IBM 1401

SIMULATOR

supporting

- **a graphical control panel**

- **an assembler (for both Autocoder as well as SPS),** The default is Autocoder, to use an SPS source program, enter "S" in place of "A" in the box below the source file name.

- **a loader**

- **an execution phase with console (MU) support**

- **a core dump feature**

- **a trace feature**

- **many sample test programs**

- **two sundial programs for a horizontal and a vertical dial**

Simon Wheaton-Smith
August 14, 2011
IBM1401autocoderGUIsws.doc

www.illustratingshadows.com
August 14, 2011 Simon Wheaton-Smith

TO GET STARTED WITH THIS IBM 1401 SIMULATOR

1.   Unzip the sim1401c.zip file in any folder you so choose

2.   Obviously run your virus checker, although all files on
        www.illustratingshadows.com
        are virus and spy-ware checked before all uploads

3.   using MY COMPUTER go to the folder you just used

4.   double click   **system1401project.exe**
        or even simpler,   **~0start here.bat**

5.   click the POWER ON button, then the START button

6.   ensure the SOURCE FILE area has   **hdial.acdr** or **vdial.acdr** or
        (upper part of the right side   **hdialsw.acdr** or
         of the CPU panel)   **hdial.sps** or whatever
        *** **You must select "A" or "S" to tell it the file is Autocoder or
        SPS, it does NOT use the file suffix to make that determination** ***

7.   then click START and it assembles the file named in the SOURCE
        FILE area

8.   then click START again and this takes the compiled code and loads
        core storage.

9.   click START which runs the program. Latitude and longitude come from
        the card reader (1401cardrdr.txt)

10.   DUMP for a core dump to be taken.

11.   look at the 1401print.txt file for your output, and 1401coredump.txt for
        the core dump.

NOTE: This system uses POWER ON to establish the GUI display area, and
START to assemble, load, and execute.

NOTE: There are many small test files in the TEST folder and they are all
called TESTnn.TXT and you can move them to the simulator's folder, and
assemble them by placing their name in the **SOURCE FILE** area.

NOTE: This system provides a vertical as well as a horizontal sundial program
with latitude and longitude difference entered by cards in the card reader. The
switches could have been used, but I chose to use cards instead. A dial west of
meridian is assumed, for dials east of the legal meridian, use PM for AM and
vice versa.

NOTE: Excellent 1401 web sites are:-

        http://www.ed-thelen.org/1401Project/SimulatorStatus.html
        http://www.ed-thelen.org/1401Project/1401RestorationPage.html

TO RECOMPILE THIS IBM 1401 SIMULATOR

1.   Install the Lazarus system, see page 15 approx of this booklet
      even for Vista win64, use the 32 bit version
      do not use the version with QT in the file name

         http://www.osalt.com/lazarus   web site for Lazarus

      And locate the download link:

            http://sourceforge.net/project/showfiles.php?group_id=89339

      and locate **the Windows 32 bit** version even if you have a 64 bit machine.

YES	lazarus-0.9.26-fpc-2.2.2-win32.exe	58455268	i386
NO	lazarus-qt-0.9.26-fpc-2.2.2-win32.exe	58420736	i386

      the version for Windows XP was about 58mb:

            lazarus-0.9.26-fpc-2.2.2-win32.exe

      but **do NOT** download:

                  lazarus-**qt**-0.9.26-fpc-2.2.2-win32.exe

      because you will get very frustrated trying to locate: qtcore4.dll

2.   Unzip the sim1401c zip file in any folder you so choose

3.   Obviously run your virus checker, although all files on

            www.illustratingshadows.com

      are virus and spy-ware checked before all uploads

4.   Bring up Lazarus

5.   select PROJECT, and then OPEN PROJECT

6.   locate the folder from step 2

7   double click on the *.lpi file: ibm1401project.lpi

8.   to compile select RUN, if the compiler stops after the build and does not
      bring up the program, select RUN and RESET DEBUGGER

9.   That is all there is to it.
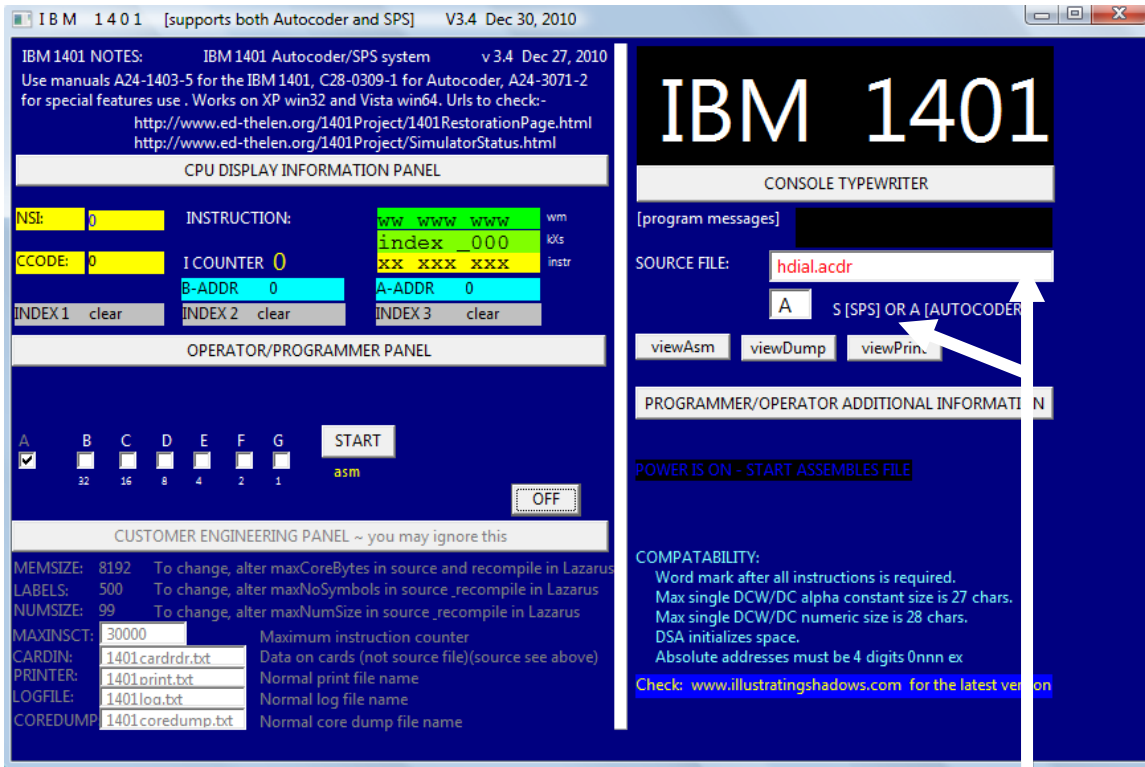
The above is the current simulator panel as of August 2011.

Subsequent pages may show earlier versions of the panel for clarity.

Power ON initializes the system.

Power OFF shuts the system down.

START assembles, then load to memory, then executes the program. It also continues after a program HALT instruction or a console output function (MU opcode). You must select "A" or "S" for Autocoder or SPS.

STOP closes the log and print files, needed for VIEWLOG and VIEWPRINT.

VIEWPRINT, VIEWLOG, VIEWDUMP, and VIEWASM allow those printouts to be immediately looked at without having to locate and double click those file names.

IBM 1401 Simulator with GUI and Autocoder and Simulator integrated

Go to this web site,   http://www.illustratingshadows.com

and there is a section on IBM System 360 as well as IBM 1401 systems. Each section has several simulators, and this is the Lazarus based IBM 1401.

Info

Current status

Controls

What START will do

Textual display and
 console input
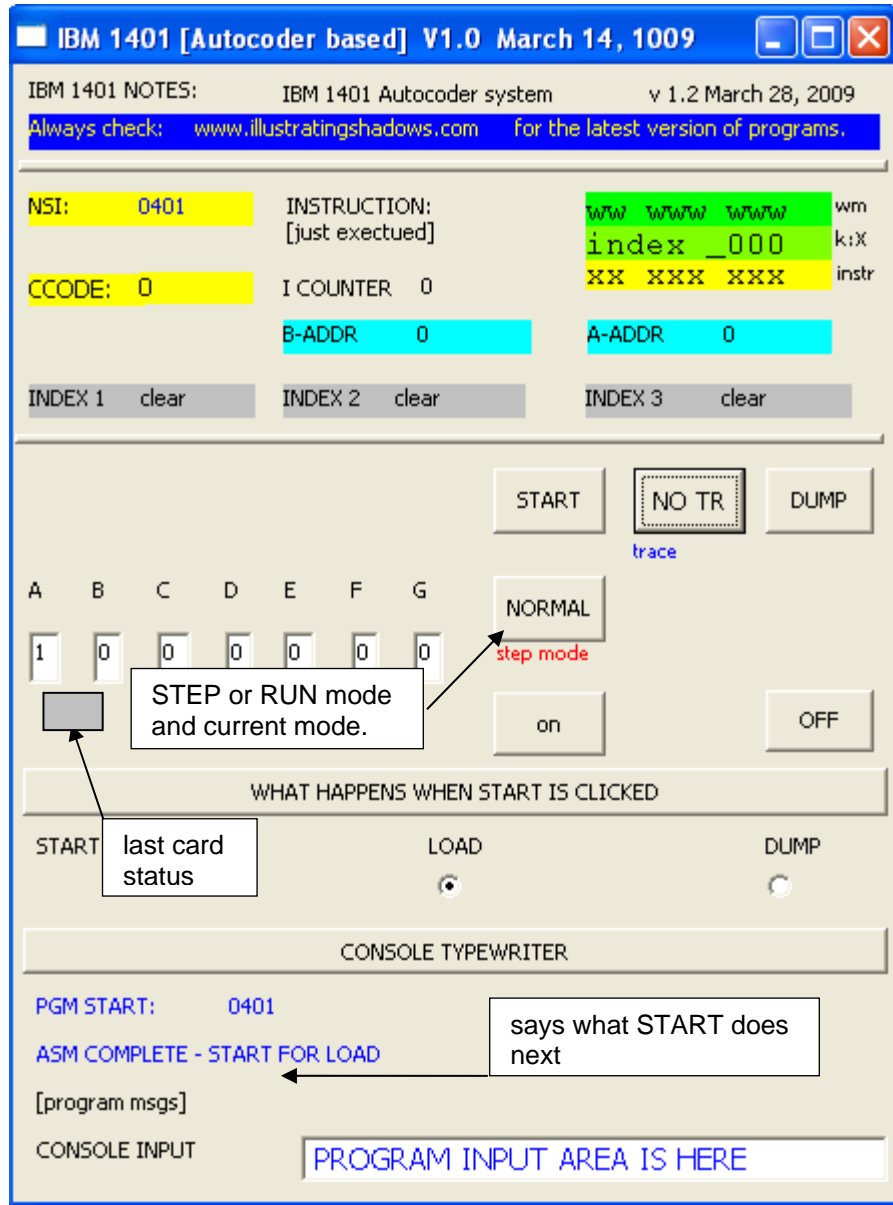
At startup time, POWER ON is visible. When pressed then other elements of the panel are displayed. START is then used to do whatever the prompt says, namely assemble (pass 1 and 2), LOAD, or EXECute the program. The actual simulator panel has more than shown above, but it can be ignored. This is a true install, power on, start and run turnkey system.

