JOHNNIAC-STYLE E

Problem No.__________ Programmer____________________ Date____________ Page____ of______

ID		LOCN		OPN		ADDR		OPN		ADDR		COMMENTS
01 02 03 04 05 06 07 08	09 10 11 12 13	14 15 16 17	18	19 20 21	22	23 24 25 26	27	28 29 30	31	32 33 34 35	36 37 38 39 40	41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

LIST OF JOHNNIAC OPERATIONS - June 1, 1963

Operation Codes are Base Eight Numbers. Notes are Base Ten

000		Proceed to next order in sequence	004	LM	Clear MQ, M → MQ
001	TNL	If A < 0, c → left command in M	005	TNR	If A < 0, c → right command in M
002	TPL	If A ≥ 0, c → left command in M	006	TPR	If A ≥ 0, c → right command in M
003	TFL	If overflow, c → left command in M	007	TFR	If overflow, c → right command in M
010	TRL	c → left command in M	014	TRR	c → right command in M
011	T1L	If T_1 on, c → left command in M	015	T1R	If T_1 on, c → right command in M
012	T2L	If T_2 on, c → left command in M	016	T2R	If T_2 on, c → right command in M
013	T3L	If T_3 on, c → left command in M	017	T3R	If T_3 on, c → right command in M
020	RA	Clear A, M → A	024	A	M + A → A
021	RS	Clear A, - M → A	025	S	M + A → A
022	RAV	Clear A, | M | → A	026	AV	| M | + A → A
023	RSV	Clear A, -| M | → A	027	SV	-| M | + A → A
030	MR	Clear A, M · MQ rounded → A	034	MB	M · MQ + 2^{-39} A + ½(1-$_1$ A $_1$) → A and MQ
031	MNR	Clear A, -M · MQ rounded → A	035	MNB	-M · MQ + 2^{-39} A + ½(1-$_1$ A $_1$) → A and MQ
032	M	Clear A, M · MQ → A and MQ	036	MA	M · MQ + 2^{-39} A → A and MQ
033	MN	Clear A, -M · MQ → A and MQ	037	MNA	-M · MQ + 2^{-39} A → A and MQ
040	DS	A ÷ M → MQ, r → A	044	D	(A + 2^{-39} MQ) ÷ M → MQ, r → A
041	DNS	A ÷ (-M) → MQ, r → A	045	DN	(A + 2^{-39} MQ) ÷ (M) → MQ, r → A
050	ST	A → M	054	SAB	$_7A_{19}$ and $_{28}A_{39}$ → $_7M_{19}$ and $_{28}M_{39}$
051	SOL	$_0A_6$ → $_0M_6$	055	SOR	$_{20}A_{27}$ → $_{20}M_{27}$
052	SAL	$_7A_{19}$ → $_7M_{19}$	056	SAR	$_{28}A_{39}$ → $_{28}M_{39}$
053	SHL	$_0A_{19}$ → $_0M_{19}$	057	SHR	$_{20}A_{39}$ → $_{20}M_{39}$
060	STQ	Clear A, MQ → A and M	064	AQS	MQ + A → A and M
061	SNQ	Clear A, -MQ → A and M	065	SQS	- MQ + A → A and M
062	SVQ	Clear A, | MQ | → A and M	066	AVS	| MQ | + A → A and M
063	SNV	Clear A, -| MQ | → A and M	067	SVS	-| MQ | + A → A and M
070	SRC	Clear MQ, shift A right n places Zeros into A_0.	074	SRH	Shift A right n places. Zeros into A_0.
071	CLC	Clear MQ, circular shift of A and MQ left n places. Couple MQ_0 to A_{39}, A_0 to MQ_{39}.	075	CLH	Circular shift of A and MQ left n places. Couple MQ_0 to A_{39}; A_0 to MQ_{39}.
072	LRC	Clear MQ, power shift A and MQ right n places. Couple A_{39} to MQ_1. 0^A0 to → $0^{MQ}0$.	076	LRH	Power shift A and MQ right n places. Couple A_{39} to MQ_1. 0^A0 → $0^{MQ}0$.
073	LLC	Clear MQ, power shift A and MQ left n places. Couple zeros into MQ_{39}; MQ_1 to A_{39}.	077	LLH	Power shift A and MQ left n places. Couple zeros into MQ_{39}, MQ_1 to A_{39}.
100	SEL	Select I-O Address Part of 100: XXX0 Pri. Feed Reader; XXX1 Sec. Feed Reader; XXX2 Feed Punch; XXX3 Feed Punch & Echo; XXX4 Sel. left 80 col. of Printer; XXX5 Sel. right 80 " " "; XXX6 Sel. Plotter	104	DIS	Display
			105	HUT	Hoot
			106	EJ	Address part of 106: XXX0 Restore one page; XXX1 Advance 1 print line; XXX2 Advance 2 print lines
101	C	M Copy Order M → 40 Leftmost Selected Col. A → 40 Rightmost " "	107		Read Cloak → A

110 RD — Read drum words to M and memory addresses following numerically. Denoting MQ as xxx $f_1f_2f_3f_4$ dpb $l_1l_2l_3l_4$, the f's determine the first drum address and the l's the last drum address. d selects the drum; p, the position of the heads; and b, the bank to be read.

111 WD — Read M and words in memory addresses following numerically to drum. MQ has the same significance as in 110.

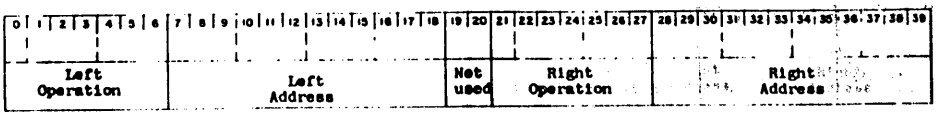
120	ZTA	Clear A to Zero	124	PI	M I A → A
121		Clear A	125	NI	- M I A → A (- denotes digit inversion
122		Clear A	126	PMI	| M | I A → A of M).
123		Clear A	127	NMI	-| M | I A → A
130	HTL	Halt c → left command in M	134	HTR	Halt c → right command in M
131	H1L	Halt if H_1 on; c → left command in M	135	H1R	Halt if H_1 on; c → right command in M
132	H2L	Halt if H_2 on; c → left command in M	136	H2R	Halt if H_2 on; c → right command in M
133	H3L	Halt if H_3 on; c → left command in M	137	H3R	Halt if H_3 on; c → right command in M
140		Write line buffer	144		Search all SCRs for Match
141		Read line buffer	145		Search all SCRs for Mismatch
142		Write SCR	146		Display Graphic I/O
143		Read SCR	147		Read Graphic I/O Tablet
150					
160					
170					

*A copy order (10.1) directed to the Plotter as selected by 10.0 XXX6 gates only the contents of the specified memory word to the Plotter register with the following meaning:

$_6M_6$-Circle Radius
$_7M_{18}$-X Magnitude
$_{21}M_{24}$-Character Selection
M_{25}-Circle Drawing
M_{26}-Line Drawing
M_{27}-Arm Selection
$_{28}M_{39}$-Y Magnitude

DEFINITIONS

A	Accumulator
MQ	Multiplier Quotient Register
M	Word in the M^{th} address in Internal Storage
$_{20}A_{39}$	Digits in position 2^{-20} through 2^{-39} of the word in A
c →	Control goes to
I	Logical (digit by digit) product or intersection



JOSS routines:

Stan

= 0 Read clock
= 1 Step one character.
= 2 Advance to nonspace
= 3 Eliminate spaces
= 4 Put (P1) in next S cell (P2)+1
= 5 Converter
= 6 Packer
= 7 Unpacker
= 8 Assign buffer.
= 9 Assign drum
= 10 Deal out a storage space
= 11 Unpack (Q3)
= 12 Push Q2-Q3 Operand PDL
= 13 Pop Q2-Q3 Operand PDL
= 14 Push Q4-Q5 Operator PDL
= 15 Pop Q4-Q5 Operator PDL
= 16 Push Q6-Q7 Auxiliary PDL
= 17 Pop Q6-Q7 Auxiliary PDL
= 18 Push Q8-Q9 Control PDL
= 19 Pop Q8-Q9 Control PDL
= 20 Evaluate unsigned numerical expression
= 21 Evaluate expression
= 22 Type time line.
= 23 Evaluate condition
= 24 Convert function to operator
= 25 Output line from S to stn. (Stan only)
= 26 Evaluate grouped list
= 27 Accumulate letters (6 max)
= 28 Erase left linked list
= 29 Erase right linked list
= 30 Check range and pack (Q3)
= 31 Add
= 32 Subtract
= 33 Multiply
= 34 Divide
= 35 Exponentiate
= 36 Convert integer for output
= 37 Simple direct tests
= 38 Simple indirect tests
= 39 Verify space and advance to nonspace in R
=40 Positive integer test on (Q3)
=41 Subscript range test and replace (Q3)
=42 Limit range test on (Q3)
=43 Deck test on (Q3)
=44 Part test on (Q3)
=45 Step test on (Q3)
=46 Form test on (Q3)
=47 Move stn to bottom of list.
=48 'TO' routine
=49 Update time of last activity
=50 Find part o/c P20
=51 Find step o/c P21
=52 Find form o/c P22
=53 Kick out current program (if any)
=54 Bring in program o/c stn (if any)
=55 Switch to user (CL+3U)
=56 Erase step o/c H21-H26.
=57 Erase form o/c H24-H26.
=58 Release current buffer (if any) for stn.
=59 Release drum to available list.
=60 Initialize drum (H19) for stn (H12).
=61 Convert time for output.
=62 Insert space in output.
=63 Copy step number from R to S.
=64 Unpack message to S for output compo
=65 Output canned message.
=66 Put (P11) in next S cell (P12)+1.
=67 Block transfer R→S.
=68 Compute part (P20) for step (P21)
=69 Insert period and CR+EOM in output
=70 Compare (H21) w (H22).
=71 Find single indexed value.
=72 Find double indexed value.
=73 Assign value.
=74 Evaluate indexed letter.
=75 Erase value(s) for letter.
=76 Erase one level of control structure.
=77 Match groupers
=78 Cancel.
=79 Verify preceding apostrophe

=80 Transmit line for drum routine.
=

Space added

X 0 ~~Drum routine~~ for logging.
X 1 Q service
X 2 Signal service
X 3 Task selection
X 4 Fire
X 5 Advance
X 6 Fetch
X 7 Ascend (or iterate)
X 8
X 9
X 10 ~~Process input~~ SIGNAL SOURCE (DRUM)
X 11 ~~Drum routine~~ for Q service
X 12 (Transmit and) switch
X 13 ~~Treat interrupts~~
X 14 Transmit and → X5
X 15 Read drum routine and fire.

D 0 Error routine
D 1 Ext of G5 for 'all ---'
D 2 Ext of G5 for spec. values.
D 3 Ext of G6 for 'all ---'
D 4 Ext of D3 for 'all values'.
D 5 Octal I-O
D 6 Ext of G6 for spec. values.
D 7 Ext of G6 for values in form.
D 8 Ext of G6 for step, part, form, size

G 5 Delete (step, part, form)
G 6 Types (via D extensions)

G 0 ~~Set~~
G 1 Do
G 2 To
G 3 Done
G 4 Cancel
G 5 Delete
G 6 Type
G 7 Line
G 8 Page
G 9 (Read)
G 10 (Punch)
G 11 Form
G 12 Stop
G 13 Go
G 14 Demand
G 15

JOSS 666 gcs 1-7-65

Region	Origin (8)	Length allowed	Unused
Signal	0000	1	0
J135	0001	5	0
Routines	0006	2266	4
W	4340	16	0
L	4360	16	0
U (U_0-U_6)	4400	88	6
J	4530	24	5
I	4560	16	6
H	4600	32	0
E	4640	32	0
K	4700	48	4
N	4760	16	4
A	5000	132	0
U7	5204	10	0
	5216	2	2
M (Drum)	5220	208	2
D	5540	12	3
B	5554	20	4
=	5600	80	0
F	5720	16	3
G	5740	16	1
X	5760	16	0
P	6000	32	0
Q	6040	16	0
Z	6060	8	7
T	6070	8	0
S	6100	75	0
V	6213	53	0
R	6300	82	0
	6422	26	26
Initial Spare Tank	6454	4	0
Available Space	6460	720	0

U_0 STATE / STN
U_1 BUFFER / STN
U_2 DRUM / STN
U_3 NEXT STN / STN
U_4 TIME / STN
U_5 NEXT BUFFER / BUFFER
U_6 NEXT DRUM / DRUM
U_7 INITIALS

User Block

JOSS Regions:

= Directory for routines.

A Character Properties of Input Characters

B Drum control words.

C System Statistics

D Drum Routinesarea

E Error Directory

F Function Directory

G Process Directory

H Context

I Integers live

J Parameters, Origins, Constants

K Constants. GENERAL

L Lists.

M Core area for drum routines.

N Table of powers of ten.

#

P Single data words.

Q Lists

R Read-write-interpret block (82 words).

S Output block.

T Temporary storage.

