TT POINTERS

WORD 0:

400000	BIT 0	NULL COUNT
200000	BIT 1	"
100000	BIT 2	"
40000	BIT 3	"
20000	BIT 4	"
10000	BIT 5	"
4000	BIT 6	(UNUSED)
2000	BIT 7	"
1000	BIT 8	"
400	BIT 9	"
200	BIT 10	IGNORE NEXT ECHO IF 1
100	BIT 11	RING MODE (ALWAYS 1)
40	BIT 12	TYPE-ACTIVE IF 1
20	BIT 13	8-BIT MODE IF 1
10	BIT 14	BEING INTERRUPTED IF 1
4	BIT 15	CONTROL MODE IF 1
2	BIT 16	FULL BUFFER ON INPUT IF 1
1	BIT 17	LINE OPEN IF 0

WORD 3:

	BITS 0-7	8 BIT MODE ALARM CHARACTER
1000	BIT 8	"SAVED 77" STATUS IN DISPATCHER
400	BIT 9	(UNUSED)
	BITS 10-17	STAT PTR TO OWNER

TT ERRORS (WORD 102)

1	NOT YOUR TELETYPE
2	INTERRUPTED BY NULL
3	TIS MAXIMUM EXCEEDED
4	ATTEMPT TO TIS INTO LOWER CORE (BELOW 40) OR INTO EXEC

FASTRAND ERRORS (WORD 102)

200000	TOTAL SYSTEM FAILURE OR DATA ERROR (NON-RECOVERABLE)
100000	IOP MAXIMUM EXCEEDED BY FASTRAND READ
40000	ILLEGAL SPECIFICATIONS OR BELOW BOUND
20000	NOT YOUR QUARTER TRACK
10000	NO MORE QUARTER TRACKS
4000	COMPARISON ERROR ON SCATTER GATHER
2000	REWRITE OR EXPUNGE ERROR (ON OWNWD OR REWRITE NUMBER)
1000	ILLEGAL ITEM, BLOCK, OR DRUM ADDRESS
400	ILLEGAL BEDSPACE CODE
200	NO SUCH UNIT NUMBER; OR BEDSPACE HAS NO PARAMETERS

TAPE ERRORS (WORD 102)

600000	TOTAL SYSTEM FAILURE OR DATA ERROR (NON-RECOVERABLE)
500000	IOP MAXIMUM EXCEEDED BY TAPE READ
440000	ILLEGAL SPECIFICATION OR BELOW BOUND
420000	NOT YOUR TAPE DRIVE
410000	NO MORE TAPE DRIVES
404000	COMPARISON ERROR ON SCATTER GATHER
402000	END OF FILE ENCOUNTERED WHILE READING OR SPACING
401000	END OF TAPE MARK ENCOUNTERED
400400	LOAD POINT ENCOUNTERED WHILE BACKSPACING TAPE
400200	ATTEMPT TO WRITE WITH WRITE RING OUT

LOWER CORE REGISTERS

ADR	NAME	USE
0	BITS	ALL T.S. STATUS BITS
1	NOCHAR	NUMBER OF CHARACTERS PROCESSED
2-3	TOTRT	TOTAL RUN TIME (IN MILLI-SECONDS)
4-5	IOPRT	TOTAL IOP TIME (IN MILLI-SECONDS)
6	BILLTT	START-UP TELETYPE NUMBER
7-10	SUDT	START-UP DATE AND TIME (DATE FIRST)
11-13	PROGNM	NAME OF PROGRAM (3 WORDS INTERNAL CODE)
14	INITIALS	INITIALS OF TYPIST
15	PRIORITY	PRIORITY WORD (TO SPECIFY BACKGROUND PROGRAM)
16-17	DDTSEG	2 DRUM ADDRESSES FOR IDDT
20-27	DDTBKP	10 WORDS OF BREAKPOINT INFORMATION FOR IDDT
30	PROGAD	ADDRESS OF PROGRAM IN START-UP TABLE
31	PIDXFL	DRUM ADDRESS OF PROGRAMMERS INDEX FILE
32	RESTPC	RESTART PC
33	HALTCR	HALT COMMUNICATION REGISTER
34	TTNN	TELETYPE NAME NUMBER
35	PC	SWAPPER REGISTER PC
36-37		SAVED FOR IDDT
40	HH16AC	HALT HORRIBLE AND IDDT AC
41	HH16PC	PC
42	HH16IO	IO
43	HH16FG	FLAGS
44	C16AC	CORE 16 AC
45	C16PC	PC
46	C16IO	IO
47	C16FLA	FLAGS
50	AC	SWAPPER REGISTERS AC
51	IO	IO
52	FLAGS	FLAGS
53	FASTAC	FASTRAND SWAPPING REGISTERS AC
54	FASTPC	PC
55-64	C16TEM	10 WORDS OF COMMON ROUTINES TEM. STORAGE
65-72	DDTS6	6 WORDS OF TEM. STORAGE FOR IDDT

