As you are now the owner of this document which should have come to you for free, please consider making a donation of £1 or more for the upkeep of the (Radar) website which holds this document. I give my time for free, but it costs me money to bring this document to you. You can donate here https://blunham.com/Misc/Texas

Many thanks.

Please do not upload this copyright pdf document to any other website. Breach of copyright may result in a criminal conviction.

This Acrobat document was generated by me, Colin Hinson, from a document held by me. I requested permission to publish this from Texas Instruments (twice) but received no reply. It is presented here (for free) and this pdf version of the document is my copyright in much the same way as a photograph would be. If you believe the document to be under other copyright, please contact me.

The document should have been downloaded from my website https://blunham.com/, or any mirror site named on that site. If you downloaded it from elsewhere, please let me know (particularly if you were charged for it). You can contact me via my Genuki email page: https://www.genuki.org.uk/big/eng/YKS/various?recipient=colin

You may not copy the file for onward transmission of the data nor attempt to make monetary gain by the use of these files. If you want someone else to have a copy of the file, point them at the website. (https://blunham.com/Misc/Texas). **Please do not point them at the file itself as it may move or the site may be updated.**

It should be noted that most of the pages are identifiable as having been processed by me.

I put a lot of time into producing these files which is why you are met with this page when you open the file.

If you find missing pages, pages in the wrong order, anything else wrong with the file or simply want to make a comment, please drop me a line (see above).

It is my hope that you find the file of use to you.

Colin Hinson
In the village of Blunham, Bedfordshire.

COMMAND LIST

```
TASK DBASIC19
NOSYMT
NOTGLOBAL
INCLUDE DBASIC.REL5.BASIC.OBJECT.START
INCLUDE DBASIC.REL4.BASIC.OBJECT.ERRECOVY
INCLUDE DBASIC.REL5.BASIC.OBJECT.SFSN
INCLUDE DBASIC.REL5.BASIC.OBJECT.JMP
INCLUDE DBASIC.REL5.BASIC.OBJECT.READ
INCLUDE DBASIC.REL5.BASIC.OBJECT.GOSUB
INCLUDE DBASIC.REL4.BASIC.OBJECT.RUN
INCLUDE DBASIC.REL5.BASIC.OBJECT.ON
INCLUDE DBASIC.REL5.BASIC.OBJECT.LET
INCLUDE DBASIC.REL5.BASIC.OBJECT.IF
INCLUDE DBASIC.REL4.BASIC.OBJECT.PRINT
INCLUDE DBASIC.REL5.BASIC.OBJECT.MOVE
INCLUDE DBASIC.REL5.BASIC.OBJECT.DIM
INCLUDE DBASIC.REL5.BASIC.OBJECT.FOR
INCLUDE DBASIC.REL5.BASIC.OBJECT.DEF
INCLUDE DBASIC.REL5.BASIC.OBJECT.UNIT
INCLUDE DBASIC.REL5.BASIC.OBJECT.RANDOM
INCLUDE DBASIC.REL0.BASIC.OBJECT.ERROR
INCLUDE DBASIC.REL0.BASIC.OBJECT.ESCAPE
INCLUDE DBASIC.REL0.BASIC.OBJECT.CALL
INCLUDE DBASIC.REL5.BASIC.OBJECT.BASE
INCLUDE DBASIC.REL3.BASIC.OBJECT.TIME
INCLUDE DBASIC.REL5.BASIC.OBJECT.EVAL
INCLUDE DBASIC.REL5.BASIC.OBJECT.IPCOM
INCLUDE DBASIC.REL5.BASIC.OBJECT.POWF
INCLUDE DBASIC.REL5.BASIC.OBJECT.LOGICOPS
INCLUDE DBASIC.REL5.BASIC.OBJECT.RELATOPS
INCLUDE DBASIC.REL5.BASIC.OBJECT.GETP2
INCLUDE DBASIC.REL5.BASIC.OBJECT.CVGC
INCLUDE DBASIC.REL3.BASIC.OBJECT.INTR
INCLUDE DBASIC.REL5.BASIC.OBJECT.FP
INCLUDE DBASIC.REL5.BASIC.OBJECT.RNDF
INCLUDE DBASIC.REL0.BASIC.OBJECT.CVBD
INCLUDE DBASIC.REL0.BASIC.OBJECT.FUNC
INCLUDE DBASIC.REL5.BASIC.OBJECT.FIX
INCLUDE DBASIC.REL5.BASIC.OBJECT.POLY
INCLUDE DBASIC.REL5.BASIC.OBJECT.LOGF
INCLUDE DBASIC.REL5.BASIC.OBJECT.EXPF
INCLUDE DBASIC.REL5.BASIC.OBJECT.SINF
INCLUDE DBASIC.REL5.BASIC.OBJECT.SQRF
INCLUDE DBASIC.REL4.BASIC.OBJECT.ATNF
INCLUDE DBASIC.REL5.BASIC.OBJECT.TICF
INCLUDE DBASIC.REL5.BASIC.OBJECT.MEMF
INCLUDE DBASIC.REL5.BASIC.OBJECT.BITF
INCLUDE DBASIC.REL5.BASIC.OBJECT.CRUF
INCLUDE DBASIC.REL5.BASIC.OBJECT.GETPARM
INCLUDE DBASIC.REL5.BASIC.OBJECT.CHRF
INCLUDE DBASIC.REL5.BASIC.OBJECT.PUTB
INCLUDE DBASIC.REL4.BASIC.OBJECT.GETLINE
INCLUDE DBASIC.REL4.BASIC.OBJECT.EDIT
INCLUDE DBASIC.REL4.BASIC.OBJECT.SIZE
INCLUDE DBASIC.REL5.BASIC.OBJECT.INPUT
INCLUDE DBASIC.REL5.BASIC.OBJECT.CVDB
INCLUDE DBASIC.REL5.BASIC.OBJECT.MWDF
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.MINUS1
END
```

COMMAND LIST

```
TASK DBASIC19
NOSYMT
NOTGLOBAL
INCLUDE DBASIC.REL5.BASIC.OBJECT.START
INCLUDE DBASIC.REL4.BASIC.OBJECT.ERRECOVY
INCLUDE DBASIC.REL5.BASIC.OBJECT.SFSN
INCLUDE DBASIC.REL5.BASIC.OBJECT.JMP
INCLUDE DBASIC.REL5.BASIC.OBJECT.READ
INCLUDE DBASIC.REL5.BASIC.OBJECT.GOSUB
INCLUDE DBASIC.REL4.BASIC.OBJECT.RUN
INCLUDE DBASIC.REL5.BASIC.OBJECT.ON
INCLUDE DBASIC.REL5.BASIC.OBJECT.LET
INCLUDE DBASIC.REL5.BASIC.OBJECT.IF
INCLUDE DBASIC.REL4.BASIC.OBJECT.PRINT
INCLUDE DBASIC.REL5.BASIC.OBJECT.MOVE
INCLUDE DBASIC.REL5.BASIC.OBJECT.DIM
INCLUDE DBASIC.REL5.BASIC.OBJECT.FOR
INCLUDE DBASIC.REL5.BASIC.OBJECT.DEF
INCLUDE DBASIC.REL5.BASIC.OBJECT.UNIT
INCLUDE DBASIC.REL5.BASIC.OBJECT.RANDOM
INCLUDE DBASIC.REL0.BASIC.OBJECT.ERROR
INCLUDE DBASIC.REL0.BASIC.OBJECT.ESCAPE
INCLUDE DBASIC.REL0.BASIC.OBJECT.CALL
INCLUDE DBASIC.REL5.BASIC.OBJECT.BASE
INCLUDE DBASIC.REL3.BASIC.OBJECT.TIME
INCLUDE DBASIC.REL5.BASIC.OBJECT.EVAL
INCLUDE DBASIC.REL5.BASIC.OBJECT.IPCOM
INCLUDE DBASIC.REL5.BASIC.OBJECT.POWF
INCLUDE DBASIC.REL5.BASIC.OBJECT.LOGICOPS
INCLUDE DBASIC.REL5.BASIC.OBJECT.RELATOPS
INCLUDE DBASIC.REL5.BASIC.OBJECT.GETP2
INCLUDE DBASIC.REL5.BASIC.OBJECT.CVGC
INCLUDE DBASIC.REL3.BASIC.OBJECT.INTR
INCLUDE DBASIC.REL5.BASIC.OBJECT.FP
INCLUDE DBASIC.REL5.BASIC.OBJECT.RNDF
INCLUDE DBASIC.REL0.BASIC.OBJECT.CVBD
INCLUDE DBASIC.REL0.BASIC.OBJECT.FUNC
INCLUDE DBASIC.REL5.BASIC.OBJECT.FIX
INCLUDE DBASIC.REL5.BASIC.OBJECT.POLY
INCLUDE DBASIC.REL5.BASIC.OBJECT.LOGF
INCLUDE DBASIC.REL5.BASIC.OBJECT.EXPF
INCLUDE DBASIC.REL5.BASIC.OBJECT.SINF
INCLUDE DBASIC.REL5.BASIC.OBJECT.SQRF
INCLUDE DBASIC.REL4.BASIC.OBJECT.ATNF
INCLUDE DBASIC.REL5.BASIC.OBJECT.TICF
INCLUDE DBASIC.REL5.BASIC.OBJECT.MEMF
INCLUDE DBASIC.REL5.BASIC.OBJECT.BITF
INCLUDE DBASIC.REL5.BASIC.OBJECT.CRUF
INCLUDE DBASIC.REL5.BASIC.OBJECT.GETPARM
INCLUDE DBASIC.REL5.BASIC.OBJECT.CHRF
INCLUDE DBASIC.REL5.BASIC.OBJECT.PUTB
INCLUDE DBASIC.REL4.BASIC.OBJECT.GETLINE
INCLUDE DBASIC.REL4.BASIC.OBJECT.EDIT
INCLUDE DBASIC.REL4.BASIC.OBJECT.SIZE
INCLUDE DBASIC.REL5.BASIC.OBJECT.INPUT
INCLUDE DBASIC.REL5.BASIC.OBJECT.CVDB
INCLUDE DBASIC.REL5.BASIC.OBJECT.MWDF
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.MINUS1
END
```

COMMAND LIST

LINK MAP

CONTROL FILE = BD1.DBASIC.REL5.BASIC.LINK.CNTL

LINKED OUTPUT FILE = BD1.DBASIC.REL5.BASIC.LINK.OBJECT

LIST FILE = BD1.DBASIC.REL5.BASIC.LINK.LIST

OUTPUT FORMAT = ASCII

PHASE 0, DBASIC19 ORIGIN = 0000 LENGTH = 31D0

MODULE	NO	ORIGIN	LENGTH	TYPE	DATE	TIME	CREATOR
START	1	0000	03E2	INCLUDE	05/22/79	14:26:00	SDSMAC
ERRECOVY	2	03E2	00AC	INCLUDE	05/02/79	16:19:51	SDSMAC
SFSN	3	048E	0014	INCLUDE	05/30/79	14:10:22	SDSMAC
JMP	4	04A2	001C	INCLUDE	05/30/79	13:58:04	SDSMAC
READ	5	04BE	00AA	INCLUDE	05/30/79	11:14:28	SDSMAC
GOSUB	6	0568	00A0	INCLUDE	05/30/79	13:52:34	SDSMAC
RUN	7	0608	00F6	INCLUDE	05/02/79	19:54:58	SDSMAC
ON	8	06FE	002C	INCLUDE	05/30/79	14:03:59	SDSMAC
LET	9	072A	0144	INCLUDE	05/30/79	11:59:02	SDSMAC
IF	10	086E	0080	INCLUDE	05/30/79	13:53:40	SDSMAC
PRINT	11	08EE	012A	INCLUDE	05/02/79	14:14:44	SDSMAC
MOVE	12	0A18	0060	INCLUDE	05/30/79	14:03:07	SDSMAC
DIM	13	0A78	0088	INCLUDE	05/30/79	13:43:59	SDSMAC
FOR	14	0B30	0182	INCLUDE	05/30/79	13:46:42	SDSMAC
DEF	15	0CB2	0026	INCLUDE	05/30/79	13:43:15	SDSMAC
UNIT	16	0CD8	0014	INCLUDE	05/30/79	14:14:10	SDSMAC
RANDOM	17	0CEC	0008	INCLUDE	05/30/79	14:07:33	SDSMAC
ERROR	18	0CF4	0008	INCLUDE	08/20/78	17:37:21	SDSMAC
ESCAPE	19	0CFC	0014	INCLUDE	08/20/78	17:37:36	SDSMAC
CALL	20	0D10	0032	INCLUDE	09/26/78	13:37:56	SDSMAC
BASE	21	0D42	0008	INCLUDE	05/30/79	13:38:00	SDSMAC
TIME	22	0D4A	00CA	INCLUDE	03/28/79	18:23:12	SDSMAC
EVAL	23	0E14	0540	INCLUDE	05/30/79	11:00:29	SDSMAC
IPCOM	24	1354	003A	INCLUDE	05/30/79	13:57:01	SDSMAC
POWF	25	138E	0022	INCLUDE	05/30/79	14:05:43	SDSMAC
LOGICOPS	26	13B0	0050	INCLUDE	05/30/79	14:01:01	SDSMAC
RELATOPS	27	1400	003C	INCLUDE	05/30/79	14:08:28	SDSMAC
GETP2	28	143C	001A	INCLUDE	05/30/79	13:50:34	SDSMAC
CVGC	29	1456	005E	INCLUDE	05/30/79	13:42:28	SDSMAC
INTR	30	14B4	006E	INCLUDE	03/28/79	18:24:22	SDSMAC
FP	31	1522	0348	INCLUDE	05/30/79	13:48:02	SDSMAC
RNDF	32	186A	0030	INCLUDE	05/30/79	14:09:29	SDSMAC
CVBD	33	189A	0216	INCLUDE	09/01/78	14:06:31	SDSMAC
FUNC	34	1AB0	0072	INCLUDE	09/26/78	15:39:35	SDSMAC
FIX	35	1B22	003E	INCLUDE	05/30/79	13:45:58	SDSMAC
POLY	36	1B60	0068	INCLUDE	05/30/79	14:04:45	SDSMAC
LOGF	37	1BC8	00EC	INCLUDE	05/30/79	13:59:06	SDSMAC
EXPF	38	1CB4	00C8	INCLUDE	05/30/79	13:44:58	SDSMAC
SINF	39	1D7C	00A0	INCLUDE	05/30/79	14:11:13	SDSMAC
SQRF	40	1E1C	006C	INCLUDE	05/30/79	14:12:17	SDSMAC
ATNF	41	1E88	0008	INCLUDE	05/02/79	00:04:26	SDSMAC
TICF	42	1E90	0030	INCLUDE	05/30/79	14:13:17	SDSMAC
MEMF	43	1EC0	0036	INCLUDE	05/30/79	14:02:13	SDSMAC
BITF	44	1EF6	00S2	INCLUDE	05/30/79	13:38:51	SDSMAC
CRUF	45	1F48	0080	INCLUDE	05/30/79	13:41:04	SDSMAC
GETPARM	46	1FC8	0020	INCLUDE	05/30/79	13:51:39	SDSMAC
CHRF	47	1FE8	003C	INCLUDE	05/30/79	13:39:56	SDSMAC
PUTB	48	2024	004E	INCLUDE	05/30/79	14:06:37	SDSMAC
GETLINE	49	2072	014E	INCLUDE	04/23/79	16:32:33	SDSMAC
EDIT	50	21C0	0AAC	INCLUDE	05/21/79	11:14:57	SDSMAC
SIZE	51	2C6C	0084	INCLUDE	04/30/79	11:44:02	SDSMAC
INPUT	52	2CF0	0104	INCLUDE	05/30/79	13:55:16	SDSMAC
CVDB	53	2DF4	01AE	INCLUDE	05/30/79	11:16:41	SDSMAC

MODULE	NO	ORIGIN	LENGTH	TYPE	DATE	TIME	CREATOR
MWDF	54	2FA2	002E	INCLUDE	05/22/79	14:17:21	SDSMAC
MINUS1	55	2FD0	0200	INCLUDE	09/18/78	18:04:43	SDSMAC

