

This image shows a microfiche card containing 100 individual pages of program code. The pages are arranged in a 10x10 grid. Each page contains a page number in the top left corner and a block of code or data. The code is printed in a monospaced font, typical of early computer systems. The pages contain various types of code, including what appears to be assembly language or low-level program instructions, with some lines starting with labels like 'LDA', 'STA', and 'MOV'. The overall appearance is that of a dense, organized collection of program fragments.

PDP11

PROGRAM GENERATOR
MD-11-DTMGA-C

EP-DTMGA-C-DL-A
COPYRIGHT © 1976
FICHE 2 OF 2

NOV 1976
digital
MADE IN USA

This microfiche card contains a grid of frames. The frames are arranged in approximately 12 rows and 3 columns. Each frame contains a small, high-contrast image of data, likely representing a portion of a program or a specific data set. The data appears to be organized in a structured format, possibly a table or a list of values. The frames are separated by thin white lines, and the overall layout is typical of a microfiche card used for data storage and retrieval.

.SBTTL REVISION HISTORY

001

- .IF DF MM
.IFF
- JUL 76 DTMGA-C RELEASE
- MAY 76 ADDED THE "PATB" PATTERN FORMAT.
- MAY 76 CORRECTED ANOTHER COMPILER BUG WITH THE "BREAK V"
AND "BREAK V ON V" INSTRUCTIONS.
- MAY 76 CHANGED SIGNED BRANCH INSTRUCTIONS TO UNSIGNED ON
THE BINARY CODE GENERATED FOR ALL RELATIVE "IF"
INSTRUCTIONS.
- MAY 76 CORRECTED COMPILER BUG WHICH OCCURRED WHEN AN OCTAL
OR DECIMAL ADDRESS WAS ENTERED ON AN I/O INSTRUCTION.
- APR 76 DTMGA-B RELEASE
- FEB 76 ADDED THE "WORD" INSTRUCTION.
- JAN 76 ADDED A TEST FOR THE CACHE MEMORY HIT/MISS REGISTER.
IF FOUND, THE DEVICE ROUTINES' TIME OUT VALUE
WILL BE EFFECTIVELY MULTIPLIED BY 2.
- JAN 76 IF AN I/O TIMEOUT ERROR OCCURS, IT WILL NOW
BE TREATED THE SAME AS OTHER DEVICE ERRORS. IT
IS NO LONGER REQUIRED TO RESTART THE PROGRAM WITH
THE RUN COMMAND.
- DEC 75 ADDED THE ABILITY TO PROCESS VARIABLE LENGTH
FILENAME ON USER PROGRAMS.
- DEC 75 ADDED SEVERAL SECTIONS OF CODE THAT PERFORMS
EQUIVALENT FUNCTIONS THAT WERE REQUIRED FOR THE
MEMORY MANAGEMENT VERSION.
- DEC 75 BUS ADDRESSES PASSED TO DEVICE ROUTINES FOR NPR
DEVICES WILL NOW BE TWO WORDS. THIS PROVIDES
DEVICE ROUTINE COMPATIBILITY BETWEEN THE DIFFERENT
VERSIONS OF MPG.
- NOV 75 INITIAL VECTOR HOUSEKEEPING IS NOW PERFORMED BY
MPG'S ONE TIME HOUSEKEEPING ROUTINE.
- NOV 75 ADDED THE "PATTERN" FORMAT TO THE 'FILL' COMMAND.
- NOV 75 ADDED THE "FAST" OPTION TO THE /LIST COMMAND.
- NOV 75 ADDED A TEST FOR THE LSI-11 AND ALTERED ALL PSW
ACCESSES FOR 11 FAMILY COMPATIBILITY. IF AN LSI -11,
ALL BUS REQUEST PRIORITIES IN DEVICE ROUTINES WILL
BE SET TO 4 WHEN THEY ARE LOADED.

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152

153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183

000

NOV 75 REMOVED THE VALID DEVICE TABLE FROM MPG AND ADDED
CODE THAT WILL LOAD IT FROM A SEPARATE FILE INTO
MPG'S AREA.

OCT 75 ADDED CODE THAT TRANSFERS CONTROL TO A DEVICE ROUTINE
IMMEDIATELY AFTER LOADING IT FOR INITIAL HC'USE-
KEEPING AND TAILORING.

OCT 75 CORRECTED THE BUG WITH THE "BREAK" INSTRUCTION.

OCT 75 ADDED DISPLAY OF THE LOADED DEVICE ROUTINE'S
FILENAME.

AUG 75 DTMGA-A INITIAL RELEASE
.IFT

JUL 76 DTMMA-B RELEASE

MAY 76 ADDED THE "PATB" PATTERN FORMAT.

MAY 76 CORRECTED ANOTHER COMPILER BUG WITH THE "BREAK V"
AND "BREAK V ON V" INSTRUCTIONS.

MAY 76 CHANGED SIGNED BRANCH INSTRUCTIONS TO UNSIGNED ON
THE BINARY CODE GENERATED FOR ALL RELATIVE "IF"
INSTRUCTIONS.

MAY 76 CORRECTED COMPILER BUG WHICH OCCURRED WHEN AN OCTAL
OR DECIMAL ADDRESS WAS ENTERED ON AN I/O INSTRUCTION.

APR 76 DTMMA-A INITIAL RELEASE
.ENDC

185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240

001

000
000000'

001

000

.SBTTL BUILD INFORMATION AND EQUATES
.IF DF MM
.IFF
.TITLE MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
.IFT
.TITLE MAINDEC-11-DTMMA-B MPG CONTROL ROUTINE
.ENDC

.CSECT MPG
.DSABL GBL

.IF DF MM
.IFF
;REVISION "C"

;FILENAME OF "TMGACO.MPG" ON MPG/XXDP MEDIA

.IFT
;REVISION "B"

;FILENAME OF "TMMABO.MPG" ON MPG/XXDP MEDIA

.ENDC

MACY11:

WITHOUT MEM MGMNT: DTMGA?,DTMGA?/CRF:SYM/DOC=DTMGA?.P11
WITH MEM MGMNT: DTMMA?,DTMMA?/CRF:SYM/EG:MM/DOC=DTMGA?.P11

NOTES: "/EQ:MLIST" WILL SUPPRESS DISPLAY OF ALL
UNUSED CONDITIONAL CODE IN THE LISTING.

"/EQ:MIMIC" WILL GENERATE A VERSION
CAPABLE OF RUNNING UNDER MIMIC.

LNKX11:

DTMGA?.MPG/B:0+DTMGA?/E
DTMMA?.MPG/B:0+DTMMA?/E

FOR PAPER TAPE OUTPUT:

PUNCH DTMGA?.MPG/FILE:ELEV
PUNCH DTMMA?.MPG/FILE:ELEV

:NOTE: ANY STATEMENTS, WHOSE COMMENT FIELD BEGINS
: WITH :*; ARE STORAGE AREAS WHOSE CONTENTS
: WILL BE DIFFERENT BETWEEN THE TWO VERSIONS
: OF MPG. ON THE MEMORY MANAGEMENT VERSION, THEIR

241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007
001200

001
002

001
000
001

172516

000
001

: CONTENTS WILL BE THE HIGH 16 BITS OF 22 BIT
: ABSOLUTE ADDRESSES. ON THE NON-MEMORY MANAGEMENT
: VERSION, THEIR CONTENTS WILL BE 16 BIT ABSOLUTE
: ADDRESSES.

: RD= %0
: R1= %1
: R2= %2
: R3= %3
: R4= %4
: R5= %5
: R6= %6
: R7= %7
: SP= %6
: PC= %7

STACK= 1200

.IF DF MM ;MEMORY MANAGEMENT EQUATES
MMMR0= 177572
.IF DF MIMIC & UMTEST
.IFF
MMMR1= 177574
.IFT
MMMR1= 574
.ENDC
MMMR2= 177576
.ENDC
.IF DF MIMIC & UMTEST
.IFF
MMMR3= 172516
.IFT
MMMR3= 576
.ENDC
.IF DF MM

KPAR0= 172340 ;KERNEL INST SPACE ADR REG'S
KPAR1= 172342
KPAR2= 172344
KPAR3= 172346
KPAR4= 172350
KPAR5= 172352
KPAR6= 172354
KPAR7= 172356

KPDR0= 172300 ;KERNEL INST SPACE DESCR. REG'S
KPDR1= 172302
KPDR2= 172304
KPDR3= 172306
KPDR4= 172310
KPDR5= 172312
KPDR6= 172314
KPDR7= 172316

UPARO= 177640 ;USER INST SPACE ADR REG'S

```
297          UPAR1= 177642
298          UPAR2= 177644
299          UPAR3= 177646
300          UPAR4= 177650
301          UPAR5= 177652
302          UPAR6= 177654
303          UPAR7= 177656
304
305          UPDR0= 177600          ;USER INST SPACE DESCR. REG'S
306          UPDR1= 177602
307          UPDR2= 177604
308          UPDR3= 177606
309          UPDR4= 177610
310          UPDR5= 177612
311          UPDR6= 177614
312          UPDR7= 177616
313
314          .IF DF MIMIC & UMTEST
315          .IFF
316          UBMAP0= 170200          ;UNIBUS MAP REGISTERS
317          UBMAP1= 170204
318          UBMAP2= 170210
319          UBMAP3= 170214
320          UBMAP4= 170220
321          UBMAP5= 170224
322          UBMAP6= 170230
323          UBMAP7= 170234
324          UBMAP8= 170240
325          .IFT
326          UBMAP0= 600          ;UNIBUS MAP REGISTERS
327          UBMAP1= 604
328          UBMAP2= 610
329          UBMAP3= 614
330          UBMAP4= 620
331          UBMAP5= 624
332          UBMAP6= 630
333          UBMAP7= 634
334          UBMAP8= 640
335          .ENDC
336
337          P0CONS= 000000          ;PAGE SELECT EQUATES
338          P1CONS= 020000
339          P2CONS= 040000
340          P3CONS= 060000
341          P4CONS= 100000
342          P5CONS= 120000
343          P6CONS= 140000
344          P7CONS= 160000
345
346          IOPAGE= 177600          ;MISC. EQUATES
347          PDRCON= 077406
348          USRPSW= 140000
349          KERPSW= 0
350          .ENDC
```

002

001

000

352					.SBTTL EXEC INTERFACE ADDRESS TABLE	
353						
354						
355	000000'	053124	040504	037477	CLOCZ: .ASCII 'TVDA??'	;CURRENT FILENAME IN ASCII ;PRESET TO THE 'VALID DEVICES' FILE
356						
357						
358	000006'	000000			BOOT: .WORD	XXXX
359	000010'	000000			BUF: .WORD	XXXX
360	000012'	000000			CFTCHI: .WORD	XXXX
361	000014'	000000			CLISTI: .WORD	XXXX
362	000016'	000000			CLOSE: .WORD	XXXX
363	000020'	000000			CPRT: .WORD	XXXX
364	000022'	000000			CREATE: .WORD	XXXX
365	000024'	000000			CSAVEI: .WORD	XXXX
366	000026'	000000			CTFLGW: .WORD	XXXX
367	000030'	000000			CTRD: .WORD	XXXX
368	000032'	000000			CTRDEX: .WORD	XXXX
369	000034'	000000			CTRDNE: .WORD	XXXX
370	000036'	000000			CTUSWR: .WORD	XXXX
371	000040'	000000			CTWR: .WORD	XXXX
372	000042'	000000			CUSPRT: .WORD	XXXX
373	000044'	000000			DELETE: .WORD	XXXX
374	000046'	000000			GET: .WORD	XXXX
375	000050'	000000			LIST: .WORD	XXXX
376	000052'	000000			LOADVR: .WORD	XXXX
377	000054'	000000			OPENL: .WORD	XXXX
378	000056'	000000			PUT: .WORD	XXXX
379	000060'	000000			ZERO: .WORD	XXXX
380	000062'	000000			REALNM: .WORD	XXXX

;ABSOLUTE ADDRESSES OF ROUTINES
;IN THE EXECUTIVE WHICH ARE
;USED BY MPG.

```

382 .SBTTL MPG CONTROL ROUTINE INITIAL HOUSEKEEPING
383
384
385 ;ENTRY POINT FROM STAND ALONE EXEC
386
387 000064' 004567 043476 START: JSR R5,C1THKP ;DO 1 TIME HOUSEKEEPING
388
389 000070' 010700 RST1: MOV PC,R0 ;SET UP CNTRL ROUT TBL ADR
390 000072' 062700 034124 ADD #CRTBL-. ,R0
391 000075' 012702 000020 MOV #MAXPRG,R2 ;SET UP # OF PROG'S
392 000102' 005020 10$: CLR (R0)+ ;CLEAR TBL ENTRY
393 000104' 005302 DEC R2 ;DECR PROG COUNT
394 000106' 001375 BNE 10$ ;DONE ALL ENTRIES? (Y,N-10$)
395 000110' 016767 001340 001330 MOV CPAREA,CFMST ;INITIALIZE START OF FREE MEM ADR
396 000116' 016767 001330 001324 MOV CRMEND,CFMEND ;INIT END OF FREE MEMORY ADR
397 001 .IF DF MM
398 MOV CPAREA,R0 ;GET START OF USR PROG AREA
399 ADD #577,R0 ;ALLOW FOR 12K WORDS
400 CMP R0,CRMEND ;ARE THERE MORE THAN 12K WDS?
401 BHIS RST2 ;Y,N-RST2
402 MOV R0,CFMEND ;STORE END OF 12K AREA ADR
403 CLR MAPMAP ;RESET UNIBUS MAP USAGE
404 CLR MAPMAP+2 ;BIT MAP
405 .ENDC
406 000124' 012706 001200 RST2: MOV #STACK,R6 ;SET UP STACK POINTER
407 000130' 042767 177400 034056 BIC #177400,CSYSFW ;INITIALIZE SYSTEM FLGWD
408 001 .IF DF MM
409 MOV #KPAR0,R0 ;SET UP ADR'S OF KERNEL MODE'S
410 MOV #KPDRO,R1 ;MM REGS
411 MOV #8.,R2 ;SET UP REGISTER COUNT
412 CLR R3 ;SET MEM BASE ADR TO 0
413 5$: MOV R3,(R0)+ ;STORE PAGE'S MEM BASE ADR
414 MOV #PDRCON,(R1)+ ;STORE DESCRIPTOR REG VALUE
415 ADD #200,R3 ;POINT BASE ADR TO NXT MEM PAGE
416 DEC R2 ;DONE ALL REGISTERS?
417 BNE 5$ ;Y,N-5$
418 MOV #IOPAGE,-(R0) ;SET PAR 7 TO THE I/O PAGE
419 BIT #CPU40,CSYSFW ;IS THIS AN 11/40?
420 BNE 10$ ;N,Y-10$
421 MOV MR3VAL,MMMR3 ;INIT MMR3 IF UMAP & 22 BITS
422 10$: MOV #1001,MMMR0 ;ENABLE MEM MANAGEMENT
423
424 ; NOW RUNNING UNDER MEMORY MANAGEMENT
425
426 BIT #UNIMAP,CSYSFW ;ARE WE USING THE UNIBUS MAP?
427 BEQ 15$ ;Y,N-15$
428 MOV #UBMAP0,R0 ;POINT AT FIRST MAP REG
429 CLR (R0)+ ;INITIALIZE REGS 0 & 1 TO
430 CLR (R0)+ ;FIRST BK WDS OF MEM USED
431 MOV #20000,(R0)+ ;BY EXEC & MPG
432 CLR (R0)+
433 ADD #16.,R0 ;BYPASS REGS 2 THRU 5
434 MOV PC,R1 ;GET ADR OF SHARED CODE PAGE
435 ADD #PG6BA-. ,R1
436 MOV R1,(R0)+ ;STORE IT IN MAP REG 6
437 CLR (R0)

```

```

438      000      000      .ENDC
439 000136' 010700      15$: MOV      PC,RO      ;SET UP ADR OF CNTRL ROUT TBL
440 000140' 062700 034056 ADD      #CRTBL-.,RO
441      001      .IF DF MM
442      001      .IFF
443 000144' 012701 000020 MOV      #MAXPRG,R1      ;HOUSEKEEP PROG FLGWD IN CASE
444 000150' 005720      20$: TST      (RO)+      ;WE'RE DOING A RST2 RESTART
445 000152' 001403      BEQ      30$
446 000154' 042750 100070 BIC      #ACTIVE+WT4IOT+CTPRIO+SETDED,2-(RO)
447 000160' 005720      TST      (RO)+
448 000162' 005301      30$: DEC      R1
449 000164' 001371      BNE      20$
450
451      20$: .IFT
452      CMP      #177777,(RO)      ;END OF THE TBL?
453      BEQ      40$      ;N,Y-40$
454      TST      (RO)      ;ENTRY EMPTY?
455      BEQ      30$      ;N,Y-30$
456      MOV      (RO),2#KPAR6      ;SET PROG'S BASE ADR IN KERNEL'S PAR 6
457      BIC      #ACTIVE+WT4IOT+CTPRIO+SETDED,2#P&CONS
458      30$: TST      (RO)+      ;POINT TO NXT TBL ENTRY
459      BR      20$      ;GO CK NXT ENTRY
460      40$: MOV      PC,RO      ;SET UP ADR OF INTERRUPT
461      ADD      #VECTBL+4-.,RO      ;INTERCEPT VECTOR TABLE
462      MOV      #128.,R1      ;# OF ENTRIES
463      50$: CLR      (RO)+      ;CLEAR THIS ENTRY
464      CLR      (RO)+
465      ADD      #4,RO      ;POINT TO NEXT ENTRY
466      DEC      R1      ;DONE ALL ENTRIES?
467      BNE      50$      ;Y,N-50$
468      .ENDC
469 000166' 016700 177634 MOV      CTFLGW,RO      ;GET ADR OF CONSOLE TERM FLGWD
470 000172' 005020      CLR      (RO)+      ;CLEAR THE FLGWD AND WRITE
471 000174' 005010      CLR      (RO)      ;COUNT
472 000176' 005067 001242 CLR      CRFLGW      ;HSPK CNTRL ROUT FLGWD
473 000202' 012767 000003 000226 MOV      #3,CMGCNT      ;INITIALIZE TO NON-INT MODE
474 000210' 116767 007120 007115 MOV      CASMSG+3,CASMSG+2
475 000216' 004567 010546 JSR      R5,CSDNRC      ;SHIFT ANY PROGS DOWN & RECOMPILE
476      001      .IF DF MM
477      MOV      #-1,CSCFWD      ;INIT TO NO PROG #
478      .ENDC
479 000222' 000167 004026 JMP      CMAP      ;GO DISP MEM MAP
480
481      ;USER INTERRUPT (+C) ENTRY POINT
482
483 000226' 012600      000226' 012600      040666      040666      USRINT: MOV      (SP)+,RO      ;GET STACK ADR AT TIME OF INT
484 000230' 004567      JSR      R5,BINASC      ;CONVERT IT TO ASCII &
485 000234' 007614      .WORD      CINTSP-      ;PUT IN MSG
486 000236' 016600 000014      MOV      12.(SP),RO      ;GET PC VALUE AT INTERRUPT
487 000242' 004567 040654      JSR      R5,BINASC      ;CONVERT IT TO ASCII &
488 000246' 007566      .WORD      CINTPC-      ;PUT IN CONSOLE MSG
489      001      .IF DF MM
490      MOV      #KPAR4,RO      ;SET UP ADR OF KERNEL WORK PAGES
491      MOV      (RO)+,-(SP)      ;SAVE WORK PAGES' BASE ADRS
492      MOV      (RO)+,-(SP)
493      MOV      (RO),-(SP)
    
```

494					MOV	#KN,CINTMD		;STORE "KNRL" AS
495					MOV	#RL,CINTMD+2		;THE MODE
496					BIT	#140000,20.(SP)		;WERE WE IN KERNEL MODE?
497					BEQ	10\$;N.Y-10\$
498					MOV	#US,CINTMD		;STORE "USER" AS
499					MOV	#ER,CINTMD+2		;THE MODE
500		000			.ENDC			
501	000250'	005767	034412	10\$:	IST	CPGCPT		;USER PROG IN CONTROL?
502	000254'	001412			BEQ	20\$;Y.N-20\$
503	000256'	016704	034210		MOV	CPGCPE,R4		;GET CURR CNTRL ROUT TBL ENTRY ADR
504	000262'	004767	013322		JSR	PC,CPNASC		;CONVERT IT TO ASCII PROG #
505	000266'	010167	007574		MOV	R1,CINTPN		;MOVE PROG # TO MSG
506	000272'	112767	000040	007570	MOVB	#40,CINTPN+2		;SET THIRD BYTE TO A SPACE
507	000300'	000406			BR	30\$;GO TO ISSUE MSG
508	000302'	012767	050115	007556	20\$:	MOV	#MP,CINTPN	;SET UP "MPG" I.D.
509	000310'	112767	000107	007552	MOVB	#G,CINTPN+2		
510	000316'	042767	010000	033670	30\$:	BIC	#CNTRLO,CSYSFW	;RESET SUPPRESS TYPING FLAG
511	000324'	004567	005674		JSR	R5,CTYPE		;ISSUE PC & SP VALUES MSG
512	000330'	000400			.WORD	400		
513	000332'	007464			.WORD	CINTMG-		
514	000334'	012767	000004	000074	MOV	#4,CMGCNT		;SET UP CNT FOR "I*"
515	000342'	112767	000111	006763	MOVB	#I,CASMSG+2		;PROMPTING MSG
516								;DROP INTO COMMAND DECODER

.SBTTL COMMAND DECODE & DISPATCHER ROUTINE

```

*****
THIS ROUTINE REQUESTS THE ENTRY OF USER COMMANDS THROUGH THE
CONSOLE KEYBOARD. UPON RECEIVING A COMMAND, THE COMMAND
NAME WILL FIRST BE VALIDATED FOLLOWED BY THE EXTRACTION OF THE
PROCESSING ROUTINE ADDRESSES FOR THE COMMAND. NEXT, THE COMMAND
TYPE IS TESTED AND IF IT REQUIRES AN OPERAND, THE COMMAND DATA
WILL BE CHECKED TO INSURE THAT ONE WAS ENTERED.
*****

```

```

519
519
520
521
522
523
524
525
526
527
528
529
530
531
532 000350' 042767 010000 033636 CFCMND: BIC      #CNTRLO,CSYSFW      ;RESET SUPPRESS TYPING FLG
533 000356' 004567 005642          JSR      R5,CTYPE      ;ISSUE 'ENTER CMND' MSG
534 000362' 000400          .WORD    400
535 000364' 006730          .WORD    CECMSG-.
536 000366' 010700          CNCMND: MOV      PC,R0      ;INITIALIZE RETURN ADR FOR
537 000370' 005740          TST      -(R0)        ;ERROR ROUTINE
538 000372' 010067 001204          MOV      R0,CERRET
539 000376' 042767 010000 033610 BIC      #CNTRLO,CSYSFW      ;RESET SUPPRESS TYPING FLG
540 000404' 005767 033604          TST      CSYSFW        ;DOING A CONS REQ?
541 000410' 100406          BMI      1$           ;N,Y-1$
542 000412' 022706 001200          CMP      #STACK,SP     ;CHECK FOR STACK ERROR
543 000416' 001403          BEQ      1$           ;Y,N-1$
544 000420' 000000          HALT
545 000422' 012706 001200          MOV      #STACK,R6     ;RE-INITIALIZE STACK PNTR
546 000426' 004567 005572          1$: JSR      R5,CTYPE      ;ISSUE "*" W/O TRAILING CR/LF
547 000432' 000420          .WORD    420
548 000434' 006675          .WORD    CASMSG-.
549 000436' 000003          CMGCNT: .WORD    3
550 000440' 004577 177364          JSR      R5,ACTRD      ;ISSUE KEYBOARD READ
551 000444' 001020          .WORD    CCMND-.
552 000446' 000107          .WORD    71
553 000450' 004567 007542          JSR      R5,CSCNSP     ;SCAN FOR FIRST WORD OF CMND
554 000454' 010700          MOV      PC,R0      ;SET UP ADR OF CONTROL
555 000456' 062700 000766          ADD      #CRFLGW-. ,R0 ;ROUTINE FLAGWORD
556 000462' 032702 000010          BIT      #CSCFND,R2   ;ANY DATA ENTERED ON THE CMND?
557 000466' 001020          BNE      CKCMND       ;N,Y-CKCMND
558 000470' 005767 033520          CGORUN: TST      CSYSFW   ;CONSOLE REQ BEING PROCESSED?
559 000474' 100402          BMI      2$           ;N,Y-2$
560 000476' 000167 033732          JMP      CPGDSP        ;GO TO PROGRAM DISPATCHER
561 000502' 042767 100000 033504 2$: BIC      #CTRQBP,CSYSFW   ;RESET CT REQ BEING PROC FLAG
562 000510' 012767 000003 177720          MOV      #3,CMGCNT    ;RESTORE PROMPTING MSG TO "*"
563 000516' 116767 006612 006607          MOV      CASMSG+3,CASMSG+2
564          .IF DF MM
565          MOV      #KPAR6,R0 ;SET UP ADR OF KERNEL WORK PAGES
566          MOV      (SP)+,(R0) ;RELOAD ORIGINAL VALUES
567          MOV      (SP)+,-(R0)
568          MOV      (SP)+,-(R0)
569          .ENDC
570 000524' 000177 177302          JMP      ACTRDEX       ;GO RESTORE & EXIT FROM INT
571 000530' 052710 100000          CKCMND: BIS      #CDDFP,(R0) ;SET FIRST PASS FLAG
572 000534' 010702          10$: MOV      PC,R2     ;SET UP START ADR OF VALID
573 000536' 062702 000424          ADD      #CMDTBL-. ,R2 ;CMND TBL

```

574	000542'	012705	000007		MOV	#7,R5	:SET COMPARE COUNT TO 7
575	000546'	016703	001024	20\$:	MOV	CDSTAD,R3	:GET DATA WORD START ADR
576	000552'	010501		30\$:	MOV	R5,R1	:MOVE COMPARE CNT TO WORK CNT REG
577	000554'	005710			TST	(R0)	:FIRST PASS FLAG SET?
578	000556'	100404			BMI	40\$:N,Y-40\$
579	000560'	012701	000002		MOV	#2,R1	:SET WORK CNT TO 2
580	000564'	010504			MOV	R5,R4	:SAVE COMPARE CNT -2
581	000566'	160104			SUB	R1,R4	
582	000570'	122712	000377	40\$:	CMPB	#377,(R2)	:IS ENTRY'S CODE BYTE ALL 1'S
583	000574'	001013			BNE	60\$:Y,N-60\$
584	000576'	122705	000002		CMPB	#2,R5	:THIS LAST GROUP OF CMND'S?
585	000602'	001403			BEQ	50\$:N,Y-50\$
586	000604'	005305			DEC	R5	:DECR COMPARE CNT BY 1
587	000606'	105722			TSTB	(R2)+	:ADD 1 TO COMPARE TBL ADR
588	000610'	000760			BR	30\$:GO DO NEXT CMND GROUP
589	000612'	005710		50\$:	TST	(R0)	:THIS THE FIRST PASS?
590	000614'	100045			BPL	100\$:Y,N-100\$
591	000616'	042710	100000		BIC	#CDDFP,(R0)	:RESET FIRST PASS FLAG
592	000622'	000744			BR	10\$:DO SECOND PASS WITH 2 CHAR COMPARES
593	000624'	112246		60\$:	MOVB	(R2)+,-(SP)	:SAVE FIRST CODE BYTE
594	000626'	000316			SWAB	(SP)	:SAVE SECOND CODE BYTE
595	000630'	112216			MOVB	(R2)+,(SP)	
596	000632'	005301		70\$:	DEC	R1	:DECR WORK COMPARE CNT BY 1
597	000634'	122223			CMPB	(R2)+,(R3)+	:CMND TBL CHAR = DATA WORD CHAR?
598	000636'	001406			BEQ	90\$:N,Y-90\$
599	000640'	005710			TST	(R0)	:FIRST PASS FLAG SET?
600	000642'	100401			BMI	80\$:N,Y-80\$
601	000644'	060402			ADD	R4,R2	:ADD CMP CNT -2 TO TBL ADR
602	000646'	060102		80\$:	ADD	R1,R2	:ADD REMAINING CNT TO TBL ADR
603	000650'	005726			TST	(SP)+	:TAKE CODE BYTES OFF THE STACK
604	000652'	000735			BR	20\$:GO CHECK NEXT TBL ENTRY
605	000654'	005701		90\$:	TST	R1	:WORK COMPARE CNT = 0?
606	000656'	001365			BNE	70\$:Y,N-70\$
607	000660'	012604			MOV	(SP)+,R4	:GET THIS CMND'S SAVED CODE BYTES
608	000662'	032704	010000		BIT	#10000,R4	:CMND POSSIBLY HAVE IMBEDDED PROG #?
609	000666'	001404			BEQ	95\$:Y,N-95\$
610	000670'	004567	012576		JSR	R5,CISDNM	:GO CHECK IF # IS INCLUDED
611	000674'	005701			TST	R1	:WAS THERE A #?
612	000676'	001020			BNE	105\$:N,Y-105\$
613	000700'	122713	000040	95\$:	CMPB	#40,(R3)	:NEXT CHAR AFTER CMND I.D. A SPACE,
614	000704'	001415			BEQ	105\$:COMMA, CR, OR LF?
615	000706'	122713	000054		CMPB	#054,(R3)	:N,Y-105\$
616	000712'	001412			BEQ	105\$	
617	000714'	122713	000015		CMPB	#015,(R3)	
618	000720'	001407			BEQ	105\$	
619	000722'	122713	000012		CMPB	#012,(R3)	
620	000726'	001404			BEQ	105\$	
621	000730'	004567	005270	100\$:	JSR	R5,CTYPE	:GO REPORT INV CMND ERROR
622	000734'	000112			.WORD	112	:RETURN AT LABEL "CNCMD"
623	000736'	005702			.WORD	CERM04-	
624	000740'	005767	033250	105\$:	TST	CSYSFW	:CONSOLE REQ BEING PROCESSED?
625	000744'	100007			BPL	CNCTU	:Y,N-CNCTU
626	000746'	032704	040000		BIT	#40000,R4	:THIS A RESTRICTED CMND?
627	000752'	001404			BEQ	CNCTU	:Y,N-CNCTU
628	000754'	004567	005244	CRSCMD:	JSR	R5,CTYPE	:ISSUE 'RESTRICTED CMND' MSG
629	000760'	000002			.WORD	002	

630	000762'	005643			.WORD	CERM03-	
631	000764'	032704	020000	CNCTU:	BIT	#20000,R4	; THIS CMND REQUIRE OPERANDS?
632	000770'	001007			BNE	107\$; Y,N-107\$
633	000772'	005767	007216		TST	CSCFWD	; IS THERE AT LEAST ONE OPERAND?
634	000776'	100004			BPL	107\$; N,Y-107\$
635	001000'	004567	005220		JSR	R5,CTYPE	; REPORT "NO OPERAND" ERROR
636	001004'	000042			.WORD	042	; RETURNS AT LABEL "CNCMD"
637	001006'	005663			.WORD	CERM07-	
638	001010'	110405		107\$:	MOVB	R4,R5	; COMPUTE DISPLACEMENT INTO
639	001012'	060705			ADD	PC,R5	; CMND ROUT ADR TBL
640	001014'	062705	000060		ADD	#CRATBL-. ,R5	
641	001020'	011505			MOV	(R5),R5	; GET REL ADR OF CMND ROUT
642	001022'	060705			ADD	PC,R5	; MAKE IT ABSOLUTE
643	001024'	010567	000556	CBASE2:	MOV	R5,CR2ADR	; STORE CMND ROUT ADR AT STD LOC
644	001030'	005704			TST	R4	; THIS CMND USE A COM PROC ROUTINE?
645	001032'	100001			BPL	110\$; N,Y-110\$
646	001034'	000115			JMP	(R5)	; GO DIRECTLY TO THIS CMND'S ROUTINE
647	001036'	000304		110\$:	SWAB	R4	; COMPUTE DISPLACEMENT INTO COM
648	001040'	042704	177760		BIC	#177760,R4	; PROC ROUT ADR TBL
649	001044'	060704			ADD	PC,R4	
650	001046'	062704	000016		ADD	#CPATBL-. ,R4	
651	001052'	011404			MOV	(R4),R4	; GET REL ADR OF COM PROC ROUT
652	001054'	060704			ADD	PC,R4	; MAKE IT ABSOLUTE
653	001056'	010467	000522	CBASE1:	MOV	R4,CR1ADR	; STORE THIS ADR AT STD LOC
654	001062'	000114			JMP	(R4)	; GO TO CMND'S COM PROC ROUTINE

;COMMON PROCESSING ROUTINES ADDRESS TABLE

659	001064'	000632		CPATBL:	.WORD	CCFMT1-CBASE1	
660	001066'	000726			.WORD	CCFMT2-CBASE1	
661	001070'	000742			.WORD	CCFMT3-CBASE1	
662	001072'	000620			.WORD	CCFMT4-CBASE1	

;COMMAND ROUTINES ADDRESS TABLE

666	001074'	002214		CRATBL:	.WORD	CDISPL-CBASE2	; DISPLAY CMND
667	001076'	004254			.WORD	CASSIG-CBASE2	; ASSIGN
668	001100'	001370			.WORD	CDELET-CBASE2	; DELETE
669	001102'	001410			.WORD	CMODIF-CBASE2	; MODIFY
670	001104'	002322			.WORD	CREPOR-CBASE2	; REPORT
671	001106'	001014			.WORD	CENTER-CBASE2	; ENTER
672	001110'	003544			.WORD	CFETCH-CBASE2	; /FETCH
673	001112'	001452			.WORD	CCONT-CBASE2	; CONT
674	001114'	002434			.WORD	CFILL-CBASE2	; FILL
675	001116'	001612			.WORD	COPSW-CBASE2	; O.SW
676	001120'	003466			.WORD	CSAVE-CBASE2	; /SAVE
677	001122'	001436			.WORD	CSTOP-CBASE2	; STOP
678	001124'	005110			.WORD	CBOC-CBASE2	; BOC
679	001126'	004672			.WORD	CRDM-CBASE2	; RDM
680	001130'	001734			.WORD	CRUN-CBASE2	; RUN
681	001132'	004534			.WORD	CWRM-CBASE2	; WRM
682	001134'	003230			.WORD	CMAF-CBASE2	; MM
683	001136'	001332			.WORD	CENTPB-CBASE2	; FM
684	001140'	005016			.WORD	CADD-CBASE2	; ADD
685	001142'	005062			.WORD	CSJB-CBASE2	; S.B

```

686 001144' 005140 .WORD CCDB-CBASE2 ;CDB
687 001146' 005164 .WORD CCBD-CBASE2 ;CBD
688 001150' 001474 .WORD CKILL-CBASE2 ;KILL
689 001152' 003774 .WORD CSLDEL-CBASE2 ;/DELETE
690 001154' 004016 .WORD CLST-CBASE2 ;/LIST
691 001156' 004104 .WORD CZERO-CBASE2 ;/ZERO
692 001160' 004244 .WORD CBOOT-CBASE2 ;/BOOT
693 001 .IF DF MM
694 .WORD CSHIFT-CBASE2 ;SHIFT
695 .WORD CUBMAP-CBASE2 ;UBMAP
696 000 .ENDC

```

;VALID COMMAND TABLE

;FORMAT OF CODE BYTES

;FIRST CODE BYTE:

```

; BITS 0 THRU 3 = DISPLACEMENT INTO COM PROC ROUT ADR TBL
; BIT 4 = 1 = CMND MAY CONTAIN IMBEDDED PROG # (OPSW)
; BIT 5 = 1 = CMND DOES NOT REQUIRE OPERANDS
; BIT 6 = 1 = CMND RESTRICTED DURING USER INTERRUPT
; BIT 7 = 1 = CMND DOES NOT USE COM PROC ROUT

```

;SECOND CODE BYTE:

; BITS 0 THRU 6 = DISPLACEMENT INTO CMND ROUT ADR TBL

```

716 001162' 000000 044504 050123 CMOTBL: .ASCII <00><00>'DISPLAY' ;7 CHARACTER CMNDS
717 001170' 040514 131
001173' 00 027456 042504 .ASCII <200><56>' /DELETE'
001200' 042514 042524
718 001204' 377 .BYTE 377
719 001205' 004 040402 051523 .ASCII <04><02>'ASSIGN' ;6 CHARACTER CMNDS
001212' 043511 116
720 001215' 100 042004 046105 .ASCII <100><04>'DELETE'
001222' 052105 105
721 001225' 100 046406 042117 .ASCII <100><06>'MODIFY'
001232' 043111 131
722 001235' 000 051010 050105 .ASCII <00><10>'REPORT'
001242' 051117 124
723 001245' 104 027414 042506 .ASCII <104><14>' /FETCH'
001252' 041524 110
724 001255' 377 .BYTE 377
725 001256' 005100 047105 042524 .ASCII <100><12>'ENTER' ;5 CHARACTER CMNDS
001264' 122
726 001265' 240 027460 044514 .ASCII <240><60>' /LIST'
001272' 052123
727 001274' 012000 051457 053101 .ASCII <00><24>' /SAVE'
001302' 105
728 001303' 240 027462 042532 .ASCII <240><62>' /ZERO'
001310' 047522
729 001312' 032240 041057 047517 .ASCII <240><64>' /BOOT'
001320' 124

```



```

730          001          .IF DF MM
731          .ASCII <100><66>'SHIFT'
732          .ASCII <100><70>'UBMAP'
733          000          .ENDC
734 001321' 377          .BYTE 377
735 001322' 007000 047503 052116 .ASCII <00><16>'CONT' ;4 CHARACTER CMNDS
736 001330' 010004 044506 046114 .ASCII <04><20>'FILL'
737 001336' 011220 050117 053523 .ASCII <220><22>'OPSW'
738 001344' 013000 052123 050117 .ASCII <00><26>'STOP'
739 001352' 026000 044513 046114 .ASCII <00><54>'KILL'
740 001360' 377          .BYTE 377
741 001361' 002 041030 041517 .ASCII <02><30>'BOC' ;3 CHARACTER CMNDS
742 001366' 015002 042123 115 .ASCII <02><32>'RDM'
743 001373' 046 051034 047125 .ASCII <46><34>'RUN'
744 001400' 017002 051127 115 .ASCII <02><36>'WRM'
745 001405' 200 040444 042104 .ASCII <200><44>'ADD'
746 001412' 023200 052523 102 .ASCII <200><46>'SUB'
747 001417' 200 041450 041104 .ASCII <200><50>'CDB'
748 001424' 025200 041103 104 .ASCII <200><52>'CBO'
749 001431' 377          .BYTE 377
750 001432' 020240 046515 .ASCII <240><40>'MM' ;2 CHARACTER CMNDS
751 001436' 021340 046506 .ASCII <340><42>'FM'
752 001442' 377          .BYTE 377 ;TABLE TERMINATOR
753 001444'          .EVEN
754
755
756          ;CONTROL ROUTINE CONSTANTS AND WORK AREAS
757
758
759 001444' 000000 CRFLGW: .WORD 0 ;CONTROL ROUTINE FLAGWORD
760          020000 CVLFLG= 20000
761          040000 CFTCHF= 40000
762          100000 CDDFP= 100000
763
764          001          .IF NOF MM
765 001446'          CVMST:          .ENDC
766          000          .ENDC
767 001446' 000000 CFMST: .WORD XXXX ;*;START OF FREE MEMORY ADR
768          001          .IF NOF MM
769 001450'          CVMEND:          .ENDC
770          000          .ENDC
771 001450' 000000 CFMEND: .WORD XXXX ;*;END OF FREE MEMORY ADR
772 001452' 000000 CRMEND: .WORD XXXX ;*;END OF REAL MEMORY
773 001454' 000000 CPAREA: .WORD XXXX ;*;START ADR OF USER PROGRAM AREA
774          001          .IF DF MM
775          CVMST: .WORD P4CONS
776          CVMEND: .WORD P4CONS+24574.
777          .ENDC
778
779 001456' 000000 CURCTE: .WORD 0 ;ADR OF CURRENT CNTRL ROUT TBL ENTRY
780 001460' 000000 CURPTA: .WORD 0 ;ADR OF CURRENT PROGRAM TABLE
781          001          .IF DF MM
782          CURPTH: .WORD 0 ;*;HIGH 16 BITS OF CUR PROG TBL ADR
783          .ENDC
784
785 001462' 001464'          CPNTR: .WORD CCMDRD ;CURRENT CMND RD DATA POINTER

```

```

786 001464' 000110          CCMDRD: .BLKB  110          ;CONTROL ROUTINE CONS TERM READIN AREA
787 001574'   015          012          .BYTE  015,012
788
789 001576' 000000          CDSTAD: .WORD  0          ;START ADR OF WORD IN CMND DATA
790 001E00' 000000          CDCNT:  .WORD  0          ;NUMBER OF CHARS IN WORD FOUND
791
792 001602' 000000          CERRET: .WORD  0          ;PRESTORED RETURN ADR FOR ERROR ROUT
793
794 001604' 000000          CR1ADR: .WORD  0          ;ADR OF COM PROC ROUT FOR CURR CMND
795 001606' 000000          CR2ADR: .WORD  0          ;ADR OF CMND ROUTINE FOR CURR CMND
796
797          000000          XXXX=   0          ;DATA THAT WILL BE TAILORED BY MPG
798          000400          CIOALT= 256.      ;PROG I/O AREA LENGTH DEFINITION
799
800 001610' 000003          CWA:    .BLKW  3          ;3 WORD WORK AREA
801
802 001616' 000000          CDREND: .WORD  0          ;END OF DEVICE ROUTINE ADR
803 001620' 000012          CMCDVN: .BLKW 10.      ;MODEL CODE & DEV #'S STORAGE AREA
804
805 001644' 051117 044120 047101 CDFNAM: .ASCII /ORPHAN/   ;DEFAULT PROGRAM NAME
806          001          .IF DF MM
807
808          MR3VAL: .WORD  0          ;MMMR3 VALUE - SET BY 1 TIME HSKP
809
810          MAPMAP: .WORD  0          ;BIT MAP FOR UNIBUS MAP
811          ;
812          ;
813          ;
814          .WORD  0          ;REGISTER USAGE (REGS 30 - 7)
815          ;
816          ;
817          ;
818          ;
819          ;
820 001652'          .ENDC
821          PATCH:  .REPT 10.      ;***** PATCH AREA *****
822          .WORD  0
          .ENDR

```

:				28	25	
:			30	27	24	
:			29	26	23	
:			REGISTER USAGE (REGS 30 - 7)			
:		21	18	15	12	9
:		20	17	14	11	8
:	22	19	16	13	10	7

824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879

.SBTTL COMMON PROCESSING ROUTINES FOR COMMANDS

THIS ROUTINE PROVIDES COMMON PROCESSING AND CHECKS FOR THE FOLLOWING COMMANDS WHICH HAVE A PROGRAM NUMBER AS THE FIRST OPERAND:

DISPLAY CONT
DELETE /SAVE
MODIFY STOP
REPORT RUN (CCFMT4)
ENTER KILL
 .IF DF MM
SHIFT MM
.ENDC

001
000

UPON BRANCH TO CMND ROUTINE:

R3 = CURRENT PROG TABLE ADR
R4 = CURRENT CONTROL ROUT TBL ENTRY ADR

ALSO PRESENT ARE END OF COMMAND CHECKS WHICH MAY BE USED BY ALL COMMANDS

CCFMT4: TST CSCFWD ;PROG # PRESENT?
 BPL CCFMT1 ;N,Y-CCFMT1
 JMP CGORUN ;GO INITIATE PROG EXECUTION

CCFMT1: JSR R5,CVPNUM ;GO VALIDATE PROG # IN CMND DATA
 .IF DF MM
 JSR PC,UPAPAR ;MAP PAR'S TO USER'S PROG AREA
 .ENDC
 MOV PC,R0 ;SET UP ADR OF ENTER CMND ROUT
 ADD #CENTER-,R0
 CMP R0,CR2ADR ;IS THIS THE ENTER CMND?
 BEQ CCFMTJ ;N,Y-CCFMTJ
 TST R3 ;PROG EXIST FOR THIS #?
 BEQ CCFPDE ;Y,N-CCFPDE
CCFMTJ: JMP CR2ADR ;GO TO ROUTINE FOR THIS CMND

CCFPDE: JSR R5,CTYPE ;GO REPORT PROG DOES NOT EXIST ERROR
 .WORD 006
 .WORD CERMOS-

; COMMON COMMAND RETURN POINTS

CCFMTA: TST CSCFWD ;ANOTHER OPERAND IN CMND?
 BPL CCFMT1 ;N,Y-CCFMT1
CCFRET: JMP CNCMND ;GO REQUEST NEXT CMND
CCFMTB: TST CSCFWD ;ANOTHER OPERAND IN CMND?
 BMI CCFRET ;Y,N-CCFRET
 JSR R5,CSCN ;GO SCAN IT

F02

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE

MACY11 27(732) 24-SEP-76 13:54 PAGE 7-1

SEQ 0019

DTMGAC.P11 COMMON PROCESSING ROUTINES FOR COMMANDS

880 001774' 004567 004224
881 002000' 000102
882 002002' 004661

JSR R5,CTYPE
.WORD 102
.WORD CERM06-

;GO REPORT UNNECESSARY OPERAND ERROR

```

884 :*****
885 :
886 : THIS ROUTINE PROVIDES COMMON PROCESSING FOR THE FOLLOWING COMMANDS
887 : WHICH HAVE A WORD ADDRESS IN OCTAL AS THE FIRST OPERAND:
888 :
889 :     BOC
890 :     RDM
891 :     WRM
892 :
893 :     .IF DF MM
894 :     .IFF
895 : ; R4 CONTAINS CONVERTED BINARY WORD UPON BRANCH TO CMND ROUTINE
896 :     .IFT
897 : ; R3 & R4 CONTAINS CONVERTED BINARY WORDS UPON BRANCH TO CMND ROUTINE
898 :     .ENDC
899 :
900 :*****
901 :
902 002004' 004567 006224 CCFMT2: JSR     R5,CSCNB8      ;SCAN OPERAND & CONVERT TO BINARY
903 002010' 004567 011330      JSR     R5,CKODAD      ;GO CHECK FOR ODD ADR
904 002014' 010004          MOV     R0,R4          ;SAVE 16 BIT ADR IN R4
905          001          .IF DF MM
906          .MOV     R2,R3      ;SAVE HIGH 6 BITS IN R3
907          000          .ENDC
908 002016' 000746          BR      CCFMTJ      ;GO TO ROUTINE FOR THIS CMND
909
910 :*****
911 : THIS ROUTINE PROVIDES COMMON PROCESSING FOR THE FOLLOWING
912 : COMMANDS WHICH HAVE AN ASCII WORD AS THE FIRST OPERAND:
913 :
914 :     ASSIGN
915 :     FETCH
916 :     FILL
917 :
918 :*****
919 :
920 :*****
921 :
922 002020' 004567 006176 CCFMT3: JSR     R5,CSCN      ;SCAN FIRST OPERAND
923 002024' 005702          TST     R2              ;ANOTHER OPERAND PRESENT?
924 002026' 100342          BPL     CCFMTJ          ;N,Y-CCFMTJ
925 002030' 004567 004170 CN20ER: JSR     R5,CTYPE      ;REPORT NO 2ND OPERAND ERROR
926 002034' 000042          .WORD  042
927 002036' 004643          .WORD  CERM08-
    
```

.SBTTL ENTER, DELETE, & MODIFY COMMAND ROUTINES

929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984

*
* ENTER COMMAND *
*

002040' 042767 040000 177376
002046' 010700
002050' 062700 177574
002054' 005005
002056' 000167 002440
001

CENTER: BIC #CFTCHF,CRFLGW
MOV PC,R0
ADD #CDFNAM-.,R0
CLR R5
JMP CSVCOM
.IF DF MM
.IFT
CENTPA: JSR R5,CTYPE
.WORD 420
.WORD ADDRREQ-
.WORD 17.
JSR R5,ACTRD
.WORD CCMRD-
.WORD 71.
MOV PC,R0
ADD #CENTPA-.,R0
MOV R0,CERRET
TST (R4)
BEQ 25
JSR PC,MAPHKP
CLR (R4)
TST CRTEXT(R4)
BMI 15
JSR R5,CSDNRC
15: CLR CRTEXT(R4)
25: CLR CENADR
JSR R5,CSCNSP
BIT #CSCFND,R2
BEQ CENPAA
JSR R5,ASOCBN
.WORD CENERB-
JSR PC,CKPGAD
MOV R2,CENADR
CMP R2,HIMPGP
BHIS 35
JMP SHPGE2
35: MOV R2,R1
JSR PC,CKF3PG
TST R2
BEQ CENTPA
CENPAA: JSR R5,CTYPE
.IFF
CENTPA: JSR R5,CTYPE
.ENOC
.WORD 420
.WORD CADMSG-

;RESET THE FETCH FLAG
;SET ADR OF PROG'S NAME TO
;THE DEFAULT NAME
;RESET "SAVE" FLAG
;GO PROCESS PROG NAME
;ASK FOR PROG AREA ADR
;ISSUE READ FOR THE REPLY
;SET COMMON ERROR RET ADR
;TO ISSUE OF 'PROG AREA ADR' MSG
;PROG ALREADY EXIST IN THIS SLOT?
;Y,N-25
;CK & DEALLOCATE UNIBUS MAP REGS
;RESET PROG'S ADR IN SLOT TBL
;WAS IT A FIXED ADR PROG?
;N,Y-15
;SHIFT DOWN MPG CONTROLLED PROGS
;HSPK SLOT'S EXTENSION WORD
;RESET PROG AREA ADR
;SCAN FOR THE REPLY
;WAS ANY DATA ENTERED?
;Y,N-CENPAA
;CONVERT THE REPLY TO BINARY
;GO CK FOR VALID PROG ADR
;SAVE HI 16 BITS OF ADR
;ADR WITHIN MPG CONTROLLED PROGS?
;Y,N-35
;ISSUE 'PROG AREA IN USE' ERR MSG
;PROG ADR TO S/R'S REG
;CK IF AREA IS OCCUPIED
;WAS AREA OCCUPIED?
;N,Y-CENTPA
;ISSUE 'ASGN DEV:' MSG
;ISSUE 'ASGN DEV' MSG

985	002072'	000013			.WORD	11.		
986	002074'	004577	175730		JSR	R5,ACTRD		;ISSUE RD FOR REPLY
987	002100'	177364			.WORD	CCMDRD-		
988	002102'	000107			.WORD	71.		
989	002104'	010700			MOV	PC,RO		;SET COMMON ERROR RET ADR
990		001			.IF DF	MM		
991					.IFF			
992	002106'	062700	177754		ADD	#CENTPA-,RO		;TO ISSUE OF 'ASGN DEV:' MSG
993					.IFT			
994					ADD	#CENPAA-,RO		;TO ISSUE OF 'ASGN DEV:' MSG
995		000			.ENDC			
996	002112'	010067	177464		MOV	RO,CERRET		
997		001			.IF DF	MM		
998					MOV	#P4CONS,CVMST		;INIT VIRTUAL MEM START ADR
999		000			.ENDC			
1000	002116'	004567	010446		JSR	R5,CVLM2		;GO VALIDATE MODEL & DEV #'S
1001	002122'	122767	000377	177470	CMPB	#377,CMCDVN		;THIS DEFAULT TYPE OF DEV?
1002	002130'	001006			BNE	5\$;Y,N-5\$
1003	002132'	126767	040441	177461	CMPB	CNONID+5,CMCDVN+1		;IS IT "NONE"?
1004	002140'	001402			BEQ	5\$;N,Y-5\$
1005	002142'	000167	003472		JMP	CASER1		;GO REPORT INV DEV TYPE
1006	002146'	004767	011226	5\$:	JSR	PC,CSUERR		;SET UP ERROR RETURN ADR
1007		001			.IF DF	MM		
1008					TST	CENADR		;WAS AN ADR SPECIFIED FOR THIS PROG?
1009					BEQ	7\$;Y,N-7\$
1010					BIS	#FIXADR,CRTEXT(R4)		;SET ITS FIXED ADR FLAG
1011		000			.ENDC			
1012	002152'	004567	006416	7\$:	JSR	R5,CSPGUP		;SHIFT UP HIGHER PROGRAMS IF NEEDED
1013		001			.IF DF	MM		
1014					.IFT			
1015					MOV	CENADR,R3		;GET ENTERED PROG AREA ADR
1016					BNE	9\$;WAS THERE ONE? (N,Y-9\$)
1017					MOV	CURPTH,R3		;GET START ADR FOR THIS PROG
1018				9\$:	JSR	PC,UPAPAR		;MAP KERNEL PAR'S TO USR PROG AREA
1019					.IFF			
1020	002156'	016703	177276		MOV	CURPTA,R3		;GET START ADR FOR THIS PROG
1021		000			.ENDC			
1022	002162'	010300			MOV	R3,RO		;SET UP END OF PROG TBL WORK ADR
1023	002164'	062700	000242		ADD	#PTLGTH,RO		
1024	002170'	020067	177254		CMP	RO,CVMEND		;ENOUGH ROOM FOR PROG TBL?
1025	002174'	103102			BHIS	CENERR		;Y,N-CENERR
1026	002176'	012740	000242		MOV	#PTLGTH,-(RO)		;STORE PROG TBL LENGTH WD
1027	002202'	012701	000120		MOV	#PTLGTH-2/2,R1		;GET # OF WORDS TO CLEAR
1028	002206'	005040		10\$:	CLR	-(RO)		;CLEAR WORD OF PROG TBL
1029	002210'	005301			DEC	R1		;DECR WORD CNT
1030	002212'	001375			BNE	10\$;CNT=0? (Y,N-10\$)
1031	002214'	012763	100000	000002	MOV	#100000,POPSW(R3)		;INITIALIZE OPSW WORD
1032	002222'	004767	011362		JSR	PC,CPNASC		;GET PROG # IN ASCII
1033	002226'	010163	000006		MOV	R1,PASCIN(R3)		;STORE IT IN PROG TBL
1034	002232'	010301			MOV	R3,R1		;GET ADR OF PROG NAME IN PROG TBL
1035	002234'	062701	000010		ADD	#PNAME,R1		
1036	002240'	010700			MOV	PC,RO		;GET ADR OF STORED NAME
1037	002242'	062700	175536		ADD	#CLOCZ-,RO		
1038	002246'	012021			MOV	(RO)+,(R1)+		;MOVE NAME TO PROG TBL
1039	002250'	012021			MOV	(RO)+,(R1)+		
1040	002252'	012021			MOV	(RO)+,(R1)+		

1041	002254'	004567	011134		JSR	R5,CSTDVI		;GO STORE DEV INFO IN PROG TBL
1042	002260'	004567	007356		JSR	R5,CLDEVR		;GO LOAD THIS DEV ROUT
1043	002264'	000116			.WORD	CENERR-		INSUF MEM ERR RET ADR
1044	002266'	000110			.WORD	CENDER-		DEV ERROR ERR RET ADR
1045	002270'	032767	040000	177146	BIT	#CFTCHF,CRFLGW		'FETCH' FLAG SET?
1046	002276'	001417			BEQ	30\$		Y,N-30\$
1047	002300'	010300			MOV	R3,RO		;GET ADR OF SOURCE FILE NAME
1048	002302'	062700	000010		ADD	#PNAME,RO		
1049	002306'	010701			MOV	PC,R1		;GET EXEC INTERFACE STORAGE ADR
1050	002310'	062701	175470		ADD	#CLOCZ-,R1		
1051	002314'	012021			MOV	(RO)+,(R1)+		;STORE PROG'S NAME THERE
1052	002316'	012021			MOV	(RO)+,(R1)+		
1053	002320'	011011			MOV	(RO),(R1)		
1054	002322'	004577	175526		JSR	R5,OPENL		;GO OPEN SOURCE FILE
1055		001			.IF DF	MM		
1056					.IFF			
1057	002326'	000050			.WORD	CENDER-		
1058					.IFT			
1059					.WORD	CENRSM-		
1060		000			.ENDC			
1061	002330'	004567	011400		JSR	R5,FETCH		;GO TO FETCH & COMPILE PROG ROUT
1062	002334'	000406			BR	CENTPC		;GO CK FOR SHIFT DOWN
1063	002336'	004567	003662	30\$:	JSR	R5,CTYPE		;ISSUE "ENTER STMT'S" MSG
1064	002342'	000600			.WORD	600		
1065	002344'	005367			.WORD	CESMSG-		
1066	002346'	004567	011266		JSR	R5,ENTER		;GO TO ENTER & COMPILE PROG ROUT
1067	002352'	010067	177070	CENTPC:	MOV	RO,CVMST		;STORE NEW FREE MEM START ADR
1068		001			.IF DF	MM		
1069					MOV	#P4CONS,R3		;RESTORE PROG TBL ADR
1070					BIC	R3,PRDIOA(R3)		;ADJ PROG TBL ADR'S TO
1071					BIC	R3,PWRIOA(R3)		;USER PAGES
1072					BIC	R3,PSRCST(R3)		
1073					BIC	R3,POBJST(R3)		
1074					MOV	PLNGTH(R3),RO		;GET PROG'S LENGTH
1075					BIT	#77,RO		;PROG END AT 32 WD BNDRY?
1076					BEQ	80\$;N,Y-80\$
1077					ADD	#100,RO		;ADJ IT TO NXT HIGHEST 32 WD BNDRY
1078				80\$:	MOV	#6,R1		;SET UP LOOP CNT
1079				90\$:	ASR	RO		;CONVERT IT TO A CNT OF 32 WD
1080					DEC	R1		;BLOCKS
1081					BNE	90\$		
1082					MOV	CURCTE,R4		;RESTORE CNTL ROUT TBL ADR
1083					BIS	RO,CRTXT(R4)		;STORE LGTH IN CNTRL ROUT TBL EXT.
1084		000			.ENDC			
1085	002356'	004567	006406	CENTPB:	JSR	R5,CSDNRC		;SHIFT DOWN PROG'S
1086	002362'	004567	003636		JSR	R5,CTYPE		;ISSUE PGMS COMP/MEM FMTTED MSG
1087	002366'	000400			.WORD	400		
1088	002370'	005363			.WORD	CPCMRF-		
1089	002372'	000167	175770		JMP	CNCMND		;RETURN FOR NEXT CMND
1090								
1091		001			.IF DF	MM		
1092				CENADR:	.WORD	0		;*;PROG ADR HOLD
1093								
1094				CENRSM:	JSR	PC,MAPHKP		;HSPK MAPMAP WDS IF NEEDED
1095		000			.ENDC			
1096	002376'	005014		CENDER:	CLR	(R4)		;RESET PROG'S TBL ENTRY


```

1097 002400' 000766          BK      CENTPB          ;GO SHIFT ANY PROGS DOWN
1098
1099 002402' 005014          CENERR: CLR      (R4)          ;RESET THIS PROG'S CTRL ROUT TBL ADR
1100 002404' 004567 003614    CENERA: JSR      R5,CTYPE      ;ISSUE INSUFF MEM MSG
1101 002410' 000006          .WORD 006
1102 002412' 004455          .WORD CERM23-.
1103          001          .IF DF MM
1104
1105          CENERB: JSR      R5,CTYPE      ;ISSUE DATA CONV
1106          .WORD 302          ;ERROR MSG
1107          .WORD CERM02-.
1108          000          .ENDC
1109
1110
1111          ; *****
1112          ; *
1113          ; * DELETE COMMAND *
1114          ; *
1115          ; *****
1116
1117
1118 002414' 004767 010760    CDELET: JSR      PC,CSUERR      ;GO SET UP ERROR RETURN ADR
1119          001          .IF DF MM
1120          JSR      PC,MAPHKP      ;CK & DEALLOCATE UNIBUS MAP REGS
1121          000          .ENDC
1122 002420' 005014          CLR      (R4)          ;CLEAR THIS PROG'S CTRL ROUT TBL ENTRY
1123 002422' 005767 005566    TST      CSCFWD      ;ANOTHER PROG # IN CMND
1124 002426' 100753          BMI      CENTPB      ;Y,N-CENTPB
1125 002430' 000167 177254    JMP      CCFMT1      ;GO SCAN NXT PROG #

```

```

1127 ; *****
1128 ; *
1129 ; *   MODIFY COMMAND   *
1130 ; *
1131 ; *****
1132 ;
1133 ;
1134 002434' 004767 010740   CMODIF: JSR      PC, CSUERR           ;SET UP ERROR RETURN ADR
1135                001           .IF DF MM
1136                .IFT
1137                MOV      (R4), CENADR       ;STORE HI 16 BITS OF PROG AREA ADR
1138                MOV      R3, CVMST         ;INIT VIRTUAL MEM START ADR
1139                BIS      R3, PRDIOA(R3)    ;ADJ PROG TBL ADR'S TO
1140                BIS      R3, PWRIOA(R3)    ;PAGE 4
1141                BIS      R3, PSRCST(R3)
1142                BIS      R3, POBJST(R3)
1143                .IFTF
1144 002440' 004567 006130   10$: JSR      R5, CSPGUP           ;GO SHIFT UP HIGHER PROGRAMS
1145                .IFT
1146                MOV      (R4), R3         ;GET PROG AREA ADR
1147                JSR      PC, UPAPAR       ;MAP KERNEL'S PAR'S TO USR PRG AREA
1148                .ENDC
1149 002444' 004567 003554   JSR      R5, CTYPE           ;ISSUE "ENTER STMT'S" MSG
1150 002450' 000600
1151 002452' 005261
1152 002454' 004567 011242   JSR      R5, MODIFY         ;GO PROCESS CHANGES & RECOMPILE
1153 002460' 000734           BR       CENTPC             ;GO SHIFT DOWN PROG'S
1154                001           .IF DF MM
1155                .PAGE
1156                .SBTTL  SHIFT AND UBMAR COMMANDS
1157 ;
1158 ;
1159 ; *****
1160 ; *
1161 ; *   SHIFT COMMAND   *
1162 ; *
1163 ; *****
1164 ;
1165 ;
1166 CSHIFT: TST      CSCFWD           ;ANOTHER OPERAND IN CMND?
1167         BPL      10$             ;N, Y-10$
1168         JMP      CN20ER         ;GO REPORT NO 2ND OPER ERROR
1169 10$:    CMPB     #'T, CPNTR       ;IS ITS FIRST CHAR A "T"?
1170         BEQ      20$             ;N, Y-20$
1171         JSR      R5, CTYPE       ;ISSUE 'TO' NOT SPECIFIED ERROR
1172         .WORD   042             ;MSG
1173         .WORD   CERM36-
1174 20$:    JSR      R5, CSCN         ;SCAN OVER THE 'TO'
1175         TST      R2             ;DESTINATION OPERAND SUPPLIED?
1176         BPL      30$             ;N, Y-30$
1177         JSR      R5, CTYPE       ;ISSUE NO DESTINATION ERROR MSG
1178         .WORD   042
1179         .WORD   CERM37-
1180 30$:    MOV      CPNTR, -(SP)     ;SAVE NXT OPERAND PNTR
1181         JSR      R5, CSCN         ;SCAN DESTINATION OPERAND
1182         MOV      (SP)+, CPNTR     ;RESTORE PNTR IN CASE OF A PROG ADR

```

```

1183          CMPB    #'M,DCDSTAD          ;IS ITS FIRST CHAR AN 'M'?
1184          BEQ     SHTMPG                ;N,Y-SHTMPG
1185          JSR     R5,ASDECO              ;CONVERT DATA TO PACKED DECIMAL
1186          .WORD   SHPGNA-
1187          JSR     R5,DECBIN              ;CONVERT PKED DEC TO BINARY
1188          .WORD   SHPGNA-
1189          CMP     R0,#MAXPRG             ;# LARGER THAN HIGHEST PROG #?
1190          BHI     SHPGNA                 ;N,Y-SHPGNA
1191
1192          ;SHIFT PROGRAM TO DIFFERENT SLOT
1193
1194          SHMVSL: MOV    R4,R5              ;SAVE ORIG SLOT ADR
1195          JSR     R5,CVPM1                ;GO VALIDATE DEST OPER AS A PROG #
1196          CMP     R4,R5                  ;THE YO-YO SPECIFY SAME SLOT #'S?
1197          BEQ     SHPGND                  ;N,Y-SHPGND
1198          TST     CRTEXT(R5)              ;IS ORIG PROG AT A FIXED ADR?
1199          BMI     SHMVFA                  ;N,Y-SHMVFA
1200          JSR     PC,CSUERR               ;SET UP ERROR RET ADR
1201          JSR     R5,CSPGUP               ;SHIFT UP ANY PROGS ABOVE NEW SLOT
1202          MOV     (R4),R1                 ;SET UP OPEN AREA ADR
1203          JSR     PC,CKF3PG               ;GET SIZE OF AVAIL MEM IN R2
1204          MOV     CRTEXT(R5),R3          ;GET PROG'S LENGTH
1205          CMP     R2,R3                  ;ENOUGH ROOM FOR PROG?
1206          BLO     SHPGE1                  ;Y,N-SHPGE1
1207          MOV     (R4),-(SP)              ;SAVE ITS NEW ADR
1208          MOV     R5,-(SP)                ;SAVE ORIG SLOT ADR
1209          MOV     (R5),(R4)               ;SET UP 'FROM' ADR
1210          MOV     R1,R5                  ;SET UP DEST. ADR
1211          JSR     PC,MOVPGD               ;MOVE PROG TO NEW MPG ADR
1212          MOV     (SP)+,R5                ;RESTORE ORIG SLOT ADR
1213          MOV     (SP)+,(R5)              ;STORE NEW PROG ADR
1214          SHMVFA: MOV    (R5),(R4)         ;MOVE PROG ADR TO NEW SLOT
1215          MOV     CRTEXT(R5),CRTEXT(R4)  ;MOVE ITS EXTENSION WORD ALSO
1216          CLR     (R5)                   ;RESET OLD PROG SLOT ENTRY
1217          MOV     (R4),R3                 ;GET ITS START ADR
1218          JSR     PC,UPAPAR               ;MAP PAR'S TO USER'S AREA
1219          JSR     PC,CPNASC               ;GET PROG'S NEW SLOT # IN ASCII
1220          MOV     R1,PASCIN(R3)           ;STORE IT IN PROG TBL
1221          SHPGEX: JMP    CENTPB            ;GO RECOMPILE PROG'S AND EXIT
1222          SHPGND: JMP    CNCMND           ;RETURN - NO PROGS MOVED
1223
1224          ;SHIFT PROGRAM TO MPG CONTROL OF ITS ADDRESS
1225
1226          SHTMPG: TST    CRTEXT(R4)        ;IS IT ALREADY UNDER MPG CONTROL?
1227          BPL     SHPGND                  ;N,Y-SHPGND
1228          JSR     PC,CSUERR               ;SET UP ERROR RET ADR
1229          MOV     R4,R0                   ;SET PROG SLOT ADR IN R0
1230          JSR     R5,CSPNFP               ;SHIFT UP ANY MPG CONTROLLED PROGS
1231          MOV     CSPNPA,R1                ;SET UP OPEN AREA START ADR
1232          MOV     R1,R5                   ;SAVE IT FOR MOVE S/R
1233          JSR     PC,CKF3PG               ;GET SIZE OF AVAIL AREA IN R2
1234          MOV     CRTEXT(R4),R3          ;GET PROG'S LENGTH
1235          BIC     #FIXADR,R3              ;RESET ITS FIXED ADR FLAG
1236          CMP     R2,R3                  ;ENOUGH ROOM FOR PROG?
1237          BLO     SHPGE1                  ;Y,N-SHPGE1
1238          JSR     PC,MOVPGD               ;MOVE PROG INTO MPG'S AREA

```

```

1239          MOV      CSPNPA, (R4)          ;STORE ITS NEW ADR IN SLOT TBL
1240          BIC      #FIXADR, CRTEXT(R4)  ;RESET ITS FIXED ADR FLAG
1241          BR       SHPGEX                ;GO TO CMND EXIT
1242
1243          ;SHIFT PROGRAM TO NEW FIXED ADDRESS
1244
1245          SHPGNA: JSR      R5, CSCN          ;RE-SCAN THE DEST OPERAND
1246          JSR      R5, ASOCBN          ;CONVERT ASCII DATA TO BINARY
1247          .WORD    CENERB-
1248          JSR      PC, CKPGAD          ;CK IF PROG ADR IS VALID
1249          CMP      R2, (R4)          ;NEW ADR SAME AS OLD?
1250          BEQ      SHPGND          ;N, Y-SHPGND
1251          CMP      R2, HIMPGP          ;ADR WITHIN MPG CONTROLLED PROGS?
1252          BLO      SHPGE2          ;N, Y-SHPGE2
1253          MOV      R2, R5          ;SAVE SUPPLIED ADR
1254          MOV      R2, R1          ;SET IT IN S/R'S REG
1255          JSR      PC, CKF3PG          ;CHECK TO SEE HOW MUCH MEM AVAIL
1256          TST      R2          ;ANY ROOM FOR PROG?
1257          BEQ      SHPGND          ;Y, N-SHPGND
1258          MOV      CRTEXT(R4), R3      ;GET PROG'S LGTH IN 32 WD BLKS
1259          BIC      #FIXADR, R3        ;RESET FIXED ADR FLG IF THERE
1260          CMP      R2, R3          ;ENOUGH ROOM FOR THIS PROG?
1261          BLO      SHPGE1          ;Y, N-SHPGE1
1262          MOV      R5, -(SP)          ;SAVE ITS STARTING ADR
1263          JSR      PC, MOVPGD          ;MOV PROG TO ITS NEW AREA
1264          MOV      (SP)+, (R4)        ;STORE ITS START ADR
1265          BIS      #FIXADR, CRTEXT(R4) ;MAKE IT A FIXED ADR PROGRAM
1266          MOV      (R4), R3          ;SET UP ITS BASE ADR
1267          JSR      PC, UPAPAR          ;MAP PAR'S TO THE NEW AREA
1268          BIT      #UNIMAP, CSYSFW    ;USING UNIBUS MAP?
1269          BNE      SHPGEX          ;N, Y-SHPGEX
1270          BIT      #USEUBM, (R3)      ;PROG USE UNIBUS MAP?
1271          BEQ      SHPGEX          ;Y, N-SHPGEX
1272          JSR      PC, CKUBAD          ;CK IF HE'S OUT OF RANGE
1273          BR       SHPGEX          ;GO TO EXIT
1274
1275          SHPGE1: JSR      R5, CTYPE      ;GO REPORT INSUF. ROOM FOR PROG
1276          .WORD    002
1277          .WORD    CERM38-
1278          SHPGE2: JSR      R5, CTYPE      ;ISSUE 'PROG AREA IN USE'
1279          .WORD    002                ;ERROR MSG
1280          .WORD    CERM34-
1281
1282          ;VALIDATE PROGRAM AREA ADR
1283
1284          ;JSR      PC, CKPGAD          S/R CALL
1285
1286          ;R0 & R2 CONTAINS A 22 BIT ADR
1287
1288          ;ON EXIT: R2 CONTAINS HI 16 BITS OF ADR
1289          ;DESTROYS R0, R1
1290
1291          CKPGAD: MOV      #10., R1      ;GET HI 16 BITS OF ADR
1292          LOS:      ROL      R0          ;IN R2
1293          ROL      R2
1294

```

1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330

```

DEC      R1
BNE     10$
BIT     #176000,R0      ;ADR ON A 32 WD BNDRY?
BEQ     20$            ;N Y-20$
INC     R2            ;POINT TO NXT 32 WD BLK
20$:    CMP     R2,CPAREA ;IS ADR WITHIN MPG AREA?
        BHIS    30$
        JSR     R5,CTYPE ;ISSUE "ADR WITHIN MPG" ERROR
        .WORD   003      ;MSG
        .WORD   CERM35-.
30$:    CMP     R2,CRMEND ;IS ADR > MEM?
        BLOS    40$
        JSR     R5,CTYPE ;ISSUE "ADR > MEM" ERROR
        .WORD   003      ;MSG
        .WORD   CERM09-.
40$:    RTS      PC      ;EXIT IN-LINE

```

```

:      *****
:      *
:      *  UBMAP COMMAND  *
:      *
:      *****

```

```

CUBMAP: BIT     #UNIMAP,CSYSFW ;USING THE UNIBUS MAP?
        BEQ     SHPGND        ;Y,N-SHPGND
        TST     PUBMAP(R3)    ;THIS PROG USE UNIBUS MAP?
        BEQ     SHPGND        ;Y,N-SHPGND
        JSR     PC,MAPHKP     ;RESET REGS CURRENTLY USING
        CLR     PUBMAP(R3)    ;RESET THIER POINTERS
        JSR     R5,CROUBM     ;GO REQUEST NEW MAP REGS
        NOP
        NOP
        BR      SHPGEX
        .ENDC

```

000

```

1332          .SBTTL  STOP, CONT, KILL, & OPSW COMMAND ROUTINES
1333
1334          :          *****
1335          :          *          *
1336          :          *  STOP COMMAND  *
1337          :          *          *
1338          :          *****
1339
1340 002462' 005713  CSTOP:  TST      (R3)          ;THIS PROG ACTIVE?
1341 002464' 100011      BPL      CINACT        ;Y,N-CINACT
1342 002466' 052713 000002  BIS      #URSTOP,(R3)    ;SET USER STOP FLAG FOR THIS PROG
1343 002472' 000167 177252  CLSJMP: JMP      CCFMTR        ;GO CHECK FOR ANOTHER PROG *
1344
1345
1346          :          *****
1347          :          *          *
1348          :          *  CONT COMMAND  *
1349          :          *          *
1350          :          *****
1351
1352 002476' 005713  CCONT:  TST      (R3)          ;THIS PROG ACTIVE?
1353 002500' 100003      BPL      CINACT        ;Y,N-CINACT
1354 002502' 042713 000006  BIC      #URSTOP+ERSTOP,(R3) ;RESET USER & ERROR STOP FLAGS
1355 002506' 000771      BR       CLSJMP        ;EXIT
1356
1357 002510' 004567 003510  CINACT: JSR      R5,CTYPE      ;ISSUE 'INACTIVE' ERROR MSG
1358 002514' 000006      .WORD   006
1359 002516' 004445      .WORD   CERM27-.
1360
1361
1362          :          *****
1363          :          *          *
1364          :          *  KILL COMMAND  *
1365          :          *          *
1366          :          *****
1367
1368 002520' 005713  CKILL:  TST      (R3)          ;THIS PROG ACTIVE?
1369 002522' 100372      BPL      CINACT        ;Y,N-CINACT
1370 002524' 042713 100000  BIC      #ACTIVE,(R3)    ;RESET ITS ACTIVE FLAG
1371 002530' 005767 031460  TST      CSYSFW        ;CONSOLE REQ BEING PROCESSED?
1372 002534' 100356      BPL      CLSJMP        ;Y,N-CLSJMP
1373 002536' 032713 000040  BIT      #SETDED,(R3)   ;THIS PROG SET PRT DEV DEDICATED FLG?
1374 002542' 001415      BEQ      5$          ;Y,N-5$
1375 002544' 032767 004000 031442  BIT      #CHPINT,CSYSFW ;IS THE PRINTER BEING USED?
1376 002552' 001404      BEQ      2$          ;Y,N-2$
1377 002554' 016700 175234  MOV      CLISTI,RO      ;GET ADR OF PRINTER INFO
1378 002560' 005070 000010  CLR      28.(RO)       ;RESET PRINTER'S INT ENABLE
1379 002564' 042767 064000 031422 2$:  BIC      #CTUDED+CRUDED+CHPINT,CSYSFW ;RESET PRT DEV DED. FLAGS
1380 002572' 042713 000040  BIC      #SETDED,(R3)   ;RESET THIS PROG'S "I SET IT" FLG
1381 002576' 021467 032064 5$:  CMP      (R4),CPGCPT    ;IS THIS THE PROG THAT'S RUNNING?
1382 002602' 001005      BNE      10$          ;Y,N-10$
1383          .IF DF MM
1384          .IFF
1385 002604' 010700      MOV      PC,RO          ;SET UP ABORT RET ADR FOR PROG
1386 002606' 062700 032406  ADD      #CKILEX-,RO
1387 002612' 010066 000014  MOV      RO,12.(SP)    ;STORE ADR IN PROG'S PC ON STACK

```

```

1388 .IFT
1389 MOV #CKILEX-PG6BA+P6CONS,18.(SP) ;SET UP ABORT RET ADR FOR PROG
1390 BIT #140000,20.(SP) ;WAS USER IN KERNEL MODE?
1391 BNE 10$ ;Y,N-10$
1392 MOV PC,RO ;SET UP KERNEL MODE
1393 ADD #CKILEK-. ,RO ;ABORT RETURN ADR
1394 MOV RO,18.(SP) ;STORE IT INSTEAD
1395 .ENDC
1396 002616' 010300 000302 10$: MOV R3,RO ;GET ADR OF DEV ROUTINE KILL
1397 002620' 062700 ADD #PTLGTH+DKILAD,RO ;ROUT ADR
1398 002624' 005710 TST (RO) ;IS THERE A KILL ROUT IN THE
1399 002626' 001721 BEQ CLSJMP ;DEV ROUT? (Y,N-CLSJMP)
1400 002630' 061000 ADD (RO),RO ;MAKE ITS ADR ABSOLUTE
1401 001 ;
1402 .IF DF MM
1403 MOV SPPAR6,2#KPAR6 ;POINT PAR 6 TO SHARED CODE
1404 .ENDC
1405 002632' 004510 JSR R5,(RO) ;GO TO KILL ROUTINE
1406 002634' 000716 BR CLSJMP ;EXIT
1407
1408 ; *****
1409 ; * OPSW COMMAND *
1410 ; *****
1411 ;
1412 ;
1413 ;
1414 002636' 004567 010630 COPSW: JSR R5,CISDNM ;GO ISOLATE PROG # IN CMND
1415 002642' 005701 TST R1 ;WAS THERE A NUMBER?
1416 002644' 001030 BNE COPSWN ;N,Y-COPSWN
1417 002646' 004567 005362 JSR R5,CSCNCB ;SCAN AND CONVERT OPSW VALUE
1418 002652' 010705 MOV PC,R5 ;SET UP ADR OF CNTRL ROUT TBL
1419 002654' 062705 031342 ADD #CRTBL-. ,R5
1420 002660' 005004 CLR R4 ;RESET PROG FND FLAG
1421 002662' 005715 10$: TST (R5) ;THIS ENTRY IN USE?
1422 002664' 001404 BEQ 20$ ;Y,N-20$
1423 001 .IF DF MM
1424 .IFF
1425 002666' 011501 MOV (R5),R1 ;GET PROGRAM TBL ADR
1426 .IFT
1427 MOV RO,-(SP) ;SAVE OPSW VALUE
1428 MOV (R5),CWA ;STORE PROG'S HI 16 BIT ADR
1429 JSR R5,G22BAD ;GENERATE 22 BIT ADR
1430 .WORD CWA-
1431 JSR PC,CVTVAD ;CONVERT IT TO A VIRTUAL ADR
1432 MOV (SP)+,RO ;RESTORE OPSW VALUE
1433 .ENDC
1434 002670' 010061 000002 MOV RO,POPSW(R1) ;MOVE NEW VALUE TO OPSW
1435 002674' 050504 BIS R5,R4 ;SET PROG FND FLAG
1436 002676' 005725 20$: TST (R5)+ ;INCR TO NEXT TBL ENTRY
1437 002700' 022715 177777 CMP #177777,(R5) ;END OF THE TABLE?
1438 002704' 001366 BNE 10$ ;Y,N-10$
1439 002706' 005704 TST R4 ;WERE PROG'S FOUND?
1440 002710' 001402 BEQ COPER ;Y,N-COPER
1441 002712' 000167 177044 CCFPBJ: JMP CCFMTB ;GO CHECK FOR NO MORE DATA
1442 002716' 004567 003302 COPER: JSR R5,CTYPE ;ISSUE 'NO PROG'S FND' MSG
1443 002722' 000002 .WORD 002

```

```

1444 002724' 004041          .WORD  CERM14-.
1445
1446                          ; ** OPSWN FORMAT **
1447
1448 002726' 010267 176530    COPSWN: MOV    R2,CPNTR          ;SET UP START ADR OF *
1449 002732' 004567 005546    JSR    R5,CVPNUM        ;GO VALIDATE THIS PROG *
1450                          .IF DF MM
1451 JSR    PC,UPAPAR        ;MAP PAR'S TO USER'S PROG AREA
1452                          .ENDC
1453 002736' 005703          TST    R3                ;PROG EXIST FOR THIS *?
1454 002740' 001002          BNE    30$               ;N.Y-30$
1455 002742' 000167 176772    JMP    CCFPDE            ;GO REPORT ERROR
1456 002746' 004567 005262    30$: JSR    R5,CSCNCB      ;SCAN & CONVERT OPSW VALUE
1457 002752' 010063 000002    MOV    R0,POPSW(R3)    ;MOVE OPSW TO THIS PROG
1458 002756' 000755          BR     CCFPBJ           ;EXIT

```



```

1460 .SBTTL RUN, DISPLAY, & REPORT COMMAND ROUTINES
1461
1462 : *****
1463 : * *
1464 : * RUN COMMAND *
1465 : * *
1466 : *****
1467
1468
1469 002760' 032713 000100 CRUN: BIT #OCPRES, (R3) ;OBJ CODE PRESENT FOR THIS PROG?
1470 002764' 001505 BEQ CRUER1 ;Y, N-CRUER1
1471 002766' 005767 031222 TST CSYSFW ;CONS REQ BEING PROCESSED?
1472 002772' 100015 BPL 5$ ;Y, N-5$
1473 002774' 005713 TST (R3) ;THIS PROG ALREADY ACTIVE?
1474 002776' 100013 BPL 5$ ;Y, N-5$
1475 003000' 021467 031652 CMP (R4), CPGCPT ;THIS PROG IN CONTROL?
1476 003004' 001511 BEQ CRUER4 ;N, Y-CRUER4
1477 003006' 032713 000040 BIT #SETDED, (R3) ;DID IT SET "PRT DEV DEDICATED"?
1478 003012' 001405 BEQ 5$ ;Y, N-5$
1479 003014' 042767 060000 031172 BIC #CTUDED+CRUDED, CSYSFW ;RESET SYSTEM'S DEDICATED FLAGS
1480 003022' 042713 000040 BIC #SETDED, (R3) ;RESET PROG'S DEDICATED FLAG
1481 003026' 105063 000034 5$: CLR8 PDPNTR(R3) ;CLEAR DEV # POINTER
1482 003032' 004767 032332 JSR PC, CPGCDN ;POINT AT FIRST DEV #
1483 003036' 111263 000035 MOV8 (R2), PCURDV(R3) ;STORE IT AS CURR DEV
1484 003042' 004567 032336 JSR R5, CPGH5K ;GO DO PROG HOUSEKEEPING
1485 003046' 000001 .WORD 1 ;UNCOND. DO HSKP FLAG
1486 003050' 042713 000036 BIC #URSTOP+ERSTOP+WT4IOT+CTPRIO, (R3) ;RESET STOP & WAIT FLAGS
1487 003054' 052713 100000 BIS #ACTIVE, (R3) ;SET IT'S ACTIVE FLAG
1488 001
1489 .IF DF MM
1490 .ENDC
1491 003060' 005767 005130 JSR PC, MAPUPG ;MAP USER'S PAR'S & PDR'S STORAGE
1492 003064' 100002 TST CSCFWD ;ANOTHER OPERAND IN CMND?
1493 003066' 000167 175376 10$: BPL 15$ ;N, Y-15$
1494 003072' 122777 000101 176362 15$: JMP CGORUN ;GO INITIATE PROG EXECUTION
1495 003100' 001120 CMPB #'A, JCPNTR ;ITS FIRST CHAR AN 'A'?
1496 003102' 004567 005114 BNE COMRET ;Y, N-COMRET
1497 003106' 042713 100000 JSR R5, CSCN ;SCAN OVER THE 'AT'
1498 003112' 005702 BIC #ACTIVE, (R3) ;RESET THE ACTIVE FLAG FOR NOW
1499 003114' 100435 TST R2 ;IS THERE A LINE #?
1500 003116' 004567 005120 BMI CRUER2 ;Y, N-CRUER2
1501 003122' 105720 JSR R5, CSCNCD ;SCAN & CONVERT IT TO PACKED DEC
1502 003124' 001035 TSTB (R0)+ ;LINE # TOO BIG?
1503 003126' 016301 000022 BNE CRUER3 ;N, Y-CRUER3
1504 001 MOV PSRCST(R3), R1 ;GET SOURCE STMENTS START ADR
1505 .IF DF MM
1506 .ENDC BIS #P4CONS, R1 ;ADJ TO KERNEL'S PAR'S
1507 003132' 111102 20$: MOV8 (R1), R2 ;GET STMENT LENGTH
1508 003134' 026110 000002 CMP 2(R1), (R0) ;LINE #'S MATCH?
1509 003140' 001406 BEQ 30$ ;N, Y-30$
1510 003142' 122761 000240 000001 CMPB #240, 1(R1) ;THIS THE 'END' STMENT?
1511 003150' 001423 BEQ CRUER3 ;N, Y-CRUER3
1512 003152' 060201 ADD R2, R1 ;INCR TO NXT STMENT
1513 003154' 000766 BR 20$ ;GO PROCESS NXT STMENT
1514 003156' 016163 000004 000236 30$: MOV 4(R1), PUSRPC(R3) ;STORE THIS STMENT'S START ADR
1515 003164' 052713 100000 BIS #ACTIVE, (R3) ;SET PROG'S ACTIVE FLAG

```

```

1516 003170' 005767 005020          TST      CSCFWD      ;ANOTHER OPERAND IN CMND?
1517 003174' 100734          BMI      10$        ;Y,N-10$ -- GO TO EXECUTION
1518 003176' 000461          BR       COMRET     ;GO CK FOR ANOTHER PROG #
1519
1520
1521 003200' 004567 003020          CRUER1: JSR      R5,CTYPE ;GO ISSUE NO OBJ CCDE MSG
1522 003204' 000006          .WORD   006
1523 003206' 003575          .WORD   CERM15-.
1524 003210' 004567 003010          CRUER2: JSR      R5,CTYPE ;REPORT NO LINE # ERROR
1525 003214' 000046          .WORD   046
1526 003216' 003601          .WORD   CERM16-.
1527 003220' 004567 003000          CRUER3: JSR      R5,CTYPE ;REPORT INV LINE # ERROR
1528 003224' 000116          .WORD   116
1529 003226' 003571          .WORD   CERM16-.
1530 003230' 004567 002770          CRUER4: JSR      R5,CTYPE ;REPORT PROG IN CONTROL
1531 003234' 000006          .WORD   006
1532 003236' 003736          .WORD   CERM30-.
1533
1534
1535          :          *****
1536          :          *
1537          :          * DISPLAY COMMAND *
1538          :          *
1539          :          *****
1540
1541
1542 003240' 016367 000006 004312 CDISPL: MOV      PASCIN(R3),CDISP ;MOVE PROG # TO MSG
1543 003246' 010300          MOV      R3,R0      ;GET ADR OF PROG'S NAME
1544 003250' 062700 000010          ADD      #PNAME,R0
1545 003254' 010701          MOV      PC,R1      ;GET ADR OF PLACE IN MSG
1546 003256' 062701 004306          ADD      #CNAME-,R1
1547 003262' 012021          MOV      (R0)+,(R1)+ ;MOVE NAME TO MSG
1548 003264' 012021          MOV      (R0)+,(R1)+
1549 003266' 011011          MOV      (R0),(R1)
1550 003270' 004567 032304          JSR      R5,CLIST   ;ISSUE DISPLAY HEADER MSG
1551 003274' 004253          .WORD   CDSHDR-.
1552 003276' 000040          .WORD   32.
1553 003300' 005767 004710          TST      CSCFWD     ;ANOTHER OPERAND IN CMND?
1554 003304' 100414          BMI      50$        ;Y,N-50$
1555 003306' 122777 000103 176146  CMPB     #'C,PCPNTR ;IT'S FIRST CHAR A "C"?
1556 003314' 001010          BNE      50$        ;Y,N-50$
1557 003316' 004567 004700          JSR      R5,CSCN    ;SCAN OVER THE WORD "CODE"
1558 003322' 032713 000100          BIT      #OCPRES,(R3) ;OBJ CODE PRESENT?
1559 003326' 001724          BEQ      CRUER1    ;Y,N-CRUER1
1560 003330' 004567 021142          JSR      R5,DISPCD ;GO LIST SOURCE & OBJ CODE
1561 003334' 000402          BR       COMRET     ;RETURN FOR NXT PROG # CHECK
1562 003336' 004567 021126          50$:    JSR      R5,DISP ;GO LIST ONLY SOURCE STMENTS
1563 003342' 000167 176402          COMRET: JMP     CCFMTA ;GO CK FOR ANOTHER PROG #

```

```

1565          : *****
1566          : *
1567          : * REPORT COMMAND *
1568          : *
1569          : *****
1570
1571
1572 003346' 016746 032150 CREPOR: MOV CRPFLG, -(SP) ;SAVE FLAG IN CASE ITS A USER INT
1573 003352' 012767 100003 032142 MOV #100003, CRPFLG ;SET FLG TO DO BOTH REPORTS
1574 003360' 005767 004630 TST CSCFWD ;ANOTHER OPERAND IN CMND?
1575 003364' 100420 BMI 30$ ;Y, N-30$
1576 003366' 122777 000123 176066 CMPB #'S, @CPNTR ;IS ITS FIRST CHAR AN 'S'?
1577 003374' 001003 BNE 20$ ;Y, N-20$
1578 003376' 005367 032120 DEC CRPFLG ;LEAVE STATUS REPORT FLAG SET
1579 003402' 000407 BR 25$ ;GO TO DO REPORT
1580 003404' 122777 000103 176050 20$: CMPB #'C, @CPNTR ;IS IT A 'C'?
1581 003412' 001005 BNE 30$ ;Y, N-30$
1582 003414' 012767 100001 032100 MOV #100001, CRPFLG ;SET COUNTS REPORT FLAG
1583 003422' 004567 004574 25$: JSR R5, CSCN ;SCAN OVER VALID OPERAND
1584 003426' 005763 000300 30$: TST PTLGTH+DERPAD(R3) ;IS THERE AN ADR FOR ERROR REPORT?
1585 003432' 001005 BNE 40$ ;N, Y-40$
1586 003434' 004567 002564 JSR R5, CTYPE ;ISSUE NO REPT AVAIL MSG
1587 003440' 000604 .WORD 604
1588 003442' 004145 .WORD CRENPA-.
1589 003444' 000402 BR 50$ ;GO TO CMND EXIT
1590 001
1591 .IF DF MM
1592 003446' 004767 032032 40$: JSR PC, CGORPT ;GO HAVE REPORT ISSUED
1593 .IFT
1594 40$: MOV SPPAR6, @KPAR6 ;POINT PAR 6 TO SHARED CODE
1595 JSR PC, CGORPT ;GO HAVE REPORT ISSUED
1596 .ENDC
1597 003452' 012667 032044 50$: MOV (SP)+, CRPFLG ;RESTORE ORG FLAG
1598 003456' 000731 BR COMRET ;GO CK FOR NXT PROG *

```

```

1600 .SBTTL FILL COMMAND ROUTINE
1601
1602 :
1603 *
1604 * FILL COMMAND *
1605 *
1606 *****
1607
1608
1609 003460' 122777 000103 176110 CFILL: CMPB #'C, @CDSTAD ;OPERAND = COM?
1610 003466' 001007 BNE 2$ ;Y, N-2$
1611 003470' 010704 MOV PC, R4 ;SET UP COM00 AS START ADR
1612 003472' 062704 036346 ADD #LCOM0-. , R4
1613 003476' 010405 MOV R4, R5 ;SET UP COM09+2 AS END ADR
1614 003500' 062705 000024 ADD #20. , R5
1615 003504' 000412 BR 4$ ;GO CK FOR #
1616 003506' 122777 000102 176062 2$: CMPB #'B, @CDSTAD ;OPERAND = BUF?
1617 003514' 001156 BNE CFIER1 ;Y, N-CFIER1
1618 003516' 010704 MOV PC, R4 ;SET START ADR TO BUF 0 BEGIN &
1619 003520' 062704 035720 ADD #CBUF00-. , R4 ;END ADR TO END OF BUF 15
1620 003524' 010405 MOV R4, R5
1621 003526' 062705 000400 ADD #256. , R5
1622 003532' 004567 007734 4$: JSR R5, CISDNM ;GO ISOLATE BUFFER/COM #
1623 003536' 005701 TST R1 ;WAS THERE A #?
1624 003540' 001435 BEQ 15$ ;Y, N-15$
1625 003542' 004567 036602 JSR R5, ASDECO ;CONVERT IT TO BINARY
1626 003546' 000314 .WORD CFIER2-
1627 003550' 004567 036342 JSR R5, DECBIN
1628 003554' 000306 .WORD CFIER2-
1629 003556' 122777 000103 176012 CMPB #'C, @CDSTAD ;IS THIS "COM" FORMAT?
1630 003564' 001010 BNE 6$ ;Y, N-6$
1631 003566' 020027 000011 CMP R0, #9. ;COM # TOO BIG?
1632 003572' 101133 BHI CFIER2 ;N, Y-CFIER2
1633 003574' 006300 ASL R0 ;MULT # BY 2
1634 003576' 060004 ADD R0, R4 ;SET UP COM START ADR
1635 003600' 010405 MOV R4, R5 ;SET UP COM END ADR
1636 003602' 005725 TST (R5)+
1637 003604' 000413 BR 15$ ;GO PROCESS DATA
1638 003606' 020027 000017 6$: CMP R0, #15. ;BUF # TOO LARGE?
1639 003612' 101123 BHI CFIER2 ;N, Y-CFIER2
1640 003614' 006300 ASL R0 ;COMPUTE BUFFER'S START & END
1641 003616' 006300 ASL R0 ;ADR'S
1642 003620' 006300 ASL R0
1643 003622' 006300 ASL R0
1644 003624' 060004 ADD R0, R4
1645 003626' 010405 MOV R4, R5
1646 003630' 062705 000020 ADD #16. , R5
1647 003634' 004567 004362 15$: JSR R5, CSCN ;SCAN NXT CMND OPERAND
1648 003640' 122777 000127 175730 CMPB #'W, @CDSTAD ;IS IT THE 'WITH' OPERAND?
1649 003646' 001111 BNE CFIER3 ;Y, N-CFIER3
1650 003650' 005702 TST R2 ;ANOTHER OPER PRESENT?
1651 003652' 100513 BMI CFIER4 ;Y, N-CFIER4
1652 003654' 005003 CLR R3 ;CLEAR CHAR CNT
1653 003656' 005002 CLR R2 ;RESET INCLUDE CR/LF FLAG
1654 003660' 122777 000052 175574 CMPB #'*, @CPNTR ;FIRST CHAR OF NXT WD = *?
1655 003666' 001435 BEQ 55$ ;N, Y-55$

```

1656	003670'	122777	000043	175564	CMPB	#', @CPNTR	; IS IT A NBR SIGN?
1657	003676'	001433			BEQ	60\$; N, Y-60\$
1658	003700'	122777	000120	175554	CMPB	#'P, @CPNTR	; IS IT A "P" FOR PAT?
1659	003706'	001505			BEQ	CFIPAT	; N, Y-CFIPAT
1660							
1661						;OCTAL/BINARY FORMAT	
1662							
1663	003710'	004567	004320		20\$: JSR	R5, CSCNCB	; SCAN & CONVERT DATA WORD
1664	003714'	010024			MOV	RO, (R4)+	; MOV DATA TO THE BUFFER
1665	003716'	005723			TST	(R3)+	; ADD 2 TO CHAR CNT
1666	003720'	020405			CMP	R4, R5	; IS THE BUFFER FULL?
1667	003722'	001552			BEQ	CFIEX	; N, Y-CFIEX
1668	003724'	005767	004264		TST	CSCFWD	; ANOTHER DATA WORD PRESENT?
1669	003730'	100367			BPL	20\$; N, Y-20\$
1670	003732'	005703			30\$: TST	R3	; CHAR CNT=0?
1671	003734'	001462			BEQ	CFIER4	; N, Y-CFIER4
1672	003736'	010401			MOV	R4, R1	; SET UP BUF START ADR
1673	003740'	160301			SUB	R3, R1	
1674	003742'	010302			40\$: MOV	R3, R2	; SET UP WORK CHAR CNT
1675	003744'	010100			MOV	R1, RO	; SET UP WORK BUFFER START ADR
1676	003746'	112024			50\$: MOVB	(RO)+, (R4)+	; PROPAGATE A CHARACTER
1677	003750'	020405			CMP	R4, R5	; BUFFER FULL?
1678	003752'	001536			BEQ	CFIEX	; N, Y-CFIEX
1679	003754'	005302			DEC	R2	; DECR CHAR CNT BY 1
1680	003756'	001373			BNE	50\$; CNT=0? (Y, N-50\$)
1681	003760'	000770			BR	40\$; GO START AT BEGINNING OF DATA
1682							
1683						;ASCII FORMAT	
1684							
1685	003762'	005202			55\$: INC	R2	; SET 'INCLUDE CR/LF' FLAG
1686	003764'	005745			TST	-(R5)	; ALLOW ROOM FOR CR/LF
1687	003766'	016700	175470		60\$: MOV	CPNTR, RO	; GET DATA ADR
1688	003772'	005200			INC	RO	; POINT PASS THE "*"/*"
1689	003774'	020405			70\$: CMP	R4, R5	; BUFFER FULL?
1690	003776'	001411			BEQ	80\$; N, Y-80\$
1691	004000'	122710	000015		CMPB	#015, (RO)	; CHAR A CARR RET?
1692	004004'	001406			BEQ	80\$; N, Y-80\$
1693	004006'	122710	000012		CMPB	#012, (RO)	; IS IT A LINE FEED?
1694	004012'	001403			BEQ	80\$; N, Y-80\$
1695	004014'	112024			MOVB	(RO)+, (R4)+	; MOVE CHAR TO THE BUFFER
1696	004016'	005203			INC	R3	; ADD 1 TO CHAR CNT
1697	004020'	000765			BR	70\$; RETURN FOR NXT CHAR
1698	004022'	005702			80\$: TST	R2	; INCLUDE CR/LF?
1699	004024'	001407			BEQ	90\$; Y, N-90\$
1700	004026'	112724	000015		MOVB	#015, (R4)+	; SET CR/LF IN OUTPUT AREA
1701	004032'	112724	000012		MOVB	#012, (R4)+	
1702	004036'	062703	000002		ADD	#2, R3	; INCLUDE THEM IN CHAR CNT
1703	004042'	005725			TST	(R5)+	; RE-ADJUST END ADR
1704	004044'	020405			90\$: CMP	R4, R5	; BUFFER FULL?
1705	004046'	001331			BNE	30\$; Y, N-30\$
1706	004050'	000477			BR	CFIEX	; GO TO CMND EXIT
1707							
1708							
1709	004052'	004567	002146		CFIER1: JSR	R5, CTYPE	; REPORT NO BUF OPERAND
1710	004056'	000142			.WORD	142	
1711	004060'	002647			.WORD	CERM10-	

