General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

Produced by the NASA Center for Aerospace Information (CASI)
SUTIL: System Utilities Routines
Revision C

by: Dale Harper, Sigma Corporation

Prepared for:

NASA Johnson Space Center
Houston, Texas 77058

December, 1976
This report was prepared under Contract NAS9-14520, "Development of the Engineering Design Integration System". The study was performed during a period from January 1975 through December 1976. The study was funded by the National Aeronautics and Space Administration, Lyndon B. Johnson Space Center, Houston Texas. The prime contractor for the study was Sigma Corporation and the study was monitored by Mr. Robert Abel of the NASA Engineering Analysis Division.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTRODUCTION</td>
<td>2</td>
</tr>
<tr>
<td>SUBROUTINE DESCRIPTIONS</td>
<td></td>
</tr>
<tr>
<td>Get value from operator</td>
<td></td>
</tr>
<tr>
<td>IGET</td>
<td>3</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>4</td>
</tr>
<tr>
<td>Arguments</td>
<td>4</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>4</td>
</tr>
<tr>
<td>RGET</td>
<td>5</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>6</td>
</tr>
<tr>
<td>Arguments</td>
<td>6</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>6</td>
</tr>
<tr>
<td>GGET</td>
<td>7</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>8</td>
</tr>
<tr>
<td>Arguments</td>
<td>8</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>8</td>
</tr>
<tr>
<td>Get AMOS string from operator</td>
<td>9</td>
</tr>
<tr>
<td>SGET</td>
<td>9</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>9</td>
</tr>
<tr>
<td>Arguments</td>
<td>9</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>9</td>
</tr>
<tr>
<td>Get yes or no answer from operator</td>
<td>10</td>
</tr>
<tr>
<td>YESNO</td>
<td>10</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>13</td>
</tr>
<tr>
<td>Arguments</td>
<td>13</td>
</tr>
<tr>
<td>Monitor listing device</td>
<td>12</td>
</tr>
<tr>
<td>MLIST</td>
<td>12</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>14</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>14</td>
</tr>
<tr>
<td>Time Delay</td>
<td>16</td>
</tr>
<tr>
<td>IDELAY</td>
<td>16</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>16</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>16</td>
</tr>
<tr>
<td>Ring TTY bell</td>
<td>16</td>
</tr>
<tr>
<td>PBELL</td>
<td>16</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>16</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>16</td>
</tr>
<tr>
<td>Free format ANE/TTY input</td>
<td>17</td>
</tr>
<tr>
<td>RDLIN</td>
<td>17</td>
</tr>
<tr>
<td>Calling Sequence</td>
<td>17</td>
</tr>
<tr>
<td>Error Conditions</td>
<td>17</td>
</tr>
<tr>
<td>EXAMPLES</td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td>11</td>
</tr>
<tr>
<td>Test1</td>
<td>15</td>
</tr>
</tbody>
</table>
INTRODUCTION

SUTIL is a package of FORTRAN callable subroutines which allow efficient communication of data between users and programs.

Calls to SUTIL routines which output text to the screen will unconditionally load monitor overlay buffer 1 with the LABEL segment (if not already loaded). I.e. Calls into the SUTIL package must not be concurrent with FORTRAN use of monitor overlay buffer 1.

Text and values output to the screen by SUTIL routines IGET, RGET, OGET, SGET, and YESNO cannot be printed on the PFL, as PFL driver uses monitor overlay buffer 1.

Proper utilization of the SUTIL package will significantly reduce program core requirements and expedite program development.
IGET

IGET is a FORTRAN callable routine designed to display a message for the operator on the AGT screen.

Routine will display the minimum and maximum bounds and current value for an integer variable, accept the current value with a C/R, or a new typein value from the operator.

If an image is being displayed prior to calling IGET a subimage link is formed to enable the image to remain on the screen with message, bounds and current value display, this link is removed on return from subroutine.