U Tables

V Variables

W Words

X Executive Routines

Y

Z Run context and statistics

User block

```
A---E000+    A  0 100 0000 000,0000    TERMINATION SIGNAL
A---E001     A  1 000 0000 001,6001    1
A---E002     A  2 000 0000 001,6002    2
A---E003     A  3 000 0000 001,5003    3
A---E004     A  4 000 0000 001,6004    4
A---E005     A  5 000 0000 001,5005    5
A---E006     A  6 000 0000 001,5006    6
A---E007     A  7 000 0000 001,4007    7
A---E008     A  8 000 0000 001,6010    8
A---E009     A  9 000 0000 001,5011    9
A---E010     A 10 000 0000 000,3161
A---E011     A 11 000 0000 000,3161
A---E012     A 12 000 0000 000,3161
A---E013     A 13 000 0000 000,3161
A---E014     A 14 000 0000 000,4016    SPACE
A---E015     A 15 000 0000 000,3161
A---E016     A 16 000 0000 000,6020    SEMICOLON
A---E017     A 17 001 V  1 002,5021    LCA
A---E018     A 18 002 V  2 002,5022    LCB
A---E019     A 19 003 V  3 002,4023    LCC
A---E020     A 20 004 V  4 002,5024    LCD
A---E021     A 21 005 V  5 002,4025    LCE
A---E022     A 22 006 V  6 002,4026    LCF
A---E023     A 23 007 V  7 002,3027    LCG
A---E024     A 24 010 V  8 002,5030    LCH
A---E025     A 25 011 V  9 002,4031    LCI
A---E026     A 26 000 0000 000,3161
A---E027     A 27 000 0000 000,3033    .
A---E028     A 28 000 0000 000,3161
A---E029     A 29 000 0000 000,3161
A---E030     A 30 000 0000 000,3036    EJECT+CR+EOM
A---E031     A 31 000 0000 000,2037    EJECT+CR
A---E032     A 32 000 = 33 240,6040    MULTIPLY DOT
A---E033     A 33 012 V 10 002,5041    LCJ
A---E034     A 34 013 V 11 002,5042    LCK
A---E035     A 35 014 V 12 002,4043    LCL
A---E036     A 36 015 V 13 002,5044    LCM
A---E037     A 37 016 V 14 002,4045    LCN
A---E038     A 38 017 V 15 002,4046    LCO
A---E039     A 39 020 V 16 002,3047    LCP
A---E040     A 40 021 V 17 002,5050    LCQ
A---E041     A 41 022 V 18 002,4051    LCR
A---E042     A 42 100 0000 000,4052    CR+EOM+TERMINAL SIGNAL IN SIGN
A---E043     A 43 002 0000 004,3053    =
A---E044     A 44 000 = 32 140,4054    -
A---E045     A 45 100 0000 000,3416    SPECIAL SPACE
A---E046     A 46 000 0000 000,3161
A---E047     A 47 000 0000 000,3161
A---E048     A 48 000 0000 001,5060    ZERO
A---E049     A 49 000 = 34 240,4061    /
A---E050     A 50 023 V 19 002,4062    LCS
A---E051     A 51 024 V 20 002,3063    LCT
A---E052     A 52 025 V 21 002,4064    LCU
A---E053     A 53 026 V 22 002,3065    LCV
A---E054     A 54 027 V 23 002,3066    LCW
```

```
A---E055    A 55 030 V 24 002,2067    LCX
A---E056    A 56 031 V 25 002,4070    LCY
A---E057    A 57 032 V 26 002,3071    LCZ
A---E058    A 58 000 0000 000,3161
A---E059    A 59 000 0000 000,2073    ,
A---E060    A 60 000 = 31 140,3074    +
A---E061    A 61 000 0000 000,2075    TAB
A---E062    A 62 000 0000 000,3161
A---E063    A 63 000 0000 000,3161
A---E064    A 64 000 0000 000,3161
A---E065    A 65 000 0000 000,5101    PRIME
A---E066    A 66 000 0000 000,5102    QUOTE MARKS
A---E067    A 67 000 0000 000,4016    STRIKE OUT CHARACTER = SPACE
A---E068    A 68 000 0000 000,5104    $
A---E069    A 69 003 0000 004,4105    LESS THAN OR =
A---E070    A 70 006 0000 004,4106    GREATER THAN OR =
A---E071    A 71 001 0000 004,3107    LESS THAN
A---E072    A 72 004 0000 004,5110    GREATER THAN
A---E073    A 73 000 A112 020,4111    (
A---E074    A 74 000 0000 000,3161
A---E075    A 75 000 0000 000,3161
A---E076    A 76 000 0000 000,3161
A---E077    A 77 000 0000 000,3161
A---E078    A 78 000 0000 000,3161
A---E079    A 79 000 0000 000,3161
A---E080    A 80 000 0000 000,5120    COLON
A---E081    A 81 041 V 27 002,4121    A
A---E082    A 82 042 V 28 002,4122    B
A---E083    A 83 043 V 29 002,3123    C
A---E084    A 84 044 V 30 002,4124    D
A---E085    A 85 045 V 31 002,3125    E
A---E086    A 86 046 V 32 002,3126    F
A---E087    A 87 047 V 33 002,2127    G
A---E088    A 88 050 V 34 002,4130    H
A---E089    A 89 051 V 35 002,3131    I
A---E090    A 90 000 0000 000,3161
A---E091    A 91 000 A123 010,2133    RIGHT BRACKET
A---E092    A 92 000 0000 000,3161
A---E093    A 93 000 0000 000,3161
A---E094    A 94 000 0000 000,3161
A---E095    A 95 000 0000 000,3161
A---E096    A 96 000 A 96 030,5140    ABS VAL BAR
A---E097    A 97 052 V 36 002,4141    J
A---E098    A 98 053 V 37 002,4142    K
A---E099    A 99 054 V 38 002,3143    L
A---E100    A100 055 V 39 002,4144    M
A---E101    A101 056 V 40 002,3145    N
A---E102    A102 057 V 41 002,3146    O
A---E103    A103 060 V 42 002,2147    P
A---E104    A104 061 V 43 002,4150    Q
A---E105    A105 062 V 44 002,3151    R
A---E106    A106 000 0000 000,3161
A---E107    A107 005 0000 004,2153    NOT =
A---E108    A108 000 0000 000,3154    UNDERSCORE
A---E109    A109 000 0000 000,2433    PERIOD WITH FLAG (CHAR 155 CAN'T GET
```

```
A---E110    A110 000 0000 000,3161
A---E111    A111 000 0000 000,3161
A---E112    A112 000 A 73 010,4160      )
\---E113    A113 000 0000 000,3161      QUESTION MARK
A---E114    A114 063 V 45 002,3162      S
A---E115    A115 064 V 46 002,2163      T
A---E116    A116 065 V 47 002,3164      U
A---E117    A117 066 V 48 002,2165      V
A---E118    A118 067 V 49 002,2166      W
A---E119    A119 070 V 50 002,1167      X
A---E120    A120 071 V 51 002,3170      Y
A---E121    A121 072 V 52 002,2171      Z
A---E122    A122 000 0000 000,3161
A---E123    A123 000 A091 020,1173      LEFT BRACKET
A---E124    A124 000 = 35 440,2174      *
A---E125    A125 000 0000 000,1175      TAB
A---E126    A126 000 0000 000,3161
A---E127    A127 000 0000 000,3161
A---E128    A128 000 0000 000,0037      EJECT
A---E129    A129 000 0000 000,0056      CARRIAGE RETURN
A---E130    A130 000 0000 000,0155      PSEUDO PERIOD
A---E131    A131 000 0000 000,0055      PSEUDO SPACE
```

```
D000E000+        $                    M  0     BODY OF ERROR ROUTINE
D000E010     *   1 020 S   0 050 P 12          SET POINTER
D000E015           020 P  18 124 K  6
D000E020           025 *  90 050 T  0
D000E025           024 *  91 052 *  6
D000E030           021 T   0 002 *  8          JUMP IF E0
D000E035           072     21 050 P 18
D000E040           020 Q   8 002 * 23          JUMP IF INDIRECT
D000E045           020 P  18 024 I  6
D000E050           050 T   0 023 T  0
D000E055           002 *   8 010 * 24          JUMP IF DIRECT AND E6
D000E060     *  23 020 P  23 001 * 24          JUMP UNLESS TROUBLE IN X7
D000E062           021 T   0 024 K  2
D000E064           001 *  24 010 $  1          JUMP IF NOT OVERFLOW IN X7
D000E066           020 $   0 010 = 13          POP Q3
D000E068           014 X   7                   BACK TO X7 IGNORING OVERFLOW
D000E070     *  24 020 $   0 010 = 64          'ERROR-'
D000E080                 *  81      * 81
D000E085           020 Q   8 001 *  4          JUMP IF AT TOP LEVEL
D000E090           020 P  23 001 *  2          JUMP IF NO TROUBLE IN X7
D000E100           020 $   0 010 = 64          'DURING-'
D000E110                 *  82      * 82
D000E120           020 Q   8 024 K  1
D000E125           056 *   9 014 *  9
D000E130     *   9                020 ----
D000E135           001 *   4 050 P 21          JUMP IF TROUBLE IN TOP LEVEL DO-FOR
D000E140           010 *   3
D000E160     *   2 020 $   0 010 = 64          'AT-'
D000E170                 *  83      * 83
D000E180     *   3 020 $   0 010 = 64
D000E190                 *  84      * 84       'STEP-'
D000E200           020 K   4 050 P 16          CLEAR P16
D000E205           020 P  23 002 * 10          JUMP IF TROUBLE IN X7
D000E210           020 Q   9 050 P 21
D000E220     *  10 020 $   0 010 = 51          FIND STEP
D000E230           001 *   7 056 P  3
D000E240           020 $   0 010 =  7          UNPACK
D000E242           020 J  12 050 P  2          COPY STEP NUMBER FROM R TO S
D000E244     *  31 020 $   0 010 =  2
D000E246           124 K  35 025 K 35
D000E248           002 *  32 010 $  1
D000E250           020 P   1 124 A 27
D000E252           025 A  27 001 *  5
D000E254     *  32 020 P  12 024 K  3
D000E256           050 P  12 056 * 33
D000E258     *  33 020 P   1 050 ----
D000E260           020 *  31 010 =  1
D000E270     *   4 020 $   0 010 = 64          'ABOVE'
D000E280                 *  85      * 85
D000E290     *   5 020 $   0 010 = 64          COLON AND TWO SPACES
D000E300                 *  86      * 86
D000E302           020 P  18 024 I  2
D000E304           050 T   0 023 T  0
D000E306           002 *  94 010 $  1          JUMP IF E2
```

```
D000E310      *   8 004 H 31 110 * 80
D000E320      *   6 004 ---- 110 * 80
D000E330            020 $   0 010 = 64      ERROR MESSAGE
D000E340                * 80      * 80
D000E350            010 * 11
D000E360      *   7 020 S   0 050 P 12
D000E365            020 $   0 010 = 64      'IT'S A MESS.  LET'S START OVER.'
D000E370                * 87      * 87
D000E380            020 Q   8 001 * 22      JUMP IF NO CONTROL STRUCTURE
D000E385      *  21 020 $   0 010 = 76      ERASE ONE LEVEL OF CONTROL STRUCTURE
D000E390            020 Q   8 002 * 21      LOOP IF MORE
D000E395      *  22 020 K   4 050 P 23      RESET P23 FLAG
D000E400      *  11 020 Q   2 001 * 12      CLEAN UP OPERAND STACK
D000E410            020 $   0 010 = 13
D000E420            010 * 11
D000E430      *  12 020 Q   4 001 * 13      CLEAN UP OPERATOR STACK
D000E440            020 $   0 010 = 15
D000E450            010 * 12
D000E460      *  13 020 Q   6 001 * 14      CLEAN UP AUXILIARY STACK
D000E470            020 $   0 010 = 17
D000E480            010 * 13
D000E490      *  14 020 Q 14 052 * 15      CLEAN UP Q14
D000E500      *  15 020 ---- 005 * 16
D000E510            052 * 15 071    40
D000E520            020 $   0 010 = 29
D000E530      *  16 010 * 15 004 Q 14
D000E540            020 $   0 010 = 28
D000E550            120     0 050 Q 14
D000E560            004 Q 15 010 $   1      CLEAN UP Q15
D000E570            020 $   0 010 = 29
D000E580            120     0 050 Q 15
D000E590            020 Q   1 002 * 20
D000E600            020 Q   0 056 * 17      REFILL SPARE TANK
D000E610      *  17 050 Q   1 020 ----
D000E620            056 * 18 004 K   4
D000E630      *  18 056 * 19 020 ----
D000E640      *  19 050 Q   0 060 ----
D000E650      *  20 010 X 12              EXIT TO TRANSMIT AND SWITCH
D000E680      *  80                       ERROR MSG INDEX AND TEXT BLOCK
D000E690         $                  $ 31
D000E700      *  81 052,4511,222,3051     'ERROR-'
D000E710            007,2000,000,0000
D000E720      *  82 012,0641,221,4445     'DURING-'
D000E730            013,4164,000,0000
D000E740      *  83 010,4630,350,0000     'AT-'
D000E750      *  84 031,0630,522,3416     'STEP-'
D000E760            100,0000,000,0000
D000E770      *  85 010,4221,143,2425     'ABOVE'
D000E780            100,0000,000,0000
D000E790      *  86 050,0160,350,0000     COLON AND TWO SPACES
D000E800      *  87 054,4632,023,1016     'IT'S A MESS. LET'S START OVER.'
D000E810            010,4161,101,2462
D000E820            031,0330,340,7143
D000E830            012,4632,023,1016
D000E840            031,0630,422,4463
```

```
D000E850          007,0461,521,2451
D000E860          015,4524,000,0000
D000E870     * 90      E  0
D000E880     * 91      * 80
D000E900     * 94 020 H 27 050 P 11      CONSTRUCT UNDEFINED MESSAGE
D000E902          020 $  0 010 = 66      PLACE LETTER
D000E904          020 H 28 001 *  8      JUMP IF NO INDICES TO COMPLETE MESSAG
D000E906          072   12 050 H 20
D000E908          020 J 20 052 * 96      =77 LEFT RIGHT GROUPER IN J20
D000E909     * 96 020 ---- 050 P 11
D000E910          020 $  0 010 = 66      PLACE LEFT GROUPER
D000E912          020 $  0 010 = 36      CONVERT FIRST INDEX
D000E914                 1        2
D000E916          020 H 29 001 * 95      JUMP IF NO SECOND INDEX
D000E918          072   12 050 H 20
D000E920          020 A 59 050 P 11
D000E922          020 $  0 010 = 66      PLACE COMMA
D000E924          020 $  0 010 = 36      CONVERT SECOND INDEX
D000E926                 1        2
D000E928     * 95 020 J 20 050 P 11      =77 LEFT RIGHT GROUPER IN J20
D000E930          020 $  0 010 = 66      PLACE RIGHT GROUPER
D000E932          010 *  8          ,    JUMP BACK TO COMPLETE MESSAGE
D000E999        /     D  0 010,7000/
```

```
D001E000+       $                   M  0    BRANCH OF G5 DELETE
D001E010          020 P  1 124 K 19
D001E020          025 K 19 002 * 40         JUMP IF TERMINAL
D001E030          020 $  0 010 = 39         VERIFY SPACE AND ADVANCE TO NONSPACE
D001E040          020 P  1 124 K 19
D001E050          025 K 19 002 * 40         JUMP IF TERMINAL
D001E060          020 $  0 010 = 27         ACCUMULATE WORD
D001E070          020 $  0 010 =  3         ELIMINATE SPACES
D001E080          124 K 19 025 K 19
D001E090          001 $  1 004 H 19         MALFORMED IF NOT TERMINAL
D001E100          020 W 12 065 T  0
D001E110          023 T  0 002 * 10         JUMP IF 'VALUES'
D001E120          020 W 10 065 T  0
D001E130          023 T  0 002 * 20         JUMP IF 'STEPS'
D001E140          020 W  8 065 T  0
D001E150          023 T  0 002 * 20         JUMP IF 'PARTS'
D001E160          020 W  4 065 T  0
D001E170          023 T  0 002 * 30         JUMP IF 'FORMS'
D001E180          010 E  6                  ELSE MALFORMED
D001E190    * 10  020 $  0 010 * 50         ERASE ALL VALUES
D001E200          010 X  5
D001E210    * 20  020 Q  8 002 E  9         ERASE ALL STEPS (OR PARTS) - ILLEGAL
D001E220          020 $  0 010 * 60
D001E230          010 X  5
D001E240    * 30  020 Q  8 002 E  9         ERASE ALL FORMS - ILLEGAL INDIRECT
D001E250          020 $  0 010 * 70
D001E260          010 X  5
D001E270    * 40  020 Q  8 002 E  9         ERASE ALL - ILLEGAL INDIRECT
D001E280          020 $  0 010 * 50
D001E290          020 $  0 010 * 60
D001E300          020 $  0 010 * 70
D001E310          010 X  5
D001E320    * 50  024 K  2 052 * 59         ERASE ALL VALUES (SUBROUTINE)
D001E330          020 * 58 052 * 51
D001E340    * 51  020 ---- 050 H 27
D001E350          124 K 34 025 K 34
D001E360          001 * 52 010 $  1
D001E370          020 $  0 010 = 75
D001E380    * 52  020 * 51 024 K  2
D001E390          052 * 51 020 H 27
D001E400          124 A121 025 A121
D001E410    * 59  002 ---- 010 * 51
D001E420    * 58      A 17
D001E430    * 60  024 K  2 052 * 69         ERASE ALL STEPS (SUBROUTINE)
D001E440          020 Q 10 050 * 90
D001E450    * 61  020 * 90 056 * 62
D001E460          124 K  5 025 K  1
D001E470    * 62  005 * 66 020 ----
D001E480          050 * 90 050 * 91
D001E490    * 63  020 * 91 052 * 64
D001E500          124 K  6 025 K  1
D001E510          005 * 65 010 $  1
D001E520    * 64  004 ---- 060 * 91
D001E530          020 $  0 010 = 29
```

```
D001E540      * 65 010 * 63 004 * 90
D001E550           020 $  0 010 = 28
D001E560      * 66 010 * 61 004 Q 10
D001E570           020 $  0 010 = 29
D001E580           120    0 050 Q 10
D001E582           020 Q 12 001 * 69      JUMP IF NO SUSPENDED TASK
D001E584           050 Q  8 020 K  4
D001E586           050 Q 12 050 Q 13
D001E588           050 P 17 050 P 23      RESET P17 AND P23 FLAGS
D001E590      * 67 020 $  0 010 = 76      ERASE ONE LEVEL
D001E592           020 Q  8 002 * 67
D001E600      * 69 010 ----
D001E610      * 70 024 K  2 052 * 79      ERASE ALL FORMS (SUBROUTINE)
D001E620           020 Q 11 050 * 91
D001E630      * 71 020 * 91 052 * 72
D001E640           124 K  6 025 K  1
D001E650           005 * 73 010 $  1
D001E660      * 72 004 ---- 060 * 91
D001E670           020 $  0 010 = 29
D001E680      * 73 010 * 71 004 Q 11
D001E690           020 $  0 010 = 28
D001E700           120    0 050 Q 11
D001E710      * 79 010 ----
D001E720      * 90                        LOCAL STORAGE
D001E730      * 91                   ,    LOCAL STORAGE
D001E999         /     D001 010,7000/
```

```
D002E000+       $                   M  0    BRANCH OF G5 DELETE
D002E005          020 $  0 010 *  70        BORROW SPARE TANK
D002E010    *  0 020 P  1 124 K  34
D002E020          025 K 34 001 E   6        VERIFY LETTER
D002E030          020 P  1 050 H  27        SAVE IT
D002E040          020 $  0 010 =   1        ADVANCE ONE
D002E050          020 P  1 124 K  31
D002E060          025 K 31 002 *   1        JUMP IF LEFT GROUPER
D002E070          020 $  0 010 =   3        ELIMINATE SPACES
D002E080          020 $  0 010 =  12        PUSH Q3
D002E090          020 K  4 050 H  28
D002E100          050 H 29 010 $   1
D002E102          020 H 27 052 *  22
D002E104    * 22 022 ---- 002 *   3        JUMP IF VALUE FOUND
D002E106          010 E  2                  ELSE ERROR
D002E110    *  1 020 H 27 050 *  90        SAVE LETTER
D002E120          020 $  0 010 =  26        EVALUATE SUBSCRIPTS
D002E130          020 * 90 050 H  27        SET UP LETTER AGAIN
D002E140          020 Q  7 050 H  29
D002E150          020 Q  6 001 *   2        JUMP IF Q7 EMPTY
D002E160          020 $  0 010 =  17        POP Q7
D002E170          020 Q  6 002 E   6        MALFORMED IF EXTRA INDICES
D002E180    *  2 020 $  0 010 =  41        CHECK AND CONVERT FIRST INDEX
D002E190          020 Q  3 050 H  28        SET UP FIRST INDEX
D002E200          020 H 29 002 *  21        JUMP IF DOUBLY INDEXED
D002E201          020 H 27 052 *  23
D002E202    * 23 020 ---- 002 E   2
D002E203          071     6 002 E   2
D002E204          020 $  0 010 =  71
D002E205          002 *  3 010 E   2
D002E210    * 21 050 Q  3 010 $   1
D002E220          020 $  0 010 =  41        CHECK AND CONVERT SECOND INDEX
D002E230          020 Q  3 050 H  29        SET UP SECOND INDEX
D002E232          020 $  0 010 =  72        FIND VALUE
D002E234          002 $  1 010 E   2        JUMP IF FOUND, ELSE ERROR
D002E240    *  3 020 H 27 124 K  17        PACK INFO INTO Q3 FOR LATER DELETE
D002E250          050 Q  3 020 H  28
D002E260          001 *  4 024 K  26
D002E270          024 Q  3 050 Q   3
D002E280          020 H 29 001 *   4
D002E290          071     9 024 K  26
D002E300          024 Q  3 050 Q   3
D002E310    *  4 020 P  1 124 A  59
D002E320          025 A 59 001 *   5        JUMP IF NOT COMMA
D002E330          020 $  0 010 =   2        ADVANCE TO NONSPACE
D002E340          010 *  0                  LOOP BACK
D002E350    *  5 020 $  0 010 =   3        ELIMINATE SPACES
D002E360          124 K 19 025 K  19        VERIFY TERMINAL
D002E370          001 E  6 010 $   1
D002E380    *  6 020 Q  3 024 J  13        UNPACK AND DELETE
D002E390          056 *  7 010 $   1
D002E400          020 K  4 050 H  28
D002E410    *  7 050 H 29 020 ----
D002E420          050 H 27 020 Q   3
```

```
D002E430          025 K 26 002 *  8     JUMP IF INDEXED
D002E440          020 $  0 010 = 75     DELETE VALUE(S) FOR LETTER
D002E450          010 * 19
D002E460     *  8 124 K 42 050 H 28
D002E470          020 Q  3 025 K 26
D002E480          025 K 26 002 * 13     JUMP IF DOUBLE INDEX
D002E490          020 H 27 052 * 14     SINGLE INDEX
D002E500     * 14 020 ---- 002 * 19     JUMP IF NOT A VECTOR
D002E510          071    6 002 * 19
D002E520          020 $  0 010 = 71     FIND VALUE
D002E530          001 * 19 010 $  1     JUMP IF CAN'T
D002E540     * 20 020 H 21 056 *  9
D002E550          020 H 22 056 * 10
D002E560     *  9 020 H 23 056 ----
D002E570     * 10 020 Q  0 050 ----
D002E580          020 H 22 124 K  5
D002E590          050 Q  0 020 H 27
D002E600          052 * 11 052 * 12
D002E610     * 11 020 ---- 124 K  5
D002E620          025 K  1 002 * 19     JUMP IF DONE
D002E630          020 K  4 010 $  1
D002E640     * 12 050 ---- 010 * 19     UNDEFINE LETTER IF NO VALUES AND JUMP
D002E650     * 13 070    9 124 K 42     DOUBLE INDEX
D002E660          050 H 29 010 $  1
D002E670          020 H 27 052 * 15
D002E680     * 15 020 ---- 002 * 19     JUMP IF NOT A MATRIX
D002E690          071    5 002 * 19
D002E700          020 $  0 010 = 72     FIND VALUE
D002E710          001 * 19 020 H 24     JUMP IF CAN'T
D002E720          052 * 16 020 H 25
D002E730          052 * 17 020 H 26
D002E740     * 16 052 ---- 020 Q  0
D002E750     * 17 050 ---- 020 H 25
D002E760          070   21 050 Q  0
D002E770          020 H 22 056 * 18
D002E780     * 18 020 K  6 124 ----
D002E790          025 K  1 001 * 20     JUMP IF NO MORE VALUES IN ROW
D002E800     * 19 020 $  0 010 = 13     POP Q3
D002E810          020 Q  2 002 *  6     LOOP IF MORE TO DELETE
D002E815          020 $  0 010 * 80     REFILL SPARE TANK
D002E820          010 X  5              EXIT
D002E830     * 70 024 K  2 052 * 79     BORROW SPARE TANK (SUBROUTINE)
D002E835          020 Q  1 056 * 71
D002E840     * 71              020 ----
D002E845          056 * 72 010 $  1
D002E850     * 72 020 Q  0 050 ----
D002E855          020 Q  1 050 Q  0
D002E860          020 K  4 050 Q  1
D002E865     * 79 010 ----
D002E870     * 80 024 K  2 052 * 89     REFILL SPARE TANK
D002E875          020 Q  0 056 * 81
D002E880     * 81 050 Q  1 020 ----
D002E885          056 * 82 004 K  4
D002E890     * 82 056 * 83 020 ----
D002E895     * 83 050 Q  0 060 ----
```

```
D002E900      * 89 010 ----
D002E910      * 90                        ,     LOCAL STORAGE
D002E999        /      D002 010,7000/
```

```
D003E000+       $                       M  0    EXTENSION OF G6 FOR 'ALL ----'
D003E010          020 $  0 010 =  3      ELIMINATE SPACES
D003E020          020 P  1 124 K 19
D003E030          025 K 19 002 * 10      JUMP IF TERMINAL
D003E040          020 $  0 010 = 27      ACCUMULATE WORD
D003E050          020 $  0 010 =  3      ELIMINATE SPACES
D003E060          124 K 19 025 K 19      VERIFY TERMINAL
D003E070          001 E  6 004 H 19
D003E080          020 W 10 065 T  0
D003E090          023 T  0 002 * 20      JUMP IF 'STEPS'
D003E100          020 W  8 065 T  0
D003E110          023 T  0 002 * 20      JUMP IF 'PARTS'
D003E120          020 W  4 065 T  0
D003E130          023 T  0 002 * 30      JUMP IF 'FORMS'
D003E140          020 W 12 065 T  0
D003E150          023 T  0 002 * 40      JUMP IF 'VALUES'
D003E160          010 E  6               ELSE MALFORMED OR ILLEGAL
D003E170    * 10  020 $  0 010 * 50      TYPE ALL (STEPS FIRST)
D003E180          020 $  0 010 * 60      THEN FORMS
D003E190          010 D  4               THEN WADE INTO D4 FOR VALUES
D003E200    * 20  020 $  0 010 * 50      TYPE ALL STEPS (OR PARTS)
D003E210          010 X  5
D003E220    * 30  020 $  0 010 * 60      TYPE ALL FORMS
D003E230          010 X  5
D003E240    * 40  010 D  4               WADE INTO D4 FOR VALUES
D003E250    * 50  024 K  2 052 P 26      TYPE ALL STEPS (SUBROUTINE)
D003E260          020 Q 10 050 P 18      INITIALIZE P18
D003E270    * 51  020 A 42 050 S  1
D003E280          020 $  0 010 = 78      LINE FEED
D003E290          020 P 18 056 * 52
D003E300    * 52                020 ----
D003E310          001 P 26 050 P 18      EXIT VIA P26 WHEN DONE
D003E320          050 P 19 010 $  1
D003E330    * 53  020 P 19 052 * 54      TYPE STEPS WITHIN PART
D003E340    * 54  020 ---- 001 * 51
D003E350          050 P 19 056 P  3
D003E360          020 $  0 014 =  7      UNPACK TO S
D003E380          020 P 15 002 * 70      JUMP IF INTERRUPT SIGNAL
D003E390          020 $  0 010 = 78      TRANSMIT STEP
D003E400          010 * 53
D003E410    * 60  024 K  2 052 P 26      TYPE ALL FORMS (SUBROUTINE)
D003E430          020 Q 11 050 P 19
D003E440    * 61  020 A 42 050 S  1
D003E450          020 $  0 010 = 78      LINE FEED
D003E460          020 P 19 052 * 62
D003E470          024 K  2 052 * 63
D003E480    * 62  020 ---- 001 P 26      EXIT VIA P26 IF NO MORE FORMS
D003E490          050 P 19 056 P  3
D003E500          020 S  0 050 P 12      INITIALIZE OUTPUT STRING POINTER
D003E510          020 $  0 010 = 64      INSERT 'FORM' IN OUTPUT STRING
D003E520              * 66       * 66
D003E525          020 $  0 010 = 62      INSERT SPACE
D003E530    * 63  020 ---- 050 T  0
D003E540          124 K 13 050 T  1
```

```
D003E550          020 T  0 124 K 11
D003E560          070   10 050 T  2
D003E570          020 J 14 025 T  2
D003E580          052 * 64 010 $  1
D003E590          120    0 004 T  1
D003E600     * 64 044 ---- 060 H 20
D003E610          020 $  0 010 = 36      CONVERT FORM NUMBER TO OUTPUT STRING
D003E620                 1         9
D003E630          020 A 80 050 P 11      INSERT COLON
D003E640          020 $  0 010 = 66
D003E650          020 A 42 050 P 11      INSERT CR+EOM
D003E660          020 $  0 010 = 66
D003E670          020 P 15 002 * 70      JUMP IF INTERRUPT SIGNAL
D003E680          020 $  0 010 = 78      TRANSMIT FORM NUMBER LINE
D003E690          020 $  0 014 =  7      UNPACK TO S
D003E710          020 $  0 010 = 78      TRANSMIT FORM
D003E720          010 * 61               LOOP
D003E760     * 66 053,0461,222,2200      'FORM'
D003E770     * 70 020 H 12 024 K 32      FAKE AN 'RI'
D003E780          050 H  9 010 X 10,       IN H9 AND JUMP
D003E999        /     D003 010,7000/
```

```
D004E000+       $                   M  0    EXTENSION OF G6 VIA D3 FOR 'ALL VALUE
D004E010          120     0 050 P 18        INITIALIZE PHASE TO 0
D004E015          020 A 42 050 S  1
D004E020          020 $  0 010 =  78        LINE FEED
D004E025    *  0 020 K  4 050 P  19        RESET LINE FEED FLAG
D004E030          020 * 90 050 P  30        INITIALIZE POINTER
D004E040    *  1 020 P 30 052 *  2
D004E050    *  2 020 ---- 050 H  27        PICK UP NEXT CHARACTER WORD
D004E060          052 *  3 124 K  34
D004E070          025 K 34 001 *  5         JUMP IF NOT LETTER
D004E080    *  3 020 ---- 050 H  30        PICK UP HEAD OR VALUE
D004E090          002 *  4 071     5        JUMP IF SCALAR
D004E100          001 * 20 071     1        JUMP IF MATRIX
D004E110          001 * 10 010 *  5         JUMP IF VECTOR ELSE UNDEFINED
D004E120    *  4 023 P 18 001 *  5         JUMP UNLESS SCALAR PHASE
D004E125          050 P 19 020 K  4         SET LF FLAG
D004E130          050 H 28 050 H  29
D004E140          020 $  0 010 *  50        TYPE SCALAR VALUE
D004E150    *  5 020 P 30 024 K  3         ADVANCE SCANNER
D004E152          050 P 30 025 *  91
D004E154          001 *  1 010 $  1         LOOP IF MORE
D004E156          020 P 19 001 *  6         JUMP IF NO LINE FEED FLAG
D004E158          020 A 42 050 S  1
D004E160          020 $  0 010 =  78        LINE FEED
D004E165    *  6 020 P 18 024 K  26        ADVANCE PHASE
D004E170          050 P 18 025 K  36
D004E175          001 *  0 010 X  5         JUMP IF MORE PHASES, ELSE EXIT
D004E180    * 10 020 P 18 071     6         VECTOR
D004E185          002 *  5 020 H  27        JUMP IF NOT VECTOR PHASE
D004E190          050 P 27 020 H  30
D004E200    * 11 050 P 28 020 P  28
D004E205          056 * 12 024 K  1
D004E210    * 12 056 * 13 020 ----
D004E215          001 * 30 010 $  1
D004E220          050 P 28 124 K  42
D004E230    * 13 050 H 28 020 ----
D004E235          050 H 30 010 $  1
D004E240          020 P 27 050 H  27
D004E245          020 K  4 050 H  29
D004E250          020 $  0 010 *  50        TYPE VECTOR ELEMENT
D004E255          014 * 11                  LOOP
D004E260    * 20 020 P 18 071     5         MATRIX
D004E265          002 *  5 010 $  1
D004E270          020 H 27 050 P  27
D004E275          020 H 30 050 P  28
D004E280    * 21 020 P 28 056 * 22
D004E290    * 22           020 ----
D004E300          001 * 30 050 P  28
D004E310          050 P 29 010 $  1
D004E320    * 23 020 P 29 052 * 24
D004E330          024 K  2 052 * 25
D004E340    * 24 020 ---- 001 * 21
D004E350          050 P 29 124 K  42
D004E360          050 H 29 020 P  28
```

```
D004E370          124 K 42 050 H 28
D004E380          020 P 27 050 H 27
D004E390     * 25 020 ---- 050 H 30
D004E400          020 $  0 010 * 50      TYPE MATRIX ELEMENT
D004E410          010 * 23               LOOP
D004E420     * 30 020 A 42 050 S  1
D004E430          020 $  0 010 = 78      LINE FEED
D004E440          010 *  5
D004E500     * 50 050 P 31 010 $  1      SAVE LINK
D004E502          021 I  7 050 * 93
D004E504          020 H 28 001 * 53
D004E506          025 * 92 002 * 51
D004E508          020 I  3 014 * 51
D004E510     * 51 020 I  4 024 * 93
D004E512          050 * 93 010 $  1
D004E514          020 H 29 001 * 53
D004E516          025 * 92 002 * 52
D004E518          020 I  2 014 * 52
D004E520     * 52 020 I  3 024 * 93
D004E522          050 * 93 010 $  1
D004E524     * 53 020 S  0 050 P 12      INITIALIZE OUTPUT LINE
D004E526     * 69 020 * 93 002 * 54      JUMP WHEN ALL SPACES ARE IN
D004E528          024 I  1 050 * 93
D004E530          020 $  0 010 = 62      INSERT SPACE
D004E532          010 * 69               LOOP
D004E534     * 54 020 H 27 050 P 11
D004E536          020 $  0 010 = 66      INSERT LETTER
D004E538          020 H 28 001 * 56      JUMP IF NO SUBSCRIPTS
D004E540          020 A 73 050 P 11
D004E542          020 $  0 010 = 66      INSERT LEFT PARENTHESIS
D004E544          020 H 28 070    12
D004E546          050 H 20 010 $  1
D004E548          020 $  0 010 = 36      CONVERT FIRST SUBSCRIPT
D004E550                   1       2
D004E552          020 H 29 001 * 55      JUMP IF NO 2ND SUBSCRIPT
D004E554          020 A 59 050 P 11
D004E556          020 $  0 010 = 66      INSERT COMMA
D004E558          020 H 29 070    12
D004E560          050 H 20 010 $  1
D004E562          020 $  0 010 = 36      CONVERT SECOND SUBSCRIPT
D004E564                   1       2
D004E566     * 55 020 A112 050 P 11
D004E568          020 $  0 010 = 66      INSERT RIGHT PAREN
D004E570     * 56 020 $  0 010 = 62      INSERT SPACE
D004E572          020 A 43 050 P 11
D004E574          020 $  0 010 = 66      INSERT '='
D004E576          020 $  0 010 = 62      INSERT SPACE
D004E578          020 H 30 124 K 11
D004E580          071     1 072    32
D004E582          050 * 94 024 I  3
D004E584          001 * 65 025 I  9      JUMP IF SCIENTIFIC NOTATION REQUIRED
D004E586          002 * 65 020 * 94      JUMP IF SCIENTIFIC NOTATION REQUIRED
D004E588          002 * 71 021 I  1      JUMP IF NEGATIVE XP
D004E589     * 71 025 I  6 050 * 93      SET SPACE COUNTER
D004E590          020 H 30 071     9
```

```
D004E592          002 * 57 020 * 93
D004E594          024 I  1 050 * 93      ALLOW FOR MINUS SIGN
D004E596    * 57 020 * 93 002 * 58
D004E598          024 K  1 050 * 93
D004E600          020 $  0 010 = 62      INSERT SPACE
D004E602          010 * 57
D004E604    * 58 020 H 30 124 K 12
D004E606          025 K 12 001 * 59      JUMP IF PLUS
D004E608          020 A 44 050 P 11
D004E610          020 $  0 010 = 66      INSERT MINUS SIGN
D004E612    * 59 020 * 94 001 * 63
D004E614          024 I  1 056 * 61
D004E616          020 J 14 025 * 94
D004E618          056 * 60 020 I  8
D004E620          025 * 94 056 * 62
D004E622          020 H 30 124 K 13
D004E624    * 60 071    40 044 ----
D004E626          050 * 94 060 H 20
D004E628          020 $  0 010 = 36      CONVERT INTEGRAL PART
D004E630    * 61           1     ----
D004E632          023 * 94 002 * 68      JUMP IF NO FRACTIONAL PART
D004E634    * 64 020 A 27 050 P 11
D004E636          020 $  0 010 = 66      INSERT DECIMAL POINT
D004E638          020 * 94 050 H 20
D004E640          020 $  0 010 = 36      CONVERT FRACTIONAL PART
D004E642    * 62           4     ----
D004E644          010 * 68               JUMP TO PUT IN CR+EOM
D004E646    * 63 020 I  8 025 * 94
D004E647          056 * 62 010 $  1
D004E648          020 H 30 124 K 13
D004E650          050 * 94 010 * 64
D004E652    * 65 020 $  0 010 = 64      SCIENTIFIC NOTATION
D004E653               * 96     * 96     INSERT 5 SPACES
D004E654          020 H 30 124 K 12
D004E655          025 K 12 001 * 66      JUMP IF NO MINUS SIGN
D004E656          020 A 44 014 * 66      INSERT MINUS
D004E658    * 66 020 A 14 050 P 11      INSERT SPACE
D004E660          020 $  0 010 = 66
D004E662          020 H 30 124 K 13
D004E664          071    40 044 N  8
D004E666          050 * 93 060 H 20
D004E668          020 $  0 010 = 36      CONVERT WHOLE NUMBER OF COEF
D004E670                1          1
D004E671          023 * 93 002 * 70      JUMP IF NO FRACTIONAL PART
D004E672          020 A 27 050 P 11
D004E674          020 $  0 010 = 66      INSERT DECIMAL PT
D004E676          020 * 93 050 H 20
D004E678          020 $  0 010 = 36      CONVERT FRACTION OF COEF
D004E680                4          8
D004E682    * 70 020 $  0 010 = 64      INSERT •10*
D004E684               * 95     * 95
D004E686          020 * 94 006 * 67      JUMP IF SFX PLUS
D004E688          020 A 73 050 P 11      INSERT (
D004E690          020 $  0 010 = 66
D004E692          020 A 44 050 P 11      INSERT -
```

```
D004E694          020 $  0 010 = 66
D004E696     * 67 021 * 94 050 H 20
D004E698          020 $  0 010 = 36      CONVERT SFX
D004E700                 1        2
D004E702          020 * 94 002 * 68      JUMP IF SFX PLUS
D004E704          020 A112 050 P 11      INSERT )
D004E706          020 $  0 010 = 66
D004E708     * 68 020 A 42 050 P 11      INSERT CR+EOM
D004E710          020 $  0 010 = 66
D004E712          020 P 15 001 * 72      JUMP IF NO INTERRUPT
D004E713          020 H 12 024 K 32      FAKE AN 'RI'
D004E714          050 H  9 010 X 10        IN H9 AND JUMP
D004E715     * 72 020 P 31 010 = 78      TRANSMIT (PASS LINK ON TO =80)
D004E716     * 90      A 17      A 17
D004E718     * 91      A122      A122
D004E720     * 92+             10   27
D004E722     * 93                       LOCAL STORAGE
D004E724     * 94                       LOCAL STORAGE
D004E726     * 95 020,0011,407,6200      .10*
D004E728     * 96 007,0160,340,7016      FIVE SPACES
D004E730          100,0000,000,0000,
D004E999        /      D  4 010,7000/
```

```
D005E000+        $                       M  0      OCTAL IN AND OUT
D005E010           020 $  0 010 =  1
D005E020           020 P  1 124 A 14
D005E030           025 A 14 001 E  6
D005E040           020 $  0 010 =  1
D005E050           020 P  1 124 A 61
D005E060           025 A 61 001 E  6
D005E065           020 $  0 010 =  1
D005E070           120    0 050 P 18
D005E080     *  0 020 P  1 124 K 35
D005E090           025 K 35 001 *  4
D005E100           020 P  1 072    3
D005E110           020 P 18 077    3
D005E120           050 P 18 010 $  1
D005E130           020 $  0 010 =  1
D005E140           010 *  0
D005E150     *  4 020 $  0 010 =  3
D005E160           001 * 10 010 $  1
D005E170           120    0 050 P 19
D005E180     *  1 020 P  1 124 K 35
D005E190           025 K 35 001 *  2
D005E200           020 P  1 072    3
D005E210           020 P 19 077    3
D005E220           050 P 19 010 $  1
D005E230           020 $  0 010 =  2
D005E240           010 *  1
D005E250     *  2 020 P 18 056 *  3
D005E260     *  3 020 P 19 050 ----
D005E270           014 X 12
D005E280     * 10 020 P 18 056 * 11
D005E290     * 11          020 ----
D005E300           050 P 19 010 $  1
D005E310           020 A 61 050 S  1
D005E320           050 S  3 020 A 14
D005E330           050 S  2 020 S  0
D005E340           024 K 43 050 P 12
D005E360           020 P 18 071   68
D005E370           020 $  0 010 * 50
D005E380                           4
D005E390           020 $  0 010 = 62
D005E400           020 $  0 010 = 62
D005E410           020 P 19 071   38
D005E420           020 $  0 010 * 50
D005E430                           3
D005E440           020 $  0 010 = 62
D005E450           020 $  0 010 * 50
D005E460                           4
D005E470           020 $  0 010 = 62
D005E480           020 $  0 010 * 50
D005E490                           3
D005E500           020 $  0 010 = 62
D005E510           020 P 19 071   68
D005E520           020 $  0 010 * 50
D005E530                           4
```

```
D005E540           020 A 42 050 P 11
D005E550           020 $  0 010 = 66
D005E560           010 X 14
D005E570      * 50 024 K  2 052 * 51
D005E580           024 K  2 052 * 59
D005E590      * 51 021 ---- 050 P 27
D005E600      * 53 120    0 075    3
D005E610           025 K  1 002 * 52
D005E620           020 A 48 014 * 52
D005E630      * 52 024 K  1 050 P 11
D005E640           020 $  0 010 = 66
D005E650           020 P 27 024 K  1
D005E660           050 P 27 001 * 53
D005E670      * 59 010 ----        ,
D005E999         /     D005 010,7000/
```

```
D006E000+       $                       M  0    EXTENSION OF G6 FOR VALUES
D006E005    *   0 020 Q  6 001 X  5     QUIT WHEN Q6-Q7 EMPTY
D006E015          020 Q  7 002 *  20    JUMP IF EXPRESSION + VALUE
D006E020          022 Q  7 001 *  10    JUMP IF -1 FOR BLANK
D006E025          020 Q  7 052 *   1
D006E030    *   1 020 ---- 071      5
D006E035          001 * 40 010 *  30    JUMP IF MATRIX ELSE VECTOR
D006E040    *  10 020 A 42 050 S   1    UNDERLINE CALLS FOR LINE FEED
D006E045          020 $  0 010 =  17    POP Q7
D006E050          020 P 15 002 *  80    JUMP IF INTERRUPT
D006E052    *  11 020 Q  6 001 X  14    EXIT VIA X14 IF LAST ITEM
D006E055          020 $  0 010 =  78    TRANSMIT
D006E060          010 *  0
D006E062    *  20 020 S  0 050 P  12
D006E065          020 Q  7 050 H  30    MOVE VALUE
D006E066          020 $  0 010 =  17    POP Q7
D006E067          020 Q  7 124 K   5
D006E070          025 I  7 002 *  22    JUMP IF EXPRESSION IS 8 OR MORE CHARA
D006E075          020 Q  7 124 K   5
D006E080          025 I  7 050 P  18
D006E085    *  21 020 $  0 010 =  62    INSERT SPACE
D006E090          020 P 18 024 K   1
D006E095          050 P 18 001 *  21    LOOP UNTIL ALL LEADING SPACES INSERTE
D006E100    *  22 020 Q  7 052 *  23
D006E105    *  23 020 ---- 050 P  11
D006E110          020 $  0 010 =  66    INSERT CHARACTER
D006E115          020 * 23 024 K   2
D006E120          052 * 23 020 Q   7
D006E125          124 K  5 025 K   1
D006E130          056 Q  7 002 *  23    LOOP UNTIL EXPRESSION COPIED TO S
D006E135          020 $  0 010 =  17    POP Q7
D006E145          020 $  0 010 *  60    CONVERT VALUE ETC
D006E160          010 * 11
D006E165    *  30 020 Q  7 125 K   4    VECTOR
D006E170          050 P 21 052 *  31
D006E175          020 $  0 010 =  17    POP Q7
D006E180    *  31 020 ---- 050 P  19
D006E185    *  32 020 P 19 056 *  33
D006E190          024 K  1 010 $   1
D006E195    *  33 056 * 34 020 ----
D006E200          001 *  0 010 $   1    JUMP WHEN DONE
D006E205          050 P 19 124 K  42
D006E210    *  34 050 H 28 020 ----
D006E215          050 H 30 010 $   1
D006E220          020 K  4 050 H  29
D006E225          020 $  0 010 *  50    CONVERT LABEL
D006E230          020 $  0 010 *  60    CONVERT VALUE
D006E235          020 $  0 010 =  78    TRANSMIT
D006E240          010 * 32              LOOP
D006E245    *  40 020 Q  7 125 K   4    MATRIX
D006E250          050 P 21 052 *  41
D006E255          020 $  0 010 =  17    POP Q7
D006E260    *  41 020 ---- 050 P  18
D006E265    *  42 020 P 18 056 *  43
```

```
D006E270     * 43                020 ----
D006E275           001 *   0 050 P  18     JUMP WHEN DONE
D006E280           050 P  19 010 $   1
D006E285     * 44 020 P  19 052 *  45
D006E290           024 K   2 052 *  46
D006E295     * 45 020 ----  001 *  42     JUMP IF DONE WITH ROW
D006E300           050 P  19 124 K  42
D006E305           050 H  29 020 P  18
D006E310           124 K  42 050 H  28
D006E315     * 46 020 ----  050 H  30
D006E320           020 $   0 010 *  50     CONVERT LABEL
D006E325           020 $   0 010 *  60     CONVERT VALUE
D006E330           020 $   0 010 =  78     TRANSMIT
D006E335           010 *  44
D006E340     * 50 024 K   2 052 *  59     CONVERT LABEL SUBROUTINE
D006E342           020 S   0 050 P  12
D006E345           020 H  28 025 *  92     COMPUTE AND INSERT INITIAL SPACES
D006E350           006 *  51 021 I   4
D006E355     * 51 010 *  52 021 I   3
D006E360     * 52 050 *  90 020 H  29
D006E365           001 *  55 025 *  92
D006E370           006 *  53 020 I   2
D006E375     * 53 010 *  54 020 I   3
D006E380     * 54 024 *  90 050 *  90
D006E385     * 55 020 *  90 002 *  56     JUMP WHEN ALL SPACES ARE IN
D006E390           024 I   1 050 *  90
D006E395           020 $   0 010 =  62
D006E400           010 *  55
D006E405     * 56 020 P  21 050 P  11     INSERT LETTER
D006E410           020 $   0 010 =  66
D006E415           020 A  73 050 P  11     INSERT LEFT PARENTHESIS
D006E420           020 $   0 010 =  66
D006E425           020 H  28 070    12
D006E430           050 H  20 010 $   1
D006E435           020 $   0 010 =  36     CONVERT FIRST INDEX
D006E440                   1         2
D006E445           020 H  29 001 *  57     JUMP IF NO SECOND INDEX
D006E450           020 A  59 050 P  11     INSERT COMMA
D006E455           020 $   0 010 =  66
D006E460           020 H  29 070    12
D006E465           050 H  20 010 $   1
D006E470           020 $   0 010 =  36     CONVERT SECOND INDEX
D006E475                   1         2
D006E480     * 57 020 A112 050 P  11
D006E485           020 $   0 010 =  66     INSERT RIGHT PARENTHESIS
D006E490     * 59 010 ----
D006E495     * 60 024 K   2 052 P  26     CONVERT VALUE ETC
D006E500           020 P  15 002 *  80     JUMP IF INTERRUPT
D006E505           020 P  12 025 *  93
D006E510           001 *  61 010 $   1     JUMP IF VALUE WILL FIT ON SAME LINE
D006E515           020 A  42 050 P  11     INSERT CR+EOM
D006E520           020 $   0 010 =  66
D006E522           020 H  30 050 P  30     SAVE VALUE AGAINST POSSIBLE CHOKE
D006E525           020 $   0 010 =  78     TRANSMIT FIRST PART
D006E527           020 P  30 050 H  30     RESTORE VALUE AFTER POSSIBLE CHOKE
```

```
D006E530           020 S  0 050 P 12
D006E535           020 $  0 010 = 64
D006E540               * 94     * 94
D006E545      * 61 020 $  0 010 = 62      INSERT SPACE
D006E550           020 A 43 050 P 11      INSERT '='
D006E555           020 $  0 010 = 66
D006E556           020 $  0 010 = 62      INSERT SPACE
D006E560           020 * 96 050 * 90      SET POTENTIAL SIGN POSITION
D006E565           020 H 30 124 K 11
D006E570           071    1 072   32
D006E575           050 * 91 024 I  3
D006E580           001 * 62 025 I  9      JUMP IF SCI NOTATION REQUIRED
D006E585           002 * 62 020 * 91      JUMP IF SCI NOTATION REQUIRED
D006E590           002 * 65 021 I  1
D006E592      * 65 050 T  0 020 * 90      ADJUST SIGN POSITION FOR SCALE FACTOR
D006E594           025 T  0 050 * 90
D006E596           071   21 052 * 90
D006E600      * 62 020 P 12 024 K  3
D006E605           025 * 90 002 * 63      JUMP IF SIGN POSITION IS NEXT
D006E610           020 $  0 010 = 62      INSERT SPACE
D006E615           010 * 62               LOOP
D006E620      * 63 020 H 30 124 K 12
D006E625           025 K 12 001 * 64      JUMP IF PLUS
D006E630           020 A 44 014 * 64      INSERT MINUS
D006E635      * 64 020 A 14 050 P 11      INSERT SPACE
D006E640           020 $  0 010 = 66
D006E642           020 * 91 024 I  3
D006E644           001 * 75 025 I  9      JUMP IF SCI NOTATION REQUIRED
D006E646           002 * 75 010 $  1      JUMP IF SCI NOTATION REQUIRED
D006E650           020 * 91 001 * 73
D006E655           024 I  1 056 * 71
D006E660           020 J 14 025 * 91
D006E665           056 * 70 020 I  8
D006E670           025 * 91 056 * 72
D006E675           020 H 30 124 K 13
D006E680      * 70 071   40 044 ----
D006E685           050 * 91 060 H 20
D006E690           020 $  0 010 = 36      CONVERT INTEGRAL PART
D006E695      * 71         1     ----
D006E700           023 * 91 002 * 78      JUMP IF NO FRACTIONAL PART
D006E705      * 74 020 A 27 050 P 11      INSERT DECIMAL POINT
D006E710           020 $  0 010 = 66
D006E715           020 * 91 050 H 20
D006E720           020 $  0 010 = 36      CONVERT FRACTIONAL PART
D006E725      * 72         4     ----
D006E730           010 * 78               JUMP TO CLOSE LINE
D006E735      * 73 020 I  8 025 * 91
D006E740           056 * 72 020 H 30
D006E745           124 K 13 050 * 91
D006E750           010 * 74
D006E775      * 75 020 H 30 124 K 13
D006E780           071   40 044 N  8
D006E785           050 * 90 060 H 20
D006E790           020 $  0 010 = 36      CONVERT WHOLE NUMBER OF COEF
D006E795                 1       1
```

```
D006E797          023 * 90 002 * 79     JUMP IF NO FRACTIONAL PART
D006E800          020 A 27 050 P 11     INSERT DECIMAL PT
D006E805          020 $  0 010 = 66
D006E810          020 * 90 050 H 20
D006E815          020 $  0 010 = 36     CONVERT FRACTION OF COEF
D006E820                4          8
D006E825     * 79 020 $  0 010 = 64     INSERT .10*
D006E830               * 95      * 95
D006E835          020 * 91 006 * 77     JUMP IF SFX PLUS
D006E840          020 A 73 050 P 11     INSERT (
D006E845          020 $  0 010 = 66
D006E850          020 A 44 050 P 11     INSERT -
D006E855          020 $  0 010 = 66
D006E860     * 77 021 * 91 050 H 20
D006E865          020 $  0 010 = 36     CONVERT SFX
D006E870                1          2
D006E875          020 * 91 002 * 78     JUMP IF SFX PLUS
D006E880          020 A112 050 P 11     INSERT )
D006E885          020 $  0 010 = 66
D006E890     * 78 020 A 42 050 P 11     INSERT CR+EOM
D006E895          020 $  0 010 = 66
D006E900          010 P 26              EXIT VIA P26
D006E902     * 81 020 $  0 010 = 17     POP Q6-Q7
D006E904     * 80 020 Q  6 002 * 81     JUMP IF Q6 HAS SOMETHING
D006E906          020 H 12 024 K 32     FAKE AN 'RI'
D006E908          050 H  9 010 X 10       IN H9 AND JUMP
D006E910     * 90                       LOCAL STORAGE
D006E915     * 91                       LOCAL STORAGE
D006E920     * 92+           10   27
D006E925     * 93      S 50      S 50
D006E930     * 94 007,0160,340,7016     EIGHT SPACE MSG
D006E935          007,0160,350,0000
D006E940     * 95 020,0011,407,6200     .10*
D006E945     * 96      S 17      S 17,
D006E999        /     D006 010,7000/
```

```
D007E000+       $                       M  0    EXTENSION OF G6 FOR VALUES IN A FORM
D007E010    *   0 020 S   0 050 P 12    SET POINTER FOR FORM
D007E012    *   1 020 K   4 050 H 21    RESET POINTERS TO -1
D007E014          050 H 22 050 H 23
D007E016    *   2 020 $   0 010 * 85    ADVANCE (OK EVEN AT RIGHT MARGIN)
D007E018          002 * 10 010 *  5      JUMP IF UNDERLINE, JUMP IF OTHER
D007E020          020 P 12 050 H 22      PERIOD--RECORD LOC'N OF POINT
D007E022          020 $   0 010 * 85     ADVANCE (PERIOD SO FAR)
D007E024          002 * 12 010 *  1      JUMP IF UNDERLINE, LOOP IF OTHER
D007E026          020 H 22 050 H 23      PERIOD--RECORD SCI NOTATION BEGINNING
D007E028    *   3 020 $   0 010 * 85     ADVANCE
D007E030          005 *  4 010 E 30      JUMP IF OTHER, ELSE ERROR
D007E032    *   4 010 *  3 020 P 12      LOOP IF PERIOD
D007E034          025 H 23 025 * 97
D007E036          002 * 20 010 E 15      JUMP IF 7 OR MORE PERIODS, ELSE ERROR
D007E038    *   5 020 P 11 002 *  2      JUMP UNLESS END OF FORM
D007E040    *   6 020 Q   6 001 X 14     EXIT TO TRANSMIT IF NO EXCESS VALUES
D007E042          010 E  8               ELSE ERROR
D007E050    *  10 020 P 12 050 H 21      RECORD WHOLE (UNDERLINE SO FAR)
D007E052    *  11 020 $   0 010 * 85     ADVANCE
D007E054          002 * 11 010 * 20      LOOP IF UNDERLINE, JUMP IF OTHER
D007E056          020 P 12 050 H 22      RECORD LOC'N OF POINT
D007E058          020 $   0 010 * 85     ADVANCE (AFTER THE POINT)
D007E060          002 * 12 010 * 20      JUMP IF UNDERLINE, JUMP IF OTHER
D007E062          010 E 30               ERROR IF ANOTHER PERIOD
D007E064    *  12 020 $   0 010 * 85     ADVANCE
D007E066          002 * 12 010 * 20      LOOP IF UNDERLINE, JUMP IF OTHER
D007E068          020 P 12 050 P 30      SAVE POINTER IF PERIOD
D007E070          020 $   0 010 * 85     ADVANCE TO CHECK NEXT CHARACTER
D007E072          002 E 30 010 * 17      ERROR IF UNDERLINE, JUMP IF OTHER
D007E074          010 E 30               ERROR IF PERIOD
D007E100    *  20 020 P 12 050 P 30      SAVE POINTER FOR END OF FIELD
D007E101    *  17 020 Q   6 001 E 24     ERROR IF NO ENTRY FOR THE FIELD
D007E102          020 Q   7 002 * 22     JUMP IF NOT VACUOUS
D007E104          022 Q   7 002 E 25     ERROR IF NOT SIMPLY -1 VACUOUS FLAG
D007E106          020 H 21 002 * 19
D007E108          020 H 22 010 * 19
D007E112    *  18 020 A 14 050 ----
D007E114    *  21 020 P 12 024 K  3
D007E116    *  19 050 P 12 056 * 18
D007E118          025 P 30 001 * 18      LOOP TILL FIELD FILLED WITH SPACES
D007E120          010 * 56               JUMP TO POP Q7 AND CARRY ON
D007E122    *  22 020 H 23 002 * 60      JUMP IF SCIENTIFIC NOTATION
D007E124          120    0 050 P 18      CLEAR W,D TO ZERO
D007E126          050 P 19 020 H 21
D007E128          001 * 24 020 H 22      JUMP IF NO WHOLE NUMBER
D007E130          002 * 23 020 P 30      JUMP IF THERE WAS A POINT
D007E132    *  23 025 H 21 050 P 18      SET W
D007E134          020 H 22 001 * 25      JUMP IF NO POINT
D007E136    *  24 020 P 30 025 K  3
D007E138          025 H 22 050 P 19      SET D
D007E140    *  25 020 Q   7 124 K 11
D007E142          071    1 072   11
D007E144          050 P 27 072   21
```

```
D007E146          024 P 27 050 P 27     SET S (+ OR -) IN P27
D007E148          020 Q  7 124 K 13
D007E150          050 H 26 020 * 98
D007E152          025 P 27 025 P 19     (NOTE MQ IS CLEAR FOR FOLLOWING)
D007E154          002 $  1 120    0     JUMP IF DIGITS TO BE ROUNDED
D007E156          025 * 95 005 $  1     JUMP IF LESS THAN 10 TO BE ROUNDED
D007E158          120    0 024 * 94
D007E160          052 * 26 056 * 27
D007E162    * 26  020 ---- 072    1
D007E164          024 H 26 050 T  0
D007E166    * 27  072   39 044 ----     ROUND
D007E168    * 28  060 H 26 050 P 28
D007E170          050 P 29 023 H 26
D007E171          001 * 34 050 P 27     JUMP IF NONZERO AFTER ROUNDING
D007E172          050 H 24 020 H 21
D007E173          002 * 36 010 * 50     JUMP IF WHOLE PART ELSE FOR FRACTION
D007E174    * 34  020 T  0 025 N  9
D007E176          001 * 30 020 P 27     JUMP UNLESS ROUNDED TO EXTRA DIGIT
D007E178          024 K  3 050 P 27     INCREASE S BY ONE
D007E180    * 30  020 Q  7 124 K 12
D007E182          071    9 006 * 31
D007E186    * 31  020 K  3 050 H 24     SET Z A/C SIGN OF NUMBER
D007E192    * 36  021 P 27 025 K  3
D007E194    * 39  001 * 32 120    0     JUMP IF S NONNEGATIVE
D007E196    * 32  025 H 24 024 P 18
D007E198          001 E 31 024 H 21     ERROR IF NUMBER WON'T FIT FIELD
D007E200          001 * 50 050 H 25     JUMP IF NO WHOLE PART TO FIELD
D007E202          020 H 21 025 K  3
D007E204          050 P 12 014 * 35
D007E206    * 33  020 $  0 010 = 62     INSERT SPACE
D007E208    * 35  020 P 12 024 K  3
D007E210          025 H 25 001 * 33     LOOP UNTIL ALL SPACES ARE IN
D007E212    * 37  021 H 24 002 * 40     JUMP IF NUMBER NON NEGATIVE (Z=0)
D007E214          020 A 44 050 P 11
D007E216          020 $  0 010 = 66     INSERT MINUS SIGN
D007E218    * 40  020 P 27 001 * 50     JUMP IF S NEGATIVE
D007E220          025 * 98 005 * 41     JUMP IF S LESS THAN 8
D007E222          120    0 050 P 29     SET FRACTION TO ZERO
D007E224    * 41  120    0 024 * 98
D007E226          024 K  3 056 * 45     SET TO CONVERT S+1(LIMIT 9) WHOLE DIG
D007E228          020 * 98 025 P 27
D007E230          001 * 44 025 P 19     JUMP IF S EXCEEDS 8
D007E232          001 * 42 120    0
D007E234    * 42  024 P 19 024 J 11
D007E236          056 * 43 120    0
D007E238    * 43  004 H 26 044 ----     SPLIT OFF FRACTIONAL DIGITS
D007E240          050 P 29 060 P 28
D007E242    * 44  020 P 28 050 H 20
D007E244          020 $  0 010 = 36     CONVERT WHOLE NUMBER DIGITS
D007E245    * 45          3     ----
D007E246          023 H 26 001 * 47     JUMP IF NONZERO OUTPUT
D007E247          023 P 19 002 * 47     JUMP IF NO FRACTION FIELD
D007E248          020 P 12 056 * 48
D007E249    * 48  020 A 14 050 ----     REPLACE ZERO BY SPACE
D007E250    * 47  020 H 22 001 * 54     JUMP IF NO POINT
```

```
D007E251          050 H 25 010 $  1
D007E252          020 $  0 010 * 80     FILL WITH ZEROS TO POINT
D007E256    * 50  020 H 22 050 P 12     SET POINTER AT DECIMAL POINT
D007E258          023 P 29 002 * 54     JUMP IF ZERO FRACTION
D007E260          020 P 27 024 K  3
D007E262          002 * 51 020 H 22
D007E264          025 P 27 050 H 25
D007E266          020 $  0 010 * 80     FILL LEAD ZEROS OF FRACTION FIELD
D007E268    * 51  020 P 27 002 * 58     JUMP IF S NONNEGATIVE
D007E270          024 P 19 025 * 98
D007E271          001 * 52 120    0     JUMP IF D+S EXCEEDS 8, ELSE SET 9 DIG
D007E272    * 52  024 * 99 014 * 57     SET D+S+1 DIGITS
D007E273    * 58  020 * 98 025 P 27
D007E274          025 P 19 002 * 57     JUMP IF 8-S EXCEEDS D
D007E275          024 P 19 014 * 57     SET 8-S DIGITS
D007E276    * 57  020 P 19 056 * 53     SET D DIGITS
D007E278          020 P 29 050 H 20
D007E280          020 $  0 010 = 36     CONVERT FRACTION DIGITS
D007E282    * 53        3      ----
D007E284    * 54  020 P 30 050 H 25
D007E286          020 $  0 010 * 80     FILL TRAILING ZEROS
D007E288    * 55  020 $  0 010 = 17     POP Q  7
D007E290    * 56  020 $  0 010 = 17     POP Q  7
D007E292          020 P 30 056 * 59
D007E294    * 59  050 P 12 020 ----     SET P12
D007E296          001 *  6 010 *  1     JUMP IF END OF FORM, ELSE LOOP BACK
D007E300    * 60  050 P 12 056 * 64     SCIENTIFIC NOTATION
D007E302          023 Q  7 001 * 62     JUMP IF NONZERO
D007E304          020 P 12 056 * 61     ZERO TREATMENT
D007E306    * 61  020 A 14 050 ----     OUTPUT SPACE
D007E308          020 A 48 050 P 11
D007E310          020 $  0 010 = 66     OUTPUT ZERO
D007E312          020 $  0 010 = 17     POP Q7
D007E314          010 * 21              JUMP TO FILL WITH SPACES ETC
D007E316    * 62  020 Q  7 124 K 12
D007E318          025 K 12 002 * 64
D007E320    * 63  020 A 14 014 * 64
D007E322    * 64  020 A 44 050 ----
D007E323          020 P 30 025 P 12
D007E324          025 * 96 025 * 98
D007E326          001 * 65 120    0
D007E328    * 65  024 * 98 056 * 70
D007E330          050 T  0 010 $  1
D007E332          020 J 14 025 T  0
D007E334          056 * 67 052 * 69
D007E336    * 66  020 Q  7 124 K 13
D007E338    * 67  071    1 024 ----
D007E340          070    1 025 N  9
D007E342          001 * 68 020 Q  7     JUMP UNLESS ROUNDS TO EXTRA DIGIT
D007E344          024 K 14 124 K 11
D007E346          024 N  8 050 Q  7
D007E348          071    1 025 * 93
D007E350          001 * 66 010 E 31     ERROR IF OVERFLOW ON EXPONENT
D007E352    * 68  024 N  9 071    40
D007E354          044 N  8 050 P 29
```

```
D007E356          060 H 20 010 $  1
D007E358          020 $  0 010 = 36     OUTPUT FIRST DIGIT
D007E360                3          1
D007E362          020 P 12 024 K  3
D007E364          050 P 12 010 $  1     SKIP OVER DECIMAL POINT
D007E366          120    0 004 P 29
D007E368     * 69 044 ---- 060 H 20
D007E370          020 $  0 010 = 36     OUTPUT FRACTION
D007E372     * 70          3      ----
D007E374          020 P 30 025 K 43
D007E376          050 H 25 010 $  1
D007E378          020 $  0 010 * 80     FILL TRAILING ZEROS OF DIGITS FIELD
D007E380          020 Q  7 071    1
D007E382          072   32 050 H 20
D007E384          002 * 71 010 * 72
D007E386     * 71 020 A 14 014 * 73
D007E388     * 72 021 H 20 050 H 20
D007E390     * 73 020 A 44 050 P 11
D007E391          020 $  0 010 = 66     OUTPUT SIGN OF EXPONENT PART
D007E392          020 $  0 010 = 36     OUTPUT EXPONENT
D007E394                3          2
D007E396          010 * 55              JUMP TO POP Q7 TWICE AND LOOP
D007E400     * 80 024 K  2 052 * 82     SUBROUTINE TO FILL ZEROS UP TO (T2)
D007E402     * 81 020 P 12 024 K  3       LEAVES P12 AT (T2)-1
D007E404          025 H 25 004 A 48
D007E406     * 82 002 ---- 024 H 25
D007E408          054 P 12 052 * 83
D007E410     * 83 060 ---- 010 * 81
D007E450     * 85 024 K  2 052 * 89     LOCAL SUBROUTINE TO ADVANCE ONE IN FO
D007E452          024 K  2 052 * 87       EXIT TO $1 WITH ACC + IF UNDERLINE
D007E454          020 P 12 024 K  3       EXIT TO $1 WITH ACC - IF OTHER
D007E456          054 P 12 052 * 86       EXIT TO $2 IF PERIOD
D007E458     * 86 020 ---- 050 P 11
D007E460          124 A 27 025 A 27
D007E462     * 87 002 ---- 020 P 11
D007E464          124 A108 025 A108
D007E466     * 89 010 ----
D007E500     * 93+          100    7     OVERFLOW TEST VALUE
D007E502     * 94       N 10      N 10
D007E504     * 95         10        10
D007E506     * 96          6         6
D007E508     * 97          7         7
D007E510     * 98          8         8
D007E512     * 99          9         9,
D007E999        /      D  7 010,7000/
```

```
D008E000+       $                       M   0       EXTENSION OF G6
D008E010            004 H 19 010 $   1
D008E015            063 T  0 002 *  30  JUMP IF H19=0=QUOTES
D008E020            020 W  9 065 T   0
D008E030            023 T  0 002 *  10  JUMP IF 'STEP'
D008E040            020 W  7 065 T   0
D008E050            023 T  0 002 *   5  JUMP IF 'PART'
D008E060            020 W  3 065 T   0
D008E070            023 T  0 002 *  15  JUMP IF 'FORM'
D008E080            020 W 13 065 T   0
D008E090            023 T  0 002 *  20  JUMP IF 'SIZE'
D008E100            020 W 11 065 T   0
D008E110            023 T  0 002 *  40  JUMP IF 'USERS'
D008E112            020 W  1 065 T   0
D008E114            023 T  0 002 *  50  JUMP IF 'TIME'
D008E120            130 $  0 130 $   0  ELSE MACHINE ERROR
D008E130      *   5 020 $  0 010 =  39  VERIFY SPACE AND ADVANCE TO NONSPACE
D008E140            020 $  0 010 =  21  EVALUATE PART NUMBER
D008E150            020 P  1 124 K  19  VERIFY TERMINAL
D008E160            025 K 19 001 E   6
D008E170            020 $  0 010 =  44  VALIDATE PART NUMBER
D008E180            020 Q  3 050 P  20  SET PART CONTEXT
D008E190            020 $  0 010 =  13  POP Q3
D008E200            020 $  0 010 =  50  FIND PART
D008E210            002 *  8 010 E  17  ERROR IF CAN'T
D008E220      *   8 020 H 23 050 P  19
D008E230      *   6 020 P 19 052 *   7
D008E240      *   7 020 ---- 001 X   5  TO ADVANCE WHEN DONE
D008E250            050 P 19 056 P   3
D008E260            020 $  0 014 =   7  UNPACK TO S
D008E280            020 P 15 002 X  13  JUMP IF INTERRUPT
D008E290            020 $  0 010 =  78  TRANSMIT
D008E300            010 *  6
D008E310      *  10 020 $  0 010 =  39  VERIFY SPACE AND ADVANCE TO NONSPACE
D008E320            020 $  0 010 =  21  EVALUATE STEP NUMBER
D008E330            020 P  1 124 K  19  VERIFY TERMINAL
D008E340            025 K 19 001 E   6
D008E350            020 $  0 010 =  45  VALIDATE STEP NUMBER
D008E360            020 Q  3 050 P  21  SET STEP CONTEXT
D008E370            020 $  0 010 =  13  POP Q3
D008E380            020 $  0 010 =  51  FIND STEP
D008E390            002 * 11 010 E  16  ERROR IF CAN'T
D008E400      *  11 020 H 26 056 P   3
D008E410            020 $  0 014 =   7  UNPACK TO S
D008E430            010 X 14            EXIT VIA X14
D008E440      *  15 020 $  0 010 =  39  VERIFY SPACE AND ADVANCE
D008E450            020 $  0 010 =  21  EVALUATE FORM NUMBER
D008E460            020 P  1 124 K  19  VERIFY TERMINAL
D008E470            025 K 19 001 E   6
D008E480            020 $  0 010 =  46  VALIDATE FORM NUMBER
D008E490            020 Q  3 050 P  22  SET FORM CONTEXT
D008E500            020 $  0 010 =  13  POP Q3
D008E510            020 $  0 010 =  52  FIND FORM
D008E520            002 * 11 010 E  19  ERROR IF CAN'T
```

```
D008E530     * 20 020 $  0 010 =  3     ELIMINATE SPACES
D008E540          020 P  1 124 K 19
D008E550          025 K 19 001 E  6     VERIFY TERMINAL
D008E560          120    0 050 T  0     COMPUTE SIZE
D008E570          020 * 22 056 * 21
D008E580     * 21          020 ----
D008E590          001 * 23 056 * 21
D008E600          020 T  0 024 I  1
D008E610          050 T  0 014 * 21
D008E620     * 22               Q  0
D008E630     * 23 020 T  0 050 H 20
D008E640          020 S  0 050 P 12
D008E650          020 $  0 010 = 36     CONVERT SIZE
D008E660               2          3
D008E670          020 A 42 050 S  4
D008E680          010 X 14              EXIT VIA X14
D008E740     * 30 020 S  0 050 P 12     SET OUTPUT FOR QUOTED MESSAGE
D008E750     * 31 020 $  0 010 =  1     ADVANCE
D008E760          001 E  6 124 A 66
D008E770          025 A 66 002 * 32     JUMP IF QUOTES
D008E780     * 33 020 P 12 024 K  3     INSERT CHARACTER IN S
D008E785          050 P 12 056 * 35
D008E790     * 35 020 P  1 050 ----
D008E795          020 * 31 010 =  1     ADVANCE AND LOOP
D008E800     * 32 020 P  2 050 * 90     SAVE P2
D008E810          020 $  0 010 =  2     ADVANCE TO NONSPACE
D008E820          020 * 90 050 P  2     RESTORE P2
D008E830          020 P  1 124 K 19
D008E840          025 K 19 002 * 34     JUMP IF TERMINAL
D008E850          020 A 66 050 P  1
D008E860          010 * 33
D008E870     * 34 020 A 42 050 P 11     INSERT CR+EOM
D008E880          020 $  0 010 = 66
D008E890          010 X 14              EXIT VIA X14
D008E900     * 40 020 $  0 010 =  3     ELIMINATE SPACES
D008E905          020 P  1 124 K 19
D008E910          025 K 19 001 E  6     VERIFY TERMINAL
D008E915          120    0 050 S  1     INITIALIZE COUNT
D008E920          020 U  2 010 * 42     INITIALIZE
D008E925     * 41 020 * 90 024 K  3
D008E930     * 42 050 * 90 052 * 43
D008E935          025 U  3 002 * 44     JUMP AT END OF TABLE
D008E940     * 43 020 ---- 001 * 41
D008E945          020 S  1 024 I  1     COUNT ONE ACTIVE USER
D008E950          050 S  1 010 * 41
D008E955     * 44 020 A 42 050 S  2     SET CR+EOM
D008E960          010 X 14              EXIT VIA X14
D008E964     * 50 020 $  0 010 =  3     ELIMINATE SPACES
D008E965          020 P  1 124 K 19
D008E966          025 K 19 001 E  6     VERIFY TERMINAL
D008E967          020 S  0 050 P 12
D008E970          020 $  0 010 = 61     CONVERT TIME
D008E975          020 A 42 050 S  5
D008E980          010 X 14              EXIT VIA X14
D008E990     * 90                       WORKING STORAGE
```

```
D008E999          /      D008 010,7000/
```

```
E---E000+     E  0 020 E  0 010 *  0     OUT OF SPACE
E---E001      E  1 020 E  1 010 *  0     OVERFLOW
E---E002      E  2 020 E  2 010 *  0     UNDEFINED
E---E003      E  3 020 E  3 010 *  0     DIVISION BY ZERO
E---E004      E  4 020 E  4 010 *  0     ILLEGAL INDEX
E---E005      E  5 020 E  5 010 *  0     NUMERAL HAS MORE THAN NINE SIGNIF DIGI
E---E006      E  6 020 E  6 010 *  0     EH
E---E007      E  7 020 E  7 010 *  0     'GO' BUT NO TASK
E---E008      E  8 020 E  8 010 *  0     TOO MANY VALUES FOR FORM
E---E009      E  9 020 E  9 010 *  0     ILLEGAL INDIRECT
E---E010      E 10 020 E 10 010 *  0     ILLEGAL DIRECT
E---E011      E 11 020 E 11 010 *  0     ILLEGAL STEP LABEL
E---E012      E 12 020 E 12 010 *  0     ILLEGAL FORM NUMBER
E---E013      E 13 020 E 13 010 *  0     ILLEGAL STEP NUMBER
E---E014      E 14 020 E 14 010 *  0     ILLEGAL ARG FOR SQRT
E---E015      E 15 020 E 15 010 *  0     FIELD HAS FEWER THAN 7 PERIODS
E---E016      E 16 020 E 16 010 *  0     NO SUCH STEP
E---E017      E 17 020 E 17 010 *  0     NO SUCH PART
E---E018      E 18 020 E 18 010 *  0     ILLEGAL PART NUMBER
E---E019      E 19 020 E 19 010 *  0     NO SUCH FORM
E---E020      E 20 020 E 20 010 *  0     ILLEGAL ARG FOR LOG
E---E021      E 21 020 E 21 010 *  0     NEGATIVE TO NON INTEGER POWER
E---E022      E 22 020 E 22 010 *  0     ILLEGAL ARG FOR SIN OR COS
E---E023      E 23 020 E 23 010 *  0     ZERO TO NEGATIVE POWER
E---E024      E 24 020 E 24 010 *  0     TOO FEW VALUES FOR FORM
E---E025      E 25 020 E 25 010 *  0     INDIVIDUAL VALUES ARE REQUIRED FOR FOR
E---E026      E 26 020 E 26 010 *  0     CAN'T FIND PART FOR ITERATION
E---E027      E 27 020 E 27 010 *  0     CAN'T FIND STEP FOR ITERATION
E---E028      E 28 020 E 28 010 *  0     ILLEGAL SET OF VALUES FOR ITERATION
E---E029      E 29
E---E030      E 30 020 E 30 010 *  0     FIELDS RUN TOGETHER
E---E031      E 31 020 E 31 010 *  0     CAN'T EXPRESS VALUE IN GIVEN FORM
E---E040      *  0 050 P 18 010 D  0,
```

```
E---E100      *  0      * 50 012 * 51
E---E101      *  1      * 51 012 * 52
E---E102      *  2      * 52 012 * 53
E---E103      *  3      * 53 012 * 54
E---E104      *  4      * 54 012 * 55
E---E105      *  5      * 55 012 * 56
E---E106      *  6      * 56 012 * 57
E---E107      *  7      * 57 012 * 58
E---E108      *  8      * 58 012 * 59
E---E109      *  9      * 59 012 * 60
E---E110      * 10      * 60 012 * 61
E---E111      * 11      * 61 012 * 62
E---E112      * 12      * 62 012 * 63
E---E113      * 13      * 63 012 * 64
E---E114      * 14      * 64 012 * 65
E---E115      * 15      * 65 012 * 66
E---E116      * 16      * 66 012 * 67
E---E117      * 17      * 67 012 * 68
E---E118      * 18      * 68 012 * 69
E---E119      * 19      * 69 012 * 70
E---E120      * 20      * 70 012 * 71
E---E121      * 21      * 71 012 * 72
E---E122      * 22      * 72 012 * 73
E---E123      * 23      * 73 012 * 74
E---E124      * 24      * 74 012 * 75
E---E125      * 25      * 75 012 * 76
E---E126      * 26      * 76 012 * 77
E---E127      * 27      * 77 012 * 78
E---E128      * 28      * 78 012 * 79
E---E129      * 29      * 79 012 * 80
E---E130      * 30      * 80 012 * 81
E---E131      * 31      * 81 012 * 82
E---E500      * 50 054,4161,121,2425     'I NEED MORE STORAGE SPACE.'
E---E501           012,0161,102,3051
E---E502           012,4161,443,1446
E---E503           024,4210,561,2416
E---E504           031,0470,421,1425
E---E505           015,4524,000,0000
E---E510      * 51 054,4160,601,0465     'I HAVE AN OVERFLOW.'
E---E511           012,4160,422,2416
E---E512           023,0650,522,4426
E---E513           021,4461,541,5452
E---E514           100,0000,000,0000
E---E520      * 52 007,0311,440,7064     ' IS UNDEFINED.'
E---E521           022,4240,521,3031
E---E522           022,4250,501,5452
E---E523           100,0000,000,0000
E---E530      * 53 054,4160,601,0465     'I HAVE A ZERO DIVISOR.'
E---E531           012,4160,420,7071
E---E532           012,4511,140,7024
E---E533           014,4650,623,1046
E---E534           024,4331,250,0000
E---E540      * 54 054,4450,501,2467     'INDEX VALUE MUST BE INTEGER AND
E---E541           007,0650,422,1464       0-INDEX-99.'
```

```
E---E542          012,4161,103,2062
E---E543          031,4160,441,2416
E---E544          014,4451,461,2427
E---E545          012,4510,341,0445
E---E546          012,0161,404,2431
E---E547          022,4240,523,3505
E---E548          004,4110,662,5200
E---E550      * 55 063,4430,521,0462    'PLEASE LIMIT NUMBERS TO 9 SIGNIFICAN
E---E551          012,4161,061,4444    DIGITS.'
E---E552          014,4630,342,2464
E---E553          022,0220,522,4462
E---E554          007,0631,140,7011
E---E555          007,0620,621,3445
E---E556          014,4260,621,1421
E---E557          022,4630,341,2031
E---E558          013,4311,463,1033
E---E559          025,2000,000,0000
E---E560      * 56 052,4303,422,5200    'EH.'
E---E570      * 57 074,0461,500,7030    'YOU HAVEN'T TOLD ME TO DO
E---E571          010,4650,522,2501      ANYTHING YET.'
E---E572          031,4161,462,3043
E---E573          012,0161,101,2416
E---E574          031,4460,341,2046
E---E575          007,0211,123,4063
E---E576          014,0311,121,3416
E---E577          034,0251,461,5452
E---E578          100,0000,000,0000
E---E580      * 58 054,4160,601,0465    'I HAVE TOO MANY VALUES FOR THE FORM.
E---E581          012,4161,462,3046
E---E582          007,0440,422,2470
E---E583          007,0650,422,1464
E---E584          012,4620,341,3046
E---E585          024,4161,461,4025
E---E586          007,0261,142,4444
E---E587          015,4524,000,0000
E---E590      * 59 052,0461,124,0463    'DON'T GIVE THIS COMMAND INDIRECTLY.'
E---E591          007,0270,623,2425
E---E592          007,0630,601,4462
E---E593          007,0231,142,2044
E---E594          010,4450,500,7031
E---E595          022,4240,622,4425
E---E596          011,4631,063,4033
E---E597          025,2000,000,0000
E---E600      * 60 052,0461,124,0463    'DON'T GIVE THIS COMMAND DIRECTLY.'
E---E601          007,0270,623,2425
E---E602          007,0630,601,4462
E---E603          007,0231,142,2044
E---E604          010,4450,500,7024
E---E605          014,4510,521,1463
E---E606          021,4700,662,5200
E---E610      * 61 063,4430,521,0462    'PLEASE LIMIT STEP LABELS TO 9 DIGITS
E---E611          012,4161,061,4444
E---E612          014,4630,343,1063
E---E613          012,4470,342,1421
E---E614          011,0251,063,1016
```

```
E---E615          031,4460,340,4416
E---E616          012,0310,561,4463
E---E617          031,0331,250,0000
E---E620      * 62 053,0461,222,2016     'FORM NUMBER MUST BE INTEGER AND
E---E621          022,4641,101,1025       0-FORM-10*9.'
E---E622          024,4161,103,2062
E---E623          031,4160,441,2416
E---E624          014,4451,461,2427
E---E625          012,4510,341,0445
E---E626          012,0161,404,3426
E---E627          023,0511,104,3401
E---E628          030,1740,221,5452
E---E629          100,0000,000,0000
E---E630      * 63 071,0630,522,3416     'STEP NUMBER MUST SATISFY 1-STEP-10*9
E---E631          022,4641,101,1025
E---E632          024,4161,103,2062
E---E633          031,4161,441,0463
E---E634          014,4620,543,4016
E---E635          000,5051,443,1425
E---E636          023,5070,023,0174
E---E637          004,4331,250,0000
E---E640      * 64 054,4160,601,0465     'I HAVE A NEGATIVE ARGUMENT FOR SQRT.
E---E641          012,4160,420,7045
E---E642          012,4270,423,1431
E---E643          032,4250,341,0451
E---E644          013,4641,101,2445
E---E645          031,4160,542,3051
E---E646          007,0621,202,4463
E---E647          015,4524,000,0000
E---E650      * 65 053,0310,522,1424     'FIELD DEFINITION HAS FEWER THAN
E---E651          007,0240,521,3031     7 PERIODS.'
E---E652          022,4311,461,4446
E---E653          022,4160,601,0462
E---E654          007,0260,523,3025
E---E655          024,4161,461,4021
E---E656          022,4160,160,7047
E---E657          012,4510,622,3024
E---E658          031,0331,250,0000
E---E660      * 66 054,4160,461,0445     'I CAN'T FIND THE REQUIRED STEP.'
E---E661          040,4630,341,3031
E---E662          022,4240,343,1430
E---E663          012,4161,221,2450
E---E664          032,0311,221,2424
E---E665          007,0621,461,2447
E---E666          015,4524,000,0000
E---E670      * 67 054,4160,461,0445     'I CAN'T FIND THE REQUIRED PART.'
E---E671          040,4630,341,3031
E---E672          022,4240,343,1430
E---E673          012,4161,221,2450
E---E674          032,0311,221,2424
E---E675          007,0470,422,4463
E---E676          015,4524,000,0000
E---E680      * 68 063,4211,223,1416     'PART NUMBER MUST BE INTEGER AND
E---E681          022,4641,101,1025       0-PART-10*9.'
E---E682          024,4161,103,2062
```

```
E---E683          031,4160,441,2416
E---E684          014,4451,461,2427
E---E685          012,4510,341,0445
E---E686          012,0161,404,3447
E---E687          010,4511,464,3401
E---E688          030,1740,221,5452
E---E689          100,0000,000,0000
E---E690     * 69 054,4160,461,0445     'I CAN'T FIND THE REQUIRED FORM.'
E---E691          040,4630,341,3031
E---E692          022,4240,343,1430
E---E693          012,4161,221,2450
E---E694          032,0311,221,2424
E---E695          007,0261,142,4444
E---E696          015,4524,000,0000
E---E700     * 70 054,4160,601,0465     'I HAVE AN ARGUMENT LTEO FOR LOG.'
E---E701          012,4160,422,2416
E---E702          010,4510,563,2044
E---E703          012,4451,460,7105
E---E704          007,0600,341,3046
E---E705          024,4161,062,3027
E---E706          015,4524,000,0000
E---E710     * 71 054,4160,601,0465     'I HAVE A NEGATIVE BASE TO A FRACTION
E---E711          012,4160,420,7045        POWER.'
E---E712          012,4270,423,1431
E---E713          032,4250,341,1021
E---E714          031,0250,343,1446
E---E715          007,0210,341,3051
E---E716          010,4231,461,4446
E---E717          022,4211,060,7047
E---E718          023,0660,522,4433
E---E719          025,2000,000,0000
E---E720     * 72 063,4430,521,0462     'PLEASE KEEP 1X1 LT 100 FOR SIN(X)
E---E721          012,4161,041,2425        OR COS(X).'
E---E722          023,4163,003,3540
E---E723          043,4011,403,0016
E---E724          013,0461,220,7062
E---E725          014,4452,223,3560
E---E726          007,0461,220,7023
E---E727          023,0622,223,3560
E---E728          015,4524,000,0000
E---E730     * 73 054,4160,601,0465     'I HAVE ZERO TO A NEGATIVE POWER.'
E---E731          012,4161,621,2451
E---E732          023,0161,462,3016
E---E733          010,4161,121,2427
E---E734          010,4630,623,2425
E---E735          007,0471,143,3025
E---E736          024,4331,250,0000
E---E740     * 74 054,4160,601,0465     'I HAVE TOO FEW VALUES FOR THE FORM.'
E---E741          012,4161,462,3046
E---E742          007,0260,523,3016
E---E743          032,4211,063,2025
E---E744          031,0160,542,3051
E---E745          007,0630,601,2416
E---E746          013,0461,222,2033
E---E747          025,2000,000,0000
```

```
E---E750     * 75 054,4161,121,2425     'I NEED INDIVIDUAL VALUES FOR A FORM.
E---E751          012,0160,622,2424
E---E752          014,4650,621,2064
E---E753          010,4430,343,2421
E---E754          021,4640,523,1016
E---E755          013,0461,220,7021
E---E756          007,0261,142,4444
E---E757          015,4524,000,0000
E---E760     * 76 054,4160,461,0445     'I CAN'T FIND THE PART FOR ITERATION.
E---E761          040,4630,341,3031
E---E762          022,4240,343,1430
E---E763          012,4161,161,0451
E---E764          031,4160,542,3051
E---E765          007,0311,461,2451
E---E766          010,4630,622,3045
E---E767          015,4524,000,0000
E---E770     * 77 054,4160,461,0445     'I CAN'T FIND THE STEP FOR ITERATION.
E---E771          040,4630,341,3031
E---E772          022,4240,343,1430
E---E773          012,4161,443,1425
E---E774          023,4160,542,3051
E---E775          007,0311,461,2451
E---E776          010,4630,622,3045
E---E777          015,4524,000,0000
E---E780     * 78 054,4431,061,2427     'ILLEGAL SET OF VALUES FOR ITERATION.
E---E781          010,4430,343,1025
E---E782          031,4161,141,3016
E---E783          032,4211,063,2025
E---E784          031,0160,542,3051
E---E785          007,0311,461,2451
E---E786          010,4630,622,3045
E---E787          015,4524,000,0000
E---E790     * 79
E---E800     * 80 074,0461,502,4416     'YOUR FIELDS RUN TOGETHER IN THE FORM
E---E801          013,0310,522,1424
E---E802          031,0161,223,2045
E---E803          007,0631,141,3425
E---E804          031,4300,522,4416
E---E805          014,4450,343,1430
E---E806          012,4160,542,3051
E---E807          022,0331,250,0000
E---E810     * 81 054,4160,461,0445     'I CAN'T EXPRESS VALUE IN YOUR FORM.'
E---E811          040,4630,341,2467
E---E812          023,4510,523,1062
E---E813          007,0650,422,1464
E---E814          012,4160,622,2416
E---E815          034,0461,502,4416
E---E816          013,0461,222,2033
E---E817          025,2000,000,0000
E---E820     * 82                    ,
```

```
F000E000+     F  0 010 $  0                    Y = SQRT(X)
F000E010          024 K  2 052 *  9           SET EXIT
F000E015          020 Q  6 002 E  6           MALFORMED IF AUX NOT EMPTY
F000E020          023 Q  3 002 *  9           TRANSFER IF X=0
F000E030          020 $  0 010 =  11          UNPACK X INTO P4, P5, P6
F000E040          023 P  5 001 E  14          TRANSFER IF X NEGATIVE
F000E050          020 P  4 071    8           COMPUTE 1/2 SFX
F000E060          050 T  1 060 T  2           STORE IN P4
F000E070          071    32 072   1
F000E080          050 P  4 020 T  1           TEST FOR ODD OR EVEN
F000E090          004 P  6 002 *  1           IF ODD 10.P6 TO TO
F000E100          032 N  1 060 T  0           AND 10E9 TO P6 AS Y(0)
F000E110          020 N  9 010 *  2
F000E120      *   1 060 T  0 020 *  90        SFX EVEN, X TO TO
F000E130      *   2 050 P  6 004 T  0         AND SQRT(10) + EPSILON TO P6
F000E140          032 N  8 044 P  6           T1=Y(N)-(10E8.X/Y(N))=DELTA
F000E150          020 P  6 065 T  1
F000E160          020 P  6 073    1           P6=Y(N+1)=(2Y-DELTA)/2
F000E170          025 T  1 072    1
F000E180          050 P  6 020 T  1           IF DELTA GREATER THAN OR EQUAL TO
F000E190          025 N  4 006 *  2           10E4, ITERATE
F000E200          020 P  6 004 T  0
F000E210          037 N  8 044 P  6           IF (Y(N+1) + (Y(N+1)-A.10E8)/Y(N+1)))
F000E220          020 P  6 064 T  2           IS LESS THAN ZERO, ADD ONE TO
F000E230          002 *  3 020 P  6           Y(N+1)
F000E240          024 K  1 050 P  6
F000E250      *   3 020 $  0 010 =  30        PACK RESULT
F000E260      *   9 010 ----                  EXIT
F000E270      *  90+   316227800   39,        SQRT(10) + EPSILON
```

```
F001E000+    F  1 010 $  0                    Y=LOG(X)---BASE E
F001E010         024 K  2 052 *  9            SET EXIT
F001E015         020 Q  6 002 E  6            MALFORMED IF AUX NOT EMPTY
F001E020         023 Q  3 002 E 20            X=0 IS AN ERROR
F001E030         020 $  0 010 = 11            UNPACK
F001E040         023 P  5 001 E 20            X NEGATIVE IS AN ERROR
F001E050         020 P  4 072   31            SFX AS AN INTEGER
F001E060         050 P  4 020 J 11            INITIALIZE ADDRESS
F001E061         024 K002 010 * 21
F001E070    * 21 052 *  6 120
F001E080    *  1 050 T  1 020 P  6            SET Q=0
F001E090         025 * 91 002 *  2            IF COEF(X) LESS THAN LOWER BOUND,
F001E100         020 P  6 073    1            2 COEF(X) REPLACES IT AND
F001E110         050 P  6 020 T  1            Q+1 REPLACES Q
F001E120         024 I  1 010 *  1            COMPUTE Z FOR SERIES AS EQUAL TO
F001E130    *  2 020 P  6 024 * 92            (COEF(X)-(10*8)(2*3) /
F001E140         050 T  2 020 P  6            (COEF(X)+(10*8)(2*3)
F001E150         025 * 92 040 T  2
F001E160         060 T  2 030 T  2            Z IN T2, Z*2 IN T3
F001E170         050 T  3 120                 COMPUTE 1/2.LOG(X)=Z+1/3(Z*3)+
F001E180         050 T  0 020 * 95            1/5(Z*5)+...+1/13(Z*13)
F001E190    *  3 050 T  4 020 I  1            T4=DELTA=13
F001E200         040 T  4 020 T  0            T0=LOG(X)=0, INITIALLY
F001E210         064 T  0 004 T  0            T0=LOG(X)+(1/DELTA)
F001E220         030 T  3 050 T  0            T0=(LOG(X)+(1/DELTA)).Z*2
F001E230         020 T  4 025 I  4            TRANSFER IF DELTA WAS 3
F001E240         005 *  4 024 I  2            DELTA=DELTA-2
F001E250    *  4 010 *  3 004 T  0            (Z.LOG(X))+Z
F001E260         030 T  2 024 T  2            T0=LOG(X)/4
F001E270         072    1 050 T  0
F001E280         020 I  3 025 T  1
F001E290         050 T  1 004 * 93
F001E300         032 T  1 060 T  5            T5=(3-Q).LOG(2)/4
F001E310         024 T  0 004 * 94            LOG(X)=4.(EXP(X).LOG(10)+T5)
F001E320         036 P  4 077    2            INTEGER IN ACCUMULATOR AND T0, T5
F001E330         050 T  0 050 T  5            FRACTION IN MQ AND T1
F001E340         001 * 12 060 T  1            TRANSFER IF LOG IS NEGATIVE
F001E350    *  5 023 T  0 006 * 14            ..... NORMALIZE .....
F001E360    * 18 020 I  1 050 T  3            IF T0 IS GREATER THAN 10*I, THEN
F001E370    *  6 021 ---- 024 T  0            SET I=I+1, ITERATE
F001E380         005 *  7 020 *  6
F001E390         024 K  2 052 *  6
F001E400         020 T  3 024 I  1
F001E410    *  7 014 * 18 020 T  3            IF T0 IS LESS THAN 10*I, THEN
F001E420         025 I  1 050 P  4            P4=EXP(Y)=I-1 AND THE MAGNITUDE
F001E430         020 * 97 025 T  3            /FY=(T0+(2*(-39)).T1).10*(9-I)
F001E440         056 *  8 056 * 10
F001E450    *  8 004 T  1 030 ----
F001E460    * 10 004 T  0 036 ----
F001E470         060 P006 025 N009            TEST FINAL ROUND
F001E471         001 * 22 020 N008
F001E472         050 P006 020 P004
F001E473         024 I001 010 * 11
F001E474    * 22 020 P004 010 * 11
```

```
F001E480      * 11 073    31 050 P  4      P4=SFY
F001E490           020 T  5 124 K  4      SET SIGN OF LOG
F001E500           070    9 050 P  5
F001E510           020 $  0 010 = 30      PACK RESULT
F001E520      *  9 010 ----               EXIT
F001E530      * 12 061 T  1 006 * 13
F001E540           125 K  4 050 T  1      GET ABSOLUTE VALUE OF NEGATIVE LOG
F001E550           022 T  0 025 I  1
F001E560      * 20 050 T  0 010 *  5
F001E570      * 13 050 T  1 022 T  0
F001E580      * 14 010 * 20 023 T  1      LOG IS LESS THAN ONE, TEST FOR
F001E590           002 * 19 004 T  1      LOG=ZERO
F001E600      * 15 020 I  1 050 T  3
F001E610           032 N  1 025 I  1
F001E620           002 * 16 020 T  3
F001E630           024 I  1 014 * 15
F001E640      * 16 020 T  3 025 I  1
F001E650           024 J 11 056 * 17
F001E660      * 17 004 T  1 032 ----
F001E670           032 N  9 050 P  6
F001E680           021 T  3 010 * 11
F001E690      * 19 050 Q  3 010 *  9
F001E700      * 90+   316227800    39      SQRT(10)+EPSILON
F001E710      * 91+   565685425    39      (2E3)(10E8)(SQRT(2)/2)
F001E720      * 92+   800000000    39      (2E3)(10E8)
F001E730      * 93+.17328679514    00      LOG(2)/4
F001E740      * 94+.57564627325    00      LOG(10)/4
F001E750      * 95+          13    39
F001E770      * 97     N  9     N  9,
```

```
F002E000+    F  2 010 $  0                    Y = E*X = EXP(X)
F002E010        024 K  2 052 *  9             SET EXIT
F002E015        020 Q  6 002 E  6             MALFORMED IF AUX NOT EMPTY
F002E020        023 Q  3 001 *  1             IF X=0, SET Y=1 AND EXIT
F002E030    *  3 020 N  8 050 Q  3
F002E040        010 *  9
F002E050    *  1 020 $  0 010 =  11          UNPACK NON-ZERO X
F002E060        020 P  6 040 N  9             NEW COEF(X) IS COEF(X) / 10*9
F002E070        060 T  0 020 P  4             SFX SHIFTED TO 2*-39 POSITION
F002E080        072   31 050 T  1             STORE IN T1
F002E090        002 *  2 024 I  9             IF, SFX IS LESS THAN -9, SET Y=1
F002E100        001 *  3 022 T  1
F002E105        025 I  1 010 $  1             ADJUST SFX BY ONE
F002E110        024 J 11 056 *  4
F002E120        120      050 P  4             SFX FOR RESULT
F002E130    *  4 004 T  0 044 ----
F002E140        060 T  0 070    2             COEF(X) /4 FOR SERIES
F002E150        050 T  0 014 *  5             IF SFX IS 3 OR MORE, CHECK SIGN
F002E160    *  2 025 I  3 001 *  6           OF COEF(X)
F002E170    *  8 023 P  5 002 E  1           OVERFLOW FOR POSITIVE NO.
F002E180        120      014 *  3           SET Y=0 FOR NEGATIVE NO.
F002E190    *  6 024 I  1 005 *  7           IF SFX=2, COEF(X) MUST BE LESS
F002E200        020 P  6 025 * 92             THAN LOG(10) TIMES (10*8) + 1
F002E210    *  7 002 *  8 020 T  1           IF GREATER, TEST SIGN OF COEF
F002E220        024 I  1 024 J 11
F002E230        056 * 10 004 T  0             (COEF(X)) (LOG(E)) (LESS THAN ONE)
F002E240        030 * 93 050 T  0             IN T0
F002E250    * 10 004 T  0 032 ----            T0 TIMES 10*(SFX PLUS 1)
F002E260        050 P  4 060 T  0             INTEGER PART TO SFX FOR RESULT
F002E270        072    2 044 * 93             FRACTIONAL PART /4 TO XBAR
F002E280    *  5 060 T  0 023 P  5
F002E290        006 * 11 021 T  0             XBAR FOR SERIES IN T0, USE 12 TERMS
F002E300        050 T  0 021 P  4
F002E310    * 11 050 P  4 020 * 95            SET DIVISOR FOR SERIES
F002E320        050 T  2 020 K 21             SET SERIES SUM = 1/2
F002E330    * 12 050 T  1 020 T  0
F002E340        076   39 044 T  2             COMPUTE 1/2 (E*XBAR) = Z
F002E350        030 T  1 024 K 21
F002E360        050 T  1 020 T  2
F002E370        025 I  1 050 T  2
F002E380        025 I  1 006 * 12             (Z*4) (10*10) (16) = E*X
F002E390        004 T  1 030 T  1
F002E400        050 T  1 004 T  1
F002E410        030 T  1 010 * 18
F002E420    * 18 050 T  1 004 T  1
F002E430        032 N 10 077    4
F002E440        050 P  6 004 I  3             NORMALIZE E*X
F002E450        025 N 11 002 * 13             RESULT LESS THAN 10*12
F002E460        004 I  2 020 P  6             AND GREATER THAN 10*8
F002E470        025 N 10 002 * 13
F002E480        004 I  1 020 P  6
F002E490        025 N  9 001 * 14
F002E500    * 13 020 J 11 064 T  1
F002E510        056 * 15 056 * 19
```

```
F002E520          020 P   4 010 $  1
F002E530     * 19 064 P   4 020 ----      SCALE E*X AND ROUND
F002E540          072     1 024 P  6
F002E550     * 15 072    39 044 ----
F002E560          060 P   6 025 N  9      TEST FINAL ROUND
F002E570          001 * 14 020 N  8
F002E580          050 P   6 020 P  4
F002E590          024 I   1 014 * 14
F002E600     * 14 020 P   4 025 I  2      ADJUST SFX BY 2
F002E605          073    31 050 P  4
F002E610          120       050 P  5      CLEAR SIGN
F002E620     * 17 020 $   0 010 = 30      PACK RESULT
F002E630     *  9 010 ----               EXIT
F002E650     * 92+    230258510   39      LOG(10), BASE E, TIMES 10*8 PLUS 1
F002E660     * 93+.43429448190    00      LOG(E), BASE 10
F002E680     * 95+           12   39,
```

```
F003E000+    F  3 004 K   4 010 $  0     SIN(X)--SET FLAG IN MQ=-1
F003E010        024 K   2 052 *  9     SET EXIT
F003E015        020 Q   6 002 E  6     MALFORMED IF AUX NOT EMPTY
F003E020        060 T   0 010 $  1     STORE FLAG IN TO
F003E030        020 $   0 010 = 11     UNPACK
F003E035        020 P   4 072   31     SHIFT SFX TO 2*(-39) POSITION
F003E040        050 P   4 020 T  0     FOR SINE SET RESULT = Q3 IF SFX
F003E050        002 *   1 020 P  4     IS -5 OR LESS, SHIFT P5 TO SIGN BIT
F003E060        024 I   4 001 *  9
F003E070        020 P   5 071    9
F003E080    *  1 050 P   5 020 P  4     TRANSFER IF SFX IS NEGATIVE
F003E090        005 *   2 025 I  2     NO. OUT OF RANGE IF SFX IS 2 OR MORE
F003E100        002 E 22 024 I  4     LOCATION (10*(SFX+2))
F003E110        024 J 11 056 $  1
F003E120        004 P   6 032 ----    X=X.(10*(SFX+10))
F003E130        044 * 91 050 T  1     COMPUTE X/2PI
F003E140        025 * 92 001 *  3     X=REMAINDER IF X LESS THAN PI,
F003E150        050 T   1 020 P  5     OTHERWISE X=X-PI
F003E160        024 K   4 050 P  5     CHANGE SIGN
F003E170    *  3 020 T   1 040 * 92     X/PI IN MQ
F003E180    *  2 014 *   4 024 * 93     IF SFX LESS THAN -11, SET X=0
F003E190        006 *   5 004 K  0
F003E200    *  5 014 *   4 024 J 11     LOCATION (10*(SFX+11))
F003E210        056 * 15 020 P  6
F003E220    * 15 040 N   9 032 ----    X=((X/10*9).10*(11+SFX))/PI.10*10
F003E230    *  4 044 * 92 020 K 21     X= 1/2 - X FOR COS
F003E240        065 T   1 020 T  0     TRANSFER TO SINE OR COS
F003E250        002 *   6 020 K 21     FOR SINE, X = 1/2 - ABSOLUTE VALUE
F003E260        027 T   1 050 T  1     OF T1 (1/2-X)
F003E270    *  6 004 T   1 033 T  1     -X*2
F003E280        077     2 050 T  2     -4(X*2)
F003E290        020 * 98 052 *  7     INITIALIZE SERIES FOR 1/4 SINE
F003E300    *  8 004 * 95 031 T  2
F003E310    *  7 024 ---- 050 P  6
F003E320        004 P   6 020 *  7
F003E330        024 K   2 052 *  7
F003E340        025 * 97 005 *  8
F003E350        032 T   1 050 P  6
F003E360        124 K   4 024 P  5
F003E370        070     9 050 P  5
F003E380        007 $   0 022 P  6
F003E390        073     2 003 * 10     TEST FOR OVERFLOW (RESULT=1)
F003E400        071    40 032 N  9     MULTIPLY RESULT AND SAVE BOTH PARTS
F003E410        050 P   6 060 T  2     FOR NORMALIZATION
F003E415        020 P   6 010 $  1
F003E420        025 N   9 002 * 10
F003E430        023 P   6 002 * 16     TEST FOR RESULT = 0
F003E440        021 I   9 050 P  4
F003E450    * 13 020 * 99 050 * 11     PRESET INSTRUCTION FOR NORMALIZATION
F003E460    * 11 020 P   6 025 N  8
F003E470        002 * 12 020 * 11
F003E480        025 K   1 014 * 13     NORMALIZE RESULT AND
F003E490    * 12 020 * 99 025 * 11
F003E500        050 T   1 024 J 11     ADJUST SCALE FACTOR
```

```
F003E505          056 * 17 010 $  1
F003E510          056 * 14 021 T  1
F003E520          024 I  8 024 P  4
F003E530          073   31 050 P  4
F003E535     * 17 004 T  2 030 ----
F003E540     * 14 004 P  6 036 ----
F003E542          060 P  6 025 N  9      TEST FOR ROUNDING UP TO 10*9
F003E544          001 * 16 020 N  8
F003E546          050 P  6 020 P  4
F003E550          024 K 14 050 P  4
F003E560     * 16 020 $  0 010 = 30      PACK RESULT
F003E570     *  9 010 ----               EXIT
F003E580     * 10 020 P  5 024 N  8      RESULT IS ONE
F003E590          050 Q  3 010 *  9      COMBINE WITH SIGN AND EXIT
F003E620     * 91+ 62831853072   39      2 PI . 10*10
F003E630     * 92+ 31415926536   39      PI.10*10
F003E640     * 93+           11  39      ELEVEN
F003E660     * 95 000,0000,003,5173      COEFFICIENTS A13,A11,...,A1
F003E670     * 96 177,7777,416,5664      TAKEN FROM ILLIAC SINE ROUTINE T5
F003E680          000,0025,016,6553
F003E690          177,6632,264,6777
F003E700          002,4315,361,4634
F003E710          153,2504,143,1673
F003E720          062,2077,325,0420      A1
F003E730     * 97 024 * 97 050 P  6      TEST WORD FOR END OF SERIES
F003E740     * 98      * 96      * 96
F003E750     * 99 020 P  6 025 N  8,     INITIALIZER FOR *11
```

```
F004E000+     F   4 004 K   0 014 F   3,     COS(X)--SET FLAG IN MQ=0.
```

```
F005E000+     F  5 010 $  0                      IP(X)
F005E010         024 K  2 052 *  9
F005E015         020 Q  6 002 E  6               MALFORMED IF AUX NOT EMPTY
F005E020         020 $  0 010 = 11               UNPACK (Q3)
F005E030         023 Q  3 002 *  9               ZERO RESULT FOR ZERO ARGUMENT
F005E040         020 P  4 006 *  1
F005E050         120    0 050 Q  3               ZERO RESULT IF NEGATIVE SFX
F005E060      *  1 010 *  9 072    31
F005E070         050 T  0 025 I  8
F005E080         002 *  9 020 I  8               RESULT = ARGUMENT IF SFX BIG AS 8.
F005E090         025 T  0 024 J 11
F005E095         056 *  2 010 $  1
F005E100         056 *  3 120    0
F005E110      *  2 004 P  6 044 ----              CHOP FRACTIONAL PART
F005E120      *  3 000    0 032 ----
F005E130         060 P  6 010 $  1
F005E140         020 $  0 010 = 30               PACK RESULT INTO Q3
F005E150      *  9 010 ----          ,
```

```
F006E000+     F  6 010 $  0                    FP(X)
F006E010         024 K  2 052 *  9
F006E015         020 Q  6 002 E  6             MALFORMED IF AUX NOT EMPTY
F006E020         020 $  0 010 = 12             PUSH OPERANDS
F006E030         020 $  0 010 F  5             IP(X)
F006E040         020 $  0 010 = 32             X-IP(X)=FP(X)
F006E050      *  9 010 ----          ,
```

```
F007E000+     F  7 010 $  0                        XP(X)
F007E010           024 K  2 052 *  9
F007E015           020 Q  6 002 E  6               MALFORMED IF AUX NOT EMPTY
F007E020           020 $  0 010 = 11               UNPACK (Q3)
F007E040           020 P  4 001 *  1
F007E050           120    0 014 *  2
F007E060      *  1 021 P  4 050 P  4
F007E070      *  2 020 K 12 050 P  5
F007E080           020 P  4 072    31
F007E090           050 P  6 025 N  1
F007E100           002 *  3 120    0
F007E110           004 N  8 010 *  4
F007E120      *  3 020 K 14 004 N  7
F007E130      *  4 050 P  4 032 P  6
F007E140           060 P  6 010 $  1
F007E150           020 $  0 010 = 30               PACK RESULT INTO Q3
F007E160      *  9 010 ----                      ,
```

```
F008E000+     F  8 024 K  2 010 $  0     DP(X)
F008E010         052 *  9 020 Q  6
F008E015         002 E  6 020 Q  3     MALFORMED IF AUX NOT EMPTY
F008E020         125 K 11 050 Q  3
F008E030      *  9 010 ----           ,
```

```
F009E000+     F  9 010 $  0                        SGN(X)
F009E010           024 K  2 052 *  9
F009E015           020 Q  6 002 E  6               MALFORMED IF AUX NOT EMPTY
F009E020           023 Q  3 002 *  9
F009E030           020 Q  3 124 K 12
F009E040           024 N  8 050 Q  3
F009E050      *  9 010 ----                       ,
```

```
F010E000+     F 10 010 $  0                    ARG(X,Y)    X IN Q3,    Y IN Q7
F010E010           024 K  2 052 *  9
F010E020           120       050 * 80         SET FLAG=0
F010E030           020 Q  3 124 K 12
F010E040           050 * 82 020 Q  3          STORE SIGN OF X IN * 82
F010E050           125 K 12 050 * 81          STORE ABSOLUTE VALUE OF X IN * 81
F010E055           020 Q  6 001 E  6          MALFORMED IF NOT ENOUGH ARGUMENTS
F010E060           020 Q  7 124 K 12
F010E070           050 * 84 020 Q  7          STORE SIGN OF Y IN * 84
F010E080           125 K 12 050 * 83          STORE ABSOLUTE VALUE OF Y IN * 83
F010E090           020 $  0 010 = 17          POP Y
F010E100           020 Q  6 002 E  6          MALFORMED IF TOO MANY ARGUMENTS
F010E110           023 * 83 002 *  1          IF Y=0, SET RESULT TO ZERO, GO TO * 1
F010E120           023 * 81 001 *  3          IF X=0, SET RESULT TO 1/4(PI/2),
F010E130           020 * 91 010 *  1          THEN DETERMINE QUADRANT
F010E140      *  3 020 * 83 004 * 81
F010E145           075    1 010 $  1
F010E150           065 T  0 023 T  0          IF Y=X, SET RESULT TO 1/4(PI/4),
F010E160      *  7 005 *  5 020 * 92          THEN DETERMINE QUADRANT
F010E170      *  5 010 *  1 021 T  0
F010E180           002 *  6 050 * 80         SET FLAG NON-ZERO
F010E190           004 * 81 020 * 83          INTERCHANGE Y AND X
F010E200           050 * 81 060 * 83
F010E210      *  6 020 * 83 050 Q  3
F010E220           020 $  0 010 = 12
F010E230           020 * 81 050 Q  3
F010E240           020 $  0 010 = 34          COMPUTE Y/X
F010E245           023 Q  3 002 *  1          TEST FOR RATIO EQUAL TO ZERO
F010E265           020 $  0 010 = 11          UNPACK
F010E270           020 P  4 072    31
F010E275           050 P  4 024 * 97          IF SFX IS LESS THAN -11, TREAT AS
F010E280           006 *  8 120               ZERO RATIO
F010E285      *  8 010 *  1 021 P  4
F010E290           025 I  1 024 J 11          COMPUTE FIXED POINT RATIO = Z
F010E295           056 * 10 020 P  6          DIVIDE BY 10*9, THEN 10*(-SFX-1)
F010E300           040 N  9 120               SET P6=0
F010E305      * 10 050 P  6 044 ----
F010E310           060 T  2 025 * 96          IF Z LESS THAN TAN(PI/8) USE K=1
F010E315           002 * 11 020 * 92          IF Z GREATER OR EQUAL USE K=3
F010E320           004 * 94 010 * 12
F010E325      * 11 020 * 93 004 * 95
F010E330      * 12 050 * 85 061 T  0          *85=K(PI/16)
F010E335           024 T  2 050 T  0          T1=Z-TAN (K.PI/16)
F010E340           032 T  2 076    1
F010E345           024 K 21 050 T  1          1/2(1+Z(TAN(K.PI/16))
F010E350           020 T  0 072    1
F010E355           044 T  1 060 T  0          T0=V, ARGUMENT FOR SERIES
F010E360           032 T  0 050 T  1          T1=V*2
F010E365           020 I  1 050 T  2          SET FLAG FOR CONSTANT + OR -
F010E370      * 13 020 * 98 050 T  3          SET INITIAL COEFFICIENT D=15
F010E375           021 T  2 050 T  2          (+ OR -1)/D
F010E380           040 T  3 020 P  6          SERIES SUM + 1/D = SUM
F010E385           064 P  6 004 T  1          (V*2).SERIES
F010E390           030 P  6 050 P  6
```

```
F010E395          020 T  3 025 I  4     D=D-2, IF NEGATIVE, TERMINATE SERIES
F010E397          005 * 24 024 I  2
F010E400    * 24 014 * 13 004 T  0     V.SERIES PLUS V
F010E405          030 P  6 024 T  0
F010E410          024 * 85 072    2     STORE 1/4 ANGLE
F010E415    *  1 050 T  0 023 * 80     IF FLAG IS NON-ZERO, SET ANGLE TO
F010E420          002 *  2 020 * 91     (1/4)PI/2 - ANGLE
F010E425          025 T  0 050 T  0
F010E430    *  2 023 * 82 006 *  4     TEST SIGN OF X, IF MINUS
F010E435          020 * 90 025 T  0     (1/4)PI - ANGLE
F010E440    *  4 050 T  0 023 * 84     TEST SIGN OF Y, IF MINUS
F010E445          006 * 14 021 T  0     -ANGLE
F010E450    * 14 050 T  0 070    0     CLEAR MQ AND P4
F010E455          060 P  4 021 T  0
F010E460          005 * 15 050 T  0     STORE MAGNITUDE OF ANGLE AND SIGN
F010E465    * 15 004 K 12 060 P  5
F010E470          120      004 T  0
F010E475          077    2 025 I  1     IF ANGLE IS LESS THAN ONE, MULTIPLY
F010E480          002 * 16 032 N  9     BY 10*9, SAVE BOTH PARTS OF RESULT
F010E483          050 P  6 060 T  2
F010E495    * 17 023 P  6 010 $  1     TEST FOR ZERO
F010E500          002 * 18 020 J 14     INITIALIZE COMPARE INSTRUCTION
F010E510    * 21 056 * 19 020 P  4     TO N8
F010E520          025 I  1 050 P  4     DECREASE SFX BY ONE
F010E530    * 19 020 P  6 025 ----
F010E540          002 * 20 020 * 19
F010E550          025 I  1 010 * 21
F010E560    * 20 020 J 11 025 I  1     COMPUTE POWER OF 10 FOR NORMALIZATION
F010E570          025 P  4 056 * 22
F010E575          056 * 25 020 P  4
F010E580          073   31 050 P  4
F010E583    * 25 004 T  2 030 ----
F010E586    * 22 004 P  6 036 ----
F010E590          060 P  6 025 N  9     TEST FOR ROUNDING UP TO 10*9
F010E595          001 * 23 020 N  8
F010E600          050 P  6 020 P  4
F010E605          024 K 14 050 P  4
F010E610    * 23 020 $  0 010 = 30     PACK
F010E620    *  9 010 ----              EXIT
F010E630    * 16 020 K 21 072    2     RESULT IS GREATER THAN ONE
F010E640          004 N  8 036 T  0     MULTIPLY BY 10*8 AND ROUND,
F010E650          077    2 010 * 18     MULTIPLY BY 4
F010E655    * 18 050 P  6 010 * 23
F010E660    * 80                       FLAG=0 IF Y LESS THAN X
F010E670    * 81                       ABSOLUTE VALUE OF X
F010E680    * 82                       SIGN OF X
F010E690    * 83                       ABSOLUTE VALUE OF Y
F010E700    * 84                       SIGN OF Y
F010E710    * 85                       K(PI/16)
F010E720    * 90 062,2077,325,0421     PI/4= .7853981633974
F010E730    * 91 031,1037,552,4210     PI/8= .3926990816987
F010E740    * 92 014,4417,665,2104     PI/16= .1963495408494
F010E750    * 93 045,5457,437,6314     3 PI/16= .5890486225481
F010E760    * 94 014,5657,536,0125     TAN(PI/16)= .1989123673
F010E770    * 95 052,6067,012,5337     TAN(3 PI/16)= .6681786379
```

```
F010E780      * 96 032,4047,463,1771      TAN (PI/8)= .4142135623
F010E790      * 97                  11
F010E800      * 98                  15,
```