D E F I N I T I O N S

NAME	VALUE	NO	NAME	VALUE	NO	NAME	VALUE	NO	NAME	VALUE	NO
ABSF	1AEA	34	ADDF	1256	23	AINC	001C*	1	ANDF	1400	27
ASCF	1EF2	43	ATNF	1E88	41	ATNFE	3022*	1	B01	1201	23
B05	1205	23	B20	06FA	7	B3F	06F8	7	B4A	1FE6	46
BASY	0D42	21	BAUD	0134	1	BCRU	0046*	1	*BEGN	02D8	1
BITF	1F28	44	BITY	1EF6	44	BMVE	0A18	12	BUS	000A*	1
C1	1200	23	C4	1202	23	C4A00	06F8	7	C6	120A	23
CCNT	0034*	1	CK302	21C0	50	CKEX	0E56	23	CLKADR	FE56*	1
CLKT01	FE5C*	1	CLKT02	FE5E*	1	*CLKWS	FE54*	1	CLLY	0D10	20
CLRV	0608	7	COSF	1D7C	39	CRBF	1FAE	45	CRBY	1F62	45
CRCASS	1700*	1	CRFF	1F80	45	CRFY	1F48	45	CRLF	02EE	1
CRLF1	0302	1	CSCLZ	300E*	1	CSOPZ	3002*	1	CSRDZ	3006*	1
*CSWP	FFBC*	1	CSWTZ	300A*	1	CVBD	18A6	33	CVBFP	3016*	1
CVBFR	1A12	33	CVBI	189E	33	CVC10	147E	29	CVCH	FEEC*	1
CVDB20	2F2A	53	CVDIFZ	2DF8	53	CVDIZ	2DF4	53	CVGCN	1456	29
CVGCN1	145E	29	CVHD	FEF2*	1	CVHD01	FEF3*	1	CVHD15	FF01*	1
CWNDO	FE9A*	1	DATXB	06FD	7	DCNT	0022*	1	DDM	002C*	1
DEFY	0CB2	15	DIMY	0A78	13	DLC	002A*	1	DLIM	0028*	1
DS	FE60*	1	DS1	FE66*	1	*DS2	FF8C*	1	*DS3	FFC2*	1
EBP	FEF2*	1	EDER	218C	49	EDIT	229C	50	EFLG	0036*	1
ELNM	003A*	1	ELSF	003C*	1	ENUM	0038*	1	EORBUS	000C*	1
ERM1	0404	2	ERRLS1	301A*	1	ERRLS2	301E*	1	ERRS	03E2	2
*ERRS0	03EA	2	ERRS2	0486	2	ERRY	0CF4	18	ESCY	0CFC	19
EUS	0002*	1	EVAL	0FDC	23	EVALS2	0FD4	23	EVARZ	0E52	23
EVERZ	0E6C	23	EVFX	0E84	23	EVOP3A	111E	23	EVSDZ	0E14	23
EVSFR	0FB4	23	EVSKB	FEF2*	1	EVSKE	FF94*	1	EXPF	1CB8	38
FAD	153C	31	FADDI	1814	31	FCLR	1744	31	FDD	1778	31
FFLG	003E*	1	FIX	1B22	35	FLDD	152A	31	FLOAT	15F6	31
FMD	1658	31	FNEG	15EC	31	FNRM	161E	31	FNS	0004*	1
*FNSZ	000A*	1	FOR2	0B70	14	FORY	0B30	14	FPAC	FF9C*	1
FPAC2	FF9E*	1	FPAC4	FFA0*	1	FPWP	FF9C*	1	FSCL	183C	31
FSD	1532	31	FSRD	1522	31	FSUBI	1828	31	FUNBK	1B60	36
FUNFX	1B8C	36	FUZZ	3800*	1	GETC	039E	1	GETCR	038C	1
GETP2	143C	28	GOS1	05B4	6	GOS2	05C8	6	GOSB1	0568	6
GOSON	05A8	6	GOSY	0576	6	GOTY	05A8	6	GPRM	1FC8	46
GPRM1	1FCE	46	GPRM2	1FD0	46	GSC	0018*	1	*GSIM	0C8C	14
GSS	0006*	1	*GSSZ	0014*	1	*GTICZ	1E90	42	GTLN	2090	49
HFLG	0040*	1	HOUT	09E0	11	ID302	00A6	1	IFLG	0042*	1
IFY	086E	10	IMKY	14D0	30	INPEND	136A	24	INPF	1ACE	34
INPY	2CF0	52	*INTFLG	FE40*	1	INTSB	FE20*	1	INTSP	FE42*	1
IOB	0000*	1	IRTY	14FA	30	*ISTCK	FE20*	1	ITAB	FE00*	1
JMPR0	04AA	4	JMPR0A	04A2	4	LANDF	13DE	26	*LCN1	2929	50
*LCN2	2848	50	*LCN3	28A8	50	*LCN4	005C*	50	*LDEF	0044*	50
LDP5	228C	50	LDPY	21E6	50	LDSTRT	21F8	50	LEC	0030*	1
LENF	201A	47	LETY	072A	9	LINE	065E	7	LINE0	0672	7
LINE2	068C	7	LINE5	0664	7	LNOTF	13F1	26	LNSZ	0084*	1
LNUM	002E*	1	*LNXT	0010*	50	LOGF	1BCA	37	LORF	13CC	26
LSTL	2A46	50	LXORF	13D4	26	MCHF	2010	47	MEMF	1EDC	43
MEMY	1EC0	43	MODE	0024*	1	MOVEB	2CE6	51	MOVEL	2CE8	51
MWDF	2FBC	54	MWDY	2FA2	54	NB	FE9E*	1	*NEWP	02AA	1

NAME	VALUE	NO	NAME	VALUE	NO	NAME	VALUE	NO	NAME	VALUE	NO
NEWY	02A0	1	NIC	0064*	1	NKYF	1B06	34	NLIN	0658	7
NLIN0	0654	7	NOEY	0D0A	19	NOTF	1431	27	*NRV	0004*	1
NVD	0014*	1	NVS	0016*	1	NXTXB	06FC	7	NXTY	0C2A	14
OCWNDO	FE94*	1	ONY	06FE	8	ORF	141C	27	OUTL1	1354	24
OWNDO	FE96*	1	PFIX	1B4E	35	PLC	0026*	1	PLF	0044*	1
PLYX	1BB2	36	PLYXX	1BA6	36	POPY	05F6	6	POWF	138E	25
*PRDY	02E4	1	PROGM	3012*	1	PRTEND	1362	24	PRTY	08EE	11
PUTB	2024	48	RANDS	FE52*	1	RANDZ	1882	32	RANY	0CEC	17
RDDY	0512	5	REST	061A	7	RNWY	054E	5	RTNY	05D4	6
RUNP	0626	7	RUNP1	062E	7	SAVY	299E	50	SFSN	048E	3
SINF	1D90	39	SIZE	2C6C	51	*SIZE1	2CD2	51	SLN	0032*	1
SLT	000E*	1	*SLTN	0020*	1	SQRF	1E1C	40	*SQRI	0004*	1
SRHF	1FE8	47	SSP	FF76*	1	STPY	0460	2	STPYA	0428	2
STRTC	00A4	1	SUBF	127E	23	SYSF	1AB0	34	TEMP	FF94*	1
TEMP2	FF96*	1	TEMP4	FF98*	1	TEMP6	FF9A*	1	TICF	1EA8	42
TIMY	0D4A	22	TRPY	14B4	30	*TY0	001A*	1	TYP0	030A	1
TYP11	0308	1	TYPB	031C	1	TYPBE	031A	1	TYPC	0314	1
TYPE	031E	1	*UCCNT	137A	24	UFT	0008*	1	UMSK	0003*	1
UNIT	001E*	1	UNTY	0CD8	16	VDT	0012*	1	VNT	0010*	1
*WPR1	FE74*	1	WPR103	FE7A*	1	*WPR104	FE7C*	1	WPR108	FE84*	1
WPR2	FFDC*	1									