USER-MODIFIED LOWER CORE REGISTERS

73	RESTSU	START-UP LOCATION FOR RESTARTS
74	IOPMAX	I-O PROCESSOR MAXIMUM (INCLUSIVE)
75	TISMAX	MAXIMUM FOR TIS (INCLUSIVE)
76	TTNO	TELETYPE NUMBER (LOWER 6 BITS)
77	TRAPPC	PC FROM IOP OR TT ERROR TRAP
100-101		RESERVED FOR CAL
102-103	ERCODE	2 ERROR CODE WORDS
104	OWNWD	OWN-WORD FOR WRITE ITEMS
105	TTTSU	TELETYPE TRAP START-UP
106	IOPTSU	I-O PROCESSOR TRAP START-UP LOCATION

WORD 0: (BITS)

400000	BIT 0	RUNNING UNDER IDDT
200000	BIT 1	HELD BY XDDT
100000	BIT 2	IN RESTART MODE
40000	BIT 3	RESTART DEBREAK LEGAL
20000	BIT 4	REAL HALT IS LEGAL (SET FOR HALT PROGRAM)
10000	BIT 5	HALTED HORRIBLY (SIGNAL TO HALT PROGRAM)

DICTIONARY OF SYMBOLS IN THE SYSTEM

ADDLP=IOT 1300	Add list pairs for Event Detector, +1 for TOT
BGI=IOT 12200	Initialize item buffer for PUT (see summary)
C16AC=44	Core 16 AC
C16IO=46	Core 16 IO
C16FLA=47	Core 16 Flags
C16RET=IOT 4000	Core 16 Executive IOT for Core 16 only
DDT=IOT 11515	Decode date (FSA), 2 returns, date in AC
DDTGO=IOT 10063	Used by DDT - Start up a program
DELAY=IOT 1600	1 Return, No parameters, +40 delay n seconds, n=(AC)
DGQN=IOT 14370	Down level and generate question number
DGQNT=IOT 14374	Down level and generate question number and type it
DNM=IOT 10600	Decode number, 2 returns, (FSA) number returned in AC
DTM=IOT 11415	Decode time (FSA), 2 returns, Time in AC
DUN=IOT 10714	Decode unit number, (FSA), 2 returns, AC + IO hold unit number on return 2
EAB=IOT 6240	Expunge block, IO=DRA
EAI=IOT 6200	Expunge item, AC=Buffer containing item, IO=DRA
EDIT=IOT 10100	Edit input string, AC = Character Pointer, 2 returns
EPM=IOT 5600	Expunge parameters
ERCODE=102	First of 2 error words
ERIM=IOT 10300	Enter simulated READIN MODE
ETQ=IOT 5440	Expunge quantity, AC is ptr to params, IO quantity to expunge
FAC=123	First word of Floating Accumulator
FADD=IOT 12700	FADD X (Macro)
FAIL=IOT 14570	Kill current question and type FIX
FDAC=IOT 13300	FDAC X (Macro)
FDIV=IOT 13100	FDIV X (Macro), R1 is error return
FIX=IOT 13600	FIX N (Macro) Single Precision
FIX2=IOT 13700	FIX2 N (Macro) Double Precision
FLAC=IOT 13200	FLAC N (Macro)
FLIP=IOT 12600	Floating input, R1 error
FLOAT=IOT 13400	FLOAT N (Macro) Single Precision
FLOAT2=IOT 13500	FLOAT2 N (Macro) Double Precision
FLOP=IOT 14000	FLOP (Macro) flags, acc, dig, col
FMUL=IOT 13000	FMUL X (Macro)
FOV=125	Start up location for F. P. overflow and underflow
FSA=107	Text pointer
FSUB=IOT 12740	FSUB X (Macro)
GET=IOT 12300	See summary
GETLP=IOT 1400	+0 = Get list pairs EVD, +1 = TOT

GETPF=IOT 1740	Get pending file # N
	R1 = not yours
	R2 = yours

GPUN=IOT 4400	Get punch, 2 returns
GQN=IOT 14310	Generate question number, word following IOT is PTR to a) Question short form b) Question long form c) Explanation of answer
GQNT=IOT 14314	Generate question number and type
GRDR=IOT 4100	Get reader, 2 returns
GTD=IOT 3400	GTD Get time (IO) and date (AC) of start up GTD+1 Actual
GTEXT=IOT 14600	Get text from drum, call with PTR in AC (FSA)
GTY=IOT 600	Get Teletype, (TTNO) 2 returns = always get 2nd