If operator types in a new value, it is compared to the upper and lower bound. If within bounds, new value is returned to the calling program. If out of bounds, variable is set to the value of the crossed boundary, the TTY bell is rung, and the following error message is displayed for the operator:

LIMIT EXCEEDED, DEFAULT VALUE ASSUMED
(MIN=XXX, MAX=XXX) CURRENT=XXX

Simple math may be included in typein, however it must be in polish string format.
i.e., no () and solution is from left to right.

Characters which may be included in type string are:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Add</td>
</tr>
<tr>
<td>-</td>
<td>Subtract</td>
</tr>
<tr>
<td>*</td>
<td>Multiply</td>
</tr>
<tr>
<td>/</td>
<td>Divide</td>
</tr>
<tr>
<td>=</td>
<td>Sum</td>
</tr>
<tr>
<td>X</td>
<td>Change sign</td>
</tr>
</tbody>
</table>

Other characters cause the TTY bell to ring and the following error message is displayed for the operator:

ERROR: ILLEGAL CHARACTER

i.e., 100/25*25-50+50 equals 100
100/2.5+60 equals 100
100 + 50X + 50 equals 100 (X = change sign)

Typein followed by an '=' sign will cause the sum to replace typein. This sum may be accepted with a C/R or a new VALUE typed.
CALLING SEQUENCE

CALL IGET (IVALUE, MIN, MAX, "MESSAGE", IFLAG)

ARGUMENTS

1 IVALUE Variable (I/O integer).
2 MIN Minimum bound for IVALUE (input integer).
3 MAX Maximum bound for IVALUE (input integer).
4 MESSAGE ASCII string (input).
   An ASCII character string to be displayed for the user. The MESSAGE is small characters at left side of screen unless ASCII control characters are in MESSAGE string. i.e., 0X00, 0Y00, 0C, 0F, 0E, 0S characters will change MESSAGE size and position.
5 IFLAG (input integer).
   If positive, clear prior MESSAGE(s) before displaying MESSAGE. Else add this MESSAGE to any previous MESSAGE(s).

ERROR CONDITIONS

All errors are properly handled internally. If numbers are out of range as input, TTY bell rings and a message to the user will appear requesting a new input.
RGET

RGET is a FORTRAN callable routine designed to display a message for the operator on the HGT screen.

Routine will display the minimum and maximum bounds and current value for a real variable, accept the current value with a C/R, or a new typein value from the operator.

If an image is being displayed prior to calling RGET a subimage link is formed to enable the image to remain on the screen with message, bounds and current value display, this link is removed on return from subroutine.

If operator types in a new value, it is compared to the upper and lower bound. If within bounds, new value is returned to the calling program. If out of bounds, variable is set to the value of the crossed boundary, the TTY bell is rung, and the following error message is displayed for the operator:

LIMIT EXCEEDED, DEFAULT VALUE ASSUMED
<MIN=XXX,MAX=XXX> CURRENT=XXX

Simple math may be included in typein, however it must be in polish string format, i.e. no () and solution is from left to right.

Characters which may be included in type string are:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Add</td>
</tr>
<tr>
<td>-</td>
<td>Subtract</td>
</tr>
<tr>
<td>*</td>
<td>Multiply</td>
</tr>
<tr>
<td>/</td>
<td>Divide</td>
</tr>
<tr>
<td>=</td>
<td>Sum</td>
</tr>
<tr>
<td>X</td>
<td>Change sign</td>
</tr>
</tbody>
</table>

Other characters cause the TTY bell to ring and the following error message is displayed for the operator:

ERROR: ILLEGAL CHARACTER

i.e. 100/25*25-50+50 equals 100
100/2.5+60 equals 100
100 + 50X + 50 equals 100 (X = change sign)

Typein followed by an '=' sign will cause the sum to replace typein. This sum may be accepted with a C/R or a new VALUE typed.
CALLING SEQUENCE

CALL RGET (VALUE,RMIN, RMAX, "MESSAGE", IFLAG)

ARGUMENTS

1 VALUE Variable (I/O real).
2 RMIN Minimum bound for VALUE (input real).
3 RMAX Maximum bound for VALUE (input real).
4 MESSAGE ASCII string (input).
   An ASCII character string to be displayed for the user.
   The MESSAGE is small characters at left side of screen
   unless ASCII control characters are in MESSAGE string.
   i.e., $X000, Y000, SC, SF, SE characters will change
   MESSAGE size and position.
5 IFLAG (input integer).
   If positive, clear prior MESSAGE(s) before displaying
   MESSAGE.
   Else add this MESSAGE to any previous MESSAGE(s).