**** LINKING COMPLETED
*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.START
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.START
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.START
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```



```
0001                    IDT 'START'
0002            *       22 MAY   1979
0004            ********************************************************
0005            *                                                      *
0006            *       DEVELOPMENT PBASIC/D  TMS9900                  *
0007            *                                                      *
0008            *       COPYRIGHT 1978  TEXAS INSTRUMENTS INC.         *
0009            *                                                      *
0010            ********************************************************
0011            *
0012                    DEF NRV,LNSZ,NIC,UMSK
0013                    DEF FUZZ,FNSZ,GSSZ
0014                    DEF SQRI,RANDS
0015                    DEF CLKWS,CLKADR,CLKT01,CLKT02
0016                    DEF DS,DS1,DS2,DS3
0017                    DEF WPR1,WPR103,WPR104,WPR108
0018                    DEF IOB,EUS,FNS,GSS
0019                    DEF UFT,BUS,EORBUS,SLT
0020                    DEF VNT,VDT,NVD,NVS
0021                    DEF GSC,TY0,AINC
0022                    DEF UNIT,SLTN,DCNT,MODE
0023                    DEF PLC,DLIM,DLC,DDM
0024                    DEF LNUM,LEC,SLN,CCNT
0025                    DEF EFLG,ENUM,ELNM,ELSF
0026                    DEF FFLG,HFLG,IFLG,PLF
0027                    DEF BCRU,BAUD
0028                    DEF TEMP,TEMP2,TEMP4,TEMP6
0029                    DEF FPAC,FPAC2,FPAC4,FPWP
0030                    DEF CVCH,CVHD,CVHD01,CVHD15
0031                    DEF EVSKB,EBP,SSP,EVSKE
0032                    DEF WPR2,CSWP
0033                    DEF NEWY,NEWP,BEGN,PRDY
0034                    DEF CRLF,CRLF1
0035                    DEF TYP11,TYP0,TYPC,TYPE
0036                    DEF TYPB,TYPBE
0037                    DEF GETCR,GETC
0038                    DEF ITAB,INTSB,INTSP
0039                    DEF INTFLG,ISTCK
0040                    DEF STRTC,ID302
0041                    DEF CSOPZ,CSRDZ,CSWTZ,CSCLZ
0042                    DEF PROGM,ERRLS1,ERRLS2,CRCASS
0043                    DEF CVBFP,STRTC
0044                    DEF OCWNDO,OWNDO,CWNDO,NB              !HJC -27 MAR 1979
0045                    DEF ATNFE                              !HJC - 1 MAY 1979
0046            *
0047                    REF CVBFR
0048                    REF C1
0049                    REF FLDD,FSRD,FAD,FSD
0050                    REF FMD,FDD,FSCL,FNRM
0051                    REF FCLR,FNEG,FLOAT,EVFX
0052                    REF CVBD,CVBI,ERRS
0053                    REF CKEX,CLRV,EDIT,GTLN
0054                    REF STPY,LDSTRT,NLIN
0055                    REF OCRLOC,GOS1  ! HJC-27 MAR 1979
0056                    REF EVSFR,FUNFX,PLYXX,PLYX
0057            *
0058            *ALLOCATION ADDRESSES
0059            *
0060    4000    BRAM    EQU >4000       ;BEGINNING OF RAM
0061    0000    ERAM    EQU >0000       ;END OF RAM
```

```
0062      0000  BROM    EQU >0000          ;BEGINNING OF ROM
0063            *EROM   EQU EROMP          ;END OF ROM
0064            *   PRAM CHANGED FROM ERAM ->01F4 TO ->0202 28 MAR 79
0065            *                                 TO ->0200 23 APRIL 79
0066      FE00  PRAM    EQU ERAM->0200     ;PERMANENT RAM
0067            *
0068            *EQUATES
0069            *
0070      0004  NRV     EQU 4              ;NUMBER OF RESERVED VARIABLES
0071            *                          ;NRV IS ALSO EQUATED IN MODULE 2
0072      0084  LNSZ    EQU 132            ;I/O BUFFER SIZE
0073      0064  NIC     EQU LNSZ-32        ;NUMBER ON INPUT CHARACTERS
0074      0003  UMSK    EQU >3             ;ALLOW TTO,LPT
0075      3B00  FUZZ    EQU >3B00          ;APPROXIMATELY 1E-7
0076            *                          ;COMS IS ALSO EQUATED IN MODULE 2
0077      000A  FNSZ    EQU 10             ;NEXTED FOR/NEXT STACK SIZE
0078      0014  GSSZ    EQU 20             ;NEXTED GOSUB STACK SIZE
0079      0004  SQRI    EQU 4              ;# OF SQR NEWTON ITERATIONS
0080            *
0081            *DEFINE FLOATING POINT XOPS
0082            *
0083                    DXOP LOADF,0             ;LOAD FPAC
0084                    DXOP STORE,1             ;STORE FPAC
0085                    DXOP FADD,2              ;ADD TO FPAC
0086                    DXOP FSUB,3              ;SUBTRACT FROM FPAC
0087                    DXOP FMUL,4              ;MULTIPLY FPAC
0088                    DXOP FDIV,5              ;DIVIDE FPAC
0089                    DXOP SCALE,6             ;SCALE FPAC
0090                    DXOP NORMAL,7            ;NORMALIZE FPAC
0091                    DXOP CLEAR,8             ;CLEAR FPAC
0092                    DXOP NEGATE,9            ;NEGATE FPAC
0093                    DXOP FLOATF,10           ;FLOAT FPAC
0094            *
0095                    DXOP EVFIX,11            ;EVALUATE AND FIX
0096                    DXOP OUTFP,12            ;OUT FLOATING POINT #
0097                    DXOP OUTINT,13           ;OUT INTEGER
0098      2F80  ERROR   EQU >2F80                ;XOP XX,14  (ERROR CALL)
0099      2FA0  ERROR2  EQU ERROR+>20
```

```
0101            *FIRST BYTE     INTERNAL BYTES
0102            *
0103            *COMMANDS       00 CR           40 ;
0104            * 00 -                          41 ?
0105            * 01 GOTO*      FUNCTIONS       42 %
0106            * 02 GOSUB*     01-1A FNA-Z     43 $
0107            * 03 ELSE*                       44 "
0108            * 04 REM*       SYSTEM FUNCS    45 '
0109            * 05 FOR*       1B ABS          46 \
0110            * 06 (LET)*     1C ASC          47 !
0111            * 07 DATA       1D ATN          48 &
0112            * 08 NEXT       1E COS          49
0113            * 09 ERROR      1F EXP          4A
0114            * 0A PRINT      20 INP          4B
0115            * 0B CALL       21 LOG          4C (
0116            * 0C LOAD       22 NKY          4D )
0117            * 0D INPUT      23 SIN
0118            * 0E READ       24 SQR          OPERATERS
0119            * 0F RESTOR     25 SYS          4E OR
0120            * 10 RETURN     26 TIC          4F LOR
0121            * 11 STOP       27 BIT          50 AND
0122            * 12 UNIT       28 CRB          51 LAND
0123            * 13 TIME       29 CRF          52 NOT
0124            * 14 SAVE       2A MEM          53 LNOT
0125            * 15 BASE       2B LEN          54 LXOR
0126            * 16 ESCAPE     2C MCH          55 ==
0127            * 17 NOESC      2D SRH          56 =
0128            * 18 RANDOM     2E              57 >
0129            * 19 IMASK      2F              58 >=
0130            * 1A IRET       30              59 <
0131            * 1B TRAP       31              5A <=
0132            * 1C BAUD       32              5B <>
0133            * 1D POP        33              5C -
0134            * 1E DIM        34              5D +
0135            * 1F LET        35              5E /
0136            * 20 ON         36              5F *
0137            * 21 IF         37              60 ↑
0138            * 22 DEF                        61 (UNARY MINUS)
0139            * 23 NEW        DELIMITERS
0140            * 24 END        38 TO           VARIABLES
0141            * 25            39 TAB          62-6C -1 TO 9
0142            * 26            3A STEP         6D 2 BYTE INTEGER
0143            * 27            3B THEN         6E 2 BYTE HEX INTEGER
0144            * 28            3C ::           6F 6 BYTE FP NUMBER
0145            * 29            3D @            70-72 FUNCTION DUMMIES
0146            * 2A            3E #            73 RND
0147            * 2B            3F ,            74 COM[9]
0148            * 2D                            75-FF 139 VARIABLES
```

```
0150            *ERROR NUMBERS AND MESSAGES:
0151            *
0152            *       01 = SYNTAX ERROR
0153            *       02 = UNMATCHED PARENTHESIS
0154            *       03 = INVALID LINE NUMBER
0155            *       04 = ILLEGAL VARIABLE NAME
0156            *       05 = TOO MANY VARIABLES
0157            *       06 = ILLEGAL CHARACTER
0158            *       07 = EXPECTING OPERATOR
0159            *       08 = ILLEGAL FUNCTION NAME
0160            *       09 = ILLEGAL FUNCTION ARGUMENT
0161            *       10 = STORAGE OVERFLOW
0162            *       11 = STACK OVERFLOW
0163            *       12 = STACK UNDERFLOW
0164            *       13 = NO SUCH LINE NUMBER
0165            *       14 = EXPECTING STRING VARIABLE
0166            *       15 = INVALID SCREEN COMMAND
0167            *       16 = EXPECTING DIMENSIONED VARIABLE
0168            *       17 = SUBSCRIPT OUT OF RANGE
0169            *       18 = TOO FEW SUBSCRIPTS
0170            *       19 = TOO MANY SUBSCRIPTS
0171            *       20 = EXPECTING SIMPLE VARIABLE
0172            *       21 = DIGITS OUT OF RANGE  (0<D<12)
0173            *       22 = EXPECTING VARIABLE
0174            *       23 = READ OUT OF DATA
0175            *       24 = READ TYPE DIFFERS FROM DATA TYPE
0176            *       25 = SQUARE OF NEGATIVE NUMBER
0177            *       26 = LOG OF NON-POSITIVE NUMBER
0178            *       27 = EXPRESSION TOO COMPLEX
0179            *       28 = DIVISION BY ZERO
0180            *       29 = FLOATING POINT OVERFLOW
0181            *       30 = FIX ERROR
0182            *       31 = FOR W/O NEXT
0183            *       32 = NEXT W/O FOR
0184            *       33 = EXP FUNCTION HAS INVALID ARGUMENT
0185            *       34 = UNNORMALIZED NUMBER
0186            *       35 = PARAMETER ERROR
0187            *       36 = MISSORG ASSIGNMENT OPERATOR
0188            *       37 = ILLEGAL DELIMITER
0189            *       38 = UNDEFINED FUNCTION
0190            *       39 = UNDIMENSIONED VARIABLE
0191            *       40 = INVALID FILE NAME
0192            *       41 = ROM NOT INSTALLED
0193            *       42 = INTERRUPT W/O TRAP
0194            *       43 = INVALID BAUD RATE
0195            *       44 = TAPE READ ERROR
0196            *       45 = EPROM VERIFY ERROR
```

```
0198            *MEMORY USAGE
0199            *
0200            * NAME    LOC   DATA
0201            * ----    ----  ----
0202            * PBASIC  0000  XXXX    PBASIC INTREPRETER
0203            *         ....  ....
0204            *         4000  A5A5    RUNID   (EPROM HEADER)
0205            *         4002  XXXX    SLT
0206            *         4004  XXXX    VNT
0207            *         4006  XXXX    NVD-VDT
0208            * BUS     4008  0575    FOR I=1 TO 10   BEGINNING USER STORA
0209            *         400A  5664
0210            *         400C  386D
0211            *         400E  000A
0212            *         4010  0000
0213            *         4012  0875      PRINT I
0214            *         4014  0000
0215            *         4016  0675    NEXT I
0216            *         4018  0000
0217            * SLT     401A  001E    LINE 30         STATEMENT LOCATION T
0218            *         401C  000E
0219            *         401E  0014    LINE 20
0220            *         4020  000A
0221            *         4022  000A    LINE 10
0222            *         4024  0000
0223            *         4026  0000    SLT TERMINATER
0224            * VNT     4028  0000    DUMMY #1        VARIABLE NAME TABLE
0225            *         402A  0000    DUMMY #2
0226            *         402C  0000    DUMMY #3
0227            *         402E  11D2    RND
0228            *         4030  2400    1
0229            * VDT     4032  0000    ADR D#1         VARIABLE DEFINITION
0230            *         4034  0000    ADR D#2
0231            *         4036  0000    ADR D#3
0232            *         4038  0000    ADR RND
0233            *         403A  XXXX    ADR I
0234            * NVD     403C                          NEXT VARIABLE DEFINI
0235            *         ....
0236            * NVS     5012  0000    I STORAGE       NEXT VARIABLE STORAG
0237            *         5014  0000
0238            *         5016  0000
0239            * IOB     AED0  ....                    INPUT/OUTPUT BUFFER
0240            *               ....
0241            * UFT     XXXX  0000    FNA ADR (PBC)   USER FUNCTION TABLE
0242            *         XXXX  0000        # OF ARGS
0243            *         ....  ....
0244            *         XXXX  0000    FNZ ADR (PBC
0245            *         XXXX  0000        # OF ARGS
0246            * GSS     XXXX  0000    LEVEL #1 PBC    GOSUB STACK
0247            *         XXXX  0000             PLC
0248            *         ....  ....
0249            * FNS     XXXX  0000    LEVEL #1 VAR    FOR/NEXT STACK
0250            *         XXXX  0000             STEP
0251            *         XXXX  0000             -
0252            *         XXXX  0000             -
0253            *         XXXX  0000             TO
0254            *         XXXX  0000             -
0255            *         XXXX  0000             -
0256            *         XXXX  0000             PBC
0257            *         XXXX  0000             PLC
```

```
0258            *           ....  ....
0259            * EUS       XXXX  0000          END OF-USER-STORAGE
0260            * PRAM      XXXX  0000          PERMANENT RAM
0261            * EMEM      FFFE  0000
```

```
0263                ************************************************
0264                *
0265                *       RAM
0266                *
0267                ************************************************
0268                *
0269 FE00                   DORG PRAM
0270                *
0271                *INTERRUPT REGISTERS
0272                *
0273 FE00           ITAB    BSS 16*2          ;TRAP TABLE
0274       FE20     INTSB   EQU $
0275 FE20           IOWSTK  BSS 16*2          ;INTERRUPT OPEN WINDOW STACK     -HJ
0276       FE20     ISTCK   EQU IOWSTK
0277                *
0278       FE2C     INTWP1  EQU $-20
0279       FE32     INTWP2  EQU $-14
0280 FE40 0000      INTFLG  DATA 0            ;INTERRUPT FLAG
0281 FE42 0000      IOWSP   DATA 0            ;INTERRUPT OPEN WINDOW STACK PTR -HJ
0282       FE42     INTSP   EQU IOWSP
0283 FE44                   BSS 7*2
0284                *
0285                ********CLEAR TO HERE ON NEW
0286                *
0287 FE52 0000      RANDS   DATA 0            ;RANDOM SEED
0288                *
0289                *CLOCK INTERRUPT REGISTERS
0290                *
0291 FE54 0000      CLKWS   DATA 0            ;FINE COUNTER
0292 FE56 0000      CLKADR  DATA 0            ;HOURS
0293 FE58 0000              DATA 0            ;MINUTES
0294 FE5A 0000              DATA 0            ;SECONDS
0295 FE5C 0000      CLKT01  DATA 0            ;TIC COUNTER
0296 FE5E 0000      CLKT02  DATA 0            ;TIC COUNTER
0297 FE60 0000      DS      DATA 0,0,0        ;TEMPORARY DATA STORAGE
     FE62 0000
     FE64 0000
0298 FE66 0000      DS1     DATA 0,0,0
     FE68 0000
     FE6A 0000
0299 FE6C 0000              DATA 0,0,0,0      ;R12,R13,R14,R15
     FE6E 0000
     FE70 0000
     FE72 0000
0300                *
0301 FE74           WPR1    BSS 3*2           ;INITIAL WORKSPACE  (R0-R2)
0302 FE7A 0000      WPR103  DATA 0            ;R3
0303 FE7C 0000      WPR104  DATA 0            ;R4
0304 FE7E                   BSS 3*2           ;R5-R7
0305 FE84 0000      WPR108  DATA 0            ;R8
0306 FE86 FEA0              DATA PTRS         ;POINTER REGISTER
0307 FE88                   BSS 6*2           ;R10-R15
0308                ************************************************************
0309                *       OPEN AND CLOSE WINDOW INSTRUCTION SEQUENCE
0310                *          FOR INTERRUPT CHANGES  -HJC 27 MAR 79
0311                *
0312                * LIMI INSTRUCTIONS TO OPEN INTERRUPT WINDOW AFTER EACH
0313                * STATEMENT LINE ,THESE INSTRUCTIONS ARE CREATED BY CODE
0314                * IN THE START ROUTINE.
0315                ************************************************************
```

```
0316                    *
0317          FE94      OCWNDO  EQU $
0318 FE94 0300          OWLIMI  LIMI 0              ;OPEN WINDOW
     FE96 0000
0319 FE98 0300          CWLIMI  LIMI 0              ;CLOSE WINDOW
     FE9A 0000
0320 FE9C 045B          OCRET   RT                  ;RETURN TO RUN MODULE  23 APRIL 1979
0321          FE96      OWNDO   EQU  OWLIMI+2       ;OPEN WINDOW VALUE
0322          FE9A      CWNDO   EQU  CWLIMI+2       ;CLOSE WINDOW VALUE
0323 FE9E 0000          NB      DATA 0              ;NATIVE / BASIC INTERRRUPT THRESHOLD
0324                    *
0325                    ********************************************************************
0326                    *  WARNING          WARNING          WARNING
0327                    *
0328                    *  THESE TABLES ARE USED IN THE 302 EXTENDED AREA.  IF THEY
0329                    *  ARE CHANGED, PLEASE CHECK THE "INTERFAC" MODULE IN THE
0330                    *  EXTENDED LINK FOR THESE REFERENCES.
0331                    *
0332                    *  WARNING          WARNING          WARNING
0333                    ********************************************************************
0334                    *
0335                    *SYSTEM POINTERS
0336                    *
0337          FEA0      PTRS    EQU $               ;SYSTEM POINTERS
0338          0000      IOB     EQU $-PTRS
0339 FEA0 FC0E          DATA IOBP                   I/O BUFFER
0340          0002      EUS     EQU $-PTRS
0341 FEA2 FDFE          DATA EUSP                   END-USER-STORAGE
0342          0004      FNS     EQU $-PTRS
0343 FEA4 FD4A          DATA FNSP                   FOR/NEXT STACK
0344          0006      GSS     EQU $-PTRS
0345 FEA6 FCFA          DATA GSSP                   GOSUB STACK
0346          0008      UFT     EQU $-PTRS
0347 FEA8 FC8E          DATA UFTP-4                 USER FUNCTION TABLE
0348          000A      BUS     EQU $-PTRS
0349 FEAA 4008          DATA BUSP                   BEGINNING OF USER STORAGE
0350          000C      EORBUS  EQU $-PTRS
0351 FEAC 4008          DATA BUSP                   END OF RAM - BEGINNING OF USER STORG
0352                    ******                      END INIT SECTION
0353          000E      SLT     EQU $-PTRS
0354 FEAE 4008          DATA SLTP                   STATEMENT LOCATION TABLE
0355          0010      VNT     EQU $-PTRS
0356 FEB0 400A          DATA VNTP                   VARIABLE ~~DEFINITION~~ NAME TABLE
0357          0012      VDT     EQU $-PTRS
0358 FEB2 4012          DATA VDTP                   ~~NEXT~~ VARIABLE DEFINITION TABLE
0359          0014      NVD     EQU $-PTRS
0360 FEB4 401A          DATA NVDP                   NEXT VARIABLE POINTER
0361          0016      NVS     EQU $-PTRS
0362 FEB6 FC0E          DATA NVSP                   NEXT VARIABLE STORAGE
0363          0018      GSC     EQU $-PTRS
0364 FEB8 FCFA          DATA GSSP                   GOSUB STACK COUNTER
0365          001A      TYO     EQU $-PTRS          OUTPUT CHARACTER BUFFER
0366 FEBA 0000          DATA 0
0367          001C      AINC    EQU $-PTRS          AUTO-INCREMENT
0368 FEBC 0000          DATA 0
0369          001E      UNIT    EQU $-PTRS
0370 FEBE 0001          DATA 1                      OUTPUT UNIT
0371          0020      SLTN    EQU $-PTRS
0372 FEC0 0000          DATA 0                      CHANNEL #
0373          0022      DCNT    EQU $-PTRS
```

```
0374 FEC2 0000          DATA 0          INDENT COUNTER
0375      0024  MODE    EQU $-PTRS
0376 FEC4 0000          DATA 0          MODE SWITCH
0377      0026  PLC     EQU $-PTRS
0378 FEC6 0000          DATA 0          PROGRAM LINE COUNTER
0379      0028  DLIM    EQU $-PTRS
0380 FEC8 0000          DATA 0          DELIMITER
0381      002A  DLC     EQU $-PTRS
0382 FECA 0000          DATA 0          DATA LINE COUNTER
0383      002C  DDM     EQU $-PTRS
0384 FECC 0000          DATA 0          DATA DELIMITER PTR
0385      002E  LNUM    EQU $-PTRS
0386 FECE 0000          DATA 0          LAST ENTERED LINE #
0387      0030  LEC     EQU $-PTRS
0388 FED0 0000          DATA 0          LAST ENTERED CHARACTER
0389      0032  SLN     EQU $-PTRS
0390 FED2 0000          DATA 0          SAVED LINE NUMBER
0391      0034  CCNT    EQU $-PTRS
0392 FED4 0000          DATA 0          COLUMN COUNTER
0393      0036  EFLG    EQU $-PTRS
0394 FED6 0000          DATA 0          ERROR FLAG
0395      0038  ENUM    EQU $-PTRS
0396 FED8 0000          DATA 0          LAST ERROR NUMBER
0397      003A  ELNM    EQU $-PTRS
0398 FEDA 0000          DATA 0          LAST ERROR LINE #
0399      003C  ELSF    EQU $-PTRS
0400 FEDC 0000          DATA 0          ELSE FLAG
0401      003E  FFLG    EQU $-PTRS
0402 FEDE 0000          DATA 0          FORMATTING FLAG
0403      0040  HFLG    EQU $-PTRS
0404 FEE0 0000          DATA 0          HELP FLAG
0405      0042  IFLG    EQU $-PTRS
0406 FEE2 0000          DATA 0          INPUT FLAG
0407      0044  PLF     EQU $-PTRS
0408 FEE4 0000          DATA 0          PROGRAM LOAD FLAG
0409      0046  BCRU    EQU $-PTRS
0410 FEE6 0000          DATA 0          CRU BASE ADDRESS
0411      0048  ASR0    EQU $-PTRS
0412 FEE8 0000          DATA 0          ASR FLAGS
0413      004A  ASR1    EQU $-PTRS
0414 FEEA 0000          DATA 0
0415            *
0416      004C  PTRSZ   EQU $-PTRS
0417            *
0418            *
0419            *STORAGE POINTERS
0420            *
0421            *
0422      FDFE  EUSP    EQU PRAM-2
0423      FD4A  FNSP    EQU EUSP-(FNSZ*18)
0424      FCFA  GSSP    EQU FNSP-(GSSZ*4)
0425      FC92  UFTP    EQU GSSP-(26*4)
0426      FC0E  IOBP    EQU UFTP-LNSZ
0427      FC0E  NVSP    EQU IOBP
0428            *
0429      4008  BUSP    EQU BRAM+8
0430      4008  SLTP    EQU BUSP
0431      400A  VNTP    EQU SLTP+2
0432      4012  VDTP    EQU VNTP+(NRV*2)
0433      401A  NVDP    EQU VDTP+(NRV*2)
```

```
0435                    *
0436                    *STACKS FOR EDIT,LIST, AND EVAL
0437                    *
0438 FEEC 0000  CVCH    DATA 0,0,0
     FEEE 0000
     FEF0 0000
0439       FEF2 CVHD    EQU $
0440       FEF3 CVHD01  EQU CVHD+1
0441       FF01 CVHD15  EQU CVHD+15
0442       FEF2 EVSKB   EQU $              ;BEGINNING OF STACK
0443 FEF2       EBP     BSS 132            ;EDIT BUFFER
0444 FF76       SSP     BSS 30             ;SUBSCRIPT STACK
0445       FF94 EVSKE   EQU $              ;END STACK
0446                    *
0447                    *FLOATING POINT WORKSPACE AND TEMPORARY STORAGE
0448                    *
0449 FF94 0000  TEMP    DATA 0             ;FP TEMP REGISTER
0450 FF96 0000  TEMP2   DATA 0
0451 FF98 0000  TEMP4   DATA 0
0452 FF9A 0000  TEMP6   DATA 0
0453 FF9C 0000  FPAC    DATA 0             ;FP ACCUMULATER
0454 FF9E 0000  FPAC2   DATA 0
0455 FFA0 0000  FPAC4   DATA 0
0456 FFA2               BSS 13*2
0457       FF9C FPWP    EQU FPAC           ;FLOATING POINT WORKSPACE
0458                    *
0459                    *
0460                    *
0461                    ********************************************************************
0462                    *  WARNING         WARNING        WARNING
0463                    *
0464                    *  THESE TABLES ARE USED IN THE 302 EXTENDED AREA.  IF THEY
0465                    *  ARE CHANGED, PLEASE CHECK THE "INTERFAC" MODULE IN THE
0466                    *  EXTENDED LINK FOR THESE REFERENCES.
0467                    *
0468                    *  WARNING         WARNING        WARNING
0469                    ********************************************************************
0470                    *
0471       FFBC CSWP    EQU $
0472 FFBC 0000  DS2     DATA 0,0,0
     FFBE 0000
     FFC0 0000
0473 FFC2 0000  DS3     DATA 0,0,0
     FFC4 0000
     FFC6 0000
0474 FFC8               BSS 10*2
0475                    *
0476                    *2ND LEVEL REGISTERS
0477                    *
0478 FFDC       WPR2    BSS 16*2           ;EVAL,TRIG,CONV REGISTERS
0479       FFFC EMEM    EQU $
```

```
0481 0000                    RORG BROM
0482                  *
0483 0000 FFDC  IVZERO  DATA WPR2,START      ;INTERRUPT VECTOR LABELS ADDED
     0002 0198'
0484 0004 FE2C  IV1     DATA INTWP1,INTIN    ;HJC - 27 MARCH 1979
     0006 00AC'
0485 0008 FE2C  IV2     DATA INTWP1,INTIN
     000A 00AC'
0486 000C FE54  IV3     DATA CLKWS,CLKI
     000E 00EE'
0487 0010 FE2C  IV4     DATA INTWP1,INTIN
     0012 00AC'
0488 0014 FE2C  IV5     DATA INTWP1,INTIN
     0016 00AC'
0489 0018 FE2C  IV6     DATA INTWP1,INTIN
     001A 00AC'
0490 001C FE2C  IV7     DATA INTWP1,INTIN
     001E 00AC'
0491 0020 FE2C  IV8     DATA INTWP1,INTIN
     0022 00AC'
0492 0024 FE2C  IV9     DATA INTWP1,INTIN
     0026 00AC'
0493 0028 FE2C  IVA     DATA INTWP1,INTIN
     002A 00AC'
0494 002C FE2C  IVB     DATA INTWP1,INTIN
     002E 00AC'
0495 0030 FE2C  IVC     DATA INTWP1,INTIN
     0032 00AC'
0496 0034 FE2C  IVD     DATA INTWP1,INTIN
     0036 00AC'
0497 0038 FE2C  IVE     DATA INTWP1,INTIN
     003A 00AC'
0498 003C FE2C  IVF     DATA INTWP1,INTIN
     003E 00AC'
0499                  *
0500 0040 FF9C          DATA FPWP,FLDD  ;XOP XX,0  LOAD FPAC
     0042 0000
0501 0044 FF9C          DATA FPWP,FSRD  ;XOP XX,1  STORE FPAC
     0046 0000
0502 0048 FF9C          DATA FPWP,FAD   ;XOP XX,2  ADD TO FPAC
     004A 0000
0503 004C FF9C          DATA FPWP,FSD   ;XOP XX,3  SUBTRACT FROM FPAC
     004E 0000
0504 0050 FF9C          DATA FPWP,FMD   ;XOP XX,4  MULTIPLY FPAC
     0052 0000
0505 0054 FF9C          DATA FPWP,FDD   ;XOP XX,5  DIVIDE FPAC
     0056 0000
0506 0058 FF9C          DATA FPWP,FSCL  ;XOP XX,6  SCALE FPAC
     005A 0000
0507 005C FF9C          DATA FPWP,FNRM  ;XOP XX,7  NORMALIZE
     005E 0000
0508 0060 FF9C          DATA FPWP,FCLR  ;XOP XX,8  CLEAR
     0062 0000
0509 0064 FF9C          DATA FPWP,FNEG  ;XOP XX,9  NEGATE
     0066 0000
0510 0068 FF9C          DATA FPWP,FLOAT ;XOP XX,10 FLOAT FPAC
     006A 0000
0511 006C FFDC          DATA WPR2,EVFX  ;XOP XX,11 EVALUATE AND FIX
     006E 0000
0512 0070 FFDC          DATA WPR2,CVBD  ;XOP XX,11 OUT FP #
```

```
     0072 0000
0513 0074 FFDC          DATA WPR2,CVBI  ;XOP XX,13 OUT INTEGER #
     0076 0000
0514 0078 FE74          DATA WPR1,ERRS  ;USER ERROR
     007A 0000
0515 007C FFFF          DATA -1,-1
     007E FFFF
0516                    *
0517                    *-------VECTOR TABLE TO 302 PROM-------*
0518                    *
0519                    ************************************************************
0520                    *  WARNING        WARNING        WARNING
0521                    *
0522                    *  THESE TABLES ARE USED IN THE 302 EXTENDED AREA.  IF THEY
0523                    *  ARE CHANGED, PLEASE CHECK THE "INTERFAC" MODULE IN THE
0524                    *  EXTENDED LINK FOR THESE REFERENCES.
0525                    *
0526                    *  WARNING        WARNING        WARNING
0527                    ************************************************************
0528                    *
0529 0080 0460          B @GETC
     0082 039E'
0530 0084 0460          B @PRDY
     0086 02E4'
0531 0088 0460          B @TYPE
     008A 031E'
0532 008C 0460          B @TYPO
     008E 030A'
0533 0090 0460          B @CVBFR
     0092 0000
0534 0094 0460          B @EVSFR        1 MAY 1979
     0096 0000
0535 0098 0460          B @FUNFX        1 MAY 1979
     009A 0000
0536 009C 0460          B @PLYXX        1 MAY 1979
     009E 0000
0537 00A0 0460          B @PLYX         1 MAY 1979
     00A2 0000
0538                    *
0539      3002  CSOPZ   EQU  >3002
0540      3006  CSRDZ   EQU  >3006
0541      300A  CSWTZ   EQU  >300A
0542      300E  CSCLZ   EQU  >300E
0543      3012  PROGM   EQU  >3012
0544      3016  CVBFP   EQU  >3016
0545      301A  ERRLS1  EQU  >301A
0546      301E  ERRLS2  EQU  >301E
0547      3022  ATNFE   EQU  >3022        1 MAY 1979
0548      1700  CRCASS  EQU  >1700
0549 00A4 A5A5  STRTC   DATA >A5A5
0550 00A6 A55A  ID302   DATA >A55A
```

```
0552                *GATHER INTERRUPTS
0553                *
0554                *   THE INTERRUPT CODE HERE WAS CHANGED 27 MAR 1979 (HJC)
0555                *   TO USE THE OPEN WINDOW/CLOSE WINDOW TECHNIQUE
0556                *
0557 00A8 FE32      INTINZ  DATA INTWP2,INTIN1
     00AA 00B0'
0558                *
0559 00AC 0420      INTIN   BLWP @INTINZ      ;DELAY AND GET LEVEL
     00AE 00A8'
0560                *
0561 00B0 0300      INTIN1  LIMI 0            ;DISABLE INTERRUPTS
     00B2 0000
0562 00B4 C24F              MOV R15,R9        ;GET INTERRUPT LEVEL
0563 00B6 0249              ANDI R9,>F
     00B8 000F
0564 00BA 0A19              SLA R9,1          ;X 2
0565                *  PUT TRAP LINE NUMBER INTO R1 OF WPR1 FOR GOSUB
0566                *
0567 00BC CAA9              MOV @ITAB+2(9),@2(10)
     00BE FE02
     00C0 0002
0568                *
0569 00C2 1313              JEQ INTER         ;IF NO TRAP ADDRESS ,THEN ERROR
0570                *  NOW WE KNOW INTERRUPT LEVEL IS OK WE CAN PROCESS IT
0571 00C4 0288              CI  R8,INTSB      ;GET R8 POINTING IN OPEN WINDOW STAC
     00C6 FE20
0572 00C8 1402              JHE INTIN2
0573 00CA 0208              LI R8,INTSB
     00CC FE20
0574                *
0575      00CE'     INTIN2  EQU  $
0576 00CE CE20              MOV @OWNDO,*R8+ ;STACK OPEN WINDOW
     00D0 FE96
0577 00D2 0819              SRA R9,>1         ;SET R9 TO INTERRUPT LEVEL -1
0578 00D4 C809              MOV R9,@OWNDO     ;SET OPEN WINDOW TO INTERRUPT LEVEL-
     00D6 FE96
0579 00D8 C3E0              MOV @NB,R15       ;SET SAVED STATUS TO NB
     00DA FE9E
0580 00DC C34A              MOV R10,R13       ;SET SAVED WP=WPR1
0581 00DE 020E              LI  R14,@INTIN3 ;SET SAVED PC TO WHERE WE ARE GOING
     00E0 00E4'
0582 00E2 0380              RTWP
0583                *
0584 00E4 C188      INTIN3  MOV R8,R6         ;SET UP BYTE POINTER
0585 00E6 0460              B   @GOS1         ;TREAT AS A GOSUB
     00E8 0000
0586                *
0587 00EA 2FA0      INTER   DATA ERROR2,42
     00EC 002A
```

```
0589                    *CLOCK INTERRUPT ROUTINE
0590                    *
0591                    *       R0=FINE COUNTER
0592                    *       R1=HOURS
0593                    *       R2=MINUTES
0594                    *       R3=SECONDS
0595                    *       R4-R5=TIC COUNTS
0596                    *
0597 00EE 020C  CLKI    LI R12,2*>80    ;POINT TO 9901
     00F0 0100
0598 00F2 1E00          SBZ 0           ;SELECT CLOCK
0599 00F4 1D03          SBO 3           ;CLEAR INTERRUPT 3
0600 00F6 0585          INC R5          ;COUNT TICS
0601 00F8 1601            JNE $+4
0602 00FA 0584            INC R4
0603 00FC 0280          CI R0,24        ;25 TICS?
     00FE 0018
0604 0100 1402            JHE CLKI1     ;Y, 1 SECOND
0605 0102 0580          INC R0          ;N, COUNT
0606 0104 0380          RTWP            ;RESTORE MASK
0607                    *
0608 0106 04C0  CLKI1   CLR R0          ;CLEAR FINE COUNTER
0609 0108 04CC          CLR R12         ;POINT TO LED
0610 010A 06C3          SWPB R3         ;POSITION SECONDS
0611 010C 3043          LDCR R3,1       ;BLINK LIGHT
0612 010E 06C3          SWPB R3         ;RESTORE R3
0613 0110 0583          INC R3          ;COUNT SECONDS
0614 0112 0283          CI R3,60        ;<60?
     0114 003C
0615 0116 110B            JLT CLKI2     ;Y, RETURN
0616 0118 04C3          CLR R3          ;N
0617 011A 0582          INC R2          ;INCREMENT MINUTE
0618 011C 0282          CI R2,60        ;<60?
     011E 003C
0619 0120 1106            JLT CLKI2     ;Y, RETURN
0620 0122 04C2          CLR R2          ;N,
0621 0124 0581          INC R1          ;INCREMENT HOUR
0622 0126 0281          CI R1,24        ;<24 HOURS?
     0128 0018
0623 012A 1101            JLT CLKI2     ;Y, RETURN
0624 012C 04C1          CLR R1          ;N
0625 012E 0380  CLKI2   RTWP
```

```
0627 0130 2FA0  BAUDE  DATA ERROR2,43
     0132 002B
0628            *
0629            *BAUD COMMAND
0630            *
0631 0134 2EC1  BAUD   EVFIX R1          ;GET DEVICE
0632 0136 C0C9         MOV R9,R3
0633 0138 0223         AI R3,ASR0        ;GET FLAG ADR
     013A 0048
0634 013C 020C         LI R12,>40*2
     013E 0080
0635 0140 C041         MOV R1,R1         ;NON-ZERO?
0636 0142 1303           JEQ BAUD1       ;Y
0637 0144 020C         LI R12,>C0*2      ;N, SELECT 2ND TERMINAL
     0146 0180
0638 0148 05C3         INCT R3
0639            *
0640 014A 0280  BAUD1  CI R0,>3F00        ;,?
     014C 3F00
0641 014E 16F0           JNE BAUDE       ;N, ERROR
0642 0150 2EC2         EVFIX R2          ;GET INDEX #
0643 0152 0282         CI R2,6           ;CHECK FOR BAUD OUT OF RANGE
     0154 0006
0644 0156 1BEC           JH BAUDE
0645 0158 0A22         SLA R2,2          ;X 4
0646 015A 1D1F         SBO 31            ;RESET UART
0647 015C 3220         LDCR @B62,8       ;SET CONTROL REGISTER
     015E 03E0'
0648 0160 1E0D         SBZ 13            ;NOT CLOCK
0649 0162 3322         LDCR @SBAUDT+2(2),12
     0164 017C'
0650            *
0651 0166 04D3         CLR *R3
0652 0168 0282         CI R2,>10         ;CHECK PTR IN TABLE FOR 1200
     016A 0010
0653 016C 1103           JLT BAUD2       ;>1200
0654 016E 1303           JEQ BAUD3       ;=1200
0655 0170 0713         SETO *R3          ;<1200
0656 0172 1001         JMP BAUD3
0657            *
0658 0174 0593  BAUD2  INC *R3           ;>1200
0659            *
0660 0176 0460  BAUD3  B @NLIN
     0178 0000
0661            *
0662            *BAUD RATE TABLE
0663            *
0664 017A 0007  SBAUDT  DATA >7,>1A      ;19200 BAUD
     017C 001A
0665 017E 000E          DATA >E,>34      ;9600 BAUD
     0180 0034
0666 0182 001D          DATA >1D,>68     ;4800 BAUD
     0184 0068
0667 0186 003B          DATA >3B,>D0     ;2400 BAUD
     0188 00D0
0668 018A 0075  B1200   DATA >75,>1A0    ;1200 BAUD
     018C 01A0
0669            *       DATA >EA,>340    ;600 BAUD
0670 018E 0246          DATA >246,>4D0   ;300 BAUD
     0190 04D0
```

```
0671 0192 7FFF           DATA >7FFF,>638 ;110 BAUD
     0194 0638
0672                *
0673                *      MODEL INSTRUCTIONS TO BUILD LIMI INSTRUCTIONS IN
0674                *      RAM TO OPEN AND CLOSE INTERRUPT WINDOW
0675                *       HJC 27 MAR 1979
0676                *
0677 0196 0300   MODLM   DATA  >0300      ;MODEL LIMI INSTRUCTION
0678                *
```

```
0680 0198 0205  START   LI R5,PRAM       ;GET PERMANENT RAM POINTER
     019A FE00
0681 019C 02A2          STWP R2          ;GET WORKSPACE POINTER
0682              *
0683 019E 0642  START1  DECT R2           ;BACKUP
0684 01A0 C485          MOV R5,*R2       ;STORE PATTERN
0685 01A2 04C1          CLR R1           ;CLEAR BUS
0686 01A4 8485          C R5,*R2         ;SAME?
0687 01A6 1604            JNE START2     ;N
0688 01A8 04D2          CLR *R2          ;Y, ZERO MEMORY
0689 01AA 0282          CI R2,BRAM
     01AC 4000
0690 01AE 18F7            JH START1
0691              *
0692 01B0 05C2  START2  INCT R2           ;MOVE TO RAM
0693 01B2 0282          CI R2,>F000      ;EXPANSION MEMORY?
     01B4 F000
0694 01B6 1A0A            JL START3      ;Y
0695 01B8 C805          MOV R5,@>EFFE    ;MAYBE
     01BA EFFE
0696 01BC 04C1          CLR R1           ;CLEAR BUS
0697 01BE 8160          C @>EFFE,R5      ;MEMORY AT @>EFFE?
     01C0 EFFE
0698 01C2 1604            JNE START3     ;N
0699 01C4 0202          LI R2,>F000      ;Y, HOLE
     01C6 F000
0700 01C8 C142          MOV R2,R5        ;SET PRAM AND SIZE AGAIN
0701 01CA 10E9          JMP START1
0702              *
0703              *       R2=BRAM
0704              *       R5=PRAM
0705              *
0706 01CC 0201  START3  LI R1,PTRS+2     ;GET POINTER
     01CE FEA2
0707 01D0 0645          DECT R5          ;RESERVE WORD AT ERAM
0708 01D2 CC45          MOV R5,*R1+      ;SAVE EUS
0709 01D4 0225          AI R5,-FNSZ*18
     01D6 FF4C
0710 01D8 CC45          MOV R5,*R1+      ;SAVE FNS
0711 01DA 0225          AI R5,-GSSZ*4
     01DC FFB0
0712 01DE CC45          MOV R5,*R1+      ;SAVE GSS
0713 01E0 0225          AI R5,-26*4-4
     01E2 FF94
0714 01E4 CC45          MOV R5,*R1+      ;SAVE UFT
0715 01E6 0225          AI R5,-LNSZ+4
     01E8 FF80
0716 01EA C805          MOV R5,@PTRS     ;SAVE IOB
     01EC FEA0
0717 01EE 0222          AI  R2,8         ;RESERVE 4 WORDS FOR EPROM POINTERS
     01F0 0008
0718 01F2 CC42          MOV R2,*R1+      ;SAVE BUS
0719 01F4 C442          MOV R2,*R1       ;SAVE EORBUS
0720 01F6 02E0          LWPI WPR1        ;GET INITIAL WORKSPACE
     01F8 FE74
0721 01FA 02A0          STWP R0
0722 01FC C800          MOV R0,@>FFFC    ;SET LOAD VECTOR FOR WARM START
     01FE FFFC
0723 0200 0200          LI R0,BEGN
     0202 02D8'
```

```
0724 0204 C800          MOV R0,@>FFFE
     0206 FFFE
0725 0208 0209          LI R9,PTRS      ;SET POINTERS
     020A FEA0
0726                *
0727                *       BUILD INSTRUCTIONS IN RAM TO OPEN AND CLOSE
0728                *       INTERRUPT WINDOW -HJC 27 MAR 79
0729                *
0730 020C C820          MOV   @MODLM,@OWLIMI
     020E 0196'
     0210 FE94
0731 0212 C820          MOV   @MODLM,@CWLIMI
     0214 0196'
     0216 FE98
0732 0218 C820          MOV   @GETC3,@OCRET
     021A 038C'
     021C FE9C
0733                *
0734                *       SET NATIVE/BASIC INTERRUPT THRESHOLD
0735                *       NB TAKES THE VALUE OF THE PRIORITY OF THE LOWEST PRI
0736                *       INTERRUPT ROUTINE NOT HANDLED BY A BASIC ROUTINE.
0737                *       ALL BASIC TRAP STATEMENTS MUST BE FOR A LEVEL GT NB
0738                *
0739 021E 0203          LI    R3,12
     0220 000C
0740 0222 0202          LI    R2,IVF+4
     0224 0040'
0741 0226 0204          LI    R4,ESTART
     0228 03E0'
0742 022A 0642  RECOMP  DECT  R2
0743 022C 0642          DECT  R2
0744 022E 8122          C     @2(R2),R4
     0230 0002
0745 0232 1B02          JH    NBFND
0746 0234 0603          DEC   R3
0747 0236 15F9          JGT   RECOMP
0748 0238 0223  NBFND   AI    R3,3
     023A 0003
0749 023C C803          MOV   R3,@NB
     023E FE9E
0750                *
0751                *  GUARANTEE THAT AUDIO CASSETTES ARE STOPPED
0752                *
0753 0240 020C          LI  R12,CRCASS  ;GET CRU BASE OF AUDIO CASSETTES
     0242 1700
0754 0244 1E00          SBZ 0           ;STOP CASSETTE DRIVE #1
0755 0246 1E01          SBZ 1           ;STOP CASSETTE DRIVE #2
0756                *
0757 0248 0202          LI R2,>3000
     024A 3000
0758 024C 84A0          C @STRTC,*R2    ;RUN MODULE IN POSITION?
     024E 00A4'
0759 0250 1606            JNE SBAUD     ;N
0760 0252 06A0          BL @NEWP1       ;Y, DO NEW
     0254 02AE'
0761 0256 0729          SETO @MODE(9)   ;SET TO RUN
     0258 0024
0762 025A 0460          B @LDSTRT       ;GO
     025C 0000
```

```
0764                    *INITIALIZE BAUD RATE
0765                    *
0766 025E 020C  SBAUD   LI R12,>40*2     ;GET CRU BASE
     0260 0080
0767 0262 04E9          CLR @ASR0(9)     ;CLEAR ASR FLAG  ( 1200 BAUD)
     0264 0048
0768 0266 1D1F          SBO 31           ;RESET TMS 9902 UART
0769 0268 3220          LDCR @B62,8      ;INITIALIZE TMS9902 CONTROL REG
     026A 03E0'
0770 026C 1E0D          SBZ 13           ;DO NOT INIT INTERVAL REG
0771 026E 04C3          CLR R3           ;CLEAR COUNTER
0772                    *
0773 0270 1F0F  SBAUD0  TB 15            ;START?
0774 0272 13FE            JEQ SBAUD0     ;N
0775                    *
0776 0274 0583  SBAUD1  INC R3           ;Y, COUNT
0777 0276 1F0F          TB 15            ;DONE?
0778 0278 16FD            JNE SBAUD1     ;N
0779 027A 0204          LI R4,SBAUDT-2   ;GET TABLE ADDRESS
     027C 0178'
0780                    *
0781 027E 05C4  SBAUD2  INCT R4          ;MOVE TO NEXT
0782 0280 8D03          C R3,*R4+        ;MATCH?
0783 0282 18FD            JH SBAUD2      ;N, LOOP TO NEXT
0784 0284 3314          LDCR *R4,12      ;INITIALIZE REC/XMT DATA RATE
0785                    *
0786 0286 1F15  SBAUD3  TB  21           ;WAIT FOR CHARACTER
0787 0288 16FE             JNE  SBAUD3   ;NOT DONE
0788 028A 1E12          SBZ 18           ;CLEAR
0789 028C 0284          CI R4,B1200+2    ;1200 BAUD?
     028E 018C'
0790                    *
0791 0290 1104          JLT  SBAUD4      ;HIGH BAUD RATE ( >1200 BAUD)
0792 0292 1308          JEQ  NEWP        ;LEAVE ASR FLAG ALONE ( =1200 BAUD)
0793 0294 0729          SETO @ASR0(9)    ;SET ASR FLAG  ( <1200 BAUD)
     0296 0048
0794 0298 1008          JMP  NEWP        ;CONTINUE
0795 029A 05A9  SBAUD4  INC @ASR0(9)     ;SHOW NO CARRIAGE RETURN DELAY
     029C 0048
0796 029E 1005          JMP  NEWP        ;CONTINUE
```

```
0798                *NEW STATEMENT
0799                *
0800 02A0 06A0      NEWY    BL @CKEX          ;EXPRESSION?
     02A2 0000
0801 02A4 1002                JMP NEWP        ;N
0802 02A6 2EE9              EVFIX @EORBUS(9)    ;Y, GET NEW EORBUS
     02A8 000C
0803                *
0804                *NEW COMMAND
0805                *
0806 02AA 020B      NEWP    LI R11,BEGN
     02AC 02D8'
0807                *
0808      02AE'     NEWP1   EQU $
0809 02AE CA69              MOV @EORBUS(9),@BUS(9) ;BUS=END OF RAM BUS
     02B0 000C
     02B2 000A
0810 02B4 C069              MOV @BUS(9),R1  ;GET BEGINNING USER STORAGE
     02B6 000A
0811 02B8 CA41              MOV R1,@SLT(9)  ;SET STATEMENT LOCATION TABLE
     02BA 000E
0812 02BC 04F1              CLR *R1+        ;LEAVE NULL
0813 02BE CA41              MOV R1,@VNT(9)  ;SET VARIABLE DEFINITION TABLE
     02C0 0010
0814 02C2 C0C1              MOV R1,R3
0815 02C4 0223              AI R3,NRV*2     ;SKIP RESERVED WORDS
     02C6 0008
0816 02C8 CA43              MOV R3,@VDT(9)  ;SET NEXT VARIABLE DEFINITION
     02CA 0012
0817 02CC 0223              AI R3,NRV*2
     02CE 0008
0818 02D0 CA43              MOV R3,@NVD(9)  ;SET NEXT VARIABLE POINTER
     02D2 0014
0819 02D4 0460              B @CLRV         ;CLEAR R1 THRU EUS
     02D6 0000
0820                *
0821                *WARM START
0822                *
0823                *       -SET MODE TO IDLE
0824                *       -CLEAR PROGRAM LOAD FFLAG
0825                *
0826 02D8 0201      BEGN    LI R1,MBEGN     ;GET MESSAGE
     02DA 03BE'
0827 02DC CA60              MOV @C1,@UNIT(9)
     02DE 0000
     02E0 001E
0828 02E2 1002              JMP PRDY1
```

```
0830                    *
0831                    *PRINT '*READY' MESSAGE
0832                    *
0833 02E4 0201  PRDY    LI R1,MPRDY       ;GET MESSAGE
     02E6 03D5'
0834 02E8 04E9  PRDY1   CLR @AINC(9)      ;CLEAR AUTO-INC FLAG
     02EA 001C
0835 02EC 1002          JMP CRLF0
0836                    *
0837 02EE 0201  CRLF    LI R1,MCRLF       ;OUT CRLF
     02F0 03DD'
0838                    *
0839 02F2 06A0  CRLF0   BL @TYPE          ;OUT STRING R1
     02F4 031E'
0840 02F6 04E9          CLR @MODE(9)      ;SET MODE TO IDLE
     02F8 0024
0841 02FA 04E9          CLR @PLF(9)       ;CLEAR PROGRAM LOAD
     02FC 0044
0842 02FE 06A0          BL @GTLN          ;GET INPUT LINE
     0300 0000
0843                    *
0844 0302 06A0  CRLF1   BL @EDIT          ;EDIT STATEMENT
     0304 0000
0845 0306 10F3          JMP CRLF
```

```
0847                *OUTPUT ROUTINES
0848                *
0849                *       TYP0            OUT R0
0850                *       TYPC            OUT CRLF
0851                *       TYPB            OUT IOB
0852                *       TYPE            OUT (R1) STRING
0853                *
0854                *TYPE 0(11)
0855                *
0856 0308 C03B     TYP11    MOV *R11+,R0    ;GET CHARACTER(S)
0857                *
0858                *TYPE R0
0859                *
0860 030A C049     TYP0     MOV R9,R1       ;GET ADR
0861 030C 0221              AI R1,TY0       ;ADD DISPLACEMENT
     030E 001A
0862 0310 C440              MOV R0,*R1      ;SAVE CHARACTER(S)
0863 0312 1005              JMP TYPE        ;OUT
0864                *
0865                *TYPE CRLF
0866                *
0867 0314 0201     TYPC     LI R1,MCRLF     ;LOAD PTR
     0316 03DD'
0868 0318 1002              JMP TYPE        ;OUT
0869                *
0870                *TYPE I/O BUFFER
0871                *
0872 031A 75D7     TYPBE    SB *R7,*R7      ;END STRING & OUT BUFFER
0873                *
0874 031C C059     TYPB     MOV *R9,R1
0875                *
0876 031E D031     TYPE     MOVB *R1+,R0    ;GET BYTE
0877 0320 134D               JEQ GETC3      ;DONE
0878 0322 C329              MOV @UNIT(9),R12
     0324 001E
0879 0326 091C              SRL R12,1       ;UNIT 1?
0880 0328 1704               JNC TYPE1      ;N
0881 032A 0420              BLWP @TYPEZ
     032C 0340'
0882 032E 0080               DATA >40*2,ASR0
     0330 0048
0883                *
0884 0332 091C     TYPE1    SRL R12,1       ;UNIT 2?
0885 0334 17F4               JNC TYPE       ;N
0886 0336 0420              BLWP @TYPEZ
     0338 0340'
0887 033A 0180               DATA >C0*2,ASR1
     033C 004A
0888 033E 10EF              JMP TYPE
0889                *
0890 0340 FFDC     TYPEZ    DATA WPR2,OUTC
     0342 0344'
0891                *
0892 0344 C33E     OUTC     MOV *R14+,R12   ;GET BASE
0893 0346 C06D              MOV @18(13),R1
     0348 0012
0894 034A C081              MOV R1,R2
0895 034C A0BE              A *R14+,R2      ;POINT TO ASR FLAG
0896 034E C01D              MOV *R13,R0     ;GET CHARACTER
0897 0350 0240              ANDI R0,>7F00   ;MASK TO 7 BITS
```

```
     0352 7F00
0898 0354 1D10          SBO 16          ;SET RTSON
0899                *
0900 0356 1F16  OUTC1   TB 22           ;READY?
0901 0358 16FE           JNE OUTC1
0902 035A 1F1B          TB   27
0903 035C 1302          JEQ  GOON
0904 035E 1F1B  BACK    TB   27
0905 0360 16FE          JNE  BACK
0906 0362 3200  GOON    LDCR R0,8       ;SEND CHARACTER
0907 0364 1E10          SBZ 16          ;TURN RTSOFF
0908 0366 0280          CI R0,>0D00
     0368 0D00
0909 036A 160B            JNE ASR733    ;N, DO WE NEED 3 NULLS  OR?
0910 036C 04E1          CLR @CCNT(1)    ;Y, CLEAR COLUMN COUNTER
     036E 0034
0911                *     MOV *R2,R0    ;ARE WE USING FAST BAUD RATE
0912                *       JGT OUTC4   ;N, IGNORE WAIT LOOPS
0913 0370 0200          LI R0,30000     ;LOOP COUNT FOR CARRIAGE RETURN
     0372 7530
0914                *
0915                *  THIS ROUTINE WAITS .2 SECONDS FOR THE CARRIAGE RETURN
0916                *  PROVIDED THAT THE BAUD RATE IS LESS THAN OR EQUAL TO
0917                *  1200 BAUD; FASTER RATES HAVE NO DELAY.  IF THE TERMINAL
0918                *  IS OPERATES AT 1200 BAUD OR LESS, EACH CHARACTER IS
0919                *  PADDED WITH 25 MSEC TO REDUCE THE EFFECTIVE TRANSFER
0920                *  RATE TO 300 BAUD.
0921                *
0922 0374 1F16  OUTC2   TB   22         ;IS TRANSMIT BUFFER EMPTY
0923 0376 16FE          JNE  OUTC2      ;IF NOT, WAIT UNTIL IT IS
0924 0378 1F17          TB   23         ;IS SHIFT REGISTER EMPTY
0925 037A 16FC          JNE  OUTC2      ;IF NOT, WAIT UNTIL IT IS
0926                *
0927 037C 0600  OUTC3   DEC R0          ;WAIT LOOP
0928 037E 16FE            JNE OUTC3     ;IF NOT DONE, JUMP BACK
0929 0380 0380  OUTC4   RTWP            ;Y, CONTINUE
0930                *
0931 0382 C012  ASR733  MOV *R2,R0      ;ASR733 AT 1200 BAUD
0932 0384 16FD            JNE OUTC4     ;NO, GET NEXT CHARACTER
0933 0386 0200          LI R0,3750      ;YES, WAIT 3 NULL TIMES
     0388 0EA6
0934 038A 10F4          JMP OUTC2       WAIT
```

```
0936                *CHARACTER ACCESS ROUTINES
0937                *
0938                *       GETCR   = GET CHARACTER
0939                *
0940 038C C28B     GETCR   MOV R11,R10     ;SAVE RETURN
0941                *
0942 038E 06A0     GETCR1  BL @GETC        ;GET CHARACTER
     0390 039E'
0943 0392 10FD              JMP GETCR1     ;NOTHING
0944 0394 045A              B *R10         ;CHARACTER, RETURN
0945 0396 C041            MOV R1,R1        ;ESC, LOOK AT MODE, IDLE?
0946 0398 13A5              JEQ PRDY       ;Y
0947 039A 0460            B @STPY
     039C 0000
0948                *
0949                *GET CHARACTER FROM KEYBOARD
0950                *       BL @GETC
0951                *         NOTHING
0952                *         R0=CHARACTER
0953                *       ESCAPE
0954                *
0955 039E 020C     GETC     LI R12,>40*2    ;GET CRU BASE
     03A0 0080
0956 03A2 1F15              TB 21          ;CHARACTER?
0957 03A4 160B              JNE GETC3      ;N
0958 03A6 04C0            CLR R0           ;Y
0959 03A8 3SC0            STCR R0,7        ;GET CHARACTER
0960 03AA 1E12            SBZ 18
0961 03AC C069     CHAR   MOV @MODE(9),R1 ;ESC ALLOWED?
     03AE 0024
0962 03B0 1504              JGT GETC2      ;N
0963 03B2 0280            CI R0,>1B00      ;ESC?
     03B4 1B00
0964 03B6 1601              JNE GETC2      ;N
0965 03B8 05CB     GETC1    INCT R11       ;Y, JMP 4(11) ESCAPE
0966 03BA 05CB     GETC2    INCT R11       ;JMP 2(11) R0=CHARACTER
0967 03BC 045B     GETC3    RT             ;RETURN,USED AS MODEL INSTRUCTION
```