At any time the simulator can enter INSTRUCTION STEP mode or resume normal mode, the STEP or RUN button is used for this and the current mode is shown below that button.

OUTPUT LISTINGS

SOURCE CODE:-

```
* TEST
          ORG  501
START     H    PQR
*
*
CONSTNUMW DCW  12345678901234567890123456 7890
CONSTALFW DCW  @12345678901234567890123456 7890@
*
CONSTNUM  DC   12345678901234567890123456 7890
CONSTALF  DC   @12345678901234567890123456 7890@
*
TXCON     DCW  PQR
*
SPACENUM  DS   12345678901234567890123456 7890
SPACEALF  DC   @12345678901234567890123456 7890@
*
*
          ORG  999
PQR       H    PQR
          END  START
```

PASS 1 OUTPUT TO PASS 2:-

```
0000   > ?                                        * TEST
0501   > G                                                  ORG  501
0501 04 . EOJ                                      START    H    EOJ
0505   > ?                                         *
. . . .
. . . .
0635   > ?                                         *
0635   > ?                                         *
0999   > G                                                  ORG  999
0999 04 . EOJ                                      EOJ      H    EOJ
1003   > $ 0501                                             END  START
```

PASS 2 OUTPUT TO LOADER

```
0000   >                                          * TEST
0501   >                                                    ORG  501
0501 04 . [0]999[0]                               START     H    EOJ
0505   >                                          *
0505   >                                          *
0505 20 > [12345678901234567890]                  CONSTNUM  DC   12345678901234567890.....
0525 19 > [1234567890123456789]                   CONSTALF  DC   @1234567890123456789.....
0544   >                                          *
0544 22 > {1234567890123456789012}                CON$TNUM  DCW  1234567890123456789012...
0566 27 > {123456789012345678901234567}           CON$TALF  DCW  @123456789012345678901234567...
0593   >                                          *
0593 03 > {PQR}                                   TXCON     DCW  PQR
0596   >                                          *
0596 20 > [12345678901234567890]                  SPACENUM  DS   12345678901234567890...
0616 19 > [1234567890123456789]                   SPACEALF  DC   @1234567890123456789...
0635   >                                          *
0635   >                                          *
0999   >                                                    ORG  999
0999 04 . [0]999[0]                               EOJ       H    EOJ
1003   >                                                    END  START
```

NOTE: The maximum reliable constant size in this assembler is 27 characters. Long constants can be coded as a DCW first with no label, followed by a DC the last of which has a label. This works because constants are addressed by the low order character whereas instructions are addressed by their high order position.

THE SYMBOL AND ADVISORY TABLE

```
***********************************************************************
* SYMBOL           TYPE          ADDRESS   SIZE     END-ADDR
*
* START            OP-CODE        0501       4       504
* CONSTNUM         NO WM CONSTANT 0505      20       524
* CONSTALF         NO WM CONSTANT 0525      19       543
* CON$TNUM         CONSTANT       0544      22       565
* CON$TALF         CONSTANT       0566      27       592
* TXCON            CONSTANT       0593       3       595
* SPACENUM         STORAGE        0596      20       615
* SPACEALF         NO WM CONSTANT 0616      19       634
* EOJ              OP-CODE        0999       4      1002
*
* INSTRUCTIONS ADDRESSED BY HI ORDER, DATA ADDRESSED BY LOW ORDER.
*
* CODING ABSOLUTE ADDRESSES:  0NNN AND NOT NNN (EG H, CS, ETC)
*
* DSA SPACE HOLDER ACTS LIKE DC, SO CORE STORE IS INITIALIZED
*
* DCW MAX GUARANTEED LENGTH OF USER DATA IS 27 CHARACTERS
*
* ENSURE OPCODES HAVE A WORD MARK AFTER THEM.
*
***********************************************************************
```

THE CORE DUMP

```
* CORE DUMP BEGIN *

IFETCHADDR: 1407
A REGISTER: 1903
B REGISTER: 1463

* CORE DUMP BEGIN – CORE STORAGE *

000000  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
                                                                                      ......  ..   .W .

        02                                                                                 012 000 007
000100  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
        ...................................................................................................


000200  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100

                            DOWNLOAD MICRO-SHADOWS.PDF FROM THE WEBSITE FOR TIPS AND FAQS
000300  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
                        .................................................................................


000400  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
        W...WWW......WW...W.....WW...WWW......WW.....W......W....WW...WWW....W...W.....W...W.
              1           1           1          1      2  1 1  1       1         1  2   2   2 1       1
        /332/2M5272902/290M5922902/29021M0024141M002018Y414430M600250M4302602/320/2L430328B276M338640/280M64
000500  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
        ...W....W....WW....W...W....WW...W....W......W....W......W....W.....W...W.....W...WW...
            1      1          2       2          2  2   2  2   2  2   2      1 2   2
        0260M636250M4352572/2802M01826 0M0282502/2802/280?018055A055055A055055M053249Y414055M0552602/2802/290
000600  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
        W......WW...WWWW...W.....WW...W.....W......WW...WW....W....W...W.....W...W.....W...W....W....W...
               1           1           1      1         2   1 1   1      1        1 1   1       1      1 1
        M5922902/290222/280M6922702/280M967253M9672632/2802M063290?969975,097M975099C977975B917UM975230C9699
000700  1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95..100
        .W...W.....W....W....W....W....W....W.....W......W....W....W...W.....W...W.....W...W....W...
              1 1   1 2   1 2   2   2 1         2    2 1         1   2 1  1 1        1        1   1 2
        75B900S?975989?983245?989254B080M264999,246B7630200A018999B778B9009750S018999Y647999,237M999239M9990
```

THE LOG

```
POWER ON:
SOURCE CODE FILE OPENED: hdial.acdr
ASM PASS 1: STARTED
ASM PASS 1: ENDED
ASM PASS 2: STARTED
ASM PASS 2: ENDED
LOAD PHASE: STARTED
LOAD PHASE: ENDED
EXEC PHASE: STARTED
EXEC PHASE: IN PROGRESS
EXECUTING: 00401 I COUNTER: 000001 INSTRUCTION=/332/2M527 A REG: 000000 B REG: 000000
                CCODE: 0          -----                1         X1:clear X2:clear
X3:clear
                                             W...WWW...

EXECUTING: 00405 I COUNTER: 000002 INSTRUCTION=/2M5272902 A REG: 000299 B REG: 000000
                CCODE: 0          -----                1         X1:0 X2:0 X3:0
                                             WWW......W

EXECUTING: 00406 I COUNTER: 000003 INSTRUCTION=2M5272902/ A REG: 000199 B REG: 000000
                CCODE: 0          -----                1         X1:0 X2:0 X3:0
                                             WW......WW

. . .
. . .

EXECUTING: 01398 I COUNTER: 009662 INSTRUCTION=2mTO 445W. A REG: 001903 B REG: 000228
                CCODE: 2          -----                1         X1:12 X2:0 X3:7
                                             WW.......W

EXECUTING: 01399 I COUNTER: 009663 INSTRUCTION=mTO 445W.4 A REG: 001903 B REG: 000333
                CCODE: 2          -----                1         X1:12 X2:0 X3:7
                                             W.......W.

    MESG:
EXEC PHASE.
POWER OFF:
```

THE CARD FILE FOR HDIAL AND VDIAL

```
33          *
02
WEST
END         *
```

Card 1 is two digits for latitude, card 2 is two digits for longitude, card 3 is East or West, while card four is not used for now.

INDEX REGISTER AND ADDRESS 1000s IN THE ASSEMBLER PASS 2 LISTING

INSTR LEN   OP   000   ADDRESS   INDEX
07   M   [3]   830   [1]

opcode is M

address is 3830 plus index register 1

seven char opcode
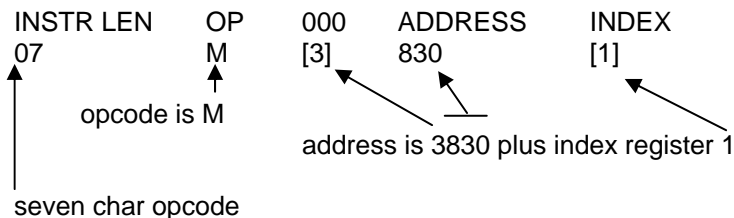
INCOMPATIBILITIES AND DIFFERENCES AND LISTINGS -   DSA and DCA and DS

The IBM 1401 specifications for Autocoder has DCW LABEL as generating an address constant however, this was more of an equate with assembly insertion. This assembler uses DCA LABEL which is not a true IBM 1401 feature, and it is an address constant in the sense that the term was used with the IBM 360 and later systems. Incompatibilities thus are that (1) DCA is specific to this simulator, and that (2) DSA is not implemented except as the same as DS. The coding below should clear up any possible misunderstanding.

ASSUME THE FOLLOWING SOURCE:-

```
        * TEST10.ACDR  CONSTANT TYPES
                ORG  501
        START     H    EOJ
        *
        *
        CONSTNUM  DC   12345678901234567890123456 7890
        CONSTALF  DC   @12345678901234567890123456 7890@
        *
        CON$TNUM  DCW  12345678901234567890123456 7890
        CON$TALF  DCW  @12345678901234567890123456 7890@
        *
        ADCON     DCA  EOJ
        *
        SPACENUM  DS   12345678901234567890123456 7890
        SPACEALF  DC   @12345678901234567890123456 7890@
        *
        *
                ORG  999
        EOJ       H    EOJ
                  END  START
```

PASS 1 OUTPUT FOR PASS 2:-

```
0593   > ?                                  *
0593 03 > A               COL:66            ADCON      DCA   EOJ
0596   > ?                                  *
```

PASS 2 OUTPUT FOR LOADER:-

```
0593   >                                    *
0593 03 > {999}                             ADCON    DCA   EOJ
0596   >                                    *
```

CORE DUMP:

```
* CORE DUMP BEGIN *

IFETCHADDR: 999
A REGISTER: 0
B REGISTER: 0

* CORE DUMP BEGIN - CORE STORAGE *

        1...5...10...15...20///..70...75...80...85...90...95..100
000500  W... .............///............................W.. ....
        . ...................///........................ .. ....
        .9991234567890123456///234567890123456789012345679 9912345

000600  .............. ....///..................................
        .............. ....///..................................
        67890123456789012345///

* CORE DUMP END *
```

CONSOLE SUPPORT FOR THE IBM 1407

The "MU" opcode is supported as "m" as opposed to "M". Otherwise this is compatible with normal Autocoder. The simulator does an internal HALT after console output, and this allows the user to enter data into the type in area, and to see each message if there are several messages.