ERROR CONDITIONS

All errors are properly handled internally. If numbers are out of range as
input, TTY bell rings and a message to the user will appear requesting a
new input.
OGET is a FORTRAN callable routine designed to display a message for the operator on the CRT screen.

Routine will display the minimum and maximum bounds and current value for an octal variable. Accept the current value with a C/R, or a new typein value from the operator.

If an image is being displayed prior to calling OGET a subimage link is formed to enable the image to remain on the screen with message, bounds and current value display, this link is removed on return from subroutine.

If operator types in a new value, it is compared to the upper and lower bound. If within bounds, new value is returned to the calling program. If out of bounds, variable is set to the value of the crossed boundary, the TTY bell is rung, and the following error message is displayed for the operator:

LIMIT EXCEEDED, DEFAULT VALUE ASSUMED
(MIN=XXX, MAX=XXX) CURRENT=XXX

Simple math may be included in typein, however it must be in polish string format.
I.e. no ( ) and solution is from left to right.

Characters which may be included in type string are:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Add</td>
</tr>
<tr>
<td>-</td>
<td>Subtract</td>
</tr>
<tr>
<td>*</td>
<td>Multiply</td>
</tr>
<tr>
<td>/</td>
<td>Divide</td>
</tr>
<tr>
<td>=</td>
<td>Sum</td>
</tr>
<tr>
<td>!</td>
<td>Change sign</td>
</tr>
</tbody>
</table>

Other characters cause the TTY bell to ring and the following error message is displayed for the operator:

ERROR: ILLEGAL CHARACTER

I.e. 40o + 40o equals 100o

Typein followed by an '=' sign will cause the sum to replace typein. This sum may be accepted with a C/R or a new VALUE typed.
CALLING SEQUENCE

CALL OGET (OVALUE, OMIN, OMAX, "MESSAGE", IFLAG)

ARGUMENTS

1 OVALUE Variable (input, output octal).
2 OMIN Minimum bound for OVALUE (input octal).
3 OMAX Maximum bound for OVALUE (input octal).
4 MESSAGE ASCII string (input).
   An ASCII character string to be displayed for the user.
   The MESSAGE is small characters at left side of screen
   unless ASCII control characters are in MESSAGE string,
   i.e., $X000, $Y000, $C, $F, $E characters will change
   MESSAGE size and position.

5 IFLAG (input integer).
   If positive, clear prior MESSAGE(s) before displaying
   MESSAGE. Else add this MESSAGE to any previous
   MESSAGE(s).

ERROR CONDITIONS

All errors are properly handled internally. If numbers are out of range as
input, TTY bell rings and a message to the user will appear requesting a
new input.
SGET is a FORTRAN callable routine designed to display a message for the operator on the AGT screen, and accept a typed string from the operator.

If an image is being displayed prior to calling SGET a subimage link is formed to enable the image to remain on the screen with message, bounds and current value display, this link is removed on return from subroutine.

CALL SEQUENCE

CALL SGET (ARRAY, MESSAGE, KFLAG)

ARGUMENTS

1 ARRAY
AMOS character array (output).
Minimum Length of array should be determined as follows:
(use a multiple of 5 for maximum characters)
If (maximum characters .LE. 5) length = 1
else
Length = maximum characters / 5 + 1.
If typed string is .GT. 5 first word in array is pointer to characters in form: -number words in address of string.

2 MESSAGE
ASCII string (input).
An ASCII character string to be displayed for the user. The MESSAGE is small characters at left side of screen unless ASCII control characters are in MESSAGE string. i.e., $X0000,Y0000,0C,0F,0E characters will change MESSAGE size and position.

3 KFLAG
Optional (input integer).
Maximum number of characters to accept from operator.
(range = ± 1-132)
If KFLAG argument is neg. prior MESSAGE(s) is not cleared.
If KFLAG argument is omitted, prior MESSAGE(s) is cleared, and 10 set as maximum output characters.