039E B @ PKBD

```
PKBD   LI   R12,>~~00~~120
       TB   ?
       JNE  NOCHAR
       LDCR R0,8
       SBO  STROBE
       B    @CHAR
NOCHAR LI   R12,>80
       B    @GETC
```

```
 0969                   *TEXT STORAGE
 0970                   *
 0971 03BE 0A0D  MBEGN  DATA >0A0D
 0972 03C0   54         TEXT 'TM990 BASIC REV D.1.9'
      03C1   4D
      03C2   39
      03C3   39
      03C4   30
      03C5   20
      03C6   42
      03C7   41
      03C8   53
      03C9   49
      03CA   43
      03CB   20
      03CC   52
      03CD   45
      03CE   56
      03CF   20
      03D0   44
      03D1   2E
      03D2   31
      03D3   2E
      03D4   39
 0973 03D5   0A  MPRDY  BYTE >0A,>0D
      03D6   0D
 0974 03D7   2A         TEXT '*READY'
      03D8   52
      03D9   45
      03DA   41
      03DB   44
      03DC   59
 0975 03DD   0A  MCRLF  BYTE >0A,>0D,0
      03DE   0D
      03DF   00
 0976 03E0   62  B62    BYTE >62
 0977      03E0' ESTART EQU  $-1          ! ATTACH LABEL TO LAST BYTE- HJC 27
 0978 03E2              EVEN
 0979                    END
NO ERRORS,     NO WARNINGS
```