TEST11.ACDR demonstrates console usage.

```
* TEST11.ACDR  TEST CONSOLE FUNCTION.
          ORG  501
*
START     MU   %TO,AREA,W
*              THE REAL 1401 HAS HI ORDER ADDRESSED
*              AND LOW ORDER HAS A WM, OPPOSITE TO
*              ALL OTHER DATA ADDRESSING WHICH IS TO
*              ADDRESS LOW ORDER AND STOP ON HI ORDER
*              WM. THIS IS BECAUSE OF SERIAL DATA XFR
*              TO THE TYPEWRITER.
*              THE SIMULATOR HALTS ON A TYPE SO THEY CAN TYPE
          MU   %TO,INAR,R
          H    EOJ
*
          ORG  601
*
*              AREA IS THE HIGH ORDER ADDRESS, AND WE
*              HAVE A WORD MARK ON THE LAST CHAR
AREA      DC   @T@
          DC   @HIS A TEST MESSAG@
          DCW  @E@
*
          ORG  701
INAR      DC   @*@
          DC   @****************@
          DCW  @*@
*
*
*
          ORG  999
*
EOJ       H    EOJ
          END  START
```

The largest DC or DCW constant is 27 characters or 28 if numeric. However, there can be multiple DCs in a sequence. The simulator does its own add and subtract thus there is no actual limit. A LAZARUS compile time limit is set at 99 characters, however this can be any number at all.

Autocoder is the primary language for which this simulator was designed.

The original source for the horizontal dial is in a file called:-

hdial.acdr	reads cards for lat, long.diff, and EAST or WEST
hdialsw.acdr	reads sense switches for lat then for long.diff and EAST or WEST
vdial.acdr	reads cards for lat, long.diff, and EAST or WEST

and whereas SPS was fixed format, Autocoder was free format, and coded on worksheets such as shown below. This simulator can convert SPS to Autocoder, see next page.

The code is compiled into the executable program. This is a two phase assembler. The version implemented with this particular simulator has the appearance of being one pass, in fact it is one large program that is part of the IDE.

The IBM 1401 was a decimal machine, with 6 bits and with two left over bits for things like "word marks" which are how the computer knows (a) when an instructions operands have ended, and (b) how the computer knows when to stop moving or adding data.

IBM 1401   SPS

        ZA   NUM32   LATUD   zero LATUD and then add NUM32

IBM 1401   Autocoder

        ZA   NUM32,LATUD   zero LATUD and then add NUM32

This simulator uses three arrays for each core storage character. One is for word marks, the second is for address-thousands and for indexing, the last is the actual character.

This simulator does not translate ASCII to BCD, and little is lost by that omission.

SPS preceded Autocoder, and this simulator can convert SPS to Autocoder.

hDial.sps

and whereas Autocoder was free format, SPS was fixed format, and coded on worksheets such as shown below. *** **NOTE** *** to use SPS, you must enter "S" in place of "A" in the box below the source file name.

The program begins by looking something like the following. This code states that this is a HORIZONTAL DIAL and after looking at the sense switches A through G, it has determined the desired latitude:-

```
00001        ORG0500                    START AT 500
00002 START  CS 0332                    CLEAR THE
00003        CS                         PRINT AREA
      *
      *      SAY THIS IS A HORIZONTAL DIAL
      *
00004        MCWMESSG      0258         MOVE TEXT TO PA
00005        W                          PRINT IT
00006        NOP                        GET LATITUDE
00007 GETLT  CS 0332                    CLEAR THE
00008        CS                         PRINT AREA
      *
00009 *      DETERMINE LATITUDE FROM SWITCHES
      *
00010        ZA LATZZ      LAT          ZERO DESIRED LATITUDE
00011        B  IS64                    ASWITCH IS 64
00012        B  NO64                     SKIP ADDING 64
00013 IS64   A  LAT64      LAT           ADD 64 TO LATITUDE
00014 NO64   B  IS32                    BSWITCH IS 32
00015        B  NO32                     DONT ADD 32
00016 IS32   A  LAT32      LAT           ADD 32 TO LATITUDE
00017 NO32   B  IS16                    CSWITCH IS 16
00018        B  NO16                     DONT ADD 16
00019 IS16   A  LAT16      LAT           ADD 16 TO LATITUDE
00020 NO16   B  IS08                    DSWITCH IS 08
00021        B  NO08                     DONT ADD 08
      ...
      ...
      ...
```

Clear Storage

Instruction Format.

Mnemonic	Op Code	A-address
CS	/	xxx

Function. As many as 100 positions of core storage can be cleared of data and word marks when this instruction is executed. Clearing starts at the A-address and continues leftward to the nearest hundreds position. The cleared area is set to blanks.

Word Marks. Word marks are not required to stop the operation.

CS	0320	clear storage to 300
CS		no operand means continue below, ie 299 to 200

In the days of the IBM 1401, core storage (memory) was limited. So one feature that saved program size was the internal registers, if operands were omitted then the registers were used. This was called chaining. Chaining is used above on a second CS with no operands.

chaining discussed ⟶

By taking advantage of the fact that the A- and B-address registers contain the necessary information to perform the next instruction, this same sequence of operations can be executed as follows:

A	700	850
A		
A		
A		

Connecting instructions together in this manner is called *chaining*. The first add instruction contains both

As an aside note, the concept of chaining made automated conversion of IBM 1401 programs at the source code level to the new (then) IBM 360, very difficult. This was because automation of the source code conversion had to determine the context of operands, as well as how the machine worked, and build new operands into the new IBM 360 source code. The author wrote such conversion programs, and also wrote an IBM 30 simulator for the IBM 1401.

And, additionally, while the IBM card reader always put cards at 001 through 080 of memory, punched from 101 through 180, and printed from 201 through 332, the IBM 360 did not have that concept. Automated conversion programs that converted Autocoder or SPS could manage that.

And, additionally, document R29-0044 states on page 8 that only BRANCH IF NOT EQUAL was standard, HI, LO, EQ were special features. In those days the systems were truly very basic.

This simulator supports COMPARE BH/BL. Thus the ATAN routine is shorter than the SPS version. The simulator using SPS did not always support BH/BL after a compare.

Operand address arithmetic was made possible by the index registers. This feature would be used by the latitude look up operations. NOTE: never compare addresses on the 1401, high order bits were used and comparing produces erroneous results.

Indexing

The indexing portion of the advanced-programming feature provides three 3-position index locations (registers) that can be used to modify addresses automatically. These three index registers are part of core storage and can be used as normal storage positions when not being used as index-register locations. The assigned core-storage addresses and index register numbers are:

Index-Register No.	Core-Storage Positions
1	087 - 089
2	092 - 094
3	097 - 099

The index factor can be placed in the index register by normal programming (add or move operations, for example) and the factor can be changed (add operations normally). In these instances, a word mark should be initially set in the high-order position of the index register before inserting or changing the index factor.

Both the A-address and/or the B-address can be modified by the index factor in any one of the three index registers; however, only core-storage addresses can be modified.

C   FROM&X1,TO

the &X1 causes Autocoder to modify the address by setting index register 1.

Addressing op-codes (high order) and data (low order):

Instruction addressed by high-order position

STORAGE ADDRESS	400	401	402	403	404	405	406	407 (NSI)
INSTRUCTION	A	5	4	2	5	6	0	WM Op code

The word mark associated with the next sequential instruction (NSI) stops the reading of this instruction.

A-address

STORAGE ADDRESS	536	537	538	539	540	541	542	543
DATA	0	0	2	5	3	4	7	8

A-field

Word mark identifies high-order position of A-field.

B-address

STORAGE ADDRESS	553	554	555	556	557	558	559	560	561
DATA	0	4	6	0	1	2	3	1	4

B-field

Word mark identifies high-order position of B-field.

High order position of both data and op-codes has the word mark (WM)
Addressing an op-code or data however was different...
        For data: Low order (right) position is what is addressed
        For instructions: High order (left) position is what is addressed

Numeric data has sign in low order position zone bits

Branching:
BWZ as opposed to B

Branch if Word Mark and/or Zone

Instruction Format.

Mnemonic	Op Code	I-address	B-address	d-character
BWZ	<u>V</u>	xxx	xxx	x

Function. The single character at the B-address is examined for a particular bit configuration, as specified by the d-character. If the bit configuration is present as specified, the program branches to the I-address for the next instruction:

d-character	Condition
1	Word mark
2	No zone (No-A, No-B-bit)
B	12-zone (AB-bits)
K	11-zone (B, No-A-bit)
S	Zero-zone (A, No-B-bit)
3	Either a word mark, or no zone
C	Either a word mark, or 12-zone
L	Either a word mark, or 11-zone
T	Either a word mark, or zero-zone

Branching:
B as opposed to BWZ

| Logic	Mnemonic	Description		
	B	Branch Unconditional	B	
	BAV	Branch on Arithmetic Overflow	B	Z
†BBE		Branch if Bit Equal	W	d
	BC9	Branch on Carriage Channel 9	B	9
	BCV	Branch on Carriage Over-flow (12)	B	@
	BE	Branch on Equal Compare (B=A)	B	S
	BEF	Branch on End of File or End of Reel	B	K
	BER	Branch on Tape Transmission Error	B	L
	BH	Branch on High Compare (B > A)	B	U
†BIN		Branch on Indicator	B	d
	BL	Branch on Low Compare (B < A)	B	T
	BLC	Branch on Last Card (Sense Switch A)	B	A
	BM	Branch on Minus (11-zone)	V	K
	BPCB	Branch Printer Carriage Busy	B	R
	BPB	Branch Printer Busy	B	P
	BU	Branch on Unequal Compare (B ≠ A)	B	/
	BW	Branch on Word Mark	V	1
†BWZ		Branch on Word Mark or Zone	V	d
†BCE		Branch if Character Equal	B	d
†BSS		Branch if Sense Switch On	B	A-G
	C	Compare	C	

NOTE: Whereas the IBM 360/370 use B to A operands in its design, the IBM 1401 used A to B operands with the exception of compare BH and BL. That is the way they designed it, don't blame me. Also, the original IBM 1401 had HI/LO as a special feature. See the SPS HDIAL program for ATAN showing how HI/LO compares were not used and only BE/BU were used. The BH/BL was a special feature on the IBM 1401. A pain in the rear.

NOTE: Also, BIN and BSS convert to the same opcode, so you may use either.

Other useful information:-

Memory is 7 bit BCD characters:-

6	5	4	3	2	1	0
word mark	B bit	A bit	8	4	2	1
	← zone →		←	digit		→
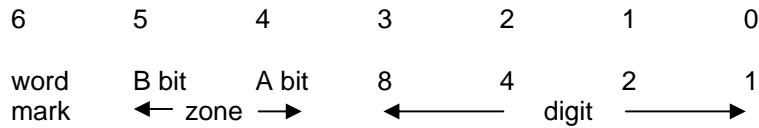
In BCD, signs are encoded in the zone bits, with 00, 01, and 11 being positive, and 10 being negative.

Memory allocation:-

```
   0....5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80...85...90...95...99
0  1.......................................................................80   card in area
                                                                               87.89 index 1
                                                                               92.94 index 2
                                                                               97.99 index 3
100 101.....................................................................180  card punch area

201 201....................................................................//
        ...............................232                                      print area
```

For assembly, in SPS all absolute addresses are 4 digits, even though assembled into 3 characters, if you coded:

    CS   280

you would get an error, you must code:

    CS   0280

Also, high order bits in the first and third characters of an address manage addresses above 0999, see the table to the right.

Also, the high order bits of the second character of an address field identify the index register, if one is being used. There are three of them, at locations 0087-0089, 0092-0092, and 0097-0099.

This explains why you should not compare addresses for high or low. Those high order bits confuse the results.