14 Sept. 1966

```
HALT=IOT 12401      Types time and executes a HLT
HALTCR=33           Communication to halt program
INIT=IOT 14200      Marks beginning of Job Hunter Program (JBUFP)
INITIA=14           Initials of Typist
IOPMAX=74           IOP maximum (inclusive)
IOPTSU=106          IOP trap start up location
ITQ=IOT 5400        Index quantity; AC is ptr to params, IO quantity
                    to index
IVNE=IOT6660        Invariant number expunge
                    C(AC) = PTR to 3 word block
                    C(IO) = Value to store
                    Block:  Invariant number
                            own-word
                            rewrite number
IVNR=IOT 6500       Invariant number read; AC=Core Adr, IO=Number
IVNW=IOT 6600       Invariant number write
IVNRW=IOT 6640      Invariant number rewrite
JANS=162            Job Hunter; register containing core ptr to answer
JBF=161             Job Hunter; brief mode flag (0,1,2, or 3)
JBUFP=150           Job Hunter; 6 register parameters area for drum
                    buffering
JDANS=163           Job Hunter; Drum PTR to answer
JDTR=164            Job Hunter drum relocation factor (used by
                    Job Hunter)
JHTSU=201           Job Hunter; transfers here if user types ^.
JMODE=111           Job Hunter Mode Flag (-1,0,1,2, or 3)
JORG=200            Job Hunter; Pointer to location after INIT
JQN=157             Job Hunter; 2 register display of current question
                    number
KILL=IOT 14520      Kill question whose number follows the IOT
                    (2 registers for number)
MCNT=IOT 7000       Get tape unit; IO = Reel #
MERS=IOT 7500       Erase tape; AC = # of inches/5;  IO = unit
MPM=IOT 5700        Swap parameters
MREW=IOT 7100       Rewind; IO = Unit# ; +40 rewind with interlock
MRPR=IOT 7400       Mag tape read; AC = CORE ADR, IO = Unit,
                    word after IOT=length
MSPC=IOT 7200       Space N blocks forward AC = N, IO = Unit
                    +40 = backward
MWEOF=IOT 7300      Write end of file:  IO = Unit
MWPR=IOT 7420       Mag tape write:  AC = CORE ADR, IO = Unit
                    C (word after IOT) = length
ORG=210             First free location
OWNWD=104           Password for write and rewrite items
PEEK=IOT 3700       Read location, address in AC; return contents
                    in AC
PIDXFL=31           Drum address of programmer's index file
PPA=IOT 4500        Punch alpha character from IO bits 0-7; 1 return
PPB=IOT 10400       Punch binary word from IO (paper tape)
PRIORI=15           Priority word (to specify background program)
PSU=IOT 1500        Program startup - a running program may start up
                    another program
PUT=IOT 12340       See summary
```

14 Sept. 1966

R/B=IOT 6140 Read block: AC = Buffer, IO = DRA

R/I=IOT 6100 Read item: AC = Buffer, IO = DRA

RCK=IOT 4700 Read Millisecond clock into the AC

RMASK=IOT 1456Ø Job Hunter; Kill current question (don't type FIX)

RNCONS=IOT 1454Ø Reconsider question:
+Ø get question number from 2 words following IOT
+10 get number from AC + IO

RSTAR=IOT 145ØØ Generalized Kill and Restart IOT
+20 Kill a question
640 Reconsider a question
+60 Kill current question
+70 Kill current question and type FIX

RSTPC=32 Restart PC

RSTSU=73 Start up location for restarts

RLSPF=IOT 17ØØ Release pending file #N (in AC)

RPA=IOT 4200 Read paper tape alpha into IO bits Ø-7; 1 return

RPB=IOT 1Ø2ØØ Read binary word (paper tape); return IO

RTY=IOT 700 Release Teletype (TTNO); 2 returns

RPM=IOT 5500 Read parameters

RPUN=IOT 4600 Release punch; 1 return

RQTRK=IOT 6000 Release quarter-track; IO is any DRA in quarter-track

RRDR=IOT 4300 Release paper tape reader; 1 return

RSMC=IOT 1200 RSMC leave RESTART mode
RSMC+1 enter RESTART mode
RSMC+2 debreak and leave RESTART mode
RSMC+3 debreak and enter RESTART mode

SGB=IOT 534Ø Scatter Gather block; item

SGI=IOT 5300 Ø - read
Ø1 - skip
Ø2 - continue on compare
Ø3 - continue on non-compare
Ø4 - write
Ø5 - zero
77 - term