F011E000+ F 11 004 * 91 010 $ 0 MAX

```
F012E000      F 12 004 * 92 010 $  0      MIN
F012E010           024 K  2 052 *  9
F012E020           060 *  2 010 $  1
F012E030           020 Q  6 001 E  6      MALFORMED IF NOT MULTIPLE ARGUMENT
F012E040      *  1 020 Q  3 050 H 21
F012E050           020 Q  7 050 H 22
F012E060           020 $  0 010 = 17
F012E070           020 $  0 010 = 70      COMPARE
F012E080      *  2 --- ---- --- ----
F012E090      *  3 020 H 22 050 Q  3
F012E100      *  4 020 Q  7 002 *  1
F012E110      *  9 010 ----
F012E120      * 91 071    4 001 *  4      FOR MAX
F012E130      * 92 071    6 001 *  4,     FOR MIN
```

```
G000E000+     G  0 010 *  6 010 *  0     SET (014=SHORT FORM)
G000E010      *  0 020 J 12 050 P  2     INITIALIZE
G000E020           020 $  0 010 =  2     ADVANCE TO NONSPACE
G000E030           020 $  0 010 * 80     LOCAL ROUTINE (GIVES E6 IN IND. SHORT)
G000E040           020 P  1 001 * 10
G000E050           124 K 19 010 *  7
G000E060      *  6 020 $  0 010 = 39     VERIFY SPACE AND ADVANCE TO NONSPACE
G000E070           020 $  0 010 * 80     LOCAL ROUTINE (COMMON TO G001)
G000E080           020 P  1 124 K 19
G000E090      *  7 025 K 19 001 E  6     ERROR IF NOT TERMINAL
G000E100      * 10 020 * 90 050 H 27     SET LETTER
G000E110           020 * 91 050 H 28     SET INDEX 1
G000E120           020 * 92 050 H 29     SET INDEX 2
G000E130           020 Q  3 050 H 30     SET VALUE
G000E140           020 $  0 010 = 13     POP Q3
G000E150           020 $  0 010 = 73     ASSIGN VALUE
G000E160           010 X  5              EXIT TO X5 ADVANCE (NO COMMA HERE)
```

```
G001E000+    G  1 010 $  0                    DO (SAME CONTEXT AS 'SET')
G001E010        020 $  0 010 =  39            VERIFY SPACE AND ADVANCE TO NONSPACE
G001E020        020 $  0 010 =  27            ACCUMULATE WORD
G001E030        025 W  7 050 *  90            SAVE DIFFERENCE IN *90
G001E040        023 *  90 002 *   1           JUMP IF 'PART'
G001E045        020 H  19 025 W   9
G001E050        050 T  0 023 T   0
G001E055        002 *  1 010 E   6           JUMP IF 'STEP' ELSE MALFORMED
G001E060    *   1 020 $  0 010 =  39          VERIFY SPACE AND ADVANCE TO NONSPACE
G001E061        020 $  0 010 =  21            EVALUATE PART OR STEP NUMBER
G001E062        020 P  1 124 K  19
G001E063        025 K  19 002 *  25           JUMP IF TERMINAL
G001E064        020 $  0 010 =  79            VERIFY SPACE BEFORE CHECKING 'FOR'
G001E065        020 $  0 010 =  27            ACCUMULATE WORD
G001E066        025 W  2 050 T   0
G001E067        023 T  0 001 E   6           MALFORMED IF NOT 'FOR'
G001E068        020 $  0 010 =  39            VERIFY SPACE AND ADVANCE TO NONSPACE
G001E069        020 P  1 124 K  19
G001E070        025 K  19 002 E   6           ERROR IF TERMINAL
G001E071    *  25 023 *  90 001 *   4         JUMP IF 'STEP'
G001E072        020 $  0 010 =  44            VALIDATE
G001E075        020 Q  3 050 P  20            SET PART
G001E080        020 $  0 010 =  50            FIND PART
G001E085        001 E  17 052 *   2           ERROR IF CAN'T
G001E090        024 K  2 052 *   3
G001E095    *   2 020 ---- 050 P  16
G001E100    *   3 020 ---- 050 P  21
G001E105        010 *  5
G001E120    *   4 020 $  0 010 =  45          VALIDATE
G001E125        020 Q  3 050 P  21            SET STEP
G001E130        020 K  21 050 Q  14           SET STEP FLAG IN Q14
G001E135        020 K  4 050 P  20            MARK P20 MINUS AS FLAG ALSO
G001E140        020 $  0 010 =  51            FIND STEP
G001E145        001 E  16 050 P  16           ERROR IF CAN'T
G001E150    *   5 020 $  0 010 =  13          POP Q3
G001E155 ]a     023 *  91 002 *  13          JUMP IF 'FOR'
G001E155 ]b     020 P  1 124 K  19
G001E160        025 K  19 001 *  13           JUMP IF NOT TERMINAL
G001E165    *  12 020 Q  12 001 *   8         JUMP IF NO SUSPENDED TASK
G001E170        050 Q  8 020 K   4            CANCEL SUSPENDED TASK
G001E171        050 Q  12 050 Q  13
G001E172    *  11 020 $  0 010 =  76
G001E173        020 Q  8 002 *  11
G001E175    *   8 020 $  0 010 =  18          PUSH Q9
G001E180        020 P  21 050 Q   9           SET Q9 TO FIRST STEP
G001E185        020 Q  14 053 Q   8           BRING ALONG ANY STRUCTURE
G001E190        120    0 050 Q  14            RELIEVE Q14 OF RESPONSIBILITY
G001E195        020 K  4 050 P  17            RESET FLAGS
G001E200        050 P  23 010 X   6           AND FETCH
G001E240    *  13 020 $  0 010 *  80          LOCAL ROUTINE COMMON TO 'SET'
G001E245        020 Q  3 050 H  30            SAVE FIRST VALUE FOR USE IN *40
G001E246        020 P  1 124 K  19
G001E247        025 K  19 001 *   9           JUMP IF NOT SINGLE VALUE
G001E248        020 $  0 010 =  13            POP Q3
```

