C+1 PT
1124 1651          1712 C SR  WPT
1125 1652          1420 LC    12          CREATE LINK CHAR.
1126 1653          34 PT=    3          PUT POINTER BACK
1127 1654          412 A=C    WPT          STORE LINK
1128 1655          630 C=M
1129 1656          1 GOSUB  PTLNKA
1129 1657          0
1130 1660          630 C=M          FIX UP
1131 1661          1740 RTN          DONE
*
* INSSUB - INSERT SUBROUTINE
* THIS SUBROUTINE SETS UP THE CALCULATOR FOR AN INSERT.
* IF THE LINE NUMBER # 0 OR THE CURRENT LINE # END, THEN
*- THE PC IN ADVANCED PAST THE CURRENT LINE, S9 IS SET TO 1,
*- AND THE LINE # IS INCREMENTED BY 1.
* CC13-10J IS SAVED IN AC13-10J.
* USES A[0-3],B[0-3],C,S[0-7,9].
*
1141 1662 INSSUB  1 GOSUB  PATCH3          THIS PATCH CHECKS PRIVACY FOR DATA E
1141 1663          0
1142 1664          1514 ?S12=1          IS THIS A PRIVATE PROGRAM?
1143 1665          1 GOLC  ERRPR          YES, SAY PRIVATE.
1143 1666          3
1144 1667 INSUBA  256 AC EX          SAVE C IN A
1145 1670          1 GOSUB  GETPC
1145 1671          0
1146 1672          1104 S9=    0          DISABLE BACKSTEP IN INBYT ERROR
1147 1673          1770 C=REGN 15          GET LINE NUMBER
1148 1674          1346 ? C#0 X          NON ZERO LINE NUMBER?
1149 1675          63 GONC  INSUB1 (1703) NO, DON'T SKPLIN, BUT INC LINNUM.
1150 1676          1 GOSUB  SKPLIN          SKIP A LINE
1150 1677          0
1151 1700          514 ?S6=1          HIT AN END?
1152 1701          67 GOC  INSUB2 (1707) YES, DON'T INCREMENT LINNUM.
1153 1702          1110 S9=    1          ENABLE BACKSTEP ON ERROR.
1154 1703 INSUB1  1 GOSUB  GETLIN          INCREMENT LINE NUMBER
1154 1704          0
1155 1705          1046 C=C+1 X
1156 1706          1750 REGN=C 15          STORE AWAY AGAIN
1157 1707 INSUB2  1 GOLONG PUTPC          PUT ADDRESS AWAY AND RETURN
1157 1710          2
1158
1159
1160
1161
1162
1163
1164
*
* GOSUB0,GOSUB1,GOSUB2,GOSUB3 - GOSUB LONG WITHIN A 4K ROM TO AN
*- ADDRESS WITHIN A SPECIFIED 1K ROM.
* THESE ROUTINES ARE THE SAME AS GOSUB EXCEPT INSTEAD OF ASSUMING THAT
*- THE DESTINATION ADDRESS IS WITHIN THE CURRENT 1024 WORD ROM, THE

```

*- DESTINATION ROM IS SPECIFIED BY THE CALL. I.E. TO GOSUB TO A
 *- SUBROUTINE IN ROM1 OF A 4 ROM CHIP, ONE WOULD USE:

```
*   GOSUB GOSUB1
*   XDEF <NAME>
```

* WARNING!!! IF YOU SPECIFY THE WRONG ROM, THE CALL WILL GO TO THE
 *- ADDRESS YOU SPECIFY RATHER THAN THE CORRECT ONE. THIS IS A PAINFUL
 *- ERROR TO FIND SINCE IT RESULTS IN JUMPS TO THE MIDDLE OF NOWHERE.

*
 * USES C PLUS 1 ADDITIONAL SUBROUTINE LEVEL TEMPORARILY.

*
 * GOL0,GOL1,GOL2,GOL3 - GOLONG TO ANYWHERE IN A 4K ROM.
 * SAME AS GOLONG EXCEPT DESTINATION 1K ROM SPECIFIED AS IN GOSUB[0-3]
 * USES C PLUS 1 SUBROUTINE LEVEL TEMPORARILY.

*
 * INTERNAL SUBROUTINE FOR GOSUB[0-3] AND GOL[0-3]

```
*
1187          ENTRY GOL0
1188          ENTRY GOSUB0
1189          ENTRY GOL1
1190          ENTRY GOSUB1
1191          ENTRY GOL2
1192          ENTRY GOSUB2
1193          ENTRY GOL3
1194          ENTRY GOSUB3
1195 1711 GSUBS1 1460 CXISA          GET 10 LSB OF ADDRESS PLUS 00
1196 1712          1732 C SR      M    PREPARE TO CONCATINATE 12 BITS TO
1197 1713          1732 C SR      M    TOP 4 BITS OF GOSUB ADDRESS
1198 1714          1732 C SR      M
1199 1715          756 C=C+C          ALIGN SO THAT A MANTISSA INCREMENT
1200 1716          756 C=C+C          WILL CHANGE BIT 10 OF THE FINAL ADDRE
1201 1717          1740 RTN
1202 1720 GOL0    660 C=STK          GET CALLING ADDRESS
1203 1721          53 GOTO      GSB0A (1726) GO TO IT (IN ROM 0)
1204 1722 GOSUB0  660 C=STK          GET CALLING ADDRESS
1205 1723          1072 C=C+1 M      INCREMENT PAST ARGUMENT
1206 1724          560 STK=C        PUT BACK FOR SUBROUTINE RETURN
1207 1725          1172 C=C-1 M     DECREMENT TO GET ARGUMENT
1208          LEGAL
1209 1726 GSB0A    1 GOSUB      GSUBS1    PREPARE ADDRESS
1209 1727          0
1210 1730          363 GOTO      GSB00 (1766) FINISH UP
1211 1731 GOL1    660 C=STK
1212 1732          53 GOTO      GSB1A (1737)
1213 1733 GOSUB1  660 C=STK
1214 1734          1072 C=C+1 M
1215 1735          560 STK=C
1216 1736          1172 C=C-1 M
1217          LEGAL
1218 1737 GSB1A    1 GOSUB      GSUBS1
1218 1740          0
1219 1741          243 GOTO      GSB01 (1765)
1220 1742 GOL2    660 C=STK
1221 1743          53 GOTO      GSB2A (1750)
1222 1744 GOSUB2  660 C=STK
1223 1745          1072 C=C+1 M
1224 1746          560 STK=C
1225 1747          1172 C=C-1 M
1226          LEGAL
1227 1750 GSB2A    1 GOSUB      GSUBS1
```



```

1227 1751      0
1228 1752      123 GOTO GSBQ2 (1764)
1229 1753 GOL3  660 C=STK
1230 1754      53 GOTO GSB3A (1761)
1231 1755 GOSUBZ 660 C=STK
1232 1756      1072 C=C+1 M
1233 1757      560 STK=C
1234 1760      1172 C=C-1 M
1235          LEGAL
1236 1761 GSB3A  1 GOSUB GSUBS1
1236 1762      0
1237 1763 GSBQ3  1072 C=C+1 M
1238 1764 GSBQ2  1072 C=C+1 M
1239 1765 GSBQ1  1072 C=C+1 M
1240 1766 GSBQ0  1574 RCR 12
1241 1767      756 C=C+C
1242 1770      756 C=C+C
1243 1771      740 GOTOC

SELECT ROM 3 OF CHIP
SELECT ROM 2 OF CHIP
SELECT ROM 1 OF CHIP
MOVE ADDRESS ALMOST INTO PLACE
ALIGN DESTINATION ADDRESS ON
DIGIT BOUNDARIES.
GO!

```

* GSB000,GSB256,GSB512,GSB768 - FAST ABSOLUTE GOSUB
 * THESE FOUR ENTRY POINTS TO THE SAME ROUTINE PROVIDE A MEANS
 *- FOR FAST 2 WORD GOSUBS IN PORT ADDRESSED MICROCODED PLUG-IN
 *- ROMS. THE SUBROUTINE CALLED MUST HAVE ITS FIRST WORD LOCATED
 *- ON A LOCAL 256(DEC) BOUNDARY AND THE GOSUBS REFERENCING THE
 *- SUBROUTINE MUST BE LOCATED WITHIN THAT 256 WORD BLOCK.
 * I.E. A SUBROUTINE COULD BE LOCATED STARTING AT LOCATION 512
 *- (1000 OCT) IN SOME ROM AND BE CALLED WITH A SINGLE 2 WORD
 *- GOSUB [GOSUB GSB256] ANYWHERE BETWEEN LOCATIONS 512 AND 767.
 *
 * BEWARE!!! - THIS ROUTINE IS DUMB. IF YOU CALL GSB256 FROM LOCATION
 *- 700 IT WILL NOT GO TO 256 BUT TO 512 AS THE LABELS ARE FOR
 *- PROGRAMMING CONVENIENCE ONLY. BE CAREFUL WHEN YOU USE THESE ROUTINES.
 *
 * USES ONLY C[2-6] PLUS 1 SUBROUTINE LEVEL TEMPORARILY.
 *

```

1261          ENTRY GSB000
1262          ENTRY GSB256
1263          ENTRY GSB512
1264          ENTRY GSB768
1265          GSB000
1266          GSB256
1267          GSB512
1268 1772 GSB768  660 C=STK
1269 1773          560 STK=C
1270 1774          1074 RCR 2
1271 1775          106 C=0 X
1272 1776          1574 RCR 12
1273 1777          740 GOTOC
1274
1275
1276
1277
1278          UNLIST
1281          END

```

ERRORS : 0

SYMBOL TABLE

CLPERR	1246	-	1243
CLRP1	1237	-	1217
CLRP2	1224	-	1245
CLRP3	1277	-	1236
CLRPGM	1214	-	
CLRREG	525	-	
CPY100	556	-	
CPY110	565	-	561
CPY120	571	-	550
CPY130	575	-	570
CPY140	607	-	603
CPY145	651	-	702
CPY150	652	-	661
CPY160	662	-	656
CPY170	703	-	674
CPYNE	554	-	557
DELLIN	1406	-	
DELLN1	1426	-	1420
DELLN2	1421	-	1430
DELNHH	1250	-	
ERR110	1373	-	
ERR120	1377	-	1374
ERR130	1401	-	1376
ERROR	1365	-	
ERRPR	604	-	
ERRRAM	562	-	572
ERRSUB	1350	-	
ERRTN	1364	-	1405
GBA1	1045	-	1023
GBA2	1043	-	1025
GBA3	1041	-	1027
GBA4	1037	-	1031
GBA5	1015	-	1033
GBA6	1034	-	
GBA6A	1035	-	1064
GENLK1	1643	-	1637
GENLNK	1632	-	
GOL0	1720	-	
GOL1	1731	-	
GOL2	1742	-	
GOL3	1753	-	
GOSUB0	1722	-	
GOSUB1	1733	-	
GOSUB2	1744	-	
GOSUB3	1755	-	
GSB000	1772	-	
GSB0A	1726	-	1721
GSB1A	1737	-	1732
GSB256	1772	-	
GSB2A	1750	-	1743
GSB3A	1761	-	1754
GSB512	1772	-	
GSB768	1772	-	
GSBQ0	1766	-	1730
GSBQ1	1765	-	1741
GSBQ2	1764	-	1752

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

GSEBQ3	1763	-			
GSUBS1	1711	-			
GTFEN1	353	-			
GTFEND	350	-			
GTLINK	1116	-			
GTLNKA	1107	-	1117	1104	
INSSUB	1662	-			
INSUB1	1703	-	1675		
INSUB2	1707	-	1701		
INSUBA	1667	-			
MOVRI0	542	-	533		
MOVREG	534	-			
NFRKBX	722	-	766		
PACK	3	-	1		
PACKE	2	-			
PACKN	0	-			
PAK100	24	-	32		
PAK102	31	-	53		
PAK105	34	-	30		
PAK108	51	-	23		
PAK110	60	-	33		
PAK115	72	-	57		
PAK117	117	-	112	105	
PAK118	120	-	116	114	110
PAK120	54	-	61		
PAK130	124	-			
PAK200	125	-			
PAK210	127	-	141		
PAK220	142	-	137		
PAK230	154	-	151	147	
PAK240	170	-	155		
PAK250	172	-	153	145	143
PAK260	173	-	171		
PAK265	206	-	253	201	
PAK270	213	-	235		
PAK275	231	-	225		
PAK280	240	-	205	203	177
PAKEND	254	-			
PAKSPC	362	-			
PATCH1	734	-			
PATCH2	741	-			
PATCH3	756	-			
PATCH5	763	-			
PKASN1	446	-	442		
PKASN2	451	-	445		
PKASN3	453	-	476	450	433
PKEND1	271	-	276		
PKEND2	301	-	272		
PKEND3	317	-	300		
PKEND4	277	-	310		
PKEND5	324	-			
PKEND6	337	-	347		
PKIO10	461	-	437		
PKIO20	473	-	506		
PKIOAS	424	-			
PKN10	434	-	457		
PKSPC1	373	-	365		
PKSPC3	406	-			
PR15RT	1337	-			
PTBYTA	1443	-			

PTBYTP	1450	--	
PTLINK	1432	-	
PTLNKA	1433	-	
PTLNKE	1441	-	1453
PUTPC	1467	-	1460 1456
PUTPC1	1501	-	1477
PUTPC2	1504	-	1502
PUTPC3	1507	-	1505 1475
PUTPCA	1471	-	
PUTPCD	1454	-	
PUTPCF	1461	-	
PUTPCX	1457	-	
PUTREG	536	-	
SKPD1	1575	-	1541
SKPD2	1621	-	1615
SKPD3	1622	-	1620
SKPD4	1542	-	1534
SKPD6	1565	-	1554
SKPD7	1544	-	1531
SKPDEL	1511	-	
SKPOL	1514	-	
SSTBST	1335	-	
TABUPL	1000	-	
TBLGBA	1020	-	
TRGSET	724	-	
ULINK1	1077	-	1074
ULINK2	1101	-	1076
ULINK3	1105	-	1102
UPLB0	1056	-	1001
UPLB1	1047	-	1003
UPLB2	1045	-	1005
UPLB3	1043	-	1007
UPLB4	1041	-	1011
UPLB5	1037	-	1013
UPLB6	1014	-	
UPLINK	1065	-	
XBST	1120	-	
XBST1	1131	-	1172
XCLPX1	1264	-	1221
XCOPY	545	-	
XDELA	1261	-	1256
XDELET	1257	-	
XDELEX	1333	-	1364 1346 1327 1263 1255 1252
XDELM1	1300	-	1307
XDELM2	1310	-	1304
XPACK	0	-	
XSST	1140	-	
XSSTR	1173	-	1153
XSSTR1	1177	-	1174

ENTRY TABLE

CLRPGM	1214	-
CLRREG	525	-
DELLIN	1406	-
DELHNN	1250	-
ERR110	1373	-
ERR120	1377	-
ERROR	1365	-
ERRPR	604	-
ERRRAM	562	-
ERRSUB	1350	-
GENLNK	1632	-
GOL0	1720	-
GOL1	1731	-
GOL2	1742	-
GOL3	1753	-
GOSUB0	1722	-
GOSUB1	1733	-
GOSUB2	1744	-
GOSUB3	1755	-
GSB000	1772	-
GSB256	1772	-
GSB512	1772	-
GSB768	1772	-
GSUBS1	1711	-
GTFEN1	353	-
GTFEND	350	-
GTLINK	1116	-
GTLNKA	1107	-
INSSUB	1662	-
MOVREG	534	-
PACKE	2	-
PACKN	0	-
PAK200	125	-
PAKEND	254	-
PAKSPC	362	-
PATCH1	734	-
PATCH2	741	-
PATCH3	756	-
PATCH5	763	-
PKIOAS	424	-
PR15RT	1337	-
PTBYTA	1443	-
PTBYTP	1450	-
PTLINK	1432	-
PTLNKA	1433	-
PTLNKB	1441	-
PUTPC	1467	-
PUTPCA	1471	-
PUTPCD	1454	-
PUTPCF	1461	-
PUTPCX	1457	-
PUTREG	536	-
SKPDEL	1511	-
SSTBST	1335	-
TRGSET	724	-
UPLINK	1065	-

XBST	1120	-
XCOPY	545	-
XDELET	1257	-
XPACK	0	-
XSST	1140	-

EXTERNAL REFERENCES

ANNOUT	1341			
ANNOUT	1342			
ASRCH	551	1237		
ASRCH	552	1240		
BSTCAT	1123			
BSTCAT	1124			
BSTEP	1127	1377		
BSTEP	1130	1400		
BSTEPH	1331			
BSTEPH	1332			
CLRREG	451	501	534	
CLRREG	452	502	535	
CPGMHD	754			
CPGMHD	755			
DCPL00	340			
DCPL00	341			
DCPLRT	64			
DCPLRT	65			
DECAD	1454			
DECAD	1455			
DECADA	77	246	1522	
DECADA	100	247	1523	
DECMPL	720			
DECMPL	721			
DELLIN	1301			
DELLIN	1302			
DFKBCK	1177			
DFKBCK	1200			
DFRST9	1131			
DFRST9	1132			
DRSY25	1134			
DRSY25	1135			
ERRNE	554	1246		
ERRNE	555	1247		
ERROR	562	604	760	
ERROR	563	605	761	
ERRPR	1665			
ERRPR	1666			
ERRSUB	1365			
ERRSUB	1366			
FIXEND	742	1315		
FIXEND	743	1316		
FLINK	1224			
FLINK	1225			
FLINKA	1516			
FLINKA	1517			
FLINKP	1310			
FLINKP	1311			
FSTIN	54			
FSTIN	55			
GENLNK	322	377		
GENLNK	323	400		
GETLIN	1157	1204	1324	1703
GETLIN	1160	1205	1325	1704
GETPC	102	1147	1222	1267 1670
GETPC	103	1150	1223	1270 1671

GSUBS1	1726	1737	1750	1761	
GSUBS1	1727	1740	1751	1762	
GTFEND	16				
GTFEND	17				
GTLINK	51	362	1526	1573	
GTLINK	52	363	1527	1574	
GTLNKA	1601				
GTLNKA	1602				
GTO.5	1322				
GTO.5	1323				
INCAD2	37	70	402	410	1546
INCAD2	40	71	403	411	1547
INCADA	72	163	735	1422	1514 1571
INCADA	73	164	736	1423	1515 1572
INSSUB	1265				
INSSUB	1266				
LINNUM	1343	1403			
LINNUM	1344	1404			
MEMLFT	611				
MEMLFT	612				
MOVREG	446				
MOVREG	447				
MSG	3				
MSG	4				
MSGC	1371				
MSGC	1372				
MSGPR	606				
MSGRAM	564				
MSGROM	762				
MSGWR	5				
NFRKB	722	1333			
NFRKB	723	1334			
NFRKB1	1136				
NFRKB1	1137				
NULTST	1202				
NULTST	1203				
NXBYTA	133	213	415	1550	
NXBYTA	134	214	416	1551	
NXLDEL	206	1413	1630		
NXLDEL	207	1414	1631		
NXLSST	1155				
NXLSST	1156				
NXLTX	421				
NXLTX	422				
PACKE	620				
PACKE	621				
PAK200	236				
PAK200	237				
PAKEND	371				
PAKEND	372				
PAKSPC	1512				
PAKSPC	1513				
PATCH2	1226				
PATCH2	1227				
PATCH3	1662				
PATCH3	1663				
PATCH5	1272				
PATCH5	1273				
PKIOAS	10				
PKIOAS	11				

PTBYTA	45	161	1424
PTBYTA	46	162	1425
PTBYTM	750		
PTBYTM	751		
PTBYTP	221		
PTBYTP	222		
PTLINK	1623		
PTLINK	1624		
PTLNKA	1656		
PTLNKA	1657		
PUTPC	1170	1707	
PUTPC	1171	1710	
PUTPCL	1230		
PUTPCL	1231		
PUTPCX	250	716	
PUTPCX	251	717	
PUTREG	504		
PUTREG	505		
ROMH05	566		
ROMH05	567		
ROMHED	573		
ROMHED	574		
RSTMS0	6	1350	
RSTM30	7	1351	
RSTSEQ	1337		
RSTSEQ	1340		
RUNNK	1212		
RUNNK	1213		
SETSST	1210		
SETSST	1211		
SKPLIN	1676		
SKPLIN	1677		
SRBMAP	1563		
SRBMAP	1564		
SSTBST	1125	1145	
SSTBST	1126	1146	
SSTCAT	1143		
SSTCAT	1144		
STBT30	43		
STBT30	44		
STBT31	20		
STBT31	21		
STOPS	1401		
STOPS	1402		
TBITMP	1561		
TBITMP	1562		
UPLINK	24		
UPLINK	25		

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES 022000-23777

*

4	FILE	CN9B
5	ENTRY	SAR021
6	ENTRY	SAR022

8		ENTRY	SERR		
9		ENTRY	SEARCH		
10		ENTRY	ASRCH		
11		ENTRY	SNRON		
12		ENTRY	DOSRCH		
13		ENTRY	XGHN10		
14		ENTRY	XGHN12		
15		ENTRY	XEQC01		
16		ENTRY	GTSPCH		
17		ENTRY	SNR12		
18		ENTRY	SNR10		
19		ENTRY	SGTO19		
20		ENTRY	SAROM		
21		ENTRY	XRTN		
22		ENTRY	RTN30		
23		ENTRY	XGI		
24		ENTRY	ROW0		
25		ENTRY	ROW11		
26		ENTRY	ROW12		
27		ENTRY	XXEQ		
28		ENTRY	XGTO		
29					
30					
31					
32					
33	0	SNROM	146 AB EX	X	AL1:03_LBL
34	1		674 RCR	11	SET ADDR TO NXT WORD
35	2	SNR09	1434 PT=	1	-
36	3	SNR010	1072 C=C+1	M	-
37	4	SNR012	1460 CXISA		GET ROM WORD
38	5		1166 C=C-1	XS	1ST BYTE?
39	6		1366 ? C#0	XS	1ST BYTE?
40	7		1747 GOC	SNR010 (	3) NOPE
41	10		1342 ? C#0	PT	ROW0?
42	11		77 GOC	SNR020 (	20) NOPE
43	12		1352 ? C#0	WPT	NULL?
44	13		1703 GONC	SNR010 (	3) YES
45	14		1146 C=C-1	X	LBL_LBL-1
46	15		1552 ? A#C	WPT	CORRECT SHORT LBL?
47	16		1657 GOC	SNR010 (	3) NOPE
48	17		403 GOTO	SNR040 (	57) RETURN
49					
50	20	SNR020	1042 C=C+1	PT	ROW 12?
51	21		1042 C=C+1	PT	-
52	22		1042 C=C+1	PT	-
53	23		1042 C=C+1	PT	-
54	24		1573 GONC	SNR010 (	3) NOPE
55	25		1634 PT=	0	LONG LBL?
56	26		1042 C=C+1	PT	-
57	27		237 GOC	SNR030 (	52) YES
58	30		1042 C=C+1	PT	CHAIN?
59	31		1517 GOC	SNR09 (	2) NOPE
60	32		1072 C=C+1	M	GET 3RD BYTE
61	33		1072 C=C+1	M	-
62	34		1460 CXISA		-
63	35		1434 PT=	1	ALBL?
64	36		1042 C=C+1	PT	-
65	37		1447 GOC	SNR010 (	3) YES
66	40		116 C=0		-

```

67 41      1114 ?S9=1      2ND END?
68 42      177 GOC      SNR050 ( 61)
69 43      1110 S9=      1      1ST END FOUND
70 44      216 B=A      SAVE LBL
71 45      1 GSBLNG:IRONHED      GET BEGIN ADDR
71 46      0
72 47      156 AB EX      PUT BACK LBL
73 50      674 RCR      11      -
74 51      1313 GOTO      SNR09 ( 2) CONTINUE SEARCH
75
76 52 SNR030 1072 C=C+1 M      CORRECT LONG LBL?
77 53      1460 CXISA      -
78 54      1546 ? A#C      X      -
79 55      1257 GOC      SNR09 ( 2) NOPE
80 56      1172 C=C-1 M      POSITION ADDRESS
81 57 SNR040 1172 C=C-1 M      -
82 60      74 RCR      3      C[13:0]_ROM ADDRESS
83 61 SNR050 34 PT=      3
84 62      1740 RTH
85
86
87
88
89
* SEARCH - SEARCH FOR NUMERIC LABEL
*- SEARCH THE CURRENT PROGRAM FOR THE DESIGNATED
*- LONG OR SHORT NUMERIC LABEL. (SEARCH'S IN ROM
*- OR RAM)
*- IN: A,X=NUMERIC LABEL (IF PC IS IN RAM, A[2] MAY BE NON-ZERO)
*- PT= 3
*- OUT: C=0 IMPLIES THE LABEL WAS NOT FOUND
* OTHERWISE
*- C[13:0]= LABEL ADDRESS (ADDRESS OF BYTE BEFORE LABEL)
*- PT= 3
*- CHIP 0 SELECTED
*- USES: STATUS BITS 9,6,0, G, A[13:0], C[13:0],B[13:0]
* S6=1 IMPLIES PROGRAM COUNTER IS AT THE FIRST BYTE
* OF A THREE BYTE INSTRUCTION ON INPUT. THIS
* ONLY OCCURS WHEN EXECUTING LONG GTONN AND XEQNN
* OUT OF PROGRAM MEMORY.
* S6=0 IMPLIES PROGRAM COUNTER IS AT A STANDARD POSITION
* (I.E. AT THE BYTE BEFORE THE FIRST BYTE OF A LINE).
*- USES: 2 SUBROUTINE LEVELS
109
110
111
112      ENTRY SEARCH1
113 63 SEARCH 504 S6=      0
114 64 SEARCH1 1104 S9=      0      1ST END NOT FOUND
115 65      206 B=A      X      SAVE A
116 66      1 GSBLNG:GETPC      GET ADDR
116 67      0
117 70      314 ?S10=1      ROM?
118 71      1077 GOC      SNR0M ( 0) YES
119 72      306 C=B      X      G_LBL
120 73      674 RCR      11      -
121 74      130 G=C      -
122 75      514 ?S6=1      -
123 76      33 GONC      SNR12 ( 101)
124 77 SNR10 1 GSBLNG:INCAD2

```

124	100	0			
125	101	SNR12	1	GSBLNG INKBYTA	GET A BYTE
125	102	0			
126	103	1574	ROR	12	SET PTR
127	104	1202	C=C	PT	1 BYTE FC?
128	105	742	C=C+C	PT	-
129	106	1737	GOC	SNR12 (101)	YES
130	107	742	C=C+C	PT	2 BYTE FC?
131	110	467	GOC	SNR70 (156)	YES
132	111	742	C=C+C	PT	3 BYTE FC?
133	112	1657	GOC	SNR10 (77)	YES
134	113	742	C=C+C	PT	ROW 0?
135	114	43	GONC	SNR50 (120)	YES
136	115	1	GSBLNG INXLTX		ITS A TEXT FC
136	116	0			
137	117	1623	GOTO	SNR12 (101)	-
138					
139	120	SNR50	1366	? C#0 XS	NULL?
140	121		1603	GONC SNR12 (101)	YES
141	122		1604	SO= 0	-
142	123		1166	C=C-1 XS	LBL_LBL-1
143	124		1074	ROR 2	-
144	125	SNR55	256	AC EX	CC[6:3]_RAM ADDRESS
145	126		674	ROR 11	-
146	127		1634	PT= 0	CC[1:0]_LBL
147	130		230	C=G	-
148	131		426	A=C XS	-
149	132		34	PT= 3	-
150	133		1546	? A#C X	CORRECT LBL?
151	134		147	GOC SNR60 (150)	NOPE
152	135		74	ROR 3	YES, PNT AT PREV STEP
153	136		416	A=C	-
154	137		1	GSBLNG DECADA	DEC RAM ADDR
154	140		0		
155	141		1614	?SO=1	LNG LBL?
156	142		1	GSUBC DECADA	YES, DEC RAM ADDR AGAIN
156	143		1		
157	144		116	C=0	RE-ENABLE CHIP 0
158	145		1160	DADD=C	
159	146		252	AC EX WPT	CC[3:0]_LBL ADDR
160	147		1740	RTN	RETURN FROM SEARCH
161					
162	150	SNR60	74	ROR 3	AI[3:0]_RAM ADDR
163	151		416	A=C	-
164	152		1273	GOTO SNR12 (101)	-
165					
166	153	SNR65	1	GSBLNG INCADA	INC RAM ADDR
166	154		0		
167	155		1243	GOTO SNR12 (101)	-
168					
169	156	SNR70	1342	? C#0 PT	ROW 12?
170	157		1747	GOC SNR65 (153)	NOPE
171	160		1066	C=C+1 XS	LONG LBL?
172	161		307	GOC SNR80 (211)	YES
173	162		1066	C=C+1 XS	X<>?
174	163		1707	GOC SNR65 (153)	YES
175	164		1	GSBLNG GTLINK	CC[13]_3RD BYT OF CHAIN
175	165		0		
176	166		1076	C=C+1 S	ALBL?
177	167		1647	GOC SNR65 (153)	YES

NOMAS
 Not Manufacturer Supported
 Resident Address NOT to contain manufacturer

```

178 170      1114 ?S9=1      2ND END?
179 171      57 GOC      SNR72 ( 176 ) YES
180 172      1 GOSUB      CPGMHD      GOTO PGM HEAD
180 173      0
181 174      1110 S9=      1      1ST END FOUND
182 175      1043 GOTO      SNR12 ( 101 )
183      SNR72
* SEARCH GIVES UP HERE
185 176      514 ?S6=1      PGM CTR IN ODD PLACE?
186 177      73 GONC      SNR73 ( 206 ) NO
187 200      1 GOSUB      GETPC      YES
187 201      0
188 202      1 GOSUB      INCAD2      SET IT TO END OF 3 BYTE FC
189 203      0
189 204      1 GOSUB      PUTPC
189 205      0
190 206 SNR73 116 C=0      NOT FOUND
191 207      1160 DADD=C      RE-ENABLE CHIP 0
192 210      1740 RTN      -
193
194 211 SNR80 1 GSBLNG INXBYTA      GET 2ND BYT OF LNG LBL
194 212      0
195 213      1610 S0=      1      LNG LBL
196 214      1113 GOTO      SNR55 ( 125 ) CORRECT LNG LBL?
197
198
199
200
201
202      ENTRY XGA00

```

```

*
* XGA - XEQ/GTO ALPHA
*- PLACE THE PROGRAM COUNTER AT THE SPECIFIED ALPHA
*- STRING LABEL ADDRESS. IN THE CASE OF A XEQ, THE
*- RETURN STACK IS PUSHED THROUGH A TRANSFER TO "XEQC"
*- IN: S7= 1/0 IMPLIES XEQ/GTO FUNCTION
*- S9= 1 IMPLIES AN ALPHA SEARCH HAS BEEN
*- PREVIOUSLY PERFORMED
*- M[3:0]=ADDRESS OR N=ALPHA STRING
*
213 215 XGA00 1 GOSUB SAVRC      SAVE RETURN ADDRESS
213 216      0
214 217      630 C=M      C=ALBL OR ADDRESS
215 220      1314 ?S13=1      RUNNING?
216 221      277 GOC      XGI52 ( 250 ) YES
217 222      114 ?S4=1      SST?
218 223      257 GOC      XGI52 ( 250 ) YES
* MUST BE FROM KEYBOARD
220 224      1114 ?S9=1      ALREADY FOUND?
221 225      233 GONC      XGI52 ( 250 ) NO. (AGTO FROM KEYBOARD)
222 226      313 GOTO      XGI54 ( 257 ) MUST BE AXEQ OF USER LABEL
*
* XGI - XEQ/GTO INDIRECT
*- PLACE THE PROGRAM COUNTER AT THE NUMERIC OR
*- ALPHA LABEL FOUND IN THE SPECIFIED REGISTER.
*- IN THE CASE OF AN XEQ, THE SUBROUTINE RETURN
*- STACK IS PUSHED THROUGH A TRANSFER TO "XEQC".
*- IN: STATUS= 2ND BYTE OF FUNCTION CODE
*- STATUS BIT 7 = 0/1 IMPLIES GTO/XEQ FUNCTION
*- OUT: GTO- CHIP 0 SELECTED

```