SICKTT=IOT 15Ø64	Job Hunter EOT - Hang Break - Types out "interrupted" and executes a Restar+64
SNM=IOT 11ØØØ	Sets up number, convert number in AC to internal (STS)
SOT=IOT 52ØØ	Type on SOROBAN. AC contains byte pointer to concise text string terminated by 56 code. SOT+20: in addition IO is typed out as octal number
SPM=IOT 56ØØ	Set a register in EVD's class + item type table; 2 returns
STD=IOT 113ØØ	Set up time and date (STS)
STS=11Ø	Storing text pointer
STV=IOT 117ØØ	Call Syntax Verifier
STVL=IOT 121ØØ	Return to STV in lose mode
STVW=IOT 12ØØØ	Return to STV in win mode
SUA=35	STV interpretive PC
SUDT=7	Start up date
SUE=137	Syntax Verifier - push down list Maximum+1
SUG=143	STV output pointer
SUN=IOT 1117Ø	Set up unit number = convert unit number in AC and IO to internal (STS)
SUP=143	Syntax Verifier - beginning pointer for pushdown list
SUPGO=IOT 141ØØ	Complete core seg (36-7777)
SUS=IOT 15Ø	STV ignore register non-general
SUTM=1Ø	Start up time
SUQ=146	STV resets output pointer to C(SUG)
SUR=145	Syntax Verifier -relocation register
SUS=146	Syntax Verifier - 15 bit atom table definition
TDNUM=IOT 112ØØ	Break down time and date
TIS=IOT 4ØØ	Typeout string; character PTR in AC
TISMAX=75	TIS maximum (inclusive)
TOS=IOT 5ØØ	Type in string; buffer PTR in AC
TRAPPC=77	Address where IOT error occurred
TTCKS=IOT 1ØØØ	Teletype check status R1, with AC = C(ERCODE) if error

TTMODE=IOT 1ØØØ	TTMODE+2Ø - enter control mode TTMODE - leave control mode TTMODE+6Ø - enter 8-bit mode TTMODE+4Ø - leave 8-bit mode
TTNN=34	Teletype name number
TTNO=76	Teletype number
TTON=IOT 1Ø5ØØ	Turn on a Teletype (TTNO) 2 returns
TTTSU=1Ø5	Teletype trap startup location
TYI=IOT 2ØØ	Input Teletype character into bits Ø-5 of the AC
TYIHNG=1ØØ	Hang until interrupted by null
TYIN=IOT 144ØØ	No argument; waits for type in
TYIV=IOT 144Ø2	Type in and verify; word after IOT contains PTR to the definition
TYO=IOT 3ØØ	Output Teletype character from bits Ø-5 of the AC
TYOC=IOT 147ØØ	Type out a question: the argument follows the IOT
TYOQ=IOT 143Ø1	Type out the previously generated question
UGQN=IOT 1633Ø	Up level and generate question number
UGQNT=IOT 14334	Up level and generate question number and type it
WAB=IOT 644Ø	Write block addressed; AC=CORE ADR, IO = DRA
WAI=IOT 64ØØ	Write item addressed; AC=CORE ADR, IO = DRA
WAIA=IOT 641Ø	Write item addressed in APR; like WAI but verifies chaining
WNBF=IOT 634Ø	Write block non-addressed free, AC=CORE ADR
WNBH=IOT 636Ø	Write block non-addressed held, AC=CORE ADR
WNIA=IOT 67ØØ	Write item non-addressed on the APR
WNIF=IOT 63ØØ	Write item non-addressed free, AC=CORE ADR
WNIH=IOT 632Ø	Write item non-addressed held, AC=CORE ADR
WPID=IOT 674Ø	Write patient identifier item
WPP=IOT 32ØØ	Write p-pointer; AC is address; IO quantity
XDGQN=IOT 1435Ø	Index, down level, generate question number
XDGQNT=IOT 14354	Index, down level, generate question number and type it

PROGRAMMING SYSTEM

1) Programmer's Index (PIDXFL)

address of next block (Ø if last block)	
	word count
	type
	name
	name
8 words	version
per file	date
	time
	DRA
	garbage

type
- Ø binary
- 1 relocatable binary
- 2 core image
- 3 english
- 4 symbols
- 5 symbols
- 6 teco

2) Handle (SA 2ØØ)

programmer's index commands (terminated with alt mode)

- add name,ver,type
- b (list with addresses)
- delete name,ver,type
- edit (EOM or KEOM)
- find name,ver,type
- halt
- index (list whole MPI)
- kill (entire word necessary)
- list
- terminate (connection with this index)

mag tape commands (tape rewinding at end)

- m reel number
- org (release tape)
- print
- read index,vers
- seduce
- write index,vers
- z (rewind to load point)