ERROR CONDITIONS

All errors are properly handled internally.
YESNO is a FORTRAN function designed to display a message for the operator on the AGT screen and accept the first character of operator type in string as a YES/NO answer.

1. Yes...
2. "Y", "C/R", or "C/R".
3. No...anything else.

Calling program can branch on the returned function value.

If an image is being displayed prior to calling YESNO a subimage link is formed to enable the image to remain on the screen with message display, this link is removed on return from subroutine.

**CALLING SEQUENCE**

```fortran
IF (YESNO("MESSAGE",NFLAG)) 10,20 or LLL=YESNO("MESSAGE",NFLAG))
```

**ARGUMENTS**

1. **MESSAGE**
   - ASCII string (input),
   - An ASCII character string to be displayed for the user. The MESSAGE is small characters at left side of screen unless ASCII control characters are in MESSAGE string,
   - I.e., @X0000, @Y000, @C, @F, @E characters will change MESSAGE size and position.

2. **NFLAG**
   - (input integer).
   - If positive, clear prior MESSAGE(s) before displaying MESSAGE. Else add this MESSAGE to any previous MESSAGE(s).

**EXAMPLE**

```fortran
LOGICAL YESNO
EXTERNAL YESNO

IF (YESNO("IS EVERYTHING OK",NFLAG)) 10,20
```

If the typed response is "Y", "C/R", or "C/R", the function value will be true, and statement 10 will be executed. Otherwise statement 20 will result from the false function value.
PROGRAM TEST
LOGICAL YESNO
IVAL = RVAL = 0.0
IMIN = RMIN = -9999.0
IMAX = RMAX = 9999.0
IOCTHL = 101B; MINOCTHL = 101B; MAXOCTHL = 2337B
C PUT UP MESSAGE ON SCREEN AND ACCEPT TYPEIN IN RANGE RMIN TO RMAX
10 CALL RGET(IVAL, RMIN, RMAX, "@GET THIS MESSAGE WILL APPEAR AS @
1SMALL CHARACTERS, AT@GETBUFFER LEFT OF SCREEN,
1GETTYPEIN IN A REAL VALUE : $\$, 1)
C ADD ANOTHER MESSAGE, ACCEPT INTEGER TYPE IN
C CALL IGET (IVAL, IMIN, IMAX, "@GET THIS MESSAGE IS ADDED ALSO, @
1THIS TIME WE ARE CALLING THE INTEGER ROUTINE, IF FRACTIONS @GETF@GET
1ARE TYPED IN, ONLY THE INTEGER PORTION IS RETURNED, @GETF@GET
1INTEGER VALUE = $\$, -1)
C ADD ANOTHER MESSAGE
CALL IGET (IVAL, IMIN, IMAX, "@GET SIMPLE MATH MAYBE INCLUDED &
1IN TYPEIN, @GET HOWEVER IT MUST BE IN POLISH STRING FORMAT, @
1IE, ( ) AND SOLUTION IS FROM LEFT TO RIGHT, @
1GET ONLY + - * / X = ARE ALLOWED, @GET
1NY OTHER CHARACTERS CAUSE THE TTY BELL TO RING AND A NEW @
1INPUT REQUESTED, @GET
1IE, 100/25+25-50+50 EQUALS 100@GETF#
10R@GETF#
1 100/2.5+60 IS ALSO 100@GETF#
10R@GETF#
1 100 + 50X + 50 EQUALS 100 (X = CHANGE SIGN)@GET
1GETTYPEIN FOLLOWED BY AN '=' SIGN WILL CAUSE THE SUM TO
1GET REPLACE TYPEIN, THIS SUM MAY BE ACCEPTED WITH A C/R @
1GET FOR A NEW VALUE MAY BE TYPED IN.@
1GET@GETFINTEGER VALUE = $\$, -1)
C PLACE ANSWER IN RVHL AS CURRENT VALUE
RVAL = IVAL
C ASK USER FOR A DATA PACK-VOLUME
CALL OGET (IOCTHL, MINOCTHL, MAXOCTHL, "@GET @GET @GETF@GETF@GET
1TYPE IN PACK-VOLUME FOR DATA, @
1GET@GET@GETF@GETFPV = "", -1)
C DO NOT CLEAR SCREEN, ASK IF OPERATOR WANTS TO REPEAT.
CIF (YESNO ("@GET @GETF@GETWANT TO GO THRU AGAIN?: ", -1)) 10, 20
C C/R OR Y C/R RETURN TO 10, ANYTHING ELSE RETURNS FROM PROGRAM
20 CALL EXIT
END
MLIST is a collection of FORTRAN callable subroutines which allow convenient access to the Adage monitor listing device routines.