Could have dispensed with these.

G001E066 & 067 should have used *91 for TO 30

G001E165a would have worked as a replacement for G001E165b and ... Somehow I didn't get the full correction made ... squeeze out a ... work.

JCP 4-10-67

```
G001E249          010 * 41                          JUMP
G001E250     *  9 020 $  0 010 = 10
G001E260          056 * 14 071   21
G001E270          052 Q 14 052 * 93
G001E280     * 14 120    0 050 ----                  CLEAR TOP OF 'PART' ELEMENT
G001E290          020 * 93 024 K  2
G001E300          052 * 15 020 P 20
G001E310     * 15 050 ---- 020 * 93                  STORE PART NO. IN STRUCTURE
G001E320          052 * 16 010 $  1
G001E330          020 $  0 010 = 10
G001E340          056 * 17 071   21
G001E350     * 16 052 ---- 052 * 93
G001E360     * 17 120    0 050 ----                  CLEAR TOP OF 'VARIABLE' ELEMENT
G001E370          020 * 93 024 K  2
G001E380          052 * 18 020 * 90
G001E390     * 18 050 ---- 010 $  1                  STORE VARIABLE
G001E400          020 * 91 001 * 30                  JUMP IF SCALAR
G001E410          020 $  0 010 = 10
G001E420          056 * 93 056 * 19
G001E430          024 K  1 056 * 20
G001E440     * 19 120    0 050 ----                  CLEAR TOP OF INDEX ELEMENT
G001E450     * 20 020 * 91 050 ----                  STORE INDEX
G001E460          020 * 93 052 * 21
G001E470     * 21 056 ---- 010 $  1                  TIE INTO STRUCTURE
G001E480          020 * 92 001 * 30                  JUMP IF NOT DOUBLY INDEXED
G001E490          020 $  0 010 = 10
G001E500          056 T  0 056 * 22
G001E505          024 K  1 056 * 23
G001E510     * 22 120    0 050 ----                  CLEAR TOP OF INDEX ELEMENT
G001E515     * 23 020 * 92 050 ----                  STORE INDEX
G001E520          020 * 93 056 * 24
G001E525     * 24 020 T  0 056 ----                  TIE INTO STRUCTURE
G001E530     * 30 020 * 93 052 * 31                  SET UP VALUE ELEMENT
G001E535          020 $  0 010 = 10
G001E540          056 * 93 056 * 32
G001E545          024 K  1 056 * 33
G001E550          025 K  1 071   21
G001E555     * 31 052 ---- 052 * 93
G001E560     * 32 120    0 050 ----                  CLEAR TOP OF VALUE ELEMENT
G001E565     * 33 020 Q  3 050 ----                  STORE VALUE
G001E570          020 $  0 010 = 13                  POP Q3
G001E575     * 34 020 P  1 124 A 59
G001E580          025 A 59 001 * 40                  JUMP IF NOT COMMA
G001E585          020 $  0 010 =  2                  ADVANCE TO NONSPACE
G001E590          020 $  0 010 = 21                  EVALUATE EXPRESSION
G001E595          010 * 30
G001E600     * 40 020 P  1 124 K 19
G001E605          025 K 19 001 * 50                  JUMP IF NOT TERMINAL
G001E610     * 41 020 * 90 050 H 27
G001E615          020 * 91 050 H 28
G001E620          020 * 92 050 H 29
G001E640          020 $  0 010 = 73                  ASSIGN FIRST VALUE
G001E645          010 * 12
G001E650     * 50 020 P  1 124 A 73
3001E652          025 A 73 002 * 51                  JUMP IF LEFT PARENTHESIS
```

```
G001E654          020 P   1 124 A123
G001E656          025 A123  001 E   6     ERROR IF NEITHER LEFT PAREN OR BRACKE
G001E660    * 51  020 $   0 010 =  14     PUSH Q4-Q5
G001E665          020 P   1 050 Q   5     SAVE LEFT GROUPER AS OPERATOR
G001E667          020 $   0 010 =   2
G001E670          020 $   0 010 =  21     EVALUATE INCREMENT EXPRESSION
G001E675          020 $   0 010 =  77     MATCH GROUPERS
G001E680          020 $   0 010 =   2     ADVANCE TO NONSPACE
G001E685          020 $   0 010 =  21     EVALUATE TERMINAL EXPRESSION
G001E690          020 Q   3 050 H  22     MOVE TERMINAL VALUE
G001E695          020 $   0 010 =  13     POP Q3
G001E700          020 Q   3 050 H  23     MOVE INCREMENT
G001E705          020 $   0 010 =  13     POP Q3
G001E710          020 *  93 024 K   1
G001E715          071    21 052 *  59
G001E720    * 59  020 ----  050 H  21     COPY INITIAL VALUE
G001E730          020 $   0 010 =  70     COMPARE INITIAL AND TERMINAL VALUES
G001E735          050 T   0 124 K  25
G001E740          025 K  25 002 *  34     JUMP BACK IF LIMITS EQUAL
G001E745          023 H  23 002 E  28     ERROR IF INCREMENT EQUALS ZERO
G001E750          020 T   0 070    3      NOW TEST EXCLUSIVE-OR OF LESS SIGNAL
G001E755          024 H  23 124 K  12       WITH SIGN OF INCREMENT
G001E760          025 K  12 001 E  28     ERROR IF INCREMENT INCOMPATIBLE
G001E765    * 52  020 *  93 056 *  53     (SPACE OK TO ADD STRUCTURE)
G001E770    * 53  020 Q   0 056 ----
G001E775          056 *  54 024 K   1
G001E780    * 54  056 *  57 020 ----
G001E785          056 *  55 056 *  56
G001E790          056 *  93 024 K   1
G001E795    * 55  056 *  58 004 ----     SAVE LINK TO AVAILABLE SPACE
G001E800    * 56  120     0 050 ----     CLEAR TOP OF TERMINAL ELEMENT
G001E805    * 57  020 H  23 050 ----     SET INCREMENT
G001E810    * 58  020 H  22 050 ----     SET TERMINAL VALUE
G001E815          060 Q   0 010 *  34     SET Q0 AND LOOP
G001E820    * 80  024 K   2 052 *  89     LOCAL ROUTINE COMMON TO 'SET' AND 'DO
G001E830          020 P   1 050 *  90     STORE CHARACTER
G001E835          124 K  34 025 K  34
G001E840          001 E   6 020 K   4     ERROR IF NOT LETTER
G001E845          050 *  91 050 *  92     SET INDEX FLAGS
G001E850          020 $   0 010 =   1     ADVANCE ONE
G001E855          124 K  31 025 K  31
G001E860          001 *  82 010 $   1     JUMP IF NOT LEFT GROUPER
G001E865          020 $   0 010 =  26     EVALUATE GROUPED LIST
G001E870          020 Q   6 001 *  83     JUMP IF Q7 EMPTY
G001E872          020 Q   7 050 *  92     STORE 2ND INDEX
G001E874          020 $   0 010 =  17     POP Q7
G001E876          020 Q   6 002 E   6     MALFORMED IF EXTRA INDICES
G001E878    * 83  020 $   0 010 =  41     CHECK AND CONVERT FIRST INDEX
G001E880          020 Q   3 050 *  91     STORE CONVERTED INDEX LOCALLY
G001E885          020 *  92 001 *  81     JUMP IF SINGLY INDEXED
G001E890          050 Q   3 010 $   1
G001E895          020 $   0 010 =  41     CHECK AND CONVERT SECOND INDEX
G001E900          020 Q   3 050 *  92     STORE CONVERTED INDEX LOCALLY
G001E910    * 81  020 $   0 010 =  13     POP Q3
G001E915    * 82  020 $   0 010 =   3     ELIMINATE SPACES
```

```
G001E920           124 A 43 025 A 43
G001E925           002 $  1 010 E  6      MALFORMED IF NOT '='
G001E930           020 $  0 010 =  2      ADVANCE TO NONSPACE
G001E935           020 $  0 010 = 21      EVALUATE EXPRESSION
G001E940      * 89 010 ----               EXIT *80 ROUTINE
G001E945      * 90                        LETTER
G001E950      * 91                        INDEX
G001E955      * 92                        INDEX
G001E960      * 93      ----     ----,
```

```
G002E000+     G  2 010 $   0                    TO
G002E010           020 Q   8 001 E  10          ERROR IF DIRECT
G002E015           071     1 001 X   7          JUMP TO ASCEND IF 'DO STEP'
G002E020           020 $   0 010 =  39          VERIFY SPACE AND ADVANCE
G002E030           020 $   0 010 =  27          PICK UP WORD
G002E040           025 W   9 050 T   0
G002E050           023 T   0 001 *   1          JUMP IF NOT 'STEP'
G002E060           020 $   0 010 =  39          VERIFY SPACE AND ADVANCE
G002E070           020 $   0 010 =  21          EVALUATE EXPRESSION
G002E080           020 P   1 124 K  19          VERIFY TERMINAL
G002E081           025 K  19 001 E   6
G002E085           020 $   0 010 =  45          VALIDATE
G002E090           020 Q   3 050 P  21          SET CONTEXT
G002E100           020 $   0 010 =  13          POP Q3
G002E110           020 $   0 010 =  51          FIND STEP
G002E120           001 E  16 050 P  16          ERROR IF CAN'T ELSE SET P16
G002E125           020 P  21 014 *   4
G002E140     *  1  020 H  19 025 W   7
G002E150           050 T   0 023 T   0
G002E160           002 *   2 010 E   6          ERROR IF NOT 'PART'
G002E170     *  2  020 $   0 010 =  39          VERIFY SPACE AND ADVANCE
G002E180           020 $   0 010 =  21          EVALUATE EXPRESSION
G002E190           020 P   1 124 K  19          VERIFY TERMINAL
G002E191           025 K  19 001 E   6
G002E195           020 $   0 010 =  44          VALIDATE
G002E200           020 Q   3 050 P  20          SET CONTEXT
G002E210           020 $   0 010 =  13          POP Q3
G002E220           020 $   0 010 =  50          FIND PART
G002E230           001 E  17 052 *   3          ERROR IF CAN'T
G002E235           024 K   2 052 *   4
G002E240     *  3  020 ---- 050 P  16          ELSE SET P16 FOR FIRST STEP
G002E245     *  4  020 ---- 050 Q   9          SET Q9
G002E250           010 X   6          ,         JUMP TO FETCH
```

```
G003E000+     G  3 010 $  0                    DONE
G003E010           020 $  0 010 = 38           SIMPLE-INDIRECT TEST
G003E020           010 X  7          ,         JUMP TO ASCEND
```

```
G004E000+       $                       M  0      CANCEL
G004E010          020 $  0 010 = 37     SIMPLE DIRECT TESTS
G004E020          020 Q 12 001 *  2     JUMP IF NO STRUCTURE
G004E030          050 Q  8 020 K  4     ELSE MOVE STRUCTURE BACK TO Q8
G004E040          050 Q 12 050 Q 13
G004E050          050 P 17 050 P 23     RESET P17 AND P23 FLAGS
G004E060    *   1 020 $  0 010 = 76     ERASE ONE LEVEL OF STRUCTURE
G004E070          020 Q  8 002 *  1     JUMP IF MORE
G004E080    *   2 014 X 12          ,   EXIT TO X12 TO SWITCH
G004E999        /     G004 010,7000/
```

```
G005E000+       $                       M  0    DELETE
G005E010           020 $  0 010 = 39    VERIFY SPACE AND ADVANCE TO NONSPACE
G005E020           020 P  1 124 K 34
G005E030           025 K 34 001 E  6    VERIFY LETTER
G005E040           020 P  2 024 K  3
G005E050           052 *  1 010 $  1
G005E060     *  1 020 ---- 124 K 34
G005E070           025 K 34 001 D  2    JUMP IF NOT A WORD
G005E080           020 $  0 010 = 27    ACCUMULATE WORD
G005E090           025 W  0 050 T  0
G005E100           023 T  0 002 D  1    JUMP IF 'ALL'
G005E110           020 $  0 010 = 39    VERIFY SPACE AND ADVANCE TO NONSPACE
G005E120           004 H 19 010 $  1
G005E130           020 W  9 065 T  0
G005E140           023 T  0 002 * 30    JUMP IF 'STEP'
G005E150           020 W  7 065 T  0
G005E160           023 T  0 002 * 40    JUMP IF 'PART'
G005E170           020 W  3 065 T  0
G005E180           023 T  0 002 * 50    JUMP IF 'FORM'
G005E190           010 E  6             ELSE MALFORMED
G005E200     * 30  020 Q  8 002 E  9     STEP-ILLEGAL INDIRECT
G005E210           020 $  0 010 * 70    BORROW SPARE TANK
G005E220           020 $  0 010 = 21    EVALUATE STEP NUMBER
G005E230           020 P  1 124 K 19    VERIFY TERMINAL
G005E231           025 K 19 001 E  6
G005E240           020 $  0 010 = 45    VALIDATE STEP NUMBER
G005E250           020 Q  3 050 P 21    SET STEP IN CONTEXT
G005E260           020 $  0 010 = 13    POP Q3
G005E270           020 $  0 010 = 51    FIND STEP
G005E280           001 E 16 010 $  1    ERROR IF CAN'T
G005E290           020 $  0 010 = 56    ERASE STEP A/C H21-H26
G005E300     * 31  020 $  0 010 * 80    RESTORE SPARE TANK
G005E310           010 X  5             EXIT
G005E320     * 40  020 Q  8 002 E  9     PART - ILLEGAL INDIRECT
G005E330           020 $  0 010 * 70    BORROW SPARE TANK
G005E340           020 $  0 010 = 21    EVALUATE PART NUMBER
G005E350           020 P  1 124 K 19    VERIFY TERMINAL
G005E351           025 K 19 001 E  6
G005E360           020 $  0 010 = 44    VALIDATE PART NUMBER
G005E370           020 Q  3 050 P 20    SET PART IN CONTEXT
G005E380           020 $  0 010 = 13    POP Q3
G005E390           020 $  0 010 = 50    FIND PART
G005E400           001 E 17 010 $  1    ERROR IF CAN'T
G005E410           020 H 23 050 * 91    SET POINTERS IN *91
G005E420     * 41  020 * 91 052 * 42
G005E430           124 K  6 025 K  1
G005E440           001 * 43 010 $  1    JUMP IF NO MORE STEPS
G005E450     * 42  004 ---- 060 * 91    SET PARAMETER AND UPDATE POINTER
G005E460           020 $  0 010 = 29    ERASE STEP STRING EXCEPT HEAD
G005E470           010 * 41             LOOP
G005E480     * 43  004 H 22 075   21     SET PARAMETER
G005E490           020 $  0 010 = 28    ERASE LEFT LINKED LIST
G005E500           020 H 21 056 * 44
G005E510     * 44  020 H 23 056 ----
```

```
G005E520     * 45 020 $  0 010 * 80     RESTORE SPARE TANK
G005E530          010 X  5              EXIT
G005E540     * 50 020 Q  8 002 E  9     FORM-ILLEGAL INDIRECT
G005E550          020 $  0 010 * 70     BORROW SPARE TANK
G005E560          020 $  0 010 = 21     EVALUATE FORM NUMBER
G005E570          020 P  1 124 K 19     VERIFY TERMINAL
G005E571          025 K 19 001 E  6
G005E580          020 $  0 010 = 46     VALIDATE FORM NUMBER
G005E590          020 Q  3 050 P 22     SET FORM IN CONTEXT
G005E600          020 $  0 010 = 13     POP Q3
G005E610          020 $  0 010 = 52     FIND FORM
G005E620          001 E 19 010 $  1     ERROR IF CAN'T
G005E630          020 $  0 010 = 57     ERASE FORM A/C H24-H26
G005E640     * 51 020 $  0 010 * 80     RESTORE SPARE TANK
G005E650          010 X  5              EXIT
G005E700     * 70 024 K  2 052 * 79     BORROW SPARE TANK (SUBROUTINE)
G005E710          020 Q  1 056 * 71
G005E720     * 71             020 ----
G005E730          056 * 72 010 $  1
G005E740     * 72 020 Q  0 050 ----
G005E750          020 Q  1 050 Q  0
G005E760          020 K  4 050 Q  1
G005E770     * 79 010 ----
G005E780     * 80 024 K  2 052 * 89     RESTORE SPARE TANK (SUBROUTINE)
G005E790          020 Q  0 056 * 81
G005E800     * 81 050 Q  1 020 ----
G005E810          056 * 82 004 K  4
G005E820     * 82 056 * 83 020 ----
G005E830     * 83 050 Q  0 060 ----
G005E840     * 89 010 ----
G005E850     * 91                ,     LOCAL STORAGE
G005E999       /     G005 010,7000/
```

```
G006E000+       $                   M  0    TYPE
G006E010          020 $  0 010 =  39        VERIFY SPACE AND ADVANCE TO NONSPACE
G006E020          020 P  2 050 *  90        SAVE POINTER
G006E022          120    0 050 H  19        CLEAR H19
G006E024          020 P  1 124 A  66
G006E026          025 A 66 002 D   8        JUMP IF QUOTES
G006E030          020 $  0 010 =  27        ACCUMULATE WORD
G006E040          025 W  0 050 T   0
G006E050          023 T  0 002 D   3        JUMP IF 'ALL' TO D3 EXTENSION
G006E060          004 H 19 010 $   1
G006E070          020 W  9 065 T   0
G006E080          023 T  0 002 D   8        JUMP IF 'STEP'
G006E090          020 W  7 065 T   0
G006E100          023 T  0 002 D   8        JUMP IF 'PART'
G006E110          020 W  3 065 T   0
G006E120          023 T  0 002 D   8        JUMP IF 'FORM'
G006E130          020 W 13 065 T   0
G006E140          023 T  0 002 D   8        JUMP IF 'SIZE'
G006E150          020 W 11 065 T   0
G006E160          023 T  0 002 D   8        JUMP IF 'USERS'
G006E162          020 W  1 065 T   0
G006E164          023 T  0 002 D   8        JUMP IF 'TIME'
G006E170          020 * 90 050 P   2        ELSE EVALUATE VALUES
G006E180          052 *  1 010 $   1
G006E190     *  1 020 ---- 050 P   1        RESET P1 AND P2
G006E200     *  2 020 P  1 124 A108
G006E210          025 A108 001 *   4        JUMP IF NOT UNDERLINE
G006E220          020 $  0 010 =  12        PUSH Q3
G006E230          020 K  4 050 Q   3        STACK UNDEFINED SYMBOL IN Q3
G006E240     *  3 020 $  0 010 =   1
G006E250          124 A108 025 A108
G006E260          002 *  3 010 $   1        LOOP IF UNDERLINE
G006E270          020 $  0 010 =   3        ELIMINATE SPACES
G006E280          010 * 12
G006E290     *  4 020 P  1 124 K  34
G006E300          025 K 34 001 *   9        JUMP IF NOT LETTER
G006E310          020 P  2 024 K   3
G006E320          052 *  5 052 *   6
G006E330     *  5 020 ---- 124 K  34
G006E340          025 K 34 002 *   9        JUMP IF NEXT CHARACTER ALSO LETTER
G006E350     *  6 020 ---- 124 K  31
G006E360          025 K 31 002 *   9        JUMP IF NEXT CHARACTER A LEFT GROUPEF
G006E370          020 P  1 052 *   7
G006E380     *  7 020 ---- 002 *   9        JUMP IF SCALAR
G006E390          071    5 001 *   8        JUMP IF MATRIX
G006E400          071    1 002 *   9        JUMP IF NOT VECTOR (IE UNDEFINED)
G006E410     *  8 020 $  0 010 =  12        PUSH Q3
G006E420          020 P  1 024 K   4
G006E430          050 Q  3 010 $   1        STACK INDEXED LETTER PLUS FLAG
G006E440          020 $  0 010 =   2        ADVANCE TO NONSPACE
G006E450          010 * 12
G006E460     *  9 020 P  2 050 P  18        SET STARTING POINTER
G006E470          020 $  0 010 =  21        EVALUATE EXPRESSION
G006E480          020 P  2 050 P  19
```

```
G006E490     * 10 020 P 19 025 K  3
G006E500          052 * 11 050 P 19
G006E510     * 11 020 ---- 124 A 14
G006E520          025 A 14 002 * 10
G006E530          020 P 19 025 P 18
G006E540          056 P 18 010 $  1
G006E550          020 $  0 010 = 12      PUSH Q3
G006E560          020 P 18 050 Q  3      STACK EXPRESSION RANGE INDICATORS
G006E570     * 12 020 P  1 124 A 59
G006E580          025 A 59 001 * 13      JUMP IF NOT A COMMA
G006E590          020 $  0 010 =  2      ADVANCE TO NONSPACE
G006E600          010 *  2               LOOP
G006E610     * 13 020 P  1 124 K 19
G006E620          025 K 19 002 * 20      JUMP IF TERMINAL
G006E625          020 $  0 010 = 79      VERIFY PRECEDING SPACE
G006E630          020 $  0 010 = 27      ACCUMULATE WORD
G006E640          025 W  6 050 T  0
G006E650          023 T  0 001 E  6      ERROR IF NOT 'IN'
G006E660          020 $  0 010 = 39      VERIFY SPACE AND ADVANCE TO NONSPACE
G006E670          020 $  0 010 = 27      ACCUMULATE WORD
G006E680          025 W  3 050 T  0
G006E690          023 T  0 001 E  6      ERROR IF NOT 'FORM'
G006E700          020 $  0 010 = 39      VERIFY SPACE AND ADVANCE TO NONSPACE
G006E710          020 $  0 010 = 21      EVALUATE FORM NUMBER
G006E720          020 P  1 124 K 19      VERIFY TERMINAL
G006E730          025 K 19 001 E  6
G006E740          020 $  0 010 = 46      VALIDATE FORM NUMBER
G006E750          020 Q  3 050 P 22      PUT FORM NUMBER IN CONTEXT
G006E760          020 $  0 010 = 13      POP Q3
G006E770          020 $  0 010 = 52      FIND FORM
G006E780          056 P  3 001 E 19      ERROR IF CAN'T
G006E790          020 $  0 014 =  7      UNPACK TO S
G006E800     * 14 020 Q  2 001 D  7      JUMP TO D7 EXT WHEN Q2 EMPTIED TO Q6.
G006E810          020 Q  2 056 * 15
G006E820          056 * 16 024 K  1
G006E830          056 * 17 056 * 18
G006E840     * 15 004 Q  2 020 ----
G006E850          050 Q  2 010 $  1
G006E860     * 16 020 Q  6 050 ----
G006E870     * 17 060 Q  6 020 ----
G006E880          004 Q  3 050 Q  3
G006E890     * 18 020 Q  7 050 ----
G006E900          060 Q  7 010 * 14
G006E905     * 20 020 Q  2 001 D  6      JUMP TO D6 EXT. WHEN Q2 EMPTIED TO Q6
G006E910          020 Q  2 056 * 21
G006E915          056 * 22 024 K  1
G006E920          056 * 23 056 * 24
G006E925     * 21 004 Q  2 020 ----
G006E930          050 Q  2 010 $  1
G006E935     * 22 020 Q  6 050 ----
G006E940     * 23 060 Q  6 020 ----
G006E945          004 Q  3 050 Q  3
G006E950     * 24 020 Q  7 050 ----
G006E955          060 Q  7 010 * 20
G006E960     * 90                   ,    LOCAL STORAGE
```

G006E999 / G006 010,7000/

```
G007E000+    G  7 010 $  0                    LINE
G007E010          020 $  0 010 =  3           ELIMINATE SPACES
G007E020          124 K 19 025 K 19           ERROR IF NOT.TERMINAL
G007E030          001 E  6 020 A 42           SET UP CR+EOM
G007E040          050 S  1 010 X 14,          TRANSMIT AND TO X5 VIA X14.
```

```
G008E000+     G  8 010 $  0                          PAGE
G008E010           020 $  0 010 =  3                 ELIMINATE SPACES
G008E020           071   31 002 E  6                 ERROR IF NOT TERMINAL
G008E030           020 $  0 010 = 22                 CALL FOR NEW PAGE
G008E040           010 X  5          ,
```