1712	004062'	004567	002136	CFIER2:	JSR	R5,CTYPE		;REPORT INV BUF #
1713	004066'	000112			.WORD	112		
1714	004070'	002647			.WORD	CERM11-		
1715	004072'	004567	002126	CFIER3:	JSR	R5,CTYPE		;REPORT "WITH" NOT SPEC
1716	004076'	000142			.WORD	142		
1717	004100'	002651			.WORD	CERM12-		
1718	004102'	004567	002116	CFIER4:	JSR	R5,CTYPE		;REPORT NO DATA SPEC
1719	004106'	000042			.WORD	042		
1720	004110'	002650			.WORD	CERM13-		
1721	004112'	004567	002106	CFIER5:	JSR	R5,CTYPE		;REPORT INV PAT #
1722	004116'	000112			.WORD	112		
1723	004120'	003115			.WORD	CERM32-		
1724								
1725								
1726								;PATTERN FORMAT
1727								
1728	004122'	010467	000114	CFIPAT:	MOV	R4,110\$;STORE DATA ADR
1729	004126'	160405			SUB	R4,R5		;COMPUTE BYTE COUNT
1730	004130'	010567	000110		MOV	R5,112\$;STORE BYTE COUNT
1731	004134'	004567	004062		JSR	R5,CSCN		;SCAN THE WORD "PAT"
1732	004140'	004567	007326		JSR	R5,CISDNM		;SCAN WORD FOR DECIMAL #
1733	004144'	005701			TST	R1		;WAS THERE A #?
1734	004146'	001014			BNE	100\$;N,Y-100\$
1735	004150'	122710	000101		CMPB	#'A,(R0)		;IS THIS "PATA"?
1736	004154'	001003			BNE	95\$;Y,N-95\$
1737	004156'	012700	000012		MOV	#10.,R0		;SET UP PATA VALUE
1738	004162'	000417			BR	105\$;GO DO PATA
1739	004164'	122710	000102	95\$:	CMPB	#'B,(R0)		;IS IT "PATB"?
1740	004170'	001350			BNE	CFIERS		;Y,N-CFIERS
1741	004172'	012700	000013		MOV	#11.,R0		;SET UP PATB VALUE
1742	004176'	000411			BR	105\$;GO DO PATB
1743	004200'	004567	036144	100\$:	JSR	R5,ASDECO		;CONVERT # TO BINARY
1744	004204'	177706			.WORD	CFIERS-		
1745	004206'	004567	035704		JSR	R5,DECBIN		
1746	004212'	177700			.WORD	CFIERS-		
1747	004214'	020027	000014		CMP	R0,#LPATEN-LPATOP/2		;PATTERN # TOO LARGE?
1748	004220'	103334			BHIS	CFIERS		;N,Y-CFIERS
1749	004222'	006300		105\$:	ASL	R0		;MULT # BY 2 TO GET DISPL.
1750	004224'	060700			ADD	PC,R0		;INDEX INTO TABLE
1751	004226'	062700	035636		ADD	#LPATOP-.,R0		
1752	004232'	010067	000010		MOV	R0,114\$;STORE ADR OF PATTERN'S REL ADR
1753	004236'	004567	032064		JSR	R5,LFILP		;GO GENERATE THE PATTERN
1754	004242'	000000		110\$:	.WORD	XXXX		;ADDRESS
1755	004244'	000000		112\$:	.WORD	XXXX		;BYTE COUNT
1756	004246'	000000		114\$:	.WORD	XXXX		;REL PATTERN ADR
1757	004250'	000167	174112	CFIEX:	JMP	CNCMND		;RETURN FOR NEXT CMND

```

1759 .SBTTL MEMORY MAP COMMAND ROUTINE
1760
1761 : *****
1762 : * *
1763 : * MEM MAP COMMAND *
1764 : * *
1765 : *****
1766
1767
1768 004254' 004567 001744 CMAP: JSR R5,CTYPE ;ISSUE MEM MAP HEADER MSG
1769 004260' 000600 .WORD 600
1770 004262' 003134 .WORD CMMHDR-.
1771 001 .IF DF MM
1772 TST CSCFWD ;ANOTHER OPERAND IN CMND?
1773 BMI CMPALL ;Y,N-CMPALL
1774 ADD #CMP1PG-CMAP,CR2ADR ;ADJ RETURN ADR OF CCFMTA S/R
1775 JMP CCFMT1 ;GO DO COM PROC FOR PROG #
1776 CMP1PG: MOV R4,-(SP) ;SET 1 PROG MAP FLAG
1777 BR CMPPPG ;GO PROCESS THIS PROG
1778 CMPALL: CLR -(SP) ;RESET 1 PROG MAP FLG (WILL DO ALL)
1779 .ENDC
1780 004264' 010704 MOV PC,R4 ;SET UP CTRL ROUT TBL ADR
1781 004266' 062704 027730 ADD #CRTBL-.,R4
1782 004272' 005714 CMPCKE: TST (R4) ;ENTRY IN USE?
1783 004274' 001005 BNE CMPPPG ;N,Y-CMPPPG
1784 004276' 005724 CMPNXP: TST (R4)+ ;INCR TO NXT ENTRY
1785 004300' 022714 177777 CMP #177777,(R4) ;END OF TBL?
1786 004304' 001372 BNE CMPCKE ;Y,N-CMPCKE
1787 004306' 000436 BR CMPPFM ;GO DO FREE MEM MSGS
1788 004310' 011400 CMPPPG: MOV (R4),RO ;GET PROG START ADR
1789 004312' 004567 034604 JSR R5,BINASC ;CONVERT IT TO ASCII & PUT
1790 004316' 003136 .WORD CMMSTA-. ;IN MSG
1791 001 .IF DF MM
1792 .IFF
1793 004320' 011400 MOV (R4),RO ;GET PROG END ADR
1794 004322' 066000 000726 ADD PLNGTH(RO),RO
1795 004326' 005300 DEC RO
1796 .IFT
1797 MOV (R4),R3 ;SET HI 16 BITS IN CORRECT REG
1798 JSR PC,UPAPAR ;MAP KERNEL PAR'S TO THIS PROG
1799 MOV #40,CMMFPI ;RESET FIXED PROG CHAR IN MSG
1800 MOV CRTXT(R4),RO ;GET PROG'S LENGTH IN 32 WD BLKS
1801 BPL 10$ ;FIXED ADR PROG? (Y,N-10$)
1802 MOV #'F,CMMFPI ;SET AN "F" IN THE MSG
1803 BIC #FIXADR,RO ;RESET FIXED ADR FLG
1804 10$: DEC RO ;ADJ IT
1805 ADD (R4),RO ;ADD IN START ADR
1806 .IFTF
1807 004330' 004567 034566 JSR R5,BINASC ;CONVERT TO ASCII & PLACE
1808 004334' 003132 .WORD CMMEND-. ;IN MSG
1809 .IFF
1810 004336' 011400 MOV (R4),RO ;GET PROG TBL ADR AGAIN
1811 .IFT
1812 MOV R3,RO ;GET PROG TBL ADR AGAIN
1813 .ENDC
1814 004340' 062700 000010 ADD #PNAME,RO ;POINT AT PROG'S NAME

```

1815	004344'	010701		MOV	PC,R1		;GET ADR OF PLACE IN MSG
1816	004346'	062701	003130	ADD	#CMMPNM-.,R1		
1817	004352'	012021		MOV	(R0)+,(R1)+		;MOVE PROG'S NAME TO MSG
1818	004354'	012021		MOV	(R0)+,(R1)+		
1819	004356'	012011		MOV	(R0)+,(R1)		
1820	004360'	016001	000014	MOV	PMDLCD-PNAME-6(R0),R1		;GET PROG'S DEV MDL CODE
1821	004364'	004567	007156	JSR	R5,FDVNAM		;GET DEV NAME FOR THIS MDL CODE &
1822	004370'	003116		.WORD	CMMDL-		;PUT IN THE MSG
1823	004372'	004567	001626	JSR	R5,CTYPE		;GO ISSUE PROG # & START & END
1824	004376'	000404		.WORD	404		;ADR'S & NAME & MDL MSG
1825	004400'	003051		.WORD	CMMSG-		
1826		001		.IF DF	MM		
1827				.IFF			
1828	004402'	000735		BR	CMPNXP		;GO PROCESS NXT PROG
1829				.IFT			
1830				TST	(SP)		;DOING ONLY 1 PROG?
1831				BEQ	CMPNXP		;Y,N-CMPNXP
1832				ADD	#PUPARS,R3		;POINT AT PAR STORAGE IN PROG TBL
1833				MOV	R3,R5		;SET UP ADR OF PDR REGS
1834				ADD	#PUPDRS-PUPARS,R5		
1835				TST	CSYSFW		;CONS REG BEING PROCESSED?
1836				BPL	20\$;Y,N-20\$
1837				TST	CPGCPT		;USER PROG IN CONTROL?
1838				BEQ	20\$;Y,N-20\$
1839				CMP	R4,CPGCPE		;IS IT THIS PROG?
1840				BNE	20\$;Y,N-20\$
1841				MOV	#UPARO,R3		;SET UP USER PAR REG ADR
1842				MOV	#UPDRO,R5		;SET UP USER PDR REG ADR
1843			20\$:	JSR	PC,PRTPAD		;GO PRINT PAR & PDR VALUES
1844				MOV	#P4CONS+PRDIOX,R3		;POINT AT 18/22 BIT ABS ADRS IN PROG TBL
1845				JSR	PC,PRTAAD		;DISPLAY RDIO/WRIO ABS ADRS
1846				BIT	#UNIMAP,CSYSFW		;USING THE UNIBUS MAP?
1847				BEQ	25\$;Y,N-25\$
1848				MOV	#P4CONS,R3		;RESTORE PROG TBL PNTR
1849				TST	PUBMAP(R3)		;THIS PROG USING UNIBUS MAP?
1850				BEQ	25\$;Y,N-25\$
1851				ADD	#PRDIOV,R3		;POINT AT 18 BIT VIRT ADRS
1852				JSR	PC,PRTVAD		;DISPLAY RDIO/WRIO VIRT ADRS
1853				MOV	#P4CONS+PUBMAP,R3		;SET UP NEW TBL PNTR
1854				MOVB	(R3)+,R0		;GET 1ST REG #
1855				MOVB	(R3),R1		;GET # OF REG BITS
1856				CMP	R1,#3		;CK HOW MANY REGS
1857				BLO	24\$;BR IF 1
1858				BEQ	23\$;BR IF 2
1859				SUB	#3,R1		;ADJ FOR 3 REGS
1860			23\$:	DEC	R1		;ADJ FOR 2 REGS
1861			24\$:	JSR	PC,PRTUMR		;DISPLAY PROG'S UNIBUS MAP REGS
1862			25\$:	JSR	R5,CTYPE		;ISSUE A PAIR OF CR/LF'S
1863				.WORD	600		
1864				.WORD	MMCRLF-		
1865			30\$:	TST	CSCFWD		;IS THERE ANOTHER PROG #?
1866				BMI	CMPPFM		;Y,N-CMPPFM
1867				TST	(SP)+		;REMOVE 1 PROG FLAG
1868				JMP	CCFMT1		;GO VALIDATE ITS #
1869				.IFF			
1870	004404'	016700	175036	CMPPFM:	MOV	CFMST,R0	;GET FREE MEM START ADR


```

1871 004410' 010003      MOV      R0,R3      ;SAVE IT
1872 004412' 004567 034504 JSR      R5,BINASC  ;CONVERT TO ASCII & PUT
1873 004416' 003036      .WORD   CMMSTA-    ;IN MSG
1874 004420' 016700 175024 MOV      CFMEND,R0  ;GET END OF FREE MEM ADR
1875 004424' 005200      INC      R0
1876 004426' 010004      MOV      R0,R4      ;SAVE IT
1877 004430' 004567 034466 JSR      R5,BINASC  ;CONVERT TO ASCII & PUT
1878 004434' 003032      .WORD   CMMEND-    ;IN MSG
1879 004436' 004567 001562 JSR      R5,CTYPE   ;ISSUE FREE MEMORY MSG
1880 004442' 000420      .WORD   420
1881 004444' 002771      .WORD   CFMMSG-    ;
1882 004446' 000040      .WORD   32.
1883 004450' 005204      INC      R4          ;ADD 1 TO END OF MEM ADR
1884 004452' 160304      SUB      R3,R4      ;GET SIZE OF "FREE"
1885 004454' 000241      CLC
1886 004456' 006004      ROR      R4          ;DIVIDE IT BY 2
1887 004460' 042704 000001 BIC      #1,R4
1888 004464' 060403      ADD      R4,R3      ;MAKE IT EVEN
1889 004466' 010300      MOV      R3,R0      ;COMPUTE ADR OF "MIDL"
1890 004470' 004567 034426 JSR      R5,BINASC  ;CONVERT IT TO ASCII &
1891 004474' 003027      .WORD   CMIDL-     ;PUT IN THE MSG
1892 004476' 004567 001522 JSR      R5,CTYPE   ;ISSUE "MIDL" ADR MSG
1893 004502' 000400      .WORD   400
1894 004504' 003007      .WORD   CMIDLM-    ;
1895 004506' 000167 173654 JMP      CNCMND     ;RETURN FOR NEXT CMND
1896
1897          CMPPFM: MOV      PC,R3      ;SET UP ADR OF MIDL'S 18/22
1898          ADD      #MIDLAA-.,R3  ;ABS ADRS
1899          JSR      PC,PRTFAA     ;DISPLAY FREE/MIDL ABS ADRS
1900          TST      (SP)+        ;DOING ONLY 1 PROG?
1901          BEQ      CMPPRM       ;Y,N-CMPPRM
1902          BIT      #UNIMAP,CSYSFW ;USING THE UNIBUS MAP?
1903          BEQ      CMPPRM       ;Y,N-CMPPRM
1904          MOV      PC,R3      ;SET UP ADR OF MIDL'S 18
1905          ADD      #MIDLVA-.,R3  ;BIT VIRT ADR
1906          JSR      PC,PRTFVA     ;DISPLAY FREE/MIDL VIRT ADRS
1907          MOV      CFMEND,R2    ;GET FREE MEM END ADR HI 16 BITS
1908          INC      R2
1909          SUB      CFMST,R2     ;ADJ IT
1910          CLR      R1          ;GET SIZE OF FREE
1911          INC      R1          ;INITIALIZE # OF REGS
1912          SUB      #200,R2      ;ADD 1 TO REG CNT
1913          BGT      40$         ;SUB 4K WORDS
1914          MOV      #3,R0       ;FND FREE'S END? (Y,N-40$)
1915          JSR      PC,PRTUMR     ;SET UP FREE'S 1ST REG #
1916          MOV      CFMEND,R0    ;DISPLAY FREE'S UNIMAP REGS
1917          JSR      R5,BINASC     ;GET END OF REAL MEM ADR
1918          .WORD   CMMEMA-     ;CONVERT TO ASCII &
1919          JSR      R5,CTYPE     ;PUT IN THE MSG
1920          .WORD   400
1921          .WORD   CMMERM-     ;DISPLAY REAL MEM END ADR
1922          JMP      CNCMND     ;RETURN FOR NEXT CMND
1923
1924
1925          ;PRINT PAR & PDR VALUES
1926

```

```

1927      ;JSR   PC,PRTPAD      S/R CALL
1928
1929      ;R3 = UPARS' ADR
1930      ;R5 = UPDRS' ADR
1931
1932      ;DESTROYS R0,R1,R2,R3
1933
1934      PRTPAD: MOVB   #'A,CMPAR+3      ;INITIALIZE MSG TO PAR
1935      1CS:  MOV    #B.-,(SP)        ;SET UP REG CNT
1936      MOV    PC,R1                ;SET UP MSG ADR
1937      ADD    #CMPRD-.,R1
1938      2CS:  MOV    (R3)+,R0        ;GET PAR/PDR VALUE
1939      JSR    R5,BINAS1           ;CONVERT DATA TO ASCII & PUT IN MSG
1940      INC    R1                  ;POINT TO NXT FIELD IN MSG
1941      DEC    (SP)               ;DONE ALL REGS?
1942      BNE   2CS                ;Y,N-2CS
1943      TST   (SP)+              ;H$KP STK
1944      JSR   R5,CTYPE           ;DISPLAY THE PAR/PDR MSG
1945      .WORD 400
1946      .WORD CMPAR-
1947      CMPB  #'D,CMPAR+3        ;JUST DO THE PDR MSG?
1948      BEQ   3CS                ;N,Y-3CS
1949      MOVB  #'D,CMPAR+3        ;SET MSG TO PDR
1950      MOV   R5,R3              ;SET UP ADR OF PDR REGS
1951      BR    1CS                ;GO TAILOR & ISSUE PDR MSG
1952      3CS:  RTS    PC            ;EXIT IN-LINE
1953
1954
1955      ;DISPLAY RDIO/WRIO 18/22 BIT ABSOLUTE OR VIRTUAL ADDRESSES
1956
1957      ;JSR   PC,PRTAAD      ABS ADRS S/R CALL
1958      ;JSR   PC,PRTVAD     VIRT ADRS S/R CALL
1959
1960      ;R3 = ADR OF 1ST 18/22 BIT ADR
1961
1962      ;DESTROYS R0,R1,R2,R3
1963
1964      PRTAAD: MOV    #'AB,CMMUAD+2    ;TAILOR MSG TO "ABS"
1965      MOVB   #'S,CMMUAD+4
1966      BR    PRTAVC              ;GO TO COMMON POINT
1967
1968      PRTVAD: MOV    #'UB,CMMUAD+2    ;TAILOR MSG TO "UBM"
1969      MOVB   #'M,CMMUAD+4
1970      PRTAVC: MOV    (R3)+,R2        ;GET RDIO'S 18/22 BIT ADR
1971      MOV    (R3)+,R0
1972      JSR    R5,BINASB           ;CONVERT TO ASCII &
1973      .WORD  CMMUA1-            ;PLACE IN MSG
1974      ADD    #4,R3              ;POINT AT CORRESPONDING WRIO ADR
1975      MOV    (R3)+,R2        ;GET WRIO'S 18/22 BIT ADR
1976      MOV    (R3),R0
1977      JSR    R5,BINASB           ;CONVERT TO ASCII &
1978      .WORD  CMMUA2-            ;PLACE IN MSG
1979      JSR    R5,CTYPE           ;ISSUE RDIO & WRIO ADRS MSG
1980      .WORD 400
1981      .WORD CMMUAD-
1982      RTS    PC                ;EXIT IN-LINE

```

1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038

;DISPLAY FREE/MIDL 18/22 BIT ABSOLUTE OR VIRTUAL ADDRESSES

```

;JSR PC,PRTFAA ABS ADRS S/R CALL
;JSR PC,PRTFVA VIRT ADRS S/R CALL
;R3 = ADR OF MIDL'S ABS OR VIRT ADR
;DESTROYS R0,R1,R2,R3,R4
    
```

```

PRTFAA: MOV #*AB,CFMMSG+2 ;TAILOR MSG TO ABS
        MOVB #*S,CFMMSG+4
        MOV CFMST,R4 ;GET FREE'S ABS START ADR
        BR PRTFAV ;GO TO COMMON POINT

PRTFVA: MOV #*UB,CFMMSG+2 ;TAILOR MSG TO "UBM"
        MOVB #*M,CFMMSG+4

PRTFAV: MOV #600,R4 ;SET UP FREE'S VIRT START ADR
        MOV R4,R0 ;SET UP START ADR
        JSR R5,BINASC ;CONVERT TO ASCII & PUT
        .WORD CMSTA- ;IN THE MSG
        MOV CFMEND,R0 ;GET ABS FREE MEM END ADR
        SUB CFMST,R0 ;GET FREE'S SIZE
        ADD R4,R0 ;ADD IN ABS/VIRT START ADR
        JSR R5,BINASC ;CONVERT TO ASCII & PUT
        .WORD CMEND- ;IN THE MSG
        MOV (R3)+,R2 ;GET MIDL'S ADR
        MOV (R3),R0
        JSR R5,BINASC ;CONVERT TO ASCII & PUT
        .WORD CMIDL- ;IN THE MSG
        JSR R5,CTYPE ;ISSUE 1ST PART OF FREE MEM MSG
        .WORD 420
        .WORD CFMMSG-
        .WORD 47
        JSR R5,CTYPE ;ISSUE "MIDL" PORTION OF THE MSG
        .WORD 400
        .WORD CMIDLM-
        RTS PC ;EXIT IN-LINE
        .PAGE
    
```

;PRINT THE CONTENTS OF 1 TO 3 UNIBUS MAP REGISTERS

```

;JSR PC,PRTUMR S/R CALL
;R0 = 1ST UNIBUS MAP REG #
;R1 = # OF UNIBUS MAP REGS
;DESTROYS R0,R1,R2,R3,R4,R5
    
```

```

PRTUMR: MOV R0,R3 ;SAVE REG #
        MOV R0,R4 ;GET UNIBUS MAP REG ADR
        ASL R4 ;IN R4
        ASL R4
        ADD #UBMAP0,R4
    
```

```

2039          MOV      PC,R5          ;POINT AT TBL WITH MSG ADRS
2040          ADD      #PRTUMA-,R5
2041          MOV      #3,PRTUBC      ;INITIALIZE MSG LENGTH
2042          MOV      R1,-(SP)      ;SET UP LOOP COUNT
2043          PRTULP:  MOV      R3,R0  ;GET REG # IN CORR REG
2044          JSR      PC,CPNAS1     ;CONVERT REG # TO DEC ASCII
2045          MOV      PC,R0        ;GET ADR OF WHERE THE # GCES
2046          PRTUB1:  ADD      (R5)+,R0 ;IN THE MSG
2047          MOV      R1,(R0)      ;STORE IT
2048          MOV      (R4)+,R0     ;GET MAP REG CONTENTS
2049          MOV      (R4)+,R2
2050          MOV      (R5)+,PRTUB2 ;GET ITS MSG ADR
2051          JSR      R5,BINAS8     ;CONVERT MAP REG CONTENTS TO
2052          PRTUB2:  .WORD      XXXX ;ASCII & PLACE IN THE MSG
2053          INC      R3          ;POINT TO NEXT UB MAP REG
2054          ADD      #22.,PRTUBC  ;INCREASE MSG LGTH FOR THIS REG
2055          DEC      (SP)        ;DONE ALL REGS?
2056          BNE     PRTULP      ;Y,N-PRTULP
2057          TST     (SP)+       ;H$KP STK
2058          JSR      R5,CTYPE     ;ISSUE UNIBUS MAP CONTENTS MSG
2059          .WORD      620
2060          .WORD      CMMUBM-.
2061          PRTUBC:  .WORD      XXXX
2062          RTS      PC          ;EXIT IN-LINE
2063
2064          PRTUMA:  .WORD      CMMUB1-PRTUB1
2065          .WORD      CMMUB2-PRTUB2
2066          .WORD      CMMUB3-PRTUB1
2067          .WORD      CMMUB4-PRTUB2
2068          .WORD      CMMUB5-PRTUB1
2069          .WORD      CMMUB6-PRTUB2
2070          .ENDC

```

000

```

2072 .SBTTL /SAVE AND /FETCH COMMAND ROUTINES
2073
2074 : *****
2075 *
2076 * /SAVE COMMAND *
2077 *
2078 : *****
2079
2080
2081 004512' 010300 CSAVE: MOV R3,RO ;SET UP ADR OF NAME IN
2082 004514' 062700 000010 ADD #PNAME,RO ;PROG TBL
2083 004520' 010705 MOV PC,R5 ;SET THE "SAVE" FLAG
2084 004522' 005767 003466 CSVCOM: TST CSCFWD ;ANOTHER OPERAND IN CMND?
2085 004526' 100005 BPL 10$ ;N,Y-10$
2086 004530' 012702 000006 MOV #6,R2 ;SET UP NAME SIZE
2087 004534' 004567 000162 JSR R5,CMVNAM ;MOVE NAME TO COMMON AREA
2088 004540' 000405 BR 20$ ;GO CK CMND TYPE
2089 004542' 004567 000052 10$: JSR R5,CSFCKA ;CHECK FOR WORD "AS"
2090 004546' 100510 BMI CSANNM ;IS THERE A NAME? (Y,N-CSANNM)
2091 004550' 004567 000072 JSR R5,CSACNR ;SCAN NAME & VALIDATE IT
2092 004554' 005705 20$: TST R5 ;THIS THE ENTER CMND?
2093 004556' 001416 BEQ CGOENT ;N,Y-CGOENT
2094 004560' 004567 020116 JSR R5,SAVE ;GO TO SAVE ROUTINE
2095 004564' 000167 173576 JMP CNCMND ;RETURN FOR NEXT CMND
2096
2097
2098 : *****
2099 *
2100 * /FETCH COMMAND *
2101 *
2102 : *****
2103
2104
2105 004570' 004567 000056 CFETCH: JSR R5,CFECNR ;VALIDATE THE NAME
2106 004574' 004567 000020 JSR R5,CSFCKA ;CHECK FOR WORD "AS"
2107 004600' 100503 BMI CSANPN ;PROG # FOLLOW? (Y,N-CSANPN)
2108 004602' 004567 003676 JSR R5,CVPNUM ;VALIDATE PROG #
2109 001 .IF DF MM
2110 JSR PC,UPAPAR ;MAP PAR'S TO USER'S PROG AREA
2111 .ENDC
2112 004606' 052767 040000 174630 CGOENT: BIS #CFTCHF,CRFLGW ;SET THE FETCH FLAG
2113 004614' 000167 175242 JMP CENTPA ;GO TO 'ENTER' CMND ROUTINE
2114
2115 ; ***** SAVE/FETCH COMMON S/R'S *****
2116
2117
2118 004620' 005767 003370 CSFCKA: TST CSCFWD ;ANOTHER WORD IN CMND?
2119 004624' 100455 BMI CSANAS ;Y,N-CSANAS
2120 004626' 004567 003370 JSR R5,CSCN ;SCAN FOR THE WORD "AS"
2121 004632' 122777 000101 174736 CMPB #'A,DCDSTAD ;FIRST CHAR AN "A"?
2122 004640' 001047 BNE CSANAS ;Y,N-CSANAS
2123 004642' 005702 TST R2 ;CHECK FOR ANOTHER OPERAND
2124 004644' 000205 CMVNEX: RTS R5 ;EXIT IN-LINE
2125
2126
2127 004646' 004567 003350 CSACNR: JSR R5,CSCN ;SCAN PROGRAM NAME

```

2128	004652'	020127	000006	CFECNR:	CMP	R1,#6	; IS IT > 6 CHAR'S LONG?
2129	004656'	101050			BHI	CSAINM	; N,Y-CSAINM
2130	004660'	010102			MOV	R1,R2	; SAVE CHAR CNT
2131	004662'	121027	000132	10\$:	CMPB	(R0),#'Z	; IS CHARACTER OF NAME 0-3 OR
2132	004666'	101044			BHI	CSAINM	; A-Z?
2133	004670'	121027	000101		CMPB	(R0),#'A	; Y,N-CSAINM
2134	004674'	103006			BHIS	20\$	
2135	004676'	121027	000071		CMPB	(R0),#'9	
2136	004702'	101036			BHI	CSAINM	
2137	004704'	121027	000060		CMPB	(R0),#'0	
2138	004710'	103433			BLO	CSAINM	
2139	004712'	005301		20\$:	DEC	R1	; DECR CHAR CNT
2140	004714'	001402			BEQ	CMVNAM	; CK ALL CHAR'S? (N,Y-CMVNAM)
2141	004716'	005300			DEC	R0	; POINT AT NXT CHAR
2142	004720'	000760			BR	10\$; RETURN FOR NXT CHAR
2143	004722'	012746	000006	CMVNAM:	MOV	#6,-(SP)	; SET UP MAX # OF CHAR'S
2144	004726'	160216			SUB	R2,(SP)	; GET # OF FILL SPACES
2145	004730'	010701			MOV	PC,R1	; SET UP STORAGE ADR
2146	004732'	062701	173046		ADD	#CLOCZ-,R1	
2147	004736'	112021		40\$:	MOVB	(R0)+,(R1)+	; MOVE CHAR TO OUTPUT STORAGE
2148	004740'	005302			DEC	R2	; DECR CHAR CNT
2149	004742'	001375			BNE	40\$; MOVED ALL CHAR'S? (Y,N-40\$)
2150	004744'	012602			MOV	(SP)+,R2	; GET # OF FILL SPACES
2151	004746'	001736		50\$:	BEQ	CMVNEX	; ANY TO DO? (Y,N-CMVNEX)
2152	004750'	112721	000040		MOVB	#40,(R1)+	; STORE A FILL SPACE
2153	004754'	005302			DEC	R2	; DECR FILL CNT
2154	004756'	000773			BR	50\$; GO CK IF DONE
2155							
2156							
2157	004760'	004567	001240	CSANAS:	JSR	R5,CTYPE	; REPORT NO "AS" ERROR
2158	004764'	000043			.WORD	043	
2159	004766'	002063			.WORD	CERM21-	
2160	004770'	004567	001230	CSANNM:	JSR	R5,CTYPE	; REPORT "NO NAME" ERROR
2161	004774'	000042			.WORD	042	
2162	004776'	002043			.WORD	CERM19-	
2163	005000'	004567	001220	CSAINM:	JSR	R5,CTYPE	; REPORT "INVALID NAME" ERROR
2164	005004'	000013			.WORD	013	
2165	005006'	002033			.WORD	CERM19-	
2166	005010'	004567	001210	CSANPN:	JSR	R5,CTYPE	; ISSUE "NO PROG #" MSG
2167	005014'	000042			.WORD	042	
2168	005016'	001562			.WORD	CERM01-	

```

2170 .SBTTL /DELETE, /LIST, /ZERO, & /BOOT COMMANDS
2171
2172 *****
2173 *
2174 * /DELETE COMMAND *
2175 *
2176 *****
2177
2178 005020' 005767 003170 CSLDEL: TST CSCFWD ;ANOTHER PROG NAME?
2179 005024' 100437 BMI CDLCEX ;Y,N-CDLCEX -- GO TO EXIT
2180 005026' 004567 177614 JSR R5,CSACNR ;SCAN PROG NAME & VALIDATE IT
2181 005032' 004577 173006 JSR R5,DELETE ;GO DELETE THIS PROG
2182 005036' 173330 .WORD CNCMND-
2183 005040' 000767 BR CSLDEL ;CHECK FOR ANOTHER NAME
2184
2185 *****
2186 *
2187 * /LIST COMMAND *
2188 *
2189 *****
2190
2191
2192 005042' 005046 CLST: CLR -(SP) ;RESET "ALL" & "FAST" FLAGS
2193 005044' 005767 003144 2$: TST CSCFWD ;ANOTHER OPERAND IN CMND?
2194 005050' 100417 BMI 10$ ;Y,N-10$
2195 005052' 122777 000101 174402 CMPB #'A,PCPNTR ;IS IT "ALL"?
2196 005060' 001002 BNE 4$ ;Y,N-4$
2197 005062' 052716 000001 BIS #1,(SP) ;SET THE "ALL" FLAG
2198 005066' 122777 000106 174366 4$: CMPB #'F,PCPNTR ;IS IT "FAST"?
2199 005074' 001002 BNE 6$ ;Y,N-6$
2200 005076' 052716 000400 BIS #400,(SP) ;SET THE "FAST" FLAG
2201 005102' 004567 003114 6$: JSR R5,CSCN ;SCAN OVER THE OPERAND
2202 005106' 000756 BR 2$ ;GO CK FOR ANOTHER OPERAND
2203 005110' 012667 000004 10$: MOV (SP)+,20$ ;STORE THE "ALL" FLAG
2204 005114' 004577 172730 JSR R5,LIST ;GO LIST FETCH DEV DIRECTORY
2205 005120' 000000 20$: .WORD XXXX
2206 005122' 173244 .WORD CNCMND-
2207 005124' 000167 173236 CDLCEX: JMP CNCMND ;RETURN FOR NEXT CMND
2208
2209 *****
2210 *
2211 * /ZERO COMMAND *
2212 *
2213 *****
2214
2215
2216 005130' 016700 172670 CZERO: MOV CSAVEI,R0 ;GET ADR OF SAVE DEV INFO
2217 005134' 062700 000004 ADD #4,R0 ;POINT AT CURRENT DEV
2218 005140' 012001 MOV (R0)+,R1 ;GET ITS MDL CODE
2219 005142' 011003 MOV (R0),R3 ;GET ITS UNIT #
2220 005144' 004567 006376 20$: JSR R5,F0VNAM ;FIND ITS DEV NAME & PUT
2221 005150' 002744 .WORD CZWMSG+2- ;INTO THE MSG
2222 005152' 105703 TSTB R3 ;VALID UNIT #?
2223 005154' 100004 BPL 25$ ;N,Y-25$
2224 005156' 012767 020040 002734 MOV #20040,CZWMSG+6 ;WIPE OUT COMMA & UNIT # IN MSG
2225 005164' 000407 BR 30$ ;GO TO TYPE

```

```

2226 005166' 052703 000060      25$:  BIS      #60,R3      ;MAKE UNIT # ASCII
2227 005172' 112767 000054 002720  MOVB     #' ,CZWMSG+6 ;MOVE A COMMA TO THE MSG
2228 005200' 110367 002715      MOVB     R3,CZWMSG+7 ;STORE UNIT # IN MSG
2229 005204' 004567 001014      30$:  JSR      R5,CTYPE  ;ISSUE WARNING MSG
2230 005210' 000420      .WORD    420
2231 005212' 002700      .WORD    CZWMSG-.
2232 005214' 000041      .WORD    33.
2233 005216' 004577 172606      JSR      R5,ACTRD   ;ISSUE READ FOR THE REPLY
2234 005222' 174242      .WORD    CCMRD-.
2235 005224' 000036      .WORD    30.
2236 005226' 004567 002764      JSR      R5,CSCNSP  ;SCAN FOR THE REPLY
2237 005232' 032702 000010      BIT      #CSCFND,R2 ;ANY DATA ENTERED?
2238 005236' 001762      BEQ      30$       ;Y,N-30$
2239 005240' 122777 000116 174330      CMPB     #'N,ACDSTAD ;WAS REPLY AN "N"?
2240 005246' 001726      BEQ      CDLCEX    ;N,Y-CDLCEX
2241 005250' 122777 000131 174320      CMPB     #'Y,ACDSTAD ;WAS REPLY A "Y"?
2242 005256' 001352      BNE      30$       ;Y,N-30$
2243 005260' 004577 172574      JSR      R5,ZERO   ;GO TO ZERO DIRECTORY ROUTINE
2244 005264' 173102      .WORD    CNCMND-.
2245 005266' 000716      BR       CDLCEX    ;RETURN FOR NXT CMND
2246
2247
2248
2249
2250
2251
2252
2253
2254 005270' 004577 172512      CBOOT: JSR      R5,ZBOOT ;GO TO EXEC'S BOOT ROUTINE
2255 005274' 173072      .WORD    CNCMND-.
2256 005276' 000712      BR       CDLCEX    ;RET FOR NXT CMND IF ABORT

```

```

: *****
: *
: * /BOOT COMMAND *
: *
: *****

```



```

2258 .SBTTL ASSIGN COMMAND ROUTINE
2259
2260
2261
2262
2263
2264
2265
2266 005300' 004567 005254 CASSIG: JSR R5,CVLM D1 ;GO VALIDATE MDL NAME & DEV #'S
2267 005304' 016701 174310 MOV CMCDVN,R1 ;GET MODEL CODE
2268 005310' 005767 002700 TST CSCFWD ;ANOTHER OPERAND IN CMND?
2269 005314' 100635 BMI CSANPN ;Y,N-CSANPN
2270 005316' 016700 174140 MOV CPNTR,R0 ;GET ADR OF NXT OPER FIRST CHAR
2271 005322' 122710 000106 CMPB #'F,(R0) ;FIRST CHAR AN "F"?
2272 005326' 001443 BEQ CASFTH ;N,Y-CASFTH
2273 005330' 122710 000123 CMPB #'S,(R0) ;IS IT AN "S"?
2274 005334' 001446 BEQ CASAVE ;N,Y-CASAVE
2275 005336' 122710 000114 CMPB #'L,(R0) ;AN "L"?
2276 005342' 001454 BEQ CASLST ;N,Y-CASLST
2277 005344' 005767 026644 TST CSYSFW ;CONSOLE REQ BEING PROCESSED?
2278 005350' 100002 BPL IOS ;Y,N-IOS
2279 005352' 000167 173376 JMP CRSCMD ;GO REPORT 'RESTRICTED CMND' ERROR
2280 005356' 004567 003122 IOS: JSR R5,CVPNUM ;GO CHECK IT FOR A PROG #
2281 001 OC1 ;IF DF MM
2282 JSR PC,UPAPAR ;MAP PAR'S TO USER'S PROG AREA
2283 000 .ENDC
2284 005362' 005703 TST R3 ;PROG EXIST FOR THIS #?
2285 005364' 001002 BNE IOS ;N,Y-IOS
2286 005366' 000167 174346 JMP CCFPDE ;REPORT PROG DOES NOT EXIST ERROR
2287 005372' 126367 000033 174221 20$: CMPB PMDLC+1(R3),CMCDVN+1 ;SAME DEVICE ROUTINE?
2288 005400' 001123 BNE CASER2 ;Y,N-CASER2
2289 005402' 004567 006006 JSR R5,CSTDVI ;STORE DEV INFO IN PROG TBL
2290 005406' 042713 100000 BIC #ACTIVE,(R3) ;RESET PROG'S ACTIVE FLAG
2291 005412' 122767 000377 174200 CMPB #377,CMCDVN ;IS IT THE "NONE" DEV ROUT?
2292 005420' 001504 BEQ CASEX ;N,Y-CASEX
2293 005422' 010304 MOV R3,R4 ;SET UP ADR OF DEV REG &
2294 005424' 062704 000266 ADD #PTLGTH+DEV DRA,R4 ;INT VECT INFO
2295 005430' 004567 003536 JSR R5,CQADR ;ALLOW DEVICE INFO CHANGE IF NEEDED
2296 005434' 000476 BR CASEX ;RETURN FOR NEXT CMND
2297
2298 ; FETCH ASSIGN
2299
2300 005436' 032701 000004 CASFTH: BIT #04,R1 ;VALID FETCH DEV?
2301 005442' 001476 CASE1J: BEQ CASER1 ;Y,N-CASER1
2302 005444' 016702 172342 MOV CFTCHI,R2 ;SET UP FETCH DEV INFO ADR
2303 005450' 000405 BR CASFCM ;GO TO SAV/FTH COMMON
2304
2305 ; SAVE ASSIGN
2306
2307 005452' 032701 000002 CASAVE: BIT #02,R1 ;VALID SAVE DEV?
2308 005456' 001470 BEQ CASER1 ;Y,N-CASER1
2309 005460' 016702 172340 MOV CSAVEI,R2 ;SET UP SAVE DEV INFO ADR
2310 005464' 005046 CASFCM: CLR -(SP) ;RESET REQ ADR'S FLAG
2311 005466' 016746 035126 MOV CLDID+4,-(SP) ;GET "LOAD" MDL CODE
2312 005472' 000411 BR CASCOM ;GO TO COMMON PROCESSING
2313

```

```

2314          : LIST ASSIGN
2315
2316 005474' 032701 000001 CASLST: BIT #01,R1 ;VALID LIST DEV?
2317 00550J' 001457 BEQ CASER1 ;Y,N-CASER1
2318 005502' 016702 172306 MOV CLISTI,R2 ;SET UP LIST DEV INFO ADR
2319 005506' 012746 000001 MOV #1,-(SP) ;SET REQ ADR'S FLAG
2320 005512' 016746 035066 MOV CKBDID+4,-(SP) ;GET "KYBD/KBD" MDL CODE
2321
2322 005516' 020126 CASCOM: CMP R1,(SP)+ ;ACCEPTABLE DEFAULT NAME?
2323 005520' 001403 BEQ 60$ ;N,Y-60$
2324 005522' 020167 035050 CMP R1,CNONID+4 ;IS THIS A 'NONE' ASSIGN?
2325 005526' 001017 BNE 70$ ;Y,N-70$
2326 005530' 026702 172260 60$: CMP CLISTI,R2 ;IS THIS FOR THE PRINTER?
2327 005534' 001006 BNE 65$ ;Y,N-65$
2328 005536' 032767 004000 026450 BIT #CHPINT,CSYSFW ;INT ENABLE BEING HELD SET?
2329 005544' 001402 BEQ 65$ ;Y,N-65$
2330 005546' 005072 000010 CLR #8,(R2) ;RESET PRINTER'S INT ENB
2331 005552' 012200 65$: MOV (R2)+,R0 ;GET DEFAULT MDL CODE
2332 005554' 012201 MOV (R2)+,R1 ;& UNIT #
2333 005556' 010022 MOV R0,(R2)+ ;STORE THEM AS CURRENT
2334 005560' 110112 MOV# R1,(R2) ;VALUES
2335 005562' 005726 TST (SP)+ ;CLEAN UP STACK
2336 005564' 000422 BR CASEX ;GO TO CMND EXIT
2337 005566' 122701 000377 70$: CMP# #377,R1 ;THIS DEFAULT TYPE OF DEV?
2338 005572' 001421 BEQ CASERO ;N,Y-CASERO
2339 005574' 042701 000360 BIC #360,R1 ;RESET DIFFERENTIATOR BITS IN MDL CD
2340 005600' 062702 000004 ADD #4,R2 ;POINT AT CURR DEV WORDS
2341 005604' 010122 MOV R1,(R2)+ ;STORE DEV'S MDL CODE WORD
2342 005606' 116712 174011 MOV# CMCDVN+3,(R2) ;STORE FIRST DEV #
2343 005612' 005726 TST (SP)+ ;REQUEST DEV ADR'S?
2344 005614' 001406 BEQ CASEX ;Y,N-CASEX
2345 005616' 005722 TST (R2)+ ;POINT AT DEVICE INFO
2346 005620' 010204 MOV R2,R4 ;SET DEV INFO ADR FOR S/R
2347 005622' 004567 002374 JSR R5,CSCN ;SCAN LAST WORD & CK REPLY TERMINATOR
2348 005626' 004567 003340 JSR R5,CRQADR ;GO REQUEST DEV ADR'S
2349 005632' 000167 172530 CASEX: JMP CNCMND ;RETURN FOR NEXT CMND
2350
2351 005636' 005726 CASERO: TST (SP)+ ;TAKE REQ ADR FLG OFF STACK
2352 005640' 004567 000360 CASER1: JSR R5,CTYPE ;GO REPORT INV DEV TYPE
2353 005644' 000012 .WORD 012
2354 005646' 001210 .WORD CERM22-
2355 005650' 004567 000350 CASER2: JSR R5,CTYPE ;TELL HIM HE CAN'T
2356 005654' 000002 .WORD 002 ;CHANGE DEVICES
2357 005656' 001331 .WORD CERM31-

```

.SBTTL RDM & WRM COMMAND ROUTINES

```

2359
2360
2361      :      *****
2362      :      *      *
2363      :      * WRM COMMAND *
2364      :      *      *
2365      :      *****
2366
2367 005660' 005767 002330  CWRM:  TST      CSCFWD      ; IS THERE A DATA OPERAND?
2368 005664' 100002          BPL      CWRMOK      ; N,Y-CWRMOK
2369 005666' 000167 174136  CN20PJ: JMP      CN20ER      ; GO REPORT NO SECOND OPER ERROR
2370 005672' 004567 005546  CWRMOK: JSR     R5,CMEMCK  ; GO CK FOR OUT OF MEM ADR
2371 005676' 004567 002332  JSR     R5,CSCNCB  ; GO CONVERT NXT OPER TO BINARY
2372          C01          .IF DF MM
2373          .IFF
2374 005702' 010024          MOV     RD,(R4)+  ; WRITE WORD TO MEMORY
2375          .IFT
2376          MOV     RD,-(SP) ; SAVE DATA WORD
2377          MOV     R4,RD    ; MOVE ADR TO CORRECT REGS
2378          MOV     R3,R2
2379          JSR     PC,CVTVAD ; CONVERT IT TO A VIRTUAL ADR
2380          MOV     (SP)+,(R1) ; WRITE WORD TO MEMORY
2381          ADD     #2,R4    ; POINT TO NXT MEM LOCATION
2382          ADC     R3
2383          .ENDC
2384 005704' 005767 002304  TST      CSCFWD      ; STILL ANOTHER OPERAND?
2385 005710' 100370          BPL      CWRMOK      ; N,Y-CWRMOK
2386 005712' 000167 172450  JMP      CNCMND      ; RETURN FOR ANOTHER CMND
2387
2388      :      *****
2389      :      *      *
2390      :      * RDM COMMAND *
2391      :      *      *
2392      :      *****
2393
2394          C01          .IF DF MM
2395          .IFF
2396 005716' 010405          CRDM:  MOV     R4,R5    ; SAVE FIRST ADR AS START & END ADR
2397          .IFT
2398  CRDM:  MOV     #1,R5    ; SET CNT TO 1 WORD
2399          .IFTF
2400 005720' 005767 002270  TST      CSCFWD      ; IS THERE A SECOND OPERAND?
2401 005724' 100405          BMI     40$         ; Y,N-40$
2402 005726' 004567 002302  JSR     R5,CSCN88  ; GO SCAN & CONVERT IT TO BINARY
2403 005732' 004567 005406  JSR     R5,CKODAD  ; GO CHECK FOR ODD ADR
2404          .IFF
2405 005736' 010005          MOV     RD,R5      ; SET ADR UP AS END ADR
2406 005740' 010400          40$: MOV     R4,RD      ; SET UP CURR MEM ADR
2407          .IFT
2408          SUB     R4,RD    ; COMPUTE # OF BYTES
2409          SBC     R2
2410          SUB     R3,R2
2411          BMI     40$
2412          MOV     RD,R5
2413          ASR     R5
2414          INC     R5

```

2415			40\$:	MOV	R4,RO	;SET CURRENT ADR IN
2416				MOV	R3,R2	;CORRECT REGS
2417				.IFTF		
2418	005742'	004567	033154	JSR	R5,BINASB	;CONVERT IT TO ASCII &
2419	005746'	001431		.WORD	CRDMG1-	;PUT IN DISPLAY MSG
2420	005750'	004567	005470	JSR	R5,CMEMCK	;GO CK FOR OUT OF MEM ADR
2421				.IFF		
2422	005754'	011400		MOV	(R4),RO	;GET DATA AT CURR MEM ADR
2423				.IFT		
2424				MOV	R4,RO	;MOVE ADR TO CORRECT REGS
2425				MOV	R3,R2	
2426				JSR	PC,CVTVAD	;CONVERT IT TO A VIRTUAL ADR
2427				MOV	(R1),RO	;GET THE DATA WORD
2428				.IFTF		
2429	005756'	004567	033140	JSR	R5,BINASC	;CONVERT IT TO ASCII &
2430	005762'	001425		.WORD	CRMDT-	;PUT IN DISPLAY MSG
2431	005764'	004567	000234	JSR	R5,CTYPE	;ISSUE ADR & DATA DISPLAY
2432	005770'	000400		.WORD	400	;MSG
2433	005772'	001405		.WORD	CRDMG1-	
2434				.IFF		
2435	005774'	020405		CMP	R4,R5	;START ADR >= END ADR?
2436	005776'	103402		BLO	50\$;Y,N-50\$
2437				.IFT		
2438				DEC	R5	;DECR WORD CNT
2439				BNE	50\$;CNT = 0?
2440				.IFTF		
2441	006000'	000167	173756	JMP	CCFMTB	;EXIT
2442				.IFF		
2443	006004'	005724		TST	(R4)+	;INCR CURR MEM ADR
2444	006006'	020405		CMP	R4,R5	;THIS LAST LOC TO DISPLAY?
2445	006010'	001753		BEQ	40\$;N,Y-50\$
2446				.IFT		
2447			50\$:	ADD	#2,R4	;ADD 2 TO THE ADR
2448				ADC	R3	
2449				CMP	R5,#1	;THIS LAST LOC TO DISPLAY?
2450				BEQ	40\$;N,Y-40\$
2451				.IFTF		
2452	006012'	032704	000007	BIT	#7,R4	;THIS ADR AT OCT 10 BOUNDARY?
2453	006016'	001750		BEQ	40\$;N,Y-40\$
2454				.IFF		
2455	006020'	011400		MOV	(R4),RO	;GET DATA AT THIS ADR
2456				.IFT		
2457				MOV	R4,RO	;ADR TO CORRECT REGS
2458				MOV	R3,R2	
2459				JSR	PC,CVTVAD	;CONVERT IT TO A VIRTUAL ADR
2460				MOV	(R1),RO	;GET THE DATA WORD
2461				.IFTF		
2462	006022'	004567	033074	JSR	R5,BINASC	;CONVERT IT TO ASCII &
2463				.IFF		
2464	006026'	001361		.WORD	CRMDT-	;PUT IN DATA MSG
2465				.IFT		
2466				.WORD	CRMD2-	
2467				.IFTF		
2468	006030'	004567	000170	JSR	R5,CTYPE	;DISPLAY DATA ONLY
2469	006034'	000400		.WORD	400	
2470	006036'	001350		.WORD	CRDMG2-	

M04

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 RDM & WRM COMMAND ROUTINES

MACY11 27(732) 24-SEP-76 13:54 PAGE 16-2

SEQ 0052

2471
2472
2473
2474 006040' 000761
2475 000

.IFT
DEC R5 ;DECR WORD COUNT
.IFTF
BR 50\$;GO PROCESS NEXT WCRD
.ENDC

.SBTTL ADD, SUB, BOC, CDB, & CBD COMMANDS

```

2477
2478
2479
2480
2481
2482
2483
2484
2485
2486 006042' 005004 CADD: CLR R4 ;CLEAR ACCUMULATOR REG
2487 001 .IF DF MM
2488 CLR R3
2489 000 .ENDC
2490 006044' 005767 002144 10$: TST CSCFWD ;IS THERE ANOTHER OPERAND?
2491 006050' 100404 BMI CARITH ;Y,N-CARITH
2492 006052' 004567 002156 JSR R5,CSCN88 ;SCAN & CONVERT IT
2493 006056' 060004 ADD R0,R4 ;ADD IT TO RUNNING TOTAL
2494 001 .IF DF MM
2495 ADC R3
2496 ADD R2,R3
2497 000 .ENDC
2498 006060' 000771 BR 10$ ;GO CK FOR ANOTHER OPERAND
2499 006062' 010400 CARITH: MOV R4,R0 ;SET UP ANS FOR S/R
2500 001 .IF DF MM
2501 MOV R3,R2
2502 000 .ENDC
2503 006064' 004567 033032 CARIT1: JSR R5,BINAS8 ;CONVERT TO ASCII
2504 006070' 001634 .WORD CANS-
2505 006072' 004567 000126 CARDSP: JSR R5,CTYPE ;ISSUE ANSWER MSG
2506 006076' 000407 .WORD 400
2507 006100' 001616 .WORD CANSMSG-
2508 006102' 000167 172c50 CARRET: JMP CNCMND ;RET FOR NXT CMND
2509
2510
2511
2512
2513
2514
2515
2516
2517 006106' 004567 002122 CSUB: JSR R5,CSCN88 ;SCAN & CONVERT SUBTRAHEND
2518 006112' 010004 MOV R0,R4 ;SAVE IT
2519 001 .IF DF MM
2520 .IFT
2521 MOV R2,R3
2522 .IFTF
2523 006114' 005000 CLR R0 ;CLEAR MINUEND REG
2524 .IFT
2525 CLR R2
2526 000 .ENDC
2527 006116' 005767 002072 TST CSCFWD ;IS THERE A MINUEND?
2528 006122' 100402 BMI 30$ ;Y,N-30$
2529 006124' 004567 002104 30$: JSR R5,CSCN88 ;SCAN & CONVERT IT
2530 006130' 160400 SUB R4,R0 ;SUBTRACT TWO NUMBERS
2531 001 .IF DF MM
2532 SBC R2

```

B05

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE MACY11 27(732) 24-SEP-76 13:54 PAGE 17-1
DTMGAC.P11 ADD. SUB. BOC. CDB, & CBD COMMANDS

SEQ 0054

2533
2534
2535 006:32' 000754

SUB R3,R2
.ENDC
BR CARIT1

:GO DISPLAY ANSWER

2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592

006134' 005767 002054
006140' 100652
006142' 062704 000002
001
000
006146' 004567 002062
006152' 004567 005166
006155' 160400
001
006160' 006200
000
006162' 000740
006164' 004567 002052
006170' 004567 033722
006174' 000004
001
006176' 000732
000
006200' 004567 000020
006204' 000012
006206' 000724

CBOC:
CDB:
CCDBS:
CCDBER:
CCDBER:

```
*****  
*  
* BRANCH OFFSET CALCULATION COMMAND *  
*  
*****  
TST CSCFWD ;IS THERE A DEST ADR?  
BMI CN20PJ ;Y,N-CN20PJ  
ADD #2,R4 ;POINT INST ADR AT NXT INST  
.IF DF MM  
ADC R3  
.ENDC  
JSR R5,CSCNBB ;CONVERT DEST ADR TO BINARY  
JSR R5,CKODAD ;CK FOR AN ODD ADR  
SUB R4,R0 ;SUB INST ADR FROM DEST ADR  
.IF DF MM  
.IFT  
SBC R2  
SUB R3,R2  
ASR R2 ;DIVIDE ANS BY 2  
ROR R0  
.IFF  
ASR R0 ;DIVIDE ANS BY 2  
.ENDC  
BR CARIT1 ;GO DISPLAY THE ANSWER  
*****  
*  
* CONVERT DECIMAL TO BINARY *  
*  
*****  
JSR R5,CSCNCD ;SCAN & CONVERT TO PKED DEC  
JSR R5,DECBIN ;CONVERT IT TO BINARY  
.WORD CCDBER-  
.IF DF MM  
.IFF  
BR CARIT1 ;GO DISPLAY ANSWER  
.IFT  
JSR R5,BINASC ;CONVERT IT TO ASCII  
.WORD CCDBAN-  
CCDBS: JSR R5,CTYPE ;ISSUE CONV ANSWER MSG  
.WORD 400  
.WORD CCDBMG-  
BR CARRET ;RETURN FOR NEXT CMND  
.ENDC  
CCDBER: JSR R5,CTYPE ;GO ISSUE INV NUM MSG  
.WORD 012  
.WORD CERM24-
```

```
*****  
*  
* CONVERT BINARY TO DECIMAL *  
*
```



```

2593      ;      *
2594      ;      *****
2595      ;
2596      006210' 004567 002020      CCBD: JSR      RS,CSCNCB      ;SCAN & CONVERT # TO BINARY
2597      006214' 004567 032752      JSR      RS,BTASLZ      ;CONVERT BINARY # TO DECIMAL ASCII
2598      001      .IF DF MM
2599      .IFF
2600      006220' 001504      .WORD  CANS-
2601      006222' 000723      BR      CARDSP      ;GO DISPLAY ANSWER
2602      .IFT
2603      .WORD  CCDBAN-
2604      BR      CCDBDS      ;GO DISPLAY ANSWER
2605      .ENDC
          CCC

```

2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662

.SBTTL CONSOLE TERMINAL MESSAGE FORMATTING ROUTINE

THIS ROUTINE FORMATS ALL MESSAGES ISSUED TO THE CONSOLE TERMINAL. IT PROVIDES SEVERAL FUNCTIONS WHICH ARE CONTROLLED BY BITS IN THE SUPPLIED FUNCTION WORD. OPTIONS INCLUDE REMOVING OF A WORD FROM THE STACK, PRECEDING THE MSG WITH '*ER*', PRECEDING THE MSG WITH 'PROG # XX', COMPUTING THE MSG LENGTH OR USING A SUPPLIED COUNT, TYPING OF AN EXTRA CR/LF, DISPLAYING OF THE LAST WORD SCANNED (INVALID DATA) IN THE COMMAND READ-IN AREA, AND EITHER EXITING IN-LINE OR TO AN ADDRESS STORED AT A STANDARD LOCATION. WHEN COMPUTING THE CHAR COUNT, TERMINATION WILL OCCUR ON THE FIRST 00 CHAR. THIS CHAR WILL NOT BE INCLUDED IN THE DATA SENT TO THE TERMINAL. A CR/LF WILL BE ISSUED INSTEAD.

LINKAGE:	JSR	R5,CTYPE	ROUTINE CALL
	.WORD	FUNCWD	FUNCTION WORD
	.WORD	MSGADR-	RELATIVE ADR OF MSG
	.WORD	CHARCNT	OPTIONAL MSG CHAR CNT; THIS IS REQUIRED WHEN 2--4 OF THE FUNCTION WORD IS SET.

FUNCTION WORD FORMAT:

- 2--0 = 1 = REMOVE WORD BEFORE THIS JSR'S R5 FROM STACK
- 2--1 = 1 = PRECEDE MSG WITH '*ER*'
- 2--2 = 1 = PRECEDE MSG WITH 'PROG # XX' AND TAILOR XX TO THE CURRENT PROGRAM NUMBER
- 2--3 = 1 = PRECEDE MSG WITH 'INV'
- 2--4 = 1 = DO NOT COMPUTE MSG LENGTH, USE SUPPLIED COUNT INSTEAD
- 2--5 = 1 = TYPE 'NOT SPECIFIED' AFTER THE MSG
- 2--6 = 1 = DISPLAY INVALID COMMAND DATA
- 2--7 = 1 = TYPE AN EXTRA CR/LF AFTER THE MSG
- 2--8 = 1 = EXIT IN-LINE
- = 0 = EXIT TO ADDRESS STORED AT "CERRET"

BIT LAYOUT:

EXIT IN-LINE	TYPE 'NOT SPEC'	TYPE 'PROG # XX'
TYPE XTRA CR/LF	USE CHAR CNT	TYPE '*ER*'
DISP INV DATA	TYPE 'INV'	CLEAR STACK

DESTROYS REGISTERS: R0,R1,R2

NOTE: WHEN 2--2 OF THE FUNCTION WORD IS SET (TYPE PRG #), R4 MUST CONTAIN THE ADR OF THE CURRENT CONTROL ROUTINE TABLE ENTRY. IT WILL BE LEFT UNCHANGED.

2663									
2664									
2665	006224'	032715	000001		CTYPE: BIT	#1, (R5)			; CLEAR A WORD OFF STACK?
2666	006230'	001401			BEQ	10\$; Y, N-10\$
2667	006232'	012616			MOV	(SP)+, (SP)			; REMOVE WORD BEFORE RETURN ADR
2668	006234'	012546		10\$:	MOV	(R5)+, -(SP)			; SAVE ROUTINE'S FUNCTION WORD
2669	006236'	032716	000002		BIT	#2, (SP)			; ISSUE *ER* AT BEGINNING OF LINE?
2670	006242'	001407			BEQ	20\$; Y, N-20\$
2671	006244'	042767	010000	025742	BIC	#CNTRLO, CSYSFW			; RESET SUPPRESS TYPING FLG IF SET
2672	006252'	004577	171562		JSR	R5, @CTWR			; TYPE THE *ER*
2673	006256'	001102			.WORD	CERID-			
2674	006260'	000005			.WORD	5.			
2675	006262'	032716	000004	20\$:	BIT	#4, (SP)			; ISSUE 'PROG # XX' DATA?
2676	006266'	001410			BEQ	30\$; Y, N-30\$
2677	006270'	004767	005314		JSR	PC, CPNASC			; COMPUTE PROG # IN ASCII
2678	006274'	010167	001074		MOV	R1, CEPNUM			; MOVE PROG # TO MSG
2679	006300'	004577	171534		JSR	R5, @CTWR			; TYPE 'PROG # XX' DATA
2680	006304'	001061			.WORD	CPNMSG-			
2681	006306'	000012			.WORD	10.			
2682	006310'	032716	000010	30\$:	BIT	#10, (SP)			; ISSUE 'INV' AT BEGINNING OF MSG?
2683	006314'	001404			BEQ	33\$; Y, N-33\$
2684	006316'	004577	171516		JSR	R5, @CTWR			; TYPE 'INV' DATA
2685	006322'	000721			.WORD	CIVMSG-			
2686	006324'	000004			.WORD	4			
2687	006326'	010500		33\$:	MOV	R5, R0			; GET MSG START ADR AND
2688	006330'	062500			ADD	(R5)+, R0			; MAKE IT ABSOLUTE
2689	006332'	010001			MOV	R0, R1			; SAVE START ADR
2690	006334'	010702			MOV	PC, R2			; MAKE IT RELATIVE TO THIS
2691	006336'	062702	000042		ADD	#50\$-, R2			; ROUTINE AND STORE IN CTWR
2692	006342'	160200			SUB	R2, R0			; LINK INFO
2693	006344'	010022			MOV	R0, (R2)+			
2694	006346'	032716	000020		BIT	#20, (SP)			; MSG CHAR CNT SUPPLIED?
2695	006352'	001402			BEQ	35\$; Y, N-35\$
2696	006354'	012512			MOV	(R5)+, (R2)			; STORE SUPPLIED CNT IN LINK INFO
2697	006356'	000406			BR	45\$; GO ISSUE MSG
2698	006360'	005000		35\$:	CLR	R0			; CLEAR CHAR CNT REG
2699	006362'	005200		40\$:	INC	R0			; INCR CHAR CNT BY 1
2700	006364'	105721			TSTB	(R1)+			; IS MSG CHAR A 00?
2701	006366'	001375			BNE	40\$; Y, N-40\$
2702	006370'	005300			DEC	R0			; DON'T COUNT THE 00 BYTE
2703	006372'	010012			MOV	R0, (R2)			; STORE COMPUTED CNT IN LINK INFO
2704	006374'	004577	171440	45\$:	JSR	R5, @CTWR			; ISSUE MSG SPECIFIED
2705	006400'	000000		50\$:	.WORD	XXXX			
2706	006402'	000000			.WORD	XXXX			
2707	006404'	032716	000040		BIT	#40, (SP)			; ISSUE 'NOT SPECIFIED' AFTER THE MSG?
2708	006410'	001404			BEQ	53\$; Y, N-53\$
2709	006412'	004577	171422		JSR	R5, @CTWR			; TYPE 'NOT SPECIFIED' DATA
2710	006416'	000631			.WORD	CNSMSG-			
2711	006420'	000016			.WORD	14.			
2712	006422'	032716	000020	53\$:	BIT	#20, (SP)			; WAS CHAR CNT SUPPLIED?
2713	006426'	001004			BNE	56\$; N, Y-56\$
2714	006430'	004577	171404		JSR	R5, @CTWR			; ISSUE A CR/LF
2715	006434'	000660			.WORD	CECMG-			
2716	006436'	000002			.WORD	2			
2717	006440'	032716	000100	56\$:	BIT	#100, (SP)			; ISSUE INVALID DATA MSG?
2718	006444'	001437			BEQ	90\$; Y, N-90\$

2719	006446'	016700	173124	MOV	COSTAD, R0	; GET ADR OF CMND WORD
2720	006452'	016701	173122	MOV	CDCNT, R1	; GET # OF CHARS IN WORD
2721	006456'	012702	000014	MOV	#12, R2	; SET UP MAX COUNT OF DEC 12
2722	006462'	020102		CMP	R1, R2	; WORD LENGTH > 12?
2723	006464'	101401		BLOS	60\$; Y, N-60\$
2724	006466'	010201		MOV	R2, R1	; SET WORD LENGTH TO 12
2725	006470'	010546	60\$:	MOV	R5, -(SP)	; SAVE WORK REG
2726	006472'	010705		MOV	PC, R5	; GET MSG STORAGE ADR
2727	006474'	062705	000604	ADD	#CERDTA-, R5	
2728	006500'	005701		TST	R1	; IS WORD LENGTH = 0?
2729	006502'	001404		BEQ	70\$; N, Y-70\$
2730	006504'	112025	65\$:	MOVB	(R0)+, (R5)+	; MOVE WORD CHAR TO ERROR MSG
2731	006506'	005302		DEC	R2	; DECR MAX COUNT
2732	006510'	005301		DEC	R1	; DECR WORD CNT
2733	006512'	001374		BNE	65\$; WD CNT = 0? (Y, N-65\$)
2734	006514'	005702	70\$:	TST	R2	; REMAINING MAX CNT = 0?
2735	006516'	001405		BEQ	80\$; N, Y-80\$
2736	006520'	012701	000040	MOV	#40, R1	; SET UP SPACE CHAR
2737	006524'	110125	75\$:	MOVB	R1, (R5)+	; FILL AFTER DATA WITH A SPACE
2738	006526'	005302		DEC	R2	; DECR MAX CNT
2739	006530'	001375		BNE	75\$; MAX CNT = 0? (Y, N-75\$)
2740	006532'	012605	80\$:	MOV	(SP)+, R5	; RESTORE WORK REG
2741	006534'	004577	171300	JSR	R5, @CTWR	; DISPLAY INVALID DATA
2742	006540'	000525		.WORD	CERMSG-	
2743	006542'	000031		.WORD	25.	
2744	006544'	032716	000200	BIT	#200, (SP)	; ISSUE EXTRA CR/LF?
2745	006550'	001404		BEQ	95\$; Y, N-95\$
2746	006552'	004577	171262	JSR	R5, @CTWR	; TYPE A CR/LF
2747	006556'	000536		.WORD	CECMG-	
2748	006560'	000002		.WORD	2	
2749	006562'	032726	000400	BIT	#400, (SP)+	; EXIT IN-LINE?
2750	006566'	001401		BEQ	100\$; Y, N-100\$
2751	006570'	000205		RTS	R5	; EXIT IN-LINE
2752						
2753	006572'	012605	100\$:	MOV	(SP)+, R5	; CLEAR REG R5 OFF THE STACK
2754	006574'	000177	173002	JMP	@CERRÉT	; EXIT TO PRE-STORED RETURN ADR

```

2756
2757 ;      **** MESSAGE STORAGE AREA ****
2758
2759
2760
2761      .NLIST BEX
2762
2763 006600' 051120 043517 021440 CERM01: .ASCIZ /PROG #/
2764 006607'      104 052101 020101 CERM02: .ASCIZ /DATA CONV ERR/
2765 006625'      122 051505 051124 CERM03: .ASCII /RESTRICTED /
2766 006640' 046503 042116      000 CERM04: .ASCIZ /CMND/
2767 006645'      104 042517 047123 CERM05: .ASCIZ /DOESN'T EXIST/
2768 006663'      125 047116 041505 CERM06: .ASCII /UNNEC /
2769 006671'      117 042520 040522 CERM07: .ASCIZ /OPERAND/
2770 006701'      062 042116 047440 CERM08: .ASCIZ /2ND OPERAND/
2771 006715'      101 051104 037040 CERM09: .ASCIZ /ADR > MEM/
2772 006727'      102 043125 041457 CERM10: .ASCIZ 'BUF/COM'
2773 006737'      102 043125 041457 CERM11: .ASCIZ 'BUF/COM #'
2774 006751'      047 044527 044124 CERM12: .ASCIZ /'WITH'/
2775 006760' 040504 040524      000 CERM13: .ASCIZ /DATA/
2776 006765'      116 020117 051120 CERM14: .ASCIZ /NO PROG'S FND/
2777 007003'      116 020117 041117 CERM15: .ASCIZ /NO OBJ CODE/
2778 007017'      114 047111 020105 CERM16: .ASCIZ /LINE #/
2779 007026' 047125 052111 021440 CERM17: .ASCIZ /UNIT #/
2780 007035'      115 046104      040 CERM18: .ASCII /MDL /
2781 007041'      116 046501 000105 CERM19: .ASCIZ /NAME/
2782 007046' 051102      000 CERM20: .ASCIZ /BR/
2783 007051'      047 051501 000047 CERM21: .ASCIZ /'AS'/
2784 007056' 042504 020126 054524 CERM22: .ASCIZ /DEV TYPE/
2785 007067'      111 051516 043125 CERM23: .ASCIZ 'INSUF MEM FOR DEV ROUT & I/O AREAS'
2786 007132' 052516 041115 051105 CERM24: .ASCIZ /NUMBER/
2787 007141'      117 042104 040440 CERM25: .ASCIZ /ODD ADR/
2788 007151'      101 051104 051040 CERM26: .ASCIZ /ADR RANGE/
2789 007163'      111 040516 052103 CERM27: .ASCIZ /INACTIVE/
2790 007174' 047111 041440 047117 CERM30: .ASCIZ /IN CONTROL/
2791 007207'      103 047101 052047 CERM31: .ASCIZ /CAN'T CHANGE DEV TYPE/
2792 007235'      120 052101 021440 CERM32: .ASCIZ /PAT #/
2793      001
2794      CERM33: .ASCIZ /RDIO+WRIO > 24,250/
2795      CERM34: .ASCIZ /PROG AREA IN USE/
2796      CERM35: .ASCIZ /ADR WITHIN MPG/
2797      CERM36: .ASCIZ /'TO'/
2798      CERM37: .ASCIZ /DESTINATION/
2799      CERM38: .ASCIZ /INSUF ROOM FOR PROG/
2800      CERM39: .ASCIZ /REG # (SHD BE 7 - 28,29,30)/
2801      CERM40: .ASCIZ /INSUF REGS AVAIL/
2802      CERM41: .ASCIZ /REGS IN USE/
2803      CERM42: .ASCIZ 'WARNING!! PROG I/O AREA > 18 BIT ADR - MAY NOT RUN'
2804      000      .ENDC
2805
2806
2807
2808 007243'      111 053116      040 CIVMSG: .ASCII /INV /
2809 007247'      040 047516 020124 CNSMSG: .ASCII / NOT SPECIFIED/
2810 007265'      111 053116 042040 CERMSG: .ASCII /INV DATA = /
2811 007300' 000014
;MUST PRECEDE CECMSG
    
```

```

2812 007314' 005015 047105 042524 CECMSG: .ASCIZ <015><012>/ENTER CMND/
2813 007331' 015 025012 052 CASMSG: .ASCII <015><012>/**/
2814 001 .IF DF MM
2815 .IFF
2816 007335' 130 054130 054130 CAAMSG: .ASCIZ /XXXXXX ADJ TO EVEN/
2817 .IFT
2818 CAAMSG: .ASCIZ /XXXXXXXX ADJ TO EVEN/
2819 .ENDC
2820 000 .EVEN
2821 007360' 042452 025122 040 CERID: .ASCII /*ER* /
2822 007365' 120 047522 020107 CPNMSG: .ASCII /PROG # /
2823 007374' 054130 040 CEPNUM: .ASCII /XX / ;WD BNDRY
2824 001 .IF DF MM
2825 .IFF
2826 007377' 130 054130 054130 CRDMG1: .ASCII /XXXXXX=/
2827 007406' 011 CRDMG2: .BYTE 011
2828 007407' 130 054130 054130 CRDMDT: .ASCIZ /XXXXXX/
2829 .IFT
2830 CRDMG1: .ASCII /XXXXXXXXX= /
2831 CRDMDT: .ASCIZ /XXXXXX/
2832 CRDMG2: .ASCII <011>/ /
2833 CRDMD2: .ASCIZ /XXXXXX/
2834 000 .ENDC
2835 007416' 005015 020052 042515 CMMHDR: .ASCIZ <015><012> /* MEM MAP */
2836 001 .IF DF MM
2837 .IFF
2838 .ODD
2839 007435' 015 043012 042522 CFMMSG: .ASCII <015><012>/FREE MEM /
2840 .IFT
2841 .EVEN
2842 CFMMSG: .ASCII <015><012>/XXX ADRS: FREE MEM /
2843 .ENDC
2844 007451' 075 020040 CMMMSG: .ASCII /= /
2845 001 .IF DF MM
2846 .IFF
2847 007454' 054130 054130 054130 CMMSTA: .ASCII /XXXXXX TO /
2848 007466' 054130 054130 054130 CMMEND: .ASCII /XXXXXX /
2849 .IFT
2850 CMMSTA: .ASCII /XXXXXXXX00 TO /
2851 CMMEND: .ASCII /XXXXXXXX77 /
2852 CMMFPI: .ASCII /X /
2853 000 .ENDC
2854 007476' 054130 054130 054130 CMMPNM: .ASCII /XXXXXX /
2855 007506' 054130 054130 000 CMMMDL: .ASCIZ /XXXX/
2856 007513' 050 044515 046104 CMIDLM: .ASCII /(MIDL = /
2857 001 .IF DF MM
2858 .IFF
2859 007523' 130 054130 054130 CMIDL: .ASCIZ /XXXXXX)/
2860 .IFT
2861 CMIDL: .ASCIZ /XXXXXXXXX)/
2862 CMMERM: .ASCII <015><012>/END OF MEM = /
2863 CMMEMA: .ASCIZ /XXXXXXXX77/
2864 CMMPAR: .ASCII <015><012>/PXR'S 0 - 7 = /
2865 CMMPRD: .ASCII /XXXXXX XXXXXX XXXXXX XXXXXX /
2866 .ASCII /XXXXXX XXXXXX /
2867 TRPSP: .ASCIZ /XXXXXX XXXXXX/

```

```

2868 .EVEN
2869 CMMUAD: .ASCII <015><012>/XXX ADRS: RDIO = /
2870 CMMUA1: .ASCII /XXXXXXXX WRIO = /
2871 CMMUA2: .ASCIZ /XXXXXXXX/
2872 .EVEN
2873 CMMUBM: .ASCII <015><012>/UBM REG # /
2874 CMMUB1: .ASCII /XX = /
2875 CMMUB2: .ASCII /XXXXXXXX REG # /
2876 CMMUB3: .ASCII /XX = /
2877 CMMUB4: .ASCII /XXXXXXXX REG # /
2878 CMMUB5: .ASCII /XX = /
2879 CMMUB6: .ASCII /XXXXXXXX/
2880 .ENDC
2881 007533' 000 051501 047107 CADMSG: .ASCII /?ASGN DEV: /
2882 007547' 007547' .ODD
2883 007547' 015 050012 047522 CDSHDR: .ASCII <015><012>/PROG # /
2884 007560' 054130 024040 CDISP: .ASCII /XX (/
2885 007564' 054130 054130 054130 CDNAME: .ASCII /XXXXXX) DISPLAY/<015><012><015><012>
2886 007607' 116 020117 042522 CRENPA: .ASCIZ /NO REPT AVAIL/
2887 007625' 077 042504 020126 CRQDRM: .ASCII /?DEV REG = /
2888 007640' 054130 054130 054130 CRQDRD: .ASCII 'XXXXXX / '
2889 007651' 077 047111 020124 .ASCII /?INT VEC = /
2890 007664' 054130 054130 054130 .ASCII 'XXXXXX / '
2891 007675' 077 052502 020123 .ASCII /?BUS REQ = /
2892 007710' 026130 020130 020057 .ASCII 'X,X / '
2893 007716' 047101 020123 020075 CANSMSG: .ASCII /ANS = /
2894 001 .IF DF MM
2895 .IFF
2896 007724' 054130 054130 054130 CANS: .ASCIZ /XXXXXX/
2897 .IFT
2898 CANS: .ASCIZ /XXXXXXXXXX/
2899 CCDBMG: .ASCII /ANS = /
2900 CCDBAN: .ASCIZ /XXXXXX/
2901 .ENDC
2902 007733' 000 042412 052116 CESMSG: .ASCIZ <015><012>/ENTER STMT'S/
2903 007753' 015 050012 046507 CPCMRP: .ASCIZ <015><012>'PGMS COMPILED / MEM REFORMATTED'
2904 010016' 010016' .EVEN
2905 010016' 005015 052101 044440 CINTMG: .ASCII <015><012>/AT INT: PC= /
2906 010034' 054130 054130 054130 CINTPC: .ASCII /XXXXXX; SP= /
2907 010050' 054130 054130 054130 CINTSP: .ASCII /XXXXXX; PROG= /
2908 001 .IF DF MM
2909 .IFF
2910 010066' 054130 000130 CINTPN: .ASCIZ /XXX/
2911 .IFT
2912 CINTPN: .ASCII /XXX /
2913 CINTMD: .ASCIZ /XXXX MODE/
2914 .ENDC
2915 000 .ODD
2916 010073' 007 042122 047511 CIOSZM: .ASCII '?RDIO = 256 / '
2917 010112' 010112' .EVEN
2918 010112' 020077 054130 054130 CZWMSG: .ASCII '? XXXX,X - ARE YOU SURE? (Y/N) / '
2919 010154' 010154' .EVEN
2920 010154' 005015 042504 020126 FLLDMG: .ASCII <015><012>'DEV ROUT "'
2921 010170' 054130 054130 054130 FLLDNM: .ASCIZ 'XXXXXX.MPG" LOADED'
2922 001 .IF DF MM
2923 ADRREQ: .ASCII '?PROG AREA ADR / '

```

2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949

000
010214'

UMRGRQ: .ASCII <015><012>'?UBMAP REG # (1ST OF '
UMNREG: .ASCII 'X) / '
.EVEN
UMRGUD: .ASCII <015><012>/1ST REG USED IS /
UMRGNM: .ASCIZ /XX/
TRPMSG: .ASCII <015><012>/*ER* TRAP AT VECTOR /
TRPADR: .ASCII /XXXXXX/<015><012><015><012>
.ASCII /AT TIME OF TRAP:/<015><012><015><012>/REG'S/
TRPPSM: .ASCII <015><012>/PSW= /
TRPPSW: .ASCII /XXXXXX/<015><012><015><012>/MMR0= /
TRPMR0: .ASCII /XXXXXX MMR2= /
TRPMR2: .ASCII /XXXXXX/
TRP4OM: .ASCII / /<015><012><015><012>/MMR1= /
TRPMR1: .ASCII /XXXXXX MMR3= /
TRPMR3: .ASCIZ /XXXXXX/
TRPFSM: .ASCII <015><012>/FAILING MODE:/
MMCRLF: .ASCIZ / /
TRPISV: .ASCIZ <015><012>/INT BEING SERVICED/
.ODD
TRPPMG: .ASCII <015><012>/CULPRIT IS USER PROG # /
TRPPNM: .ASCIZ /XX/
TRPMPG: .ASCIZ <015><012>/CULPRIT IS MPG/
.ENDC
.EVEN
.LIST BEX

.SBTTL SCAN ENTERED COMMAND DATA S/R

2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001

THIS S/R WILL SCAN THE DATA IN THE COMMAND FROM ITS CURRENT POSITION TO THE END OF THE NEXT WORD OR TO THE END OF THE LINE, WHICHEVER COMES FIRST. IF A WORD IS FOUND, SCANNING WILL CONTINUE TO DETERMINE IF THERE IS ANOTHER WORD OR IF THE ONE FOUND WAS THE LAST. OPTIONAL ENTRY POINTS PROVIDE FOR AUTOMATIC CONVERSION OF THE FOUND WORD TO EITHER PACKED DECIMAL OR BINARY (PACKED OCTAL). RETURN WILL BE IN-LINE UNLESS A CONVERSION IS SPECIFIED AND AN ERROR OCCURS ON THE CONVERSION. IN THIS CASE, RETURN WILL BE MADE TO THE ERROR ROUTINE.

LINKAGE: JSR R5,CSCN SCAN FOR NEXT WORD
,CSCNSP SET DATA PNTR & SCAN ONLY
,CSCNMS SCAN & CK FOR MINUS SIGN
,CSCNBB OR
,CSCNCB SCAN & CONVERT TO BINARY
,CSCNCD SCAN & CONVERT TO PACKED DECIMAL

DESTROYS REGISTERS: R0,R1,R2

OUTPUT INFO:

CDSTAD = START ADR OF WORD IN CMND DATA
CPNTR = START ADR OF NEXT WORD IN CMND OR CR/LF ADR
CDCNT = NUMBER OF CHARACTERS IN WORD FOUND
DECBUF = START ADR OF 3 BYTES OF DATA IN PACKED DECIMAL
BINARY = ADR OF WORD WITH DATA IN BINARY

CSCN &
CSCNSP EXIT: R0 CONTAINS ADR OF RIGHTMOST DATA BYTE IN WORD
R1 CONTAINS # OF CHAR'S IN WORD
R2 CONTAINS CSCN FLAGWORD (CSCFWD)
CSCNCB EXIT: R0 CONTAINS BINARY WORD
CSCNBB EXIT: R0 & R2 CONTAINS 22 BIT BINARY WORD
CSCNCD EXIT: R0 CONTAINS ADR OF 'DECBUF' (DATA STORAGE)

010214' 000000

CSCFWD: .WORD 0 ;SCAN S/R FLAG WORD
; = 1 ; CONVERT TO BINARY
; = 2 ; CONVERT TO PACKED DECIMAL
; = 4 ; STOP SCAN ON MINUS SIGN
CSCFND= 10 ; DATA FOUND
CSCTCS= 20 ; TRAILING COMMA/SPACE
CSCTMS= 40 ; TRAILING MINUS SIGN
CSCRLF= 100000 ; CR/LF TERMINATOR

000010
000020
000040
100000

M05

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 SCAN ENTERED COMMAND DATA S/R

MACY11 27(732) 24-SEP-76 13:54 PAGE 19-1

SEQ 0065

```

3003                                     ;SET UP DATA POINTER & SCAN ONLY ENTRY POINT
3004
3005 010216' 004767 003106      CSCNSP: JSR      PC,CSUPTR          ;INITIALIZE READ DATA POINTER
3006
3007                                     ;SCAN ONLY ENTRY POINT
3008
3009 010222' 005002      CSCN:   CLR      R2          ;RESET OCTAL & DEC CONV FLAGS
3010 010224' 000410      BR       C$CCOM          ;GO TO COMMON PROCESSING
3011 010226' 012702 000004      CSCNMS: MOV     #4,R2          ;SU FOR CK OF MINUS SIGN
3012 010232' 000405      BR       C$CCOM          ;GO TO COMMON POINT
3013
3014                                     ;SCAN AND CONVERT TO 16 BIT BINARY (PACKED OCTAL) ENTRY POINT
3015
3016                                     .IF NDF MM
3017 010234'          CSCNBB:
3018          000          .ENDC
3019 010234' 012702 000001      CSCNCB: MOV     #1,R2          ;SET OCTAL FLAG & RESET DECIMAL
3020 010240' 000402      BR       C$CCOM          ;GO TO COMMON PROCESSING
3021
3022                                     ;SCAN AND CONVERT TO PACKED DECIMAL ENTRY POINT
3023
3024 010242' 012702 000002      CSCNCD: MOV     #2,R2          ;SET DECIMAL FLAG & RESET OCTAL
3025 010246' 005001      C$CCOM: CLR     R1          ;CLEAR CHARACTER COUNT
3026 010250' 016700 171206      MOV     CPNTR,R0          ;GET CURR POSITION ADR OF DATA
3027 010254' 122710 000040      10$:   CMPB   #40,(R0)          ;DATA CHAR A SPACE?
3028 010260' 001431      BEQ     40$          ;(N,Y-40$)
3029 010262' 122710 000054      CMPB   #54,(R0)          ;IS IT A COMMA?
3030 010266' 001426      BEQ     40$          ;(N,Y-40$)
3031 010270' 122710 000015      CMPB   #15,(R0)          ;IS IT A CARR RETURN?
3032 010274' 001442      BEQ     60$          ;(N,Y-60$)
3033 010276' 122710 000012      CMPB   #12,(R0)          ;IS IT A LINE FEED?
3034 010302' 001437      BEQ     60$          ;(N,Y-60$)
3035 010304' 122710 000055      CMPB   #55,(R0)          ;IS IT A MINUS SIGN?
3036 010310' 001423      BEQ     50$          ;N,Y-50$
3037 010312' 032702 000010      15$:   BIT     #C$CFND,R2          ;'DATA FOUND' FLAG SET?
3038 010316' 001004      BNE     20$          ;(N,Y-20$)
3039 010320' 052702 000010      BIS     #C$CFND,R2          ;SET THE 'FND' FLAG
3040 010324' 010067 171246      MOV     R0,CD$TAD          ;SAVE DATA START ADDRESS
3041 010330' 032702 000060      20$:   BIT     #C$CTCS+C$CTMS,R2          ;'TRAILING COMMA/SPACE/- SIGN' FLAGS SET?
3042 010334' 001024      BNE     70$          ;(N,Y-70$)
3043 010336' 005201      INC     R1          ;ADD 1 TO CHAR COUNT
3044 010340' 005200      30$:   INC     R0          ;ADD 1 TO DATA ADR
3045 010342' 000744      BR      10$          ;GO CHECK NEXT CHAR
3046 010344' 032702 000010      40$:   BIT     #C$CFND,R2          ;'FND' FLAG SET?
3047 010350' 001773      BEQ     30$          ;(Y,N-30$)
3048 010352' 052702 000020      BIS     #C$CTCS,R2          ;SET 'TRAILING COMMA/SPACE' FLAG
3049 010356' 000770      BR      30$          ;GO CHECK NEXT CHAR
3050 010360' 032702 000004      50$:   BIT     #4,R2          ;LOOKING FOR A MINUS SIGN?
3051 010364' 001752      BEQ     15$          ;Y,N-15$
3052 010366' 032702 000010      BIT     #C$CFND,R2          ;'FND' FLAG SET?
3053 010372' 001762      BEQ     30$          ;Y,N-30$
3054 010374' 052702 000040      BIS     #C$CTMS,R2          ;SET TRAILING MINUS SIGN FLG
3055 010400' 000757      BR      30$          ;GO CK NEXT CHAR
3056 010402' 052702 100000      60$:   BIS     #C$CRLF,R2          ;SET THE 'CR/LF TERMINATOR' FLAG
3057 010406' 010267 177602      70$:   MOV     R2,C$CFWD          ;STORE FLAG WORD
3058 010412' 010067 171044      MOV     R0,CPNTR          ;STORE NEXT WORD/CR-LF ADR

```

3059	010416'	010167	171156	MOV	R1,CDCNT	;STORE LENGTH OF THE WORD
3060	010422'	016700	171150	MOV	CDSTAD,R0	;SET UP RIGHT HAND END
3061	010426'	060100		ADD	R1,R0	;ADR OF DATA
3062	010430'	005300		DEC	R0	
3063	010432'	032702	000010	BIT	#CSCFND,R2	;WAS ANY DATA FOUND?
3064	010436'	001415		BEQ	CSCNEX	;Y,N-CSCNEX
3065	010440'	032702	000003	BIT	#3,R2	;EITHER CONVERT FLAG SET?
3066	010444'	001412		BEQ	CSCNEX	;(Y,N-CSCNEX)
3067	010446'	004567	031676	JSR	R5,ASDECO	;GO CONVERT TO PACKED DECIMAL
3068	010452'	000022		.WORD	CSCERR-	;CONVERSION ERROR ADR
3069	010454'	032767	000001 177532	BIT	#1,CSCFWD	;CONVERT TO BINARY FLAG SET?
3070	010462'	001403		BEQ	CSCNEX	;(Y,N-CSCVEX)
3071	010464'	004567	031442	JSR	R5,OCTBIN	;GO CONVERT PACKED OCTAL TO BINARY
3072	010470'	000004		.WORD	CSCERR-	;CONVERSION ERROR ADR
3073	010472'	000205		CSCNEX: RTS	R5	;S/R EXIT
3074		001		.IF DF MM		
3075						
3076						;SCAN AND CONVERT TO 24 BIT BINARY (PACKED OCTAL) ENTRY POINT
3077						
3078						;INPUT & OUTPUT INFO SAME AS CSCNCB EXCEPT FOR R2
3079						
3080				CSCNBB: JSR	R5,CSCN	;SCAN THE NEXT WORD
3081				JSR	R5,ASOCBN	;CONVERT * TO BINARY
3082				.WORD	CSCERR-	
3083				RTS	R5	;EXIT IN-LINE
3084		000		.ENDC		
3085						
3086	010474'	004567	175524	CSCERR: JSR	R5,CTYPE	;GO REPORT CONVERSION ERROR
3087	010500'	000103		.WORD	103	
3088	010502'	176105		.WORD	CERM02-	

.SBTTL VALIDATE PROGRAM # S/R

3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145

010504' 004567 177512
010510' 004567 031634
010514' 000050
010516' 004567 031374
010522' 000042
010524' 005700
010526' 001416
010530' 020027 000020
010534' 101013
010536' 010704
010540' 062704 023454
010544' 006300
010546' 060004
010550' 010467 170702
010554' 011403
010556' 010367 170676
010562' 000205

010564' 004567 175434
010570' 000113
010572' 176006
001

THE NEXT WORD IN THE CMND DATA WILL BE SCANNED AND VALIDATED
AS A CORRECT PROGRAM NUMBER. IF VALID, THE ADDRESSES OF ITS CONTROL
ROUTINE TABLE ENTRY AND ITS PROGRAM TABLE, IF PRESENT, WILL BE
STORED AT STANDARD LOCATIONS. IF AN INVALID PROGRAM NUMBER, EXIT
WILL BE MADE TO THE ERROR S/R TO REPORT THIS CONDITION.
LINKAGE: JSR R5,CVPNUM SUBROUTINE CALL
DESTROYS REGISTERS: R0,R1,R2,R3,R4
EXIT: CURCTE & R4 = ADDRESS OF CONTROL ROUTINE TABLE ENTRY
CURPTA & R3 = ADDRESS OF PROGRAM TABLE

CVPNUM: JSR R5,CSCN ;SCAN PROG #
JSR R5,ASDECO ;CONVERT IT TO PACKED DECIMAL
.WORD CUPER-
JSR R5,DECBIN ;CONVERT PKED DEC TO BINARY
.WORD CUPER-
CVPNUM1: TST R0 ;IS IT 0?
BEQ CUPER ;N.Y-CUPER
CMP R0,#MAXPRG ;IS IT TOO LARGE?
BHI CUPER ;N.Y-CUPER
MOV PC,R4 ;COMPUTE CNTRL ROUT TBL ENTRY
ADD #CRTBL-2-.,R4
ASL R0
ADD R0,R4
MOV R4,CURCTE ;STORE CNTRL ROUT ENTRY ADR
MOV (R4),R3 ;GET PROG TBL ADR
MOV R3,CURPTA ;STORE ITS PROG TABLE ADR
RTS R5 ;EXIT IN-LINE

CUPER: JSR R5,CTYPE ;GO REPORT INV PROG # ERROR
.WORD 113
.WORD CERMO1-. ;ERROR MSG ADR
.IF DF MM
.PAGE
.SBTTL INITIALIZE MM PAR'S TO USER PROGRAM AREA

THIS SUBROUTINE USES THE CONTENTS OF R3 AS THE HIGH 16 BITS OF
AN ABSOLUTE ADDRESS AND MAPS KERNEL'S PAR'S 4 THRU 6 TO A THREE
PAGE AREA. UPON RETURN, R3 WILL CONTAIN THE ADDRESS OF THE FIRST
WORD OF THIS AREA BASED UPON PAR 4. THIS ADDRESS WILL ALSO BE
STORED AT THE LOCATION "CURPTA". THE ORIGINAL CONTENTS OF R3
WILL ALWAYS BE STORED AT "CURPTH". IF R3 IS ZERO UPON ENTRY,
NO OTHER ACTIONS WILL TAKE PLACE.
LINKAGE: JSR PC,UPAPAR S/R CALL

3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164

```

: DESTROYS R0,R1 - ALTERS R3
:*****
UPAPAR: MOV      R3,CURPTH      :SAVE HI 16 BITS OF ADR
        TST      R3            :R3 CONTAIN A BASE ADR?
        BEQ      UPAPEX        :Y,N-UPAPEX
        MOV      #KPAR4,R0     :SET UP ADR OF PAR 4 REG
        MOV      #3,R1         :SET UP LOOP CNT
IOS:    MOV      R3,(R0)+      :STORE MEM BASE ADR
        ADD      #200,R3       :POINT TO NEXT MEM PAGE
        DEC      R1            :DONE 3 PAGES?
        BNE      IOS           :Y,N-10$
        MOV      #P4CONS,R3    :POINT R3 AT 1ST WD OF THE AREA
        MOV      R3,CURPTA     :STORE IT IN MEMORY ALSO
UPAPEX: RTS      PC           :EXIT IN-LINE
        .ENDC

```

000

.SBTTL SHIFT HIGHER PROGRAMS UP S/R

3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221

```

*****
THIS SUBROUTINE WILL SEARCH THE CONTROL ROUTINE TABLE FROM
ITS CURRENT ENTRY TO ITS END. IF ANY HIGHER NUMBERED
PROGRAMS ARE FOUND, THEY WILL BE RELOCATED TO THE HIGH
END OF MEMORY OR, IF THE MEMORY MANAGEMENT VERSION, TO THE
LOWEST FIXED ADDRESS USER PROGRAM. THE CONTROL ROUTINE TABLE
ENTRIES WILL REFLECT THEIR NEW LOCATIONS.

LINKAGE: JSR R5,CSPGUP SUBROUTINE CALL

DESTROYS REGISTERS: R0,R1,R2
*****

```

010574' 016700 170556
001

```

CSPGUP: MOV CURCTE,R0 ;GET CURRENT CNTRL ROUT ENTRY ADR
        .IF DF MM
        TST CRTEXT(R0) ;THIS A FIXED ADR PROG?
        BPL CSPNFP ;Y,N-CSPNFP
        MOV CENADR,R1 ;GET ITS HI 16 BITS ADR
        MOV R1,(R0) ;STORE PROG ADR IN CTRL ROUT TBL
CSPGSZ: MOV R1,CURPTH ;STORE IT
        JSR PC,CKF3PG ;GET AVAIL PROG AREA SIZE IN R2
        MOV #6,R1 ;CONVERT IT TO A BYTE CNT
SS: ASL R2
    DEC R1
    BNE SS
    SUB #2,R2 ;LOWER ADR TO BELOW THE BNDRY
    ADD #P4CONS,R2 ;CONVERT IT TO A VIRTUAL ADR
    MOV R2,CVMEND ;STORE IT AS VIRTUAL MEM END
    RTS R5 ;EXIT IN-LINE
        .ENDC

```

010500' 010046
010602' 005046
001

```

CSPNFP: MOV R0,-(SP) ;SAVE ENTRY ADR
        CLR -(SP) ;RESET THE 'SHIFT' FLAG
        .IF DF MM
        MOV CPAREA,CSPNPA ;INIT NEXT PROG AREA ADR
        MOV CRMEND,CSPLFP ;INIT LOWEST FIXED PROG ADR
        MOV PC,R0 ;SET UP CNTRL ROUT TBL START ADR
        ADD #CRTBL-2-,R0 ;BEGIN SCAN AT BEGINNING OF TBL
        .ENDC

```

010604' 005720
010606' 022710 177777
010612' 001404
010614' 005710
010616' 001772
001

```

10S: TST (R0)+ ;INCR TO NXT TBL ENTRY
      CMP #177777,(R0) ;END OF THE TABLE?
      BEQ 20S ;N,Y-20S
      TST (R0) ;ENTRY IN USE?
      BEQ 10S ;Y,N-10S
      .IF DF MM
      .IFT
      TST CRTEXT(R0) ;THIS A FIXED ADR PROG?
      BMI 15S ;N,Y-15S
      CMP R0,2(SP) ;BELOW ORG PROG SLOT?
      BLO 12S ;N,Y-12S
      BEQ 10S ;AT ORG SLOT? (N,Y-10S)
      MOV R0,(SP) ;SET SHIFT FLAG