```

*-      XEQ- PT= 3
*-      C[3:0]= LABEL ADDRESS
*-      CHIP 0 SELECTED
*-  USES: A,B,C,M,N,G,STATUS BITS, STATUS BITS 8 & 9
*-      REG 9, REG 10 DIGITS 0,1
*-  USES: 4 SUBROUTINE LEVELS
238
239
240
241
242 227 XGI30 1534 PT= 12          TEST FOR NULL LBL
243 230      102 C=0      PT      -
244 231      1356 ? C#0      -
245 232      453 GONC      SERRXF ( 277 ) YES
246 233      416 A=C      FORMAT
247 234      116 C=0      -
248 235      1150 DADD=C      -
249 236      1434 PT= 1      -
250 237 XGI40 1512 ? A#0 WPT      - (END OF STRING2)
251 240      63 GONC      XGI50 ( 246 ) - (YES)
252 241      252 AC EX WPT      -
253 242      1574 RCR 12      -
254 243      1616 A SR      -
255 244      1616 A SR      -
256 245      1723 GOTO      XGI40 ( 237 ) -
257 246 XGI50 1074 RCR 2      -
258 247      530 M=C      -
259 250 XGI52 1150 REGN=C 9      -
260 251      1 GSBLNG ASRCH      SEARCH FOR ALPHA LBL
260 252      0      -
261 253      1356 ? C#0      FOUND?
262 254      233 GONC      SERRXF ( 277 ) NO
263 255      1014 ?S2=1      ROM?
264 256      37 GOC      XGI55 ( 261 ) YES
265 257 XGI54 304 S10= 0      NO. MUST BE RAM
266 260      223 GOTO      XGI60 ( 302 )
267 261 XGI55 1114 ?S9=1      MICROCODE?
268 262      173 GONC      XGI57 ( 301 ) NO. MUST BE USER LANG
269 263      214 ?S5=1      MAINFRAME?
270 264      647 GOC      SERR ( 350 ) YES. ERROR
271 265      530 M=C      SAVE ADDRESS IN M
272 266      1270 C=REGN 10      RETRIEVE RTN ADDR
273 267      1346 ? C#0 X      XEQ?
274 270      603 GONC      SERR ( 350 ) NO. AGTD ILLEGAL FOR
275      MICROCODE
276 271      1670 C=REGN 14      RESTORE SS0
277 272      1530 ST=C      -
278 273      630 C=M      C[3:0]=XADR
279 274      674 RCR 11      C[6:3]=XADR
280 275      1460 CXISA      GET WORD AT XADR
281 276      1346 ? C#0 X      PROGRAMMABLE FCN?
282 277 SERRXF 513 GONC      SERR ( 350 ) NO. ERROR
283 300      740 GOTOC      -
*
* 295      ENTRY XGI57
* XROM ENTERS AT XGI57
* ON ENTRY, ADDRESS OF FIRST BYTE OF DESTINATION LABEL IS IN C[3:0],
* AND RETURN ADDRESS IS IN R10[3:0] ALREADY PACKED FOR PUSH ONTO
* SUBROUTINE STACK
290 301 XGI57 310 S10= 1      ROM USER LANGUAGE

```

```

291 302 XGI60      34 PT=      3      -
292 303           412 A=C      WPT      -
293 304           1 GSBLNG-DECAD      -
293 305           0
294 306           243 GOTO      XGI07 ( 332) -
295
296 307 XGI        1 GOSUB      SAVRC      SAVE RETURN ADDRESS
296 310           0
297 311           1204 S7=      0      CLEAR S7 FOR ADRFCH
298 312           1 GSBLNG-ADRFCH      C[13:0]_REG CONTENTS
298 313           0
299 314           1176 C=C-1      S
300 315           1376 ? C#0      S      VALID # LABEL?
301 316           1113 GONC      XGI30 ( 227) NOPE
302 317           1 GSBLNG-BCDBIN      CNVRT BCD TO BINARY
302 320           0
303 321           406 A=C      X      VALID LABEL?
304 322           460 LDI
305 323           144 CON      100
306 324           1406 ? A<C      X
307 325           233 GONC      SERR ( 350) NOPE, # TOO BIG
308 326           1 GSBLNG-DOSRC1      SEARCH FOR LBL
308 327           0
309 330 XGI05      34 PT=      3      -
310 331           412 A=C      WPT      A[3:0]_LBL ADDR
*
* XGI07 - ENTRY POINT TO DO XEQ [RAM ADDRESS]
*         ADDED FOR WAND 2/13/80 JAVB.
*
* ON INPUT
* CHIP 0 ENABLED
* REG10 HAS PACKED RETURN ADDRESS (SEE SAVRTN)
* A[3:0] HAS RAM ADDRESS
* PT = 3
* NEVER RETURNS TO CALLING PROGRAM
*
321           ENTRY      XGI07
322 332 XGI07      1270 C=REGN 10      RETRIEVE RTN ADDR
323 333           1346 ? C#0      X      XEQ?
324 334           663 GONC      XGNN10 ( 422) NO, GTO
325 335           212 B=A      WPT      LBL ADDR TO B[3:0]
326 336           143 GOTO      XEQC01 ( 352)
327
328
329
329
330
331 337 GTSRCH      152 AB EX      WPT      A[1:0]_CORRESPONDING SHORT LBL
332 340           1606 A SR      X
333 341           1606 A SR      X
334 342           646 A=A-1      X
335           ENTRY      DOSRC1
336 343 DOSRC1      504 S6=      0      PGM CTR IS IN A STD PLACE
337 344 DOSRCH      1 GSBLNG-SEARCH1      SEARCH FOR NUMERIC LABEL
337 345           0
338 346           1356 ? C#0
339 347           1540 RTH C      FOUND?
340 350 SERR        1 GOLONG-ERRHE      YES
340 351           2      REPORT ERROR
341
342
343           "NON-EXISTENT"

```

344

```

* XEQC01 - XEQ COMMON LOGIC
*- IF IN KEYBOARD MODE, THE SUBROUTINE RETURN STACK
*- IS CLEARED, & THE PROGRAM IS SET TO RUNNING,
*- OTHERWISE THE SUBROUTINE STACK IS PUSHED AND THE
*- PROGRAM COUNTER IS SET TO THE DESIGNATED LABEL
*- ADDRESS
*- IN: 013:01= LABEL ADDRESS
*-      PT= 3
*-      REG 10 13:01 = RETURN ADDRESS ALREADY PACKED
*- OUT: CHIP 0 SELECTED
*- USES: 013:01, 0113:01, A113:01
*- USES: 1 SUBROUTINE LEVEL
*
* XEQ20 - SAME AS XEQC01 EXCEPT DOESN'T CHECK FOR KEYBOARD MODE
*

```

```

360
361
362
363 352 XEQC01 105 C=0      X          SELECT CHIP 0
364 353      1160 DADD=C          -
365 354      1314 ?S13=1        RUNNING?
366 355      57 GOC      XEQ20  ( 362 ) YES
367 356      1670 C=REGN 14      S8TFLAG?
368 357      1530 ST=C          -
369 360      114 ?S4=1          -
370 361      203 GONC      XEQ50  ( 401 ) NOPE
371 362 XEQ20 1270 C=REGN 10      GET RETURN ADDRESS
372 363      412 A=C      WPT      PUT RTH ADDR TO A13:01
373 364      1470 C=REGN 12
374 365      252 AC EX  WPT
375 366      374 RCR      10      PUSH STACK
376 367      416 A=C          -
377 370      1370 C=REGN 11      -
378 371      374 RCR      10      -
379 372      252 AC EX  WPT      -
380 373      1350 REGN=C 11      -
381 374      256 AC EX          -
382 375      1 GSBLNG.CLR5B3      FINISH PUSH
383 376      0
384 377      253 GOTO      XGNN12 ( 424 )
385 400 XEQ49 352 BC EX  WPT      KEYBOARD PATH
386 401 XEQ50 1 GSBLNG.CLR5B2      CLEAR RTH STACK
387 402      0
388 403      1 GOLONG RUN
389 404      2
390

```

```

* XGNN - XEQ/GTO NUMERIC (LONG FORM GTO)
*- PLACE THE PROGRAM COUNTER AT THE SPECIFIED NUMERIC
*- LABEL ADDRESS, COMPILING A DISPLACEMENT TO BE
*- STORED WITH THE FUNCTION UPON THE FIRST ENCOUNTER
*- OF THAT FUNCTION. (FOLLOWING A DECOMPILE) IN THE
*- CASE OF AN XEQ, THE RETURN STACK IS PUSHED THROUGH
*- A TRANSFER TO "XEQC".
*- IN: S1= 0/1 IMPLIES GTO/XEQ FUNCTION
*-      S9=1 IMPLIES A NUMERIC SEARCH HAS BEEN
*-      PREVIOUSLY PERFORMED

```



```

*-      C[1:0]= NUMERIC LABEL
*- OUT:  GTO- CHIP 0 SELECTED
*-      XEQ- C[3:0]= LABEL ADDRESS
*-      PT= 3
*-      CHIP 0 SELECTED
*- USES: STATUS BITS 0,1,6,9,9, A[13:0], B[13:0], C[13:0],
*-      M[13:0], G
*- USES: 4 SUBROUTINE LEVELS
409
410
411
412
413 405 XGTO 1404 S1= 0 GTO NH
414 406 XGNN 1314 ?S13=1 RUNNING?
415 407 177 GOC XGNN02 ( 426 ) YES
416 410 114 ?S4=1 SSTFLAG?
417 411 157 GOC XGNN02 ( 426 ) YES
418 412 406 A=C X A[1:0]_# LBL
419 413 630 C=M
420 414 1114 ?S9=1
421 415 1 GSUBNO: DOSRC1
421 416 0
422 417 1414 ?S1=1 XEQ?
423 420 1607 GOC XEQ49 ( 400 )
424 421 412 A=C WPT
425 422 XGNN10 1 GSBLNG: PUTFCX -
425 423 0
426 424 XGNN12 1 GOLONG: NFRPU -
426 425 2
427 426 XGNN02 314 ?S10=1 RON?
428 427 463 GONC XGNN20 ( 475 ) NOPE
429 430 412 A=C WPT -
430 431 2 A=0 PT -
431 432 1470 C=REGN 12 NO, A[2:0]_FULL REL ADDR.
432 433 674 RCR 11 -
433 434 1072 C=C+1 M -
434 435 1460 CXISA -
435 436 246 AC EX X -
436 437 266 AC EX XS -
437 440 1072 C=C+1 M DETERMINE SIGN
438 441 1460 CXISA -
439 442 1374 RCR 13 -
440 443 746 C=C+C X -
441 444 107 GOC XGNN15 ( 454 ) ADD.
442 445 1470 C=REGN 12 PGMCTR_PGMCTR-REL ADDR.
443 446 252 AC EX WPT -
444 447 712 A=A-C WPT -
445 450 XGNN05 1414 ?S1=1 XEQ?
446 451 XGNN06 1513 GONC XGNN10 ( 422 ) NOPE
447 452 212 B=A WPT B[3:0]_LBL ADDRESS
448 453 1073 GOTO XEQ20 ( 362 )
449
450 454 XGNN15 1470 C=REGN 12 PGMCTR_PGMCTR+REL ADDR.
451 455 512 A=A+C WPT -
452 LEGAL
453 456 1723 GOTO XGNN05 ( 450 ) -
*
455
456 457 XEQ 1410 S1= 1 XEQ
457 460 356 BC EX SAVE FC & 2ND BYTE IN B

```

458	461	1	GOSUB	GETPC	CALC RETURN ADDRESS
458	462	0			
459	463	1	GOSUB	INCAD	INCREMENT OVER 2ND AND
459	464	0			
460	465	1	GOSUB	INCAD	3RD BYTES OF XEQ IN
460	466	0			
461	467	252	C=A	WPT	COPY ADDR TO C13:01
461	470	412			
462	471	1	GOSUB	SAVR10	REQ ADDR IN BOTH A AND C
462	472	0			
463					SAVES RTH ADDR IN R10
464	473	356	BC EX		RESTORE C
465	474	1123	GOTO	XGNN (406)	
466					
467					
468	475	XGNN20	414	?S8=1	FULL REL ADDR?
469	476		73	GONC XGNN25 (505)	YES
470	477		352	BC EX WPT	NO, C12:01_FULL REL ADDR
471	500		1	GSBLNG GETPCA	-
471	501		0		
472	502		1	GSBLNG NXBYTA	-
472	503		0		
473	504		326	C=B XS	-
474	505	XGNN25	1346	? C#0 X	COMPILE?
475	506		463	GONC XGNN65 (554)	YES
476	507		102	C=0 PT	UNPACK REL ADDR &
477	510		752	C=C+C WPT	INC BYTE BY 2
478	511		752	C=C+C WPT	-
479	512		752	C=C+C WPT	-
480	513		1042	C=C+1 PT	-
481	514		742	C=C+C PT	-
482	515		746	C=C+C X	-
483	516		227	GOC XGNN50 (540)	-
484	517		1706	C SR X	-
485	520	XGNN30	352	BC EX WPT	B13:01_REL ADDR (MM)
486	521		1	GSBLNG GT3DET	GET 3RD BYTE
486	522		0		
487	523		730	MC EX	M_ORG STATUS, C_PGMCTR
488	524		152	AB EX WPT	A13:01_REL ADDR
489	525		1214	?S7=1	SUBTRACT?
490	526		203	GONC XGNN60 (546)	YES
491	527		506	A=A+C X	-
492	530		502	A=A+C PT	A13:01_LBL ADDR
493	531		123	GONC XGNN55 (543)	-
494	532		546	A=A+1 X	-
495	533	XGNN35	630	C=M	RESTORE ORG STATUS
496	534		1530	ST=C	-
497				ENTRY XGNN40	
498	535	XGNN40	106	C=0 X	RE-ENABLE CHIP 0
499	536		1160	DADD=C	
500	537		1113	GOTO XGNN05 (450)	
501					
502	540	XGNN50	1706	C SR X	-
503	541		1066	C=C+1 XS	-
504				LEGAL	
505	542		1563	GOTO XGNN30 (520)	-
506					
507	543	XGNN55	642	A=A-1 PT	BYTE_BYTE-2
508	544		642	A=A-1 PT	-
509				LEGAL	

```

510 545      1663 GOTO   XGNN35 ( 533) -
511
512 546 XGNN60 252 AC EX  WPT          A13:0J_PGMCTR-REL ADDR
513 547      1142 C=C-1  PT          -
514 550      1142 C=C-1  PT          -
515          LEGAL
516 551          1 GSBLNG:CALDSP      CALCULATE DISPLACEMENT
516 552          0
517 553      1603 GOTO   XGNN35 ( 533) -
518
519
520 554 XGNN65 410 S8=    1          -
521 555          1 GSBLNG:GT3DBT      GET 3RD BYTE
521 556          0
522 557      1204 S7=    0          DECOMPILE DOESN'T CLEAR
523                                     BIT 7 OF THE THIRD BYTE
524 560      1730 CST EX
525 561      406 A=C    X
526 562      510 S6=    1          PGM PTR IS AT 1ST BYTE
527                                     OF 3 BYTE FC
528 563          1 GSBLNG:DOSRCH      SEARCH RAM FOR LBL
528 564          0
529 565      412 A=C    WPT          CALCULATE DISPLACEMENT
530 566      730 MC EX          - (M_LBL ADDRESS)
531 567      1546 ? A#C  X
532 570      47 GOC    XGNN70 ( 574) -
533 571      1402 ? A<C  PT
534 572      63 GONC   XGNN80 ( 600) -
535 573      33 GOTO    XGNN75 ( 576) -
536 574 XGNN70 1406 ? A<C  X
537 575      33 GONC   **3      ( 600) -
538 576 XGNN75 404 S8=    0
539 577      252 AC EX  WPT
540 600 XGNN80 1106 C=A-C  X          - (#REGS)
541 601      1102 C=A-C  PT          - (#BYTES)
542 602      43 GONC   **4      ( 606) - (NO CARRY)
543 603      1146 C=C-1  X          - (REG_REG-1)
544 604      1142 C=C-1  PT          - (BYTE_BYTE-2)
545 605      1142 C=C-1  PT
546 606      746 C=C+C  X          PACK DISPLACEMENT
547 607      746 C=C+C  X
548 610      746 C=C+C  X
549 611      746 C=C+C  X
550 612      23 GONC   **2      ( 614) -
551 613      1042 C=C+1  PT
552 614      1712 C SR   WPT
553 615      346 BC EX  X          REL ADDR_DISPLACEMENT
554 616          1 GSBLNG:GETPC
554 617          0
555 620          1 GSBLNG:GTBYTA
555 621          0
556 622      1574 RCR    12
557 623      306 C=B    X
558 624      1074 RCR    2
559 625      372 BC EX  H
560 626      376 BC EX  S
561 627          1 GSBLNG:PTBYTA
561 630          0
562 631          1 GSBLNG:INCADA
562 632          0

```

567	633	316	C=B	-
564	634	1574	RCR 12	-
565	635	1	GSBLNG IPTBYTA	-
565	636	0		
566	637	1	GSBLNG JHXBYTA	SET BIT 8 OF LABEL BYTE
566	640	0		
567	641	1730	CST EX	-
568	642	1204	S7= 0	-
569	643	414	?S8=1	-
570	644	23	GONC **2 (646)	-
571	645	1210	S7= 1	-
572	646	1730	CST EX	-
573	647	1	GSBLNG PTBYTA	-
573	650	0		
574	651	630	C=M	AC[370]_LBL ADDRESS
575	652	412	A=C WPT	-
576	653	1	GOLONG XGNH40	-
576	654	2		
577				
* GT3DBT MOVED TO CNO				
579				
580				
581	655 ROW11	1	GSBLNG INCGT2	GET 2ND BYTE
581	656	0		
582	657	1526	? A#0 XS	SHORT GTO?
583	660	1640	RTN NC	NOPE
584	661	212	B=A WPT	-
585	662	26	A=0 XS	-
586	663	314	?S10=1	ROM?
587	664	303	GONC SGT025 (714)	NOPE
588	665	1506	? A#0 X	NEED SEARCH?
589	666	213	GONC SGT015 (707)	YES
590	667	2	A=0 PT	-
591	670	246	AC EX X	STATUS_BYTE2
592	671	1530	ST=C	-
593	672	1214	?S7=1	SUBTRACT?
594	673	73	GONC SGT010 (702)	YES
595	674	1204	S7= 0	PGMCTR_PGMCTR+ REL ADDR
596	675	1630	C=ST	-
597	676	406	A=C X	-
598	677	1470	C=REGN 12	-
599	700	512	A=A+C WPT	-
600			LEGAL	
601	701	113	GOTO SGT020 (712)	-
602				
603	702 SGT010	406	A=C X	PGMCTR_PGMCTR-REL ADDR
604	703	1470	C=REGN 12	-
605	704	252	AC EX WPT	-
606	705	712	A=A-C WPT	-
607			LEGAL	
608	706	43	GOTO SGT020 (712)	-
609				
610	707 SGT015	1	GSBLNG IGTSRCH	SEARCH FOR NUMERIC SHORT LBL
610	710	0		
611	711 SGT019	412	A=C WPT	PGMCTR_LBL ADDRESS
612	712 SGT020	1	GOLONG XGNH10	-
612	713	2		
613				
614				
615	714 SGT025	1506	? A#0 X	NEED COMPILER?

616	715	273	GONC	SGT040 (744)	YES
617	716	1752	A SL	WPT	UNPACK REL ADDR(EXCEPT FOR +- BIT)
618	717	1752	A SL	WPT	-
619	720	1606	A SR	X	-
620	721	1606	A SR	X	-
621	722	1470	C=REGN	12	AC[3:0]_PGMCTR
622	723	742	C=C+C	PT	-
623	724	252	AC EX	WPT	CF[3:0]_REL ADDR
624	725	742	C=C+C	PT	ADD?
625	726	47	GOC	SGT030 (732)	YES
626	727	1	GSBLNG	CALDSP	CALCULATE DISPLACEMENT
626	730	0			
627	731	1613	GOTO	SGT020 (712)	-
628					
629	732	SGT030	506	A=A+C	X
630	733		1042	C=C+1	PT
631	734		1042	C=C+1	PT
632	735		502	A=A+C	PT
633	736		33	GONC	SGT035 (741)
634	737		546	A=A+1	X
635				LEGAL	
636	740		1523	GOTO	SGT020 (712)
637					
638	741	SGT035	642	A=A-1	PT
639	742		642	A=A-1	PT
640				LEGAL	
641	743		1473	GOTO	SGT020 (712)
642					
643	744	SGT040	404	SR=	0
644	745		1	GSBLNG	IGTSCRCH
644	746		0		
645	747		530	N=C	
646	750		1	GSBLNG	IGETPC
646	751		0		
647	752		160	N=C	
648	753		630	C=M	
649	754		1546	? A#0	X
650	755		47	GOC	SGT045 (761)
651	756		1402	? A#0	PT
652	757		63	GONC	SGT055 (765)
653	760		33	GOTO	SGT050 (763)
654	761	SGT045	1406	? A#0	X
655	762		33	GONC	**+3 (765)
656	763	SGT050	410	SR=	1
657	764		252	AC EX	WPT
658	765	SGT055	1	GSBLNG	CALDSP
658	766		0		
659	767		1526	? A#0	XS
660	770		227	GOC	SGT060 (1012)
661	771		1746	A SL	X
662	772		1526	? A#0	XS
663	773		177	GOC	SGT060 (1012)
664	774		1746	A SL	X
665	775		1612	A SR	WPT
666	776		260	C=N	
667	777		252	AC EX	WPT
668	1000		414	?SR=1	
669	1001		23	GONC	**+2 (1003)
670	1002		1042	C=C+1	PT
671	1003		752	C=C+C	WPT

672	1004	752	C=C+C	WPT	-
673	1005	752	C=C+C	WPT	-
674	1006	746	C=C+C	X	-
675	1007	1074	RCR	2	-
676	1010	1	GSBLNG	PTBYTA	-
676	1011	0			
677	1012	SGT060	630	C=M	PGMCTR_LBL ADDR
678	1013	1	GOLONG	SGT019	-
678	1014	2			
679					
* CALDSP MOVED TO CH0					
681					
682					
683	1015	SAR0M	1034	PT=	2
684	1016		620	LC	6
685	1017		74	RCR	3
686	1020		436	A=C	S
687	1021		630	C=M	-
688	1022	SAR002	356	BC EX	CONVERT ASCII CHAR TO LCD
689	1023		152	AB EX	WPT
690	1024		1	GSBLNG	MASK
690	1025		0		
691	1026		0	NOP	
692	1027		1434	PT=	1
693	1030		1366	? C#0	XS
694	1031		33	GONC	**+3 (1034)
695	1032		420	LC	4
696	1033		1434	PT=	1
697	1034		356	BC EX	PLACE LCD CHAR IN STRING
698	1035		352	BC EX	WPT
699	1036		1074	RCR	2
700	1037		1352	? C#0	WPT
701	1040		43	GONC	SAR004 (1044)
702	1041		676	A=A-1	S
703	1042		57	GOC	SAR006 (1047)
704	1043		1573	GOTO	SAR002 (1022)
705	1044	SAR004	1074	RCR	2
706	1045		1352	? C#0	WPT
707	1046		1263	GONC	**+2 (1044)
708	1047	SAR006	530	M=C	M_LCD CHAR STRING
709	1050		204	SS=	0
710	1051		534	PT=	6
711	1052		116	C=0	
712	1053		1160	DADD=C	
713	1054		520	LC	5
714	1055	SAR011	372	BC EX	M
715	1056	SAR010	332	C=B	M
716	1057		1634	PT=	0
717	1060		1460	CXISA	
718	1061		130	G=C	
719	1062		1072	C=C+1	M
720	1063		1460	CXISA	
721	1064		1346	? C#0	X
722	1065		127	GOC	SAR020 (1077)
723	1066	SAR015	534	PT=	6
724	1067		332	C=B	M
725	1070		1042	C=C+1	PT
726	1071		1643	GONC	SAR011 (1055)
727	1072		120	LC	1
728	1073		320	LC	3

PT_1 & RE131_0

NOT to Manufacturer Supported
display driver NOT to contact manufacturer

CONVERT ASCII CHAR TO LCD

SPECIAL CHARACTER?

ADJUST SPECIAL CHARACTER

PLACE LCD CHAR IN STRING

DONE?

YES

? CHARS?

YES

NXT CHAR

RIGHT JUSTIFY

M_LCD CHAR STRING

MAINFRAME TBL 3RD

BIMJ_CIMJ_56K

(SEL CHIP 0)

TABLE THERE?

(G_ROM ID)

!!!!!!SHOULD BE CXISA!!!!!!

!!!!!!SHOULD BE CXISA!!!!!!

YES

ADJUST ADDR

LOAD MAIN ADDR - 1 (11777 OCT)

729	1074	1720	LC	15	
730	1075	1720	LC	15	
731	1076	210	SS=	1	SEARCH MAINFRAME TBL NOW
732	1077	SAR020	1434	PT=	1
733	1100	SAR021	1072	C=C+1	M
734	1101	SAR022	1460	CXISA	-
735	1102		214	?SS=1	MAINFRAME SEARCH?
736	1103		627	GOC	SAR042 (1165) YES
737	1104		346	BC EX	X
738	1105		1072	C=C+1	M
739	1106		1460	CXISA	-
740	1107		1306	? B#0	X
741	1110		37	GOC	SAR025 (1113) -
742	1111		1346	? C#0	X
743	1112		1543	GONC	SAR015 (1066) - (YES)
744	1113	SAR025	346	BC EX	X
745	1114		416	A=C	-
746	1115		274	RCR	5
747	1116		502	A=A+C	PT
748	1117		1174	RCR	9
749	1120		246	AC EX	X
750	1121		1574	RCR	12
751	1122		312	C=B	WPT
752	1123		160	N=C	SAVE LBL ADDR IN N
753	1124		674	RCR	11
754	1125		246	AC EX	X
755	1126		766	C=C+C	XS
756	1127		766	C=C+C	XS
757	1130		766	C=C+C	XS
758	1131		643	GONC	SAR045 (1215) -
759	1132		1072	C=C+1	M
760	1133		1072	C=C+1	M
761	1134		416	A=C	-
762	1135		1460	CXISA	-
763	1136		1474	RCR	1
764	1137		436	A=C	S
765	1140		1170	C=REGN	9
766	1141		256	AC EX	-
767	1142		1176	C=C-1	S
768	1143		1072	C=C+1	M
769	1144	SAR030	1072	C=C+1	M
770	1145		1460	CXISA	CC[1:0]_1 LBL CHAR
771	1146		1552	? A#C	WPT
772	1147		147	GOC	SAR040 (1163) NO
773	1150		1176	C=C-1	S
774	1151		1616	A SR	DEC LBL COUNT
775	1152		1616	A SR	SHIFT A TO NXT CHAR
776	1153		1376	? C#0	S
777	1154		57	GOC	SAR035 (1161) NOPE
778	1155		1512	? A#0	WPT
779	1156		57	GOC	SAR040 (1163) NOPE
780	1157		1104	SS=	0
781	1160		743	GOTO	SAR055 (1254) -
782	1161	SAR035	1512	? A#0	WPT
783	1162		1627	GOC	SAR030 (1144) NOPE, TST NXT CHAR
784	1163	SAR040	274	RCR	5
785	1164		1143	GOTO	SAR021 (1100) -
786					
787	1165	SAR042	34	PT=	3
788	1166		406	A=C	X
					SET PTR
					AC[2:0]_LBL ADDR

789	1167	1506 ? A#0	X	END OF MAINFRAME TBL?
790	1170	37 GOC	**3 (1173)	NOPE
791	1171	116 C=0		RETURN W/ERROR
792	1172	1740 RTN		-
793	1173	460 LDI		HOLE?
794	1174	1737 CON	@1737	-
795	1175	1406 ? A<C	X	-
796	1176	107 GOC	SAR043 (1206)	NOPE
797	1177	706 A=A-C	X	SUBTRACT OFFSET
798	1200	74 RCR	3	TBL ADDR_ TBL ADDR+DISPLACEMENT
799	1201	2 A=0	PT	-
800	1202	1012 C=A+C	WPT	-
801	1203	674 RCR	11	-
802	1204	1 GOLONG	SAR022	CHECK NEXT TBL ENTRY
802	1205	2		-
803	1206	SAR043 246 AC EX	X	-
804	1207	1374 RCR	13	N_LBL ADDR
805	1210	1712 C SR	WPT	-
806	1211	1042 C=C+1	PT	-
807	1212	1434 PT=	1	-
808	1213	160 N=C		-
809	1214	674 RCR	11	TEST FOR ALBL MATCH
810	1215	SAR045 416 A=C		A_ALPHA STRING
811	1216	630 C=M		-
812	1217	256 AC EX		-
813	1220	SAR047 1172 C=C-1	M	GET NXT CHAR
814	1221	1460 CXISA		-
815	1222	1352 ? C#0	WPT	IS THERE A PROMPT STRING?
816	1223	203 GONC	SAR048 (1243)	NO
817	1224	404 S8=	0	S8_END BIT
818	1225	1730 CST EX		-
819	1226	1214 ?S7=1		-
820	1227	33 GONC	**3 (1232)	-
821	1230	1204 S7=	0	-
822	1231	410 S8=	1	-
823	1232	1730 CST EX		-
824	1233	1552 ? A#C	WPT	EQUAL?
825	1234	77 GOC	SAR048 (1243)	NOPE
826	1235	1616 A SR		-
827	1236	1616 A SR		-
828	1237	414 ?S8=1		END OF LBL?
829	1240	117 GOC	SAR050 (1251)	YES
830	1241	1512 ? A#0	WPT	END OF CHARS?
831	1242	1567 GOC	SAR047 (1220)	NOPE
832	1243	SAR048 174 RCR	4	GET NXT ENTRY
833	1244	214 ?S5=1		REALIGN OLD ADDR
834	1245	27 GOC	**2 (1247)	-
835	1246	1474 RCR	1	-
836	1247	1 GOLONG	SAR021	-
836	1250	2		-
837	1251	SAR050 1512 ? A#0	WPT	END OF CHARS?
838	1252	1717 GOC	SAR048 (1243)	NOPE
839	1253	1110 S9=	1	UCODE_TRUE
840	1254	SAR055 260 C=N		CI3:0]_ADDR & F.C.
* NEXT TWO INSTRUCTIONS (PT=7, LC 0) MAY NOT BE NECESSARY.				
842	1255	1234 PT=	7	-
843	1256	20 LC	0	-
844	1257	214 ?S5=1		XROM?
845	1260	237 GOC	SAR060 (1303)	NOPE
846	1261	416 A=C		CI7:4]_XROM F.C.