CALLING SEQUENCE

Select and initialize listing device.
CALL INLIST (ICODE)
or
CALL INLIST (ICODE, 'NAME', IPRV)

Output an AMOS string with 0 to NCR carriage returns prior to string.
CALL FHMOS ('AA', NCR)

Output date.
CALL IDATE (date)

Output an ASCII string (listing device 1 only (ICODE =1)).
CALL FHSCI I('BB')

Tab to specified column of listing device.
CALL TABTO (ICOL)

Output a carriage return and line feed.
CALL OSTCR

Eject current page.
CALL EJECT

Unselect listing device.
CALL UNSELECT

Output decimal integer.
CALL I2DEC (VALUE, IDIV, ITS)

Output floating point number.
CALL R2DEC (VALUE, ITOTAL, IFRAC)

After initial call to INLIST, the following parameters, may be accessed at: (use FORTRAN equivalence statement). 

70007 = current character position in line.
70010 = current line count.
70011 = maximum lines/page for selected device.
70012 = busy flag (negative if busy).

TN 76-123
ARGUMENTS

<table>
<thead>
<tr>
<th>ARGUMENT</th>
<th>TYPE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICODE</td>
<td>input integer</td>
<td>Desired listing device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = TTY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = SCREEN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = PPL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = ATEXT disk file</td>
</tr>
<tr>
<td>NAME</td>
<td>input AMOS string</td>
<td>If ICODE = 3, the output disk file name is set = NAME</td>
</tr>
<tr>
<td>PVV</td>
<td>input octal</td>
<td>Output pack-volume for disk file.</td>
</tr>
<tr>
<td>AA</td>
<td>input AMOS string</td>
<td>String for output.</td>
</tr>
<tr>
<td>NCR</td>
<td>input integer</td>
<td>Number of carriage returns to output prior to AA.</td>
</tr>
<tr>
<td>DATE</td>
<td>input date</td>
<td>Address of a cell containing a data in AMOS format, usually #DATE or #FILEDATE</td>
</tr>
<tr>
<td>BB</td>
<td>input ASCII string</td>
<td>String for output.</td>
</tr>
<tr>
<td>ICOL</td>
<td>input integer</td>
<td>Desired character column.</td>
</tr>
<tr>
<td>IVALUE</td>
<td>input integer</td>
<td>Integer value to be output.</td>
</tr>
<tr>
<td>DIGITS</td>
<td>integer input</td>
<td>Total digits, range = ±10.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If positive ... right justify.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If negative ... left justify.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Round off will occur if value exceeds number of digits specified.</td>
</tr>
<tr>
<td>ARGUMENT</td>
<td>TYPE</td>
<td>FUNCTION</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VALUE</td>
<td>(input real)</td>
<td>Real value to be output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If VALUE exceeds 536,870,656, 'E' format is used.</td>
</tr>
<tr>
<td>ITOTAL</td>
<td>(integer input)</td>
<td>Total digits, range = ± 10.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If positive, right justify (integer part).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If negative, left justify (integer part).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Round off will occur if value exceeds number of digits specified.</td>
</tr>
<tr>
<td>IFRAC</td>
<td>(input integer)</td>
<td>Fractional digits.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Round off will occur if value exceeds number of digits specified.</td>
</tr>
</tbody>
</table>