CODED ADDRESSES IN STORAGE

ACTUAL ADDRESSES		3-CHARACTER ADDRESSES
000 to 999	No zone bits	000 to 999
1000 to 1099		‡00 to ‡99
1100 to 1199		/00 to /99
1200 to 1299		S00 to S99
1300 to 1399		T00 to T99
1400 to 1499	A-bit,	U00 to U99
1500 to 1599	using 0-zone	V00 to V99
1600 to 1699		W00 to W99
1700 to 1799		X00 to X99
1800 to 1899		Y00 to Y99
1900 to 1999		Z00 to Z99
2000 to 2099		I00 to I99
2100 to 2199		J00 to J99
2200 to 2299		K00 to K99
2300 to 2399		L00 to L99
2400 to 2499	B-bit,	M00 to M99
2500 to 2599	using 11-zone	N00 to N99
2600 to 2699		*O00 to O99
2700 to 2799		P00 to P99
2800 to 2899		Q00 to Q99
2900 to 2999		R00 to R99
3000 to 3099		?00 to ?99
3100 to 3199		A00 to A99
3200 to 3299		B00 to B99
3300 to 3399		C00 to C99
3400 to 3499	A-B-bit,	D00 to D99
3500 to 3599	using 12-zone	E00 to E99
3600 to 3699		F00 to F99
3700 to 3799		G00 to G99
3800 to 3899		H00 to H99
3900 to 3999		I00 to I99

* Letter O followed by Zero Zero

NOTE: The 6 bits shown above are implemented with three arrays, one for the word mark, another for address thousands as well as indexing, the third for actual data. Nothing is lost by this simplification.

The IBM 1401 op-codes are shown to the right.

Multiply and Divide were special features. The author had to write a 1401 simulator for the IBM 360 because non of the (very few) available simulators had the sterling feature, and at that time the UK currency was still pounds, shillings, and pence.

Compare and testing for results had to be double checked, and even HI, LO, EQ were special features.

There were two assembler languages, one was SPS, which this program uses, the other was the improved Autocoder, which the author used when programming the IBM 1401.

AREA DEFINITION		
Mnemonic Operation Code	Description	
DCW	Define Constant With Word Mark	
DC	Define Constant (No Word Mark)	
DS	Define Symbol	
DSA	Define Symbol Address	

INSTRUCTIONS			
Type	Mnemonic Operation Code	Description	Machine Language Equivalent
Arithmetic	A	Add	A
	S	Subtract	S
	*M	Multiply	@
	*D	Divide	%
	ZA	Zero and Add	? (Prints as &)
	ZS	Zero and Subtract	! (Prints as −)
Data Control	MCW	Move Characters to A or B Word Mark	M
	*MCM	Move Characters to Record or Group Mark	P
	MCS	Move Characters and Suppress Zeros	Z
	MN	Move Numeric	D
	MZ	Move Zone	Y
	MCE	Move Characters and Edit	E
	LCA	Load Characters to A Word Mark	L
	SW	Set Word Mark	,
	CW	Clear Word Mark	□
	CS	Clear Storage	/
	*MIZ	Move and Insert Zeros (for reading 7070 Compressed Tape)	X
	*MA	Modify Address	#
	*SAR	Store A Address Register	Q
	*SBR	Store B Address Register	H
Logic Control	B	Branch	B
	BWZ	Branch if Word Mark and/or Zone	V
	C	Compare	C
	NOP	No Operation	N
	H	Halt	.
	*BBE	Branch if Bit Equal	W
System Control	R	Read a Card	1
	W	Write a Line	2
	WR	Write and Read	3
	P	Punch a Line	4
	RP	Read and Punch	5
	WP	Write and Punch	6
	WRP	Write, Read and Punch	7
	*SRF	Start Read Feed	8
	*SPF	Start Punch Feed	9
	SS	Select Stacker	K
	CC	Control Carriage	F
	CU	Control Unit	U
	MU	Move Unit	M
	LU	Load Unit	L

PROCESSOR CONTROL OPERATIONS		
Mnemonic Operation Code	Description	
CTL	Control	
ORG	Origin	
END	End	
EX	Execute	

*Pertains to an optional feature.

The IBM 1401 didn't really have "words" as commonly used, things were variable length and delineated by word marks, and they could be set and cleared by SW and CW instructions. Normal MOVE instructions did not copy a word mark, that required a LOAD instruction.

Below is a data field showing the word mark:-

```
400  401  402  403  404  405  406  407
wm
char char char char char char char char
                                   ↑
this is the field's address: ──────┘
i.e.  A-address, and B-address of op-codes
```

Op-codes have a word mark with the one character op-code, and there must be another word mark to mark the end of the instruction. This is usually implied since on instruction follows another. However, the last instruction must have a word mark after it. This is because instructions are variable length, unlike the IBM 360 which used 2, 4, or 6 byte (aligned on half word boundaries) op-codes.

This variable length op-code is what provided the ability for op-codes to be chained, since the A and B registers kept track of where things were with data operands, and if omitted, the A or B or both registers were used in lieu of actual operands.

```
CS      0320        / 3 2 0
                    wm
CS                  /
                    wm
xxx                 opcode
                    wm
```

The first instruction clears storage starting at 0320, note that four digits are used for SPS, however the actual compiled address uses only three positions.

CS clears down to the next lowest hundreds, and like all other instructions the A-register moved down. So the next instruction "/" has no operands, thus the A-register is used, clearing the next 100 positions. This is called chaining.

Some op-codes could be added together! For example read a card was 1, print a line was 2, and punch a card was 4. You could add them so that 7 for example would read a card, print a line and punch a card. In days when memory was very valuable, every trick was used to maximize its usage.

The core dump extracts below show the systems dump process and its easy to use features:-
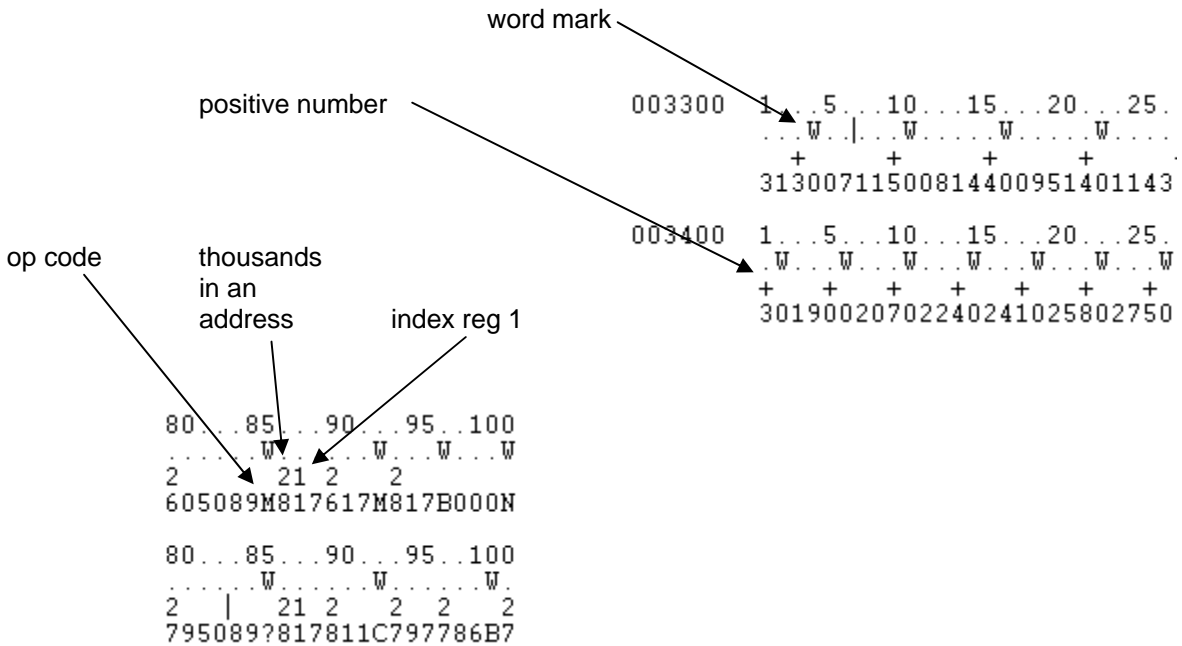
Collating sequence for the IBM 1401

PRINTS AS	DEFINED CHARACTER	CARD CODE	BCD CODE
	BLANK		C
.	.	12-3-8	B A 8 2 1
□	□	12-4-8	C B A 8 4
	(Left Parenthesis (Special Character)	12-5-8	B A 8 4 1
	< Less Than (Special Character)	12-6-8	B A 8 4 2
	⧻ Group Mark (Note 1)	12-7-8	C B A 8 4 2 1
&	&	12	C B A
$	$	11-3-8	C B 8 2 1
*	*	11-4-8	B 8 4
	) Right Parenthesis (Special Char.)	11-5-8	C B 8 4 1
	; Semicolon (Special Character)	11-6-8	C B 8 4 2
	Δ Delta (Mode Change)	11-7-8	B 8 4 2 1
—	—	11	B
/	/	0-1	C A 1
,	,	0-3-8	C A 8 2 1
%	%	0-4-8	A 8 4
	= Word Separator	0-5-8	C A 8 4 1
	' Apostrophe (Special Character)	0-6-8	C A 8 4 2
	" Tape Segment Mark	0-7-8	A 8 4 2 1
‡	¢ Cent (Special Character Note 2)		A
#	#	3-8	8 2 1
@	@	4-8	C 8 4
	: Colon (Special Character)	5-8	8 4 1
	> Greater Than (Special Character)	6-8	8 4 2
	√ Tape Mark	7-8	C 8 4 2 1
&	? (Plus Zero)	12-0	C B A 8 2
A	A	12-1	B A 1
B	B	12-2	B A 2
C	C	12-3	C B A 2 1
D	D	12-4	B A 4
E	E	12-5	C B A 4 1
F	F	12-6	C B A 4 2

PRINTS AS	DEFINED CHARACTER	CARD CODE	BCD CODE
G	G	12-7	B A 4 2 1
H	H	12-8	B A 8
I	I	12-9	C B A 8 1
—	! (Minus Zero)	11-0	B 8 2
J	J	11-1	C B 1
K	K	11-2	C B 2
L	L	11-3	B 2 1
M	M	11-4	C B 4
N	N	11-5	B 4 1
O	O	11-6	B 4 2
P	P	11-7	C B 4 2 1
Q	Q	11-8	C B 8
R	R	11-9	B 8 1
‡	‡ Record Mark	0-2-8	A 8 2
S	S	0-2	C A 2
T	T	0-3	A 2 1
U	U	0-4	C A 4
V	V	0-5	A 4 1
W	W	0-6	A 4 2
X	X	0-7	C A 4 2 1
Y	Y	0-8	C A 8
Z	Z	0-9	A 8 1
0	0	0	C 8 2
1	1	1	1
2	2	2	2
3	3	3	C 2 1
4	4	4	4
5	5	5	C 4 1
6	6	6	C 4 2
7	7	7	4 2 1
8	8	8	8
9	9	9	C 8 1

HORIZONTAL SUNDIAL – **hdial.acdr**

The output is in a file called:-   1401print.txt

```
HORIZONTAL SUNDIAL PROGRAM ON THE IBM 1401 - - -  S WHEATON-SMITH
==================================================================
                  LATITUDE        33
                   SIN LAT       .544
                  LNG DIFF         2
            MINUTES DIFF          08
==================================================================

   HR    HRANGL    TAN.-------    TAN.HLA    HLA
                        *1K          *1K

   00                                                        PM HOURS
   01     013      000230        000125      08
   02     028      000531        000288      17
   03     043      000932        000507      27
   04     058      001600        000870      42
   05     073      003270        001778      61
   06     088      028636        015577      87


                                                            AM HOURS
   01     017      000305        000165      10
   02     032      000624        000339      19
   03     047      001072        000583      31
   04     062      001880        001022      46
   05     077      004331        002356      68
   06     092

   00     002      000034        000018      02         --NOON--


   HOUR LINE ANGLES ARE CLOSE
   HNOON ANGLE IS PRETTY GOOD

   FREE SUNDIAL NOTES AT    WWW.ILLUSTRATINGSHADOWS.COM

   THIS DIAL IS WEST OF MERIDIAN. IF EAST OF MERIDIAN THEN SWITCH AM FOR PM
   CHECK     WWW.ILLUSTRATINGSHADOWS.COM FOR THE LATEST PROGRAMS
   DOWNLOAD    MICRO-SHADOWS.PDF   FROM THE WEBSITE FOR TIPS AND FAQS
```

Each latitude entry covered hours from noon to 6 from noon. The latitude is read from the first of three cards in the 1401, longitude difference in card two, and east/west of meridian in card three.