```

847 1262      1074 RCR      2      CONSTRUCT TABLE INDEX PART
848 1263      1172 C=C-1    M      -
849 1264      1172 C=C-1    M      -
850 1265      772 C=C+C     M      -
851 1266      772 C=C+C     M      -
852 1267      772 C=C+C     M      -
853 1270      1732 C SR     M      -
854 1271      772 C=C+C     M      CONSTRUCT ROM ID PART
855 1272      772 C=C+C     M      -
856 1273      234 PT=       5      -
857 1274      230 C=G       -      -
858 1275      772 C=C+C     M      -
859 1276      772 C=C+C     M      -
860 1277      1234 PT=       7      CONSTRUCT XROM FC PART
861 1300      1220 LC       10     -
862 1301      34 PT=        3      -
863 1302      252 AC EX     WPT    C[3:0]_ROM ADDR & C[5:4]_F.C.
864 1303 SAR060 1010 S2=     1      -
865 1304      1740 RTH      -      RETURN
866
867
868
* ASRCH - ALPHA SEARCH
*- LOCATE THE ADDRESS OF AN ALPHA STRING. THE ALPHA
*- STRING MAY APPLY TO AN ALPHA LABEL IN RAM OR A
*- FUNCTION IN THE MAIN FRAME OR PLUG-IN ROMS. IF THE
*- FUNCTION IS LOCATED IN A PLUG-IN ROM, RETURN THE
*- XROM FUNCTION CODE. IF THE FUNCTION IS LOCATED IN
*- THE MAINFRAME, RETURN ITS FUNCTION CODE. IF THE
*- FUNCTION IS LOCATED IN RAM, RETURN THE ALPHA LABEL
*- ADDRESS.
*-
*- IN:  M[13:0] AND REG 9[13:0] = ALPHA LABEL (2 COPIES)
*-
*- OUT: C[3:0]= ADDRESS (IF USER LANG, THIS IS ADDRESS OF FIRST
*       BYTE OF LABEL)
*-      C[7:4]= FUNCTION CODE
*-      S2=1/0 IMPLIES ROM/RAM ADDRESS
*-      C=0 IMPLIES NOT FOUND
*-      S9=1/0 IMPLIES MICROCODE/USER CODE
*-      S5=1 IMPLIES A MAINFRAME FUNCTION
*-      CHIP 0 ENABLED
*-
*- USES: M,A,B,C,G,N,STATUS,PTR P,REG 9
*-      STATUS BITS 2,3,5,8,9
*- USES: 2 SUBROUTINE LEVELS
893
894
895 1305 ASRCH  1570 C=REGH  13      A[3:0]_END ADDR (RAM 1ST)
896 1306      34 PT=        3      -
897 1307      420 LC        4      C[2:0]_END LINK
898 1310      34 PT=        3      -
899 1311      412 A=C       WPT    -
900 1312      1160 DADD=C    -      -
901 1313      70 C=DATA     -      -
902 1314      1074 RCR      2      -
903 1315 SARA10 1346 ? C#0    X      END?
904 1316      1 GOLINC SAROM    YES
904 1317      2
905 1320      1 GSBLNG UPLINK    GET NXT LINK ADDR

```

```

905 1321          0
906 1322          1076 C=C+1 S      ALBL?
907 1323          1723 GONC SARA10 (1315) NOPE
908 1324 SARA20 1174 RCR 9      G_# ALPHA LBL CHARS
909 1325          1142 C=C-1 PT -
910 1326          130 G=C -
911 1327          212 B=A WPT    AC[7:0]_LBL ADDR & CHAR ADDR
912 1330          312 C=B WPT -
913 1331          374 RCR 10 -
914 1332          312 C=B WPT -
915 1333          416 A=C      GET 1ST CHAR
916 1334          1 GSBLNG INCAD2 -
916 1335          0 -
917 1336          1 GSBLNG INCAD -
917 1337          0 -
918 1340          630 C=M      B[13:0]_ALPHA STRING
919 1341          356 BC EX -
920 1342          156 AB EX -
921 1343 SARA30 156 AB EX      GET NXT BYT
922 1344          34 PT= 3 -
923 1345          1 GSBLNG INXBYTA -
923 1346          0 -
924 1347          156 AB EX -
925 1350          1434 PT= 1 -
926 1351          1552 ? A#C WPT EQUAL?
927 1352          127 GOC SARA40 (1364) NOPE
928 1353          1616 A SR      SHIFT TO NXTCHAR
929 1354          1616 A SR -
930 1355          230 C=G      DEC COUNT LBL CHARS
931 1356          1142 C=C-1 PT -
932 1357          130 G=C -
933 1360          1342 ? C#0 PT END LBL CHARS?
934 1361          113 GONC SARA50 (1372) YES
935 1362          1512 ? A#0 WPT END STR CHARS?
936 1363          1607 GOC SARA30 (1343) NOPE
937 1364 SARA40 34 PT= 3      GET NXT LINK
938 1365          316 C=B -
939 1366          174 RCR 4 -
940 1367          412 A=C WPT -
941 1370          274 RCR 5 -
942 1371          1243 GOTO SARA10 (1315) -
943 1372 SARA50 1512 ? A#0 WPT END STR CHARS?
944 1373          1717 GOC SARA40 (1364) NOPE
945 1374          106 C=0 X      ENABLE CHIP 0
946 1375          1160 DADD=C -
947 1376          316 C=B      C[13:0]_ADDR
948 1377          174 RCR 4 -
949 1400          1004 S2= 0      RAM
950 1401          1104 S9= 0      USERCODE_TRUE
951 1402          1740 RTN      RETURN

```

```

*
* RTN - RETURN
*-- POPS THE SUBROUTINE RETURN STACK IF RUNNING,
*-- OTHERWISE, IT PLACES THE PROGRAM COUNTER AT THE
*-- BEGINNING OF THE CURRENT PROGRAM
*-- IN: CHIP 0 SELECTED
*-- OUT:
*-- USES: STATUS BITS 13 & 12, C[13:0], A[13:0],
*--       B[13:0], M[13:0]
*-- USES: 2 SUBROUTINE LEVELS

```

```

962
963
964 1403 XRTN 1104 S9= 0 REMEMBER THIS IS RTN
965 1404 34 PT= 3 -
966 1405 RTN00 1314 ?S13=1 RUNNING?
967 1406 37 GOC RTN10 (1411) YES
968 1407 114 ?S4=1 SSTFLAG?
969 1410 473 GONC RTN30 (1457) NOPE
970 1411 RTN10 1370 C=REGN 11 POP RTN STK
971 1412 416 A=C -
972 1413 1470 C=REGN 12 -
973 1414 252 AC EX WPT -
974 1415 174 RCR 4 -
975 1416 1352 ? C#0 WPT - (POP ZERO?)
976 1417 353 GONC RTE21 (1454) - (YES)
977 1420 310 S10= 1 ASSUME NEW PC IS IN ROM
978 1421 1342 ? C#0 PT - (NEED UNPACK?)
979 1422 107 GOC RTN15 (1432) - (NOPE)
980 1423 304 S10= 0 NEW PC IS IN RAM
981 1424 752 C=C+C WPT - (UNPACK)
982 1425 752 C=C+C WPT -
983 1426 752 C=C+C WPT -
984 1427 746 C=C+C X -
985 1430 157 GOC RTN25 (1445) -
986 1431 1706 C SR X -
987 1432 RTN15 1450 REGN=C 12 -
988 1433 256 AC EX -
989 1434 112 C=0 WPT -
990 1435 174 RCR 4 -
991 1436 1350 REGN=C 11 -
992 1437 1770 C=REGN 15 LINE_FFF
993 1440 106 C=0 X -
994 1441 1146 C=C-1 X -
995 1442 1750 REGN=C 15 -
996 1443 1 GOLONG-CHKRPC
996 1444 2
997
998 1445 RTN25 1706 C SR X -
999 1446 1066 C=C+1 XS -
1000 LEGAL
1001 1447 1633 GOTO RTN15 (1432) -

```

```

*
* XEND - EXECUTE END
* WHEN EXECUTING FROM THE KEYBOARD, OR WHEN RUNNING AND
*- IF THE SUBROUTINE STACK IS EMPTY, THEN THE
*- PROGRAM COUNTER IS PLACED AT THE CURRENT
*- PROGRAM HEAD, OTHERWISE, A RETURN FUNCTION
*- IS PERFORMED
*- IN:
*- OUT:
*- USES: C[13:0], A[3:0], B[4:0]
*- USES: 2 SUBROUTINE LEVELS

```

```

1013
1014
1015 ENTRY XEND
1016 XEND

```

```

1017 ROW LOGIC MAY LEAVE
1018 SOME CHIP OTHER THAN
1019 0 ENABLED.
1020 1450 116 C=0 SELECT CHIP 0

```

```

1021 1451      1160 DADD=C      -
1022 1452      1110 S9=      1      REMEMBER THIS IS END
1023 1453      1323 GOTO      RTN00 (1405)
1024
1025 1454 RTN21  1304 S13=      0      CLEAR RUNNING FLAG
1026 1455      1114 ?S9=1      IS THIS END?
1027 1456      1640 RTN NC      NO, MUST BE RTN
1028 1457 RTN30  1770 C=REGN 15      LINE #_0
1029 1460      106 C=0      X      -
1030 1461      1750 REGN=C 15      -
1031 1462      314 ?S10=1      ROM?
1032 1463      127 GOC      RTN35 (1475) YES
1033 1464      1 GSBLNG FLINHP      GET END ADDRESS
1034 1465      0      -
1035 1466      474 ROR      8      -
1036 1467      412 A=C      WPT      -
1037 1470      1 GSBLNG:CPGMHD      C13:01_HEAD ADDRESS
1038 1471      0      -
1039 1472 RTN33  1104 S9=      0      TELL DORT10 TO RTN
1040 1473      1 GOLONG:DCRT10      GO CLEAR SUBROUTINE STACK
1041 1474      2      -
1042
1043
1044
1045
1046
1047 RTN35      1 GSBLNG:ROMHD      A13:01_CPGMHD
1048      0      -
1049 1477      1733 GOTO      RTN33 (1472) -

```

```

*- ROW12 - ROW TWELVE LOGIC
*- DISTINGUISHES LONG NUMERIC LABELS, <X> FUNCTION,
*- END FUNCTION, AND ALPHA LABELS
*-
*- IN:  C13:2]= FUNCTION CODE
*-      CHIP 0 SELECTED
*- USES: M13:0], C13:0], AND A13:0]
*- USES: 1 SUBROUTINE LEVEL
1055
* NOTE PARSE GENERATES CD FOR THE FC OF "ALBL". LOGIC AT ROW12A
* IS FOR KEYBOARD EXECUTION ONLY.

```

```

1058
1059 1500 ROW12A 1066 C=C+1  XS      ALBL F.C.?
1060 1501      257 GOC      ALBL (1526) YES, ALPHA LABEL
1061 1502 XENDA  1463 GOTO      XEND (1450) NO, MUST BE END
1062
1063 1503 ROW12  530 M=C      SAVE F.C.
1064 1504      1066 C=C+1  XS      LONG LBL?
1065 1505      307 GOC      LBL (1535) YES
1066 1506      1066 C=C+1  XS      -
1067 1507      1 GOLC      <X>ROW      -
1068 1510      3      -
1069 1511      1314 ?S13=1      RUNNING?
1070 1512      37 GOC      RW10 (1515) YES
1071 1513      114 ?S4=1      SSTFLAG?
1072 1514      1643 GONC      ROW12A (1500) NOFE
1073 1515 RW10  1 GSBLNG:GETPCA      ALBL?
1074 1516      0      -
1075 1517      1 GSBLNG:INCD      -
1076 1520      0      -

```

```

1074 1521          1 GSBLNG:NXTBYT      -
1074 1522          0
1075 1523          1074 RCR      2      -
1076 1524          1076 C=C+1  S      -
1077 1525          1553 GONC      XENDA (1502) GOTO END
1079
* ALBL - ALPHA LABEL
*- INCREMENT THE PROGRAM COUNTER PAST THE ALPHA LABEL,
* AND DROP INTO SLBL
*- IN:  M[3:2]= ALPHA LABEL FUNCTION CODE
*- OUT: CHIP 0 SELECTED
*- USES: C[13:0], A[13:0], STATUS BITS 1 & 2, B[13:0]
*-      M[13:0]
*- USES: 2 SUBROUTINE LEVELS
1087
1088 1526 ALBL      630 C=M              RECOVER F.C.
1089 1527          416 A=C              -
1090 1530          106 C=0      X      -
1091 1531          1160 DADD=C          -
1092 1532          1 GSBLNG:GTAINC      ADVANCE PGMCTR
1092 1533          0
1093 1534          33 GOTO      SLBL      (1537)
*
* LBL/SLBL - (NUMERIC) LABEL/SHORT LABEL
*- INCREMENTS THE PROGRAM COUNTER PAST A NUMERIC
*- LABEL, AND ROTATES THE GOOSE RIGHT ONE POSITION
*-
*- USES: 1 SUBROUTINE LEVEL
1100
1101
1102 1535 LBL          1 GSBLNG:INCQT2      INC PGMCTR
1102 1536          0
1103 1537 SLBL      214 ?SS=1              DISPLAY GOT SOMETHING?
1104                                (MSGFLG?)
1105 1540          1540 RTN C              YES
1106 1541          1 GOSUB      ENLCD
1106 1542          0
1107 1543          1670 RABCR              ROTATE GOOSE
1108 1544          1 GOLONG:ENCF00
1108 1545          2
1109
* ROW0 - ROW ZERO LOGIC
*- DISTINGUISHES NULLS FROM SHORT LABELS
*- SKIPS ALL NULLS
*- IN:  C[3:] = FUNCTION CODE
*-      PT= 3
*- OUT: PT= 3
*- USES: C[2]
1117
1118
1119
1120 1546 ROW0      1166 C=C-1  XS      -
1121 1547          1703 GONC      SLBL      (1537) SHORT LBL
1122 1550 NULL      1 GOLONG:RUNING      SKIP ALL NULLS
1122 1551          2
1123
1124
1125
1126          ENTRY  ASH20
1127          ENTRY  XASH

```

```

1128
1129
1130
* ASN - ASSIGN FUNCTION TO KEYCODE
*- THIS CODE PERFORMS AN ASSIGNMENT FUNCTION AND ALSO
*- CLEARS ASSIGNMENTS. ROM FUNCTIONS ARE ASSIGNED
*- BY PLACING THE FUNCTION CODE & KEYCODE IN AN
*- ASSIGNMENT TABLE. RAM FUNCTIONS ARE ASSIGNED BY
*- PLACING THE KEYCODE IN THE CORRESPONDING ALPHA
*- LABEL. THE ASSIGNMENT BIT MAP IS MAINTAINED AND
*- ASN TABLE REGISTERS ARE CREATED BY THIS CODE ALSO.
*- IN: A[1:0]= KEYCODE TO BE ASSIGNED/CLEARED
*- REG 9 = ALPHA STRING/ZERO
*- OUT:
*- USES: A,B,C,H,H,C,REG 9,REG 10,STATUS BITS 3,8,9,2,5
*- USES: 3 SUBROUTINE LEVELS
1144
1145
1146 1552 XASN      1170 C=REGN 9          REMOVE ASSIGNMENT?
1147 1553          530 M=C              -
1148 1554          1356 ? C#0            -
1149 1555          1 GOLNC   ASN20       YES
1149 1556          2                      -
1150 1557          1270 C=REGN 10        SAVE KEYCODE IN REG 10
1151 1560          252 AC EX   WPT       -
1152 1561          1250 REGN=C 10        -
1153 1562          1 GSBLNG ASRCH       C[3:0]_ALBL ADDR
1153 1563          0                      -
1154 1564          1356 ? C#0            ERROR?
1155 1565          1 GOLNC   SERR       YES
1155 1566          2                      -
1156 1567          1150 REGN=C 9          REG 9_ALBL ADDR & F.C.
1157 1570          1270 C=REGN 10        A[2:1]_K.C.
1158 1571          406 A=C   X          -
1159 1572          1 GSBLNG TBITMA      TEST BIT MAP
1159 1573          0                      -
1160 1574          1356 ? C#0            BIT SET?
1161 1575          73 GONC   XASN02 (1604) NO
1162 1576          1270 C=REGN 10        CLEAR KEYCODE ENTRY
1163 1577          416 A=C              -
1164 1600          1410 S1= 1           -
1165 1601          1 GSBLNG GCPCPC      -
1165 1602          0                      -
1166 1603          33 GOTO   **+3      (1606)
1167 1604 XASN02    1 GSBLNG SRBMAP      SET BIT
1167 1605          0                      -
1168 1606          1270 C=REGN 10        A[3:2]_K.C.  A[1:0]_0
1169 1607          406 A=C   X          -
1170 1610          1756 A SL            -
1171 1611          1756 A SL            -
1172 1612          1170 C=REGN 9        B[3:0]_F.C.
1173 1613          174 RCR   4          -
1174 1614          356 BC EX            -
1175 1615          1014 ?S2=1           PLACE IN RAM?
1176 1616          377 GOC   XASN05 (1655) NOPE
1177 1617          1170 C=REGN 9        C[3:0]_ALBL ADDRESS
1178 1620          256 AC EX            YES
1179 1621          1074 RCR   2          -
1180 1622          356 BC EX            SAVE K.C.
1181 1623          34 PT= 3             -

```

1182 1624	1 GSBLNG INCAD2	-
1182 1625	0	-
1183 1626	1 GSBLNG INCADA	-
1183 1627	0	-
1184 1630	1 GSBLNG IGTTYT	GET KEYCODE BYTE
1184 1631	0	-
1185 1632	156 AB EX	TEST FOR ASSIGN SAME KEY
1186 1633	1434 PT= 1	-
1187 1634	1552 ? A#C WPT	-
1188 1635	1640 RTN NC	KEYS EQUAL
1189 1636	156 AB EX	PLACE KEY CODE
1190 1637	34 PT= 3	-
1191 1640	530 M=C	-
1192 1641	356 BC EX	-
1193 1642	1 GSBLNG IPTBYTA	-
1193 1643	0	-
1194 1644	1434 PT= 1	TEST TO UPDATE BIT MAP
1195 1645	630 C=M	-
1196 1646	1352 ? C#0 WPT	-
1197 1647	1640 RTN NC	NOT NEEDED
1198 1650	256 AC EX	-
1199 1651	106 C=0 X	-
1200 1652	1160 DADD=C	-
1201 1653	1 GOLONG ITSTMAP	UPDATE BIT MAP
1201 1654	2	-
1202 1655 XASN05	1 GSBLNG IGCPKC	PLACE K.C. & F.C.
1202 1656	0	-
1203 1657	14 ?S3=1	DONE?
1204 1660	1540 RTN C	YES
1205 1661	116 C=0	CONSTRUCT REGISTER TO INSERT
1206 1662	312 C=B WPT	-
1207 1663	374 RCR 10	-
1208 1664	252 AC EX WPT	-
1209 1665	1074 RCR 2	-
1210 1666	1176 C=C-1 S	-
1211 1667	356 BC EX	B_REG TO INSERT
1212 1670	1 GSBLNG AVAILA	ANY ROOM?
1212 1671	0	-
1213 1672	1356 ? C#0	-
1214 1673	77 GOC ASN15 (1702)	YES
1215 1674	1160 DADD=C	ENABLE CHIP 0
1216 1675	156 AB EX	-
1217 1676	1 GOSUB TSTMAP	CLEAR BIT IN BIT MAP
1217 1677	0	-
1218 1700	1 GOLONG PACKE	-
1218 1701	2	-
1219		-
1220	ENTRY ASN15	ENTRY POINT MADE FOR CARD READER
*		
1222 1702 ASN15	1160 DADD=C	SELECT REGISTER
1223 1703	530 M=C	SAVE ADDR
1224 1704	70 C=DATA	SWITCH REG CONTENTS
1225 1705	356 BC EX	-
1226 1706	1360 DATA=C	-
1227 1707	630 C=M	INCREMENT ADDR
1228 1710	1046 C=C+1 X	-
1229 1711	1406 ? A<C X	END?
1230 1712	1703 GONC ASN15 (1702)	-
1231 1713	1740 RTN	RETURN
1232 1714 ASN20	216 B=A	SAVE F.C. & K.C.

NOMAS
 Not Manufacturer Supported
 recipient agrees NOT to contact manufacturer

1233 1715	1	GSBLNGTSTMAP	UPDATE BIT MAP
1233 1716	0		
1234 1717	156	AB EX	AC1103 K.C.
1235 1720	1410	SI= 1	-
1236 1721	1	GOLONGGCPKC	CLEAR K.C.
1236 1722	2		

270

NOMAS
 NOT Manufacturer Supported
 recipient agrees NOT to contact manufacturer

* SAVRTN - SAVE RETURN ADDRESS IN REG 10 [3:0]
 * SAVRC - SAVE RETURN CONDITIONED ON S7
 * SAVR10 - SAVE THE ADDRESS IN THE A AND C REGISTERS [3:0]
 * SAVR10 REQUIRES PT=3 ON ENTRY

1243		ENTRY	SAVRTN	
1244		ENTRY	SAVRC	
1245		ENTRY	SAVR10	
1246 1723 SAVRTN	1	GOSUB	GETPC	RTNS PC IN BOTH A AND C
1246 1724	0			
1247 1725 SAVR10	314	?S10=1		ROMFLAG?
1248 1726	57	GOC	SAVR20 (1733)	YES
1249 1727	106	C=0	X	PACK RAM ADDRESS INTO 3 DIGITS
1250 1730	1712	C SR	WPT	
1251 1731	506	A=A+C	X	
1252 1732	2	A=0	PT	
1253 1733 SAVR20	1270	C=REGH 10		
1254 1734	252	AC EX	WPT	
1255 1735	1250	REGH=C 10		
1256 1736	1740	RTN		

1258 1737 SAVRC	1214	?S7=1		XEQ?
1259 1740	1637	GOC	SAVRTN (1723)	YES
1260 1741	6	A=0	X	SAVE X000 TO
1261 1742	34	PT=	3	REMEMBER THIS IS
1262 1743	1703	GOTO	SAVR20 (1733)	GTO

1266		ENTRY	IORUN	
1267 1744 IORUN	460	LDI		
1268 1745	13	CON	11	MAIN RUNNING LOOP
1269				FALL INTO ROMCHK HERE

* ROMCHK - PLUG-IN ROM CHECK SUBROUTINE
 * LOOKS AT LOCATIONS AT THE END OF ROM CHIPS 5-F
 * IF THE LOCATION IS NON-ZERO, THEN DOES A GOTOC TO THAT LOCATION
 * LOCATIONS TO BE CHECKED ARE SPECIFIED IN C.X ON ENTRY:
 * PAUSE LOOP (-FF4) ... C.X=12 NOTE - MUST RETURN IN A MULTIPLE
 * OF 80 STATES AND ADJUST PAUSETIMER ACCORDINGLY
 * MAIN RUNNING LOOP (-FF5) ... C.X=11
 * WAKE UP FROM DEEP SLEEP WITH NO KEY DOWN (-FF6) ... C.X=10
 * OFF LOCATION (-FF7) ... C.X=9
 * I/O SERVICE (-FF8) ... C.X=8
 * WAKEUP FROM DEEP SLEEP (-FF9) ... C.X=7
 * COLD START (-FFA) ... C.X=6
 *
 * FOR ENTRY: HEX MODE, P SELECTED, SS0 UP, CHIP 0 SELECTED
 * PLUG-IN ROMS MUST PRESERVE C[10:3] AND RETURN TO RMCK10 WITH
 * HEX MODE, P SELECTED, STATUS SET 0 UP, AND CHIP 0 SELECTED.
 * PLUG-IN ROMS MAY RETURN TO RMCK15 (SAVING ONE WORD TIME) IF
 * ALL OF THE ABOVE CONDITIONS ARE SATISFIED AND IN ADDITION
 * PTR P=6.

- * ALL SUBROUTINE LEVELS ARE AVAILABLE EXCEPT IN I/O SERVICE ENTRY.
- * IF PKSEQ IS SET THEN I/O SERVICE ROUTINES MUST EITHER PRESERVE
- * THREE SUBROUTINE RETURNS ON THE SUBROUTINE STACK OR ELSE TERMINATE
- * THE PARTIAL KEY SEQUENCE.

*

```

1295                      ENTRY  ROMCHK
1296 1746 ROMCHK  406 A=C    X          SAVE ADDR IN A.X
1297 1747                      1004 S2=    0          CLEAR IOFLG
1298 1750                      1670 C=REGN 14
1299 1751                      1630 C=ST
1300 1752                      1650 REGN=C 14          STORE SS0
1301 1753                      246 AC EX  X          RESTORE ADDR
1302                      ENTRY  RMCK05          PAUSE LOOP ENTERS HERE
1303          RMCK05
1304 1754                      1474 RCR    1
1305 1755                      660 C=STK
1306 1756                      1374 RCR    13
1307 1757                      34 PT=    3
1308 1760                      420 LC     4
1309 1761                      1206 C=-C   X
1310 1762                      674 RCR    11

```

- * C NOW HAS ROMCHK'S RETURN ADDRESS IN DIGITS 10:7 AND THE TARGET
- * ADDRESS IN THE PLUG-IN ROMS IN DIGITS 6:3 (NOTE CHIP # IS 4 AT
- * PRESENT, BUT WILL BE INCREMENTED TO 5, THE LOWEST POSSIBLE
- * PLUG-IN ADDRESS).

```

1315                      ENTRY  RMCK10
1316 1763 RMCK10  534 PT=    6
1317                      ENTRY  RMCK15
1318 1764 RMCK15 1042 C=C+1  PT
1319 1765                      33 GONC   RMCK20 (1770)
1320 1766                      174 RCR    4
1321 1767                      740 GOTOC          RETURN TO CALLING PGM
1322
1323 1770 RMCK20 1460 CXISA
1324 1771                      1346 ? C#0  X
1325 1772                      1723 GONC   RMCK15 (1764)
1326 1773                      740 GOTOC
1327
1328
1329

```

*

```

1331                      UNLIST
1334                      END

```

ERRORS : 0

SYMBOL TABLE

ALBL	1526	-	1501
ASN15	1702	-	1712 1673
ASN20	1714	-	
ASRCH	1305	-	
DOSRC1	343	-	
DOSRCH	344	-	
GTSRCH	337	-	
IGRUN	1744	-	
LBL	1535	-	1505
NULL	1550	-	
RMCK05	1754	-	
RMCK10	1763	-	
RMCK15	1764	-	1772
RMCK20	1770	-	1765
ROMCHK	1746	-	
ROW0	1546	-	
ROW11	655	-	
ROW12	1503	-	
ROW12A	1500	-	1514
RTN00	1405	-	1453
RTN10	1411	-	1406
RTN15	1432	-	1447 1422
RTN21	1454	-	1417
RTN25	1445	-	1430
RTN30	1457	-	1410
RTN33	1472	-	1477
RTN35	1475	-	1462
RU10	1515	-	1512
SARA10	1315	-	1371 1323
SARA20	1324	-	
SARA30	1343	-	1363
SARA40	1364	-	1373 1352
SARA50	1372	-	1361
SAR002	1022	-	1043
SAR004	1044	-	1040
SAR006	1047	-	1042
SAR010	1056	-	
SAR011	1055	-	1071
SAR015	1066	-	1112
SAR020	1077	-	1065
SAR021	1100	-	1164
SAR022	1101	-	
SAR025	1113	-	1110
SAR030	1144	-	1162
SAR035	1161	-	1154
SAR040	1163	-	1156 1147
SAR042	1165	-	1103
SAR043	1206	-	1176
SAR045	1215	-	1131
SAR047	1220	-	1242
SAR048	1243	-	1252 1234 1223
SAR050	1251	-	1240
SAR055	1254	-	1160
SAR060	1303	-	1260
SAROM	1015	-	
SAVR10	1725	-	

SAVR20	1733	-	1743	1726					
SAVRC	1737	-							
SAVRTH	1723	-	1740						
SEARC1	64	-							
SEARCH	63	-							
SERR	350	-	325	277	270	264			
SERRXF	277	-	254	232					
SGT010	702	-	673						
SGT015	707	-	666						
SGT019	711	-							
SGT020	712	-	743	740	731	706	701		
SGT025	714	-	664						
SGT030	732	-	726						
SGT035	741	-	736						
SGT040	744	-	715						
SGT045	761	-	755						
SGT050	763	-	760						
SGT055	765	-	757						
SGT060	1012	-	773	770					
SLBL	1537	-	1547	1534					
SNR10	77	-	112						
SNR12	101	-	175	155	152	121	117	106	76
SNR50	120	-	114						
SNR55	125	-	214						
SNR60	150	-	134						
SNR65	153	-	167	163	157				
SNR70	156	-	110						
SNR72	176	-	171						
SNR73	206	-	177						
SNR80	211	-	161						
SNR010	3	-	37	24	16	13	7		
SNR012	4	-							
SNR020	20	-	11						
SNR030	52	-	27						
SNR040	57	-	17						
SNR050	61	-	42						
SNR09	2	-	55	51	31				
SNROM	0	-	71						
XASN	1552	-							
XASN02	1604	-	1575						
XASN05	1655	-	1616						
XEND	1450	-	1502						
XENDA	1502	-	1525						
XEQ20	362	-	453	355					
XEQ49	400	-	420						
XEQ50	401	-	361						
XEQC01	352	-	336						
XGA00	215	-							
XGI	307	-							
XGI05	330	-							
XGI07	332	-	306						
XGI30	227	-	316						
XGI40	237	-	245						
XGI50	246	-	240						
XGI52	250	-	225	223	221				
XGI54	257	-	226						
XGI55	261	-	256						
XGI57	301	-	262						
XGI60	302	-	260						
XGNN	406	-	474						

XGNN02	426	-	411	407
XGNN05	450	-	537	456
XGNN06	451	-		
XGNN10	422	-	451	334
XGNN12	424	-	377	
XGNN15	454	-	444	
XGNN20	475	-	427	
XGNN25	505	-	476	
XGNN30	520	-	542	
XGNN35	533	-	553	545
XGNN40	535	-		
XGNN50	540	-	516	
XGNN55	543	-	531	
XGNN60	546	-	526	
XGNN65	554	-	506	
XGNN70	574	-	570	
XGNN75	576	-	573	
XGNN80	600	-	572	
XGTO	405	-		
XRTN	1403	-		
XXEQ	457	-		

ENTRY TABLE

ASN15	1702	-
ASN20	1714	-
ASRCH	1305	-
DOSRC1	343	-
DOSRCH	344	-
GTSRCH	337	-
IORUN	1744	-
RMCK05	1754	-
RMCK10	1763	-
RMCK15	1764	-
ROMCHK	1746	-
ROW0	1546	-
ROW11	655	-
ROW12	1503	-
RTN30	1457	-
SAR021	1100	-
SAR022	1101	-
SAROM	1015	-
SAVR10	1725	-
SAVRC	1737	-
SAVRTH	1723	-
SEARC1	64	-
SEARCH	63	-
SERR	350	-
SGT019	711	-
SNR10	77	-
SNR12	101	-
SHROM	0	-
XASN	1552	-
XEND	1450	-
XEQC01	352	-
XGA00	215	-
XGI	307	-
XGI07	332	-
XGI57	301	-
XGNN10	422	-
XGNN12	424	-
XGNN40	535	-
XGTO	405	-
XRTN	1403	-
XXEQ	457	-

EXTERNAL REFERENCES

ADRFCH	312				
ADRFCH	313				
ASH20	1555				
ASN20	1556				
ASRCH	251	1562			
ASRCH	252	1563			
AVAILA	1670				
AVAILA	1671				
BCDBIN	317				
BCDBIN	320				
CALDSP	551	727	765		
CALDSP	552	730	766		
CHKRPC	1443				
CHKRPC	1444				
CLRSB2	401				
CLRSB2	402				
CLRSB3	375				
CLRSB3	376				
CPGMHD	172	1470			
CPGMHD	173	1471			
DCRT10	1473				
DCRT10	1474				
DECAD	304				
DECAD	305				
DECADA	137	142			
DECADA	140	143			
DOSRC1	326	415			
DOSRC1	327	416			
DOSRCH	563				
DOSRCH	564				
ENCP00	1544				
ENCP00	1545				
ENLCD	1541				
ENLCD	1542				
ERRNE	350				
ERRNE	351				
FLINKP	1464				
FLINKP	1465				
GCPKC	1601	1655	1721		
GCPKC	1602	1656	1722		
GETPC	66	200	461	616	750 1723
GETPC	67	201	462	617	751 1724
GETPCA	500	1515			
GETPCA	501	1516			
GT3DBT	521	555			
GT3DBT	522	556			
GTAINC	1532				
GTAINC	1533				
GTBYT	1630				
GTBYT	1631				
GTBYTA	620				
GTBYTA	621				
GTLINK	164				
GTLINK	165				
GTSRCH	707	745			
GTSRCH	710	746			

INCA0	463	465	1517		
INCA0	464	466	1520		
INCA02	77	202	1334	1624	
INCA02	100	203	1335	1625	
INCAD4	153	631	1336	1626	
INCAD4	154	632	1337	1627	
INCGT2	655	1535			
INCGT2	656	1536			
MASK	1024				
MASK	1025				
NFRPU	424				
NFRPU	425				
NXBYTA	101	211	502	637	1345
NXBYTA	102	212	503	640	1346
NXLTX	115				
NXLTX	116				
NXTBYT	1521				
NXTBYT	1522				
PACKE	1700				
PACKE	1701				
PTBYTA	627	635	647	1010	1642
PTBYTA	630	636	650	1011	1643
PUTPC	204				
PUTPC	205				
PUTPCX	422				
PUTPCX	423				
ROMHED	45	1475			
ROMHED	46	1476			
RUN	403				
RUN	404				
RUNING	1550				
RUNING	1551				
SAR021	1247				
SAR021	1250				
SAR022	1204				
SAR022	1205				
SAROM	1316				
SAROM	1317				
SAVR10	471				
SAVR10	472				
SAVRC	215	307			
SAVRC	216	310			
SEARC1	344				
SEARC1	345				
SERR	1565				
SERR	1566				
SGT019	1013				
SGT019	1014				
SRBMAP	1604				
SRBMAP	1605				
TBITMA	1572				
TBITMA	1573				
TSTMAP	1653	1676	1715		
TSTMAP	1654	1677	1716		
UPLINK	1320				
UPLINK	1321				
X<>ROW	1507				
X<>ROW	1510				
XGNN10	712				
XGNN10	713				

XGNN40 653
XGNN40 654

278

End of VASM assembly

VASM ROM ASSEMBLY

REV. 6/81A

OPTIONS: L C S

* HP41C MAINFRAME MICROCODE ADDRESSES 024000-25777

*

4	FILE	CN10B
5	ENTRY	AVAIL
6	ENTRY	AVAILA
7	ENTRY	BSTEP
8	ENTRY	BSTE2
9	ENTRY	BKROM2
10	ENTRY	BSTE
11	ENTRY	BSTEPA
12	ENTRY	DECAD
13	ENTRY	DECADA
14	ENTRY	FINEND
15	ENTRY	FLINK
16	ENTRY	GETPC
17	ENTRY	INCAD
18	ENTRY	INCADA
19	ENTRY	LINNM1
20	ENTRY	LINN1A
21	ENTRY	LINNUM
22	ENTRY	NXLDEL
23	ENTRY	NXLSST
24	ENTRY	FLINKA
25	ENTRY	FLINKM
26	ENTRY	FLINKP
27	ENTRY	GTBYT
28	ENTRY	GTBYTO
29	ENTRY	GETPCA
30	ENTRY	GTBYTA
31	ENTRY	GTONN
32	ENTRY	GTO.5
33	ENTRY	IN3B
34	ENTRY	INBYT
35	ENTRY	INBYTC
36	ENTRY	INBYTP
37	ENTRY	INBYT0
38	ENTRY	INBYT1
39	ENTRY	INCAD2
40	ENTRY	INCADP
41	ENTRY	INEX
42	ENTRY	INLIN
43	ENTRY	INSLIN
44	ENTRY	INLIN2
45	ENTRY	INSTR
46	ENTRY	INTXC
47	ENTRY	NR00M3
48	ENTRY	NXBYTA
49	ENTRY	NXBYT3
50	ENTRY	NXL1B
51	ENTRY	NXL3B2
52	ENTRY	NXLCHN