9 September 1966

3) Midas (SA 1Ø1)

```
1 nam,ver       do pass 1
2 nam,ver       do pass 2
C nam,ver       continue current pass
J               jump block at end
B nam,ver       file binary in index (must be done before S)
S nam,ver       file symbols in index
A               alphabetic symbol print
P               symbol punch on paper tape
H               halt
T nam,ver       load binary symbols
I               initialize symbols
E num           extra features
                 bit 14, define usw's=Ø
                 bit 15, lgl typeout
                 bit 16, print all
```

4) Editor (SA 1ØØ)

Restart at 1ØØ to save buffer, 1Ø1 to kill buffer

```
          "A", LOC"A"                 Append
     LOC"C", LOC;LOC"C"               Change
     LOC"K", LOC;LOC"K"               Delete
          "P", LOC;LOC"P"             Print
Name,vers"S"                          File in index
Name,vers"U", LOC;name,vers"U"        Append from index
          1"H"                        Halt
          "R", LOC"R"                 Append internal paper tape
          "Y", LOC"Y"                 Append concise paper tape
          "W", LOC"W"                 Append old system internal tape
          "T", LOC;LOC"T"             Punch internal paper tape
     NUM"Z"                           Tape feed paper tape
          "V", LOC;LOC"V"             Verify internal paper tape
          "X"                         Seg to lister

       LOC/                           Open line
     RUBOUT                           Delete open line
          .                           Value of current line
          ↑                           Open previous line
     RETURN                           Open next line
          @                           Value of last line
       LOC=, LOC;LOC=                 Number of lines (octal)
```

5) Lister

E	Seg to editor
R RE	Append internal paper tape
Y YE	Append concise paper tape
FNUMBER	Append saved file
W "W" W2,7,6-@	Write (print) pages
K K3,9,14-17	Delete pages
Halt ←-	Halt
@	Number of pages
C5Ø	Wait at beginning of page
B1Ø	Ring bell
S72	Width
L2	Space between lines
T1Ø,14,12	Tabs
167,67,66	Page length
M6Ø	Lines/page
#L #P #L,P #	Number pages, number lines
P "P"	Proceed
O	Octal
D	Decimal

6) DDT

CONTROL CHARACTERS:

"B"	Flush all breakpoints
ARG;ARG"B"	Set specified breakpoint
ARG"B"	Set next free breakpoint
ARG"C"	Change cores
"D"	Turn off TT and hang
ARG"E"	Effective address search (M#,M#+1,M#+2)
"F"	See next page
ARG"G"	Set PC, go
1"H"	Halt
"K"	Kill all but initial symbols
1"L"	Punch, see next page
ARG"N"	Not word search (M#,M#+1,M#+2)
"O"	Type "Q" and next location as time and date
ARG"P"	Multiple proceeds
"P"	Proceed
"Q"	Value of last quantity typed
ARG"R"	Set radix of numeric typeout
"S"	Type as squoze
-ARG"T"	Clear buffer; read symbol tape
ARG"T"	Read symbol table image from drum (F#)
"U"	Logical OR
"V"	Verify core against binary tape image (M#+1,M#+2,F#)
1"V"	Clear buffer; verify core against paper tape (M#+1, M#+2)
ARG"W"	Word search (M#,M#+1,M#+2)
ARG"X"	Execute ARG
1"Y"	Clear buffer; yank binary tape (M#+1,M#+2)
"Y"	Read binary tape image from drum (M#+1,M#+2,F#)
ARG;ARG"Z"	Zero core (G#)

FILE HANDLING:

C"P"	Get startup program
F"P"	Find file (F#-2 to F#+2)
S"F"	Save core image on held information
U"F"	Unsave core image

PUNCHING:

1"L"	Punch loader
-Ø"L"	Punch loader; set to punch only non-zero core (G#)
ARG;ARG%	Punch data
*	Punch register
ARG@	Punch jump block

TYPEOUT MODES:

A!	Numeric location
R!	Symbolic location
F!	Floating contents
C!	Numeric contents
I!	Internal code contents
S!	Symbolic contents

SPECIAL REGISTER EXAMINATION:

B#/	Location of breakpoint Bn#/ loc of breakpoint n,n=Ø-3
C#/	Which core image is being examined
E#/	Floating format word
T#/	Bottom of symbol table
G#/	Number which is generated all over core ("Z")
M#/	Mask, lower limit, upper limit
F#-2/	Mask for file operations
F#-1/	Type
F#/	Name, left
F#+1/	Name, right
F#+2/	Version
F#+3/	Date
F#+4/	Time
F#+5/	Address

OTHER CHARACTERS:

ARG/		Open register
ARG:		Open register, 16-bit address
ARG[	(K)	Open register, force numeric typeout
ARG]	(M)	Open register, force symbolic typeout
=		Numeric typeout of "Q"
~		Full symbolic typeout of "Q"
?		Internal code typeout of "Q"
ARG"		Take ARG as internal code
LINEFEED		Close register
RETURN		Close register, open next
ALT MODE		Close register, open register addressed by "Q"
>		Close register, open register addressed by "Q", same sequence
↑		Close register, open previous
&		Logical and
.		Current location, take number as decimal, symbol constituent
'		Open in type in mode
$		Print as floating point

PUT, GET and BGI Summary

BGI -

AC = Map address
IO = Item address

Creates an "empty" item starting at location in AC, as described by map.