1401cardrdr.txt
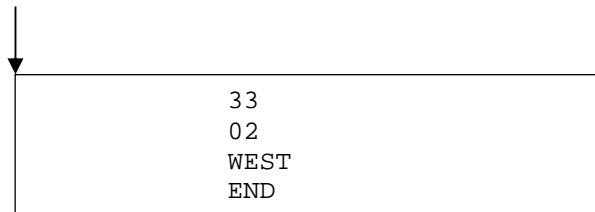
```
33
02
WEST
END
```

SAMPLE CODE FOR THE HDIAL PROGRAM (CHECK FILE FOR CURRENT VERSION)

```
              JOB   HORIZONTAL DIAL
              CTL   6611
      *
      *     HORIZONTAL SUNDIAL V4     WRITTEN IN IBM 1401 AUTOCODER
      *
      *     APRIL 11, 2009          SIMON WHEATON-SMITH
      *
      *     WWW.ILLUSTRATINGSHADOWS.COM
      *     DOCUMENTATION FILE              IBM1401AUTOCODERGUISWS.pdf
      *     ON WEB PAGE                     IBM 1401
      *
      *     DISPLAYS THE HOUR LINE ANGLES
      *
      *     LATITUDE     IS READ AS CARD 1     NN     ( EG 33 )
      *     LONGITUDE    IS READ AS CARD 2     NN     ( EG 02 )
      *     EAST OR WEST IS READ AS CARD 3     CCCC   ( EG WEST )
      *
      *     DOES NOT USE MULTIPLY NOR DIVIDE BUT USES SUBROUTINE HEREIN
      *     DOES USE INDEX REGISTER 1 SYNTAX IS LABEL&X1 NOT LABEL&1
      *     USES SUBROUTINES SIN TAN MULT AND ATAN
      *     ENTIRE MATH PACKAGE IS HERE FOR SIN COS TAN COT ATAN MULT
      *
      *
      *
      * HORIZONTAL SUNDIAL PROGRAM ON THE IBM 1401 - - -  S WHEATON-SMITH
      * ================================================================
      *                 LATITUDE       33
      *                 SIN LAT      .544
      *                 LNG DIFF       2         WEST OF MERIDIAN
      *           MINUTES DIFF        08
      * ================================================================
      *
      *    HR    HRANGL   TAN.-------   TAN.HLA   HLA
      *                      *1K         *1K
      *
      *    00                                                PM HOURS
      *    01     013       000230      000125     08
      *    02     028       000531      000288     17
      *    03     043       000932      000507     27
      *    04     058       001600      000870     42
      *    05     073       003270      001778     61
      *    06     088       028636      015577     87
      *
      *                                                      AM HOURS
      *    01     017       000305      000165     10
      *    02     032       000624      000339     19
      *    03     047       001072      000583     31
      *    04     062       001880      001022     46
      *    05     077       004331      002356     68
      *    06     092
      *
      *    00     002       000034      000018     02         --NOON--
      *
      *    HOUR LINE ANGLES ARE CLOSE
      *    NOON ANGLE IS PRETTY GOOD
      *
      *    FREE SUNDIAL NOTES AT   WWW.ILLUSTRATINGSHADOWS.COM
      *
      *    THIS DIAL IS WEST OF MERIDIAN. IF EAST OF MERIDIAN THEN SWITCH AM FOR PM
      *    CHECK   WWW.ILLUSTRATINGSHADOWS.COM FOR THE LATEST PROGRAMS
      *    DOWNLOAD   MICRO-SHADOWS.PDF  FROM THE WEBSITE FOR TIPS AND FAQS
      *
      *
      *
      *
              ORG   401                 GOOD STARTING PLACE
      START   CS    0332                CLEAR THE
              CS                        PRINT AREA
              W                         BLANK LINE
```

```
*
*       SAY THIS IS A HORIZONTAL DIAL
*
        MCW   DTYPE,0290         MOVE HDIAL HEADER
        W                        PRINT IT
        CS    0290               CLEAR STORAGE
        MCW   EQUAL,0290         MOVE EAUALS
        W                        PRINT THEM
        CS    0290               THEN PROCEED
        W                        SPARE LINE
********************************************************************************
*       CARD 1     LATITUDE                                                  ***
********************************************************************************
        R                        READ LATITUDE OR 1ST CARD
        MCW   0002,LATZZ          SAVE LATITUDE
********************************************************************************
*       CARD 2     DIFF LONG                                                 ***
********************************************************************************
        R                        READ DIFF LONG OR 2ND CARD
        MCW   0002,LNGDL          SAVE LONGITUDE DIFFERENCE
********************************************************************************
*       CARD 3     EAST OR WEST                                             ***
********************************************************************************
        R                        READ EAST OR WEST ON 3RD CARD
        BCE   WEST,0001,W         IF WEST OK
        MCW   LNGEOW,LNGES        ELSE MAKE EAST
*
WEST    MZ    LATZZ,LAT           CLEAR ZONES
        MCW   ISLAT,0250          DESCRIPTION
        MCW   LAT,0260            AND DISPLAY IT
        W                        PRINT IT
        CS    0320               CLEAR PRINT AREA
        CS                       ALL OF IT
        W                        SPARE LINE
*
*       LOCATE SIN OF THE DESIRED LATITUDE
*
        LCA   LAT,SININ           * SET SIN IN
        B     SINFN               * DO SIN FUNCTION
        MCW   SINOUT,SINLAT       * GET SIN OUT
        CS    0280               CLEAR PRINT AREA
        MCW   SINLAT,0260         SAY SIN LAT
        MCW   SLATMS,0250         SAY WHAT THIS IS
        MCW   DOT,0257            MAKE DECIMAL PRETTY
        W                        DISPLAY IT
        CS    0280               AND THEN DO A
        W                        BLANK LINE
*
*       DETERMINE DIF IN LONGITUDES
*
WESTS   MCW   LNGDL,0260          SAY LONGITUDE DIFFERENCE
        MCW   LNGMSG,0250         SAY WHAT THIS IS
        C     LNGES,LNGEOW        IF EQ THEN EAST
        BE    EASTMSG
        MCW   HEADRA,0285         PROMPTS
        B     WESTMSG
EASTMSG MCW   HEADRAE,0285        PROMPTS
WESTMSG NOP
        W                        PRINT
        CS    0290               CLEAR
        W                        PRINT
        CS    0280               CLEAR
        ZA    LNGDL,LNGMIN        GET DEGREES
        A     LNGMIN,LNGMIN       TIMES 2
        A     LNGMIN,LNGMIN       TIMES 4
        MCW   MINMSG,0249         SAY MINUTES DIFF
        MZ    LATZZ,LNGMIN        CLEAR ZONES
        MCW   LNGMIN,0260         AND ITS VALUE
        W                        PRINT
        CS    0280               CLEAR
        W                        BLANK LINE
```

```
          CS    0290                 CLEAR STORAGE
          MCW   EQUAL,0290           MOVE EQUALS
          W                          PRINT THEM
          CS    0290                 THEN PROCEED
          W                          SPARE LINE
          W                          SPARE LINE
          W                          SPARE LINE
* - - - - - - - - - - - - - - - - - - - - - - - - - - -
* END OF LONGITUDE FROM LEGAL MERIDIAN CALCULATIONS
* - - - - - - - - - - - - - - - - - - - - - - - - - - -
          CS    0280
          MCW   HEADR,0270           HR,HRANGL,TAN, ETC
* /////////////////////////////////////////////////////
*         -----------------------------------------------
*         HR   HRANGL   TAN.-------   TAN.HLA   HLA
*         *        *              *         *     *
*         230      239            253       263   270
*         -----------------------------------------------
* /////////////////////////////////////////////////////
          W                          PRINT HEADER
          CS    0280                 CLEAR PRINT AREA
          MCW   HDRTAN,0253          SAY TIMES 1000
          MCW   HDRTAN,0263          AND TIMES 1000
          W
          CS    0280
          W
*
*     FIRST - - - - LOOP ON HR AFTERNOON HOURS
*
          MCW   PMMSG,0290           SAY PM HOURS
          ZA    HR00,HRWKG           PRIME INITIAL HOURS FROM NOON
*
HLOOP     SW    0097                 WM FOR INDEX 3
          MCW   HRWKG,0099           FOR DEBUGGING PUT HRWKG IN X3
          C     HRLIM,HRWKG          HAVE WE HIT A LIMIT
          BH    HHALT                IF WKG GT LIM
*
          MCW   HRWKG,0230           HR OF DAY FROM NOON
          C     HR00,HRWKG           IF NOON SKIP IT
          BE    SKIP01               AS NOON DONE ELSEWHERE
          ZA    HRWKG,HRWKGM         COPY HOUR TO HOUR ANGLE
          ZA    HRDEG,MPLIER         * SET MULTILPIER   * 15 DEG PER HR
          ZA    HRWKGM,MCAND         * SET MULTIPLICAND * WORKING HR
          B     MULTP                * DO MULTIPLY      * MULTIPLY
          MCW   MULTPP,HRANGL        * GET PRODUCT      * RESULTS
          SW    0246                 LIMIT RESULT OF HR ANGLE SIZE
*     NOW WE ADD OR SUBTRACT SOME HOUR ANGLE DEGREES BASED
*        ON         1 LNGES DCW* 0 WHERE 0 IS WEST 1 IS EAST
*        AMOUNT     2 LNGDL DCW* 00LONGITUDE DIFFERENCE
          B     SBWEST,LNGES,0       MEANS WEST SO SUB
          A     LNGDL,HRANGL         HRANGL IS FIXED
          B     ADEAST               ADDED AS EAST
SBWEST    B     SKIP01,HRWKG,0       DO NOTHING AS NEGATIVE INDEX
          S     LNGDL,HRANGL         SUB IF WE ARE WEST HOWEVER
*                                    UPSETS THINGS
ADEAST    MZ    ZERO,HRANGL          GET NICE ZONES
          SW    0237                 LIMIT SIZE OF HRANGL TO 3 CHARS
          MCW   HRANGL,0239          RESULTS TO PRINT AND WAS 10 CHARS
          MCW   HRANGL,CURHRA        GET 2 CHARS OF HOUR ANGLE ADJUSTED
          B     SKIP01,CURHRA-001,9     ANGLE 90 OR GREATER
          MCW   HRANGL,TANIN         * SET TAN 2 CHARS  *
          B     TANFN                * DO TAN FUNCTION  *
          MCW   TANOUT,HRATAN        * GET TAN OUT      *
          MCW   HRATAN,0253          PRINT ATAN HRA
          ZA    SINLAT,MPLIER        * SET MULTILPIER   *