$		03E2'		0274	0278	0279	0317	0337	0338	0340	0342	0344
				0346	0348	0350	0353	0355	0357	0359	0361	0363
				0365	0367	0369	0371	0373	0375	0377	0379	0381
				0383	0385	0387	0389	0391	0393	0395	0397	0399
				0401	0403	0405	0407	0409	0411	0413	0416	0439
				0442	0445	0471	0479	0575	0601	0808	0977	
AINC	D	001C	0367	0021	0834							
ASR0		0048	0411	0633	0767	0793	0795	0882				
ASR1		004A	0413	0887								
ASR733		0382'	0931	0909								
ATNFE	D	3022	0547	0045								
B1200		018A'	0668	0789								
B62		03E0'	0976	0647	0769							
BACK		035E'	0904	0905								
BAUD	D	0134'	0631	0027								
BAUD1		014A'	0640	0636								
BAUD2		0174'	0658	0653								
BAUD3		0176'	0660	0654	0656							
BAUDE		0130'	0627	0641	0644							
BCRU	D	0046	0409	0027								
BEGN	D	02D8'	0826	0033	0723	0806						
BRAM		4000	0060	0429	0689							
BROM		0000	0062	0481								
BUS	D	000A	0348	0019	0809	0810						
BUSP		4008	0429	0349	0351	0430						
C1	R	02DE'	0048	0827								
CCNT	D	0034	0391	0024	0910							
CKEX	R	02A2'	0053	0800								
CLEAR			0091									
CLKADR	D	FE56	0292	0015								
CLKI		00EE'	0597	0486								
CLKI1		0106'	0608	0604								
CLKI2		012E'	0625	0615	0619	0623						
CLKT01	D	FE5C	0295	0015								
CLKT02	D	FE5E	0296	0015								
CLKWS	D	FE54	0291	0015	0486							
CLRV	R	02D6'	0053	0819								
CRCASS	D	1700	0548	0042	0753							
CRLF	D	02EE'	0837	0034	0845							
CRLF0		02F2'	0839	0835								
CRLF1	D	0302'	0844	0034								
CSCLZ	D	300E	0542	0041								
CSOPZ	D	3002	0539	0041								
CSRDZ	D	3006	0540	0041								
CSWP	D	FFBC	0471	0032								
CSWTZ	D	300A	0541	0041								
CVBD	R	0072'	0052	0512								
CVBFP	D	3016	0544	0043								
CVBFR	R	0092'	0047	0533								
CVBI	R	0076'	0052	0513								
CVCH	D	FEEC	0438	0030								
CVHD	D	FEF2	0439	0030	0440	0441						
CVHD01	D	FEF3	0440	0030								
CVHD15	D	FF01	0441	0030								
CWLIMI		FE98	0319	0322	0731							
CWNDO	D	FE9A	0322	0044								
DCNT	D	0022	0373	0022								
DDM	D	002C	0383	0023								
DLC	D	002A	0381	0023								
DLIM	D	0028	0379	0023								

```
DS      D  FE60     0297  0016
DS1     D  FE66     0298  0016
DS2     D  FFBC     0472  0016
DS3     D  FFC2     0473  0016
EBP     D  FEF2     0443  0031
EDIT    R  0304'    0053  0844
EFLG    D  0036     0393  0025
ELNM    D  003A     0397  0025
ELSF    D  003C     0399  0025
EMEM       FFFC     0479
ENUM    D  0038     0395  0025
EORBUS  D  000C     0350  0019  0802  0809
ERAM       0000     0061  0066
ERRLS1  D  301A     0545  0042
ERRLS2  D  301E     0546  0042
ERROR      2F80     0098  0099
ERROR2     2FA0     0099  0587  0627
ERRS    R  007A'    0052  0514
ESTART     03E0'    0977  0741
EUS     D  0002     0340  0018
EUSP       FDFE     0422  0341  0423
EVFIX              0095
EVFX    R  006E'    0051  0511
EVSFR   R  0096'    0056  0534
EVSKB   D  FEF2     0442  0031
EVSKE   D  FF94     0445  0031
FAD     R  004A'    0049  0502
FADD               0085
FCLR    R  0062'    0051  0508
FDD     R  0056'    0050  0505
FDIV               0088
FFLG    D  003E     0401  0026
FLDD    R  0042'    0049  0500
FLOAT   R  006A'    0051  0510
FLOATF             0093
FMD     R  0052'    0050  0504
FMUL               0087
FNEG    R  0066'    0051  0509
FNRM    R  005E'    0050  0507
FNS     D  0004     0342  0018
FNSP       FD4A     0423  0343  0424
FNSZ    D  000A     0077  0013  0423  0709
FPAC    D  FF9C     0453  0029  0457
FPAC2   D  FF9E     0454  0029
FPAC4   D  FFA0     0455  0029
FPWP    D  FF9C     0457  0029  0500  0501  0502  0503  0504  0505  0506  0507
                          0508  0509  0510
FSCL    R  005A'    0050  0506
FSD     R  004E'    0049  0503
FSRD    R  0046'    0049  0501
FSUB               0086
FUNFX   R  009A'    0056  0535
FUZZ    D  3B00     0075  0013
GETC    D  039E'    0955  0037  0529  0942
GETC1      0388'    0965
GETC2      038A'    0966  0962  0964
GETC3      038C'    0967  0732  0877  0957
GETCR   D  038C'    0940  0037
GETCR1     038E'    0942  0943
GOON       0362'    0906  0903
```

```
GOS1    R  00E8'   0055  0585
GSC     D  0018    0363  0021
GSS     D  0006    0344  0018
GSSP       FCFA    0424  0345  0364  0425
GSSZ    D  0014    0078  0013  0424  0711
GTLN    R  0300'   0053  0842
HFLG    D  0040    0403  0026
ID302   D  00A6'   0550  0040
IFLG    D  0042    0405  0026
INTER      00EA'   0587  0569
INTFLG  D  FE40    0280  0039
INTIN      00AC'   0559  0484  0485  0487  0488  0489  0490  0491  0492  0493
                         0494  0495  0496  0497  0498
INTIN1     00B0'   0561  0557
INTIN2     00CE'   0575  0572
INTIN3     00E4'   0584  0581
INTINZ     00A8'   0557  0559
INTSB   D  FE20    0274  0038  0571  0573
INTSP   D  FE42    0282  0038
INTWP1     FE2C    0278  0484  0485  0487  0488  0489  0490  0491  0492  0493
                         0494  0495  0496  0497  0498
INTWP2     FE32    0279  0557
IO8     D  0000    0338  0018
IOBP       FC0E    0426  0339  0427
IOWSP      FE42    0281  0282
IOWSTK     FE20    0275  0276
ISTCK   D  FE20    0276  0039
ITAB    D  FE00    0273  0038  0567
IV1        0004'   0484
IV2        0008'   0485
IV3        000C'   0486
IV4        0010'   0487
IV5        0014'   0488
IV6        0018'   0489
IV7        001C'   0490
IV8        0020'   0491
IV9        0024'   0492
IVA        0028'   0493
IVB        002C'   0494
IVC        0030'   0495
IVD        0034'   0496
IVE        0038'   0497
IVF        003C'   0498  0740
IVZERO     0000'   0483
LDSTRT  R  025C'   0054  0762
LEC     D  0030    0387  0024
LNSZ    D  0084    0072  0012  0073  0426  0715
LNUM    D  002E    0385  0024
LOADF              0083
MBEGN      038E'   0971  0826
MCRLF      03DD'   0975  0837  0867
MODE    D  0024    0375  0022  0761  0840  0961
MODLM      0196'   0677  0730  0731
MPRDY      03D5'   0973  0833
NB      D  FE9E    0323  0044  0579  0749
NBFND      0238'   0748  0745
NEGATE             0092
NEWP    D  02AA'   0806  0033  0792  0794  0796  0801
NEWP1      02AE'   0808  0760
NEWY    D  02A0'   0800  0033
```

```
NIC     D  0064     0073  0012
NLIN    R  0178'    0054  0660
NORMAL             0090
NRV     D  0004     0070  0012  0432  0433  0815  0817
NVD     D  0014     0359  0020  0818
NVDP       401A     0433  0360
NVS     D  0016     0361  0020
NVSP       FC0E     0427  0362
OCRET      FE9C     0320  0732
OCRLOC  R          0055
OCWNDO  D  FE94     0317  0044
OUTC       0344'    0892  0890
OUTC1      0356'    0900  0901
OUTC2      0374'    0922  0923  0925  0934
OUTC3      037C'    0927  0928
OUTC4      0380'    0929  0932
OUTFP              0096
OUTINT             0097
OWLIMI     FE94     0318  0321  0730
OWNDO   D  FE96     0321  0044  0576  0578
PLC     D  0026     0377  0023
PLF     D  0044     0407  0026  0841
PLYX    R  00A2'    0056  0537
PLYXX   R  009E'    0056  0536
PRAM       FE00     0066  0269  0422  0680
PRDY    D  02E4'    0833  0033  0530  0946
PRDY1      02E8'    0834  0828
PROGM   D  3012     0543  0042
PTRS       FEA0     0337  0306  0338  0340  0342  0344  0346  0348  0350  0353
                          0355  0357  0359  0361  0363  0365  0367  0369  0371
                          0373  0375  0377  0379  0381  0383  0385  0387  0389
                          0391  0393  0395  0397  0399  0401  0403  0405  0407
                          0409  0411  0413  0416  0706  0716  0725
PTRSZ      004C     0416
R0         0000           0603  0605  0608  0640  0721  0722  0723  0724  0856
                          0862  0876  0896  0897  0906  0908  0913  0927  0931
                          0933  0958  0959  0963
R1         0001           0621  0622  0624  0631  0635  0635  0685  0696  0706
                          0708  0710  0712  0714  0718  0719  0810  0811  0812
                          0813  0814  0826  0833  0837  0860  0861  0862  0867
                          0874  0876  0893  0894  0945  0945  0961
R10        000A           0580  0940  0944
R11        000B           0806  0856  0940  0965  0966
R12        000C           0597  0609  0634  0637  0753  0766  0878  0879  0884
                          0892  0955
R13        000D           0580  0896
R14        000E           0581  0892  0895
R15        000F           0562  0579
R2         0002           0617  0618  0620  0642  0643  0645  0652  0681  0683
                          0684  0686  0688  0689  0692  0693  0699  0700  0717
                          0718  0719  0740  0742  0743  0744  0757  0758  0894
                          0895  0931
R3         0003           0610  0611  0612  0613  0614  0616  0632  0633  0638
                          0651  0655  0658  0739  0746  0748  0749  0771  0776
                          0782  0814  0815  0816  0817  0818
R4         0004           0602  0741  0744  0779  0781  0782  0784  0789
R5         0005           0600  0680  0684  0686  0695  0697  0700  0707  0708
                          0709  0710  0711  0712  0713  0714  0715  0716
R6         0006           0584
R7         0007           0872  0872
```