**ERROR CONDITIONS**

If ICODE is invalid or incorrect number of arguments default is to TTY. ASCII output is legal to device 1 only.
PROGRAM TEST1
REAL INBUFF(100), OBUFF1(100), OBUFF2(100)
A = B = 10.0
C GET DESIRED LISTING DEVICE FROM USER.
IDEVICE = 1
CALL IGET (IDEVICE, 0, 3, "OUTPUT DEVICE OPTIONS:"
10 = TTY@C@F@
11 = SCREEN@C@F@
12 = PPL@C@F@
13 = DISK FILE@C@F@
1 TYPE IN DESIRED OPTION : ", 1)
C INITIALIZE INBUFF
DO 10 J = 1, 100
INBUFF(J) = A
A = A + B
IF (A .EQ. 250.0) B = -10.0
IF (A .EQ. -250.0) B = 10.0
10 CONTINUE
C CLEAR OBUFF 1 & 2
DO 11 J = 1, 100
OBUFF1(J) = OBUFF2(J) = 0.0
11 CONTINUE
C PLACE SAMPLES IN OBUFF1 AT 2 TIMES SAMPLE RATE OF INBUFF
CALL NTPOLATE (INBUFF(1), OBUFF1(1), 50, 0.5)
C PLACE SAMPLES IN OBUFF2 AT 1/2 SAMPLE RATE OF INBUFF
CALL NTPOLATE (INBUFF(1), OBUFF2(1), 50, 2.0)
C INITIALIZE LISTING DEVICE
CALL IINLIST (IDEVICE)
C OUTPUT HEADER FOR LISTING
CALL FAMOS ('BUFFER LISTING OF INTERPOLATION RESULTS', 0)
CALL FAMOS ('INBUFF', 2)
CALL TABTO (20)
CALL FAMOS ('OBUFF1', 0)
CALL TABTO (40)
CALL FAMOS ('OBUFF2', 0)
CALL OSTCR
C LIST CONTENTS OF BUFFERS
DO 20 J = 1, 50
CALL R2DEC (INBUFF(J), 8, 3)
CALL TABTO (20)
CALL R2DEC (OBUFF1(J), 8, 3)
CALL TABTO (40)
CALL R2DEC (OBUFF2(J), 8, 3)
CALL OSTCR
20 CONTINUE
C EJECT PAGE, AND UNSELECT DEVICE
CALL EJECT
CALL UNSELECT
RETURN
END
IDELAY

IDELAY is a FORTRAN callable Subroutine which causes a time delay.

CALLING SEQUENCE

Call IDELAY (ICOUNT)

ARGUMENTS

ICOUNT (Input integer)  Number of milliseconds of desired delay, range 1 to 4095.

ERROR CONDITIONS

Out of range number defaults to a one second delay.

RBELL

RBELL is a FORTRAN callable routine to ring the TTY bell.

CALLING SEQUENCE

CALL RBELL

ERROR CONDITIONS

None
RDLIN

RDLIN is a FORTRAN callable routine designed to allow free format input from the ANK/TTY.

RDLIN will unconditionally load the LABEL segment into monitor overlay buffer 1 (unless already loaded).

Typein is stored in Global array 'PLIST' as left justified 'PADKR' filled AMOS characters.

Typein is also stored in Global array 'ULIST', right justified, one character per word.

Global variable 'NICH' contains the number of typed characters.

Global variable 'MXXAR' controls the maximum number of typed characters allowed (range 1-132).

Global variable 'PADKR' (normally 4096 space) is used as pad character for left justified strings.
If 'PADKR' is changed by calling program it should be restored to 4096.

Global flag 'NOESP', if negative assumes external display of monitor label buffer (#BUFF).

Cursor appears on screen 2 spaces to the right of last output to label buffer, i.e. a write to unit 25 or SUTIL routine MLIST.

Calling Sequence

CALL RDLIN

Error Conditions

After number of typed characters .EQ. 'MXXAR' no further characters are accepted.
LANGUAGE

ADEPT (Adage machine language)

SUBROUTINE LENGTH

Approximately 1500 words.

EQUIPMENT

AGT-10 through AGT-340

AUTHORSHIP

D. HARPER, Sigma Corp.