```

3222				BR	10\$;GO CK NXT ENTRY
3223				MOV	(R0),CSPNPA		;CALC & SAVE MEM ADR OF AREA
3224			12\$:	ADD	CRTEXT(R0),CSPNPA		;AFTER THIS PROG
3225				BR	10\$;GO CK NXT SLOT
3226				.IFF			
3227	010620'	010016		MOV	R0,(SP)		;SET SHIFT FLAG
3228	010622'	000770		BR	10\$;GO CK NXT ENTRY
3229				.IFT			
3230			15\$:	CMP	(R0),CSPLFP		;THIS PROG THE LOWEST SO FAR?
3231				BHIS	10\$;Y,N-10\$
3232				MOV	(R0),CSPLFP		;SAVE ITS ADR AND
3233				DEC	CSPLFP		;POINT IT TO NXT LOW 32 WD BLK
3234				BR	10\$;GO CK NEXT ENTRY
3235		000		.ENDC			
3236	010624'	005726	20\$:	TST	(SP)+		; 'SHIFT' FLAG SET?
3237	010626'	001006		BNE	25\$;N,Y-25\$
3238	010630'	012601		MOV	(SP)+,R1		;GET ORG TBL ENTRY ADR
3239	010632'	005711		TST	(R1)		;PROG ALREADY EXIST FOR ORG ENTRY?
3240	010634'	001022		BNE	40\$;N,Y-40\$
3241		001		.IF DF MM			
3242				.IFF			
3243	010636'	016711	170604	MOV	CFMST,(R1)		;SET UP START ADR OF FREE MEM
3244				.IFT			
3245				MOV	CSPNPA,(R1)		;GET ADR OF NXT MEM AREA
3246		000		.ENDC			
3247	010642'	000417		BR	40\$;GO TO S/R EXIT
3248	010644'	012601	25\$:	MOV	(SP)+,R1		;GET ORG TBL ENTRY ADR
3249	010646'	01034E		MOV	R3,-(SP)		;SAVE WORK REG'S
3250	010650'	010446		MOV	R4,-(SP)		
3251		001		.IF DF MM			
3252				.IFF			
3253	010652'	016704	170572	MOV	CFMEND,R4		;GET END OF FREE MEM ADR
3254				.IFT			
3255				MOV	CSPLFP,R4		;GET ADR OF LOWEST FIXED PROG
3256		000		.ENDC			
3257	010656'	005740	30\$:	TST	-(R0)		;DECR TBL ENTRY PNTR TO NXT LOWEST
3258	010660'	020001		CMP	R0,R1		;BACK TO ORG ENTRY?
3259	010662'	001012		BNE	50\$;Y,N-50\$
3260		001		.IF NDF MM			
3261	010664'	010467	170560	MOV	R4,CFMEND		;STORE NEW FREE MEM END ADR
3262		000		.ENDC			
3263	010670'	005711		TST	(R1)		;PROG ALREADY EXIST FOR ORG ENTRY?
3264	010672'	001001		BNE	35\$;N,Y-35\$
3265	010674'	010311		MOV	R3,(R1)		;SET UP START ADR FOR THIS PROG
3266	010676'	012604	35\$:	MOV	(SP)+,R4		;RESTORE WORK REG'S
3267	010700'	012603		MOV	(SP)+,R3		
3268		001		.IF DF MM			
3269				.IFF			
3270	010702'	011167	170552	MOV	(R1),CURPTA		;STORE NEW CURR PROG TBL ADR
3271	010706'	000205	40\$:	RTS	R5		;EXIT IN-LINE
3272				.IFT			
3273			40\$:	MOV	(R1),R1		;GET PROG AREA ADR IN R1
3274				BR	CSPG\$Z		;GO COMPUTE AREA SIZE & EXIT
3275		000		.ENDC			
3276	010710'	005710	50\$:	TST	(R0)		;IS ENTRY IN USE?
3277	010712'	001761		BEG	30\$;Y,N-30\$

3278		001		.IF DF MM		
3279				TST	CRTEXT(R0)	;FIXED ADR PROG?
3280				BMI	30\$;N,Y-30\$
3281		000		.ENDC		
3282	010714'	011003		MOV	(R0),R3	;GET PROG'S START ADR
3283		001		.IF DF MM		
3284				.IFF		
3285	010716'	016302	000026	MOV	PLNGTH(R3),R2	;GET PROG'S LENGTH
3286	010722'	060203		ADD	R2,R3	;SET UP PROG END ADR
3287	010724'	006202		ASR	R2	;DIVIDE LNPTH BY 2
3288	010726'	005724		TST	(R4)+	;ADD 2 TO END OF FREE MEM ADR
3289	010730'	014344		MOV	-(R3),-(R4)	;MOVE WORD OF THE PROG
3290	010732'	005302		DEC	R2	;DECR WORD CNT BY 1
3291	010734'	001375		BNE	60\$;CNT = 0? (Y,N-60\$)
3292	010736'	010402		MOV	R4,R2	;COMPUTE RELOCATION FACTOR
3293	010740'	160302		SUB	R3,R2	
3294	010742'	060264	000016	ADD	R2,PROIOA(R4)	;ADJ ADR'S OF SHIFTED PROG
3295	010746'	060264	000020	ADD	R2,PWRIOA(R4)	
3296	010752'	060264	000022	ADD	R2,PSRCST(R4)	
3297	010756'	060264	000024	ADD	R2,PCBJST(R4)	
3298	010762'	010410		MOV	R4,(R0)	;SET UP NEW PROG ADR IN CTRL ROUT TBL
3299	010764'	005744		TST	-(R4)	;SET UP NEW END OF FREE MEM ADR
3300	010766'	000733		BR	30\$;GO CK NXT TBL ENTRY
3301				.IFT		
3302				MOV	CRTEXT(R0),R2	;GET PROG'S LENGTH
3303				MOV	R0,-(SP)	;SAVE R0 & R1
3304				MOV	R1,-(SP)	
3305				JSR	PC,MOVPGU	;MOVE THE PROG UP IN MEM
3306				MOV	(SP)+,R1	;RESTORE R0 & R1
3307				MOV	(SP)+,R0	
3308				MOV	R4,(R0)	;STORE ITS NEW ADR
3309				DEC	R4	;POINT TO NXT LOW 32 WD BLK
3310				BR	30\$;GO CK NXT TBL ENTRY
3311						
3312						
3313			CSPNPA:	.WORD	0	*;NEXT PROG AREA ADR
3314			CSPLFF:	.WORD	0	*;ADR OF LOWEST FIXED ADR PROG
3315		000		.ENDC		

.SBTTL SHIFT PROGRAMS DOWN & RECOMPILE S/R

```

*****
THIS S/R PERFORMS THREE BASIC FUNCTIONS. FIRST, IT
WILL SCAN THE CONTROL ROUTINE TABLE FOR EXISTING PROGRAMS AND
DETERMINES IF THERE ARE ANY UNUSED MEMORY AREAS. IF THERE
ARE, PROGRAMS WILL BE SHIFTED DOWN TO ELIMINATE THESE AREAS.
AFTER THE PROGRAMS HAVE BEEN SHIFTED DOWN, IF ANY, THE
FLAGWORD FOR EACH PROGRAM WILL BE RE-INITIALIZED SO THAT
PROGRAMS CAN ONLY BE INITIATED BY THE "RUN" COMMAND AND
ALL PROGRAMS WILL THEN BE RECOMPILED.
LINKAGE: JSR R5,CSDNRC SUBROUTINE CALL
DESTROYS REGISTERS: R0,R1,R2
*****

```

3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361
3362
3363
3364
3365
3366
3367
3368
3369
3370
3371
3372

```

010770' 010346
010772' 010446
010774' 010546
010776' 005046
011000' 016705 170450
      001
011004' 010704
011006' 062704 023210
011012' 005714
011014' 001021
      001
011016' 005724
011020' 022714 177777
011024' 001372
011026' 005726
011030' 001007
011032' 010446
      001
011034' 016767 170412 170406
011042' 010567 170400
      000
011046' 000756

```

```

CSDNRC: MOV R3,-(SP) ;SAVE WORK REG'S
        MOV R4,-(SP)
        MOV R5,-(SP)
        CLR -(SP) ;RESET 'END SHIFT' FLAG
        MOV CPAREA,R5 ;INITIALIZE PREV PROG END ADR
                                ;TO START OF PROG AREA
HIMPGP: .IF DF MM
        MOV R5,(PC)+ ;INIT HIGHEST MPG PROG ADR
        .WORD 0
        .ENDC
10$: MOV PC,R4 ;SET UP START ADR OF CNTRL
      ADD #CRTBL-.,R4 ;ROUT TBL
20$: TST (R4) ;THIS ENTRY IN USE?
      BNE 50$ ;N,Y-50$
      .IF DF MM
      CLR CRTEXT(R4) ;HSPK ITS TBL EXTENSION WORD
      .ENDC
30$: TST (R4)+ ;INCR TO NXT TBL ENTRY
      CMP #177777,(R4) ;END OF CNTRL ROUT TBL?
      BNE 20$ ;Y,N-20$
      TST (SP)+ ;'END SHIFT' FLAG SET?
      BNE 40$ ;N,Y-40$
      MOV R4,-(SP) ;SET END SHIFT FLAG
      .IF DF MM
      .IFF
      MOV CRMEND,CFMEND ;SET FREE MEM END TO REAL MEM END
      MOV R5,CFMST ;MOVE PREV PROG END TO FREE MEM START
      .IFT
      JSR PC,FINDFM ;GO DETERMINE FREE MEM AREA
      MOV R0,CFMST ;STORE ITS START ADR
      ADD R0,R1
      DEC R1
      MOV R1,CFMEND ;STORE ITS END ADR
      JSR PC,SUMIFM ;SET UP MIDL & FREE'S ADRS
      .ENDC
BR 10$ ;GO DO SECOND PASS OF TBL

```

```

3373 011050' 012605          40$:  MOV      (SP)+,R5          ;RESTORE WORK REG'S
3374 011052' 012604          MOV      (SP)+,R4
3375 011054' 012603          MOV      (SP)+,R3
3376 011056' 000205          RTS      R5          ;EXIT IN-LINE
3377 011060' 011403          50$:  MOV      (R4),R3      ;GET THIS PROG TBL ADR
3378                001          .IF DF MM
3379                000          JSR      PC,UPAPAR    ;MAP KERNEL'S PAR'S TO USR PROG
3380                000          .ENDC
3381 011062' 005716          TST      (SP)        ;'END SHIFT' FLAG SET?
3382 011064' 001411          BEQ      60$         ;Y,N-60$
3383 011066' 011463 000004  MOV      (R4),PFWADR(R3) ;STORE FLAGWORD ADR IN PROG TBL
3384                001          .IF DF MM
3385                000          JSR      PC,MAPUPG    ;MAP USER'S PAR & PDR STORAGE
3386                000          MOV      (R4),(PC)+   ;GET HIGH 16 BITS ADR
3387                000          52$:  .WORD    XXXX
3388                000          JSR      R5,G22BAD    ;CONVERT IT TO A 22 BIT ADR
3389                000          .WORD    52$-
3390                000          MOV      R2,PGABAD   ;STORE PROG'S ABS ADR
3391                000          MOV      R0,PGABAD+2 ;FOR COMPILER USE
3392                000          MOV      R3,R1      ;POINT AT STORAGE AREA FOR
3393                000          ADD      #PRDIOX,R1 ;18/22 BIT ABS ADR OF RDIO
3394                000          MOV      R2,(R1)+   ;STORE HIGH BITS OF BASE ADR
3395                000          MOV      PRDIOA(R3),(R1) ;STORE RDIO DISPL
3396                000          ADD      R0,(R1)   ;ADD IN LOW BITS OF BASE ADR
3397                000          ADC      -(R1)
3398                000          ADD      #8,R1      ;POINT AT WRIO STORAGE
3399                000          MOV      R2,(R1)+   ;STORE HIGH BITS OF BASE ADR
3400                000          MOV      PWRIOA(R3),(R1) ;STORE WRIO DISPL
3401                000          ADD      R0,(R1)   ;ADD IN LOW BITS OF BASE ADR
3402                000          ADC      -(R1)
3403                000          BIT      #UNIMAP,CSYSFW ;WE USING THE UNIBUS MAP?
3404                000          BEQ      58$         ;Y,N-58$
3405                000          TST      PUBMAP(R3)    ;DOES PROG USE UNIBUS MAP?
3406                000          BEQ      58$         ;Y,N-58$
3407                000          MOVVB   PUBMAP(R3),R1    ;GET 1ST MAP REG #
3408                000          ASL      R1          ;COMPUTE DISPL INTO UNIBUS
3409                000          ASL      R1          ;MAP REGS
3410                000          ADD      #UBMAP0,R1
3411                000          MOVVB   PUBMAP+1(R3),-(SP) ;GET # OF MAP REG BITS
3412                000          CMPB    (SP),#3      ;HOW MANY REGS BEING USED?
3413                000          BLO      54$         ;BR IF 1
3414                000          BEQ      53$         ;BR IF 2
3415                000          SUB      #3,(SP)    ;ADJ FOR 3 REGS
3416                000          53$:  DECB    (SP)      ;ADJ FOR 2 REGS
3417                000          54$:  MOV      R0,(R1)+   ;STORE UNIBUS MAP REG VALUE
3418                000          MOV      R2,(R1)+
3419                000          ADD      #20000,R0
3420                000          ADC      R2
3421                000          DECB    (SP)      ;DONE ALL REGS?
3422                000          BNE      54$         ;Y,N-54$
3423                000          TST      (SP)+
3424                000          MOVVB   PUBMAP(R3),R2    ;H$KP STK
3425                000          CLR      R0          ;GET 1ST UNIBUS MAP REG #
3426                000          MOV      #3,R1
3427                000          56$:  ASR      R2          ;SET UP LOOP COUNT
3428                000          ROR      R0          ;CONVERT REG # TO UNIBUS MAP
;REG SELECT BITS

```

3429				DEC	R1	
3430				BNE	56\$	
3431				MOV	R2,PGUBMA	; STORE UNIMAP REG SELECT BITS FOR
3432				MOV	R0,PGUBMA+2	; COMPILER USE
3433				MOV	R3,R1	; POINT AT STORAGE AREA FOR 18
3434				ADD	#PRDIOV,R1	; BIT VIRT ADRS OF RDIO & WRIO
3435				MOV	R2,(R1)+	; STORE HIGH REG SELECT BITS
3436				MOV	PRDIOA(R3),(R1)	; STORE RDIO DISPL
3437				ADD	R0,(R1)	; ADD IN LOW REG SELECT BITS
3438				ADC	-(R1)	
3439				ADD	#8,R1	; POINT AT WRIO STORAGE
3440				MOV	R2,(R1)+	; STORE HIGH REG SELECT BITS
3441				MOV	PWRIOA(R3),(R1)	; STORE WRIO DISPL
3442				ADD	R0,(R1)	; ADD IN LOW REG SELECT BITS
3443				ADC	-(R1)	
3444		000		.ENDC		
3445	011072'	032713	000100	58\$: BIT	#OCPRES,(R3)	; THIS PROG ALREADY HAVE OBJ CODE?
3446	011076'	001747		BEQ	30\$; Y,N-30\$
3447		001		.IF DF	MM	
3448				.IFF		
3449	011100'	005013		CLR	(R3)	; RESET THIS PROG'S FLGWD
3450				.IFT		
3451				BIC	#177777-USEUBM,(R3)	; HSKP THIS PROG'S FLGWD
3452		000		.ENDC		
3453	011102'	004567	003722	JSR	R5,COMPIL	; GO RECOMPILE THIS PROG
3454	011106'	000743		BR	30\$; GO CK NXT TBL ENTRY
3455	011110'	016301	000026	60\$: MOV	PLNGTH(R3),R1	; GET PROG'S LENGTH
3456	011114'	001002		BNE	65\$; LENGTH = 0? (Y,N-65\$)
3457	011116'	005014		CLR	(R4)	; RESET ITS CTRL ROUT TBL ENTRY
3458		001		.IF DF	MM	
3459				.IFF		
3460	011120'	000736		BR	30\$; GO CK NXT PROG
3461	011122'	020305	65\$:	CMP	R3,R5	; PROG'S START ADR = PREV PROG END ADR?
3462	011124'	001002		BNE	70\$; Y,N-70\$
3463	011126'	060105		ADD	R1,R5	; ADD PROG LENGTH TO ITS START ADR &
3464						; SAVE AS PREV PROG END ADR
3465	011130'	000732		BR	30\$; GO CHECK NEXT PROG SLOT
3466	011132'	010302	70\$:	MOV	R3,R2	; COMPUTE RELOCATION FACTOR
3467	011134'	160502		SUB	R5,R2	
3468	011136'	160263	000016	SUB	R2,PRDIOA(R3)	; ADJ PROG TBL ENTRIES TO NEW LOC
3469	011142'	160263	000020	SUB	R2,PWRIOA(R3)	
3470	011146'	160263	000022	SUB	R2,PSRCST(R3)	
3471	011152'	160263	000024	SUB	R2,POBJST(R3)	
3472	011156'	006001		ROR	R1	; DIVIDE PROG LENGTH BY 2
3473	011160'	010514		MOV	R5,(R4)	; SET NEW PROG ADR IN CNTRL ROUT TBL
3474	011162'	012325	80\$:	MOV	(R3)+,(R5)+	; MOV 1 WORD OF PROG DOWN
3475	011164'	005301		DEC	R1	; DECR WORD COUNT
3476	011166'	001375		BNE	80\$; CNT = 0? (Y,N-80\$)
3477	011170'	000712		BR	30\$; GO CHECK NXT PROG SLOT
3478				.IFT		
3479				CLR	CRTEXT(R4)	; RESET CNTRL ROUT TBL EXT
3480				BR	80\$; GO CK NXT PROG
3481			65\$:	TST	CRTEXT(R4)	; THIS A FIXED ADR PROG?
3482				BMI	80\$; N,Y-80\$
3483				CMP	(R4),R5	; PROG'S START ADR = PREV PROG END ADR?
3484				BNE	70\$; Y,N-70\$

J06

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 SHIFT PROGRAMS DOWN & RECOMPILE S/R

MACY11 27(732) 24-SEP-76 13:54 PAGE 22-3

SEQ 0075

```
3485          ADD      CRTEXT(R4),R5          ;ADD PROG LGTH TO ITS START ADR &  
3486          ;SAVE AS PREV PROG ADR  
3487          BR       75$                      ;GO STORE NEW ADR  
3488          70$:    MOV      CRTEXT(R4),R3    ;GET PROG'S LENGTH  
3489          JSR      PC,MOVPGD                ;MOVE THE PROG DOWN  
3490          75$:    MOV      R5,HIMPGP        ;STORE AS HIGHEST MPG PROG ADR  
3491          80$:    JMP      30$              ;GO CK NXT PROG SLOT  
3492          PGABAD: .WORD    0                ;UP TO 22 BIT ABSOLUTE ADR (16 - 21)  
3493          ;OF CURRENT PROG'S AREA (0 - 15)  
3494          PGUBMA: .WORD    0                ;CURRENT PROG'S UNIBUS MAP (16 & 17)  
3495          ;REG SELECT BITS (0 - 15)  
3496          .WORD    0  
3497          .ENDC  
000
```

.SBTTL REQUEST DEVICE & INT ADR'S S/R

```

*****
THIS S/R USES A 4 WORD AREA AS INPUT. THESE WORDS SHOULD
CONTAIN THE DEV REG ADR, INT VECTOR ADR, SINGLE OR READ
INT PROCESSOR STATUS WORD, AND THE WRITE INT STATUS WORD
(OR A WORD OF ZEROS) IN THAT ORDER. REG R4 MUST CONTAIN AN
ADDRESS THAT POINTS TO THE FIRST WORD.

THE THREE QUERY MESSAGES WILL BE DISPLAYED WITH THE
EXISTING DATA BEING TAILORED AND INCLUDED. THE USER
MAY MAKE AN ENTRY THAT REPLACES THE EXISTING DATA OR MAKE A
NULL ENTRY WHICH CAUSES THE EXISTING DATA TO BE RETAINED.

LINKAGE:      JSR      R5,CRQADR      SUBROUTINE CALL

INPUT:        R4 MUST CONTAIN ADR OF DEVICE'S 4 INFO WORDS

DESTROYS REGISTERS: R0,R1,R2
*****

```

3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554

```

011172' 122777 000012 170262 CRQADR: CMPB    #012, @CPNTR      ;LINE FEED TERMINATE PREV REPLY?
011200' 001002          BNE      2$          ;Y,N-2$
011202' 000167 000400          JMP      CRQEX      ;GO TO EXIT - BYPASS REQUESTS
011206' 010346          2$:    MOV     R3,-(SP)  ;SAVE WORK REG'S
011210' 010446          MOV     R4,-(SP)
011212' 010546          MOV     R5,-(SP)
011214' 005046          CLR     -(SP)      ;RESET LOOP CNT
011216' 012703 176265          5$:    MOV     #CRQDRM-CRQLNK,R3 ;SET UP ADR OF FIRST MSG
011222' 010705          MOV     PC,R5      ;SET UP ADR OF DATA IN 1ST MSG
011224' 062705 176414          ADD     #CRQDRD-. ,R5
011230' 010702          CRQNXM: MOV    PC,R2      ;SET UP LINK INFO ADR
011232' 062702 000106          ADD     #CRQLNK-. ,R2
011236' 010322          MOV     R3,(R2)+   ;STORE MSG ADR IN LINK INFO
011240' 012712 000013          MOV     #11. ,(R2) ;SET UP MSG ONLY LENGTH
011244' 011400          MOV     (R4),R0    ;GET PRESET DATA WORD
011246' 122716 000002          CMPB   #2,(SP)     ;THIS BUS REQ MSG?
011252' 001406          BEQ    10$         ;N,Y-10$
011254' 062712 000011          ADD     #9. ,(R2)  ;SET UP MSG & DATA LENGTH
011260' 010501          MOV     R5,R1      ;SET UP OUTPUT ADR FOR S/R
011262' 004567 027640          JSR    R5,BINAS1   ;GO CONVERT DATA TO ASCII
011266' 000421          BR     CRQLPT      ;GO ISSUE DEV REG OR INT VECT MSG
011270' 062712 000006          10$:  ADD     #6. ,(R2)  ;SET UP BUS REQ MSG LGTH
011274' 004367 000310          JSR    R3,CRQCTA   ;CONVERT FIRST PS WD TO ASCII #
011300' 016400 000002          MOV     2(R4),R0   ;GET WRITE PS WORD
011304' 005700          TST    R0          ;ANY DATA?
011306' 001005          BNE    20$         ;N,Y-20$
011310' 012701 000040          MOV     #40,R1     ;CLEAR 2ND BR TO SPACES
011314' 110125          MOVB   R1,(R5)+
011316' 110115          MOVB   R1,(R5)
011320' 000404          BR     CRQLPT      ;GO ISSUE SINGLE BR MSG
011322' 112725 000054          20$:  MOVB   #054,(R5)+ ;MOVE COMMA TO MSG
011326' 004367 000256          JSR    R3,CRQCTA   ;CONVERT SECOND PS WD TO ASCII

```

3555	011332'	004567	174666	CRQLPT:	JSR	R5,CTYPE		;ISSUE INFO REQUEST MSG
3556	011336'	000420			.WORD	420		
3557	011340'	000000		CRQLNK:	.WORD	XXXX		
3558	011342'	000000			.WORD	XXXX		
3559	011344'	004577	166460		JSR	R5,ACTRD		;ISSUE READ FOR THE REPLY
3560	011350'	170114			.WORD	CCMDRD-		
3561	011352'	000107			.WORD	107		
3562	011354'	004767	001750		JSR	PC,CSUPTR		;INITIALIZE READ DATA PNTR
3563	011360'	004567	176636	25\$:	JSR	R5,CSCN		;SCAN FOR REPLY
3564	011364'	032702	000010		BIT	#CSCFND,R2		;ANY DATA ENTERED ON REPLY?
3565	011370'	001465			BEQ	60\$;Y,N-60\$
3566	011372'	004567	030752	30\$:	JSR	R5,ASDECO		;CONVERT DATA TO BINARY
3567	011376'	000232			.WORD	CRQER-		
3568	011400'	004567	030526		JSR	R5,OCTBIN		
3569	011404'	000224			.WORD	CRQER-		
3570	011406'	122716	000002		CMPB	#2,(SP)		;THIS BUS REQ MSG?
3571	011412'	001026			BNE	44\$;Y,N-44\$
3572	011414'	020027	000007		CMP	RO,#7		;VALID BUSS REQ #?
3573	011420'	101405			BLOS	40\$;N,Y-40\$
3574	011422'	004567	174576		JSR	R5,CTYPE		;ISSUE 'INV BR' MSG
3575	011426'	000712			.WORD	712		
3576	011430'	175416			.WORD	CERM20-		
3577	011432'	000737			BR	CRQLPT		;GO ISSUE MSG AGAIN
3578	011434'	000300		40\$:	SWAB	RO		;SET UP PS WORD
3579	011436'	006200			ASR	RO		
3580	011440'	006200			ASR	RO		
3581	011442'	006200			ASR	RO		
3582	011444'	010024			MOV	RO,(R4)+		;STORE NEW PS WORD
3583	011446'	032716	000400		BIT	#400,(SP)		;WAS THIS 2ND PS WORD?
3584	011452'	001035			BNE	62\$;N,Y-62\$
3585	011454'	005767	176534		TST	CSCFWD		;IS THERE ANOTHER BR # ENTERED?
3586	011460'	100432			BMI	62\$;Y,N-62\$
3587	011462'	052716	000400		BIS	#400,(SP)		;SET SECOND PS WORD FLAG
3588	011466'	000734			BR	25\$;GO PROCESS SECOND PS WORD
3589	011470'	032700	000001	44\$:	BIT	#1,RO		;IS MEM ADR ODD?
3590	011474'	001405			BEQ	48\$;Y,N-48\$
3591	011476'	004567	174522		JSR	R5,CTYPE		;ISSUE 'ODD ADR' ERR MSG
3592	011502'	000702			.WORD	702		
3593	011504'	175435			.WORD	CERM25-		
3594	011506'	000711			BR	CRQLPT		;GO RE-ISSUE MSG
3595	011510'	105716		48\$:	TSTB	(SP)		;THIS DEV REG ADR MSG?
3596	011512'	001010			BNE	54\$;Y,N-54\$
3597	011514'	020027	160010		CMP	RO,#160010		;IS ADR IN DEV REG RANGE?
3598	011520'	103010			BHIS	58\$;N,Y-58\$
3599	011522'	004567	174476	50\$:	JSR	R5,CTYPE		;ISSUE 'INV ADR RANGE' MSG
3600	011526'	000712			.WORD	712		
3601	011530'	175421			.WORD	CERM26-		
3602	011532'	000677			BR	CRQLPT		;GO RE-ISSUE MSG
3603	011534'	020027	001000	54\$:	CMP	RO,#1000		;IS ADR IN INT VECT RANGE?
3604	011540'	103370			BHIS	50\$;Y,N-50\$
3605	011542'	010014		58\$:	MOV	RO,(R4)		;STORE ENTERED DATA
3606	011544'	005724		60\$:	TST	(R4)+		;INCR TO NXT STORAGE ADR
3607	011546'	005216		62\$:	INC	(SP)		;ADD 1 TO LOOP COUNT
3608	011550'	062703	000024		ADD	#20.,R3		;SET MSG PNTR TO NXT MSG
3609	011554'	062705	000024		ADD	#20.,R5		;SET TAILORED DATA PNTR TO NXT MSG
3610	011560'	122777	000012	167674	CMPB	#012,ACPNT		;LINE FEED TERMINATE PREV REPLY?

M06

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 REQUEST DEVICE & INT ADR'S S/R

MACY11 27(732) 24-SEP-76 13:54 PAGE 23-2

SEQ 0078

3611	011566'	001403			BEG	70\$;N,Y-70\$
3612	011570'	122716	000003		CMPB	#3,(SP)		;DONE ALL MSG'S?
3613	011574'	001215			BNE	CRQNXM		;Y,N-CRQNXM
3614	011576'	005726		70\$:	TST	(SP)+		;TAKE FLGWD OFF THE STACK
3615	011600'	012605			MOV	(SP)+,R5		;RESTORE REG'S
3616	011602'	012604			MOV	(SP)+,R4		
3617	011604'	012603			MOV	(SP)+,R3		
3618	011606'	000205		CRQEX:	RTS	R5		;EXIT IN-LINE
3619								
3620								
3621	011610'	006300		CRQCTA:	ASL	RO		;CONVERT PRIORITY FIELD TO
3622	011612'	006300			ASL	RO		;TO A 1 DIGIT ASCII NUMBER
3623	011614'	006300			ASL	RO		
3624	011616'	000300			SWAB	RO		
3625	011620'	052700	000060		BIS	#60,RO		
3626	011624'	110025			MOVB	RO,(R5)+		;STORE BUS REQ # IN MSG
3627	011626'	000203			RTS	R3		;EXIT IN-LINE
3628								
3629								
3630	011630'	004567	174370	CRQER:	JSR	R5,CTYPE		;ISSUE DATA CONV ERR MSG
3631	011634'	000702			.WORD	702		
3632	011636'	174751			.WORD	CERMO2-		
3633	011640'	000634			BR	CRQLPT		;GO ISSUE MSG AGAIN

.SBTTL LOAD DEVICE ROUTINE S/R

THIS S/R WILL LOAD THE SPECIFIED DEVICE ROUTINE FROM THE SYSTEM LOAD DEVICE AND PLACE IT IN MEMORY IMMEDIATELY FOLLOWING THE CURRENT PROGRAM TABLE. INITIALIZATION OF CERTAIN PROGRAM TABLE ENTRIES WILL ALSO OCCUR. THE 'REQUEST ADR'S' S/R WILL BE USED TO UPDATE THE DEV ROUTINE'S DEVICE ADR'S.

LINKAGE: JSR R5,CLDEV R SUBROUTINE CALL
.WORD ERRADR-. INSUF MEM REL ERR RET ADR
.WORD DERADR-. DEVICE ERROR REL ERR RET ADR

INPUT: R3 MUST CONTAIN ADR OF CURRENT PROG TABLE

DESTROYS REGISTERS: R0,R1,R2

3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690

```

011642' 012767 000400 000544 CLDEV: MOV #CIOALT,RDIOSZ ;INITIALIZE I/O AREA SIZES
011650' 012767 000400 000546 MOV #CIOALT,WRIOSZ
011656' 122777 000012 167576 CMPB #012,ACPNTN ;BYPASS PRESET MESSAGES?
011664' 001460 BEQ CLOPDR ;N,Y-CLOPDR
011666' 012767 042122 176200 MOV #*RD,CIOSZM+1 ;INITIALIZE MSG TO "RDIO"
011674' 004567 174324 CSZREQ: JSR R5,CTYPE ;ISSUE REQ MSG
011700' 000420 .WORD 420
011702' 176171 .WORD CIOSZM-.
011704' 000016 .WORD 14.
011706' 004577 166116 JSR R5,ACTRD ;ISSUE RD FOR THE REPLY
011712' 167552 .WORD CCMRD-.
011714' 000036 .WORD 30.
011716' 004567 176274 JSR R5,CSCNSP ;SCAN FOR THE REPLY
011722' 032702 000010 BIT #CSCFND,R2 ;DATA RECEIVED?
011726' 001433 BEQ 115$ ;Y,N-115$
011730' 004567 030414 JSR R5,ASDECO ;CONVERT DECIMAL DATA TO BINARY
011734' 000552 .WORD CLDER3-
011736' 004567 030154 JSR R5,DECBIN
011742' 000544 .WORD CLDER3-
011744' 032700 000001 BIT #1,R0 ;WAS AN ODD SIZE ENTERED?
011750' 001401 BEQ 100$ ;Y,N-100$
011752' 005200 INC R0 ;MAKE IT EVEN
011754' 122767 000122 176112 100$: CMPB #*R,CIOSZM+1 ;THIS RDIO MSG?
011762' 001012 BNE 110$ ;Y,N-110$
011764' 010067 000424 MOV R0,RDIOSZ ;SAVE DATA AS RDIO SIZE
011770' 122777 000012 167464 105$: CMPB #012,ACPNTN ;LINE FEED TERMINATE RDIO REPLY?
011776' 001413 BEQ CLOPDR ;N,Y-CLOPDR
012000' 012767 051127 176066 MOV #*WR,CIOSZM+1 ;SET UP FOR WRIO MSG
012006' 000732 BR CSZREQ ;GO GET WRIO SIZE
012010' 010067 000410 110$: MOV R0,WRIOSZ ;STORE DATA AS WRIO SIZE
012014' 000404 BR CLOPDR ;GO TO DVR LOAD
012016' 122767 000122 176050 115$: CMPB #*R,CIOSZM+1 ;THIS RDIO MSG?
012024' 001761 BEQ 105$ ;N,Y-105$
001 .IF DF MM
.IFT

```



```

3691          CLDFDR: MOV      RDIOSZ,R0      ;COMBINE SIZES OF RDIO AND
3692          ADD      WRIOSZ,R0      ;WRIO
3693          CMP      R0,#24250.      ;THEIR SIZE GREATER THAN 2.9 PAGES?
3694          BLOS     CLDPDX         ;Y,N-CLDPDX
3695          JSR      R5,CTYPE        ;ISSUE ERROR MSG
3696          .WORD    402
3697          .WORD    CERM33-.
3698          CLRB     JCPNTR         ;RESET LINE FEED TERM IF THERE
3699          BR       CLDEVR         ;GO REQUEST NEW VALUES
3700          CLDPDX: MOV      R3,R2      ;GET DEV ROUT AREA START ADR
3701          .IFF
3702 012026' 010302          CLDPDR: MOV      R3,R2      ;GET DEV ROUT AREA START ADR
3703          .ENDC
3704 012030' 062702 000242          ADD      #PTLGTH,R2
3705 012034' 010267 000202          MOV      R2,CDVRST      ;STORE IT IN S/R LINK INFO
3706 012040' 062702 000012          ADD      #10,R2        ;ALLOW FOR I/O AREAS & "END" STMT
3707 012044' 066702 000344          ADD      RDIOSZ,R2
3708 012050' 066702 000350          ADD      WRIOSZ,R2
3709 012054' 016701 167370          MOV      CVMEND,R1      ;GET END OF FREE MEM ADR
3710 012060' 020201          CMP      R2,R1        ;ENOUGH ROOM SO FAR?
3711 012062' 103402          BLO      55
3712 012064' 000167 000406          25:     JMP      CLDER1      ;Y,N-CLDER1
3713 012070' 160201          55:     SUB      R2,R1      ;GET SIZE OF AVAILABLE MEM
3714 012072' 010167 000146          MOV      R1,CDVRSZ      ;STORE IT IN LINK INFO
3715 012076' 026767 030474 167514          CMP      CNONID+4,CMCDVN ;THIS THE "NONE" DEV?
3716 012104' 001035          BNE      205           ;Y,N-205
3717 012106' 022701 000120          CMP      #DRTLTH+2,R1   ;ENOUGH ROOM FOR "NONE" DEV ROUT?
3718 012112' 101364          BHI      25           ;Y,N-CLDER1
3719 012114' 016700 000122          MOV      CDVRST,R0      ;GET DEV ROUT AREA START ADR
3720 012120' 012720 000120          MOV      #DRTLTH+2,(R0)+ ;STORE "NONE" DEV ROUT LENGTH
3721 012124' 012701 000022          MOV      #DIOEAD/2,R1  ;GET # OF WORDS TO CLEAR
3722 012130' 005020          105:    CLR      (R0)+      ;CLEAR WORD FOR "NONE" DEV ROUT
3723 012132' 005301          DEC      R1            ;DECR WORD COUNT
3724 012134' 001375          BNE      105          ;CNT = 0? (Y,N-105)
3725 012136' 062700 000050          ADD      #DRTEND-DEVI0B,R0 ;POINT AT END OF THE TBL
3726 012142' 012710 177777          MOV      #177777,(R0)  ;STORE TERMINATOR WORD
3727 012146' 012740 000002          MOV      #2,-(R0)      ;STORE TABLE POINTERS
3728 012152' 012740 000004          MOV      #4,-(R0)
3729 012156' 012740 000006          MOV      #6,-(R0)
3730 012162' 012740 000010          MOV      #10,-(R0)
3731 012166' 012740 000012          MOV      #12,-(R0)
3732 012172' 012740 000014          MOV      #14,-(R0)
3733 012176' 000477          BR       CLDISU        ;GO SET UP PROG TBL & DEV ROUT INFO
3734 012200' 112767 000124 165572          205:    MOVB     #'T,CLOCZ      ;STORE "TXXA?" CODE AND
3735 012206' 116767 000516 165565          MOVB     DVRID,CLOCZ+1  ;DVR I.D.
3736 012214' 116767 000511 165560          MOVB     DVRID+1,CLOCZ+2 ;CODE
3737 012222' 112767 000101 165553          MOVB     #'A,CLOCZ+3    ;IN DEVICE
3738 012230' 012767 037477 165546          MOV      #'?',CLOCZ+4  ;ROUT NAME AREA
3739 012236' 004577 165610          JSR      R5,LOADVR     ;GO LOAD THIS DEV ROUT
3740 012242' 000000          CDVRST: .WORD    XXXX   ;DEV ROUT MEM START ADR
3741 012244' 000000          CDVRSZ: .WORD    XXXX   ;DEV ROUT AREA MAX SIZE
3742 012246' 000234          .WORD    CLDER2-.     ;DEV ERROR RETURN ADR
3743 012250' 000226          .WORD    CLDER1-.     ;INSUFF MEMORY ERR RET ADR
3744 012252' 000002          .WORD    CLDNRT-.     ;NORMAL RET ADR
3745 012254' 010302          CLDNRT: MOV      R3,R2   ;POINT AT DEV ROUTINE
3746 012256' 062702 000244          ADD      #PTLGTH+DEVFWD,R2 ;FLAGWORD

```

3747	012262'	005712		TST	(R2)		; IS IT ZERO?
3748	012264'	001402		BEQ	30\$; N, Y-30\$
3749	012266'	061202		ADD	(R2), R2		; IT'S AN ADR - MAKE IT ABS
3750	012270'	004512		JSR	R5, (R2)		; GO TO DEV'S LOAD HOUSEKEEPING
3751	012272'	032767	000004 02:714 30\$:	BIT	#LSI11, CSYSFW		; WE ON AN LSI-11?
3752	012300'	001413		BEQ	40\$; Y, N-40\$
3753	012302'	010302		MOV	R3, R2		; GET ADR OF DEV'S BUS REQUEST
3754	012304'	062702	000272	ADD	#PTLGTH+DEV RPS, R2		; POINT AT READ OR ONLY PRIORITY
3755	012310'	005712		TST	(R2)		; IS IT 0?
3756	012312'	001406		BEQ	40\$; N, Y-40\$
3757	012314'	012722	000200	MOV	#200, (R2)+		; UNCOND. SET IT TO BR 4
3758	012320'	005712		TST	(R2)		; IS THERE A WRITE PRIORITY?
3759	012322'	001402		BEQ	40\$; Y, N-40\$
3760	012324'	012712	000200	MOV	#200, (R2)		; SET IT TO BR 4 ALSO
3761	012330'	016700	165526	MOV	REALNM, R0		; GET ADR OF FILE'S NAME
3762	012334'	012067	175630	MOV	(R0)+, FLLDNM		; MOVE NAME TO THE MSG
3763	012340'	012067	175626	MOV	(R0)+, FLLDNM+2		
3764	012344'	011067	175624	MOV	(R0), FLLDNM+4		
3765	012350'	004567	173650	JSR	R5, CTYPE		; ISSUE MSG WITH FILENAME
3766	012354'	000600		.WORD	600		
3767	012356'	175576		.WORD	FLLDMG-		
3768	012360'	010446		MOV	R4, -(SP)		; SAVE R4
3769	012362'	010304		MOV	R3, R4		; SET UP ADR OF DEV ROUT
3770	012364'	062704	000266	ADD	#PTLGTH+DEV DRA, R4		; DEV INFO
3771	012370'	004567	176576	JSR	R5, CROADR		; GO REQUEST DEV INFO CHANGES
3772	012374'	012604		MOV	(SP)+, R4		; RESTORE R4
3773	012376'	017701	177640	MOV	#CDVRST, R1		; GET DEV ROUT LENGTH
3774	012402'	066701	177634	ADD	CDVRST, R1		; POINT AT END OF DEV ROUT
3775	012406'	010163	000016	MOV	R1, PROIOA(R3)		; SET UP ADR OF RDIO AREA
3776	012412'	062701		ADD	(PC)+, R1		; ADJ ADR TO WRIO
3777	012414'	000400		.WORD	256		
3778	012416'	010163	000020	MOV	R1, PWIOA(R3)		; STORE WRIO AREA ADR
3779	012422'	062701		ADD	(PC)+, R1		; INCR PAST WRIO AREA
3780	012424'	000400		.WORD	256		
3781	012426'	010163	000022	MOV	R1, PSRCST(R3)		; STORE SOURCE STMENTS START ADR
3782	012432'	010167	167010	MOV	R1, CVMST		; STORE NEW FREE MEM START ADR
3783	012436'	010700		MOV	PC, R0		; GET TABLE ADR OF ADR'S TO
3784	012440'	062700	000062	ADD	#CLDTBL-., R0		; BE PLACED IN DEV ROUT
3785	012444'	010302		MOV	R3, R2		; SET UP THEIR ADR IN DEV ROUT
3786	012446'	062702	000310	ADD	#PTLGTH+DEVI08, P2		
3787	012452'	012001		CLDLOP: MOV	(R0)+, R1		; GET REL OR VIRTUAL ADR
3788		001		.IF DF MM			
3789		000		BMI	CLDBAS		; PAGE 6 ADR? (N, Y-CLDBAS)
3790		000		.ENOC			
3791	012454'	060701		ADD	PC, R1		; ITS REL. MAKE IT ABSOLUTE
3792	012456'	010122		CLDBAS: MOV	R1, (R2)+		; STORE IT IN DEV ROUT
3793	012460'	005710		TST	(R0)		; END OF TBL?
3794	012462'	001373		BNE	CLDLOP		; Y, N-CLDLOP
3795		001		.IF DF MM			
3796				TST	PTLGTH+DEVIW7(R3)		; THIS DEV ROUT NEED UNIBUS MAP?
3797				BEQ	CLDEX		; Y, N-CLDEX
3798				BIS	#USEUBM, (R3)		; SET ITS USE UNIBUS MAP FLG
3799				BIT	#UNIMAP, CSYSFW		; USING THE UNIBUS MAP?
3800				BNE	CRQUBM		; N, Y-CRQUBM
3801				JSR	PC, CKUBAD		; GO CK IF PROG WILL RUN
3802				BR	CLDEX		; GO TO EXIT

3803	CRQUBM:	MOV	PSRCST(R3),R0	;GET SIZE OF DEV ROUT + RDIO + WRIO
3804		BIC	#P4CONS,R0	;RESET PAGE BIT
3805		MOV	#1,R1	;INITIALIZE # OF BITS FOR REGS
3806		MOV	R1,R2	;SET UP BINARY # OF REGS
3807	50\$:	SUB	#8192.,R0	;DECR SIZE BY 4K WORDS
3808		BLE	55\$;FND AREA END? (N,Y-55\$)
3809		ASL	R1	;ADD ANOTHER BIT TO # OF REGS
3810		INC	R1	
3811		INC	R2	;ADD 1 TO BINARY CNT
3812		BR	50\$;GO CK FOR ANOTHER REG
3813	55\$:	MOVB	R1,PUBMAP+1(R3)	;STORE # OF REGS NEEDED IN PROG TBL
3814		MOV	#31.,CLDUMV	;SET UP MAX VALUE FOR THIS
3815		SUB	R2,CLDUMV	;# OF REGS
3816		BIS	#60,R2	;CONVERT BIN # TO ASCII
3817		MOVB	R2,UMNREG	;STORE # IN MSG
3818	CLDAUB:	JSR	R5,CTYPE	;ASK FOR 1ST UNIBUS MAP REG
3819		.WORD	420	
3820		.WORD	UMRGRO-	
3821		.WORD	28.	
3822		JSR	R5,ACTRD	;ISSUE READ FOR REPLY
3823		.WORD	CCMORD-	
3824		.WORD	30.	
3825		JSR	R5,CSCNSP	;SCAN READ DATA FOR REPLY
3826		BIT	#CSCFND,R2	; <CR> OR <LF> ONLY ENTRY?
3827		BEQ	CLDUMT	;N,Y-CLDUMT
3828		JSR	R5,ASDECO	;CONVERT IT TO PACKED DECIMAL
3829		.WORD	CLDER4-	
3830		JSR	R5,DECBIN	;CONVERT PKED DEC TO BINARY
3831		.WORD	CLDER4-	
3832		CMP	R0,#7	;IS # < 7?
3833		BLO	CLDER5	;N,Y-CLDER5
3834		CMP	R0,(PC)+	;IS # > THAN MAX VALUE?
3835	CLDUMV:	.WORD	30.	
3836		BHI	CLDER5	;N,Y-CLDER5
3837		MOV	R0,-(SP)	;SAVE BINARY REG #
3838		MOVB	PUBMAP+1(R3),R2	;GET # OF REG BITS
3839		CLR	R1	;CLEAR HI BITS
3840		SUB	#6,R0	;ADJ REG # FOR BIT POSITION
3841	60\$:	DEC	R0	;SHIFTED BITS ENOUGH?
3842		BEQ	65\$;N,Y-65\$
3843		ASL	R2	;SHIFT BITS ONE POSITION
3844		ROL	R1	
3845		BR	60\$;GO CK BIT SHIFT CNT
3846	65\$:	BIT	R1,MAPMAP	;ANY OF HIGH REGS IN USE?
3847		BNE	CLDER6	;N,Y-CLDER6
3848		BIT	R2,MAPMAP+2	;HOW ABOUT LOW REGS?
3849		BNE	CLDER6	;N,Y-CLDER6
3850	CLDUME:	BIS	R1,MAPMAP	;SET BITS FOR THIS PROG'S REGS
3851		BIS	R2,MAPMAP+2	;IN UB MAP REG USAGE BIT MAP
3852		MOVB	(SP)+,PUBMAP(R3)	;STORE 1ST REG BIN # IN PROG TBL
3853		BR	CLDEX	;GO TO EXIT
3854	CLDUMT:	MOV	#7,-(SP)	;INITIALIZE 1ST REG #
3855		MOVB	PUBMAP+1(R3),R2	;GET # OF REG BITS
3856		CLR	R1	
3857		MOV	#24.,R0	;SET UP MAX # OF SHIFTS ALLOWED
3858		CMP	R2,#3	;HOW MANY REGS NEEDED?

3859				BLO	75\$:BR IF 1
3860				BEQ	70\$:BR IF 2
3861				DEC	R0		:ADJ FOR 3 REGS
3862			70\$:	DEC	R0		:ADJ FOR 2 REGS
3863			75\$:	BIT	R1,MAPMAP		:BITS IN USE IN HI REGS?
3864				BNE	80\$:N,Y-80\$
3865				BIT	R2,MAPMAP+2		:BITS IN USE IN LOW REGS?
3866				BEQ	90\$:Y,N-90\$
3867			80\$:	DEC	R0		:DONE ALL SHIFTS POSSIBLE?
3868				BEQ	CLDER7		:N,Y-CLDER7
3869				INC	(SP)		:BUMP UP 1ST REG #
3870				ASL	R2		:SHIFT TO NEXT GROUP OF REGS
3871				ROL	R1		
3872				BR	75\$:GO CK NEXT GROUP OF REGS
3873			90\$:	MOV	(SP),R0		:GET REG #
3874				JSR	R4,SREG		:SAVE R0 - R4
3875				JSR	PC,CPNAS1		:CONVERT REG # TO DECIMAL ASCII
3876				MOV	R1,UMRGNM		:STORE REG # IN MSG
3877				JSR	R5,CTYPE		:DISPLAY # OF REG SELECTED
3878				.WORD	600		
3879				.WORD	UMRGUD-		
3880				JSR	R4,RREG		:RESTORE R0 - R4
3881				BR	CLDUME		:GO UPDATE REG MAP
3882			000	.ENDC			
3883	012464'	005063	000262	CLDEX:	CLR	PTLGTH+DEVIW7(R3,	:RESET UB MAP USAGE FLAG
3884	012470'	062705	000004		ADD	#4,R5	:INCR PAST ERR RET ADR'S
3885	012474'	000205			RTS	R5	:EXIT IN-LINE
3886							
3887							
3888	012476'	061505		CLDER1:	ADD	(R5),R5	:SET UP ERR RET ADR
3889	012500'	000205			RTS	R5	:EXIT TO ERROR ADR
3890	012502'	005725		CLDER2:	TST	(R5)+	:POINT AT DEV ERR RET ADR
3891	012504'	000774			BR	CLDER1	:GO TO ERROR EXIT
3892	012506'	004567	173512	CLDER3:	JSR	R5,CTYPE	:ISSUE DATA CONV ERROR
3893	012512'	000702			.WORD	702	
3894	012514'	174073			.WORD	CERM02-	
3895	012516'	000167	177152		JMP	CSZREQ	:GO RE-ISSUE MSG REQUEST
3896		001			.IF DF	MM	
3897				CLDER4:	JSR	R5,CTYPE	:ISSUE DATA CONV ERROR
3898					.WORD	702	
3899					.WORD	CERM02-	
3900					BR	CLDERT	:GO RE-ISSUE UB MAP MSG
3901				CLDER5:	JSR	R5,CTYPE	:ISSUE INV REG # MSG
3902					.WORD	712	
3903					.WORD	CERM39-	
3904					BR	CLDERT	:GO RE-ISSUE UB MAP MSG
3905				CLDER6:	JSR	R5,CTYPE	:ISSUE REGS IN USE ERROR MSG
3906					.WORD	603	
3907					.WORD	CERM41-	
3908				CLDERT:	JMP	CLDAUB	:GO RE-ISSUE UB MAP MSG
3909				CLDER7:	TST	(SP)+	:MSKP STACK
3910					JSR	R5,CTYPE	:ISSUE INSUF REGS AVAIL MSG
3911					.WORD	203	
3912					.WORD	CERM40-	
3913			000		.ENDC		
3914							

3915
 3916 001
 3917 012522' 022540
 3918 012524' 022302
 3919 012526' 023162
 3920 012530' 023122
 3921 012532' 026444
 3922 012534' 026514
 3923 012536' 026346
 3924 012540' 021536
 3925 012542' 026214
 3926 012544' 026236
 3927 012546' 026256
 3928 012550' 026206
 3929 012552' 026310
 3930 012554' 026314
 3931 012556' 000000
 3932 000
 3933 001
 3934
 3935
 3936
 3937
 3938
 3939
 3940
 3941
 3942
 3943
 3944
 3945
 3946
 3947
 3948
 3949
 000

```

    CLDTBL: .IF NDF MM
             .WORD  CIOBSY-CLDBAS
             .WORD  CUPGER-CLDBAS
             .WORD  ULIST-CLDBAS
             .WORD  CLIST-CLDBAS
             .WORD  BINASC-CLDBAS
             .WORD  BTASLZ-CLDBAS
             .WORD  DECASC-CLDBAS
             .WORD  CSYSFW-CLDBAS
             .WORD  SETVEC-CLDBAS
             .WORD  CLRVEC-CLDBAS
             .WORD  TSTVEC-CLDBAS
             .WORD  RTNINT-CLDBAS
             .WORD  GETBYT-CLDBAS
             .WORD  PUTBYT-CLDBAS
             .WORD  0
             .ENDC
  
```

```

    CLDTBL: .IF DF MM
             .WORD  CIOBSY-PG6BA+P6CONS
             .WORD  CUPGER-PG6BA+P6CONS
             .WORD  ULIST-PG6BA+P6CONS
             .WORD  CLIST-CLDBAS
             .WORD  BINASC-PG6BA+P6CONS
             .WORD  BTASLZ-PG6BA+P6CONS
             .WORD  DECASC-PG6BA+P6CONS
             .WORD  CSYSFW-PG6BA+P6CONS
             .WORD  SETVEC-PG6BA+P6CONS
             .WORD  CLRVEC-PG6BA+P6CONS
             .WORD  TSTVEC-PG6BA+P6CONS
             .WORD  RTNINT-PG6BA+P6CONS
             .WORD  GETBYT-PG6BA+P6CONS
             .WORD  PUTBYT-PG6BA+P6CONS
             .WORD  0
             .ENDC
  
```

```

;ALL LABELS CONTAINED IN THIS
;TABLE MUST HAVE A MEMORY
;ADDRESS WHICH IS HIGHER THAN
;"CLDBAS".
  
```

.SBTTL VALIDATE MODEL NAME & DEV #'S S/R

```

*****
THIS S/R SCANS THE KEYBOARD READ-IN AREA AND EXPECTS TO
FIND A 4 CHARACTER ASCII MODEL NAME. FOR THE 'ASSIGN' CMD
ENTRY POINT (CVLMD1), THE MODEL NAME HAS ALREADY BEEN SCANNED
BY THE CCFMT3 ROUTINE. THE MODEL NAME WILL BE LOOKED UP IN THE
VALID MODEL NAME TABLE AND ITS MODEL CODE WORD WILL BE EXTRACTED
AND STORED. NEXT, THE KEYBOARD DATA WILL BE SCANNED FOR UP TO
SIXTEEN DEVICE #'S. IF ANY ARE FOUND, THEY WILL BE STORED
AT "CMCDVN" ALSO.

LINKAGE:      JSR      R5,CVLMD1      "ASSIGN" S/R CALL
              JSR      R5,CVLMD2      "ENTER" S/R CALL

DESTROYS REGISTERS:  R0,R1,R2
*****

```

```

3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972 012560' 052767 020000 166656 CVLMD1: BIS      #CVLFLG,CRFLGW      ;SET THE TRAILING OPERAND FLAG
3973 012566' 000412 BR          CVLCOM          ;GO TO COMMON PROCESSING
3974
3975
3976 012570' 004567 175422          CVLMD2: JSR      R5,CSCNSP      ;SCAN FOR THE MDL NAME
3977 012574' 032702 000010          BIT      #CSCFND,R2      ;WAS ANY DATA FOUND?
3978 012600' 001002          BNE      5$              ;N,Y-5$
3979 012602' 000167 000402          JMP      CVLER1          ;GO ISSUE ERROR MSG
3980 012606' 042767 020000 166630 5$: BIC      #CVLFLG,CRFLGW      ;RESET TRAILING OPERAND FLAG
3981 012614' 022701 000004          CVLCOM: CMP      #4,R1      ;MDL NAME = 4 CHAR'S?
3982 012620' 001177          BNE      CVLER2          ;Y,N-CVLER2
3983 012622' 010700          MOV      PC,R0          ;SET UP MDL NAME TBL START ADR
3984 012624' 062700 027746          ADD      #CMDLNM-.,R0
3985 012630' 016701 166742          10$: MOV      COSTAD,R1      ;GET KEYBOARD DATA START ADR
3986 012634' 005760 000004          TST      4(R0)          ;END OF THE MDL NAME TABLE?
3987 012640' 100567          BMI      CVLER2          ;N,Y-CVLER2
3988 012642' 012702 000004          MOV      #4,R2          ;SET UP COMPARE COUNT
3989 012646' 122021          20$: CMPB   (R0)+,(R1)+      ;DATA CHAR = TBL CHAR?
3990 012650' 001403          BEQ      30$            ;N,Y-30$
3991 012652' 060200          ADD      R2,R0          ;POINT TO NXT TBL ENTRY
3992 012654' 005200          INC      R0
3993 012656' 000764          BR       10$            ;GO CK NXT ENTRY
3994 012660' 005302          30$: DEC      R2            ;DECR CHAR CNT
3995 012662' 001371          BNE      20$            ;CNT = 0? (Y,N-20$)
3996 012664' 010346          MOV      R3,-(SP)        ;SAVE WORK REG'S
3997 012666' 010446          MOV      R4,-(SP)
3998 012670' 010703          MOV      PC,R3          ;SET UP STORAGE ADR
3999 012672' 062703 166726          ADD      #CMCDVN-.,R3
4000 012676' 011023          MOV      (R0),(R3)+
4001 012700' 005023          CLR      (R3)+
4002 012702' 116002 000001          MOVB   1(R0),R2          ;STORE MODEL CODE BYTES
4003 012706' 032702 000001          BIT      #1,R2          ;RESET DEV # PNTR & CURR DEV #
4004 012712' 001007          BNE      DFLTMM          ;GET OVR I.D. CODE DISPL VALUE
4005 012714' 010704          MOV      PC,R4          ;THIS A DEFAULT MDL NAME?
4006 012716' 062704 027652          ADD      #DIDTAD-.,R4      ;N,Y-DFLTMM

```


4063	013154'	005704		CVLFIL:	TST	R4		;DEV # CNT = 0?
4064	013156'	001404			BEQ	100\$;N,Y-100\$
4065	013160'	112723	000377	90\$:	MOV	#377,(R3)+		;MOV FILL CHAR TO DEV # STORAGE
4066	013164'	005304			DEC	R4		;DECR DEV # CNT
4067	013166'	001374			BNE	90\$;CNT = 0? (Y,N-90\$)
4068	013170'	012600		100\$:	MOV	(SP)+,R0		;RESTORE WORK REG'S
4069	013172'	012604			MOV	(SP)+,R4		
4070	013174'	012603			MOV	(SP)+,R3		
4071	013176'	010046			MOV	R0,-(SP)		
4072	013200'	032767	020000 166236		BIT	#CVLFLG,CRFLGW		;TEST TRAILING OPERAND FLAG
4073	013206'	000205			RTS	R5		;EXIT IN-LINE
4074								
4075								
4076	013210'	004567	173010	CVLER1:	JSR	R5,CTYPE		;ISSUE NO MDL NAME ERROR
4077	013214'	000043			.WORD	043		
4078	013216'	173617			.WORD	CERM18-		
4079	013220'	004567	173000	CVLER2:	JSR	R5,CTYPE		;ISSUE INV MDL NAME ERROR
4080	013224'	000113			.WORD	113		
4081	013226'	173607			.WORD	CERM18-		
4082	013230'	012604		CVLER3:	MOV	(SP)+,R4		;RESTORE R3 & R4
4083	013232'	012603			MOV	(SP)+,R3		
4084	013234'	004567	172764	CVLR3A:	JSR	R5,CTYPE		;GO REPORT INV UNIT #
4085	013240'	000113			.WORD	113		
4086	013242'	173564			.WORD	CERM17-		
4087	013244'	012605		CVLER4:	MOV	(SP)+,R5		;CLEAR REG R5
4088	013246'	000167	171536		JMP	CSANPN		;GO ISSUE NO PROG # MSG

.SBTTL USER PROGRAM FLAGWORD MODIFICATION S/R'S

```

*****
THESE SUBROUTINES PROVIDE QUICK AND EASY ACCESS TO THE
FLAGWORD CONTAINED IN A USER PROGRAM AREA. THEY ARE
REQUIRED FOR THE EXECUTIVE IN THE MEMORY MANAGEMENT
VERSION. EQUIVALENT S/R'S ARE INCLUDED IN THE NON-MEMORY
MANAGEMENT VERSION FOR COMPATIBILITY PURPOSES.
*****

```

: SET BITS IN THE USER'S FLAGWORD

```

:         JSR      R5,UFWSET      S/R CALL
:         .WORD    BITMSK        BITS TO BE SET
:         .WORD    ADR-          REL ADR OF WD THAT
:                               CONTAINS USER FLGWD ADR.
:                               IF MM, CONTAINS HIGH 16 BITS.

```

: DESTROYS R1

.IF DF MM

.IFTF

```

UFWSET: MOV      (R5)+,10$      ;GET BIT MASK
        MOV      R5,R1        ;GET REL ADR AND MAKE
        ADD      (R5)+,R1     ;IT ABS
        .IFT
        JSR      PC,UFWSUV    ;SET UP A VIRTUAL ADR
        BIS      (PC)+,(R1)   ;SET SPECIFIED BITS
        .IFF
        BIS      (PC)+,2(R1)+ ;SET SPECIFIED BITS
10$:    .IFTF
        .WORD    XXXX
        BR       UFWMEX      ;GO TO EXIT
        .ENDC

```

: CLEAR BITS IN THE USER'S FLAGWORD

```

:         JSR      R5,UFWCLR    S/R CALL
:         .WORD    BITMSK        BITS TO BE CLEARED
:         .WORD    ADR-          REL ADR OF WD THAT
:                               CONTAINS USER FLGWD ADR.
:                               IF MM, CONTAINS HIGH 16 BITS.

```

: DESTROYS R1

.IF DF MM

.IFTF

```

UFWCLR: MOV      (R5)+,10$    ;GET BIT MASK
        MOV      R5,R1        ;GET REL ADR AND MAKE
        ADD      (R5)+,R1     ;IT ABS
        .IFT
        JSR      PC,UFWSUV    ;SET UP A VIRTUAL ADR
        BIC      (PC)+,(R1)   ;CLEAR SPECIFIED BITS
        .IFF

```

```

4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
4101
4102
4103
4104
4105
4106
4107
4108
4109
4110
4111
4112      001
4113
4114 013252' 012567 000006
4115 013256' 010501
4116 013260' 062501
4117
4118
4119
4120
4121 013262' 052731
4122
4123 013264' 000000
4124 013266' 000417
4125      000
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137      001
4138
4139 013270' 012567 000006
4140 013274' 010501
4141 013276' 062501
4142
4143
4144
4145

```

K07

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 USER PROGRAM FLAGWORD MODIFICATION S/R'S

MACY11 27(732) 24-SEP-76 13:54 PAGE 26-1

SEQ 0029

```
4146 013300' 042731          BIC      (PC)+,2(R1)+          ;CLEAR SPECIFIED BITS
4147                                     .IFTF
4148 013302' 000000          10$:      .WORD      XXXX
4149 013304' 000410          BR        UFWMEX          ;GO TO EXIT
4150                                     .ENDC
```

```

4152
4153
4154           ; TEST BITS IN THE USER'S FLAGWORD
4155
4156           JSR      R5,UFWTST      S/R CALL
4157           .WORD    BITMSK        BITS TO BE TESTED
4158           .WORD    ADR-          REL ADR OF WD THAT
4159                                     CONTAINS USER FLGWD ADR.
4160                                     IF MM, CONTAINS HIGH 16 BITS.
4161           BR       LABEL          EXECUTED IF BITS NOT SET (BEG)
4162
4163           ; DESTROYS R1
4164
4165           CO1
4166           .IF DF MM
4167           .IFTF
4167 013306' 012567 000006 UFWTST: MOV      (R5)+,10$      ;GET BIT MASK
4168 013312' 010501          MOV      R5,R1          ;GET REL ADR AND MAKE
4169 013314' 062501          ADD      (R5)+,R1      ;IT ABS
4170           .IFT
4171           JSR      PC,UFWSUV      ;SET UP A VIRTUAL ADR
4172           BIT      (PC)+,(R1)    ;TEST SPECIFIED BITS
4173           .IFF
4174 013316' 032731          BIT      (PC)+,2(R1)+    ;TEST SPECIFIED BITS
4175           .IFTF
4176 013320' 000000 10$:   .WORD    XXXX
4177 013322' 001401          BEQ      UFWMEX          ;ANY BITS SET? (Y,N-UFWMEX)
4178 013324' 005725          TST      (R5)+          ;BYPASS THE BR INST
4179           .IFF
4180 013326' 000205 UFWMEX: RTS      R5          ;EXIT IN-LINE
4181           .IFT
4182 UFWMEX: MOV      (SP)+,2*KPARE6  ;RESTORE KERNEL'S PAR 6
4183           RTS      R5          ;EXIT IN-LINE
4184
4185           ;SET UP VIRTUAL ADR OF UFW
4186
4187 UFWSUV: MOV      (SP),-(SP)      ;MAKE ROOM ON THE STK
4188           MOV      2*KPARE6,2(SP) ;SAVE KERNEL'S PAR 6
4189           JSR      R4,SREG        ;SAVE REG'S R0 - R4
4190           MOV      PC,R2          ;GET ADR OF S/R LINK WD
4191           ADD      #20$-.,R2
4192           SUB      R2,R1          ;MAKE ADR REL AGAIN
4193           MOV      R1,(R2)        ;STORE IT
4194           JSR      R5,G22BAD      ;CONVERT IT TO A 22 BIT ADR
4195 20$:   .WORD    XXXX
4196           JSR      PC,CVTVAD      ;CONVERT IT TO A VIRTUAL ADR
4197           MOV      R1,2(SP)       ;PUT ADR IN R1 ON THE STK
4198           JSR      R4,RREG        ;RESTORE REGISTERS
4199           RTS      PC          ;EXIT IN-LINE
4200           .ENDC
4201           .IF DF MM
4202           .PAGE
4203           .SBTTL CHECK FOR UP TO A 3 PAGE MEMORY AREA
4204
4205           ;*****
4206           ;
4207           ; USING THE CONTENTS OF R1 AS THE HIGH 16 BITS OF A 22 BIT ADDRESS,

```

THIS SUBROUTINE WILL DETERMINE THE AMOUNT OF AVAILABLE MEMORY FROM THIS ADDRESS UP TO THE NEXT USER PROGRAM OR THE END OF MEMORY. THE SIZE OF THIS AREA (MAX OF 12K WORDS) IS RETURNED IN R2 AS A COUNT OF 32 WORD BLOCKS. AN ERROR WILL BE REPORTED IF THE MEMORY AREA IS ALREADY OCCUPIED AND R2 WILL BE SET TO 0.

LINKAGE: JSR PC,CKF3PG S/R CALL
ENTRY: R1 = HIGH 16 BIT ADR OF AREA TO BE CHECKED
EXIT: R2 = AREA LENGTH IN 32 WORD BLOCKS
DESTROYS R2

```

4208      ;
4209      ;
4210      ;
4211      ;
4212      ;
4213      ;
4214      ;
4215      ;
4216      ;
4217      ;
4218      ;
4219      ;
4220      ;
4221      ;
4222      ;
4223      ;
4224      ;
4225      ;
4226      ;
4227      ;
4228      ;
4229      ;
4230      ;
4231      ;
4232      ;
4233      ;
4234      ;
4235      ;
4236      ;
4237      ;
4238      ;
4239      ;
4240      ;
4241      ;
4242      ;
4243      ;
4244      ;
4245      ;
4246      ;
4247      ;
4248      ;
4249      ;
4250      ;
4251      ;
4252      ;
4253      ;
4254      ;
4255      ;
4256      ;
4257      ;
4258      ;
4259      ;
4260      ;
4261      ;
4262      ;
4263      ;

CKF3PG: JSR      R4,SREG      ;SAVE REGS 0 - 4
        MOV      PC,R0      ;SET UP CNTRL ROUT TBL ADR
        ADD      #CRTBL-2-.,R0
        MOV      R1,R3      ;MOVE AREA START ADR TO R3
        MOV      R3,R4      ;SET AREA END ADR TO AREA
        ADD      #600,R4     ;START + 12K WORDS
        MOV      CRMEND,R2   ;GET REAL MEM END ADR
        INC      R2         ;ADJUST IT
        CMP      R4,R2      ;IS END ADR OUT OF MEM?
        BLOS    10$        ;Y,N-10$
        MOV      R2,R4      ;USE END OF MEM AS AREA END ADR
10$:    TST      (R0)+      ;POINT TO NXT TBL ENTRY
        TST      (R0)       ;THIS PROG ENTRY EMPTY?
        BEQ     10$        ;N,Y-10$
        CMP      (R0),#177777 ;END OF THE TBL?
        BEQ     50$        ;N,Y-50$
        MOV      CRTEXT(R0),R2 ;GET PROG'S LGTH IN 32 WD BLKS
        BIC      #FIXADR,R2  ;RESET FIX ADR FLG IF THERE
        ADD      (R0),R2     ;GET PROG'S END ADR
        CMP      R3,(R0)     ;AREA START ADR VS. PROG'S START ADR
        BLO     20$        ;BR IF AREA ADR IS LOWER
        BHI     30$        ;BR IF AREA ADR IS HIGHER
        CMP      R0,CURCTE   ;THIS SAME PROG SLOT?
        BEQ     10$        ;N,Y-10$
        BR      40$        ;GO TO ERROR POINT
20$:    CMP      R4,(R0)     ;AREA END ADR =< PROG'S START ADR?
        BLOS    10$        ;N,Y-10$
        MOV      (R0),R4     ;USE PROG'S ST ADR AS NEW AREA END ADR
        BR      10$        ;GO CK NXT ENTRY
30$:    CMP      R3,R2      ;AREA START ADR => PROG'S END ADR?
        BHI     10$        ;N,Y-10$
40$:    MOV      R3,R4      ;SET AREA END ADR TO AREA START ADR
        JSR      R5,CTYPE    ;ISSUE "AREA ALREADY OCCUPIED"
        .WORD   402         ;ERROR MSG
50$:    SUB      R3,R4      ;GET LGTH OF THE AREA
        MOV      R4,4(SP)    ;STORE IT IN R2 ON THE STK
        JSR      R4,RREG     ;RESTORE REGS 0 - 4
        RTS     PC          ;EXIT IN-LINE
        .PAGE
    
```

.SBTTL MOVE USER PROGRAMS UP AND DOWN S/R'S

```

*****
THE FOLLOWING TWO SUBROUTINES ARE USED TO MOVE USER PROGRAMS IN
MEMORY. THE ADDRESSES SUPPLIED ARE THE HIGH 16 BITS OF 22 BIT
ADDRESSES AND THE PROGRAM'S SIZE IS SPECIFIED IN THE NUMBER OF
32 WORD BLOCKS. THE MOVE UP S/R, WHICH IS USED TO RELOCATE
PROGRAMS UPWARDS IN MEMORY, STARTS AT THE END OF THE PROGRAM
AND MOVES CONSECUTIVELY LOWER BLOCKS. THE MOVE DOWN S/R, WHICH
IS USED TO RELOCATE PROGRAMS DOWNWARD IN MEMORY, STARTS AT THE
BEGINNING OF THE PROGRAM AND MOVES CONSECUTIVELY HIGHER BLOCKS.
*****
    
```

;MOVE USER PROGRAM UP

```

;JSR    PC,MOVPGU      S/R CALL
;R2 = # OF 32 WORD BLOCKS
;R3 = "FROM" 16 HI BIT ADR
;R4 = "TO" 16 HI BIT ADR
;DESTROYS R0,R1,R2,KPAR6 - ALTERS R3,R4
    
```

```

MOVPGU:  MOV    R2,-(SP)          ;SAVE PROG LENGTH
          ADD    R2,R3          ;POINT TO END OF "FROM" AREA
          DEC    R3
10$:     MOV    R3,MOVADR        ;STORE "FROM" HI 16 BIT ADR
          JSR    PC,GET32W      ;GET 32 WDS OF PROG
          MOV    R4,MOVADR        ;STORE "TO" HI 16 BIT ADR
          JSR    PC,PUT32W      ;PUT 32 WDS IN NEW AREA
          DEC    (SP)          ;DECR PROG LENGTH CNT
          BEQ    MOVEX          ;CNT = 0? (N,Y-MOVEX)
          DEC    R3            ;DECR PROG AREA ADR'S
          DEC    R4
          BR     10$           ;GO DO NXT 32 WDS
MOVEX:   TST    (SP)+          ;CLEAN UP STACK
          RTS    PC            ;EXIT IN-LINE
    
```

;MOVE USER PROGRAM DOWN

```

;JSR    PC,MOVPGD      S/R CALL
;R3 = # OF 32 WD BLKS
;R4 = ADR OF "FROM" HI 16 BIT ADR
;R5 = "TO" HI 16 BIT ADR
;DESTROYS R0,R1,R2,R3,KPAR6 - ALTERS (R4),R5
    
```

```

MOVPGD:  MOV    R5,-(SP)          ;SAVE PROG'S NEW AREA ADR
30$:     MOV    (R4),MOVADR      ;STORE "FROM" HI 16 BIT ADR
          JSR    PC,GET32W      ;GET 32 WDS OF PROG
          MOV    R5,MOVADR        ;STORE "TO" HI 16 BIT ADR
          JSR    PC,PUT32W      ;PUT 32 WDS IN NEW AREA
          INC    (R4)          ;ADD 1 TO "FROM" ADR
          INC    R5            ;ADD 1 TO "TO" ADR
          DEC    R3            ;DONE ALL 32 WD BLKS?
          BNE   30$           ;Y,N-30$
    
```

4264
4265
4266
4267
4268
4269
4270
4271
4272
4273
4274
4275
4276
4277
4278
4279
4280
4281
4282
4283
4284
4285
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319

```

4320          MOV      (SP)+,(R4)          ;STORE PROG'S NEW ADR
4321          RTS      PC                  ;EXIT IN-LINE
4322
4323
4324          GET32W: MOV      #012120,GPINST ;SET UP "MOV (R1)+,(R0)+" INST
4325                BR      GP32CM        ;GO TO COM PROC
4326
4327          PUT32W: MOV      #012021,GPINST ;SET UP "MOV (R0)+,(R1)+" INST
4328          GP32CM: JSR      R5,G22BAD    ;GET ADR AS 22 BITS
4329                .WORD    MOVADR-
4330                JSR      PC,CVTVAD    ;SET UP A VIRTUAL ADR IN R1
4331                MOV      PC,R0        ;SET UP WORK BUFFER ADR
4332                ADD      #USTKWK-.,R0
4333          GPINST: MOV      #32.,R2      ;SET UP WORD CNT
4334                NOP                    ;GET OR PUT A WORD
4335                DEC      R2            ;DECR WD CNT
4336                BNE     GPINST        ;CNT = 0? (Y,N-GPINST)
4337                RTS      PC          ;EXIT IN-LINE
4338
4339          MOVADR: .WORD    0            ;*;HI 16 BIT ADR OF MEM BLK
4340                .PAGE
4341                .SBTTL  FIND LARGEST/LOWEST FREE MEMORY AREA
4342
4343
4344
4345
4346
4347
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358
4359
4360
4361
4362
4363
4364
4365
4366
4367
4368
4369
4370
4371
4372
4373
4374
4375

```

```

*****
THIS SUBROUTINE SCANS THE UNUSED AREAS OF MEMORY LOOKING FIRST
FOR A 12K WORD AREA. IF NONE ARE FOUND, THE NEXT LARGEST AREA
IS SELECTED. IF TWO OR MORE AREAS ARE THE SAME SIZE AS THE LARGEST, THE
AREA LOWEST IN MEMORY WILL BE SELECTED. THE STARTING ADDRESS OF THE AREA
AND ITS SIZE WILL BE RETURNED IN REGISTERS. IF THE UNIBUS MAP IS NOT
BEING USED, THE SELECTED AREA WILL BE WITHIN THE 18 BIT ADDRESSING AREA.
LINKAGE:      JSR      PC,FINDFM
EXIT:         R0 = HI 16 BITS OF AREA
              R1 = # OF 32 WD BLKS
*****

```

```

4360          FINDFM: JSR      R4,SREG      ;SAVE REGS 0 - 4
4361                MOV      PC,R0        ;SET UP CNTRL ROUT TBL ADR
4362                ADD      #CRTBL-.,R0
4363                MOV      PC,R1        ;SET UP WORK BUFFER ADR
4364                ADD      #USTKWK-.,R1
4365          10$:   MOV      R1,R2          ;SAVE IT IN R2 FOR LATER
4366                MOV      CRTEXT(R0),CRTEXT(R1) ;MOVE TBL EXT WD TO WK BUF
4367                BIC      #FIXADR,CRTEXT(R1) ;RESET FIXED ADR FLG IF THERE
4368                MOV      (R0)+,(R1)+ ;STORE USER PROG ADR
4369                CMP      #177777,-2(R0) ;END OF CNTRL ROUT TBL?
4370                BNE     10$           ;Y,N-10$
4371                MOV      R2,R1        ;SET UP ADR OF 2ND
4372                ADD      #68.,R1      ;WORK AREA
4373          20$:   MOV      R1,-(SP)     ;SAVE 2ND AREA ADR ON STK
4374                MOV      CRMEND,-(SP) ;PUT END OF MEM ADR ON STK
4375                CLR      R3          ;RESET ENTRY FOUND FLG
4376                MOV      R2,R0      ;POINT AT 1ST AREA IN WK BUF

```

4376	30\$:	TST	(R0)	: ENTRY EMPTY?
4377		BEQ	40\$: N, Y-40\$
4378		BIT	#UNIMAP, CSYSFW	: USING THE UNIBUS MAP?
4379		BNE	35\$: N, Y-35\$
4380		BIT	#BITS22, CSYSFW	: USING 22 BIT ADRS?
4381		BEQ	35\$: Y, N-35\$
4382		CMP	(R0), #10000	: THIS PROG ABOVE 18 BIT ADRS?
4383		BLO	35\$: Y, N-35\$
4384		CLR	(R0)	: RESET ITS ENTRY
4385		BR	40\$: GO CK NEXT SLOT
4386	35\$:	INC	R3	: SET THE ENTRY FND FLG
4387		CMP	(R0), (SP)	: THIS LOWEST ADR SO FAR?
4388		BHIS	40\$: Y, N-40\$
4389		MOV	(R0), (SP)	: SAVE ITS ADR AS NEW LOWEST
4390		MOV	R0, R4	: SAVE ITS ENTRY ADR
4391	40\$:	TST	(R0)+	: POINT TO NXT TBL ENTRY
4392		CMP	#177777, (R0)	: END OF THE TBL?
4393		BNE	30\$: Y, N-30\$
4394		TST	(SP)+	: TAKE OLD ADR OFF STK
4395		TST	R3	: ANY PROG'S LEFT?
4396		BEQ	50\$: Y, N-50\$
4397		MOV	CRTEXT(R4), CRTEXT(R1)	: STORE THIS PROG'S INFO AS THE
4398		MOV	(R4), (R1)+	: NEXT HIGHEST IN MEM
4399		CLR	(R4)	: RESET THIS PROG'S SLOT
4400		BR	20\$: GO LOOK FOR NXT HIGHEST PROG
4401	50\$:	MOV	#177777, (R1)+	: STORE A TBL TERMINATOR
4402		MOV	(SP)+, R0	: GET 2ND AREA ADR AGAIN
4403		MOV	CPAREA, R1	: INIT PREV PROG END ADR
4404		MOV	R1, -(SP)	: INIT PREV PROG-1 END ADR
4405		MOV	R1, -(SP)	: INIT PREV PROG START ADR
4406		CLR	R4	: RESET LONGEST AREA CNT
4407	60\$:	CMP	#177777, (R0)	: END OF SORTED PROG ADR TBL?
4408		BNE	70\$: Y, N-70\$
4409		MOV	CRMEND, R2	: GET END OF MEM ADR
4410		INC	R2	: ADJ IT
4411		BIT	#UNIMAP, CSYSFW	: USING THE UNIBUS MAP?
4412		BNE	65\$: N, Y-65\$
4413		BIT	#BITS22, CSYSFW	: USING 22 BIT ADRS?
4414		BEQ	65\$: Y, N-65\$
4415		CMP	R2, #10000	: MEM END > 18 BITS?
4416		BLOS	65\$: Y, N-65\$
4417		MOV	#10000, R2	: SET UP NEW MEM ADR
4418		CMP	R1, R2	: PREV PROG END AT 18 BITS OR ABOVE?
4419		BLO	65\$: Y, N-65\$
4420		MOV	(SP), R2	: USE IT START ADR AS END OF MEM ADR
4421		MOV	2(SP), R1	: GET END ADR OF PROG BEFORE IT
4422	65\$:	SUB	R1, R2	: SUB PREV PROG END ADR FROM IT
4423		CMP	R2, #600	: AREA => 3 PAGES IN SIZE?
4424		BHIS	100\$: N, Y-100\$
4425		CMP	R2, R4	: IS IT LARGER THAN PREV LARGEST?
4426		BLOS	120\$: Y, N-120\$
4427		MOV	R2, R4	: SET UP ITS LENGTH INSTEAD
4428		BR	110\$: GO TO FINAL PROCESSING
4429	70\$:	CMP	(R0), R1	: THIS PROG BUTT AGAINST PREV PROG?
4430		BNE	90\$: Y, N-90\$
4431	80\$:	MOV	R1, 2(SP)	: SAVE AS PREV PROG-1 END ADR

```

4432      MOV      (R0),(SP)      ;SAVE AS PREV PROG START ADR
4433      MOV      (R0),R1        ;USE ITS ADR + LGTH AS NEW
4434      ADD      CRTEXT(RC),R1   ;PREV PROG END ADR
4435      TST      (R0)+          ;POINT TO NXT PROG SLOT
4436      BR       60$           ;GO CK NXT PROG
4437      90$:    MOV      (R0),R2  ;GET ITS ADR
4438      SUB      R1,R2          ;SUB PREV PROG END ADR FROM IT
4439      CMP      R2,#600       ;OPEN AREA => 3 PAGES?
4440      BHIS    100$          ;N,Y-100$
4441      CMP      R2,R4        ;IS IT THE LARGEST SO FAR?
4442      BLOS    80$           ;Y,N-80$
4443      MOV      R1,R3        ;SAVE PREV PROG END ADR AS OPEN AREA ADR
4444      MOV      R2,R4        ;SAVE ITS SIZE
4445      BR       80$           ;GO CK FOR LARGER AREA
4446      100$:  MOV      #600,R4 ;SET AREA SIZE TO 3 PAGES
4447      110$:  MOV      R1,R3   ;USE PREV PROG END ADR AS START ADR
4448      120$:  ADD      #4,SP    ;REMOVE ADRS FROM STACK
4449      MOV      R3,(SP)      ;STORE ADR IN R0 ON THE STK
4450      MOV      R4,2(SP)     ;STORE LGTH IN R1 ON THE STK
4451      JSR      R4,RREG      ;RESTORE REGS 0 - 4
4452      RTS      PC          ;EXIT IN-LINE
4453      .PAGE
4454      .SBTTL  SET UP MIDL'S AND FREE MEM'S ABSOLUTE ADDRESSES

```

```

*****
: THIS S/R WILL FIRST STORE THE 18/22 BIT ABSOLUTE ADDRESS OF
: "FREE" AND THEN CALCULATE AND STORE THE 18/22 BIT ABSOLUTE
: ADDRESS OF "MIDL". NEXT, THE PAR/PDR STORAGE AREAS FOR THE
: "FREE" AREA ARE TAILORED FOR FREE'S LOCATION AND LENGTH. IF
: THE UNIBUS MAP IS BEING USED, 18 BIT VIRTUAL ADDRESSES, WHICH
: ARE BASED UPON MAP REGISTERS 3 THRU 5, WILL ALSO BE STORED.
: THESE UNIBUS MAP REGISTERS WILL BE INITIALIZED WITH THE
: CORRECT ABSOLUTE ADDRESSES AT THIS TIME.
: LINKAGE:      JSR      PC,SUMIFM
: DESTROYS R0,R1,R2
*****

```

```

4471      SUMIFM: MOV      R3,-(SP)      ;SAVE R3 & R4
4472      MOV      R4,-(SP)
4473      JSR      R5,G22BAD          ;CONVERT FREE MEM START TO
4474      .WORD    CFMST-            ;22 BITS
4475      MOV      PC,R3             ;GET ADR OF STORAGE AREA
4476      ADD      #FREEAD+2-.,R3   ;STORE FREE'S 18/22 BIT
4477      MOV      R2,(R3)+         ;ABS ADR
4478      MOV      R0,(R3)+         ;USING THE UNIBUS MAP
4479      BIT      #UNIMAP,CSYSFW    ;Y,N-20$
4480      BEQ      20$              ;STORE FREE'S 18 BIT ADR BASED
4481      CLR      (R3)+             ;UPON MAP REGS 3 THRU 5
4482      MOV      #P3CONS,(R3)+    ;SET UP LOOP CNT
4483      MOV      #3,R1            ;INITIALIZE MAP REG ADR
4484      MOV      #UBMAP3,R4       ;STORE ADR IN MAP REG
4485      10$:  MOV      R0,(R4)+
4486
4487

```



```

4488      MOV      R2,(R4)+
4489      ADD      #20000,R0          ;POINT TO NEXT 4K WORDS
4490      ADC      R2
4491      DEC      R1                ;DONE 3 REGS?
4492      BNE     10$              ;Y,N-10$
4493      SUB     #60000,R0        ;RE-ADJUST FREE'S ABS ADR
4494      SBC      R2
4495      CLR     MIDADJ          ;RESET ADJ FACTOR
4496      MOV     CFMEND,R1       ;GET END OF FREE MEM
4497      INC     R1              ;ALLOW FOR LAST 32 WD BLK
4498      SUB     CFMST,R1        ;FIND FREE MEM SIZE
4499      MOV     R1,-(SP)        ;SAVE IT FOR LATER
4500      CLC
4501      ROR     R1              ;DIVIDE IT BY 2
4502      BCC     30$            ;BR IF EVEN # OF 32 WD BLKS
4503      MOV     #40,MIDADJ     ;STORE ADJ FACTOR
4504      MOV     #6,R4          ;CONVERT 32 WD CNT TO A
4505      ASL     R1              ;BYTE CNT
4506      DEC     R4
4507      BNE     40$
4508      BIS     (PC)+,R1        ;SET IN ADJ FACTOR
4509      MIDADJ: .WORD XXXX
4510      MOV     PC,R3           ;SET UP ADR OF MIDL'S
4511      ADD     #MIDLAD-,R3     ;STORAGE AREA
4512      MOV     #P3CONS,(R3)   ;STORE MIDL'S 16 BIT VIRTUAL ADR
4513      ADD     R1,(R3)+
4514      ADD     R1,R0          ;ADD DISPL TO FREE'S ADR
4515      ADC     R2
4516      MOV     R2,(R3)+       ;STORE MIDL'S 18/22 BIT
4517      MOV     R0,(R3)+       ;ABS ADR
4518      BIT     #UNIMAF,CSYSFW ;USING THE UNIBUS MAP?
4519      BEQ     60$            ;Y,N-60$
4520      CLR     (R3)+           ;SET MIDL'S 18 BIT VIRT ADR BASED
4521      MOV     -8,(R3),(R3)    ;UPON MAP REGS 3 THRU 5
4522      MOV     (SP)+,R1        ;GET FREE'S SIZE SAVED EARLIER
4523      MOV     CFMST,R3        ;GET FREE'S START HI 16 BITS
4524      MOV     PC,R4          ;SET UP STORAGE AREA ADR
4525      ADD     #FMPAR3-,R4
4526      MOV     #3,-(SP)        ;SET UP PAGE CNT
4527      MAPEXT: CMP     R1,#200 ;SIZE < 4K WORDS?
4528      70$:   BLO     80$      ;N,Y-80$
4529      MOV     #PDRCON,16,(R4) ;STORE 4K PDR VALUE
4530      MOV     R3,(R4)+        ;STORE HI 16 BITS OF ADR
4531      ADD     #200,R3         ;BUMP UP ADR BY 4K
4532      SUB     #200,R1         ;DECREASE SIZE BY 4K
4533      DEC     (SP)           ;DONE 3 PAGES?
4534      BNE     70$            ;Y,N-70$
4535      BR     100$           ;GO TO EXIT
4536      80$:   TST     R1        ;PARTIAL PAGE?
4537      BEQ     90$            ;Y,N-90$
4538      DEC     R1              ;ADJ # OF BLKS FOR PLF
4539      SWAB    R1              ;SET UP PLF FIELD
4540      BIS     #6,R1           ;SET IN ACF BITS
4541      MOV     R1,16,(R4)     ;STORE PARTIAL PAGE PDR
4542      MOV     R3,(R4)+        ;STORE ITS START ADR
4543      DEC     (SP)           ;DONE 3 PAGES?

```

4544
4545
4546
4547
4548
4549
4550
4551
4552
4553
4554
4555
4556
4557
4558
4559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588
4589
4590
4591
4592
4593
4594
4595

```

90$: BEQ 100$ ;N.Y-100$
      CLR 16.(R4) ;RESET UNUSED PDR
      CLR (R4)+ ;RESET UNUSED PAR
      DEC (SP) ;DONE 3 PAGES?
      BNE 90$ ;Y.N-90$
100$: TST (SP)+ ;TAKE CNT OFF STK
      MOV (SP)+,R4 ;RESTORE R3 & R4
      MOV (SP)+,R3
      RTS PC ;EXIT IN-LINE
      .PAGE
      .SBTTL MAP USER PROGRAM'S PAR & PDR REGISTER VALUES

```

THIS SUBROUTINE WILL INITIALIZE THE PROGRAM TABLE STORAGE AREA FOR THIS PROGRAM'S PAR AND PDR REGISTERS. THE FOLLOWING ARE THE REGISTER ASSIGNMENTS:

```

PAR/PDR 0,1,2 = PROGRAM'S AREA
          3,4,5 = FREE MEMORY AREA
          6     = MPG'S SHARED CODE
          7     = I/O PAGE

```

USES PART OF THE "SUMIFM" S/R TO MAP PAR/PDR'S 0,1,2.

LINKAGE: JSR PC,MAPUPG

```

ENTRY: R3 CONTAINS PROG TBL ADR
        R4 CONTAINS PROGRAM'S CONTROL ROUTINE TABLE ENTRY ADR

```

DESTROYS R0,R1,R2

```

MAPUPG: MOV R3,-(SP) ;SAVE R3 & R4
        MOV R4,-(SP)
        MOV PC,R0 ;SET UP ADR OF VALUES FOR
        ADD #FMPAR3-.,R0 ;PAR'S & PDR'S 3 THRU 7
        MOV R3,R1 ;POINT TO THEIR STORAGE AREA IN
        ADD #PUPARS+6,R1 ;THIS PROG'S TBL
        MOV #5,R2 ;SET UP REG CNT
10$: MOV 16.(R0),16.(R1) ;STORE PDR VALUE
      MOV (R0)+,(R1)+ ;STORE PAR VALUE
      DEC R2 ;DONE ALL REGS?
      BNE 10$ ;Y.N-10$
      MOV CRTEXT(R4),R1 ;GET PROG'S LGTH IN 32 WD BLKS
      BIC #FIXADR,R1 ;RESET FIXED ADR FLG IF THERE
      MOV R3,R4 ;SET UP ADR OF STORAGE AREA IN
      ADD #PUPARS,R4 ;THIS PROG'S TBL
      MOV PFWADR(R3),R3 ;GET STARTING HI 16 BIT ADR
      BR MAPEXT ;USING PART OF SUMIFM S/R, DO MAPPING
      .ENDC

```

000

4597
4598
4599
4600
4601
4602
4603
4604
4605
4606
4607
4608
4609 013330' 010746
4610 013332' 062716 166132
4611 013336' 012667 166120
4612 013342' 000207
4613
4614
4615
4616
4617
4618
4619 001
4620
4621 000
4622
4623
4624
4625
4626
4627
4628
4629 001
4630
4631
4632
4633
4634 013344' 010046
4635 000
4636 013346' 032700 000001
4637 013352' 001410
4638 013354' 005316
4639 013356' 004567 025540
4640 013362' 173753
4641 013364' 004567 172634
4642 013370' 000400
4643 013372' 173743
4644 013374' 012600
4645 001
4646
4647 000
4648 013376' 000205

.SBT'L CONTROL ROUTINE SMALL S/R AREA

: THIS S/R INITIALIZES THE CURRENT COMMAND DATA PCOUNTER
: TO THE START OF THE CONSOLE READ-IN AREA.
: LINKAGE: JSR PC,CSUPTR

: CSUPTR: MOV PC,-(SP) ;RESET DATA PNTR TO FIRST
: ADD #CCMDRD-(SP) ;CHAR OF READ IN AREA
: MOV (SP)+,CPNTR
: RTS PC ;EXIT IN-LINE

: THIS S/R TESTS THE OCTAL ADDRESS IN R0 AND IF ODD,
: SETS IT TO EVEN AND ISSUES A MSG INDICATING THIS FACT.
: IF DF MM
: R2 CONTAINS HIGH 6 BITS OF THE OCTAL ADDRESS
: ENDC

LINKAGE: JSR R5,CKODAD
MAY DESTROY REGISTERS: R0,R1,R2

: IF DF MM
: IFT
CKODAD: MOV R2,-(SP) ;SAVE ORG 22 BIT ADR
: MOV R0,-(SP)
: IFF
CKODAD: MOV R0,-(SP) ;SAVE ORG ADR
: ENDC
: BIT #1,R0 ;IS OCTAL ADR ODD?
: BEQ 10\$;(Y,N-10\$)
: DEC (SP) ;SET IT TO NXT LOWEST EVEN ADR
: JSR R5,BINAS8 ;CNVRT ORG ADR TO ASCII &
: .WORD CAAMSG- ;PUT IN MSG
: JSR R5,CTYPE ;ISSUE ADR ADJUSTED MSG
: .WORD 400
: .WORD CAAMSG-
10\$: MOV (SP)+,R0 ;RESTORE FINAL ADR
: IF DF MM
: MOV (SP)+,R2
: ENDC
: RTS R5 ;EXIT IN-LINE

4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660 013400' 010746
4661 013402' 062716 166754
4662 013406' 012667 166170
4663 013412' 000207
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677 013414' 010302
4678 013416' 062702 000032
4679 013422' 010701
4680 013424' 062701 166174
4681 013430' 012700 000012
4682 013434' 012122
4683 013436' 005300
4684 013440' 001375
4685 013442' 000205
4686
4687
4688
4689
4690 001
4691
4692
4693
4694
4695 000
4696
4697
4698
4699
4700
4701
4702
4703 001
4704
4705 013444' 020467 166002

```

*****
:
: THE ADDRESS CONTAINED AT THE STANDARD ERROR RETURN
: ADDRESS STORAGE LOCATION "CERRET" WILL BE SET TO
: "CENTPB" BY THIS S/R.
:
: LINKAGE: JSR PC,CSUERR
:
*****
CSUERR: MOV PC, -(SP) ;GET ADR OF ERROR RETURN POINT
ADD #CENTPB, (SP)
MOV (SP)+, CERRET ;STORE IT IN ERROR REPORTING ROUT
RTS PC ;EXIT IN-LINE

*****
:
: THIS S/R MOVES THE 10 WORDS OF DEVICE INFO AT "CMCDVN"
: TO THE PROGRAM TABLE POINTED TO BY R3.
:
: LINKAGE: JSR R5,CSTDVI
:
: DESTROYS REGISTERS: R0,R1,R2
:
*****
CSTDVI: MOV R3,R2 ;SET UP PROG AREA WORK ADR
ADD #PMDLCD,R2 ;SET UP DEVICE INFO STORAGE ADR
MOV PC,R1 ;GET ADR OF DEVICE INFO
ADD #CMCDVN-,R1
MOV #10,R0 ;SET UP WD CNT
20$: MOV (R1)+,(R2)+ ;MOVE DEV INFO TO PROG TBL
DEC R0
BNE 20$
RTS R5 ;EXIT IN-LINE

*****
:
: IF DF MM
: IFF
: THIS S/R VALIDATES THAT THE ADDRESS IN R4 IS EITHER
: IFT
: THIS S/R VALIDATES THAT THE ADDRESS IN R3 & R4 IS EITHER
: ENDC
: WITHIN REAL MEMORY OR IN THE DEVICE REGISTERS' AREA.
: IF NOT, AN ERROR IS REPORTED.
:
: LINKAGE: JSR R5,CMEMCK
:
*****
:
: IF DF MM
: IFF
CMEMCK: CMP R4,CRMEND ;ADR HIGHER THAN REAL MEM ADR?

```

```

4706 013450' 101403      BLOS      24$      ;Y,N-24$
4707 013452' 020427 160000  CMP      R4,#160000 ;IN DEV REG RANGE?
4708 013456' 103401      BLO      26$      ;Y,N-26$
4709 013460' 000205      RTS      R5      ;EXIT IN-LINE
4710 013462' 004567 172536 24$: JSR      R5,CTYPE ;GO REPORT OUT OF MEM ADR
4711 013466' 000003      .WORD   003
4712 013470' 173225      .WORD   CERM09-.
4713
4714 CMEMCK: JSR      R5,G22BAD ;CONVERT REAL MEM END ADR TO
4715      .WORD   CRMEND-. ;22 BITS
4716      BIS      #77,R0 ;POINT TO END OF 32 WD AREA
4717      CMP      R3,R2 ;COMPARE HIGH 6 BITS
4718      BHI      20$ ; > MEM? (N,Y-20$)
4719      BLO      50$ ; < MEM? (N,Y-50$)
4720      CMP      R4,R0 ;COMPARE LOW 16 BITS
4721      BLOS     50$ ; = < MEM? (N,Y-50$)
4722 20$: MOV      #3,R2 ;SET UP HIGH BITS FOR 18 BITS
4723      BIT      #BITS22,CSYSFW ;USING 22 BIT ADDRESSING?
4724      BEQ      30$ ;Y,N-30$
4725      MOV      #77,R2 ;SET UP HIGH BITS FOR 22 BITS
4726 30$: CMP      R3,R2 ;COMPARE HIGH 6 BITS
4727      BNE      60$ ; = I/O PAGE ADR? (Y,N-60$)
4728      CMP      R4,#160000 ;COMPARE LOW 16 BITS
4729      BLO      60$ ; < I/O PAGE ADR? (N,Y-60$)
4730 50$: RTS      R5 ;EXIT IN-LINE
4731
4732 60$: JSR      R5,CTYPE ;GO REPORT OUT OF MEM
4733      .WORD   003
4734      .WORD   CERM09-.
4735      .ENDC
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753 013472' 010346      CISDNM: MOV      R3,-(SP) ;SAVE WORK REG
4754 013474' 016700 166076  MOV      CDSTAD,R0 ;GET DATA STARTING ADR
4755 013500' 016703 166074  MOV      CDCNT,R3 ;GET DATA LENGTH
4756 013504' 060300      ADD      R3,R0 ;SET UP RIGHT HAND END ADR
4757 013506' 005300      DEC      R0
4758 013510' 010002      MOV      R0,R2 ;MOVE IT TO WORK REG
4759 013512' 005001      CLR      R1 ;CLEAR CHAR CNT REG
4760 013514' 121227 000060 10$: CMPB     (R2),#'0 ;CHAR IN OPER FROM 0 - 9?
4761 013520' 103407      BLO      20$ ;Y,N-20$

```

000

THIS S/R SCANS THE LAST CMND WORD BACKWARDS LOOKING FOR NON-DECIMAL CHARACTERS.