R8		0008		0571	0573	0576	0584					
R9		0009		0562	0563	0564	0577	0578	0632	0725	0860	0874
RANDS	D	FE52	0287	0014								
RECOMP		022A'	0742	0747								
SBAUD		025E'	0766	0759								
SBAUD0		0270'	0773	0774								
SBAUD1		0274'	0776	0778								
SBAUD2		027E'	0781	0783								
SBAUD3		0286'	0786	0787								
SBAUD4		029A'	0795	0791								
SBAUDT		017A'	0664	0649	0779							
SCALE			0089									
SLN	D	0032	0389	0024								
SLT	D	000E	0353	0019	0811							
SLTN	D	0020	0371	0022								
SLTP		4008	0430	0354	0431							
SQRI	D	0004	0079	0014								
SSP	D	FF76	0444	0031								
START		0198'	0680	0483								
START1		019E'	0683	0690	0701							
START2		01B0'	0692	0687								
START3		01CC'	0706	0694	0698							
STORE			0084									
STPY	R	039C'	0054	0947								
STRTC	D	00A4'	0549	0040	0043	0758						
TEMP	D	FF94	0449	0028								
TEMP2	D	FF96	0450	0028								
TEMP4	D	FF98	0451	0028								
TEMP6	D	FF9A	0452	0028								
TY0	D	001A	0365	0021	0861							
TYP0	D	030A'	0860	0035	0532							
TYP11	D	0308'	0856	0035								
TYPB	D	031C'	0874	0036								
TYPBE	D	031A'	0872	0036								
TYPC	D	0314'	0867	0035								
TYPE	D	031E'	0876	0035	0531	0839	0863	0868	0885	0888		
TYPE1		0332'	0884	0880								
TYPEZ		0340'	0890	0881	0886							
UFT	D	0008	0346	0019								
UFTP		FC92	0425	0347	0426							
UMSK	D	0003	0074	0012								
UNIT	D	001E	0369	0022	0827	0878						
VDT	D	0012	0357	0020	0816							
VDTP		4012	0432	0358	0433							
VNT	D	0010	0355	0020	0813							
VNTP		400A	0431	0356	0432							
WPR1	D	FE74	0301	0017	0514	0720						
WPR103	D	FE7A	0302	0017								
WPR104	D	FE7C	0303	0017								
WPR108	D	FE84	0305	0017								
WPR2	D	FFDC	0478	0032	0483	0511	0512	0513	0890			

*

```
SOURCE ACCESS NAME=      DBASIC.REL4.BASIC.SOURCE.ERRECOVY
OBJECT ACCESS NAME=      DBASIC.REL4.BASIC.OBJECT.ERRECOVY
LISTING ACCESS NAME=     DBASIC.REL4.BASIC.LIST.ERRECOVY
ERROR ACCESS NAME=
OPTIONS=                 X
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'ERRECOVY'
0002              *     26 APRIL 1979
0003              *
0004                    DEF ERRS,ERM1,STPY,ERRS2,ERRS0
0005                    DEF STPYA                ;SPR 35 HJC 1 MAY 1979
0006              *
0007                    REF B20,CRLF,EFLG,ENUM
0008                    REF FFLG,GOSB1,MODE
0009                    REF MOVEB,MOVEL,PLC,SLN
0010                    REF TYPBE
0011                    REF ERRLS1,ID302
0012                    REF SLT
0013                    DXOP OUTINT,13
0014              *     08/10/78
0015              *
0016              *ERROR RECOVERY ROUTINE
0017              *
0018 0000 C06E    ERRS  MOV @-2(14),R1   ;GET INSTRUCTION
     0002 FFFE
0019 0004 0241          ANDI R1,>3F      ;ISOLATE ERROR
     0006 003F
0020              *
0021 0008 CA41    ERRS0 MOV R1,@ENUM(9)  ;SAVE ERROR #
     000A 0000
0022 000C 04E9          CLR @FFLG(9)     ;RESET INPUT & FORMATTING FLAG
     000E 0000
0023 0010 8820          C   @ID302,@>3000  ;IS >A55A AT LOCATION >3000
     0012 0000
     0014 3000
0024 0016 1603          JNE ERRS01       ;NO, OUTPUT ERROR NUMBER
0025 0018 06A0          BL  @ERRLS1      ;OUTPUT ERROR COMMENT
     001A 0000
0026 001C 1008          JMP ERRS02       ;CONTINUE
0027 001E 06A0  ERRS01  BL @MOVEB        ;'*ERROR '
     0020 0000
0028 0022 0A0D  ERM1      DATA >0A0D
0029 0024   2A            TEXT '*ERROR '
     0025   45
     0026   52
     0027   52
     0028   4F
     0029   52
     002A   20
0030 002B   00            BYTE 0
0031 002C 2F41          OUTINT R1        ;CONVERT
0032 002E DDE0  ERRS02  MOVB @B20,*R7+   ;OUT SPACE
     0030 0000
0033              *
0034 0032 C029          MOV @MODE(9),R0  ;CHECK MODE
     0034 0000
0035 0036 1336            JEQ ERRS2      ;IDLE
0036 0038 C069          MOV @EFLG(9),R1  ;ERROR COMMAND EXECUTED?
     003A 0000
0037 003C 1326            JEQ ERRS1      ;N
0038 003E 04E9          CLR @EFLG(9)     ;Y, CLEAR IN CASE NOT FOUND
     0040 003A'
0039 0042 0460          B @GOSB1         ;DO GOSUB
     0044 0000
0040              *     ENTRY POINT FOR STOP STATEMENT  SPR 35 HJC 1 MAY 79
0041              *     UPDATE IS MADE SO WE CAN SQUEEZE CODE WHEN IT COMES TO
```

```
0042                    *    THE CRUNCH.    INSURANCE
0043 0046 06A0  STPYA   BL @MOVEB
     0048 0020'
0044 004A 0A0D          DATA >0A0D
0045 004C   53          TEXT 'STOP '
     004D   54
     004E   4F
     004F   50
     0050   20
0046 0051   00          BYTE 0
0047                    *
0048 0052 C069          MOV @PLC(9),R1
     0054 0000
0049 0056 0221          AI  R1,-4           ; CHECK TO SEE IF THERE IS A
     0058 FFFC
0050 005A 8A41          C   R1,@SLT(9)      ; STATEMENT AFTER THE STOP
     005C 0000
0051 005E 1A15          JL  ERRS1
0052 0060 C069          MOV @PLC(9),R1
     0062 0054'
0053 0064 C061          MOV @-6(1),R1       ;GET LINE NUMBER FOR CONT
     0066 FFFA
0054 0068 CA41          MOV R1,@SLN(9)      ; SAVE FOR CONTINUE
     006A 0000
0055 006C C069          MOV @PLC(9),R1      ;RESET R1
     006E 0062'
0056 0070 C061          MOV @-2(1),R1
     0072 FFFE
0057 0074 06A0          BL  @MOVEL
     0076 0000
0058 0078   41          TEXT 'AT '
     0079   54
     007A   20
0059 007B   00          BYTE 0
0060 007C 1010          JMP  CLFLAG
0061                    *
0062 007E 06A0  STPY    BL @MOVEB           ;       INPUT/ESC ENTRY, OUT 'STOP'
     0080 0048'
0063 0082 0A0D            DATA >0A0D
0064 0084   53            TEXT 'STOP '
     0085   54
     0086   4F
     0087   50
     0088   20
0065 0089   00            BYTE 0
0066                    *
0067 008A C069  ERRS1   MOV @PLC(9),R1      ;ERR ENTRY
     008C 006E'
0068 008E C061          MOV @-2(1),R1       ;GET LINE #
     0090 FFFE
0069 0092 06A0          BL @MOVEL           ;RUN, OUT LINE #
     0094 0076'
0070 0096   41            TEXT 'AT '
     0097   54
     0098   20
0071 0099   00            BYTE 0
0072 009A CA41          MOV R1,@SLN(9)
     009C 006A'
0073 009E 04E9  CLFLAG  CLR @FFLG(9)        ;CLEAR FORMATTING
     00A0 000E'
```

E

HECK TO SEE IF THERE IS A

TATEMENT AFTER THE STOP

T LINE NUMBER FOR CONT

AVE FOR CONTINUE

SET R1

INPUT/ESC ENTRY, OUT 'STOP'

ENTRY

LINE #

, OUT LINE #

AR FORMATTING

```
 0074 00A2 2F41            OUTINT R1         ;CONVERT
 0075                 *
 0076 00A4 06A0  ERRS2     BL @TYPBE         ;END & OUT
      00A6 0000
 0077 00A8 0460            B @CRLF
      00AA 0000
 0078                      END
NO ERRORS
```

LABEL		VALUE	DEFN	REFERENCES								
B20	R	0030'	0007	0032								
CLFLAG		009E'	0073	0060								
CRLF	R	00AA'	0007	0077								
EFLG	R	0040'	0007	0036	0038							
ENUM	R	000A'	0007	0021								
ERM1	D	0022'	0028	0004								
ERRLS1	R	001A'	0011	0025								
ERRS	D	0000'	0018	0004								
ERRS0	D	0008'	0021	0004								
ERRS01		001E'	0027	0024								
ERRS02		002E'	0032	0026								
ERRS1		008A'	0067	0037	0051							
ERRS2	D	00A4'	0076	0004	0035							
FFLG	R	00A0'	0008	0022	0073							
GOSB1	R	0044'	0008	0039								
ID302	R	0012'	0011	0023								
MODE	R	0034'	0008	0034								
MOVEB	R	0080'	0009	0027	0043	0062						
MOVEL	R	0094'	0009	0057	0069							
OUTINT			0013									
PLC	R	008C'	0009	0048	0052	0055	0067					
R0		0000		0034								
R1		0001		0018	0019	0021	0031	0036	0048	0049	0050	0052
				0053	0054	0055	0056	0067	0068	0072	0074	
R7		0007		0032								
SLN	R	009C'	0009	0054	0072							
SLT	R	005C'	0012	0050								
STPY	D	007E'	0062	0004								
STPYA	D	0046'	0043	0005								
TYPBE	R	00A6'	0010	0076								

*

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.SFSN
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.SFSN
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.SFSN
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'SFSN'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       SFSN              ;SEARCH FOR STATEMENT #
0009          *
0010          * COPY: (NONE)
0011          *
0012          * MACRO: (NONE)
0013          *
0014          * EXTERNAL ROUTINES:
0015          *
0016    2F80  ERROR   EQU >2F80                ;XOP XX,14  (ERROR CALL)
0017          *
0018          * EXTERNAL DATA:
0019          *
0020                REF SLT                  ;STATEMENT LOCATION TABLE
0021          *
0022          * MODULE EQUATES: (NONE)
0023          *
0024          *
0025          * MODULE VARIABLES AND CONSTANTS:
0026          *
0027                TITL "SEARCH FOR STATEMENT NUMBER"
```