PUT -

AC = pointer to parameters block.
Parameters:
Word 0: Information address (Byte pointer for variable information, full word for fixed length).
Word 1: Field No. (address relative to beginning of map).
Word 2: Item address.
Word 3: Map address.
Word 4: Address of subscript list (1 word for each level in secondary information tree).

GET -

AC = Pointer to parameters block
Parameters:
Word 0: Address of first 3 error locations
Word 1-4: Same as PUT
Errors:
Word 0: Address of field number too high error routine
Word 1: Address of subscript too high error routine
Word 2: Address of attempt to retrieve pointer error routine
Returns:
AC = pointer to information sought (byte or bit pointer).
IO = 3rd word of field map
Sign = 1 if fixed length
0 if variable
Bits 1-8 - transform code
Bits 9-12 - length in words (for fixed length)
Bits-17 - residual bit length

MIDAS ERROR COMMENTS

Error	Cause	Effect
	Undefined Symbols	
usα	Undefined symbol (α indicates where found):	set symbol=Ø (TW option)
c	in a constant	
d	in size of dimension array	
f	in offset count	
i	in argument of Øif or 1if	
l	in a location assignment	
m	in storage word generated by a macro call	
o	in argument of equals or opsyn	
p	in a parameter assignment	
r	in the count of a repeat	
s	in the argument of a start	
t	in a multi-syllabic address tag	
w	in a storage word	
uwd	undefined symbol in argument of a pseudo-instruction	

Error	Cause	Effect
	Relocation Errors	
irα	Illegal relocation; see undefined symbol section for meaning of α	Relocation

Error	Cause	Effect
	Multiple Definitions	
mdt	Multiply defined tag	no redefinition
mdv	Multiply defined variable	no redefinition
mdd	Multiply defined dimension	no redefinition

Error	Cause	Effect
	Other Errors	
mnd	Macro name disagrees. (define-terminate)	First name used.
ich	Illegal character	The character is ignored
ilf	Illegal format	Characters ignored until next tab or carriage return.
ipa	Improper parameter assignment.	The assignment is ignored.
vld	Variables location disagrees on different passes.	Condition ignored.
lgi	Location gone indefinite	If bit 15 set, this error message printed on Pass 1.
irx	Illegal operation with relocatable symbols.	
pnt	Not an error. Result of print pseudo-instruction.	
eof	No start at end of English	

In the event of the following error conditions, assembly cannot continue.

cld	Constants location disagrees on different passes.
tmc	Too many constants
tmp	Too many parameters. (macros)
tmv	Too many variables.
sce	Storage capacity exceeded
iae	Call Nancy

FUNCTION BITS

	-WORD 1-
	HOSPITAL PROGRAMS
1	RESEARCH CYCLE
2	SAVED JOSS PROGRAMS
4	X-MODE (NON-RESEARCH-CYCLE)
10	Y-MODE
20	CENSUS OPERATIONS
40	INTERNAL LAB OPERATIONS
100	NURSING STATION - FLOOR - OPERATIONAL
400000	HOSPITAL EXECUTIVE PROGRAMS (POSSIBLY BEDDES, MEDFML, ETC.)
	-WORD 2-
1	BASIC PROGRAMS (HARMLESS - FOR SCHOOLS, ETC.) (UTTEST, UDOWN, TELCOMP, TELHOW, ABACUS, TTT, ...)
2	UTILITY PROGRAMS (PEEK, UTTP, ULIB, ?, STAT, SNAPSHOT, TELBRIEF, ...)
4	PROGRAMMING SYSTEM (DDT, ETC.)
10	M.I.T.D. DEMONSTRATION PROGRAMS (DEM...)
*20	HOSPITAL PROGRAMS (CALLABLE BY PROGRAMMERS)
40	RESTRICTED UTILITY PROGRAMS (PFC, UTTC, SFC, SDT, ...)
400000	COMPUTER ROOM ONLY (SYSDOWN, SYSSTAT, ...)
-SPECIAL-	
400	NON-HCP TELETYPE
1000	PROGRAM CALLABLE UNDER DDT BY NON-HCP TELETYPE

* THIS BIT IS SET IN ADDITION TO A BIT IN WORD 1

12 September 1966

HOLLERITH CODE (MAG TAPE)