          ZA    HRATAN,MCAND         * SET MULTIPLICAND *
          B     MULTP                * DO MULTIPLY      *
          MCW   MULTPP-003,HLANGT    * GET PRODUCT      *
          MCW   HLANGT,0263          PRINT IT
          ZA    HLANGT,ATNIN         * SET ATAN 6 CHARS *
          B     ATNFN                * DO ATAN FUNCTION *
```

```
        MCW   ATNOUT,HLANGL      * GET ANGLE 2 CHARS*
        MZ    HR00,HLANGL        LOSE THE ZONE
        MCW   HLANGL,0270
*
*     ONE COMPLETE LINE DERIVED
*
SKIP01  W                        PRINT THE DATA
        CS    0320               CLEAR PRINT AREA
        CS                       ALL OF IT
        A     ONE,HRWKG          ADD ONE TO HRWKG
        B     HLOOP              DO AGAIN
***
***   NOW - - - LOOP ON HR AFTERNOON HOURS
***
HHALT   CS    0320               CLEAR PRINT AREA
        CS                       ALL OF IT
        W                        SEPARATOR LINE
        MCW   AMMSG,0290         SAY AM HOURS
* //////////////////////////////////////////////////////////
*      ------------------------------------------------
*         HR    HRANGL   TAN.-------   TAN.HLA    HLA
*          *       *              *         *       *
*         230     239            253       263     270
*      ------------------------------------------------
* //////////////////////////////////////////////////////////
        ZA    HR00,HRWKG         PRIME INITIAL HOURS FROM NOON
HLOOP1  MZ    HR00,HRWKG         ZA STILL LEFT A ZONE
        C     HRLIM,HRWKG        HAVE WE HIT A LIMIT
        BH    HHALT2             IF WKG GT LIM
        C     HR00,HRWKG         IF NOON SKIP IT
        BE    SKIP02             AS NOON DONE ELSEWHERE
        MCW   HRWKG,0230         HR OF DAY FROM NOON
        ZA    HRWKG,HRWKGM       COPY HOUR TO HOUR ANGLE
        ZA    HRDEG,MPLIER       * SET MULTILPIER  * 15 DEG PER HR
        ZA    HRWKGM,MCAND       * SET MULTIPLICAND * WORKING HR
        B     MULTP              * DO MULTIPLY      * MULTIPLY
        MCW   MULTPP,HRANGL      * GET PRODUCT      * RESULTS
        SW    0237               LIMIT RESULT OF HR ANGLE SIZE
        B     SUWEST,LNGES,0     0 MEANS WEST SO ADD IF MORNING
        S     LNGDL,HRANGL       HRANGL IS FIXED
        B     ADEEST             SUBTRACTED AS EAST
SUWEST  B     SKIP02,HRWKG,0     DO NOTHING AS NEGATIVE INDEX
        A     LNGDL,HRANGL       SUB IF WE ARE WEST HOWEVER
ADEEST  MZ    ZERO,HRANGL        GET NICE ZONES
        MCW   HRANGL,0239        RESULTS TO PRINT AND WAS 10 CHARS
        MCW   HRANGL,CURHRA      GET 2 CHARS OF HOUR ANGLE ADJUSTED
        B     SKIP02,CURHRA-001,9   ANGLE 90 OR GREATER
        MCW   HRANGL,TANIN       * SET TAN 2 CHARS  *
        B     TANFN              * DO TAN FUNCTION  *
        MCW   TANOUT,HRATAN      * GET TAN OUT      *
        MCW   HRATAN,0253        PRINT ATAN HRA
        ZA    SINLAT,MPLIER      * SET MULTILPIER   *
        ZA    HRATAN,MCAND       * SET MULTIPLICAND *
        B     MULTP              * DO MULTIPLY      *
        MCW   MULTPP-003,HLANGT  * GET PRODUCT      *
        MCW   HLANGT,0263        PRINT IT
        ZA    HLANGT,ATNIN       * SET ATAN 6 CHARS *
        B     ATNFN              * DO ATAN FUNCTION *
        MCW   ATNOUT,HLANGL      * GET ANGLE 2 CHARS*
        MZ    HR00,HLANGL        LOSE THE ZONE
        MCW   HLANGL,0270
***
***   ONE COMPLETE LINE DERIVED
***
SKIP02  W                        PRINT THE DATA
        CS    0320               CLEAR PRINT AREA
        CS                       ALL OF IT
        A     ONE,HRWKG          ADD ONE TO HRWKG
        B     HLOOP1             DO AGAIN
***
***   DO THE NOON HOUR DATA
```

```
***
HHALT2    W                         GET A CLEAR
          CS   0290                  LINE
* /////////////////////////////////////////////////////
*        ---------------------------------------------
*          HR    HRANGL   TAN.-------   TAN.HLA   HLA
*           *       *              *        *       *
*          230     239            253      263     270
*        ---------------------------------------------
* /////////////////////////////////////////////////////
          MCW  NNMSG,0290            SAY NOON HOURS
          MCW  HR00,0230             HR OF DAY FROM NOON
          ZA   LNGDL,HRANGL          * GET PRODUCT      * RESULTS
          MZ   ZERO,HRANGL           CLEAR THE ZONE
          SW   0237                  LIMIT SIZE OD HR ANGL
          MCW  HRANGL,0239           RESULTS TO PRINT AND WAS 10 CHARS
          MCW  HRANGL,CURHRA         GET 2 CHARS OF HOUR ANGLE ADJUSTED
          MCW  HRANGL,TANIN          * SET TAN 2 CHARS  *
          B    TANFN                 * DO TAN FUNCTION  *
          MCW  TANOUT,HRATAN         * GET TAN OUT      *
          MCW  HRATAN,0253           PRINT ATAN HRA
          ZA   SINLAT,MPLIER         * SET MULTILPIER   *
          ZA   HRATAN,MCAND          * SET MULTIPLICAND *
          B    MULTP                 * DO MULTIPLY      *
          MCW  MULTPP-003,HLANGT     * GET PRODUCT      *
          MCW  HLANGT,0263           PRINT IT
          ZA   HLANGT,ATNIN          * SET ATAN 6 CHARS *
          B    ATNFN                 * DO ATAN FUNCTION *
          MZ   ZERO,ATNOUT           CLEAR THE ZONE
          MCW  ATNOUT,0270           * GET ANGLE 2 CHARS*
          W                          PRINT ANY RESIDUAL DATA
          CS   0320                  CLEAR PRINT AREA
          CS                         ALL OF IT
          W                          PRINT A BLANK LINE
          MCW  HEADR3,0259           ADVISE ON ACCURACY
          W                          SAY SO
          MCW  HEADR4,0259           ADVISE ON NOON ISSUES
          W
          CS   0320                  CLEAR PRINT AREA
          CS                         ALL OF IT
          W                          PRINT A BLANK LINE
          MCW  HEADR5,0254           GET HELP AT THIS URL
          MCW  HEADR6,0284           GET HELP AT THIS URL
          W
          CS   0290                  CLEAR
          W
          MCW  EQUAL,0290            MOVE EQUALS
          W                          PRINT THEM
          CS   0290                  CLEAR
          W
          C    LNGES,LNGEOW          IF EQ THEN EAST
          BE   EASTHDR
          MCW  HEADR7,0283           PROMPTS
          B    WESTHDR
EASTHDR   MCW  HEADR7E,0283          PROMPTS
WESTHDR   NOP
          W
          MCW  HEADR8,0285           PROMPTS
          W
          MCW  HEADR9,0289           PROMPTS
          W
*         THE REAL 1401 HAS HI ORDER ADDRESSED
*         AND LOW ORDER HAS A WM, OPPOSITE TO
*         ALL OTHER DATA ADDRESSING WHICH IS TO
*         ADDRESS LOW ORDER AND STOP ON HI ORDER
*         WM. THIS IS BECAUSE OF SERIAL DATA XFR
*         TO THE TYPEWRITER.
          MU   %TO,HMSG,W            TYPE WRITER MESSAGE
          H    START                 *** END PROGRAM ***
*
**
```

```
****
********
****
**
*
********************************************************************************
*                  L A T I T U D E    D A T A                                  *
********************************************************************************
BASLAT    DCW  00                      A BASE LATITUDE - SINCE SW A NOT USABLE  *
*              30                      A GOOD BASE LATITUDE IS 00 OR 30         *
********************************************************************************
*
LATZZ     DCW  32
LAT64     DCW  64
LAT32     DCW  32
LAT16     DCW  16
LAT08     DCW  08
LAT04     DCW  04
LAT02     DCW  02
LAT01     DCW  01
LAT       DCW  33
*
L91       DCW  91                      TEST FOR BAD LATITUDE
L90       DCW  90                      CORRECT BAD LATITUDE
DOT       DCW  @.@                     MAKE SENSE OF DECIMAL ON PRINTOUT
BLAT      DCW  @LAT GT 90@             ERROR MESSAGE
*
*              AREA IS THE HIGH ORDER ADDRESS, AND WE
*              HAVE A WORD MARK ON THE LAST CHAR
HMSG      DC   @*@
          DC   @** NORMAL EOJ **@
          DCW  @*@
*
          DCW  @HORIZONTAL SUNDIAL @
          DC   @PROGRAM ON THE IBM @
          DC   @1401 - - -  S @
DTYPE     DC   @WHEATON-SMITH@
*
          DCW  @=================@
          DC   @==================@
          DC   @=============@
EQUAL     DC   @============@
*
ISLAT     DCW  @LATITUDE@         NAME THE NEXT FIELD TO PRINT
BSLAT     DCW  @BASE LATITUDE@    NAME THE NEXT FIELD TO PRINT
USLAT     DCW  @ENTERED LATITUDE@ NAME THE NEXT FIELD TO PRINT
SLATMS    DCW  @SIN LAT@          NAM SIN LAT
SINLAT    DCW  0000              1000 * SIN OF LATITUDE
*
INDX      DCW  000               DESIRED LATITUDE READY FOR MULTIPLY
K032      DCW  032               SIZE OF AN ENTRY FOR MULTIPLY
ZERO      DCW  0                 INITIAL INDEX VALUE
*
CTR       DCW  00                COUNT UP TO LATITUDE IN LOOP
ONE       DCW  01                DECREMENT AMOUNT
*
*
*     OTHER DATA
*
*            HR    HRANGL   TAN.-------   TAN.HLA   HLA
*             *       *             *       *       *
*            230     239          253     263     270
*
*
          DCW  @HR   HRANGL   TAN.@
          DC   @-------    TAN.HLA @
HEADR     DC   @   HLA@
*
          DCW  @HOUR LINE ANGLES @
HEADR3    DC   @ARE CLOSE@
*
```

```
            DCW   @NOON ANGLE IS ALS@
HEADR4      DC    @O CLOSE  @
*
            DCW   @FREE SUNDIAL @
HEADR5      DC    @NOTES AT@
*
HEADR6      DCW   @WWW.ILLUSTRATINGSHADOWS.COM@
*
            DCW   @THIS DIAL IS WEST @
            DC    @OF MERIDIAN. IF EA@
HEADR7      DC    @ST SWITCH AM FOR PM@
*
            DCW   @THIS DIAL IS EAST @
            DC    @OF MERIDIAN. IF WE@