```
0029            * ABSTRACT:
0030            *
0031            *       SEARCH IN STATEMENT-LOCATION-TABLE FOR THE LINE NUMB
0032            *       THE SLT IS ORGANIZED AS FOLLOWS FOR STATEMENTS 100,2
0033            *
0034            *                           SLT     300
0035            *                                   PLC
0036            *
0037            *                                   200
0038            *                                   PLC
0039            *
0040            *                                   100
0041            *                                   PLC
0042            *
0043            *                                   0
0044            *
0045            * CALLING SEQUENCE:
0046            *
0047            *       BL @SFSN
0048            *
0049            *      IN  - R1 = #
0050            *      OUT - R8 = PLC
0051            *
0052            *      NORMAL EXIT - RETURN
0053            *      ERROR EXIT TO ERROR ROUTINE
0054            *
0055            * EXCEPTIONS AND CONDITIONS:
0056            *
0057            *       ERROR 13 RESULTS IF LINE IS NOT FOUND
0058            *
0059            *
0060            * EXTERNAL ROUTINE LIST:
0061            *
0062            *      (SEE EXTERNAL ROUTINES)
0063            *
0064            * LOCAL DATA:
0065            *
0066            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0068                    * ENTRY POINT:
0069                    *
0070                            DEF SFSN
0071                    *
0072 0000 C229  SFSN    MOV @SLT(9),R8  ;START AT TOP
     0002 0000
0073                    *
0074 0004 C338  SFSN1   MOV *R8+,R12    ;DONE?
0075 0006 1305            JEQ SFSN3     ;Y
0076 0008 804C          C R12,R1        ;FOUND?
0077 000A 1302            JEQ SFSN2     ;Y
0078 000C 05C8          INCT R8         ;N, CONTINUE
0079 000E 10FA          JMP SFSN1
0080                    *
0081 0010 045B  SFSN2   RT              ;RETURN
0082                    *
0083 0012 2F8D  SFSN3   DATA ERROR+13   ;NO SUCH LINE #
0084                    END
NO ERRORS,      NO WARNINGS
```

```
ERROR      2F80     0016  0083
R1         0001           0076
R12        000C           0074  0076
R8         0008           0072  0074  0078
SFSN    D  0000'    0072  0070
SFSN1      0004'    0074  0079
SFSN2      0010'    0081  0077
SFSN3      0012'    0083  0075
SLT     R  0002'    0020  0072
*
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.JMP
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.JMP
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.JMP
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'JMP'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       JMPR0A           ;JUMP ON BYTE
0009          *
0010          * COPY: (NONE)
0011          *
0012          * MACRO: (NONE)
0013          *
0014          * EXTERNAL ROUTINES:  (NONE)
0015          *
0016          * EXTERNAL DATA:
0017          *
0018                REF DLIM          ;PROGRAM DELIMITER
0019          *
0020          * MODULE EQUATES:  (NONE)
0021          *
0022          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0025            * ABSTRACT:
0026            *
0027            *       JMPR0A IS SIMPLY A ROUTINE TO SAVE
0028            *       MEMORY WHEN DECIDING UPON AN ACTION
0029            *       FROM A TABLE OF ALTERNATIVES SELECTED
0030            *       BY A BYTE CODE.
0031            *
0032            * CALLING SEQUENCE:
0033            *
0034            *       BL @JMPR0
0035            *TABLE    BYTE DISP,BYTE
0036            *         BYTE DISP,BYTE
0037            *         ...
0038            *         DATA 0
0039            *
0040            *       IN R8 = PBC
0041            *       OUT DLIM UPDATED
0042            *
0043            *       EXIT TO TABLE+DISP ON BYTE FOUND
0044            *            OTHERWISE AFTER TABLE
0045            *
0046            *
0047            *       BL @JMPR0
0048            *TABLE    BYTE DISP,BYTE
0049            *         BYTE DISP,BYTE
0050            *         ...
0051            *         DATA 0
0052            *
0053            *       IN R0 = BYTE CODE
0054            *
0055            *       EXIT TO TABLE+DISP ON BYTE FOUND
0056            *            OTHERWISE AFTER TABLE
0057            *
0058            * EXCEPTIONS AND CONDITIONS:  (NONE)
0059            *
0060            * EXTERNAL ROUTINE LIST:
0061            *
0062            *      (SEE EXTERNAL ROUTINES)
0063            *
0064            * LOCAL DATA:
0065            *
0066            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0068                    * ENTRY POINT:
0069                    *
0070                            DEF JMPR0A
0071                    *
0072 0000 04C0  JMPR0A  CLR R0              ;GET DELIMITER
0073 0002 D038          MOVB *R8+,R0
0074 0004 CA40          MOV R0,@DLIM(9) ;SAVE
     0006 0000
0075                    *
0076                    * ENTRY POINT:
0077                    *
0078                            DEF JMPR0
0079                    *
0080 0008 C08B  JMPR0   MOV R11,R2          ;GET TABLE ADR
0081                    *
0082 000A D07B  JMPR01  MOVB *R11+,R1       ;GET DISPLACEMENT, DONE?
0083 000C 1305            JEQ JMPR02        ;Y
0084 000E 9EC0          CB R0,*R11+         ;N, CODE FOUND?
0085 0010 16FC            JNE JMPR01        ;N, KEEP LOOKING
0086 0012 0871          SRA R1,7            ;Y, POSITION
0087 0014 A081          A R1,R2             ;ADD DISPLACEMENT
0088 0016 0452          B *R2               ;GOTO ROUTINE
0089                    *
0090 0018 058B  JMPR02  INC R11             ;MOVE TO NEXT LINE
0091 001A 045B          RT
0092                    END
NO ERRORS,      NO WARNINGS
```

DLIM	R	0006'	0018	0074			
JMPR0	D	0008'	0080	0078			
JMPR01		000A'	0082	0085			
JMPR02		0018'	0090	0083			
JMPR0A	D	0000'	0072	0070			
R0		0000		0072	0073	0074	0084
R1		0001		0082	0086	0087	
R11		0008		0080	0082	0084	0090
R2		0002		0080	0087	0088	
R8		0008		0073			

*

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.READ
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.READ
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.READ
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'READ'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *           30 MAY 79          ;COPY & UPDATE TO PROD RELS
0006              * ROUTINE LIST:
0007              *
0008              *       RDDY              ;READ COMMAND
0009              *       RNWY              ;RESTOR COMMAND
0010              *
0011              * COPY: (NONE)
0012              *
0013              * MARCO: (NONE)
0014              *
0015              * EXTERNAL ROUTINES:
0016              *
0017                      REF BMVE                  ;MOVE BYTE
0018                      REF CKEX                  ;CHECK FOR EXPRESSION
0019                      REF EVARZ                 ;GET VARIABLE
0020                      REF EVERZ                 ;EVALUATE EXPRESSION
0021                      REF EVSDZ                 ;EVALUATE STRING
0022                      REF NLIN                  ;EXIT TO MULTIPLEXOR
0023                      REF NLIN0                 ;EXIT TO MULTIPLEXOR
0024                      REF REST                  ;RESTORE DATA PTR
0025                      REF SFSN                  ;SEARCH FOR STATEMENT NUMBER
0026                      DXOP EVFIX,11             ;EVALUATE AND FIX
0027    2F80  ERROR       EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0028              *
0029              * EXTERNAL DATA:
0030              *
0031                      REF BUS           ;BEGINNING OF USER STORAGE
0032                      REF DATXB         ;DATA INTERNAL BYTE CODE
0033                      REF DDM           ;DATA DELIMITER
0034                      REF DLC           ;DATA LINE COUNTER
0035                      REF DLIM          ;LINE DELIMITER
0036                      REF SLT           ;STATEMENT LOCATION TABLE
0037              *
0038              * MODULE EQUATES:  (NONE)
0039              *
0040              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0043            * ABSTRACT:
0044            *
0045            *       READ COMMAND USES FDATA TO PICK
0046            *       UP NEXT ITEM FROM DATA STATEMENTS.
0047            *       SINCE THE EVALUATOR IS USED, ANY
0048            *       EXPRESSION CAN APPEAR IN THE DATA
0049            *       STATEMENT.
0050            *
0051            * CALLING SEQUENCE:
0052            *
0053            *       B @RDDY
0054            *       B @RNWY
0055            *
0056            *       EXIT TO NLIN
0057            *
0058            * EXCEPTIONS AND CONDITIONS:
0059            *
0060            *       ERROR 24 FOR READING INTO STRING CONSTANT
0061            *       ERROR 23 FOR READ WITHOUT DATA
0062            *
0063            * EXTERNAL ROUTINE LIST:
0064            *
0065            *      (SEE EXTERNAL ROUTINES)
0066            *
0067            * LOCAL DATA:
0068            *
0069            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0071                    *FIND DATA
0072                    *       BL @FDATA
0073                    *         STRING
0074                    *       #
0075                    *
0076 0000 C188  FDATA   MOV R8,R6       ;SAVE PBC
0077 0002 C169          MOV @DLIM(9),R5 ;SAVE DLIM
     0004 0000
0078 0006 C229          MOV @DDM(9),R8  ;GET DELIMITER
     0008 0000
0079 000A 1307            JEQ FDATA1    ;MOVE TO NEXT LINE
0080 000C 04C0          CLR R0
0081 000E D038          MOVB *R8+,R0    ;LOOK AT DELIMITER
0082 0010 1304            JEQ FDATA1    ;EOF, MOVE TO NEXT
0083 0012 0280          CI R0,>3F00     ;,?
     0014 3F00
0084 0016 1310            JEQ FDATA2    ;Y, EVALUATE
0085                    *
0086 0018 2F97  ERR23   DATA ERROR+23   ;READ W/O DATA
0087                    *
0088 001A C229  FDATA1  MOV @DLC(9),R8  ;GET DLC
     001C 0000
0089 001E 0228          AI R8,-4        ;MOVE TO NEXT LINE
     0020 FFFC
0090 0022 8A48          C R8,@SLT(9)    ;ANY MORE LINES?
     0024 0000
0091 0026 1AF8            JL ERR23      ;N, ERROR
0092 0028 CA48          MOV R8,@DLC(9)  ;Y, SAVE DLC
     002A 001C'
0093 002C C218          MOV *R8,R8      ;GET PBC
0094 002E A229          A @BUS(9),R8    ;MAKE DISPLACEMENT INTO POINTER
     0030 0000
0095 0032 9838          CB *R8+,@DATXB  ;DATA STATEMENT?
     0034 0000
0096 0036 16F1            JNE FDATA1    ;N, CONTINUE LOOKING
0097                    *
0098 0038 0420  FDATA2  BLWP @EVSDZ     ;EVALUATE
     003A 0000
0099 003C 1004            JMP FDATA3    ;"
0100 003E 1003            JMP FDATA3    ;$
0101 0040 0420          BLWP @EVERZ     ;#
     0042 0000
0102 0044 05CB          INCT R11        ;RETURN 2(11)
0103                    *
0104 0046 0608  FDATA3  DEC R8          ;BACKUP TO DELIMITER
0105 0048 CA48          MOV R8,@DDM(9)  ;SAVE IN DELIMITER PTR
     004A 0008'
0106 004C C206          MOV R6,R8       ;RESTORE R8
0107 004E CA45          MOV R5,@DLIM(9) ;RESTORE DLIM
     0050 0004'
0108 0052 045B          RT
```

```
0110                *ENTRY POINT:
0111                *
0112                        DEF RDDY
0113                *READ COMMAND
0114                *
0115 0054 0420  RDDY    BLWP @EVSDZ      ;SEE IF STRING
     0056 003A'
0116 0058 2F98  ERR24     DATA ERROR+24  ;", ERROR
0117 005A 1010            JMP RDD2       ;$
0118 005C 0420          BLWP @EVARZ      ;WANTS #
     005E 0000
0119 0060 C1C2          MOV R2,R7        ;SAVE
0120 0062 06A0          BL @FDATA        ;FIND DATA
     0064 0000'
0121 0066 10F8            JMP ERR24      ;STRING, ERROR
0122 0068 CDF2          MOV *R2+,*R7+    ;STORE NUMBER
0123 006A CDF2          MOV *R2+,*R7+
0124 006C C5D2          MOV *R2,*R7
0125                *
0126 006E C029  RDD1    MOV @DLIM(9),R0 ;LOOK AT DELIMITER
     0070 0050'
0127 0072 0280          CI R0,>3F00      ;,?
     0074 3F00
0128 0076 13EE            JEQ RDDY       ;Y, DO AGAIN
0129 0078 0460          B @NLIN          ;N
     007A 0000
0130                *
0131 007C C1C2  RDD2    MOV R2,R7        ;SAVE ADR
0132 007E 06A0          BL @FDATA        ;LOOKING FOR STRING DATA
     0080 0000'
0133 0082 1001            JMP RDD3       ;FOUND
0134 0084 10E9          JMP ERR24
0135                *
0136 0086 0205  RDD3    LI R5,80         ;MAX OF EIGHTY CHARACTERS
     0088 0050
0137 008A 06A0          BL @BMVE         ;MOVE FROM *R2 TO *R7
     008C 0000
0138 008E 10EF          JMP RDD1
```

```
0140                    * ENTRY POINT:
0141                    *
0142                            DEF RNWY
0143                    *
0144                    *RENEW COMMAND
0145                    *
0146 0090 06A0  RNWY    BL @REST        ;RESTORE TO PROGRAM BEGINNING
     0092 0000
0147 0094 06A0          BL @CKEX        ;CHECK FOR EXPRESSION
     0096 0000
0148 0098 1006            JMP RNWY1     ;NONE
0149 009A 2EC1          EVFIX R1        ;FIX PARAMETER
0150 009C 06A0          BL @SFSN        ;LOOK FOR STATEMENT #
     009E 0000
0151 00A0 8E38          C *R8+,*R8+     ;MOVE BACK 1 LINE
0152 00A2 CA48          MOV R8,@DLC(9)  ;SAVE PLC
     00A4 002A'
0153 00A6 0460  RNWY1   B @NLIN0
     00A8 0000
0154                    END
NO ERRORS,      NO WARNINGS
```