```

53      ENTRY  NXLIN
54      ENTRY  NXLINA
55      ENTRY  NXLIN3
56      ENTRY  NXLTx
57      ENTRY  PTBYTM
58      ENTRY  PUTPCL
59      ENTRY  SKPLIN

```

* PUT BYTE BRANCH TABLE

```

61      0 TELPBA 70 C=DATA 7 ENTRY POINTS(0,2,4,...,12)
62      1      443 GOTO PBA0 ( 45) BYTE 0
63      2      70 C=DATA
64      3      363 GOTO PBA1 ( 41) BYTE 1
65      4      70 C=DATA
66      5      303 GOTO PBA2 ( 35)
67      6      70 C=DATA
68      7      223 GOTO PBA3 ( 31)
69      10     70 C=DATA
70      11     143 GOTO PBA4 ( 25)
71      12     70 C=DATA
72      13     63 GOTO PBA5 ( 21)
73      14 PBA6 70 C=DATA BYTE 6
74      15     1574 RCR 12 ROTATE PROPER BYTE INTO POSITION
75      16     312 C=B WPT STORE 1 (OR 2) BYTE(S)
76      17     1074 RCR 2 RESTORE BYTE(S) TO PROPER POSITION
77      20     263 GOTO PBAEND ( 46) CLEAN UP
78      21 PBA5 374 RCR 10
79      22     312 C=B WPT
80      23     174 RCR 4
81      24     223 GOTO PBAEND ( 46)
82      25 PBA4 474 RCR 8
83      26     312 C=B WPT
84      27     574 RCR 6
85      30     163 GOTO PBAEND ( 46)
86      31 PBA3 574 RCR 6
87      32     312 C=B WPT
88      33     474 RCR 8
89      34     123 GOTO PBAEND ( 46)
90      35 PBA2 174 RCR 4
91      36     312 C=B WPT
92      37     374 RCR 10
93      40     63 GOTO PBAEND ( 46)
94      41 PBA1 1074 RCR 2
95      42     312 C=B WPT
96      43     1574 RCR 12
97      44     23 GOTO PBAEND ( 46)
98      45 PBA0 312 C=B WPT NO ROTATION NEEDED HERE
99      46 PBAEND 1360 DATA=C RESTORE REGISTER IN MEMORY
100     47      34 PT= 3 RESTORE POINTER
101     50     1740 RTN DONE!
102     51 INB1 1074 RCR 2
103     52     352 BC EX WPT
104     53     1574 RCR 12
105     54     443 GOTO INBENA ( 120)

```

*

```

107      ENTRY  ERRDE
108     55 ERRDE 1 GOSUB ERROR
108     56      0
109     57      0 XDEF MSGDE
110      FILLTO 0057

```

* PUT LINK BRANCH TABLE

112	60	TBLPTL	70	C=DATA		7 ENTRY POINTS(0,2,4,...,12)
113	61		403	GOTO	PTL0	(121) SPECIAL CASE ON BOUNDARY
114	62		70	C=DATA		
115	63		1623	GOTO	PBA0	(45)
116	64		70	C=DATA		
117	65		1543	GOTO	PBA1	(41)
118	66		70	C=DATA		
119	67		1463	GOTO	PBA2	(35)
120	70		70	C=DATA		
121	71		1403	GOTO	PBA3	(31)
122	72		70	C=DATA		
123	73		1323	GOTO	PBA4	(25)
124	74		70	C=DATA		
125	75		1243	GOTO	PBA5	(21)
126	76	PBA6A	1163	GOTO	PBA6	(14)
127				FILLTO	0077	

230

NOMAS
 NOT Manufacturer Supported
 repaired across NOT to contact manufacturer

* INSERT BYTE BRANCH TABLE						
129	100	TBLINB	70	C=DATA		INSERT BYTE BRANCH TABLE
130	101		533	GOTO	INB0	(154) ON 16 WORD BOUNDARY
131	102		70	C=DATA		
132	103		1463	GOTO	INB1	(51)
133	104		70	C=DATA		
134	105		433	GOTO	INB2	(150)
135	106		70	C=DATA		
136	107		353	GOTO	INB3	(144)
137	110		70	C=DATA		
138	111		273	GOTO	INB4	(140)
139	112		70	C=DATA		
140	113		213	GOTO	INB5	(134)
141	114	INB6	70	C=DATA		GET REGISTER TO INSERT INTO.
142	115		1574	RCR	12	POSITION BYTE OF INTEREST IN C00-13
143	116		352	BC EX	WPT	EXCHANGE BYTE WITH BYTE IN REGISTER
144	117		1074	RCR	2	PUT BYTE BACK INTO RIGHT POSITION IN
145	120	INBEXA	353	GOTO	INBEX	(155) FINISH UP
146	121	PTL0	1574	RCR	12	PUT FIRST BYTE IN
147	122		302	C=B	PT	
148	123		326	C=B	XS	
149	124		1074	RCR	2	
150	125		1360	DATA=C		
151	126		252	AC EX	WPT	PUT IN SECOND BYTE
152	127		412	A=C	WPT	IN THE NEXT REGISTER
153	130		1156	C=C-1		
154	131		1160	DADD=C		
155	132		1434	PT=	1	
156	133		1433	GOTO	PBA6A	(76)
157	134	INB5	374	RCR	10	
158	135		352	BC EX	WPT	
159	136		174	RCR	4	
160	137		163	GOTO	INBEX	(155)
161	140	INB4	474	RCR	8	
162	141		352	BC EX	WPT	
163	142		574	RCR	6	
164	143		123	GOTO	INBEX	(155)
165	144	INB3	574	RCR	6	
166	145		352	BC EX	WPT	
167	146		474	RCR	8	
168	147		63	GOTO	INBEX	(155)
169	150	INB2	174	RCR	4	
170	151		352	BC EX	WPT	

171	152	374	ROR	10	
172	153	23	GOTO	INDEX	(155)
173	154	INB0	352	BC EX	WPT
174	155	INDEX	1312	? B#0	WPT
175	156		107	GOC	INLIN (166)
176	157		1360	DATA=C	PUT REGISTER BACK
177	160		630	C=M	INCREMENT CT BY 1
178	161		1076	C=C+1	S
179	162		256	AC EX	RESTORE REGISTERS
180	163		106	C=0	X WAKE UP CHIP 0
181	164		1160	DADD=C	
182	165		1740	RTN	DONE
183	166	INLIN	1	GOSUB	AVAIL CHECK IF EMPTY REG. AVAILABLE
183	167		0		
184	170	NR00M	1356	? C#0	EMPTY REGISTER?
185	171		723	GONC	NOR00M (263) NO, ERROR EXIT
* INSERT ASSURED HERE.					
187	172		1	GOSUB	FLINKM FIND LINKS TO FIX UP CHAIN
187	173		0		
188	174		1	GOSUB	FIXEND FIX UP END
188	175		0		
189	176		174	ROR	4 FIX UP FOLLOWING LINK
190	177		1	GOSUB	GTLNKA GET LINK
190	200		0		
191	201		1346	? C#0	X TOP ELEMENT OF CHAIN?
192	202		23	GONC	**+2 (204) YES, DO NOTHING
193	203		1056	C=C+1	ADD 1 TO REGISTER COUNT
194				LEGAL	
195	204		1	GOSUB	PTLINK PUT LINK BACK
195	205		0		
196	206		630	C=M	
197	207		412	A=C	WPT
198	210		1474	ROR	1 PUT BYTE # IN C[XS]
199	211		1434	PT=	1 SET PT TO TOP DIGIT OF THE BYTE SPEC
200	212		43	GOTO	INL1 (216) C[XS].
201	213	INL2	1166	C=C-1	XS
202	214		1734	INC PT	
203	215		1734	INC PT	
204	216	INL1	1166	C=C-1	XS
205	217		1743	GONC	INL2 (213)
206	220		116	C=0	FIX UP CHAINHEAD
207	221		1160	DADD=C	
208	222		1570	C=REGN	13
209	223		1156	C=C-1	
210	224		1550	REGN=C	13 PUT BACK
211	225	INL3	1056	C=C+1	MOVE REGISTERS DOWN
212	226		1160	DADD=C	
213	227		346	BC EX	X
214	230		70	C=DATA	
215	231		346	BC EX	X
216	232		1146	C=C-1	X
217	233		1160	DADD=C	
218	234		346	BC EX	X
219	235		1360	DATA=C	
220	236		346	BC EX	X
221	237		1046	C=C+1	X
222	240		1546	? A#C	X FINISHED?
223	241		1647	GOC	INL3 (225) NO, MOVE ANOTHER LINE.
224	242		346	BC EX	X CLEAN UP
225	243		256	AC EX	

226	244	1360 DATA=C	STORE A TEMPORARILY
227	245	116 C=0	CREATE NULL REGISTER
228	246	252 AC EX WPT	PLACE NULLS IN BEGINNING OF REGISTER
229	247	256 AC EX	
230	250	70 C=DATA	RETRIEVE OLD A
231	251	256 AC EX	
232	252	1360 DATA=C	STORE LAST PART OF REGISTER
233	253	630 C=M	
234	254	1160 DADD=C	
235	255	70 C=DATA	
236	256	112 C=0 WPT	STORE LAST PART OF REGISTER WITH 1 0
237	257	1360 DATA=C	MORE NULLS
238	260	630 C=M	GO BACK AND FINISH THE INSERT
239	261	1 GOLONG INBYT1	
239	262	2	
240	263	NOROOM 630 C=M	NO ROOM - ERROR EXIT
241	264	412 A=C WPT	ZERO PREVIOUSLY INSERTED STEPS
242	265	103 GOTO NROOM2 (275)	
243	266	NROOM1 1 GOSUB DECADA	
243	267	0	
244	270	106 C=0 X	
245	271	1 GOSUB PTBYTM	
245	272	0	
246	273	1176 C=C-1 S	DONE?
247	274	530 M=C	
248	275	NROOM2 1376 ? C#0 S	
249	276	1707 GOC NROOM1 (266)	NO, ZERO OUT SOME MORE BYTES.
250	277	1114 ?S9=1	SET RUNNING TO ASSURE BACK STEP
251	300	23 GONC **2 (302) IF S9=1	
252	301	1310 S13= 1	
253	302	NROOM3 1 GOLONG:PACKE	
253	303	2	

*
 * AVAIL - FIND AN AVAILABLE REGISTER
 * THIS SUBROUTINE PLACES CHAINHEAD-1 IN AIXJ AND LOOKS
 *- TO SEE IF CHAINHEAD-1 IS AVAILABLE FOR USE IN INSERTING
 *- OR IN ASSIGNING. CIXJ IS RETURNED AS 0 IF THERE IS NO ROOM.
 *- IF THERE IS ROOM, CIXJ IS RETURNED AS DECIMAL 192.
 *- NOTHING IS ASSUMED AND PT IS RETURNED AS 3
 * AVAILA - SAME AS AVAIL EXCEPT PT NOT SET AND REG 0 ASSUMED
 *--SELECTED.

264	304	AVAIL	116 C=0	
265	305		34 PT= 3	
266	306		1160 DADD=C	
267	307	AVAILA	1570 C=REGN 13	GET CHAINHEAD ADDRESS
268	310		1156 C=C-1	
269	311		1160 DADD=C	SELECT CHAINHEAD-1 REGISTER
270	312		406 A=C X	SAVE ADDRESS IN A
271	313		70 C=DATA	GET THE REGISTER
272	314		1356 ? C#0	NON ZERO REGISTER?
273	315		57 GOC	AVAIL1 (322) YES, ERROR EXIT
274	316		460 LDI	
275	317		300 CON 192	REGISTER EXISTENT?
276	320		1406 ? A<C X	
277	321		1640 RTN NC	YES, SUCCESS EXIT
278	322	AVAIL1	116 C=0	FAILURE EXIT
279	323		1740 RTH	

*
 * BSTEP - BACK STEP

* WHEN CALLED, ASSUMES THE PROGRAM COUNTER IS POINTING
 *- AT AN UNKNOWN LINE. THIS ROUTINE MOVES THE
 *- PROGRAM COUNTER TO POINT AT THE BYTE JUST PRECEDING THE UNKNOWN
 *- LINE.
 * WORKS IN ROM OR RAM
 * WILL BACK STEP PAST BEGINNING OF MEMORY TO END
 *- IF LINE NUMBER = 1 OR 0.
 * ASSUMES NOTHING
 * USES A,B[0-3],C,M,N,S[0-7],3 SUB LEVELS
 *

* BACK AROUND TO END CASE OF BACK STEP

293	324	BSTEP2	314	?S10=1	ROM FLAG?
294	325		207	GOC	BSTEP3 (345) YES, DO BACK STEP IN ROM
295	326		1	GOSUB	FLINKP FIND THE END OF THE PROGRAM
295	327		0		
296	330		474	RCR	8
297	331		412	A=C	WPT
298					MOVE BACK ONE BYTE
299	332		1	GOSUB	PUTPCD PUT PC THERE
299	333		0		
300	334		1	COLONG	LINNM1 CALCULATE NEW LINE NUMBER AND RTH
300	335		2		
301	336	BSTEP	1	GOSUB	LINNUM GET THE LINENUMBER
301	337		0		
302	340		1146	C=C-1	X 0?
303	341		1637	GOC	BSTEP2 (324) YES, GO TO END.
304	342		1346	? C#0	X 1?
305	343		1613	GONC	BSTEP2 (324) YES, GO TO END.
306	344		1750	REGH=C	15 NO, FIX LINE NUMBER
307	345	BSTEP3	1	GOSUB	GETPC GET PROGRAM COUNTER
307	346		0		
308	347		314	?S10=1	ROM FLAG
309	350		457	GOC	BKROM (415) ROM-
310	351		1	GOSUB	FLINK RAM
310	352		0		
311	353	BSTEPSA	730	CM EX	SAVE ADDRESSES IN M
312	354		1512	? A#0	WPT TOP OF MEMORY?
313	355		47	GOC	BST1 (361) NO
314	356		1	GOSUB	FSTIN YES - MOVE TO BEFORE FIRST INSTRUCTIE
314	357		0		
315	360		33	GOTO	BST2 (363)
316	361	BST1	1	GOSUB	DECAIDA MOVE TO ADDRESS BEFORE LINK
316	362		0		
317	363	BST2	1	GOSUB	GTBYTA GET BYTE
317	364		0		
318	365		1574	RCR	12
319	366		1704	CLR ST	SET UP FOR NXLIN
320	367		110	S4=	1
321	370		73	GOTO	BSTNL2 (377) GO FIND THE END OF THE CURRENT LINE
* REVISED BACK STEP MAIN LOOP					
323	371	BSTNL	374	RCR	10 SAVE PREVIOUS 2 ADDRESSES
324	372		312	C=B	WPT
325	373		256	AC EX	
326	374		730	CM EX	GET REGISTER BACK
327	375		1	GOSUB	NXLIN MOVE UP ONE LINE
327	376		0		
328	377	BSTNL2	730	CM EX	
329	400		212	B=A	WPT
330	401		256	AC EX	
331	402		1406	? A<C	X MORE?

```

332 403      1667 GOC      BSTML ( 371) YES!
333 404      1546 ? A#C    X      SAME REG?
334 405      57 GOC      BSTML1 ( 412) NO, DONE
335 406      1402 ? AKC    PT      MORE?
336 407      1627 GOC      BSTML ( 371) YES
337 410      1552 ? A#C    WPT      DONE?
338 411      1603 CONC     BSTML ( 371) NO, DON'T QUIT ON EQUAL
339 412 BSTML1 474 RCR      8      DONE-GET OLD ADDRESS
340 413 BSTE    1 GOLONG BSTE2    PUT IN PC
340 414      2
* ROM BACK STEP-HERE
342 415 BKROM   674 RCR      11     PUT PC IN PLACE
343 416      410 SS=      1      SET CTNN BIT
344 417 BKROM1 1460 CXISA      GET BYTE
345 420      1172 C=C-1    M      MOVE TO PREVIOUS BYTE
346 421      1166 C=C-1    XS     STARTING BYTE?
347 422      1757 GOC      BKROM1 ( 417) NO
348 423      74 RCR      3      PUT IN PLACE
349 424      1176 C=C-1    S      BEGIN?
350 425      1 GOLHC      BKROM2    YES, GOTO LINE FFF OF PROG
350 426      2
351 427      1643 GOTO     BSTE ( 413) NO, DONE!
*
* FIXEND - FIX END
*-SETS DECOMPILE AND PACK BITS IN AN END SPECIFIED
*-BY C[8-11]
*-PT=3 IN AND OUT
*-USES 1 SUB LEVEL
* USES A[0-3],B[0-3],M
*
360 430 FIXEND  530 M=C      SAVE ADDRESSES
361 431      474 RCR      8      GET END ADDRESS
362 432      412 A=C      WPT
363 433      1 GOSUB      INCAD2    GET END BYTE
363 434      0
364 435      1 GOSUB      GTBYTA
364 436      0
365 437      1634 PT=      0      PUT DECOMPILE AND PACK BITS IN END
366 440      1720 LC      15
367 441 PTBYTN  1 GOSUB      PTBYTA
367 442      0
368 443      630 C=M      RESTORE ADDRESS
369 444      1740 RTN
*
* FLINK - FIND LINKS
* GIVEN AN ADDRESS IN MM FORM IN C[0-3]
* RETURNS THE FOLLOWING IN MM-FORM:
*
* FLINKP - SAME AS FLINK EXCEPT USES THE PC AS THE INPUT ADDRESS
*
* FLINKA - SAME AS FLINK EXCEPT INPUT ADDRESS IN A[0-3].
*
* A[0-3]- THE ADDRESS OF THE LINK PRECEEDING (HIGHER REG #)
*-THE INPUT ADDRESS.
* C[0-3]- THE INPUT ADDRESS
* C[4-7]- THE ADDRESS OF THE NEXT LINK FOLLOWING THE INPUT
*- ADDRESS.
* C[8-11]- THE ADDRESS OF THE FIRST END FOLLOWING THE INPUT
*- ADDRESS
* M[0-2] AND M[13] - THE LINK PRECEEDING THE INPUT ADDRESS

```

* NOTE- IF NO LINK PRECEDES THE INPUT ADDRESS [TOP OF MEMORY]

*- THEN A[0-3] AND M SET TO 0

* USES A[0-3], B[0-3], C[0-11], M, 1 SUB LEVEL

* PT=3 ON RETURN

391	445	FLINKP	1	GOSUB	GETPC	GET PROGRAM COUNTER
391	446		0			
392	447	FLINKA	252	AC EX	WPT	MOVE ADDRESS TO C
393	450	FLINK	730	CM EX		
394	451	FLINKM	1	GOSUB	GTFEND	GET THE FINAL END
394	452		0			
395	453		1074	RCR	2	PUT IN PLACE
396	454		33	GOTO	FLINK2 (457)	
397	455	FLINK1	1	GOSUB	UPLINK	MOVE UP 1 LINK
397	456		0			
398	457	FLINK2	730	CM EX		RETRIEVE ADDRESSES
399	460		1406	? AC	X	SEE IF DONE
400	461		77	GOC	FLINK3 (470)	NO WAY - TRY AGAIN
401	462		1546	? A#C	X	CHECK FOR SAME REGISTER CASE
402	463		1540	RTN C		DONE!
403	464		1402	? AC	PT	CHECK BYTE
404	465		37	GOC	FLINK3 (470)	TRY AGAIN
405	466		1542	? A#C	PT	ON LINK?
406	467		1540	RTN C		ALL DONE IF NOT EQUAL
407	470	FLINK3	174	RCR	4	GO UP ANOTHER LINK
408	471		212	B=A	WPT	PUT ADDRESS IN FOLLOWING LINK SPOT
409	472		312	C=B	WPT	
410	473		374	RCR	10	
411	474		730	CM EX		CHECK FOR END
412	475		1076	C=C+1	S	
413	476		147	GOC	FLINK5 (512)	CARRY IF ALPHA LABEL
414	477		730	CM EX		PUT ADDRESS IN FOLLOWING END SPOT
415	500		474	RCR	8	
416	501		312	C=B	WPT	
417	502		574	RCR	6	
418	503		730	CM EX		
419	504		1504	S12=	0	CLEAR PRIVACY STATUS BIT
420	505		1176	C=C-1	S	RESTORE END BYTE
421	506		776	C=C+C	S	CHECK PRIVATE BIT FOR THIS PROGRAM
422	507		776	C=C+C	S	IS IT 1
423	510		23	GONC	++2 (512)	NO, LEAVE PRIVACY RESET
424	511		1510	S12=	1	YES, SET PRIVACY STATUS
425	512	FLINK5	1346	? C#0	X	CHECK FOR END OF CHAIN
426	513		1427	GOC	FLINK1 (455)	NON ZERO - TRY AGAIN
427	514		12	A=0	WPT	FIX UP END OF CHAIN EXIT
428	515		116	C=0		
429	516		730	CM EX		
430	517		1740	RTN		

*

* GETPC - RETRIEVES THE PC AFTER SELECTING CHIP 0 AND CREATES

*- THE MM ADDRESS FORM BY DOUBLING THE BYTE NUMBER IF THE

*- S10=0. THE RESULTING MM ADDRESS IS

*- STORED IN A[0-3]

* GETPCA - SAME AS GETPC EXCEPT CHIP 0 ASSUMED SELECTED.

* SETS PT=3

* USES A[0-3], AND C

*

440	520	GETPC	116	C=0
441	521		1160	DADD=C
442	522	GETPCA	1470	C=REGN-12
443	523		34	PT= 3

```

444 524      314 7810=1
445 525      27 GOC      *+2      ( 527 )
446 526      742 C=C+C    PT
447 527      412 A=C      WPT
448 530      1740 RTH

```

*
 * GTONN - GOTO LINE NNN OF THE CURRENT PROGRAM
 * REPLACES THE PROGRAM COUNTER WITH THE ADDRESS OF
 *-- THE LINE SPECIFIED BY A0-23
 * FFF IN A0-23 MEANS GTO..
 * USES A,B10-31C,H,M,P,Q,S10-73
 *

```

456 531 GTONN      34 PT=      3      RSTORE POINTER TO MM PLACE
457 532      256 AC EX      PUT LINE# IN C
458 533      1346 ? C#0    X      GOTO 0?
459 534      1 GOLNC      RTH30    YES, EXECUTE KEYBOARD RETURN
459 535      2
460 536      132 C=0      M      CLEAN UP THE REGISTER
461 537      1046 C=C+1    X      GTO..?
462 540      217 GOC      GTO.. ( 561 ) YES.
463 541      1046 C=C+1    X      GTO. ALPHA?
464 542      73 GONC      GTONN2 ( 551 ) NOPE, GO ON
465 543      460 LDI      CREATE GOTO ALPHA FUNCTION CODE
466 544      35 CON2      1      13
467 545      1574 RCR      12      PUT IN PLACE
468 546      1104 S9=      0      CLEAR ALPHA SEARCH BIT
469 547      1 GOLONG      XROW1    GO TO THE LABEL
469 550      2
470 551 GTONN2     1146 C=C-1    X
471 552      1146 C=C-1    X      RESTORE LINE NUMBER
472 553      374 RCR      10      PUT IN PLACE
473 554      410 S8=      1      SET GTONN BIT
474 555      1 GOSUB      LINN1A    GO DO IT
474 556      0
475 557      1 GOLONG      NFRC     RETURN
475 560      2
476 561 GTO..      1570 C=REGN    13    GET CHAINHEAD ADDRESS
477 562      420 LC      4
478 563      34 PT=      3      SAVE FOR LATER
479 564      412 A=C      WPT
480 565      160 N=C
481 566      1 GOSUB      GTLINK    IS THE PREVIOUS LINK AN END?
481 567      0
482 570      1346 ? C#0    X      IS IT THE TOP OF MEMORY?
483 571      47 GOC      GTO.4 ( 575 ) NO, SEE IF PREVIOUS LINK IS AN END
484 572      1 GOSUB      FSTIN     GO TO THE TOP OF MEMORY
484 573      0
485 574      73 GOTO      GTO.2 ( 603 ) SEE IF .END. IS THE FIRST INST.
486 575 GTO.4      1 GOSUB      UPLINK GET THE LINK
486 576      0
487 577      1076 C=C+1    S      IS IT AN ALPHA LABEL?
488 600      157 GOC      GTO.1 ( 615 ) YES, PUT AN END IN.
489 601      1 GOSUB      INCAD2    GO TO THE NEXT INSTRUCTION
489 602      0
490 603 GTO.2      1 GOSUB      NXBYTA FIND THE ADDRESS OF THE NEXT LINE
490 604      0
491 605      1574 RCR      12      NULL?
492 606      1342 ? C#0    PT
493 607      37 GOC      GTO.2A ( 612 )
494 610      1366 ? C#0    XS

```



```

495 611          1723 GONC   GTO.2  ( 603) YES, NULL, KEEP LOOKING
496 612 GTO.2A   260 C=N      COMPARE ADDRESSES
497 613          1552 ? A#C   WPT      SAME AS THE FINAL END?
498 614          333 GONC   GTO.3  ( 647) YES, NO INSERT NEEDED.
* CREATE NEW FINAL END HERE
500 615 GTO.1     1 GOSUB   AVAIL
500 616          0
501 617          1356 ? C#0      IS THERE ROOM?
502 620          1 GOLNC   PACKE   NO, GO PACK!
502 621          2
503 622          234 PT=    5      MAKE NEW FINAL END
504 623          1420 LC     12
505 624          34 PT=    3
506 625          460 LDI
507 626          440 CON    0440   LINK= 1 REG.
508 627          1360 DATA=C
509 630          260 C=N      FIX OLD END
510 631          1160 DADD=C
511 632          1146 C=C-1  X      CONSTRUCT ADDR OF NEW
512 633          412 A=C     WPT    FINAL END AND SAVE IN
* A[3:0] FOR THE PUTPCD CALL LATER ON
* NOTE C[3] HERE IS 4, THE MM-BYTE COUNT FOR THE
* FIRST BYTE OF THE END.  N[3:0] WAS SET UP BACK AT GTO..
516 634          70 C=DATA      GET OLD FINAL END
517 635          1730 CST EX     TURN OFF FINAL END STATUS BIT
518 636          204 S5=    0
519 637          1010 S2=    1      TURN OF PACK BIT
520 640          1730 CST EX
521 641          1360 DATA=C     RETURN
522 642          116 C=0         FIX CHAINHEAD
523 643          1160 DADD=C
524 644          1570 C=REGN-13
525 645          1146 C=C-1  X
526 646          1550 REGN=C 13
527 647 GTO.3    304 S10=    0      TURN OFF THE ROM FLAG
528                                FIX PC
529 650          1 GOSUB   PUTPCD
529 651          0
530 652 GTO.5     1 GOSUB   PACKN   PACK MEMORY
530 653          0
531 654          1 GOSUB   RTH30     MAKE A ZERO LINE NUMBER
531 655          0
532 656          1 GOLONG NFRPU     CAN'T USE A RTN HERE.
532 657          2
* THE GOLONG NFRPU IS NECESSARY HERE INSTEAD OF A SIMPLE
* RETURN BECAUSE WE GET HERE FROM CLP VIA DELLIN AND DELLIN
* USES UP ALL THE SUBROUTINE LEVELS, PUSHING THE NFRPU OFF
* THE TOP OF THE STACK.
*
* GTBYT - GET BYTE
* GENERALIZED ROUTINE FOR GETTING A BYTE OUT OF ROM OR RAM.
* GETS THE BYTE POINTED TO BY A[0-3] IN MM ADDRESS FORM AND
* PLACES IT IN C[0-1]
* GTBYTA - SAME AS GTBYT EXCEPT RAM ADDRESS'S ONLY WORK.
* USES A[0-3], AND C
*
545 660 GTBYT     314 ?S10=1      ROM FLAG?
546 661          123 GONC   GTBYTA ( 673) NO, GET RAM BYTE
547 662 GTBYTO    256 AC EX      YES, GET ROM BYTE
548 663          416 A=C

```