G009E000+ G 9 010 E 6

G010E000+ G 10 010 E 6

```
G011E000+     G 11 010 $  0                      FORM
G011E010          020 $  0 010 = 39              VERIFY SPACE AND ADVANCE
G011E012          020 Q  9 002 E  9              ERROR IF INDIRECT
G011E015          020 P 10 002 E  6              ERROR IF CONDITION ATTACHED
G011E020          020 $  0 010 = 21              EVALUATE FORM NUMBER
G011E030          020 $  0 010 = 46              VALIDATE
G011E040          020 Q  3 050 P 22              SET CONTEXT
G011E050          020 $  0 010 = 13              POP Q3
G011E060          020 $  0 010 =  3              ELIMINATE SPACES
G011E070          124 A 80 025 A 80
G011E080          002 *  1 010 E  6              MALFORMED IF NOT COLON
G011E090     *  1 020 $  0 010 =  1              STEP ONE CHARACTER
G011E100          001 *  2 010 E  6              MALFORMED IF NOT TERMINAL MINUS
G011E110     *  2 020 I  1 050 P 13              SET SUBSTATE FOR FORM
G011E120          020 $  0 010 = 55              SWITCH TO USER
G011E130          020 $  0 010 = 53              KICK OUT PROGRAM
G011E140          010 X  1         ,             JUMP TO Q SERVICE
```

```
G012E000+       $                   M  0      STOP
G012E010          020 $  0 010 = 38           SIMPLE-INDIRECT TESTS
G012E020          120    0 050 P 17           SET 'STOPPED' FLAG
G012E030          020 S  0 050 P 12
G012E040          020 $  0 010 = 64           PREPARE MESSAGE
G012E042              * 90     * 90
G012E044          020 J 12 050 P  2           COPY STEP NUMBER FROM R TO S
G012E046    *  1 020 $  0 010 =  2
G012E048          124 K 35 025 K 35
G012E050          002 *  2 010 $  1
G012E052          020 P  1 124 A 27
G012E054          025 A 27 001 *  4
G012E056    *  2 020 P 12 024 K  3
G012E058          050 P 12 056 *  3
G012E060    *  3 020 P  1 050 ----
G012E062          020 *  1 010 =  1
G012E070    *  4 020 A 27 050 P 11
G012E080          020 $  0 010 = 66           ADD PERIOD
G012E090          020 A 42 050 P 11
G012E100          020 $  0 010 = 66           ADD CR+EOM
G012E110          010 X 12                    TRANSMIT AND SWITCH VIA X12
G012E150    * 90 071,0631,142,3447            'STOPPED BY STEP'
G012E160          012,4240,341,1070
G012E170          007,0621,461,2447
G012E180          007,2000,000,0000,
G012E999        /     G 12 010,7000/
```

```
G013E000+       $                       M  0      GO
G013E010          020 $  0 010 = 37     SIMPLE-DIRECT TESTS
G013E020          020 Q 12 050 Q  8
G013E030          020 Q 13 050 Q  9
G013E040          020 K  4 050 Q 12
G013E050          050 Q 13 010 $  1
G013E052          020 P 23 001 *  4     JUMP IF NO SPACE PROBLEM IN X7
G013E054          020 K  4 050 P 23     RESET P23 FLAG
G013E056          010 X  7              JUMP TO ASCEND
G013E060    *  4 020 P 17 001 *  1     JUMP IF NOT 'STOP'
G013E070          020 K  4 050 P 17     RESET 'STOP' FLAG
G013E080          010 X  5              JUMP TO ADVANCE
G013E090    *  1 020 Q  8 001 E  7     ERROR IF NO OPEN TASK
G013E100          020 Q  9 050 P 21
G013E110          020 $  0 010 = 51     FIND CURRENT STEP
G013E120          002 *  2 010 X  5     JUMP IF FOUND, ELSE ADVANCE
G013E130    *  2 050 P 16 010 X  6,    SET P16 AND FETCH
G013E999       /     G 13 010,7000/
```

```
G014E000+       $                       M  0    DEMAND
G014E010          020 P 13 025 I  3
G014E020          050 T  0 023 T  0
G014E030          002 * 50 010 $  1     JUMP IF EXPECTING VALUE
G014E035          020 Q  8 001 E 10     ERROR IF DIRECT
G014E040          020 $  0 010 = 39     VERIFY SPACE AND ADVANCE TO NONSPACE
G014E050          050 P 27 124 K 34
G014E060          025 K 34 001 E  6     ERROR IF NOT LETTER
G014E070          020 K  4 050 P 28     RESET INDEX FLAGS
G014E080          050 P 29 010 $  1
G014E090          020 $  0 010 =  1     ADVANCE ONE
G014E100          124 K 31 025 K 31
G014E110          001 *  3 010 $  1     JUMP IF NOT LEFT GROUPER
G014E120          020 $  0 010 = 26     EVALUATE GROUPED LIST
G014E130          020 Q  6 001 *  1     JUMP IF NO 2ND INDEX
G014E140          020 Q  7 050 P 29     STORE 2ND INDEX
G014E150          020 $  0 010 = 17     POP Q7
G014E160          020 Q  6 002 E  6     ERROR IF EXTRA INDICES
G014E170     *  1 020 $  0 010 = 41     CHECK AND CONVERT FIRST INDEX
G014E180          020 Q  3 050 P 28     STORE CONVERTED INDEX
G014E190          020 P 29 001 *  2     JUMP IF SINGLE INDEX
G014E200          050 Q  3 010 $  1
G014E210          020 $  0 010 = 41     CHECK AND CONVERT SECOND INDEX
G014E220          020 Q  3 050 P 29     STORE CONVERTED INDEX
G014E230     *  2 020 $  0 010 = 13     POP Q3
G014E240     *  3 020 $  0 010 =  3     ELIMINATE SPACES
G014E250          124 K 19 025 K 19
G014E260          002 *  4 010 E  6     ERROR IF NOT AT TERMINAL
G014E270     *  4 021 I  7 050 * 90
G014E280          020 P 28 001 *  7
G014E290          025 * 91 002 *  5
G014E300          020 I  3 014 *  5
G014E310     *  5 020 I  4 024 * 90
G014E320          050 * 90 010 $  1
G014E330          020 P 29 001 *  7
G014E340          025 * 91 002 *  6
G014E350          020 I  2 014 *  6
G014E360     *  6 020 I  3 024 * 90
G014E370          050 * 90 010 $  1
G014E380     *  7 020 S  0 050 P 12     INITIALIZE OUTPUT LINE
G014E390     *  8 020 * 90 002 *  9     JUMP WHEN ALL SPACES ARE IN
G014E400          024 I  1 050 * 90
G014E410          020 $  0 010 = 62     INSERT SPACE
G014E420          010 *  8             LOOP
G014E430     *  9 020 P 27 050 P 11
G014E440          020 $  0 010 = 66     INSERT LETTER
G014E450          020 P 28 001 * 11     JUMP IF NO SUBSCRIPTS
G014E460          020 J 20 052 * 12     =77 LEFT RIGHT GROUPER IN J20
G014E465     * 12 020 ---- 050 P 11
G014E470          020 $  0 010 = 66     INSERT LEFT GROUPER
G014E480          020 P 28 070    12
G014E490          050 H 20 010 $  1
G014E500          020 $  0 010 = 36     CONVERT FIRST SUBSCRIPT
G014E510                1         2
```

```
G014E520          020 P 29 001 * 10     JUMP IF NO 2ND SUBSCRIPT
G014E530          020 A 59 050 P 11
G014E540          020 $  0 010 = 66     INSERT COMMA
G014E550          020 P 29 070   12
G014E560          050 H 20 010 $  1
G014E570          020 $  0 010 = 36     CONVERT SECOND SUBSCRIPT
G014E580                   1        2
G014E590     * 10 020 J 20 050 P 11     =77 LEFT RIGHT GROUPER IN J20
G014E600          020 $  0 010 = 66     INSERT RIGHT GROUPER
G014E610     * 11 020 $  0 010 = 62     INSERT SPACE
G014E620          020 A 43 050 P 11
G014E630          020 $  0 010 = 66     INSERT '='
G014E635          020 $  0 010 = 62     INSERT SPACE
G014E640          020 A131 050 P 11
G014E650          020 $  0 010 = 66     INSERT EOM
G014E660          020 P  0 025 I  1     COMPENSATE FOR =25 COUNTING UP ONE
G014E670          050 P  0 010 $  1
G014E680          020 I  3 050 P 13     SET SUBSTATE TO EXPECT VALUE
G014E690          010 X 12              EXIT TO TRANSMIT AND SWITCH
G014E700     * 50 020 I  0 050 P 13     RESET SUBSTATE TO ZERO
G014E705          020 Q 12 050 Q  8     RESTORE CONTROL PDL
G014E710          020 Q 13 050 Q  9
G014E715          020 K  4 050 Q 12
G014E720          050 Q 13 010 $  1
G014E721          020 P  1 124 A124
G014E722          025 A124 002 * 51     JUMP IF LAST CHARACTER IS '*'
G014E725          020 J 12 050 P  2     INITIALIZE POINTER FOR SCAN
G014E730          020 $  0 010 =  2     ADVANCE TO NONSPACE
G014E740          020 $  0 010 = 21     EVALUATE EXPRESSION
G014E750          020 P  1 001 * 53     JUMP IF TERMINAL MINUS
G014E752          124 A 27 025 A 27     ALLOW A PERIOD
G014E754          002 * 52 010 E  6     ERROR IF NOT A PERIOD
G014E756     * 52 020 $  0 010 =  1     ADVANCE ONE
G014E758          001 * 53 010 E  6     ERROR IF NOT TERMINAL MINUS
G014E760     * 53 020 P 27 050 H 27     SET LETTER
G014E770          020 P 28 050 H 28     SET INDEX
G014E780          020 P 29 050 H 29     SET INDEX
G014E790          020 Q  3 050 H 30     SET VALUE
G014E800          020 $  0 010 = 13     POP Q3
G014E810          020 $  0 010 = 73     ASSIGN VALUE
G014E820          010 X  5              EXIT TO X5 ADVANCE
G014E830     * 51 020 Q  9 050 P 21     SET STEP (MUST EXIST)
G014E840          020 $  0 010 = 51     FIND CURRENT STEP (MUST EXIST)
G014E850          050 P 16 010 $  1     SET P16
G014E860          020 $  0 010 =  0     READ CLOCK
G014E870          050 H  8 010 X  6     SET H8 START TIME AND EXIT TO FETCH
G014E880     * 90                       LOCAL STORAGE
G014E890     * 91+          10   27,
G014E999        /     G 14 010,7000/
```

```
H---E000+     H  0                                LAST 16 BIT CLOCK READING
H---E001      H  1                                TIME OF DAY (IN CLOCK COUNTS)
H---E002      H  2                                DATE
H---E003      H  3                                DATE (CONTINUED)
H---E004      H  4 100 0000 000 0000              STN DUE FOR Q MSG
H---E005      H  5                          0     TIME TO LOG
H---E006      H  6                          0     TIME TO R0
H---E007      H  7      ----      ----            STN IN CORE
H---E008      H  8                                START TIME OF CURRENT SHOT
H---E009      H  9                                SCR CONTENTS
H---E010      H 10         0          0           NEXT AVAILABLE BUFFER
H---E011      H 11         0 000      0           NEXT AVAILABLE DRUM SLOT
H---E012      H 12      ----      ----            STN
H---E013      H 13      ----      ----            BUFFER
H---E014      H 14      ----      ----            DRUM
H---E015      H 15 100 0000 000 0000              += 'OF' DELAYED A/C READING
H---E016      H 16 100 0000 000 0000              += 'OF' DELAYED A/C PUNCHING
H---E017      H 17                                CURRENT DRUM ROUTINE CONTROL WORD
H---E018      H 18                                TEMP FOR =21, =26.
H---E019      H 19                                ACCUMULATED CHARACTERS RESULT OF =27
H---E020      H 20                                INTEGER FOR OUTPUT
H---E021      H 21                                SEARCH POINTER
H---E022      H 22                                SEARCH POINTER
H---E023      H 23                                SEARCH POINTER
H---E024      H 24                                SEARCH POINTER
H---E025      H 25                                SEARCH POINTER
H---E026      H 26                                SEARCH POINTER
H---E027      H 27                                LETTER
H---E028      H 28                                FIRST INDEX
H---E029      H 29                                SECOND INDEX
H---E030      H 30                                VALUE
H---E031      H 31 000,1000,012,1037              DRUM WORD FOR ERROR MSG INDEX
```

```
I---E000+     I  0          0      SMALL INTEGERS
I---E001      I  1          1
I---E002      I  2          2
I---E003      I  3          3
I---E004      I  4          4
I---E005      I  5          5
I---E006      I  6          6
I---E007      I  7          7
I---E008      I  8          8
I---E009      I  9          9
```

J---E000+	J 0			60	MINUTES PER HOUR
J---E001	J 1			1800	COUNTS PER MINUTE
J---E002	J 2			55	LINE NUMBER FOR EJECT
J---E003	J 3			15	PREFERRED PROCESSING PERIOD
J---E004	J 4			30	SHOT TIME FOR PROCESSING
J---E005	J 5			120	SHOT TIME FOR READING
J---E006	J 6			120	SHOT TIME FOR PUNCHING
J---E007	J 7			450	OVERDUE TIME FOR PROCESSING
J---E008	J 8			300	OVERDUE TIME FOR READING
J---E009	J 9			300	OVERDUE TIME FOR PUNCHING
J---E010	J 10	Q	11	Q 10	
J---E011	J 11	N	0	N 0	
J---E012	J 12	R	0	R 0	
J---E013	J 13	A	0	A 0	
J---E014	J 14	N	8	N 8	
J---E015	J 15				
J---E016	J 16			4	CHOKE NUMBER
J---E017	J 17			1	UNCHOKE NUMBER
J---E018	J 18+			9 8	CONSTANT
J---E019	J 19				
J---E020	J 20				COMMUNICATION CELL =77 TO D0 AND G14
J---E021	J 21			----	NUMBER OF INPUTS
J---E022	J 22			----	NUMBER OF OUTPUTS
J---E023	J 23			----	TIME WORKED

```
K---E000+    K  0 000,0000,000,0000    ZERO   (GENERAL CONSTANTS)
K---E001     K  1 000,0000,000,0001    ADDRESS MODIFIER
K---E002     K  2 000,0001,000,0000    ADDRESS MODIFIER
K---E003     K  3 000,0001,000,0001    ADDRESS MODIFIER
K---E004     K  4 100,0000,000,0000    SIGN BIT
K---E005     K  5 000,0000,000,7777    EXTRACTOR RIGHT ADDRESS
K---E006     K  6 000,7777,000,0000    EXTRACTOR LEFT ADDRESS
K---E007     K  7 000,7777,000,7777    EXTRACTOR BOTH ADDRESSES
K---E008     K  8 000,0000,177,0000    MASK INPUT MESSAGES
K---E009     K  9 000,0000,020,0000    COUNTER FOR EXTENDED CLOCK
K---E010     K 10 000,0000,033,0000    MASK FOR DRUM SECTION CODE
K---E011     K 11 077,6000,000,0000    EXTRACTOR SFX
K---E012     K 12 000,1000,000,0000    EXTRACTOR SIGN(CF)
K---E013     K 13 000,0777,777,7777    EXTRACTOR MAG(CF)
K---E014     K 14 000,2000,000,0000    UNIT FOR SFX (AND TEST IN =27)
K---E015     K 15 177,7777,777,7777    ALL ONES
K---E016     K 16 000,0017,000,0000    BUFFER MASK
K---E017     K 17 000,0000,000,0177    CHARACTER CODE MASK AND STN MASK.
K---E018     K 18 000,0017,000,0177    BUFFER + STN MASK
K---E019     K 19 000,0000,000,0400    BIT OF DISTINCTION
K---E020     K 20 000,0000,000,0017    EXTRACTOR NUMERIC CODE
K---E021     K 21 040,0000,000,0000    EN BIT
K---E022     K 22 020,0000,000,0000    DS BIT
K---E023     K 23 010,0000,000,0000    RO BIT
K---E024     K 24 004,0000,000,0000    TL BIT OR GREATER
K---E025     K 25 002,0000,000,0000    CL BIT OR EQUAL OR MATRIX
K---E026     K 26 001,0000,000,0000    SU BIT OR LESS OR VECTOR
K---E027     K 27 000,0000,400,0000    *      OR FUNCTION OR SUBSCRIPT
K---E028     K 28 000,0000,200,0000    MPY, /
K---E029     K 29 000,0000,100,0000    ON BIT OR +, -
K---E030     K 30 000,0000,040,0000    OF BIT OR ARITH OPERATOR
K---E031     K 31 000,0000,020,0000    TC BIT OR LEFT GROUPER
K---E032     K 32 000,0000,010,0000    RI BIT OR RIGHT GROUPER
K---E033     K 33 000,0000,004,0000    RC BIT OR RELATION
K---E034     K 34 000,0000,002,0000    EJ BI) OR LETTER
K---E035     K 35 000,0000,001,0000    TO BIT OR DIGIT
K---E036     K 36 003,0000,000,0000    CL+SU OR CONSTANT
K---E037     K 37 000,0000,700,0000    PRECEDENCE BITS
K---E038     K 38 102,0000,000,0000    MATRIX HEAD
K---E039     K 39 101,0000,000,0000    VECTOR HEAD
K---E040     K 40 000,0000,177,0000    MAJOR CYCLE MSGS
K---E041     K 41 000,0000,177,0000    MINOR CYCLE MSGS
K---E042     K 42 000,0000,177,0000    MASK FOR INDEX
K---E043     K 43         3         3
```

```
L---E000+     L  0          0          0      Q FOR ENABLE
L---E001      L  1 100 0000 000 0000          L ENABLED AND IDLE
L---E002      L  2 100 0000 000 0000          Q FOR CARD READER
L---E003      L  3 100 0000 000 0000          S WAITING FOR READER-READY SIGNAL
L---E004      L  4 100 0000 000 0000          S READING CARDS
L---E005      L  5 100 0000 000 0000          Q FOR DRUM ASSIGNMENTS
L---E006      L  6 100 0000 000 0000          L DRAINING BUFFERS BEFORE CL+SU
L---E007      L  7 100 0000 000 0000          Q FOR INPUT BUFFER AND CL+SU
L---E008      L  8 100 0000 000 0000          Q FOR CARD PUNCH
L---E009      L  9 100 0000 000 0000          S PUNCHING CARDS
L---E010      L 10 100 0000 000 0000          L GREEN (P13 HOLDS SUBSTATE)
L---E011      L 11 100 0000 000 0000          Q FOR BUFFER AND PREFERRED PROCESSING
L---E012      L 12 100 0000 000 0000          Q FOR PROCESSING
L---E013      L 13 100 0000 000 0000          S PROCESSING
L---E014      L 14 100 0000 000 0000          L CHOKED
L---E015      L 15 100 0000 000 0000
```

N---E000+	N	0+	1	39	POWERS OF TEN.
N---E001	N	1+	10	39	
N---E002	N	2+	100	39	
N---E003	N	3+	1000	39	
N---E004	N	4+	10000	39	
N---E005	N	5+	100000	39	
N---E006	N	6+	1000000	39	
N---E007	N	7+	10000000	39	
N---E008	N	8+	100000000	39	
N---E009	N	9+	1000000000	39	
N---E010	N	10+	10000000000	39	
N---E011	N	11+	100000000000	39	

```
P---E000+    P  0                          --    LINE NUMBER ON TYPEWRITER PAGE
P---E001     P  1 --- ---- --- ----         ----  PRESENT CHARACTER WORD IN R
P---E002     P  2      ----      ----         ----  ADDRESS OF PRESENT CHARACTER WORD IN
P---E003     P  3                ----         ----  LOCN FOR PACKING OR UNPACKING
P---E004     P  4                              SFX   (EXTENDED TO SIGN POSITION)
P---E005     P  5                              SIGN   (EXTRACTED)
P---E006     P  6                              CF   (MAGNITUDE)
P---E007     P  7                              SFX   (EXTENDED TO SIGN POSITION)
P---E008     P  8                              SIGN   (EXTRACTED)
P---E009     P  9                              CF   (MAGNITUDE)
P---E010     P 10      ----      ----         ----  LOCN OF SPACE AFTER 'IF', ELSE -1.
P---E011     P 11 --- ---- --- ----         ----  PRESENT CHARACTER WORD IN S
P---E012     P 12      ----      ----         ----  ADDRESS OF PRESENT CHARACTER WORD INS
P---E013     P 13                          2     SUB(L10)=0 INST, 1 FORM, 2 INIT, 3 DE
P---E014     P 14 010 ----                       LINK FROM =25 (OUTPUT LINE)
P---E015     P 15 100 0000 000 0000              + = RI WHILE L11 OR L14
P---E016     P 16 100 0000 000 0000              + = LOCN OF CURRENT STEP IN RT ADDR
P---E017     P 17 100 0000 000 0000              + = STOPPED (USED BY G13 'GO')
P---E018     P 18                                WORKING STORE
P---E019     P 19                                WORKING STORE
P---E020     P 20                                CONTEXT CELL FOR PART ETC.
P---E021     P 21                                CONTEXT CELL FOR STEP ETC.
P---E022     P 22                                CONTEXT CELL FOR FORM ETC.
P---E023     P 23 100 0000 000 0000              + = PROBLEM IN X7
P---E024     P 24 000 ---- --- ----              DRUM ROUTINE CONTROL WORD FOR =78
P---E025     P 25 010 ----                       LINK FROM =78 (TRANSMIT LINE)
P---E026     P 26 010 ----                       LINK FOR TYPE ROUTINES
P---E027     P 27                                WORKING STORE
P---E028     P 28                                WORKING STORE
P---E029     P 29                                WORKING STORE
P---E030     P 30                                WORKING STORE
P---E031     P 31                                WORKING STORE
```

Q---E000+	Q	0				----	AVAILABLE SPACE PDL LINK
Q---E001	Q	1				----	SPARE TANK PDL LINK
Q---E002	Q	2	100	0000	000	0000	OPERAND PDL LINK
Q---E003	Q	3	100	0000	000	0000	OPERAND
Q---E004	Q	4	100	0000	000	0000	OPERATOR PDL LINK
Q---E005	Q	5	100	0000	000	0000	OPERATOR
Q---E006	Q	6	100	0000	000	0000	AUXILIARY PDL LINK
Q---E007	Q	7	100	0000	000	0000	AUXILIARY ITEM
Q---E008	Q	8	100	0000	000	0000	CONTROL PDL LINK
Q---E009	Q	9	100	0000	000	0000	CONTROL (STEP NUMBER)
Q---E010	Q	10	000	0000	000	0000	LIST OF STEPS (AND PARTS)
Q---E011	Q	11	000	0000	000	0000	LIST OF FORMS
Q---E012	Q	12	100	0000	000	0000	HIDEOUT FOR (Q8)
Q---E013	Q	13	100	0000	000	0000	HIDEOUT FOR (Q9)
Q---E014	Q	14	000	----	000	0000	TEMPORARY LIST STRUCTURE
Q---E015	Q	15	000	0000	000	----	TEMPORARY LINEAR LIST

```
R---E000+     R  0 100 0000 000 0000      INPUT AND INTERPRETATION BLOCK
R---E001      R  1
R---E002      R  2
R---E003      R  3
R---E004      R  4
R---E005      R  5
R---E006      R  6
R---E007      R  7
R---E008      R  8
R---E009      R  9
R---E010      R 10
R---E011      R 11
R---E012      R 12
R---E013      R 13
R---E014      R 14
R---E015      R 15
R---E016      R 16
R---E017      R 17
R---E018      R 18
R---E019      R 19
R---E020      R 20
R---E021      R 21
R---E022      R 22
R---E023      R 23
R---E024      R 24
R---E025      R 25
R---E026      R 26
R---E027      R 27
R---E028      R 28
R---E029      R 29
R---E030      R 30
R---E031      R 31
R---E032      R 32
R---E033      R 33
R---E034      R 34
R---E035      R 35
R---E036      R 36
R---E037      R 37
R---E038      R 38
R---E039      R 39
R---E040      R 40
R---E041      R 41
R---E042      R 42
R---E043      R 43
R---E044      R 44
R---E045      R 45
R---E046      R 46
R---E047      R 47
R---E048      R 48
R---E049      R 49
R---E050      R 50
R---E051      R 51
R---E052      R 52
R---E053      R 53
```

R---E054 R 54
R---E055 R 55
R---E056 R 56
R---E057 R 57
R---E058 R 58
R---E059 R 59
R---E060 R 60
R---E061 R 61
R---E062 R 62
R---E063 R 63
R---E064 R 64
R---E065 R 65
R---E066 R 66
R---E067 R 67
R---E068 R 68
R---E069 R 69
R---E070 R 70
R---E071 R 71
R---E072 R 72
R---E073 R 73
R---E074 R 74
R---E075 R 75
R---E076 R 76
R---E077 R 77
R---E078 R 78
R---E079 R 79
R---E080 R 80
R---E081 R 81

```
S---E000+     S  0     S  0     S  0     OUTPUT BLOCK
S---E001      S  1
S---E002      S  2
S---E003      S  3
S---E004      S  4
S---E005      S  5
S---E006      S  6
S---E007      S  7
S---E008      S  8
S---E009      S  9
S---E010      S 10
S---E011      S 11
S---E012      S 12
S---E013      S 13
S---E014      S 14
S---E015      S 15
S---E016      S 16
S---E017      S 17
S---E018      S 18
S---E019      S 19
S---E020      S 20
S---E021      S 21
S---E022      S 22
S---E023      S 23
S---E024      S 24
S---E025      S 25
S---E026      S 26
S---E027      S 27
S---E028      S 28
S---E029      S 29
S---E030      S 30
S---E031      S 31
S---E032      S 32
S---E033      S 33
S---E034      S 34
S---E035      S 35
S---E036      S 36
S---E037      S 37
S---E038      S 38
S---E039      S 39
S---E040      S 40
S---E041      S 41
S---E042      S 42
S---E043      S 43
S---E044      S 44
S---E045      S 45
S---E046      S 46
S---E047      S 47
S---E048      S 48
S---E049      S 49
S---E050      S 50
S---E051      S 51
S---E052      S 52
S---E053      S 53
```

```
S---E054      S 54
S---E055      S 55
S---E056      S 56
S---E057      S 57
S---E058      S 58
S---E059      S 59
S---E060      S 60
S---E061      S 61
S---E062      S 62
S---E063      S 63
S---E064      S 64
S---E065      S 65
S---E066      S 66
S---E067      S 67
S---E068      S 68
S---E069      S 69
S---E070      S 70
S---E071      S 71
S---E072      S 72
S---E073      S 73 100 0000 000,4052      (NEEDED FOR D7)
S---E074      S 74 100 0000 000,4052      (NEEDED FOR D7)
```

```
T---E000+     T  0
T---E001      T  1
T---E002      T  2
T---E003      T  3
T---E004      T  4
T---E005      T  5
T---E006      T  6
T---E007      T  7
```