A	61
B	62
C	63
D	64
E	65
F	66
G	67
H	7Ø
I	71
J	41
K	42
L	43
M	44
N	45
O	46
P	47
Q	5Ø
R	51
S	22
T	23
U	24
V	25
W	26
X	27
Y	3Ø
Z	31
Ø	12
1	Ø1
2	Ø2
3	Ø3
4	Ø4
5	Ø5
6	Ø6
7	Ø7
8	1Ø
9	11
[&] +	6Ø
-	4Ø
/	21
[#] =	13
$	33
.	53
[@] '	73
[%] (	14
*	34
[□])	54
blank	74
	ØØ

TELETYPE CODE CONVERSION FOR THE NEW SYSTEM

INTERNAL	ASCII	CHARACTER
00	040	SPACE
01	041	!
02	042	"
03	043	#
04	044	$
05	045	%
06	046	& ^
07	047	'
10	050	(
11	051	)
12	052	*
13	053	+
14	054	,
15	055	-
16	056	.
17	057	/
20	060	0 (ZERO)
21	061	1
22	062	2
23	063	3
24	064	4
25	065	5
26	066	6
27	067	7
30	070	8
31	071	9
32	072	:
33	073	;
34	074	<
35	075	=
36	076	>
37	077	?

INTERNAL	ASCII	CHARACTER
40	100	@ ∨
41	101	A
42	102	B
43	103	C
44	104	D
45	105	E
46	106	F
47	107	G
50	110	H
51	111	I
52	112	J
53	113	K
54	114	L
55	115	M
56	116	N
57	117	O (OH)
60	120	P
61	121	Q
62	122	R
63	123	S
64	124	T
65	125	U
66	126	V
67	127	W
70	130	X
71	131	Y
72	132	Z
73	133	[
74	175,176, 033	EOM
75	135	]
76	015-012	CARRIAGE RETURN-LINE FEED
77		WARNING

INTERNAL	ASCII	CHARACTER
7700	000	NULL OR BREAK OR "@ "
7701	001	"A"
7702	002	"B"
7703	003	"C"
7704	004	EOT
7705	005	"E" OR WRU
7706	006	"F" OR RU
7707	007	"G" OR BELL
7710	010	"H"
7711	011	TAB
7712	012	LINE FEED
7713	013	"K" OR VT
7714	014	"L" OR FORM FEED
7715	015	CARRIAGE RETURN (OUTPUT ONLY)
7716	016	"N"
7717	017	"O"
7720	020	"P"
7721	021	"Q"
7722	022	"R" OR TAPE
7723	023	"S" OR RDR OFF
7724	024	"T"
7725	025	"U"
7726	026	"V"
7727	027	"W"
7730	030	"X"
7731	031	"Y"
7732	032	"Z"
7733	033	"["
7734	034	SHIFT "L"
7735	035	"]"
7736	036	"↑"
7737	037	"←"
7740-7743	UNUSED	
7744	134	BACKSLASH
7745	UNUSED	
7746	136	↑
7747	137	←
7750-7773	UNUSED	
7774	177	RUB OUT
7775-7777	UNUSED	

CONCISE CODE

Lower + Upper Case	
A	61
B	62
C	63
D	64
E	65
F	66
G	67
H	70
I	71
J	41
K	42
L	43
M	44
N	45
O	46
P	47
Q	50
R	51
S	22
T	23
U	24
V	25
W	26
X	27
Y	30
Z	31

Lower Case	Upper Case	
0	→	20
1	"	01
2	'	02
3	~	03
4	⊃	04
5	∨	05
6	∧	06
7	<	07
8	>	10
9	↑	11
(	[	57
)	]	55
_	—	56
=	+	54
.	.	40
,	=	33
/	x	73
	?	21

Lower Case	72
Upper Case	74
Space	0
Backspace	75
Tab	36
Carriage Return	77
Red	35
Black	34
Stop Code	13

HOSPITAL COMPUTER INSTRUCTIONS

See "Revised New Instructions", EDG, 12-8-64 and "PDP-1 Manual", Digital Equipment Corp., 1962 for a detailed explanation of the instructions.