```

549 664      674 RCR      11
550 665      1460 CXISA
551 666      1740 RTN      DONE
*
* NXBYTA - GET THE NEXT BYTE
* INCREMENTS A[0-3] IN MM FORMAT AND RETURNS THE BYTE
*-- POINTED TO BY THIS ADDRESS IN C[0-1].
* ASSUMES PT=3 ON ENTRY
* RAM ONLY!
* USES 1 SUB LEVEL
*
* NXBYT3 - GET THE NEXT 3RD BYTE
* SAME AS NXBYTA EXCEPT INCREMENTS THE ADDRESS 3 BYTES INSTEAD
*-- OF 1 BEFORE GETTING THE BYTE
*
564 667 NXBYT3      1 GOSUB  INCAD2
564 670      0
565 671 NXBYTA      1 GOSUB  INCADA
565 672      0
* GET A BYTE FROM RAM HERE
567 673 GTRYTA 252 AC EX WPT      GET BYTE OUT OF RAM
568 674      412 A=C WPT      SET UP TABLE ADDRESS
569 675      1160 DADD=C
570 676      174 RCR      4
571 677      460 LDI      TABLE ON 16 WORD BOUNDARY
* TABLE JUMP
573 700      1041 CON      01041      TABLE -GET BYTE
574 701      374 RCR      10
575 702      740 GOTOC      7 WAY BRANCH
576      ENTRY CALDSP
577 703 CALDSP 706 A=A-C X      A[3:0]_PGMCTR-REL ADDR
578 704      702 A=A-C PT      -
579 705      1640 RTN NC
580 706      273 GOTO      INC2      ( 735)
*
* INCAD - INCREMENT ADDRESS
* INCREMENTS ROM OR RAM(MM FORM) ADDRESS IN A[0-3] IN PLACE.
* INCADA - SAME AS INCAD EXCEPT ASSUMES PT=3 AND ONLY RAM ADDRESSES.
* INCADP - SAME AS INCADA EXCEPT SET PT=3 ON ENTRY
* INCAD2 - INCREMENT ADDRESS BY TWO BYTES
* SAME AS 2 CALLS TO INCADA EXCEPT FASTER.
*
* DECAD - DECADA DECREMENT ADDRESS
*--DECADA - ASSUMES ADDRESS IS A RAM ADDRESS
*--DECAD - RAM OR ROM
*--ADDRESS EXPECTED IN A[0-3] IN MM FORMAT
*--PT EXPECTED AT 3 FOR DECADA, ALWAYS RETURNED AT 3
594 707 DECAD      34 PT=      3
595 710      314 ?S10=1      ROM FLAG?
596 711      267 GOC      DECADB ( 737)
597 712 DECADA 542 A=A+1 PT
598 713      542 A=A+1 PT
599 714      542 A=A+1 PT
600      LEGAL
601 715      1 GOLONG PATCH1      0740 IN QUAD 8
601 716      2
602 717 INCAD      314 ?S10=1      ROM FLAG?
603 720      217 GOC      INCADS ( 741) YES, ROM INCREMENT.
604 721 INCADP      34 PT=      3      NO, RAM ADDRESS TO INCREMENT
605 722      43 GOTO      INCADA ( 726)

```

```

606 723 INCAD2 642 A=A-1 PT BYTE 0?
607 724 67 GOC INC21 < 732> YES, GO TO NEXT REG, BYTE 5
608 725 642 A=A-1 PT FINISH THE FIRST INCREMENT
609 726 INCADA 642 A=A-1 PT BYTE=0?
610 727 57 GOC INC1 < 734> YES, GO TO NEXT REG, BYTE 6
611 730 642 A=A-1 PT NO, FINISH MOVING TO NEXT BYTE
612 731 1740 RTN DONE
613 732 INC21 642 A=A-1 PT 2 INC CASE, BYTE 5 DESIRED
614 733 642 A=A-1 PT
615 734 INC1 642 A=A-1 PT SET C[PT] TO 12 (BYTE 6)
616 735 INC2 642 A=A-1 PT
617 736 642 A=A-1 PT
618 737 DECADB 656 A=A-1 DECREMENT REGISTER BY 1
619 740 1740 RTN DONE
620 741 INCADB 556 A=A+1 FROM INCREMENT
621 742 1740 RTN DONE
*
* INBYT0 - INSERT A ZERO BYTE INTO MEMORY
* CONDITIONS THE SAME AS INBYT EXCEPT THAT G NEED NOT BE
*- SPECIFIED.
*
* INBYTC - SPECIAL INBYT ENTRY WHERE THE BYTE TO BE INSERTED IS
*- FOUND IN C[0-11].
*
* INBYTP - SAME AS INBYT EXCEPT THAT THE PT POINTS TO THE
*- LAST DIGIT OF THE BYTE IN C TO BE INSERTED.
*
633 743 INBYT0 106 C=0 X
634 744 INBYTC 1634 PT= 0
635 745 INBYTP 130 G=C
*
* INBYT - INSERT BYTE INTO PROGRAM MEMORY
*- INCREMENT A[0-3] IN MM FORMAT AND INSERT THE BYTE IN G INTO
*- PROGRAM MEMORY AT THAT LOCATION. MAKE SPACE OF NECESSARY
*- BY INCREASING PROG LENGTH BY 1 REGISTER. FIX UP CHAINHEAD,
*- CURRENT PROGRAM HEAD AND CLOSEST LINK. ALSO INCREMENT CT
*- IN A[13]. CT IS USED TO KEEP TRACK OF THE NUMBER OF
*- SUCCESSFUL INSERTS IN A LINE IN CASE OF RUNNING OUT OF ROOM.
* DOES NOT RETURN IF NO ROOM. THE PREVIOUS [CT] BYTES SET TO 0
*- IF S9=1 THEN BACK STEP FORCED IN ERROR CASE.
* ASSUMES NOTHING.
* RETURNS CHIP 0 SELECTED AND PT=1
* USES A[0-11], B[0-3], C[0-3], M, G, PT, S9, AND 2 SUB LEVELS
*
* NOTE- INBYT0 MUST BE LOCATED RIGHT ABOVE HERE.
*
652 746 INBYT 1 GOSUB INCADP MOVE TO RIGHT BYTE
652 747 0
653 750 256 AC EX SAVE ADDRESS
654 751 530 M=C
655 752 INBYT1 1160 DADD=C WAKE UP THE RIGHT REGISTER
656 753 174 RCR 4 DO 7 WAY BRANCH JUST LINE GTBYTA
657 754 1634 PT= 0
658 755 230 C=G PUT BYTE TO INSERT INTO B[0-11]
659 756 346 BC EX X
660 757 1434 PT= 1 SET POINTER TO INSERT 2 DIGITS(1 BYTE
661 760 460 LDI
* TABLE JUMP
663 761 1204 CON @1204
664 762 374 RCR 10

```

*
 * INSLIN - INSERT LINE
 * THIS ROUTINE INSERTS A LINE AFTER THE CURRENT LINE POINTED TO
 * BY THE PC. DOES NOT SKIP CURRENT LINE IF THE LINE IS AN END OR
 * THE LINE NUMBER IS 0.
 * LEAVES THE PC POINTING TO THE NEW LINE AND INCREMENTS THE LINE
 * NUMBER BY 1. IF NO ROOM, THE ENTIRE INSERT IS IGNORED.
 * DIGIT OR TEST ENTRY NOT HANDLED BY THIS ROUTINE.
 * THE LINE TO BE INSERTED HAS ITS FIRST BYTE IN C[13-12], THE
 * SECOND BYTE, IF NEEDED, IS IN C[11-10]. ALPHA OPERANDS ARE
 * IN REGISTER 9
 * USES A,B,C,M,G,S[0-9]

679	764	INSLIN	1	GOSUB	INSSUB	INITIALIZE
679	765		0			
680	766	INLIN2	216	B=A		SAVE BYTES FOR LATER
681	767		316	C=B		GET FIRST BYTE READY FOR INSERT
682	770		1534	PT=	12	
683	771		130	G=C		
684	772		36	A=0	S	INITIALIZE CT TO 0
685	773		1	GOSUB	INBYT	INSERT 1 BYTE
685	774		0			
686	775		1236	C=-C	S	DECODE NUMBER OF BYTES TO INSERT
687	776		776	C=C+C	S	1 BYTE?
688	777		133	GONC	IN2B	(1012) NO, MORE DECODE
689	1000		1076	C=C+1	S	LINE 1?
690	1001		1076	C=C+1	S	
691	1002		273	GONC	INEXA	(1031) NO, DONE!
* ALPHA OPERANDS HERE						
693	1003		1	GOSUB	INTXC	PREPARE TEXT CHARACTER
693	1004		0			
694	1005		1	GOSUB	INBYTC	INSERT TEXT CHAR.
694	1006		0			
695	1007		1	GOSUB	INSTR	OUTPUT TEXT STRING
695	1010		0			
696	1011		203	GOTO	INEXA	(1031) DONE!
697	1012	IN2B	776	C=C+C	S	2 BYTE?
698	1013		1	GOLNC	IN3B	NO, MORE DECODE
699	1014		2			
699	1015		1376	? C#0	S	ROW 12?(NO ROW 11 CODES)
700	1016		767	GOC	IN2BA	(1114) NO, ROWS 9-10
701	1017		1534	PT=	12	CHECK FOR LBL NN
702	1020		1042	C=C+1	PT	LBL NN?
703	1021		113	GONC	IN2BB	(1032) NO, MORE DECODE
704	1022		374	RCR	10	YES, CHECK FOR SHORT FORM
705	1023		1046	C=C+1	X	NOTE-FF IS ILLEGAL ADDRESS
706	1024		1434	PT=	1	15 CARRIED TO 10?
707	1025		1342	? C#0	PT	OR >15?
708	1026		717	GOC	INN2B	(1117) YES, LONG FORM, NORMAL 2 BYTE
709				ENTRY	INSHRT	
710	1027	INSHRT	1	GOSUB	PTBYTA	SHORT FORM
710	1030		0			
711	1031	INEXA	613	GOTO	INEX	(1112) DONE!
712	1032	IN2BB	1042	C=C+1	PT	CK>REG?
713	1033		647	GOC	INN2B	(1117) YES, NORMAL 2 BYTE
* LINKS OF THE CHAIN INSERTED HERE						
715	1034		404	S8=	0	CLEAR END BIT
716	1035		1042	C=C+1	PT	END?
717	1036		27	GOC	**2	(1040) NO, ALPHA LABEL

NOMAS
 NO MANUFACTURING SUPPORT
 NO MANUFACTURING SUPPORT

718 1037	410 S8= 1	SET END BIT
719 1040	1 GOSUB INBYT0	PUT IN BYTE FOR LINK
719 1041	0	
720 1042	460 LDI	PUT OF IN EXP FIELD
721 1043	17 CON 15	
722 1044	414 ?S8=1	END?
723 1045	43 GONC INLNK1 (1051)	NO, ALPHA LABEL
724 1046	1 GOSUB INBYTC	OUTPUT OF (END BYTE)
724 1047	0	
725 1050	123 GOTO INLNK2 (1062)	GO FIX LINKS
726 1051 INLNK1	1 GOSUB INTXC	MAKE TEXT CHARACTER
726 1052	0	
727 1053	1046 C=C+1 X	COUNT BYTE FOR KEYCODE
728	LEGAL	
729 1054	1 GOSUB INBYTC	INSERT TEXT COUNT
729 1055	0	
730 1056	1 GOSUB INBYT0	INSERT ZERO BYTE FOR KEYCODE
730 1057	0	
731 1060	1 GOSUB INSTR	INSERT STRING
731 1061	0	
732 1062 INLNK2	1 GOSUB GETPC	FIX LINKS
732 1063	0	
733 1064	1 GOSUB INCADA	POINT TO FIRST BYTE OF NEW LINK
733 1065	0	
734 1066	1 GOSUB FLINKA	FIND LINKS
734 1067	0	
735 1070	1 GOSUB GENLNK	FIX CURRENT LINK
735 1071	0	
736 1072	174 RCR 4	
737 1073	1 GOSUB GENLNK	FIX PREVIOUS LINK
737 1074	0	
738 1075	414 ?S8=1	IF END, FIX PREVIOUS END
739 1076	143 GONC INEX (1112)	NO, DONE
740 1077	374 RCR 10	YES, PUT DECOMPILE BITS
741 1100	1 GOSUB FIXEND	IN PREVIOUS END
741 1101	0	
742 1102	412 A=C WPT	MOVE PC TO END OF "END"
743 1103	1 GOSUB INCAD2	
743 1104	0	
744 1105	1 GOSUB PUTPC	
744 1106	0	
745 1107	1770 C=REGN 15	SET LINE # TO 000
746 1110	106 C=0 X	
747 1111	1750 REGN=C 15	
748 1112 INEX	1 GOLONG MFRC	DONE
748 1113	2	
749 1114 IN2BA	776 C=C+C S	SEPARATE ROWS 9 AND 10
750 1115	1376 ? C#0 S	
751 1116	67 GOC IN2R9 (1124)	ROW 9, MORE TO DO
752 1117 INH2B	316 C=8	ROW 10, NORMAL 2 BYTE
753 1120	334 PT= 10	
754 1121	1 GOSUB INBYTP	INSERT SECOND BYTE
754 1122	0	
755 1123	1673 GOTO INEX (1112)	DONE
* ROW 9 HERE		
757 1124 IN2R9	1534 PT= 12	
758 1125	1142 C=C-1 PT	RCL?
759 1126	103 GONC IN2ST0 (1136)	NO, CHECK FOR STORE
760 1127	634 PT= 11	YES, CHECK FOR SHORT FORM
761 1130	1342 ? C#0 PT	<16?

762	1131	1667	GOC	INN2B	(1117)	NO, LONG FORM.
763	1132	220	LC	2		YES, MAKE SHORT FORM
764	1133	374	RCR	10		
765	1134	1	GOLONG	INSHRT		INSERT IT
765	1135	2				
766	1136	1142	C=C-1	PT		STO?
767	1137	1603	GONC	INN2B	(1117)	NO, STANDARD 2 BYTE
768	1140	634	PT=	11		SHORT FORM?
769	1141	1342	? C#0	PT		
770	1142	1557	GOC	INN2B	(1117)	NO, LONG FORM
771	1143	320	LC	3		
772	1144	1673	GOTO	INRCLS	(1133)	SHORT FORM
* 3 BYTE FUNCTIONS HERE						
774	1145	1	GOSUB	INBYT0		PUT OUT SECOND BYTE FOR COMPILE
774	1146	0				
775	1147	776	C=C+C	S		XEQ OR GTO?
776	1150	1376	? C#0	S		
777	1151	1463	GONC	INN2B	(1117)	XEQ-INSERT NORMAL ADDRESS
778	1152	374	RCR	10		CHECK FOR SHORT FORM
779	1153	1046	C=C+1	X		NOTE-FF IS ILLEGAL
780	1154	1434	PT=	1		
781	1155	1342	? C#0	PT		SHORT FORM?
782	1156	1417	GOC	INN2B	(1117)	NO, INSERT NORMAL ADDRESS
783	1157	1320	LC	11		YES, SHORT FORM
784	1160	1	GOSUB	DECAD		OVERWRITE FIRST BYTE
784	1161	0				
785	1162	1523	GOTO	INSHR2	(1134)	

*

* INSTR - INSERT STRING

* GIVEN REG A IN THE PROPER FORMAT FOR INBYT, INSERTS A

*- LABEL STRING FROM REG 9 INTO PROGRAM MEMORY

* USES THE SAME REGISTERS AS INBYT AND IN ADDITION ALL

*- OF C.

* USES 3 SUB LEVELS. RETURNS PT=1

*

794	1163	INSTR	1170	C=REGN-9		GET THE REST OF THE STRING
795	1164	INSTR1	1356	? C#0		ALL DONE?
796	1165		1640	RTN NC		YES, GO BACK.
797	1166		1	GOSUB	INBYTC	NO, INSERT ANOTHER CHAR.
797	1167		0			
798	1170		1170	C=REGN-9		SHIFT OUT INSERTED CHAR.
799	1171		1716	C SR		
800	1172		1716	C SR		
801	1173		1150	REGN=C 9		
802	1174		1703	GOTO	INSTR1 (1164)	GO AROUND AGAIN

*

* INTXC - PREPARE TEST CHARACTER FOR INSERT

* PLACES A TEXT CHARACTER OF THE PROPER SIZE IN C[0-1]

* WHICH IS NEEDED TO PRECEED THE TEXT STRING IN REG 9.

* USES ONLY C[X] AND M.

*

809	1175	INTXC	460	LDI		CREATE TEXT CHARACTER
810	1176		360	CON2	15	0
811	1177		256	AC EX		PLACE TEXT CHAR IN A
812	1200		530	M=C		SAVE A FOR LATER
813	1201		1170	C=REGN 9		GET TEXT CHAR
814	1202		43	GOTO	INTXC2 (1206)	
815	1203	INTXC1	546	A=A+1	X	ADD 1 TO TEXT CHAR
816	1204		1716	C SR		MOVE TO NEXT CHAR.
817	1205		1716	C SR		

818 1206	INTXC2	1356 ? C#0	ALL DONE?
819 1207	1747 GOC	INTXC1 (1203)	NO, COUNT SOME MORE.
820 1210	630 C=M		DONE, PUT THINGS BACK
821 1211	256 AC EX		RESTORE A
822 1212	1740 RTN		DONE

*

* LINNUM - LINE NUMBER

* WHEN CALLED EITHER RECALLS THE BINARY LINE NUMBER

*- OF THE CURRENT LINE FROM REGISTER 15 OR ELSE COMPUTES IT

*- IF THE LINE NUMBER STORED IS INVALID, IN ALL CASES THE

*- CORRECT LINE NUMBER IS RETURNED IN C[0-2]. IF COMPUTED,

*- THE PROPER LINE NUMBER IS STORED.

* ASSUMES CHIP 0 SELECTED ON INPUT.

* WORKS IN ROM OR RAM.

* USES 2 SUBROUTINE LEVELS.

* USES A,C,M,N,P,Q,B[0-3],S[0-8]. RETURNS P SELECTED IF LINE NUMBER

*- IS COMPUTED.

835 1213	LINNUM	1770 C=REGN 15	GET LINE NUMBER
836 1214		1046 C=C+1 X	VALID?
837 1215		37 GOC LINNM1 (1220)	NO, GO COMPUTE IT.
838 1216		1146 C=C-1 X	RESTORE THE CORRECT NUMBER
839 1217		1740 RTN	
840 1220	LINNM1	404 S8= 0	CLEAR GTONN BIT
841 1221	BKROM2	116 C=0	
842 1222		1156 C=C-1	SET TARGET LINE# = FFF
843 1223	LINN1A	240 SEL P	COMPUTE LINE NUMBER IN RAM
844 1224		134 PT= 4	SET UP POINTERS FOR LATER
845 1225		340 SEL Q	
846 1226		160 N=C	STORE TARGET LINE#
847 1227		314 ?S10=1	ROM FLAG?
848 1230		647 GOC LINROM (1314)	YES, COMPUTE LINE# IN ROM
849 1231		1 GOSUB FLINKP	FIND THE PREVIOUS LINK
849 1232		0	
850 1233		414 ?S8=1	GTONN?
851 1234		23 GONC **2 (1236)	NO, USE PC ADDRESS
852 1235		474 RCR 8	YES, USE END ADDRESS AS TARGET
853 1236		122 C=0 PQ	PREPARE FOR LINE NUMBER IN PQ FIELD
854 1237		730 CM EX	
855 1240		1512 ? A#0 WPT	TOP OF MEMORY?
856 1241		103 GONC LINNM5 (1251)	YES, GO TO FIRST INST.
857 1242		33 GOTO LINNM2 (1245)	GO FIND IT
858 1243	LINNM3	1 GOSUB UPLINK	FIND PREVIOUS END
858 1244		0	
859 1245	LINNM2	1076 C=C+1 S	END?
860 1246		63 GONC LINNM4 (1254)	YES, MOVE TO FINAL BYTE
861 1247		1346 ? C#0 X	TOP OF MEMORY?
862 1250		1737 GOC LINNM3 (1243)	NO, CONTINUE
863 1251	LINNM5	1 GOSUB FSTIN	YES, POSITION JUST BEFORE 1ST INST.
863 1252		0	
864 1253		63 GOTO LINNM6 (1261)	GO COUNT LINES.
865 1254	LINNM4	1 GOSUB INCADA	END! SET UP FOR COUNTING LOOP
865 1255		0	
866 1256		1 GOSUB NXBYTA	POSITION TO LAST BYTE OF END
866 1257		0	
867 1260		1574 RCR 12	
868 1261	LINNM6	1704 CLR ST	SET UP FOR NXLIN
869 1262		110 S4= 1	
870 1263		212 B=A WPT	SAVE COUNTING ADDRESS IN B
871 1264		730 CM EX	STORE MEM REG IN C
872 1265		416 A=C	GET TARGET ADDRESS AND LINE CT TO A

873	1266	260	C=N		RETRIEVE THE TARGET LINE NUMBER
874	1267	312	C=B	WPT	MERGE WITH THE COUNTING ADDRESS
* MAIN COUNTING LOOP					
876	1270	LINML	160	N=C	SAVE THE CURRENT ADDRESS
877	1271		256	AC EX	
878	1272		730	CM EX	SAVE ADDRESS, GET STEPS
879	1273	LINML1	1	GOSUB	MOVE TO THE NEXT LINE
879	1274		0		
880	1275		730	CM EX	GET ADDRESS
881	1276		1062	C=C+1	ADD 1 TO LINE COUNT
882	1277		256	AC EX	TEST FOR DONE
883	1300		1562	? A#C	REACHED LINE NN
884	1301		113	GONC	LINML2 (1312) YES, GTOHN EXIT.
885	1302		1406	? A#C	MORE?
886	1303		1657	GOC	LINML (1270) YES.
887	1304		1546	? A#C	SAME REGISTER?
888	1305		57	GOC	LINML2 (1312) NO, DONE!
889	1306		1402	? A#C	MORE?
890	1307		1617	GOC	LINML (1270) YES
891	1310		1542	? A#C	MORE? - DON'T STOP ON EQUAL.
892	1311		1573	GONC	LINML (1270) YES
893	1312	LINML2	256	AC EX	SAVE NUMBER IN C
894	1313		343	GOTO	LINEND (1347) ALL DONE!
* CALCULATE LINE NUMBER IN ROM					
896	1314	LINROM	1	GOSUB	ROMHED
896	1315		0		
897	1316		116	C=0	PREPARE A ZERO MANTISSA
898	1317		252	AC EX	WPT
899	1320		530	M=C	SAVE COUNTERS IN M
900	1321		652	A=A-1	WPT
901	1322		414	?S8=1	SET A[0-3]=FFFF
902	1323		1	GOSUB	GETPC
902	1324		0		NO, GET ENDING ADDRESS
903	1325		260	C=N	GET ENDING LINE#
904	1326		252	AC EX	WPT
905	1327		730	CM EX	FORM TARGET STRING
906	1330		416	A=C	GET READY FOR LOOP
907	1331		504	S6=	0
908	1332		53	GOTO	LINRM4 (1337)
909	1333	LINRM2	1	GOSUB	SKPLIN
909	1334		0		MOVE TO THE NEXT LINE
910	1335		514	?S6=1	HIT AN END?
911	1336		77	GOC	LINRM3 (1345) YES, DONE!
912	1337	LINRM4	562	A=A+1	NO, ADD 1 TO LINE #
913	1340		630	C=M	GET TARGETS
914	1341		1422	? A#C	PO
915	1342		33	GONC	LINRM3 (1345) YES, DONE!
916	1343		1412	? A#C	WPT
917	1344		1677	GOC	LINRM2 (1333) REACHED THE ADDRESS?
918	1345	LINRM3	256	AC EX	NO, TRY AGAIN.
919	1346		160	N=C	DONE!
920	1347	LINEND	174	RCR	4
921	1350		406	A=C	X
922	1351		1	GOSUB	GETLIN
922	1352		0		PLACE NUMBER IN REGISTER 15
923	1353		240	SEL P	
924	1354		34	PT=	3
925	1355		1514	?S12=1	PRIVATE PROGRAM?
926	1356		1540	RTN C	YES, RETURN FFF.
927	1357		246	AC EX	X
					PUT LINE NUMBER IN PLACE

928 1360	1750 REGN=C 15	PUT BACK
929 1361	260 C=N	
930 1362 BSTE2	412 A=C WPT	
931 1363 PUTPCL	1 COSUB PUTPC	PUT THE NEW ADDRESS IN THE PC
931 1364	0	
932 1365	1770 C=REGN 15	GET THE LINE NUMBER
933 1366	1740 RTH	

*
 * NXLIN - MOVE TO THE NEXT LINE
 * SPECIAL RAM PROGRAM MEMORY TRAVERSAL SUBROUTINE
 * GIVEN THE ADDRESS OF THE LAST BYTE OF A LINE IN MM FORMAT IN
 *- A[0-3], AND ALSO IN C THE REGISTER POINTED TO BY A[0-2]
 *- ROTATED SO THAT THE BYTE POINTED TO BY A[3] IS IN C[3-2].
 * THE ROUTINE RETURNS A & C IN THE SAME FORMAT AS THEY WERE
 *- INPUT BUT REFERRING TO THE NEXT LINE IN PROGRAM MEMORY.
 * NOTE- IF THE BYTE NUMBER=0 THEN C[3-2] IS CORRECT, BUT THE
 *- REST OF C MAY BE FROM A DIFFERENT REGISTER ON RETURN. ON
 *- INPUT, C NEED NOT BE SPECIFIED.
 * TRAILING NULLS ARE TREATED AS PART OF THE CURRENT PROGRAM STEP.
 * USES B[0-3].
 * PT=3 IN AND OUT
 *

*
 * SKPLIN -SKIP A LINE
 * GIVEN THE ADDRESS OF THE LAST BYTE OF A PROGRAM LINE
 *- IN A[0-3] IN MM FORMAT, RETURNS THE ADDRESS OF THE
 *- LAST BYTE OF THE NEXT LINE IN A[0-3].
 * NULLS FOLLOWING THE CURRENT LINE ARE PROPERLY SKIPPED
 * THE ROUTINE DOES NOTHING IF THE LINE TO BE SKIPPED IS
 *- AND END.
 *

* NXLSST - SAME AS SKPLIN, BUT THIS ENTRY WILL SKIP ENDS
 *- BY GOING TO STEP 1 OF THE CURRENT PROGRAM.
 *

* S6 SET TO 1 WHEN ENCOUNTERING AN END.
 * USES A[0-3],B[0-3],C,S0-7,1 SUB LEVEL
 *

* NXLDEL - A SPECIAL ENTRY POINT INTO NXLIN HAS BEEN
 *- CREATED FOR DELETE OPERATIONS. THIS ENTRY POINT
 *- EXPECTS S7=1 AND GOES ON TO THE NORMAL RAM LINE SKIPPING
 *- LOGIC. IF A CHAIN ELEMENT IS TO BE SKIPPED, SPECIAL
 *- DELETE LOGIC IS EMPLOYED:
 * THE PREVIOUS LINK IS ELARGED TO BRIDGE THE GAP
 * IF AN END IS TO BE DELETED, SET S5=0. OTHERWISE ALL ENDS
 *- ARE TREATED AS THE FINAL END.
 * IF THE FINAL END, RETURN WITH THE SAME ADDRESS AS INPUT.
 *

974 1367 NXLSST	1704 CLR ST	SINGLE STEP ENTRY
975 1370	33 GOTO	NXLSS1 (1373)
976 1371 SKPLIN	1704 CLR ST	
977 1372	210 S5=	1 SET BIT TO BACK UP ON END
978 1373 NXLSS1	314 ?S10=1	ROM TO SKIP?
979 1374	47 GOC	SKPR0M (1400) YES GO DO IT
980 1375 NXLDEL	1 COSUB	NXBYTA DELETE ENTRY
980 1376	0	
981 1377	403 GOTO	NXLINA (1437)

* ROM SKIP LINE HERE
 983 1400 SKPR0M 252 AC EX WPT
 984 1401 674 RCR 11
 985 1402 SKPR10 1072 C=C+1 M

986	1403	1460	CXISA			
987	1404	1366	? C#0	XS		1ST BYTE OF NEW FC?
988	1405	1753	GONC	SKPR10 (1402)		NO. SKIP THIS NULL.
989	1406	SKPR20	1072 C=C+1	M		SKIP THIS BYTE
990	1407	1460	CXISA			
991	1410	1166	C=C-1	XS		CONTINUATION BYTE?
992	1411	1757	GOC	SKPR20 (1406)		YES
993	1412	SKPR30	1172 C=C-1	M		MUST BE 3RD BYTE OF END
994						OR 1ST BYTE OF 2ND NEW FC
995						BACK UP ONE BYTE
996	1413	1166	C=C-1	XS		WAS IT 1ST BYTE OF 2ND FC?
997	1414	1	GOLC	ROMH35		YES
997	1415	3				
998	1416	510	S6=	1		MARK THE END
999	1417	214	?S5=1			STOP AT END?
1000	1420	1	GOLNC	ROMHED		NO. GO TO TOP
1000	1421	2				
1001	1422	1172	C=C-1	M		BACK UP 2ND BYTE
1002			LEGAL			
1003	1423	1673	GOTO	SKPR30 (1412)		GO BACK UP 1 MORE & EXIT
* NXLIN RAM TRAVERSAL LOGIC HERE						
1005	1424	NXLIN	642 A=A-1	PT		MOVE TO THE NEXT BYTE
1006	1425		113 GONC	NXLIN1 (1436)		
1007	1426		252 AC EX	WPT		GET THE NEXT REGISTER
1008	1427		1142 C=C-1	PT		SET BYTE NO. TO 6
1009	1430		1142 C=C-1	PT		
1010	1431		1146 C=C-1	X		MOVE TO THE NEXT REGISTER
1011	1432		1160 DADD=C			GET IT
1012	1433		412 A=C	WPT		SAVE THE NEW ADDRESS
1013	1434		70 C=DATA			
1014	1435		1574 RCR	12		MOVE BYTE 6 INTO BYTE 0 POSITION
1015	1436	NXLIN1	642 A=A-1	PT		FINISH CHANGING THE BYTE NO.
1016	1437	NXLIN1A	1574 RCR	12		MOVE NEW BYTE INTO POSITION
1017	1440		1202 C=-C	PT		START DECODE
1018	1441		742 C=C+C	PT		1 BYTE?
1019	1442		373 GONC	NXLIN2 (1501)		NO, MORE DECODE
1020	1443	NXLIN1B	1042 C=C+1	PT		1 BYTE INST. HERE
1021	1444		1042 C=C+1	PT		ROW 1?
1022	1445		1640 RTN NC			NO, ALL DONE!
1023	1446		766 C=C+C	XS		DIG 0-7?
1024	1447		53 GONC	NXLDE (1454)		YES
1025	1450		766 C=C+C	XS		DIG 8-9, ., EEX?
1026	1451		33 GONC	NXLDE (1454)		YES
1027	1452		1366 ? C#0	XS		GTO ALPHA, XEQ ALPHA?
1028	1453		1517 GOC	NXLIN (1424)		YES, GET TEXT
* DIGIT ENTRY HERE						
1030	1454	NXLDE	352 BC EX	WPT		SAVE THE CURRENT BYTE IN B
1031	1455		312 C=B	WPT		RESTORE C
1032	1456		1 GOSUB	NXL3B2		GET THE NEXT BYTE
1032	1457		0			
1033	1460		1202 C=-C	PT		SEARCH FOR NON DIGIT ENTRY CODE
1034	1461		742 C=C+C	PT		1 BYTE FN?
1035	1462		123 GONC	NXLDE2 (1474)		NO, BACK UP 1 BYTE
1036	1463		1042 C=C+1	PT		
1037	1464		1042 C=C+1	PT		ROW 1?
1038	1465		73 GONC	NXLDE2 (1474)		NO, BACK UP
1039	1466		766 C=C+C	XS		DIG 0-7?
1040	1467		1653 GONC	NXLDE (1454)		YES, KEEP GOING
1041	1470		766 C=C+C	XS		DIG 8-9, ., EEX?
1042	1471		1633 GONC	NXLDE (1454)		YES, KEEP GOING

1043	1472	1366 ? C#0	XS	CHS?
1044	1473	1613 GONC	NXLDE (1454)	YES, KEEP GOING
1045	1474 NXLDE2	1 GOSUB	DECADA	BACK UP ONE BYTE
1045	1475	0		
1046	1476	1074 RCR	2	RESTORE THE REGISTER
1047	1477 NXLTX1	312 C=B	WPT	RETRIEVE OLD BYTE
1048	1500	1740 RTN		DONE!
* 2 BYTE INSTRUCTIONS HERE				
1050	1501 NXLIN2	742 C=C+C	PT	2 BYTE?
1051	1502	353 GONC	NXLIN3 (1537)	NOPE, MORE DECODE
1052	1503	1342 ? C#0	PT	ROW 12?
1053	1504	377 GOC	NXL3B2 (1543)	NO, INCREMENT 1 BYTE
1054	1505	1066 C=C+1	XS	LBL NN?
1055	1506	357 GOC	NXL3B2 (1543)	YES, SIMPLE INCREMENT
1056	1507	1066 C=C+1	XS	X<>NN?
1057	1510	337 GOC	NXL3B2 (1543)	YES
1058	1511 NXLCHN	1214 ?S7=1		DELETE?
1059	1512	1 GOLC	SKPDEL	YES, SPECIAL LOGIC
1059	1513	3		
1060	1514	1 GOSUB	NXL3B2	GET THE THIRD BYTE
1060	1515	0		
1061	1516	1 GOSUB	NXL3B2	
1061	1517	0		
1062	1520	1042 C=C+1	PT	ALPHA LABEL?
1063	1521	467 GOC	NXLTX (1567)	YES, GO DO THE TEXT
1064	1522	510 S6=	1	NO, END - MARK IT
1065	1523	114 ?S4=1		NORMAL CASE?
1066	1524	1540 RTN C		YES, DONE!
1067	1525	1 GOSUB	DECADA	RESTORE ADDRESS OF 1ST BYTE OF LINK.
1067	1526	0		
1068	1527	1 GOSUB	DECADA	
1068	1530	0		
1069	1531	214 ?S5=1		SKPLIN?
1070	1532	1427 GOC	NXLDE2 (1474)	YES, BACK UP
1071	1533	1 GOSUB	GTLINK	NO, SINGLE STEP
1071	1534	0		
1072	1535	1 GOLONG	ICPGMHD	GO TO THE TOP OF THE PROGRAM
1072	1536	2		
* 3 BYTE INSTRUCTIONS HERE				
1074	1537 NXLIN3	742 C=C+C	PT	3 BYTE?
1075	1540	213 GONC	NXLIN4 (1561)	NO, MORE DECODE
1076	1541	1 GOSUB	NXL3B2	INCREMENT THE FIRST BYTE
1076	1542	0		
1077	1543 NXL3B2	642 A=A-1	PT	INCREMENT 1 BYTE
1078	1544	123 GONC	NXL2B1 (1556)	
1079	1545	252 AC EX	WPT	
1080	1546	1420 LC	12	
1081	1547	34 PT=	3	
1082	1550	1146 C=C-1	X	
1083	1551	1160 DADD=C		
1084	1552	412 A=C	WPT	
1085	1553	70 C=DATA		
1086	1554	374 RCR	10	
1087	1555	1740 RTN		
1088	1556 NXL2B1	642 A=A-1	PT	
1089	1557	1574 RCR	12	
1090	1560	1740 RTN		
* TEXT AND ROW 0 HERE				
1092	1561 NXLIN4	742 C=C+C	PT	TEXT?
1093	1562	57 GOC	NXLTX (1567)	YES, GO TRAVERSE IT.