TEMPORARY STORAGE REGION
(NOT GUARANTEED OVER SUBROUTINES)

```
U000E000+     U  0     $  0     $  0     STN TABLE -- STATE
U000E001      *  0     L  0     L  0
U000E002      *  1     L  0     L  0
U000E003      *  2     L  0     L  0
U000E004      *  3     L  0     L  0
U000E005      *  4     L  0     L  0
U000E006      *  5     L  0     L  0
U000E007      *  6     L  0     L  0
U000E008      *  7     L  0     L  0
U000E009      *  8     L  0     L  0
U000E010      *  9     L  0     L  0,
```

```
U001E000+     U  1      $  0      $  0      STN TABLE -- CURRENT BUFFER
U001E001      *  0 100 0000 000 0000
U001E002      *  1 100 0000 000 0000
U001E003      *  2 100 0000 000 0000
U001E004      *  3 100 0000 000 0000
U001E005      *  4 100 0000 000 0000
U001E006      *  5 100 0000 000 0000
U001E007      *  6 100 0000 000 0000
U001E008      *  7 100 0000 000 0000
U001E009      *  8 100 0000 000 0000
U001E010      *  9 100 0000 000 0000,
```

```
U002E000+     U   2       $  0       $  0      STN TABLE -- DRUM ASSIGNMENT
U002E001      *   0 100 0000 000 0000
U002E002      *   1 100 0000 000 0000
U002E003      *   2 100 0000 000 0000
U002E004      *   3 100 0000 000 0000
U002E005      *   4 100 0000 000 0000
U002E006      *   5 100 0000 000 0000
U002E007      *   6 100 0000 000 0000
U002E008      *   7 100 0000 000 0000
U002E009      *   8 100 0000 000 0000
U002E010      *   9 100 0000 000 0000,
```

```
U003E000+     U   3       $  0       $  0      STN TABLE -- NEXT STN IN SAME STATE
U003E001      *   0          1          1
U003E002      *   1          2          2
U003E003      *   2          3          3
U003E004      *   3          4          4
U003E005      *   4          5          5
U003E006      *   5          6          6
U003E007      *   6          7          7
U003E008      *   7          8          8
U003E009      *   8          9          9
U003E010      *   9 100 0000 000 0000,
```

```
U004E000+     U   4       $  0       $  0      STN TABLE -- TIME OF LAST ACTIVITY
U004E001      *   0 100 0000 000 0000
U004E002      *   1 100 0000 000 0000
U004E003      *   2 100 0000 000 0000
U004E004      *   3 100 0000 000 0000
U004E005      *   4 100 0000 000 0000
U004E006      *   5 100 0000 000 0000
U004E007      *   6 100 0000 000 0000
U004E008      *   7 100 0000 000 0000
U004E009      *   8 100 0000 000 0000
U004E010      *   9 100 0000 000 0000,
```

```
U005E000+     U  5      $  0      $  0     BUFFER TABLE--NEXT OPEN BUFFER
U005E001      *  0         1         1
U005E002      *  1         2         2
U005E003      *  2         3         3
U005E004      *  3         4         4
U005E005      *  4         5         5
U005E006      *  5         6         6
U005E007      *  6         7         7
U005E008      *  7         8         8
U005E009      *  8         9         9
U005E010      *  9        10        10
U005E011      * 10        11        11
U005E012      * 11        12        12
U005E013      * 12        13        13
U005E014      * 13        14        14
U005E015      * 14        15        15
U005E016      * 15 100 0000 000 0000,
```

```
U006E000+     U  6      $  0      $  0     DRUM SLOT TABLE--NEXT OPEN SLOT
U006E001      *  0         1 001     1
U006E002      *  1         2 002     2
U006E003      *  2         3 003     3
U006E004      *  3         4 020     4
U006E005      *  4         5 021     5
U006E006      *  5         6 022     6
U006E007      *  6         7 023     7
U006E008      *  7 100 0000 000 0000,
```

```
U007E000+     U  7      $  0      $  0      DRUM TABLE--INITIALS
U007E001      *  0 005,4130,270,0000
U007E002      *  1 005,4130,270,0000
U007E003      *  2 005,4130,270,0000
U007E004      *  3 005,4130,270,0000
U007E005      *  4 005,4130,270,0000
U007E006      *  5 005,4130,270,0000
U007E007      *  6 005,4130,270,0000
U007E008      *  7 005,4130,270,0000
U007E009      *  8 005,4130,270,0000
U007E010      *  9 005,4130,270,0000,
```

```
V---E000+    V  0      V  0      V  0     VARIABLE TABLE (ORIGIN)
V---E001     V  1 100 0000 000 0000     LCA
V---E002     V  2 100 0000 000 0000     LCB
V---E003     V  3 100 0000 000 0000     LCC
V---E004     V  4 100 0000 000 0000     LCD
V---E005     V  5 100 0000 000 0000     LCE
V---E006     V  6 100 0000 000 0000     LCF
V---E007     V  7 100 0000 000 0000     LCG
V---E008     V  8 100 0000 000 0000     LCH
V---E009     V  9 100 0000 000 0000     LCI
V---E010     V 10 100 0000 000 0000     LCJ
V---E011     V 11 100 0000 000 0000     LCK
V---E012     V 12 100 0000 000 0000     LCL
V---E013     V 13 100 0000 000 0000     LCM
V---E014     V 14 100 0000 000 0000     LCN
V---E015     V 15 100 0000 000 0000     LCO
V---E016     V 16 100 0000 000 0000     LCP
V---E017     V 17 100 0000 000 0000     LCQ
V---E018     V 18 100 0000 000 0000     LCR
V---E019     V 19 100 0000 000 0000     LCS
V---E020     V 20 100 0000 000 0000     LCT
V---E021     V 21 100 0000 000 0000     LCU
V---E022     V 22 100 0000 000 0000     LCV
V---E023     V 23 100 0000 000 0000     LCW
V---E024     V 24 100 0000 000 0000     LCX
V---E025     V 25 100 0000 000 0000     LCY
V---E026     V 26 100 0000 000 0000     LCZ
V---E027     V 27 100 0000 000 0000     A
V---E028     V 28 100 0000 000 0000     B
V---E029     V 29 100 0000 000 0000     C
V---E030     V 30 100 0000 000 0000     D
V---E031     V 31 100 0000 000 0000     E
V---E032     V 32 100 0000 000 0000     F
V---E033     V 33 100 0000 000 0000     G
V---E034     V 34 100 0000 000 0000     H
V---E035     V 35 100 0000 000 0000     I
V---E036     V 36 100 0000 000 0000     J
V---E037     V 37 100 0000 000 0000     K
V---E038     V 38 100 0000 000 0000     L
V---E039     V 39 100 0000 000 0000     M
V---E040     V 40 100 0000 000 0000     N
V---E041     V 41 100 0000 000 0000     O
V---E042     V 42 100 0000 000 0000     P
V---E043     V 43 100 0000 000 0000     Q
V---E044     V 44 100 0000 000 0000     R
V---E045     V 45 100 0000 000 0000     S
V---E046     V 46 100 0000 000 0000     T
V---E047     V 47 100 0000 000 0000     U
V---E048     V 48 100 0000 000 0000     V
V---E049     V 49 100 0000 000 0000     W
V---E050     V 50 100 0000 000 0000     X
V---E051     V 51 100 0000 000 0000     Y
V---E052     V 52 100 0000 000 0000     Z
```

```
W---E000+     W  0 000,0000,001,1414     ALL
W---E001      W  1 000,0002,411,1505     TIME
W---E002      W  2 000,0000,006,1722     FOR
W---E003      W  3 000,0000,617,2215     FORM
W---E004      W  4 000,0061,722,1523     FORMS
W---E005      W  5 000,0000,000,1106     IF
W---E006      W  6 000,0000,000,1116     IN
W---E007      W  7 000,0002,001,2224     PART
W---E008      W  8 000,0200,122,2423     PARTS
W---E009      W  9 000,0002,324,0520     STEP
W---E010      W 10 000,0232,405,2023     STEPS
W---E011      W 11 000,0252,305,2223     USERS
W---E012      W 12 002,6011,425,0523     VALUES
W---E013      W 13 000,0002,311,3205     SIZE
W---E014      W 14 000,0000,000,1722     OR
W---E015      W 15 000,0000,001,1604     AND
```

```
Z---E000+     Z  0 177,7777,777,7777     USER'S INITIALS (8-BIT CHAR STRING)
```

```
X000E000+       $                       M  0    LOGGING ROUTINE
X000E005    *   0 013 $   1 010 *  10           SKIP IF T3 OFF
X000E010          020 H   1 024 *   1           UPDATE TIME TO R0
X000E015          050 H   6 010 *   2
X000E020    *   1                  300          TEN SECOND INTERVAL FOR R0'S
X000E025    *   2 120     0 050 T   0
X000E030    *   3 020 T   0 024 K  23           R0 LOOP
X000E035          142 K  23 020 T   0
X000E040          024 K   1 050 T   0
X000E045          025 N   1 001 *   3
X000E050    *  10 020 H   1 025 H   5
X000E055          001 X   1 120     0
X000E060          004 H   1 044 J   1
X000E065          060 *  64 032 J   1
X000E070          020 J   1 064 H   5
X000E075          120     0 050 T   0           CLEAR IMAGE IN R AND S
X000E080    *  11 020 T   0 024 *  13
X000E085          054 *  12 120     0
X000E090    *  12 050 ---- 050 ----
X000E095          020 T   0 024 K   3
X000E100          050 T   0 025 *  14
X000E105          001 *  11 010 *  20
X000E110    *  13      S   0      R   0
X000E115    *  14        41         41
X000E120    *  20 120     0 004 *  64
X000E130          044 J   0 050 *  64
X000E135          036 N   2 060 H  20
X000E140          020 K   4 050 *  91
X000E145          020 $   0 010 *  70
X000E150          000                 4
X000E155          020 *  91 070      1
X000E160          050 *  91 020 H   2
X000E165          071    48 050 *  21
X000E170          075     8 050 *  22
X000E175          075    16 050 *  24
X000E180          075     8 050 *  25
X000E185          004 H   3 075     16
X000E190          050 *  27 075      8
X000E195          050 *  28 010 *  30
X000E200    *  21 --- ---- --- ----
X000E205    *  22 --- ---- --- ----
X000E210    *  23                  11
X000E215    *  24 --- ---- --- ----
X000E220    *  25 --- ---- --- ----
X000E225    *  26                  11
X000E230    *  27 --- ---- --- ----
X000E235    *  28 --- ---- --- ----
X000E240          100 0000 000 0000
X000E245    *  29     * 21      * 21
X000E250    *  30 020 * 29 052 * 31
X000E255    *  31 020 ---- 005 * 34
X000E260          124 K  20 024 *  95
X000E265          056 *  32 010 $   1
X000E270    *  32 056 *  33 020 ----
```

```
X000E275     * 33 024 * 91 050 ----
X000E280          020 * 31 024 K  2
X000E285          052 * 31 020 * 91
X000E290          070    1 050 * 91
X000E295     * 34 010 * 31 020 * 91
X000E300          070    2 050 * 91
X000E305     * 40 120    0 050 H 20      CONVERT NUMBER OF SLOTS ASSIGNED
X000E310          050 T  0 010 $  1
X000E315     * 41 020 T  0 024 U  2
X000E320          056 * 42 014 * 42
X000E325     * 42             020 ----
X000E330          001 * 43 020 H 20
X000E335          024 K  1 050 H 20
X000E340     * 43 020 T  0 024 K  1
X000E345          050 T  0 025 N  1
X000E350          001 * 41 010 $  1
X000E355          020 H 20 050 * 97
X000E360          020 $  0 010 * 70
X000E365          010                2
X000E370          120    0 050 H 20      CONVERT NUMBER OF GREENS
X000E375          020 L 10 050 T  0
X000E380     * 45 020 T  0 001 * 47
X000E385          024 U  3 052 * 46
X000E390     * 46 020 ---- 050 T  0
X000E395          020 H 20 024 K  1
X000E400          050 H 20 010 * 45
X000E405     * 47 020 * 97 025 H 20
X000E410          050 * 90 010 $  1
X000E415          020 $  0 010 * 70
X000E420          010                2
X000E425          020 * 90 050 H 20      CONVERT NUMBER WORKING
X000E430          020 $  0 010 * 70
X000E435          010                2
X000E440          120    0 050 H 20      CONVERT NUMBER IN QUEUE
X000E445          020 L  5 050 T  0
X000E450     * 48 020 T  0 001 * 44
X000E455          024 U  3 052 * 49
X000E460     * 49 020 ---- 050 T  0
X000E465          020 H 20 024 K  1
X000E470          050 H 20 010 * 48
X000E475     * 44 020 $  0 010 * 70
X000E480          010                2
X000E485          020 * 91 070    3
X000E490          050 * 91 010 $  1
X000E495          020 J 21 050 H 20      CONVERT NUMBER OF INPUTS
X000E500          120    0 050 J 21
X000E505          020 $  0 010 * 70
X000E510          010                3
X000E515          020 J 22 050 H 20      CONVERT NUMBER OF OUTPUTS
X000E520          120    0 050 J 22
X000E525          020 $  0 010 * 70
X000E530          010                4
X000E535          020 J  1 072    1      CONVERT EFFICIENCY IN PER CENT
X000E540          004 J 23 036 N  2
X000E545          044 J  1 060 H 20
```

```
X000E550           120     0 050 J 23
X000E555           020 $   0 010 * 70
X000E560           010                4
X000E565     * 50  120     0 050 H 20      CONVERT INITIALS FOR STATIONS
X000E570     * 51  020 $   0 010 * 80
X000E575           020 H 20 024 K  1
X000E580           050 H 20 025 N  1
X000E585           001 * 51 010 $  1
X000E590     * 60  100     5 010 $  1      COPY PRINT IMAGE
X000E595           120     0 050 T  0
X000E600     * 61  020 T   0 024 * 13
X000E605           054 * 62 010 $  1
X000E610     * 62  020 ---- 101 ----
X000E615           020 T   0 024 K  3
X000E620           050 T   0 025 * 14
X000E621           001 * 61 010 $  1
X000E622           020 * 64 024 K  1
X000E623           025 J   0 002 * 63
X000E624           106     1 010 * 65
X000E625     * 63  106     0 010 * 65      PAGE IF NEXT ENTRY IS ON THE HOUR
X000E626     * 64                          WORKING STORAGE
X000E630     * 65  013 $   1 010 X  1
X000E631           023 * 97 001 X  1       EXIT UNLESS NO USERS
X000E632           130 X   1               HALT IF T3 AND NO USERS
X000E635     * 70  024 K   2 052 * 71      SUBROUTINE TO CONVERT INTEGER
X000E640           024 K   2 052 * 79
X000E645     * 71  020 ---- 051 * 75
X000E650           124 K   5 025 K  1
X000E655           050 T   0 010 $  1
X000E660           020 K   4 050 T  1
X000E665     * 72  020 T   0 024 J 11
X000E670           056 * 73 120     0
X000E675     * 73  004 H 20 044 ----
X000E680           050 H 20 060 T  2
X000E685           025 K   1 001 * 74
X000E690           050 T   1 014 * 75
X000E695     * 74  023 T   0 006 * 75
X000E700           020 T   1 006 * 75
X000E705     * 75  --- * 78 020 T  2
X000E710           024 * 95 056 * 76
X000E715     * 76  056 * 77 020 ----
X000E720     * 77  024 * 91 050 ----
X000E725     * 78  020 * 91 070     1
X000E730           050 * 91 020 T  0
X000E735           025 K   1 050 T  0
X000E740     * 79  001 ---- 010 * 72
X000E745     * 80  024 K   2 052 * 89      SUBROUTINE TO CONVERT INITIALS
X000E750           020 H 20 024 U  7
X000E755           056 * 82 020 H 20
X000E760           071     2 056 * 81
X000E765     * 81  020 K   4 070 ----
X000E770     * 82  050 * 91 020 ----
X000E775           050 T   0 010 $  1
X000E780     * 83  020 T   0 001 * 89
X000E785           071     2 001 * 85
```

```
X000E790           071     1 001 * 84
X000E795           120     0 014 * 86
X000E800     * 84  020 * 92 014 * 86
X000E805     * 85  071     1 001 * 86
X000E810           020 * 93 014 * 86
X000E815     * 86  020 * 94 050 T  1
X000E820           020 T  0 071     4
X000E825           071     4 050 T  0
X000E830           020 T  1 064 T  1
X000E835           024 * 96 056 * 87
X000E840     * 87  056 * 88 020 ----
X000E845     * 88  024 * 91 050 ----
X000E850           020 * 91 070     1
X000E855           050 * 91 010 * 83
X000E860     * 89  010 ----
X000E865     * 90                      WORKING STORAGE
X000E870     * 91                      STROBE
X000E875     * 92                 14
X000E880     * 93                 23
X000E885     * 94                 31
X000E890     * 95               R  0   BASE FOR LEFT IMAGE
X000E895     * 96               S  0   BASE FOR RIGHT IMAGE
X000E900     * 97                  ,   NUMBER OF USERS
X000E999       /      X  0 010,7000/
```

```
X001E000+    X  1 131 $  0                    QUEUE SERVICE (HALT HERE IF H1)
X001E010         020 $  0 010 =  0            READ CLOCK
X001E020         013 *  1 020 L  0            JUMP IF T3 FOR WINDUP
X001E030         002 X 11 010 *  2            JUMP IF L0 FOR ENABLE, ELSE SKIP
X001E040    *  1 025 H  6 002 X  0            JUMP IF TIME TO R0 IN WINDUP
X001E050         020 L  1 002 X 11            JUMP IF L1 TO DISABLE STN
X001E060    *  2 020 L  5 001 *  3            JUMP IF NO L5 Q FOR DRUM SLOT
X001E070         020 H 11 002 X 11            JUMP IF DRUM SLOT AVAILABLE
X001E080    *  3 020 H  4 001 *  4            JUMP IF NO Q MESSAGE DUE
X001E090         020 H 10 002 X 11            JUMP IF BUFFER AVAILABLE FOR MSG
X001E100    *  4 020 H  1 025 H  5
X001E110         002 X  0 010 X  2,           JUMP IF TIME TO LOG, ELSE TO X2
```

```
X002E000+    X  2 010 $  0                    SIGNAL SERVICE
X002E010         120    0 145 K 40           SCAN FOR SIGNALS
X002E020         001 X  3 050 H  9           JUMP IF NO SIGNALS ELSE SET H9
X002E030         124 K 17 050 H 12
X002E040         071   21 052 H 12           SET H12 FOR SIGNALLING STATION
X002E050         020 H  9 071   21
X002E060         001 X 10 071    1           JUMP IF 'ON'
X002E070         001 X 10 071    1           JUMP IF 'OF'
X002E080         001 X 10 071    2           JUMP IF 'TC'
X002E090         001 * 20 071    1           JUMP IF 'RC'
X002E100         001 * 30 071    1           JUMP IF 'EJ'
X002E110         001 * 10 010 X 10           JUMP IF 'TO' ELSE ASSUME 'RI'
X002E120    * 10 020 $  0 010 = 48           'TO' TREAT BY SUBROUTINE
X002E130         010 X  2                    LOOP BACK TO SIGNAL SERVICE
X002E140    * 20 020 H 12 142 K 33           'RC' ERASE SIGNAL
X002E150         010 * 31                    MERGE WITH 'EJ' TREATMENT
X002E160    * 30 020 H 12 142 K 34           'EJ' ERASE SIGNAL
X002E170    * 31 020 $  0 010 = 54           BRING IN USER'S BLOCK
X002E180         020 $  0 010 = 47           CHANGE STATE TO 'PROCESSING'
X002E190              L 13    L 13
X002E200         020 H  9 141 R  1           READ BUFFER
X002E210         020 $  0 010 = 58           RELEASE BUFFER
X002E220         020 $  0 010 =  5           CONVERT
X002E230         020 H  9 071   26
X002E240         001 * 32 020 P  0           JUMP IF 'EJ' ELSE 'RC'
X002E250         024 I  1 050 P  0
X002E260         025 J  2 001 * 33           JUMP IF STILL ROOM ON PAGE
X002E270         020 * 32 010 = 22           EJECT, TYPE TIME LINE AND SKIP
X002E280    * 32 020 $  0 014 = 22           TYPE TIME LINE AFTER 'EJ'
X002E290    * 33 020 J 21 024 K  1           COUNT ONE INPUT
X002E292         050 J 21 010 $  1
X002E294         023 P 13 002 * 40           JUMP IF WAITING FOR INSTRUCTION
X002E300         024 I  1 002 * 70           JUMP IF WAITING FOR FORM
X002E310         024 I  1 002 * 35           JUMP IF WAITING FOR INITIALS
X002E320         024 I  1 002 G 14           JUMP IF WAITING FOR VALUE DEMANDED
X002E330         130 $  0                    ELSE PHONY SUBSTATE CODE
X002E335    * 35 050 H  9 010 X 10           CLEAR H9 AND JUMP
X002E340    * 40 020 R  1 005 X 12           JUMP IF EMPTY TO SWITCH VIA X12
X002E350         020 P  1 050 P 18           SAVE LAST CHARACTER
X002E360         124 A124 025 A124
X002E370         006 X 12 020 P  2           JUMP IF LINE ENDS WITH '*' TO SWITCH
X002E380         056 * 41 056 * 53           SET ADDRESSES FOR LAST CHARACTER
X002E390         020 J 12 050 P  2           INTIALIZE POINTER
X002E400         020 $  0 010 =  2           ADVANCE TO NONSPACE
X002E402         020 R  1 124 A 61
X002E404         025 A 61 002 D  5           JUMP IF TAB
X002E410         020 P  1 124 A124
X002E415         025 A124 006 X 12           JUMP IF '*'
X002E420         020 P  1 124 K 34
X002E430         025 K 34 001 * 50           JUMP IF NOT LETTER, ASSUME STEP LABEL
X002E440         020 P 18 124 A 27           ASSUME DIRECT COMMAND
X002E450         025 A 27 001 * 42           JUMP UNLESS LAST CHARACTER WAS PERIOD
X002E460    * 41 020 A109 050 ----           REPLACE BY TERMINAL PERIOD
X002E470    * 42 020 P 10 001 X  4           FIRE IF UNCONDITIONAL
```

```
X002E480          025 K 43 056 * 43
X002E485          025 P  2 001 E  6     ERROR IF NOTHING PRECEDES 'IF'
X002E490     * 43 020 A 45 050 ----     SET SPECIAL SPACE
X002E500          010 X  4               FIRE
X002E510     * 50 020 P  2 050 P 19
X002E515          020 $  0 010 = 20
X002E520          020 T  0 070   39     PICK UP DEC PT INFO FROM =20 IN TO
X002E525          024 P  2 025 P 19
X002E530          124 K  5 025 I  1
X002E535          025 N  1 002 E 11     ERROR IF 10 OR MORE DIGITS
X002E540          020 P  1 001 * 52     JUMP IF AT TERMINAL MINUS
X002E545          020 $  0 010 = 39     VERIFY SPACE AND ADVANCE TO NONSPACE
X002E550          020 P  2 025 K  3
X002E555          052 * 51 020 A131
X002E560     * 51 050 ---- 010 $  1
X002E570     * 52 020 $  0 010 = 45     VALIDATE STEP LABEL
X002E580          020 Q  3 050 P 21     SET STEP NUMBER IN CONTEXT
X002E590          020 $  0 010 = 13     POP Q3
X002E600          020 P 18 124 A 27
X002E610          025 A 27 001 * 54     JUMP UNLESS LAST CHARACTER WAS PERIOD
X002E620     * 53 020 A130 050 ----     REPLACE FINAL PERIOD BY PSEUDO-PERIOD
X002E630     * 54 020 P 10 001 * 56     JUMP IF NO CONDITION
X002E640          025 K 43 056 * 55
X002E650     * 55 020 A131 050 ----     SET SPECIAL SPACE BEFORE 'IF'
X002E660     * 56 020 $  0 010 =  6     PACK STRING
X002E670          020 Q  0 056 * 57     VERIFY TWO MORE SPACE UNITS
X002E680     * 57 001 E  0 020 ----
X002E690          001 E  0 010 $  1
X002E700          020 $  0 010 = 51     FIND STEP IF CAN
X002E705          001 * 58 010 $  1     JUMP IF CAN'T
X002E710          020 $  0 010 = 56     ERASE DUPLICATE
X002E715          020 $  0 010 = 51     FIND PLACE AGAIN
X002E720     * 58 020 H 23 002 * 62     JUMP IF PART EXISTS
X002E725          020 $  0 010 = 68     COMPUTE PART (P20) FOR STEP (P21)
X002E730          020 H 21 056 * 59     SET UP PART ID IN QUO STRUCTURE
X002E735          020 $  0 010 = 10
X002E740     * 59 056 * 60 056 ----
X002E745          024 K  1 056 * 61
X002E750     * 60 020 H 22 056 ----
X002E755     * 61 020 P 20 050 ----
X002E760          020 * 60 124 K  5
X002E765          071   21 050 H 24
X002E770          120    0 050 H 25
X002E775     * 62 020 H 24 052 * 63     SET UP STEP IN Q10 STRUCTURE
X002E780          020 $  0 010 = 10
X002E785          071   21 052 * 65
X002E790     * 63 052 ---- 024 K  2
X002E795          052 * 64 020 P 21
X002E800     * 64 050 ---- 020 H 25
X002E805          124 K  6 024 P  3
X002E810     * 65 050 ---- 120    0
X002E815          050 Q 15 014 X 12     RELIEVE Q15 AND SWITCH VIA X12
X002E820     * 70 120    0 050 P 13     CLEAR 'WTG FOR FORM' SUBSTATE
X002E825          020 $  0 010 =  6     PACK STRING
X002E830          020 $  0 010 = 52     FIND FORM IF CAN A/C P22
```

```
X002E835          001 * 71 010 $  1      JUMP IF CAN'T
X002E840          020 $  0 010 = 57      ERASE DUPLICATE
X002E845          020 $  0 010 = 52      FIND PLACE AGAIN
X002E850    * 71 020 H 24 052 * 72
X002E855          020 $  0 010 = 10
X002E860          071   21 052 * 74
X002E865    * 72 052 ---- 024 K  2
X002E870          052 * 73 020 P 22
X002E875    * 73 050 ---- 020 H 25
X002E880          124 K  6 024 P  3
X002E885    * 74 050 ---- 120    0
X002E890          050 Q 15 014 X 12,     RELIEVE Q15 AND SWITCH VIA X12
```

```
X003E000+     X   3 010 $   0                       TASK SELECTION
X003E010          011 *   1 010 *   2               TO *1 IF 'READER READY',ELSE TO *2
X003E020      *   1 020 L   3 001 *   2             JUMP IF NO STN WAITING FOR READER REA
X003E030          020 L   9 001 X   8               JUMP IF NO STN PUNCHING, INITIATE REA
X003E050      *   2 020 L   4 001 *   3             TO *3 IF NO STN READING
X003E060          024 U   4 052 $   1
X003E070          020 ---- 024 J   8
X003E080          025 H   1 005 X   8               TO X8 IF READING IS OVERDUE.
X003E090          010 *   5
X003E100      *   3 020 L   2 002 *  20             JUMP IF STN WAITING TO READ
X003E110          020 L   9 001 *   4               JUMP IF NO STN PUNCHING
X003E120          024 U   4 052 $   1
X003E130          020 ---- 024 J   9
X003E140          025 H   1 005 X   9               TO X9 IF PUNCHING IS OVERDUE
X003E150          010 *   5
X003E160      *   4 020 L   8 001 *   5             JUMP IF NO STN WAITING TO PUNCH
X003E170          020 L   4 001 X   9               JUMP IF NO STN READING, INITIATE PUNC
X003E180      *   5 020 L  12 001 *   6             TO *6 IF NO STN WTG FOR NORMAL PROCES
X003E190          024 U   4 052 $   1
X003E200          020 ---- 024 J   7
X003E210          025 H   1 001 *  40               TO *40 IF STN OVERDUE FOR PROCESSING
X003E220      *   6 020 L  11 001 *   7             TO *7 IF NO STN WTG FOR BUFFER
X003E230          020 H  10 002 *  50               TO *50 IF BUFFER AVAILABLE
X003E240      *   7 020 L  12 002 *  40             TO *40 IF STN WTG FOR PROCESSING
X003E250          020 L   4 006 X   8               JUMP IF STN READING
X003E260          020 L   9 006 X   9               JUMP IF STN PUNCHING
X003E270          010 X   1                         NOTHING TO DO,GO BACK TO Q SERVICE
X003E280      *  20 105     0 011 $   0             LOOP TILL T1 TURNED OFF
X003E500      *  40 020 L  12 050 H  12             SET STN CONTEXT
X003E510          020 $   0 010 =  47               CHANGE STATE TO L13
X003E520              L  13     L  13
X003E530          020 $   0 010 =  54               BRING IN PROGRAM
X003E540          010 X   4                         FIRE
X003E560      *  50 020 L  11 050 H  12             SET STN CONTEXT
X003E570          020 $   0 010 =  47               CHANGE STATE TO L13
X003E580              L  13     L  13
X003E590          020 $   0 010 =  54               BRING IN PROGRAM
X003E600          014 =  25                   ,     REENTER =25
```

```
X004E000+     X  4 010 $  0                    FIRE
X004E010           020 P 10 001 *  2           JUMP IF UNCONDITIONAL
X004E020           020 P  1 050 P 11           SAVE SETUP
X004E030           020 P  2 050 P 12
X004E040           020 A 14 050 P  1
X004E050           020 P 10 050 P  2
X004E060           020 $  0 010 = 23           EVALUATE CONDITION
X004E070           002 *  1 010 X  5           JUMP TO X5 ADVANCE IF DOESN'T HOLD
X004E100      *  1 020 P 11 050 P  1           RESTORE SETUP
X004E110           020 P 12 050 P  2
X004E120      *  2 020 $  0 010 = 27           INTERPRET VERB
X004E130           020 * 50 052 * 11
X004E140      * 11 020 ---- 005 G  0           JUMP TO TRY SHORT-SET IF NOT VALID VE
X004E150           025 H 19 001 * 12
X004E160           025 K  1 001 * 13
X004E170      * 12 020 * 11 024 K  2
X004E180           052 * 11 010 * 11
X004E190      * 13 020 * 11 025 * 50
X004E200           024 * 51 052 *  9
X004E210      *  9 010 ----                    GO TO GXXX
X004E220      * 50      * 60
X004E230      * 51      G  0
X004E240      * 60 000,0000,063,0524           SET
X004E250           000,0000,000,4417           DO
X004E260           000,0000,000,6417           TO
X004E270           000,0004,417,1605           DONE
X004E280           004,3011,603,0514           CANCEL
X004E290           004,4051,405,2405           DELETE
X004E300           000,0006,431,2005           TYPE
X004E310           000,0005,411,1605           LINE
X004E320           000,0006,001,0705           PAGE
X004E330           000,0006,205,0104           READ
X004E340           000,0602,516,0310           PUNCH
X004E350           000,0004,617,2215           FORM
X004E360           000,0006,324,1720           STOP
X004E370           000,0000,000,4717           GO
X004E380           004,4051,501,1604           DEMAND
X004E390           100,0000,000,0000,
```

```
X005E000+     X  5 010 $   0                    ADVANCE
X005E010           020 Q   8 001 *   5          JUMP IF TASK FINISHED
X005E020           071     1 001 X   7          JUMP IF 'STEP'
X005E030           020 P  16 002 *   1          JUMP IF P16 STILL GOOD
X005E035           020 Q   9 050 P  21          SET STEP
X005E040           020 $   0 010 =  51          FIND CURRENT STEP
X005E050           002 *   1 010 $   1          TO *1 IF FOUND
X005E060           020 H  23 001 X   7          TO X7 (ASCEND) IF DONE.
X005E070           020 H  26 001 X   7          TO X7 (ASCEND) IF DONE.
X005E080           020 H  25 010 *   1          GIMMICK TO GET NEXT
X005E090     *  1 052 *   2 010 $   1
X005E095           024 K   2 052 *   6
X005E096     *  6 020 ---- 050 Q   9          SET Q9
X005E100     *  2 020 ---- 001 *   4
X005E110     *  3 050 P  16 010 X   6          SET P16 FOR NEXT STEP AND GO TO X6(FE
X005E120     *  4 050 P  16 010 X   7          SET P16-AND GO TO X7 (ASCEND)
X005E130     *  5 020 $   0 010 =  55          SWITCH TO USER--TASK COMPLETE
X005E140           020 $   0 010 =  53          KICK OUT PROGRAM
X005E150           010 X   1          ,         GO BACK TO Q SERVICE.
```

```
X006E000+     X  6 010 $  0                    FETCH
X006E010           020 P 15 001 *  1           JUMP IF NO POSTPONED 'RI'
X006E020           020 H 12 024 K 32           FAKE AN 'RI'
X006E030           050 H  9 010 X 10             IN H9 AND JUMP
X006E035      *  1 020 P 16 056 P  3
X006E040           020 $  0 010 =  7           UNPACK
X006E050           020 $  0 010 =  1           ADVANCE TO VERB
X006E060      *  2 020 $  0 010 =  0           READ CLOCK
X006E070           025 H  8 025 J  4
X006E080           002 *  6 024 J  4           JUMP IF SHOT TIME IS UP
X006E090           025 J  3 002 *  4           JUMP IF PREFERRED PERIOD IS UP
X006E100           120    0 145 K 35
X006E110           001 X  4 050 H  9           JUMP IF NO 'TO'
X006E120      *  3 020 $  0 010 = 48           TREAT 'TO' BY SUBROUTINE
X006E140           010 *  2                    LOOP BACK
X006E150      *  4 120    0 145 K 41
X006E160           001 *  5 050 H  9           JUMP IF NO MINOR CYCLE MSG
X006E170           124 K 35 025 K 35
X006E180           002 *  3 010 *  6           JUMP IF 'TO' ELSE JUMP TO CUTOFF
X006E190      *  5 020 L 11 001 X  4           JUMP IF NO L11
X006E200           020 H 10 001 X  4           JUMP IF NO BUFFER
X006E210      *  6 020 $  0 010 = 47           CUTOFF
X006E220              L 12      L 12
X006E230           020 $  0 010 = 53           KICK OUT
X006E240           010 X  1          ,         JUMP TO Q SERVICE
```

```
X007E000+     X  7 010 $  0 010 * 30     ASCEND (OR ITERATE) (014=OVERFLOW)
X007E010           020 K  4 050 P 16     RESET P16
X007E020           020 Q  8 052 *  1
X007E030     *  1 020 ---- 006 *  3     JUMP IF POSSIBLY MORE ITERATIONS
X007E040     *  2 020 $  0 010 = 76     ASCEND REALLY
X007E050           020 K  4 050 P 23     RESET P23
X007E060           020 Q  8 002 X  5     EXIT TO X5 IF MORE TO DO ON TASK
X007E070           020 $  0 010 = 55     SWITCH TO USER
X007E080           020 $  0 010 = 53     KICK OUT PROGRAM
X007E090     *  3 010 X  1 052 *  5     EXIT TO X1 FOR QUEUE SERVICE
X007E100           120    0 050 P 23     SET P23
X007E110           020 Q  0 056 *  4     CHECK FOR TWO SPACES
X007E120     *  4 001 E  0 020 ----
X007E130           002 *  5 010 E  0
X007E140     *  5 020 ---- 052 *  6
X007E150     *  6 023 ---- 002 *  2     JUMP IF NO MORE ITERATIONS
X007E160     * 10 020 *  1 024 K  2     SET UP Q9 FOR NEXT ITERATION
X007E170           052 * 11 010 $  1
X007E180     * 11 020 ---- 001 * 14     JUMP IF MINUS INDICATING 'STEP'
X007E190           050 P 20 010 $  1
X007E200           020 $  0 010 = 50     FIND PART
X007E210           001 E 26 052 * 13     ERROR IF CAN'T
X007E220           024 K  2 052 * 12
X007E230     * 12 020 ---- 050 Q  9
X007E240     * 13 020 ---- 014 * 15
X007E250     * 14 020 Q  9 050 P 21
X007E260           020 $  0 010 = 51     FIND STEP
X007E270     * 15 001 E 27 050 P 16     ERROR IF CAN'T
X007E280     * 20 020 *  6 052 * 21     GET NEXT VALUE FOR ITERATION
X007E290     * 21 020 ---- 056 * 22
X007E300     * 22 050 P 18 020 ----
X007E310           001 * 40 056 * 31     JUMP IF NO INCREMENT
X007E320           024 K  1 056 * 25
X007E330           020 *  6 024 K  2
X007E340           052 * 23 020 * 22
X007E350           024 K  1 056 * 24
X007E355           020 * 23 052 * 33
X007E360           020 $  0 010 = 12     PUSH Q3
X007E370     * 23 020 ---- 050 Q  3
X007E380           020 $  0 010 = 12     PUSH Q3
X007E390     * 25 070    0 020 ----     PICK UP LIMIT
X007E400     * 24 050 H 22 020 ----     PICK UP INCREMENT
X007E410           050 Q  3 050 H 23
X007E420           020 $  0 010 = 31     COMPUTE NEXT VALUE
X007E430           050 H 21 010 $  1
X007E440           020 $  0 010 = 13     POP Q3
X007E450           020 $  0 010 = 70     COMPARE NEW VALUE WITH LIMIT VALUE
X007E460           071    5 001 * 30     JUMP IF NEW VALUE EQUALS LIMIT VALUE
X007E470           070    8 024 H 23
X007E480           071    9 002 * 30     JUMP IF NEW VALUE EXCEEDS LIMIT VALUE
X007E490           020 H 21 010 * 33
X007E500     * 30 020 * 31 056 * 32     DELETE INCREMENT-LIMIT PAIR
X007E510           020 *  6 052 * 32
X007E520     * 31 004 Q  0 020 ----
```

```
X007E530      * 32 056 ---- 060 ----
X007E540           020 * 22 124 K  5
X007E550           050 Q  0 020 H 22     USE LIMIT NEXT IF EQUALED OR EXCEEDED
X007E560      * 33 050 ---- 010 * 50     UPDATE CURRENT VALUE AND JUMP TO ASSI
X007E610      * 40 020 *  5 052 * 43     DELETE USED VALUE
X007E620           020 *  6 052 * 42
X007E630           124 K  6 070   21
X007E640      * 41 004 Q  0 050 Q  0
X007E650      * 42 060 ---- 020 P 18
X007E660      * 43 052 ---- 010 * 50
X007E670      * 50 020 *  5 052 * 51     ASSIGN NEW VALUE
X007E680           024 K  2 052 * 52
X007E690      * 51 020 ---- 056 * 53
X007E700           024 K  3 056 * 54
X007E710      * 52 004 ---- 052 * 58
X007E720      * 53 060 H 27 020 ----
X007E730           056 * 55 024 K  1
X007E740           056 * 56 006 * 54     (USING TRICKS HERE -- BEWARE)
X007E750      * 54 010 * 55 020 ----
X007E760      * 55 050 H 28 020 ----
X007E770      * 56 001 * 57 020 ----
X007E780      * 57 050 H 29 010 $  1
X007E790      * 58 020 ---- 050 H 30
X007E800           020 $  0 010 = 73
X007E810           020 K  4 050 P 23     RESET P23
X007E820           010 X  6         ,    EXIT TO X6 FETCH
```

```
X010E000+        $                   M  0     SIGNAL SERVICE (FROM DRUM)
X010E001            020 J 21 024 K  1        COUNT ONE INPUT
X010E002            050 J 21 010 $  1
X010E005            020 H  9 071    21
X010E010            001 * 10 071     1        JUMP IF 'ON'
X010E015            001 * 20 071     1        JUMP IF 'OF'
X010E020            001 * 40 071     1        JUMP IF 'TC'
X010E025            001 * 50 025 K  1        JUMP IF 'RI'
X010E026            001 * 70 134 $  0        JUMP IF H9 FLAGGED FOR INITIALS
X010E030     * 10   020 H 12 142 K 29        'ON'  ERASE SIGNAL
X010E035            020 $  0 010 = 47
X010E040                L  5     L  5
X010E045            020 H  4 002 * 11        JUMP IF H4 SET ALREADY
X010E050            020 H 12 050 H  4        ELSE SET H4 FOR POSSIBLE Q MSG
X010E055     * 11   010 X  1                  JUMP BACK TO Q SERVICE
X010E060     * 20   020 H 12 142 K 30        'OF'  ERASE SIGNAL
X010E065            024 U  0 052 * 21
X010E070     * 21   020 ---- 124 K  5
X010E075            025 * 36 001 * 22
X010E080            025 I  2 001 * 25        JUMP IF L3 OR L4
X010E085            025 I  4 001 * 22
X010E090            025 I  1 001 * 26        JUMP IF L9
X010E095     * 22   020 L  5 014 * 23        ADJUST H4 IF NECESSARY
X010E100     * 23   020 ---- 001 * 24        NO ADJUSTMENT IF NOT IN L5.
X010E105            050 T  0 025 H 12
X010E110            050 T  1 023 T  1
X010E115            002 * 27 020 T  0
X010E120            025 H  4 050 T  1
X010E125            023 T  1 002 * 24        NO ADJUSTMENT IF DOWNSTREAM FROM
X010E130            020 T  0 024 U  3          NEXT STN FOR Q MSG
X010E135            052 * 23 010 * 23
X010E140     * 27   020 T  0 024 U  3
X010E145            052 * 28 010 $  1
X010E150     * 28   020 ---- 050 H  4        SET SUCCESSOR IN H4 FOR NEW Q MSGS
X010E155     * 24   020 $  0 010 = 58        RELEASE BUFFERS (IF ANY)
X010E160            002 * 24 010 $  1
X010E165            020 H 12 024 U  2        RELEASE-DRUM TREATMENT
X010E170            052 * 32 056 * 29
X010E175     * 29   056 * 33 020 ----
X010E180            001 * 34 020 * 35        JUMP IF NO DRUM ASSIGNED
X010E185     * 30   052 * 31 056 * 32
X010E190     * 31   020 ---- 001 * 32
X010E195            024 U  6 010 * 30
X010E200     * 32   020 ---- 050 ----
X010E205     * 33   020 K  4 050 ----        UNASSIGN STN
X010E206            020 H 12 024 U  7
X010E207            052 * 38 020 * 37
X010E208     * 38   050 ---- 010 $  1        RESET INITIALS IN U7
X010E210     * 34   020 $  0 010 = 47        CHANGE STATE TO L0
X010E215                L  0     L  0
X010E220            010 X  2
X010E225     * 25   120     0 050 H 15        SET DELAYED 'OF' FOR READING
X010E230            010 X  2
X010E235     * 26   120     0 050 H 16        SET DELAYED 'OF' FOR PUNCHING
```

```
X010E240          010 X  2
X010E245     * 35      H 11      H 11
X010E250     * 36                L  3
X010E251     * 37 005,4130,270,0000     THREE MINUS'S FOR UNASSIGNED DRUM
X010E255     * 40 020 H 12 142 K 31     'TC'
X010E260          020 $  0 010 = 54     BRING IN USER'S BLOCK
X010E265          023 P 13 002 * 41     JUMP IF GREEN SUBSTATE WAS 0
X010E270     * 45 020 H 12 024 K 26     ELSE KEEP SAME BUFFER AND
X010E275          142 K 26 010 X  2       SWITCH TO USER
X010E280     * 41 020 H  9 141 R  1     READ BUFFER
X010E285          020 $  0 010 =  5     CONVERT
X010E290          020 R  1 002 * 45     JUMP IF PARTIAL INPUT LINE
X010E295          020 $  0 010 = 58     RELEASE BUFFER
X010E300          020 S  0 050 P 12
X010E305          013 * 42 010 $  1     JUMP IF T3
X010E310          020 $  0 010 = 64
X010E315              * 43      * 43
X010E320          010 X 12              TRANSMIT AND SWITCH VIA X12
X010E325     * 42 020 $  0 010 = 64
X010E330              * 44      * 44
X010E335          010 X 12              TRANSMIT AND SWITCH VIA X12
X010E340     * 43 051,4211,062,1416     'CALL 233 FOR LATEST JOSS INFORMATION
X010E345          001,0030,060,7026
X010E350          023,0510,342,1421
X010E355          031,4251,443,1416
X010E360          060,5463,447,1016
X010E365          014,4450,542,3051
X010E370          022,0211,461,4446
X010E375          022,4331,250,0000
X010E380     * 44 063,4430,521,0462     'PLEASE WIND UP YOUR WORK AND TURN OF
X010E385          012,4161,541,4445     AS SOON AS POSSIBLE.'
X010E390          012,0161,502,3416
X010E395          034,0461,502,4416
X010E400          033,0461,222,1016
X010E405          010,4450,500,7063
X010E410          032,0511,120,7046
X010E415          013,0260,341,0462
X010E420          007,0621,142,3045
X010E425          007,0211,440,7047
X010E430          023,0621,441,4422
X010E435          021,4250,662,5200
X010E440     * 50 020 H 12 142 K 32     'RI' RESET SIGNAL
X010E445          024 U  0 052 * 51
X010E450     * 51 020 ---- 025 * 53
X010E455          024 * 54 056 * 52
X010E460     * 52 025 * 55 001 ----
X010E465          130 $  0
X010E470     * 53      L  0      L  0
X010E475     * 54      * 56      * 56
X010E480     * 55      * 57      * 57
X010E485     * 56 130 $  0              SWITCH TABLE . L0
X010E490          130 $  0              L1
X010E495          010 * 60              L2
X010E500          010 X  2              L3
X010E505          010 X  2              L4
```

```
X010E510          010 X  2                    L5
X010E515          010 X  2                    L6
X010E520          010 X  2                    L7
X010E525          010 * 60                    L8
X010E530          010 X  2                    L9
X010E535          130 $  0                    L10
X010E540          010 * 57                    L11
X010E545          010 * 60                    L12
X010E550          010 * 66                    L13
X010E555          010 * 57                    L14
X010E560          130 $  0                    L15
X010E570    * 57  020 $  0 010 = 54           BRING IN USER BLOCK
X010E580          120    0 050 P 15           SET RI-FLAG FOR L11,L14
X010E590          020 $  0 010 = 53           KICK OUT USER BLOCK
X010E595          010 X .2                    EXIT TO SIGNAL SERVICE
X010E600    * 60  020 $  0 010 = 54           BRING IN USER BLOCK
X010E610    * 66  020 Q  8 001 * 61           JUMP IF DIRECT
X010E620          020 S  0 050 P 12           SET UP 'INTERRUPTED' MESSAGE
X010E625          020 $  0 010 = 64
X010E630              * 90     * 90
X010E635          020 P 16 056 P  3           P16 OK HERE
X010E640          020 $  0 010 =  7           UNPACK
X010E645          020 J 12 050 P  2           COPY STEP NUMBER FROM R TO S
X010E650    * 62  020 $  0 010 =  2
X010E655          124 K 35 025 K 35
X010E660          002 * 63 010 $  1
X010E665          020 P  1 124 A 27
X010E670          025 A 27 001 * 65
X010E675    * 63  020 P 12 024 K  3
X010E680          050 P 12 056 * 64
X010E685    * 64  020 P  1 050 ----
X010E690          020 * 62 010 =  1
X010E695    * 65  020 A 27 050 P 11           INSERT PERIOD
X010E700          020 $  0 010 = 66
X010E705          020 A 42 050 P 11           INSERT CR+EOM
X010E710          020 $  0 010 = 66
X010E715          010 X 12                    TRANSMIT AND SWITCH VIA X12
X010E720    * 61  020 S  0 050 P 12           SET UP 'REVOKED' MESSAGE
X010E725          020 $  0 010 = 64
X010E730              * 91     * 91
X010E735          010 X 12                    TRANSMIT AND SWITCH VIA X12
X010E740    * 70  020 R  1 001 * 73           (INITIALS) JUMP IF NO CHARACTERS
X010E745          020 R  5 002 * 73           JUMP IF MORE THAN FOUR CHARACTERS
X010E750          020 K 15 050 Z  0           INITIALIZE
X010E755    * 71  020 P  1 124 K 34
X010E760          025 K 34 001 * 73           JUMP IF NOT LETTER
X010E765          020 P  1 004 Z  0
X010E770          075   32 050 Z  0
X010E775          020 P  2 025 K  3
X010E780          054 P  2 052 * 72
X010E785    * 72  020 ---- 050 P  1
X010E790          006 * 71 020 H  7           LOOP IF MORE
X010E791          024 U  7 056 * 74
X010E792    * 74  020 Z  0 050 ----           SET INITIALS IN U7
X010E795          120    0 050 P 13           CLEAR 'WTG FOR INITIALS' SUBSTATE
```

```
X010E800          020 $  0 010 = 22      TYPE TIME LINE AT TOP OF PAGE
X010E805          014 X 12               SWITCH AND CARRY ON VIA X12.
X010E810     * 73 020 S  0 050 P 12      INITIALIZE
X010E815          020 $  0 010 = 64
X010E820              * 92     * 92
X010E825          010 X 12               TRANSMIT AND SWITCH VIA X12
X010E850     * 90 054,5011,100,7021      'I'M AT STEP'
X010E855          031,4161,443,1425
X010E860          023,4164,000,0000
X010E865     * 91 064,4251,522,3042      'REVOKED.'
X010E870          012,4240,662,5200
X010E875     * 92 060,4641,443,1416      'JUST ONE TO FOUR LETTERS PLEASE.'
X010E880          023,0450,520,7063      'INITIALS PLEASE-
X010E885          023,0160,542,3064
X010E890          024,4161,061,2463
X010E895          031,4251,223,1016
X010E900          023,4430,521,0462
X010E905          012,4331,345,4445
X010E910          014,4630,621,0443
X010E915          031,0161,162,1425
X010E920          010,4620,525,0016
X010E925          007,0554,000,0000,
X010E999        /     X 10 010,7000/
```

```
X011E000+       $                   M  0    DRUM ROUTINE FOR QUEUE SERVICE
X011E010    *   1 013 *  5 010 $  1    JUMP IF T3 ON
X011E020          020 L  0 002 *  10    JUMP IF Q FOR ENABLE
X011E030    *   2 020 L  5 001 *  3     JUMP IF NO Q FOR DRUM SLOT
X011E040          020 H 11 002 *  20    JUMP IF DRUM SLOT OPEN
X011E050    *   3 020 H  4 001 *  4     JUMP IF NO Q MSG DUE
X011E060          020 H 10 002 *  30    JUMP IF BUFFER AVAILABLE
X011E070    *   4 010 X  1              LOOP BACK TO X1
X011E072    *   5 020 L  1 001 *  2     JUMP IF NO STN IN L1
X011E074          050 H 12 024 K  22
X011E075          142 K 22 010 $  1     DISABLE STN IN L1
X011E076          020 $  0 010 =  47    MOVE STN TO L0
X011E078              L  0     L  0
X011E079          010 *  1
X011E080    *  10 020 L  0 050 H  12    CONSIDER STN AT TOP OF L0
X011E090          024 K 21 142 K  21    ENABLE IT
X011E100          020 $  0 010 =  47    CHANGE STATE TO L1
X011E110              L  1     L  1
X011E120          010 *  1              LOOP FOR MORE
X011E130    *  20 020 L  5 050 H  12    CONSIDER STN AT TOP OF L5
X011E140          004 * 29 110,6000     INITIALIZE CORE BLOCK
X011E145          020 H  1 050 H  8     SET H8 TO KEEP TIME WORKED HONEST
X011E150          020 U  2 024 H  12    ADJUST DRUM TABLES
X011E160          052 * 21 020 H  11
X011E170    *  21 050 ---- 024 U  6     ASSIGN DRUM SLOT
X011E180          052 * 22 056 * 23
X011E190    *  22 020 ---- 050 H  11    BRING UP NEXT DRUM SLOT
X011E200    *  23 020 K  4 050 ----     SET NULL SUCCESSOR TO ASSIGNED SLOT
X011E210          020 H 12 050 H  7     ESTABLISH STN AS IN CORE
X011E220          020 $  0 010 =  47    CHANGE STATE (TO POP L5)
X011E230              L 13     L  13
X011E240          020 L  5 050 H  4     SET H4 FOR Q MSGS IF ANY
X011E250          120    0 050 P  0     CLEAR LINE NUMBER
X011E260          020 S  0 050 P  12    INITIALIZE OUTPUT LINE
X011E270          020 $  0 010 =  64    UNPACK MSG TO S
X011E280              * 28     * 28
X011E282          020 U  7 024 H  12    SET TEMPORARY INITIALS $$$
X011E284          052 * 24 020 *  27
X011E286    *  24 050 ---- 010 X  12    EXIT TO X12 TO TRANSMIT REQ AND SWITC
X011E290    *  27 010,0200,410,0000     TEMPORARY INITIALS $$$
X011E300    *  28 027,1413,147,1162     'JOSS AT YOUR SERVICE.'
X011E305          007,0211,460,7070     'INITIALS PLEASE-
X011E310          023,0641,220,7062
X011E320          012,4511,521,4423
X011E330          012,4331,345,4445
X011E340          014,4630,621,0443
X011E350          031,0161,162,1425
X011E360          010,4620,525,0016
X011E370          007,0554,000,0000
X011E380    *  29 000,0000,013,1777     DRUM CW FOR INITIAL BLOCK
X011E390    *  30 020 H  4 050 H  12
X011E400          024 U  3 052 *  31
X011E410    *  31 020 ---- 050 H  4
X011E420          020 * 36 050 S  1
```

```
X011E425          020 * 37 050 P 12
X011E430          020 $  0 010 = 61
X011E440          020 $  0 010 = 64
X011E450              * 38     * 38
X011E460          120    0 050 H 20
X011E470          020 L  5 050 T  0
X011E480     * 32 020 H 20 024 I  1
X011E490          050 H 20 020 T  0
X011E500          025 H 12 001 * 33
X011E510          025 K  1 001 * 35
X011E520     * 33 020 T  0 024 U  3
X011E530          052 * 34 010 $  1
X011E540     * 34 020 ---- 050 T  0
X011E550          002 * 32 134 $  0
X011E560     * 35 020 $  0 010 = 36
X011E570                1        1
X011E580          020 $  0 010 = 64
X011E590              * 39     * 39
X011E600          120    0 050 P  0
X011E610          020 $  0 010 = 25     TRANSMIT Q MSG (BUFFER OPEN)
X011E615          010 *  3
X011E620     * 36 000,0000,000,0056     CR
X011E625     * 37      S  1      S  1
X011E630     * 38 007,0163,602,3064     '  YOU ARE NOW NUMBER '
X011E640          007,0211,221,2416
X011E650          022,4461,540,7045
X011E660          032,0440,441,2451
X011E670          007,2000,000,0000
X011E680     * 39 007,0311,120,7063     'IN THE QUEUE.'
X011E690          014,0250,342,4064
X011E700          012,4640,521,5452
X011E710          100,0000,000,0000,
X011E999        /      X 11 010,7000/
```

```
X012E000+    X 12 010 *  1 010 *  2     TRANSMIT (IF 010) AND SWITCH
X012E010     *  1 020 $  0 010 = 25     TRANSMIT
X012E020     *  2 020 $  0 010 = 55     SWITCH
X012E030        020 $  0 010 = 53     KICK OUT
X012E040        010 X  1         ,    EXIT TO Q SERVICE
```

```
X014E000+     X 14 010 $  0                    SPECIAL TRANSMIT THEN GO TO X5
X014E010           020 $  0 010 = 25
X014E020           010 X  5          ,
```

```
X015E000+     X 15 010 $  0                    READ ROUTINE FROM DRUM AND EXECUTE
X015E010         050 T  0 020 H 17             SAVE ACCUMULATOR
X015E020         065 T  1 023 T  1
X015E030         002 *  1 060 H 17             JUMP IF ROUTINE ALREADY IN CORE
X015E040         110 M  0 010 $  1             ELSE READ IT IN
X015E050      *  1 020 T  0 010 M  0,          RESET ACCUMULATOR AND JUMP
```

```
=000E000+     =   0 010 $   0                       READ CLOCK
=000E010          024 K   2 052 *   9
=000E020          107     0 004 H   0
=000E030          050 H   0 065 T   0
=000E040          002 *   1 024 K  31     ADD 2*16 IF CLOCK TURNED OVER
=000E050      *   1 024 H   1 050 H   1
=000E060      *   9 010 ----          ,     EXITS WITH TIME IN ACC AND H1
```

```
=001E000+     =   1 010 $   0                    STEP ONE CHARACTER
=001E010            024 K   2 052 *   9
=001E020            020 P   2 024 K   3
=001E030            052 *   1 054 P   2
=001E040      *   1 020 ----- 050 P   1          LEAVES NEW CHARACTER IN ACC
=001E050      *   9 010 ----
```

```
=002E000+     =   2 024 K   2 010 $   0      ADVANCE TO NONSPACE
=002E010            052 *   9 010 *   1
```

```
=003E000      =  3 010 $  0                        ELIMINATE SPACES
=003E010         024 K  2 052 *  9
=003E020         020 P  1 010 *  3
=003E030      *  1 020 P  2 024 K  3
=003E040         052 *  2 054 P  2
=003E050      *  2 020 ---- 050 P  1
=003E060      *  3 124 A 14 025 A 14
=003E070         002 *  1 020 P  1
=003E080      *  9 010 ----          ,     LEAVES NONSPACE IN ACC
```

```
=004E000+      =   4 130 =   4 134 =   4
```

```
=005E000+     =   5 010 $  0                    CONVERTER
=005E010          024 K  2 052 *  9
=005E020      *   1 020 A 42 050 R 73           FILL WITH CR+EOM+TERMINAL FROM RIGHT
=005E030          020 *  1 014 *  3
=005E040      *   2 020 A 42 050 ----
=005E050      *   3 020 *  2 025 K  1
=005E060          056 *  2 056 *  4
=005E070      *   4 020 *  5 004 ----
=005E080          065 T  0 023 T  0
=005E090          002 *  2 020 *  6
=005E100          065 T  0 023 T  0
=005E105          002 *  2 010 * 10
=005E110      *   5                ,0016        SPACE
=005E115      *   6                ,0103        STRIKE OUT
=005E120      *  10 020 J 12 050 P  2
=005E130      *  11 020 K  4 050 P 10           CLEAR 'IF'
=005E140      *  12 020 $  0 010 * 20
=005E150      *  13 020 P  1 124 A 14           SPACE
=005E160          025 A 14 001 * 14
=005E170      *  15 020 $  0 010 * 20
=005E180          020 P  1 124 A 25           I
=005E190          025 A 25 001 * 13
=005E200          020 $  0 010 * 20
=005E210          020 P  1 124 A 22           F
=005E220          025 A 22 001 * 13
=005E230          020 $  0 010 * 20
=005E240          020 P  1 124 A 14           SPACE
=005E250          025 A 14 001 * 14
=005E260          020 P  2 050 P 10           RECORD 'IF' BY LOCN OF FOLLOWING SPAC
=005E270          010 * 15
=005E280      *  14 020 P  1 124 A 66
=005E290          025 A 66 001 * 12
=005E300          010 * 11                    CLEAR 'IF' IF QUOTE
=005E310      *  20 024 K  2 052 * 29
=005E320          020 P  2 024 K  3
=005E330          050 P  2 056 * 21
=005E340      *  21 056 * 23 020 ----
=005E350          024 J 13 056 * 22
=005E360      *  22 001 * 24 020 ----
=005E370      *  23 050 P  1 050 ----
=005E380      *  29 010 ----                 EXIT FROM *20
=005E390      *  24 020 P  2 025 K  3
=005E400          050 P  2 052 * 25
=005E410      *  25 020 ---- 050 P  1
=005E420      *   9 010 ----       ,          EXIT WITH LAST CHAR IN CONTEXT AND AC
```

```
=006E000+     =   6 010 $   0                    PACKER
=006E010            024 K   2 052 *   9          LEAVES LINK TO STRING IN P3 AND ACC.
=006E030            020 R   1 001 *   8          TO *8 IF VACUOUS
=006E040            020 Q   0 056 Q  15          SET
=006E050            020 J  12 050 P   2
=006E060            020 $   0 010 =   1
=006E070      *   1 020 $   0 010 =  10
=006E080            056 *   3 024 K   1
=006E090            056 *   2 070      0
=006E100            020 $   0 010 *  20
=006E110            020 $   0 010 *  20
=006E120            020 $   0 010 *  20
=006E130            020 $   0 010 *  20
=006E140            060 T   0 071     12
=006E150            050 T   1 070      0
=006E160            020 $   0 010 *  20
=006E170            020 $   0 010 *  20
=006E180            020 $   0 010 *  20
=006E190            020 $   0 010 *  20
=006E200            020 $   0 010 *  20
=006E210            060 T   0 010 $   1
=006E220      *   2 071     5 050 ----
=006E230            020 P   1 001 *   3
=006E240            020 Q   0 056 T   1
=006E250      *   3 020 T   1 050 ----
=006E260            020 P   1 002 *   1
=006E270      *   8 020 Q  15 050 P   3          POSITION OUTPUT IN P3
=006E280      *   9 010 ----
=006E290      *  20 024 K   2 052 *  29
=006E300            120      0 075      7
=006E310            020 P   1 124 K  17
=006E320            064 T   0 004 T   0
=006E330            020 P   2 024 K   3
=006E340            052 *  21 054 P   2
=006E350      *  21 020 ---- 050 P   1
=006E360      *  29 010 ----                    ,
```

```
=007E000+     =   7 024 K   4 010 $   0     UNPACK TO R(IF 010), TO S(IF 014).
=007E010          024 K   2 052 *   9
=007E020          006 *   5 020 J  12
=007E025      *   5 010 *   6 020 S   0
=007E030      *   6 052 *  23 020 K   4
=007E035          050 P   2 050 P  10
=007E040          020 P   3 050 T   0     PICK UP SOURCE FROM P3.
=007E050      *   1 023 T   0 001 *   2     TO *2 IF MORE
=007E060          020 A  42 010 *  21     SUPPLY CR+EOM AND EXIT VIA *21.
=007E070      *   2 020 T   0 010 $   1
=007E080          056 *   3 024 K   1
=007E090      *   3 056 *   4 004 ----
=007E100          020 $   1 010 *  20
=007E110          020 $   1 010 *  20
=007E120          020 $   1 010 *  20
=007E130          020 $   1 010 *  20
=007E140          075    12 010 $   1
=007E150      *   4 056 T   0 004 ----
=007E160          020 $   1 010 *  20
=007E170          020 $   1 010 *  20
=007E180          020 $   1 010 *  20
=007E190          020 $   1 010 *  20
=007E200          020 $   1 010 *  20
=007E210          023 T   0 014 *   1
=007E230      *  20 050 *  29 075     7
=007E240      *  21 124 K  17 024 J  13
=007E245          056 *  22 010 $   1
=007E250          020 *  23 024 K   2
=007E260      *  22 052 *  23 020 ----
=007E270      *  23 050 ---- 002 *  29
=007E280          124 A  45 025 A  45
=007E290      *   9 001 ---- 020 *  23     EXIT
=007E300          124 K   6 050 T   1
=007E310          074    21 056 T   1
=007E320          020 P   2 001 *  24     JUMP IF P2 NOT SET
=007E330          020 T   1 024 K  43
=007E340          050 P  10 010 *  29     SET P10
=007E350      *  24 020 T   1 050 P   2
=007E370      *  29 --- ---- --- ----,
```

```
=008E000+     =   8 010 $   0                       ASSIGN BUFFER TO STN
=008E010              024 K   2 052 *   9
=008E020              020 H  10 050 H  13           SET H13
=008E030              001 *   9 024 U   5           EXIT MINUS IF NO BUFFER AVAILABLE
=008E040              052 *   1 056 *   5
=008E050      *   1 020 ----  050 H  10           UPDATE H10
=008E060              021 J  16 050 T   0
=008E070              020 H  12 024 U   1
=008E080              052 *   2 056 *   6
=008E090      *   2 020 ----  002 *   4
=008E100              020 H  12 050 T   1           SET BUFFER BITS IN SCR FOR NEW BUFFER
=008E110              020 H  13 052 T   1
=008E120              020 T   1 142 K  16
=008E130              010 *   5
=008E140      *   3 020 ----  001 *   5
=008E150      *   4 024 U   5 052 *   3
=008E160              056 *   6 020 T   0
=008E170              024 I   1 050 T   0
=008E180              001 *   3 020 H  13
=008E190              050 H  10 020 K   4           BACKTRACK - STN ALREADY HAS LIMIT
=008E200              050 H  13 010 *   9
=008E210      *   5 020 K   4 050 ----              SET SUCCESSOR TO -1.
=008E220      *   6 020 H  13 050 ----              PUT NEW BUFFER ON END OF LIST FOR STN
=008E230      *   9 010 ----                    ,   EXIT WITH BUFFER (OR-1) IN H13 AND AC
```