HEADR7E     DC    @ST SWITCH AM FOR PM@
*
HEADRA      DCW   @WEST OF MERIDIAN@
HEADRAE     DCW   @EAST OF MERIDIAN@
*
            DCW   @CHECK WWW.ILLUS@
            DC    @TRATINGSHADOWS.COM@
            DC    @ FOR THE LATEST PR@
HEADR8      DC    @OGRAMS@
*
            DCW   @DOWNLOAD MICRO-@
            DC    @SHADOWS.PDF FROM@
            DC    @ THE WEBSITE FOR T@
HEADR9      DC    @IPS AND FAQS@
*
HDRTAN      DCW   @*1K@                 SAYS TIMES 1000
*
HR00        DCW   00                    START HR FROM NOON
HRM6        DCW   06                    MINUS 6
HRK6        DCW   06                    SIX
HRWKG       DCW   00                    WORKING HOUR
HRLIM       DCW   06                    SIX HOURS
*                                       A TWO DIGIT LIMIT ON SIN COS TAN
HRDEG       DCW   000015                15 DEGREES PER HOUR
HRWKGM      DCW   000000                WORKING HOUR FOR MULTIPLY
HRANGL      DCW   0000000000            HR ANGLE IS HR FR NOON * 15
HLANGT      DCW   000000                HR LINE ANGLE AS A TAN
HRATAN      DCW   000000                HR ANGLE BUT TAN
HLANGL      DCW   00                    RESULTS OF ATAN
CURHRA      DCW   00                    HOUR ANGLE AFTER LONG DIFF CALC
*
LNGZZ       DCW   00                    LONGITUDE DIFFERENCE ZEROES
LNGDL       DCW   3                     LONGITUDE DIFFERENCE
LNGEOW      DCW   1                     1 IS EAST
LNGES       DCW   0                     0 IS WEST 1 IS EAST
LNGMSG      DCW   @LNG DIFF@            NAME THE NEXT FIELD TO PRINT
REFMSG      DCW   @1-EAST 0-WEST@       NAM EAST OR WEST
MINMSG      DCW   @MINUTES DIFF@        SAY MINUTES OF DIFF
LNGMIN      DCW   00                    VALUE OF MINUTES OF DIFF
*
PMMSG       DCW   @PM HOURS@
AMMSG       DCW   @AM HOURS@
NNMSG       DCW   @--NOON--@
*
*
****************************************************************************
*
**********************************************************************
*                                                                    *
* - - M A T H   F U N C T I O N S   F O L L O W   - - - - - - -*
*                                                                    *
**********************************************************************
*
*
*
* - - M U L T I P L Y - - - - - - - - - - - - - - - - - - - - - - -
*
```

```
*
**********************************************************
*     BEGIN MULTIPLY SUBROUTINE * USES SBR AND 3 PARMS *
**********************************************************
*     MPLIER - -MULTP - - MULTPP  ARE RESERVED LABELS   *
*     MCAND- - -                  SET WITH ZA NOT MCW   *
*     6                 10        SIMPLE PRODUCT         *
**********************************************************
*    INPUT EACH  6 CHARS MAX   * OUTPUT 10 CHARS        *
**********************************************************
*    ZA VAL1        MPLIER      * SET MULTILPIER         *
*    ZA VAL2        MCAND       * SET MULTIPLICAND       *
*    B  MULTP                   * DO MULTIPLY            *
*    MCWMULTPP      XXXXX       * GET PRODUCT            *
**********************************************************
MULTP     SBR  MULTPV-001         SAVE RETURN ADDRESS
          LCA  MULTPC,MULTPA       CLEAR ENTIRE AREA
          LCA  MPLIER,MULTPA-020   LOAD MULTIPLIER TO -20
          ZA   MULT00,MULTPP       SET PRODUCT TO 0
          C    MPLIER,MULT00       IS MULTIPLIER 0
*                                  U IF SO STOP - U MEANS B GT A
*                                  T IF SO STOP - T MEANS B LT  A
*                                  / IF SO STOP - / MEANS B NE  A
*                                  S IF SO STOP - S MEANS B EQ  A
          BE   MULTPR              IF ZERO THEN EXIT
          C    MCAND,MULT00        IS MULTIPLICAND 0
          BE   MULTPR              S IF ZERO THEN EXIT
MULTPZ    MN   MULTPA-020,MULTPT   MOVE CURRENT LOW ORDER CHAR
          BCE  MULTPM,MULTPT,0     BRANCH ZERO
BEGIN     A    MCAND,MULTPA-010    ADD MCAND
          S    MULTP1,MULTPA-020   SUB 1 FROM MULTIPLIER
          B    MULTPZ              REPEAT
MULTPM    BWZ  MULTPX,MULTPA-020,1 TEST FOR WM
          LCA  MULTPA-001,MULTPA   SHIFT AREA RIGHT ONE POS
          B    MULTPZ              REPEAT FOR THIS PART
MULTPX    MCW  MULTPA-002,MULTPP   MULTIPLY IS COMPLETE
MULTPR    B    0000                ADDRESS COMES FROM SBR
MULTPV    NOP                      SBR TO HERE -1 AS PLUS
*                                  UPSET SPS ASSEMBLER
*
*                    -20        -10         -0
*                ----10---*----10---*----10---*
*                  MPLIER     MCAND      MULTPP
MULTPC    DCW  00000000000000000000000000000000
*
MULTPA    DCW  00000000000000000000000000000000   MPLIER/MCAND/PRODUCT
MPLIER    DCW  000000000            MULTIPLIER - - 9 CHARS
MCAND     DCW  000000000            MULTIPLICAND - 9 CHARS
MULTPP    DCW  0000000000           PRODUCT - - - 10 CHARS
MULT00    DCW  000000000            0 BUT IS 9 CHARS FOR COMPARE
MULTP1    DCW  1                    VALUE OF 1 FOR SUBTRACT
MULTPT    DCW  0                    TEST AREA FOR ONE BYTE
**********************************************************
*    END MULTIPLY SUBROUTINE                            *
**********************************************************
*
*
*
* - - S I N - - - - - - - - - - - - - - - - - - - - - - - - - - - -
*
*
*
**********************************************************
*    BEGIN SIN SUBROUTINE       * USES SBR AND 2 PARMS *
**********************************************************
*    INPUT       2 CHARS        * OUTPUT  4 CHARS       *
**********************************************************
*    SININ - - SINFN - - SINOUT  ARE RESERVED LABELS   *
*    2                 4         SIN * 1000             *
**********************************************************
*    LCAVAL1        SININ      * SET SIN IN             *
```

```
*     B   SINFN               * DO SIN FUNCTION     *
*     MCWSINOUT    XXXXX      * GET SIN OUT         *
************************************************************
SINFN     SBR  SINEXT-001          SAVE RETURN ADDRESS
          LCA  SININ0,SININI       RESET IN CASE USED BEFORE
          MCW  SININ,SININI        LOAD THEIR ANGLE IN DEGREES
          A    SININI,SININI       DOUBLE IT
          A    SININI,SININI       NOW FOUR TIMES
          MCW  SININI,0089         87-89 IS INDEX REG 1
*                                  92-94 IS INDEX 2, 97-99 IS INDEX 3
* ------------------------------- USE INDEX REGISTER 1
          MCW  SIN00&X1,SINOUT     GET SIN00 PLUS INDEX 1
*              &X1                 THE  1 MEANS INDEX WITH REGISTER 1
          B    0000                ADDRESS COMES FROM SBR
SINEXT    NOP                      SBR TO HERE -1 AS PLUS
*                                  UPSET SPS ASSEMBLER
SININ     DCW  00                  SIN ANGLE IN DEGREES INCOMING
SININ0    DCW  000                 MAKES THIS SERIALLY REUSABLE
SININI    DCW  000                 WORKING ANGLE IN DEGREES
SINOUT    DCW  0000                SIN OF ANGLE * 1000 OUTGOING
************************************************************
*   END SIN FUNCTION SUBROUTINE                           *
************************************************************
*
*
*
*
*
*
* - - C O S I N E - - - - - - - - - - - - - - - - - - - - - - - - - -
*
*
*
************************************************************
*     BEGIN COS SUBROUTINE      * USES SBR AND 2 PARMS *
************************************************************
*     LOGIC IS TO SUB THEIR ANGLE FROM 90 AND THEN USE *
*     THAT AS ANGLE INTO THE SIN SERIES TO SAVE SPACE  *
*     WHEREAS COS BY ITSELF HAS ITS OWN TABLE          *
************************************************************
*     INPUT      2 CHARS      * OUTPUT  4 CHARS      *
************************************************************
*     COSIN - - COSFN - - COSOUT  ARE RESERVED LABELS  *
*     2             4        COS * 1000               *
************************************************************
*     LCAVAL2      COSIN       * SET COS IN           *
*     B  COSFN                 * DO COS FUNCTION      *
*     MCWCOSOUT    XXXXX       * GET COS OUT          *
************************************************************
COSFN     SBR  COSEXT-001          SAVE RETURN ADDRESS
          MCW  COS90,COSINI        SET 90 DEGREES TO OUR ANGLE
          S    COSIN,COSINI        SUB THEIR ANGLE FROM 90
          MN   COSINI,COSWK        GET LOW ORDER AND LOSE ZONES
          MCW  COS00,COSINI        ONE BYTE ONLY
          MN   COSWK,COSINI        NOW COSINI HAS NO ZONES
*     NOW THIS IS SIN
          A    COSINI,COSINI       DOUBLE IT
          A    COSINI,COSINI       NOW FOUR TIMES
          MCW  COSINI,0089         87-89 IS INDEX REG 1
*                                  92-94 IS INDEX 2, 97-99 IS INDEX 3
* ----------------------------- USE INDEX REGISTER 1
          MCW  SIN00&X1,COSOUT     GET COS00 IE SIN00 NOW PLUS INDEX 1
*              &X1                 THE 1 MEANS INDEX WITH REGISTER 1
          B    0000                ADDRESS COMES FROM SBR
COSEXT    NOP                      SBR TO HERE -1 AS PLUS
*                                  UPSET SPS ASSEMBLER
COSIN     DCW  00                  SIN ANGLE IN DEGREES INCOMING
COSINI    DCW  000                 WORKING ANGLE IN DEGREES
COS90     DCW  090                 90 FOR 90-ANGLE HENCE SIN
COS00     DCW  0                   0 FOR INITIALIZING
COSWK     DCW  0                   FOR LOSING ZONES
COSOUT    DCW  0000                COS OF ANGLE * 1000 OUTGOING
```

```