LINKAGE: JSR R5,CISDNM S/R CALL

DESTROYS REGISTERS: R0,R1,R2

UPON EXIT: R0 = RIGHT END ADR OF WORD
R1 = # OF DECIMAL DIGITS
R2 = ADR OF LEFTMOST DECIMAL DIGIT

J08

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 CONTROL ROUTINE SMALL S/R AREA

MACY11 27(732) 24-SEP-76 13:54 PAGE 27-3

SEQ 0101

4762 013522' 121227 000071
4763 013526' 101004
4764 013530' 005302
4765 013532' 005201
4766 013534' 020103
4767 013536' 001366
4768 013540' 005202
4769 013542' 012603
4770 013544' 000205

20\$:

CMPB (R2),#9
BHI 20\$
DEC R2
INC R1
CMP R1,R3
BNE 10\$
INC R2
MOV (SP)+,R3
RTS R5

;DECR DATA ADR
;INCR CHAR CNT
;SCANNED ENTIRE WORD?
;Y,N-10\$
;POINT ADR AT PREV CHAR
;RESTORE WORK REG
;EXIT IN-LINE

4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827

013546' 010700
013550' 062700 027022
013554' 010502
013556' 062502
013560' 005760 000004
013564' 100406
013566' 020160 000004
013572' 001403
013574' 062700 000006
013600' 000767
013602' 012022
013604' 011012
013606' 000205

```
FDVNAM: MOV PC,R0 ;SET UP ADR OF DEV NAME TBL
         ADD #CMDLNM-.,R0
         MOV R5,R2 ;GET REL OUTPUT ADR &
         ADD (R5)+,R2 ;MAKE IT ABS
10$:     TST 4(R0) ;END OF THE TABLE?
         BMI 20$ ;N,Y-20$
         CMP R1,4(R0) ;THIS THE MODEL CODE?
         BEQ 20$ ;N,Y-20$
         ADD #6,R0 ;POINT AT NXT TBL ENTRY
         BR 10$ ;GO CK IT
20$:     MOV (R0)+,(R2)+ ;STORE DEV NAME
         MOV (R0),(R2)
         RTS R5 ;EXIT IN-LINE
```

USING THE DEVICE MODEL CODE IN R1, FIND THE DEVICE'S NAME IN
THE VALID DEVICE NAME TABLE AND MOVE IT TO THE ADR SPECIFIED.
LINKAGE: JSR R5,FDVNAM S/R CALL
.WORD ADR- REL ADR TO PUT 4 CHAR NAME
R1 CONTAINS MODEL CODE WORD
DESTROYS REGISTERS: R0,R2

USING THE CURRENT CONTROL ROUTINE TABLE ENTRY ADDRESS
IN R4, THIS S/R WILL COMPUTE THE ENTRY'S EQUIVALENT
PROGRAM NUMBER IN ASCII.
LINKAGE; JSR PC,CPNASC SUBROUTINE CALL
ENTRY: R4 CONTAINS ADR OF CURR CONTROL ROUT TBL ENTRY
EXIT: R1 CONTAINS EQUIVALENT PROG # IN ASCII
DESTROYS REGISTERS: R0,R1,R2

013610' 010701
013612' 062701 020402
013616' 010400
013620' 160100
013622' 006000
013624' 004567 025334
013630' 165760
013632' 016701 165756
013636' 000207
001

```
CPNASC: MOV PC,R1 ;SET UP CTRL ROUT TBL START ADR
         ADD #CRTBL-2-.,R1
         MOV R4,R0 ;COMPUTE ENTRY'S NUMBER
         SUB R1,R0
         ROR R0
CPNAS1: JSR R5,BTASLB ;CONVERT # TO DECIMAL ASCII
         .WORD CWA-
         MOV CWA+4,R1 ;GET LAST TWO DIGITS
         RTS PC ;EXIT IN-LINE
         .IF DF MM
         .PAGE
*****
```

4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883

```

: THIS S/R TAKES THE HIGH ORDER 16 BITS OF AN ADR
: STORED AT THE SUPPLIED LABEL AND CONVERTS IT TO A 22 BIT
: ADR. THE LOW 16 BITS WILL BE IN R0 AND THE HIGH 6 BITS
: IN R2. THE LOW 6 BITS OF THE 22 BIT ADR WILL BE 0'S.

```

```

LINKAGE: JSR R5,G22BAD S/R CALL
          .WORD LABEL- REL ADR OF HIGH 16 BITS

```

```

DESTROYS R0,R1,R2

```

```

G22BAD: MOV R5,R1 ;MAKE REL ADR ABSOLUTE
        ADD (R5)+,R1
        MOV (R1),R0 ;GET HIGH 16 BITS
        CLR R2 ;RESET REG FOR HIGH 6 BITS
10$: MOV #6,R1 ;SET UP LOOP CNT
      ASL R0 ;SHIFT BITS ONE TO THE LEFT
      ROL R2
      DEC R1 ;SHIFTED 6 BITS?
      BNE 10$ ;Y,N-10$
      RTS R5 ;EXIT IN-LINE

```

```

: THE 22 BIT ADDRESS IN R0 & R2 WILL BE CONVERTED TO A
: VIRTUAL ADDRESS BASED UPON PAR 6 WITH THE CORRECT
: DISPLACEMENT. THIS ADDRESS WILL BE RETURNED IN R1.
: ALSO, KERNEL'S PAR 6 WILL BE INITIALIZED TO THE CORRECT
: BASE ADDRESS.

```

```

LINKAGE: JSR PC,CVTVAD S/R CALL

```

```

DESTROYS R1 AND KPAR6

```

```

CVTVAD: JSR R4,SREG ;SAVE REG'S R0 - R4
        MOV R0,R1 ;EXTRACT LOW 6 BITS
        BIC #177700,R1
        BIS #P6CONS,R1 ;BASE IT UPON PAR 6
10$: MOV #6,R3 ;SET UP LOOP CNT
      ASR R2 ;SHIFT ADR RIGHT 1 BIT
      ROR R0
      DEC R3 ;DONE 6 BITS?
      BNE 10$ ;Y,N-10$
      MOV R0,#KPAR6 ;STORE HIGH 16 BITS IN KPAR 6
      MOV R1,2(SP) ;STORE VIRTUAL ADR IN R1 ON THE STK
      JSR R4,RREG ;RESTORE REGISTERS
      RTS PC ;EXIT IN-LINE

```

.PAGE

```

: THIS SUBROUTINE WILL RESET THE UNIBUS MAP USAGE

```


4884
4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939

.; BITS IN THE "MAPMAP" WORDS FOR THE CURRENT PROGRAM.
.; IF THE CURRENT PROGRAM DOES NOT USE THE UNIBUS MAP,
.; NO ACTIONS WILL TAKE PLACE.

LINKAGE: JSR PC,MAPHKP S/R CALL

R3 = PROG TABLE ADR
R4 = CONTROL ROUT TBL ENTRY ADR

DESTROYS R0,R1,R2

```
MAPHKP: TST     PUBMAP(R3)           ;THIS PROG USE THE UNIBUS MAP?
        BEQ     30$                ;Y,N-30$
        MOVB   PUBMAP(R3),R0       ;GET 1ST MAP REG #
        MOVB   PUBMAP+1(R3),R2    ;GET # OF REG BITS
        CLR    R1
        SUB    #6,R0              ;CHG REG # INTO SHIFT CNT
10$:    DEC     R0                  ;DONE ALL SHIFTS?
        BEQ     20$                ;N,Y-20$
        ASL    R2                  ;SHIFT REG BITS ONE POSITION
        ROL    R1
        BR     10$
20$:    BIC    R1,MAPMAP           ;GO CK SHIFT CNT NOW
        BIC    R2,MAPMAP+2        ;RESET THIS PROG'S UNIBUS
30$:    RTS    PC                  ;MAP USAGE BITS
        ;EXIT IN-LINE
```

.PAGE

THIS SUBROUTINE CHECKS THAT A PROGRAM'S I/O AREA
(PROG TBL + DEVICE ROUTINE + RDIO + WRIO) DOES NOT
EXCEED 18 BIT ADDRESSING. THIS CHECK IS MADE IF 22 BIT
ADDRESSING IS BEING USED AND THE UNIBUS MAP IS NOT.
ALSO, IT IS MADE ONLY FOR PROGRAMS THAT WOULD NORMALLY
REQUIRE THE UNIBUS MAP. IF IT FAILS THE TEST, A
MESSAGE WILL BE ISSUED TO WARN THE USER THAT HIS PROGRAM
MIGHT NOT WORK.

LINKAGE: JSR PC,CKUBAD S/R CALL

KPAR4 = PROG'S HI 16 BIT BASE ADR
R3 = PROG TBL ADR

DESTROYS R0,R1,R2

```
CKUBAD: BIT     #BITS22,CSYSFW     ;USING 22 BIT ADDRESSING?
        BEQ     CKUBAX            ;Y,N-CKUBAX
        MOV    PSRCST(R3),R0      ;GET SIZE OF DEV ROUT+RDIO+WRIO
        BIC    #P4CONS,R0        ;IN 32 WD BLKS
10$:    MOV    #6,R1
        ASR    R0
        DEC    R1
```

```

4940      BNE      10$
4941      INC      RO
4942      ADD      2*PAR4,RO      ;ADD IN BASE ADR
4943      CMP      RO,#10000     ;PROG I/O AREA > 18 BIT ADR?
4944      BLOS    CKUBAX        ;Y,N-CKUBAX
4945      JSR      R5,CTYPE      ;ISSUE WARNING MSG THAT PROG
4946      .WORD   400           ;MAY NOT RUN
4947      .WORD   CERM42-.
4948      CKUBAX: RTS          ;EXIT IN-LINE
4949      .PAGE
4950
4951      ;*****
4952      ;
4953      ; THIS ROUTINE IS A HANDLER FOR 'TRAP' INSTRUCTIONS USED
4954      ; BY THE MEMORY MANAGEMENT VERSION OF MPG AND IS ENTERED
4955      ; FROM THE TRAP VECTOR AT LOCATION 34. MPG USES TRAPS
4956      ; WHENEVER IT IS IN USER MODE AND DESIRES TO GO TO KERNEL
4957      ; MODE. THE TRAP NUMBER DETERMINES WHICH POINT IT WILL
4958      ; BRANCH TO WITHIN MPG. INVALID TRAPS WILL RESULT IN AN
4959      ; ERROR REPORT.
4960      ;
4961      ;*****
4962
4963      TRP34: MOV      RO,-(SP)      ;SAVE RO
4964      MOV      2(SP),RO          ;GET ADR OF TRAP INST + 2
4965      MFPI     -2(RO)           ;GET TRAP INST ON STACK
4966      MOVB    (SP)+,RO         ;GET TRAP I.D. BYTE
4967      BEQ     TRAPER          ;IS IT 0? (N,Y-TRAPER)
4968      BIT     #1,RO           ;IS IT ODD?
4969      BNE     TRAPER          ;N,Y-TRAPER
4970      CMP     RO,#MAXTRP      ;IS IT A VALID TRAP?
4971      BHI     TRAPER          ;Y,N-TRAPER
4972      ADD     PC,RO           ;INDEX INTO TRAP ADR TBL
4973      ADD     #TRPTBL-2-.,RO
4974      ADD     (RO),RO         ;MAKE TRAP ADR ABS
4975      RTS     RO              ;GO TO THIS TRAP'S POINT
4976
4977      TRAPER: MOV     (SP)+,RO    ;RESTORE RO
4978      BR      TRP34          ;GO TO INVALID TRAP ROUT
4979
4980      TRPTBL: .WORD   UPGRET-.    ; 2 - USER PROGRAM RETURN TRAP
4981      .WORD   KSETVC-.        ; 4 - SET INTERRUPT VECTOR TRAP
4982      .WORD   KCLRVC-.        ; 6 - CLEAR INTERRUPT VECTOR TRAP
4983      .WORD   UPRTKM-.        ;10 - USER PROGRAM PRINT TRAP
4984
4985      MAXTRP= 10
4986      .PAGE
4987      ;*****
4988      ;
4989      ; THIS ROUTINE PROCESSES ALL INVALID TRAPS WHICH OCCUR ON
4990      ; VECTORS 4, 10, 34, 114, AND 250. REGISTERS RO THRU R7, THE PSW,
4991      ; MMRO THRU MMR3, AND THE PAR/PDR REGISTERS OF THE FAILING
4992      ; STATE WILL BE DISPLAYED. IF THE TRAP OCCURRED DURING A USER
4993      ; PROGRAM, AN ATTEMPT WILL BE MADE TO KILL IT AND KEEP
4994      ; RUNNING. IF IT OCCURRED DURING MPG'S OPERATIONS, A HALT
4995

```

```

4996      ; WILL RESULT.
4997      ;
4998      ;*****
4999
5000      TRP04: MOV      #4, -(SP)      ;SET UP TRAP 04 ADR
5001      BR          TRPCOM          ;GO TO COMMON CODE
5002
5003      TRP10: MOV      #10, -(SP)     ;SET UP TRAP 10 ADR
5004      BR          TRPCOM          ;GO TO COMMON CODE
5005
5006      TRP34I: MOV     #34, -(SP)     ;SET UP TRAP 34 ADR
5007      BR          TRPCOM          ;GO TO COMMON CODE
5008
5009      TRP114: MOV     #114, -(SP)    ;SET UP TRAP 114 ADR
5010      BR          TRPCOM          ;GO TO COMMON CODE
5011
5012      TRP250: MOV     #250, -(SP)    ;SET UP TRAP 250 ADR
5013
5014      TRPCOM: MOV     R5, -(SP)      ;SAVE REGS R0 THRU R5
5015      JSR          R4, SREG
5016      MOV          12, (SP), R0     ;GET TRAP'S ADR
5017      JSR          R5, BINASC       ;CONVERT TO ASCII & PUT
5018      .WORD        TRPADR-         ;IN THE MSG
5019      CLR          TRPSTK+2         ;SET STK DISPL TO 0
5020      MOV          #8, R3           ;SET UP # OF WORDS ON THE STK
5021      MOV          PC, R1           ;GET ADR IN MSG WHERE ASCII
5022      ADD          #CMPRD-, R1      ;DATA GOES
5023      TRPSTK: MOV     0(SP), R0      ;GET WORD FROM STK
5024      JSR          R5, BINASC       ;CONVERT IT TO ASCII & PUT IN MSG
5025      INC          R1               ;POINT TO NXT FLD IN MSG
5026      ADD          #2, TRPSTK+2    ;POINT AT NEXT WD ON STK
5027      DEC          R3               ;DONE ALL WDS ON STK?
5028      BNE         TRPSTK           ;Y N-TRPSTK
5029      MOV          SP, R4           ;CALC STK PNTR AT TIME
5030      ADD          #16, R4          ;OF THE TRAP
5031      MOV          (R4)+, R0        ;GET PSW
5032      MOV          R0, R5           ;SAVE IT
5033      JSR          R5, BINASC       ;CONVERT PSW TO ASCII & PUT
5034      .WORD        TRPPSW-         ;IN THE MSG
5035      CLRB         TRP40M          ;INITIALIZE MSG TO AN 11/40
5036      MOV          #MMR0, R0        ;GET MMR0 CONTENTS
5037      JSR          R5, BINASC       ;CONVERT & PUT IN THE MSG
5038      .WORD        TRPMR0-
5039      MOV          #MMR2, R0        ;GET MMR2 CONTENTS
5040      JSR          R5, BINASC       ;CONVERT & PUT IN THE MSG
5041      .WORD        TRPMR2-
5042      BIT          #CPU40, CSYSFW   ;IS THIS AN 11/40?
5043      BNE         10$              ;N, Y-10$
5044      MOVB        #40, TRP40M      ;SET MSG FOR OTHER THAN AN 11/40
5045      MOV          #MMR1, R0        ;GET MMR1 CONTENTS
5046      JSR          R5, BINASC       ;CONVERT & PUT IN THE MSG
5047      .WORD        TRPMR1-
5048      MOV          #MMR3, R0        ;GET MMR3 CONTENTS
5049      JSR          R5, BINASC       ;CONVERT & PUT IN THE MSG
5050      .WORD        TRPMR3-
5051      10$: BIT     #140000, R5      ;IN USER MODE AT TIME OF TRAP?

```

5052		BEQ	20\$; Y, N-20\$
5053		MFPI	SP	; GET USER MODE STK PNTR
5054		MOV	(SP)+, R0	
5055		BR	30\$; BYPASS KERNEL MODE PROC
5056	20\$:	MOV	R4, R0	; SET UP KERNEL STK PNTR
5057	30\$:	JSR	R5, BINASC	; CONVERT STK PNTR & PUT
5058		.WORD	TRPSP-	; IN THE MSG
5059		JSR	R5, CTYPE	; ISSUE MSGS WITH VALUES OF
5060		.WORD	420	; ALL REGISTERS
5061		.WORD	TRPMMSG-	
5062		.WORD	57	
5063		JSR	R5, CTYPE	
5064		.WORD	400	
5065		.WORD	CMMPAR+7-	
5066		JSR	R5, CTYPE	
5067		.WORD	400	
5068		.WORD	TRPPSM-	
5069		JSR	R5, CTYPE	; ISSUE FAILING STATE MSG
5070		.WORD	400	
5071		.WORD	TRPFMS-	
5072		MOV	#UPARD, R3	; SET UP USER MODE PAR &
5073		MOV	#UPDRD, R5	; PDR ADR'S
5074		BIT	#140000, 16.(SP)	; WERE WE IN USER MODE?
5075		BNE	35\$; N, Y-35\$
5076		MOV	#KPARD, R3	; SET UP KERNEL'S PAR &
5077		MOV	#KPDRO, R5	; PDR ADR'S
5078	35\$:	JSR	PC, PRTPAD	; DISPLAY PAR & PDR REGS
5079		BIT	#CUSPGR, CSYSFW	; IS A USER PROG RUNNING?
5080		BNE	40\$; N, Y-40\$
5081		JMP	MPGTRP	; GO TO MPG TRAP PROCESSING
5082	40\$:	MOV	CPGCPT, R5	; INITIALIZE USER PROG ADR
5083		BIT	#340, 16.(SP)	; WAS INT BEING SERVICED?
5084		BEQ	41\$; Y, N-41\$
5085		JSR	R5, CTYPE	; ISSUE 'INT BEING SERVICED'
5086		.WORD	400	; MSG
5087		.WORD	TRPISV-	
5088		MOV	#KPAR5, R5	; GET PROG ADR FRM INT SRV BASE PAGE
5089		BR	43\$; BYPASS OTHER TEST
5090	41\$:	TST	CPGCPT	; USER PROG IN CONTROL?
5091		BEQ	MPGTRP	; Y, N-MPGTRP
5092	43\$:	TST	CSYSFW	; CONS REQ BEING PROCESSED?
5093		BMI	MPGTRP	; N, Y-MPGTRP
5094		MOV	#KPAR4, -(SP)	; SAVE KERNEL'S PAR 4
5095		MOV	R5, #KPAR4	; SET UP PROG'S BASE ADR
5096		MOV	#P4CONS, R3	; SET UP FLGWD ADR
5097		MOV	PASCIN(R3), TRPPNM	; MOVE PROG # TO PROG # MSG
5098		BIC	#ACTIVE, (R3)	; RESET ITS ACTIVE FLG
5099		BIT	#SETDED, (R3)	; DID IT SET PRT DEV DEDICATED?
5100		BEQ	50\$; Y, N-50\$
5101		BIT	#CHPINT, CSYSFW	; IS THE PRINTER BEING USED?
5102		BEQ	45\$; Y, N-45\$
5103		MOV	CLISTI, R0	; GET ADR OF PRINTER INFO
5104		CLR	#8, (R0)	; RESET PRINTER'S INT ENABLE
5105	45\$:	BIC	#CTUDED+CRUDED+CHPINT, CSYSFW	; RESET PRT DEV DED FLGS
5106		BIC	#SETDED+CTPRIO, (R3)	
5107	50\$:	MOV	R3, R0	; GET ADR OF DEV ROUT KILL

```

S108      ADD      #PTLGTH+DKILAD,RO      ;ROUT ADR
S109      TST      (RO)                   ;IS THERE A KILL ROUT?
S110      BEQ      55$                    ;Y,N-55$
S111      ADD      (RO),RO                 ;MAKE ITS ADR ABS
S112      MOV      @#KPAR6, -(SP)         ;SAVE PAR 6
S113      MOV      SPPAR6,@#KPAR6        ;SET UP SHARED CODE BASE ADR
S114      JSR      R5,(RO)                ;GO TO KILL ROUT
S115      MOV      (SP)+,@#KPAR6         ;RESTORE PAR 6
S116      55$:    MOV      (SP)+,@#KPAR4   ;RESTORE PAR 4
S117      JSR      R5,CTYPE               ;DISPLAY USER PROG #
S118      .WORD   400
S119      .WORD   TRPPMG-
S120      MOV      #-1,STKADJ            ;HSKP STACK ADJ VALUE
S121      MOV      #CKILEX-PG6BA+P6CONS,RO ;SET UP USR MODE ABORT ADR
S122      BIT      #140000,16.(SP)       ;WAS USER IN KERNEL MODE?
S123      BNE      64$                    ;Y,N-64$
S124      BIT      #340,16.(SP)          ;SERVICING AN INTERRUPT?
S125      BNE      58$                    ;N,Y-58$
S126      MOV      PC,RO                  ;SET UP KERNEL MODE ABORT ADR
S127      ADD      #CKILEK-.,RO
S128      BR       64$
S129      58$:    MOV      SP,R1           ;GO STORE THIS ADR
S130      ADD      #16.,R1                ;POINT TO USER WORDS - 1
S131      MOV      PC,R2                  ;ON THE STK
S132      ADD      #INTRET-.,R2          ;SET UP ADR FOR CK IF USER'S
S133      60$:    TST      (R1)+          ;DEV TYPE OF INT
S134      CMP      (R1),R2               ;POINT TO NXT STK WD
S135      BEQ      62$                    ;THIS USER INT RET ADR?
S136      CMP      R1,#STACK-2          ;N,Y-62$
S137      BLO      60$                   ;REACHED STK UPPER LIMIT?
S138      BR       66$
S139      62$:    SUB      #12.,R1        ;CAN'T FIND RETURN - LEAVE RET ADR ALONE
S140      MOV      SP,R2                 ;ALLOW FOR R5 THRU R0
S141      ADD      #18.,R2               ;DETERMINE IF USER WORDS ARE ON
S142      MOV      R1,R3                 ;THE STK
S143      SUB      R2,R3
S144      BMI      66$
S145      MOV      R3,STKADJ             ;BR IF STK NOT AS EXPECTED
S146      MOV      16.(SP),-(R1)         ;STORE STK ADJ VALUE
S147      MOV      PC, -(R1)             ;MOVE PSW UP ON STK
S148      ADD      #UTRPRT-.,(R1)       ;SET UP NEW RET ADR
S149      BR       66$
S150      64$:    MOV      RO,14.(SP)    ;BYPASS ADR STORING
S151      66$:    JSR      R4,RREG        ;STORE IT IN PC ON THE STK
S152      MOV      (SP)+,R5              ;RESTORE REGS R0 THRU R5
S153      CMP      (SP)+,#250           ;WAS THIS A MEM MGMNT TRAP?
S154      BNE      68$                    ;Y,N-68$
S155      MOV      #1001,@#MMMR0        ;CLEAR MEM MGMNT ERRORS
S156      68$:    TST      STKADJ        ;IS STK TO BE ADJUSTED?
S157      BMI      UTRPEX                ;Y,N-UTRPEX
S158      ADD      (PC)+,SP              ;ADJ STK PNTR IF TRAP IN USR INT
S159      STKADJ: .WORD   0
S160      UTRPEX: RTI                    ;RETURN & TRY TO KEEP RUNNING
S161
S162      UTRPRT: MOV      (SP)+,R5      ;RESTORE REGS R5 THRU R0
S163      MOV      (SP)+,R4

```

```

S164      MOV      (SP)+,R3
S165      MOV      (SP)+,R2
S166      MOV      (SP)+,R1
S167      MOV      (SP)+,R0
S168      JMP      RTNINT          ;GO TO USER INT EXIT
S169
S170
S171      MPGTRP: JSR      RS,CTYPE      ;ISSUE MPG I.D. MSG
S172      .WORD    400
S173      .WORD    TRPMPG-
S174      CLR      @MMMRO          ;DISABLE MEM MGMNT
S175      HALT
S176      JMP      RST2          ;TRAP WHILE MPG IN CONTROL
S177      .ENDC          ;GO TO RESTART 2
000
S178
S179      001
S180
S181      .IF DF MM
S182      .IFT
S183      .TITLE  MAINDEC-11-DTMGA-C    MPG INTERPRETER
S184      .TITLE  MAINDEC-11-DTMA-B    MPG INTERPRETER
000      .ENDC

```

.SBTTL USER PROGRAM LANGUAGE PROCESSOR

THIS SUBROUTINE PROCESSES USER ENTRIES EITHER FROM THE CONSOLE TERMINAL OR FROM THE PROGRAM SAVE DEVICE. IT ACCEPTS THE ENTRIES ONE LINE AT A TIME, PACKS IT INOT A COMPRESSED FORM IN AN INTERNAL BUFFER, AND THEN INSERTS THIS PACKED STRING IN THE PROPER POSITION IN RELATION TO OTHER PACKED ENTIERES AS DETERMINED BY THE LINE NUMBER. CHECKS ARE MADE TO INSURE THAT THE PACKED CODE AND THE EVENTUAL MACHINE CODE WILL BOTH FIT WITHIN THE REMAINING MEMORY SPACE. THIS PROCESS CONTINUES UNTIL THE WORD 'DONE' IS DETECTED IN PLACE OF A LINE NUMBER.

CALLING SEQUENCE JSR R5,ENTER (MODIFY)(FETCH(COMPIL)
ENTRY- R3 POINTS TO PROGRAM TABLE
EXIT- BRANCHES TO COMPILER

5186
5187
5188
5189
5190
5191
5192
5193
5194
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237
5238
5239
5240
5241

013640' 105267 002217
013644' 116767 002226 002214
013652' 116767 002220 002207
013660' 112767 000061 002202
013666' 116767 002204 002175
013674' 105267 002202
013700' 016363 000022 000024
013706' 016367 000022 001130
013714' 160367 001124
013720' 000402
013722' 105267 002134
013726' 004567 001272
013732' 000415
013734' 105267 001074
013740' 016363 000022 000024
013746' 016367 000022 001070
013754' 160367 001064
013760' 004567 002120
013764' 000042
013766' 010367 005276
013772' 004567 002310
013776' 105767 007247
014002' 001011
014004' 105767 007237
014010' 001051
014012' 105767 007232
014016' 001414
014020' 105067 007224

ENTER: INCB LENTER ;SET 'ENTER' FLAG
MOVBLASZER,LINNR ;SET
MOVBLASZER,LINNR+1 ;SAVED
MOVBL#61,LINNR+2 ;LINE NUMBER
MOVBLASZER,LINNR+3 ;TO 0010
INCBFSTIME ;SET FIRST TIME FLAG
MOVPSRCST(R3),POBJST(R3) ;INIT OBJECT START
MOVPSRCST(R3),LPROLN
SUBR3,LPROLN ;INIT PROG LENGTH
BRLLPGFC
MODIFY: INCB LMODFY ;SET 'MODIFY' FLAG
LLPGFC: JSR R5,UPCI ;GET LINE FROM CONSOLE
BRLLPPAK ;THEN GO PACK IT
FETCH: INCB LFETCH ;SET 'FETCH' FLAG
MOVPSRCST(R3),POBJST(R3) ;INIT OBJECT START
MOVPSRCST(R3),LPROLN
SUBR3,LPROLN ;INIT PROGRAM LENGTH
JSRR5,UPLIO ;GET LINE FROM 'SAVE' DEV
WORDLLPTFF-
LLPPAK: MOV R3,LPKPTP ;SAVE PROG TBL PNTR
JSRR5,PACK ;PACK THE ENTRY
TSTBLEERROR ;ERROR FLAG SET?
BNE LLPTFF ;YES- GO TEST FETCH FLAG
TSTBLDELET ;TEST DELETE FLAG
BNE LLPDEL ;BRANCH IF SET
TSTBLIGNOR ;IGNORE FLAG SET?
BEQLLPCOL ;NO -GO CK OBJ LENGTH
CLRBLIGNOR ;RESET IGNORE FLAG

5242	014024'	000506		BR	LLPGAL		;GO GET ANOTHER LINE
5243	014026'	105767	001002	LLPTFF: TSTB	LFETCH		;FETCH FLAG SET?
5244	014032'	001503		BEQ	LLPGAL		;NO-GET ANOTHER LINE
5245	014034'	004567	172164	JSR	R5,CTYPE		;TYPE FETCH ERROR MSG
5246	014040'	000402		.WORD	402		
5247	014042'	001004		.WORD	LFTMSG-		
5248	014044'	000167	000630	JMP	LLPIEN		;GO INSERT END
5249	014050'	010700		LLPCOL: MOV	PC,RO		
5250	014052'	062700	007212	ADD	#LPKBUF-. ,R0		;POINT R0 AT PACK BUFR
5251	014056'	004567	013350	JSR	R5,LCKOBL		;GO CHECK OBJECT LENGTH
5252	014062'	016701	000756	MOV	LPROLN,R1		;GET OLD PROG LENGTH
5253	014066'	060001		ADD	RO,R1		;ADD NEW OBJ LENGTH
5254	014070'	010702		MOV	PC,R2		
5255	014072'	062702	007172	ADD	#LPKBUF-. ,R2		;POINT R2 AT PACK BUFR
5256	014076'	111200		MOVSB	(R2),RO		
5257	014100'	060001		ADD	RO,R1		;ADD NEW SOURCE LENGTH
5258	014102'	062701	000012	ADD	#10. R1		;ADD STORAGE FOR 'END'
5259	014106'	060301		ADD	R3,R1		;ADD PROG TBL PNTR
5260	014110'	020167	165334	CMP	R1,CVMEND		;IS THERE ROOM
5261	014114'	101402		BLOS	LLPTDN		;YES-TEST DONE FLAG
5262	014116'	000167	000546	JMP	LLPWRN		;NO-TYPE WARNING
5263	014122'	105767	007120	LLPTDN: TSTB	LDONE		;DONE FLAG SET?
5264	014126'	001500		BEQ	LLPTFS		;NO-TEST IF FIRST STMT
5265	014130'	000167	000504	JMP	LLPTEN		;YES-TEST END
5266	014134'	004567	013230	LLPDEL: JSR	R5,LFINLN		;SEARCH FOR EXISTING LINE
5267	014140'	000130		.WORD	LLPLNF-		
5268	014142'	010100		MOV	R1,RO		
5269	014144'	004567	013262	JSR	R5,LCKOBL		;GET OBJ LENGTH
5270	014150'	160067	000670	SUB	RO,LPROLN		;SUB OBJ LENGTH
5271	014154'	111100		MOVSB	(R1),RO		;GET LENGTH OF DELETED ENTRY
5272	014156'	060100		ADD	R1,RO		;ADD TO ENTRY POINTER
5273	014160'	020063	000024	LLPLP1: CMP	RO,POBJST(R3)		
5274	014164'	001402		BEQ	LLPDDN		
5275	014166'	012021		MOV	(RO)+,(R1)+		;MOVE REST DOWN ON TOP
5276	014170'	000773		BR	LLPLP1		
5277	014172'	010163	000024	LLPDDN: MOV	R1,POBJST(R3)		;UPDATE OBJ START ADDR
5278	014176'	160100		SUB	R1,RO		;RECOVER ENTRY LENGTH
5279	014200'	160067	000640	SUB	RO,LPROLN		;SUB SOURCE LENGTH
5280	014204'	010701		MOV	PC,R1		
5281	014206'	062701	020060	ADD	#LINBUF+4-. ,R1		
5282	014212'	010700		MOV	PC,RO		
5283	014214'	062700	000726	ADD	#LDLMSG-. ,RO		
5284	014220'	012702	000011	MOV	#9. R2		
5285	014224'	112021		LLPLP2: MOVSB	(RO)+,(R1)+		
5286	014226'	005302		DEC	R2		
5287	014230'	001375		BNE	LLPLP2		
5288	014232'	004567	171766	JSR	R5,CTYPE		;TYPE 'NNNN DELETED'
5289	014236'	000400		.WORD	400		
5290	014240'	020022		.WORD	LINBUF-		
5291	014242'	105767	000566	LLPGAL: TSTB	LFETCH		;FETCH FLAG SET
5292	014246'	001404		BEQ	LLPMFC		;YES-GET LINE FROM SAVE
5293	014250'	004567	001656	JSR	R5,UPLI		
5294	014254'	177552		.WORD	LLPTFF-		
5295	014256'	000643		BR	LLPPAK		
5296	014260'	004567	000740	LLPMFC: JSR	R5,UPCI		;NO-GET LINE FROM CONSOLE
5297	014264'	000167	177476	LLPPKB: JMP	LLPPAK		

5298	014270'	010701		LLPLNF:	MOV	PC,R1	
5299	014272'	062701	017775		ADD	#LINBUF+5-. ,R1	
5300	014276'	010700			MOV	PC,R0	
5301	014300'	062700	000653		ADD	#LNFMMSG-. ,R0	
5302	014304'	012702	000012		MOV	#10. ,R2	
5303	014310'	112021		LLPLP3:	MOVB	(R0)+,(R1)+	
5304	014312'	005302			DEC	R2	
5305	014314'	001375			BNE	LLPLP3	
5306	014316'	004567	171702		JSR	R5,CTYPE	;TYPE 'NNNN NOT FOUND'
5307	014322'	000402			.WORD	402	
5308	014324'	017736			.WORD	LINBUF-	
5309	014326'	000745			BR	LLPGAL	
5310	014330'	016301	000022	LLPTFS:	MOV	PSRCST(R3),R1	;TEST IF FIRST STATEMENT
5311	014334'	020163	000024	LLPLP4:	CMP	R1,POBJST(R3)	
5312	014340'	001401			BEQ	LLPADL	;YES-GO ADD A LINE
5313	014342'	000417			BR	LLPCNH	;NO CHECK IF NEXT OLD HIGHER
5314	014344'	111200		LLPADL:	MOVB	(R2),R0	;GET SOURCE LENGTH
5315	014346'	060063	000024		ADD	R0,POBJST(R3)	;ADD LENGTH TO OBJ START
5316	014352'	060067	000466		ADD	R0,LPROLN	;ADD TO PROG LENGTH
5317	014356'	010200			MOV	R2,R0	
5318	014360'	004567	013046		JSR	R5,LCKOBL	;GO CHECK OBJ LENGTH
5319	014364'	060067	000454		ADD	R0,LPROLN	;ADD TO PROG LENGTH
5320	014370'	111200			MOVB	(R2),R0	;GET SOURCE LENGTH AGAIN
5321	014372'	112221		LLPLP5:	MOVB	(R2)+,(R1)+	;MOVE NEW STATEMENT
5322	014374'	005300			DEC	R0	;IN AS FINAL STATEMENT
5323	014376'	001375			BNE	LLPLP5	
5324	014400'	000720		LLPGLB:	BR	LLPGAL	;GO GET ANOTHER LINE
5325	014402'	126261	000002	LLPCNH:	CMPB	2(R2),2(R1)	;CHECK NEW LN VS OLD
5326	014410'	001402			BEQ	LLPCLS	;IF = CHECK LSB LIN NBR
5327	014412'	103473			BLO	LLPINL	;IF < INSERT NEW LINE
5328	014414'	000405			BR	LLPMBH	;MUST BE >
5329	014416'	126261	000003	LLPCLS:	CMPB	3(R2),3(R1)	;CHECK LSB OF LINE NBR
5330	014424'	001404			BEQ	LLPRPL	;IF = REPLACE LINE
5331	014426'	103465			BLO	LLPINL	;IF < INSERT NEW LINE
5332	014430'	111100		LLPMBH:	MOVB	(R1),R0	
5333	014432'	060001			ADD	R0,R1	;BUMP R1 TO END OF OLD
5334	014434'	000737			BR	LLPLP4	;NOW SEE IF IT FITS
5335	014436'	010100		LLPRPL:	MOV	R1,R0	
5336	014440'	004567	012766		JSR	R5,LCKOBL	;GET OLD OBJ LENGTH
5337	014444'	160067	000374		SUB	R0,LPROLN	;SUB FROM PROG LENGTH
5338	014450'	010200			MOV	R2,R0	
5339	014452'	004567	012754		JSR	R5,LCKOBL	;GET NEW OBJ LENGTH
5340	014456'	060067	000362		ADD	R0,LPROLN	;ADD TO PROG LENGTH
5341	014462'	010267	000350		MOV	R2,LLPSNP	;SAVE NEW POINTER
5342	014466'	010167	000346		MOV	R1,LLPSOP	;SAVE OLD POINTER
5343	014472'	111200			MOVB	(R2),R0	;GET NEW LENGTH
5344	014474'	111102			MOVB	(R1),R2	;GET OLD LENGTH
5345	014476'	160200			SUB	R2,R0	;SUBTRACT OLD LENGTH
5346	014500'	060067	000340		ADD	R0,LPROLN	;ADJUST PROG LENGTH
5347	014504'	060063	000024		ADD	R0,POBJST(R3)	;AND OBJECT START
5348	014510'	005700			TST	R0	;IF R0 NEGATIVE
5349	014512'	100411			BMI	LLPMDN	;HOLE TOO BIG
5350							
5351	014514'	016302	000024	LLPMUP:	MOV	POBJST(R3),R2	
5352	014520'	160002			SUB	R0,R2	
5353	014522'	016300	000024		MOV	POBJST(R3),R0	;POINT R0 AT THE TOP

5354	014526'	014240		LLPLP6:	MOV	-(R2),-(R0)		;MOVE EVERYTHING UP
5355	014530'	020001			CMP	R0,R1		;TO MAKE A HOLE
5356	014532'	001375			BNE	LLPLP6		;FOR THE NEW STATEMENT
5357	014534'	000411			BR	LLPFLI		;GO FILL IN THE HOLE
5358	014536'	111102		LLPMDN:	MOVB	(R1),R2		
5359	014540'	060201			ADD	R2,R1		;R1 POINTS AT NEXT
5360	014542'	010102			MOV	R1,R2		;OLD STATEMENT
5361	014544'	060002			ADD	R0,R2		;R2 POINTS WHERE IT SHD BE
5362	014546'	020263	000024	LLPLP7:	CMP	R2,POBJST(R3)		;ARE WE DONE ?
5363	014552'	001402			BEQ	LLPFLI		;YES - BRANCH
5364	014554'	012122			MOV	(R1)+,(R2)+		;MOVE EVERYTHING DOWN
5365	014556'	000773			BR	LLPLP7		;TO MAKE HOLE SMALLER
5366	014560'	016702	000252	LLPFLI:	MOV	LLPSNP,R2		;RESTORE NEW AND
5367	014564'	016701	000250		MOV	LLPSOP,R1		;OLD POINTERS
5368	014570'	111200			MOVB	(R2),R0		;GET LENGTH OF NEW ENTRY
5369	014572'	112221		LLPLP8:	MOVB	(R2)+,(R1)+		;MOVE NEW ENTRY
5370	014574'	005300			DEC	R0		;INTO PREPARED HOLE
5371	014576'	001375			BNE	LLPLP8		
5372	014600'	000677			BR	LLPGLB		;THEN GET ANOTHER LINE
5373	014602'	010267	000230	LLPINL:	MOV	R2,LLPSNP		;SAVE NEW POINTER
5374	014606'	010167	000226		MOV	R1,LLPSOP		;SAVE OLD POINTER
5375	014612'	010200			MOV	R2,R0		
5376	014614'	004567	012612		JSR	R5,LCKOBL		;GET OBJECT LENGTH
5377	014620'	060067	000220		ADD	R0,LPROLN		;UPDATE PROG LENGTH
5378	014624'	111200			MOVB	(R2),R0		;GET SOURCE LENGTH
5379	014626'	060067	000212		ADD	R0,LPROLN		;UPDATE PROG LENGTH
5380	014632'	060063	000024		ADD	R0,POBJST(R3)		;UPDATE OBJ START ADDR
5381	014636'	000726			BR	LLPMUP		;GO MOVE EVERYTHING UP
5382	014640'	016300	000024	LLPTEN:	MOV	POBJST(R3),R0		
5383	014644'	162700	000005		SUB	#5,R0		
5384	014650'	122710	000240		CMPB	#240,(R0)		;LAST STATEMENT AN END
5385	014654'	001440			BEQ	LLPFEX		;YES-LETS GE OUT OF HERE
5386	014656'	004567	171342		JSR	R5,CTYPE		;NO-TELL USER
5387	014662'	000402			.WORD	402		
5388	014664'	000230			.WORD	LNEMSG-		
5389	014666'	000644			BR	LLPGLB		;THEN GET ANOTHER LINE
5390	014670'	004567	171330	LLPWRN:	JSR	R5,CTYPE		;TYPE MEM LIMIT MSG
5391	014674'	000402			.WORD	402		
5392	014676'	000267			.WORD	LMLMSG-		
5393	014700'	016302	000024	LLPIEN:	MOV	POBJST(R3),R2		;NOW WE INSERT AN END
5394	014704'	062763	000006		ADD	#6,POBJST(R3)		;BUMP OBJECT START BY 6
5395	014712'	112722	000006		MOVB	#6,(R2)+		;STORE LENGTH OF 6
5396	014716'	112722	000240		MOVB	#240,(R2)+		;STORE FUNCT. CODE FOR END
5397	014722'	012722	114631		MOV	#114631,(R2)+		;STORE 9999 FOR LINE NBR
5398	014726'	005022			CLR	(R2)+		;STORE 0 FOR PHYSICAL ADDR.
5399	014730'	062767	000012		ADD	#10,LPROLN		;UP PROG LENGTH BY 10
5400	014736'	016302	000024		MOV	POBJST(R3),R2		;GET OBJECT START
5401	014742'	010267	164500		MOV	R2,CVMST		;PUT IN FREE MEM START
5402	014746'	160302			SUB	R3,R2		;SUB START OF P TABLE
5403	014750'	010263	000026		MOV	R2,PLNGTH(R3)		;UPDATE P LENGTH IN TBL
5404	014754'	000417			BR	LLPRFL		;EXIT
5405	014756'	016302	000024	LLPFEX:	MOV	POBJST(R3),R2		;GET OBJECT START
5406	014762'	010267	164460		MOV	R2,CVMST		;PUT IN FREE MEM START
5407	014766'	160302			SUB	R3,R2		;SUB START OF P TABLE
5408	014770'	010263	000026		MOV	R2,PLNGTH(R3)		;UPDATE P LENGTH IN TBL
5409	014774'	105767	000034		TSTB	LFETCH		;TEST FETCH FLAG

```

5410 015000' 001405          BEQ      LLPRFL          ;IF RESET, CONT EXIT
5411 015002' 004577 163040   JSR      RS, GET          ;GO VALIDATE CHECKSUM
5412 015006' 177020          .WORD    LLPTFF-        ;ERROR IF NO GOOD
5413 015010' 000000          .WORD    0
5414 015012' 000000          .WORD    0
5415 015014' 105067 001043   LLPRFL: CLRB   LENTER
5416 015020' 105067 001036   CLRB   LMODFY          ;CLEAR INTERNAL FLAGS
5417 015024' 105067 000004   CLRB   LFETCH
5418 015030' 000167 015230   COMPIL: JMP    LCPIL
5419 015034' 000000          LFETCH: .WORD  0          ;EXIT
5420 015036' 000000          LLPSNP: .WORD  0          ;SAVED NEW POINTER
5421 015040' 000000          LLPSOP: .WORD  0          ;SAVED OLD POINTER
5422 015042' 000000          LOBJLN: .WORD  0
5423 015044' 000000          LPROLN: .WORD  0          ;TOTAL PROGRAM LENGTH
5424
5425 015046' 054523 052116 054101 LFTMSG: .ASCIZ /SYNTAX ERROR ON SAVED PROG-END FORCED/
      015054' 042440 051122 051117
      015062' 047440 020116 040523
      015070' 042526 020104 051120
      015076' 043517 042455 042116
      015104' 043040 051117 042503
      015112' 000104
5426 015114' 040514 052123 051440 LNEMSG: .ASCIZ /LAST STMT NOT AN END/
      015122' 046524 052116 047040
      015130' 052117 040440 020116
      015136' 047105 000104
5427 015142' 042011 046105 052105 LDLMSG: .ASCIZ <011>/DELETED/
      015150' 042105 000
5428 015153' 116 052117 043040 LNFMSG: .ASCIZ /NOT FOUND/
      015160' 052517 042116 000
5429 015165' 115 046505 046040 LMLMSG: .ASCIZ /MEM LIMIT EXCEEDED-END FORCED/
      015172' 046511 052111 042440
      015200' 041530 042505 042504
      015206' 026504 047105 020104
      015214' 047506 041522 042105
      015222' 000
5430 015224'          .EVEN

```

.SBTTL USER PROGRAM CONSOLE INTERFACE

THIS ROUTINE CONTROLS THE CONSOLE TERMINAL
 INPUT AND OUTPUT TO PROVIDE THE NEXT
 SEQUENTIAL LINE NUMBER, OR ACCEPT THE
 ONE SUPPLIED BY THE USER. ITS MAIN FUNCTION
 IS TO INPUT ONE LINE OF THE USERS PROGRAM.
 ONCE ENTERED, IT DOES NOT EXIT UNTIL
 A CARRIAGE RETURN IS RECEIVED FROM THE KEYBOARD.

CALLING SEQUENCE- JSR R5,UPCI

5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487

015224'			
015224'	012700	000110	
015230'	010701		
015232'	062701	017030	
015236'	112721	000015	
015242'	005300		
015244'	001374		
015246'	004567	170752	
015252'	000420		
015254'	000604		
015256'	000001		
015260'	136767	000576	000606
015266'	001130		
015270'	004577	162540	
015274'	000604		
015276'	000001		
015300'	126767	000574	017122
015306'	001006		
015310'	004567	170710	
015314'	000420		
015316'	017112		
015320'	000002		
015322'	000751		
015324'	126767	000550	000532
015332'	001415		
015334'	004567	170664	
015340'	000420		
015342'	000536		
015344'	000001		
015346'	116767	000526	016706
015354'	004577	162450	
015360'	016703		
015362'	000107		
015364'	000475		
015366'	105767	000510	
015372'	001043		
015374'	105767	003674	

```

UPCI:      MOV      #72, R0
           MOV      PC, R1
           ADD      #LINBUF-, R1
LCLBUF:    MOVB     #015, (R1)+ ; CLEAR INPUT AREA
           DEC      R0
           BNE     LCLBUF
LREADY:    JSR     R5, CTYPE ; TYPE A GTR THAN SIGN
           .WORD   420
           .WORD   LGTRTN-.
           .WORD   1
           BITB    LMODFY, LON ; TEST MODIFY FLAG
           BNE     LRD72 ; IF SET, GO READ 72 CHARS
           JSR     R5, JCTRDNE ; READ FROM TTY - NO ECHO
           .WORD   LCBUF-.
           .WORD   1
           CMPB   LCBUF, LCRLF ; TEST FOR A CARRIAGE RETURN
           BNE     LSPTST ; GO TEST FOR A SPACE
           JSR     R5, CTYPE ; TYPE CARRIAGE RETURN-LF
           .WORD   420
           .WORD   LCRLF-.
           .WORD   2
           BR     LREADY ; GO GET NEW INPUT
LSPTST:    CMPB   LCBUF, LSPACE ; TEST IF IT WAS A SPACE
           BEQ     LINCLN ; IF IT WAS, INCREMENT LINE NBR
           JSR     R5, CTYPE ; OTHERWISE ECHO THE CHARACTER
           .WORD   420
           .WORD   LCBUF-.
           .WORD   1
           MOVB   LCBUF, LINBUF ; AND MOVE IT TO THE LINE BUFFER
           JSR     R5, JCTRD ; READ FROM TTY
           .WORD   LINBUF+1-.
           .WORD   71
           BR     LUPDLN ; GO UPDATE THE LINE NUMBER
LINCLN:    TSTB   FSTTME ; IF FIRST TIME THRU
           BNE     LINCNDN ; DONT INCR LINE NBR
           TSTB   LBDLIN ; TEST BAD LIN NBR FLAG
    
```

5488	015400'	001003		BNE	LDOINC		; IF SET, DO INCREMENT
5489	015402'	105767	005643	TSTB	LERROR		; IF ERROR SET
5490	015406'	001035		BNE	LINCDN		; DONT INCR LINE NBR
5491	015410'	005067	003660	LDOINC:	CLR	LBDLIN	; CLR BAD LIN NBR FLAG
5492	015414'	010700		MOV	PC,R0		; GET ADDRESS OF
5493	015416'	062700	000453	ADD	#LINNBR+3-.,R0		; LEAST SIGNIF. ASCII DIGIT
5494	015422'	116710	000450	MOVB	LASZER,(R0)		
5495	015426'	005300		DEC	R0		
5496	015430'	121067	000443	CMPB	(R0),LASNIN		; IS IT A NINE ?
5497	015434'	001021		BNE	LINC0G		; NO-INCREMENT DIGIT AND EXIT
5498	015436'	116710	000434	MOVB	LASZER,(R0)		; YES, REPLACE IT WITH A ZERO
5499	015442'	005300		DEC	R0		; MOVE POINTER TO NEXT DIGIT
5500	015444'	121067	000427	CMPB	(R0),LASNIN		; IS IT A NINE ?
5501	015450'	001013		BNE	LINC0G		; NO-INCREMENT DIGIT AND EXIT
5502	015452'	116710	000420	MOVB	LASZER,(R0)		; YES, REPLACE IT WITH A ZERO
5503	015456'	005300		DEC	R0		; MOVE POINTER TO NEXT DIGIT
5504	015460'	121067	000413	CMPB	(R0),LASNIN		; IS IT A NINE ?
5505	015464'	001005		BNE	LINC0G		; NO-INCREMENT DIGIT AND EXIT
5506	015466'	116720	000405	MOVB	LASNIN,(R0)+		; YES-MAKE NEXT LEAST A NINE
5507	015472'	116710	000401	MOVB	LASNIN,(R0)		; AND LSD A NINE
5508	015476'	000401		BR	LINCDN		; EXIT
5509	015500'	105210		LINC0G:	INCB	(R0)	; INCREMENT DIGIT
5510	015502'	004567	170516	LINCDN:	JSR	R5,CTYPE	; TYPE OUT LINE NUMBER
5511	015506'	000420		.WORD	420		
5512	015510'	000356		.WORD	LINNBR-		
5513	015512'	000005		.WORD	5		
5514	015514'	016767	000346	016540	MOV	LINNBR,LINBUF	; MOVE COMPUTED LINE NUMBER
5515	015522'	016767	000342	016534	MOV	LINNBR+2,LINBUF+2	
5516	015530'	116767	000336	016530	MOVB	LINNBR+4,LINBUF+4	; TO LINE BUFFER
5517	015536'	004577	162266	JSR	R5,ACTRD		; READ FROM TTY
5518	015542'	016525		.WORD	LINBUF+5-		
5519	015544'	000103		.WORD	67		
5520	015546'	000404		BR	LUPDLN		; GO UPDATE LINE NUMBER
5521	015550'	004577	162254	LRD72:	JSR	R5,ACTRD	; READ FROM TTY
5522	015554'	016506		.WORD	LINBUF-		
5523	015556'	000110		.WORD	72		
5524	015560'	010701		LUPDLN:	MOV	PC,R1	
5525	015562'	062701	016500	ADD	#LINBUF-.,R1		; SET POINTER AT LINE START
5526	015566'	121127	000012	10\$:	CMPB	(R1),#12	; LINE FEED?
5527	015572'	001404		BEQ	20\$; YES - CHANGE TO CR
5528	015574'	122127	000015	CMPB	(R1)+,#15		; CR?
5529	015600'	001403		BEQ	30\$; YES - END OF LINE FOUND
5530	015602'	000771		BR	10\$; NO - CHECK NEXT BYTE
5531	015604'	112711	000015	20\$:	MOVB	#15,(R1)	; STORE CR
5532	015610'	010701		30\$:	MOV	PC,R1	
5533	015612'	062701	016450	ADD	#LINBUF-.,R1		; SET POINTER AT LINE START
5534	015616'	105067	005427	CLRB	LERROR		; RESET ERROR FLAG
5535	015622'	105067	000254	CLRB	FSTTME		; CLEAR FIRST TIME FLAG
5536	015626'	012700	000003	MOV	#3,R0		; SET DIGIT COUNTER TO 3
5537	015632'	121167	000226	LUPDL1:	CMPB	(R1),LSPACE	; IF NOT BLANK
5538	015636'	001011		BNE	LUPDL2		; CHECK FOR NUMERIC
5539	015640'	121167	000226	CMPB	(R1),LTAB		; IF IT WAS A TAB
5540	015644'	001401		BEQ	LCKCR		; CHECK FOR CARRIAGE RETN
5541	015646'	000405		BR	LUPDL2		; OTHERWISE CHECK FOR NUMERIC
5542	015650'	005201		LCKCR:	INC	R1	; ADVANCE CHAR POINTER
5543	015652'	122711	000015	CMPB	#15,(R1)		; IF NOT A CARRIAGE RETURN

5544	015656'	001365		BNE	LUPDL1				;CHECK NEXT CHARACTER
5545	015660'	000476		BR	LUPCIX				;OTHERWISE EXIT
5546	015662'	121167	000210	LUPDL2:	CMPB	(R1),LASZER			;IF LESS THAN ASCII ZERO
5547	015666'	103473		BLO	LUPCIX				;EXIT
5548	015670'	121167	000203	CMPB	(R1),LASNIN				;IF MORE THAN ASCII NINE
5549	015674'	101070		BHI	LUPCIX				;EXIT
5550	015676'	005201		INC	R1				;ADVANCE CHAR POINTER
5551	015700'	121167	000160	CMPB	(R1),LSPACE				;IF SPACE NEXT
5552	015704'	001410		BEQ	LJSTFY				;START TO JUSTIFY
5553	015706'	121167	000160	CMPB	(R1),LTAB				;IF TAB NEXT
5554	015712'	001405		BEQ	LJSTFY				;START TO JUSTIFY
5555	015714'	121127	000015	CMPB	(R1),#15				;IF CARRIAGE RETURN
5556	015720'	001402		BEQ	LJSTFY				;LGTRTN TO JUSTIFY
5557	015722'	005300		DEC	RO				;DECREMENT DIGIT COUNT
5558	015724'	000756		BR	LUPDL2				;CHECK NEXT CHARACTER
5559	015726'	010701		LJSTFY:	MOV	PC,R1			;SET R1 TO ADDRESS
5560	015730'	062701	016332	ADD	#LINBUF-. ,R1				;OF LINE BUFFER
5561	015734'	020027	000000	CMP	RO,#0				;CHECK DIGIT COUNT
5562	015740'	001440		BEQ	LSAVLN				;IF ZERO, DONT JUSTIFY
5563	015742'	002417		BLT	LLJUST				;IF NEGATIVE, LEFT JUSTIFY
5564	015744'	062701	000110	ADD	#72, R1				;IF POSITIVE, RIGHT JUSTIFY
5565	015750'	010102		MOV	R1,R2				;SET R1 & R2 TO LINBUF + 72
5566	015752'	160001		SUB	RO,R1				;SEPARATE BY AMMOUNT TO JUSTIFY
5567	015754'	010700		MOV	PC,RO				;SET RO TO ADDRESS
5568	015756'	062700	016304	ADD	#LINBUF-. ,RO				;OF LINE BUFFER
5569	015762'	114142		RJUST1:	MOVB	-(R1),-(R2)			;START SHIFTING LINE
5570	015764'	020100		CMP	R1,RO				;SHIFT COMPLETE ?
5571	015766'	001375		BNE	RJUST1				;NO, SHIFT SOME MORE
5572	015770'	116742	000102	LRJST2:	MOVB	LASZER,-(R2)			;FILL IN LEADING ZEROES
5573	015774'	020200		CMP	R2,RO				
5574	015776'	001374		BNE	LRJST2				
5575	016000'	000420		BR	LSAVLN				;GO SAVE JUSTIFIED LINE NBR
5576	016002'	010102		LLJUST:	MOV	R1,R2			;SET R2 TO LINBUF ADDRESS
5577	016004'	160002		SUB	RO,R2				;SEPARATE BY AMMOUNT TO JUSTIFY
5578	016006'	010100		MOV	R1,RO				
5579	016010'	116720	000050	LSPFL:	MOVB	LSPACE,(RO)+			;CLEAR LEADING ZEROES
5580	016014'	020002		CMP	RO,R2				;ALL CLEAR ?
5581	016016'	001374		BNE	LSPFL				;IF NOT, CLEAR SOME MORE
5582	016020'	111221		LLJST1:	MOVB	(R2),(R1)+			;KEEP SHIFTING
5583	016022'	116722	000036	MOVB	LSPACE,(R2)+				;AND INSERTING
5584	016026'	121267	000032	CMPB	(R2),LSPACE				;IF LAST CHAR NOT A LSPACE
5585	016032'	001403		BEQ	LSAVLN				;SHIFT SOME MORE
5586	016034'	121267	000032	CMPB	(R2),LTAB				;IF LAST CHAR NOT A LTAB
5587	016040'	001367		BNE	LLJST1				;SHIFT SOME MORE
5588	016042'	016767	016214	LSAVLN:	MOV	LINBUF,LINNR			;SAVE CURRENT
5589	016050'	016767	016210	000016	000012	MOV	LINBUF+2,LINNR+2		;LINNR
5590	016056'	000205		LUPCIX:	RTS	R5			;RETURN TO CALLING PROGRAM
5591									
5592	016060'	000076		LGTRTN:	.ASCIZ	/>/			
5593	016062'	000		LMOFY:	.BYTE	0			
5594	016063'	000		LENTER:	.BYTE	0			
5595	016064'	000040		LSPACE:	.ASCIZ	/ /			
5596	016066'	030060	030060	LINNR:	.ASCII	/0000/			;LINE NUMBER
5597	016072'	000011		LTAB:	.ASCIZ	<11>			;HORIZLONTAL LTAB
5598	016074'	001		LON:	.BYTE	1			
5599	016075'	000		LOFF:	.BYTE	0			

5600	016076'	060	LASZER:	.ASCII	/0/	;ASCII ZERO
5601	015077'	071	LASNIN:	.ASCII	/9/	;ASCII NINE
5602	015100'	000	LCBUF:	.BYTE	0	;CHARACTER BUFFER
5603	015101'	000		.BYTE	0	
5604	015102'	000000	FSTTME:	.WORD	0	;FIRST TIME FLAG

.SBTTL USER PROGRAM LIBRARY INTERFACE

THIS ROUTINE READS PROGRAM SOURCE STATEMENTS
IN BLOCKS FROM THE LIBRARY DEVICE
AND TRANSFERS ONLY ONE STATEMENT TO THE
LINE BUFFER EACH TIME IT IS CALLED.

CALLING SEQUENCE

JSR R5,UPLIO FIRST CALL
.WORD ERATN-.

JSR R5,UPLI SUBSEQUENT CALLS
.WORD ERATN-.

5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626 016104' 004577 161736
5627 016110' 000130
5628 016112' 000000
5629 016114' 000000
5630 016116' 016767 177770 000160
5631 016124' 016767 177764 000150
5632 016132' 010702
5633 016134' 062702 016126
5634 016140' 016704 000140
5635 016144' 005767 000132
5636 016150' 001410
5637 016152' 112422
5638 016154' 005367 000122
5639 016160' 126227 177777 000012
5640 016166' 001430
5641 016170' 000765
5642 016172' 016704 161612
5643 016176' 005714
5644 016200' 001413
5645 016202' 004577 161640
5646 016206' 000032
5647 016210' 000000
5648 016212' 000000
5649 016214' 016704 177770
5650 016220' 016767 177766 000054
5651 016226' 000746
5652 016230' 004567 167770
5653 016234' 000412
5654 016236' 000022
5655 016240' 105267 005005
5656 016244' 061505
5657 016246' 000205
5658 016250' 010467 000030
5659 016254' 005725
5660 016256' 000205
5661

UPLIO: JSR R5,@GET ;GET FIRST BLOCK
.WORD LLIERT-.
LLIEW1: .WORD 0 ;ADDRESS SUPPLIED BY EXEC
LLIEW2: .WORD 0 ;BYTE COUNT ALSO SUPPLIED
MOV LLIEM1,LLIBPT ;UPDATE BLOCK POINTER
MOV LLIEM2,LLIBCT ;UPDATE DATA LENGTH
UPLI: MOV PC,R2
ADD #LINBUF-.,R2 ;SET LINE POINTER
MOV LLIEM3,R4 ;GET BLOCK POINTER
LLITBC: TST LLIEM3 ;BYTES LEFT IN BLOCK?
BEQ LLITMB ;NO - TEST IF MORE BLOCKS
MOVB (R4)+,(R2)+ ;MOVE CHAR TO LINE BUF
DEC LLIEM3 ;DECREMENT DATA LENGTH
CMPB -1(R2),#12 ;WAS IT A LINE FD?
BEQ LLIEXT ;IF YES EXIT
BR LLITBC ;CHECK IF END OF BLOCK
LLITMB: MOV BUF,R4
TST (R4) ;MORE BLOCKS LEFT?
BEQ LLIERR ;ERROR IF NOT
JSR R5,@GET ;GET NEXT BLOCK
.WORD LLIERT-.
LLIEW3: .WORD 0 ;ADDRESS SUPPLIED BY EXEC
LLIEW4: .WORD 0 ;BYTE COUNT ALSO SUPPLIED
MOV LLIEM3,R4 ;POINT R4 AT DATA
MOV LLIEM4,LLIBCT ;UPDATE DATA LENGTH
BR LLITBC
LLIERR: JSR R5,CTYPE ;TYPE FETCH ERROR MSG
.WORD 412
LLIERT: .WORD LLIEM1-.
INCB LERROR ;SET ERROR FLAG
ADD (R5),R5 ;CALC ERROR RETURN
RTS R5 ;AND TAKE IT
LLIEXT: MOV R4,LLIBPT ;SAVE BLOCK BUF PNTR
TST (R5)+ ;BYPASS ERR RETURN ADR
RTS R5 ;RETURN

5662 016260' 046506 020124 047117 LLIEM1: .ASCIZ /FMT ON SAVED FILE/
016266' 051440 053101 042105
016274' 043040 046111 000105

5663
5664 016302' 000000 LLIBCT: .EVEN
5665 016304' 000000 LLIBPT: .WORD 0 :DATA LENGTH
0 :BLOCK CHAR POINTER

.SBTTL PACK SUBROUTINE

5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722

016306' 105067 004737
016312' 005067 004730
016316' 010346
016320' 010446
016322' 012700 000110
016326' 010703
016330' 062703 004734
016334' 105023
016336' 005300
016340' 001375
016342' 162703 000110
016346' 010702
016350' 062702 015711
016354' 010267 010244
016360' 004567 010022
016364' 010702
016366' 062702 004664
016372' 012704 000003
016376' 160400
016400' 122220
016402' 001025
016404' 005304
016406' 001374
016410' 005267 004632
016414' 000416
016416' 105267 004626
016422' 000413
016424' 105267 004617
016430' 000410
016432' 004567 167566
016436' 000412
016440' 170357
016442' 105267 002626
016446' 105267 004577
016452' 000167 000756
016456' 005304
016460' 060400
016462' 004567 023662
016466' !77744

```

*****
THIS SUBROUTINE EXAMINES ONE LINE OF A
USERS PROGRAM ONE WORD AT AT TIME AND
CONVERTS IT TO AN INTERNAL PACKED CODE.

CALLING SEQUENCE      JSR      RS,PACK

ENTRY  USERS STATEMENT IS IN LINBUF

EXIT   PACKED STATEMENT IS IN LPKBUF
*****

PACK:  CLRB      LERROR      ;RESET ERROR FLAG
        CLR      LDONE       ;RESET DONE AND DELETE FLAGS
        MOV      R3,-(SP)
        MOV      R4,-(SP)      ;SAVE R3 AND R4
        MOV      #72,R0
        MOV      PC,R3
        ADD      #LPKBUF-.,R3
LPKCBF: CLRB      (R3)+      ;CLEAR PACK BUFFER
        DEC      R0
        BNE     LPKCBF
        SUB      #72,R3      ;SET OUTPUT POINTER
        MOV      PC,R2
        ADD      #LINBUF-1-.,R2 ;SET INPUT POINTER
        MOV      R2,LSCNPT    ;AND SAVE IT
        JSR     RS,SCAN      ;SCAN FIRST WORD
        MOV      PC,R2
        ADD      #LASDON-.,R2
        MOV      #3,R4
        SUB      R4,R0
LPKLP1: CMPB     (R2)+,(R0)+  ;WAS FIRST WORD 'DONE'
        BNE     LPKCLN      ;NO-GO CONVERT LINE NBR
        DEC      R4
        BNE     LPKLP1
LPKSDN: INC      LDONE       ;YES-SET DONE FLAG
        BR      LGOEXT      ;THEN FIND EXIT
LPKSIG: INCB     LIGNOR      ;OTHERWISE SET IGNOR
        BR      LGOEXT      ;AND EXIT
LPKSDL: INCB     LDELET      ;SET DELETE FLAG
        BR      LGOEXT      ;AND EXIT
LPKMLN: JSR     RS,CTYPE
        .WORD   412
        .WORD   CERM16-.
        INCB     LBOLIN      ;SET BAD LINE NBR FLAG
        INCB     LERROR      ;SET ERROR FLAG
LGOEXT: JMP      LPKEXT
LPKCLN: DEC      R4
        ADD      R4,R0
        JSR     RS,ASDECO    ;CONVERT LINE NUMBER
        .WORD   LPKNLN-.    ;TO PACKED DECIMAL

```

DTMGAC.P11

PACK SUBROUTINE

5723	016470'	116063	000001	000002		MOVB	1(R0),2(R3)	
5724	016476'	116063	000002	000003		MOVB	2(R0),3(R3)	; STORE IN PACK BUFFER
5725	016504'	004567	007576			JSR	R5,SCAN	; GET NEXT WORD
5726	016510'	122710	000015			CMPB	#15,(R0)	; IF CARRIAGE RETURN
5727	016514'	001740				BEQ	LPK5IG	; GO SET IGNORE FLAG
5728	016516'	010702			LPKILU:	MOV	PC,R2	
5729	016520'	062702	003230			ADD	#LPKTBE--,R2	
5730	016524'	010212				MOV	R2,(R2)	
5731	016526'	010702				MOV	PC,R2	
5732	016530'	062702	002560			ADD	#LPKTBL+6--,R2	; SET POINTER TO PACK TABLE
5733	016534'	005267	002536			INC	LPKIFL	; SET INTERNAL TBL FLAG
5734	016540'	005200			LPKL25:	INC	R0	
5735	016542'	026702	003202		LPKCTE:	CMP	LPKTBE,R2	; ARE WE AT TBL END?
5736	016546'	001034				BNE	LPKSCC	; NO-SAVE CHAR COUNT
5737	016550'	005767	002522			TST	LPKIFL	; IF INTERNAL FLAG CLR
5738	016554'	001401				BEQ	LPKEB1	; TAKE ERROR BRANCH
5739	016556'	000402				BR	LPKSEX	; SET UP FOR EXTERNAL TBL
5740	016560'	000167	000644		LPKEB1:	JMP	LPKERR	; ELSE TAKE ERROR EXIT
5741	016564'	005067	002506		LPKSEX:	CLR	LPKIFL	; CLEAR INTERNAL TBL FLAG
5742	016570'	016702	002474			MOV	LPKPTP,R2	; DETERMINE END
5743	016574'	062702	000352			ADD	#PTLGTH+DVVTEP,R2	; OF
5744	016600'	061202				ADD	(R2),R2	; EXTERNAL PACK TABLE
5745	016602'	062702	000006			ADD	#6,R2	
5746	016606'	010267	003136			MOV	R2,LPKTBE	
5747	016612'	016702	002452			MOV	LPKPTP,R2	; POINT R2
5748	016616'	062702	000350			ADD	#PTLGTH+DVPTEP,R2	; AT START OF
5749	016622'	061202				ADD	(R2),R2	; EXT PACK TABLE
5750	016624'	026702	003120			CMP	LPKTBE,R2	; ARE WE AT TBL END?
5751	016630'	001753				BEQ	LPKEB1	; YES - ERROR
5752	016632'	062702	000006			ADD	#6,R2	; ADJUST R2
5753	016636'	000741				BR	LPKCTE	; CHECK IF START = END
5754	016640'	010104			LPKSCC:	MOV	R1,R4	; SAVE CHAR COUNT
5755	016642'	010267	004412			MOV	R2,LPKPSV	; AND TABLE POINTER
5756	016646'	124042			LPKLP3:	CMPB	-(R0),-(R2)	; COMPARE ONE CHAR
5757	016650'	001011				BNE	LPKNXI	; IF NOT=CHECK NEXT INST.
5758	016652'	005304				DEC	R4	; DECREMENT CHAR COUNT
5759	016654'	001374				BNE	LPKLP3	; LOOP IF MORE TO GO
5760	016656'	022701	000006			CMP	#6,R1	; IF WORD COUNT = 6
5761	016662'	001414				BEQ	LPKIFD	; INSTRUCTION FOUND
5762	016664'	122742	000040			CMPB	#40, -(R2)	; IF NO LEADING BL IN TBL
5763	016670'	001366				BNE	LPKLP3	; CHECK NEXT INST
5764	016672'	000410				BR	LPKIFD	; INSTRUCTION FOUND!
5765	016674'	062767	000010	004356	LPKNXI:	ADD	#8,LPKPSV	; BUMP POINTER
5766	016702'	016702	004352			MOV	LPKPSV,R2	; PUT IT IN R2
5767	016706'	016700	007712			MOV	LSCNPT,R0	
5768	016712'	000712				BR	LPKL25	; CHECK NEXT TABLE ENTRY
5769	016714'	016702	004340		LPKIFD:	MOV	LPKPSV,R2	
5770	016720'	121227	000257			CMPB	(R2),#257	; WAS IT DELETE ?
5771	016724'	001637				BEQ	LPKSDI	; YES - GO SET FLAG
5772	016726'	121227	000256			CMPB	(R2),#256	; WAS IT DONE ?
5773	016732'	001626				BEQ	LPKSDN	; YES-SET DONE FLAG
5774	016734'	111263	000001			MOVB	(R2),1(R3)	; STORE BASIC FUNCTION CODE
5775	016740'	105762	000001			TSTB	1(R2)	; CHECK FLAG BYTE
5776	016744'	001402				BEQ	LPKCIO	
5777	016746'	000167	001104			JMP	LPKSPH	; SET MEANS SPECIAL HANDLING
5778	016752'	121227	000317		LPKCIO:	CMPB	(R2),#317	

5779	016756'	101014		BHI	LPKCODE		;BRANCH IF I/O CMD
5780	016760'	121227	000120	CMPB	(R2), #120		
5781	016764'	103417		BLO	LPKSSP		;BRANCH IF NOT READ
5782	016766'	121227	000137	CMPB	(R2), #137		
5783	016772'	101406		BLOS	LPKCODE		;WRITE
5784	016774'	121227	000160	CMPB	(R2), #160		
5785	017000'	103411		BLO	LPKSSP		;OR BREAK
5786	017002'	121227	000167	CMPB	(R2), #167		
5787	017006'	101006		BHI	LPKSSP		
5788	017010'	111200		LPKCODE: MOV	(R2), R0		;FUNCTION CODE TO R0
5789	017012'	016702	002252	MOV	LPKPTP, R2		;P TBL PNTR TO R2
5790	017016'	004567	010612	JSR	R5, FENTRY		;CHECK IF FC SUPPORTED
5791	017022'	000352		.WORD	LPKDER-		
5792	017024'	016767	007574	007576 LPKSSP: MOV	LSCNPT, LSPTS		;SAVE SCAN POINTER
5793	017032'	016702	004222	MOV	LPKPSV, R2		;GET PACK POINTER
5794	017036'	121227	000237	CMPB	(R2), #237		;IF NOT MULTIPLE FORMAT
5795	017042'	101045		BHI	LPKSV		;DONT GO ADD WC TO FC
5796	017044'	005004		CLR	R4		;SET WORD COUNT=0
5797	017046'	004567	007334	LPKLP4: JSR	R5, SCAN		;SCAN FOR NEXT WORD
5798	017052'	122710	000015	CMPB	#15, (R0)		;WAS IT A CARRIAGE RET?
5799	017056'	001402		BEQ	LPKCBK		;YES-CHECK IF BREAK
5800	017060'	105204		INCB	R4		;NO-INCREMENT WORD COUNT
5801	017062'	000771		BR	LPKLP4		;SCAN NEXT WORD
5802	017064'	120427	000003	LPKCBK: CMPB	R4, #3		;WORD COUNT < 3 ?
5803	017070'	103430		BLO	LPKUFC		;YES - CONTINUE
5804	017072'	126327	000001	000160 CMPB	1(R3), #160		;IF NOT A BREAK
5805	017100'	103407		BLO	LPKCRW		;CHECK FOR RD OR WR
5806	017102'	126327	000001	000167 CMPB	1(R3), #167		;IF NOT RD, WR OR BK
5807	017110'	101020		BHI	LPKUFC		;GO UPDATE F C
5808	017112'	112704	000003	MOV	#3, R4		;JAM 3 INTO WD CNT
5809	017116'	000415		BR	LPKUFC		
5810	017120'	120427	000005	LPKCRW: CMPB	R4, #5		;WORD COUNT < 5
5811	017124'	103412		BLO	LPKUFC		;YES - CONTINUE
5812	017126'	126327	000001	000120 CMPB	1(R3), #120		;IF NOT A READ
5813	017134'	103406		BLO	LPKUFC		;CONTINUE
5814	017136'	126327	000001	000137 CMPB	1(R3), #137		;IF NOT A WRITE
5815	017144'	101002		BHI	LPKUFC		;CONTINUE
5816	017146'	112704	000005	MOV	#5, R4		;JAM 5 INTO WORD CNT
5817	017152'	150463	000001	LPKUFC: BISB	R4, 1(R3)		;ADD WORD COUNT TO F.C.
5818	017156'	010704		LPKSV: MOV	PC, R4		;SET POINTER TO
5819	017160'	062704	002572	ADD	#LAMPNTR-, R4		;MODEL STATEMENT VECTOR TEL
5820	017164'	121463	000001	LPKLP5: CMPB	(R4), 1(R3)		;CHECK F.C. IN TABLE
5821	017170'	001450		BEQ	LPKSP		;BRANCH IF FOUND
5822	017172'	122714	000377	CMPB	#377, (R4)		;TEST IF END INT TBL
5823	017176'	001403		BEQ	LPKSEV		;YES-SET EXT VECTOR
5824	017200'	062704	000004	ADD	#4, R4		;BUMP POINTER
5825	017204'	000767		BR	LPKLP5		;CHECK NEXT ENTRY
5826	017206'	016704	002056	LPKSEV: MOV	LPKPTP, R4		;DETERMINE END
5827	017212'	062704	000354	ADD	#PTLGTH+DVCTEP, R4		;OF
5828	017216'	061404		ADD	(R4), R4		;EXTERNAL VECTOR TBL
5829	017220'	010467	002054	MOV	R4, LASTBE		
5830	017224'	016704	002040	MOV	LPKPTP, R4		;POINT R4 AT START
5831	017230'	062704	000352	ADD	#PTLGTH+DVVTEP, R4		;OF
5832	017234'	061404		ADD	(R4), R4		;EXTERN. VECTOR TABLE
5833	017236'	020467	002036	LPKLP6: CMP	R4, LASTBE		;R4 AT TBL END?
5834	017242'	001472		BEQ	LPKERR		;YES - ERROR

5835	017244'	121463	000001		CMPB	(R4), 1(R3)	;CHECK F.C. IN TABLE
5836	017250'	001403			BEQ	LPKSEP	;BRANCH IF FOUND
5837	017252'	062704	000004		ADD	#4, R4	;BUMP POINTER
5838	017256'	000767			BR	LPKLP6	;CHECK NEXT ENTRY
5839	017260'	016767	007344	007336	LPKSEP: MOV	LSPTSV, LSCNPT	;RESTORE SCAN POINTER
5840	017266'	062704	000002		ADD	#2, R4	
5841	017272'	062703	000006		ADD	#6, R3	;ADJUST PACK POINTER
5842	017276'	016702	001766		MOV	LPKPTP, R2	
5843	017302'	062702	000242		ADD	#PTLGTH, R2	
5844	017306'	061402			ADD	(R4), R2	;SET EXTERNAL MDL PTR
5845	017310'	000411			BR	LBASE1	
5846	017312'	016767	007312	007304	LPKSMP: MOV	LSPTSV, LSCNPT	;RESTORE SCAN POINTER
5847	017320'	062704	000002		ADD	#2, R4	
5848	017324'	062703	000006		ADD	#6, R3	;ADJUST PACK POINTER
5849	017330'	011402			MOV	(R4), R2	
5850	017332'	060702			ADD	PC, R2	;SET MODEL POINTER
5851	017334'	010267	003722		LBASE1: MOV	R2, LPKMPS	
5852	017340'	122712	000377		LPKCNB: CMPB	#377, (R2)	
5853	017344'	001503			BEQ	LPKVAR	;BRANCH FOR VARIABLE
5854	017346'	122712	000376		CMPB	#376, (R2)	
5855	017352'	001440			BEQ	LPKLIN	;BRANCH FOR LINE NBR
5856	017354'	122712	000373		CMPB	#373, (R2)	;IF END OF PRINT STATEMENT
5857	017360'	001425			BEQ	LPKEXT	;EXIT
5858	017362'	122712	000375		CMPB	#375, (R2)	
5859	017366'	001005			BNE	LPKCEM	
5860	017370'	000167	000546		JMP	LPKBIT	;BRANCH FOR BIT NBR
5861	017374'	105267	003651		LPKDER: INCB	LERROR	;SET ERROR FLAG
5862	017400'	000415			BR	LPKEXT	;GO TO EXIT
5863	017402'	105712			LPKCEM: TSTB	(R2)	;END OF MODEL ?
5864	017404'	001404			BEQ	LPKCUE	;YES-CHECK USER ENTRY
5865	017406'	005202			INC	R2	;BUMP MODEL POINTER
5866	017410'	010267	003646		MOV	R2, LPKMPS	;AND SAVE IT
5867	017414'	000747			BR	LBASE1	;CHECK NEXT BYTE OF MDL
5868	017416'	004567	006764		LPKCUE: JSR	R5, SCAN	;END OF USER ENTRY?
5869	017422'	122710	000015		CMPB	#15, (R0)	
5870	017426'	001402			BEQ	LPKEXT	;YES-EXIT
5871	017430'	004567	004664		LPKERR: JSR	R5, SYNTAX	;NO-REPORT ERROR
5872	017434'	010702			LPKEXT: MOV	PC, R2	
5873	017436'	062702	003626		ADD	#LPKBUF--, R2	
5874	017442'	160203			SUB	R2, R3	;CALCULATE PACKED LENGTH
5875	017444'	110312			LPKSTL: MOV	R3, (R2)	;STORE IN FIRST BYTE
5876	017446'	012604			MOV	(SP)+, R4	
5877	017450'	012603			MOV	(SP)+, R3	;RESTORE STACK
5878	017452'	000205			RTS	R5	;RETURN TO CALLING PGM
5879	017454'	004567	006726		LPKLIN: JSR	R5, SCAN	;FIND NEXT USER WORD
5880	017460'	122710	000015		CMPB	#15, (R0)	;WAS IT CARRIAGE RET
5881	017464'	001761			BEQ	LPKERR	;YES MEANS ERROR
5882	017466'	112723	000114		MOV	#'L, (R3)+	;STORE L FOR LINE NBR
5883	017472'	004567	022652		JSR	R5, ASDECO	;CONVERT TO
5884	017476'	177732			.WORD	LPKERR-	;PACKED DECIMAL
5885	017500'	111023			MOV	(R0), (R3)+	
5886	017502'	116023	000001		MOV	1(R0), (R3)+	;STORE IN PACKED CODE
5887	017506'	116023	000002		MOV	2(R0), (R3)+	
5888	017512'	105710			TSTB	(R0)	;ERROR IF LINE NBR
5889	017514'	001345			BNE	LPKERR	;GREATER THAN 4 DIGITS
5890	017516'	112763	000377	177775	MOV	#377, -3(R3)	;REDUCE TO 4 DIGITS

5891	017524'	005267	003532		_LPKAMP:	INC	LPKMPS		;ADVANCE MODEL POINTER
5892	017530'	004567	006652			JSR	R5,SCAN		;SCAN CONNECTING WORD
5893	017534'	016702	003522			MOV	LPKMPS,R2		;RESTORE MODEL POINTER
5894	017540'	122710	000015			CMPB	#15,(R0)		;WAS IT A CARRIAGE RET?
5895	017544'	001273				BNE	LBASE!		;NO-CHECK NEXT BYTE
5896	017546'	105712				TSTB	(R2)		;END OF MODEL STATEMENT?
5897	017550'	001327				BNE	LPKERR		;NO-ERROR
5898	017552'	000730				BR	LPKEXT		;YES-EXIT
5899	017554'	004567	006626		LPKVAR:	JSR	R5,SCAN		;SCAN VARIABLE
5900	017560'	122710	000015			CMPB	#15,(R0)		;WAS IT A CARRIAGE RET?
5901	017564'	001721				BEQ	LPKERR		;YES-ERROR
5902	017566'	005301				DEC	R1		
5903	017570'	001425				BEQ	LPKCOCT		
5904	017572'	160100				SUB	R1,R0		
5905	017574'	126067	000001	176274		CMPB	1(R0),LASZER		;SECOND DIGIT NON NUMERIC?
5906	017602'	103457				BLO	LPKTAG		;YES-PACK TAG
5907	017604'	126067	000001	176265		CMPB	1(R0),LASNIN		;SECOND DIGIT NON NUMERIC?
5908	017612'	101053				BHI	LPKTAG		;YES-PACK TAG
5909	017614'	122710	000104			CMPB	#'D,(R0)		;FIRST CHAR A 'D'
5910	017620'	001431				BEQ	LPKDEC		;YES-PACK DECIMAL
5911	017622'	122710	000117			CMPB	#'0,(R0)		;FIRST CHAR AN '0'
5912	017626'	001406				BEQ	LPKCOCT		;YES-PACK OCTAL
5913									
5914	017630'	121067	176242			CMPB	(R0),LASZER		;FIRST DIGIT NON NUMERIC?
5915	017634'	103442				BLO	LPKTAG		;YES-PACK TAG
5916	017636'	121067	176235			CMPB	(R0),LASNIN		;FIRST DIGIT NON NUMERIC?
5917	017642'	101037				BHI	LPKTAG		;YES-PACK TAG
5918	017644'	112723	000040		LPKCOCT:	MOVB	#40,(R3)+		;STORE A BLANK
5919	017650'	060100				ADD	R1,R0		;POINTER TO RIGHT DIGIT
5920	017652'	005201				INC	R1		;RESTORE CHAR COUNT
5921	017654'	004567	022462			JSR	R5,ASDEC1		;CONVERT TO PACKED DECIMAL
5922	017660'	177550				.WORD	LPKERR-		;REPRESENTATION OF OCT
5923	017662'	111023				MOVB	(R0),(R3)+		
5924	017664'	116023	000001			MOVB	1(R0),(R3)+		;STORE PACKED DECIMAL (OCTAL)
5925	017670'	116023	000002			MOVB	2(R0),(R3)+		
5926	017674'	004567	022232			JSR	R5,OCTBIN		;CONVERT TO BINARY
5927	017700'	177530				.WORD	LPKERR-		;JUST TO CHECK FOR ERROR
5928	017702'	000710				BR	LPKAMP		;IF OK ADVANCE MODEL PTR.
5929	017704'	112723	000104		LPKDEC:	MOVB	#'D,(R3)+		;STORE A 'D'
5930	017710'	060100				ADD	R1,R0		;POINTER TO RIGHT DIGIT
5931	017712'	004567	022424			JSR	R5,ASDEC1		;CONVERT TO PACKED DECIMAL
5932	017716'	177512				.WORD	LPKERR-		
5933	017720'	111023				MOVB	(R0),(R3)+		
5934	017722'	116023	000001			MOVB	1(R0),(R3)+		;STORE PACKED DECIMAL
5935	017726'	116023	000002			MOVB	2(R0),(R3)+		
5936	017732'	004567	022160			JSR	R5,DECBIN		;CONVERT TO BINARY
5937	017736'	177472				.WORD	LPKERR-		;JUST TO CHECK FOR ERROR
5938	017740'	000671				BR	LPKAMP		;IF OK ADVANCE MODEL PTR.
5939	017742'	005201			LPKTAG:	INC	R1		
5940	017744'	020127	000004			CMP	R1,#4		;CHECK IF TAG LENGTH<=4
5941	017750'	101401				BLOS	LPKCKT		
5942	017752'	000626			JLPKER:	BR	LPKERR		;ERROR IF IT ISNT
5943	017754'	010367	001312		LPKCKT:	MOV	R3,LPKPKP		
5944	017760'	016703	001304			MOV	LPKPTP,R3		
5945	017764'	004567	006642			JSR	R5,TAGLUP		;CHECK IF TAG VALID
5946	017770'	177440				.WORD	LPKERR-		

6003	020224'	177204				.WORD	LPKERR-	
6004	020226'	010023				MOV	R0,(R3)+	;STORE IT IN PACKED CODE
6005	020230'	016767	006370	006372	LPKSR1:	MOV	LSCNPT,LSPTSV	;SAVE SCAN POINTER
6006	020236'	004567	006144			JSR	R5,SCAN	;SCAN NEXT WORD
6007	020242'	122710	000015			CMPB	#15,(R0)	;CARRIAGE RET ?
6008	020246'	001005				BNE	LPKSR3	;NO-KEEP SEARCHING
6009	020250'	016767	006354	006346	LPKSR2:	MOV	LSPTSV,LSCNPT	;RESTORE SCAN POINTER
6010	020256'	000167	177242			JMP	LPKAMP	;GO ADVANCE MODEL POINTER
6011	020262'	126027	177777	000124	LPKSR3:	CMPB	-1(R0),#T	
6012	020270'	001357				BNE	LPKSR1	
6013	020272'	121027	000117			CMPB	(R0),#0	;LAST 2 LETTERS 'TO' ?
6014	020276'	001354				BNE	LPKSR1	;NO-KEEP SEARCHING
6015	020300'	000763				BR	LPKSR2	;YES-GO PROCESS BRANCH
6016	020302'	004567	006100		LPKIF:	JSR	R5,SCAN	
6017	020306'	004567	006074			JSR	R5,SCAN	;SCAN AHEAD
6018	020312'	005301				DEC	R1	
6019	020314'	160100				SUB	R1,R0	
6020	020316'	122710	000102			CMPB	#B,(R0)	;LAST WORD 'BIT'
6021	020322'	001475				BEQ	LPKBT1	;YES-ANALYZE IF BIT INSTR.
6022	020324'	122710	000101			CMPB	#A,(R0)	;LAST WORD 'AT'
6023	020330'	001010				BNE	LPKCR1	;YES CHECK RELATIONSHIP
6024	020332'	062713	004000			ADD	#4000,(R3)	;ADD 8 TO FUNCTION CODE
6025	020336'	004567	006044			JSR	R5,SCAN	
6026	020342'	004567	006040			JSR	R5,SCAN	;SCAN AHEAD
6027	020346'	005301				DEC	R1	
6028	020350'	160100				SUB	R1,R0	
6029	020352'	020127	000001		LPKCR1:	CMP	R1,#1	;CHECK WORD COUNT
6030	020356'	001422				BEQ	LPKWD2	;BRANCH IF TWO
6031	020360'	103401				BLO	LPKWD1	;BRANCH IF ONE
6032	020362'	000440				BR	LPKEJ2	;ERROR IF 3 OR MORE
6033	020364'	062713	001000		LPKWD1:	ADD	#1000,(R3)	;ADD 2 TO FUNCTION CODE
6034	020370'	122710	000075			CMPB	#=(R0)	;WAS IT EQUAL
6035	020374'	001435				BEQ	LPKRFD	;YES-RELATION FOUND
6036	020376'	105263	000001			INCB	1(R3)	;INCREMENT F.C.
6037	020402'	122710	000076			CMPB	#>(R0)	;WAS IT GTR THAN ?
6038	020406'	001430				BEQ	LPKRFD	;YES-RELATION FOUND
6039	020410'	105263	000001			INCB	1(R3)	;INCREMENT F.C.
6040	020414'	122710	000074			CMPB	#<(R0)	;WAS IT LESS THAN ?
6041	020420'	001423				BEQ	LPKRFD	;YES-RELATION FOUND
6042	020422'	000420				BR	LPKEJ2	;ERROR IF NONE OF ABOVE
6043	020424'	062713	002400		LPKWD2:	ADD	#2400,(R3)	;ADD 5 TO FUNCTION CODE
6044	020430'	152010				BISB	(R0)+,(R0)	;COMBINE=AND > ETC.
6045	020432'	122710	000077			CMPB	#077,(R0)	;WAS IT => OR >=
6046	020436'	001414				BEQ	LPKRFD	;YES-RELATION FOUND
6047	020440'	105263	000001			INCB	1(R3)	;INCREMENT F.C.
6048	020444'	122710	000075			CMPB	#075,(R0)	;WAS IT <= OR =<
6049	020450'	001407				BEQ	LPKRFD	;YES-RELATION FOUND
6050	020452'	105263	000001			INCB	1(R3)	;INCREMENT F.C.
6051	020456'	122710	000076			CMPB	#076,(R0)	;WAS IT <> OR ><
6052	020462'	001402				BEQ	LPKRFD	;YES-RELATION FOUND
6053	020464'	000167	176740		LPKEJ2:	JMP	LPKERR	;ERROR IF NONE OF ABOVE
6054	020470'	126327	000001	000112	LPKRFD:	CMPB	1(R3),#112	;IF FC 112 OR LESS
6055	020476'	101405				BLOS	LPKRFD	;CONTINUE
6056	020500'	126327	000001	000117		CMPB	1(R3),#117	;IF FC 117 OR MORE
6057	020506'	103001				BHIS	LPKRFD	;CONTINUE
6058	020510'	000765				BR	LPKEJ2	;OTHERWISE REPORT ERROR

6059	020512'	000167	176440	LPKRFF: JMP	LPKSVP	;GO SET MODEL VECTOR PTR
6060	020516'	004567	005664	LPKBT1: JSR	R5,SCAN	;SCAN IF BIT STRING
6061	020522'	122710	000015		CMPB #15,(R0)	;CHECK FOR CAR RET
6062	020526'	001001			BNE LPKTG4	
6063	020530'	000755			BR LPKEJ2	;ERROR IF FOUND
6064	020532'	005301		LPKTG4: DEC	R1	
6065	020534'	160100			SUB R1,R0	;POINTER TO FIRST CHAR.
6066	020536'	122710	000103		CMPB #'C,(R0)	
6067	020542'	001404			BEQ LPKTG5	;BRANCH IF C (CLEAR)
6068	020544'	122710	000123		CMPB #'S,(R0)	
6069	020550'	001403			BEQ LPKTG6	;BRANCH IF S (SET)
6070	020552'	000761			BR LPKBT1	;SCAN NEXT WORD
6071	020554'	105263	000001	LPKTG5: INCB	1(R3)	;INCREMENT FUNCTION CODE
6072	020560'	000167	176372	LPKTG6: JMP	LPKSVP	;GO SET MODEL VECTOR PTR
6073						
6074	020564'	004567	005616	LPKPRT: JSR	R5,SCAN	
6075	020570'	122710	000052		CMPB #'*(R0)	;IS FIRST CHAR A *
6076	020574'	001502			BEQ LPKXT	;YES-GO PACK TEXT
6077	020576'	122710	000043		CMPB #'*(R0)	;IS FIRST CHAR A *
6078	020602'	001474			BEQ LPKTR	;YES - PACK TEXT WITH NO RETN
6079	020604'	004567	005576		JSR R5,SCAN	;SCAN SECOND WORD
6080	020610'	022701	000002		CMP #2,R1	;DOES LENGTH=2
6081	020614'	001013			BNE LPKTG7	;NO-SCAN MORE
6082	020616'	005301			DEC R1	
6083	020620'	160100			SUB R1,R0	
6084	020622'	122027	000111		CMPB (R0)+, #'I	;1ST CHAR AN I?
6085	020626'	001006			BNE LPKTG7	;NO-SCAN MORE
6086	020630'	121027	000116		CMPB (R0), #'N	;2ND CHAR AN N?
6087	020634'	001003			BNE LPKTG7	;NO-SCAN MORE
6088	020636'	062713	002000		ADD #2000,(R3)	;INCR F.C. BY 4
6089	020642'	000422			BR LPKTGA	
6090	020644'	004567	005536	LPKTG7: JSR	R5,SCAN	;SCAN NEXT WORD
6091	020650'	122710	000015		CMPB #15,(R0)	;CARRIAGE RETURN?
6092	020654'	001001			BNE LPKTG8	;NO-CONTINUE
6093	020656'	000702			BR LPKEJ2	;YES-ERROR
6094	020660'	004567	005522	LPKTG8: JSR	R5,SCAN	;SCAN NEXT WORD
6095	020664'	005301			DEC R1	
6096	020666'	160100			SUB R1,R0	
6097	020670'	122027	000111		CMPB (R0)+, #'I	;1ST CHAR AN I?
6098	020674'	001003			BNE LPKTGA	;NO-ERROR
6099	020676'	121027	000116		CMPB (R0), #'N	;2ND CHAR AN N?
6100	020702'	001402			BEQ LPKTGA	;YES-CONTINUE
6101	020704'	000167	176520	LPKTG9: JMP	LPKERR	;NO-ERROR
6102	020710'	004567	005472	LPKTGA: JSR	R5,SCAN	;SCAN NEXT WORD
6103	020714'	005301			DEC R1	
6104	020716'	160100			SUB R1,R0	
6105	020720'	121027	000107		CMPB (R0), #'G	
6106	020724'	001661			BEQ LPKRFD	;BRANCH IF G (GRAPHICS)
6107	020726'	121027	000101		CMPB (R0), #'A	
6108	020732'	001656			BEQ LPKRFD	;BRANCH IF A (ASCII)
6109	020734'	105263	000001		INCB 1(R3)	;INCREMENT F.C.
6110	020740'	121027	000117		CMPB (R0), #'0	
6111	020744'	001651			BEQ LPKRFD	;BRANCH IF 0 (OCTAL)
6112	020746'	105263	000001		INCB 1(R3)	;INCREMENT F.C.
6113	020752'	121027	000104		CMPB (R0), #'D	
6114	020756'	001644			BEQ LPKRFD	;BRANCH IF D (DECIMAL)

6115	020760'	105263	000001		INCB	1(R3)	; INCREMENT F.C.
6116	020764'	121027	000102		CMPB	(R0), #'B	
6117	020770'	001637			BEQ	LPKRF0	; BRANCH IF B (BINARY)
6118	020772'	000744			BR	LPKTG9	; ERROR IF NONE OF ABOVE
6119	020774'	062713	044400	LPKTNR:	ADD	#44400, (R3)	; ADD 111 TO F.C.
6120	021000'	000402			BR	LPKADP	; GO ADJUST PACK POINTER
6121	021002'	062713	034400	LPKTX:	ADD	#34400, (R3)	; ADD 71 TO F.C.
6122	021006'	062703	000006	LPKADP:	ADD	#6, R3	; ADJUST PACK POINTER
6123	021012'	005200		LPKITX:	INC	R0	
6124	021014'	112023		LPKLP8:	MOVB	(R0)+, (R3)+	; MOVE WORD TO PACKED CODE
6125	021016'	122760	000015	177777	CMPB	#15, -1(R0)	; FOUND CARRIAGE RET ?
6126	021024'	001373			BNE	LPKLP8	; NO-MOVE ANOTHER BYTE
6127	021026'	032703	000001	LPKEVN:	BIT	#1, R3	; NEXT ADDRESS EVEN ?
6128	021032'	001405			BEQ	LPKTXD	; YES - TEXT DONE
6129	021034'	112763	000040	177777	MOVB	#40, -1(R3)	; STORE BLANK IN PLACE OF CR
6130	021042'	112723	000015		MOVB	#15, (R3)+	; STORE CR TO MAKE IT EVEN
6131	021046'	000167	176362	LPKTXD:	JMP	LPKEXT	; YES-EXIT
6132	021052'	005004		LPKFIL:	CLR	R4	; SET WORD COUNT=0
6133	021054'	004567	005326	LPKFL1:	JSR	R5, SCAN	; SCAN A WORD
6134	021060'	005204			INC	R4	; INCREMENT WORD COUNT
6135	021062'	005301		LPKFT1:	DEC	R1	
6136	021064'	160100			SUB	R1, R0	
6137	021066'	121027	000122		CMPB	(R0), #'R	; 1ST CHAR R (RANDOM)
6138	021072'	001415			BEQ	LPKFL2	; YES-BRANCH
6139	021074'	121027	000107		CMPB	(R0), #'G	; 1ST CHAR G (GRAPHICS)
6140	021100'	001424			BEQ	LPKFL3	; YES-BRANCH
6141	021102'	121027	000101		CMPB	(R0), #'A	; 1ST CHAR A (ASCII) ?
6142	021106'	001421			BEQ	LPKFL3	; YES - BRANCH
6143	021110'	121027	000120		CMPB	(R0), #'P	; 1ST CHAR P (PATTERN)
6144	021114'	001434			BEQ	LPKFL5	
6145	021116'	121027	000015		CMPB	(R0), #15	; CARRIAGE RETURN?
6146	021122'	001425			BEQ	LPKFL4	; YES-BRANCH
6147	021124'	000753			BR	LPKFL1	; SCAN SOME MORE
6148	021126'	004567	005254	LPKFL2:	JSR	R5, SCAN	
6149	021132'	005204			INC	R4	; INCREMENT WORD COUNT
6150	021134'	121027	000015		CMPB	(R0), #15	; LAST WORD ?
6151	021140'	001350			BNE	LPKFT1	; NO-DISREGARD
6152	021142'	022704	000004		CMP	#4, R4	; WORD COUNT=4
6153	021146'	001446			BEQ	LPKFA1	; YES
6154	021150'	000443			BR	LPKFA2	; NO
6155	021152'	004567	005230	LPKFL3:	JSR	R5, SCAN	
6156	021156'	005204			INC	R4	; INCREMENT WORD COUNT
6157	021160'	121027	000015		CMPB	(R0), #15	; LAST WORD ?
6158	021164'	001336			BNE	LPKFT1	; NO-DISREGARD
6159	021166'	022704	000004		CMP	#4, R4	; WORD COUNT=4
6160	021172'	001430			BEQ	LPKFA3	; YES
6161	021174'	000425			BR	LPKFA4	; NO
6162	021176'	022704	000004	LPKFL4:	CMP	#4, R4	; WORD COUNT=4
6163	021202'	001420			BEQ	LPKFA5	; YES
6164	021204'	000415			BR	LPKFA6	
6165	021206'	004567	005174	LPKFL5:	JSR	R5, SCAN	
6166	021212'	005204			INC	R4	; INCREMENT WORD COUNT
6167	021214'	121027	000015		CMPB	(R0), #15	; LAST WORD ?
6168	021220'	001320			BNE	LPKFT1	; NO - DISREGARD
6169	021222'	022704	000004		CMP	#4, R4	; WORD COUNT = 4?
6170	021226'	001402			BEQ	LPKFA7	; YES