```

* ROW 0 HERE
1095 1563 NXLR0 1366 ? C#0 XS SHORT LABELS?
1096 1564 1540 RTH C YES, ALL DONE!
1097 1565 1 GOLONG NXLIN SKIP OVER NULLS
1097 1566 2

* TEXT HERE
1099 1567 NXLTIX 1166 C=C-1 XS
1100 1570 1540 RTH C EXIT FOR FUNCTION CODE F0
1101 1571 NXLTIX2 352 BC EX WPT SAVE BYTE COUNT IN B
1102 1572 1 GOSUB NXL3B2 MOVE TO THE NEXT CHAR.
1102 1573 0
1103 1574 352 BC EX WPT RETRIEVE REMAINING CHAR COUNT.
1104 1575 1166 C=C-1 XS DEC. THE CHAR COUNT
1105 1576 1733 GONG NXLTIX2 (1571) DONE?
1106 1577 1003 GOTO NXLTIX1 (1477) YES, RESTORE THE C REGISTER
1107
1108
1109
1110 ENTRY GCPK04
1111 ENTRY GCPK05
1112 ENTRY GCPK0
1113 ENTRY GCP112
1114

* GCPKC - GET/CLEAR/PLACE KEYCODE
*- DEPENDING UPON THE INPUT CONDITIONS, THIS SUBROUTINE
*- WILL GET, CLEAR OR PLACE A KEYCODE IN THE ASN
*- FUNCTION TABLE OR PROGRAM MEMORY, WHICHEVER IS
*- APPLICABLE.
*-
*- GET- IN: A[1:0]= LOGICAL KEYCODE
*- STATUS BIT 1= 0
*- OUT: CHIP 0 SELECTED
*- C[3:0]= CORRESPONDING FUNCTION CODE IF ROM
*- = CORRESPONDING LABEL ADDRESS IF RAM
*- S3= 1 IMPLIES C[3:0] IS A RAM LABEL ADDRESS
*- (IF DIGIT 3 = 0 THEN FUNCTION CODE IS 1 BYTE
*- FUNCTION CODE)
*-
*- CLEAR- IN: A[1:0]= LOGICAL KEYCODE
*- STATUS BIT 1 = 1
*- OUT: CHIP 0 SELECTED
*-
*- PLACE- IN: A[3:2]= LOGICAL KEYCODE
*- A[1:0]= ZERO
*- B[3:0] = FUNCTION CODE
*- OUT: S3=1 IMPLIES FUNCTION WAS PLACED
*- USES: A,B,C,M,N, STATUS BIT 3
*- USES: 1 SUBROUTINE LEVEL
1140
1141
1142
1143 1600 GCPKC 4 S3= 0 -
1144 1601 1570 C=REGN 13 M_CHAINHEAD
1145 1602 34 PT= 3
1146 1603 420 LC 4 C[3:0]=FINAL END ADDR
1147 1604 530 M=C SAVE FINAL ENS ADDR IN M
1148 1605 34 PT= 3
1149 1606 374 RCR 10
1150 1607 312 C=B WPT
1151 1610 356 BC EX W SAVE .END. ADDR IN B[4:7] TOO

```

```

*
1153          ENTRY  GCPKC0          SEARCH ALBLS FROM ANY LINK
1154 1611 GCPKC0  460 LDI              CI[2:0] _ 1ST REG
1155 1612          277 CON            191          -
1156 1613 GCPK10 1046 C=C+1          X          -
1157 1614          256 AC EX          -          CHAINHEAD=REG?
1158 1615          730 MC EX          -          -
1159 1616          1546 ? A#C          X          -
1160 1617          553 GONC          GCPK04 (1674) YES, SEARCH ALBLS
1161 1620          730 MC EX          RESTORE REGS
1162 1621          256 AC EX          -
1163 1622          1160 DADD=C          C_REG
1164 1623          160 N=C          -
1165 1624          70 C=DATA          -
1166 1625          1076 C=C+1          S          END ASNS?
1167 1626          503 GONC          GCPK05 (1676) YES
1168 1627          1176 C=C-1          S          -
1169 1630          436 A=C            S          -
1170 1631 GCPK70 1434 PT=            1          INITIALIZE
1171 1632          1552 ? A#C          WPT          1ST KEYCODE?
1172 1633          43 GONC            GCPK80 (1637) YES
1173 1634          574 RCR            6          2ND KEYCODE?
1174 1635          1552 ? A#C          WPT          -
1175 1636          347 GOC            GCPK20 (1672) NOPE
1176 1637 GCPK80 1512 ? A#0          WPT          PLACE?
1177 1640          177 GOC            GCP100 (1657) NOPE
1178 1641          34 PT=            3          -
1179 1642          1612 A SR          WPT          CI[1:0]_K.C.
1180 1643          1612 A SR          WPT          -
1181 1644          246 AC EX          X          -
1182 1645          266 AC EX          XS          -
1183 1646          1074 RCR            2          PLACE THE FUNCTION CODE
1184 1647          312 C=B            WPT          -
1185 1650          10 S3=            1          FUNCTION PLACED
1186 1651 GCPK90 574 RCR            6          RESTORE REGISTER
1187 1652          1576 ? A#C          S          -
1188 1653          23 GONC            **2      (1655) -
1189 1654          574 RCR            6          -
1190 1655          1360 DATA=C          RESTORE REGISTER
1191 1656          73 GOTO            GCP112 (1665) -
1192 1657 GCP100 1414 ?S1=1          CLEAR?
1193 1660          43 GONC            GCP110 (1664) NOPE
1194 1661          112 C=0            WPT          ZERO OUT KEYCODE
1195 1662          1074 RCR            2          RESTORE REGISTER
1196 1663          1663 GOTO          GCPK90 (1651) -
1197 1664 GCP110 1074 RCR            2          CI[3:0]_FUNCTION CODE
1198 1665 GCP112 730 MC EX          SELECT CHIP 0
1199 1666          106 C=0            X          -
1200 1667          1160 DADD=C          -
1201 1670          730 MC EX          -
1202 1671          1740 RTN          RETURN
1203
1204 1672 GCPK20 260 C=N          INCREMENT TO NXT REG
1205 1673          1203 GOTO          GCPK10 (1613)-
1206
1207 1674 GCPK04 730 MC EX          RESTORE REGISTERS
1208 1675          23 GOTO            GCPK06 (1677)
1209 1676 GCPK05 256 AC EX          B[5:4]_K.C.
1210 1677 GCPK06 374 RCR            10          -
1211 1700          356 BC EX          -

```

1212	1701	160	N=C		SAVE F.C. IN N
1213	1702	174	RCR	4	CC[3:0]_CHAINHEAD
1214	1703	34	PT=	3	
1215	1704	412	A=C	WPT	
1216	1705	1	GOSUB	GTLINK	
1216	1706	0			
1217	1707	GCPK15 1346	? C#0	X	END OF CHAIN?
1218	1710	313	GONC	GCPK55 (1741)	YES, NOT FOUND
1219	1711	1	GSBLNG	UPLINK	GET NEXT LINK
1219	1712	0			
1220	1713	1076	C=C+1	S	ALBL?
1221	1714	1733	GONC	GCPK15 (1707)	NOPE
1222	1715	374	RCR	10	SAVE LINK & ADDR IN M
1223	1716	252	AC EX	WPT	-
1224	1717	416	A=C		-
1225	1720	530	M=C		-
1226	1721	1	GSBLNG	INCD2	GET KEYCODE BYTE
1226	1722	0			
1227	1723	1	GSBLNG	NXBYTA	-
1227	1724	0			
1229	1725	212	B=A	WPT	-
1229	1726	1434	PT=	1	CORRECT K.C.?
1230	1727	332	C=B	M	-
1231	1730	174	RCR	4	-
1232	1731	412	A=C	WPT	-
1233	1732	374	RCR	10	-
1234	1733	1552	? A#C	WPT	-
1235	1734	267	GOC	GCPK45 (1762)	NOPE
1236	1735	GCPK25 1512	? A#0	WPT	PLACE?
1237	1736	127	GOC	GCPK28 (1750)	NO
1238	1737	106	C=0	X	-
1239	1740	34	PT=	3	-
1240	1741	GCPK55 332	C=B	M	AC[1:0]_K.C.
1241	1742	1150	DADD=C		-
1242	1743	174	RCR	4	-
1243	1744	416	A=C		-
1244	1745	260	C=N		RESTORE F.C.
1245	1746	356	BC EX		-
1246	1747	1163	GOTO	GCP112 (1665)	-
1247	1750	GCPK28 1414	?S1=1		GET?
1248	1751	63	GONC	GCPK35 (1757)	YES
1249	1752	106	C=0	X	CLEAR K.C.
1250	1753	156	AB EX		-
1251	1754	1	GSBLNG	PTBYTA	-
1251	1755	0			
1252	1756	1073	GOTO	GCP112 (1665)	-
1253	1757	GCPK35 630	C=M		CC[3:0]_LBL ADDR
1254	1760	10	S3=	1	RAM ADDR
1255	1761	1043	GOTO	GCP112 (1665)	-
1256					
1257	1762	GCPK45 34	PT=	3	PREPARE TO GET NXT LINK
1258	1763	630	C=M		-
1259	1764	412	A=C	WPT	-
1260	1765	174	RCR	4	-
1261	1766	1213	GOTO	GCPK15 (1707)	-

* 1263 ENTRY LEFTJ

* LEFT JUSTIFY LCD

*

NOMAS
 Not Manufacturer Supported
 Incident #6799 NOT to contact manufacturer

1267	1767	LEFTJ	460	LDI	
1268	1770		40	CON	32
1269	1771		406	A=C	X
1270	1772		1434	PT=	1
1271	1773	LEFTJ1	1770	RABCL	
1272	1774		1552	? A#C	WPT
1273	1775		1763	GONC	LEFTJ1 (1773)
1274	1776		1670	RABCR	
1275	1777		1740	RTH	
1276					
1277					
1278					
1279					

UNLIST

ERRORS : 0

SYMBOL TABLE

AVAIL	304	-			
AVAIL1	322	-	315		
AVAILA	307	-			
BKRON	415	-	350		
BKROM1	417	-	422		
BKROM2	1221	-			
BST1	361	-	355		
BST2	363	-	360		
BSTE	413	-	427		
BSTE2	1362	-			
BSTEP	336	-			
BSTEP2	324	-	343	341	
BSTEP3	345	-	325		
BSTEPA	353	-			
BSTML	371	-	411	407	403
BSTML1	412	-	405		
BSTML2	377	-	370		
CALDSP	703	-			
DECAD	707	-			
DECADA	712	-			
DECADB	737	-	711		
ERRDE	55	-			
FIXEND	430	-			
FLINK	450	-			
FLINK1	455	-	513		
FLINK2	457	-	454		
FLINK3	470	-	465	461	
FLINK5	512	-	476		
FLINKA	447	-			
FLINKM	451	-			
FLINKP	445	-			
GCP100	1657	-	1640		
GCP110	1664	-	1660		
GCP112	1665	-	1761	1756	1747 1656
GCPK04	1674	-	1617		
GCPK05	1676	-	1626		
GCPK06	1677	-	1675		
GCPK10	1613	-	1673		
GCPK15	1707	-	1766	1714	
GCPK20	1672	-	1636		
GCPK25	1735	-			
GCPK28	1750	-	1736		
GCPK35	1757	-	1751		
GCPK45	1762	-	1734		
GCPK55	1741	-	1710		
GCPK70	1631	-			
GCPK80	1637	-	1633		
GCPK90	1651	-	1663		
GCPKC	1600	-			
GCPKC0	1611	-			
GETPC	520	-			
GETPCA	522	-			
GTBYT	660	-			
GTBYTA	673	-	661		
GTBYTO	662	-			
GTO..	561	-	540		

GT0.1	615	-	600						
GT0.2	603	-	611	574					
GT0.2A	612	-	607						
GT0.3	647	-	614						
GT0.4	575	-	571						
GT0.5	652	-							
GT0NN	531	-							
GT0NN2	551	-	542						
IN2B	1012	-	777						
IN2BA	1114	-	1016						
IN2BB	1032	-	1021						
IN2R9	1124	-	1116						
IN2ST0	1136	-	1126						
IN3B	1145	-							
INB0	154	-	101						
INB1	51	-	103						
INB2	150	-	105						
INB3	144	-	107						
INB4	140	-	111						
INB5	134	-	113						
INB6	114	-							
INBEX	155	-	153	147	143	137	120		
INBEXA	120	-	54						
INBYT	746	-							
INBYT0	743	-							
INBYT1	752	-							
INBYTC	744	-							
INBYTP	745	-							
INC1	734	-	727						
INC2	735	-	706						
INC21	732	-	724						
INCAD	717	-							
INCAD2	723	-							
INCADA	726	-	722						
INCADB	741	-	720						
INCADP	721	-							
INEX	1112	-	1123	1076	1031				
INEXA	1031	-	1011	1002					
INL1	216	-	212						
INL2	213	-	217						
INL3	225	-	241						
INLIN	166	-	156						
INLIN2	766	-							
INLNK1	1051	-	1045						
INLNK2	1062	-	1050						
INN2B	1117	-	1156	1151	1142	1137	1131	1033	1026
INRCLS	1133	-	1144						
INSHR2	1134	-	1162						
INSHRT	1027	-							
INSLIN	764	-							
INSTR	1163	-							
INSTR1	1164	-	1174						
INTXC	1175	-							
INTXC1	1203	-	1207						
INTXC2	1206	-	1202						
LEFTJ	1767	-							
LEFTJ1	1773	-	1775						
INHEND	1347	-	1313						
INML	1270	-	1311	1307	1303				
INML1	1273	-							

LINML2	1312	-	1305	1301					
LINN1A	1223	-							
LINNM1	1220	-	1215						
LINNM2	1245	-	1242						
LINNM3	1243	-	1250						
LINNM4	1254	-	1246						
LINNM5	1251	-	1241						
LINNM6	1261	-	1253						
LINNUM	1213	-							
LINRM2	1333	-	1344						
LINRM3	1345	-	1342	1336					
LINRM4	1337	-	1332						
LINROM	1314	-	1230						
NOROOM	263	-	171						
NROOM	170	-							
NROOM1	266	-	276						
NROOM2	275	-	265						
NROOM3	302	-							
NXBYT3	667	-							
NXBYTA	671	-							
NXL1B	1443	-							
NXL2B1	1556	-	1544						
NXL3B2	1543	-	1510	1506	1504				
NXLCHN	1511	-							
NXLDE	1454	-	1473	1471	1467	1451	1447		
NXLDE2	1474	-	1532	1465	1462				
NXLDEL	1375	-							
NXLIN	1424	-	1453						
NXLIN1	1436	-	1425						
NXLIN2	1501	-	1442						
NXLIN3	1537	-	1502						
NXLIN4	1561	-	1540						
NXLINA	1437	-	1377						
NXLR0	1563	-							
NXLSS1	1373	-	1370						
NXLSST	1367	-							
NXLTX	1567	-	1562	1521					
NXLTX1	1477	-	1577						
NXLTX2	1571	-	1576						
PBA0	45	-	63	1					
PBA1	41	-	65	3					
PBA2	35	-	67	5					
PBA3	31	-	71	7					
PBA4	25	-	73	11					
PBA5	21	-	75	13					
PBA6	14	-	76						
PBA6A	76	-	133						
PBAEND	46	-	44	40	34	30	24	20	
PTBYTM	441	-							
PTL0	121	-	61						
PUTPCL	1363	-							
SKPLIN	1371	-							
SKPR10	1402	-	1405						
SKPR20	1406	-	1411						
SKPR30	1412	-	1423						
SKPROM	1400	-	1374						
TBLINB	100	-							
TBLPBA	0	-							
TBLPTL	60	-							

ENTRY TABLE

AVAIL	304	-
AVAILA	307	-
BKROM2	1221	-
BSTE	413	-
BSTE2	1362	-
BSTEP	336	-
BSTPA	353	-
CALDSP	703	-
DECAD	707	-
DECADA	712	-
ERRDE	55	-
FIXEND	430	-
FLINK	450	-
FLINKA	447	-
FLINKM	451	-
FLINKP	445	-
GCP112	1665	-
GCPK04	1674	-
GCPK05	1676	-
GCPKC	1600	-
GCPKC0	1611	-
GETPC	520	-
GETPCA	522	-
GTBYT	660	-
GTBYTA	673	-
GTBYTO	662	-
GTO.5	652	-
GTONN	531	-
IN3B	1145	-
INBYT	746	-
INBYT0	743	-
INBYT1	752	-
INBYTC	744	-
INBYTP	745	-
INCAD	717	-
INCAD2	723	-
INCADA	726	-
INCADP	721	-
INEX	1112	-
INLIN	166	-
INLIN2	766	-
INSHRT	1027	-
INSLIN	764	-
INSTR	1163	-
INTXC	1175	-
LEFTJ	1767	-
LINN1A	1223	-
LINN1	1220	-
LINNUM	1213	-
NROOM3	302	-
NXBYT3	667	-
NXBYTA	671	-
NXL1B	1443	-
NXL3B2	1543	-
NXLCHN	1511	-
NXLDEL	1375	-

NXLIN	1424	-
NXLIN3	1537	-
NXLINA	1437	-
NXLSST	1367	-
NXLTX	1567	-
PTBYTM	441	-
PUTPCL	1363	-
SKPLIN	1371	-

EXTERNAL REFERENCES

AVAIL	166	615		
AVAIL	167	616		
BKROM2	425			
BKROM2	426			
BSTE2	413			
BSTE2	414			
CPGMHD	1535			
CPGMHD	1536			
DECAD	1160			
DECAD	1161			
DECADA	266	361	1474	1525 1527
DECADA	267	362	1475	1526 1530
ERROR	55			
ERROR	56			
FIXEND	174	1100		
FIXEND	175	1101		
FLINK	351			
FLINK	352			
FLINKA	1066			
FLINKA	1067			
FLINKM	172			
FLINKM	173			
FLINKP	326	1231		
FLINKP	327	1232		
FSTIN	356	572	1251	
FSTIN	357	573	1252	
GENLNK	1070	1073		
GENLNK	1071	1074		
GETLIN	1351			
GETLIN	1352			
GETPC	345	445	1062	1323
GETPC	346	446	1063	1324
GTBYTA	363	435		
GTBYTA	364	436		
GTFEND	451			
GTFEND	452			
GTLINK	566	1533	1705	
GTLINK	567	1534	1706	
GTLNKA	177			
GTLNKA	200			
IN3B	1013			
IN3B	1014			
INBYT	773			
INBYT	774			
INBYT0	1040	1056	1145	
INBYT0	1041	1057	1146	
INBYT1	261			
INBYT1	262			
INBYTC	1005	1046	1054	1166
INBYTC	1006	1047	1055	1167
INBYTP	1121			
INBYTP	1122			
INCAD2	433	601	667	1103 1721
INCAD2	434	602	670	1104 1722
INCADA	671	1064	1254	
INCADA	672	1065	1255	

INCADP	746			
INCADP	747			
INSHRT	1134			
INSHRT	1135			
INSSUB	764			
INSSUB	765			
INSTR	1007	1060		
INSTR	1010	1061		
INTXC	1003	1051		
INTXC	1004	1052		
LINN1A	555			
LINN1A	556			
LINNMI	334			
LINNMI	335			
LINNUM	336			
LINNUM	337			
MSGDE	57			
HFRC	557	1112		
HFRC	560	1113		
HFRPU	656			
HFRPU	657			
NXBYTA	603	1256	1375	1723
NXBYTA	604	1257	1376	1724
NXL3B2	1456	1514	1516	1541 1572
NXL3B2	1457	1515	1517	1542 1573
NXLIN	375	1273	1565	
NXLIN	376	1274	1566	
PACKE	302	620		
PACKE	303	621		
PACKN	652			
PACKN	653			
PATCH1	715			
PATCH1	716			
PTBYTA	441	1027	1754	
PTBYTA	442	1030	1755	
PTBYTM	271			
PTBYTM	272			
PTLINK	204			
PTLINK	205			
PUTPC	1105	1363		
PUTPC	1106	1364		
PUTPCD	332	650		
PUTPCD	333	651		
ROMH35	1414			
ROMH35	1415			
ROMHED	1314	1420		
ROMHED	1315	1421		
RTN30	534	654		
RTN30	535	655		
SKPDEL	1512			
SKPDEL	1513			
SKPLIN	1333			
SKPLIN	1334			
UPLINK	455	575	1243	1711
UPLINK	456	576	1244	1712
XROW1	547			
XROW1	550			

End of VASM assembly

OPTIONS: L C S

309

* HP41C MAINFRAME MICROCODE ADDRESSES 026000-27777

* CONTENTS:

*

5	FILE	CN11B
6	ENTRY	TXTLBL
7	ENTRY	TXTLB1
8	ENTRY	AOUT15
9	ENTRY	APHST*
10	ENTRY	APNDNM
11	ENTRY	APPEND
12	ENTRY	ARGOUT
13	ENTRY	ASCCLD
14	ENTRY	CLLCDE
15	ENTRY	CLRLCD
16	ENTRY	DAT106
17	ENTRY	DAT231
18	ENTRY	DAT260
19	ENTRY	DAT280
20	ENTRY	DAT300
21	ENTRY	DAT320
22	ENTRY	DAT400
23	ENTRY	DAT500
24	ENTRY	DATENT
25	ENTRY	DECMPL
26	ENTRY	INBCHS
27	ENTRY	INBYTJ
28	ENTRY	MASK
29	ENTRY	NXBYTO
30	ENTRY	NXTBYT
31	ENTRY	OPROMT
32	ENTRY	OUTLCD
33	ENTRY	ROLBAK
34	ENTRY	SCROLO
35	ENTRY	SCROLL
36	ENTRY	STBT10
37	ENTRY	STOLCC
38	ENTRY	TEXT
39	ENTRY	XROMNF
40	ENTRY	XECROM

*

* SPECIAL CHAR TABLE

*

44	0	ASCTBL	177	CON	@177	LAZY T
45	1		141	CON	@141	SMALL A
46	2		142	CON	@142	SMALL B
47	3		143	CON	@143	SMALL C
48	4		144	CON	@144	SMALL D
49	5		145	CON	@145	SMALL E

* LCD 106 OVERBAR ... HELIOS 0 SMALL DIAMOND

51	6		0	CON	@0	LCD 106
52	7		140	CON	96	SUPERSCRIPT T

* LCD 108 ONE-LEGGED HANGMAN ... HELIOS 6 UPPER CASE GAMMA

54	10		6	CON	@6	LCD 108

* LCD 109 TWO-LEGGED HANGMAN ... HELIOS 4 ALPHA

56	11		4	CON	4	LCD 109

* LCD 10A TWO-LEGGED ONE-ARMED HANGMAN ... HELIOS 5 BETA

58	12	5 CON	5	LCD 10A
* LCD 10B	COMPLETE HANGMAN ...	HELIOS 1	LITTLE X	
60	13	1 CON	1	LCD 10B
61	14	14 CON	0014	MU
62	15	35 CON	29	NOT EQUAL SIGN
63	16	176 CON	0176	SIGMA SIGN
64	17	15 CON	13	ANGLE SIGN

```

*
*
* ARGOUT - OUTPUT ALPHA REGISTER TO DISPLAY
* CALLING SEQUENCE:
* IF S8=1, NO SCROLL, PROMPT
* IF S8=0, SCROLL, NO PROMPT
* IF S8=0, THEN S9 INDICATES WHETHER THE KEYBOARD HAS BEEN RESET
*   S9=1 : KEYBOARD ALREADY BEEN RESET
*   S9=0 : KEYBOARD NOT BEEN RESET
* BY SET/RESET S8,S9 THE KEYBOARD WILL REMAIN ALIVE DURING SCROLLING.
*   GOSUB ARGOUT
* ASSUME NOTHING, RETURN WITH CHIP 0 ENABLE
* USED A,B,C. CALLED NXBYTA, ASCLCD. 2 SUB LEVELS.
*

```

```

79 20 ARGOUT 116 C=0
80 21        1760 PFAD=C
81 22        134 PT= 4          LOAD FIRST CHAR ADDR
82 23        620 LC 6          = 6008 (BYTE 3, REG.8)
83 24        460 LDI
84 25        214 CON2 8 12
85 26        1474 RCR 1
86 27        416 A=C
87 30 AOUT05 1 GOSUB NXBYTA
87 31        0
88 32        1434 PT= 1
89 33        1352 ? C#0 WPT      IS A LEADING BLANK ?
90 34        127 GOC AOUT10 ( 46) NO
91 35        460 LDI
92 36        5 CON2 0 5        CHECK END OF AREG.
93 37        34 PT= 3
94 40        102 C=0 PT
95 41        1552 ? A#C WPT      LAST CHAR IN AREG. ?
96 42        1667 GOC AOUT05 ( 30) NO
97 43        1 GOSUB CLLCDE     CLEAR LCD
97 44        0
98 45        153 GOTO AOUTR0 ( 62)
99 46 AOUT10 1074 RCR 2
100 47        1 GOSUB CLLCDE
100 50        0
101 51        1340 DISOFF
102 52        453 GOTO AOUT20 ( 117)
103 53 AOUT15 156 AB EX W
104 54        460 LDI
105 55        5 CON2 0 5
106 56        34 PT= 3
107 57        102 C=0 PT
108 60        1552 ? A#C WPT      END OF ALPHA REG. ?
109 61        237 GOC AOUT18 ( 104) NOT YET
110 62 AOUTR0 236 B=A S        B(13) _ LCD COUNTER
111 63        460 LDI
112 64        37 CON 037
113 65        414 ?S8=1          PROMPT ?
114 66        43 GONC AOUT16 ( 72) NO

```

NOMAS
 Not Manufacturer Supported
 recipient agrees NOT to contact manufacturer


```

115 67      1750 SLSABC
116 70      676 A=A-1 S      LCD FULL ?
117 71      67 GOC      AOUTRT ( 77 ) YES
118 72 AOUT16 1046 C=C+1 X      Q37+1 = Q40
119 73 AOUT17 676 A=A-1 S      DO WE HAVE TO LEFT JUSTIFY ?
120 74      37 GOC      AOUTRT ( 77 ) NO
121 75      1750 SLSABC
122 76      1753 GOTO      AOUT17 ( 73 )
123 77 AOUTRT 1340 DISOFF
124 100     1440 DISTOG      TURN DISPLAY ON AGAIN
125 101     336 C=B S
126 102     1 GOLONG:STOLCC      SAVE THE LCD COUNTER
126 103     2
127 104 AOUT18 1536 ? A#0 S      LCD FULL ?
128 105     47 GOC      AOUT19 ( 111 ) NO
129 106     414 ?S8=1      SCROLL NEEDED ?
130 107     1 GSUBNC:SCROLL      YES
130 110     0
131 111 AOUT19 1 GOSUB ENCP00
131 112     0
132 113     34 PT= 3
133 114     1 GOSUB NXBYTA      GET NEXT CHAR
133 115     0
134 116     1074 RCR 2
135 117 AOUT20 216 B=A W
136 120     1 GOSUB ENLCD      ENABLE LCD
136 121     0
137 122     1574 RCR 12      C(1:0) _ CHAR
138 123     126 C=0 XS      C(2) _ 0
139 124     1 GOSUB ASCLCD      SEND IT TO LCD
139 125     0
140 126     1253 GOTO      AOUT15 ( 53 )

```

```

*
* ASCLCD - SEND A ASCII CHAR TO LCD
* CALLED WITH ASCII IN C(1:0)
* ASSUME LCD ENABLE, RETURN WITH LCD ENABLE.
* USED A,X, B,S, C, 1 SUB LEVEL.
* GOSUB ASCLCD
*

```

```

148 127 COLON 460 LDI
149 130      200 CON2 8 0
150 131      243 GOTO PUNC ( 155 )
151 132 COMMA 460 LDI
152 133      300 CON2 12 0
153 134      213 GOTO PUNC ( 155 )
154 135 ASCLCD 406 A=C X
155 136      26 A=0 XS
156 137      460 LDI
157 140      72 CON2 3 10
158 141      1546 ? A#C X      IS THIS A COLON ?
159 142      1653 GONC COLON ( 127 ) YES
160 143      460 LDI
161 144      54 CON2 2 12
162 145      1546 ? A#C X      IS THIS A COMMA ?
163 146      1643 GONC COMMA ( 132 ) YES
164 147      460 LDI
165 150      56 CON2 2 14
166 151      1546 ? A#C X      IS THIS A PERIOD ?
167 152      367 GOC MASK ( 210 ) NO
168 153 PERIOD 460 LDI

```

```

169 154      100 CON2  4      0
170 155 PUNC      406 A=C    X
171 156      1670 FRSABC
172 157      1730 CST EX
173 160      514 ?S6=1
174 161      137 GOC      PUNC10 ( 174 ) YES
175 162      1214 ?S7=1
176 163      117 GOC      PUNC10 ( 174 ) YES
177 164      1730 CST EX
178 165      1334 PT=      13
179 166      1420 LC      12
180 167      436 A=C      S
181 170      336 C=B      S
182 171      1576 ? A#C    S
183 172      63 GONC      OUTLCD ( 200 ) YES
184 173      123 GOTO      PUNC20 ( 205 )
185 174 PUNC10 1730 CST EX
186 175      1750 SLSABC
187 176      460 LDI
188 177      40 CON      Q40
189 200 OUTLCD 1336 ? B#0    S
190 201      43 GONC      PUNC20 ( 205 )
191 202      176 AB EX    S
192 203      676 A=A-1    S
193 204      176 AB EX    S
194 205 PUNC20 1560 C=CORA-
195 206      1750 SLSABC
196 207      1740 RTN

```