*********************************************************
*   END COS FUNCTION SUBROUTINE                        *
*********************************************************
*
*
*
* - - T A N G E N T - - - - - - - - - - - - - - - - - - - - - - - - - - -
*
*
*
*********************************************************
*     BEGIN TAN SUBROUTINE     * USES SBR AND 2 PARMS *
*********************************************************
*     INPUT     2 CHARS     * OUTPUT  6 CHARS     *
*********************************************************
*     TANIN - - TANFN - - TANOUT  ARE RESERVED LABELS  *
*     2            6          TAN * 1000          *
*********************************************************
*     LCAVAL1        TANIN      * SET TAN IN         *
*     B  TANFN                  * DO TAN FUNCTION     *
*     MCWTANOUT     XXXXX       * GET TAN OUT         *
*********************************************************
TANFN      SBR  TANEXT-001        SAVE RETURN ADDRESS
           LCA  TANIN0,TANINI     RESET IN CASE USED BEFORE
           MCW  TANIN,TANINI      LOAD THEIR ANGLE IN DEGREES
           A    TANINI,TANINI     DOUBLE IT
           A    TANINI,TANINI     NOW FOUR TIMES
           A    TANIN,TANINI      NOW FIVE TIMES
           A    TANIN,TANINI      NOW SIX TIMES
           MCW  TANINI,0089       87-89 IS INDEX REG 1
*                                 92-94 IS INDEX 2, 97-99 IS INDEX 3
* ------------------------------- USE INDEX REGISTER 1
           MCW  TAN00&X1,TANOUT   GET TAN00 PLUS INDEX 1
*               &X1               THE 1 MEANS INDEX WITH REGISTER 1
           B    0000              ADDRESS COMES FROM SBR
TANEXT     NOP                    SBR TO HERE -1 AS PLUS
*                                 UPSET SPS ASSEMBLER
TANIN      DCW  00                SIN ANGLE IN DEGREES INCOMING
TANIN0     DCW  000               MAKES THIS SERIALLY REUSABLE
TANINI     DCW  000               WORKING ANGLE IN DEGREES
TANOUT     DCW  000000            TAN OF ANGLE * 1000 OUTGOING
*********************************************************
*   END TAN FUNCTION SUBROUTINE                        *
*********************************************************
*
*
*
*
* - - C O T A N - - - - - - - - - - - - - - - - - - - - - - - - - - - -
*
*
*
*********************************************************
*     BEGIN COT SUBROUTINE     * USES SBR AND 2 PARMS *
*********************************************************
*     LOGIC IS TO SUB THEIR ANGLE FROM 90 AND THEN USE *
*     THAT AS ANGLE INTO THE TAN SERIES TO SAVE SPACE  *
*     WHEREAS TAN BY ITSELF HAS ITS OWN TABLE          *
*********************************************************
*     INPUT     2 CHARS     * OUTPUT  4 CHARS     *
*********************************************************
*     COTIN - - COTFN - - COTOUT  ARE RESERVED LABELS  *
*     2            4          COS * 1000          *
*********************************************************
*     LCAVAL2        COTIN      * SET COS IN         *
*     B  COTFN                  * DO COS FUNCTION     *
*     MCWCOTOUT     XXXXX       * GET COS OUT         *
*********************************************************
COTFN      SBR  COTEXT-001        SAVE RETURN ADDRESS
           MCW  COT90,COTINI      SET 90 DEGREES TO OUR ANGLE
           S    COTIN,COTINI      SUB THEIR ANGLE FROM 90
```

```
        MN   COTINI,COTWK         GET LOW ORDER AND LOSE ZONES
        MCW  COT00,COTINI         ONE BYTE ONLY
        MN   COTWK,COTINI         NOW COTINI HAS NO ZONES
        MCW  COTINI,COTWIN        SAVE WHAT WAS IN FROM 90
*    NOW THIS IS TAN
        A    COTINI,COTINI        DOUBLE IT
        A    COTINI,COTINI        NOW FOUR TIMES
        A    COTWIN,COTINI        NOW FIVE TIMES
        A    COTWIN,COTINI        NOW SIX TIMES
        MCW  COTINI,0089          87-89 IS INDEX REG 1
*                                 92-94 IS INDEX 2, 97-99 IS INDEX 3
* ---------------------------- USE INDEX REGISTER 1
        MCW  TAN00&X1,COTOUT      GET COT00 IE TAN00 NOW PLUS INDEX 1
*            &X1                  THE 1 MEANS INDEX WITH REGISTER 1
        MCW  TAN00    10220
        B    0000                 ADDRESS COMES FROM SBR
COTEXT  NOP                       SBR TO HERE -1 AS PLUS
*                                 UPSET SPS ASSEMBLER
COTIN   DCW  00                   TANANGLE IN DEGREES INCOMING
COTINI  DCW  000                  WORKING ANGLE IN DEGREES
COTWIN  DCW  000                  USED AS WE MULT BY 6
COT90   DCW  090                  90 FOR 90-ANGLE HENCE TAN
COT00   DCW  0                    0 FOR INITIALIZING
COTWK   DCW  0                    FOR LOSING ZONES
COTOUT  DCW  0000                 COT OF ANGLE * 1000 OUTGOING
*********************************************************
*  END COT FUNCTION SUBROUTINE                          *
*********************************************************
*
* - - A R C T A N - - - - - - - - - - - - - - - - - - - - - - - - - -
*
*
*
*********************************************************
*     BEGIN ATN SUBROUTINE      * USES SBR AND 2 PARMS *
*********************************************************
*     INPUT     6 CHARS         * OUTPUT  2 CHARS      *
*********************************************************
*     LOGIC     THE COMPARE HI AND LO WORKS ON THE     *
*               SIM THAT SUPPORTS THIS AUTOCODER SO    *
*               THAT IS WHY ATTOCODER HAS JUST ONE     *
*               PASS AND WHY SPS HAS MULTIPLE PASSES   *
*               IT IS A SIMULATOR ISSUE NOT AN SPS     *
*               NOR AUTOCODER ISSUE.                   *
*********************************************************
*     ATNIN - - ATNFN - - ATNOUT  ARE RESERVED LABELS  *
*     6                          TAN * 1000 IN AND     *
*                     2          ANGLE IS 2 BACK       *
*********************************************************
*    ZA VAL1       ATNIN       * SET ATN IN            *
*    B  ATNFN                  * DO ATN FUNCTION       *
*    MCWATNOUT    XXXXX        * GET ATN OUT IN DEGREES *
*********************************************************
ATNFN   SBR  ATNEXT-001          SAVE RETURN ADDRESS
        ZA   ATNZRO,ATNANG       SET 000 TO RESULTING ANGLE TO 00
        ZA   ATNZRO,ATNNDX       COPY OF I1
        MZ   ATNZO,ATNNDX        LOSE ZONES
        MZ   ATNZO,ATNNDX-1      LOSE ZONES
        MZ   ATNZO,ATNNDX-2      LOSE ZONES
        MCW  ATNNDX,0089         SET INDEX 1 AT 0089 TO 0 INITIALLY
*                                87-89 IS INDEX REG 1
*                                92-94 IS INDEX 2, 97-99 IS INDEX 3
*
*    LOOP 1 LOOKS FOR EXACT MATCH
*
ATNFN1  MZ   ATNZO,ATNANG        CLEAR ZONE
        ZA   ATNANG,ATNOUT       SAVE ANGLE AS OUTPUT JUST IN CASE
        MCW  ATNNDX,0089         SET INDEX 1 AT 0089 TO NEW VALUE
*            &X1*----------MEANS USE INDEX 1
        ZA   TAN00&X1,ATNWK      COPY TABLE INDEXED FOR COMPARE
        C    ATN90,ATNANG        COMPARE 90 TO CURRENT ANGLE SO FAR
```

```
          BH    ATNFN2              IF 90 LOWER THAN WORKING ANGLE THEN EXIT
          BE    ATNFN2              IF 90 EQUALS WORKING ANGLE THEN EXIT
          MZ    ATNZO,ATNWK         CLEAR
          MZ    ATNZO,ATNIN         ZONES FOR COMPARE
          C     ATNWK,ATNIN         COMPARE CURRENT TAN TABLE ENTRY TO INPUT
          BE    ATNEXX              ATN00 TABLE ENTRY EQUALS OUR PARAMETER
          BL    ATNEXX              IF IN IS LOW THEN USE WHAT WE GOT
*     WORKING ANGLE LT 90 AND NOT EQUL WHERE WE ARE IN TABLE
          A     ATNSIX,ATNNDX       SO ADD 6 TO OUR COPY OF INDEX
          A     ATNONE,ATNANG       AND ADD 1 TO FINAL ANGLE
          B     ATNFN1              AND DO IT AGAIN
*
ATNFN2    ZA    ATN90,ATNOUT        SET 90 IF THINGS ARE BALLED UP THEN EXIT
ATNEXX    MZ    ATNZO,ATNOUT        FIX ZONE
          B     0000                ADDRESS COMES FROM SBR
ATNEXT    H     0999                SBR TO HERE -1 AS PLUS
*
*     LOOP 2 IS NOT USED IN THE AUTOCODER VERSION BECAUSE
*     THE SIMULATOR COMPARE BH BL WORKS HERE
*
ATNZRO    DCW   000                 ZERO TO START SEARCH
ATNZO     DCW   00                  ZERO FOR COMPARE
ATNANG    DCW   00                  ANGLE I.E. NTH ENTRY IN TAN TABLE
ATNSIX    DCW   006                 INCREMENT SIZE FOR TAN TABLE
ATNONE    DCW   001                 INCREMENT SIZE FOR DEGREES
ATNNDX    DCW   000                 INDEX 0 6 12 ETC FOR I1
ATN90     DCW   90                  LIMIT
*
ATNOUT    DCW   00                  ANGLE IN DEGREES OUTGOING
ATNIN     DCW   000000              TAN OF ANGLE * 1000 INCOMING
ATNWK     DCW   000000              WKAREA - COPY FROM TAN00 INDEXED
*
*******************************************************
*  END ATN FUNCTION SUBROUTINE                        *
*******************************************************
*
*
*
* - - T A B L E S    F O R    T A N    A N D    A R C T A N - - - -
*
*      *** NOTE *** HERE ARE NO MINOR CHANGES OF ABOUT 1/10 DEGREE
*      HERE COMPARED TO THE SPS ATAN - THIS IS BECAUSE MULTIPLE
*      PASSES ARE NOT NEEDED SO THAT ADJUSTMENT IS NOT NEEDED.
*
*
TAN00     DCW   000000              EACH ENTRY IS 1000 * TAN
          DCW   000017
          DCW   000034
          DCW   000052
          DCW   000069
...
...
...
          DCW   028636
          DCW   057289
          DCW   999999
*
*
*
* - - T A B L E S    F O R    S I N    A N D    C O S I N E - -
*
*
*
SIN00     DCW   0000                EACH ENTRY IS 1000 * SIN
          DCW   0017
          DCW   0034
...
...
...
          DCW   0999
          DCW   0999
```

```
SIN90     DCW  1000
*
          END  START
```