6171	021230'	105263	000001		INCB	1(R3)		;NO
6172	021234'	105263	000001		LPKFA7:	INCB	1(R3)	
6173	021240'	105263	000001		LPKFA6:	INCB	1(R3)	;ADD
6174	021244'	105263	000001		LPKFA5:	INCB	1(R3)	;APPROPRIATE
6175	021250'	105263	000001		LPKFA4:	INCB	1(R3)	;AMOUNT
6176	021254'	105263	000001		LPKFA3:	INCB	1(R3)	;TO
6177	021260'	105263	000001		LPKFA2:	INCB	1(R3)	;FUNCTION
6178	021264'				LPKFA1:			;CODE
6179	021264'	000167	175666		JMP	LPKSVF		;GO SET VECTOR POINTER
6180								
6181								
6182	021270'	000000			LPKPTP:	.WORD	0	;P TBL POINTER SAVE
6183	021272'	000000			LPKPKP:	.WORD	0	;PACK POINTER SAVE
6184	021274'	000000			LBDLIN:	.WORD	0	;BAD LINE NBR FLAG
6185	021276'	000000			LPKIFL:	.WORD	0	;INTERNAL TBL FLG
6186	021300'	000000			LMSTBE:	.WORD	0	;MODEL STMT TBL END
6187								
6188								
6189								
6190	021302'	047522	040524	042524	LPKTBL:	.ASCII	/ROTATE/	;PACK/UNPACK TABLE
6191	021310'	000	000		.BYTE	0,0		
6192	021312'	020040	047111	051103	.ASCII	/ INCR/		
6193	021320'	010	000		.BYTE	010,0		
6194	021322'	020040	042504	051103	.ASCII	/ DECR/		
6195	021330'	020	000		.BYTE	020,0		
6196	021332'	020040	047515	042526	.ASCII	/ MOVE/		
6197	021340'	030	000		.BYTE	030,0		
6198	021342'	020040	040440	042104	.ASCII	/ ADD/		
6199	021350'	040	000		.BYTE	040,0		
6200	021352'	020040	051440	041125	.ASCII	/ SUB/		
6201	021360'	050	000		.BYTE	050,0		
6202	021362'	020040	020040	043111	.ASCII	/ IF/		
6203	021370'	100	377		.BYTE	100,377		
6204	021372'	020040	020040	043111	.ASCII	/ IF/		
6205	021400'	110	377		.BYTE	110,377		
6206	021402'	020040	042522	042101	.ASCII	/ READ/		
6207	021410'	120	000		.BYTE	120,0		
6208	021412'	053440	044522	042524	.ASCII	/ WRITE/		
6209	021420'	130	000		.BYTE	130,0		
6210	021422'	042526	044522	054506	.ASCII	/VERIFY/		
6211	021430'	140	000		.BYTE	140,0		
6212	021432'	050040	044522	052116	.ASCII	/ PRINT/		
6213	021440'	150	377		.BYTE	150,377		
6214	021442'	041040	042522	045501	.ASCII	/ BREAK/		
6215	021450'	160	000		.BYTE	160,0		
6216	021452'	020040	044506	046114	.ASCII	/ FILL/		
6217	021460'	170	377		.BYTE	170,377		
6218	021462'	020040	042440	042116	.ASCII	/ END/		
6219	021470'	240	000		.BYTE	240,0		
6220	021472'	050040	044522	052116	.ASCII	/ PRINT/		
6221	021500'	241	000		.BYTE	241,0		
6222	021502'	020040	047514	042101	.ASCII	/ LOAD/		
6223	021510'	242	000		.BYTE	242,0		
6224	021512'	042040	046105	054501	.ASCII	/ DELAY/		
6225	021520'	243	000		.BYTE	243,0		
6226	021522'	043440	020117	047524	.ASCII	/ GO TO/		

6227	021530'	244	000		.BYTE	244,0	
6228	021532'	020040	047507	047524	.ASCII	/ GOTO/	
6229	021540'	244	000		.BYTE	244,0	
6230	021542'	020040	044514	045516	.ASCII	/ LINK/	
6231	021550'	245	000		.BYTE	245,0	
6232	021552'	042522	052524	047122	.ASCII	/RETURN/	
6233	021560'	246	000		.BYTE	246,0	
6234	021562'	020040	051440	052105	.ASCII	/ SET/	
6235	021570'	247	377		.BYTE	247,377	
6236	021572'	041440	042514	051101	.ASCII	/ CLEAR/	
6237	021600'	250	377		.BYTE	250,377	
6238		001			.IF NDF MM		
6239	021602'	042526	052103	051117	.ASCII	/VECTOR/	
6240	021610'	251	000		.BYTE	251,0	
6241		000			.ENDC		
6242	021612'	046040	052105	047507	.ASCII	/ LETGO/	
6243	021620'	252	000		.BYTE	252,0	
6244	021622'	042440	052116	054522	.ASCII	/ ENTRY/	
6245	021630'	253	000		.BYTE	253,0	
6246	021632'	020040	054105	052111	.ASCII	/ EXIT/	
6247	021640'	254	000		.BYTE	254,0	
6248		001			.IF NDF MM		
6249	021642'	051040	051505	052105	.ASCII	/ RESET/	
6250	021650'	255	000		.BYTE	255,0	
6251		000			.ENDC		
6252	021652'	020040	047504	042516	.ASCII	/ DONE/	
6253	021660'	256	000		.BYTE	256,0	
6254	021662'	042504	042514	042524	.ASCII	/DELETE/	
6255	021670'	257	000		.BYTE	257,0	
6256	021672'	042516	040507	042524	.ASCII	/NEGATE/	
6257	021700'	260	000		.BYTE	260,0	
6258	021702'	050040	044522	052116	.ASCII	/ PRINT/	
6259	021710'	261	000		.BYTE	261,0	
6260	021712'	050040	052501	042523	.ASCII	/ PAUSE/	
6261	021720'	262	000		.BYTE	262,0	
6262	021722'	020040	047527	042122	.ASCII	/ WORD/	
6263	021730'	263	000		.BYTE	263,0	
6264	021732'	020040	020040	020040	.ASCII	/ /	
6265	021740'	377	000		.BYTE	377,0	;END OF TABLE
6266	021742'	000006			.BLKB	6	
6267	021750'	000000			LPKTBE: .WORD	0	
6268							
6269							
6270							
6271							
6272							
6273							
6274	021752'	000242	003046		MPNTR: .WORD	242,LLOAD-LBASE1	
6275	021756'	000011	003055		.WORD	011,LINCR1-LBASE1	
6276	021762'	000021	003055		.WORD	021,LDECR1-LBASE1	
6277	021766'	000013	003057		.WORD	013,LINCR2-LBASE1	
6278	021772'	000023	003057		.WORD	023,LDECR2-LBASE1	
6279	021776'	000030	003632		.WORD	030,LMOVE0-LBASE1	
6280	022002'	000033	003064		.WORD	033,LMOVE1-LBASE1	
6281	022006'	000035	003071		.WORD	035,LMOVE2-LBASE1	
6282	022012'	000043	003064		.WORD	043,LADD1-LBASE1	

6283	022016'	000053	003101	.WORD	053,LSUB1-LBASE1
6284	022022'	000045	003071	.WORD	045,LADD2-LBASE1
6285	022026'	000055	003110	.WORD	055,LSUB2-LBASE1
6286	022032'	000260	003055	.WORD	260,LNEGAT-LBASE1
6287	022036'	000247	003122	.WORD	247,LSET-LBASE1
6288	022042'	000250	003122	.WORD	250,LCLEAR-LBASE1
6289	022046'	000100	003130	.WORD	100,LIFST-LBASE1
6290	022052'	000101	003150	.WORD	101,LIFCLR-LBASE1
6291	022056'	000102	003172	.WORD	102,LIFEQ1-LBASE1
6292	022062'	000103	003204	.WORD	103,LIFGT1-LBASE1
6293	022066'	000104	003216	.WORD	104,LIFLT1-LBASE1
6294	022072'	000105	003230	.WORD	105,LIFGE1-LBASE1
6295	022076'	000106	003243	.WORD	106,LIFLE1-LBASE1
6296	022102'	000107	003256	.WORD	107,LIFNE1-LBASE1
6297	022106'	000112	003271	.WORD	112,LIFEQ2-LBASE1
6298	022112'	000117	003306	.WORD	117,LIFNE2-LBASE1
6299	022116'	000244	003324	.WORD	244,LGOTO-LBASE1
6300	022122'	000245	003324	.WORD	245,LLINK-LBASE1
6301	022126'	000246	003326	.WORD	246,LRETN-LBASE1
6302	022132'	000241	003330	.WORD	241,LPRN1-LBASE1
6303	022136'	000261	003333	.WORD	261,LPRNR-LBASE1
6304	022142'	000154	003336	.WORD	154,LPRNG1-LBASE1
6305	022146'	000155	003351	.WORD	155,LPRN01-LBASE1
6306	022152'	000156	003364	.WORD	156,LPRN01-LBASE1
6307	022156'	000157	003401	.WORD	157,LPRN01-LBASE1
6308	022162'	000150	003415	.WORD	150,LPRNG2-LBASE1
6309	022166'	000151	003433	.WORD	151,LPRN02-LBASE1
6310	022172'	000152	003451	.WORD	152,LPRN02-LBASE1
6311	022176'	000153	003471	.WORD	153,LPRN02-LBASE1
6312	022202'	000161	003055	.WORD	161,LBRK1-LBASE1
6313	022206'	000163	003704	.WORD	163,LBRK2-LBASE1
6314	022212'	000170	003510	.WORD	170,LFILR1-LBASE1
6315	022216'	000172	003526	.WORD	172,LFILG1-LBASE1
6316	022222'	000174	003543	.WORD	174,LFILV1-LBASE1
6317	022226'	000171	003552	.WORD	171,LFILR2-LBASE1
6318	022232'	000173	003573	.WORD	173,LFILG2-LBASE1
6319	022236'	000175	003613	.WORD	175,LFILV2-LBASE1
6320	022242'	000176	003543	.WORD	176,LFILP1-LBASE1
6321	022246'	000177	003613	.WORD	177,LFILP2-LBASE1
6322	022252'	000001	003055	.WORD	001,LROT1-LBASE1
6323	022256'	000003	003625	.WORD	003,LROT2-LBASE1
6324	022262'	000243	003055	.WORD	243,LDELAY-LBASE1
6325	022266'	000120	003632	.WORD	120,LREAD1-LBASE1
6326	022272'	000121	003055	.WORD	121,LREAD2-LBASE1
6327	022276'	000123	003633	.WORD	123,LREAD3-LBASE1
6328	022302'	000125	003642	.WORD	125,LREAD4-LBASE1
6329	022306'	000130	003632	.WORD	130,LWRIT1-LBASE1
6330	022312'	000131	003055	.WORD	131,LWRIT2-LBASE1
6331	022316'	000133	003656	.WORD	133,LWRIT3-LBASE1
6332	022322'	000135	003665	.WORD	135,LWRIT4-LBASE1
6333	022326'	000143	003543	.WORD	143,LVERF1-LBASE1
6334	022332'	000145	003613	.WORD	145,LVERF2-LBASE1
6335		001			
6336	022336'	000251	003677	.WORD	251,LVECT-LBASE1
6337		000			
6338	022342'	000252	003632	.WORD	252,LETGO-LBASE1

.IF NDF MM
.ENDC

6339	022346'	000253	003632		.WORD	253,LENTRY-LBASE1	
6340	022352'	000254	003632		.WORD	254,LEXIT-LBASE1	
6341		001			.IF NDF	MM	
6342	022356'	000255	003632		.WORD	255,LRESET-LBASE1	
6343		000			.ENDC		
6344	022362'	000262	003632		.WORD	252,LPAUSE-LBASE1	
6345	022366'	000263	003055		.WORD	263,LWORD-LBASE1	
6346	022372'	000240	003632		.WORD	240,LEND-LBASE1	
6347	022376'	000377	003711		.WORD	377,LTBEND-LBASE1	
6348							
6349							
6350							
6351							
6352							
6353							
6354	022402'	053777	052111	177510	LOAD:	.ASCIZ	<377>/WITH/<377>
	022410'	000					

;END OF TABLE

MODEL STATEMENT TABLE

6355							
6356	022411'				LWORD:		
6357	022411'				LBRK1:		
6358	022411'				LWRT2:		
6359	022411'				LINCR1:		
6360	022411'				LNEGAT:		
6361	022411'				LROT1:		
6362	022411'				LDELAY:		
6363	022411'				LREAD2:		
6364	022411'	377	000		LDECR1:	.BYTE	377,0
6365	022413'				LINCR2:		
6366	022413'	377	054502	000377	LDECR2:	.ASCIZ	<377>/BY/<377>
6367	022420'				LADD1:		
6368	022420'	052377	177517	000	LMOVE1:	.ASCIZ	<377>/TO/<377>
6369	022425'				LADD2:		
6370	022425'	377	052101	052377	LMOVE2:	.ASCIZ	<377>/AT/<377>/TO/<377>
	022432'	177517	000				
6371	022435'	377	051106	046517	LSUB1:	.ASCIZ	<377>/FROM/<377>
	022442'	000377					
6372	022444'	040777	177524	051106	LSUB2:	.ASCIZ	<377>/AT/<377>/FROM/<377>
	022452'	046517	000377				
6373	022456'				LSET:		
6374	022456'	041377	052111	000375	LCLEAR:	.ASCIZ	<377>/BIT/<375>
6375	022464'	041377	052111	051775	LIFST:	.ASCIZ	<377>/BIT/<375>/SET GO TO/<376>
	022472'	052105	043440	020117			
	022500'	047524	000376				
6376	022504'	041377	052111	041775	LIFCLR:	.ASCIZ	<377>/BIT/<375>/CLEAR GO TO/<376>
	022512'	042514	051101	043440			
	022520'	020117	047524	000376			
6377	022526'	036777	043777	020117	LIFEQ1:	.ASCIZ	<377>/=/<377>/GO TO/<376>
	022534'	047524	000376				
6378	022540'	037377	043777	020117	LIFGT1:	.ASCIZ	<377>/>/<377>/GO TO/<376>
	022546'	047524	000376				
6379	022552'	036377	043777	020117	LIFLT1:	.ASCIZ	<377>/</<377>/GO TO/<376>
	022560'	047524	000376				
6380	022564'	036777	177476	047507	LIFGE1:	.ASCIZ	<377>/=)/<377>/GO TO/<376>
	022572'	052040	177117	000			
6381	022577'	377	036474	043777	LIFLE1:	.ASCIZ	<377>/<=/<377>/GO TO/<376>
	022604'	020117	047524	000376			

DTMGAC.P11 PACK SUBROUTINE

6382	022612'	036377	177476	047507	LIFNE1: .ASCIZ	<377>/(<377>/GO TO/<376>
	022620'	052040	177117	000		
6383	022625'	377	052101	036777	LIFEQ2: .ASCIZ	<377>/AT/<377>/=<377>/GO TO/<376>
	022632'	043777	020117	047524		
	022640'	000376				
6384	022642'	040777	177524	037074	LIFNE2: .ASCIZ	<377>/AT/<377>/(<377>/GO TO/<376>
	022650'	043777	020117	047524		
	022656'	000376				
6385	022660'				LGOTO:	
6386	022660'	376	000		LLINK: .BYTE	376,0
6387	022662'	373	000		LRETN: .BYTE	373,0
6388	022664'	176052	000		LPRNI: .ASCIZ	/*/<374>
6389	022667'	043	000374		LPRNPR: .ASCIZ	/#/<374>
6390	022672'	044777	020116	051501	LPRNG1: .ASCIZ	<377>/IN ASCII/'373>
	022700'	044503	175511	000		
6391	022705'	377	047111	047440	LPRN01: .ASCIZ	<377>/IN OCTAL/<373>
	022712'	052103	046101	000373		
6392	022720'	044777	020116	042504	LPRND1: .ASCIZ	<377>/IN DECIMAL/<373>
	022726'	044503	040515	175514		
	022734'	000				
6393	022735'	377	047111	041040	LPRNB1: .ASCIZ	<377>/IN BINARY/<373>
	022742'	047111	051101	175531		
	022750'	000				
6394	022751'	377	052101	044777	LPRNG2: .ASCIZ	<377>/AT/<377>/IN ASCII/<373>
	022756'	020115	051501	044503		
	022764'	175511	000			
6395	022767'	377	052101	044777	LPRN02: .ASCIZ	<377>/AT/<377>/IN OCTAL/<373>
	022774'	020116	041517	040524		
	023002'	175514	000			
6396	023005'	377	052101	044777	LPRND2: .ASCIZ	<377>/AT/<377>/IN DECIMAL/<373>
	023012'	020116	042504	044503		
	023020'	040515	175514	000		
6397	023025'	377	052101	044777	LPRNB2: .ASCIZ	<377>/AT/<377>/IN BINARY/<373>
	023032'	020116	044502	040516		
	023040'	054522	000373			
6398	023044'	053777	052111	020110	LFILR1: .ASCIZ	<377>/WITH RANDOM/<373>
	023052'	040522	042116	046517		
	023060'	000373				
6399	023062'	053777	052111	020110	LFILG1: .ASCIZ	<377>/WITH ASCII/<373>
	023070'	051501	044503	175511		
	023076'	000				
6400	023077'				LFILP1:	
6401	023077'				LVERF1:	
6402	023077'	377	044527	044124	LFILV1: .ASCIZ	<377>/WITH/<377>
	023104'	000377				
6403	023106'	040777	177524	044527	LFILR2: .ASCIZ	<377>/AT/<377>/WITH RANDOM/<373>
	023114'	044124	051040	047101		
	023122'	047504	175515	000		
6404	023127'	377	052101	053777	LFILG2: .ASCIZ	<377>/AT/<377>/WITH ASCII/<373>
	023134'	052111	020110	051501		
	023142'	044503	175511	000		
6405	023147'				LFILP2:	
6406	023147'				LVERF2:	
6407	023147'	377	052101	053777	LFILV2: .ASCIZ	<377>/AT/<377>/WITH/<377>
	023154'	052111	175510	000		
6408	023161'	377	052101	000377	LR0T2: .ASCIZ	<377>/AT/<377>

```

6409 023166'          LMOVED:
6410 023166'          LWRT1:
6411 023166'          LREAD1:
6412 023166'          LETGO:
6413 023166'          LENTRY:
6414 023166'          LEXIT:
6415          001      .IF NDF MM
6416 023166'          LRESET:
6417          000      .ENDC
6418 023166'          LPAUSE:
6419 023166'          000      LEND: .BYTE 0
6420 023167'          377 047111 047524 LREAD3: .ASCIZ <377>/INTO/<377>
        023174' 000377
6421 023176' 044777 052116 177517 LREAD4: .ASCIZ <377>/INTO/<377>/FROM/<375>
        023204' 051106 046517 000375
6422 023212' 043377 047522 177515 LWRT3: .ASCIZ <377>/FROM/<377>
        023220' 000
6423 023221' 377 051106 046517 LWRT4: .ASCIZ <377>/FROM/<377>/TO/<375>
        023226' 052377 176517 000
6424          001      .IF NDF MM
6425 023233' 377 047524 000376 LVECT: .ASCIZ <377>/TO/<376>
6426          000      .ENDC
6427 023240' 047777 176516 000 LBRK2: .ASCIZ <377>/ON/<375>
6428 023245' 000      LTBEND: .BYTE 000 ;END OF TABLE
6429          .EVEN
6430
6431
6432 023246' 000      LDONE: .BYTE 0 ;DONE FLAG
6433 023247' 000      LDELET: .BYTE 0 ;DELETE FLAG (MUST FOLLOW LDONE)
6434 023250' 000      LIGNOR: .BYTE 0 ;IGNORE FLAG
6435 023251' 000      LERROR: .BYTE 0 ;ERROR FLAG
6436
6437 023252' 047504 042516 LASDON: .ASCII /DONE/
6438 023256' 000000 LPKEPT: .WORD 0
6439 023260' 000000 LPKPSV: .WORD 0
6440 023262' 000000 LPKMPS: .WORD 0
6441
6442
6443 023264' 000110 LPKBUF: .BLKB 72. ;PACK BUFFER

```



```

6445
6446
6447
6448
6449
6450
6451
6452
6453
6454
6455
6456
6457
6458
6459
6460
6461
6462
6463 023374' 010346
6464 023376' 010446
6465 023400' 010546
6466 023402' 005067 000702
6467 023406' 010705
6468 023410' 062705 010763
6469 023414' 012700 000111
6470 023420' 112745 000040
6471 023424' 005300
6472 023426' 001374
6473 023430' 016704 000660
6474 023434' 112467 000651
6475 023440' 111403
6476 023442' 112714 000377
6477 023446' 010400
6478 023450' 010501
6479 023452' 004567 015346
6480 023456' 110314
6481 023460' 010105
6482 023462' 010004
6483 023464' 012467 000622
6484 023470' 112725 000011
6485 023474' 010700
6486 023476' 062700 176254
6487 023502' 010302
6488 023504' 122710 000377
6489 023510' 001405
6490 023512' 120310
6491 023514' 001437
6492 023516' 062700 000004
6493 023522' 000770
6494 023524' 016700 000566
6495 023530' 062700 000354
6496 023534' 061000
6497 023536' 010067 175536
6498 023542' 016700 000550
6499 023546' 062700 000352
E500 023552' 061000
    
```

.SBTTL UNPACK SUBROUTINE

```

*****
THIS SUBROUTINE EXPANDS ONE LINE OF
PACKED CODE BACK TO A FORMAT SIMILAR
TO THE USERS ORIGINAL ENTRY.

CALLING SEQUENCE - JSR R5,UNPACK

ENTRY - LUPKPT POINTS TO PACKED CODE.

EXIT - R0 CONTAINS UNPACKED CHARACTER COUNT
      R1 AND LUPKPT POINT TO NEXT PACKED ENTRY
*****
    
```

```

UNPACK: MOV R3,-(SP)
        MOV R4,-(SP) ;SAVE R3, R4 AND R5
        MOV R5,-(SP)
        CLR LENDFL ;RESET END FLAG
        MOV PC,R5
        ADD #LINBUF+73.-.,R5 ;SET OUTPUT POINTER
        MOV #73.,R0
LUPLP1: MOVB #40,-(R5) ;CLEAR OUTPUT LINE BFR
        DEC R0
        BNE LUPLP1
        MOV LUPKPT,R4 ;POINT R4 AT PACKED CODE
        MOVB (R4)+,LPLNTH ;GET LENGTH
        MOVB (R4),R3 ;GET FUNCTION CODE
        MOVB #377,(R4) ;PUT NULL (ONES) FOR F.C.
        MOV R4,R0
        MOV R5,R1
        JSR R5,DECASC ;CONVERT LINE # TO ASCII
        MOVB R3,(R4) ;RESTORE F.C.
        MOV R1,R5
        MOV R0,R4
        MOV (R4)+,LPHYSL ;GET AND SAVE PHYSICAL ADDRESS
        MOVB #11,(R5)+ ;STORE A TAB
        MOV PC,R0
        ADD #LMPNTR-.,R0 ;SET R0 TO MODEL VECTOR TBL
        MOV R3,R2 ;SAVE FUNCTION CODE
LUPLP2: CMPB #377,(R0)
        BEQ LUPSEV ;GO SET EXTERNAL VECTOR
        CMPB R3,(R0)
        BEQ LUPSEV
        ADD #4,R0
        BR LUPLP2
LUPSEV: MOV LUPPTP,R0 ;DETERMINE END
        ADD #PTLGTH+DVCTEP,R0
        ADD (R0),R0 ;OF EXT VEC TBL
        MOV R0,LASTBE
        MOV LUPPTP,R0 ;POINT R0 AT START
        ADD #PTLGTH+DVVTEP,R0 ;OF
        ADD (R0),R0 ;EXTERN VECTOR TBL
    
```

6501	023554'	020067	175520	10\$:	CMP	RO,LMSTBE	;RO AT TBL END?
6502	023560'	001520			BEQ	LUPEXT	;YES - EXIT
6503	023562'	121003			CMPB	(RO),R3	;CHECK F.C. IN TBL
6504	023564'	001403			BEQ	LUPSEP	;BRANCH IF FOUND
6505	023566'	062700	000004		ADD	#4,RO	;BUMP POINTER
6506	023572'	000770			BR	10\$;CHECK NEXT ENTRY
6507	023574'	062700	000002	LUPSEP:	ADD	#2,RO	
6508	023600'	016703	000512		MOV	LUPPTP,R3	
6509	023604'	062703	000242		ADD	#PTLGTH,R3	
6510	023610'	061003			ADD	(RO),R3	;SET EXTERN MDL POINTER
6511	023612'	000406			BR	LUPTMF	;GO TEST MULT FORMAT
6512	023614'	062700	000002	LUPSMP:	ADD	#2,RO	;RO CONTAINS MODEL ADDR
6513	023620'	011003			MOV	(RO),R3	
6514	023622'	060703			ADD	PC,R3	
6515	023624'	062703	173510	LBASE2:	ADD	#LBASE1-LBASE2,R3	;R3 NOW POINTS TO MODEL
6516	023630'	120227	000237	LUPTMF:	CMPB	R2,#237	
6517	023634'	101002			BHI	LUPNPI	
6518	023636'	042702	000007		BIC	#7,R2	
6519	023642'	010701		LUPNPI:	MOV	PC,R1	
6520	023644'	062701	175444		ADD	#LPKTBL+6-,R1	;POINT R1 AT PACK TBL
6521	023650'	122711	000377	LUPLF3:	CMPB	#377,(R1)	
6522	023654'	001405			BEQ	LUPSEX	;EXIT IF F.C. NOT FOUND
6523	023656'	120211			CMPB	R2,(R1)	
6524	023660'	001433			BEQ	LUPIFD	;BRANCH IF INSTR. FOUND
6525	023662'	062701	000010		ADD	#8,R1	
6526	023666'	000770			BR	LUPLP3	
6527	023670'	016701	000422	LUPSEX:	MOV	LUPPTP,R1	;DETERMINE END
6528	023674'	062701	000352		ADD	#PTLGTH+DVVTEP,R1	;OF
6529	023700'	061101			ADD	(R1),R1	;EXT PACK TABLE
6530	023702'	062701	000006		ADD	#6,R1	;ADJUST R1
6531	023706'	010167	176036		MOV	R1,LPKTBE	
6532	023712'	016701	000400		MOV	LUPPTP,R1	;POINT R1
6533	023716'	062701	000350		ADD	#PTLGTH+DVPTEP,R1	;AT START
6534	023722'	061101			ADD	(R1),R1	;OF EXT PACK TABLE
6535	023724'	062701	000006		ADD	#6,R1	;ADJUST R1
6536	023730'	020167	176014	10\$:	CMP	R1,LPKTBE	;R1 AT TBL END?
6537	023734'	001432			BEQ	LUPEXT	;YES - EXIT
6538	023736'	120211			CMPB	R2,(R1)	
6539	023740'	001403			BEQ	LUPIFD	;BRANCH IF INSTR FND
6540	023742'	062701	000010		ADD	#8,R1	
6541	023746'	000770			BR	10\$;ELSE KEEP LOOKING
6542	023750'	012700	000006	LUPIFD:	MOV	#6,RO	
6543	023754'	160001			SUB	RO,R1	;BACK UP TO INST WORD
6544	023756'	122711	000040	LUPLP4:	CMPB	#40,(R1)	;CHAR A BLANK?
6545	023762'	001004			BNE	LUPSIN	;NO-STORE INST WORD
6546	023764'	005201			INC	R1	;BUMP R1
6547	023766'	005300			DEC	RO	;DECREMENT CHARA COUNT
6548	023770'	001372			BNE	LUPLP4	;LOOP IF ANY REMAINING
6549	023772'	000413			BR	LUPEXT	;EXIT IF ALL BLANKS
6550	023774'	112125		LUPSIN:	MOVB	(R1)+,(R5)+	;STORE INSTRUCTION WORD
6551	023776'	005300			DEC	RO	
6552	024000'	001375			BNE	LUPSIN	
6553	024002'	112725	000040		MOVB	#40,(R5)+	;STORE A BLANK
6554	024006'	122702	000240		CMPB	#240,R2	;WAS IT AN 'END'
6555	024012'	001035			BNE	LUPEMB	;NO-CONTINUE
6556	024014'	012767	000001 000266		MOV	#1,LENDFL	;YES-SET END FLAG

6557	024022'	112725	000015		LUPEXT: MOVB	#15, (R5)+		;STORE A CR/LF
6558	024026'	112725	000012			MOVB	#12, (R5)+	
6559	024032'	010700				MOV	PC, R0	
6560	024034'	062700	010226			ADD	#LINBUF--, R0	
6561	024040'	160005				SUB	R0, R5	
6562	024042'	010500				MOV	R5, R0	;R0=UNPACKED CHAR COUNT
6563	024044'	116701	000241			MOVB	LPLNTH, R1	
6564	024050'	001003				BNE	LPLNNZ	;CONTINUE IF LENGTH NOT 0
6565	024052'	112767	000001	000230	LPLNNZ: MOVB	#1, LENDFL		;OTHERWISE SET END FLAG
6566	024060'	060167	000230			ADD	R1, LUPKPT	;POINT LUPKPT AND
6567	024064'	016701	000224			MOV	LUPKPT, R1	;R1 AT NEXT PACKED ENTRY
6568	024070'	016702	000216			MOV	LPHYSL, R2	;R2 AT PHYSICAL ADDRESS
6569	024074'	012605				MOV	(SP)+, R5	
6570	024076'	012604				MOV	(SP)+, R4	;RESTORE R3, R4, R5
6571	024100'	012603				MOV	(SP)+, R3	
6572	024102'	000205				RTS	R5	;RETURN TO CALLING PGM
6573	024104'	005203			LUPAMP: INC	R3		;ADVANCE MODEL POINTER
6574	024106'	122713	000377		LUPEMB: CMPB	#377, (R3)		;EXAMINE MODEL BYTE
6575	024112'	001446				BEQ	LUPVAR	;GO UNPACK VARIABLE
6576	024114'	122713	000376			CMPB	#376, (R3)	
6577	024120'	001425				BEQ	LUPLIN	;GO UNPACK LINE NBR
6578	024122'	122713	000375			CMPB	#375, (R3)	
6579	024126'	001412				BEQ	LUPBIT	;GO UNPACK BIT NBR
6580	024130'	122713	000374			CMPB	#374, (R3)	
6581	024134'	001460				BEQ	LUIMP	;GO UNPACK IMM PRINT
6582	024136'	122713	000373			CMPB	#373, (R3)	
6583	024142'	001727				BEQ	LUPEXT	;TAKE SPECIAL EXIT
6584	024144'	105713				TSTB	(R3)	;IF MODEL BYTE=0
6585	024146'	001725				BEQ	LUPEXT	;EXIT
6586	024150'	111325				MOVB	(R3), (R5)+	;STORE MODEL CHARACTER
6587	024152'	000754				BR	LUPAMP	;THEN ADVANCE MODEL POINTER
6588	024154'	012400			LUPBIT: MOV	(R4)+, R0		
6589	024156'	112725	000040			MOVB	#40, (R5)+	
6590	024162'	010501				MOV	R5, R1	
6591	024164'	004567	001640			JSR	R5, BTUPAK	;UNPACK BIT NUMBERS
6592	024170'	010105				MOV	R1, R5	
6593	024172'	000744				BR	LUPAMP	;THEN ADVANCE MODEL POINTER
6594	024174'	005204			LUPLIN: INC	R4		;BUMP R4 PAST 'L'
6595	024176'	112725	000040			MOVB	#40, (R5)+	;STORE A BLANK
6596	024202'	000401				BR	LUPTG1	
6597	024204'	112425			LUPCTA: MOV	(R4)+, (R5)+		;STORE FIRST BYTE
6598	024206'	010400			LUPTG1: MOV	R4, R0		
6599	024210'	010501				MOV	R5, R1	
6600	024212'	004567	014606			JSR	R5, DECASC	;CONVERT LINE # TO ASCII
6601	024216'	010004				MOV	R0, R4	
6602	024220'	010105				MOV	R1, R5	;RESTORE POINTERS
6603	024222'	112725	000040			MOVB	#40, (R5)+	;STORE A BLANK
6604	024226'	000726				BR	LUPAMP	;THEN ADVANCE MODEL POINTER
6605	024230'	112725	000040		LUPVAR: MOV	#40, (R5)+		;STORE A BLANK
6606	024234'	121427	000040			CMPB	(R4), #40	;FIRST CHAR A BLANK ?
6607	024240'	001761				BEQ	LUPCTA	;YES-UNPACK OCTAL NBR
6608	024242'	121427	000104			CMPB	(R4), #104	;FIRST CHAR A "D" ?
6609	024246'	001004				BNE	LUPTAG	;NO-GO UNPACK TAG
6610	024250'	126427	000001	000360		CMPB	1(R4), #360	;IF 2ND CHAR > 360
6611	024256'	101352				BHI	LUPCTA	;GO UNPACK DECIMAL NBR
6612	024260'	112425			LUPTAG: MOV	(R4)+, (R5)+		

6613	024262'	112425		MOV B	(R4)+, (R5)+	
6614	024264'	112425		MOV B	(R4)+, (R5)+	
6615	024266'	112425		MOV B	(R4)+, (R5)+	
6616	024270'	112725	000040	MOV B	#40, (R5)+	; MOVE TAG AS IS
6617	024274'	000703		BR	LUPAMP	; STORE A BLANK
6618	024276'	112425		LUPIMP: MOV B	(R4)+, (R5)+	; THEN ADVANCE MODEL POINTER
6619	024300'	121427	000015	CMP B	(R4), #15	; MOVE STORED BYTE
6620	024304'	001374		BNE	LUPIMP	; UNTIL CARRIAGE RET
6621	024306'	000676		BR	LUPAMP	; THEN ADVANCE MODEL PTR
6622						
6623	024310'	000		LENDL: .BYTE	0	
6624	024311'	000		LPLNTH: .BYTE	0	
6625	024312'	000000		LPHYSL: .WORD	0	
6626	024314'	000000		LUPKPT: .WORD	0	
6627	024316'	000000		LUPPTP: .WORD	0	

.SBTTL SYNTAX ERROR SUBROUTINE

```

*****
THIS SUBROUTINE PRINTS A LINE UNDERNEATH THE USERS
ENTRY, WHEN ANYTHING IN THAT ENTRY CANNOT BE INTERPRETED.
THIS LINE CONSISTS OF ALL BLANKS EXCEPT FOR THE BAD
ENTRY SURROUNDED BY QUESTION MARKS

CALLING SEQUENCE      JSR      R5,SYNTAX
*****

```

```

6629
6630
6631
6632
6633
6634
6635
6636
6637
6638
6639
6640
6641
6642
6643 024320' 010700
6644 024322' 062700 007740
6645 024326' 016701 002274
6646 024332' 016702 002266
6647 024336' 012767 000005 000114
6648 024344' 005202
6649 024346' 112722 000077
6650 024352' 112722 000015
6651 024356' 112712 000012
6652 024362' 060167 000072
6653 024366' 162702 000003
6654 024372' 160102
6655 024374' 111262 177777
6656 024400' 112712 000077
6657 024404' 020200
6658 024406' 001414
6659 024410' 122762 000011 177777
6660 024416' 001003
6661 024420' 005302
6662 024422' 005267 000032
6663 024426' 112742 000040
6664 024432' 005267 000022
6665 024436' 000762
6666 024440' 012722 051105
6667 024444' 112712 000052
6668 024450' 004567 161550
6669 024454' 000420
6670 024456' 007603
6671 024460' 000001
6672 024462' 105267 176563
6673 024466' 000205

```

```

SYNTAX: MOV      PC,R0
        ADD      #LINBUF-.,R0      ;SET PNTR TO LINE BUFF
        MOV      LSCNCT,R1        ;GET CHAR COUNT AND
        MOV      LSCNPT,R2        ;LAST POS OF BAD WORD
        MOV      #5,LTTYCT        ;INIT OUTPUT COUNT
        INC      R2                ;POINT R2 PAST BAD WORD
        MOVB     #'?',(R2)+        ;STORE A ?
        MOVB     #15,(R2)+        ;A CARRIAGE RETURN
        MOVB     #12,(R2)         ;AND A LINE FEED
        ADD      R1,LTTYCT        ;ADJUST OUTPUT COUNT
        SUB      #3,R2
        SUB      R1,R2            ;POINT R2 BEFORE BAD WORD
        MOVB     (R2),-1(R2)      ;MOVE THAT CHAR 1 BACK
        MOVB     #'?',(R2)        ;INSERT A ?
LSNLP1: CMP      R2,R0            ;ARE WE AT LINE START ?
        BEQ      LSNTG2          ;YES - GET READY TO TYPE
        CMPB     #11,-1(R2)       ;IS THIS A TAB ?
        BNE      LSNTG1          ;NO - CONTINUE
        DEC      R2              ;YES - SKIP IT
        INC      LTTYCT          ;AND BUMP OUTPUT COUNT
LSNTG1: MOVB     #40,-(R2)        ;INSERT A BLANK
        INC      LTTYCT          ;BUMP OUTPUT COUNT
        BR       LSNLP1         ;CHECK IF DONE
LSNTG2: MOV      #'ER',(R2)+      ;STORE 'ER'
        MOVB     #'*',(R2)        ;THEN ANOTHER *
        JSR      R5,CTYPE        ;TYPE THE LINE
        .WORD    420
        .WORD    LINBUF-1-.
LTTYCT: .WORD    1
        INCB     LERROR          ;SET ERROR FLAG
        RTS      R5              ;RETURN TO CALLING PGM.

```

.SBTTL DISPLAY COMMAND HANDLER

```

*****
THIS SUBROUTINE HANDLES THE DISPLAY N
AND DISPLAY N CODE COMMANDS. THE OUTPUT
IS ON THE DISPLAY DEVICE AND CONSISTS
OF A LISTING OF THE USERS STATEMENTS
ALONG WITH THE MACHINE CODE GENERATED
BY THE COMPILER (IF REQUESTED).

CALLING SEQUENCE      JSR      P5,DISP (DISPCD)

ENTRY   R3 POINTS TO PROGRAM TABLE
*****

```

6675
6676
6677
6678
6679
6680
6681
6682
6683
6684
6685
6686
6687
6688
6689
6690
6691
6692
6693
6694
6695
6696
6697
6698
6699
6700
6701
6702
6703
6704
6705
6706
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716
6717
6718
6719
6720
6721
6722
6723
6724
6725
6726
6727
6728
6729
6730

```

024470' 105067 000164
024474' 000402
024476' 105267 000156
024502' 010367 177610
024506' 010346
024510' 010446
024512' 016367 000022 177574
      001
      000
024520' 004567 176650
024524' 010067 000010
024530' 010204
      001
      000
024532' 004567 011042
024536' 007524
024540' 000000
024542' 105767 000112
024546' 001006
024550' 105767 177534
024554' 001761
024556' 012604
024560' 012603
024562' 000205
      001
024564' 005704
      000
024566' 001770
024570' 105767 177514
024574' 001025
024576' 016702 177512
024602' 016203 000004
024606' 001760

```

```

DISP:  CLRB      LSTOBJ      ;CLEAR LIST OBJ FLAG
        BR       LDISVR     ;THEN SAVE REGISTERS
DISPCD: INCB     LSTOBJ     ;SET LIST OBJ FLAG
LDISVR: MOV      R3,LUPPTP  ;SAVE PTBL POINTER
        MOV      R3,-(SP)
        MOV      R4,-(SP)  ;SAVE R3 AND R4
        MOV      PSRCST(R3),LUPKPT ;INIT UNPACK POINTER
        .IF DF MM
        BIS      #P4CONS,LUPKPT ;SET PAR4 SELECT BITS
        .ENDC
LDILP1: JSR      R5,UNPACK  ;UNPACK ONE LINE
        MOV      R0,LDITBC
        MOV      R2,R4     ;SAVE PHYSICAL ADDRESS
        .IF DF MM
        BIS      #P4CONS,R4  ;SET PAR4 SELECT BITS
        .ENDC
        JSR      R5,CLIST  ;PRINT IT
        .WORD   LINBUF-
LDITBC: .WORD   0
        TSTB    LSTOBJ     ;TEST LIST OBJ FLAG
        BNE     LDIGPA    ;BRANCH IF SET
LDITEF: TSTB    LENDFL    ;TEST END FLAG
        BEQ     LDILP1   ;LOOP IF NOT SET
        MOV     (SP)+,R4
        MOV     (SP)+,R3  ;RESTORE REGISTERS
        RTS     R5       ;RETURN TO CALLING PGM
        .IF DF MM
        .IFF
LDIGPA: TST     R4       ;IF PHYSICAL ADDR ZERO
        .IFT
LDIGPA: CMP     R4,#P4CONS ;IF PHYSICAL ADR ZERO
        .ENDC
        BEQ     LDITEF   ;GO CHECK END FLAG
        TSTB    LENDFL   ;DETOUR
        BNE     LDIREF   ;IF END FLAG SET
        MOV     LUPKPT,R2
        MOV     4(R2),R3 ;GET NEXT PHYSICAL ADDR
        BEQ     LDITEF   ;IF ZERO DONT CONTINUE

```

```

6731          001          .IF DF MM
6732          .IFT
6733          LDILCD: MOV      2#KPAR4,(PC)+      ;GET 22 BIT ADDRESS
6734          LPARSV: .WORD      0
6735          JSR      R5,G22BAD      ;OF OBJ CODE WORD
6736          .WORD      LPARSV-
6737          MOV      R4,-(SP)
6738          SUB      #P4CONS,(SP)
6739          ADD      (SP)+,R0      ;PUT PHYSICAL ADR IN
6740          ADC      R2      ;R2 AND R0
6741          JSR      R5,BINASB      ;CONVERT TO ASCII
6742          .WORD      LDI188-
6743          MOV      R4,R0      ;GET VIRTUAL ADR OF OBJ CODE WD
6744          BIC      #P4CONS,R0      ;RESET PAGE SELECT BITS
6745          .IFF
6746          LDILCD: MOV      R4,R0      ;GET ADR OF OBJ CODE WORD
6747          .ENDC
6748          JSR      R5,BINASC      ;CONVERT OBJ ADR
6749          .WORD      LDIADR-      ;TO ASCII
6750          MOV      (R4)+,R0
6751          JSR      R5,BINASC      ;CONVERT CONTENTS TO ASCII
6752          .WORD      LDI0BJ-
6753          JSR      R5,CLIST      ;PRINT OUT ABOVE
6754          .WORD      LDIMSG-
6755          .IF DF MM
6756          .IFF
6757          .WORD      17.
6758          .IFT
6759          .WORD      29.
6760          .ENDC
6761          MOV      R4,R0
6762          .IF DF MM
6763          BIC      #P4CONS,R0      ;CLEAR PAR4 SELECT BITS
6764          .ENDC
6765          CMP      R0,R3      ;CURRENT ADDR >=NEXT
6766          BLO      LDILCD      ;NO - LIST MORE CODE
6767          BR      LDITEF      ;YES - CHECK FOR MORE
6768          LDIREF: MOV      R4,R3      ;GET CURRENT PHYSICAL ADDR
6769          .IF DF MM
6770          BIC      #P4CONS,R3      ;RESET PAR4 SELECT BITS
6771          .ENDC
6772          ADD      #4,R3      ;ADD 4 STORE AS NEXT
6773          BR      LDILCD      ;GO PRINT CODE
6774
6775
6776          LSTOBJ: .BYTE      0      ;LIST OBJECT FLAG
6777
6778          LDIMSG: .ASCII      <011>
6779          .IF DF MM
6780          .IFF
6781          LDIADR: .ASCII      /XXXXXX= /
6782
6783          LDI188: .ASCII      /XXXXXXXX (/
6784          LDIADR: .ASCII      /XXXXXX) = /
6785          .ENDC

```

M11

MAINDEC-11-DTMGA-C MPG INTERPRETER MACY11 27(732) 24-SEP-76 13:54 PAGE 34-2
DTMGAC.P11 DISPLAY COMMAND HANDLER

SEQ 0143

6786 024672' 054130 054130 054130 LDI0BJ: .ASCII /XXXXXX/<015><012>
024700' 005015
6787 .EVEN

.SBTTL SAVE COMMAND HANDLER

```

*****
THIS SUBROUTINE HANDLES THE SAVE COMMAND.
IT DETERMINES THE NUMBER OF BLOCKS REQUIRED
AND OPENS THE FILE. IT UNPACKS THE INTERNAL
PACKED CODE BACK TO ITS ORIGINAL FORM AND
STORES IT ON THE LIBRARY DEVICE.

CALLING SEQUENCE JSR R5,SAVE

ENTRY R3 POINTS TO PROGRAM TABLE
*****

```

6789
6790
6791
6792
6793
6794
6795
6796
6797
6798
6799
6800
6801
6802
6803
6804
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816
6817
6818
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828
6829
6830
6831
6832
6833
6834
6835
6836
6837
6838
6839
6840
6841
6842
6843
6844

024702' 005067 000402
024706' 010367 177404
024712' 016367 000022 177374
001
000
024720' 005067 000366
024724' 012767 000754 000354
024732' 016704 153052
024736' 062704 000010
024742' 005067 000174
024746' 005767 000336
024752' 001003
024754' 005267 000330
024760' 000402
024762' 005067 000322
024766' 004567 176402
024772' 010702
024774' 062702 007266
025000' 160067 000302
025004' 100461
025006' 112224
025010' 005300
025012' 001375
025014' 105767 177270
025020' 001762
025022' 010702
025024' 062702 007236
025030' 012722 047504
025034' 012722 042516
025040' 016722 007364
025044' 012700 000006
025050' 160002
025052' 160067 000230
025056' 100434
025060' 112224
025062' 005300
025064' 001375

```

SAVE: CLR LSVFPS ;CLEAR FIRST PASS SW
MOV R3,LUPPTP ;SAVE P TBL POINTER
LSVPS2: MOV PSRCST(R3),LUPKPT ;INIT UNPACK POINTER
.IF DF MM
BIS #P4CONS,LUPKPT ;SET PAR4 SELECT BITS
.ENDC
CLR LSVRTL ;CLEAR RUNNING TOTAL
MOV #492,LSVBCT ;SET BYTES REMAIN=492
MOV BUF,R4 ;GET BLOCK BUF POINTER
ADD #8,R4 ;SKIP FIRST 8 BYTES
CLR LSVBNM ;SET NBR BLKS=0
TST LSVFPS ;TEST FIRST PASS SW
BNE LSVRSW ;IF SET-RESET IT
INC LSVFPS ;OTHERWISE SET IT
BR LSVUPL
LSVRSW: CLR LSVFPS ;RESET FIRST PASS SW
LSVUPL: JSR R5,UNPACK ;UNPACK ONE LINE
MOV PC,R2
ADD #LINBUF-,R2 ;POINTER R2 TO LINBUF

SUB R0,LSVBCT ;SUBTRACT CHAR COUNT
BMI LSVWFT ;BRANCH IF IT WONT FIT
LSVMLB: MOVB (R2)+,(R4)+ ;MOVE LINE TO BLOCK
DEC R0
BNE LSVMLB
LSVTEN: TSTB LENDFL ;TEST END FLAG
BEQ LSVUPL ;IF RESET UNPACK MORE
MOV PC,R2
ADD #LINBUF-,R2
MOV #DO,(R2)+ ;MOVE 'DONE'
MOV #NE,(R2)+ ;TO LINE BUFFER
MOV LCRLF,(R2)+
MOV #6,R0
SUB R0,R2
SUB R0,LSVBCT ;SUBTRACT CHAR CNT
BMI LSVWFT ;BRANCH IF IT WONT FIT
LSVMOB: MOVB (R2)+,(R4)+ ;MOVE 'DONE' TO BLOCK
DEC R0
BNE LSVMOB

```

6845	025066'	005767	000216	TST	LSVFPS	;TEST FIRST PASS SW
6846	025072'	001017		BNE	LSVBBC	;IF SET BUMP BLK CNT
6847	025074'	016704	152710	MOV	BUF,R4	;GET BLOCK BUF POINTER
6848	025100'	062704	000004	ADD	#4,R4	;BUMP IT BY 4
6849	025104'	012701	000762	MOV	#498,R1	
6850	025110'	166701	000172	SUB	LSVBCT,R1	;CALC BYTES + 6
6851	025114'	010124		MOV	R1,(R4)+	;STORE IN BUFFER
6852	025116'	016714	000170	MOV	LSVRTL,(R4)	;ALSO STORE RUNNING TOTAL
6853	025122'	004577	152670	JSR	RS,@CLOSE	;IF NOT CLOSE FILE
6854	025126'	000002		.WORD	LSVEXT-	
6855	025130'	000205		LSVEXT: RTS	RS	;RETURN TO CONTROL ROUT.
6856	025132'	005267	000004	LSVBBC: INC	LSVBNM	;BUMP BLOCK COUNTER
6857	025136'	004577	152660	JSR	RS,@CREATE	;OPEN A FILE
6858	025142'	000000		LSVBNM: .WORD	0	;WITH THIS NBR BLKS
6859	025144'	177764		.WORD	LSVEXT-	
6860	025146'	000661		BR	LSVPS2	;GO TO SECOND PASS
6861	025150'	060067	000132	LSVWFT: ADD	RO,LSVBCT	;ADD CHAR COUNT BACK
6862	025154'	005767	000126	TST	LSVBCT	;REMAINING BYTE COUNT = 0 ?
6863	025160'	001405		BEQ	LSVBCZ	;YES GOTO LSVBCZ
6864	025162'	112224		LSVMOC: MOVB	(R2)+,(R4)+	;MOVE ONE CHARACTER
6865	025164'	005300		DEC	RO	;AT A TIME
6866	025166'	005367	000114	DEC	LSVBCT	
6867	025172'	001373		BNE	LSVMOC	
6868	025174'	005767	000110	LSVBCZ: TST	LSVFPS	;TEST FIRST PASS SW
6869	025200'	001022		BNE	LSVDWR	;DONT WRITE IF SET
6870	025202'	016704	152602	MOV	BUF,R4	;GET BLOCK BUF POINTER
6871	025206'	062704	000004	ADD	#4,R4	;BUMP IT BY 4
6872	025212'	012701	000762	MOV	#498,R1	
6873	025216'	166701	000064	SUB	LSVBCT,R1	;CALC BYTES + 6
6874	025222'	010124		MOV	R1,(R4)+	;STORE IN BUFFER
6875	025224'	016714	000062	MOV	LSVRTL,(R4)	;ALSO STORE RUNNING TOTAL
6876	025230'	162701	000006	SUB	#6,R1	
6877	025234'	060167	000052	ADD	R1,LSVRTL	;UPDATE RUNNING TOTAL
6878	025240'	004577	152612	JSR	RS,@PUT	;WRITE ONE BLOCK
6879	025244'	177664		.WORD	LSVEXT-	
6880	025246'	005267	177670	LSVDWR: INC	LSVBNM	;BUMP BLOCK COUNTER
6881	025252'	016704	152532	MOV	BUF,R4	;GET BLOCK BUF POINTER
6882	025256'	012767	000754	MOV	#492,LSVBCT	;SET BYTES REMAIN=492
6883	025264'	062704	000010	ADD	#8,R4	;SKIP FIRST 8 BYTES
6884	025270'	112224		LSVMRL: MOVB	(R2)+,(R4)+	;MOVE REST OF LINE
6885	025272'	005367	000010	DEC	LSVBCT	
6886	025276'	005300		DEC	RO	
6887	025300'	001373		BNE	LSVMRL	
6888	025302'	000167	177506	JMP	LSVTEN	;THEN GO TEST END FLAG
6889						
6890	025306'	000000		LSVBCT: .WORD	0	;REMAINING BYTE COUNT
6891	025310'	000000		LSVFPS: .WORD	0	;FIRST PASS SWITCH
6892	025312'	000000		LSVRTL: .WORD	0	;RUNNING TOT - BYTES WRITTEN

.SBTTL BIT PACKER SUBROUTINE

```

*****
THIS SUBROUTINE INTERPRETS THE USER STATEMENT
WORDS N AND N THRU N OR N AND N -N AND SETS
THE APPROPRIATE BITS IN A 16 BIT WORD.

CALLING SEQUENCE      JSR      R5,BTPACK
                       .WORD   ERRATN-.

ENTRY  RD CONTAINS THE NUMBER OF USER WORDS IN THE
       STRING (EXAMPLE -7 FOR N & N & N - N)
       LSCNPT POINTS TO THE WORD BEFORE THE STRING

EXIT   RD CONTAINS THE 16 BIT WORD WITH BITS SET
*****

```

```

6894
6895
6896
6897
6898
6899
6900
6901
6902
6903
6904
6905
6906
6907
6908
6909
6910
6911
6912
6913
6914
6915 025314' 010067 000460
6916 025320' 012767 000001 000450
6917 025326' 004567 001054
6918 025332' 004567 015012
6919 025336' 000434
6920 025340' 004567 014552
6921 025344' 000426
6922 025346' 010067 000430
6923 025352' 010067 000430
6924 025356' 005367 000416
6925 025362' 005067 000416
6926 025366' 005767 000410
6927 025372' 001410
6928 025374' 006367 000376
6929 025400' 001002
6930 025402' 000167 000364
6931 025406' 005367 000370
6932 025412' 001370
6933 025414' 056767 000356 000362
6934 025422' 005767 000352
6935 025426' 001555
6936 025430' 004567 000752
6937 025434' 005301
6938 025436' 160100
6939 025440' 010702
6940 025442' 062702 000346
6941 025446' 005367 000326
6942 025452' 001547
6943 025454' 012701 000004
6944 025460' 121012
6945 025462' 001003
6946 025464' 062702 000002
6947 025470' 011207
6948 025472' 062702 000004
6949 025476' 005301

```

```

BTPACK: MOV      R0,LNBRWD      ;SAVE NBR OF WORDS
        MOV      #1,LBTPNT     ;SET BIT POINTER TO BIT 0
        JSR      R5,SCAN       ;FIND FIRST WORD
        JSR      R5,ASDECO     ;CONVERT TO BCD
        .WORD   LBPERX-
        JSR      R5,DECBIN     ;CONVERT BCD TO BINARY
        .WORD   LBPERX-
        MOV      R0,LBTNBR     ;STORE RESULT AS BIT NBR
        MOV      R0,LBTNSV     ;AND SAVE FOR LATER
        DEC      LNBRWD        ;DECREMENT NUMBER OF WORDS
        CLR      LBPRWD        ;SET RESULT WORD TO ZEROES
        TST      LBTNBR        ;IF REQUESTED BIT NBR 0
        BEQ      LBPBS         ;GO SET THAT BIT
        ASL      LBTPNT        ;SHIFT BIT POINTER LEFT
        BNE      LBPTG1        ;ERROR IF SHIFT >16
        JMP      LBPERX
LBPTG1: DEC      LBTNBR        ;DECREMENT BIT NUMBER
        BNE      LBPLP1        ;SHIFT TO CORRECT POSITION
LBPBS:  BIS      LBTNT,LBPRWD  ;SET A BIT
LBPMOR: TST      LNBRWD
        BEQ      LBPEXT        ;EXIT IF ALL WORDS PROCESSED
        JSR      R5,SCAN
        DEC      R1
        SUB      R1,R0         ;MOVE POINTER TO WORD START
        MOV      PC,R2
        ADD      #LBPTB-. ,R2  ;POINT R2 AT LOOKUP TABLE
        DEC      LNBRWD        ;DECREMENT NUMBER OF WORDS
        BEQ      LBPERX        ;ERROR IF EVEN # WORDS
        MOV      #4,R1
        LBPLP2: CMPB      (R0),(R2) ;IS WORD A TABLE ENTRY
        BNE      LBPNF         ;BRANCH IF NO
        ADD      #2,R2
        MOV      (R2),PC       ;IF YES TAKE BRANCH
LBPNF:  ADD      #4,R2         ;BUMP TABLE POINTER
        DEC      R1            ;END OF TABLE

```

6950	025500'	001367			BNE	LBPLP2		;NO-CHECK NEXT ENTRY
6951	025502'	000533			BR	LBPERX		;YES-TAKE ERROR EXIT
6952	025504'	004567	000676		LBPRND:	JSR	RS,SCAN	;FIND NEXT WORD
6953	025510'	005367	000264			DEC	LNBRWD	;DECREMENT NBR OF WORDS
6954	025514'	004567	014630			JSR	RS,ASDECO	;CONVERT WORD (BIT*) TO BCD
6955	025520'	000252				.WORD	LBPERX-	
6956	025522'	004567	014370			JSR	RS,DECBIN	;CONVERT BCD TO BINARY
6957	025526'	000244				.WORD	LBPERX-	
6958	025530'	010067	000246			MOV	RO,LBTNBR	;STORE RESULT AS BIT NBR
6959	025534'	010067	000246			MOV	RO,LBTNSV	;AND SAVE FOR LATER
6960	025540'	012767	000001	000230		MOV	#1,LBTPNT	;SET BIT POINTER TO BIT 0
6961	025546'	005767	000230			TST	LBTNBR	;IF BIT NUMBER 0
6962	025552'	001406				BEQ	LBPSB0	;GO SET IT
6963	025554'	006367	000216		LBPLP3:	ASL	LBTPNT	;SHIFT BIT POINTER LEFT
6964	025560'	001504				BEQ	LBPERX	;ERROR IF SHIFT >16
6965	025562'	005367	000214			DEC	LBTNBR	;DECREMENT BIT NBR
6966	025566'	001372				BNE	LBPLP3	;SHIFT TO CORRECT POSITION
6967	025570'	056767	000202	000206	LBPSB0:	BIS	LBTPNT,LBPRWD	;SET A BIT
6968	025576'	005767	000176			TST	LNBRWD	;TEST IF NBR OF WORDS =0
6969	025602'	001467				BEQ	LBPEXT	;EXIT IF YES
6970	025604'	000706				BR	LBPMOR	;IF NO TEST MORE WORDS
6971	025606'	004567	000574		LBPTRU:	JSR	RS,SCAN	;FIND NEXT WORD
6972	025612'	005367	000162			DEC	LNBRWD	;DECREMENT NBR OF WORDS
6973	025616'	004567	014526			JSR	RS,ASDECO	;CONVERT WORD (BIT*) TO BCD
6974	025622'	000150				.WORD	LBPERX-	
6975	025624'	004567	014266			JSR	RS,DECBIN	;CONVERT BCD TO BINARY
6976	025630'	000142				.WORD	LBPERX-	
6977	025632'	016701	000150			MOV	LBTNSV,R1	;GET PRIOR BIT*
6978	025636'	010067	000144			MOV	RO,LBTNSV	;SAVE LATEST
6979	025642'	020001				CMP	RO,R1	
6980	025644'	002403				BLT	LBPPGL	;IF LATEST >PRIOR
6981	025646'	010067	000130			MOV	RO,LBTNBR	;STORE LATEST
6982	025652'	000403				BR	LBPLGP	;IF PRIOR >LATEST
6983	025654'	010167	000122		LBPPGL:	MOV	R1,LBTNBR	;STORE PRIOR
6984	025660'	010001				MOV	RO,R1	;LATEST TO R1
6985	025662'	012767	000001	000106	LBPLGP:	MOV	#1,LBTPNT	;SET BIT POINTER TO BIT 0
6986	025670'	005701				TST	R1	;IF BIT * IS NOT 0
6987	025672'	001001				BNE	LBPLP4	;CONTINUE
6988	025674'	000407				BR	LBPTG2	;OTHERWISE SET BIT 0
6989	025676'	006367	000074		LBPLP4:	ASL	LBTPNT	;SHIFT BIT POINTER LEFT
6990	025702'	001433				BEQ	LBPERX	;ERROR IF SHIFT >16
6991	025704'	005367	000072			DEC	LBTNBR	;DECREMENT LARGER BIT*
6992	025710'	005301				DEC	R1	;DECREMENT SMALLER BIT*
6993	025712'	001371				BNE	LBPLP4	;SHIFT TO CORRECT POSITION
6994	025714'	020167	000062		LB ^r :	CMP	R1,LBTNBR	;CHECK IF PRIOR =LATEST
6995	025720'	001414				BEQ	LBPST1	;GO SET ONE BIT
6996	025722'	056767	000050	000054	LBPLP5:	BIS	LBTPNT,LBPRWD	;SET A BIT
6997	025730'	006367	000042			ASL	LBTPNT	;SHIFT BIT POINTER LEFT
6998	025734'	001001				BNE	LBPTG3	
6999	025736'	103015				BCC	LBPERX	;EXIT IF SHIFT > 16
7000	025740'	005201			LBPTG3:	INC	R1	;INCREMENT SMALLER NBR
7001	025742'	020167	000034			CMP	R1,LBTNBR	;COMPARE TO LARGER
7002	025746'	101765				BLOS	LBPLP5	;LOOP IF < OR =
7003	025750'	000624				BR	LBPMOR	
7004	025752'	056767	000020	000024	LBPST1:	BIS	LBTPNT,LBPRWD	;SET ONE BIT
7005	025760'	000620				BR	LBPMOR	

7006	025762'	016700	000016	LBPEXT: MUV	LBPRWD, R0	; RESULT TO R0
7007	025766'	005725		TST	(R5)+	; BYPASS ERR RETURN ADR
7008	025770'	000205		RTS	R5	; RETURN TO CALLING PGM
7009						
7010	025772'	061505		LBPERX: ADD	(R5), R5	; CALCULATE ERROR RETURN
7011	025774'	000205		RTS	R5	; AND TAKE IT
7012						
7013	025776'	000000		LBPNT: .WORD	0	; BIT POINTER
7014	026000'	000000		LNBRWD: .WORD	0	; NUMBER OF WORDS
7015	026002'	000000		LBTNBR: .WORD	0	; BIT NUMBER
7016	026004'	000000		LBPRWD: .WORD	0	; BIT PACKER RESULT
7017	026006'	000000		LBTNSV: .WORD	0	; BIT NUMBER SAVE
7018						
7019	026010'	000101		LBPVTB: .ASCIZ	/A/	; AND-THRU LOOKUP TABLE
7020	026012'	025504'		LBPR1: .WORD	LBPRWD	
7021	026014'	000046		.ASCIZ	/8/	
7022	026016'	025504'		LBPR2: .WORD	LBPRWD	
7023	026020'	000124		.ASCIZ	/T/	
7024	026022'	025606'		LBPR3: .WORD	LBPRWD	
7025	026024'	000055		.ASCIZ	/-/	
7026	026026'	025606'		LBPR4: .WORD	LBPRWD	

.SBTTL BIT UNPACKER SUBROUTINE

THIS SUBROUTINE CONVERTS A 16 BIT
BINARY NUMBER INTO THE FORM OF
N THRU N AND N AS AN ASCII CHARACTER STRING

CALLING SEQUENCE JSR RS,BTUPAK

ENTRY R0 CONTAINS THE 16 BIT NUMBER
R1 POINTS TO WHERE ASCII STRING DESIRED

EXIT R1 POINTS TWO SPACES BEYOND LAST
WORD GENERATED.

7028
7029
7030
7031
7032
7033
7034
7035
7036
7037
7038
7039
7040
7041
7042
7043
7044
7045
7046
7047
7048
7049
7050
7051
7052
7053
7054
7055
7056
7057
7058
7059
7060
7061
7062
7063
7064
7065
7066
7067
7068
7069
7070
7071
7072
7073
7074
7075
7076
7077
7078
7079
7080
7081
7082
7083

026030' 010067 000310
026034' 012767 000001 000262
026042' 005067 000260
026046' 036767 000252 000270
026054' 001011
026056' 005267 000244
026062' 006367 000236
026066' 100367
026070' 036767 000230 000246
026076' 001504
026100' 016702 000222
026104' 006302
026106' 060702
026110' 062702 000236
026114' 112221
026116' 111221
026120' 022767 000017 000200
026126' 001470
026130' 006367 000170
026134' 005267 000166
026140' 036767 000160 000176
026146' 001433
026150' 012700 000006
026154' 010702
026156' 062702 000152
026162' 112221
026164' 005300
026166' 001375
026170' 006367 000130
026174' 005267 000126
026200' 036767 000120 000136
026206' 001405
026210' 022767 000017 000110
026216' 101364
026220' 000727
026222' 000241

BTUPAK: MOV R0,LBUWRD
MOV #1,LBUBPT
CLR LBUNBR
LBULP1: BIT LBUBPT,LBUWRD
BNE LBUSAS
INC LBUNBR
ASL LBUBPT
BPL LBULP1
BIT LBUBPT,LBUWRD
BEQ LBUEXT
LBUSAS: MOV LBUNBR,R2
ASL R2
ADD PC,R2
ADD #LBUAST-,R2
MOVB (R2),(R1)+
MCVB (R2),(R1)+
CMP #15,LBUNBR
BEQ LBUEXT
ASL LBUBPT
INC LBUNBR
BIT LBUBPT,LBUWRD
BEQ LBUEXT
MOV #6,R0
MOV PC,R2
ADD #LBUTRU-,R2
LBULP2: MOVB (R2),(R1)+
DEC R0
BNE LBULP2
LBULP3: ASL LBUBPT
INC LBUNBR
BIT LBUBPT,LBUWRD
BEQ LBUBKU
CMP #15,LBUNBR
BHI LBULP3
BR LBUSAS
LBUBKU: CLC

;SAVE BINARY NUMBER
;SET BIT 0 OF UNPACK PTR
;SET BIT NBR TO 0
;TEST IF BIT SET
;BRANCH IF IT IS
;INCR BIT NBR
;SHIFT BIT POINTER LEFT
;REPEAT IF NOT BIT 15
;TEST IF BIT 15 SET
;EXIT IF NOT
;GET BIT NUMBER
;DOUBLE IT
;CALCULATE
;LOOKUP TABLE POSITION
;PRINT FIRST DIGIT
;THEN SECOND
;TEST IF LAST BIT # 15
;EXIT IF YES
;ADVANCE BIT POINTER
;INCREMENT BIT NUMBER
;TEST IF BIT SET
;BRANCH IF NOT
;STORE WORD 'THRU'
;ADVANCE BIT POINTER
;INCREMENT BIT NUMBER
;TEST IF BIT SET
;BRANCH IF NOT
;TEST IF BIT 15
;LOOP IF NOT
;STORE 15 IF IT IS

DTMGAC.P11 BIT UNPACKER SUBROUTINE

```

7084 026224' 006067 000074          ROR      LBUBPT          ;BACK UP POINTER
7085 026230' 005367 000072          DEC      LBUNBR          ;DECR BIT #
7086 026234' 000721          BR       LBUSAS          ;THEN STORE NUMBER
7087 026236' 022767 000017 0C0062  LBUFBT:  CMP      #15, LBUNBR
7088 026244' 001421          BEQ     LBUEXT          ;EXIT IF BIT # = 15
7089 026246' 006367 000052          ASL     LBUBPT          ;ADVANCE BIT POINTER
7090 026252' 005267 000050          INC     LBUNBR
7091 026256' 036767 000042 000060  BIT     LBUBPT, LBUWRD  ;TEST IF BIT SET
7092 026264' 001764          BEQ     LBUFBT          ;LOOP IF NOT
7093 026266' 012700 000005          MOV     #5, R0
7094 026272' 010702          MOV     PC, R2
7095 026274' 062702 000042          ADD     #LBUAND-, R2
7096 026300' 112221          LBU4P4:  MOVB   (R2)+, (R1)+  ;STORE WORD "AND"
7097 026302' 005300          DEC     R0
7098 026304' 001375          BNE     LBU4P4
7099 026306' 000674          BR      LBUSAS          ;STORE BIT NUMBER
7100 026310' 010702          LBUEXT:  MOV     PC, R2
7101 026312' 062702 000030          ADD     #LBUSPC-, R2
7102 026316' 112221          MOVB   (R2)+, (R1)+  ;STORE TWO SPACES
7103 026320' 111211          MOVB   (R2), (R1)
7104 026322' 000205          RTS     R5          ;RETURN TO CALLING PGM
7105
7106 026324' 000000          LBUBPT:  .WORD  0          ;UNPACK BIT POINTER
7107 026326' 000000          LBUNBR:  .WORD  0          ;UNPACK BIT NUMBER
7108 026330' 052040 051110 020125  LBUTRU:  .ASCII  / THRU /
7109 026336' 040440 042116          LBUAND:  .ASCII  / AND /
7110 026342' 020040          LBUSPC:  .ASCII  / /
7111 026344' 000000          LBUWRD:  .WORD  0
7112 026346' 030040          LBUAST:  .ASCII  / 0 /
7113 026350' 030440          .ASCII  / 1 /
7114 026352' 031040          .ASCII  / 2 /
7115 026354' 031440          .ASCII  / 3 /
7116 026356' 032040          .ASCII  / 4 /
7117 026360' 032440          .ASCII  / 5 /
7118 026362' 033040          .ASCII  / 6 /
7119 026364' 033440          .ASCII  / 7 /
7120 026366' 034040          .ASCII  / 8 /
7121 026370' 034440          .ASCII  / 9 /
7122 026372' 030061          .ASCII  /10 /
7123 026374' 030461          .ASCII  /11 /
7124 026376' 031061          .ASCII  /12 /
7125 026400' 031461          .ASCII  /13 /
7126 026402' 032061          .ASCII  /14 /
7127 026404' 032461          .ASCII  /15 /

```

.SBTTL SCAN SUBROUTINE

THIS SUBROUTINE SCANS A LINE OF ASCII CHARACTERS AND EXITS WHEN THE NEXT WORD OR SPECIAL SYMBOL IS FOUND. SCANNING BEGINS AT THE ADDRESS STORED IN LSCNPT. ANY INITIAL BLANKS ARE IGNORED. WHEN THE FIRST NON-BLANK CHARACTER IS ENCOUNTERED SCAN BEGINS COUNTING. WHEN THE NEXT BLANK, DASH, COMMA, TAB, AMPERSAND OR CARRIAGE RETURN IS ENCOUNTERED, SCAN EXITS. R0 AND LSCNPT THEN CONTAIN THE ADDRESS OF THE LAST CHARACTER OF THE WORD, AND R1 CONTAINS A COUNT OF THE NUMBER OF CHARACTERS.

CALLING SEQUENCE JSR R5,SCAN

7129			
7130			
7131			
7132			
7133			
7134			
7135			
7136			
7137			
7139			
7139			
7140			
7141			
7142			
7143			
7144			
7145			
7146			
7147			
7148			
7149			
7150			
7151			
7152	026406'	016700	000212
7153	026412'	005200	
7154	026414'	005001	
7155	026416'	112002	
7156	026420'	122702	000015
7157	026424'	001454	
7158	026426'	122702	000055
7159	026432'	001451	
7160	026434'	122702	000054
7161	026440'	001446	
7162	026442'	122702	000052
7163	026446'	001443	
7164	026450'	122702	000043
7165	026454'	001440	
7166	026456'	122702	000046
7167	026462'	001435	
7168	026464'	122702	000040
7169	026470'	001752	
7170	026472'	122702	000011
7171	026476'	001747	
7172	026500'	005201	
7173	026502'	112002	
7174	026504'	122702	000015
7175	026510'	001420	
7176	026512'	122702	000055
7177	026516'	001415	
7178	026520'	122702	000054
7179	026524'	001412	
7180	026526'	122702	000046
7181	026532'	001407	
7182	026534'	122702	000040
7183	026540'	001415	
7184	026542'	122702	000011

```

SCAN:  MOV    LSCNPT,R0      ;GET SCAN POINTER
        INC    R0            ;ADVANCE IT BY ONE
        CLR    R1            ;CLEAR CHARACTER COUNT
LSKBL:  MOVB   (R0)+,R2      ;GET CHAR, ADVANCE POINTER
        CMPB  #15,R2        ;CARRIAGE RETURN ?
        BEQ   LSCOUT        ;YES - GET OUT
        CMPB  #55,R2        ;HYPHEN ?
        BEQ   LSCOUT        ;YES - GET OUT
        CMPB  #54,R2        ;COMMA ?
        BEQ   LSCOUT        ;YES - GET OUT
        CMPB  #'*,R2        ;ASTERISK ?
        BEQ   LSCOUT        ;YES - GET OUT
        CMPB  #'#,R2        ;NBR SGN ?
        BEQ   LSCOUT        ;YES - GET OUT
        CMPB  #46,R2        ;AMPERSAND ?
        BEQ   LSCOUT        ;YES - GET OUT
        CMPB  #40,R2        ;SPACE ?
        BEQ   LSKBL         ;YES - CHECK NEXT CHARACTER
        CMPB  #11,R2        ;TAB ?
        BEQ   LSKBL         ;YES - CHECK NEXT CHARACTER
LCTCHR: INC    R1            ;INCREMENT CHAR. COUNT
        MOVB  (R0)+,R2      ;GET CHAR, ADVANCE POINTER
        CMPB  #15,R2        ;CARRIAGE RETURN ?
        BEQ   LSBKUP        ;YES - DONE
        CMPB  #55,R2        ;HYPHEN ?
        BEQ   LSBKUP        ;YES - DONE
        CMPB  #54,R2        ;COMMA ?
        BEQ   LSBKUP        ;YES - DONE
        CMPB  #46,R2        ;AMPERSAND ?
        BEQ   LSBKUP        ;YES - DONE
        CMPB  #40,R2        ;SPACE ?
        BEQ   LSCCKGO       ;YES-CHECK FOR GO
        CMPB  #11,R2        ;TAB ?

```



```

7185 026546' 001401      BEQ      LSBKUP      ; YES - DONE
7186 026550' 000753      BR       LCTCHR     ; NO-GET AND COUNT NEXT CHAR
7187 026552' 005300      LSBKUP: DEC      RO      ; BACK UP SCAN POINTER
7188 026554' 000401      BR       LSCNEX     ; EXIT
7189 026556' 005201      LSCOUT: INC      R1     ; INCREMENT CHARACTER COUNT
7190 026560' 005300      LSCNEX: DEC      RO      ; BACK UP SCAN POINTER
7191 026562' 010067      000036      MOV      RO,LSCNPT  ; STORE SCAN POINTER
7192 026566' 010167      000034      MOV      R1,LSCNCT  ; SAVE SCAN CHAR COUNT
7193 026572' 000205      RTS       R5        ; RETURN TO CALLING PROGRAM
7194 026574' 022701      000002      LSCKGO: CMP      #2,R1 ; CHAR COUNT = 2 ?
7195 026600' 001364      BNE      LSBKUP     ; NO-EXIT
7196 026602' 122760      000107      177775      CMPB     #'G,-3(RO)  ; FIRST CHAR 'G'
7197 026610' 001360      BNE      LSBKUP     ; NO-EXIT
7198 026612' 122760      000117      177776      CMPB     #'O,-2(RO)  ; SECOND CHAR 'O'
7199 026620' 001354      BNE      LSBKUP     ; NO-EXIT
7200 026622' 000726      BR       LCTCHR     ; YES-GET AND COUNT NEXT CHAR
7201
7202
7203 026624' 000000      LSCNPT: .WORD    0      ; SCAN POINTER
7204 026626' 000000      LSCNCT: .WORD    0      ; SCAN CHARACTER COUNT
7205 026630' 000000      LSPTSU: .WORD    0      ; SCAN POINTER SAVE

```

.SBTTL TAG LOOKUP SUBROUTINE

```

*****
THIS SUBROUTINE SEARCHES THE COMPILER SYMBOL
TABLES FOR AN ENTRY CORRESPONDING TO A TAG
ENTERED BY THE USER. IF FOUND, ITS BINARY
EQUIVALENT IS PASSED BACK IN R2, OTHERWISE
THE ERROR RETURN IS TAKEN

CALLING SEQUENCE-      JSR      R5, TAGLUP
                        .WORD    ERATN-

ENTRY-  R0  POINTS TO THE FIRST CHAR. OF THE TAG
        R1  CONTAINS CHARACTER COUNT OF THE TAG
        R3  POINTS TO PROGRAM TABLE

EXIT-   R2  CONTAINS ABSOLUTE ADDRESS OF TAG
*****

```

```

7207
7208
7209
7210
7211
7212
7213
7214
7215
7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226
7227
7228
7229 026632' 010167 000524
7230 026636' 010702
7231 026640' 062702 001174
7232 026644' 020267 000506
7233 026650' 001002
7234 026652' 000167 000274
7235 026656' 020267 000476
7236 026662' 001002
7237 026664' 000167 000312
7238 026670' 021227 177777
7239 026674' 001442
7240 026676' 121012
7241 026700' 001033
7242 026702' 005301
7243 026704' 126062 000001 000001
7244 026712' 001026
7245 026714' 005301
7246 026716' 001413
7247 026720' 126062 000002 000002
7248 026726' 001020
7249 026730' 005301
7250 026732' 001411
7251 026734' 126062 000003 000003
7252 026742' 001012
7253 026744' 000422
7254 026746' 126227 000002 000040
7255 026754' 001005
7256 026756' 126227 000003 000040
7257 026764' 001001
7258 026766' 000411
7259 026770' 016701 000366
7260 026774' 062702 000006
7261 027000' 000721
7262 027002' 016701 000354

```

```

TAGLUP: MOV      R1, LTAGCT      ;SAVE TAG CHAR. COUNT
        MOV      PC, R2
        ADD      #LCSTBL-, R2   ;POINT R2 AT SYMBOL TABLE
LTLLP1: CMP      R2, LSYMTE     ;END OF TABLE
        BNE      LTLEP2
        JMP      LSYNIT        ;YES - CHECK DEVICE ROUTINE TBL
LTLEP2: CMP      R2, LDRSTE     ;END OF DEVICE ROUT TBL ?
        BNE      LTLEP3
        JMP      LSYNDT        ;YES - CHECK INTERF WD TBL
LTLEP3: CMP      (R2), #177777  ;END OF INTERF WD SYM TBL?
        BEQ      LSYMNF        ;YES - SYMBOL NOT FOUND
        CMPB     (R0), (R2)     ;FIRST CHARS. EQUAL?
        BNE      LNOTIT        ;NO - BRANCH
        DEC      R1            ;YES - DECREMENT CHAR CNT
        CMPB     1(R0), 1(R2)   ;SECOND CHARS. EQUAL?
        BNE      LNOTIT        ;NO - BRANCH
        DEC      R1            ;YES - DECREMENT CHAR CNT
        BEQ      LTGLN2        ;BRANCH IF COUNT=0
        CMPB     2(R0), 2(R2)   ;THIRD CHARS. EQUAL?
        BNE      LNOTIT        ;NO - BRANCH
        DEC      R1            ;YES - DECREMENT CHAR CNT
        BEQ      LTGLN1        ;BRANCH IF COUNT=0
        CMPB     3(R0), 3(R2)   ;FOURTH CHARS. EQUAL?
        BNE      LNOTIT        ;NO - BRANCH
        BR       LTTBIN        ;WE FOUND IT
LTGLN2: CMPB     2(R2), #40     ;THIRD CHAR. A BLANK?
        BNE      LNOTIT        ;NO - BRANCH
LTGLN1: CMPB     3(R2), #40     ;FOURTH CHAR A BLANK?
        BNE      LNOTIT        ;NO - BRANCH
        BR       LTTBIN        ;YES - WE FOUND IT
LNOTIT: MOV      LTAGCT, R1     ;RESTORE CHAR. COUNT
        ADD      #6, R2        ;BUMP SYMBOL POINTER
        BR       LTLLP1        ;TRY AGAIN
LSYMNf: MOV      LTAGCT, R1     ;RESTORE R1

```

```

7263 027006' 061505
7264 027010' 000205
7265 027012' 010701
7266 027014' 062701 000350
7267 027020' 020267 000324
7268 027024' 103444
7269 027026' 020267 000320
7270 027032' 103431
7271 027034' 020267 000314
7272 027040' 103416
7273 027042' 020267 000310
7274 027046' 101402
7275 027050' 000167 000142
7276 027054' 062702 000004
7277 027060' 011202
7278 027062' 060302
7279 027064' 005011
7280 027066' 011261 000002
7281      001
7282
7283
7284
7285
7286
7287
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299      000
7300 027072' 000167 000236
7301 027076' 062702 000004
7302 027102' 011202
7303 027104' 005011
7304 027106' 010261 000002
7305      001
7306
7307
7308
7309
7310
7311
7312
7313
7314
7315
7316
7317      000
7318 027112' 000157 000216

```

```

      ADD      (R5),R5
      RTS
LTTBIN: MOV     PC,R1
      ADD     #LTGADR-,R1
      CMP     R2,LXRTBS
      BLO    LDIR
      CMP     R2,LIRTBS
      BLO    LINDEX
      CMP     R2,LIITBS
      BLO    LINDIR
      CMP     R2,LSYMTE
      BLOS   LININD
      JMP     LDISP
LININD: ADD     #4,R2
      MOV     (R2),R2
      ADD     R3,R2
      CLR     (R1)
      MOV     (R2),2(R1)
      .IF DF MM
      TSTB   NPRFLG
      BPL    30$
      MOV     R3,-(SP)
      ADD     #PRDIOA,(SP)
      CMP     (SP)+,R2
      BNE    10$
      ADD     #PRDIOX-PRDIOA,R2
      BR     20$
10$:  ADD     #PWRIOX-PWRIOA,R2
20$:  MOV     (R2)+,(R1)
      MOV     (R2)+,2(R1)
      BITB   #10,MASS70
      BNE    30$
      BIT    #UNIMAP,CSYSFW
      BEQ    30$
      MOV     (R2)+,(R1)
      MOV     (R2),2(R1)
      .ENDC
30$:  JMP     LTLEXT
LINDIR: ADD     #4,R2
      MOV     (R2),R2
      CLR     (R1)
      MOV     (R2),2(R1)
      .IF DF MM
      MOV     (R2)+,2(R1)
      TSTB   NPRFLG
      BPL    10$
      MOV     (R2)+,(R1)
      MOV     (R2)+,2(R1)
      BITB   #10,MASS70
      BNE    10$
      BIT    #UNIMAP,CSYSFW
      BEQ    10$
      MOV     (R2)+,(R1)
      MOV     (R2),2(R1)
      .ENDC
10$:  JMP     LTLEXT

```

```

;CALCULATE ERROR RETURN
;TAKE ERROR RETURN
;POINT R1 AT
;ADDRESS AREA
;R2 BEFORE INDEX REF START?
;MUST BE DIRECT REFERENCE
;R2 BEFORE INDIRECT REF START?
;MUST BE INDEXED REFERENCE
;R2 BEFORE INDX IND START ?
;MUST BE IND REF
;R2 AFTER MPG TBL END ?
;MUST BE DISPLACED REFERENCE
;BUMP TABLE POINTER
;GET DISPLACEMENT
;ADD TO PROG TBL START
;CLR MSW OF ADDR
;UPDATE LSW OF ADR
;TEST NPR FLAG
;IF RESET, SKIP
;SET UP COMPARISON ADR
;DOING RDIO ADR?
;Y,N-10$
;POINT AT RDIO 18 BIT ADR
;CONTINUE PROCESSING
;POINT AT WRIO 18 BIT ADR
;UPDATE MSW OF ADR
;UPDATE LSW OF ADR
;TEST MASS BUS/70 FLAG
;IF SET, LEAVE ABS ADR ALONE
;TEST UNIBUS MAP FLAG
;IF NOT SET LEAVE ABS ADR ALONE
;ELSE GET 18 BIT VADR
;EXIT
;BUMP TABLE POINTER
;GET FREE ADR PTR
;CLR MSW OF ADDR
;UPDATE LSW OF ADR
;UPDATE LSW OF ADR
;TEST MASS BUS/70 FLAG
;IF SET, LEAVE ABS ADR
;TEST UNIBUS MAP FLAG
;IF NOT SET, LEAVE ABS ADR
;ELSE GET 18 BIT VADR
;EXIT

```

7319	027116'	062702	000004	LINDEX: ADD	#4,R2	;BUMP TABLE POINTER
7320	027122'	011202		MOV	(R2),R2	;GET DISPLACEMENT
7321		001		.IF DF MM		
7322				BIC	#P4CONS,R3	;CLEAR PAR4 SELECT BITS
7323		000		.ENDC		
7324	027124'	060302		XCALC: ADD	R3,R2	;ADD TO PROG TBL START
7325	027126'	005011		CLR	(R1)	;CLR MSW OF ADR
7326	027130'	010261	000002	MOV	R2,2(R1)	;UPDATE LSW OF ADR
7327		001		.IF DF MM		
7328				BIC	#100000,2(R1)	;RESET PAR4 SELECT BITS
7329				TSTB	NPRFLG	;TEST NPR FLAG
7330				BPL	20\$;SKIP IF NOT SET
7331				MOV	2(R1),R3	;DISPLACEMENT TO R3
7332				MOV	PC,R2	;POINT R2 AT
7333				ADD	#PGABAD-,R2	;P TBL ABS ADR
7334				BIT	#UNIMAP,CSYSFW	;UNIBUS MAP?
7335				BEQ	10\$;NO, GO CALC ABS ADR
7336				BITB	#10,NPRFLG	;MASS BUS DEV?
7337				BNE	10\$;YES, GO CALC ABSADR
7338				ADD	#4,R2	;ADJUST R2 FOR PTBL VADR
7339				10\$: MOV	(R2)+(R1)	;GET APPROPRIATE
7340				MOV	(R2),2(R1)	;P TBL ADR
7341				ADD	R3,2(R1)	
7342				ADC	(R1)	
7343				BR	20\$	
7344		000		.ENDC		
7345	027134'	000477		20\$: BR	LTLEXT	;EXIT
7346	027136'	062702	000004	LDIR: ADD	#4,R2	;BUMP TABLE POINTER
7347	027142'	005011		CLR	(R1)	;CLR MSW OF ADR
7348	027144'	011261	000002	MOV	(R2),2(R1)	;UPDATE LSW OF ADR
7349		001		.IF DF MM		
7350				TSTB	NPRFLG	;TEST NPR FLAG
7351				BPL	10\$;SKIP IF RESET
7352				BIT	#UNIMAP,CSYSFW	;UNIBUS MAP?
7353				BEQ	5\$;NO, GO CALC ABS ADR
7354				BITB	#10,NPRFLG	;MASS BUS DEV?
7355				BEQ	10\$;NO, LEAVE VADR ALONE
7356				5\$: SUB	#P6CONS,2(R1)	;TAKE OUT P6 BITS
7357				MOV	PC,R2	
7358				ADD	#PG6BA-,R2	;PICK UP PAGE 6 BASE ADR
7359				ADD	R2,2(R1)	;ADD TO DISPLACEMENT
7360				ADC	(R1)	;UPDATE MSW WITH CARRY
7361		000		.ENDC		
7362	027150'	000471		10\$: BR	LTLEXT	;EXIT
7363	027152'	010302		LSYNIT: MOV	R3,R2	;POINT R2 AT PROG TBL
7364	027154'	062702	000346	ADD	#PTLGTH+DEVETP,R2	
7365	027160'	061202		ADD	(R2),R2	;ENTRY LENGTH PNTR TO R2
7366	027162'	010267	000172	MOV	R2,LDRSTE	;USE AS SYMBOL TABLE END
7367	027166'	010302		MOV	R3,R2	;POINT R2 AT PROG TABLE
7368	027170'	062702	000344	ADD	#PTLGTH+DEVSTP,R2	
7369	027174'	061202		ADD	(R2),R2	;SYMBOL TBL POINTER TO R2
7370	027176'	000167	177454	JMP	LTLEP2	;GO LOOKUP SYMBOL
7371	027202'	010302		LSYNDT: MOV	R3,R2	;POINT R2 AT PROG TBL
7372	027204'	062702	000355	ADD	#PTLGTH+DVIWSP,R2	
7373	027210'	061202		ADD	(R2),R2	;INTERFACE SYM TBL ADR TO R2
7374	027212'	000167	177452	JMP	LTLEP3	;GO LOOK UP SYMBOL

.SBTTL FIND LINE ENTRY SUBROUTINE

```

*****
THIS SUBROUTINE SEARCHES THE PACKED SOURCE ENTRIES
FOR A GIVEN LINE NUMBER. IF IT IS NOT THERE, IT
TAKES THE "NOT FOUND" EXIT.

CALLING SEQUENCE      JSR      R5,LFINLN
                       .WORD    NOTFND-.

ENTRY-  R3 POINTS TO PROGRAM TABLE
        R2 POINTS TO NEW LINE (2 BYTES BEFORE LINE #)

EXIT-   R1 POINTS TO THE REQUESTED LINE
*****

```

7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450
7451
7452
7453
7454
7455
7456
7457
7458
7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472
7473
7474

```

027370' 016301 000022
027374' 016300 000024
      001
      000
027400' 020100
027402' 001407
      001
      000
027404' 026162 000002 000002
027412' 001405
027414' 111100
      001
      000
027416' 060001
027420' 000765
      001
      000
027422' 061505
      000
027424' 000205
027426' 005725
027430' 000205

```

```

LFINLN: MOV      PSRCST(R3),R1      ;GET SOURCE START
LFLLP1: MOV      POBJST(R3),R0      ;GET OBJECT START
      .IF DF MM
      BIC      #P4CONS,R0          ;RESET PAR4 SELECT BITS
      .ENDC
      CMP      R1,R0                ;IS THIS THE END?
      BEQ      LENDFD              ;YES, EXIT
      .IF DF MM
      BIS      #P4CONS,R1          ;SET PAR 4 SELECT BITS
      .ENDC
      CMP      2(R1),2(R2)          ;IS THIS THE LINE?
      BEQ      LINFND              ;YES-EXIT
      MOV      (R1),R0              ;GET SOURCE LENGTH
      .IF DF MM
      BIC      #P4CONS,R1          ;RESET PAR4 SELECT BITS
      .ENDC
      ADD      R0,R1                ;POINT R1 AT NEXT ENTRY
      BR      LFLLP1              ;TRY AGAIN
      .IF DF MM
      .IFT
LENDFD: BIS      #P4CONS,R1          ;SET PAR4 SELECT BITS IN LINE ADR
      ADD      (R5),R5              ;CALC NOT FOUND EXIT
      .IFF
LENDFD: ADD      (R5),R5            ;CALC NOT FOUND EXIT
      .ENDC
      RTS      R5                  ;AND GO THERE
LINFND: TST      (R5)+
      RTS      R5                  ;RETURN TO CALLING PGM

```

7476
7477
7478
7479
7480
7481
7482
7483
7484
7485
7486
7487
7488
7489
7490
7491
7492
7493
7494
7495
7496
7497
7498
7499
7500
7501
7502
7503
7504
7505
7506
7507
7508
7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531

.SBTTL OBJECT SIZER SUBROUTINE

```

*****
THIS SUBROUTINE DETERMINES THE AMOUNT
OF MEMORY REQUIRED FOR THE MACHINE CODE
THAT IS GENERATED BY A GIVEN STATEMENT.

CALLING SEQUENCE      JSR      R5,LCKOBL

ENTRY-  R0 POINTS TO PACKED STATEMENT

EXIT-   R0 CONTAINS LENGTH
*****

```

```

027432' 010146      LCKOBL: MOV      R1,-(SP)      ;SAVE
027434' 010246      MOV      R2,-(SP)      ;R1 AND R2
027436' 010702      MOV      PC,R2
027440' 062702      001212      ADD      #LCMCTB-.,R2      ;POINT R2 AT COMPILER TABLE
027444' 105760      000001      LCOLP1: TSTB     1(R0)      ;IF FUNCT CODE NOT = 0
027450' 001002      BNE      LCOTG1      ;CHECK TABLE
027452' 005000      CLR      R0          ;OTHERWISE SET LENGTH = 0
027454' 000464      BR       LCOTG2
027456' 122762      000240      000001      LCOTG1: CMPB     #240,1(R2)      ;F.C. REPRESENTS END?
027464' 001405      BEQ      LCOEIT      ;YES - CHECK FURTHER
027466' 126062      000001      000001      CMPB     1(R0),1(R2)    ;COMPARE FUNCTION CODES
027474' 001433      BEQ      LCOLFD      ;BRANCH IF FOUND
027476' 000426      BR       LCOBPT      ;ELSE BUMP POINTER
027500' 126062      000001      000001      LCOEIT: CMPB     1(R0),1(R2)    ;COMPARE FUNCTION CODES
027506' 001426      BEQ      LCOLFD      ;MUST BE END
027510' 016702      171554      MOV      LPKPTP,R2     ;GET P TBL POINTER
027514' 062702      000354      ADD      #PTLGTH+DVCTEP,R2 ;POINT R2
027520' 061202      ADD      (R2),R2      ;AT COMPILE TBL EXTEN.
027522' 122762      000377      000001      10$:  CMPB     #377,1(R2)    ;END OF TBL
027530' 001410      BEQ      LCOBUG      ;YES-MUST BE BUG
027532' 126062      000001      000001      CMPB     1(R0),1(R2)    ;COMPARE FUNCTION CODES
027540' 001411      BEQ      LCOLFD      ;BRANCH IF FOUND
027542' 111201      MOVB     (R2),R1
027544' 006301      ASL     R1
027546' 060102      ADD     R1,R2          ;ELSE BUMP POINTER
027550' 000764      BR     10$           ;TRY AGAIN
027552' 000000      LCOBUG: HALT          ;YES-HALT (MPG BUG)
027554' 111201      LCOBPT: MOVB     (R2),R1
027556' 006301      ASL     R1
027560' 060102      ADD     R1,R2          ;BUMP TABLE POINTER
027562' 000730      BR     LCOLP1        ;TRY AGAIN
027564' 122760      000241      000001      LCOLFD: CMPB     #241,1(R0)    ;WAS IT A PRINT * ?
027572' 001001      BNE     LCOENR        ;NO-GET LENGTH FROM TABLE
027574' 000404      BR     LCOGPL        ;YES-GET PACKED LENGTH
027576' 122760      000261      000001      LCOENR: CMPB     #261,1(R0)    ;WAS IT A PRINT # ?
027604' 001004      BNE     LCOGLT        ;NO-GET LENGTH FROM TABLE
027606' 111000      LCOGPL: MOVB     (R0),R0    ;YES-GET PACKED LENGTH
027610' 062700      000002      ADD     #2,R0          ;THEN ADD 2
027614' 000404      BR     LCOTG2        ;AND EXIT

```

7532 027616' 111200
7533 027620' 006300
7534 027622' 162700 000002
7535 027626' 012602
7536 027630' 012601
7537 027632' 000205

LCOGLT: MOVB (R2),R0
ASL R0
SUB #2,R0
LCCTG2: MOV (SP)+,R2
MOV (SP)+,R1
RTS R5

; CONVERT LENGTH TO BYTES
; AND ADJUST
; RESTORE
; R1 AND R2
; RETURN TO CALLING PGM

7539
7540
7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572
7573
7574
7575
7576
7577
7578
7579
7580
7581
7582
7583
7584
7585
7586
7587
7588
7589
7590
7591
7592
7593
7594

.SBTTL FIND DEVICE COMMAND ENTRY SUBROUTINE

```

*****
THIS SUBROUTINE SEARCHES THE DEVICE ROUTINE
ENTRY TABLE FOR A FUNCTION CODE MATCH.
IF IT FINDS A MATCH IT RETURNS WITH THE
ABSOLUTE ADDRESS OF THE ENTRY POINT
IN R2. IF THERE IS NO MATCH, IT POINTS
AN ERROR MESSAGE IN THE FOLLOWING FORMAT:
'CANT XXXX ON YYYY', WHERE XXXX IS THE
ATTEMPTED COMMAND AND YYYY IS THE DEVICE NAME.
IT THEN TAKES THE ERROR RETURN.

CALLING SEQUENCE-      JSR      R5,FENTRY
                        .WORD    EARTN-.

ENTRY   R0 CONTAINS FUNCTION CODE
        R2 POINTS AT PROGRAM TABLE

EXIT-   R2 CONTAINS ABSOLUTE ADDRESS OF ENTRY
        R1 CONTAINS NPR FLAG BYTE
        R0 IS DESTROYED
*****

```

```

027634' 010201
027636' 062702 000346
027642' 061202
027644' 120027 000167
027650' 101002
027652' 042700 000007
027656' 122712 000377
027662' 001413
027664' 121200
027666' 001403
027670' 062702 000004
027674' 000770
027676' 116201 000001
027702' 005722
027704' 061202
      001
      000
027706' 005725
027710' 000205
027712' 010702
027714' 062702 171374
027720' 120012
027722' 001406
027724' 122712 000377
027730' 001424
027732' 062702 000010
027736' 000770
027740' 162702 000006

```

```

FENTRY: MOV      R2,R1          ;POINT R1 AT PROG TBL
        ADD      #P1LGTH+DEVETP,R2 ;GET ENTRY TBL POINTER
        ADD      (R2),R2        ;POINT R2 AT TABLE
        CMPB     RD,#167       ;IF NOT RD, WR OR BRK
        BHI     LLENLP1        ;CONTINUE
        BIC      #7,R0         ;ELSE RESET BITS 0-2
        CMPB     #377,(R2)     ;END OF TABLE?
        BEQ     LENCNT         ;TYPE 'CANT DO IT' MSG
        CMPB     (R2),R0       ;FUNCTION CODE IN TBL
        BEQ     LENFCF        ;YES - BRANCH
        ADD      #4,R2         ;NO - BUMP POINTER
        BR       LLENLP1       ;KEEP LOOKING
LENLPI: CMPB     #377,(R2)     ;GET NPR FLAG
        BEQ     LENCNT         ;R2 = ENTRY DISPLACEMENT
        BEQ     LENCNT         ;CALC ABSOLUTE ADDR
        ADD      (R2),R2
        .IF DF MM
        SUB      #P4CONS,R2   ;TAKE OUT PAR 4 BITS
        .ENDC
        TST     (R5)+         ;BYPASS ERROR RETURN ADR
        RTS     R5            ;EXIT
LENCNT: MOV      PC,R2
        ADD      #LPKTB+6-.,R2 ;POINT R2 AT PACK TBL
LENLPI2: CMPB    RO,(R2)       ;FUNCTION CODE MATCH?
        BEQ     LENCFD        ;YES - COMMAND FOUND
        BEQ     LENCNT        ;END OF TABLE?
        BEQ     LENEEX        ;YES - TAKE ERROR EXIT
        ADD     #8.,R2        ;BUMP R2
        BR      LENLPI2       ;KEEP LOOKING
LENCFD: SUB      #6,R2        ;R2 POINTS TO ASCII CMD