BASIC INSTRUCTIONS

add Y	40	Add C(Y) to C(AC)
and Y	02	Logical AND of C(Y) with C(AC)
cal	16	Equals jda 100.
dac Y	24	Deposit C(AC) in Y
dap Y	26	Deposit contents of address part of AC in Y
dch Y	14	Deposit C(AC) according to byte pointer in Y and rotate AC left 6 bits. If defer bit set, index the byte pointer in Y before depositing character.
dio Y	32	Deposit C(IO) in Y
dip Y	30	Deposit contents of instruction part of AC in Y
div Y	56	Divide C(AC,IO) by C(Y). Quotient in AC, remainder in IO. The next instruction is skipped unless the result overflows the AC.
dzm Y	34	Make C(Y) zero
idx Y	44	Index (add one to) C(Y). Leave in Y and AC.
ior Y	04	Inclusive OR of C(Y) with C(AC)
iot	72	See In-Out Transfer Group
isp Y	46	Index and skip if result is positive; see idx
jda Y	17	Equals dac Y plus jsp Y + 1
jmp Y	60	Take next instruction from Y
jsp Y	62	Jump to Y and save extended PC in AC.
lac Y	20	Load AC with C(Y)
law N	70	Load AC with the number N. If the defer bit is set, the resulting op code (71) will load AC with -N.
lch Y	12	Load AC according to byte pointer in Y and clear AC . If defer bit set, index the byte pointer in Y before loading character.
lio Y	22	Load IO with C(Y)
mul Y	54	Multiply C(AC) by C(Y). Result in AC, IO; sign in AC and IO .
opr	76	See Operate Group
sad Y	50	Skip 1 instruction if C(AC) differs from C(Y)
sas Y	52	Skip 1 instruction if C(AC) is same as C(Y)
shift	66	See Shift Group
skp	64	See Skip Group
spo	74	See Special Operate Group
sub Y	42	Subtract C(Y) from C(AC)
tad Y	36	Two's complement add C(Y) to C(AC)
xct Y	10	Perform instruction in Y
xor Y	06	Exclusive OR of C(AC) with C(Y)

OPERATE GROUP

Two or more instructions may be combined by "or-ing" their addresses. The order in which the instruction is executed is:

1. cli cla
2. clf stf lap lat
3. cmi cma hlt
4. lai lia swp

cla	760200	Clear AC
clc	761200	Clear and complement AC (claVcma)
clf	76000f	Clear selected program flag (f=flag;7=all)
cli	764000	Clear IO
cma	761000	Complement AC
cmi	770000	Complement IO
hlt	760400	Halt
lai	760040	Load AC from IO
lap	760300	Load AC from extended PC
lat	762200	Load AC from test word switches
lia	760020	Load IO from AC
nop	760000	No operation
stf	76001f	Set selected program flag (f=flag;7=all)
swp	760060	Swap AC and IO (laiVlia)

SPECIAL OPERATE GROUP

Two or more instructions may be combined by "or-ing" their addresses. See "Revised New Instructions" for exact order of execution.

aai	740003	Add AC and IO (=0 not changed to 0)
cll	740010	Clear link
cml	740004	Complement link
iai	740002	Inclusive OR to AC from IO
ida	740400	Index AC (=0 not changed to 0)
idc	741000	Index byte pointer in AC
ifi	742000	Inclusive OR to flags from IO (link,ring)
iif	744000	Inclusive OR to IO from flags (link,ring)
lfi	742050	Load flags from IO (scfVcllVifi)
lif	744100	Load IO from flags (sciViif)
nai	750000	And the AC and the IO, result in the AC
scf	740040	Special clear flags (ring,not link)
sci	740100	Special clear IO
scm	740200	Special complement AC (=0 not changed to 0)
szl	740020	Skip on ZERO link
xai	740001	Exclusive OR to AC from IO

SHIFT - ROTATE GROUP

Shift is an arithmetic operation. The sign bit is left unchanged and vacated bits are filled with the sign. Rotate is a logical operation and cycles the bits (including sign) in a closed ring. The number of steps is the number of ONE's in bits 9-17 of the instruction (9 max).

ral	661	Rotate AC left
rar	671	Rotate AC right
rcl	663	Rotate combined AC and IO left
rcr	673	Rotate combined AC and IO right
ril	662	Rotate IO left
rir	672	Rotate IO right
sal	665	Shift AC left
sar	675	Shift AC right
scl	667	Shift combined AC and IO left
scr	677	Shift combined AC and IO right
sil	666	Shift IO left
sir	676	Shift IO right

SKIP GROUP

Two or more instructions may be combined by "or-ing" their addresses. The intent of any single or combined skip group instructions may be reversed by making bit 5 equal to 1.

clo	651600	Clear overflow (spaVsmaVszoVi)
sma	640400	Skip on minus AC
sni	644000	Skip on NON-ZERO IO
spa	640200	Skip on plus AC
spi	642000	Skip on plus IO
spq	650500	Skip on plus quanity (szaVsmaVi)
sza	640100	Skip on ZERO AC
szf	64000f	Skip on ZERO flag (f=flag; 7=all flags)
szm	640500	Skip on ZERO or minus AC (szaVsma)
szo	641000	Skip on ZERO overflow (and clear overflow. Only add or sub set it)
szs	6400s0	Skip on ZERO sense switch (s=switch; 7=all)