=009E000+ = 9 130 = 9 134 = 9

```
=010E000+     = 10 024 K   2 010 $   0      DEAL OUT A STORAGE SPACE
=010E010         052 *   9 020 Q   0
=010E020         001 E   0 056 *   1
=010E030      *  1 050 T   0 020 ----
=010E040         050 Q   0 020 T   0      LEAVES ADDR IN R(ACC)
=010E050      *  9 010 ----           ,
```

```
=011E000+     = 11 024 K  2 010 $  0      UNPACK (Q3) TO P4,P5,P6.
=011E010         052 *  9 020 P  4
=011E020         050 P  7 020 P  5
=011E030         050 P  8 020 P  6
=011E040         050 P  9 020 Q  3
=011E050         124 K 11 071    1
=011E060         072    1 050 P  4
=011E070         020 Q  3 124 K 12
=011E080         050 P  5 020 Q  3
=011E090         124 K 13 050 P  6
=011E100      *  9 010 ----         ,
```

```
=012E000+     = 12 004 * 92 010 *  0      PUSH Q2-Q3 OPERANDS
```

```
=013E000      = 13 004 * 92 010 * 10      POP Q2-Q3 OPERANDS
```

```
=014E000      = 14 004 * 94 010 *  0      PUSH Q4-Q5 OPERATORS
```

```
=015E000      = 15 004 * 94 010 * 10      POP Q4-Q5 OPERATORS
```

```
=016E000      = 16 004 * 96 010 *  0      PUSH Q6-Q7 AUXILIARY
```

```
=017E000      = 17 004 * 96 010 * 10      POP Q6-Q7 AUXILIARY
```

```
=018E000      = 18 004 * 98 010 *  0      PUSH Q8-Q9 CONTROLS
```

```
=019E000      = 19 004 * 98 010 * 10      POP Q8-Q9 CONTROLS
=019E010      *  0 024 K  2 052 *  9
=019E020         060 T  0 056 *  1
=019E030         052 *  2 024 K  3
=019E040         052 *  5 020 Q  0
=019E050      *  1 001 E  0 004 ----
=019E060      *  2 050 ---- 056 *  3
=019E070         056 *  4 024 K  1
=019E080      *  3 056 *  5 020 ----
=019E090      *  4 050 Q  0 060 ----
=019E100      *  5 020 ---- 050 ----
=019E110      *  9 010 ----
=019E120      * 10 024 K  2 052 * 19
=019E130         060 T  0 052 * 11
=019E140         052 * 13 024 K  3
=019E150         052 * 14 004 Q  0
=019E160      * 11 020 ---- 124 K  5
=019E170         050 Q  0 056 * 12
=019E180         056 * 14 024 K  1
=019E190      * 12 056 * 13 020 ----
=019E200      * 13 050 ---- 020 ----
=019E210      * 14 050 ---- 060 ----
=019E220      * 19 010 ----
=019E230      * 92      Q  2      Q  2
=019E240      * 94      Q  4      Q  4
=019E250      * 96      Q  6      Q  6
=019E260      * 98      Q  8      Q  8,
```

```
=020E000+     = 20 010 $  0                    EVALUATE DECIMAL EXPRESSION (UNSIGNED
=020E010          024 K  2 052 *  9
=020E020          020 $  0 010 = 12           PUSH Q3
=020E030          120    0 050 T  1           INITIALIZE
=020E040          020 K  4 050 P  6
=020E050          050 T  0 050 T  3
=020E060          020 P  1 010 *  2
=020E070     *  1 020 $  0 010 =  1           ADVANCE (DOESN'T DISTURB T'S)
=020E080     *  2 124 * 90 025 K 35
=020E090          001 *  5 050 P  6           JUMP IF NOT DIGIT
=020E100          025 K  1 001 *  1           LOOP IF '0'
=020E110          020 P  2 050 T  3           RECORD LOC'N OF MOST SIGNIF DIGIT
=020E120     *  3 020 $  0 010 =  1           ADVANCE
=020E130          124 * 90 025 K 35
=020E140          001 *  5 050 T  2           JUMP IF NOT DIGIT
=020E150          020 P  2 025 T  3
=020E160          025 T  1 025 * 99
=020E170          006 *  4 020 P  6           JUMP IF ALREADY NINE SIGNIF. DIGITS
=020E180          073    2 024 P  6
=020E190          073    1 024 T  2
=020E200          050 P  6 010 *  3
=020E210     *  5 020 T  0 002 * 10           JUMP IF END OF FIELD
=020E220          020 P  1 124 A 27
=020E230          025 A 27 001 * 10           JUMP IF END OF FIELD
=020E240          020 P  2 050 T  0
=020E250          020 T  3 001 *  1           LOOP IF MSD NOT YET FOUND
=020E260          020 K  3 050 T  1           CHANGE T1
=020E270     *  4 010 *  3 021 T  2           LOOP //
=020E280          002 *  3 010 E  5           LOOP IF '0' ELSE TOO MANY SIGNIF. DIG
=020E290     * 10 020 P  6 001 E  6           ERROR IF NO DIGITS
=020E300          023 P  6 006 * 15           JUMP IF ZERO RESULT
=020E310     * 11 020 T  0 002 * 12           JUMP IF DECIMAL POINT
=020E320          020 P  2 025 K  3           SPECIAL FOR INTEGER
=020E330     * 12 025 T  3 025 T  1           (MUST USE RIGHT FIELD)
=020E340          073   31 050 P  4           SET XP OF RESULT
=020E350          020 * 99 025 P  2
=020E360          024 T  3 024 T  1
=020E370          001 * 14 024 J 11           JUMP IF DP OF RESULT OK
=020E380          052 * 13 004 P  6
=020E390     * 13 032 ---- 060 P  6
=020E400     * 14 020 P  4 125 K  4
=020E410     * 15 024 P  6 050 Q  3
=020E420     *  9 010 ----
=020E430     * 90 000,0000,001,0017           SPECIAL MASK FOR DIGIT AND VALUE
=020E440     * 99         9        9,
```

```
=021E000+     = 21 124 K  6 010 $  0     EVALUATE EXPRESSION
=021E010          024 K  2 050 H 18
=021E020          020 $  0 010 = 14
=021E030          020 H 18 050 Q  5     STACK EXIT IN OPERATORS
=021E040     *  1 020 $  0 010 =  3     ELIMINATE SPACES
=021E050     *  9 124 K 29 025 K 29
=021E060          001 *  2 010 $  1
=021E070          020 $  0 010 = 12
=021E080          120    0 050 Q  3
=021E090          020 $  0 010 = 14
=021E100          020 P  1 050 Q  5
=021E110     * 22 020 $  0 010 =  2     ADVANCE
=021E120     *  2 020 P  1 071    23
=021E130          001 *  8 071     3     TO *8 IF LEFT GROUPER
=021E140          001 *  3 071     1     TO *3 IF LETTER
=021E150          001 *  7 020 P  1     TO *7 IF DIGIT
=021E160          124 A 27 025 A 27
=021E170          002 *  7 020 P  1     TO *7 IF DEC PT
=021E180          124 A 68 025 A 68
=021E190          002 * 10 010 E  6     TO *10 IF CC = '$' ELSE MALFORMED
=021E200     *  3 020 P  2 024 K  2     LETTER
=021E210          052 *  4 010 $  1
=021E220     *  4 020 ---- 050 T  0     PICK UP LOOK AHEAD CHARACTER
=021E230          124 K 34 025 K 34
=021E235          002 * 13 020 P  1     JUMP IF LETTER (HENCE FUNCTION)
=021E240          052 *  5 052 *  6
=021E245     *  5 020 ---- 005 * 15     JUMP IF NOT SCALAR
=021E250          020 $  0 010 = 12     PUSH Q3
=021E255     *  6 020 ---- 050 Q  3
=021E260     * 15 010 * 29 020 T  0
=021E265          124 A 96 025 A 96
=021E270          006 * 16 020 T  0     JUMP IF ABS VAL BAR
=021E275          124 K 31 025 K 31
=021E280     * 16 002 * 28 020 P  1     JUMP IF '(' OR LEFT BRACKET
=021E285          050 H 27 020 K  4     ERROR--SET H27 AND H28 FOR E2
=021E290          050 H 28 010 E  2
=021E300     *  7 020 $  0 010 = 20     CONVERT NUMBER
=021E310          020 * 14 010 =  3     ELIMINATE SPACES AND CONTINUE AT *20
=021E330     *  8 020 $  0 010 = 14     STACK LEFT GROUPER
=021E340          020 P  1 050 Q  5
=021E350          020 *  1 010 =  2     ADVANCE TO NONSPACE AND CONTINUE AT *
=021E370     * 10 020 $  0 010 = 12     CONVERT $
=021E380          020 P  0 025 N  1
=021E390          002 * 11 004 P  0
=021E400          032 N  8 014 * 12
=021E410     * 11 004 P  0 032 N  7
=021E420     * 12 020 K 14 064 Q  3
=021E430     * 29 020 * 14 010 =  2     ADVANCE TO NONSPACE AND CONTINUE AT *
=021E440     * 13 020 $  0 010 = 24     FUNCTION
=021E450     * 14 020 $  0 010 = 26     EVALUATE GROUPED LIST
=021E615     * 20 020 P  1 124 K 30       K30 ~ ARITH OPERATOR
=021E620          025 K 30 001 * 23
=021E625          020 Q  5 124 K 30
=021E630          025 K 30 001 * 21
```

```
=021E635          020 P  1 124 K 37
=021E640          050 T  0 020 Q  5
=021E645          124 K 37 025 T  0
=021E650          002 * 24 010 $  1
=021E655     * 21 020 $  0 010 = 14      STACK OPERATOR
=021E660          020 P  1 050 Q  5
=021E665          010 * 22
=021E670     * 23 020 Q  5 124 K 30
=021E675          025 K 30 001 * 26
=021E680     * 24 020 Q  5 052 * 25
=021E685          020 $  0 024 K  2
=021E690     * 25 010 ----               FIRE OPERATOR
=021E695          020 * 14 010 = 15      POP OPERATOR AND CONTINUE AT *20
=021E705     * 26 020 Q  5 124 K 31
=021E710          025 K 31 001 * 30      JUMP TO EXIT IF NOT LEFT GROUPER
=021E712          020 $  0 010 = 77      MATCH GROUPERS
=021E714          020 P  1 124 A 96
=021E716          025 A 96 001 * 29
=021E718          020 Q  3 125 K 12      TAKE ABS VAL
=021E720          050 Q  3 010 * 29
=021E724     * 30 020 Q  5 052 * 27
=021E726          020 $  0 010 = 15      POP Q5
=021E730     * 27 010 ----               EXIT
=021E735     * 28 020 $  0 010 = 14      SUBSCRIPT
=021E740          020 P  1 124 K 17
=021E745          024 * 90 050 Q  5
=021E750          020 $  0 010 =  1
=021E755          010 * 14
=021E760     * 90 000 = 74 440 ----,
```

```
=022E000+     = 22 010 $  0 010 *  1     TYPE TIME LINE (ENTER RT SUPPRESSES E
=022E010           050 * 90 010 $  1
=022E020           020 A 31 050 S  1
=022E030           020 S  0 024 K  3
=022E040           050 P 12 010 *  2
=022E050      *  1 050 * 90 010 $  1
=022E060           020 S  0 050 P 12
=022E070      *  2 020 $  0 010 = 61     TIME
=022E080           020 $  0 010 = 62     SPACE
=022E090           020 $  0 010 = 64     DATE
=022E100                H  2       H  2
=022E110           020 $  0 010 = 62     SPACE
=022E120           020 $  0 010 = 64     INITIALS
=022E130                Z  0       Z  0
=022E140           020 $  0 010 = 64     SIX CR'S AND EOM
=022E150                * 91       * 91
=022E160           120    0 050 P  0     RESET LINE COUNTER
=022E170           020 * 90 010 = 25     EXIT VIA=25 TO OUTPUT LINE FROM S
=022E180      * 90 --- ---- --- ----     LINK
=022E190      * 91 027,0561,342,7056
=022E200           025,2000,000,0000,
```

```
=023E000+     = 23 010 $  0                    EVALUATE CONDITION
=023E010          024 K  2 052 *  9
=023E020          120    0 055 *  4           SET SWITCH (ALWAYS SAVE FIRST RESULT)
=023E030      *  0 120    0 050 * 50           SET SIGNAL +
=023E040          020 $  0 010 = 21           EVALUATE EXPRESSION
=023E050      *  1 020 Q  3 050 * 51           SAVE IT
=023E060          020 $  0 010 = 13           POP Q3
=023E070          020 P  1 124 K 33
=023E080          025 K 33 001 E  6           MALFORMED IF NOT RELATION
=023E090          020 P  1 050 * 52           SAVE IT
=023E100          020 $  0 010 =  1           STEP ONE
=023E110          020 $  0 010 = 21           EVALUATE EXPRESSION
=023E120          020 * 51 050 H 21
=023E130          020 Q  3 050 H 22
=023E140          020 $  0 010 = 70           COMPARE
=023E150          124 * 52 025 K  1
=023E160          001 *  2 010 *  3
=023E170      *  2 020 K  4 050 * 50           SET - IF NOT MET
=023E180      *  3 020 P  1 124 K 33
=023E190          025 K 33 002 *  1           JUMP IF ANOTHER RELATION
=023E200          020 $  0 010 = 13           POP Q3
=023E210      *  4 020 * 50 --- *  8           SWITCH
=023E220          050 * 53 010 $  1           STORE NEW RESULT
=023E230      *  8 020 P  1 124 A109
=023E240          025 A109 002 *  7           JUMP IF PERIOD
=023E245          020 $  0 010 = 79           VERIFY PRECEDING SPACE
=023E250          020 $  0 010 = 27           ACCUMULATE WORD
=023E260          025 W 14 050 T  0
=023E270          023 T  0 002 *  5           JUMP IF 'OR'
=023E280          020 H 19 025 W 15
=023E290          050 T  0 023 T  0
=023E300          002 *  6 010 E  6           JUMP IF 'AND' ELSE MALFORMED
=023E310      *  5 020 K 35 014 *  6           OR
=023E320      *  6 020 K 34 055 *  4           AND
=023E322          020 $  0 010 = 39           VERIFY SPACE AND ADVANCE TO NONSPACE
=023E324          010 *  0
=023E330      *  7 020 * 53 010 $  1
=023E340      *  9 010 ----                   EXIT
=023E350      * 50 --- ---- --- ----           SIGNAL
=023E360      * 51 --- ---- --- ----           LEADING TERM
=023E370      * 52 --- ---- --- ----           RELATION
=023E380      * 53 --- ---- --- ----,          FINAL SIGNAL
```

```
=024E000+     = 24 010 $  0                    CONVERT FUNCTION TO OPERATOR
=024E010          024 K  2 052 *  9
=024E020          020 $  0 010 = 14
=024E030          020 $  0 010 = 27
=024E040          050 T  0 004 K  2
=024E050          020 * 50 052 *  1
=024E060      *   1 020 ---- 001 E  6
=024E070          025 T  0 001 *  2
=024E080          025 K  1 005 *  3
=024E090      *   2 020 *  1 064 *  1
=024E100      *   3 010 *  1 020 *  1
=024E110          124 K  6 025 * 50
=024E120          024 * 51 050 Q  5          STACK FN AS OPERATOR IN Q5
=024E130      *   9 010 ----
=024E140      *  50     * 60
=024E150      *  51     F  0 440
=024E160      *  60 000,0002,321,2224         SQRT (ORDERED SAME AS F'S)
=024E170          000,0000,014,1707         LOG
=024E180          000,0000,005,3020         EXP
=024E190          000,0000,023,1116         SIN
=024E200          000,0000,003,1723         COS
=024E210          000,0000,000,1120         IP
=024E220          000,0000,000,0620         FP
=024E230          000,0000,000,3020         XP
=024E240          000,0000,000,0420         DP
=024E250          000,0000,023,0716         SGN
=024E260          000,0000,001,2207         ARG
=024E270          000,0000,015,0130         MAX
=024E280          000,0000,015,1116         MIN
=024E290          100,0000,000,0000,
```

```
=025E000+     = 25 010 $  0 010 *  1     OUTPUT LINE FROM S BLOCK TO STN
=025E010          024 K  2 052 P 14     SET EXIT IN P14 (INITIAL ENTRY)
=025E020      *  1 020 $  0 010 =  8     ASSIGN BUFFER (REENTRY POINT)
=025E030          001 *  3 140 S  1     LOAD BUFFER IF ASSIGNED
=025E040          020 H 12 024 U  1
=025E050          052 *  2 020 H 13
=025E060      *  2 025 ---- 001 *  5     EXIT VIA P14 IF CAN'T TL YET
=025E070          025 K  1 002 *  5     EXIT VIA P14 IF CAN'T TL YET
=025E080          020 H 12 024 K 24     OK TO TRANSMIT
=025E090          142 K 24 010 *  5     EXIT VIA P14
=025E100      *  3 020 H 10 001 *  4
=025E110          020 $  0 010 = 47     CHOKED-CHANGE STATE TO L14
=025E120              L 14     L 14
=025E130          010 *  6
=025E140      *  4 020 $  0 010 = 47     Q FOR BUFFER AND PREFERRED PROCESSINC
=025E150              L 11     L 11
=025E155      *  6 020 $  0 010 = 53     KICK OUT PROGRAM
=025E160          010 X  1              EXIT TO EXECUTIVE
=025E170      *  5 020 J 22 004 I  1     COUNT ONE OUTPUT
=025E172          064 J 22 020 P  0     COUNT ONE LINE NUMBER
=025E180          064 P  0 025 J  2
=025E190          001 P 14 020 P 14
=025E200          025 K  2 010 = 22,    EXIT VIA=22 IF NEW PAGE NEEDED
```

```
=026E000+     = 26 124 K   6 010 $   0     EVALUATE GROUPED LIST
=026E010           024 K   2 050 H  18     RESULT IN Q3 AND AUX
=026E020           020 $   0 010 =  14
=026E030           020 H  18 050 Q   5     STACK EXIT IN OPERATORS
=026E040           020 P   1 124 A  73
=026E045           025 A  73 002 *   1
=026E050           020 P   1 124 A123
=026E055           025 A123 001 E   6     MALFORMED IF NOT LEFT PAREN OR BRACKE
=026E060     *   1 020 $   0 010 =  14     PUSH OPERATORS
=026E070           020 P   1 050 Q   5     STACK CHARACTER IN OPERATORS
=026E080           020 $   0 010 =   1     ADVANCE
=026E090           020 $   0 010 =  21     EVALUATE EXPRESSION
=026E100           020 P   1 124 A  59
=026E110           025 A  59 002 *   1     JUMP IF COMMA
=026E120     *   2 020 Q   5 124 A  59
=026E130           025 A  59 001 *   3     JUMP IF PREV OPERATOR NOT COMMA
=026E140           020 $   0 010 =  15     POP COMMA
=026E150           020 $   0 010 =  16     PUSH AUX
=026E160           020 Q   3 050 Q   7     SAVE VALUE OF EXPR IN AUX
=026E170           020 $   0 010 =  13     POP Q3
=026E180           010 *   2
=026E190     *   3 020 $   0 010 =  77     MATCH GROUPERS
=026E195           020 $   0 010 =   2     ADVANCE TO NONSPACE
=026E200           020 Q   5 052 *   9
=026E210           020 $   0 010 =  15     POP Q5
=026E220     *   9 010 ----              ,  EXIT
```

```
=027E000+     = 27 010 $  0                        ACCUMULATE LETTER CODES (6 MAX)
=027E010           024 K  2 052 *  9               RESULT IN ACC WITH SCAN ADVANCED
=027E020           120    0 050 H 19
=027E030      *  1 020 P  1 124 K 34
=027E040           025 K 34 005 *  3
=027E050           020 H 19 004 P  1
=027E060           077    6 050 H 19
=027E070           020 P  2 024 K  3
=027E080           054 P  2 052 *  2
=027E090      *  2 020 ---- 050 P  1
=027E100           020 H 19 025 K 12
=027E110      *  3 001 *  1 020 H 19               LEAVE RESULT IN ACC AND H19
=027E120      *  9 010 ----          ,
```

```
=028E000+     = 28 010 $  0                    ERASE LEFT LINKED LIST A/C MQ
=028E010           024 K  2 052 *  9
=028E020           060 T  0 124 K  6
=028E030           050 T  0 010 $  1
=028E040      *  1 023 T  0 002 *  9
=028E050           020 T  0 052 *  2
=028E060           052 *  3 070    21
=028E070           004 Q  0 050 Q  0
=028E080      *  2 020 ---- 052 T  0
=028E090      *  3 060 ---- 010 *  1
=028E100      *  9 010 ----                  ,
```

```
=029E000+     = 29 010 $  0                    ERASE RIGHT LINKED LIST A/C MQ
=029E010           024 K  2 052 *  9
=029E020           060 T  0 124 K  5
=029E030           050 T  0 010 $  1
=029E040      *  1 023 T  0 002 *  9
=029E050           020 T  0 056 *  2
=029E060           056 *  3 004 Q  0
=029E070      *  2 050 Q  0 020 ----
=029E080      *  3 056 T  0 060 ----
=029E090           010 *  1
=029E100      *  9 010 ----                  ,
```

```
=030E000+     = 30 010 $  0                        CHECK RANGE AND PACK RESULT
=030E010          024 K  2 052 *  9
=030E015          023 P  6 006 *  3                CHK FOR ZERO
=030E020          020 P  4 005 *  1
=030E030          025 * 90 005 *  2
=030E040      *  1 010 E  1 024 * 91
=030E050          006 *  2 120    0
=030E060      *  2 014 *  3 020 P  4
=030E070          024 P  5 024 P  6
=030E080      *  3 125 K  4 050 Q  3
=030E090      *  9 010 ----
=030E100      * 90+          100    8
=030E110      * 91+           99    8,
```

```
=031E000+     = 31 010 $  0                    ADD
=031E010           024 K  2 052 *  9
=031E020           020 $  0 010 = 11          UNPACK 2ND OPERAND
=031E030           020 *  1 010 = 13          POP Q3 AND MERGE WITH =32
```

```
=032E000     = 32 010 $  0                    SUBTRACT
=032E010          024 K  2 052 *  9
=032E020          020 $  0 010 =  11          UNPACK 2ND OPERAND
=032E030          020 K 12 025 P  5           INVERT SIGN OF 2ND OPERAND
=032E040          050 P  5 010 $  1
=032E050     *  1 020 $  0 010 =  13          POP Q3
=032E060          023 P  6 002 *  9           DONE IF 2ND OPERAND IS ZERO
=032E070          023 Q  3 002 *  8           PACK UP IF 1ST OPERAND IS ZERO
=032E080          020 $  0 010 =  11          UNPACK 1ST OPERAND
=032E090          020 P  4 025 P  7
=032E100          001 *  2 025 K  1          TO *2 IF 2ND SFX IS GREATER
=032E110          002 *  3 020 P  6          TO    IF 1ST SFX IS GREATER
=032E120          025 P  9 002 *  3          TO    IF 1ST CF IS GREATER OR EQUAL
=032E130     *  2 020 P  4 004 P  7          INTERCHANGE OPERANDS
=032E140          050 P  7 060 P  4
=032E150          020 P  5 004 P  8
=032E160          050 P  8 060 P  5
=032E170          020 P  6 004 P  9
=032E180          050 P  9 060 P  6
=032E190     *  3 020 P  5 024 P  8
=032E200          071    9 002 *  4          TO *4 IF SIGNS ARE ALIKE
=032E210          021 P  9 050 P  9          ELSE COMPLEMENT CF OF SMALLER NUMBER
=032E220     *  4 020 P  4 025 P  7
=032E230          025 * 90 002 *  8          TO *8 IF SFX DIFF AS BIG AS 11
=032E240          024 * 90 072    10
=032E250          024 J 11 052 *  5
=032E260     *  5 004 ---- 060 T  0          SET DECIMAL SHIFTER
=032E270          020 P  9 077    1
=032E280          036 P  6 044 T  0          DEVELOP TWICE SUM IN MQ
=032E290          050 T  2 060 T  1          SAVE PIECES
=032E340     * 10 023 T  1 001 * 11
=032E350          023 T  2 001 * 11
=032E360          050 Q  3 010 *  9          SET ZERO RESULT AND EXIT
=032E370     * 11 020 T  1 025 * 91
=032E380          002 * 13 004 N  1
=032E390          032 T  2 044 T  0          SCALE UP ONE DIGIT
=032E400          060 T  3 004 N  1
=032E410          036 T  1 020 P  4
=032E420     * 12 025 K 14 050 P  4          ADJUST SFX
=032E430          060 T  3 025 * 91
=032E440          002 * 13 032 N  1
=032E450          020 P  4 010 * 12          SCALE UP AND GO TO *12
=032E452     * 13 060 T  3 025 * 92
=032E454          001 * 14 020 P  4
=032E456          024 K 14 050 P  4
=032E458          120    0 044 N  1
=032E460     * 14 060 T  3 024 K  1          ROUND
=032E465          072    1 050 P  6            AND SET CF OF RESULT
=032E470     *  8 020 $  0 010 =  30          PACK UP RESULT
=032E480     *  9 010 ----                    EXIT
=032E490     * 90+          11    8
=032E500     * 91+   200000000   39
=032E510     * 92+  1999999999   39,
```

```
=033E000+     = 33 010 $   0                    MULTIPLY
=033E010           024 K   2 052 *   9
=033E020           020 $   0 010 =  11
=033E030           020 $   0 010 =  13
=033E040           023 P   6 001 *   1
=033E050           050 Q   3 010 *   9
=033E060      *   1 023 Q   3 002 *   9
=033E070           020 $   0 010 =  11
=033E080           020 N   9 070     1
=033E090           004 P   9 036 P   6
=033E100           050 T   0 025 *  90
=033E110           001 *   3 025 N   0
=033E120           002 *   2 021 *  91
=033E130           064 T   1 001 *   3
=033E140      *   2 020 T   0 044 N   9
=033E150           060 P   6 020 P   4
=033E160           024 P   7 024 K  14
=033E170           050 P   4 010 *   6
=033E180      *   3 021 *  92 064 T   1
=033E190           004 T   1 001 *   4
=033E200           020 T   0 010 *   5
=033E210      *   4 020 T   0 025 N   0
=033E220      *   5 044 N   8 060 P   6
=033E230           020 P   4 024 P   7
=033E240           050 P   4 010 $   1
=033E250      *   6 020 P   5 024 P   8
=033E260          124 K  12 050 P   5
=033E270          020 $   0 010 =  30
=033E280      *   9 010 ----
=033E290      *  90+        181898    39
=033E300      *  91+517415400576      39
=033E310      *  92+   450000000      39,
```

```
=034E000+     = 34 010 $  0                    DIVIDE
=034E010           024 K  2 052 *  9
=034E020           023 Q  3 002 E  3
=034E030           020 $  0 010 = 11
=034E040           020 $  0 010 = 13
=034E050           023 Q  3 002 *  9
=034E060           020 $  0 010 = 11
=034E070           020 N  8 050 T  0
=034E080           020 P  4 025 P  7
=034E090           050 P  4 020 P  6
=034E100           025 P  9 002 *  1
=034E110           020 N  9 050 T  0
=034E120           020 P  4 025 K 14
=034E130           050 P  4 010 $  1
=034E140      *  1 020 P  9 072    1
=034E150           004 P  6 036 T  0
=034E160           044 P  9 060 P  6
=034E170           020 P  5 024 P  8
=034E180           124 K 12 050 P  5
=034E190           020 $  0 010 = 30
=034E200      *  9 010 ----          ,
```

```
=035E000+     .= 35 010 $  0                    Y = A*B
=035E010           024 K  2 052 *  9
=035E020           120       050 * 81          SET FLAGS TO ZERO
=035E030           050 * 82 050 * 83
=035E090           020 Q  3 050 * 85           STORE B
=035E100           020 $  0 010 = 11           UNPACK B
=035E110           020 P  4 050 * 86           STORE PARTS IN *86,*87,*88
=035E120           020 P  5 050 * 87
=035E130           020 P  6 050 * 88
=035E140           020 $  0 010 = 13           POP B
=035E141           023 * 88 001 *  1           DOES B=0,
=035E142      *  6 020 N  8 024 * 81           IF YES, RESULT IS ONE PLUS SIGN
=035E143      * 16 050 Q  3 010 *  9           IN FLAG 1
=035E150      *  1 023 Q  3 001 *  2           IF A=0 AND B IS POSITIVE, RESULT IS
=035E160           023 * 87 002 *  9           ZERO.  IF B IS NEGATIVE, ERROR.
=035E172           010 E 23
=035E180      *  2 020 Q  3 050 * 84           STORE A
=035E190           020 $  0 010 = 11           UNPACK A
=035E200           020 P  4 050 * 89           STORE PARTS IN *89,*90,*91
=035E210           020 P  5 050 * 90
=035E220           020 P  6 050 * 91
=035E230           020 * 85 050 Q  3
=035E240           020 $  0 010 F  6           FIND FP OF B
=035E250           023 Q  3 006 *  3           TRANSFER IF B IS AN INTEGER
=035E260           023 * 90 001 E 21           ERROR IF A IS NEGATIVE
=035E270           020 K 12 050 * 82           FLAG 2 IS 2*(-9) FOR B FRACTIONAL
=035E280      *  3 014 *  4 020 * 86
=035E290           072    31 050 T  0          IF SFB IS LESS THAN 9 COMPUTE AND
=035E300           020 I  8 025 T  0           STORE TRUE VALUE OF INTEGRAL B
=035E310           005 *  4 024 J 11          IN *80
=035E320           056 *  5 120
=035E330      *  5 004 * 88 044 ----          TEST UNITS POSITION FOR B ODD OR
=035E340           060 * 80 071    39          EVEN. (SHIFT TO SIGN BIT)
=035E350           006 *  4 020 * 90          IF B IS ODD STORE SIGN OF A
=035E360      *  4 050 * 81 020 * 84          IN FLAG 1
=035E370           125 K 12 050 * 84           STORE ABSOLUTE VALUE OF A
=035E380           025 N  8 050 T  1           IF IT EQUALS ONE, THEN RESULT
=035E390           023 T  1 002 *  6           IS PLUS OR MINUS ONE
=035E400           020 * 87 050 * 83           FLAG 3 EQUALS SIGN OF B
=035E410           020 * 85 125 K 12           STORE ABSOLUTE VALUE OF B
=035E420           050 * 85 023 * 82           IF FLAG 2 IS ZERO, B IS AN INTEGER
=035E430           002 *  7 020 * 85           TRANSFER
=035E440           025 * 93 050 T  0
=035E450           020 * 84 050 Q  3           STORE A IN Q3, THEN TEST
=035E460           023 T  0 001 *  8           IF B EQUALS 1/2
=035E470           020 $  0 010 F  0           COMPUTE SQUARE ROOT OF A
=035E480           023 * 83 002 *  9           IF FLAG 3 IS ZERO, EXIT
=035E490           020 I  1 050 * 80           TRANSFER TO INTEGER ARITHMETIC TO
=035E502           050 * 87 020 Q  3           COMPUTE RECIPROCAL OF SQUARE ROOT
=035E504           050 * 84 010 * 29
=035E530      *  8 020 $  0 010 F  1           COMPUTE E*(B.LOG(A)) FOR B
=035E540           020 $  0 010 = 11           FRACTIONAL OR INTEGER GREATER THAN 9
=035E550           020 P  5 024 * 87
=035E560           124 K 12 025 I  1           TEST SIGN FOR B.LOG(A), IF NEGATIVE
```

```
=035E570          001 * 26 020 P  4     TEST SF FOR RESULT
=035E580          024 * 86 072    31    IF SF GREATER THAN 2, E*(-B.LOG(A))=0
=035E590          025 I  3 006 * 23
=035E600    * 26 020 $  0 010 = 12
=035E610          020 * 85 024 * 87
=035E620          050 Q  3 010 $  1
=035E630          020 $  0 010 = 33     MULTIPLY B.LOG(A)
=035E680          020 $  0 010 F  2     COMPUTE  E* (B(LOG(A))
=035E690    * 20 020 Q  3 014 *  6      TRANSFER TO ADD FLAG 1
=035E700    *  7 020 * 91 025 N  8      B IS AN INTEGER, TEST A
=035E710          025 I  1 002 * 14     IF A IS NOT A POWER OF 10, TRANSFER
=035E720          022 * 86 025 K 14     IF ABSOLUTE VALUE OF SFB IS
=035E730          025 K 14 005 * 15     GREATER THAN ONE, TEST SIGN FOR
=035E740          020 * 87 024 I  1     SF FOR RESULT, + IS OVERFLOW,
=035E745          071    49 032 * 89    - IS UNDERFLOW
=035E750    * 23 002 E  1 120
=035E755    * 15 010 * 16 004 * 80
=035E760          023 * 87 002 * 18     ATTACH SIGN TO B
=035E765          061 * 80 004 * 80     B. SFA
=035E770    * 18 032 * 89 076    0      SHIFT SIGN BIT
=035E771          001 * 25 025 I  1     TEST FOR NON-ZERO BIT IN ACCUMULATOR
=035E772          002 E  1 010 * 24     (IS SF MAGNITUDE GREATER THAN 511)
=035E773    * 25 024 I  1 005 * 23
=035E775    * 24 060 P  4 020 * 81
=035E780          050 P  5 020 * 91
=035E785          050 P  6 010 * 17
=035E790    * 17 020 $  0 010 = 30     PACK
=035E795    *  9 010 ----              EXIT
=035E800    * 14 020 * 84 050 Q  3     SET Q3 EQUAL TO ABSOLUTE VALUE OF A
=035E810          020 K 14 025 * 86     IF B IS GREATER THAN 29, TRANSFER TO
=035E812          001 *  8 020 * 80    COMPUTE E*(B.LOG(A))
=035E814          025 * 94 002 *  8
=035E816    * 29 004 N  8 060 * 88     INITIALIZE PRODUCT
=035E817          023 * 87 002 * 27
=035E818          060 Q  3 010 $  1     SET Q3=1 FOR DIVISION
=035E820          020 $  0 010 = 12     PUSH
=035E822          020 * 84 050 Q  3     Q3 = A
=035E826          020 $  0 010 = 34     IF B NEGATIVE, COMPUTE 1/A
=035E828    * 27 020 Q  3 050 * 91     STORE MULTIPLIER
=035E830          020 * 80 072    1     TEST SUCCESSIVE DIGITS OF B, IF ONE
=035E832          050 * 80 061 T  0     MULTIPLY CURRENT PRODUCT BY CURRENT
=035E834          002 * 28 010 $  1     POWER OF A*N OR (1/A)*N
=035E836          020 $  0 010 = 12     PUSH A*N
=035E838          020 * 88 050 Q  3
=035E840          020 $  0 010 = 33     MULTIPLY
=035E842          020 Q  3 050 * 88
=035E844    * 28 023 * 80 002 * 20     IF B=0, DONE, OTHERWISE
=035E846          020 * 91 050 Q  3     COMPUTE NEXT POWER
=035E848          020 $  0 010 = 12     PUSH A*N
=035E850          020 $  0 010 = 33     COMPUTE A*(2N)
=035E852          010 * 27
=035E870    * 80                        B IF IT IS AN INTEGER
=035E875    * 81                        FLAG 1
=035E880    * 82                        FLAG 2
=035E885    * 83                        FLAG 3
```

```
=035E890     * 84                          A PACKED -- THEN ABSOLUTE A PACKED
=035E895     * 85                          B PACKED -- THEN ABSOLUTE B PACKED
=035E900     * 86                          SFX B
=035E905     * 87                          SIGN B
=035E910     * 88                          COEF B, REPLACED BY 10*8 IN * 29 +2
=035E915     * 89                          SFX A
=035E920     * 90                          SIGN A
=035E925     * 91                          COEF A, REPLACED BY A*N IN * 27
=035E935     * 93 077,6356,326,2400        1/2(10*8)
=035E940     * 94                  30,
```

```
=036E000+     = 36 010 $  0                    CONVERT H20 INTEGER FOR OUTPUT FROM S
=036E010           024 K  2 052 *  1            020 $  0 010 = 36
=036E020           024 K  2 052 *  9                   K         N
=036E030      *  1 020 ---- 024 * 90            WHERE K=OPTION AND N=NBR OF DIGITS
=036E040           052 * 10 124 K  5            K=1 SKIP LEADING ZEROS
=036E050           025 I  1 050 T  0            K=2 SPACE FOR LEADING ZEROS
=036E060           120    0 050 *  3            K=3 FORCE LEADING ZEROS
=036E070      * 10 010 ----                     K=4 FORCE LEADING, SKIP TRAILING ZE
=036E080      * 11 020 * 91 014 *  2
=036E090      * 12 020 * 92 014 *  2
=036E100      * 13 020 *  6 014 *  2
=036E110      * 14 020 * 94 050 *  3
=036E120      *  2 020 *  6 050 *  5
=036E130      *  3 --- ---- --- ----
=036E140           020 T  0 024 J 11
=036E150           056 *  4 120    0
=036E160      *  4 004 H 20 044 ----
=036E170           050 H 20 063 T  1
=036E180           002 *  5 020 *  6
=036E190           050 *  5 010 * 20
=036E200      *  5 --- ---- --- ----
=036E210      *  6 004 A 48 010 * 20
=036E220      *  7 004 A 14 010 * 20
=036E230      * 20 020 P 12 024 K  3
=036E240           050 P 12 056 * 21
=036E250      * 21 060 P 11 050 ----
=036E260      * 22 020 T  0 025 I  1
=036E270           050 T  0 002 *  3
=036E280      *  9 010 ----                     EXIT
=036E290      * 90      * 10
=036E300      * 91 023 T  0 001 * 22
=036E310      * 92 023 T  0 001 *  7
=036E320      * 94 023 H 20 002 *  9,
```

```
=037E000+     = 37 024 K   2 010 $   0      SIMPLE DIRECT TESTS
=037E010           052 *   9 020 Q   9
=037E020           002 E   9 010 *   1      ERROR IF INDIRECT, JUMP INTO =38
```

```
=038E000     = 38 024 K   2 010 $   0     SIMPLE INDIRECT TESTS
=038E010          052 *   9 020 Q   9
=038E020          001 E 10 010 *   1     ERROR IF DIRECT
=038E030     *  1 020 $   0 010 =   3     ELIMINATE SPACES
=038E040          124 K 19 025 K  19
=038E050     *  9 002 ---- 010 E   6,    EXIT IF TERMINAL ELSE MALFORMED
```

```
=039E000+     = 39 050 T  0 010 $  0     VERIFY SPACE AND ADVANCE TO NONSPACE
=039E010           020 P  1 124 A 14
=039E020           025 A 14 001 E  6     ERROR IF NOT SPACE
=039E030           020 T  0 010 =  2,    WADE INTO ADVANCE TO NONSPACE
```

```
=040E000+     = 40 010 $  0                    POSITIVE INTEGER TEST ON (Q3)
=040E010           024 K  2 052 *  9
=040E020           020 $  0 010 = 11
=040E030           021 P  6 001 *  1
=040E040      *  2 020 K  4 010 *  9
=040E050      *  1 021 P  5 001 *  9
=040E060           020 P  4 001 *  9
=040E070           025 J 18 002 *  2          (J18) = 9X2*(-8)
=040E080           021 P  4 072    10
=040E090           024 J 14 052 *  3
=040E100           120    0 004 P  6
=040E110      *  3 044 ---- 025 K  1
=040E120           002 *  2 120    0
=040E130      *  9 010 ----          ,         EXIT + OR - A/C TEST
```

```
=041E000+     = 41 010 $  0                    SUBSCRIPT RANGE TEST AND REPL (Q3)
=041E010           024 K  2 052 *  9
=041E020           023 Q  3 002 *  9
=041E030           020 $  0 010 = 11
=041E040           023 P  5 001 E  4
=041E050           020 P  4 001 E  4
=041E060           025 K 14 002 *  1
=041E070           020 N  8 014 *  2
=041E080     *  1 025 K 14 002 E  4
=041E090     *  2 020 N  7 050 T  0
=041E100           120    0 004 P  6
=041E110           044 T  0 025 K  1
=041E120           002 E  4 060 T  0
=041E130     *  3 071   12 050 Q  3
=041E140     *  9 010 ----                    ,
```

```
=042E000+      = 42 130 = 42 134 = 42
```

```
=043E000+      = 43 130 = 43 134 = 43
```

```
=044E000+     = 44 010 $  0                          PART NUMBER TEST ON (Q3)
=044E010           024 K  2 052 *  9
=044E020           020 $  0 010 = 40
=044E030      *  9 002 ---- 010 E 18,
```

```
=045E000+     = 45 010 $  0                          STEP NUMBER TEST ON (Q3)
=045E010           024 K  2 052 *  9
=045E020           023 Q  3 002 E 13
=045E030           020 Q  3 071    1
=045E040           001 E 13 071    8
=045E050           001 E 13 020 Q  3
=045E060           025 J 18 002 E 13
=045E070      *  9 010 ----                        ,
```

```
=046E000+     = 46 010 $  0                          FORM NUMBER TEST ON (Q3)
=046E010           024 K  2 052 *  9
=046E020           020 $  0 010 = 40
=046E030      *  9 002 ---- 010 E 12,
```

```
=047E000+     = 47 010 $  0                          PUT STN AT END OF STATE IN $1.
=047E010           024 K  2 052 *  6
=047E020           024 K  2 052 *  9
=047E030           020 H 12 024 U  0
=047E040           052 *  1 056 *  6
=047E050           020 H 12 024 U  3
=047E060           052 *  4 056 *  5
=047E070      *  1 020 ---- 056 *  2
=047E080      *  2 056 *  4 020 ----
=047E090           050 T  0 025 H 12
=047E100           005 *  3 025 K  1
=047E110      *  3 001 *  4 020 T  0
=047E120           024 U  3 014 *  1
=047E130      *  4 020 ---- 050 ----                 CLOSE GAP IN OLD STATE LIST
=047E140      *  5 020 K  4 050 ----                 SET NULL SUCCESSOR FOR STN
=047E150      *  6 020 ---- 050 ----                 SET NEW STATE FOR STN
=047E160      *  7 052 *  8 056 * 10
=047E170      *  8 020 ---- 001 * 10
=047E180           024 U  3 010 *  7
=047E190      * 10 020 H 12 050 ----                 ADD STN TO END OF NEW STATE LIST
=047E200      *  9 010 ----                    ,
```

```
=048E000+     = 48 010 $  0                        'TO' ROUTINE
=048E010           024 K  2 052 *  9
=048E030           020 H  9 124 K 17
=048E040           142 K 35 050 H 12      ERASE SIGNAL
=048E050           071   21 052 H 12      SET STN CONTEXT
=048E060           020 $  0 010 = 58      RELEASE CURRENT BUFFER
=048E070           001 *  1 020 H 12
=048E080           024 K 24 142 K 24      TL NEXT BUFFER TO STN
=048E090     *  1 020 H 12 024 U  0
=048E100           052 *  2 020 K  5
=048E110     *  2 124 ---- 025 * 90
=048E120           001 *  8 025 I  1
=048E130           001 *  3 025 I  7      TO *3 IF STATE=L6
=048E140           001 *  8 025 I  1
=048E150           001 *  5 010 *  8
=048E160     *  3 020 H 13 024 U  5      STATE=L6
=048E170           052 *  4 010 $  1
=048E180     *  4 020 ---- 002 *  8
=048E190           020 $  0 010 = 47      ON LAST BUFFER - SWITCH TO USER
=048E200              L 10     L 10
=048E210           020 H 12 024 K 36
=048E220           142 K 36 010 *  8      CL+SU THEN EXIT
=048E230     *  5 020 J 17 050 T  0      STATE = L14
=048E240           020 H 12 024 U  1
=048E250           052 *  7 010 *  7
=048E260     *  6 024 U  5 052 *  7
=048E270           020 T  0 025 I  1
=048E280           050 T  0 001 *  8      EXIT IF STILL CHOKING
=048E290     *  7 020 ---- 002 *  6      LOOP IF BUFFER TO COUNT
=048E300           020 $  0 010 = 47      ELSE UNCHOKE
=048E310              L 11     L 11
=048E315     *  8 020 L 13 050 H 12      RESET H12 TO STN IN L13
=048E320     *  9 010 ----               EXIT
=048E330     * 90                 L  6,
```

```
=049E000+     = 49 024 K   2 010 $   0      UPDATE ACTIVITY OF STN
=049E010           052 *   9 020 H  12
=049E020           024 U   4 052 *   1
=049E030           020 $   0 010 =   0
=049E040      *   1 050 ---- 010 $   1
=049E050      *   9 010 ----           ,
```

```
=050E000+     = 50 010 $  0                    FIND PART (P20)
=050E010           024 K  2 052 *  9
=050E020           020 J 10 050 H 22
=050E030           020 Q 10 050 H 23
=050E040      *  1 020 H 22 050 H 21
=050E050           020 H 23 050 H 22
=050E060           056 *  2 024 K  1
=050E070      *  2 056 *  3 020 ----
=050E080           050 H 23 001 *  9     EXIT MINUS IF CAN'T FIND
=050E090      *  3 020 P 20 025 ----
=050E100           001 *  1 025 K  1
=050E110           002 *  1 020 H 23     OUTPUT LOCATOR IN ACC
=050E120      *  9 010 ----          ,    H21, H22, H23 HOLD USEFUL POINTERS
```

```
=051E000+     = 51 010 $  0                          FIND STEP (P21)
=051E010           024 K  2 052 *  9
=051E020           020 P 21 070    31
=051E030           124 K  5 050 H 21
=051E040           020 J 14 025 H 21
=051E050           056 *  4 010 $  1
=051E060           020 J 10 050 H 22
=051E070           020 Q 10 050 H 23
=051E080      *  1 020 H 22 050 H 21
=051E090           020 H 23 050 H 22
=051E100           056 *  2 024 K  1
=051E110      *  2 056 *  3 020 ----
=051E120      *  8 050 H 23 001 *  9      EXIT MINUS IF CAN'T FIND
=051E130      *  3 020 P 21 025 ----
=051E140      *  4 001 *  8 025 ----      EXIT MINUS IF CAN'T FIND AND LEAVE H2
=051E150           002 *  1 020 H 22
=051E160           071    21 124 K  6
=051E170           050 H 25 020 H 23
=051E180           050 H 26 010 $  1
=051E190      *  5 020 H 25 050 H 24
=051E200           020 H 26 050 H 25
=051E210           052 *  6 024 K  2
=051E220           052 *  7 010 $  1
=051E230      *  6 020 ---- 050 H 26
=051E240           001 *  9 020 P 21      EXIT MINUS IF CAN'T FIND
=051E250      *  7 025 ---- 001 *  9      EXIT MINUS IF CAN'T FIND
=051E260           025 K  1 002 *  5
=051E270           020 H 26 010 $  1      OUTPUT LOCATOR IN ACC
=051E280      *  9 010 ----          ,     H21-H26 HOLD USEFUL INFO ON EXIT
```

```
=052E000+     = 52 010 $  0                          FIND FORM (P22)
=052E010           024 K  2 052 *  9
=052E020           020 J 10 050 H 25
=052E030           020 Q 11 050 H 26
=052E040      *  1 020 H 25 050 H 24
=052E050           020 H 26 050 H 25
=052E060           052 *  2 024 K  2
=052E070           052 *  3 010 $  1
=052E080      *  2 020 ---- 050 H 26
=052E090           001 *  9 020 P 22                 EXIT MINUS IF CAN'T FIND
=052E100      *  3 025 ---- 001 *  9
=052E110           025 K  1 002 *  1
=052E120           020 H 26 010 $  1                 OUTPUT LOCATOR IN ACC AND H26
=052E130      *  9 010 ----      ,                   H24 LOCATES PREDECESSOR ON EXIT
```

```
=053E000+     = 53 010 $  0                    KICK OUT CURRENT PROGRAM (IF ANY)    T
=053E010           024 K  2 052 *  9
=053E020           020 H  7 001 *  9
=053E030           024 U  2 052 *  1
=053E032           020 $  0 010 =  0           ACCUMULATE TIME WORKED
=053E034           025 H  8 024 J 23
=053E036           050 J 23 010 $  1
=053E040     *  1 020 ---- 055 * 90
=053E050           004 * 90 111,6000           (FOR 1024 WORD USER BLOCK)
=053E060           020 K  4 050 H  7
=053E070     *  9 010 ----
=053E080     * 90 000,0000,000,1777,          CONSTRUCTED DRUM CONTROL WORD
```

```
=054E000+     = 54 010 $  0                    BRING IN PROGRAM (IF ANY) A/C STN    T
=054E010          024 K  2 052 *  9
=054E020          020 $  0 010 = 53           KICK OUT CURRENT PROGRAM IF ANY
=054E030          020 H 12 024 U  2
=054E040          052 *  1 010 $  1
=054E050      *  1 020 ---- 001 *  9
=054E060          055 * 90 010 $  1
=054E070          004 * 90 110,6000           (FOR 1024 WORD USER BLOCK)
=054E080          020 H 12 050 H  7
=054E090          020 $  0 010 = 49           UPDATE STATION ACTIVITY
=054E100          050 H  8 010 $  1           SET START TIME OF CURRENT SHOT
=054E110      *  9 010 ----
=054E120      * 90 000,0000,000,1777,         CONSTRUCTED DRUM CONTROL WORD
```

```
=055E000+     = 55 010 $  0                        SWITCH TO USER (CL+SU)
=055E010           024 K  2 052 *  9               (ASSUMES SUBSTATE OF L10 ALREADY IN P
)=055E015          020 K  4 050 P 16               CLEAR P16 BEFORE SWITCHING
=055E020           050 P 15 020 H 12               ALSO CLEAR P15
=055E030           024 U  1 052 *  1
=055E032           020 Q  8 001 *  1               JUMP IF Q8 MINUS
=055E034           050 Q 12 020 Q  9               HIDE Q8-Q9
=055E036           050 Q 13 020 K  4
=055E038           050 Q  8 050 Q  9
=055E040      *  1 020 ---- 001 *  3
=055E050           024 U  5 052 *  2
=055E060      *  2 020 ---- 001 *  4
=055E070           020 $  0 010 = 47               CHANGE STATE TO L6 WHILE DRAINING
=055E080              L  6     L  6
=055E090           010 *  9
=055E100      *  3 020 $  0 010 =  8               ASSIGN INPUT BUFFER
=055E110           002 *  4 010 $  1               TO *4 IF ASSIGNED
=055E120           020 $  0 010 = 47               CHANGE STATE TO L7 IF NO BUFFER AVAIL
=055E130              L  7     L  7
=055E140           010 *  9
=055E150      *  4 020 H 12 024 K 36               OK TO SWITCH
=055E160           142 K 36 010 $  1               CL+SU
=055E170           020 $  0 010 = 47               CHANGE STATE TO L10 GREEN
=055E180              L 10     L 10
=055E190      *  9 010 ----                       ,
```

```
=056E000+     = 56 010 $  0                    ERASE STEP A/C H21-H26
=056E010          024 K  2 052 *  9
=056E020          020 $  0 010 = 57            STEP LOOKS JUST LIKE FORM HERE
=056E030          020 H 22 056 *  1
=056E040      *  1 020 K  6 124 ----
=056E050          025 K  1 002 *  9            EXIT UNLESS PART NOW EMPTY
=056E060          020 H 21 056 *  2
=056E070      *  2 020 H 23 056 ----
=056E080          020 H 22 071   61
=056E090          020 $  0 010 = 28            ERASE LEFT LINKED LIST (ONE ITEM HERE
=056E100      *  9 010 ----          ,
```

```
=057E000+     = 57 010 $  0                          ERASE FORM A/C H24-H26
=057E010          024 K  2 052 *  9
=057E020          020 H 24 052 *  1
=057E030          020 H 26 010 $  1
=057E040      *  1 052 ---- 010 $  1
=057E050          020 H 25 071    19
=057E060          020 $  0 010 = 29                  ERASE RIGHT LINKED LIST A/C MQ
=057E070      *  9 010 ----          ,
```

```
=058E000+     = 58 010 $  0                      RELEASE CURRENT BUFFER FOR STN
=058E010           024 K  2 052 *  9               SERVICES L7 WITH FREE BUFFER AND
=058E020           020 H 12 050 * 92               LEAVES NEXT BUFFER OR -1 IN ACC AND
=058E030           024 U  1 052 *  1
=058E040           056 *  8 010 $  1
=058E050      *  1 020 ---- 050 H 13
=058E055           002 $  1 010 *  9
=058E060           024 U  5 056 *  2
=058E070      *  2 056 *  3 004 ----             PICK UP NEXT BUFFER (IF ANY)
=058E080      *  3 020 K  4 050 ----             SET SUCCESSOR OF OLD BUFFER TO -1
=058E090           060 * 93 050 T  0
=058E100           020 H 12 056 T  0
=058E110           020 T  0 142 K 16               SET BUFFER BITS IN SCR FOR NEW BUFFER
=058E120           020 * 91 010 $  1               FIND TAIL OF H10 LIST
=058E130      *  4 052 *  5 056 *  6
=058E140      *  5 020 ---- 001 *  6
=058E150           024 U  5 010 *  4
=058E160      *  6 020 H 13 050 ----             PUT OLD BUFFER ON END OF H10 LIST
=058E170           020 L  7 005 *  7               SERVICE L7
=058E180           050 H 12 010 $  1
=058E190           020 $  0 010 =  8               ASSIGN BUFFER
=058E200           020 $  0 010 = 47               CHANGE STATE TO L10
=058E210                L 10     L 10
=058E220           020 H 12 024 K 36               CL+SU
=058E230           142 K 36 020 * 92
=058E240      *  7 050 H 12 020 * 93             RESTORE STN CONTEXT
=058E250      *  8 050 H 13 050 ----             LEAVE NEXT BUFFER OR -1 IN ACC AND H1
=058E260      *  9 010 ----                      EXIT
=058E270      * 91      H 10     H 10
=058E280      * 92                              PRIVATE STORAGE FOR STN
=058E290      * 93                              PRIVATE STORAGE FOR BUFFER
```