```

```

7595 027744' 012267 000044      MOV      (R2)+,LENCMD      ;MOVE
7596 027750' 012267 000042      MOV      (R2)+,LENCMD+2  ;ASCII COMMAND
7597 027754' 012267 000040      MOV      (R2)+,LENCMD+4  ;TO ERROR MESSAGE
7598 027760' 016101 000032      MOV      PMDLC0(R1),R1   ;GET MDL CODE IN R1
7599 027764' 004567 163556      JSR      R5,F0VNAM       ;FIND DEV NAME FOR THIS MCL
7600 027770' 000036              .WORD    LENEDEV-        ;CODE & PUT IN THE MSG
7601 027772' 004567 156226      JSR      R5,CTYPE        ;TYPE 'CANT XXXX ON YYYY'
7602 027776' 000402              .WORD    402
7603 030000' 000006              .WORD    LENMSG-
7604 030002' 061505      LENEEX: ADD      (R5),R5  ;CALC ERROR RETURN
7605 030004' 000205      RTS          R5          ;AND TAKE IT
7606
7607 030006' 040503 023516 020124 LENMSG: .ASCII /CAN'T /
7608 030014' 020040 020040 020040 LENCMD: .ASCII /      ON /
7609 030022' 047440 020116              .WORD
7610 030026' 020040 020040 000 LENEDEV: .ASCIZ /      /
7611 030034'              .EVEN

```

				.SBTTL COMMON STORAGE SYMBOL TABLE (DIRECT REFERENCE)	
7613					
7614					
7615					
7616		001			
7617	030034'	052502	020106	LCSTBL:	.IF NDF MM
7618	030040'	041440'			.ASCII /BUF /
7619	030042'	043102	030060		.WORD CBUF00
7620	030046'	041440'			.ASCII /BF00/
7621	030050'	043102	030460		.WORD CBUF00
7622	030054'	041460'			.ASCII /BF01
7623	030056'	043102	031060		.WORD CBUF01
7624	030062'	041500'			.ASCII /BF02/
7625	030064'	043102	031460		.WORD CBUF02
7626	030070'	041520'			.ASCII /BF03/
7627	030072'	043102	032060		.WORD CBUF03
7628	030076'	041540'			.ASCII /BF04/
7629	030100'	043102	032460		.WORD CBUF04
7630	030104'	041560'			.ASCII /BF05/
7631	030106'	043102	033060		.WORD CBUF05
7632	030112'	041600'			.ASCII /BF06/
7633	030114'	043102	033460		.WORD CBUF06
7634	030120'	041620'			.ASCII /BF07/
7635	030122'	043102	034060		.WORD CBUF07
7636	030126'	041640'			.ASCII /BF08/
7637	030130'	043102	034460		.WORD CBUF08
7638	030134'	041660'			.ASCII /BF09/
7639	030136'	043102	030061		.WORD CBUF09
7640	030142'	041700'			.ASCII /BF10/
7641	030144'	043102	030461		.WORD CBUF10
7642	030150'	041720'			.ASCII /BF11/
7643	030152'	043102	031061		.WORD CBUF11
7644	030156'	041740'			.ASCII /BF12/
7645	030160'	043102	031461		.WORD CBUF12
7646	030164'	041760'			.ASCII /BF13/
7647	030166'	043102	032061		.WORD CBUF13
7648	030172'	042000'			.ASCII /BF14/
7649	030174'	043102	032461		.WORD CBUF14
7650	030200'	042020'			.ASCII /BF15/
					.WORD CBUF15

7652	030202'	047503	030115	LCTBST:	.ASCII	/COM0/
7653	030206'	042040'			.WORD	LCOM0
7654	030210'	047503	030515		.ASCII	/COM1/
7655	030214'	042042'			.WORD	LCOM1
7656	030216'	047503	031115		.ASCII	/COM2/
7657	030222'	042044'			.WORD	LCOM2
7658	030224'	047503	031515		.ASCII	/COM3/
7659	030230'	042046'			.WORD	LCOM3
7660	030232'	047503	032115		.ASCII	/COM4/
7661	030236'	042050'			.WORD	LCOM4
7662	030240'	047503	032515		.ASCII	/COM5/
7663	030244'	042052'			.WORD	LCOM5
7664	030246'	047503	033115		.ASCII	/COM6/
7665	030252'	042054'			.WORD	LCOM6
7666	030254'	047503	033515		.ASCII	/COM7/
7667	030260'	042056'			.WORD	LCOM7
7668	030262'	047503	034115		.ASCII	/COM8/
7669	030266'	042060'			.WORD	LCOM8
7670	030270'	047503	034515		.ASCII	/COM9/
7671	030274'	042062'			.WORD	LCOM9
7672	030276'	040520	030124		.ASCII	/PAT0/
7673	030302'	042064'			.WORD	LPAT0P
7674	030304'	040520	030524		.ASCII	/PAT1/
7675	030310'	042066'			.WORD	LPAT1P
7676	030312'	040520	031124		.ASCII	/PAT2/
7677	030316'	042070'			.WORD	LPAT2P
7678	030320'	040520	031524		.ASCII	/PAT3/
7679	030324'	042072'			.WORD	LPAT3P
7680	030326'	040520	032124		.ASCII	/PAT4/
7681	030332'	042074'			.WORD	LPAT4P
7682	030334'	040520	032524		.ASCII	/PAT5/
7683	030340'	042076'			.WORD	LPAT5P
7684	030342'	040520	033124		.ASCII	/PAT6/
7685	030346'	042100'			.WORD	LPAT6P
7686	030350'	040520	033524		.ASCII	/PAT7/
7687	030354'	042102'			.WORD	LPAT7P
7688	030356'	040520	034124		.ASCII	/PAT8/
7689	030362'	042104'			.WORD	LPAT8P
7690	030364'	040520	034524		.ASCII	/PAT9/
7691	030370'	042106'			.WORD	LPAT9P
7692	030372'	040520	040524		.ASCII	/PATA/
7693	030376'	042110'			.WORD	LPATAP
7694	030400'	040520	041124		.ASCII	/PATB/
7695	030404'	042112'			.WORD	LPATBP
7696	030406'	051103	043114		.ASCII	/CRLF/
7697	030412'	042114'			.WORD	LCRETN
7698		000			.ENDC	
7699		001			.IF OF MM	
7700				LCSTBL:	.ASCII	/BUF /
7701					.WORD	CBUF00-PG6BA+P6CONS
7702					.ASCII	/BF00/
7703					.WORD	CBUF00-PG6BA+P6CONS
7704					.ASCII	/BF01/
7705					.WORD	CBUF01-PG6BA+P6CONS
7706					.ASCII	/BF02/
7707					.WORD	CBUF02-PG6BA+P6CONS

7708	.ASCII	/BF03/
7709	.WORD	CBUF03-PG6BA+P6CONS
7710	.ASCII	/BF04/
7711	.WORD	CBUF04-PG6BA+P6CONS
7712	.ASCII	/BF05/
7713	.WORD	CBUF05-PG6BA+P6CONS
7714	.ASCII	/BF06/
7715	.WORD	CBUF06-PG6BA+P6CONS
7716	.ASCII	/BF07/
7717	.WORD	CBUF07-PG6BA+P6CONS
7718	.ASCII	/BF08/
7719	.WORD	CBUF08-PG6BA+P6CONS
7720	.ASCII	/BF09/
7721	.WORD	CBUF09-PG6BA+P6CONS
7722	.ASCII	/BF10/
7723	.WORD	CBUF10-PG6BA+P6CONS
7724	.ASCII	/BF11/
7725	.WORD	CBUF11-PG6BA+P6CONS
7726	.ASCII	/BF12/
7727	.WORD	CBUF12-PG6BA+P6CONS
7728	.ASCII	/BF13/
7729	.WORD	CBUF13-PG6BA+P6CONS
7730	.ASCII	/BF14/
7731	.WORD	CBUF14-PG6BA+P6CONS
7732	.ASCII	/BF15/
7733	.WORD	CBUF15-PG6BA+P6CONS
7734	.PAGE	
7735		
7736		
7737	LCTBST: .ASCII	/COM0/
7738	.WORD	LCOM0-PG6BA+P6CONS
7739	.ASCII	/COM1/
7740	.WORD	LCOM1-PG6BA+P6CONS
7741	.ASCII	/COM2/
7742	.WORD	LCOM2-PG6BA+P6CONS
7743	.ASCII	/COM3/
7744	.WORD	LCOM3-PG6BA+P6CONS
7745	.ASCII	/COM4/
7746	.WORD	LCOM4-PG6BA+P6CONS
7747	.ASCII	/COM5/
7748	.WORD	LCOM5-PG6BA+P6CONS
7749	.ASCII	/COM6/
7750	.WORD	LCOM6-PG6BA+P6CONS
7751	.ASCII	/COM7/
7752	.WORD	LCOM7-PG6BA+P6CONS
7753	.ASCII	/COM8/
7754	.WORD	LCOM8-PG6BA+P6CONS
7755	.ASCII	/COM9/
7756	.WORD	LCOM9-PG6BA+P6CONS
7757	.ASCII	/PAT0/
7758	.WORD	LPAT0P-PG6BA+P6CONS
7759	.ASCII	/PAT1/
7760	.WORD	LPAT1P-PG6BA+P6CONS
7761	.ASCII	/PAT2/
7762	.WORD	LPAT2P-PG6BA+P6CONS
7763	.ASCII	/PAT3/

7764	.WORD	LPAT3P-PG6BA+P6CONS
7765	.ASCII	/PAT4/
7766	.WORD	LPAT4P-PG6BA+P6CONS
7767	.ASCII	/PAT5/
7768	.WORD	LPAT5P-PG6BA+P6CONS
7769	.ASCII	/PAT6/
7770	.WORD	LPAT6P-PG6BA+P6CONS
7771	.ASCII	/PAT7/
7772	.WORD	LPAT7P-PG6BA+P6CONS
7773	.ASCII	/PAT8/
7774	.WORD	LPAT8P-PG6BA+P6CONS
7775	.ASCII	/PAT9/
7776	.WORD	LPAT9P-PG6BA+P6CONS
7777	.ASCII	/PATA/
7778	.WORD	LPATAP-PG6BA+P6CONS
7779	.ASCII	/PATB/
7780	.WORD	LPATBP-PG6BA+P6CONS
7781	.ASCII	/CRLF/
7782	.WORD	LCRETN-PG6BA+P6CONS
7783	.ENDC	

000

				.SBTTL TEMPORARY STORAGE SYMBOL TABLE (INDEX REFERENCED)	
7785					
7786					
7787					
7788	030414'	046524	030060	LXTBL:	.ASCII /TM00/
7789	030420'	000056			.WORD PTEM0
7790	030422'	046524	030460		.ASCII /TM01/
7791	030426'	000060			.WORD PTEM1
7792	030430'	046524	031060		.ASCII /TM02/
7793	030434'	000062			.WORD PTEM2
7794	030436'	046524	031460		.ASCII /TM03/
7795	030442'	000064			.WORD PTEM3
7796	030444'	046524	032060		.ASCII /TM04/
7797	030450'	000066			.WORD PTEM4
7798	030452'	046524	032460		.ASCII /TM05/
7799	030456'	000070			.WORD PTEM5
7800	030460'	046524	033060		.ASCII /TM06/
7801	030464'	000072			.WORD PTEM6
7802	030466'	046524	033460		.ASCII /TM07/
7803	030472'	000074			.WORD PTEM7
7804	030474'	046524	034060		.ASCII /TM08/
7805	030500'	000076			.WORD PTEM8
7806	030502'	046524	034460		.ASCII /TM09/
7807	030506'	000100			.WORD PTEM9
7808	030510'	046524	030061		.ASCII /TM10/
7809	030514'	000102			.WORD PTEM10
7810	030516'	046524	030461		.ASCII /TM11/
7811	030522'	000104			.WORD PTEM11
7812	030524'	046524	031061		.ASCII /TM12/
7813	030530'	000106			.WORD PTEM12
7814	030532'	046524	031461		.ASCII /TM13/
7815	030536'	000110			.WORD PTEM13
7816	030540'	046524	032061		.ASCII /TM14/
7817	030544'	000112			.WORD PTEM14
7818	030546'	046524	032461		.ASCII /TM15/
7819	030552'	000114			.WORD PTEM15
7820	030554'	050117	053523		.ASCII /OPSW/
7821	030560'	000002			.WORD PPSW
7822	030562'	044523	042532		.ASCII /SIZE/
7823	030566'	000262			.WORD PTLGTH+DEVIW7
7824	030570'	051105	020122		.ASCII /ERR /
7825	030574'	000264			.WORD PTLGTH+DEVIW8
7826	030576'	041116	020122		.ASCII /NBR /
7827	030602'	000116			.WORD PNBR
7828	030604'	051123	020103		.ASCII /SRC /
7829	030610'	000120			.WORD PSRC
7830	030612'	051504	020124		.ASCII /DST /
7831	030616'	000122			.WORD PDST

```

7833                                     .SBTTL  INDIRECT REFERENCE SYMBOL TABLE
7834
7835
7836 030620' 044515 046104      LINDTB: .ASCII /MIDL/
7837                               .IF DF MM
7838                               .IFF
7839 030624' 177777          LMIDL:  .WORD  177777
7840                               .IFT
7841                               .MIDL:  .WORD  MIDLAD
7842                               .ENDC
7843 030626' 051106 042505      .ASCII /FREE/
7844                               .IF DF MM
7845                               .IFF
7846 030632' 177777          LFREE:  .WORD  177777
7847                               .IFT
7848                               LFREE:  .WORD  FREEAD
7849                               .ENDC
7850 030634' 042122 047511      LXINDT: .ASCII /RDIO/
7851 030640' 000016          .WORD  PRDIOA
7852 030642' 051127 047511      .ASCII /WRIO/
7853 030646' 000020          .WORD  PWRIOA
7854 030650' 000000          LSMEND: .WORD  0
7855                               .IF DF MM
7856
7857
7858                               FREEAD: .WORD  P3CONS      ;16 BIT VIRTUAL ADR FOR FREE
7859                               .WORD  XXXX          ;18/22 BIT ABSOLUTE ADDRESS
7860                               .WORD  XXXX          ;FOR FREE
7861                               FREEVA: .WORD  XXXX          ;18 BIT VIRTUAL ADDRESS
7862                               .WORD  XXXX          ;FOR FREE
7863
7864                               MIDLAD: .WORD  XXXX          ;16 BIT VIRTUAL ADR FOR MIDL
7865                               MIDLAA: .WORD  XXXX          ;18/22 BIT ABSOLUTE ADDRESS
7866                               .WORD  XXXX          ;FOR MIDL
7867                               MIDLVA: .WORD  XXXX          ;18 BIT VIRTUAL ADDRESS
7868                               .WORD  XXXX          ;FOR MIDL
7869                               000
7870                               .ENDC
7871
7872                               001
7873                               .IF DF MM
7874                               .IFF
7875                               .TITLE MAINDEC-11-DTMGA-C  MPG COMPILER
7876                               .IFT
7877                               .TITLE MAINDEC-11-DTMMA-B  MPG COMPILER
7878                               .ENDC

```


.SBTTL COMPILER MACHINE CODE TABLE

COMPILER UTILIZES TABLE AS FOLLOWS-

ENTRY	ACTION
>15	STORES TABLE ENTRY AS OBJECT CODE
15	STORES A VALUE OF 1 AS OBJECT CODE
14	CALCULATES DST ADDRESS FOR THIS PROGRAM
13	CALCULATES SRC ADDRESS FOR THIS PROGRAM
12	CALCULATES NBR ADDRESS FOR THIS PROGRAM
11	STORES ALL ZEROES AS OBJECT CODE
10	FINDS AND STORES DEVICE ROUTINE ENTRY ADDRESS
9	STORES A VALUE OF 5 AS OBJECT CODE
8	MODIFYS PREVIOUS CODE IF V NOT A TAG
7	CALCULATES WRIO ADDRESS FOR THIS PROGRAM
6	CALCULATES RDIO ADDRESS FOR THIS PROGRAM
5	STORES A VALUE OF 2 AS OBJECT CODE
4	INTERPRETS PACKED CODE AS BIT NUMBERS
3	MODIFYS NEXT TABLE ENTRY IF V NOT A TAG
2	INTERPRETS VARIABLE THEN SWAPS WITH PRECEEDING WORD
1	INTERPRETS PACKED CODE AS BIT NUMBER THEN SWAPS
0	INTERPRETS PACKED CODE AS VARIABLE-STORES BINARY EQUIV.

7879				
7880				
7881				
7882				
7883				
7884				
7885				
7886				
7887				
7888				
7889				
7890				
7891				
7892				
7893				
7894				
7895				
7896				
7897				
7898				
7899				
7900				
7901				
7902				
7903				
7904				
7905				
7906				
7907				
7908	030652'	004	242	
7909	030654'	012737	000000	000002
7910	030662'	003	011	
7911	030664'	005237	000000	
7912	030670'	003	021	
7913	030672'	005337	000000	
7914	030676'	004	013	
7915	030700'	062737	000000	000002
7916	030706'	004	023	
7917	030710'	162737	000000	000002
7918	030716'	012	030	
7919	030720'	013700	000014	013701
	030726'	000015		
7920	030730'	013702	000016	112122
7921	030736'	005300	001375	
7922	030742'	004	033	
7923	030744'	013737	000000	000000
7924	030752'	012	035	
7925	030754'	012700	000000	012701
	030762'	000000		
7926	030764'	012702	000000	112122
7927	030772'	005300	001375	
7928	030776'	004	043	
7929	031000'	063737	000000	000000
7930	031006'	004	053	
7931	031010'	163737	000000	000000
7932	031016'	006	045	

LCMCTB:	.BYTE	4,242		;LOAD V WITH V
	.WORD	012737,0,2		
	.BYTE	3,011		;INCR V
	.WORD	005237,0		
	.BYTE	3,021		;DECR V
	.WORD	005337,0		
	.BYTE	4,013		;INCR V BY V
	.WORD	062737,0,2		
	.BYTE	4,023		;DECR V BY V
	.WORD	162737,0,2		
	.BYTE	12,030		;MOVE
	.WORD	13700,12.,13701,13.		
	.WORD	13702,14.,112122		
	.WORD	5300,1375		
	.BYTE	4,033		;MOVE V TO V
	.WORD	013737,0,0		
	.BYTE	12,035		;MOVE V AT V TO V
	.WORD	012700,0,012701,0		
	.WORD	012702,0,112122		
	.WORD	005300,001375		
	.BYTE	4,043		;ADD V TO V
	.WORD	063737,0,0		
	.BYTE	4,053		;SUB V FROM V
	.WORD	163737,0,0		
	.BYTE	6,045		;ADD V AT V TO V

M13

MAINDEC-11-DTMGA-C MPG COMPILER MACY11 27(732) 24-SEP-76 13:54 PAGE 45-1
DTMGAC.P11 COMPILER MACHINE CODE TABLE

SEQ 0169

7933	031020'	004537				.WORD	004537	
7934	031022'	035722'	000000	000000	LRT1:	.WORD	LADDM,0,0,0	
	031030'	000000						
7935	031032'	006	055			.BYTE	6,055	;SUB V AT V FROM V
7936	031034'	004537				.WORD	004537	
7937	031036'	036020'	000000	000000	LRT2:	.WORD	LSUBM,0,0,0	
	031044'	000000						
7938	031046'	004	247			.BYTE	4,247	;SET V BIT N
7939	031050'	052737	000000	000001		.WORD	052737,0,1	
7940	031056'	004	250			.BYTE	4,250	;CLEAR V BIT N
7941	031060'	042737	000000	000001		.WORD	042737,0,1	
7942	031066'	011	100			.BYTE	9,100	;IF V BIT N SET GO TO LN
7943	031070'	013700	000000	005100		.WORD	013700,0,005100,032700,4	
	031076'	032700	000004					
7944	031102'	001002	000137	000000		.WORD	001002,000137,0	
7945	031110'	007	101			.BYTE	7,101	;IF V BIT N CLEAR GO TO LN
7946	031112'	032737	000000	000001		.WORD	032737,0,1	
7947	031120'	001002	000137	000000		.WORD	001002,000137,0	
7948	031126'	007	102			.BYTE	7,102	;IF V = V GO TO LN
7949	031130'	023737	000000	000010		.WORD	023737,0,8	
7950	031136'	001002	000137	000000		.WORD	001002,000137,0	
7951	031144'	007	103			.BYTE	7,103	;IF V > V GO TO LN
7952	031146'	023737	000000	000010		.WORD	023737,0,8	
7953	031154'	101402	000137	000000		.WORD	101402,000137,0	
7954	031162'	007	104			.BYTE	7,104	;IF V < V GO TO LN
7955	031164'	023737	000000	000010		.WORD	023737,0,8	
7956	031172'	103002	000137	000000		.WORD	103002,000137,0	
7957	031200'	007	105			.BYTE	7,105	;IF V => V GO TO LN
7958	031202'	023737	000000	000010		.WORD	023737,0,8	
7959	031210'	103402	000137	000000		.WORD	103402,000137,0	
7960	031216'	007	106			.BYTE	7,106	;IF V <= V GO TO LN
7961	031220'	023737	000000	000010		.WORD	023737,0,8	
7962	031226'	101002	000137	000000		.WORD	101002,000137,0	
7963	031234'	007	107			.BYTE	7,107	;IF V <> V GO TO LN
7964	031236'	023737	000000	000010		.WORD	023737,0,8	
7965	031244'	001402	000137	000000		.WORD	001402,000137,0	
7966	031252'	016	112			.BYTE	16,112	;IF V AT V = V GO TO LN
7967	031254'	012700	000000	012701		.WORD	012700,0,012701,0	
	031262'	000000						
7968	031264'	012702	000003	122122		.WORD	012702,3,122122,001005	
	031272'	001005						
7969	031274'	000302				.WORD	000302	
7970	031276'	005300	001373	000137		.WORD	005300,001373,000137,0	
	031304'	000000						
7971	031306'	017	117			.BYTE	17,117	;IF V AT V <> V GO TO LN
7972	031310'	012700	000000	012701		.WORD	012700,0,012701,0	
	031316'	000000						
7973	031320'	012702	000003	122122		.WORD	012702,3,122122,001004	
	031326'	001004						
7974	031330'	000302				.WORD	000302	
7975	031332'	005300	001373	000402		.WORD	005300,001373,000402,000137,0	
	031340'	000137	000000					
7976	031344'	003	244			.BYTE	3,244	;GO TO LN
7977	031346'	000137	000000			.WORD	000137,0	
7978	031352'	003	245			.BYTE	3,245	;LINK LN
7979	031354'	004737	000000			.WORD	004737,0	

7980	031360'	002	246		.BYTE	2,246		;RETURN
7981	031362'	000207			.WORD	000207		
7982	031364'	003	260		.BYTE	3,260		;NEGATE
7983	031366'	005437	000000		.WORD	005437,0		
7984	031372'	005	261		.BYTE	5,261		;PRINT * TEXT
7985	031374'	004537			.WORD	4537		
7986	031376'	040022'	031402'	000004	LRT21:	.WORD	LPRNG,.,+2,4	
7987	031404'	005	241		.BYTE	5,241		;PRINT * TEXT
7988	031406'	004537			.WORD	004537		
7989	031410'	040022'	031414'	000004	LRT3:	.WORD	LPRNG,.,+2,4	
7990	031416'	005	154		.BYTE	5,154		;PRINT V IN ASCII
7991	031420'	004537			.WORD	004537		
7992	031422'	040022'	000000	000005	LRT4:	.WORD	LPRNG,0,5	
7993	031430'	005	155		.BYTE	5,155		;PRINT V IN OCTAL
7994	031432'	004537			.WORD	004537		
7995	031434'	040126'	000000	000005	LRT5:	.WORD	LPRNO,0,5	
7996	031442'	005	156		.BYTE	5,156		;PRINT V IN DECIMAL
7997	031444'	004537			.WORD	004537		
7998	031446'	040114'	000000	000005	LRT6:	.WORD	LPRND,0,5	
7999	031454'	005	157		.BYTE	5,157		;PRINT V IN BINARY
8000	031456'	004537			.WORD	004537		
8001	031460'	040402'	000000	000005	LRT7:	.WORD	LPRNB,0,5	
8002	031466'	005	150		.BYTE	5,150		;PRINT V AT V IN ASCII
8003	031470'	004537			.WORD	004537		
8004	031472'	040022'	000000	000002	LRT8:	.WORD	LPRNG,0,2	
8005	031500'	005	151		.BYTE	5,151		;PRINT V AT V IN OCTAL
8006	031502'	004537			.WORD	004537		
8007	031504'	040126'	000000	000002	LRT9:	.WORD	LPRNO,0,2	
8008	031512'	005	152		.BYTE	5,152		;PRINT V AT V IN DECIMAL
8009	031514'	004537			.WORD	004537		
8010	031516'	040114'	000000	000002	LRT10:	.WORD	LPRND,0,2	
8011	031524'	005	153		.BYTE	5,153		;PRINT V AT V IN BINARY
8012	031526'	004537			.WORD	004537		
8013	031530'	040402'	000000	000002	LRT11:	.WORD	LPRNB,0,2	
8014	031536'	005	170		.BYTE	5,170		;FILL V WITH RANDOM
8015	031540'	004537			.WORD	004537		
8016	031542'	036216'	000000	000005	LRT12:	.WORD	LFILR,0,5	
8017	031550'	005	172		.BYTE	5,172		;FILL V WITH ASCII
8018	031552'	004537			.WORD	004537		
8019	031554'	036274'	000000	000005	LRT13:	.WORD	LFILG,0,5	
8020	031562'	006	174		.BYTE	6,174		;FILL V WITH V
8021	031564'	004537			.WORD	004537		
8022	031566'	036152'	000000	000005	LRT14:	.WORD	LFILV,0,5,0	
	031574'	000000						
8023	031576'	005	171		.BYTE	5,171		;FILL V AT V WITH RANDOM
8024	031600'	004537			.WORD	004537		
8025	031602'	036216'	000000	000002	LRT15:	.WORD	LFILR,0,2	
8026	031610'	005	173		.BYTE	5,173		;FILL V AT V WITH ASCII
8027	031612'	004537			.WORD	004537		
8028	031614'	036274'	000000	000002	LRT16:	.WORD	LFILG,0,2	
8029	031622'	006	175		.BYTE	6,175		;FILL V AT V WITH V
8030	031624'	004537			.WORD	004537		
8031	031626'	036152'	000000	000002	LRT17:	.WORD	LFILV,0,2,0	
	031634'	000000						
8032	031636'	006	176		.BYTE	6,176		;FILL V WITH PATN
8033	031640'	004537			.WORD	004537		

8034	031642'	036326'	000000	000005	LRT22:	.WORD	LFILP,0,5,0	
	031650'	000000						
8035	031652'	006	177			.BYTE	6,177	;FILL V AT V WITH PATN
8036	031654'	004537				.WORD	004537	
8037	031656'	036326'	000000	000002	LRT23:	.WORD	LFILP,0,2,0	
	031664'	000000						
8038	031666'	005	001			.BYTE	5,001	;ROTATE V
8039	031670'	004537				.WORD	004537	
8040	031672'	036732'	000007	000000	LRT18:	.WORD	LROTR,7,0	
8041	031700'	005	003			.BYTE	5,003	;ROTATE V AT V
8042	031702'	004537				.WORD	004537	
8043	031704'	036732'	000000	000002	LRT19:	.WORD	LROTR,0,2	
8044	031712'	004	243			.BYTE	4,243	;DELAY V
8045	031714'	004537				.WORD	004537	
8046	031716'	036764'	000000		LRT20:	.WORD	LDLAY,0	
8047	031722'	007	120			.BYTE	7,120	;READ
8048	031724'	004537				.WORD	004537	
8049	031726'	000012	000400	000006		.WORD	10.,00400,6,6,11.	
	031734'	000006	000013					
8050	031740'	007	121			.BYTE	7,121	;READ V
8051	031742'	004537				.WORD	004537	
8052	031744'	000012	000000	000006		.WORD	10.,0,6,6,11.	
	031752'	000006	000013					
8053	031756'	007	123			.BYTE	7,123	;READ V INTO V
8054	031760'	004537				.WORD	004537	
8055	031762'	000012	000000	000002		.WORD	10.,0,2,2,11.	
	031770'	000002	000013					
8056	031774'	007	125			.BYTE	7,125	;READ V INTO V FROM N
8057	031776'	004537				.WORD	004537	
8058	032000'	000012	000000	000002		.WORD	10.,0,2,2,4	
	032006'	000002	000004					
8059	032012'	007	130			.BYTE	7,130	;WRITE
8060	032014'	004537				.WORD	004537	
8061	032016'	000012	000400	000007		.WORD	10.,00400,7,7,11.	
	032024'	000007	000013					
8062	032030'	007	131			.BYTE	7,131	;WRITE V
8063	032032'	004537				.WORD	004537	
8064	032034'	000012	000000	000007		.WORD	10.,0,7,7,11.	
	032042'	000007	000013					
8065	032046'	007	133			.BYTE	7,133	;WRITE V FROM V
8066	032050'	004537				.WORD	004537	
8067	032052'	000012	000000	000002		.WORD	10.,0,2,2,11.	
	032060'	000002	000013					
8068	032064'	007	135			.BYTE	7,135	;WRITE V FROM V TO N
8069	032066'	004537				.WORD	004537	
8070	032070'	000012	000000	000002		.WORD	10.,0,2,2,4	
	032076'	000002	000004					
8071	032102'	006	143			.BYTE	6,143	;VERIFY V WITH V
8072	032104'	004537				.WORD	004537	
8073	032106'	037004'	000005	000000	LRT27:	.WORD	LVERFY,5,0,0	
	032114'	000000						
8074	032116'	006	145			.BYTE	6,145	;VERIFY V AT V WITH V
8075	032120'	004537				.WORD	004537	
8076	032122'	037004'	000000	000000	LRT28:	.WORD	LVERFY,0,0,0	
	032130'	000000						
8077	032132'	007	161			.BYTE	7,161	;BREAK V

8078	032134'	004537			.WORD	004537	
8079	032136'	000012	000000	000007	.WORD	10.,0,7,7,11.	
	032144'	000007	000013				
8080	032150'	007	163		.BYTE	7,163	;BREAK V ON V
8081	032152'	004537	000012		.WORD	4537,10.	
8082	032156'	000000	000007	000007	.WORD	0,7,7,4	
	032164'	000004					
8083		001			.IF NDF MM		
8084	032166'	004	251		.BYTE	4,251	;VECTOR V TO LN
8085	032170'	012737	000000	000002	.WORD	012737,0,2	
8086		000			.ENDC		
8087	032176'	003	252		.BYTE	3,252	;LETGO
8088	032200'	004537			.WORD	004537	
8089	032202'	035216'			LRT29: .WORD	CUPCR	
8090	032204'	007	253		.BYTE	7,253	;ENTRY
8091	032206'	010546	010446	010346	.WORD	10546,10446,10346	
8092	032214'	010246	010146	010046	.WORD	10246,10146,10046	
8093	032222'	010	254		.BYTE	8,254	;EXIT
8094	032224'	012600	012601	012602	.WORD	12600,12601,12602	
8095	032232'	012603	012604	012605	.WORD	12603,12604,12605,5	
	032240'	000005					
8096		001			.IF NDF MM		
8097	032242'	002	255		.BYTE	2,255	;RESET
8098	032244'	000011			.WORD	9.	
8099		000			.ENDC		
8100	032246'	002	262		.BYTE	2,262	;PAUSE
8101	032250'	000017			.WORD	15.	
8102	032252'	002	263		.BYTE	2,263	;WORD V
8103	032254'	000000			.WORD	0	
8104	032256'	003	240		.BYTE	3,240	;END (MUST BE LAST IN THIS TABLE)
8105	032260'	004537			.WORD	004537	
8106	032262'	035016'			LRT30: .WORD	CUPGEX	

8108
8109
8110
8111
8112
8113
8114
8115
8116
8117
8118
8119
8120
8121
8122
8123
8124
8125
8126
8127
8128
8129
8130
8131
8132
8133
8134
8135
8136
8137
8138
8139
8140
8141
8142
8143
8144
8145
8146
8147
8148
8149
8150
8151
8152
8153
8154
8155
8156
8157
8158
8159
8160
8161
8162
8163

.SBTTL MPG COMPILER

```

*****
THIS ROUTINE CONVERTS THE INTERNAL
PACKED CODE INTO MACHINE LANGUAGE
AND STORES IT IN MEMORY.

CALLING SEQUENCE - ENTERED FROM UPLP

ENTRY - R3 POINTS TO PROGRAM TABLE

EXIT - RETURNS TO CALLING PROGRAM (CONTROL ROUTINE)
*****

```

```

032264' 010446
032266' 010367 166776
032272' 042713 000100
032276' 016304 000022
      001
      000
032302' 016302 000024
032306' 116401 000001
032312' 010264 000004
032316' 010400
032320' 004567 175106
032324' 060002
032326' 122701 000240
032332' 001403
032334' 111400
032336' 060004
032340' 000762
      001
      000
032342' 010267 001564
      000
032346' 160302
032350' 010263 000026
      001
032354' 016767 147066 176250
032362' 016704 147062
032366' 005724
032370' 166704 147052
032374' 000241
032376' 006004
032400' 042704 000001
032404' 066704 147036
032410' 010467 176210
      000
032414' 010367 001514

```

```

LCPILE: MOV      R4, -(SP)          ;SAVE R4
          MOV      R3, LPKPTP      ;SAVE P TBL POINTER
          BIC      #0CPRES, (R3)   ;RESET OBJ CODE PRES FLG
          MOV      PSRCST(R3), R4  ;POINT R4 AT PACKED SOURCE
          .IF DF MM
          BIS      #P4CONS, R4     ;SELECT PAR4
          .ENDC
          MOV      POBJST(R3), R2  ;POINT R2 AT OBJECT ADDR
LCPLP1: MOVB     1(R4), R1          ;GET FUNCTION CODE
          MOV      R2, 4(R4)       ;STORE PHYSICAL ADDRESS
          MOV      R4, R0          ;SET PKED STMT PNTR IN R0
          JSR     R5, LCKOBL       ;GO FIND OBJECT LENGTH
          ADD      R0, R2          ;ADD TO OBJECT ADDRESS
          CMPB    #240, R1         ;WAS IT 'END'
          BEQ     LCPEFP          ;YES-END OF FIRST PASS
          MOVB    (R4), R0        ;GET PACKED LENGTH
          ADD      R0, R4          ;BUMP POINTER TO NEXT ENTRY
          BR      LCPLP1          ;DO IT AGAIN
          .IF DF MM
          .IFT
LCPEFP: BIS      #P4CONS, R2      ;SET PAR4 SELECT BITS
          MOV      R2, LCPOEN      ;STORE AS OBJ END
          .IFF
LCPEFP: MOV      R2, LCPOEN       ;STORE AS OBJ END
          .ENDC
          SUB      R3, R2          ;CALC PROG LENGTH
          MOV      R2, PLNGTH(R3)  ;AND PUT IN TABLE
          .IF NDF MM
          MOV      CFMST, LFREE    ;UPDATE SYMBOL TABLE
          MOV      CFMEND, R4      ;FREE MEM END ADDR TO R4
          TST     (R4)+           ;ADD 2 TO IT
          SUB      CFMST, R4       ;SUBTRACT FREE MEM START
          CLC      R4              ;DIVIDE BY 2
          ROR     R4
          BIC     #1, R4           ;MAKE IT AN EVEN ADDR
          ADD     CFMST, R4        ;CALC MIDDLE OF FREE
          MOV     R4, LMIDL        ;STORE IN SYMBOL TABLE
          .ENDC
          MOV     R3, LCPTPTS      ;SAVE R3

```

8164	032420'	016304	000022		MOV	PSRCST(R3),R4	;POINT R4 AT PACKED SOURCE
8165		001			.IF DF MM		
8166					BIS	#P4CONS,R4	;SELECT PAR4
8167		000			.ENDC		
8168	032424'	016367	000024	001506	MOV	POBJST(R3),LCPOBA	;SET OBJ ADDR=OBJ START
8169		001			.IF DF MM		
8170					BIS	#P4CONS,LCPOBA	;SELECT PAR4
8171		000			.ENDC		
8172	032432'	010703			LCPGNE: MOV	PC,R3	
8173	032434'	062703	176216		ADD	#LCMCTB-.,R3	;POINT R3 AT COMPILE TBL
8174	032440'	116401	000001		LCPGFC: MOV	1(R4),R1	;GET FUNCTION CODE
8175	032444'	122701	000241		CMPI	#241,R1	;WAS IT 'PRINT IMM'
8176	032450'	001003			BNE	LCPCNR	;NO-CONTINUE
8177	032452'	105067	001453		CLRB	LCPNRF	;CLEAR NO-RETURN FLAG
8178	032456'	000405			BR	LCPPIB	;YES-HANDLE SPECIAL CASE
8179	032460'	122701	000261		LCPENR: CMPI	#261,R1	;WAS IT PRINT WITH NO RETN
8180	032464'	001004			BNE	LCPFTE	;NO-CONTINUE
8181	032466'	105267	001437		INCB	LCPNRF	;SET NO-RETURN FLAG
8182	032472'	000167	001336		LCPPIB: JMP	LCPPRI	;GO HANDLE SPECIAL CASE
8183	032476'	122763	000240	000001	LCPFTE: CMPI	#240,1(R3)	;F.C. REPRESENTS END?
8184	032504'	001405			BEQ	LCPEIT	;YES - CHECK FURTHER
8185	032506'	126463	000001	000001	CMPI	1(R4),1(R3)	;COMPARE FUNCTION CODES
8186	032514'	001433			BEQ	LCPFFC	;BRANCH IF FOUND
8187	032516'	000426			BR	LCPBTP	;ELSE BUMP POINTER
8188	032520'	126463	000001	000001	LCPEIT: CMPI	1(R4),1(R3)	;COMPARE FUNCTION CODES
8189	032526'	001426			BEQ	LCPFFC	;MUST BE END
8190	032530'	016703	166534		MOV	LPKPTP,R3	;GET P TBL POINTER
8191	032534'	062703	000354		ADD	#PTLGTH+DVCTEP,R3	;POINT R2
8192	032540'	061303			ADD	(R3),R3	;AT COMPILE TBL EXTEN.
8193	032542'	122763	000377	000001	10\$: CMPI	#377,1(R3)	;END OF TBL?
8194	032550'	001410			BEQ	LCPBUG	;YES - MUST BE BUG
8195	032552'	126463	000001	000001	CMPI	1(R4),1(R3)	;COMPARE FUNCTION CODES
8196	032560'	001411			BEQ	LCPFFC	;BRANCH IF FOUND
8197	032562'	111300			MOV	(R3),R0	
8198	032564'	006300			ASL	R0	
8199	032566'	060003			ADD	R0,R3	;ELSE BUMP POINTER
8200	032570'	000764			BR	10\$;TRY AGAIN
8201	032572'	000000			LCPBUG: HALT		;YES-HALT (MPG BUG)
8202	032574'	111300			LCPBTP: MOV	(R3),R0	;GET ENTRY LENGTH
8203	032576'	006300			ASL	R0	;CONVERT TO BYTES
8204	032600'	060003			ADD	R0,R3	;BUMP POINTER TO NEXT ENTRY
8205	032602'	000735			BR	LCPFTE	;TRY AGAIN
8206	032604'	011467	001332		LCPFFC: MOV	(R4),LCPTM1	
8207	032610'	012714	177777		MOV	#177777,(R4)	
8208	032614'	010400			MOV	R4,R0	
8209	032616'	005200			INC	R0	
8210	032620'	010701			MOV	PC,R1	
8211	032622'	062701	001363		ADD	#LCPERL-.,R1	
8212	032626'	004567	006172		JSR	R5,DECASC	;PUT LINE # IN ERR MSG
8213	032632'	016714	001304		MOV	LCPTM1,(R4)	
8214	032636'	062704	000006		ADD	#6,R4	;ADJUST SOURCE POINTER
8215	032642'	112367	001270		MOV	(R3)+,LCPTM1	;GET ENTRY LENGTH
8216	032646'	112367	001272		MOV	(R3)+,LCPFCS	;SAVE FUNCTION CODE
8217	032652'	005367	001260		DEC	LCPTM1	;AND COUNT
8218	032656'	105067	001246		CLRB	LCPSWP	;CLEAR SWAP FLAG
8219	032662'	105067	001236		CLRB	LDVRF	;CLEAR DEVR INST FLAG

```

8220 032666' 011302          LCPGTE: MOV      (R3),R2          ;GET TABLE ENTRY
8221 032670' 105067 001231    CLRB     L2CODF          ;RESET 2 CODE FLAG
8222 032674' 020227 000017    CMP      R2,#15.
8223 032700' 101064          BHI      JCPTSF          ;IF >15 GO TEST SWAP FLAG
8224
8225                          ;PROCESS COMPILER CODES 0 THRU 15.
8226
8227 032702' 005702          LCPTG0: TST     R2
8228 032704' 001450          BEQ     JCPGNV          ;IF 0 GET NEXT VARIABLE
8229
8230 032706' 005302          LCPTG1: DEC     R2
8231 032710' 001005          BNE     LCPTG2
8232 032712' 112767 000001 001210  MOVB    #1,LCPSWP          ;SET SWAP FLAG
8233 032720' 012402          MOV     (R4)+,R2        ;PACKED ENTRY TO R2
8234 032722' 000453          BR      JCPTSF
8235
8236 032724' 005302          LCPTG2: DEC     R2
8237 032726' 001006          BNE     LCPTG3
8238 032730' 112767 000001 001172  MOVB    #1,LCPSWP          ;SET SWAP FLAG
8239 032736' 105267 001163    INCB    L2CODF          ;SET THE 2 CODE FLAG
8240 032742' 000431          BR      JCPGNV
8241
8242 032744' 005302          LCPTG3: DEC     R2
8243 032746' 001031          BNE     LCPTG4
8244 032750' 122714 000040    CMPB    #40,(R4)        ;IF FIRST BYTE BLANK
8245 032754' 001416          BEQ     LCPNT1          ;IT IS NOT A TAG
8246 032756' 116400 000001    MOVB    1(R4),R0        ;GET SECOND BYTE
8247 032762' 042700 000017    BIC     #17,R0          ;OF PACKED VARIABLE
8248 032766' 122700 000360    CMPB    #360,R0        ;IF LEFT 4 BITS ARE ONES
8249 032772' 001407          BEQ     LCPNT1          ;IT IS NOT A TAG
8250 032774' 012763 122122 000002  MOV     #122122,2(R3)    ;MODIFY NEXT TABLE ENTRY
8251 033002' 012763 000240 000006  MOV     #240,6(R3)      ;SET UP A NOP INST
8252 033010' 000406          BR      JCPGNV          ;GO GET NEXT VARIABLE
8253 033012' 012763 122102 000002  LCPNT1: MOV     #122102,2(R3) ;MODIFY NEXT TABLE ENTRY
8254 033020' 012763 000302 000006  MOV     #302,6(R3)      ;SET UP "SWAB R2" INST
8255 033026' 000167 000334    JCPGNV: JMP     LCPGNV   ;GO GET NEXT VARIABLE
8256
8257 033032' 005302          LCPTG4: DEC     R2
8258 033034' 001002          BNE     LCPTG5
8259 033036' 012402          MOV     (R4)+,R2        ;PACKED ENTRY TO R2
8260 033040' 000404          BR      JCPTSF
8261
8262 033042' 005302          LCPTG5: DEC     R2
8263 033044' 001004          BNE     LCPTG6
8264 033046' 012702 000002    MOV     #2,R2          ;SET R2=2
8265 033052' 000167 000650    JCPSTF: JMP     LCPTSF   ;GO TEST SWAP FLAG
8266
8267 033056' 005302          LCPTG6: DEC     R2
8268 033060' 001013          BNE     LCPTG7
8269 033062' 016702 001046    MOV     LCPPTS,R2
8270 033066' 062702 000016    ADD     #PRDIOA,R2      ;ADDRESS OF READ AREA TO R2
8271 033072' 010701          MOV     PC,R1          ;POINT R1
8272 033074' 062701 174270    ADD     #LTGADR-.,R1    ;AT ADDRESS AREA
8273 033100' 005011          CLR     (R1)          ;CLR MSW OF ADR
8274 033102' 011261 000002    MOV     (R2),2(R1)     ;UPDATE LSW OF ADR
8275
      .IF DF MM

```



```

8276                                     .IFF
8277 033106' 000420                       BR      LCP67E      ;GO SET UP CORRECT REG
8278                                     .IFT
8279                                     TSTB     NPRFLG     ;TEST IF NPR DEVICE
8280                                     BPL     LCP67E     ;IF NOT, SKIP
8281                                     ADD     #PRDIOX-PRDIOA,R2 ;POINT AT 18 BIT ADR
8282                                     BR      LCP67C     ;GO GET ADR & TEST FLAGS
8283                                     .ENDC
8284                                     000
8285 033110' 005302                       LCPTG7: DEC     R2
8286 033112' 001022                       BNE     LCPTG8
8287 033114' 016702 001014                MOV     LCPPTS,R2
8288 033120' 062702 000020                ADD     #PWRIA0,R2      ;ADDRESS OF WRITE AREA TO R2
8289 033124' 126727 001014 000001        CMPB   LCPFCS,#001     ;THIS THE 'ROTATE V' INST?
8290 033132' 001410                       BEQ     SU4ROT        ;N Y-SU4ROT
8291 033134' 010701                       MOV     PC,R1        ;POINT R1
8292 033136' 062701 174226                ADD     #LTGADR-. ,R1 ;AT ADDRESS AREA
8293 033142' 005011                       CLR     (R1)        ;CLR MSW OF ADR
8294 033144' 011261 000002                MOV     (R2),2(R1)  ;UPDATE LSW OF ADR
8295                                     001
8296                                     .IF DF MM
8297                                     TSTB     NPRFLG     ;TEST IF NPR DEVICE
8298                                     BPL     LCP67E     ;IF NOT, SKIP
8299                                     ADD     #PWRI0X-PWRIA0,R2 ;POINT AT 18 BIT ADR
8300 LCPT67C: MOV     (R2)+,(R1)
8301                                     MOV     (R2)+,2(R1)  ;UPDATE ADDRESS AREA
8302                                     BITB    #10,MASS70  ;TEST MASS BUS/70 FLAG
8303                                     BNE     LCP67E     ;IF SET, LEAVE ABS ADR ALONE
8304                                     BIT     #UNIMAP,CSYSFW ;TEST UNIBUS MAP FLAG
8305                                     BEQ     LCP67E     ;IF NOT SET, LEAVE ABS ADR ALONE
8306                                     MOV     (R2)+,(R1)  ;ELSE GET 18 BIT VADR
8307                                     MOV     (R2),2(R1)
8307                                     .ENDC
8308 033150' 010102                       LCP67E: MOV     R1,R2
8309 033152' 000543                       BR      ADR18B      ;GO STORE IN OBJ CODE
8310 033154' 011202                       SU4ROT: MOV     (R2),R2 ;GET WRIO'S 16 BIT ADR
8311 033156' 000502                       BR      BCPTSF      ;GO TEST SWAP FLAG
8312
8313 033160' 005302                       LCPTG8: DEC     R2
8314 033162' 001022                       BNE     LCPTG9
8315 033164' 122714 000040                CMPB   #40,(R4)     ;IF FIRST BYTE BLANK
8316 033170' 001413                       BEQ     LCPNT2      ;IT IS NOT A TAG
8317 033172' 116400 000001                MOVB   1(R4),R0    ;GET SECOND BYTE
8318 033176' 042700 000017                BIC    #17,R0      ;OF PACKED VARIABLE
8319 033202' 122700 000360                CMPB   #360,R0     ;IF LEFT 4 BITS SET
8320 033206' 001404                       BEQ     LCPNT2      ;IT IS NOT A TAG
8321 033210' 012761 023737 177774        MOV     #023737,-4(R1) ;MODIFY PREVIOUS CODE
8322 033216' 000463                       BR      LCPGNV      ;GO GET NEXT VARIABLE
8323 033220' 012761 023727 177774        LCPNT2: MOV    #023727,-4(R1) ;MODIFY PREVIOUS CODE
8324 033226' 000457                       BR      LCPGNV      ;GO GET NEXT VARIABLE
8325
8326 033230' 005302                       LCPTG9: DEC     R2
8327 033232' 001003                       BNE     LCPT10
8328 033234' 012702 000005                MOV     #5,R2
8329 033240' 000451                       BR      BCPTSF
8330
8331 033242' 005302                       LCPT10: DEC     R2

```

```

8332 033244' 001014          BNE      LCPT11
8333 033246' 116700 000672  MOV      LCPFCS,R0          ;PICK UP FUNCTION CODE
8334 033252' 016702 000656  MOV      LCPPTS,R2        ;PICK UP PT POINTER
8335 033256' 004567 174352  JSR      R5,FENTRY        ;FIND DEVICE ROUT ENTRY
8336 033262' 000416          .WORD   LCPABT-          ;GO HERE IF NOT THERE
8337 033264' 105267 000634  INCB    LDVRIF            ;SET THE DVR INST FLAG
8338 033270' 110167 000633  MOV      R1,NPRFLG        ;UPDATE NPR FLAG
8339          001          .IF DF MM
8340          BIT      #CPU70,CSYSFW          ;SYSTEM = 11/70?
8341          BEQ      10$                    ;NO, SKIP
8342          MOV      NPRFLG,R1              ;ELSE SET MASS70 FLAG
8343          BIC      #177767,R1            ;EQUAL TO
8344          MOV      R1,MASS70              ;MASS BUS DEV FLAG
8345          000          .ENDC
8346 033274' 000433 10$: BR      BCPTSF
8347
8348 033276' 005302  LCPT11: DEC     R2
8349 033300' 001002          BNE     LCPT12
8350 033302' 005002          CLR     R2                    ;SET R2 = 0
8351 033304' 000427          BR      BCPTSF
8352
8353 033306' 005302  LCPT12: DEC     R2
8354 033310' 001005          BNE     LCPT13
8355 033312' 016702 000616  MOV      LCPPTS,R2
8356          001          .IF DF MM
8357          BIC      #P4CONS,R2              ;SELECT PAR0
8358          000          .ENDC
8359 033316' 062702 000116  ADD      #PNBR,R2              ;ADDR OF NBR TO R2
8360 033322' 000420          BR      BCPTSF
8361
8362 033324' 005302  LCPT13: DEC     R2
8363 033326' 001005          BNE     LCPT14
8364 033330' 016702 000600  MOV      LCPPTS,R2
8365          001          .IF DF MM
8366          BIC      #P4CONS,R2              ;SELECT PAR0
8367          000          .ENDC
8368 033334' 062702 000120  ADD      #PSRC,R2              ;ADDR OF SRC TO R2
8369 033340' 000411          BR      BCPTSF
8370
8371 033342' 005302  LCPT14: DEC     R2
8372 033344' 001005          BNE     LCPT15
8373 033346' 016702 000562  MOV      LCPPTS,R2
8374          001          .IF DF MM
8375          BIC      #P4CONS,R2              ;SELECT PAR0
8376          000          .ENDC
8377 033352' 062702 000122  ADD      #PDST,R2              ;ADDR OF DST TO R2
8378 033356' 000402          BR      BCPTSF              ;GO TEST SWAP FLAG
8379
8380 033360' 012702 000001  LCPT15: MOV     #1,R2
8381 033364' 000560  BCPTSF: BR      LCPTSF          ;GO TEST SWAP FLAG
8382
8383          ;GET NEXT VARIABLE FROM PACKED SOURCE
8384
8385 033366' 122714 000040  LCPGNV: CMPB   #40,(R4)          ;IF FIRST BYTE IS A BLANK
8386 033372' 001452          BEQ     LCP0CT              ;VARIABLE IS AN OCTAL #
8387 033374' 116400 000001  MOV      1(R4),R0          ;GET SECOND BYTE

```

8388	033400'	042700	000017	BIC	#17,R0	;ZF PACKED VARIABLE
8389	033404'	122700	000360	CMPB	#360,R0	;IF LEFT FOUR BITS ARE ONES
8390	033410'	001451		BEQ	LCPNAL	;IT IS NOT A TAG
8391	033412'	010400		MOV	R4,R0	
8392	033414'	012701	000004	MOV	#4,R1	
8393	033420'	010367	000516	MOV	R3,LCPTM1	
8394	033424'	016703	000504	MOV	LCPPTS,R3	
8395	033430'	004567	173176	JSR	R5,TAGLUP	;LOOK UP TAG
8396	033434'	000234		.WORD	LCPEER-	
8397	033436'	016703	000500	MOV	LCPTM1,R3	
8398	033442'	062704	000004	ADD	#4,R4	;ADJUST SOURCE POINTER
8399	033446'	105767	000455	TSTB	NPRFLG	;TEST IF 2 WD ADR NEEDED
8400	033452'	001003		SNE	ADR188	;BRANCH IF YES
8401	033454'	016202	000002	MOV	2(R2),R2	;PUT TAG VALUE IN R2
8402	033460'	000522		BR	LCPTSF	;THEN GO TEST SWAP FLAG
8403						
8404						
8405						
8406	033462'	005367	000450	ADR188:	DEC	LCPTC
8407	033466'	005723		TST	(R3)+	;DECREMENT TBL END CTR
8408	033470'	105067	000434	CLRB	LCPSWP	;BUMP COMPILER TBL PTR
8409	033474'	105067	000427	CLRB	NPRFLG	
8410	033500'	016700	000434	MOV	LCPOBA,R0	;POINT R0 AT OBJ CODE
8411	033504'	014046		MOV	-(R0),-(SP)	;SAVE OLD CONTENTS
8412	033506'	012220		MOV	(R2)+,(R0)+	;STORE MSW OF ADR
8413	033510'	011220		MOV	(R2),(R0)+	;STORE LSW OF ADR
8414	033512'	012620		MOV	(SP)+,(R0)+	;PUT BYTE COUNT HERE
8415	033514'	010001		MOV	R0,R1	
8416	033516'	000517		BR	LCPTEN	;CONT LIKE ORIG MPG
8417						
8418						
8419						
8420	033520'	005204		LCPOCT:	INC	R4
8421	033522'	010400		MOV	R4,R0	
8422	033524'	004567	006402	JSR	R5,OCTBIN	;CONVERT OCTAL TO BINARY
8423	033530'	000140		.WORD	LCPEER-	
8424	033532'	000413		BR	LCK2WD	;GO CK IF 2 WORDS NEEDED
8425	033534'	121427	000114	LCPNAL:	CMPB	(R4),#'L
8426	033540'	001434		BEQ	LCPF1N	;IS FIRST CHAR AN 'L'
8427	033542'	121427	000104	CMPB	(R4),#'D	;YES-GO FIND LINE NBR
8428	033546'	001050		BNE	LCPEAR	;IS FIRST CHAR A 'D'
8429	033550'	005204		INC	R4	;NO-SOMETHINGS WRONG
8430	033552'	010400		MOV	R4,R0	
8431	033554'	004567	006336	JSR	R5,DECBIN	;CONVERT DECIMAL TO BINARY
8432	033560'	000110		.WORD	LCPEER-	
8433	033562'	062704	000003	LCK2WD:	ADD	#3,R4
8434	033566'	105767	000332	TSTB	LDVRIF	;THIS A DEV ROUT INST?
8435	033572'	001415		BEQ	10\$;Y,N-10\$
8436	033574'	105767	000327	TSTB	NPRFLG	;2'WORD ADR NEEDED?
8437	033600'	001412		BEQ	10\$;Y,N-10\$
8438	033602'	105767	000317	TSTB	L2CODF	;PROCESSING A 2 CODE?
8439	033606'	001407		BEQ	10\$;Y,N-10\$
8440	033610'	010702		MOV	PC,R2	;SET UP ADR OF ADR STORAGE
8441	033612'	062702	173552	ADD	#L1GADR-,R2	
8442	033616'	005012		CLR	(R2)	;CLEAR MSW OF ADR
8443	033620'	010062	000002	MOV	R0,2(R2)	;STORE LSW OF ADR

8444	033624'	000716			BR	ADR18B		;GO STORE 2 WORDS
8445	033626'	010002			10\$: MOV	RD,R2		
8446	033630'	000436			BR	LCPTSF		;GO TEST SWAP FLAG
8447	033632'	010402			LCPFLN: MOV	R4,R2		
8448	033634'	010367	000302		MOV	R3,LCPTM1		
8449	033640'	016703	000270		MOV	LCPPTS,R3		
8450	033644'	004567	173520		JSR	R5,LFINLN		:LOOK UP LINE NUMBER
8451	033650'	000020			.WORD	LCPERR-		
8452	033652'	016703	000264		MOV	LCPTM1,R3		
8453	033656'	062704	000004		ADD	#4,R4		
8454	033662'	016102	000004		MOV	4(R1),R2		
8455	033666'	000417			BR	LCPTSF		;GO TEST SWAP FLAG
8456								
8457	033670'	004567	152330		LCPERR: JSR	R5,CTYPE		;TYPE NON EXIST LINE MSG
8458	033674'	000402			.WORD	402		
8459	033676'	000250			.WORD	LCPEMS-		
8460	033700'	016701	000230		LCPABT: MOV	LCPPTS,R1		
8461	033704'	016100	000024		MOV	POBJST(R1),R0		
8462		001			.IF DF MM			
8463					BIS	#P4CONS,R0		;SELECT PAR4
8464		000			.ENDC			
8465	033710'	160100			SUB	R1,R0		;SUBTRACT OBJECT LENGTH
8466	033712'	010061	000026		MOV	R0,PLNGTH(R1)		;FROM PROGRAM LENGTH
8467	033716'	016100	000024		MOV	POBJST(R1),R0		;PASS FREE MEM START
8468		001			.IF DF MM			
8469					BIS	#P4CONS,R0		;SELECT PAR4
8470		000			.ENDC			
8471	033722'	012604			MOV	(SP)+,R4		;RESTORE R4
8472	033724'	000205			RTS	R5		;RETURN TO CALLING PGM
8473								
8474								;TEST THE SWAP FLAG
8475								
8476	033726'	016701	000206		LCPTSF: MOV	LCPOBA,R1		;RECOVER OBJECT ADDRESS
8477	033732'	105767	000172		TSTB	LCPSWP		;TEST THE SWAP FLAG
8478	033736'	001406			BEQ	LCPNSW		;BRANCH IF NO SWAP
8479	033740'	014100			MOV	-(R1),R0		;GET PREVIOUS WORD CONTENTS
8480	033742'	010221			MOV	R2,(R1)+		;STORE NEW CODE AT PREV. ADR
8481	033744'	010021			MOV	R0,(R1)+		;STORE OLD CODE A NEW ADR
8482	033746'	105067	000156		CLRB	LCPSWP		;RESET SWAP FLAG
8483	033752'	000401			BR	LCPTEN		;GO TEST FOR TABLE END
8484	033754'	010221			LCPNSW: MOV	R2,(R1)+		;STORE NEW CODE AT NEW ADR
8485	033756'	010167	000156		LCPTEN: MOV	R1,LCPOBA		;PUT BACK NEXT OBJ ADR
8486	033762'	005367	000150		DEC	LCPTEN		;DECREMENT TABLE ENTRY CNT
8487	033766'	001404			BEQ	LCPTOE		;IF ZERO-TEST OBJ END
8488	033770'	062703	000002		ADD	#2,R3		;BUMP TABLE POINTER
8489	033774'	000167	176666		JMP	LCPGTE		;GO CHECK NEXT ENTRY
8490	034000'	026767	000134	000124	LCPTOE: CMP	LCPOBA,LCPOEN		;COMPILE COMPLETE
8491	034006'	001402			BEQ	LCPOUT		;YES-GET OUT
8492	034010'	000167	176416		JMP	LCPGNE		;NO-CHECK NEXT PACKED ENTRY
8493	034014'	016701	000114		LCPOUT: MOV	LCPPTS,R1		
8494	034020'	052711	000100		BIS	#OCPRES,(R1)		;SET OBJ CODE PRESENT FLAG
8495	034024'	016700	000102		MOV	LCPOEN,R0		;PASS FREE MEM START
8496	034030'	012604			MOV	(SP)+,R4		;RESTORE R4
8497	034032'	000205			RTS	R5		;RETURN TO CALLING PGM
8498								
8499								;SPECIAL HANDLING FOR PRINT * AND PRINT *

```

8500
8501 034034' 016701 000100 LCPPRI: MOV LCPOBA,R1 ;GET CURRENT OBJ ADDR
8502 034040' 012721 004537 MOV #4537,(R1)+ ;STORE JSR
8503 034044' 016721 175340 MOV LRT3,(R1)+ ;TO PRINT ASCII PGM
8504 034050' 010100 MOV R1,R0
8505 034052' 062700 000004 ADD #4,R0
8506 001 .IF DF MM
8507 BIC #P4CONS,R0 ;SELECT PARO
8508 000 .ENDC
8509 034056' 010021 MOV RO,(R1)+ ;STORE INLINE ADDRESS
8510 034060' 111400 MOVB (R4),R0
8511 034062' 162700 000006 SUB #6,R0 ;CALCULATE BYTE COUNT
8512 034066' 010011 MOV RO,(R1) ;STORE IT
8513 034070' 005321 DEC (R1)+ ;DECR BY 1
8514 034072' 062704 000006 ADD #6,R4
8515 034076' 105767 000027 TSTB LCPNRF ;TEST NO-RETURN FLAG
8516 034102' 001402 BEQ LCPMTX ;CONTINUE IF NOT SET
8517 034104' 005461 177776 NEG -2(R1) ;OTHERWISE COMPLIMENT BC
8518 034110' 112421 LCPMTX: MOVB (R4)+,(R1)+ ;MOVE TEXT INLINE
8519 034112' 005300 DEC RO
8520 034114' 001375 BNE LCPMTX
8521 034116' 010167 000016 MOV R1,LCPOBA ;SAVE NEXT OBJ ADDR
8522 034122' 000726 BR LCPTOE ;GO TEST FOR OBJ END
8523
8524 034124' 000 LDVRIF: .BYTE 0 ;DEVICE ROUTINE INST FLAG
8525 034125' 000 L2CODF: .BYTE 0 ;COMPILER 2 CODE FLAG
8526 034126' 000 MASS70: .BYTE 0 ;MASS BUS DEV ON 11/70 FLAG
8527 034127' 000 NPRFLG: .BYTE 0 ;NPR FLAG ADDRESS
8528 034130' 000 LCPSWP: .BYTE 0 ;SWAP FLAG
8529 034131' 000 LCPNRF: .BYTE 0 ;PRINT NO RETURN FLAG
8530 034132' 000000 LCPOEN: .WORD 0 ;OBJECT END ADDRESS
8531 034134' 000000 LCPPTS: .WORD 0 ;PROG TABLE POINTER SAVE
8532 034136' 000000 LCPTEC: .WORD 0 ;TABLE ENTRY COUNT
8533 034140' 000000 LCPOBA: .WORD 0 ;OBJECT CODE ADDRESS
8534 034142' 000000 LCPTM1: .WORD 0
8535 034144' 000000 LCPFCS: .WORD 0 ;FUNCTION CODE SAVE
8536
8537 034146' 047516 026516 054105 LCPEMS: .ASCII /NON-EXISTENT LINE REF ON LINE /
034154' 051511 042524 052116
034162' 046040 047111 020105
034170' 042522 020106 047117
034176' 046040 047111 020105
034204' 040
8538 034205' 040 040 040 LCPERL: .BYTE 40,40,40,40,0,0
034210' 040 000 000
8539 034214' .EVEN
8540 001 .IF DF MM
8541
8542
8543 ALPT1= .-CLOCZ
8544 ALFCT1= ALPT1&000077/2
8545 002 .IF NE ALFCT1
8546 ALQUN1= 31.-ALFCT1
8547 .BLKW ALQUN1 ;ALIGN TO A 32 WORD BNDRY
8548 .WORD 0
8549 001 .ENDC

```

8550	000	.ENDC		
8551				
8552	001	.IF DF MM		
8553		.IFF		
8554		.TITLE	MAINDEC-11-DTMGA-C	MPG / USER PROGRAM SHARED CODE
8555		.IFT		
8556		.TITLE	MAINDEC-11-DTMMA-B	MPG / USER PROGRAM SHARED CODE
8557	000	.ENDC		

```

8559          .SBT'L  SHARED CODE CONSTANTS
8560
8561
8562          000020      MAXPRG= 16.          ;MAXIMUM # OF USER PROGRAMS
8563
8564          001        .IF DF MM
8565          .IFT
8566          PG6BA:      ;SHARED CODE PAGE BASE ADDRESS
8567
8568
8569          USTKWK: .BLKW  MAXPRG+1*4      ;USER PROGRAM'S STACK
8570          USRSTK:
8571
8572
8573          CSYSFW: .WORD  1          ;MPG'S SYSTEM FLAGWORD
8574          .IFF
8575          034214' 000000      CSYSFW: .WORD  0          ;MPG'S SYSTEM FLAGWORD
8576          000        .ENDC
8577
8578          000001      MMVER= 1          ;MPG MEMORY MANAGEMENT VERSION
8579          000002      USMTPS= 2        ;CPU REQUIRES MTPS/MFPS INST'S
8580          000004      LSI11= 4        ;CPU IS AN LSI-11
8581          000010      CPU70= 10       ;CPU = 11/70 (MASSBUS USES ABS ADRS)
8582          001        .IF DF MM
8583          CPU40= 20          ;CPU = 11/40 (NO MMMR1 OR MMMR3)
8584          UNIMAP= 40        ;USE THE UNIBUS MAP
8585          BITS22= 100      ;USE 22 BIT ADDRESSING
8586          .ENDC
8587          000200      CACHE= 200       ;CACHE MEMORY PRESENT IN CPU
8588          001000      CAIDLE= 1000    ;ALL USER PROGRAMS IDLE FLAG
8589          002000      CUSPGR= 2000   ;USER PROG RUNNING FLAG
8590          004000      CHPINT= 4000   ;'HOLD PRINTER'S INT ENB SET' FLG
8591          010000      CNTRLO= 10000  ;SUPPRESS TYPING - CNTRL/O FLAG
8592          020000      CTUDED= 20000  ;PRINT DEVICE DEDICATED FLAG
8593          040000      CRUDED= 40000  ;REQUEST USER DEDICATED RESET
8594          100000      CTRQBP= 100000 ;CONSOLE REQUEST BEING PROCESSED
8595
8596
8597          034216'      CRTBL: .REPT  MAXPRG      ;*;CONTROL ROUTINE TABLE
8598          .WORD  0          ;*;ONE WORD PER ENTRY -
8599          .ENDR          ;*;EACH ENTRY CONTAINS 16 BIT ADR OF
8600          ;*;PROG TBL FOR RESIDENT USER PROG
8601          034256' 177777      .WORD  177777    ;TABLE TERMINATOR
8602          001        .IF DF MM
8603
8604          CRTEXT= MAXPRG+1*2      ;TABLE EXTENSION ACCESS VALUE
8605
8606          CTBLEX: .REPT  MAXPRG      ;CONTROL ROUTINE TABLE EXTENSION
8607          .WORD  0          ;BIT 15 = 1 = FIXED ADR PROG
8608          .ENDR          ;BITS 8 - 0 = PROG AREA LENGTH
8609          FIXADR= 100000          ;      (# OF 32 WD BLKS)
8610
8611
8612
8613          FMPAR3: .WORD  XXXX          ;VALUES FOR USER'S PAR'S 3 THRU 7
8614          FMPAR4: .WORD  XXXX          ;FREE MEM PAR'S

```

8615		FMPAR5: .WORD	XXXX	:
8616		SPPAR6: .WORD	XXXX	; SHARED PAGE PAR
8617		IOPAR7: .WORD	007600	; I/O PAGE PAR
8618				
8619		.WORD	0,0,0	; FILL WORDS
8620				
8621				; VALUES FOR USER'S PDR'S 3 THRU 7
8622		FMPDR3: .WORD	XXXX	; FREE MEM PDR'S
8623		FMPDR4: .WORD	XXXX	:
8624		FMPDR5: .WORD	XXXX	:
8625		SPPDR6: .WORD	XXXX	; SHARED PAGE PDR
8626		IOPDR7: .WORD	077406	; I/O PAGE PDR
9627	000	.ENDC		
8628		.EVEN		
8629	034260' 025040	.ASCII	/ */	
8630	034262' 000144	_INBUF: .BLKB	100.	; PROGRAM STATEMENT BUFFER
8631	034426' 005015	.ASCII	<15><12>	
8632	034430' 005015	LCRLF: .ASCII	<15><12>	; CARRIAGE RETN, LINE FEED
8633	034432' 000000	.WORD	0	

.SBTTL USER PROGRAM DISPATCHER

```

*****
THE PROGRAM DISPATCHER ROUTINE WILL BE ENTERED FROM THE
COMMAND DECODE ROUTINE WHEN A NULL KEYBOARD ENTRY OR A R/JN CMND
IS ENTERED. IT WILL SCAN THE PROGRAM SLOTS AND LOOK FOR
A USER PROGRAM THAT IS AVAILABLE FOR EXECUTION. IF FOUND
CONTROL WILL BE TRANSFERRED TO THAT USER PROGRAM'S OBJECT
CODE. IF A PROGRAM IS NOT AVAILABLE, SCANNING OF THE PROGRAM
SLOTS WILL CONTINUE AS LONG AS A PROGRAM IS WAITING FOR
I/O TERMINATION. IF NONE ARE WAITING FOR TERMINATION OR
AVAILABLE FOR EXECUTION, RETURN WILL BE MADE TO THE
COMMAND DECODE ROUTINE WHICH REQUESTS A COMMAND ENTRY.
*****

```

```

8635
8636
8637
8638
8639
8640
8641
8642
8643
8644
8645
8646
8647
8648
8649
8650
8651
8652
8653 034434' 010705
8654 034436' 062705 177556
8655 034442' 052715 001000
8656 034446' 010704
8657 034450' 062704 177546
8658 034454' 005367 000006
8659 034460' 100003
8660 034462' 012727
8661 034464' 000000
8662 034466' 000000
8663 034470' 010427
8664 034472' 000000
8665 034474' 005714
8666 034476' 001442
8667 001
8668
8669 034500' 011403
8670
8671
8672
8673
8674 000
8675 034502' 005713
8676 034504' 100037
8677 034506' 005027
8678 034510' 000000
8679 034512' 032713 000004
8680 034516' 001032
8681 034520' 032713 000002
8682 034524' 001403
8683 034526' 032713 000040
8684 034532' 001424
8685 034534' 042715 001000
8686 034540' 052715 002000
8687 034544' 032713 000030
8688 034550' 001430
8689 034552' 032713 000020
8690 034556' 001012

```

```

CPGDSP: MOV PC,R5 ;SET UP ADR OF SYSTEM FLGWD
ADD #CSYSFW-,R5
BIS #CAIDLE,(R5) ;SET 'ALL IDLE' FLAG
MOV PC,R4 ;SET UP START ADR OF CNTRL ROUT TBL
ADD #CRTBL-,R4
DEC CTOCK ;DECR T/O CHECK VALUE
BPL CPGTTE ;IS IT EXHAUSTED? (Y,N-CPGTTE)
MOV (PC)+,(PC)+ ;RESTORE ITS VALUE

CTOAdj: .WORD 0
CTOCK: .WORD 0
CPGTTE: MOV R4,(PC)+ ;SAVE CURRENT CNTR ROUT TBL ADR
CPGCPE: .WORD 0 ;CURRENT CNTRL ROUT TBL ENTRY ADR
TST (R4) ;THIS PROG SLOT IN USE?
BEQ CPGNXT ;Y,N-CPGNXT
.IF DF MM
.IFF
MOV (R4),R3 ;GET PROG TBL ADR
.IFT
MOV (R4),#KPAR4 ;MAP KPAR 4 TO THIS PROG
MOV #P4CONS,R3 ;SET UP ITS VIRTUAL ADR
MOV SPPAR6,#KPAR6 ;MAP KPAR 6 TO SHARED CODE
.ENDC
TST (R3) ;IS PROG'S ACTIVE FLAG SET?
BPL CPGNXT ;Y,N-CPGNXT
CLR (PC)+ ;RESET T/O ERR ROUT ADR

TOEADR: .WORD 0
BIT #ERSTOP,(R3) ;ERROR STOPPED THIS PROG?
BNE CPGNXT ;N,Y-CPGNXT
BIT #URSTOP,(R3) ;USER STOP THIS PROG?
BEQ IOS ;Y,N-IOS
BIT #SETDED,(R3) ;PRT DEV STILL DEDICATED TO THIS PROG?
BEQ CPGNXT ;Y,N-CPGNXT
IOS: BIC #CAIDLE,(R5) ;RESET THE 'ALL IDLE' FLAG
BIS #CUSPRG,(R5) ;SET 'USER PROG RUNNING' FLAG
BIT #WT4IOT+CTPRIO,(R3) ;PROG WAITING FOR I/O TERM?
BEQ CEXPRG ;Y,N-CEXPGR
BIT #CTPRIO,(R3) ;THIS A CONSOLE OR PRINTER I/O?
BNE CPGNXT ;N,Y-CPGNXT

```

8691	034560'	032763	000040	000002	BIT	#DOIOT,POPSW(R3)	:DO I/O TIME OUT?
8692	034566'	001006			BNE	CPGNXT	:Y,N-CPGNXT
8693	034570'	005767	177672		TST	CTOCK	:CHECK T/O THIS LOOP?
8694	034574'	001003			BNE	CPGNXT	:Y,N-CPGNXT
8695	034576'	005363	000030		DEC	PTOCNT(R3)	:DECR TIME OUT CNT BY 1
8696	034602'	001455			BEQ	CPGTOE	:TIMED OUT? (N,Y-CPGTOE)
8697	034604'	005724			CPGNXT: TST	(R4)+	:INCR TO NXT CNTR ROUT TBL ENTRY
8698	034606'	022714	177777		CMP	#177777,(R4)	:END OF THE TABLE?
8699	034612'	001326			BNE	CPGTTE	:Y,N-CPGTTE
8700	034614'	032715	001000		BIT	#CAIDLE,(R5)	: 'ALL IDLE' FLAG SET?
8701	034620'	001705			BEQ	CPGDSP	:Y,N-CPGDSP
8702	034622'	042715	002000		BIC	#CUSPGR,(R5)	:RESET 'USER PROG RUNNING' FLAG
8703	034626'	000167	143516		JMP	CFCMND	:GO ASK FOR A CMND ENTRY
8704							
8705		001				.IF DF MM	
8706						.IFF	
8707	034632'	062703	000124		CEXPRG: ADD	#PSTKCT,R3	:GET ADR OF STACK STORAGE CNT
8708						.IFT	
8709					CEXPRG: MOV	R3,R0	:POINT AT USER'S PAR'S STORAGE
8710					ADD	#PUPARS,R0	
8711					MOV	#UPARO,R1	:SET UP PAR & PDR REG'S ADR'S
8712					MOV	#UPDRO,R2	
8713					MOV	#8,R5	:SET UP REG CNT
8714				12\$:	MOV	16.(R0),(R2)+	:LOAD USER PDR
8715					MOV	(R0)+,(R1)+	:LOAD USER PAR
8716					DEC	R5	:DONE ALL REGS?
8717					BNE	12\$:Y,N-12\$
8718					ADD	#PSTKCT,R3	:GET ADR OF STK STORAGE CNT
8719		000				.ENOC	
8720	034636'	020627	001200		CMP	SP,#STACK	:**** STACK POINTER CORRECT?
8721	034642'	001401			BEQ	15\$:**** N,Y-15\$
8722	034644'	000000			HALT		:****
8723	034646'	012300			15\$: MOV	(R3)+,R0	:GET # OF WORDS SAVED OFF OF THE STACK
8724	034650'	060300			ADD	R3,R0	:POINT PAST LAST WORD
8725		001				.IF DF MM	
8726						.IFF	
8727	034652'	020300			20\$: CMP	R3,R0	:LOADED ALL WORDS ONTO STACK?
8728	034654'	001402			BEQ	30\$:N,Y-30\$
8729	034656'	014046			MOV	-(R0),-(SP)	:MOVE PROG'S WORD ONTO STACK
8730	034660'	000774			BR	20\$:GO CK IF DONE
8731	034662'	005040			30\$: CLR	-(R0)	:CLEAR OLD STACK COUNT
8732	034664'	011427			MOV	(R4),(PC)+	:SAVE CURR PROG TBL ADR
8733	034666'	000000			CPGCPT: .WORD	0	:CURRENT PROG TABLE ADR
8734	034670'	010305			MOV	R3,R5	:MOVE PRG TBL PNTR TO WK REG
8735	034672'	062705	000074		ADD	#PSVREG-PSTKSV,R5	:POINT AT SAVED REG'S
8736	034676'	012500			MOV	(R5)+,R0	:LOAD USER REGISTERS
8737	034700'	012501			MOV	(R5)+,R1	
8738	034702'	012502			MOV	(R5)+,R2	
8739	034704'	012503			MOV	(R5)+,R3	
8740	034706'	012504			MOV	(R5)+,R4	
8741	034710'	012546			MOV	(R5)+,-(SP)	
8742	034712'	011505			MOV	(R5),R5	
8743						.IFT	
8744					MOV	PC,R1	:SET UP ADR OF USER STACK AREA
8745					ADD	#USRSTK-. ,R1	
8746					MOV	#USRSTK-2-PG68A+P6CONS,USERR6	:SET UP STK'S VIRTUAL ADR

8747				20\$:	CMP	R3,R0		;LOADED ALL WORDS ON STK?
8748					BEQ	30\$;N,Y-30\$
8749					MOV	-(R0),-(R1)		;MOVE WORD ONTO USER STK
8750					SUB	#2,USERR6		;UPDATE USER STK PNTR
8751					BR	20\$;GO CK IF DONE
8752				30\$:	CLR	-(R0)		;CLEAR OLD STK CNT
8753					MOV	(R4),(PC)+		;SAVE CURR PROG TBL ADR
8754				CPGCPT:	.WORD	0		;CURRENT PROG TABLE ADR
8755					MOV	R3,R5		;MOVE PROG TBL PNTR TO WK REG
8756					ADD	#PSVREG-PSTKSV,R5		;POINT AT SAVED REG'S
8757					MOV	10,(R5),-(R1)		;PUT USER'S R5 ON HIS STACK
8758					MOV	(R5)+,R0		;LOAD USER REGISTERS 0 - 4
8759					MOV	(R5)+,R1		
8760					MOV	(R5)+,R2		
8761					MOV	(R5)+,R3		
8762					MOV	(R5)+,R4		
8763					TST	(R5)+		;BYPASS ALREADY STORED R5
8764					MOV	(R5),R5		;LOAD USER'S PC INTO R5
8765					MOV	#USRPSW,-(SP)		;LOAD USER MODE PSW ON STK
8766					MOV	#USMENT-PG68A+P6CONS,-(SP)		;PUT ENTRY POINT ADR ON STK
8767					RTI			;GO INTO USER MODE IN-LINE
8768				USMENT:	MOV	(PC)+,R6		;LOAD USER'S STK PNTR
8769				USERR6:	.WORD	0		
8770					.IFTF			
8771	034714'	005767	177570		TST	TOEADR		;IS THIS FOR T/O ERR ROUT?
8772	034720'	001001			BNE	40\$;N,Y-40\$
8773	034722'	000205			RTS	R5		;GO TO USER'S PROG
8774								
8775	034724'	004577	177560	40\$:	JSR	R5,@TOEADR		;GO TO USER'S T/O ERR ROUT
8776	034730'	004567	000270		JSR	R5,CPGRSV		;SAVE PROG'S INFO
8777	034734'	000516			BR	CKEX		;GO SET UP STANDARD ADRS
8778		000			.ENDC			
8779								
8780	034736'	010300		CPGTOE:	MOV	R3,R0		;SET UP ADR OF T/O ROUT ADR
8781	034740'	062700	000306		ADD	#PTLGTH+DTEAD,R0		
8782	034744'	005710			TST	(R0)		;THIS DEV HAVE TIMEOUT ROUT?
8783	034746'	001716			BEQ	CPGNXT		;Y,N-CPGNXT
8784	034750'	061000			ADD	(R0),R0		;MAKE ITS ADR ABS
8785		001			.IF DF MM			
8786					BIC	#P4CONS,R0		;ADJ ADR TO USER SPACE
8787		000			.ENDC			
8788	034752'	010067	177532		MOV	R0,TOEADR		;STORE T/O ERR ROUT ADR
8789	034756'	000725			BR	CEXPGR		;GO TO USER PROG

.SBT'L USER PROGRAM RETURN ROUTINES

```

*****
THE RETURN ROUTINES PROVIDE VARIOUS ENTRY POINTS WHICH WILL SUSPEND
USER PROGRAM EXECUTION FOR EITHER ONE POLLING LOOP, UNTIL AN
I/O COMPLETES, OR INDEFINITELY. ALSO PROVIDED ARE END OF PROGRAM
AND ERROR DETECTED ENTRY POINTS. THESE ENTRY POINTS ARE AS
FOLLOWS:

CUPCR = USER PROGRAM CONTROL RELEASE
CIOBSY = USER I/O DEVICE IS ALREADY BUSY AT INITIATION
CUPGEX = END OF USER PROGRAM
CUPGER = THE USER PROGRAM'S DEVICE ROUTINE
HAS DETECTED AN ERROR.
*****
    
```

:USER PROGRAM ERROR RETURN POINT

8791								
8792								
8793								
8794								
8795								
8796								
8797								
8798								
8799								
8800								
8801								
8802								
8803								
8804								
8805								
8806								
8807								
8808								
8809								
8810								
8811								
8812								
8813	034760'	004567	000240		CUPGER:	JSR	R5,CPGRSV	;SAVE REG'S & SET UP ADR'S
8814	034764'	052713	000004			BIS	#ERSTOP,(R3)	;SET ERROR STOP FLAG
8815	034770'	032763	100000	000002		BIT	#STONER,POPSW(R3)	;STOP ON ERROR IN OPSW?
8816	034776'	001075				BNE	CKEX	;N,Y-CKEX
8817	035000'	042713	000004			BIC	#ERSTOP,(R3)	;RESET ERROR STOP IN FLGWD
8818	035004'	032763	001000	000002		BIT	#GTNXTD,POPSW(R3)	;GO TO NEXT DEVICE ON ERROR?
8819	035012'	001467				BEQ	CKEX	;Y,N-CKEX
8820	035014'	000406				BR	CKPDVP	;GO SET UP FOR NEXT DEVICE
8821								
8822								
8823								
8824								
8825								

:USER PROGRAM TERMINATION RETURN POINT

8826	035016'	004567	000202		CUPGEX:	JSR	R5,CPGRSV	;SAVE REG'S AND SET UP ADR'S
8827	035022'	032763	040000	000002		BIT	#CYCPRG,POPSW(R3)	;CYCLE PROG ON CURR DEV?
8828	035030'	001043				BNE	CKCLRG	;N,Y-CKCLRG
8829	035032'	126327	000034	000017	CKPDVP:	CMPB	PDPNTR(R3),#15.	;DEVICE # PNTR AT LAST DEV?
8830	035040'	103430				BLO	30\$;Y,N-30\$
8831	035042'	032763	002000	000002	20\$:	BIT	#CYCDVL,POPSW(R3)	;CYCLE DEVICE LIST?
8832	035050'	001403				BEQ	22\$;Y,N-22\$
8833	035052'	105063	000034			CLRB	PDPNTR(R3)	;SET DEV PNTR TO FIRST DEV
8834	035056'	000430				BR	CKCLRG	;GO RESTART AT FIRST DEV
8835	035060'	032763	000020	000002	22\$:	BIT	#AUTORP,POPSW(R3)	;DISPLAY COUNTS AUTOMATICALLY?
8836	035066'	001002				BNE	24\$;Y,N-24\$
8837	035070'	004767	000402			JSR	PC,CRPTEP	;GO ISSUE THE REPORT
8838	035074'	032763	000001	000002	24\$:	BIT	#NOCOMP,POPSW(R3)	;PRINT PROG COMPLETED MSG?
8839	035102'	001004				BNE	28\$;Y,N-28\$
8840	035104'	004567	000512			JSR	R5,ULISTM	;ISSUE PROG XX COMPLETED MSG
8841	035110'	000456				.WORD	PGCOMP-	
8842	035112'	177767				.WORD	-9.	
8843	035114'	042713	100000		28\$:	BIC	#ACTIVE,(R3)	;RESET THIS PROG'S ACTIVE FLAG
8844	035120'	000424				BR	CKEX	;GO TO ROUTINE EXIT
8845	035122'	105263	000034		30\$:	INCB	PDPNTR(R3)	;INCR DEV # PNTR
8846	035126'	004767	000236			JSR	PC,CPGCDN	;POINT AT NEXT DEV #

```

8847 035132' 122712 000377      CMPB   #377,(R2)      ;IS IT A DEV #?
8848 035136' 001741      BEQ    20$           ;Y,N-20$
8849 035140' 032763 000030 000002 CKCLRG: BIT   #AJTORP+AURPEP,POPSW(R3) ;DO AUTO DISPLAY OF COUNTS?
8850 035146' 001002      BNE    40$           ;Y,N-40$
8851 035150' 004767 000322      JSR    PC,CRPTEP    ;GO ISSUE THE REPORT
8852 035154' 004767 000210      JSR    PC,CPGCDN    ;POINT AT NEXT DEV #
8853 035160' 111263 000035      MOVB   (R2),PCURDV(R3) ;STORE IT AS CURR DEV #
8854 035164' 004567 000214      JSR    R5,CPGHSK    ;GO DO PROG HOUSEKEEPING
8855 035170' 000000      .WORD  0           ; HSKP CNTS PER OPSW FLAG
8856          001      .IF DF MM
8857          .IFT
8858          MOV    PC,R5      ;SET THE REMAP FLAG
8859          BR    CKEX1      ;GO SET UP ADRS
8860
8861          CKEX:  CLR    R5      ;RESET THE REMAP FLAG
8862          CKEX1: TRAP   2      ;GO INTO KERNEL MODE IN-LINE
8863          UPGRET: ADD    #4,SP  ;REMOVE TRAP'S PSW & PC
8864          MOV    CPGCPE,R4  ;GET CURR CNTRL ROUT TBL ADR
8865          MOV    (R4),J#KPAR4 ;MAP PAR 4 TO THIS USER
8866          MOV    #P4CONS,R3 ;SET UP HIS VIRTUAL ADR
8867          TST   R5          ;REMAP HIS PAR/PDR STORAGE
8868          BEQ   50$        ;Y,N-50$
8869          JSR   PC,MAPUPG   ;MAP PAR/PDR STORAGE
8870          50$:  MOV    PC,R5  ;SET UP SYSTEM FLGWD ADR
8871          ADD   #CSYSFW-.,R5
8872          .IFF
8873          035172' 016704 177274      CKEX:  MOV    CPGCPE,R4  ;GET CURR CNTRL ROUT TBL ADR
8874          035176' 010705      MOV    PC,R5          ;SET UP SYSTEM FLGWD ADR
8875          035200' 062705 177014      ADD   #CSYSFW-.,R5
8876          .ENDC
8877          035204' 005067 177456      CLR   CPGCPT        ;CLEAR CURR PROG TBL ADR
8878          035210' 000167 177370      JMP   CPGNXT        ;GO CHECK NEXT PROG ENTRY
8879
8880
8881          ;USER PROGRAM I/O BUSY, CONTROL
8882          ;RELEASE, AND KILL RETURN POINTS
8883
8884
8885          001
8886          .IF DF MM
8887          CKILEK: MOV   #STACK,R6      ;RESTORE KERNEL'S STK PNTR
8888          MOV   #USRPSW,-(SP)        ;LOAD USER MODE PSW ON STK
8889          MOV   #CKILEX-PG6BA+P6CONS,-(SP) ;SET UP KILL ABORT ADR
8890          RTI          ;GO INTO USER MODE FOR THE KILL
8891          .ENDC
8892          035214' 005746      CKILEX: TST   -(SP)      ;ADD A DUMMY WORD ON THE STACK
8893
8894          CUPCR:
8895          CIOSBY: JSR   R5,CPGRSV      ;SAVE REG'S & SET UP ADR'S
8896          BR    CKEX      ;GO CHECK NXT PROG SLOT
8897
8898          ;
8899          ;*****
8900          ; * USER PROGRAM DISPATCHER & RETURN SUBROUTINES *
8901          ; *
8902          ;*****

```

```

8903
8904
8905                :SAVE USER REGISTERS & STACK INFO
8906
8907                :JSR    R5,CPGRSV          S/R CALL
8908
8909 035224' 010527  CPGRSV: MOV    R5,(PC)+      :SAVE RETURN ADR
8910 035226' 000000  CPSVRS: .WORD  0
8911                .IF DF MM
8912                .IFF
8913 035230' 016705 177432  MOV    CPGCPT,R5          ;GET CURRENT PROG TBL ADR
8914                .IFT
8915                CLR    R5          ;GET VIRTUAL PROG TBL ADR
8916                .ENDC
8917 035234' 062705 000236  ADD    #PUSRPC,R5        ;POINT AT PC STORAGE AREA
8918 035240' 012615          MOV    (SP)+,(R5)        ;SAVE PROG'S PC
8919 035242' 012645          MOV    (SP)+,-(R5)       ;SAVE PROG'S R5
8920 035244' 010445          MOV    R4,-(R5)         ;SAVE R4 THRU R0
8921 035246' 010345          MOV    R3,-(R5)
8922 035250' 010245          MOV    R2,-(R5)
8923 035252' 010145          MOV    R1,-(R5)
8924 035254' 010045          MOV    R0,-(R5)
8925 035256' 010503          MOV    R5,R3          ;PROG TBL PNTR TO R3
8926 035260' 162703 000222  SUB    #PSVREG,R3
8927                .IF DF MM
8928                .IFF
8929 035264' 012700 001200  MOV    #STACK,R0        ;SET UP STACK PNTR MAX VALUE
8930                .IFT
8931                MOV    #USRSTK-PG6BA+P6CONS,R0 ;SET UP STK PNTR MAX VALUE
8932                .ENDC
8933 035270' 160600          SUB    SP,R0          ;GET # OF WORDS ON STACK
8934 035272' 100015          BPL    70$           ;STACK ADR SCREWED UP? (Y,N-70$)
8935 035274' 012767 000212 000012  MOV    #INVSTK-60$,60$ ;STORE INV STK ERR MSG ADR
8936 035302' 010645          MOV    SP,-(R5)       ;SAVE BAD STK PNTR
8937                .IF DF MM
8938                .IFF
8939 035304' 012706 001200  MOV    #STACK,R6        ;REINITIALIZE STK PNTR
8940                .IFT
8941                MOV    #USRSTK-PG6BA+P6CONS,R6 ;REINITIALIZE STK PNTR
8942                .ENDC
8943 035310' 004567 000306          JSR    R5,ULISTM        ;ISSUE "INV STK ADR" ERROR MSG OR
8944 035314' 000000          .WORD  XXXX
8945 035316' 177760          .WORD  -16.
8946 035320' 042713 100000          BIC    #ACTIVE,(R3)    ;RESET THIS PROG'S ACTIVE FLAG
8947 035324' 000722          BR     CKEX           ;GO CK FOR ANOTHER USER PROG
8948 035326' 020027 000074          CMP    R0,#PSVREG-PSTKSV ;TOO MANY WORDS ON STACK?
8949 035332' 101404          BLOS  80$           ;Y,N-80$
8950 035334' 012767 000232 177752  MOV    #STKBIG-60$,60$ ;STORE STK TOO BIG ERR MSG ADR
8951 035342' 000757          BR     50$           ;GO SHUT DOWN THIS PROG
8952 035344' 162705 000076          SUB    #PSVREG-PSTKCT,R5 ;POINT TO STACK CNT STORAGE
8953 035350' 010025          MOV    R0,(R5)+       ;STORE STACK COUNT
8954 035352' 162700 000002          SUB    #2,R0          ;DECR STACK WORD CNT
8955 035356' 100402          BMI    100$         ;ALL WORDS OFF STACK? (N,Y-100$)
8956 035360' 012625          MOV    (SP)+,(R5)+   ;STORE WORD OFF STACK
8957 035362' 000773          BR     90$           ;GO DECR CNT
8958                .IF DF MM

```

```

8959      .IFF
8960 035364' 016707 177536      100$: MOV      CPSVR5,PC      ;EXIT TO IN-LINE ADR
8961      .IFT
8962      100$: MOV      R3,RO      ;GET ADR OF PAR STORAGE
8963      ADD      #PUPARS,RO
8964      MOV      #UPARO,R1      ;GET ADRS OF USER'S MODE
8965      MOV      #UPDRO,R2      ;PAR & PDR REGS
8966      MOV      #8,R5      ;SET UP REG CNT
8967      110$: MOV      (R2)+,16.(RO) ;STORE PDR VALUE
8968      MOV      (R1)+,(RO)+      ;STORE PAR VALUE
8969      DEC      R5      ;DONE ALL REGS?
8970      BNE      110$      ;Y,N-110$
8971      MOV      CPSVR5,PC      ;RETURN IN-LINE
8972      COO
8973
8974
8975      ;GET ADR IN R2 OF UNIT # POINTED TO BY "PDPNTR"
8976
8977      ;JSR      PC,CPGCDN      S/R CALL
8978      ;R3 CONTAINS PROG TBL ADR
8979      ;DESTROYS R2
8980
8981 035370' 116302 000034      CPGCDN: MOV      PDPNTR(R3),R2      ;GET DEV PNTR
8982 035374' 060302      ADD      R3,R2      ;SET UP DEV # ADR
8983 035376' 062702 000036      ADD      #PDNUMS,R2
8984 035402' 000207      RTS      PC      ;EXIT
8985
8986
8987
8988      ;HOUSEKEEP PROG TBL & OPTIONALLY THE DEV ROUT
8989
8990      ;JSR      R5,CPGHSK      S/R CALL
8991      ;.WORD 0 OR 1      0 = DO DEV ROUT HSKP PER OPSW
8992      ;      1 = UNCOND. DO DEV ROUT HSKP
8993      ;R3 CONTAINS PROG TEL ADR
8994      ;DESTROYS R0,R1,R2
8995
8996 035404' 010300      CPGHSK: MOV      R3,RO      ;PROG TBL ADR TO WK REG
8997 035406' 062700 000056      ADD      #PTEMO,RO      ;POINT AT TMOO WORD
8998 035412' 012701 000070      MOV      #PUSRPC-PTEMO/2,R1 ;GET # OF WORDS TO CLEAR
8999 035416' 005020      10$: CLR      (RO)+      ;CLEAR TMP, STACK, & REG
9000 035420' 005301      DEC      R1      ;STORAGE FOR THIS PROG
9001 035422' 001375      BNE      10$
9002 035424' 016320 000024      MOV      POBJST(R3),(RO)+ ;SET PROG'S PC TO OBJ CODE START
9003 035430' 062700 000004      ADD      #PTSIZE-PUSRPC+DEVFWD,RO ;POINT AT DEV ROUT FLGWD
9004 035434' 012701 000011      MOV      #DEVTRA-DEVFWD/2,R1 ;GET # OF DEV ROUT WDS TO CLEAR
9005 035440' 005020      20$: CLR      (RO)+      ;CLEAR DEV ROUT FLAGWORD &
9006 035442' 005301      DEC      R1      ;INTERFACE WORDS
9007 035444' 001375      BNE      20$
9008 035446' 012567 000020      MOV      (R5)+,35$      ;STORE HOUSEKEEP FLAGWD
9009 035452' 062700 000010      30$: ADD      #DHKPAD-DEVTRA,RO ;POINT AT DEV ROUT HSKP ROUT ADR
9010 035456' 005710      TST      (RO)      ;IS THERE A HSKP ROUT?
9011 035460' 001405      BEQ      40$      ;Y,N-40$
9012 035462' 061000      ADD      (RO),RO
9013 035464' 016302 000002      MOV      POPSW(R3),R2      ;SET UP PROG'S OPSW
9014 035470' 004510      JSR      R5,(RO)      ;GO TO DEVICE'S HSKP ROUT

```

```

9015 035472' 000000      35$: .WORD   XXXX      ;HOUSEKEEP FLAGWD
9016 035474' 000205      40$:  RTS     R5       ;EXIT IN-LINE
9017
9018
9019
9020
9021
9022
9023
9024
9025
9026 035476' 012767 000001 000016 CRPTEP: MOV    #1,CRPFLG      ;SET UP END OF PASS FLGWD
9027 035504' 010300      CGORPT: MOV    R3,R0      ;POINT TO ADR OF REPORT ROUT
9028 035506' 062700 000300      ADD    #PTLGTH+DERPAD,R0
9029 035512' 005710      TST    (R0)             ;IS THERE A ROUT ADR?
9030 035514' 001403      BEQ    CGOREX           ;Y,N-CGOREX
9031 035516' 061000      ADD    (R0),R0         ;SET UP ABS ADR OF REPORT ROUT
9032 035520' 004510      JSR    R5,(R0)        ;GO TO DEV ROUT ERROR REPORT S/R
9033 035522' 000000      CRPFLG: .WORD   XXXX
9034 035524' 000207      CGOREX: RTS     PC     ;EXIT IN-LINE
9035
9036
9037 035526' 042452 025122 044440 INVSTK: .ASCII /*ER* INV STK ADR/
9038 035546' 042452 025122 051440 STKBIG: .ASCII /*ER* STK TOO BIG/
9039 035566' 047503 050115 042514 PGCOMP: .ASCII /COMPLETED/
9040
9041      035600'          .LIST   BEX
          .EVEN