```

*
* MASK - CONVERT A ASCII TO LCD CHAR FORM (NOT INCLUDING COMMA,
* PERIOD AND COLON)
* CALLED WITH ASCII IN A[2:0].
* TWO CALLING SEQUENCE:
* 1.  GOSUB MASK
*     NOP
*     CALLED MASK FOLLOWED BY A NOP, THE LCD CHAR WILL RETURN
*     IN C[2:0], CHIP ENABLE UNCHANGED.
*     USED A,X, C. ASSUME NOTHING. 1 SUB LEVEL.
* 2.  GOSUB MASK
*     <ANYTHING BUT NOP>
*     NOT FOLLOWED A NOP WILL CAUSE THE CHAR BEEN SEND TO DISPLAY,
*     RETURN WITH CHIP 0 ENABLE. USED A,X, B,S, C. 1 SUB LEVEL.
*     ASSUME LCD ENABLE.
*

```

```

213 210 MASK      26 A=0    XS
214 211      460 LDI
215 212      40 CON      Q40
216 213      1406 ? A#C    X
217 214      227 GOC      MASK10 ( 236 ) YES, SPECIAL CHAR
218 215      460 LDI
219 216      140 CON      Q140
220 217      1406 ? A#C    X
221 220      163 GONC      MASK10 ( 236 ) YES, SPECIAL CHAR
222 221      246 AC EX    X
223 222      1730 CST EX
224 223      504 S6=      0
225 224      1730 CST EX
226 225 MASKRT 406 A=C      X
227 226      660 C=STK
228 227      1460 CXISA

```

```

229 230          560 STK=C
230 231          246 AC EX  X
231 232          1506 ? A#0 X
232 233          1640 RTN NC
233 234          6 A=0      X
234 235          1433 GOTO   OUTECD ( 200)
235 236 MASK10   116 C=0
236 237          534 PT=    6          CHECK SPECIAL CHAR TABLE
237 240          220 LC      2          TABLE ENTRY AT 0000 OF QUAR 11
238 241          1420 LC     12
239 242          34 PT=     3
240 243 MASK20   1460 CXISA          LOAD 1 CHAR FROM TABLE
241 244          1546 ? A#C X          MATCH A SPECIAL CHAR ?
242 245          63 GONC          MASK30 ( 253) YES
243 246          1042 C=C+1 PT          POINT TO NEXT WORD
244 247          1743 GONC          MASK20 ( 243) GO ON !
245 250          460 LDI
246 251          72 CON          072          ALL SEGMENT IF NOT FOUND
247 252          1533 GOTO   MASKRT ( 225)
248 253 MASK30   74 RCR          3
249 254          126 C=0      XS
250 255          1066 C=C+1 XS          C[2:0] HAS THE SPECIAL CHAR
251          LEGAL
252 256          1473 GOTO   MASKRT ( 225) REPLACE IT

```

```

*
*
* TEXT FUNCTION - EXECUTION OF TEXT FC IN RUN TIME
* ASSUME PGM COUNTER POINTING THE 1ST BYTE OF THE TEXT FUNCTION
* THIS ROUTINE WILL PICK UP THE CHAR FROM NEM AND STICK IT TO
* ALPHA REG. IF THE 1ST CHAR IS A LAZY T, THE STRING WILL BE
* APPENDING TO ALPHA REG. OTHERWISE, THE ALPHA REG WILL BE CLEARED
* BEFORE THE STRING GOES IN.
* CALLED APPEND. RETURN TO NFRPU. PC WILL POINT TO LAST BYTE OF TEXT
* FC ON EXIT.
*

```

```

264 257 TEXT          16 A=0      W
265 260          1 GOSUB   GETPC          GET PRM COUNTER
265 261          0
266 262          1 GOSUB   GTBYT
266 263          0
267 264          1434 PT=    1
268 265          102 C=0      PT          C(1:0) _ STRING COUNTER
269 266          1152 C=C-1 WPT
270 267          1540 RTN C          RTN IF "F0" F.C.
271 270          374 RCR      10          C(4) _ STRING COUNTER
272 271          134 PT=     4          MOVE COUNTER TO A(4)
273 272          402 A=C      PT
274 273          1 GOSUB   NXTBYT          GET FIRST CHAR
274 274          0
275 275          216 B=A      W          SAVE THE COUNTER IN B
276 276          1634 PT=    0
277 277          130 G=C
278 300          406 A=C      X
279 301          26 A=0      XS
280 302          106 C=0      X
281 303          1160 DADD=C
282 304          460 LDI
283 305          177 CON      127          TEST FIRST CHAR
284 306          1546 ? A#C X          IS IT A LAZY "T"
285 307          1 GSUBC   INTARG          NO, INITIALIZE ALPHA REG

```

```

285 310          1
286 311 TEXT30  316 C=B      W
287 312          416 A=C      W
288 313          34 PT=       3
289 314          174 RCR       4
290 315          1146 C=C-1  X      C.X _ STRING COUNTER
291 316          1 GOLC      PUTPC  ALL DONE ?
291 317          3
292 320 TEXT40  374 RCR       10
293 321          416 A=C      W
294 322          1 GOSUB     NXTBYT  POINT TO NEXT CHAR
294 323          0
295 324          216 B=A      W      SAVE COUNTER IN B
296 325          1634 PT=      0
297 326          130 G=C
298 327          106 C=0      X
299 330          1160 DADD=C    ENABLE CHIP 0
300 331          1 GOSUB     APHDNW  STORE CHAR TO AREG.
300 332          0
301 333          1563 GOTO    TEXT30 ( 311 )

*
* SCROLL - TURN ON THE DISPLAY AND DECIDE WHETHER TO HAVE A DELAY
* AFTER PUSHING A CHAR OFF LEFT END.
* S9=1 MEANS SCROLL IS NOT REQUIRED, NO DELAY
* S9=1 MEANS KEYBOARD ALREADY BEEN RESET.
* IF ANY KEY HIT WHEN S9=1, NO DELAY ANY MORE. THIS WAY THE
* KEYBOARD WILL STAY ALIVE DURING SCROLLING.
* DESTROYS C.X MAY SET S9
* MAY USE A SUBROUTINE LEVEL TO CALL RST05
*
312 334 SCROLL  1340 DISOFF
313 335          1440 DISTOG
314 336 SCROL0  414 ?S8=1      SCROLL REQUIRED ?
315 337          1540 RTN C      NO
316 340          1114 ?S9=1     HAS KEY BOARD BEEN RESET?
317 341          77 GOC         SCROL2 ( 350 ) YES
318 342          1710 RST KB
319 343          1714 CHK KB
320 344          67 GOC         SCROL5 ( 352 ) OLD KEY STILL DOWN
321 345          1110 S9=       1 REMEMBER OLD KEY IS UP
322 346          1 GOSUB     RST05 DELAY FOR DEBOUNCE
322 347          0
323 350 SCROL2  1714 CHK KB     IS A NEW KEY DOWN ?
324 351          1540 RTN C      YES, NO SCROLL
325 352 SCROL5  460 LDI
326 353          1600 CON       01600
327 354          746 C=C+C      X      *** 01600 FOR FINAL PRODUCT *****
328
329 355 SROL10  1146 C=C-1      X
330 356          1773 GONC      *-1   ( 355 )
331 357          1740 RTN

*
* CLEAR LCD
* CLR LCD - ASSUME LCD ENABLE
* CLLCDE - ENABLE LCD & CLEAR IT
*
337 360 CLLCDE  460 LDI
338 361          20 CON2       1      0
339 362          1160 DADD=C
340 363          460 LDI
DISABLE SLEEPER CHIP

```

341	364	375	CON2	15	13	
342	365	1760	PFAD=C			ENABLE LCD CHIP
343	366	CLRLCD	634	PT=	11	
344	367		112	C=0	WPT	
345	370		334	PT=	10	
346	371		220	LC	2	
347	372		1234	PT=	7	
348	373		220	LC	2	
349	374		134	PT=	4	
350	375		220	LC	2	
351	376		1434	PT=	1	
352	377		220	LC	2	
353	400		450	SRLABC		
354	401		450	SRLABC		
355	402		450	SRLABC		
356	403		1740	RTH		

*
* NXTBYT - GET NEXT BYTE IN RAM OR ROM
*

360			ENTRY	NBYTAD	
361	404	NBYTAD	1	GOSUB	ENCF00
361	405		0		
362			ENTRY	NBYTAB	
363	406	NBYTAB	156	AB EX	
364	407	NXTBYT	34	PT=	3
365	410		314	?S10=1	ROM MEMORY ?
366	411		1	GOLNC	NXBYTA
366	412		2		NO
367	413	NXBYTO	556	A=A+1	
368				LEGAL	
369	414		1	GOLONG	GTBYTO
369	415		2		

*
* APPEND - APPEND A CHAR TO ALPHA REG
* CHAR IN G
* ASSUMED CHIP 0 ENABLE. USED A,C. 1 SUB LEVEL
* TWO ENTRIES :
* 1. APPEND : WILL GIVE A WARNING IF AREG FULL AND AUDIO ENABLE
* 2. APNDHW : NO WARNING EVEN IF AREG FULL
*

378	416	APPEND	1070	C=REGH	8	CHECK IF AREG. ALMOST FULL?
379	417		1434	PT=	1	
380	420		1074	RCR	2	CHECK SECOND LAST CHAR
381	421		1352	? C#0	WPT	STILL EMPTY
382	422		1	GOSUB	TONE7X	NO, GIVE A WARNING
382	423		1			
383	424	APNDHW	1434	PT=	1	
384	425		1070	C=REGH	8	
385	426		1574	RCR	12	
386	427		416	A=C		
387	430		770	C=REGH	7	
388	431		1574	RCR	12	
389	432		412	A=C	WPT	
390	433		256	AC EX	W	
391	434		1050	REGH=C	8	
392	435		670	C=REGH	6	
393	436		1574	RCR	12	
394	437		412	A=C	WPT	
395	440		256	AC EX	W	
396	441		750	REGH=C	7	

```

397 442      570 C=REGN 5
398 443      1574 RCR      12
399 444      412 A=C      UPT
400 445      256 AC EX    W
401 446      650 REGN=C 6
402 447      256 AC EX    W
403 450      1634 PT=      0
404 451      230 C=G
405 452      550 REGN=C 5
406 453      1740 RTN

```

```

**
* DATA ENTRY - WHEN PARSE DETECTS A DATAENTRY FC, IT PUTS THE FC
* IN CC1:0] AND BRANCH TO HERE.
*

```

```

411 454 DATENT 346 BC EX X
412 455      1 GOSUB OFSHT      RESET SHIFT
412 456      0
413 457      346 BC EX X
414 460      126 C=0      XS
415 461      406 A=C      X      COPY FC TO A.X
416 462      1634 PT=      0
417 463      130 G=C      COPY FC TO REG.G TOO
418 464      1670 C=REGN 14
419 465      574 RCR      6
420 466      1730 CST EX
421 467      1404 S1=      0      RESET CATLOG FLAG
422 470      1730 CST EX
423 471      474 RCR      8
424 472      1650 REGN=C 14
425 473      766 C=C+C      XS
426 474      766 C=C+C      XS      ALREADY IN DATAENTRY ?
427 475      457 GOC      DAT200 ( 542 ) YES
428 476      1506 ? A#0      X      BACK ARROW ?
429 477      207 GOC      DAT110 ( 517 ) NO
430 500      214 ?S5=1      MSGFLAG SET ?
431 501      347 GOC      DAT140 ( 535 ) YES
432 502 DAT102      14 ?S3=1      PROGRAM MODE ?
433 503      43 GONC      DAT105 ( 507 ) NO
434 504      460 LDI
435 505      13 CON      @013      DELETE
436 506      1740 RTN      RETURN TO PARSE
437 507 DAT105      1214 ?S7=1      ALPHA MODE ?
438 510      47 GOC      DAT106 ( 514 ) YES
439 511      460 LDI
440 512      167 CON2      7      7      CLX
441 513      1740 RTH      RETURN TO PARSE
442 514 DAT106      460 LDI
443 515      207 CON2      8      7      CLA
444 516      1740 RTN
445 517 DAT110      460 LDI
446 520      34 CON2      1      12      LOAD CHS
447 521      1546 ? A#C      X      IS IT A CHS ?
448 522      47 GOC      DAT120 ( 526 ) NO
449 523      460 LDI
450 524      124 CON2      5      4      CHS FC
451 525      1740 RTN      RETURN TO PARSE
452 526 DAT120      1 GOSUB STFLGS      SET MSGFLG & DATAENTRY FLAG
452 527      0
453
454 530      14 ?S3=1      STFLGS LEAVES SS ONE-HALF UP
                        ALPHA MODE ?

```

```

455 531          1 GOLNC DIGST*      INITIALIZE DIGIT ENTRY
455 532          2
456 533          1 GOLONG APHST*     INITIALIZE ALPHA ENTRY
456 534          2
457 535 DAT140   204 SS=      0      CLEAR MSGFLAG
458 536          1670 C=REGN 14
459 537          1630 C=ST
460 540          1650 REGN=C 14
461 541          453 GOTO   DAT220 ( 606)
462 542 DAT200   219 SS=      1      SET MSGFLAG
463 543          1670 C=REGN 14
464 544          1630 C=ST
465 545          1650 REGN=C 14
466 546          14 ?S3=1             PROGMODE ?
467 547          647 GOC   DAT300 ( 633) YES
468 550          1214 ?S7=1          ALPHA MODE ?
469 551          253 GONC   DAT235 ( 576) NO
470 552 DAT230   460 LOI
471 553          177 CON    127
472 554          406 A=C    X
473 555          1634 PT=    0
474 556          230 C=G             LOAD THE FC
475 557          1346 ? C#0 X        IS THIS A BACK ARROW ?
476 560          1 GOLNC BAKAPH     YES
476 561          2
477 562          1546 ? A#C X        IS THIS A LAZY "T"
478 563          257 GOC   DAT240 ( 610) NO
479 564          1 GOSUB  BLINK      BLINK AND IGNORE IT
479 565          0
480 566          203 GOTO   DAT220 ( 606)
481 567 DAT231   1670 C=REGN 14      SET NUMERIC DATA ENTRY FLAG
482 570          474 RCR    8        (FLAG 22)
483 571          1730 CST EX
484 572          1410 S1=    1      SET FLAG 22
485 573          1730 CST EX
486 574          574 RCR    6
487 575          1650 REGN=C 14
488 576 DAT235   1 GOSUB  DGENS8    TELL DIGENT NO CHS WHEN X=0
488 577          0
489 600          1 GOSUB  NOREG9
489 601          0
490 602          1 GOSUB  RG9LCD
490 603          0
491 604          1 GOSUB  RFDSS5
491 605          0
492 606 DAT220   1 GOLONG NFRKB
492 607          2
493 610 DAT240   1 GOSUB  APPEND     APPEND TO ALPHA REG.
493 611          0
494 612          1170 C=REGN 9
495 613          1376 ? C#0 S        LCD FULL ?
496 614          67 GOC   DAT245 ( 622) NOT YET
497 615          376 BC EX  S
498 616          1 GOSUB  ENLCD
498 617          0
499 620          1670 FRSABC
500 621          33 GOTO   DAT260 ( 624)
501 622 DAT245   1 GOSUB  ROLDAK
501 623          0
502 624 DAT260   106 C=0    X

```

```

503 625          230 C=G
504 626          1 GOSUB  ASCLCD      SEND IT TO LCD
504 627          0
505 630 DAT280   1 GOSUB  OPROMT      OUT PUT THE PROMPT
505 631          0
506 632          1543 GOTO    DAT220 ( 606)
507 633 DAT300  1214 ?S7=1      ALPHA MODE ?
508 634          1 GOLC    DAT500      YES
508 635          3

```

*
* DIGIT ENTRY IN PRM MODE
*

```

512 636          1 GOSUB  GETPC
512 637          0
513 640          1 GOSUB  DELLIN
513 641          0
514 642 DAT320   116 C=0
515 643          1160 DADD=C      ENABLE CHIP 0
516 644          1 GOSUB  DGENSB      TELL DIGENT NO CHS WHEN X=0
516 645          0
517 646          1 GOSUB  GETPC
517 647          0
518 650          36 A=0      S      INITIALIZE CT
519 651          216 B=A
520 652          1 GOSUB  NXBYTA
520 653          0
521 654          1434 PT=      1
522 655          1352 ? C#0  UPT      IS FIRST BYTE A NULL ?
523 656          53 GONC    DAT322 ( 663) YES
524 657          156 AB EX  W      OTHERWISE INSERT A NULL FIRST
525 660          1 GOSUB  INBYT0
525 661          0
526 662          36 A=0      S
527 663 DAT322   106 C=0      X
528 664          1160 DADD=C      ENABLE CHIP 0
529 665          1014 ?S2=1      MANTISSA NEGATIVE ?
530 666          33 GONC    DAT325 ( 671) NO
531 667          1 GOSUB  INBCHS      INSERT A CHS FIRST
531 670          0
532 671 DAT325   340 SEL Q
533 672          1534 PT=      12
534 673          240 SEL P
535 674          1170 C=REGH 9      LOAD D.P. POS COUNTER IN REG.9(13)
536 675          1240 SETDEC
537 676          1376 ? C#0  S
538 677          173 GONC    DAT333 ( 716)
539 700          1236 C=-C      S      TENTH COMPLEMENT
540 701          1140 SETHEX
541 702          376 CB EX  S      SAVE THE D.P. POS IN REG.B
542 703 DAT330   336 C=B      S      C.S _ D.P. POSITION
543 704          1176 C=C-1  S      OUTPUT D.P. NOW ?
544 705          133 GONC    DAT335 ( 720) NOT YET
545 706          1614 ?S0=1      D.P. HIT ?
546 707          113 GONC    DAT335 ( 720) NO
547 710          376 CB EX  S
548 711          460 LDI
549 712          32 CON2      1      10
550 713          1 GOSUB  INBYTJ      INSERT A D.P. TO MEM
550 714          0
551 715          1663 GOTO    DAT330 ( 703)

```



```

552 716 DAT333 1176 C=C-1 S
553 717 1140 SETHEX
554 720 DAT335 376 BC EX S
555 721 340 SEL Q
556 722 1324 ? PT= 13 FINISHED DIGIT 0 ?
557 723 527 GOC DAT380 ( 775) YES, ALL DONE
558 724 1170 C=REGN 9
559 725 1042 C=C+1 PT LAST DIGIT IN MANTISSA ?
560 726 257 GOC DAT350 ( 753) YES
561 727 1024 ? PT= 2 END OF MANTISSA ?
562 730 127 GOC DAT345 ( 742) YES
563 731 1142 C=C-1 PT RESTORE THE DIGIT
564 732 1734 INC PT MOVE THE DIGIT TO G
565 733 120 LC 1
566 734 130 G=C
567 735 1724 DEC PT POINT TO NEXT DIGIT
568 736 240 SEL P
569 737 1 GOSUB INBYT INSET THE DIGIT
569 740 0
570 741 1423 GOTO DAT330 ( 703)
571 742 DAT345 1414 ?S1=1 EEX HIT ?
572 743 167 GOC DAT370 ( 761) YES
573 744 1614 ?S0=1 D.P. HIT ?
574 745 303 GONC DAT380 ( 775) NO, NO PROMPT
575 746 1176 C=C-1 S D.P. AT DIGIT 3 ?
576 747 267 GOC DAT380 ( 775) YES, NO PROMPT
577 750 1176 C=C-1 S D.P. AT DIGIT 4 ?
578 751 247 GOC DAT380 ( 775) YES
579 752 303 GOTO DAT390 (1002) PROMPT
580 753 DAT350 1424 ? PT= 1 END OF EXP ?
581 754 267 GOC DAT390 (1002) YES
582 755 1624 ? PT= 0 END OF EXP ?
583 756 247 GOC DAT390 (1002) YES
584 757 DAT360 1414 ?S1=1 EEX HIT ?
585 760 223 GONC DAT390 (1002) NO, WE ARE DONE
586 761 DAT370 1434 PT= 1
587 762 240 SEL P
588 763 460 LDI
589 764 33 CON2 1 11
590 765 1 GOSUB INBYTJ INSERT A EEX
590 766 0
591 767 1170 C=REGN 9
592 770 1366 ? C#0 XS EXP NEGATIVE ?
593 771 1123 GONC DAT330 ( 703) NO
594 772 1 GOSUB INBCHS INSERT A CHS
594 773 0
595 774 1073 GOTO DAT330 ( 703)
596 775 DAT380 240 SEL P
597 776 1 GOSUB INBYT0
597 777 0
598 1000 DAT385 404 S8= 0 NO PROMPT
599 1001 53 GOTO DAT410 (1006)
600 1002 DAT390 240 SEL P
601 1003 1 GOSUB INBYT0 INSERT A NULL AT TAIL
601 1004 0
602 1005 DAT400 410 S8= 1 SAY PROMPT
603 1006 DAT410 1 GOSUB DFILLF
603 1007 0
604 1010 DAT415 1 GOLONG NFRKB
604 1011 2

```

```

605 1012 INBCHS 460 LDI
606 1013      34 CON2 1 12 LOAD A CHS
607 1014 INBYTJ 1634 PT= 0
608 1015      130 G=C
609 1016      1 GOLONG INBYT
609 1017      2

```

NOMAS

Not Manufacturer Supported
recipient agrees NOT to contact manufacturer

* ALPHA ENTRY IN PGM MODE

```

*
613 1020 DAT500 460 LDI
614 1021      177 CON 127
615 1022      1546 ? A#C X IS IT A LAZY T ?
616 1023      1623 GONC DAT400 (1005) YES, IGNORE IT
617 1024      1170 C=REGN 9
618 1025      1506 ? A#0 X IS IT A BACK ARROW ?
619 1026      227 GOC DAT510 (1050) NO
620 1027      1176 C=C-1 S STRING LENGTH -1
621 1030      1376 ? C#0 S ZERO LENGTH NOW ?
622 1031      57 GOC DAT505 (1036) NO
623 1032      1 GOSUB DATOFF RESET DATAENTRY FLAG
623 1033      0
624 1034      1 GOLONG XDELET
624 1035      2
625 1036 DAT505 256 AC EX W
626 1037      1 GOSUB PTBYTA ZERO LAST CHAR
626 1040      0
627 1041      1 GOSUB DECADA POINT BACK ONE CHAR
627 1042      0
628 1043 DAT507 106 C=0 X
629 1044      1160 DADD=C ENABLE CHIP 0
630 1045      256 AC EX W
631 1046      1150 REGN=C 9
632 1047      143 GOTO DAT520 (1063)
633 1050 DAT510 1076 C=C+1 S STRING LENGTH + 1
634 1051      43 GONC DAT515 (1055) STRING LENGTH <= 15
635 1052      1 GOSUB BLINK STRING LENGTH > 15
635 1053      0
636 1054      1343 GOTO DAT415 (1010) IGNORE THIS CHAR
637 1055 DAT515 256 AC EX W
638 1056      1 GOSUB INBYT INSERT THIS CHAR
638 1057      0
639 1060      676 A=A-1 S
640 1061      0 NOP
641 1062      1613 GOTO DAT507 (1043)
642 1063 DAT520 436 A=C S A.S _ STRING LENGTH
643 1064      1 GOSUB GETPC
643 1065      0
644 1066      1 GOSUB INCADA POINT TO 1ST BYTE OF TEXT
644 1067      0
645 1070      276 AC EX S C.S = STRING LENGTH
646 1071      436 A=C S SAVE THE LENGTH IN A.S
647 1072      1374 RCR 13
648 1073      1434 PT= 1
649 1074      1720 LC 15 C(0:1) _ FX
650 1075      1 GOSUB PTBYTA UPDATE STRING LENGTH
650 1076      0
651 1077      576 A=A+1 S LENGTH = 15 ?
652 1100      1007 GOC DAT385 (1000) YES, NO PROMPT
653 1101      1043 GOTO DAT400 (1005)
654 1102 ROLBAK 1170 C=REGN 9 LOAD LCD COUNTER

```

```

655 1103      436 A=C      S      A(13) _ LCD COUNTER
656 1104      236 B=A      S
657 1105      1 GOSUB ENLCD      ENABLE LCD CHIP
657 1106      0
658 1107 ROBK10 1536 ? A#0 S
659 1110      1640 RTN NC
660 1111      676 A=A-1 S
661 1112      1670 FRSABC
662 1113      1743 GOTO ROBK10 (1107)

```

```

*
* OPRMT - OUTPUT A PROMPT CHAR AND LEFT JUSTIFY DISPLAY AND
* AND UPDATE LCD COUNTER.
* THE LCD COUNTER IS IN B(13) WHEN COME IN, IT WILL BE UPDATED
* AND STORED TO REG.9(C13) ON RETURN.
* THE COUNTER IS SET TO 12, EVERY TIME A CHAR SHIFT FROM RIGHT
* END TO DISPLAY THE COUNTER DECREMENT BY ONE.
* ASSUME LCD ENABLE. RETURN WITH CHIP 0 ENABLE.
* USED A(13),B(13), C(13), C(210),N. 1 SUB LEVEL.
*

```

```

673 1114 OPRMT 460 LDI
674 1115      37 CON      Q37
675 1116      1750 SLSABC
676 1117      460 LDI
677 1120      40 CON      Q40
678 1121      336 C=B      S
679 1122      1176 C=C-1 S      LCD FULL ?
680 1123      157 GOC      OPMT20 (1140) YES
681 1124      436 A=C      S
682 1125      1076 C=C+1 S      RESTORE LCD COUNTER
683 1126 OPMT10 1536 ? A#0 S      STRING AT LEFT END ?
684 1127      43 GONC      STOLCC (1133) YES
685 1130      1750 SLSABC
686 1131      676 A=A-1 S
687 1132      1743 GONC      OPMT10 (1126)
688 1133 STOLCC 106 C=0 X
689 1134      1760 PFAD=C      DISABLE LCD CHIP
690 1135      1160 DADD=C      ENABLE SLEEPER CHIP
691 1136      1150 REGN=C 9
692 1137      1740 RTN
693 1140 OPMT20 136 C=0 S
694 1141      1723 GOTO      STOLCC (1133)

```

```

*
* APHST* - INITIALIZE ALPHA ENTRY
* G HAS THE CHAR.
* CALLED BY DATAENTRY AND RETURN TO DATAENTRY
*

```

```

700 1142 APHST* 1670 C=REGN 14
701 1143      1530 ST=C      LOAD SET #
702 1144      14 ?S3=1      PROGRAM MODE ?
703 1145      467 GOC      APHST4 (1213) YES
704 1146      474 RCR      8
705 1147      1730 CST EX
706 1150      1610 S0=      1      SET FLAG 23
707 1151      1730 CST EX
708 1152      574 RCR      6
709 1153      1650 REGN=C 14
710 1154      106 C=0 X
711 1155      1634 PT=      0
712 1156      230 C=G      LOAD THE CHAR
713 1157      406 A=C      X

```

```

714 1160      460 LDI
715 1161      177 CON      127
716 1162      1546 ? A#C X      IS THIS A LAZY "T" ?
717 1163      177 GOC      APHST3 (1202) NO, CLEAR ALPHA REG.
718 1164      570 C=REGH 5
719 1165      1434 PT=      1
720 1166      1352 ? C#0 WPT      ALPHA REG. EMPTY ?
721 1167      77 GOC      APHST1 (1176) NO
722 1170      1334 PT=      13
723 1171      1420 LC      12      SET LCD COUNTER
724 1172      376 BC EX S
725 1173      1 GOSUB      ENLCD
725 1174      0
726 1175      33 GOTO      APHST2 (1200)
727 1176 APHST1      1 GOSUB      ROLBAK
727 1177      0
728 1200 APHST2      1 GOLONG DAT280
728 1201      2
729 1202 APHST3      1 GOSUB      INTARG
729 1203      0
730 1204      1334 PT=      13
731 1205      1420 LC      12
732 1206      376 BC EX S
733 1207      1 GOSUB      CLLCDE
733 1210      0
734 1211      1 GOLONG DAT260
734 1212      2
735 1213 APHST4      1 GOSUB      INSSUB      INCREMENT LINE #
735 1214      0
736 1215      1634 PT=      0
737 1216      230 C=G
738 1217      160 N=C      SAVE THE CHAR IN N TEMP.
739 1220      1 GOSUB      GETPC      LOAD THE PGM COUNTER
739 1221      0
740 1222      36 A=0 S
741 1223      460 LDI
742 1224      361 CON2 15      1      F1 - ONE CHAR TEXT STRING
743 1225      1634 PT=      0
744 1226      130 G=C
745 1227      1 GOSUB      INBYT
745 1230      0
746 1231      260 C=N      LOAD THE CHAR
747 1232      1634 PT=      0
748 1233      130 G=C
749 1234      1 GOSUB      INBYT
749 1235      0
750 1236      256 AC EX W
751 1237      1176 C=C-1 S
752 1240      1150 REGN=C 9      SAVE WORKING PTR IN REG.9
753 1241      1 GOLONG DAT400      EXIT FROM ALPHA ENTRY
753 1242      2

```

```

*
* STBT10 - MOVE SOME STATUS BITS TO SCRATCH AREA (REG.8)
*   DIGIT(0) - 0 : D.P.HIT
*             1 : EEX HIT
*             2 : CHS HIT
*             3 : MANTISSA NONZERO FLAG
*   DIGIT(1) - 4 : DIGIT GROUPING FLAG
*             5 : DECIMAL POINT FLAG
*             6 : ENG

```

```

*          7 : FIX
*    DIGIT(2) - # OF DIGITS
*
766 1243 STBT10 116 C=0    W
767 1244          1160 DADD=C
768 1245          534 PT=   6
769 1246          1420 LC   12
770 1247          134 PT=   4
771 1250          1720 LC   15
772 1251          1420 LC   12
773 1252          416 A=C    W
774 1253          1670 C=REGN 14
775 1254          1660 C=C.A
776 1255          1074 RCR    2
777 1256          406 A=C    X
778 1257          772 C=C+C  N
779 1260          772 C=C+C  M
780 1261          174 RCR    4
781 1262          506 A=A+C  X
782 1263          1070 C=REGN 8
783 1264          674 RCR    11
784 1265          246 C=A    X
784 1266          406
785 1267          1530 ST=C
786 1270          74 RCR    3
787 1271          1050 REGN=C 8
788 1272          1740 RTN

A _ 0000000C0FC000

MOVE NO SEPERATOR & COMMA
TO LOWER TOW BITS IN A DIGIT

*
* DECMPL - DECOMPILE
*
* CALLING SEQUENCE :
*   GOSUB DECMPL
* ASSUME NOTHING. USED A,B,C,N, ST 0-9. 3 SUB LEVELS
* RETURN WITH CHIP 0 ENABLE AND LOAD STATUS SET 0
*   AND R14 IN C (PACH12 IN CNO DEPENDS ON R14 IN C ON RTN)
*
* PACK AND DECOMPILE SHARE COMMON TERMINATION LOGIC
* PACK TERMINATES BY GOING EITHER TO DCPL00 OR TO DCPLRT.
* SINCE PACK CAN EITHER RETURN TO THE CALLING PROGRAM OR EXIT
* VIA ERROR, STATUS BIT S9 IS USED TO CONTROL WHAT TYPE OF
* TERMINATION IS DONE. S9 IS CLEARED AT THE DECMPL ENTRY
* POINT, SO DECOMPILE ALWAYS RETURNS. PACK SETS OR RESETS
* S9 AS NECESSARY BEFORE IT COMES TO THE DCPL00 OR DCPLRT ENTRIES.
*
* S8 AND S3 ARE USED INSIDE DECOMPILE. S8 IS USED TO REMEMBER
* THE STATE OF THE DECOMPILE BIT IN ONE END WHILE TRAVELING UP
* THE LABEL CHAIN TO FIND THE NEXT PREVIOUS END. S3 IS USED TO
* REMEMBER WHETHER ANY PROGRAM HAS BEEN DECOMPILED. IF NO
* PROGRAM HAS BEEN DECOMPILED, THEN DECOMPILE SKIPS AROUND
* THE LOGIC TO ZERO OUT THE SUBROUTINE STACK.
*
813 1273 DCPL17 1346 ? C#0  X      IS THIS AN CHAIN END ?
814 1274          427 GOC    DCPL15 (1336) NO
815 1275 DCPL20 116 C=0
816 1276          160 N=C
817 1277          1 GOSUB   FSTIN      GET REG0
817 1300          0
818 1301          463 GOTO   DCPL24 (1347)