=059E000+ = 59 130 = 59 134 = 59

```
=060E000+      = 60 130 = 60 134 = 60
```

```
=061E000+     = 61 010 $  0                      CONVERT TIME FOR OUTPUT
=061E010           024 K  2 052 *  9
=061E020           020 $  0 010 =  0              READ THE CLOCK
=061E030           120    0 004 H  1
=061E040           044 J  1 120    0
=061E050           044 J  0 036 N  2
=061E060           060 H 20 010 $  1
=061E070           020 $  0 010 = 36
=061E080                  3       4
=061E090      *  9 010 ----        ,
```

```
=062E000+     = 62 050 T  0 010 $  0      INSERT SPACE IN OUTPUT REGION S
=062E010           020 A 14 050 P 11
=062E020           020 T  0 010 = 66,
```

=063E000+ = 63 130 = 63 134 = 63

```
=064E000+     = 64 010 $  0                    UNPACK MSG TO S FOR OUTPUT COMPOSITIC
=064E010           024 K  2 052 *  1            020 $  0 010 = 64
=064E020           024 K  2 052 *  9                   M         M
=064E030      *  1 020 ---- 052 *  2            WHERE 'M' IS LOCN OF MSG
=064E040      *  2 020 ---- 050 T  0
=064E050      *  3 020 T  0 001 *  9
=064E060           071    8 050 T  0
=064E070           020 P 12 024 K  3
=064E080           050 P 12 052 *  4
=064E090      *  4 060 ---- 023 T  0
=064E100           001 *  3 020 *  2
=064E110           024 K  2 014 *  1
=064E120      *  9 010 ----          ,
```

=065E000+ = 65 130 = 65 134 = 65

```
=066E000+     = 66 010 $  0                    PUT (P11) IN NEXT S CELL (P12)+1
=066E010           024 K  2 052 *  9
=066E020           020 P 12 024 K  3
=066E030           050 P 12 056 *  1
=066E040      *  1 020 P 11 050 ----
=066E050      *  9 010 ----          ,
```

=067E000+ = 67 130 = 67 134 = 67

```
=068E000+     = 68 010 $  0                          COMPUTE PART (P20) FOR STEP (P21)
=068E010           024 K  2 052 *  9
=068E020           020 P 21 124 K 11
=068E030           050 T  0 070    10
=068E040           050 T  1 020 J 14
=068E045           025 T  1 052 *  1                  ASSUMES (P21) IS LEGITIMATE
=068E050           052 *  2 020 P 21
=068E060           124 K 13 072    39
=068E070      *  1 044 ---- 020 T  0
=068E080      *  2 036 ---- 060 P 20
=068E090      *  9 010 ----
```

=069E000+ = 69 130 = 69 134 = 69

```
=070E000+     = 70 010 $  0                    COMPARE (H21) W (H22)
=070E010      *  1 024 K  2 052 *  9
=070E020            020 H 21 025 H 22
=070E030            005 *  2 025 K  1
=070E040      *  2 001 * 12 023 H 22
=070E050            005 *  3 020 H 21
=070E060            071    9 001 * 11
=070E070      *  3 010 * 13 023 H 21
=070E080            005 *  4 020 H 22
=070E090            071    9 001 * 13
=070E100      *  4 010 * 11 020 H 21
=070E110            124 K 12 024 H 22
=070E120            124 K 12 025 K 12
=070E130            005 *  5 020 H 21
=070E140            071    9 001 * 11
=070E150      *  5 010 * 13 020 H 22
=070E160            071    1 072    1
=070E170            050 T  0 020 H 21
=070E180            071    1 072    1
=070E190            025 T  0 001 *  6
=070E200            020 H 21 071    9
=070E210            002 * 13 010 * 11
=070E220      *  6 020 H 21 071    9
=070E230            002 * 11 010 * 13
=070E240      * 11 020 K 26 010 *  9     LESS
=070E250      * 12 020 K 25 010 *  9     EQUAL
=070E260      * 13 020 K 24 010 *  9     GREATER
=070E270      *  9 010 ----         ,    EXIT WITH RELATION IN ACC
```

```
=071E000+     = 71 010 $  0                    FIND SINGLE INDEXED VALUE
=071E010          024 K  2 052 *  9              H27 HOLDS LETTER
=071E020          020 H 27 052 *  1              H28 HOLDS INDEX
=071E030          070   21 050 H 22              ASSUME LETTER IS SINGLY INDEXED
=071E040     *  1 020 ---- 050 H 23
=071E050     *  2 020 H 22 050 H 21
=071E060          020 H 23 056 *  3
=071E070     *  3 050 H 22 020 ----
=071E080          050 H 23 001 *  9              JUMP IF CAN'T FIND AND EXIT MINUS
=071E090          124 K 42 025 H 28
=071E100          001 *  2 025 K  1              JUMP IF NO MATCH AND TRY NEXT
=071E110          006 *  9 020 H 22              JUMP IF NO MATCH POSSIBLE
=071E120          024 K  1 124 K  5
=071E130     *  9 010 ---- 020 K  4              LEAVE + POINTER IN ACC IF FOUND
=071E014          050 H 23 010 *  9,             EXIT - WITH H23 MINUS.
```

```
=072E000+     = 72 010 $  0                    FIND DOUBLY INDEXED VALUE
=072E010          024 K  2 052 *  9              H27 HOLDS LETTER
=072E020          020 $  0 010 = 71              H28 HOLDS FIRST INDEX
=072E030          001 *  9 020 H 22              H29 HOLDS SECOND INDEX
=072E040          071   21 050 H 25           ASSUME LETTER IS DOUBLY INDEXED
=072E050          020 H 23 050 H 26
=072E060     *  1 020 H 25 050 H 24
=072E070          020 H 26 050 H 25
=072E080          052 *  2 010 $  1
=072E090     *  2 020 ---- 050 H 26
=072E100          001 *  9 124 K 42
=072E110          025 H 29 001 *  1
=072E120          025 K  1 001 *  3
=072E130          020 K  4 010 *  9
=072E140     *  3 020 H 25 024 K  2
=072E150     *  9 010 ----          ,        EXIT
```

```
=073E000+     = 73 010 $  0                      ASSIGN VALUE A/C H27-H30
=073E010          024 K  2 052 *  9
=073E020          020 H 29 002 * 20               JUMP FOR MATRIX ELEMENT
=073E030          020 H 28 002 * 10               JUMP FOR VECTOR ELEMENT
=073E040          020 H 27 052 *  1               SIMPLE VALUE TO ASSIGN
=073E050          052 *  3 010 $  1
=073E060     *  1 020 ---- 002 *  2               JUMP IF ALREADY SIMPLE
=073E070          025 K  1 002 *  2               JUMP IF UNDEFINED
=073E080          020 $  0 010 = 75               ERASE VALUES FOR LETTER H27
=073E090     *  2 020 H 30 010 $  1
=073E100     *  3 050 ---- 010 $  1               ASSIGN SIMPLE VALUE
=073E110     *  9 010 ----                        EXIT
=073E120     * 10 020 Q  0 001 E  0               VECTOR ELEMENT TO ASSIGN
=073E130          020 H 27 052 * 11
=073E140     * 11 020 ---- 002 * 12
=073E150          071    6 001 * 14               JUMP IF LETTER NOW A VECTOR
=073E160     * 12 020 $  0 010 = 75               ERASE VALUE(S) FOR LETTER H27
=073E170          020 H 27 052 * 13
=073E180          020 K 39 010 $  1
=073E190     * 13 050 ---- 010 $  1
=073E200     * 14 020 $  0 010 = 71               FIND SINGLE INDEXED VALUE
=073E210          056 * 17 002 * 17               JUMP IF FOUND
=073E220          020 H 21 056 * 15
=073E230          020 $  0 010 = 10               GET A SPACE
=073E240     * 15 056 * 16 056 ----
=073E250          024 K  1 056 * 17
=073E260          020 H 22 124 K  5
=073E270     * 16 024 H 28 050 ----
=073E280     * 17 020 H 30 050 ----
=073E290          010 *  9
=073E300     * 20 020 Q  0 056 * 21               MATRIX ELEMENT TO ASSIGN
=073E310     * 21 001 E  0 020 ----               CHECK SPACE
=073E320          002 $  1 010 E  0
=073E330          020 H 27 052 * 22
=073E340     * 22 020 ---- 002 * 23
=073E350          071    5 001 * 25               JUMP IF LETTER NOW A MATRIX
=073E360     * 23 020 $  0 010 = 75               ERASE VALUE(S) FOR LETTER H27
=073E370          020 H 27 052 * 24
=073E380          020 K 38 010 $  1
=073E390     * 24 050 ---- 010 $  1
=073E400     * 25 020 $  0 010 = 72               FIND DOUBLE INDEXED VALUE
=073E410          052 * 31 006 * 30               JUMP IF FOUND
=073E420          020 H 23 002 * 28               JUMP IF ROW EXISTS
=073E430          020 H 21 056 * 26
=073E440          020 $  0 010 = 10               GET A SPACE
=073E450     * 26 056 * 27 056 ----
=073E460          071   21 050 H 24
=073E470          020 H 22 124 K  5
=073E480     * 27 024 H 28 050 ----
=073E490          050 H 25 010 $  1
=073E500     * 28 020 H 24 052 * 29
=073E510          020 $  0 010 = 10               GET A SPACE
=073E520          071   21 052 * 30
=073E530     * 29 052 ---- 024 K  2
```

```
=073E540          052 * 31 020 H 25
=073E550          124 K  6 024 H 29
=073E560     * 30 050 ---- 020 H 30
=073E570     * 31 050 ---- 010 *  9,
```

```
=074E000+     = 74 010 $  0                    EVALUATE INDEXED LETTER
=074E010           024 K  2 052 *  9
=074E020           020 Q  7 050 H 29
=074E030           020 Q  6 001 *  1           JUMP IF Q7 EMPTY
=074E040           020 $  0 010 = 17           POP Q7
=074E050           020 Q  6 002 E  6           MALFORMED IF EXTRA INDICES
=074E060      *  1 020 $  0 010 = 41           CHECK AND CONVERT FIRST INDEX
=074E070           020 Q  3 050 H 28
=074E080           020 Q  5 024 J 13
=074E090           071   21 052 *  2
=074E100      *  2 020 ---- 050 H 27           SET LETTER
=074E110           052 *  3 052 *  6
=074E120           020 H 29 002 *  5           JUMP IF DOUBLE INDEX
=074E130      *  3 020 ---- 002 E  2           UNDEFINED IF SCALAR
=074E140           071    6 002 E  2           UNDEFINED IF NOT VECTOR
=074E150           020 $  0 010 = 71
=074E160           001 E  2 056 *  4           UNDEFINED IF CAN'T FIND
=074E170      *  4 070    0 020 ----
=074E180           050 Q  3 010 *  9           SET RESULT AND EXIT
=074E190      *  5 050 Q  3 010 $  1
=074E200           020 $  0 010 = 41           CHECK AND CONVERT SECOND INDEX
=074E210           020 Q  3 050 H 29
=074E220      *  6 020 ---- 002 E  2           UNDEFINED IF SCALAR
=074E230           071    5 002 E  2           UNDEFINED IF NOT MATRIX
=074E240           020 $  0 010 = 72
=074E250           001 E  2 052 *  7           UNDEFINED IF CAN'T FIND
=074E260      *  7 020 ---- 050 Q  3           SET RESULT
=074E270      *  9 010 ----          ,         EXIT
```

```
=075E000+     = 75 010 $  0                    ERASE VALUE(S) FOR LETTER H27
=075E010          024 K  2 052 *  9
=075E020          020 H 27 052 *  1
=075E030          052 * 12 010 $  1
=075E040      *  1 020 ---- 002 * 11           JUMP IF LETTER HAS SIMPLE VALUE
=075E050          050 * 90 071    5
=075E060          001 * 20 071    1            JUMP IF MATRIX
=075E070          001 * 10 010 *  9            JUMP IF VECTOR ELSE UNDEFINED
=075E080      * 10 004 * 90 010 $  1           VECTOR
=075E090          020 $  0 010 = 29            ERASE RIGHT LINKED LIST
=075E100      * 11 020 K  4 010 $  1
=075E110      * 12 050 ---- 010 $  1           UNDEFINE THE LETTER
=075E120      *  9 010 ----                    EXIT
=075E130      * 20 020 * 90 014 * 21           MATRIX
=075E140      * 21 020 * 91 056 * 22
=075E150          124 K  5 025 K  1
=075E160      * 22 001 * 10 004 ----
=075E170          060 * 91 010 $  1
=075E180          020 $  0 010 = 28            ERASE LEFT LINKED LIST
=075E190          010 * 21
=075E200      * 90                             LOCAL STORAGE
=075E210      * 91                         ,   LOCAL STORAGE
```

```
=076E000+     = 76 010 $  0                    ERASE ONE LEVEL OF CONTROL STRUCTURE
=076E010           024 K  2 052 *  9
=076E020           020 Q  8 052 *  1           ASSUME STRUCTURE EXISTS OFF Q8L
=076E030      *  1 020 ---- 005 *  2
=076E040           052 *  1 075    40
=076E050           020 $  0 010 = 29
=076E060      *  2 010 *  1 004 Q  8
=076E070           020 $  0 010 = 28
=076E080           020 $  0 010 = 19
=076E090      *  9 010 ----          ,
```

```
=077E000+     = 77 010 $  0                    MATCH GROUPERS
=077E010           024 K  2 052 *  9
=077E020           020 P  1 052 *  1
=077E030           050 J 20 124 K 32          LEAVE RIGHT GROUPER FOR D0 AND G14
=077E040           025 K 32 001 E  6          ERROR IF NOT RIGHT GROUPER
=077E050      *  1 020 ---- 124 Q  5
=077E060           025 Q  5 001 E  6          ERROR IF NO MATCH WITH PREV OPERATOR
=077E070           020 $  0 010 = 15          POP OPERATOR
=077E080      *  9 010 ----          ,        EXIT
```

```
=078E000+     = 78 010 $  0                    TRANSMIT LINE AND RESTORE
=078E010           024 K  2 052 P 25          DRUM ROUTINE IF NECESSARY
=078E020           020 H 17 050 P 24          SAVE DRUM CW
=078E030           020 $  0 010 = 25          TRANSMIT
=078E040           020 H 17 025 P 24
=078E050           050 T  0 023 T  0
=078E060           002 P 25 004 P 24          EXIT VIA P25 IF STILL INTACT
=078E070           060 H 17 110 M  0          ELSE READ FROM DRUM
=078E080           010 P 25         ,         THEN EXIT
```

```
=079E000+     = 79 024 K   2 010 $   0      VERIFY PRECEDING SPACE
=079E010           052 *   9 020 P   2
=079E020           025 K   3 052 *   1
=079E030      *   1 020 ---- 124 A  14
=079E040           025 A  14 001 E   6
=079E050      *   9 010 ----           ,
```

W	,4340
L	,4360
U	,4400
J	,4530
I	,4560
H	,4600
E	,4640
K	,4700
N	,4760
A	,5000
M	,5220
D	,5540
B	,5554
=	,5600
F	,5720
G	,5740
X	,5760
P	,6000
Q	,6040
Z	,6060
T	,6070
S	,6100
V	,6213
R	,6300

```
ERR--LDR        $                ,4000
ERR--LDR        (                     (
ERR--LDR     *  0 004 * 50 111,1000
ERR--LDR           010,0001
ERR--LDR     * 50 000,1000,012,1777,
ERR--LDR        )                     )
ERR--LDR        (           010,4000)
ERR--LDR        $                ,1000
```

```
INITIATE        $                       M  0    INITIALIZER  (AT REGION M)
INITIATE        (                          (
INITIATE    *   0 020 * 90 070          0       SET ACC=,1234 AND MQ=0
INITIATE          134 $  0 071         40       HALT FOR TIME OF DAY IN MINUTES
INITIATE          032 J  1 060 H  1            CONVERT TO CLOCK COUNTS AND STORE
INITIATE          107    0 050 H  0            SYNCHRONIZE WITH 16-BIT CLOCK
INITIATE    *   1 020 * 91 050 Q  1            SET UP SPARE TANK
INITIATE          020 * 92 050,6454
INITIATE          020 K  4 050,6455
INITIATE          050,6456 050,6457
INITIATE          020 * 93 050 Q  0            SET UP AVAILABLE SPACE
INITIATE          050 * 94 010 $  1
INITIATE    *   2 020 * 94 056 *  3
INITIATE          024 I  1 056 *  4
INITIATE    *   3 024 I  1 050 ----
INITIATE          050 * 94 010 $  1
INITIATE    *   4 020 K  4 050 ----
INITIATE          020 * 94 025 * 95
INITIATE          001 *  2 020 K  4
INITIATE          050    0 050,7776
INITIATE          050,7777 106    0            RESTORE PRINTER
INITIATE          004 * 96 111,6000            SETUP PRIMER COPY OF USER BLOCK
INITIATE          010 X  1                     KICKOFF
INITIATE    * 90                ,1234          IDENTIFIER FOR TIME ENTRY
INITIATE    * 91                ,6454
INITIATE    * 92                ,6456
INITIATE    * 93                ,6460
INITIATE    * 94                 ----          WORKING STORE
INITIATE    * 95                ,7776
INITIATE    * 96 000,0000,013,1777,
INITIATE        )                   )
INITIATE        (          010 M  0)
```

```
D-LDR000+       $                ,7000      LOAD ROUTINES STYLE F TO DRUM
D-LDR005        (                     (
D-LDR010    *   1 050 * 86 020 * 80
D-LDR020          071   21 052 * 88
D-LDR030          020 * 86 124 * 87
D-LDR040          024 * 80 050 * 80
D-LDR050          025 * 83 001 *  2
D-LDR060          120    0 050 * 80
D-LDR070          020 * 88 024 * 82
D-LDR080          050 * 88 014 *  1
D-LDR090    *   2 020 * 80 025 * 81
D-LDR100          056 * 88 020 * 86
D-LDR110          052 *  3 020 * 84
D-LDR120    *   3 050 ---- 052 *  4      STORE HEADER WORD FOR DRUM ROUTINE
D-LDR130          024 * 85 050 * 84
D-LDR140          004 * 88 111 M  0
D-LDR150    *   4 060 ---- 010    1      STORE DRUM CONTROL WORD, EXIT TO F-LD
D-LDR160    *  80                 0      INITIAL NEXT AVAILABLE DRUM ADDRESS
D-LDR170    *  81                 1
D-LDR180    *  82 000,0000,001,0000      BAND COUNTER
D-LDR190    *  83 000,0000,000,2000      1024
D-LDR200    *  84 004 B  0 010 X 15      INITIAL HEADER WORD
D-LDR210    *  85       1
D-LDR220    *  86     ----      ----      PARAMETER WORD (NAME,LENGTH)
D-LDR230    *  87               ,7777     MASK
D-LDR240    *  88 000 ---- 010 ----,      INITIAL DRUM CONTROL WORD
D-LDR250       )                   )
```

```
DRUM-CLR          $                ,4000
DRUM-CLR          (                     (
DRUM-CLR      *   0 004 * 50 111,6000
DRUM-CLR            020 * 51 064 * 50
DRUM-CLR            025 * 52 001 *  0
DRUM-CLR            010,0001
DRUM-CLR      * 50 000,0000,000,1777
DRUM-CLR      * 51 000,0000,001,0000
DRUM-CLR      * 52 000,0000,024,1777,
DRUM-CLR          )                     )
DRUM-CLR          (           010,4000)
```

```
D-CON000+       $                ,7000      CONVERT DRUM ROUTINES FROM E TO F
D-CON005           020 * 82 056 * 84
D-CON010           020    $ 056 * 85
D-CON020     *  1 100    2 004 * 86      PUNCH FROM M0 TO $-1.
D-CON030     *  2 020 * 84 056 *  3
D-CON040     *  3           020 ----
D-CON050           101 * 84 020 * 84
D-CON060           024 * 81 056 * 84
D-CON070           025 * 85 002 * 10
D-CON080           075   79 002 *  2
D-CON090           010 *  1
D-CON100     * 10 020    / 124 * 83
D-CON110           024    $ 025 * 82
D-CON120           100    2 101 * 87      PUNCH CONTROL CARD
D-CON125           010    0               GO BACK TO STYLE E LOADER
D-CON130     * 81                   1
D-CON140     * 82                 M 0
D-CON150     * 83 000,7777,000,0000
D-CON160     * 84          050 ----
D-CON170     * 85          050 ----
D-CON180     * 86 000,0000,000,4000
D-CON190     * 87          010,7000,
```