```



```

9043          .SBTTL FORMATS FOR PROGRAM & DEVICE ROUTINE TABLES
9044
9045          ; PROGRAM TABLE FORMAT
9046
9047          001          .IF DF MM
9048          .IFF
9049          000242      PTLGTH= 162. ;PROGRAM TABLE LENGTH
9050          .IFT
9051          000          PTLGTH= 212. ;PROGRAM TABLE LENGTH (MEM MGMNT VERSION)
9052          .ENDC
9053
9054          000000      PFLGWD= +0. ;PROGRAM FLAG WORD - 1 WORD
9055
9056          000002      URSTOP= 2 ; 1 = USER HAS STOPPED THIS PROGRAM
9057          000004      ERSTOP= 4 ; 1 = AN ERROR HAS STOPPED THIS PROGRAM
9058          000010      WT4IOT= 10 ; 1 = WAITING FOR I/O TERMINATION
9059          000020      CTPRIO= 20 ; 1 = CONSOLE OR PRINTER I/O IN PROGRESS
9060          000040      SETDED= 40 ; 1 = THIS PROG SET THE PRT DEV DEDICATED FLAG
9061          000100      OCPRES= 100 ; 1 = OBJ CODE IS PRESENT
9062          001          .IF DF MM
9063          000          USEUBM= 200 ; 1 = THIS PROG USES THE UNIBUS MAP
9064          .ENDC
9065          100000      ACTIVE= 100000 ; 1 = PROGRAM IS ACTIVE (SPECIFIED FOR EXECUTION)
9066
9067          000002      POPSW= +2. ;PROGRAM'S OPERATION SWITCHES - 1 WORD
9068
9069          100000      STONER= 100000 ; 1 = STOP PROG EXECUTION UPON ERROR
9070          040000      CYCPRG= 40000 ; 1 = CYCLE PROGRAM (ON CURRENT DEVICE)
9071          020000      PRONER= 20000 ; 1 = DO NOT PRINT ON ERROR
9072          010000      BIT12= 10000 ; 0 = NOT USED
9073          004000      BIT11= 4000 ; 0 = NOT USED
9074          002000      CYCDVL= 2000 ; 1 = CYCLE THE DEVICE LIST
9075          001000      GTNXTD= 1000 ; 1 = CYCLE ON SAME DEVICE UPON ERROR
9076          000400      CLERCK= 400 ; 1 = DON'T DO ERROR CHECKING
9077          000200      SPOPER= 200 ; 1 = DEVICE SPECIAL OPERATION
9078          000100      BIT6= 100 ; 0 = NOT USED
9079          000040      DOIOT= 40 ; 1 = DO NOT PERFORM I/O TIMEOUT
9080          000020      AUTORP= 20 ; 1 = DO NOT AUTOMATICALLY DISPLAY COUNTS
9081          000010      AURPEP= 10 ; 1 = AUTO DISPLAY COUNTS AT END OF FINAL PASS ONLY
9082          000004      HSKPEP= 4 ; 1 = HOUSEKEEP COUNTS ONLY AT RUN COMMAND
9083          000002      PFBBOV= 2 ; 1 = PRINT FIRST BAD BYTE ONLY ON VERIFY
9084          000001      NOCOMP= 1 ; 1 = DO NOT PRINT PROG COMPLETED MSG
9085
9086          000004      PFWADR= +4. ;*;PROGRAM FLAGWORD ADDRESS - 1 WORD
9087
9088          000006      PASCIN= +6. ;PROGRAM'S NUMBER IN ASCII - 1 WORD
9089
9090          000010      PNAME= +8. ;PROGRAM'S NAME IN ASCII - 6 BYTES
9091
9092          000016      PRDIOA= +14. ;ADDRESS OF READ I/O AREA - 1 WORD
9093
9094          000020      PWRIOA= +16. ;ADDRESS OF WRITE I/O AREA - 1 WORD
9095
9096          000022      PSRCST= +18. ;SOURCE STATEMENTS START ADDRESS - 1 WORD
9097
9098          000024      POBJST= +20. ;OBJECT CODE START ADDRESS - 1 WORD
    
```

K15

MAINDEC-11-DTMGA-C MPG / USER PROGRAM SHARED CODE MACY11 27(732) 24-SEP-76 13:54 PAGE 50-1
 DTMGAC.P11 FORMATS FOR PROGRAM & DEVICE ROUTINE TABLES

SEQ 0193

9099			
9100	000026	PLNGTH= +22.	;PROG AREA LENGTH (OBJ END MINUS PROG TBL START) - 1 WORD
9101			
9102	000030	PTCNT= +24.	;I/O TIMEOUT COUNT - 1 WORD
9103			
9104	000032	PMDLCD= +26.	;DEV ROUT MODEL # CODE - 1 WORD
9105			
9106	000034	PCPNTR= +28.	;CURRENT DEVICE NUMBER POINTER - 1 BYTE
9107			
9108	000035	PCURDV= +29.	;CURRENT DEVICE # - 1 BYTE
9109			
9110	000036	PDNUMS= +30.	;DEVICE NUMBERS - 16 BYTES
9111			
9112	000056	PTEM0= +46.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9113			
9114	000060	PTEM1= +48.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9115			
9116	000062	PTEM2= +50.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9117			
9118	000064	PTEM3= +52.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9119			
9120	000066	PTEM4= +54.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9121			
9122	000070	PTEM5= +56.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9123			
9124	000072	PTEM6= +58.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9125			
9126	000074	PTEM7= +60.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9127			
9128	000076	PTEM8= +62.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9129			
9130	000100	PTEM9= +64.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9131			
9132	000102	PTEM10= +66.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9133			
9134	000104	PTEM11= +68.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9135			
9136	000106	PTEM12= +70.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9137			
9138	000110	PTEM13= +72.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9139			
9140	000112	PTEM14= +74.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9141			
9142	000114	PTEM15= +76.	;USER PROGRAM TEMPORARY STORAGE - 1 WORD
9143			
9144	000116	PNBR= +78.	;NUMBER OF BYTES TO TRANSFER ON MOVE (NBR) - 1 WORD
9145			
9146	000120	PSRC= +80.	;DATA SOURCE ADDRESS ON MOVE (SRC) - 1 WORD
9147			
9148	000122	PDST= +82.	;DATA DESTINATION ADDRESS ON MOVE (DST) - 1 WORD
9149			
9150	000124	PSTKCT= +84.	;# OF WORDS (X 2) SAVED OFF STACK - 1 WORD
9151			
9152	000126	PSTKSV= +86.	;STACK WORDS STORAGE AREA - 30 WORDS
9153			
9154	000222	PSVREG= +146.	;USER'S R0 THRU R5 REGISTERS STORAGE AREA - 6 WORDS

```

9155
9156      000236      PUSRPC= +158.      ;USER'S CURRENT PROGRAM COUNTER - 1 WORD
9157
9158      001          .IF DF MM
9159      .IFF
9160      000240      PFSIZE= +160.      ;PROGRAM TABLE SIZE IN BYTES - 1 WORD
9161
9162      000242      PTEND= +162.      ;END OF PROGRAM TABLE
9163      .IFT
9164      PRDIOX= +160. ;18/22 BIT ABSOLUTE ADDRESS OF READ I/O AREA - 2 WORDS
9165
9166      PRDIOV= +164. ;18 BIT VIRTUAL ADDRESS OF READ I/O AREA - 2 WORDS
9167
9168      PWRIOX= +168. ;18/22 BIT ABSOLUTE ADDRESS OF WRITE I/O AREA - 2 WORDS
9169
9170      PWRIOV= +172. ;18 BIT VIRTUAL ADDRESS OF WRITE I/O AREA - 2 WORDS
9171
9172      PUPARS= +176. ;STORAGE AREA FOR USER'S PAR'S 0 THRU 7 - 8 WORDS
9173
9174      PUPDRS= +192. ;STORAGE AREA FOR USER'S PDR'S 0 THRU 7 - 8 WORDS
9175
9176      PUBMAP= +208. ;1ST UNIBUS MAP REG # AND # OF REGS USED - 1 WORD
9177
9178      PFSIZE= +210. ;PROGRAM TABLE SIZE IN BYTES - 1 WORD (MEM MGMNT VERSION)
9179
9180      PTEND= +212. ;END OF PROGRAM TABLE (MEM MGMNT VERSION)
9181      000          .ENDC

```

```

9183          :      DEVICE ROUTINE TABLE
9184
9185          000116      DRTLTH= 78.      ;DEVICE ROUTINE TABLE LENGTH
9186
9187          :
9188          000000      DEVRSZ= +0.      ;DEVICE ROUTINE SIZE IN BYTES - 1 WORD
9189
9190          000002      DEVFWD= +2.      ;DEVICE ROUTINE FLAGWORD - 1 WORD
9191
9192          000004      DEVIW1= +4.      ;DEVICE INTERFACE WORD # 1 - 1 WORD
9193
9194          000006      DEVIW2= +6.      ;DEVICE INTERFACE WORD # 2 - 1 WORD
9195
9196          000010      DEVIW3= +8.      ;DEVICE INTERFACE WORD # 3 - 1 WORD
9197
9198          000012      DEVIW4= +10.     ;DEVICE INTERFACE WORD # 4 - 1 WORD
9199
9200          000014      DEVIW5= +12.     ;DEVICE INTERFACE WORD # 5 - 1 WORD
9201
9202          000016      DEVIW6= +14.     ;DEVICE INTERFACE WORD # 6 - 1 WORD
9203
9204          000020      DEVIW7= +16.     ;DEVICE INTERFACE WORD # 7 - 1 WORD (SIZE)
9205
9206          000022      DEVIW8= +18.     ;DEVICE INTERFACE WORD # 8 - 1 WORD (ERR)
9207
9208          000024      DEVVRA= +20.     ;DEVICE REGISTERS ADDRESS - 1 WORD
9209
9210          000026      DEVIVA= +22.     ;DEVICE INTERRUPT VECTOR ADDRESS - 1 WORD
9211
9212          000030      DEVRPS= +24.     ;DEVICE READ PROCESSOR STATUS WORD (BUS REQ) - 1 WORD
9213
9214          000032      DEVPWS= +26.     ;DEVICE WRITE PROC STATUS WORD (BUS REQ) - 1 WORD
9215
9216          000034      DHKPAD= +28.     ;DEVICE ROUT HOUSEKEEPING ROUT REL ENTRY ADR - 1 WORD
9217
9218          000036      DERPAD= +30.     ;DEVICE ROUT REPORT ROUT REL ENTRY ADR - 1 WORD
9219
9220          000040      DKILAD= +32.     ;DEVICE ROUT KILL ROUTINE REL ENTRY ADR - 1 WORD
9221
9222          000042      DECTAD= +34.     ;DEVICE ROUT ERROR COUNTER REL ADR - 1 WORD
9223
9224          000044      DTOEAD= +36.     ;DEVICE ROUT TIMEOUT ERR ROUT REL ENTRY ADR - 1 WORD
9225
9226          000046      DEVI0B= +38.     ;DEVICE I/O BUSY BRANCH ADDRESS (CIOBSY) - 1 WORD
9227
9228          000050      DEVDER= +40.     ;DEVICE ERROR BRANCH ADDRESS (CUPGER) - 1 WORD
9229
9230          000052      DVUPRT= +42.     ;USER MODE PRINT BRANCH ADDRESS (ULIST) - 1 WORD
9231
9232          000054      DVCPRN= +44.     ;CMND MODE PRINT BRANCH ADDRESS (CLIST) - 1 WORD
9233
9234          000056      DEVBTA= +46.     ;CONVERT BINARY TO ASCII BR ADR (BINASC) - 1 WORD
9235
9236          000060      DVBTDA= +48.     ;CONVERT BINARY TO DECIMAL ASCII BR ADR (BTASLZ) - 1 WORD
9237
9238

```

9239	000062	DV7DTA= +50.	;CONVERT PACKED DECIMAL TO ASCII BR ADR (DECASC) - 1 WORD
9240			
9241	000064	DVSFW= +52.	;MPG SYSTEM FLAGWORD ADDRESS (CSYSFW) - 1 WORD
9242			
9243	000066	DVSVEC= +54.	;SET INTERRUPT VECTOR BR ADR (SETVEC) - 1 WORD
9244			
9245	000070	DVCVEC= +56.	;CLEAR INTERRUPT VECTOR BR ADR (CLRVEC) - 1 WORD
9246			
9247	000072	DVTVEC= +58.	;TEST INTERRUPT VECTOR BR ADR (TSTVEC) - 1 WORD
9248			
9249	000074	DVRINT= +60.	;RETURN FROM INTERRUPT BR ADR (RTNINT) - 1 WORD
9250			
9251	000076	DVGETB= +62.	;GET DATA BYTE BR ADR (GETBYT) - 1 WORD
9252			
9253	000100	DVPUTB= +64.	;PUT DATA BYTE BR ADR (PUTBYT) - 1 WORD
9254			
9255	000102	DEVSTP= +66.	;DEVICE ROUT REL SYMBOL TABL F POINTER - 1 WORD
9256			
9257	000104	DEVETP= +68.	;DEVICE ROUT REL ENTRY TABLE POINTER - 1 WORD
9258			
9259	000106	DVPTEP= +70.	;PACK TABLE EXTEN. REL POINTER - 1 WORD
9260			
9261	000110	DVVTEP= +72.	;VECTOR TABLE EXTEN. REL POINTER - 1 WORD
9262			
9263	000112	DVCTEP= +74.	;COMPILER TBL EXTEN. REL POINTER - 1 WORD
9264			
9265	000114	DVIWSP= +76.	;DEVICE INTERFACE WORD SYMBOL TBL REL POINTER - 1 WORD
9266			
9267	000116	DRTEND= +78.	;END OF DEVICE ROUTINE TABLE

.SBTTL LIST CONTROL ROUTINES

```

*****
THIS ROUTINE DETERMINES IF A LIST DEVICE HAS BEEN ASSIGNED
AND THEN BRANCHES TO EITHER THE CONSOLE TERMINAL ROUTINE
OR TO THE PRINTER ROUTINE.

LINKAGE:      JSR      R5,CLIST      MPG LIST
              .WORD   TAG-          REL ADR OF I/C AREA
              .WORD   COUNT        # OF CHARS TO TRANSFER

EXITS TO EITHER: CTWR OR CPRT

*****

```

```

92769
92770
92771
92772
92773
92774
92775
92776
92777
92778
92779
92780
92781
92782
92783
92784
92785
9286 035600' 016700 142210
9287 035604' 005760 000004
9288 035610' 001002
9289 035612' 000177 142222
9290 035616' 000177 142176
9291
9292
9293
9294
9295
9296
9297
9298
9299 035622' 004567 002770
9300 035626' 010567 000040
9301 035632' 062567 000034
9302 035636' 000404
9303
9304 035640' 004567 002752
9305 035644' 012567 000022
9306 035650' 011567 000020
9307 035654' 100402
9308 035656' 005467 000012
9309 035662' 105067 002035
9310 035666' 004567 002134
9311 035672' 000000
9312 035674' 000000
9313 035676' 005725
9314 035700' 100403
9315 035702' 004567 002404
9316 035706' 000404
9317 035710' 105067 001757
9318 035714' 004567 002364
9319 035720' 000205

```

```

CLIST:  MOV      CLISTI,RO      ;GET ADR OF LIST DEV
        TST      4(RO)          ;IS THERE AN ALT LIST DEV?
        BNE     10$            ;N,Y-10$
        JMP     2CTWR          ;GO TO MPG'S CONS TERM WRITE
10$:    JMP     2CPRT          ;GO TO MPG'S PRINT

```

;USER LIST S/R FOR DEVICE ROUTINES

```

;JSR      R5,ULIST          S/R CALL
;.WORD   ADR              ABS DATA ADR
;.WORD   CNT              BYTE COUNT

ULISTM: JSR      R5,LTSTDE      ;GO TEST DEDICATED FLAGS
        MOV     R5,UPRADR      ;GET REL MSG ADR & MAKE
        ADD    (R5)+,UPRADR    ;IT ABS
        BR     ULSTCM         ;GO TO COMMON POINT

ULIST:  JSR      R5,LTSTDE      ;GO TEST DEDICATED FLAGS
        MOV     (R5)+,UPRADR   ;GET ABS DATA ADR
ULSTCM: MOV     (R5),UPRBYC    ;GET BYTE COUNT
        BMI    10$           ;CNT NEG? (N,Y-10$)
        NEG   UPRBYC         ;NEGATE THE BYTE COUNT
10$:    CLRB   LPRRDE         ;RESET REALLY DEDICATED FLAG
        JSR   R5,UPRASC      ;ISSUE MSG USING PRINT ASCII ROUT

UPRADR: .WORD   XXXX
UPRBYC: .WORD   XXXX

        TST   (R5)+         ;WAS SUPPLIED BYTE CNT NEG?
        BMI  20$           ;N,Y-20$
        JSR  R5,UPCRLF      ;ISSUE <CR> & <LF>
        BR   30$           ;GO TO EXIT
20$:    CLRB   LPRRDE         ;RESET REALLY DEDICATED FLAG
        JSR  R5,ULCRLF      ;ISSUE <CR> & <LF> FOR LAST MSG
30$:    RTS    R5           ;EXIT TO USER PROG

```

```

9321 .SBTTL SUBROUTINES USED BY USER PROGRAM STATEMENTS
9322
9323
9324
9325
9326 035722 012500 LADDM: MOV (R5)+,R0 ;GET COUNT
9327 035724 012501 MOV (R5)+,R1 ;FIRST VARIABLE POINTER
9328 035726 012502 MOV (R5)+,R2 ;SECOND VARIABLE POINTER
9329 035730 060001 ADD R0,R1 ;POINTERS
9330 035732 060002 ADD R0,R2 ;TO LSB
9331 035734 005067 000056 CLR LCARRY ;CLEAR SOFTWARE
9332 035740 000241 CLC ;AND HARDWARE CARRY BITS
9333 035742 162701 000002 SUB #2,R1
9334 035746 162702 000002 SUB #2,R2
9335 035752 111103 LADDM: MOV (R1),R3 ;GET
9336 035754 111204 MOV (R2),R4 ;OPERANDS (ONE BYTE)
9337 035756 066704 000034 ADD LCARRY,R4 ;AND PREVIOUS CARRY
9338 035762 103001 BCC LADDC ;IF NO CARRY CLEAR SOFT CARRY
9339 035764 000402 BR LADD
9340 035766 005067 000024 LADDC: CLR LCARRY
9341 035772 060304 LADD: ADD R3,R4 ;ADD TWO OPERANDS
9342 035774 005567 000016 ADC LCARRY ;ADD ANY CARRY TO SOFT CARRY
9343 036000 110412 MOV R4,(R2) ;PUT BACK RESULT OPERAND
9344 036002 005300 DEC R0
9345 036004 001403 BEQ LADDEX ;EXIT WHEN DONE
9346 036006 004567 000102 JSR R5,BUMP ;GET NEXT MSB LIKE REAL COMPUTER
9347 036012 000757 BR LADML
9348 036014 000205 LADDEX: RTS R5 ;RETURN
9349 036016 000000 LCARRY: .WORD 0
9350
9351
9352 SUBTRACT MULTIPLE SUBROUTINE
9353 036020 012500 LSUBM: MOV (R5)+,R0 ;GET COUNT
9354 036022 012501 MOV (R5)+,R1 ;FIRST VARIABLE POINTER
9355 036024 012502 MOV (R5)+,R2 ;SECOND VARIABLE POINTER
9356 036026 060001 ADD R0,R1 ;POINTERS
9357 036030 060002 ADD R0,R2 ;TO LSB
9358 036032 005067 177760 CLR LCARRY ;CLEAR SOFTWARE
9359 036036 000241 CLC ;AND HARDWARE CARRIES
9360 036040 162701 000002 SUB #2,R1
9361 036044 162702 000002 SUB #2,R2
9362 036050 111103 LSUBML: MOV (R1),R3 ;GET
9363 036052 111204 MOV (R2),R4 ;OPERANDS (ONE BYTE)
9364 036054 166704 177736 SUB LCARRY,R4 ;SUBTRACT PREVIOUS CARRY
9365 036060 103001 BCC LSUBCC ;IF NO CARRY CLEAR SOFT CARRY
9366 036062 000402 BR LSUB
9367 036064 005067 177726 LSUBCC: CLR LCARRY
9368 036070 160304 LSUB: SUB R3,R4 ;SUBTRACT TWO OPERANDS
9369 036072 005567 177720 ADC LCARRY ;REMEMBER ANY CARRY
9370 036076 110412 MOV R4,(R2) ;PUT BACK RESULT OPERAND
9371 036100 005300 DEC R0
9372 036102 001403 BEQ LSUBEX ;EXIT WHEN DONE
9373 036104 004567 000004 JSR R5,BUMP ;GET NEXT MSB LIKE REAL COMPUTER
9374 036110 000757 BR LSUBML
9375 036112 000205 LSUBEX: RTS R5 ;RETURN

```

```

9377      :
9378      :      BUMP R0 AND R1 TO NEXT MSB SUBROUTINE
9379      :
9380 036114' 032702 000001      BUMP:  BIT      #1,R2      ;IF BIT 0 RESET
9381 036120' 001403      BEQ      LBUMP1      ;GO ADD ONE
9382 036122' 162702 000003      SUB      #3,R2      ;OTHERWISE SUBTRACT 3
9383 036126' 000401      BR      LBUMP2
9384 036130' 005202      LBUMP1: INC      R2
9385 036132' 032701 000001      LBUMP2: BIT      #1,R1      ;IF BIT 0 RESET
9386 036136' 001403      BEQ      LBUMP3      ;GO ADD ONE
9387 036140' 162701 000003      SUB      #3,R1      ;OTHERWISE SUBTRACT 3
9388 036144' 000401      BR      LBUMP4
9389 036146' 005201      LBUMP3: INC      R1
9390 036150' 000205      LBUMP4: RTS      R5      ;RETURN
9391      :
9392      :      FILL WITH VARIABLE SUBROUTINE
9393      :
9394 036152' 012504      LFILV:  MOV      (R5)+,R4      ;GET ADDRESS
9395 036154' 012503      MOV      (R5)+,R3      ;BYTE COUNT
9396 036156' 012502      MOV      (R5)+,R2      ;AND VARIABLE
9397 036160' 032704 000001      LFOET:  BIT      #1,R4      ;IS ADDRESS ODD?
9398 036164' 001403      BEQ      LFOVCBC      ;NO-CHECK BYTE COUNT
9399 036166' 110224      LFOVOD: MOVB     R2,(R4)+      ;MOVE ONE BYTE OF VAR
9400 036170' 005303      DEC      R3      ;DECREMENT BYTE COUNT
9401 036172' 001410      BEQ      LFVEXT      ;EXIT IF ZERO
9402 036174' 022703 000001      LFOVCBC: CMP     #1,R3      ;CHECK BYTE COUNT
9403 036200' 001772      BEQ      LFOVOD      ;IF=1 TREAT AS ODD
9404 036202' 010224      LFVEVN: MOV     R2,(R4)+      ;MOVE ONE WORD OF VAR
9405 036204' 162703 000002      SUB      #2,R3      ;DECREMENT BYTE COUNT
9406 036210' 001401      BEQ      LFVEXT      ;EXIT IF ZERO
9407 036212' 000770      BR      LFOVCBC      ;OTHERWISE CHECK BYTE CNT
9408 036214' 000205      LFVEXT: RTS      R5      ;RETURN
9409      :
9410      :      FILL WITH RANDOM SUBROUTINE
9411      :
9412 036216' 012504      LFILR:  MOV      (R5)+,R4      ;GET ADDRESS
9413 036220' 012503      MOV      (R5)+,R3      ;AND BYTE COUNT
9414 036222' 032704 000001      LFR0ET: BIT     #1,R4      ;IS ADDRESS ODD?
9415 036226' 001406      BEQ      LFR0CBC      ;NO-CHECK BYTE COUNT
9416 036230' 004567 003070      LFR0OD: JSR     R5,LRANDOM      ;GET RANDOM NBR
9417 036234' 116724 003174      MOVB     LHNUM,(R4)+      ;MOVE IN ONE BYTE
9418 036240' 005303      DEC      R3      ;DECREMENT BYTE COUNT
9419 036242' 001413      BEQ      LFR0EXT      ;EXIT IF ZERO
9420 036244' 022703 000001      LFR0CBC: CMP     #1,R3      ;CHECK BYTE COUNT
9421 036250' 001767      BEQ      LFR0OD      ;IF=1 TREAT AS ODD
9422 036252' 004567 003046      LFR0EVN: JSR     R5,LRANDOM      ;GET RANDOM NBR
9423 036256' 016724 003152      MOV      LHNUM,(R4)+      ;MOVE IN ONE WORD
9424 036262' 162703 000002      SUB      #2,R3      ;DECREMENT BYTE COUNT
9425 036266' 001401      BEQ      LFR0EXT      ;EXIT IF ZERO
9426 036270' 000765      BR      LFR0CBC      ;OTHERWISE CHECK BYTE CNT
9427 036272' 000205      LFR0EXT: RTS     R5      ;RETURN
9428      :
9429      :      FILL WITH ASCII SUBROUTINE
9430      :
9431 036274' 012504      LFILG:  MOV      (R5)+,R4      ;GET ADDRESS
9432 036276' 012503      MOV      (R5)+,R3      ;AND BYTE COUNT

```


DTMGAC.P11

SUBROUTINES USED BY USER PROGRAM STATEMENTS

9433	036300'	112702	000040	LFGLP1:	MOVB	#40,R2	;INITIALIZE ASCII CHAR
9434	036304'	110224		LFGLP2:	MOVB	R2,(R4)+	;MOVE THE CHAR
9435	036306'	005303			DEC	R3	;DECREMENT BYTE COUNT
9436	036310'	001405			BEQ	LFGEXT	;EXIT IF ZERO
9437	036312'	105202			INCB	R2	;CREATE NEXT CHAR
9438	036314'	122702	000140		CMPB	#140,R2	;LAST CHAR OF SET
9439	036320'	001371			BNE	LFGLP2	;NO-GO MOVE IT
9440	036322'	000766			BR	LFGLP1	;YES-GO INITIALIZE IT
9441	036324'	000205		LFGEXT:	RTS	R5	;RETURN
9442				...			
9443						FILL WITH PATTERN SUBROUTINE	
9444							
9445	036326'	012504		LFILP:	MOV	(R5)+,R4	;GET ADDRESS AND
9446	036330'	012503			MOV	(R5)+,R3	;BYTE COUNT
9447	036332'	012501			MOV	(R5)+,R1	
9448	036334'	010702			MOV	PC,R2	;DETERMINE PATTERN
9449	036336'	061102		LPBASE:	ADD	(R1),R2	;GENERATION ROUTINE
9450	036340'	010207			MOV	R2,PC	;THEN GO THERE
9451							
9452	036342'	012700	000020	LPATO:	MOV	#16,R0	;WALKING ONES
9453	036346'	012702	100000		MOV	#100000,R2	
9454	036352'	004767	000132	10\$:	JSR	PC,STRPAT	
9455	036356'	005703			TST	R3	
9456	036360'	001452			BEQ	FILPEX	
9457	036362'	000241			CLC		
9458	036364'	006002			ROR	R2	
9459	036366'	005300			DEC	R0	
9460	036370'	001370			BNE	10\$	
9461	036372'	000763			BR	LPATO	
9462	036374'	012700	000020	LPAT1:	MOV	#16,R0	;WALKING ZEROES
9463	036400'	012702	077777		MOV	#77777,R2	
9464	036404'	004767	000100	10\$:	JSR	PC,STRPAT	
9465	036410'	005703			TST	R3	
9466	036412'	001435			BEQ	FILPEX	
9467	036414'	000261			SEC		
9468	036416'	006002			ROR	R2	
9469	036420'	005300			DEC	R0	
9470	036422'	001370			BNE	10\$	
9471	036424'	000763			BR	LPAT1	
9472	036426'	012700	000020	LPAT2:	MOV	#16,R0	;EXPANDING ONES
9473	036432'	012702	100000		MOV	#100000,R2	
9474	036436'	004767	000046	10\$:	JSR	PC,STRPAT	
9475	036442'	005703			TST	R3	
9476	036444'	001420			BEQ	FILPEX	
9477	036446'	006202			ASR	R2	
9478	036450'	005300			DEC	R0	
9479	036452'	001371			BNE	10\$	
9480	036454'	000764			BR	LPAT2	
9481	036456'	012700	000020	LPAT3:	MOV	#16,R0	;EXPANDING ZEROES
9482	036462'	012702	077777		MOV	#77777,R2	
9483	036466'	004767	000016	10\$:	JSR	PC,STRPAT	
9484	036472'	005703			TST	R3	
9485	036474'	001404			BEQ	FILPEX	
9486	036476'	006202			ASR	R2	
9487	036500'	005300			DEC	R0	
9488	036502'	001371			BNE	10\$	

9489	036504'	000764		BR	LPAT3	
9490	036506'	000205		FILPEX: RTS	R5	;RETURN INLINE
9491	036510'	110224		STRPAT: MOVB	R2,(R4)+	;STORE PATTERN SUBR
9492	036512'	005303		DEC	R3	
9493	036514'	001404		BEQ	STPTEX	
9494	036516'	000302		SWAB	R2	
9495	036520'	110224		MOVB	R2,(R4)+	
9496	036522'	005303		DEC	R3	
9497	036524'	000302		SWAB	R2	
9498	036526'	000207		STPTEX: RTS	PC	
9499	036530'	012700	125252	LPAT4: MOV	#125252,R0	;ALTERNATE ONES (HORIZ)
9500	036534'	012701	125252	MOV	#125252,R1	
9501	036540'	000450		BR	LPATCM	
9502	036542'	012700	052525	LPAT5: MOV	#052525,R0	;ALTERNATE ZEROES (HORIZ)
9503	036546'	012701	052525	MOV	#052525,R1	
9504	036552'	000443		BR	LPATCM	
9505	036554'	012700	177777	LPAT6: MOV	#177777,R0	;ALTERNATE ONES (VERT)
9506	036560'	012701	000000	MOV	#0,R1	
9507	036564'	000436		BR	LPATCM	
9508	036566'	012700	000000	LPAT7: MOV	#0,R0	;ALTERNATE ZEROES (VERT)
9509	036572'	012701	177777	MOV	#177777,R1	
9510	036576'	000431		BR	LPATCM	
9511	036600'	012700	070707	LPAT8: MOV	#70707,R0	;OCTAL CHECKERBOARD
9512	036604'	012701	107070	MOV	#107070,R1	
9513	036610'	000424		BR	LPATCM	
9514	036612'	012700	125252	LPAT9: MOV	#125252,R0	;BINARY CHECKERBOARD
9515	036616'	012701	052525	MOV	#52525,R1	
9516	036622'	000417		BR	LPATCM	
9517	036624'	005002		LPATA: CLR	R2	;START WITH WORD OF ZEROES
9518	036626'	110224		IOS: MOVB	R2,(R4)+	;STORE A BYTE
9519	036630'	005303		DEC	R3	
9520	036632'	001415		BEQ	LPATEX	;EXIT IF DONE
9521	036634'	000302		SWAB	R2	;SWAP BYTES WITHIN R2
9522	036636'	110224		MOVB	R2,(R4)+	;STORE A BYTE
9523	036640'	000302		SWAB	R2	;SWAP BYTES WITHIN R2
9524	036642'	005303		DEC	R3	
9525	036644'	001410		BEQ	LPATEX	;EXIT IF DONE
9526	036646'	005202		INC	R2	;ADD 1 TO R2
9527	036650'	000766		BR	IOS	;GO AGAIN
9528	036652'	012700	165555	LPATB: MOV	#165555,R0	;DISK WORST CASE
9529	036656'	012701	133333	MOV	#133333,R1	
9530	036662'	004767	000002	LPATCM: JSR	PC,PATGEN	
9531	036666'	000205		LPATEX: RTS	R5	
9532	036670'	010002		PATGEN: MOV	R0,R2	;GET WORD FROM R0
9533	036672'	110224		MOVB	R2,(R4)+	;STORE A BYTE
9534	036674'	005303		DEC	R3	
9535	036676'	001414		BEQ	PGENEX	;EXIT IF DONE
9536	036700'	000302		SWAB	R2	;SWAP BYTES WITHIN R2
9537	036702'	110224		MOVB	R2,(R4)+	;STORE A BYTE
9538	036704'	005303		DEC	R3	;DONE ?
9539	036706'	001410		BEQ	PGENEX	;YES - EXET
9540	036710'	010102		MOV	R1,R2	;GET WORD FROM R1
9541	036712'	110224		MOVB	R2,(R4)+	;STORE A BYTE
9542	036714'	005303		DEC	R3	;DONE ?
9543	036716'	001404		BEQ	PGENEX	;YES - EXIT
9544	036720'	000302		SWAB	R2	;SWAP BYTES WITHIN R2

```

9545 036722' 110224          MOVB   R2,(R4)+      ;STORE A BYTE
9546 036724' 005303          DEC    R3            ;DONE ?
9547 036726' 001360          BNE   PATGEN        ;NO - GET NEXT WORD
9548 036730' 000207          PGENEX: RTS         ;EXIT
9549
9550
9551      :
9552      :           ROTATE STRING SUBROUTINE
9553 036732' 012504          LROTR: MOV   (R5)+,R4      ;GET STRING ADDRESS
9554 036734' 012503          MOV   (R5)+,R3      ;GET # BYTES TO BE SPUN
9555 036736' 010402          MOV   R4,R2
9556 036740' 005303          DEC   R3
9557 036742' 060302          ADD   R3,R2        ;SET UP RIGHT POINTER
9558 036744' 010201          MOV   R2,R1        ;SET UP LEFT POINTER
9559 036746' 111200          MOVB  (R2),R0      ;SAVE RIGHT HAND BYTE
9560 036750' 005202          INC   R2
9561 036752' 114142          LROTRL: MOVB  -(R1),-(R2) ;MOVE OTHER BYTES RIGHT 1
9562 036754' 020204          CMP   R2,R4
9563 036756' 001375          BNE   LROTRL
9564 036760' 110012          MOVB  R0,(R2)      ;MOVE SAVED BYTE IN ON LEFT
9565 036762' 000205          RTS    R5         ;RETURN
9566
9567      :
9568      :           DELAY SUBROUTINE
9569 036764' 012500          LDLAY: MOV   (R5)+,R0      ;GET DELAY IN MILLISECONDS
9570 036766' 012701 000522  LDLLP1: MOV   #522,R1      ;LOAD CONSTANT FOR 1 MS
9571 036772' 005301          LDLLP2: DEC   R1
9572 036774' 001376          BNE   LDLLP2      ;DELAY 1 MILLISECOND
9573 036776' 005300          DEC   R0
9574 037000' 001372          BNE   LDLLP1      ;REPEAT AS PER COUNT
9575 037002' 000205          RTS    R5         ;RETURN
9576
9577      :
9578      :           VERIFY SUBROUTINE
9579 037004' 012500          LVERIFY: MOV   (R5)+,R0      ;GET BYTE COUNT
9580 037006' 012501          MOV   (R5)+,R1      ;GET 1ST DATA ADR
9581 037010' 012502          MOV   (R5)+,R2      ;GET 2ND DATA ADR
9582 037012' 005004          CLR   R4            ;INITIALIZE BYTE #
9583 037014' 004767 000254  JSR   PC,LVECMP      ;DO DATA COMPARE
9584 037020' 000401          BR    10$          ;MISCOMPARE? (N,Y-10$)
9585 037022' 000205          RTS    R5         ;EXIT IN-LINE TO USER PROG
9586
9587      :           ;VERIFY MISCOMPARE
9588
9589      :           001          .IF DF MM
9590
9591 037024' 016703 175636  10$: MOV   CPGCPT,R3      ;GET ADR OF PROG'S TABLE
9592
9593 037024' 016703 175636  10$: CLR   R3            ;GET ADR OF PROG'S TABLE
9594
9595      :           000          .ENDC
9596 037030' 010301          MOV   R3,R1        ;POINT AT ADR OF
9597 037032' 062701 000304  ADD   #PTLGTH+DECTAD,R1 ;DATA ERROR CNTR
9598 037036' 012761 000001 17776r MOV   #1,DEVIW8-DECTAD(R1) ;SET THE ERROR INDICATOR
9599 037044' 005711          TST   (R1)         ;IS THERE AN ERROR CNTR?
9600 037046' 001402          BEQ   20$          ;Y,N-20$
          ADD   (R1),R1 ;MAKE ITS ADR ABSOLUTE

```

9601	037052'	005211				INC	(R1)		;ADD 1 TO PROG'S DATA ERR CNT
9602	037054'	032763	020000	000002	20\$:	BIT	#PRONER,POPSW(R3)		;OPSW INDICATE PRINT ON ERROR?
9603	037062'	001403				BEQ	30\$;N,Y-30\$
9604	037064'	062706	000012			ADD	#10.,SP		;REMOVE ERROR DATA FROM STACK
9605	037070'	000205				RTS	R5		;EXIT IN-LINE TO USER PROG
9606	037072'	004567	001520		30\$:	JSR	R5,LTSTDE		;GO TEST DEDICATED FLAGS
9607	037076'	105067	000571			CLRB	LPRDE		;RESET REALLY DEDICATED FLAG
9608	037102'	016300	000022			MOV	PSRCST(R3),RC		;GET START ADR OF SRC STMT'S
9609	037106'	010501				MOV	R5,R1		;COMPUTE THIS STMT'S ADR
9610	037110'	162701	000012			SUB	#10.,R1		
9611	037114'	026001	000004		45\$:	CMP	4(R0),R1		;THIS THE STMT?
9612	037120'	001403				BEQ	50\$;N,Y-50\$
9613	037122'	111002				MOVB	(R0),R2		;GET THIS STMT'S LENGTH
9614	037124'	060200				ADD	R2,R0		;POINT AT NXT STMT
9615	037126'	000772				BR	45\$;GO CK IT
9616	037130'	005720			50\$:	TST	(R0)+		;POINT AT STMT #
9617	037132'	010701				MOV	PC,R1		;SET UP ADR OF AREA IN MSG
9618	037134'	062701	000262			ADD	#LVEHDN-.,R1		
9619	037140'	004567	001660			JSR	R5,DECASC		;CONVERT STMT # TO ASCII
9620	037144'	012767	006451	000250		MOV	#06451,LVEHDN+4		;RESTORE) & CR DATA IN THE MSG
9621	037152'	004567	000600			JSR	R5,LPROGN		;PRINT THE PROG #
9622	037156'	004567	000326			JSR	R5,USRPRT		;PRINT THE HEADER MSG
9623	037162'	037372'			LVECL1:	.WORD	LVEHDR		
9624	037164'	000064				.WORD	52.		
9625	037166'	010400			LVEDEP:	MOV	R4,R0		;GET THE BAD BYTE # IN R0
9626	037170'	004567	001770			JSR	R5,BTASLB		;CONVERT TO ASCII DECIMAL
9627	037174'	000262				.WORD	LVEMSG-		
9628	037176'	005201				INC	R1		;POINT AT MSG'S 1ST DATA FIELD
9629	037200'	012600				MOV	(SP)+,R0		;GET FIRST BAD DATA BYTE
9630	037202'	004767	000136			JSR	PC,LVEADB		;CONVERT IT TO ASCII
9631	037206'	012600				MOV	(SP)+,R0		;GET 2ND BAD DATA BYTE
9632	037210'	005721				TST	(R1)+		;POINT AT MSG'S 2ND DATA FIELD
9633	037212'	004767	000126			JSR	PC,LVEADB		;CONVERT IT TO ASCII
9634	037216'	004567	000534			JSR	R5,LPROGN		;PRINT THE PROG # MSG
9635	037222'	004567	000262			JSR	R5,USRPRT		;PRINT THE BAD DATA MSG
9636	037226'	037456'			LVECL2:	.WORD	LVEMSG		
9637	037230'	000032				.WORD	26.		
9638	037232'	012600				MOV	(SP)+,R0		;RESTORE THE CURR BYTE COUNT
9639	037234'	012601				MOV	(SP)+,R1		;RESTORE DATA FIELD # 1 ADR
9640	037236'	012602				MOV	(SP)+,R2		;RESTORE DATA FIELD # 2 ADR
9641	037240'	032763	000002	000002		BIT	#PFBB0V,POPSW(R3)		;OPSW SAY PRINT ONLY 1ST BAD BYTE?
9642	037246'	001003				BNE	LVEREX		;N,Y-LVEREX
9643	037250'	004767	000020			JSR	PC,LVECMP		;RESUME DATA COMPARE
9644	037254'	000744				BR	LVEDEP		;ANOTHER ERROR? (N,Y-LVEDEP)
9645	037256'	042767	020000	174730	LVEREX:	BIC	#CTUDED,CSYSFW		;RESET SYSTEM'S DEDICATED FLAG
9646	037264'	042713	000040			BIC	#SETDED,(R3)		;RESET PROG'S DEDICATED FLAG
9647	037270'	000167	175464			JMP	CUPGER		;GO TO MPG'S USER PROG ERR RETN POINT
9648									
9649									
9650	037274'	005700			LVECMP:	TST	R0		;BYTE COUNT = 0?
9651	037276'	001003				BNE	100\$;Y,N-100\$
9652	037300'	062716	000002			ADD	#2,(SP)		;POINT PAST ERROR BR INST
9653	037304'	000207				RTS	PC		;EXIT IN-LINE AFTER BR INST
9654	037306'	005204			100\$:	INC	R4		;ADD 1 TO BYTE #
9655	037310'	005300				DEC	R0		;SUB 1 FROM BYTE CNT
9656	037312'	122122				CMPB	(R1)+,(R2)+		;DATA BYTES COMPARE?

```

9657 037314' 001767          BEQ      LVECMP          ;N.Y-LVECMP
9658 037316' 012627          MOV      (SP)+,(PC)+    ;SAVE RETURN ADR
9659 037320' 000000          110$: .WORD 0
9660 037322' 010246          MOV      R2,-(SP)       ;SAVE CURR DATA 2 ADR
9661 037324' 010146          MOV      R1,-(SP)       ;SAVE CURR DATA 1 ADR
9662 037326' 010046          MOV      R0,-(SP)       ;SAVE CURR BYTE CNT
9663 037330' 116246 177777  MOVB     -1(R2),-(SP)    ;SAVE BAD DATA # 2 BYTE
9664 037334' 116146 177777  MOVB     -1(R1),-(SP)    ;SAVE BAD DATA # 1 BYTE
9665 037340' 016707 177754  MOV      110$,PC        ;EXIT IN-LINE & DO THE BR INST
9666
9667
9668 037344' 042700 177400  LVEADB: BIC      #177400,R0 ;RESET OTHER BYTE
9669 037350' 004567 001552  JSR      R5,BINAS1      ;CONVERT BYTE TO ASCII
9670 037354' 012761 020040 177772  MOV      #20040,-6(R1)  ;STORE LEADING BLANKS
9671 037362' 112761 000040 177774  MOVB     #40,-4(R1)
9672 037370' 000207          RTS      PC             ;EXIT IN-LINE
9673
9674
9675 037372' 040504 040524 042440 LVEHDR: .ASCII /DATA ERROR (STMNT # /
037400' 051122 051117 024040
037406' 052123 047115 020124
037414' 020043
9676 037416' 054130 054130 006451 LVEHDN: .ASCII /XXXX/<015><012><011>
037424' 004412
9677 037426' 054502 042524 021440 .ASCII /BYTE # DATA 1 DATA 2/<015><012>
037434' 020040 040504 040524
037442' 030440 020040 040504
037450' 040524 031040 005015
9678
9679 037456' 020040 020040 020040 LVEMSG: .EVEN
037464' 020040 020040 020040 .ASCII /
037472' 020040 020040 020040 /<015><012>
037500' 020040 020040 020040
037506' 005015
9680
          .EVEN
    
```

.SBT'L USER PRINT SUBROUTINE

THIS SUBROUTINE HANDLES ALL PRINT OPERATIONS TO THE DISPLAY DEVICE AS A RESULT OF USER INSTRUCTIONS. ITS PURPOSE IS TO DEDICATE THE PRINT DEVICE TO ONE USER PROGRAM UNTIL A MULTIPLE LINE OUTPUT IS PRINTED WHILE ALLOWING INTERACTION AMONG OTHER I/O DEVICES BETWEEN LINES.

CALLING SEQUENCE JSR R5,USRPRT
.WORD ADDRESS
.WORD BYTE COUNT

9682
9683
9684
9685
9686
9687
9688
9689
9690
9691
9692
9693
9694
9695
9696
9697
9698
9699
9700
9701 037510' 105067 000156
9702 037514' 004467 001256
9703 037520' 012504
9704 037522' 012503
9705 037524' 010301
9706 037526' 005303
9707 037530' 020127 000062
9708 037534' 003402
9709 037536' 012701 000062
9710 037542' 010167 000064
9711 037546' 010702
9712 037550' 062702 000124
9713 037554' 112422
9714 037556' 005303
9715 037560' 005301
9716 037562' 001374
9717 037564' 005703
9718 037566' 100006
9719 037570' 105767 000076
9720 037574' 001403
9721 037576' 052767 040000 174410
9722 001
9723
9724 037604' 016701 140226
9725 037610' 016700 140200
9726 037614' 005760 000004
9727 037620' 001402
9728 037622' 016701 140214
9729 037626' 004511
9730 037630' 000044
9731 037632' 000000
9732 037634' 000030
9733 037636' 000000
9734
9735
9736
9737

USRPRT: CLRB LLPRTF ; RESET LAST PRINT FLAG
USRPR1: JSR R4,SREG ; SAVE REG'S R0 - R4
MOV (R5)+,R4 ; GET TEXT ADDRESS
MOV (R5)+,R3 ; AND BYTE COUNT
MOV R3,R1 ; SET UP WORK COUNT
DEC R3 ; ADJ TOTAL CNT FOR LATER TESTING
USRLP: CMP R1,#50. ; WILL MSG FIT IN BUF?
BLE 10\$; YES - GO AHEAD
MOV #50.,R1 ; NO-TRUNCATE IT FOR NOW
10\$: MOV R1,LPRCNT ; STORE BYTE CNT
MOV PC,R2 ; POINT R2 AT
ADD #LPRBUF-,R2 ; OUTPUT BUFFER
20\$: MOVB (R4)+,(R2)+ ; MOVE MESSAGE
DEC R3 ; DECR TOTAL CNT
DEC R1 ; DECR WORK CNT
BNE 20\$; GOT ALL CHARS THIS GROUP? (Y,N-20\$)
TST R3 ; MORE DATA AFTER THIS GROUP?
BPL LPRCAL ; N,Y-LPRCAL
TSTB LLPRTF ; THIS THE LAST PRINT?
BEQ LPRCAL ; Y,N-LPRCAL
BIS #CRUDED,CSYSFW ; TELL DRIVER TO RESET DEDICATED FLAG
.IF DF MM
.IFF
LPRCAL: MOV CTUSWR,R1 ; INITIALIZE TO CONS TERM USR WRITE
MOV CLISTI,R0 ; GET ADR OF LIST DEV
TST 4(R0) ; IS THERE A PRINTER ASSIGNED?
BEQ 30\$; Y,N-30\$
MOV CUSPRT,R1 ; GET USER'S PRINT ROUT ADR
30\$: JSR R5,(R1) ; CALL FOR PRINT
.WORD LPRBUF- ; ADDRESS
LPRCNT: .WORD XXXX ; BYTE COUNT
.WORD LPRBSY- ; BUSY ADDRESS
LPRFWA: .WORD 0 ; FLAGWORD ADDRESS
.IFT
LPRCAL: TRAP 10 ; GO INTO KERNEL MODE. DOES
; INSTS AT "UPRTKM" & RETURNS
; IN-LINE IF NOT BUSY. IF BUSY,

```

9738                                     ;RETURNS AT "LPRBSY".
9739                                     .ENDC
9740 037640' 004567 175352             JSR    R5, CIOBSY      ;WAIT FOR I/O COMPLETE
9741 037644' 005703                     TST    R3              ;MORE DATA ON THIS MSG?
9742 037646' 100403                     BMI    USRPEX          ;BR IF NO - GO TO EXIT
9743 037650' 010301                     MOV    R3, R1         ;SET UP REMAINING BYTE CNT
9744 037652' 005201                     INC    R1
9745 037654' 000725                     BR     USRLP          ;GO PRINT NEXT GROUP
9746 037656' 004467 001126             USRPEX: JSR    R4, RREG ;RELOAD REGISTERS
9747 037662' 000205                     RTS    R5              ;RETURN
9748 037664' 004567 175326             LPRBSY: JSR    R5, CIOBSY ;IF I/O NOT STARTED, WAIT & THEN
9749 037670' 000745                     BR     LPRCAL         ;HIT IT AGAIN
9750                                     .IF DF MM
9751 UPRTKM: MOV    CTUSWR, R1          ;INITIALIZE TO CONS TERM USR WRITE
9752                                     MOV    CLISTI, R0      ;GET ADR OF LIST DEV
9753                                     TST    4(R0)          ;IS THERE A PRINTER ASSIGNED?
9754                                     BEQ    30$            ;Y, N-30$
9755                                     MOV    CUSPRT, R1     ;GET USER'S PRINT ROUT ADR
9756 30$: JSR    R5, (R1)          ;ISSUE THE MSG
9757                                     .WORD LPRBUF-        ;REL DATA ADR
9758 LPRCNT: .WORD XXXX          ;MSG BYTE CNT
9759                                     .WORD UPRTKB-        ;REL BUSY ADR
9760 LPRFWA: .WORD XXXX          ;FLAGWORD ADR
9761 RTI                                     ;RETURN IN-LINE FROM TRAP
9762
9763 UPRTKB: MOV    #LPRBSY-PG6BA+P6CONS, (SP) ;SET UP BUSY RETURN ADR
9764 RTI                                     ;RETURN TO BUSY ADR FROM TRAP
9765                                     .ENDC
9766 037672' 000                                LLPRTF: .BYTE 0
9767 037673' 000                                LPRRDE: .BYTE 0      ;REALLY DEDICATED FLAG
9768 037674' 000062                            LPRBUF: .BLKB 50.   ;MESSAGE BUFFER
9769
9770
9771                                     ;
9772                                     ;
9773                                     ;
9774                                     ;
9775                                     ;
9776                                     ;
9777                                     ;
9778                                     ;
9779                                     ;
9780                                     ;
9781                                     ;
9782                                     ;
9783                                     ;
9784                                     ;
9785                                     ;
9786                                     ;
9787                                     ;
9788                                     ;
9789                                     ;
9790                                     ;
9791                                     ;
9792                                     ;

```

PRINT PROGRAM NUMBER SUBROUTINE

PRINT ASCII SUBROUTINE

```

9793 040022' 004567 000570 LPRNG: JSR R5,LTSTDE ;GO TEST DEDICATE FLAG
9794 040026' 004567 177724 UPRASC: JSR R5,LPROGN ;PRINT P NN
9795 040032' 012500 MOV (R5)+,R0 ;GET ADDRESS
9796 040034' 012501 MOV (R5)+,R1 ;AND BYTE COUNT
9797 040036' 100422 BMI LPGNCT ;BRANCH IF NEG COUNT
9798 040040' 105067 177627 CLRB LPRRDE
9799 040044' 020500 LPGTIL: CMP R5,R0 ;TEXT INLINE?
9800 040046' 001005 BNE LPGTG1 ;NO-CONTINUE
9801 040050' 060105 ADD R1,R5 ;YES-ADJUST RETURN
9802 040052' 032705 000001 BIT #1,R5 ;WAS BYTE CNT ODD?
9803 040056' 001401 BEQ LPGTG1 ;Y,N-LPGTG1
9804 040060' 005205 INC R5 ;ADJ RET ADR TO EVEN
9805 040062' 010067 000010 LPGTG1: MOV R0,LPGADR ;MOVE ADDRESS
9806 040066' 010167 000006 MOV R1,LPGBYC ;AND BYTE COUNT TO CALL
9807 040072' 004567 177412 JSR R5,USRPT ;CALL USER PRINT SUB
9808 040076' 000000 LPGADR: .WORD 0
9809 040100' 000000 LPGBYC: .WORD 0
9810 040102' 000475 BR LPOFXT ;CR LF AND EXIT
9811 040104' 105267 177563 LPGNCT: INCB LPRRDE ;SET REALLY DEDICATED FL
9812 040110' 005401 NEG R1 ;GET POSITIVE COUNT
9813 040112' 000754 BR LPGTIL ;GO TEST IF TEXT INLINE
9814
9815
9816
9817
          .
          .
          .
          PRINT OCTAL OR DECIMAL SUBROUTINE
9818 040114' 004567 000476 LPRND: JSR R5,LTSTDE ;GO TEST DEDICATE FLAG
9819 040120' 005067 000254 CLR LOCTSW ;CLEAR OCTAL SWITCH
9820 040124' 000404 BR LPOSDF
9821 040126' 004567 000464 LPRNO: JSR R5,LTSTDE ;GO TEST DEDICATE FLAG
9822 040132' 005267 000242 INC LOCTSW ;SET OCTAL SWITCH
9823 040136' 012767 000010 000160 LPOSDF: MOV #8,LPOCNT ;SET FORMAT CNTR = 8
9824 040144' 004567 177606 JSR R5,LPROGN ;PRINT P NN
9825 040150' 105067 177517 CLRB LPRRDE ;CLEAR REALLY DED FLAG
9826 040154' 012504 MOV (R5)+,R4 ;GET ADDRESS
9827 040156' 012503 MOV (R5)+,R3 ;GET BYTE COUNT
9828 040160' 032704 000001 LPOOET: BIT #1,R4 ;IS ADDRESS ODD?
9829 040164' 001431 BEQ LPOCBC ;NO-CHECK BYTE COUNT
9830 040166' 112400 LPOODD: MOVB (R4)+,R0 ;GET ONE BYTE
9831 040170' 042700 177400 BIC #177400,R0
9832 040174' 004567 000152 JSR R5,LTOASC ;CONVERT
9833 040200' 010702 MOV PC,R2
9834 040202' 062702 174060 ADD #LINBUF-. ,R2
9835 040206' 012722 020040 MOV #20040,(R2)+ ;BLANK OUT
9836 040212' 112722 000040 MOVB #40,(R2)+ ;LEADING ZEROES
9837 040216' 062702 000003 ADD #3,R2
9838 040222' 012722 020040 LPOPRT: MOV #20040,(R2)+ ;STORE TWO TRAILING BLANKS
9839 040226' 004567 177256 JSR R5,USRPT ;CALL USER PRINT SUB
9840 040232' 034262' LPOADR: .WORD LINBUF
9841 040234' 000010 .WORD 8
9842 040236' 005303 DEC R3 ;DECREMENT BYTE COUNT
9843 040240' 001416 BEQ LPOFXT ;IF ZERO FIND EXIT
9844 040242' 005367 000056 DEC LPOCNT ;DECREMENT FORMAT COUNTER
9845 040246' 001427 BEQ LPORTN ;CR - LF IF ZERO
9846 040250' 022703 000001 LPOCBC: CMP #1,R3 ;IF ONLY ONE REMAINING
9847 040254' 001744 BEQ LPOODD ;TREAT AS ODD ADDRESS
9848 040256' 012400 LPOEVN: MOV (R4)+,R0 ;GET WORD

```



```

9849 040260' 004567 000066          JSR    R5,LTOASC          ;CONVERT
9850 040264' 010702          MOV    PC,R2
9851 040266' 062702 174002          ADD    #LINBUF+6--,R2
9852 040272' 005303          DEC    R3                ;DECREMENT BYTE CNT BY 1
9853 040274' 000752          BR     LPOPRT            ;GO PRINT IT
9854 040276' 105767 177371          LPOFXT: TSTB            LPRRDE            ;TEST REALLY DEDICATED FL
9855 040302' 001007          BNE   LPONOM            ;IF SET THERE'S MORE TO GO
9856 040304' 112767 000001 177360  ULCRLF: MOVB            #1,LLPRTF        ;SET THIS IS THE LAST PRT FLG
9857 040312' 004567 177176          UPCRLF: JSR            R5,USRPR1        ;PRINT A CARRIAGE RET
9858 040316' 042114'          LPOCR1: .WORD          LCRETN          ;AND LINE FEED
9859 040320' 000002          .WORD          2
9860 040322' 000205          LPONOM: RTS            R5                ;RETURN
9861 040324' 000000          LPOCNT: .WORD          0                ;FORMAT COUNTER
9862 040326' 004567 177156          LPORNT: JSR            R5,USRPR1        ;PRINT CR LF
9863 040332' 042114'          LPOCR2: .WORD          LCRETN
9864 040334' 000002          .WORD          2
9865 040336' 012767 000010 177760  MOV    #8.,LPOCNT        ;RE-INIT COUNTER
9866 040344' 004567 177406          JSR    R5,LPROGN        ;PRINT P NN
9867 040350' 000737          BR     LPOCBC            ;GO CHECK BYTE COUNT
9868 040352' 005767 000022          LTOASC: TST            LOCTSW          ;TEST OCTAL SWITCH
9869 040356' 001404          BEQ   LTOD              ;BRANCH IF NOT SET
9870 040360' 004567 000536          LTOD:  JSR            R5,BINASC        ;CONVERT TO OCTAL
9871 040364' 173676          .WORD          LINBUF-
9872 040366' 000205          RTS    R5                ;RETURN
9873 040370' 004567 000570          LTOD:  JSR            R5,BTASLB        ;CONVERT TO DECIMAL
9874 040374' 173666          .WORD          LINBUF-
9875 040376' 000205          RTS    R5                ;RETURN
9876 040400' 000000          LOCTSW: .WORD          0
9877
9878
9879
9880
9881
9882
9883
9884
9885
9886
9887
9888
9889
9890
9891
9892
9893
9894
9895
9896
9897
9898
9899
9900
9901
9902
9903
9904
040402' 004567 000210          :
040406' 012767 000002 177710  LPRNB: JSR            R5,LTSTDE        ;GO TEST DEDICATE FLAG
040414' 004567 177336          MOV    #2,LPOCNT        ;SET FORMAT CTR=2
040420' 105067 177247          JSR    R5,LPROGN        ;PRINT P NN
040424' 012504          CLR   LPRRDE            ;CLEAR REALLY DED FLAG
040426' 012503          MOV    (R5)+,R4         ;GET ADDRESS
040430' 032704 000001          MOV    (R5)+,R3         ;GET BYTE COUNT
040434' 001443          LPBOET: BIT            #1,R4          ;IS ADDRESS ODD?
040436' 010702          BEG   LPBCBC            ;NO-CHECK BYTE COUNT
040440' 062702 173622          LPBODD: MOV            PC,R2
040444' 012701 000010          ADD    #LINBUF--,R2
040450' 112722 000040          MOV    #8.,R1           ;SET SPACE COUNT IN R1
040454' 005301          LPBLP1: MOVB           #40,(R2)+      ;INSERT
040456' 001374          DEC    R1                ;8 LEADING BLANKS
040460' 012701 000010          BNE   LPBLP1
040464' 112400          MOV    #8.,R1           ;SET BIT COUNT=8
040466' 000300          MOVB   (R4)+,R0         ;GET A BYTE
040470' 006300          SWAB  R0                ;LEFT JUSTIFY
040472' 103403          LPBLP2: ASL            R0            ;SHIFT LEFT INTO CARRY
040474' 112722 000060          BCS   LPBTG1            ;IF CARRY NOT SET
040500' 000402          MOVB   #60,(R2)+       ;STORE ASCII ZERO
040502' 112722 000061          BR     LPBTG2
040506' 005301          LPBTG1: MOVB          #61,(R2)+      ;OTHERWISE ASCII ONE
040510' 001367          LPBTG2: DEC            R1            ;REPEAT UNTIL
          BNE   LPBLP2        ;BIT COUNTER=0

```

```

9905 040512' 112722 000011          MOVB      #11,(R2)+      ;STORE TWO
9906 040516' 112722 000011          MOVB      #11,(R2)+      ;TAB CHARACTERS
9907 040522' 004567 176762          JSR       R5,USRPR1     ;PRINT 18 CHARS.
9908 040526' 034262'          LPBADR: .WORD      LINBUF
9909 040530' 000022          .WORD      18.
9910 040532' 005303          DEC       R3            ;DECREMENT BYTE COUNT
9911 040534' 001660          BEQ      LPOFXT         ;IF 0, EXIT
9912 040536' 005367 177562          LPBCFM: DEC       LPOCNT  ;DECREMENT FORMAT COUNTER
9913 040542' 001413          BEQ      LPBRTN         ;IF ZERO PRINT CR-LF
9914 040544' 022703 000001          LPBCBC: CMP       #1,R3   ;IF ONLY ONE BYTE LEFT
9915 040550' 001732          BEQ      LPB00D         ;TREAT AS ODD
9916 040552' 010702          MOV      PC,R2
9917 040554' 062702 173506          ADD      #LINBUF-.,R2
9918 040560' 012400          LPBEVN: MOV      (R4)+,R0 ;GET A WORD IF EVEN
9919 040562' 012701 000020          MOV      #16.,R1      ;SET BIT COUNT=16
9920 040566' 005303          DEC       R3
9921 040570' 000737          BR       LPBLP2         ;GO STORE ZEROES & ONES
9922 040572' 004567 176712          LPBRTN: JSR       R5,USRPR1 ;PRINT CR-LF
9923 040576' 042114'          LPBCR1: .WORD      LCRETN
9924 040600' 000002          .WORD      2
9925 040602' 012767 000002 177514          MOV      #2,LPOCNT     ;RE-INIT FORMAT CTR
9926 040610' 004567 177142          JSR      R5,LPROGN     ;PRINT P NN
9927 040614' 000753          BR       LPBCBC        ;GO CHECK BYTE COUNT
9928
9929
9930 040616' 032767 020000 173370          LTSTDE: BIT      #CTUDED,CSYSFW ;TEST DEDICATE FLAG
9931 040624' 001410          BEQ      LNOTDE        ;RETURN IF NOT SET
9932          001          .IF DF MM
9933          .IFF
9934 040626' 032777 000040 174032          BIT      #SETDED,CPGCPT ;CONSOLE DED TO THIS PROG ?
9935          .IFT
9936          BIT      #SETDED,0#0 ;CONSOLE DED TO THIS PROG?
9937          .ENDC
9938 040634' 001004          BNE      LNOTDE        ;N,Y-LNOTDE
9939 040636' 162705 000004          SUB      #4,R5         ;OTHERWISE BACK UP R5
9940 040642' 000167 174350          JMP      CIOBSY        ;AND RELEASE CONTROL
9941          001          .IF DF MM
9942          .IFF
9943 040646' 016703 174014          LNOTDE: MOV      CPGCPT,R3   ;GET PROG'S TABLE ADR
9944          .IFT
9945          LNOTDE: CLR      R3            ;GET VIRTUAL ADR OF PROG'S TABLE
9946          .ENDC
9947 040652' 052713 000040          BIS      #SETDED,(R3)   ;SET PROG'S DEDICATED FLAG
9948 040656' 052767 020000 173330          BIS      #CTUDED,CSYSFW ;SET SYSTEM'S DEDICATED FLAG
9949          001          .IF DF MM
9950          .IFF
9951 040664' 010367 176746          MOV      R3,LPRFWA     ;STORE PROG'S TABLE ADR
9952          .IFT
9953          MOV      PFWADR(R3),LPRFWA ;STORE HI 16 BITS OF PRG TBL ADR
9954          .ENDC
9955 040670' 000205          RTS       R5            ;RETURN

```

.SBTTL USER PROGRAM INTERRUPT COMMON PROCESSING ROUTINES

REQUIRED IN MEMORY MANAGEMENT VERSION
INCLUDED IN BOTH VERSIONS
FOR DEVICE ROUTINE COMPATIBILITY

;SET UP INTERRUPT VECTOR S/R

JSR R5,SETVEC ;S/R CALL
.WORD VECT ADR ;INT VECT ABS ADR
.WORD PSW ;PSW VALUE
.WORD INTROUT-. ;INT ROUT REL ADR

001

.IF DF MM

.IFF

SETVEC: JSR R4,SREG ;SAVE REGS R0 - R4
MOV (R5)+,R0 ;GET VECTOR ADR
MOV (R5)+,R1 ;GET PSW
MOV R5,R2 ;GET INT ROUT ADR
ADD (R5)+,R2 ;MAKE IT ABSOLUTE
MOV R2,(R0)+ ;PUT IT IN VECTOR ADR
MOV R1,(R0) ;STORE PSW
BR VECEX ;GO TO EXIT

.IFT

SETVEC: JSR R4,SREG ;SAVE REGISTERS
MOV (R5)+,R1 ;GET VECTOR ADDRESS
MOV (R5)+,R2 ;GET PSW
MOV R5,R3 ;GET INT ROUT ADR
ADD (R5)+,R3 ;MAKE IT REL TO PAGE 0
MOV PC,R0 ;POINT R0 AT
ADD #VECTBL+4-.,R0 ;INTER-PAGE VECTOR TBL
MOV R1,R4
ASL R4 ;ADJUST R0 TO ENTRY
ADD R4,R0 ;FOR THIS VECTOR
MOV #UPARO,(R0)+ ;SAVE PARO CONTENTS FOR PAGE 5 BASE
BIS #P5CONS,R3 ;SET PARS SEL BITS IN INT ROUT ADR
MOV R3,(R0) ;SAVE IN TBL
SUB #6,R0 ;POINT R0 AT JSR IN TBL
TRAP 4 ;GO INTO KERN MODE TO SET UP VECT
BR VECEX ;DOES INST'S AT "KSETVC" & RET IN-LINE
;GO TO EXIT

KSETVC: SUB #P6CONS,R0 ;CONVERT TBL ADR TO KERNEL SPACE

ADD PC,R0
ADD #PG6BA-.,R0
MOV R0,(R1)+ ;SET HARDWARE VECTOR
MOV R2,(R1) ;AND PSW
RTI ;RETURN FROM TRAP
.IFTF

9957
9958
9959
9960
9961
9962
9963
9964
9965
9966
9967
9968
9969
9970
9971
9972
9973
9974
9975
9976
9977
9978
9979
9980
9981
9982
9983
9984
9985
9986
9987
9988
9989
9990
9991
9992
9993
9994
9995
9996
9997
9998
9999
10000
10001
10002
10003
10004
10005
10006
10007
10008
10009
10010
10011
10012

040672' 004467 000100
040676' 012500
040700' 012501
040702' 010502
040704' 062502
040706' 010220
040710' 010110
040712' 000421

```

10013 ;CLEAR INTERRUPT VECTOR S/R
10014 :
10015 : JSR R5,CLRVEC ;S/R CALL
10016 : .WORD VECT ADR ;INT VECT ABS ADR
10017 :
10018 :
10019 CLRVEC: .IFF
10020 040714' 004467 000056 JSR R4,SREG ;SAVE REGS R0 - R4
10021 040720' 012500 MOV (R5)+,R0 ;GET VECTOR ADR
10022 040722' 010001 MOV R0,R1
10023 040724' 005721 TST (R1)+ ;SET R1 TO NXT ADR
10024 040726' 010120 MOV R1,(R0)+ ;POINT VECTOR AT .+2
10025 040730' 005010 CLR (R0) ;PUT HALT AT .+2
10026 040732' 000411 BR VECEX ;GO TO EXIT
10027 :
10028 CLRVEC: .IFT
10029 JSR R4,SREG ;SAVE REGISTERS
10030 MOV (R5)+,R0 ;GET VECTOR ADDRESS
10031 MOV PC,R2 ;POINT R2 AT
10032 ADD #VECTBL+6-,R2 ;INTER PAGE VECTOR TBL
10033 MOV R0,R1
10034 ASL R1 ;ADJUST R2 TO ROUT ADR ENTRY
10035 ADD R1,R2 ;FOR THIS VECTOR
10036 MOV R0,R1
10037 TST (R1)+ ;SET R1 TO NEXT ADR
10038 TRAP 6 ;GO INTO KERN MODE TO CLEAR VECTOR
10039 : ;DOES INST'S AT "KCLRVC" & RET IN-LINE
10040 CLR (R2) ;CLEAR INTER-PAGE VECTOR
10041 BR VECEX ;GO TO EXIT
10042 :
10043 KCLRVC: MOV R1,(R0)+ ;POINT VECTOR AT .+2
10044 CLR (R0) ;PUT HALT AT .+2
10045 RTI ;RETURN FROM TRAP
10046 :
10047 : .IFTF
10048 :
10049 : ;TEST INTERRUPT VECTOR S/R
10050 :
10051 : MOV PFWADR(R?),-(SP) ;PUT PROG'S FLGWD ADR ON STK
10052 : JSR R5,TSTVEC ;S/R CALL
10053 : .WORD VECT ADR ;INT VECT ABS ADR
10054 : .WORD INTROUT- ;INT ROUT REL ADR
10055 : BR LABEL ;EXECUTED IF NOT SAME
10056 : ;DESTROYS R0
10057 :
10058 TSTVEC: .IFF
10059 040734' 012616 MOV (SP)+,(SP) ;REMOVE UNUSED FLGWD ADR
10060 040736' 004467 JSR R4,SREG ;SAVE REGS R0 - R4
10061 040742' 012500 MOV (R5)+,R0 ;GET VECTOR ADDR
10062 040744' 010501 MOV R5,R1 ;GET INT ROUT ADR
10063 040746' 062501 ADD (R5)+,R1 ;MAKE IT ABSOLUTE
10064 040750' 021001 CMP (R0),R1 ;IS IT SAME AS VECTOR ?
10065 040752' 001001 BNE VECEX ;NO-RETURN
10066 040754' 005725 TST (R5)+ ;YES-BUMP R5
10067 040756' 004467 JSR R4,RREG ;RESTORE REGS R0 - R4
10068 040762' 000205 RTS R5
10069 :
10070 TSTVEC: .IFT
10071 MOV 2(SP),R0 ;GET PROG FLGWD ADR OF STK

```

```

10069          MOV      (SP)+,(SP)          ;ELIMINATE ITS SPACE ON STK.
10070          JSR      R4,SREG             ;SAVE REGISTERS
10071          MOV      (R5)+,R1            ;GET VECTOR ADDR
10072          MOV      R5,R2              ;GET INT ROUT ADR
10073          ADD      (R5)+,R2            ;MAKE IT REL TO PAGE 0
10074          MOV      PC,R3              ;POINT R3 AT
10075          ADD      @VECTBL+6-.,R3      ;INTER-PAGE VECTOR TBL
10076          ASL      R1                  ;ADJUST R3 TO ROUT ADR ENTRY
10077          ADD      R1,R3               ;FOR THIS VECTOR
10078          BIS      @P5CONS,R2         ;SET PARS SELECT BITS IN R2
10079          CMP      R2,(R3)            ;IS VECTOR POINTING AT MY
10080          BNE      VECEX               ;VIRTUAL ADR? NO-RETURN
10081          CMP      R0,-(R3)           ;MY PROG FLGWD MATCH ITS?
10082          BNE      VECEX               ;Y,N-VECEX
10083          TST      (R5)+              ;YES, BUMP R5 & BYPASS BR INST
10084          JSR      R4,RREG             ;RESTORE REGS
10085          RTS      R5                  ;RETURN
10086
10087
10088          ;INTER-PAGE INTERRUPT VECTOR TABLE
10089          ;
10090          ;
10091
10092          VECTBL: .REPT 128.
10093          JSR      R5,INTCOM             ;TO INTERRUPT COMMON
10094          .WORD    0                    ;SAVED UPARD
10095          .WORD    0                    ;INT ROUTINE ADR
10096          .ENDR
10097          .PAGE
10098          ;INTERRUPT INTERCEPT COMMON PROCESSING S/R
10099          ;
10100          ;
10101
10102          INTCOM: MOV      @#KPAR5,-(SP)   ;SAVE KERNAL'S PAR 5 &
10103          MOV      @#KPAR6,-(SP)         ;PAR 6
10104          MOV      @#KPAR6,@#KPAR6      ;POINT PAR 6 AT SHARED CODE PAGE
10105          MOV      (R5)+,@#KPAR5        ;SET KERNEL PAR 5 = TO OLD USR PAR 0
10106          JSR      PC,@(R5)+            ;GO TO USER'S INTERRUPT ROUT
10107          INTRET: MOV      (SP)+,@#KPAR6 ;RESTORE KERNEL'S PAR 6 &
10108          MOV      (SP)+,@#KPAR5        ;PAR 5
10109          MOV      (SP)+,R5             ;RESTORE ORG R5
10110          RTI                          ;RETURN FROM INTERRUPT
10111          .IFTF
10112
10113          ;RETURN FROM INTERRUPT S/R
10114          ;
10115          ;
10116          ;      JMP      RNTINT          S/R CALL
10117          ;
10118
10119          RTNINT: .IFF
10120          RTNINT: RTI                    ;RETURN FROM INTERRUPT
10121          RTNINT: RTS      PC            ;RET TO INT COM PROC ROUT AFTER JSR
10122          .ENDC
10123
10124

```

040764' 000002

000

F01

```

10125                                     ;GET DATA BYTE FROM USER'S ABS/VIRT ADR SPACE
10126                                     :
10127                                     :       JSR      PC,GETBYT      S/R CALL
10128                                     :       RO CONTAINS ABS/VIRT MEMORY ADR
10129                                     :       RETURNS DATA BYTE IN R1
10130                                     :       RO WILL BE INCREMENTED BY 1
10131                                     :       ALL OTHER REGISTERS ARE UNCHANGED
10132                                     :
10133                                     .IF DF MM
10134                                     .IFF
10135 040766' 112001   GETBYT: MOV      (RO)+,R1      ;GET DATA BYTE
10136 040770' 000207   RTS      PC              ;EXIT IN-LINE
10137                                     .IFT
10138   GETBYT: JSR      R2,GPBCSU      ;DO COMMON SET UP
10139   GPBEX: MOV      (RO),R1        ;GET DATA BYTE IN R1
10140   MOV      (SP)+,RO            ;RESTORE RO
10141   MOV      (SP)+,@#KPDR4       ;RESTORE PDR & PAR REGS
10142   MOV      (SP)+,@#KPAR4
10143   MOV      (SP)+,R2            ;RESTORE R2
10144   INC      RO                  ;ADD 1 TO DATA ADR
10145   RTS      PC                  ;EXIT IN-LINE
10146
10147   GPBCSU: MOV      @#KPAR4,-(SP)  ;SAVE KERNEL'S PAR 4 &
10148   MOV      @#KPDR4,-(SP)      ;PDR 4
10149   MOV      RO,-(SP)           ;SAVE ORIG ADR
10150   BIC      #17777,RO          ;ISOLATE PAGE SELECT BITS
10151   SWAB     RO                  ;COMPUTE DISPL INTO USER'S
10152   ASL      RO                  ;PAR/PDR STORAGE
10153   ASL      RO
10154   ASL      RO
10155   ASL      RO
10156   SWAB     RO
10157   ADD      #P5CONS+PUPARS,RO    ;ADD IN PROG TBL BASE & PAR DISPL
10158   MOV      (RO),@#KPAR4       ;LOAD PAR 4 &
10159   MOV      16.(RO),@#KPDR4    ;PDR 4
10160   MOV      (SP),RO            ;RESTORE DATA ADR
10161   BIC      #160000,RO         ;RESET PAGE SELECT BITS
10162   BIS      #P4CONS,RO        ;BASE ADR UPON PAGE 4
10163   MOV      R2,PC              ;EXIT IN-LINE
10164   .IFTF
10165
10166                                     ;PUT DATA BYTE IN USER'S ABS/VIRT ADR SPACE
10167                                     :
10168                                     :       JSR      PC,PUTBYT      S/R CALL
10169                                     :       RO CONTAINS ABS/VIRT MEMORY ADR
10170                                     :       R1 CONTAINS THE DATA BYTE
10171                                     :       RO WILL BE INCREMENTED BY 1
10172                                     :       ALL OTHER REGISTERS ARE UNCHANGED
10173                                     :
10174                                     :
10175                                     .IFF
10176 040772' 110120   PUTBYT: MOV      R1,(RO)+      ;STORE DATA BYTE
10177 040774' 000207   RTS      PC              ;EXIT IN-LINE
10178                                     .IFT
10179   PUTBYT: JSR      R2,GPBCSU      ;DO COMMON SET UP
10180   MOV      R1,(RO)            ;STORE DATA BYTE IN MEMORY
    
```

GO1

```

10181          BR      GPBEX          ;GO TO EXIT
10182          .ENDC
10183          000
10184
10185          ;SAVE REGISTERS R0 THRU R4 S/R
10186          ;
10187          ;      JSR      R4,SREG      S/R CALL
10188
10189          SREG:  MOV      R3,-(SP)      ;SAVE REGS R3 - R0
10190          MOV      R2,-(SP)      ;R4 SAVED BY JSR TO THIS S/R
10191          MOV      R1,-(SP)
10192          MOV      R0,-(SP)
10193          MOV      R4,PC          ;EXIT IN-LINE
10194
10195
10196          ;RELOAD REGISTERS R0 THRU R4 S/R
10197          ;
10198          ;      JSR      R4,RREG      S/R CALL
10199
10200          RREG:  TST      (SP)+      ;REMOVE THIS JSR'S R4 FROM STK
10201          MOV      (SP)+,R0      ;RELOAD R0 - R3
10202          MOV      (SP)+,R1
10203          MOV      (SP)+,R2
10204          MOV      (SP)+,R3
10205          RTS      R4          ;RESTORE ORG R4 & EXIT IN-LINE
  
```

10207
10208
10209
10210
10211
10212
10213
10214
10215
10216
10217
10218
10219
10220
10221
10222
10223
10224
10225
10226
10227
10228
10229
10230
10231
10232
10233
10234
10235
10236
10237
10238
10239
10240
10241
10242
10243
10244
10245
10246
10247
10248
10249
10250
10251
10252
10253
10254
10255

.SBTTL PACKED DECIMAL TO ASCII SUBROUTINE

```

*****
PACKED DECIMAL TO ASCII SUBROUTINE

THIS SUBROUTINE CONVERTS FROM 1 TO 6 BCD
DIGITS TO EITHER 6 ASCII DIGITS WITH
LEADING ZEROES, OR AN EQUIVALENT ASCII NUMBER
WITH LEADING BLANKS IGNORED.

CALLING SEQUENCE      JSR      R5,DECASC

ENTRY  R0 POINTS TO START OF PACKED CODE
       R1 POINTS TO START OF RESULT AREA

EXIT   R0 POINTS ONE BYTE BEYOND PACKED NUMBER.
       R1 POINTS ONE BYTE BEYOND ASCII NUMBER

DESTROYS R2

*****

```

```

041024' 010346
041026' 010446
041030' 012702 000003
041034' 112003
041036' 110304
041040' 106003
041042' 106003
041044' 106003
041046' 106003
041050' 142703 000360
041054' 152703 000060
041060' 122703 000077
041064' 001401
041066' 110321
041070' 142704 000360
041074' 152704 000060
041100' 122704 000077
041104' 001401
041106' 110421
041110' 005302
041112' 001350
041114' 012604
041116' 012603
041120' 000205

```

```

DECASC: MOV      R3, -(SP)
        MOV      R4, -(SP)
        MOV      #3, R2
LDASLP: MOVB    (R0)+, R3
        MOVB    R3, R4
        RORB   R3
        RORB   R3
        RORB   R3
        RORB   R3
        BICB   #360, R3
        BISB   #60, R3
        CMPB   #77, R3
        BEQ    LSKIP1
        MOVB   R3, (R1)+
LSKIP1: BICB   #360, R4
        BISB   #60, R4
        CMPB   #77, R4
        BEQ    LSKIP2
        MOVB   R4, (R1)+
LSKIP2: DEC     R2
        BNE    LDASLP
        MOV    (SP)+, R4
        MOV    (SP)+, R3
        RTS

```

```

;SAVE HOLY REGISTERS
;SET COUNTER = 3
;GET TWO PACKED DIGITS
;SAVE SECOND DIGIT

;SHIFT FIRST RIGHT FOUR

;MASK OUT LEFT 4 BITS
;CONVERT TO ASCII
;CHECK FOR A NULL
;SKIP IF TRUE
;STORE ASCII DIGIT
;MASK LEFT FOUR BITS DIGIT 2
;CONVERT TO ASCII
;CHECK FOR A NULL
;SKIP IF TRUE
;STORE ASCII DIGIT

;LOOP FOR OTHER DIGITS

;RESTORE HOLY REGISTERS
;RETURN TO CALLING PROGRAM

```


10257
10258
10259
10260
10261
10262
10263
10264
10265
10266
10267
10268
10269
10270
10271
10272
10273
10274
10275
10276
10277
10278
10279
10280
10281
10282
10283
10284
10285
10286
10287
10288
10289
10290
10291
10292
10293
10294
10295
10296
10297
10298
10299
10300
10301
10302
10303
10304
10305
10306
10307
10308
10309
10310
10311
10312

.SBTTL BINARY TO ASCII SUBROUTINE

```

*****
THIS SUBROUTINE CONVERTS ONE OR TWO 16 BIT
BINARY NUMBERS INTO 6 OR 8 ASCII DIGITS

CALLING SEQUENCE      JSR      R5,BINASC (BINAS8) (BINAS1)
                      .WORD    RESULT-.

ENTRY  R0 CONTAINS A 16 BIT UNSIGNED NUMBER
       R2 CONTAINS UP TO 8 ADDITIONAL BITS OF
       HIGHER SIGNIFICANCE THAT WILL BE
       CONVERTED IF THE BINAS8 ENTRANCE IS
       USED. THESE BITS WILL BE RIGHT JUSTIFIED.
       RESULT IS THE LABEL OF THE AREA IN WHICH
       THE ASCII DIGITS ARE TO BE PLACED (6 IF ENTRY
       AT BINASC, 8 IF AT BINAS8).
       OUTPUT ADR IS ALREADY IN R1 ON BINAS1 ENTRY.
EXIT   R1 CONTAINS ADR OF RIGHTMOST BYTE + 1 IN THE
       OUTPUT AREA.
DESTROYS R0,R2
*****

```

001

.IF DF MM
.IFF

```

BINAS8:
BINASC: MOV      R5,R1
        ADD      (R5)+,R1          ;CALCULATE RESULT ADR
BINAS1: MOVB     #30,(R1)
        SEC
LDA1:   ROL      R0                ;CONVERT TO ASCII
        ROLB     (R1)+
        MOVB     (R1),R2
        MOVB     #206,(R1)
LDA2:   ASL      R0
        BEQ      LDA3
        ROLB     (R1)
        BCS      LDA2
        BR       LDA1
LDA3:   MOVB     R2,(R1)
        RTS      R5                ;RETURN TO CALLING PGM
        .IFT
BINAS8: MOV      R3,-(SP)          ;SAVE R3
        MOV      #8,R3            ;SET DIGIT CTR TO 8
        BR       BASCRA          ;GO CALC RESULT ADDR
BINAS1: MOV      R3,-(SP)          ;SAVE R3
        MOV      #6,R3            ;SET DIGIT CTR TO 6
        CLR      R2
        BR       BASCVT          ;GO CONVERT
BINASC: MOV      R3,-(SP)          ;SAVE R3

```

```

041122' 010501
041124' 062501
041126' 112711 000030
041132' 000261
041134' 006100
041136' 106121
041140' 111102
041142' 112711 000206
041146' 006300
041150' 001403
041152' 106111
041154' 103774
041156' 000766
041160' 110211
041162' 000205

```

```

10313          MOV      #6,R3          ;SET DIGIT CNTR TO 6
10314          CLR      R2
10315          BASCRA:  MOV      R5,R1
10316          ADD      (R5)+,R1      ;CALCULATE RESULT ADDRESS
10317          BASCVT:  ADD      R3,R1
10318          MOV      R1,-(SP)      ;SAVE RESULT END PTR
10319          MOV      R4,-(SP)      ;SAVE R4
10320          BASLP1:  MOV      R0,R4  ;GET BINARY NBR
10321          BIC      #177770,R4    ;MASK ALL BUT 3 BITS
10322          BISB     #60,R4        ;CONVERT TO ASCII
10323          MOVB     R4,-(R1)      ;STORE AS RESULT
10324          MOV      #3,R4
10325          BASLP2:  ROR      R2    ;MOVE NEXT 3 BITS
10326          ROR      R0           ;INTO POSITION
10327          DEC      R4
10328          BNE     BASLP2
10329          DEC      R3           ;COUNT DIGITS
10330          BNE     BASLP1       ;REPEAT FOR ALL
10331          MOV      (SP)+,R4     ;RESTORE R4
10332          MOV      (SP)+,R1     ;RESTORE RESULT PTR
10333          MOV      (SP)+,R3     ;RESTORE R3
10334          RTS      R5          ;RETURN
10335          .ENDC

```

000

10337
10338
10339
10340
10341
10342
10343
10344
10345
10346
10347
10348
10349
10350
10351
10352
10353
10354
10355
10356
10357
10358
10359
10360
10361
10362
10363
10364
10365
10366
10367
10368
10369
10370
10371
10372
10373
10374
10375
10376
10377
10378
10379
10380
10381
10382
10383
10384
10385
10386
10387
10388
10389
10390
10391
10392

.SBTTL BINARY TO ASCII DECIMAL SUBROUTINE

THIS SUBROUTINE CONVERTS ONE 16 BIT BINARY
NUMBER INTO ONE TO FIVE ASCII DIGITS
REPRESENTING THE DECIMAL VALUE OF THE NUMBER

CALLING SEQUENCE JSR R5,BTASLB (BTASLZ)
.WORD RESULT-

ENTRY R0 CONTAINS A 16 BIT UNSIGNED NUMBER
RESULT IS THE LABEL OF THE AREA IN
WHICH THE ASCII DIGITS ARE TO BE PLACED.
THIS AREA MUST BE 6 BYTES & DIGITS ARE
RIGHT JUSTIFIED.