*
820 1302 DECMPL 1104 S9=    0

```

821		ENTRY	DCPL00	
822	1303	1 GOSUB	GTEND	LOAD CHAIN HEAD
822	1304	0		
823	1305	252 AC EX	WPT	SAVE .END. ADDR IN N
824	1306	160 N=C		REM NO PGM BEEN DECMPL YET
825	1307	4 S3=	0	NEXT.END. ADDR IN C(3:0) HERE
826		DCPL05		LOAD END ADDR FROM C
827	1310	412 A=C	WPT	POINT TO 3RD BYTE OF END
828	1311	1 GOSUB	INCAD2	
828	1312	0		
829	1313	1 GOSUB	GTBYTA	GET 3RD BYTE OF END
829	1314	0		
830	1315	1730 CST EX		CHECK IF DECMPL BIT SET
831	1316	1414 ?S1=1		DECMPL BIT SET ?
832	1317	47 GOC	DCPL07 (1323)	YES
833	1320	404 S8=	0	REMEMBER NO DECMPL THIS PGM
834	1321	1730 CST EX		
835	1322	63 GOTO	DCPL11 (1330)	
836	1323	410 S8=	1	SET S8 REMEMBER IT
837	1324	1404 S1=	0	CLEAR DECMPL BIT
838	1325	1730 CST EX		
839	1326	1 GOSUB	PTBYTA	PUT THE BYTE BACK
839	1327	0		
840	1330	260 C=N		
841	1331	416 A=C		
842	1332	1 GOSUB	GTLINK	
842	1333	0		
843	1334	1346 ? C#0	X	CHAIN END ?
844	1335	1403 GONC	DCPL20 (1275)	YES
* MOVES UP SEARCHING FOR END OR FIRST ALBL IN MEM				
846	1336	1 GOSUB	UPLINK	MOVES UP ONE LINK
846	1337	0		
847	1340	1076 C=C+1	S	IS THIS BYTE AN END ?
848	1341	1327 GOC	DCPL17 (1273)	NO, IS AN ALBL
849	1342	252 C=A	WPT	
849	1343	412		
850	1344	160 N=C		SAVE END ADDR IN N
851	1345	1 GOSUB	INCAD2	POINT TO 3RD BYTE OF END
851	1346	0		
852	1347	414 ?S8=1		NEED DECMPL THIS PGM ?
853	1350	353 GONC	DCPL70 (1405)	NO
854	1351	10 S3=	1	REM AT LEAST DECMPL 1 PGM
855	1352	1 GOSUB	GTBYTA	
855	1353	0		
856	1354	1574 RCR	12	
857	1355	642 A=A-1	PT	NEXT BYTE ON SAME REG.?
858	1356	113 GONC	DCPL30 (1367)	YES
859	1357	252 AC EX	WPT	NEXT BYTE IN NEXT REG.
860	1360	1142 C=C-1	PT	
861	1361	1142 C=C-1	PT	
862	1362	1146 C=C-1	X	POINT TO NEXT REG.
863	1363	1160 DADD=C		
864	1364	412 A=C	WPT	
865	1365	70 C=DATA		LOAD NEXT REG.
866	1366	1574 RCR	12	
867	1367	642 A=A-1	PT	POINT TO NEXT BYTE
868	1370	1574 RCR	12	C(3:2) _ NEXT BYTE
869	1371	1202 C=-C	PT	16 COMPLEMENT
870	1372	742 C=C+C	PT	ONE BYTE ?
871	1373	1627 GOC	DCPL25 (1355)	YES, GO ON TO NEXT LINE

```

872 1374      742 C=C+C PT          THREE BYTE LINE ?
873 1375      357 GOC   DCPL40 (1432) NO, ITS TWO BYTES
874 1376      742 C=C+C PT          ROW 15 ?
875 1377      567 GOC   DCPL45 (1455) NO, ITS ROW 13 OR 14
876 1400      742 C=C+C PT          TEST ROW 0 OR ROW 15
877 1401      1543 GONC  DCPL25 (1355) ROW 0 IF NO CARRY
* TEXT ROW, LET THE "NXLIN" ROUTINE HANDLE IT
879 1402 DCPL35      1 GOSUB  NXLTX
879 1403          0
880 1404          1513 GOTO   DCPL25 (1355) GO ON TO NEXT LINE
*
882 1405 DCPL70      260 C=N          GET STARTING ADDR
883 1406          1346 ? C#0 X        JUST FINISHING 1ST PGM ?
884 1407          1017 GOC   DCPL05 (1310) NO, KEEP GOING
885 1410          1160 DADD=C
886 1411          14 ?S3=1            HAS ANY PGM BEEN DECHPL ?
887 1412          113 GONC  DCPL60 (1423) NO, DON'T CLEAR SUB STACK
888          ENTRY  DCPLRT
889 1413 DCPLRT      1 GOSUB  GETPC    CLEAR THE SUBROUTINE STACK
889 1414          0
890          ENTRY  DCRT10
891 1415 DCRT10      116 C=0
892 1416          1160 DADD=C
893 1417          1350 REGN=C 11
894 1420          1450 REGN=C 12
895 1421          1 GOSUB  PUTPC
895 1422          0
896 1423 DCPL60      1670 C=REGN 14    PUT UP SS0
897 1424          1530 ST=C
898 1425          1114 ?S9=1
899 1426          1640 RTN NC
900          ENTRY  ERRTA
901 1427 ERRTA      1 GOSUB  ERROR
901 1430          0
902 1431          0 XDEF  MSGTA
*
* TWO BYTES ROW
905 1432 DCPL40      742 C=C+C PT          ROW 9 OR 10 ?
906 1433          177 GOC   DCPL50 (1452) YES, SIMPLY SKIP 1 BYTE
907 1434          1342 ? C#0 PT          ROW 12 ?
908 1435          427 GOC   DCPL55 (1477) NO, ITS ROW 11
909 1436          1066 C=C+1 XS        IS A LBL.NN ?
910 1437          137 GOC   DCPL50 (1452) YES, SKIP 1 BYTE
911 1440          1066 C=C+1 XS        IS A X<>.NN ?
912 1441          117 GOC   DCPL50 (1452) YES
* ALBL OR END HERE
914 1442 DCPL42      1 GOSUB  INCADA    SKIP OVER THE LINK
914 1443          0
915 1444          1 GOSUB  NXBYTA    LOAD THE THIRD BYTE
915 1445          0
916 1446          1574 RCR   12        MOVE IT TO C(3:2)
917 1447          1042 C=C+1 PT        IS IT AN ALBL ?
918 1450          1327 GOC   DCPL35 (1402) YES, TEXT STRING FOLLOWS
*
920 1451          1343 GOTO   DCPL70 (1405) GOTO TAKE CARE OF END
921 1452 DCPL50      1 GOSUB  NXL3B2
921 1453          0
922 1454 DCPL51      1013 GOTO   DCPL25 (1355)
* GTO.NN & XEQ.NN HERE (CLEAR 3 DIGITS LINK)
924 1455 DCPL45      1 GOSUB  GTBYTA    GET THE FIRST BYTE AGAIN

```

```

924 1456      0
925 1457    1574 RCR      12      MOVE IT TO C(3:2)
926 1460      126 C=0      XS      ZERO FIRST DIGIT OF LINK
927 1461    1074 RCR      2
928 1462      1 GOSUB    PTBYTA    PUT IT BACK TO MEM
928 1463      0
929 1464      1 GOSUB    NXBYTA    GET NEXT BYTE
929 1465      0
930 1466    1434 PT=      1
931 1467      112 C=0      WPT      ZERO LAST TWO DIGITS OF LINK
932 1470      356 BC EX    W      SAVE THE REG. IN B
933 1471      316 C=B      W
934 1472      1 GOSUB    PTBYTA    PUT BYTE
934 1473      0
935 1474      316 C=B
936 1475    1574 RCR      12
937 1476    1543 GOTO    DCPL50 (1452) INCREMENT 1 BYTE
* GTO.0-14 HERE ( 1 BYTE LINK)
939 1477 DCPL55      1 GOSUB    NXBYTA    GET THE LINK BYTE
939 1500      0
940 1501    1434 PT=      1
941 1502      112 C=0      WPT
942 1503      356 BC EX    W      SAVE THE REG. IN B
943 1504      316 C=B      W
944 1505      1 GOSUB    PTBYTA
944 1506      0
945 1507      316 C=B      W
946 1510    1574 RCR      12
947 1511    1433 GOTO    DCPL51 (1454)

*
* XECROM - DISPLAY ROM FUNCTION
* CALLED FROM DFILLF WHEN IT HAS A XECROM FUNCTION.
* CALLED WITH A,X HAS THE 1ST BYTE OF THE 2 BYTE'S FC, PT=1.
*
953 1512 XECROM    246 AC EX    X
954 1513      1374 RCR      13
955 1514      130 G=C
956 1515      156 AB EX    W      SAVE 1ST BYTE IN G
957 1516      1 GOSUB    NXTBYT    GET THE SECOND BYTE
957 1517      0
958 1520      1034 PT=      2
959 1521      230 C=G
960 1522      1 GOSUB    GTRMAD    PUT 2 BYTES TOGETHER IN C(3:0)
960 1523      0      FIND IT IN THE ROM
961 1524      303 GOTO    XROMNF (1554) ROM NOT PLUGED IN
962 1525      256 AC EX    W
963 1526      674 RCR      11      C.M _ XADR
964 1527      1 GOSUB    ENLCD
964 1530      0
965 1531      14 ?S3=1      XTYPE=0 ?
966 1532      163 GONC      XROM10 (1550) YES, MICRO CODE FUNCTION
967 1533      1072 C=C+1    M
968 1534      1072 C=C+1    M      POINT TO THIRD BYTE OF ALBL
969 1535      1460 CXISA    GET 1ST BYTE OF TEXT STRING
970 1536      406 A=C      X
971 1537      646 A=A-1    X
972 1540      74 RCR      3      C(3:0) _ 1ST BYTE ADDR
973 1541      1056 C=C+1
974 1542      356 BC EX    W      SAVE 1ST BYTE ADDR IN B
975 1543      1 GOSUB    OUTROM    SEND "XROM" TO LCD

```



```

975 1544          0
976 1545        404 S8= 0          CLEAR S8 FOR TXRW10
977 1546          1 GOLONG TXTROM  DISPLAY TEXT STRING FROM ROM 327
977 1547          2
978          XROM10
979 1550          1 GOSUB  PROMF2
979 1551          0
980 1552 XROMRT   1 GOLONG DF150
980 1553          2

```

*
* ROM NOT PLUG IN, DISPLAY ROM ID & FC #
*

```

984 1554 XROMHF   1 GOSUB  ENLCD
984 1555          0
985 1556          1 GOSUB  OUTROM  SEND "XROM" TO LCD
985 1557          0
986 1560        316 C=B          GET ROM ID
987 1561          74 RCR      3
988 1562        406 A=C      X
989 1563          36 A=0      S
990 1564          1 GOSUB  GENNUM
990 1565          0
991 1566        1670 FRSABC
992 1567        1434 PT=      1
993 1570        1720 LC      15
994 1571        1750 SLSABC
995 1572          146 AB EX  X    GET FUNCTION #
996 1573          36 A=0      S
997 1574          1 GOSUB  GENNUM
997 1575          0
998 1576        1543 GOTO   XROMRT (1552)
999

```

```

1000
1001
1002
1003
1004
1005
1006
1007
1008
1009

```

```

ENTRY  SRBMAP
ENTRY  TBITMP
ENTRY  TBITMA
ENTRY  XROM

```

* TBITMP - TEST BIT MAP

*- TEST THE CORRECT BIT MAP (SHIFTED/UNSHIFTED) TO
*- DETERMINE WHETHER A PARTICULAR KEY HAS BEEN
*- ASSIGNED OR NOT

*- IN: A[2:1]= LOGICAL KEYCODE (0:79 FORM)
*- CHIP 0 SELECTED

*- OUT: C=0 IMPLIES BIT NOT SET

*- C#0 IMPLIES BIT SET

*- M= BIT MAP

*- CHIP 0 & APPROPRIATE REGISTER IS SELECTED

*- USES: C[13:0], A[13:0], M[13:0]

*

* TBITMA ENTRY - SAME AS TBITMP EXCEPT KC IS IN A[1:0] AND IS

* IN 1-30 FORM ON ENTRY

```
1024
```

```
1025
```

```
1026
```

```
1027 1577 TBITMA  646 A=A-1  X
```

DECREMENT K.C.

```

1028 1600      1756 A SL      AI21_COL
1029 1601 TBITMP 1034 PT=    2      CI21_4
1030 1602      420 LC      4      -
1031 1603      1604 S0=    0      S0_SHIFTSET
1032 1604      1020 LC      8      -
1033 1605      1434 PT=    1      -
1034 1606      1102 C=A-C PT      -
1035 1607      37 GOC      **3      (1612) -
1036 1610      402 A=C      PT      -
1037 1611      1610 S0=    1      -
1038 1612      234 PT=    5      POSITION PTR AT COLUMN
1039 1613      33 GOTO      **3      (1616) -
1040 1614      1734 INC PT      -
1041 1615      1734 INC PT      -
1042 1616      666 A=A-1 XS      -
1043 1617      1753 GONC      *-3      (1614) -
1044 1620      1746 A SL      X      AI21_ROW
1045 1621      1426 ? A<C XS      ROW<4?
1046 1622      37 GOC      **3      (1625) YES
1047 1623      1734 INC PT      SET PTR
1048 1624      726 A=A-C XS      -
1049 1625      116 C=0      POSITON ROW,COL BIT
1050 1626      1624 ? PT=    0      - (TOP ROW KEYS?)
1051 1627      1540 RTN C      - (YES)
1052 1630      1042 C=C+1 PT      -
1053      LEGAL
1054 1631      23 GOTO      **2      (1633) - (YES)
1055 1632      742 C=C+C PT      -
1056 1633      666 A=A-1 XS      -
1057 1634      1763 GONC      *-2      (1632) -
1058 1635      416 A=C      A_MASK
1059 1636      1770 C=REGN 15      -
1060 1637      1614 ?S0=1      SHIFTSET?
1061 1640      27 GOC      **2      (1642) NOPE
1062 1641      1270 C=REGN 10      -
1063 1642      530 M=C      M_C_BIT MAP
1064 1643      1660 C=C.A      ROW,COL BIT SET?
1065 1644      1740 RTN      -
1066
1067
1068
1069
* SRBMAP - SET/RESET BIT MAP
*- TOGGLE THE BIT DESIGNATED BY THE MASK FOUND IN
*- REGISTER C.
*- IN:  CC[13:0]= BIT MAP MASK (RESULT OF TBITMP)
*-      MC[13:0]= BIT MAP
*-      THE APPROPRIATE REGISTER MUST BE SELECTED
*- OUT: CHIP 0 SELECTED
*- USES: CC[13:0], MC[13:0], AI[13:0]
1078
1079
1080
1081 1645 SRBMAP 1356 ? C#0      SET?
1082 1646      47 GOC      SRBN10 (1652) YES,RESET
1083 1647      630 C=M      -
1084 1650      1002 C=A+C PT      SET IT
1085      LEGAL
1086 1651      43 GOTO      **4      (1655) -
1087 1652 SRBN10 630 C=M      -

```

1088 1653
1089 1654
1090 1655
1091 1656
1092

256 AC EX
1116 C=A-C
1360 DATA=C
1740 RTN
EJECT

-
-
RESTORE BIT MAP
RETURN

329

```

* XROM - EXECUTE ROM FUNCTION
*- LOCATES ROM FUNCTION AND PREPARES IT FOR EXECUTION.
*- IF THE FUNCTION IS A USER LANGUAGE PROGRAM, A TRANSFER
*- IS MADE TO THE XEGC PROGRAM SEGMENT. IF THE FUNCTION
*- IS MICRO CODED, A JUMP IS MADE DIRECTLY TO THE FUNCTION'S
*- EXECUTION POINT.
* IN:  FIRST BYTE OF FC IS IN G
*      SECOND BYTE OF FC IS IN ST AND IN C[1:0]
*      PT=2
*      NUMERIC ARGUMENT, IF ANY, IS IN B.X
*      ALPHA ARGUMENT, IF ANY, IS IN REG 9
*- OUT: FOR MICROCODE FCNS, SS0 UP, NUMERIC ARG IN A.X, ALPHA ARG IN
*      REG 9, NFRPU ON THE STACK
*      FOR USER LANGUAGE FCNS, CURRENT ADDR SAVED IN R10[3:0],
*      NEW ADDR IN C[3:0], EXITS TO XGI57
*- USES: 1 SUBROUTINE LEVEL
1109
1110 1657 XROM      316 C=B          SAVE NUMERIC ARGUMENT
1111 1660          160 N=C          IN N
1112 1661          230 C=C          RESTORE FC TO
1113 1662          1630 C=ST        C[3:0]
1114 1663          1 GOSUB  GTRMAD
1114 1664          0
1115 1665          173 GOTO  XRM20 (1704) COULDN'T FIND IT
* GTRMAD RETURNS TO P+2 WITH FOUND ADDRESS IN A[3:0]
1117 1666          14 ?S3=1        USER LANGUAGE?
1118 1667          77 GOC  XRM10 (1676) YES
1119          MICROCODE FCN
1120 1670          1670 C=REGN 14   PUT UP SS0
1121 1671          1530 ST=C
1122 1672          260 C=N          RETRIEVE NUMERIC ARG TO A.X
1123 1673          256 AC EX
1124 1674          674 RCR  11
1125 1675          740 GOTOC
1126
1127          XRM10          USER LANGUAGE FCN
1128 1676          156 AB EX        SAVE NEW ADDR IN B
1129 1677          1 GOSUB  SAVRTN  SAVE OLD ADDR IN R10
1129 1700          0
1130 1701          316 C=B          PUT NEW ADDR IN C[3:0]
1131 1702          1 GOLONG XGI57
1131 1703          2
1132
1133 1704 XRM20      1 GOLONG ERNE  REPORT ERROR
1133 1705          2
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144

```

*TXTLBL-TEXT OF LABEL STRING

*THIS IS THE FRONT END FOR TEXT LBL

NOMAS
 Not Manufacturer Supported
 recipient agrees NOT to contact manufacturer

*GIVEN A PC IN A(0,3) POINTING AT THE FIRST BYTE
 *OF AN ALPHA LBL THIS ROUTINE DISPLAYS THE ALPHA TEXT
 *STRING.

*FOR ROM S2=1. FOR RAM S2=0.

*SETS STATUS FOR NO PROMT AND LCD NOT FULL.

*
 * TXTLB1 - SAME AS TXTLBL EXCEPT CLEARS S4 ON ENTRY.
 * S4 IS USED TO DECIDE WHETHER TO CLEAR THE DISPLAY
 * BEFORE PUTTING UP THE TEXTSTRING. S4=0 IMPLIES
 * CLEAR THE DISPLAY, S4=1 IMPLIES DON'T CLEAR FIRST.

 1159 1706 TXTLB1 104 S4= 0 REMEMBER TO CLEAR DISPLAY
 1160 1707 TXTLBL 404 S8= 0 NO PROMPT
 1161 1710 1404 S1= 0 LCD NOT FULL
 1162 1711 1014 ?S2=1 ROM OR RAM?
 1163 1712 107 GDC ROMSTG (1722) ROM
 1164 1713 1 GOSUB INCADP SET PT=3 INC ADR
 1164 1714 0
 1165 1715 1 GOSUB NXBYTA GET # CHR
 1165 1716 0
 1166 1717 1 GOSUB INCADA SKIP ASSIGN BIT
 1166 1720 0
 1167 1721 53 GOTO CLRL (1726)
 1168 1722 ROMSTG 556 A=A+1 INC ADR
 1169 LEGAL
 1170 1723 1 GOSUB NXBYTO GET # CHRT
 1170 1724 0
 1171 1725 556 A=A+1
 1172 1726 CLRL 1146 C=C-1 X
 1173 1727 346 BC EX X
 1174 1730 156 AB EX COUNT IN A,ADR IN B
 1175 1731 1 GOSUB ENLCD
 1175 1732 0
 1176 1733 114 ?S4=1 SKIP CLEARING THE DISPLAY?
 1177 1734 1 GSUBNO:CLRLCD NO. CLEAR THE DISPLAY
 1177 1735 0
 1178 1736 1 GOLONG:TXTSTR
 1178 1737 2
 1179
 1180 ENTRY STBT30
 1181 ENTRY STBT31
 1182 1740 STBT30 414 ?S8=1 LAST PGM NEEDS PACK ?
 1183 1741 43 GONC STBT31 (1745) NO
 1184 1742 356 BC EX W
 1185 1743 530 M=C
 1186 1744 356 BC EX W
 1187 1745 STBT31 404 S8= 0
 1188 1746 1730 CST EX
 1189 1747 1014 ?S2=1
 1190 1750 43 GONC STBT32 (1754)
 1191 1751 410 S8= 1
 1192 1752 1004 S2= 0
 1193 1753 1410 S1= 1
 1194 1754 STBT32 1730 CST EX
 1195 1755 1740 RTH

OUTROM - SHIFT "XROM " INTO THE LCD FROM THE RIGHT END

FOR ENTRY, LCD MUST BE ENABLED
 USES C16:01 AND ONE ADDITIONAL SUBROUTINE LEVEL

```

*
1202          ENTRY  OUTROM
1203 1756 OUTROM  1 GOSUB  MESSL
1203 1757          0
1204 1760          30 CON  24          X
1205 1761          22 CON  18          R
1206 1762          17 CON  15          0
1207 1763          15 CON  13          H
1208 1764          1040 CON  01040      BLANK
1209 1765          1740 RTN
* RESERVE 2 WORDS AT THE END OF CH11 FOR THE CHIP 2 CHECKSUM AND
* TRAILER.
1212
1213          FILLTO 01775
          1766          0000 NOP
          1767          0000 NOP
          1770          0000 NOP
          1771          0000 NOP
          1772          0000 NOP
          1773          0000 NOP
          1774          0000 NOP
          1775          0000 NOP
1214 1776 REVLV2  6 CON  6          REV LEVEL= F
1215 1777 CKSUM2  0 CON  00000
1216          END

```

```

ERRORS :      0

```

SYMBOL TABLE

AOUT05	30	-	42			
AOUT10	46	-	34			
AOUT15	53	-	126			
AOUT16	72	-	66			
AOUT17	73	-	76			
AOUT18	104	-	61			
AOUT19	111	-	105			
AOUT20	117	-	52			
AOUTR0	62	-	45			
AOUTRT	77	-	74	71		
APHST*	1142	-				
APHST1	1176	-	1167			
APHST2	1200	-	1175			
APHST3	1202	-	1163			
APHST4	1213	-	1145			
APNDHW	424	-				
APPEND	416	-				
ARGOUT	20	-				
ASCLCD	135	-				
ASCTBL	0	-				
CKSUM2	1777	-				
CLLCDE	360	-				
CLRL	1726	-	1721			
CLRLCD	366	-				
COLON	127	-	142			
COMMA	132	-	146			
DAT102	502	-				
DAT105	507	-	503			
DAT106	514	-	510			
DAT110	517	-	477			
DAT120	526	-	522			
DAT140	535	-	501			
DAT200	542	-	475			
DAT220	606	-	632	566	541	
DAT230	552	-				
DAT231	567	-				
DAT235	576	-	551			
DAT240	610	-	563			
DAT245	622	-	614			
DAT260	624	-	621			
DAT280	630	-				
DAT300	633	-	547			
DAT320	642	-				
DAT322	663	-	656			
DAT325	671	-	666			
DAT330	703	-	774	771	741	715
DAT333	716	-	677			
DAT335	720	-	707	705		
DAT345	742	-	730			
DAT350	753	-	726			
DAT360	757	-				
DAT370	761	-	743			
DAT380	775	-	751	747	745	723
DAT385	1000	-	1100			
DAT390	1002	-	760	756	754	752
DAT400	1005	-	1101	1023		

DAT410	1006	-	1001	
DAT415	1010	-	1054	
DAT500	1020	-		
DAT505	1036	-	1031	
DAT507	1043	-	1062	
DAT510	1050	-	1026	
DAT515	1055	-	1051	
DAT520	1063	-	1047	
DATENT	454	-		
DCPL00	1303	-		
DCPL05	1310	-	1407	
DCPL07	1323	-	1317	
DCPL11	1330	-	1322	
DCPL15	1336	-	1274	
DCPL17	1273	-	1341	
DCPL20	1275	-	1335	
DCPL24	1347	-	1301	
DCPL25	1355	-	1454 1404 1401 1373	
DCPL30	1367	-	1356	
DCPL35	1402	-	1450	
DCPL40	1432	-	1375	
DCPL42	1442	-		
DCPL45	1455	-	1377	
DCPL50	1452	-	1476 1441 1437 1433	
DCPL51	1454	-	1511	
DCPL55	1477	-	1435	
DCPL60	1423	-	1412	
DCPL70	1405	-	1451 1350	
DCPLRT	1413	-		
DCRT10	1415	-		
DECMPL	1302	-		
ERRTA	1427	-		
INBCHS	1012	-		
INBYTJ	1014	-		
MASK	210	-	152	
MASK10	236	-	220 214	
MASK20	243	-	247	
MASK30	253	-	245	
MASKRT	225	-	256 252	
NBYTAB	404	-		
NBYTAB	406	-		
NXBYTO	413	-		
NXTBYT	407	-		
OPMT10	1126	-	1132	
OPMT20	1140	-	1123	
OPRONT	1114	-		
OUTLCD	200	-	235 172	
OUTROM	1756	-		
PERIOD	153	-		
PUNC	155	-	134 131	
PUNC10	174	-	163 161	
PUNC20	205	-	201 173	
REVLV2	1776	-		
ROBK10	1107	-	1113	
ROLBAK	1102	-		
RONSTG	1722	-	1712	
SCROL0	336	-		
SCROL2	350	-	341	
SCROL5	352	-	344	
SCROLL	334	-		

SRBM10	1652	-	1646
SRBNAP	1645	-	
SROL10	355	-	
STBT10	1243	-	
STBT30	1740	-	
STBT31	1745	-	1741
STBT32	1754	-	1750
STOLCC	1133	-	1141 1127
TBITMA	1577	-	
TBITMP	1601	-	
TEXT	257	-	
TEXT30	311	-	333
TEXT40	320	-	
TXTLB1	1706	-	
TXTLBL	1707	-	
XECROM	1512	-	
XRM10	1676	-	1667
XRM20	1704	-	1665
XROM	1657	-	
XROM10	1550	-	1532
XROMNF	1554	-	1524
XROMRT	1552	-	1576

ENTRY TABLE

AOUT15	53	-
APHST*	1142	-
APHDNW	424	-
APPEND	416	-
ARGOUT	20	-
ASCLCD	135	-
CLLCDE	360	-
CLRLCD	366	-
DAT106	514	-
DAT231	567	-
DAT260	624	-
DAT280	630	-
DAT300	633	-
DAT320	642	-
DAT400	1005	-
DAT500	1020	-
DATENT	454	-
DCPL00	1303	-
DCPLRT	1413	-
DCRT10	1415	-
DECNPL	1302	-
ERRTA	1427	-
INBCHS	1012	-
INBYTJ	1014	-
MASK	210	-
NBYTA0	404	-
NBYTAB	406	-
NXBYT0	413	-
NXTBYT	407	-
OPRDMT	1114	-
OUTLCD	200	-
OUTROM	1756	-
ROLBAK	1102	-
SCROLO	336	-
SCROLL	334	-
SRENAP	1645	-
STBT10	1243	-
STBT30	1740	-
STBT31	1745	-
STOLCC	1133	-
TBITMA	1577	-
TBITMP	1601	-
TEXT	257	-
TXTLB1	1706	-
TXTLBL	1707	-
XECROM	1512	-
XROM	1657	-
XROMNF	1554	-

EXTERNAL REFERENCES

APHST*	533					
APHST*	534					
APNDNW	331					
APNDNW	332					
APPEND	610					
APPEND	611					
ASCLCD	124	626				
ASCLCD	125	627				
BAKAPH	560					
BAKAPH	561					
BLINK	564	1052				
BLINK	565	1053				
CLLCDE	43	47	1207			
CLLCDE	44	50	1210			
CLRLCD	1734					
CLRLCD	1735					
DAT260	1211					
DAT260	1212					
DAT280	1200					
DAT280	1201					
DAT400	1241					
DAT400	1242					
DAT500	634					
DAT500	635					
DATOFF	1032					
DATOFF	1033					
DECADA	1041					
DECADA	1042					
DELLIN	640					
DELLIN	641					
DF150	1552					
DF150	1553					
DFILLF	1006					
DFILLF	1007					
DGENS8	576	644				
DGENS8	577	645				
DIGST*	531					
DIGST*	532					
ENCP00	111	404				
ENCP00	112	405				
ENLCD	120	616	1105	1173	1527	1554 1731
ENLCD	121	617	1106	1174	1530	1555 1732
ERRNE	1704					
ERRNE	1705					
ERROR	1427					
ERROR	1430					
FSTIN	1277					
FSTIN	1300					
GENNUM	1564	1574				
GENNUM	1565	1575				
GETPC	260	636	646	1064	1220	1413
GETPC	261	637	647	1065	1221	1414
GTBYT	262					
GTBYT	263					
GTBYTA	1313	1352	1455			
GTBYTA	1314	1353	1456			

GTBYT0	414							
GTBYT0	415							
GTFEND	1303							
GTFEND	1304							
GTLINK	1332							
GTLINK	1333							
GTRMAD	1522	1663						
GTRMAD	1523	1664						
INBCHS	667	772						
INBCHS	670	773						
INBYT	737	1016	1056	1227	1234			
INBYT	740	1017	1057	1230	1235			
INBYT0	660	776	1003					
INBYT0	661	777	1004					
INBYTJ	713	765						
INBYTJ	714	766						
INCAD2	1311	1345						
INCAD2	1312	1346						
INCADA	1066	1442	1717					
INCADA	1067	1443	1720					
INCADP	1713							
INCADP	1714							
INSSUB	1213							
INSSUB	1214							
INTARG	307	1202						
INTARG	310	1203						
MESSL	1756							
MESSL	1757							
MSGTA	1431							
NFRKB	606	1010						
NFRKB	607	1011						
NOREG9	600							
NOREG9	601							
NXBYTA	30	114	411	652	1444	1464	1477	1715
NXBYTA	31	115	412	653	1445	1465	1500	1716
NXBYT0	1723							
NXBYT0	1724							
NXL3B2	1452							
NXL3B2	1453							
NXLTX	1402							
NXLTX	1403							
NXTBYT	273	322	1516					
NXTBYT	274	323	1517					
OFSHFT	455							
OFSHFT	456							
OPRONT	630							
OPRONT	631							
OUTROM	1543	1556						
OUTRON	1544	1557						
PRONF2	1550							
PRONF2	1551							
PTBYTA	1037	1075	1326	1462	1472	1505		
PTBYTA	1040	1076	1327	1463	1473	1506		
PUTPC	316	1421						
PUTPC	317	1422						
RFDS55	604							
RFDS55	605							
RG9LCD	602							
RG9LCD	603							
ROLBAK	622	1176						

RULBAK	623	1177
RST05	346	
RST05	347	
SAVRTN	1677	
SAVRTN	1700	
SCROLL	107	
SCROLL	110	
STFLGS	526	
STFLGS	527	
STQLCC	102	
STQLCC	103	
TOHE7X	422	
TOHE7X	423	
TXTRON	1546	
TXTRON	1547	
TXSTR	1736	
TXSTR	1737	
UPLINK	1336	
UPLINK	1337	
XDELET	1034	
XDELET	1035	
XGIS7	1702	
XGIS7	1703	

339

End of VASM assembly

NOMAS
NOT Manufacturer Supported
recipient agrees NOT to contact manufacturer

Scan Copyright ©
The Museum of HP Calculators
www.hpmuseum.org

Original content used with permission.

Thank you for supporting the Museum of HP
Calculators by purchasing this Scan!

Please to not make copies of this scan or
make it available on file sharing services.