BTASLB IS ENTRY FOR LEADING BLANKS
BTASLZ IS ENTRY FOR LEADING ZEROES

EXIT R1 POINTS AT LAST BYTE OF RESULT

DESTROYS R0,R1,R2

041164' 012702 000040
041170' 000402
041172' 012702 000060
041176' 010346
041200' 010446
041202' 010501
041204' 062501
041206' 005004
041210' 005700
041212' 001001
041214' 005204
041216' 010703
041220' 062703 000072
041224' 110221
041226' 112711 000061
041232' 161300
041234' 103414
041236' 112702 000060
041242' 161300
041244' 103402
041246' 105211
041250' 000774
041252' 061300
041254' 022723 000001
041260' 001404
041262' 005201
041264' 000760
041266' 110211
041270' 000770

BTASLB: MOV #40,R2 ;STORE SPACE AS FILL
BR LBIASD
BTASLZ: MOV #60,R2 ;STORE ZERO AS FILL
LBIASD: MOV R3,-(SP) ;SAVE R3 & R4
MOV R4,-(SP)
MOV R5,R1 ;GET ADR OF RESULT AREA
ADD (R5)+,R1 ;CALCULATE RESULT ADDR.
CLR R4 ;CLEAR ZERO FLAG
TST R0 ;IS NBR ZERO?
BNE LBDSTP ;Y,N-LBDSTP
INC R4 ;SET THE ZERO FLAG
LBDSTP: MOV PC,R3
ADD #LDECTB+2-.,R3 ;SET TABLE POINTER
MOVB R2,(R1)+ ;STORE A FILL CHAR
LBDLP1: MOVB #61,(R1) ;STORE AN ASCII ONE
SUB (R3),R0 ;NUMBER MINUS TBL VALUE
BCS LBDSFL ;BRANCH IF NEGATIVE
MOVB #60,R2 ;SET FILL CHAR = 0
LBDLP2: SUB (R3),R0 ;NUMBER MINUS TBL VALUE
BCS LBDDDN ;DIGIT DONE IF NEG
INCB (R1) ;INCREMENT RESULT DIGIT
BR LBDLP2 ;AROUND AGAIN
LBDDDN: ADD (R3),R0 ;RESTORE POS NBR
CMP #1,(R3)+ ;CONVERSION DONE
BEQ LBOTZF ;YES-GO CHECK ZERO FLAG
INC R1 ;BUMP OUTPUT POINTER
BR LBDLP1 ;GO WORK ON NEXT DIGIT
LBDSFL: MOVB R2,(R1) ;STORE THE FILL CHAR
BR LBDDDN ;CHECK IF FINISHED

L01

MAINDEC-11-DTMGA-C MPG / USER PROGRAM SHARED CODE
DTMGAC.P11 BINARY TO ASCII DECIMAL SUBROUTINE

MACY11 27(732) 24-SEP-76 13:54 PAGE 56-1

SEQ 0219

10393	041272'	005704		LBDTZF: TST	R4		;TEST THE ZERO FLAG
10394	041274'	001402			BEQ	LBDEXT	;IF NOT SET, EXIT
10395	041276'	112711	000060		MOVB	#60, (R1)	;ELSE INSERT A ZERO
10396	041302'	012604		LBDEXT: MOV	(SP)+, R4		;RESTORE R3 & R4
10397	041304'	012603			MOV	(SP)+, R3	
10398	041306'	000205			RTS	R5	;RETURN
10399							
10400	041310'	177777		LDECTB: .WORD	65535.		;DECIMAL TO BINARY TABLE
10401	041312'	023420			.WORD	10000.	
10402	041314'	001750			.WORD	1000.	
10403	041316'	000144			.WORD	100.	
10404	041320'	000012			.WORD	10.	
10405	041322'	000001			.WORD	1	

```

10407
10408
10409
10410
10411
10412
10413
10414
10415
10416
10417
10418
10419 041324' 010346
10420 041326' 016700 000104
10421 041332' 016701 000076
10422 041336' 012703 177771
10423 041342' 005002
10424 041344' 006300
10425 041346' 006101
10426 041350' 006102
10427 041352' 005203
10428 041354' 001373
10429 041356' 066700 000054
10430 041362' 005501
10431 041364' 066701 000044
10432 041370' 005502
10433 041372' 062700 001057
10434 041376' 005501
10435 041400' 005502
10436 041402' 062701 047401
10437 041406' 005502
10438 041410' 062702 000006
10439 041414' 060200
10440 041416' 005501
10441 041420' 010067 000012
10442 041424' 010167 000004
10443 041430' 012603
10444 041432' 000205
10445 041434' 176543
10446 041436' 123456

```

.SBTTL RANDOM NUMBER GENERATOR ROUTINE

```

*****
*          JSR      R5, LRANDM          ;CALL THE ROUTINE
*          RETURN                    ;RETURN HERE THE RANDOM
*                                     ;NUMBER WILL BE IN
*                                     ;LHINUM, LLONUM
*****

LRANDM:  MOV      R3, -(SP)           ;SAVE R3
         MOV      LLONUM, R0         ;SET R0 WITH LOW
         MOV      LHINUM, R1         ;SET R1 WITH HIGH
         MOV      #7, R3             ;SET SHIFT COUNT
         CLR      R2                 ;ZERO R2
1$:      ASL      R0                 ;SHIFT R0 SEFT AND
         ROL      R1                 ;ROTATE CARRY INTO R1 AND
         ROL      R2                 ;ROTATE CARRY INTO R2
         INC      R3                 ;CHECK FOR DONE
         BNE     1$                 ;CONTINUE SHIFT $OOP
         ADD      LLONUM, R0         ;ADD NUMBER TO MAKE X 129
         ADC      R1                 ;PROPOGATE CARRY
         ADD      LHINUM, R1         ;ADD NUMBER TO MAKE X 129
         ADC      R2                 ;PROPOGATE CARRY
         ADD      #1057, R0          ;ADD LOW CONSTANT
         ADC      R1                 ;PROPOGATE CARRY
         ADC      R2                 ;PROPOGATE CARRY
         ADD      #47401, R1         ;ADD HIGH CONSTANT
         ADC      R2                 ;PROPOGATE CARRY
         ADD      #6, R2             ;ADD HIGHEST CONSTART
         ADD      R2, R0             ;REPRIME R0 WITH HIGHEST DIGIT
         ADC      R1                 ;PROPOGATE CARRY
         MOV      R0, LLONUM         ;SAVE R0
         MOV      R1, LHINUM         ;SAVE R1
         MOV      (SP)+, R3          ;RESTORE R3
         RTS      R5                 ;RETURN

LHINUM:  .WORD   176543
LLONUM:  .WORD   123456

```

		.SBTTL DATA BUFFERS, COMMON WORDS, & PATTERN ADDRESSES TABLE			
10448					
10449					
10450					
10451	041440'	000000	000000	000000	CBUF00: .EVEN .WORD 0,0,0,0,0,0,0,0 ;COMMON DATA BUFFERS
	041446'	000000	000000	000000	
	041454'	000000	000000	000000	
10452	041460'	000000	000000	000000	CBUF01: .WORD 0,0,0,0,0,0,0,0
	041466'	000000	000000	000000	
	041474'	000000	000000	000000	
10453	041500'	000000	000000	000000	CBUF02: .WORD 0,0,0,0,0,0,0,0
	041506'	000000	000000	000000	
	041514'	000000	000000	000000	
10454	041520'	000000	000000	000000	CBUF03: .WORD 0,0,0,0,0,0,0,0
	041526'	000000	000000	000000	
	041534'	000000	000000	000000	
10455	041540'	000000	000000	000000	CBUF04: .WORD 0,0,0,0,0,0,0,0
	041546'	000000	000000	000000	
	041554'	000000	000000	000000	
10456	041560'	000000	000000	000000	CBUF05: .WORD 0,0,0,0,0,0,0,0
	041566'	000000	000000	000000	
	041574'	000000	000000	000000	
10457	041600'	000000	000000	000000	CBUF06: .WORD 0,0,0,0,0,0,0,0
	041606'	000000	000000	000000	
	041614'	000000	000000	000000	
10458	041620'	000000	000000	000000	CBUF07: .WORD 0,0,0,0,0,0,0,0
	041626'	000000	000000	000000	
	041634'	000000	000000	000000	
10459	041640'	000000	000000	000000	CBUF08: .WORD 0,0,0,0,0,0,0,0
	041646'	000000	000000	000000	
	041654'	000000	000000	000000	
10460	041660'	000000	000000	000000	CBUF09: .WORD 0,0,0,0,0,0,0,0
	041666'	000000	000000	000000	
	041674'	000000	000000	000000	
10461	041700'	000000	000000	000000	CBUF10: .WORD 0,0,0,0,0,0,0,0
	041706'	000000	000000	000000	
	041714'	000000	000000	000000	
10462	041720'	000000	000000	000000	CBUF11: .WORD 0,0,0,0,0,0,0,0
	041726'	000000	000000	000000	
	041734'	000000	000000	000000	
10463	041740'	000000	000000	000000	CBUF12: .WORD 0,0,C,0,0,0,0,0
	041746'	000000	000000	000000	
	041754'	000000	000000	000000	
10464	041760'	000000	000000	000000	CBUF13: .WORD 0,0,0,0,0,0,0,0
	041766'	000000	000000	000000	
	041774'	000000	000000	000000	
10465	042000'	000000	000000	000000	CBUF14: .WORD 0,0,0,0,0,0,0,0
	042006'	000000	000000	000000	
	042014'	000000	000000	000000	
10466	042020'	000000	000000	000000	CBUF15: .WORD 0,0,0,0,0,0,0,0
	042026'	000000	000000	000000	
	042034'	000000	000000	000000	

```

10468 042040' 000000 LCOM0: .WORD 0 ;COMMON DATA WORD STORAGE
10469 042042' 000000 LCOM1: .WORD 0
10470 042044' 000000 LCOM2: .WORD 0
10471 042046' 000000 LCOM3: .WORD 0
10472 042050' 000000 LCOM4: .WORD 0
10473 042052' 000000 LCOM5: .WORD 0
10474 042054' 000000 LCOM6: .WORD 0
10475 042056' 000000 LCOM7: .WORD 0
10476 042060' 000000 LCOM8: .WORD 0
10477 042062' 000000 LCOM9: .WORD 0
10478
10479
10480 042064' 000004 LPAT0P: .WORD LPAT0-LPBASE ;PATTERN ADDRESS TABLE
10481 042066' 000036 LPAT1P: .WORD LPAT1-LPBASE
10482 042070' 000070 LPAT2P: .WORD LPAT2-LPBASE
10483 042072' 000120 LPAT3P: .WORD LPAT3-LPBASE
10484 042074' 000172 LPAT4P: .WORD LPAT4-LPBASE
10485 042076' 000204 LPAT5P: .WORD LPAT5-LPBASE
10486 042100' 000216 LPAT6P: .WORD LPAT6-LPBASE
10487 042102' 000230 LPAT7P: .WORD LPAT7-LPBASE
10488 042104' 000242 LPAT8P: .WORD LPAT8-LPBASE
10489 042106' 000254 LPAT9P: .WORD LPAT9-LPBASE
10490 042110' 000266 LPATAP: .WORD LPATA-LPBASE
10491 042112' 000314 LPATBP: .WORD LPATB-LPBASE
10492 042114' LPATEN= .
10493
10494 042114' 005015 LCRETN: .WORD 5015
10495
10496
10497 042116' PG6END= . ;CODE BEYOND THIS POINT IS NOT REQUIRED TO BE IN
10498 ;THE SHARED PAGE AREA.
10499
10500
10501 001 .IF NOF MIMIC
10502 002 .IF DF MM
10503 .IFF
10504 .TITLE MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
10505 .IFT
10506 .TITLE MAINDEC-11-DTMMA-B MPG CONTROL ROUTINE
10507 001 .ENDC
10508 000 .ENDC
10509 001 .IF DF MIMIC
10510 002 .IF DF MM
10511 .IFF
10512 .TITLE MAINXEC-11-DTMGA-C MPG CONTROL ROUTINE
10513 .IFT
10514 .TITLE MAINXEC-11-DTMMA-B MPG CONTROL ROUTINE
10515 001 .ENDC
10516 000 .ENDC
    
```

10518
10519
10520
10521
10522
10523
10524
10525
10526
10527
10528
10529
10530
10531
10532
10533
10534
10535
10536
10537
10538
10539
10540
10541
10542
10543
10544
10545
10546
10547
10548
10549
10550
10551
10552
10553
10554
10555
10556
10557
10558
10559
10560
10561
10562
10563
10564
10565
10566
10567
10568
10569
10570
10571
10572
10573

.SBTTL DECIMAL OR OCTAL TO BINARY SUBROUTINE

```

*****
THIS SUBROUTINE CONVERTS FROM 1 TO 6 PACKED BCD DIGITS (3
BYTES), THAT REPRESENT OCTAL OR DECIMAL NUMBERS, INTO ONE
16 BIT BINARY NUMBER. ANY NUMBER LARGER THAN
65,535 WILL CAUSE AN ERROR EXIT.

CALLING SEQUENCE      JSR      R5,DECBIN      (OCTBIN)
                      .WORD    ERRTN-.

ENTRY   R0 CONTAINS ADDRESS OF DECIMAL NBR (FIRST BYTE)
        ERRTN IS THE ERROR RETURN ADDRESS TAG

EXIT   R0 CONTAINS RESULT

DESTROYS R2
*****

```

```

042116' 010702
042120' 062702 177170
042124' 010146
042126' 005001
042130' 000406
042132' 010702
042134' 062702 000172
042140' 010146
042142' 012701 000001
042146' 010346
042150' 010446
042152' 010546
042154' 005003
042156' 012704 000006
042162' 111005
042164' 106005
042166' 106005
042170' 106005
042172' 106005
042174' 000401
042176' 112005
042180' 042705 177760
042184' 005701
042206' 001406
042210' 120527 000010
042214' 001432
042216' 120527 000011
042222' 001427
042224' 122705 000017
042230' 001406
042232' 105705
042234' 001404
042236' 061203

```

```

DECBIN: MOV      PC,R2          ;SET POINTER R2
        ADD      #LDECTB-. ,R2 ;TO DECIMAL TABLE
        MOV      R1,-(SP)      ;SAVE R1
        CLR      R1            ;CLEAR OCTAL FLAG
        BR       LTBSAV        ;GO SAVE REGS
OCTBIN: MOV      PC,R2          ;SET POINTER R2
        ADD      #LOCTTB-. ,R2 ;TO OCTAL TABLE
        MOV      R1,-(SP)      ;SAVE R1
        MOV      #1,R1         ;SET OCTAL FLAG
LTBSAV: MOV      R3,-(SP)
        MOV      R4,-(SP)      ;SAVE HOLY REGISTERS
        MOV      R5,-(SP)
        CLR      R3            ;CLEAR RESULT WORD
        MOV      #6,R4         ;SET UP DIGIT COUNT
LTBIN0: MOVB     (R0),R5        ;ALIGN BYTE TO GET
        RORB     R5            ;FIRST DIGIT IN BYTE
        RORB     R5
        RORB     R5
        BR       LTBIN2
LTBIN1: MOVB     (R0)+,R5
LTBIN2: BIC      #177760,R5
        TST      R1
        BEQ     LTbfd
        CMPB    R5,#8
        BEQ     LTberr
        CMPB    R5,#9
        BEQ     LTberr
LTbfd:  CMPB     #17,R5
        BEQ     LTbin4
        TSTB    R5
        BEQ     LTbin4
LTbin3: ADD      (R2),R3
        ;GO PROCESS DIGIT
        ;GET INPUT BYTE AGAIN
        ;RESET ANY OTHER BITS
        ;BRANCH IF OCTAL FLAG
        ;NOT SET
        ;CHECK IF DECIMAL NBR
        ;ERROR IF TRUE
        ;CHECK IF DECIMAL NBR
        ;ERROR IF TRUE
        ;ALL ONES ?
        ;TRY NEXT DIGIT
        ;ALL ZEROES ?
        ;TRY NEXT DIGIT
        ;ADD TABLE VALUE TO RESULT

```



```

10574 042240' 103420          BCS      LTBERR          ;ERROR IF CARRY SET
10575 042242' 005305          DEC      R5
10576 042244' 001374          BNE     LTBIN3          ;REPEAT UNTIL DIGIT = 0
10577 042246' 005722          LTBIN4: TST    (R2)+      ;BUMP TABLE POINTER
10578 042250' 005304          DEC     R4              ;DECR DIGIT CNT
10579 042252' 001404          BEQ     LTBINS          ;DONE 6 DIGITS? (N,Y-LTBINS)
10580 042254' 032704 000001  BIT     #1,R4          ;2ND DIGIT OF BYTE NEXT?
10581 042260' 001346          BNE     LTBIN1          ;N,Y-LTBIN1
10582 042262' 000737          BR      LTBINO          ;DO 1ST DIGIT OF NXT BYTE
10583 042264' 114004          LTBIN5: MOVB   -(R0),R4  ;GET BYTE WITH LAST DIGIT
10584 042266' 042704 177760  BIC     #177760,R4     ;ISOLATE LAST DIGIT
10585 042272' 001006          BNE     LTBOU          ;WAS IT ZERO? (Y,N-LTBOU)
10586 042274' 010304          MOV     R3,R4          ;MOVE RESULT TO A WORK REG
10587 042276' 005204          INC     R4              ;INCREMENT RESULT
10588 042300' 103003          BCC     LTBOU          ;ERROR IF CARRY GETS SET
10589 042302' 012605          LTBERR: MOV   (SP)+,R5  ;RESTORE R5
10590 042304' 061505          ADD     (R5),R5        ;CALCULATE ERR RET ADR
10591 042306' 000403          BR      LBTEX          ;GO TO S/R EXIT
10592 042310' 010300          LTBOU: MOV   R3,R0     ;MOVE RESULT TO RETURN REG
10593 042312' 012605          MOV     (SP)+,R5        ;RESTORE R5
10594 042314' 005725          TST    (R5)+          ;BYPASS ERR RET ADR
10595 042316' 012604          LBTEX: MOV   (SP)+,R4  ;RESTORE HOLY REGISTERS
10596 042320' 012603          MOV     (SP)+,R3
10597 042322' 012601          MOV     (SP)+,R1
10598 042324' 000205          RTS      R5            ;RETURN TO CALLING PGM
10599
10600 042326' 100000          LOCTTB: .WORD 100000   ;OCTAL TO BINARY TABLE
10601 042330' 010000          .WORD 10000
10602 042332' 001000          .WORD 1000
10603 042334' 000100          .WORD 100
10604 042336' 000010          .WORD 10
10605 042340' 000001          .WORD 1
10606          001          .IF DF MM
10607          .PAGE
10608          .SBTTL ASCII OCTAL TO BINARY SUBROUTINE

```

```

*****
THIS SUBROUTINE CONVERTS FROM 1 TO 8 ASCII
CHARACTERS THAT REPRESENT THE NUMBERS FROM
0 THRU 7, TO THE EQUIVALENT BINARY NUMBER.
NUMBERS GREATER THAN 8 DIGITS OR NON-NUMERIC
CHARACTERS WILL CAUSE AN ERROR EXIT.

CALLING SEQUENCE      JSR      R5,ASOCBN
                      .WORD    ERRTN-

ENTRY      R0 CONTAINS ADDRESS OF LAST DIGIT
           R1 CONTAINS A COUNT OF THE NUMBER OF DIGITS
           ERRTN IS THE ERROR RETURN ADDRESS TAG

EXIT      R0 CONTAINS LSB OF RESULT
           R2 CONTAINS MSB OF RESULT
*****

```

10610
10611
10612
10613
10614
10615
10616
10617
10618
10619
10620
10621
10622
10623
10624
10625
10626
10627
10628
10629

DTMGAC.P11

DECIMAL OR OCTAL TO BINARY SUBROUTINE

```

10630
10631 ASOCBN: MOV R3,-(SP) ;SAVE R3 AND R4
10632 MOV R4,-(SP)
10633 CMP R1,#8 ;HOW MANY DIGITS?
10634 BLE LASOB2 ;BR IF 8 OR LESS DIGITS
10635 MOV R0,R3 ;CK FOR & BYPASS LEADING 0'S
10636 SUB R1,R3 ;SET POINTER R3
10637 INC R3 ;TO FIRST DIGIT
10638 LASOB1: CMPB (R3),#60 ;IS IT A ZERO?
10639 BNE LASOBE ;NO-ERROR
10640 DEC R1 ;IS NEXT DIGIT
10641 CMP R1,#8 ;THE FIRST OF 8?
10642 BGT LASOB1 ;NO-CONTINUE CHECK
10643 LASOB2: MOV #9,R4 ;COMPUTE # OF FILL DIGITS + 1
10644 SUB R1,R4
10645 MOV R0,R3
10646 CLR R2 ;PUT DATA ADR IN WORK REG
10647 LASOB3: CMPB (R3),#60 ;CLEAR HIGH ORDER BIT REG
10648 BLO LASOBE ;IF LESS THAN ASCII 0
10649 CMPB (R3),#67 ;TAKE ERROR EXIT
10650 BHI LASOBE ;IF MORE THAN ASCII 7
10651 MOVB (R3),-(SP) ;TAKE ERROR EXIT
10652 BICB #370,(SP) ;ABSORB THE DIGIT
10653 JSR PC,LASHFT ;EXTRACT OCTAL DIGIT BITS
10654 DEC R3 ;SHIFT THEM INTO THE RESULT
10655 DEC R1 ;POINT TO NEXT CHAR
10656 BNE LASOB3 ;IF MORE DIGITS TO GO
10657 LASOB4: DEC R4 ;GO GET THEM
10658 BEQ LASOB5 ;DECR FILL DIGIT CNT
10659 CLR -(SP) ;ANY MORE TO DO? BR IF NO
10660 JSR PC,LASHFT ;SET UP FILL DIGIT BITS
10661 BR LASOB4 ;SHIFT IN FILL DIGIT
10662 LASOB5: TST (R5)+ ;GO CK FOR MORE
10663 LASOB6: MOV (SP)+,R4 ;BYPASS ERR RET ADR
10664 MOV (SP)+,R3 ;RESTORE R4 AND R3
10665 RTS R5 ;EXIT
10666
10667 LASOBE: ADD (R5),R5 ;CALCULATE ERROR RETN
10668 BR LASOB6 ;GO TO SUBR EXIT
10669
10670
10671 LASHFT: MOV #3,-(SP) ;SET UP LOOP CNT
10672 LASHF1: ROR 4(SP) ;SHIFT BIT OUT OF OCTAL DIGIT
10673 RORB R2 ;SHIFT IT INTO RESULT REGS
10674 ROR R0
10675 DEC (SP) ;DONE 3 BITS?
10676 BNE LASHF1 ;Y,N-LASHF1
10677 TST (SP)+ ;REMOVE LOOP CNT FROM STK
10678 MOV (SP)+,(SP) ;REMOVE DIGIT FROM STK
10679 RTS PC ;EXIT IN-LINE
10680 .ENDC

```

000

10682
10683
10684
10685
10686
10687
10688
10689
10690
10691
10692
10693
10694
10695
10696
10697
10698
10699
10700
10701
10702
10703
10704
10705
10706
10707
10708
10709
10710
10711
10712
10713
10714
10715
10716
10717
10718
10719
10720
10721
10722
10723
10724
10725
10726
10727
10728
10729
10730
10731
10732
10733
10734
10735
10736
10737

.SBTTL ASCII TO PACKED DECIMAL SUBROUTINE

```

*****
THIS SUBROUTINE CONVERTS FROM 1 TO 6 ASCII
CHARACTERS THAT REPRESENT THE NUMBERS FROM
0 THRU 9, TO THE EQUIVALENT PACKED DECIMAL
FORMAT, AND STORES THEM IN AN INTERNAL BUFFER
CALLED DECBUF. NUMBERS GREATER THAN 6 DIGITS
OR NON-NUMERIC CHARACTERS WILL CAUSE AN ERROR EXIT.

CALLING SEQUENCE      JSR      R5,ASDECO  (ASDEC1)
                      .WORD    ERRTN-.

ENTRY  R0 CONTAINS ADDRESS OF LAST DIGIT
       R1 CONTAINS A COUNT OF THE NUMBER OF DIGITS.
       ERRTN IS THE ERROR RETURN ADDRESS TAG
       ASDECO FILLS LEADING DIGITS WITH ZEROES
       ASDEC1 FILLS LEADING DIGITS WITH ONES

EXIT  DECBUF CONTAINS PACKED NUMBER (3 BYTES)
      R0 POINTS TO DECBUF

DESTROYS R1,R2
*****
    
```

042342' 012702 177777
 042346' 000401
 042350' 005002
 042352' 010346
 042354' 010446
 042356' 020127 000006
 042362' 003412
 042364' 010003
 042366' 160103
 042370' 005203
 042372' 122327 000060
 042376' 001052
 042400' 005301
 042402' 020127 000006
 042406' 003371
 042410' 012704 000006
 042414' 160104
 042416' 010603
 042420' 121027 000060
 042424' 103436
 042426' 121027 000071
 042432' 101033
 042434' 111046
 042436' 005300
 042440' 005301
 042442' 001366

```

ASDEC1: MOV      #177777,R2      ;SET UP FILL CHAR OF ALL ONES
        BR       LASDEC
ASDECO: CLR      R2             ;SET UP FILL CHAR OF 0'S
LASDEC: MOV      R3,-(SP)
        MOV      R4,-(SP)      ;SAVE HOLY REGISTERS
        CMP      R1,#6
        BLE      LASCNT        ;CONTINUE IF 6 OR LESS DIGITS
        MOV      R0,R3         ;CK FOR & BYPASS LEADING 0'S
        SUB      R1,R3         ;SET POINTER R3
        INC      R3           ;TO FIRST DIGIT
LASLPO: CMPB     (R3)+,#60      ;IS IT A ZERO ?
        BNE      LASER1        ;NO-ERROR
        DEC      R1           ;IS NEXT DIGIT
        CMP      R1,#6         ;THE FIRST OF 6 ?
        BGT      LASLPO        ;NO - CONTINUE CHECK
LASCNT: MOV      #6,R4         ;COMPUTE # OF FILL CHAR'S
        SUB      R1,R4
        MOV      SP,R3
LASLPI: CMPB     (R0),#60      ;SAVE STK PNTR IF AN ERR
        BLO      LASERR        ;IF LESS THAN ASCII 0
        BLO      LASERR        ;TAKE ERROR EXIT
        CMPB     (R0),#71      ;IF MORE THAN ASCII 9
        BHI      LASERR        ;TAKE ERROR EXIT
        MOVB     (R0),-(SP)    ;ABSORB THE DIGIT
        DEC      R0           ;POINT TO NXT CHAR
        DEC      R1           ;IF MORE DIGITS TO GO
        BNE      LASLPI        ;GO GET THEM
    
```

10738	042444'	005704		TST	R4		;NEED FILL CHAR'S?
10739	042446'	001403		BEQ	LASLP3		;Y N-LASLP3
10740	042450'	010246		LASLP2: MOV	R2,-(SP)		;STORE A FILL CHAR
10741	042452'	005304		DEC	R4		;DECR FILL CHAR CNT
10742	042454'	001375		BNE	LASLP2		;DONE ALL? (Y,N-LASLP2)
10743	042456'	004767	000046	LASLP3: JSR	PC,LASPAK		;PACK 1ST TWO DIGITS
10744	042462'	010401		MOV	R4,R1		;SAVE THE BYTE
10745	042464'	004767	000040	JSR	PC,LASPAK		;PACK NXT 2 DIGITS
10746	042470'	010402		MOV	R4,R2		;SAVE THEM
10747	042472'	004767	000032	JSR	PC,LASPAK		;PACK LAST 2 DIGITS
10748	042476'	010700		MOV	PC,R0		;SET UP ADR OF DECBUF
10749	042500'	062700	000065	ADD	#DECBUF+2-.,R0		
10750	042504'	110410		MOVB	R4,(R0)		;STORE 3 BYTES IN
10751	042506'	110240		MOVB	R2,-(R0)		;DECBUF
10752	042510'	110140		MOVB	R1,-(R0)		
10753	042512'	005725		TST	(R5)+		;BYPASS ERR RET ADR
10754	042514'	012604		LASEX: MOV	(SP)+,R4		;RESTORE HOLY REGISTERS
10755	042516'	012603		MOV	(SP)+,R3		
10756	042520'	000205		RTS	R5		;RETURN TO CALLING PGM
10757	042522'	010306		LASERR: MOV	R3,SP		;RESTORE STACK PNTR
10758	042524'	061505		LASER1: ADD	(R5),R5		;CALCULATE ERROR RETURN
10759	042526'	000772		BR	LASEX		;GO TO S/R EXIT
10760							
10761							
10762	042530'	012600		LASPAK: MOV	(SP)+,R0		;SAVE RET ADR
10763	042532'	112603		MOVB	(SP)+,R3		;GET BYTE WITH LEFT FOUR BITS
10764	042534'	042703	177760	BIC	#177760,R3		;CLEAR ANY OTHER BITS
10765	042540'	106103		ROLB	R3		;SHIFT THEM LEFT FOUR BITS
10766	042542'	106103		ROLB	R3		
10767	042544'	106103		ROLB	R3		
10768	042546'	106103		ROLB	R3		
10769	042550'	012604		MOV	(SP)+,R4		;GET BYTE WITH RIGHT FOUR BITS
10770	042552'	042704	177760	BIC	#177760,R4		;CLEAR ANY OTHER BITS
10771	042556'	050304		BIS	R3,R4		;MERGE 2 DIGITS INTO 1 BYTE
10772	042560'	010007		MOV	R0,PC		;EXIT IN-LINE
10773							
10774							
10775	042562'	000001					
10776	042563'	000003		DECBUF: .BLKB	1		

10778
10779
10780
10781
10782
10783
10784
10785
10786
10787
10788
10789
10790
10791
10792
10793
10794
10795
10796
10797
10798
10799
10800
10801
10802
10803
10804
10805
10806
10807
10808
10809
10810
10811
10812
10813
10814
10815
10816
10817
10818
10819
10820
10821
10822
10823
10824
10825
10826
10827
10828
10829
10830
10831
10832
10833

.SBTTL VALID DEVICES TABLE

;THE FOLLOWING TABLE WILL BE READ INTO MEMORY BY
;THE ONE TIME HOUSEKEEPING ROUTINE. ITS FILENAME
;ON THE MPG MEDIUM IS "TVDA".MPG".

VALID DEVICE MODEL NAME TABLE

6 BYTE ENTRY FORMAT:

BYTES 0 THRU 3 = 4 CHAR ASCII DEV NAME

BYTE 4 = FLAG BYTE

BIT 0 = 1 = VALID LIST DEV
BIT 1 = 1 = VALID SAVE DEV
BIT 2 = 1 = VALID FETCH DEV
BIT 3 = 1 = PERM SAV/FTH DEV
BIT 4 = DEVICE I.D. BITS: USED TO
& 5 = DISTINGUISH BETWEEN DIFFERENT
& 6 = MODEL NAMES AND ALSO BETWEEN
& 7 = DEVICES USING SAME DEV ROUT.

BYTE 5 = POINTER INTO THE DEVICE
ROUTINE I.D. CODE TBL FOR XX
(TXANN.MPG) OR, IF ODD, THE
I.D. CODE FOR 'NONE', 'KYBD',
'KBCJ', OR 'LOAD'

NOTE: BITS 1 & 2 OF BYTE 4 WILL
BE RESET BY MPG'S 1 TIME HSKP
FOR ALL DEVICES EXCEPT THE
LOAD DEV & PAPER TAPE.

001

.IF DF MIMIC

.IFF

VDFLSZ: .WORD 0
DIDTAD: .WORD 0
CMDLNM:
CNONID: .BLKW 3
CKBDID: .BLKW 3
CLDID: .BLKW 3

;DUMMY ALLOCATIONS USED TO
;PROPERLY ALIGN LABELS IN
;THIS AREA.

043566'

. = VDFLSZ+512.

;ALLOW FOR UP TO 512 BYTE TABLE

;MPG WILL SET THE USER PROGRAM AREA ADDRESS
;TO THE END OF THE TABLE THAT WAS READ IN.

.IFT
VDFLSZ: .WORD CTBEND-
DIDTAD: .WORD DRIDCD-

;NUMBER OF BYTES IN THIS FILE
;RELATIVE ADR OF I.D. TABLE

CMDLNM:
CNONID: .ASCII /NONE/<<377><001.>

10834	CKBDID:	.ASCII	/KYBD/<377><003.>
10835		.ASCII	/KB00/<377><003.>
10836	CLDID:	.ASCII	/LOAD/<377><005.>
10837		.ASCII	/TC11/<006><000.>
10838		.ASCII	/TU56/<026><000.>
10839		.ASCII	/RK11/<006><002.>
10840		.ASCII	/RK05/<026><002.>
10841		.ASCII	/LP11/<001><004.>
10842		.ASCII	/LS11/<021><004.>
10843		.ASCII	/LV11/<041><004.>
10844		.ASCII	/TM11/<006><006.>
10845		.ASCII	/TU10/<026><006.>
10846		.ASCII	/DJ11/<000><008.>
10847		.ASCII	/DL11/<000><010.>
10848		.ASCII	/DQ11/<000><012.>
10849		.ASCII	/PC11/<016><014.>
10850		.ASCII	/PR11/<034><014.>
10851		.ASCII	/RP04/<006><016.>
10852		.ASCII	/RP05/<026><016.>
10853		.ASCII	/RP06/<046><016.>
10854		.ASCII	/DH11/<000><018.>
10855		.ASCII	/TA11/<000><020.>
10856		.ASCII	/RF11/<000><022.>
10857		.ASCII	/RC11/<000><024.>
10858		.ASCII	/RP02/<000><026.>
10859		.ASCII	/RP03/<020><026.>
10860		.ASCII	/DN11/<000><028.>
10861		.ASCII	/DC11/<000><030.>
10862		.ASCII	/CR11/<000><032.>
10863		.ASCII	/CM11/<020><032.>
10864		.ASCII	/CD11/<000><034.>
10865		.ASCII	/KW11/<000><036.>
10866		.ASCII	/TM02/<006><038.>
10867		.ASCII	/TU16/<026><038.>
10868		.ASCII	/RX11/<006><040.>
10869		.ASCII	/RX01/<026><040.>
10870		.ASCII	/DU11/<000><042.>
10871		.ASCII	/RK06/<000><044.>
10872		.ASCII	/RS03/<000><046.>
10873		.ASCII	/RS04/<020><046.>
10874		.ASCII	/RL01/<000><048.>
10875		.ASCII	/RL02/<020><048.>
10876		.ASCII	/???/<377><377>

;TABLE TERMINATOR

;VALID DEV ROUTINE I.D. TABLE

10881	DRIDCD:	.ASCII	/TC/	;TC11/TU56
10882		.ASCII	/RK/	;RK11/RK05
10883		.ASCII	/LP/	;LP11/LS11/LV11
10884		.ASCII	/TM/	;TM11/TU10
10885		.ASCII	/DJ/	;DJ11
10886		.ASCII	/DL/	;DL11
10887		.ASCII	/DQ/	;DQ11
10888		.ASCII	/PC/	;PC11/PR11
10889		.ASCII	/RP/	;RP04/RP05/RP06

10890		.ASCII /DH/	;DH11
10891		.ASCII /MS/	;TA11
10892		.ASCII /MS/	;RF11
10893		.ASCII /MS/	;RC11
10894		.ASCII /R3/	;RP02/RP03
10895		.ASCII /MS/	;DN11
10896		.ASCII /MS/	;DC11
10897		.ASCII /MS/	;CR11/CM11
10898		.ASCII /MS/	;CD11
10899		.ASCII /MS/	;KW11
10900		.ASCII /MS/	;TM02/TU16
10901		.ASCII /MS/	;RX11/RX01
10902		.ASCII /DU/	;DU11
10903		.ASCII /R6/	;RK06
10904		.ASCII /RS/	;RS03/RS04
10905		.ASCII /RL/	;RL01/RL02
10906	COO	CTBEND =	
10907		.ENDC	

.SBTTL MPG CONTROL ROUTINE ONE TIME HOUSEKEEPING

```

*****
:
: THIS CODING WILL BE EXECUTED ONCE WHEN
: MPG IS FIRST LOADED. IT WILL BE DESTROYED
: WHEN THE FIRST USER PROGRAM IS ENTERED OR
: FETCHED.
:
*****

```

```

;R0 CONTAINS THE ADDRESS OF THE END OF MEMORY AS
;COMPUTED BY THE EXEC. (ALWAYS < 32K WORDS)
;R2 CONTAINS THE POSITION ADJUST FACTOR FOR THE
;EXEC'S GLOBAL TABLE. IT WILL BE 0'S ON THE
;NON-MEM MGMNT VERSION.

```

;STORE MEM END ADR & HSKP INTERRUPT VECTORS

```

10930 043566' 010067 135660
10931 043572' 012703 000300
10932 043576' 010304
10933 043600' 005724
10934 043602' 010423
10935 043604' 005023
10936 043606' 062704 000004
10937 043612' 020327 001000
10938 043616' 001371

```

```

C1THKP: MOV R0,CRMEND ;STORE END OF REAL MEMORY ADR
        MOV #300,R3 ;GET VECTOR'S START ADR
        MOV R3,R4 ;SET UP .+2 ADR
        TST (R4)+
10$: MOV R4,(R3)+ ;STORE .+2 ADR
     CLR (R3)+ ;STORE "HALT" INST CODE
     ADD #4,R4 ;INCR .+2 ADR
     CMP R3,#1000 ;JUST DO 774 & 776?
     BNE 10$ ;Y,N-10$

```

;TAILOR & STORE EXEC INTERFACE ADR'S

```

10942 043620' 010700
10943 043622' 062700 134156
10944 043626' 160200
10945 043630' 012701 000007
10946 043634' 010702
10947 043636' 062702 001002
10948 043642' 014203
10949 043644' 060703
10950 043646' 010340
10951 043650' 005301
10952 043652' 001373

```

```

        MOV PC,R0 ;SET UP LOC 0 MPG ADR
        ADD #CLOCZ--,R0
        SUB R2,R0 ;SUB MPG'S POSITION ADJ FACTOR
        MOV #CGBLTE-CGBLTS/2,R1 ;GET # OF WORDS IN GLOBAL TBL
        MOV PC,R2 ;SET UP GLOBAL TBL END ADR
        ADD #CGBLTE--,R2
CREL1: MOV -(R2),R3 ;GET RELATIVE ADR
        ADD PC,R3 ;MAKE IT ABSOLUTE
CHBASE: MOV R3,-(R0) ;STORE ABS ADR IN EXEC
        DEC R1 ;DECR WORD COUNT
        BNE CREL1 ;DONE ALL? (Y,N-CREL1)

```

;TAILOR AND ISSUE MPG'S TITLE MSG

```

10956 043654' 010700
10957 043656' 062700 134212
10958 043662' 004567 175234
10959 043666' 001167
10960 043670' 010700
10961 043672' 062700 134232
10962 043676' 004567 175220
10963 043702' 001171
10964 043704' 010700

```

```

        MOV PC,R0 ;SET UP RESTART POINT # 1 ADR
        ADD #RST1--,R0
        JSR R5,BINASC ;CONVERT IT TO ASCII & PUT IN MSG
        .WORD CHKRIA-
        MOV PC,R0 ;SET UP RESTART # 2 ADR
        ADD #RST2--,R0
        JSR R5,BINASC ;CONVERT IT ALSO
        .WORD CHKRA2A-
        MOV PC,R0 ;GET MPG'S BASE ADR

```



```

10965 043706' 062700 134072      ADD      #CLOCZ-- ,R0
10966 043712' 004567 175204      JSR      R5,BINASC      ;CONVERT IT TO ASCII &
10967 043716' 001212                .WORD   CBASAD--      ;PUT IN THE MSG
10968 043720' 004567 142300      JSR      R5,CTYPE      ;ISSUE MPG TITLE MSG
10969 043724' 000400                .WORD   400
10970 043726' 001074                .WORD   CTITLE--
10971
10972                                ;ADJUST COMPILER TABLES FOR RELOCATION
10973
10974 043730' 010703      LRELOC: MOV      PC,R3      ;TABLE END ADDRESS
10975 043732' 062703 001066      ADD      #LMCRTE-- ,R3      ;TO R3
10976 043736' 010700                MOV      PC,R0      ;TABLE START ADDRESS
10977 043740' 062700 000704      ADD      #LMCRTB-- ,R0      ;TO R0
10978 043744' 010701      LRLLP1: MOV      PC,R1
10979 043746' 062001      LBA:   ADD      (R0)+,R1      ;CALC TABLE ADDRESS
10980 043750' 010702                MOV      PC,R2
10981 043752' 062002      LBB:   ADD      (R0)+,R2      ;CALC SUBR ADDR
10982                                .IF DF MM
10983                                MOV      PC,R3
10984                                ADD      #PG6BA-- ,R3      ;SET R3 = PAGE 6 BASE ADR
10985                                SUB      R3,R2      ;SUB FROM CALC ADDR
10986                                ADD      #P6CONS,R2      ;ADD IN PAGE 6 CONSTANT
10987                                MOV      PC,R3
10988                                ADD      #LMCRTE-- ,R3      ;PUT R3 BACK AS IT WAS
10989                                .ENDC
10990 043754' 010211      MOV      R2,(R1)      ;UPDATE COMPILER TABLE
10991 043756' 020003      CMP      R0,R3
10992 043760' 001371      BNE      LRLLP1      ;REPEAT UNTIL COMPLETE
10993                                .IF NDF MM
10994 043762' 010703      MOV      PC,R3
10995 043764' 062703 164222      ADD      #LCSTBST+4-- ,R3
10996 043770' 010700                MOV      PC,R0
10997 043772' 062700 164046      ADD      #LCSTBL+4-- ,R0
10998 043776' 010701                MOV      PC,R1
10999 044000' 062701 175440      ADD      #CBUF00-- ,R1      ;UPDATE BUFFER ADDRESSES
11000 044004' 010110      MOV      R1,(R0)      ;UPDATE BUFFER ADDRESSES
11001 044006' 062700 000006      ADD      #6,R0      ;IN DIRECT REFERENCE
11002 044012' 010110      LRLLP2: MOV      R1,(R0)      ;SYMBOL TABLE
11003 044014' 062700                ADD      #6,R0
11004 044020' 062701 000020      ADD      #16,R1
11005 044024' 020003      CMP      R0,R3
11006 044026' 001371      BNE      LRLLP2
11007 044030' 010703                MOV      PC,R3
11008 044032' 062703 164366      ADD      #LXTBL+4-- ,R3
11009 044036' 010701                MOV      PC,R1
11010 044040' 062701 176000      ADD      #LCOMO-- ,R1
11011 044044' 010110      LRLLP3: MOV      R1,(R0)      ;UPDATE COMMON ADDRESSES
11012 044046' 062700 000006      ADD      #6,R0      ;IN DIRECT REFERENCE
11013 044052' 062701 000002      ADD      #2,R1      ;SYMBOL TABLE
11014 044056' 020003      CMP      R0,R3
11015 044060' 001371      BNE      LRLLP3
11016                                .ENDC
11017 044062' 010702      MOV      PC,R2
11018 044064' 062702 163750      ADD      #LCSTBL-- ,R2      ;CALCULATE
11019 044070' 010267 163252      MOV      R2,LDRTBS      ;DIR REF TBL START
11020 044074' 010702      MOV      PC,R2
    
```

MO2

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11

MACY11 27(732) 24-SEP-76 13:54 PAGE 62-2

SEQ 0233

MPG CONTROL ROUTINE ONE TIME HOUSEKEEPING

11021	044076'	062702	164316	ADD	#LXTBL-. ,R2	;CALCULATE
11022	044102'	010267	163242	MOV	R2,LXRTBS	;INDEX REF TBL START
11023	044106'	010702		MOV	PC,R2	
11024	044110'	062702	164510	ADD	#LINDTB-. ,R2	;CALCULATE
11025	044114'	010267	163232	MOV	R2,LIRTBS	;INDIRECT REF TBL START
11026	044120'	010702		MOV	PC,R2	
11027	044122'	062702	164512	ADD	#LXINDT-. ,R2	;CALCULATE
11028	044126'	010267	163222	MOV	R2,LIITBS	;INDEXED INDIRECT TBL START
11029	044132'	010702		MOV	PC,R2	
11030	044134'	062702	164514	ADD	#LSMEND-. ,R2	;CALCULATE
11031	044140'	010267	163212	MOV	R2,LSYMTE	;SYMBOL TABLE END
11032	044144'	010702		MOV	PC,R2	;MAKE BIT PACK TABLE
11033	044146'	062702	161336	ADD	#LBPAND-. ,R2	
11034	044152'	010267	161634	MOV	R2,LBPRT1	
11035	044156'	010267	161634	MOV	R2,LBPRT2	
11036	044162'	010702		MOV	PC,R2	;POSITION INDEPENDANT
11037	044164'	062702	161422	ADD	#LBPTRU-. ,R2	
11038	044170'	010267	161626	MOV	R2,LBPRT3	
11039	044174'	010267	161626	MOV	R2,LBPRT4	
11040		001		.IF	DF MM	
11041				.IFF		
11042	044200'	010700		MOV	PC,RO	
11043	044202'	062700	175712	ADD	#LCRETN-. ,RO	;CALC POS IND ADDR
11044				.IFT		
11045				MOV	#LCRETN-PG6BA+P6CONS,RO	;CALC POS IND ADR
11046		000		.ENDC		
11047	044206'	010067	174104	MOV	RO,LPOCR1	;FOR CR LF
11048	044212'	010067	174114	MOV	RO,LPOCR2	
11049	044216'	010067	174354	MOV	RO,LPBCR1	
11050		001		.IF	DF MM	
11051				.IFF		
11052	044222'	010700		MOV	PC,RO	;CALC POS IND ADDR
11053	044224'	062700	170036	ADD	#LINBUF-. ,RO	;FOR LINBUF
11054				.IFT		
11055				MOV	#LINBUF-PG6BA+P6CONS,RO	;CALC POS IND ADR FOR LINBUF
11056		000		.ENDC		
11057	044230'	010067	173776	MOV	RO,LPOADR	
11058	044234'	010067	174266	MOV	RO,LPBADR	
11059		001		.IF	DF MM	
11060				.IFF		
11061	044240'	010700		MOV	PC,RO	;MAKE PRINT PRG # S/R
11062	044242'	062700	173552	ADD	#LPN-. RO	;POSITION INDEPENDENT
11063	044246'	010067	173534	MOV	RO,LPNADR	
11064	044252'	010700		MOV	PC,RO	;MAKE VERIFY ROUTINE
11065	044254'	062700	173116	ADD	#LVEHDR-. ,RO	;POSITION INDEPENDANT
11066	044260'	010067	172676	MOV	RO,LVECL1	
11067	044264'	010700		MOV	PC,RO	
11068	044266'	062700	173170	ADD	#LVEMSG-. ,RO	
11069	044272'	010067	172730	MOV	RO,LVECL2	
11070				.IFT		
11071				MOV	#LPN-PG6BA+P6CONS,LPNADR	;MAKE PRT PRG # S/R POS IND
11072				MOV	#LVEHDR-PG6BA+P6CONS,LVECL1	;MAKE VERIFY S/R
11073				MOV	#LVEMSG-PG6BA+P6CONS,LVECL2	;POS IND
11074				MOV	PC,RO	;SET UP ABS FREE & MIDL ADR'S
11075				ADD	#FREEAD-. ,RO	;FOR MEM MGMNT VERSION
11076				MOV	RO,LFREE	

```

11077          MOV      PC,RO
11078          ADD      #MIDLAD-.,RO
11079          MOV      RO,LMIDL
11080          000
11081          001
11082
11083          ;READ IN THE 'VALID DEVICES TABLE' FILE
11084
11085          LDVDTB: MOV    PC,RO          ;GET ADR OF TABLE'S AREA
11086          ADD      #VDFLSZ-.,RO
11087          MOV      RO,10$          ;STORE MEM ADR
11088          JSR      R5,@LOADVR      ;GO LOAD THE FILE
11089          10$: .WORD   XXXX          ;MEM ADR
11090          .WORD   512.            ;MAX BYTE COUNT
11091          .WORD   LDVDER-        ;DEV ERR ADR
11092          .WORD   LDVDER-        ;INSUF MEM ERR ADR
11093          .WORD   20$-          ;NORMAL RET ADR
11094          20$: MOV    PC,RO          ;GET ADR OF 1ST WD IN TBL
11095          ADD      #VDFLSZ-.,RO
11096          ADD      (RO),RO        ;COMPUTE END OF TBL ADR
11097          .IF DF MM
11098          BIT      #77,RO          ;AT A 32 WORD BNDRY?
11099          BEQ      25$            ;N,Y-25$
11100          ADD      #100,RO        ;BUMP UP PAST NXT BNDRY
11101          25$: MOV    #6,R1        ;SET UP LOOP CNT
11102          35$: CLC
11103          ROR      RO
11104          DEC      R1
11105          BNE     35$
11106          .ENDC
11107          001
11108          044336' 010067 135112    MOV    RO,CPAREA          ;STORE ADR AS START OF USR PROG AREA
11109          044342' 000406          BR     HKVDTB            ;GO TO TBL HOUSEKEEPING
11110          044344' 004567 141654    LDVDER: JSR    R5,CTYPE      ;ISSUE FILE LOAD ERROR MSG
11111          044350' 000602          .WORD   602
11112          044352' 000565          .WORD   LDVDEM-
11113          044354' 000000          HALT
11114          044356' 000747          BR     LDVDTB          ;HALT FOR OPERATOR ACTION
11115          000                    .ENDC          ;GO TRY LOAD AGAIN
11116
11117          ;HOUSEKEEP SAVE/FETCH DEVICE BITS IN
11118          ;THE VALID DEVICES TABLE
11119
11120          HKVDTB: MOV    @CSAVEI,RO    ;GET LOAD DEV'S MDL CODE
11121          MOV    PC,R1          ;GET ADR OF MDL NAME TBL
11122          ADD    #CMDLNM+4-.,R1
11123          10$: MOV    (R1),R2
11124          BMI    CKMFPS          ;GET MDL CODE FROM TBL
11125          BIC    #360,R2        ;END OF THE TBL? (N,Y-CKMFPS)
11126          CMP    RO,R2          ;RESET DIFFERENTIATOR BITS
11127          BEQ    20$            ;THIS THE LOAD DEV'S ENTRY?
11128          BIT    #10,(R1)        ;N,Y-20$
11129          BNE    20$            ;PERMANENT SAV/FTH DEV?
11130          BIC    #6,(R1)         ;N,Y-20$
11131          20$: ADD    #6,R1        ;CLEAR SAVE & FETCH SUPPORT BITS
11132          BR     10$            ;POINT AT NXT TBL ENTRY
                                ;GO CK IT

```

```

11133
11134
11135 ;CHECK IF CPU NEEDS MFPS/MTPS INSTRUCTIONS AND
11136 ;IT IT DOES, CHECK IF IT IS AN LSI-11.
11137 044426' 010700 CKMFPS: MOV PC,RO ;SET UP ADR OF HOUSEKEEPING'S
11138 044430' 062700 000164 ADD #H$KTRP-.,RO ;TRAP ROUT
11139 044434' 010037 000004 MOV RO,2#4 ;STORE IN CPU ERRORS TRAP VECTOR
11140 044440' 010037 000010 MOV RO,2#10 ;STORE IN RESERVED INST TRAP VECT
11141 044444' 106700 MFPS RO ;ISSUE INST TO CK CPU TYPE
11142 044446' 052767 000002 167540 BIS #USMTPS,CSYSFW ;DID IT - SET FLG TO INDICATE CPU TYPE
11143 044454' 000401 BR 30$ ;GO TO LSI-11 CHECK
11144 044456' 000407 BR CKRSET ;IT TRAPPED - BYPASS LSI-11 TEST
11145 044460' 010700 30$: MOV PC,RO ;GET AN EVEN ADR
11146 044462' 005200 INC RO ;MAKE IT ODD
11147 044464' 005710 TST (RO) ;DO WORD INST TO AN ODD ADR
11148 ;LSI-11 WILL NOT TRAP
11149 044466' 052767 000004 167520 BIS #LSI11,CSYSFW ;NO TRAP - SET LSI-11 FLAG
11150 044474' 000402 BR CKRST2 ;BYPASS VECTOR RESET
11151 044476' 005037 000102 CKRSET: CLR 2#102 ;RESET LSI-11 CLOCK VECTOR
11152 044502' 012737 000012 000010 CKRST2: MOV #12,2#10 ;RESTORE RESERVED INST TRAP VECT
11153
11154
11155 ;TEST IF THE CPU HAS A CACHE MEMORY BY
11156 ;ATTEMPTING TO ACCESS THE HIT/MISS REGISTER.
11157
11158 044510' 005737 177752 CK4CHE: TST 2#177752 ;ACCESS THE HIT/MISS REG
11159 044514' 012767 000002 167742 MOV #2,CTOAJ ;ITS THERE: ADJ T/O VALUE
11160 044522' 000401 BR 31$ ;BYPASS TRAP BR
11161 044524' 000406 BR CK4A70 ;NOT THERE - GO TO NEXT TEST
11162 044526' 012767 002126 172234 31$: MOV #2126,LDLLP1+2 ;ADJ DELAY INST TIMING VALUE
11163 044534' 052767 000200 167452 BIS #CACHE,CSYSFW ;SET CACHE MEM PRESENT FLAG
11164
11165
11166 ;TEST IF THE CPU IS AN 11/70 BY ATTEMPTING TO SET
11167 ;THE 22 BIT ADDRESSING BIT IN MMR3.
11168
11169 044542' 012737 000020 172516 CK4A70: MOV #20,2#MMR3 ;TRY TO SET 22 BIT ADR BIT
11170 044550' 032737 000020 172516 BIT #20,2#MMR3 ;DID BIT SET?
11171 044556' 001001 BNE 32$ ;N.Y-32$
11172 044560' 000405 BR 33$ ;GO TO NEXT TEST
11173 044562' 052767 000010 167424 32$: BIS #CPU70,CSYSFW ;SET THE CPU = 11/70 FLAG
11174 044570' 005037 172516 CLR 2#MMR3 ;RESET 22 BIT ADR BIT
11175 001 .IF DF MM
11176 .IFF
11177 044574' 012737 000006 000004 33$: MOV #6,2#4 ;RESTORE TRAP VECTOR
11178
11179
11180 .IFT
11181 33$:
11182
11183
11184 ;TEST IF CPU IS AN 11/40 BY ATTEMPTING TO ACCESS
11185 ;MMR3. IF WE CAN, ITS AN 11/45 OR HIGHER.
11186
11187 CK4A40: BIS #CPU40,CSYSFW ;INITIALIZE TO AN 11/40
11188 TST 2#MMR3 ;TRY TO ACCESS MMR3

```

```

11189      BIC      #CPU40,CSYSFW      ;DID IT - RESET 11/40 FLG
11190      NOP                          ;FILL INST
11191
11192      ;INITIALIZE MEMORY MANAGEMENT REGISTERS AND
11193      ;ENABLE ITS OPERATION.
11194
11195      INITMM: MOV      #KPAR0,R0      ;SET UP ADR'S OF KERNEL MODE REGS
11196      MOV      #KPDRO,R1
11197      MOV      #8.,R2                ;SET UP REGISTER COUNT
11198      CLR      R3                    ;SET MEM BASE ADR TO 0
11199      35$:  MOV      R3,(R0)+         ;STORE PAGE'S MEM BASE ADR
1200      MOV      #PORCON,(R1)+       ;STORE DESCRIPTOR REG VALUE
1201      ADD      #200,R3              ;POINT TO NEXT MEM PAGE
1202      DEC      R2                    ;DONE ALL REGISTERS?
1203      BNE      35$                  ;Y,N-35$
1204      MOV      #IOPAGE,-(R0)       ;SET PAR 7 TO THE I/O PAGE
1205      BIT      #CPU40,CSYSFW       ;THIS AN 11/40?
1206      BNE      40$                  ;N,Y-40$
1207      CLR      #MMMR3                ;HOUSEKEEP MMR3 REG
1208      40$:  MOV      #1001,#MMMR0   ;ENABLE MEM MGMNT
1209
1210      ;NOW RUNNING UNDER MEMORY MANAGEMENT
1211
1212      ;CHECK IF THE UNIBUS MAP AND 22 BIT ADDRESSING
1213      ;ARE AVAILABLE ON THIS CPU. IF THEY ARE, ASK IF
1214      ;THEY ARE TO BE USED.
1215
1216      CKUMAP: BIT      #CPU40,CSYSFW ;THIS AN 11/40?
1217      BNE      MMSZM                 ;N,Y-MMSZM
1218      MOV      #MMMR3,R3              ;SET UP MM REG ADR
1219      MOV      #60,(R3)               ;ENABLE THE MAP & 22 BIT ADDRESSING
1220      BIT      #40,(R3)               ;DID THE UNIBUS MAP BIT SET?
1221      BEQ      CK22AD                 ;Y,N-CK22AD
1222      42$:  JSR      RS,CTYPE          ;ASK IF THEY WANT US TO USE IT
1223      .WORD    420
1224      .WORD    UBMPNG-.
1225      .WORD    25.
1226      JSR      PC,CKRPLY              ;CHECK THEIR REPLY
1227      BR      42$                     ;BR IF INV REPLY
1228      BR      CK22AD                   ;BR IF REPLY IS 'N'
1229      BIS      #UNIMAP,CSYSFW         ;SET THE "USE UNIBUS MAP" FLAG
1230      BIS      #40,MR3VAL             ;SET ITS BIT IN WD FOR MMR3 SET UP
1231      CK22AD: BIT      #20,(R3)       ;DID THE 22 BIT ADR BIT SET?
1232      BEQ      CKUMCL                 ;Y,N-CKUMCL
1233      44$:  JSR      RS,CTYPE          ;ASK IF THEY WANT TO USE
1234      .WORD    420                     ;22 BIT ADDRESSING
1235      .WORD    U22BAD-.
1236      .WORD    27.
1237      JSR      PC,CKRPLY              ;CHECK THEIR REPLY
1238      BR      44$                     ;BR IF INV REPLY
1239      BR      CKUMCL                   ;BR IF REPLY IS 'N'
1240      BIS      #BITS22,CSYSFW         ;SET THE "USE 22 BITS" FLAG
1241      BIS      #20,MR3VAL             ;SET ITS BIT IN WD FOR MMR3 SET UP
1242      MOV      #177600,IOPAR7        ;SET 22 BIT I/O PAGE PAR
1243      CKUMCL: BIC      #40,(R3)       ;RESET THE UNIBUS MAP FOR NOW
1244

```

```

11245 ;SIZE MEMORY ABOVE THE 16K MEMORY BOUNDARY
11246
11247 MMSZM: MOV #P6CONS,RO ;SET UP ADR THAT SELECTS PAR 6
11248 MOV #1000,2#KPAR6 ;INITIALIZE PAR 6 TO 16K WD BNDRY
11249 50$: BIT #BITS22,CSYSFW ;USING 22 BIT ADDRESSING?
11250 BEQ 60$ ;Y,N-60$
11251 CMP #170000,2#KPAR6 ;END OF 22 BIT MEMORY?
11252 BEQ 80$ ;N,Y-80$
11253 BR 70$ ;CONTINUE MEM TEST
11254 60$: CMP #7600,2#KPAR6 ;END OF 18 BIT MEMORY?
11255 BEQ 80$ ;N,Y-80$
11256 70$: TST (RO) ;TEST CURRENT MEM LOCATION
11257 ADD #40,2#KPAR6 ;INCR TO NEXT 1K OF MEM
11258 BR 50$ ;GO TEST NEXT LOC ADR
11259 80$: MOV 2#KPAR6,CRMEND ;STORE LAST K OF MEM ADR
11260 DEC CRMEND
11261 MOV #6,2#4 ;RESTORE CPU ERROR TRAP VECT
11262 CLR 2#MMMR0 ;DISABLE MEM MGMNT
11263
11264 ;MEMORY MANAGEMENT IS NOW DISABLED
11265
11266 ;INITIALIZE SHARED PAGE PAR & PDR VALUES
11267
11268 IPG6VL: MOV PC,R1 ;GET START ADR OF SHARED PAGE
11269 ADD #PG68A--,R1
11270 MOV PC,R0 ;GET ITS END ADR
11271 ADD #PG6END--,R0
11272 SUB R1,R0 ;GET ITS SIZE IN BYTES
11273 BIT #77,R0 ;END ON A 32 WD BNDRY?
11274 BEQ 82$ ;N,Y-82$
11275 ADD #100,R0 ;POINT IT PAST NXT BNDRY
11276 82$: MOV #6,R2 ;CONVERT ADR TO A 32 WD ADR
11277 84$: ASR R0 ;AND BYTE CNT TO A 32 WD
11278 ASR R1 ;BLK CNT
11279 DEC R2
11280 BNE 84$
11281 MOV R1,SPPAR6 ;STORE SHARED PAGE PAR VALUE
11282 DEC R0 ;ADJ BLK CNT
11283 SWAB R0 ;GET PLF BITS IN HIGH BYTE
11284 BIS #6,R0 ;SET IN ACF BITS
11285 MOV R0,SPPDR6 ;STORE SHARED PAGE PDR VALUE
11286
11287 ;INITIALIZE TRAP VECTORS
11288
11289 ITRPVC: MOV PC,R0 ;GET TRAP INST ROUT ABS ADR
11290 ADD #TRP34--,R0
11291 MOV R0,2#34 ;STORE IT AT ITS VECTOR
11292 MOV PC,R0 ;SET UP ADR OF ROUT FOR
11293 ADD #TRP04--,R0 ;CPU ERROR TRAP
11294 MOV R0,2#4 ;STORE IT
11295 MOV PC,R0 ;SET UP ADR OF ROUT FOR
11296 ADD #TRP10--,R0 ;RESERVED INST TRAP
11297 MOV R0,2#10 ;STORE IT
11298 MOV PC,R0 ;SET UP ADR OF ROUT FOR
11299 ADD #TRP114--,R0 ;MEMORY ERRORS
11300 MOV R0,2#114 ;STORE IT
    
```

E03

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11

MACY11 27(732) 24-SEP-76 13:54 PAGE 62-7
MPG CONTROL ROUTINE ONE TIME HOUSEKEEPING

SEQ 0238

```

11301          MOV      PC,R0          ;SET UP ADR OF ROUT FOR
11302          ADD      #TRP250-.,R0   ;MEM MGMNT TRAP
11303          MOV      R0,#250        ;STORE IT
11304          OOC
11305          OOI          .ENDC
11306          .IF DF MIMIC
11307          ;CHECK IF DEV ROUT IS INCLUDED & ADJUST MPG
11308
11309          CK4DVR: MOV      PC,R0          ;SET UP MPG END ADR
11310          ADD      #MPGEND-.,R0
11311          MOV      R0,R3          ;SAVE IT
11312          OOI          .IF DF MM
11313          MOV      #6,R1          ;SET UP LOOP CNT
11314          90$: CLC          ;EXTRACT HIGH 16 BITS OF THE ADR
11315          ROR      R0
11316          DEC      R1
11317          BNE      90$
11318          OOI          .ENDC
11319          MOV      R0,CPAREA        ;SAVE NEW USR PROG START ADR
11320          CMP      LIMIT+2,R3      ;IS A DEV ROUT LINKED IN?
11321          BLOS     HSKEND          ;Y,N-HSKEND
11322          JSR      R5,BINASC       ;PLACE DVR ST ADR IN MSG
11323          .WORD   CDVRBG-
11324          MOV      PC,R0          ;SET UP SLOT # 1 ADR
11325          ADD      #CRTBL-.,R0
11326          JSR      R5,BINASC       ;PLACE SLOT # 1 ADR IN MSG
11327          .WORD   CSLT1A-
11328          OOI          .IF DF MM
11329          MOV      PLNGTH(R3),R0    ;GET DEV ROUT SIZE
11330          BIT      #77,R0          ;END ON A 32 WD BNDRY?
11331          BEQ      100$           ;N,Y-100$
11332          ADD      #100,R0        ;POINT PAST NXT BNDRY
11333          100$: MOV      #6,R1     ;SET UP LOOP CNT
11334          105$: ASR      R0          ;GET # OF 32 WD BLKS
11335          DEC      R1
11336          BNE      105$
11337          MOV      R0,CRTBL+34.    ;STORE IN CNTRL ROUT TBL EXT
11338          OOI          .ENDC
11339          JSR      R5,CTYPE         ;TYPE THE "WRM" MSG
11340          .WORD   400
11341          .WORD   CWMSG-
11342          OOI          .IF DF MM
11343          BIT      #UNIMAP,CSYSFW  ;USING THE UNIBUS MAP?
11344          BNE      115$           ;N,Y-115$
11345          CLR      PUBMAP(R3)     ;RESET ITS MAP INFO
11346          OOI          .ENDC
11347          115$: ADD      #PTLGTH+DEVI0B,R3 ;POINT AT DVR ADDRESSES
11348          MOV      PC,R0          ;SET UP TABLE ADR
11349          ADD      #CLDTBL-.,R0
11350          CHKLOP: MOV      (R0)+,R2 ;GET REL OR VIRT MPG POINT ADR
11351          BEQ      HSKEND          ;END OF TBL? (N,Y-HSKEND)
11352          OOI          .IF DF MM
11353          BMI      CHKVAD          ;PAGE 6 ADR? (N,Y-CHKVAD)
11354          .ENDC
11355          001          ADD      PC,R2
11356          001          CHKBAS: SUB     #CHKBAS-CLDBAS,R2 ;MAKE IT ABS
;ADJ FOR DIFF PC ADRS

```

```

11357          CHKVAD: MOV      R2,(R3)+      ;STORE IT IN DVR
11358          BR          CHKLOP           ;GO GET NXT ADR
11359
11360          .ENDC
11361 044602' 012745 000240  HSKEND: MOV      #240,-(R5)      ;SET CALLING JSR TO NOP'S
11362 044606' 012745 000240  MOV      #240,-(R5)
11363 044612' 000205          RTS      R5          ;EXIT IN-LINE
11364
11365          ;HOUSEKEEPING'S TRAP ROUTINE
11366
11367
11368 044614' 062716 000010  HSKTRP: ADD      #8.,(SP)      ;BYPASS NEXT 4 WDS OF PROG
11369 044620' 000002          RTI          ;EXIT FROM THIS TRAP
11370
11371
11372          .IF DF MM
11373          ;GET REPLY FOR HSKP MESSAGE AND CHECK IT.
11374          ;Y, N, <CR>, & <LF> ARE ONLY VALID REPLIES.
11375          ;
11376          ;          JSR      PC,CKRPLY      S/R CALL
11377          ;          BR          LABEL      EXECUTED IF INVALID REPLY
11378          ;          BR          LABEL      EXECUTED IF A 'N' REPLY
11379          ;
11380          ;DESTROYS R0,R1,R2
11381
11382          CKRPLY: JSR      R5,ACTRD          ;ISSUE KEYBOARD READ
11383          .WORD      CCMRD-
11384          .WORD      30.
11385          JSR      R5,CSCNSP          ;SCAN READ DATA FOR THE REPLY
11386          BIT      #CSCFND,R2        ;ONLY <CR> OR <LF> ENTRY?
11387          BEQ      10$
11388          CMPB     #'N',ACDSTAD      ;IS REPLY AN 'N'?
11389          BEQ      20$
11390          CMPB     #'Y',ACDSTAD      ;IS REPLY A 'Y'?
11391          BNE      30$
11392          10$:  ADD      #2,(SP)        ;BYPASS 2ND BR INST
11393          20$:  ADD      #2,(SP)        ;BYPASS 1ST BR INST
11394          30$:  RTS      PC          ;EXIT
11395          .ENDC
11396
11397
11398 044622' 147404  CGBLTS: .WORD      UFWSET-CHBASE      ;GLOBAL ADDRESSES TABLE
11399 044624' 147422  .WORD      UFWCLR-CHBASE
11400 044626' 147440  .WORD      UFWTST-CHBASE
11401 044630' 175254  .WORD      BINASC-CHBASE
11402 044632' 175316  .WORD      BTASLB-CHBASE
11403 044634' 170346  .WORD      CSYSFW-CHBASE
11404 044636' 134360  .WORD      USRINT-CHBASE
11405          044640'  CGBLTE =.
11406
11407
11408 044640' 000000 000000  LIMIT: .LIMIT      ;MPG'S START & END ADR'S
11409
11410          ;
11411          ; COMPILER MACHINE CODE RELOCATION TABLE
11412          ;
    
```



```

11413 044644' 165054 171750      .MCRTB: .WORD  LRT1-LBA,LADD0-LBB
11414 044650' 165070 172046      .WORD  LRT2-LBA,LSUBM-LBB
11415 044654' 165442 174050      .WORD  LRT3-LBA,LPRNG-LBB
11416 044660' 165454 174050      .WORD  LRT4-LBA,LPRNG-LBB
11417 044664' 165466 174154      .WORD  LRT5-LBA,LPRNO-LBB
11418 044670' 165500 174142      .WORD  LRT6-LBA,LPRND-LBB
11419 044674' 165512 174430      .WORD  LRT7-LBA,LPRNB-LBB
11420 044700' 165524 174050      .WORD  LRT8-LBA,LPRNG-LBB
11421 044704' 165536 174154      .WORD  LRT9-LBA,LPRNO-LBB
11422 044710' 165550 174142      .WORD  LRT10-LBA,LPRND-LBB
11423 044714' 165562 174430      .WORD  LRT11-LBA,LPRNB-LBB
11424 044720' 165574 172244      .WORD  LRT12-LBA,LFILR-LBB
11425 044724' 165606 172322      .WORD  LRT13-LBA,LFILG-LBB
11426 044730' 165620 172200      .WORD  LRT14-LBA,LFILV-LBB
11427 044734' 165634 172244      .WORD  LRT15-LBA,LFILR-LBB
11428 044740' 165646 172322      .WORD  LRT16-LBA,LFILG-LBB
11429 044744' 165660 172200      .WORD  LRT17-LBA,LFILV-LBB
11430 044750' 165724 172760      .WORD  LRT18-LBA,LROTR-LBB
11431 044754' 165736 172760      .WORD  LRT19-LBA,LROTR-LBB
11432 044760' 165750 173012      .WORD  LRT20-LBA,LDLAY-LBB
11433 044764' 165430 174050      .WORD  LRT21-LBA,LPRNG-LBB
11434 044770' 165674 172354      .WORD  LRT22-LBA,LFILP-LBB
11435 044774' 165710 172354      .WORD  LRT23-LBA,LFILP-LBB
11436 045000' 166140 173032      .WORD  LRT27-LBA,LVERFY-LBB
11437 045004' 166154 173032      .WORD  LRT28-LBA,LVERFY-LBB
11438 045010' 166234 171244      .WORD  LRT29-LBA,CUPCR-LBB
11439 045014' 166314 171044      .WORD  LRT30-LBA,CUPGEX-LBB
11440 045020' 000000      LMCRTB: .WORD  0 ;END OF RELOCATION TABLE
11441
11442
11443      001      .NLIST  BEX
11444      .IF DF MM
11445 045022' 005015 052104 043515 CTITLE: .ASCII <015><012>/DTMGA-C M.P.G. RST1: /
11446      .IFT
11447 CTITLE: .ASCII <015><012>/DTMMA-B M.P.G. RST1: /
11448      .ENDC
11449 045055' 130 054130 054130 CHKR1A: .ASCII /XXXXXX; RST2: /
11450 045073' 130 054130 054130 CHKR2A: .ASCII /XXXXXX/<015><012><015><012>
11451 045105' 115 043520 041040      .ASCII /MPG BASE ADDRESS = /
11452 045130' 054130 054130 054130 CBASAD: .ASCIZ /XXXXXX/
11453      001      .IF DF MIMIC
11454 CWMSG: .ASCII <015><012>/DO: "WRM /
11455 CSLT1A: .ASCII /XXXXXX/
11456 CDVRBG: .ASCIZ /XXXXXX^, THEN "FM"/
11457      000      .ENDC
11458      001      .IF NDF MIMIC
11459 045137' 126 042114 042040 LDVDEM: .ASCIZ /VLD DEV TBL LD ERR, PRESS "CONT" TO RETRY/
11460      000      .ENDC
11461      001      .IF DF MM
11462 UBMPMG: .ASCII <015><012>'?USE UNIBUS MAP (Y/N) / '
11463 U22BAD: .ASCII <015><012>'?USE 22 BIT ADRS (Y/N) / '
11464      000      .ENDC
11465      .LIST  BEX
11466
11467      045212'      .EVEN
11468      001      .IF DF MIMIC & MM

```

11469		ALPT2=	.-CLOCZ
11470		ALFCT2=	ALPT2&000077/2
11471			.IF NE ALFCT2
11472	002	ALQUN2=	31.-ALFCT2
11473			.BLKW ALQUN2
11474			.WORD 0
11475			.ENDC
11476	001		
11477			.ENDC
11478	000	MPGEND=	.
11479	045212'		.END
11480	000001		

ACTIVE=	100000		CBUF05	041560R	002	CERM03	006625R	002	CHBASE	043646R	002	CMIDLM	007513R	002
ADR188	033462R	002	CBUF06	041600R	002	CERM04	006640R	002	CHKR1A	045055R	002	CMMEND	007466R	002
ASDECO	042350R	002	CBUF07	041620R	002	CERM05	006645R	002	CHKR2A	045073R	002	CMMHDR	007416R	002
ASDEC1	042342R	002	CBUF08	041640R	002	CERM06	006663R	002	CHPINT=	004000		CMMMDL	007506R	002
AURPEP=	000010		CBUF09	041660R	002	CERM07	006671R	002	CINACT	002510R	002	CMMMSG	007451R	002
AUTORP=	000020		CBUF10	041700R	002	CERM08	006701R	002	CINTMG	010016R	002	CMPNM	007476R	002
BCPTSF	033364R	002	CBUF11	041720R	002	CERM09	006715R	002	CINTPC	010034R	002	CMMSTA	007454R	002
BINASC	041122R	002	CBUF12	041740R	002	CERM10	006727R	002	CINTPN	010066R	002	CMODIF	002434R	002
BINAS1	041126R	002	CBUF13	041760R	002	CERM11	006737R	002	CINTSP	010050R	002	CMPCKE	004272R	002
BINAS8	041122R	002	CBUF14	042000R	002	CERM12	006751R	002	CIOALT=	000400		CMPNXP	004276R	002
BIT11 =	004000		CBUF15	042020R	002	CERM13	006760R	002	CIOBSY	035216R	002	CMPPFM	004404R	002
BIT12 =	010000		CCBO	006210R	002	CERM14	006765R	002	CIOSZM	010073R	002	CMPPPG	004310R	002
BIT6 =	000100		CCDB	006164R	002	CERM15	007003R	002	CISDMM	013472R	002	CMVNAM	004722R	002
BOOT	000006R	002	CCDBER	006200R	002	CERM16	007017R	002	CIVMSG	007243R	002	CMVNEX	004644R	002
BTASLB	041164R	002	CCFMTA	001750R	002	CERM17	007026R	002	CKBDID	042600R	002	CNCMND	000366R	002
BTASLZ	041172R	002	CCFMTB	001762R	002	CERM18	007035R	002	CKCLRG	035140R	002	CNCTU	000764R	002
BTPACK	025314R	002	CCFMTJ	001734R	002	CERM19	007041R	002	CKCMND	000530R	002	CNONID	042572R	002
BTUPAK	026030R	002	CCFMT1	001710R	002	CERM20	007046R	002	CKEX	035172R	002	CNSMSG	007247R	002
BUF	000010R	002	CCFMT2	002004R	002	CERM21	007051R	002	CKILEX	035214R	002	CNTRLO=	010000	
BUMP	036114R	002	CCFMT3	002020R	002	CERM22	007056R	002	CKILL	002520R	002	CN20ER	002030R	002
CRAMSG	007335R	002	CCFMT4	001676R	002	CERM23	007067R	002	CKMFP5	044426R	002	CN20PJ	005666R	002
CACHE =	000200		CCFPBJ	002712R	002	CERM24	007132R	002	CKODAD	013344R	002	COMPIL	015030R	002
CADD	006042R	002	CCFPDE	001740R	002	CERM25	007141R	002	CKPDVP	035032R	002	COMRET	003342R	002
CADMSG	007533R	002	CCFRET	001756R	002	CERM26	007151R	002	CKRSET	044476R	002	COPER	002716R	002
CAIDLE=	001000		CCMDRD	001464R	002	CERM27	007163R	002	CKRST2	044502R	002	COPSW	002636R	002
CANS	007724R	002	CCONT	002476R	002	CERM30	007174R	002	CK4A70	044542R	002	COPSWN	002726R	002
CANSMG	007716R	002	CDCNT	001600R	002	CERM31	007207R	002	CK4CHE	044510R	002	CPAREA	001454R	002
CARDSP	006072R	002	CDDFP =	100000		CERM32	007235R	002	CLDBAS	012456R	002	CPATBL	001064R	002
CARITH	006062R	002	CDELET	002414R	002	CERRET	001602R	002	CLDER1	012476R	002	CPCMRF	007753R	002
CARIT1	006064R	002	CDFNAM	001644R	002	CESMSG	007733R	002	CLDF R2	012502R	002	CPGCDN	035370R	002
CARRET	006102R	002	CDISPL	003240R	002	CEXPGR	034632R	002	CLDER3	012506R	002	CPGCPE	034472R	002
CASAVE	005452R	002	CDISPN	007560R	002	CFCMND	000350R	002	CLDEVR	011642R	002	CPGCPT	034666R	002
CASCOM	005516R	002	CDLCEX	005124R	002	CFECNR	004652R	002	CLDEX	012464R	002	CPGDSP	034434R	002
CASERO	005636R	002	CDNAME	007564R	002	CFETCH	004570R	002	CLDID	042614R	002	CPGHSK	035404R	002
CASER1	005640R	002	CDREND	001616R	002	CFIER1	004052R	002	CLDISU	012376R	002	CPGNXT	034604R	002
CASER2	005650R	002	CDSHOR	007547R	002	CFIER2	004062R	002	CLDLOP	012452R	002	CPGRSV	035224R	002
CASEX	005632R	002	CDSTAD	001576R	002	CFIER3	004072R	002	CLDNRT	012254R	002	CPGTOE	034736R	002
CASE1J	005442R	002	CDVRST	012242R	002	CFIER4	004102R	002	CLDPDR	012026R	002	CPGTE	034470R	002
CASFCH	005464R	002	CDVRSZ	012244R	002	CFIERS	004112R	002	CLDTBL	012522R	002	CPNASC	013610R	002
CASFTH	005436R	002	CECM5G	007314R	002	CFIEX	004250R	002	CLIST	035600R	002	CPNAS1	013624R	002
CASLST	005474R	002	CENDER	002376R	002	CFILL	003460R	002	CLISTI	000014R	002	CPNMSG	007365R	002
CASMSG	007331R	002	CENERA	002404R	002	CFIPAT	004122R	002	CLOCZ	000000R	002	CPNTR	001462R	002
CASSIG	005300R	002	CENERR	002402R	002	CFMEND	001450R	002	CLOSE	000016R	002	CPRT	000020R	002
CBASAD	045130R	002	CENTER	002040R	002	CFMMSG	007435R	002	CLRVEC	040714R	002	CPSVRS	035226R	002
CBASE1	001056R	002	CENTPA	002062R	002	CFMST	001446R	002	CLSJMP	002472R	002	CPU70 =	000010	
CBASE2	001024R	002	CENTPB	002356R	002	CFTCHF=	040000		CLST	005042R	002	CRATBL	001074R	002
CBOC	006134R	002	CENTPC	002352R	002	CFTCHI	000012R	002	CMAP	004254R	002	CRDM	005716R	002
CBOOT	005270R	002	CEPNUM	007374R	002	CGBLTE=	044640R	002	CMCDVN	001620R	002	CRDMT	007407R	002
CBUF00	041440R	002	CERDTA	007300R	002	CGBLTS	044622R	002	CMDLNM	042572R	002	CRDMG1	007377R	002
CBUF01	041460R	002	CERID	007360R	002	CGOENT	004614R	002	CMDTBL	001162R	002	CRDMG2	007406R	002
CBUF02	041500R	002	CERMSG	007265R	002	CGOREX	035524R	002	CMEMCK	013444R	002	CREATE	000022R	002
CBUF03	041520R	002	CERM01	006600R	002	CGORPT	035504R	002	CMGCNT	000436R	002	CREL1	043642R	002
CBUF04	041540R	002	CERM02	006607R	002	CGORUN	000470R	002	CMIDL	007523R	002	CRENPA	007607R	002

CREPOR	003346R	002	CSUERR	013400R	002	DERPAD=	000036	FILPEX	036506R	002	LBPLP1	025374R	002
CRFLGW	001444R	002	CSUPTR	013330R	002	DEVBTA=	000056	FLLDMG	010154R	002	LBPLP2	025460R	002
CRMEND	001452R	002	CSVCOM	004522R	002	DEVDER=	000050	FLLDMN	010170R	002	LBPLP3	025554R	002
CRPFLG	035522R	002	CSYSFW	034214R	002	DEVORA=	000024	FSTTME	016102R	002	LBPLP4	025676R	002
CRPTEP	035476R	002	CSZREQ	011674R	002	DEVETP=	000104	GET	000046R	002	LBPLP5	025722R	002
CRQADR	011172R	002	CTFLGW	000026R	002	DEVFWD=	000002	GETBYT	040766R	002	LBP MOR	025422R	002
CRQCTA	011610R	002	CTITLE	045022R	002	DEVI08=	000046	GTNXTD=	001000	002	LBP NF	025472R	002
CRQORD	007640R	002	CTOAJ	034464R	002	DEVIVA=	000026	HKVDTB	044360R	002	LBP PGL	025654R	002
CRQORD	007625R	002	CTOCK	034466R	002	DEVIW1=	000004	HSKEND	044602R	002	LBP RT1	026012R	002
CRQER	011630R	002	CTPRIO=	000020	002	DEVIW2=	000006	HSKPEP=	000004	002	LBP RT2	026016R	002
CRQEX	011606R	002	CTRD	000030R	002	DEVIW3=	000010	HSKTRP	044614R	002	LBP RT3	026022R	002
CRGLNK	011340R	002	CTRDEX	000032R	002	DEVIW4=	000012	INVSTK	035526R	002	LBP RT4	026026R	002
CRQLPT	011332R	002	CTRDNE	000034R	002	DEVIW5=	000014	JCPGV	031026R	002	LBP RW	026004R	002
CRQNXM	011230R	002	CTRQBP=	100000	002	DEVIW6=	000016	JCPTSF	033052R	002	LBP SBO	025570R	002
CRSCHD	000754R	002	CTUDED=	020000	002	DEVIW7=	000020	JLPKER	017752R	002	LBP ST1	025752R	002
CRTBL	034216R	002	CTUSWR	000036R	002	DEVIW8=	000022	LADD	035772R	002	LBP TG1	025406R	002
CRUDED=	040000	002	CTWR	000040R	002	DEV RPS=	000030	LADDC	035766R	002	LBP TG2	025714R	002
CRUER1	003200R	002	CTYPE	006224R	002	DEVRSZ=	000000	LADDEX	036014R	002	LBP TG3	025740R	002
CRUER2	003210R	002	CUPCR	035216R	002	DEVSTP=	000102	LADDM	035722R	002	LBP TRU	025606R	002
CRUER3	003220R	002	CUPGER	034760R	002	DEVWPS=	000032	LADDM	035752R	002	LBP VTB	026010R	002
CRUER4	003230R	002	CUPGEX	035016R	002	DFLT MN	012732R	LADD1	022420R	002	LBRK1	022411R	002
CRUN	002760R	002	CURCTE	001456R	002	DHKPAD=	000034	LADD2	022425R	002	LBRK2	023240R	002
CR1ADR	001604R	002	CURPTA	001460R	002	DIDTAD	042570R	LASCNT	042410R	002	LBT EX	042316R	002
CR2ADR	001606R	002	CUSPGR=	002000	002	DISP	024470R	LASDEC	042352R	002	LBT NBR	026002R	002
CSACNR	004646R	002	CUSPRT	000042R	002	DISPCD	024476R	LASDON	023252R	002	LBT NSV	026006R	002
CSAINM	005000R	002	CVL COM	012614R	002	DKILAD=	000040	LASERR	042522R	002	LBT PNT	025776R	002
CSANAS	004760R	002	CVLER1	013210R	002	DOERCK=	000400	LASER1	042524R	002	LBU AND	026336R	002
CSANNM	004770R	002	CVLER2	013220R	002	DOIOT =	000040	LASEX	042514R	002	LBU AST	026346R	002
CSANPN	005010R	002	CVLER3	013230R	002	DRTEND=	000116	LASLPO	042372R	002	LBU BKU	026222R	002
CSAVE	004512R	002	CVLER4	013244R	002	DRTLTH=	000116	LASLP1	042420R	002	LBU BPT	026324R	002
CSAVEI	000024R	002	CVLFIL	013154R	002	DTOEAD=	000044	LASLP2	042450R	002	LBU EXT	026310R	002
CSCCOM	010246R	002	CVLFLG=	020000	002	DVB TDA=	000060	LASLP3	042456R	002	LBU FB	026236R	002
CSCERR	010474R	002	CVLMD1	012560R	002	DVC PRT=	000054	LASNIN	016077R	002	LBULP1	026046R	002
CSCFND=	000010	002	CVLMD2	012570R	002	DVCTEP=	000112	LASPAK	042530R	002	LBULP2	026162R	002
CSCFND	010214R	002	CVLR3A	013234R	002	DVCVEC=	000070	LASZER	016076R	002	LBULP3	026170R	002
CSCN	010222R	002	CVMEND	001450R	002	DVGETB=	000076	LBA	043746R	002	LBULP4	026300R	002
CSCNBB	010234R	002	CV MST	001446R	002	DVIWSP=	000114	LBASE1	017334R	002	LBUMP1	036130R	002
CSCNCB	010234R	002	CVPER	010564R	002	DVNUM1	012742R	LBASE2	023624R	002	LBUMP2	036132R	002
CSCNCD	010242R	002	CVPNM1	010524R	002	DVPDTA=	000062	LBB	043752R	002	LBUMP3	036146R	002
CSCNEX	010472R	002	CVPNUM	010504R	002	DVPTEP=	000106	LBDDDN	041252R	002	LBUMP4	036150R	002
CSCNMS	010226R	002	CWA	001610R	002	DVPUTB=	000100	LBDEXT	041302R	002	LBUNBR	026326R	002
CSCNSP	010216R	002	CWRM	005660R	002	DVRID	012730R	LBDLIN	021274R	002	LBU SAS	026100R	002
CSCRLF=	100000	002	CWRMOK	005672R	002	DVRINT=	000074	LBOLP1	041226R	002	LBU SPC	026342R	002
CSC TCS=	000020	002	CYCDVL=	002000	002	DVSFWD=	000064	LBOLP2	041242R	002	LBU TRU	026330R	002
CSC TMS=	000040	002	CYCPRG=	040000	002	DVSVEC=	000066	LBD SFL	041266R	002	LBU WRD	026344R	002
CSONAC	010770R	002	CZERO	005130R	002	DVTVEC=	000072	LBDSTP	041216R	002	LCARRY	036016R	002
CSFCKA	004620R	002	CZMSG	010112R	002	DVUPRT=	000052	LBDTZF	041272R	002	LCBUF	016100R	002
CSLDEL	005020R	002	CITHKP	043566R	002	DVVTTP=	000110	LBIASD	041176R	002	LCKCR	015650R	002
CSPGUP	010574R	002	DECASC	041024R	002	ENTER	013640R	LBPAND	025504R	002	LCKOBL	027432R	002
CSPNFP	010600R	002	DECBIN	042116R	002	ERSTOP=	000004	LBPBS	025414R	002	LCKWD	033562R	002
CSTDVI	013414R	002	DECBUF	042563R	002	FDV NAM	013546R	LBP ERX	025772R	002	LCLBUF	015236R	002
CSTOP	002462R	002	DECTAD=	000042	002	FENTRY	027634R	LBPEXT	025762R	002	L CLEAR	022456R	002
C SUB	006106R	002	DELETE	000044R	002	FETCH	013734R	LBPLGP	025662R	002	LCMCTB	030652R	002

LCOBPT	027554R	002	LCPSWP	034130R	002	LDOINC	015410R	002	LFVODD	036166R	002	LLPCLS	014416R	002
LCOBUG	027552R	002	LCPTTEC	034136R	002	LDONE	023246R	002	LFVOET	036160R	002	LLPCNH	014402R	002
LCOCNR	027576R	002	LCPTEN	033756R	002	LDRSTE	027360R	002	LGOEXT	016452R	002	LLPCOL	014050R	002
LCOEIT	027500R	002	LCPTGO	032702R	002	LDRTBS	027346R	002	LGOTO	022660R	002	LLPODN	014172R	002
LCOGLT	027616R	002	LCPTG1	032706R	002	LDVDEM	045137R	002	LGTRTN	016060R	002	LLPDEL	014134R	002
LCOGPL	027606R	002	LCPTG2	032724R	002	LDVDER	044344R	002	LHINUM	041434R	002	LLPFEX	014756R	002
LCOLFD	027564R	002	LCPTG3	032744R	002	LDVDTB	044276R	002	LIFCLR	022504R	002	LLPFLI	014560R	002
LCOLP1	027444R	002	LCPTG4	033032R	002	LDVRIF	034124R	002	LIFEQ1	022526R	002	LLPGAL	014242R	002
LCOMB	027316R	002	LCPTG5	033042R	002	LENCFD	027740R	002	LIFEQ2	022625R	002	LLPGFC	013726R	002
LCOMO	042040R	002	LCPTG6	033056R	002	LENCMD	030014R	002	LIFGE1	022564R	002	LLPGLB	014400R	002
LCOM1	042042R	002	LCPTG7	033110R	002	LENCNT	027712R	002	LIFGT1	022540R	002	LLPIEN	014700R	002
LCCM2	042044R	002	LCPTG8	033160R	002	LEND	023166R	002	LIFLE1	022577R	002	LLPINL	014602R	002
LCCM3	042046R	002	LCPTG9	033230R	002	LENDEV	030026R	002	LIFLT1	022552R	002	LLPLNF	014270R	002
LCCM4	042050R	002	LCPTM1	034142R	002	LENDFD	027422R	002	LIFNE1	022612R	002	LLPLP1	014160R	002
LCCM5	042052R	002	LCPTOE	034000R	002	LENDFL	024310R	002	LIFNE2	022642R	002	LLPLP2	014224R	002
LCCM6	042054R	002	LCPTSF	033726R	002	LENEEX	030002R	002	LIFST	022464R	002	LLPLP3	014310R	002
LCCM7	042056R	002	LCPT10	033242R	002	LENFCF	027676R	002	LIGNOR	023250R	002	LLPLP4	014334R	002
LCCM8	042060R	002	LCPT11	033276R	002	LENLP1	027656R	002	LIITBS	027354R	002	LLPLP5	014372R	002
LCCM9	042062R	002	LCPT12	033306R	002	LENLP2	027720R	002	LIMIT	044640R	002	LLPLP6	014526R	002
LCOTG1	027456R	002	LCPT13	033324R	002	LENMSG	030006R	002	LINBUF	034262R	002	LLPLP7	014546R	002
LCOTG2	027626R	002	LCPT14	033342R	002	LENTER	016063R	002	LINCDG	015500R	002	LLPLP8	014572R	002
LCPABT	033700R	002	LCPT15	033360R	002	LENTY	023166R	002	LINCDN	015502R	002	LLPMBH	014430R	002
LCPBTP	032574R	002	LCPE7E	033150R	002	LERROR	023251R	002	LINCLN	015366R	002	LLPMDN	014536R	002
LCPBUG	032572R	002	LCRETN	042114R	002	LETGO	023166R	002	LINCR1	022411R	002	LLPMFC	014260R	002
LCPCNR	032460R	002	LCRLF	034430R	002	LEXIT	023166R	002	LINCR2	022413R	002	LLPMUP	014514R	002
LCPEFP	032342R	002	LCSTBL	030034R	002	LFETCH	015034R	002	LINDEX	027116R	002	LLPPAK	013766R	002
LCPEIT	032520R	002	LCTBST	030202R	002	LFEXT	036324R	002	LINDIR	027076R	002	LLPPKB	014264R	002
LCPEMS	034146R	002	LCTCHR	026500R	002	LFGLP1	036300R	002	LINDTB	030620R	002	LLPRFL	015014R	002
LCPERL	034205R	002	LDASLP	041034R	002	LFGLP2	036304R	002	LINFND	027426R	002	LLPRPL	014436R	002
LCPERR	033670R	002	LDA1	041134R	002	LFILG	036274R	002	LININD	027054R	002	LLPRTF	037672R	002
LCPFCS	034144R	002	LDA2	041146R	002	LFILG1	023062R	002	LINNBR	016066R	002	LLPSNP	015036R	002
LCPFCC	032604R	002	LDA3	041160R	002	LFILG2	023127R	002	LIRTBS	027352R	002	LLPSOP	015040R	002
LCPFLN	033632R	002	LDECR1	022411R	002	LFILP	036326R	002	LIST	000050R	002	LLPTDN	014122R	002
LCPFTE	032476R	002	LDECR2	022413R	002	LFILP1	023077R	002	LIWS	027272R	002	LLPTEN	014640R	002
LCPGFC	032440R	002	LDECTB	041310R	002	LFILP2	023147R	002	LJSTFY	015726R	002	LLPTFF	014026R	002
LCPGNE	032432R	002	LDELAY	022411R	002	LFILR	036216R	002	LLIBCT	016302R	002	LLPTFS	014330R	002
LCPGNV	033366R	002	LDELET	023247R	002	LFILR1	023044R	002	LLIBPT	016304R	002	LLPWRN	014670R	002
LCPGTE	032600R	002	LDIADR	024662R	002	LFILR2	023106R	002	LLIEM1	016260R	002	LMCRTB	044644R	002
LCPILE	032264R	002	LDIGPA	024564R	002	LFILV	036152R	002	LLIERR	016230R	002	LMCRTE	045020R	002
LCPLP1	032306R	002	LDILCD	024610R	002	LFILV1	023077R	002	LLIERT	016240R	002	LMIDL	030624R	002
LCPLTX	034110R	002	LDILP1	024520R	002	LFILV2	023147R	002	LLIEW1	016112R	002	LMLMSG	015165R	002
LCPNAL	033534R	002	LDIMSG	024661R	002	LFILN	027370R	002	LLIEW2	016114R	002	LMOFY	016062R	002
LCPNRF	034131R	002	LDIOBJ	024672R	002	LFLLP1	027374R	002	LLIEW3	016210R	002	LMOVE0	023166R	002
LCPNRW	033754R	002	LDIR	027136R	002	LFRCBC	036244R	002	LLIEW4	016212R	002	LMOVE1	022420R	002
LCPNT1	033012R	002	LDIREF	024650R	002	LFREE	030632R	002	LLIEXT	016250R	002	LMOVE2	022425R	002
LCPNT2	033220R	002	LDISP	027216R	002	LFREVN	036252R	002	LLINK	022660R	002	LMPNTR	021752R	002
LCPOBA	034140R	002	LDISVR	024502R	002	LFREXT	036272R	002	LLITBC	016144R	002	LMSTBE	021300R	002
LCPOCT	033520R	002	LDITBC	024540R	002	LFROOD	036230R	002	LLITMB	016172R	002	LNBRWD	026000R	002
LCPOEN	034132R	002	LDITEF	024550R	002	LFROET	036222R	002	LLJST1	016020R	002	LNEGAT	022411R	002
LCPOUT	034014R	002	LDLAY	036764R	002	LFTMSG	015046R	002	LLJUST	016002R	002	LNEMSG	015114R	002
LCPPIB	032472R	002	LDLLP1	036766R	002	LFVCBC	036174R	002	LLOAD	022402R	002	LNFMMSG	015153R	002
LCPPRI	034034R	002	LDLLP2	036772R	002	LFVEVN	036202R	002	LLONUM	041436R	002	LNOTDE	040646R	002
LCPPTS	034134R	002	LDLMSG	015142R	002	LFVEXT	036214R	002	LLPADL	014344R	002	LNOTIT	026770R	002

LOADVR	000052R	002	LPKADP	021006R	002	LPKLP6	017236R	002	LPNEX	040012R	002	LROT2	023161R	002
LOBJLN	015042R	002	LPKAMP	017524R	002	LPKLP8	021014R	002	LPOADR	040232R	002	LRT1	031022R	002
LOCTSW	040400R	002	LPKBIT	020142R	002	LPKL25	016540R	002	LPOCBC	040250R	002	LRT10	031516R	002
LOCTTB	042326R	002	LPKBLP	020152R	002	LPKMPS	023262R	002	LPOCNT	040324R	002	LRT11	031530R	002
LOFF	016075R	002	LPKBT1	020516R	002	LPKNLN	016432R	002	LPOCR1	040316R	002	LRT12	031542R	002
LON	016074R	002	LPKBUF	023264R	002	LPKNXI	016674R	002	LPOCR2	040332R	002	LRT13	031554R	002
LPATA	036624R	002	LPKCBF	016334R	002	LPKOCCT	017644R	002	LPOEVN	040256R	002	LRT14	031566R	002
LPATAP	042110R	002	LPKCBK	017064R	002	LPKPBT	020210R	002	LPOFXT	040276R	002	LRT15	031602R	002
LPATB	036652R	002	LPKCDE	017010R	002	LPKPKP	021272R	002	LPONOM	040322R	002	LRT16	031614R	002
LPATBP	042112R	002	LPKCEM	017402R	002	LPKPRT	020564R	002	LPOODD	040166R	002	LRT17	031626R	002
LPATCH	036662R	002	LPKCIO	016752R	002	LPKPSV	023260R	002	LPOOET	040160R	002	LRT18	031672R	002
LPATEN-	042114R	002	LPKCKT	017754R	002	LPKPTP	021270R	002	LPOPRT	040222R	002	LRT19	031704R	002
LPATEX	036666R	002	LPKCLN	016456R	002	LPKRFD	020470R	002	LPORTN	040326R	002	LRT2	031036R	002
LPATO	036342R	002	LPKCNB	017340R	002	LPKRJF	020512R	002	LPOSDF	040136R	002	LRT20	031716R	002
LPATOP	042064R	002	LPKCRL	020352R	002	LPKSCC	016640R	002	LPRBSY	037664R	002	LRT21	031376R	002
LPAT1	036374R	002	LPKCRW	017120R	002	LPKSDL	016424R	002	LPRBUF	037674R	002	LRT22	031642R	002
LPAT1P	042066R	002	LPKCTE	016542R	002	LPKSDN	016410R	002	LPRCAL	037604R	002	LRT23	031656R	002
LPAT2	036426R	002	LPKCUE	017416R	002	LPKSEP	017260R	002	LPRCNT	037632R	002	LRT27	032106R	002
LPAT2P	042070R	002	LPKDEC	017704R	002	LPKSEV	017206R	002	LPRFWA	037636R	002	LRT28	032122R	002
LPAT3	036456R	002	LPKDER	017374R	002	LPKSEX	016564R	002	LPRNB	040402R	002	LRT29	032202R	002
LPAT3P	042072R	002	LPKEB1	016560R	002	LPKSIG	016416R	002	LPRNB1	022735R	002	LRT3	031410R	002
LPAT4	036530R	002	LPKEJ1	020132R	002	LPKSJ1	020120R	002	LPRNB2	023025R	002	LRT30	032262R	002
LPAT4P	042074R	002	LPKEJ2	020464R	002	LPKSMP	017312R	002	LPRND	040114R	002	LRT4	031422R	002
LPAT5	036542R	002	LPKEPT	023256R	002	LPKSPH	020056R	002	LPRND1	022720R	002	LRT5	031434R	002
LPAT5P	042076R	002	LPKERR	017430R	002	LPKSRS	020136R	002	LPRND2	023005R	002	LRT6	031446R	002
LPAT6	036554R	002	LPKEVN	021026R	002	LPKSR1	020230R	002	LPRNG	040022R	002	LRT7	031460R	002
LPAT6P	042100R	002	LPKEXT	017434R	002	LPKSR2	020250R	002	LPRNG1	022672R	002	LRT8	031472R	002
LPAT7	036566R	002	LPKFA1	021264R	002	LPKSR3	020262R	002	LPRNG2	022751R	002	LRT9	031504R	002
LPAT7P	042102R	002	LPKFA2	021260R	002	LPKSSP	017024R	002	LPRNI	022664R	002	LSAVLN	016042R	002
LPAT8	036600R	002	LPKFA3	021254R	002	LPKSTL	017444R	002	LPRNMR	022667R	002	LSBKUP	026552R	002
LPAT8P	042104R	002	LPKFA4	021250R	002	LPKSVF	017156R	002	LPRNO	040126R	002	LSCKGO	026574R	002
LPAT9	036612R	002	LPKFA5	021244R	002	LPKTAG	017742R	002	LPRNO1	022705R	002	LSCNCT	026626R	002
LPAT9P	042106R	002	LPKFA6	021240R	002	LPKTBE	021750R	002	LPRNO2	022767R	002	LSCNEX	026560R	002
LPAUSE	023166R	002	LPKFA7	021234R	002	LPKTBL	021302R	002	LPROGN	037756R	002	LSCNPT	026624R	002
LPBAOR	040526R	002	LPKFIL	021052R	002	LPKTGA	020710R	002	LPROLN	015044R	002	LSCOUT	026556R	002
LPBASE	036336R	002	LPKFL1	021054R	002	LPKTGD	020020R	002	LPRRDE	037673R	002	LSET	022456R	002
LPBCBC	040544R	002	LPKFL2	021126R	002	LPKTG4	020532R	002	LPRANDM	041324R	002	LSI11 =	000004	
LPBCFM	040536R	002	LPKFL3	021152R	002	LPKTG5	020554R	002	LRO72	015550R	002	LSKBL	026416R	002
LPBCR1	040576R	002	LPKFL4	021176R	002	LPKTG6	020560R	002	LREADY	015246R	002	LSKIP1	041070R	002
LPBEVN	040560R	002	LPKFL5	021206R	002	LPKTG7	020644R	002	LREAD1	023166R	002	LSKIP2	041110R	002
LPBLP1	040450R	002	LPKFT1	021062R	002	LPKTG8	020660R	002	LREAD2	022411R	002	LSMEND	030650R	002
LPBLP2	040470R	002	LPKIF	020302R	002	LPKTG9	020704R	002	LREAD3	023167R	002	LSNLP1	024404R	002
LPB00D	040436R	002	LPKIFD	016714R	002	LPKTNR	020774R	002	LREAD4	023176R	002	LSNTG1	024426R	002
LPBOET	040430R	002	LPKIFL	021276R	002	LPKTXD	021046R	002	LRELOC	043730R	002	LSNTG2	024440R	002
LPBRTN	040572R	002	LPKILU	016516R	002	LPKTXT	021002R	002	LRESET	023166R	002	LSPACE	016064R	002
LPBTG1	040502R	002	LPKITX	021012R	002	LPKUFC	017152R	002	LRETN	022662R	002	LSPFL	016010R	002
LPBTG2	040506R	002	LPKLIN	017454R	002	LPKVAR	017554R	002	LRJST2	015770R	002	LSPTST	015324R	002
LPGAOR	040076R	002	LPKLP8	020044R	002	LPKWD1	020364R	002	LRLLP1	043744R	002	LSPTSV	026630R	002
LPGBYC	040100R	002	LPKLP6	020032R	002	LPKWD2	020424R	002	LRLLP2	044012R	002	LSTOBJ	024660R	002
LPGNCT	040104R	002	LPKLP1	016400R	002	LPLMNZ	024060R	002	LRLLP3	044044R	002	LSUB	036070R	002
LPGTG1	040062R	002	LPKLP3	016646R	002	LPLNTH	024311R	002	LROT1	036732R	002	LSUBCC	036064R	002
LPGTIL	040044R	002	LPKLP4	017046R	002	LPN	040014R	002	LROTRL	036752R	002	LSUBEX	036112R	002
LPHYSL	024312R	002	LPKLP5	017164R	002	LPNADR	040006R	002	LROT1	022411R	002	LSUBM	036020R	002

LSUBML	036050R	002	LTOO	040360R	002	LWRIT1	023166R	002	PSVREG=	000222	SREG	040776R	002
LSUB1	022435R	002	LTSTDE	040616R	002	LWRIT2	022411R	002	PTEM0 =	000056	STACK =	001200	
LSUB2	022444R	002	LTTBIN	027012R	002	LWRIT3	023212R	002	PTEM1 =	000060	START	000064R	002
LSVBBC	025132R	002	LTTYCT	024460R	002	LWRIT4	023221R	002	PTEM10=	000102	STKBIG	035546R	002
LSVBCT	025306R	002	LUPAMP	024104R	002	LXINDT	030634R	002	PTEM11=	000104	STONER=	100000	
LSVBCT	025174R	002	LUPBIT	024154R	002	LXRTBS	027350R	002	PTEM12=	000106	STPTEX	036526R	002
LSVBMM	025142R	002	LUPCIX	016056R	002	LXTBL	030414R	002	PTEM13=	000110	STRPAT	036510R	002
LSVDWR	025246R	002	LUPCTA	024204R	002	L2COOF	034125R	002	PTEM14=	000112	SU4ROT	033154R	002
LSVEXT	025130R	002	LUPDLN	015560R	002	MASS70	034126R	002	PTEM15=	000114	SYNTAX	024320R	002
LSVFPS	025310R	002	LUPDL1	015632R	002	MAXPRG=	000020		PTEM2 =	000062	TAGLUP	026632R	002
LSVMDB	025060R	002	LUPDL2	015662R	002	MMMR3 =	172516		PTEM3 =	000064	TOEADR	034510R	002
LSVMLB	025006R	002	LUPEMB	024106R	002	MMVER =	000001		PTEM4 =	000066	TSTVEC	040734R	002
LSVMOC	025162R	002	LUPEXT	024022R	002	MODIFY	013722R	002	PTEM5 =	000070	UFWCLR	013270R	002
LSVMRL	025270R	002	LUPIFD	023750R	002	MPGEND=	045212R	002	PTEM6 =	000072	UFWMEX	013326R	002
LSVPS2	024712R	002	LUPIMP	024276R	002	NOCOMP=	000001		PTEM7 =	000074	UFWSET	013252R	002
LSVRSW	024762R	002	LUPKPT	024314R	002	NPRFLG	034127R	002	PTEM8 =	000076	UFWTST	013306R	002
LSVRTL	025312R	002	LUPLIN	024174R	002	OCPRES=	000100		PTEM9 =	000100	ULCRLF	040304R	002
LSVTEN	025014R	002	LUPLP1	023420R	002	OCTBIN	042132R	002	PTEND =	000242	ULIST	035640R	002
LSVUPL	024766R	002	LUPLP2	023504R	002	OPENL	000054R	002	PTLGH=	000242	ULISTM	035622R	002
LSVWFT	025150R	002	LUPLP3	023650R	002	PACK	016306R	002	PTCNT=	000030	ULSTCM	035650R	002
LSYMF	027002R	002	LUPLP4	023756R	002	PASCIN=	000006		PTSIZE=	000240	UNPACK	023374R	002
LSYMT	027356R	002	LUPNPI	023642R	002	PATCH	001652R	002	PUSRPC=	000236	UPCI	015224R	002
LSYNDT	027202R	002	LUPPTP	024316R	002	PATGEN	036670R	002	PUT	000056R	UPCRLF	040312R	002
LSYNIT	027152R	002	LUPSEP	023574R	002	PC	=%000007		PUTBYT	040772R	UPLI	016132R	002
LTAB	016072R	002	LUPSEV	023524R	002	PCURDV=	000035		PWRIA=	000020	UPLIO	016104R	002
LTAGCT	027362R	002	LUPSEX	023670R	002	PDNUMS=	000036		RDIOSZ	012414R	UPRADR	035672R	002
LTBEND	023245R	002	LUPSIN	023774R	002	PDPNTR=	000034		REALNM	000062R	UPRASC	040026R	002
LTBERR	042302R	002	LUPSMP	023614R	002	PDST =	000122		RJUST1	015762R	UPRBYC	035674R	002
LTBFD	042224R	002	LUPTAG	024260R	002	PFBBOV=	000002		RREG	041010R	URSTOP=	000002	
LTBIN0	042162R	002	LUPTG1	024206R	002	PFLGWD=	000000		RST1	000070R	USMTPS=	000002	
LTBIN1	042176R	002	LUPTMF	023630R	002	PFWADR=	000004		RST2	000124R	USRINT	000226R	002
LTBIN2	042200R	002	LUPVAR	024230R	002	PGCOMP	035566R	002	RTNINT	040764R	USRLP	037530R	002
LTBIN3	042236R	002	LVEADB	037344R	002	PGENEX	036730R	002	R0	=%000000	USRPEX	037656R	002
LTBIN4	042246R	002	LVECL1	037162R	002	PG6END=	042116R	002	R1	=%000001	USRPR1	037510R	002
LTBINS	042264R	002	LVECL2	037226R	002	PLNGTH=	000026		R2	=%000002	USRPR1	037514R	002
LTBOUT	042310R	002	LVECMP	037274R	002	PMDLCD=	000032		R3	=%000003	VDFLSZ	042566R	002
LTBSAV	042146R	002	LVECT	023233R	002	PNAME =	000010		R4	=%000004	VECEX	040756R	002
LTGADR	027364R	002	LVEDEP	037166R	002	PNBR =	000116		R5	=%000005	WRIOSZ	012424R	002
LTGLN1	026756R	002	LVEHDN	037416R	002	POBJST=	000024		R6	=%000006	WT4IOT=	000010	
LTGLN2	026746R	002	LVEHOR	037372R	002	POPSW =	000002		R7	=%000007	XCALC	027126R	002
LTLEP2	026656R	002	LVEMSG	037456R	002	PRDIOA=	000016		SAVE	024702R	XXXX =	000000	
LTLEP3	026670R	002	LVEREX	037256R	002	PRONER=	020000		SCAN	026406R	ZERO	000060R	002
LTLEXT	027334R	002	LVERFY	037004R	002	PSRC =	000120		SETDED=	000040	.	= 045212R	002
. ABS.	000000	000											
	000000	001											
MPG	045212	002											

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

* ,DTMGAC/NL:TOC/DOC=DTMGAC.P11
RUN-TIME: 35 71 4 SECONDS
RUN-TIME RATIO: 141/111=1.2
CORE USED: BK (15 PAGES)

N03

MAINDEC-11-DTMGA-C MPG CONTROL ROUTINE
DTMGAC.P11 SYMBOL TABLE

MACY11 27(732) 24-SEP-76 13:54 PAGE 63-5

SEQ 0247

DOCUMENT PAGES: 246