MOV

MOV B

TO PROGRAM BEGINNING

OR EXPRESSION

AMETER
STATEMENT #

CK 1 LINE

```
                MOV  R8,R2     ; SAVE PBC
                MOV  R2,R8     ; RESTORE PBC
                B    @NLIN
```

```
BMVE    R  008C'   0017  0137
BUS     R  0030'   0031  0094
CKEX    R  0096'   0018  0147
DATXB   R  0034'   0032  0095
DDM     R  004A'   0033  0078  0105
DLC     R  00A4'   0034  0088  0092  0152
DLIM    R  0070'   0035  0077  0107  0126
ERR23      0018'   0086  0091
ERR24      0058'   0116  0121  0134
ERROR      2F80    0027  0086  0116
EVARZ   R  005E'   0019  0118
EVERZ   R  0042'   0020  0101
EVFIX              0026
EVSDZ   R  0056'   0021  0098  0115
FDATA      0000'   0076  0120  0132
FDATA1     001A'   0088  0079  0082  0096
FDATA2     0038'   0098  0084
FDATA3     0046'   0104  0099  0100
NLIN    R  007A'   0022  0129
NLIN0   R  00A8'   0023  0153
R0         0000    0080  0081  0083  0126  0127
R1         0001    0149
R11        000B    0102
R2         0002    0119  0122  0123  0124  0131
R5         0005    0077  0107  0136
R6         0006    0076  0106
R7         0007    0119  0122  0123  0124  0131
R8         0008    0076  0078  0081  0088  0089  0090  0092  0093  0093
                   0094  0095  0104  0105  0106  0151  0151  0152
RDD1       006E'   0126  0138
RDD2       007C'   0131  0117
RDD3       0086'   0136  0133
RDDY    D  0054'   0115  0112  0128
REST    R  0092'   0024  0146
RNWY    D  0090'   0146  0142
RNWY1      00A6'   0153  0148
SFSN    R  009E'   0025  0150
SLT     R  0024'   0036  0090
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.GOSUB
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.GOSUB
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.GOSUB
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'GOSUB'
0002              *
0003              * REVISION: 12/01/78 1.01          IDT 'GOSUB'
0004              *
0005              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0006              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0007              *
0008              * ROUTINE LIST:
0009              *
0010              *       GOTY              ;GOTO COMMAND
0011              *       GOSY              ;GOSUB COMMAND
0012              *       RTNY              ;RETURN COMMAND
0013              *       POPY              ;POP COMMAND
0014              *
0015              * COPY: (NONE)
0016              *
0017              * MARCO: (NONE)
0018              *
0019              * EXTERNAL ROUTINES:
0020              *
0021                      REF SFSN          ;SEARCH FOR STATEMENT NUMBER
0022                      REF NLIN0         ;ENTRIES IN MULTIPLEXOR
0023                      REF LINE0,LINE2
0024                      REF LINE5,LINE6
0025                      REF CKEX          ;CHECK FOR EXPRESSION
0026                      DXOP EVFIX,11     ;EVALUATE AND FIX
0027    2F80  ERROR       EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0028    2FA0  ERROR2      EQU ERROR+>20
0029              *
0030              * EXTERNAL DATA:
0031              *
0032                      REF ELNM          ;ERROR LINE NUMBER
0033                      REF FNS           ;FOR/NEXT STACK
0034                      REF GSC           ;GOSUB STACK COUNTER
0035                      REF GSS           ;GOSUB STACK
0036                      REF PLC           ;PROGRAM LINE COUNTER
0037                      REF SLT           ;STATEMENT LOCATION TABLE
0038                      REF BUS           ;BEGINNING USER STORAGE
0039                      REF C4            ;>0004
0040              *
0041              * MODULE EQUATES:  (NONE)
0042              *
0043              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0046            * ABSTRACT:
0047            *
0048            *       EXECUTE GOTO,GOSUB, AND SYSTEM GOSUB
0049            *       COMMANDS.  PROVIDE FOR ON COMMAND
0050            *
0051            *       STACK FORMAT:   PLC
0052            *                       PBC
0053            *
0054            *                       PLC
0055            *                       PBC
0056            *
0057            *                       ...
0058            *
0059            *       IF PBC = 0, GOTO NEXT LINE, OTHERWISE,
0060            *       CONTINUE EXECUTION ON THE SAME LINE
0061            *       AS THE CALL WAS MADE.  (FOR ON
0062            *       COMMAND WHERE YOU DO NOT WANT TO
0063            *       JUMP BACK INTO THE PARAMETER LIST.)
0064            *
0065            * CALLING SEQUENCE:
0066            *
0067            *       B @GOSB1
0068            *       B @GOSY
0069            *       B @GOTY
0070            *
0071            *       EXIT TO LINE0 IN MUX
0072            *
0073            * EXCEPTIONS AND CONDITIONS:
0074            *
0075            *       STACK OVERFLOW
0076            *       STACK UNDERFLOW
0077            *       NO SUCH LINE #
0078            *
0079            * EXTERNAL ROUTINE LIST:
0080            *
0081            *      (SEE EXTERNAL ROUTINES)
0082            *
0083            * LOCAL DATA:
0084            *
0085            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0087                    * ENTRY POINT:
0088                    *
0089                            DEF GOSY        ;GOSUB COMMAND
0090                            DEF GOS1,GOS2
0091                            DEF GOSB1       ;SYSTEM ENTRY
0092                            DEF GOSON       ;ON COMMAND ENTRY
0093                            DEF GOTY        ;GOTO ENTRY
0094                    *
0095 0000 C1A9  GOSB1   MOV @PLC(9),R6  ;GET ERROR LINE #
     0002 0000
0096 0004 CA66          MOV @-2(6),@ELNM(9)
     0006 FFFE
     0008 0000
0097 000A 04C6          CLR R6          ;SET FOR FIRST OF NEXT LINE
0098 000C 101F          JMP GOS1
0099                    *
0100                    *GOTO-GOSUB ROUTINE
0101                    *
0102 000E C188  GOSY    MOV R8,R6       ;SET TO SAVE PBC
0103 0010 05C6          INCT R6
0104 0012 04C0          CLR R0
0105 0014 D036          MOVB *R6+,R0    ;GET DELIMITER
0106 0016 1308            JEQ GOSY1     ;EOL
0107 0018 0280          CI R0,>3C00     ;:?
     001A 3C00
0108 001C 1311            JEQ GOSON     ;Y
0109 001E 0280          CI R0,>4700     ;!?
     0020 4700
0110 0022 1302            JEQ GOSY1     ;Y
0111 0024 2FA0  GOSYE   DATA ERROR2,37  ;N, ERROR
     0026 0025
0112                    *
0113 0028 C1A9  GOSY1   MOV @PLC(9),R6  ;MOVE TO NEXT LINE
     002A 0002'
0114 002C 0226          AI R6,-4
     002E FFFC
0115 0030 8A46          C R6,@SLT(9)    ;DONE?
     0032 0000
0116 0034 1AF7            JL GOSYE      ;Y, ERROR
0117 0036 CA46          MOV R6,@PLC(9)  ;N, UPDATE
     0038 002A'
0118 003A C196          MOV *R6,R6
0119 003C A1A9          A @BUS(9),R6
     003E 0000
0120                    *
0121      0040' GOTY    EQU $           ;GOTO ENTRY
0122 0040 D078  GOSON   MOVB *R8+,R1    ;GET COMMAND
0123 0042 06C1          SWPB R1
0124 0044 D078          MOVB *R8+,R1
0125 0046 06C1          SWPB R1
0126 0048 0922          SRL R2,2        ;GOTO?
0127 004A 130A            JEQ GOS2
0128                    *
0129 004C C0E9  GOS1    MOV @GSC(9),R3  ;N, GOSUB
     004E 0000
0130 0050 8A43          C R3,@FNS(9)    ;ROOM?
     0052 0000
0131 0054 1409            JHE ERR11     ;N, ERROR
0132 0056 CCC6          MOV R6,*R3+     ;Y, SAVE PBC OR NULL AND PLC
0133 0058 CCE9          MOV @PLC(9),*R3+
```

```
      005A 0038'
0134  005C CA43          MOV R3,@GSC(9)   ;UPDATE GSC
      005E 004E'
0135                *
0136  0060 06A0  GOS2    BL @SFSN         ;SEARCH FOR STATEMENT #
      0062 0000
0137  0064 0460          B @LINE0
      0066 0000
0138                *
0139  0068 2F88  ERR11   DATA ERROR+11    ;STACK OVERFLOW
0140                *
0141  006A 2F8C  ERR12   DATA ERROR+12    ;STACK UNDERFLOW
```

```
0144                    * ABSTRACT:
0145                    *
0146                    *       THE RETURN COMMAND WILL POP THE
0147                    *       GOSUB STACK (GSS) AND EFFECT A
0148                    *       TRANSFER TO THAT LOCATION.
0149                    *
0150                    * CALLING SEQUENCE:
0151                    *
0152                    *       B @RTNY
0153                    *
0154                    *       EXIT TO MULTIPLEXOR
0155                    *
0156                    * EXCEPTIONS AND CONDITIONS:
0157                    *
0158                    *       STACK UNDERFLOW
0159                    *
0160                    * EXTERNAL ROUTINES:  (NONE)
0161                    *
0162                    * LOCAL DATA:  (NONE)
0163                    *
0164                    * ENTRY POINT:
0165                    *
0166                            DEF RTNY
0167                    *
0168 006C C0E9  RTNY    MOV @GSC(9),R3  ;SEE IF STACK EMPTY
     006E 005E'
0169 0070 8A43          C R3,@GSS(9)    ;EMPTY?
     0072 0000
0170 0074 12FA            JLE ERR12     ;Y
0171 0076 0643          DECT R3         ;N, POP PLC
0172 0078 CA53          MOV *R3,@PLC(9)
     007A 005A'
0173 007C 0643          DECT R3         ;POP PBC
0174 007E CA43          MOV R3,@GSC(9)  ;UPDATE GSC
     0080 006E'
0175                    *
0176 0082 C213  RTN1    MOV *R3,R8      ;GET PBC
0177 0084 1302            JEQ RTRN2     ;0, GOTO NEXT LINE (ON)
0178 0086 0460          B @LINE2
     0088 0000
0179                    *
0180 008A 0460  RTRN2   B @LINES
     008C 0000
```

```
0183              * ABSTRACT:
0184              *
0185              *       POP THE GOSUB STACK WITHOUT EFFECTING
0186              *       PROGRAM EXECUTION.  (NO TRANSFER)
0187              *
0188              * CALLING SEQUENCE:
0189              *
0190              *       B @POPY
0191              *
0192              *       EXIT TO NLIN0
0193              *
0194              * EXCEPTIONS AND CONDITIONS:
0195              *
0196              *       STACK UNDERFLOW
0197              *
0198              * EXTERNAL ROUTINES:  (NONE)
0199              *
0200              * LOCAL DATA:  (NONE)
0201              *
0202              * ENTRY POINT:
0203              *
0204                      DEF POPY
0205              *
0206 008E 8A69   POPY    C @GSC(9),@GSS(9)
     0090 0080'
     0092 0072'
0207 0094 12EA             JLE ERR12
0208 0096 6A60           S @C4,@GSC(9)
     0098 0000
     009A 0090'
0209 009C 0460           B @NLIN0
     009E 0000
0210              *
0211                      END
NO ERRORS,     NO WARNINGS
```

```
$          00A0'          0121
BUS     R  003E'    0038  0119
C4      R  0098'    0039  0208
CKEX    R              0025
ELNM    R  0008'    0032  0096
ERR11      0068'    0139  0131
ERR12      006A'    0141  0170  0207
ERROR      2F80     0027  0028  0139  0141
ERROR2     2FA0     0028  0111
EVFIX              0026
FNS     R  0052'    0033  0130
GOS1    D  004C'    0129  0090  0098
GOS2    D  0060'    0136  0090  0127
GOSB1   D  0000'    0095  0091
GOSON   D  0040'    0122  0092  0108
GOSY    D  000E'    0102  0089
GOSY1      0028'    0113  0106  0110
GOSYE      0024'    0111  0116
GOTY    D  0040'    0121  0093
GSC     R  009A'    0034  0129  0134  0168  0174  0206  0208
GSS     R  0092'    0035  0169  0206
LINE0   R  0066'    0023  0137
LINE2   R  0088'    0023  0178
LINE5   R  008C'    0024  0180
LINE6   R              0024
NLIN0   R  009E'    0022  0209
PLC     R  007A'    0036  0095  0113  0117  0133  0172
POPY    D  008E'    0206  0204
R0         0000           0104  0105  0107  0109
R1         0001           0122  0123  0124  0125
R2         0002           0126
R3         0003           0129  0130  0132  0133  0134  0168  0169  0171  0172
                          0173  0174  0176
R6         0006           0095  0097  0102  0103  0105  0113  0114  0115  0117
                          0118  0118  0119  0132
R8         0008           0102  0122  0124  0176
RTN1       0082'    0176
RTNY    D  006C'    0168  0166
RTRN2      008A'    0180  0177
SFSN    R  0062'    0021  0136
SLT     R  0032'    0037  0115
*
```

```
SOURCE ACCESS NAME=      DBASIC.REL4.BASIC.SOURCE.RUN
OBJECT ACCESS NAME=      DBASIC.REL4.BASIC.OBJECT.RUN
LISTING ACCESS NAME=     DBASIC.REL4.BASIC.LIST.RUN
ERROR ACCESS NAME=
OPTIONS=                 X
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'RUN'
0002                *   23 APRIL 1979
0003                    DEF LINE,LINE0,LINE2,LINE5
0004                    DEF NLIN,NLIN0
0005                    DEF RUNP,RUNP1
0006                    DEF CLRV,REST
0007                    DEF B20,B3F,C4A00
0008                    DEF NXTXB,DATXB
0009                *
0010                    REF NVS,GSS,VNT,VDT
0011                    REF IMKY,IRTY,TRPY,RANDS
0012                    REF GOS1,INTFLG,BAUD,GOTY
0013                    REF ITAB,INTSB,POPY
0014                    REF DDM,DLC,GSC
0015                    REF BASY,BITY,CLLY,CRBY
0016                    REF CRFY,DEFY,DIMY
0017                    REF ERRY,ESCY,FORY
0018                    REF GOSY,IFY,INPY,LDPY
0019                    REF LETY,MEMY,NEWY,NOEY
0020                    REF NXTY,ONY,PRTY
0021                    REF RANY,RDDY,RNWY,RTNY
0022                    REF SAVY,STPY,TIMY,UNTY
0023                    REF BUS,DLIM,LEC,MODE
0024                    REF PLC,SLT,ELSF
0025                    REF CRLF,GETC,TYPC
0026                    REF OCWNDO          ;HJC 27 MAR 79
0027                    REF STPYA           ;SPR 35,HJC 1 MAY 79
0028                    REF MWDY            ;HJC 2 MAY 1979
0029                *
0030       2F80     ERROR  EQU >2F80
0031       2FA0     ERROR2 EQU ERROR+>20
0032                *
0033                *CLEAR R1 THRU EUS
0034                *
0035 0000 CA59      CLRV    MOV *R9,@NVS(9)
     0002 0000
0036 0004 CA69              MOV @GSS(9),@GSC(9)
     0006 0000
     0008 0000
0037                *
0038 000A 04F1              CLR *R1+
0039 000C 0281              CI R1,RANDS     ;DONE?
     000E 0000
0040 0010 1AFC                JL  $-6       ;N
0041                *
0042                *RESET DATA PTR
0043                *
0044 0012 CA69      REST    MOV @VNT(9),@DLC(9)
     0014 0000
     0016 0000
0045 0018 04E9              CLR @DDM(9)
     001A 0000
0046 001C 045B              RT
0047                *
0048                *RUN PROGRAM
0049                *
0050 001E 0729      RUNP    SETO @MODE(9)   ;SET MODE TO RUN
     0020 0000
0051 0022 06A0              BL @TYPC        ;OUT CRLF
     0024 0000
```

```
0052                    *
0053 0026 CA69  RUNP1   MOV @VNT(9),@PLC(9)
     0028 0014'
     002A 0000
0054                    *
0055                    *CLEAR COMMAND
0056                    *
0057 002C C069  CLRY    MOV @VDT(9),R1  ;CLEAR VDT THRU EUS
     002E 0000
0058 0030 06A0          BL @CLRV
     0032 0000'
0059 0034 1010          JMP LINE
0060                    *
0061                    *MULTIPLEXOR
0062                    *
0063 0036 0280  NLIN1   CI R0,>3C00      ;:?
     0038 3C00
0064 003A 1324            JEQ LINE2     ;Y
0065 003C 0280          CI R0,>3B00     ;THEN?
     003E 3B00
0066 0040 1321            JEQ LINE2     ;Y
0067 0042 0280          CI R0,>4700     ;!?
     0044 4700
0068 0046 1307            JEQ LINE      ;Y, CONTINUE
0069                    *
0070 0048 2FA0  ERR37   DATA ERROR2,37  ;N, ILLEGAL DELIMITER
     004A 0025
0071                    *
0072                    *NEXT LINE OR ":
0073                    *
0074 004C DA78  NLIN0   MOVB *R8+,@DLIM(9)
     004E 0000
0075                    *
0076 0050 C029  NLIN    MOV @DLIM(9),R0 ;LOOK AT DELIMITER
     0052 004E'
0077 0054 16F0            JNE NLIN1
0078                    *
0079                    *NEXT LINE
0080                    *
0081 0056 C029  LINE    MOV @MODE(9),R0 ;LOOK AT MODE
     0058 0020'
0082 005A 131B            JEQ LINE3     ;DONE
0083      005C' LINE5   EQU $
0084 005C C229          MOV @PLC(9),R8  ;GET PROGRAM LINE COUNTER
     005E 002A'
0085 0060 0228          AI  R8,-4       ;MOVE TO NEXT LINE
     0062 FFFC
0086 0064 8A48          C R8,@SLT(9)    ;ANY MORE STATEMENTS?
     0066 0000
0087 0068 1A06            JL LINE4      ;N
0088 006A CA48  LINE0   MOV R8,@PLC(9)  ;Y, UPDATE PLC
     006C 005E'
0089 006E 06A0          BL @GETC        ;Y, GET CHARACTER
     0070 0000
0090 0072 1005            JMP LINE1     ;NOTHING
0091 0074 1002            JMP LINE1A    ;R0=CHARACTER
0092 0076 0460  LINE4   B @STPY         ;ESCAPE
     0078 0000
0093                    *
0094 007A CA40  LINE1A  MOV R0,@LEC(9)  ;SAVE CHARACTER
```

```
     007C 0000
0095                 *
0096 007E C218   LINE1    MOV *R8,R8        ;GET PBC
0097 0080 A229            A @BUS(9),R8
     0082 0000
0098                 *
0099 0084 06A0   LINE2    BL  @UCWNDO       ;BRANCH TO OPEN AND CLOSE WINDOW INS
     0086 0000
0100                 *
0101                 *
0102 0088 D088   LINE2B   MOVB *R8+,R2      ;GET TYPE
0103 008A 0972            SRL R2,7          ;GET INDEX
0104 008C C2E2            MOV @MXLST-2(2),R11
     008E 009C'
0105 0090 045B            B *R11            ;GOTO COMMAND
0106                 *
0107 0092 0460   LINE3    B @CRLF
     0094 0000
0108                 *
0109                 *ELSE COMMAND
0110                 *
0111 0096 C029   ELSY     MOV @ELSF(9),R0         ;LOOK AT ELSE FLAG
     0098 0000
0112 009A 16F4              JNE LINE2             ;SET,CONTINUE TO EXECUTE
0113 009C 10DC            JMP LINE                ;RESET, IGNORE LINE
```

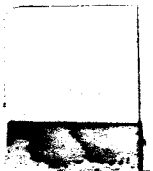
```
0115 009E 0000  MXLST   DATA GOTY         ;GOTO
0116 00A0 0000          DATA GOSY         ;GOSUB
0117 00A2 0096'         DATA ELSY         ;ELSE
0118 00A4 0056'         DATA LINE         ;REM
0119 00A6 0000          DATA FORY         ;FOR
0120 00A8 0000          DATA LETY         ;(LET)
0121 00AA 0056'         DATA LINE         ;DATA
0122      0007  DATB    EQU $-MXLST/2
0123 00AC 0000          DATA NXTY         ;NEXT
0124      0008  NXTB    EQU $-MXLST/2
0125 00AE 0000          DATA ERRY         ;ERROR
0126 00B0 0000          DATA PRTY         ;PRINT
0127 00B2 0000          DATA CLLY         ;CALL
0128 00B4 0000          DATA LDPY         ;LOAD
0129 00B6 0000          DATA INPY         ;INPUT
0130 00B8 0000          DATA RDDY         ;READ
0131 00BA 0000          DATA RNWY         ;RESTOR
0132 00BC 0000          DATA RTNY         ;RETURN
0133 00BE 0000          DATA STPYA        ;STOP     SPR 35 ,HJC,1 MAY 1979
0134 00C0 0000          DATA UNTY         ;UNIT
0135 00C2 0000          DATA TIMY         ;TIME
0136 00C4 0000          DATA SAVY         ;SAVE
0137 00C6 0000          DATA BASY         ;BASE
0138 00C8 0000          DATA ESCY         ;ESCAPE
0139 00CA 0000          DATA NOEY         ;NOESC
0140 00CC 0000          DATA RANY         ;RANDOM
0141 00CE 0000          DATA IMKY         ;IMASK
0142 00D0 0000          DATA IRTY         ;IRET
0143 00D2 0000          DATA TRPY         ;TRAP
0144 00D4 0000          DATA BAUD         ;BAUD
0145 00D6 0000          DATA POPY         ;POP
0146 00D8 0000          DATA DIMY         ;DIM
0147 00DA 00A8'         DATA LETY         ;LET
0148 00DC 0000          DATA ONY          ;ON
0149 00DE 0000          DATA IFY          ;IF
0150 00E0 0000          DATA DEFY         ;DEF
0151 00E2 0000          DATA NEWY         ;NEW
0152 00E4 0078'         DATA STPY         ;END
0153              *
0154 00E6 0000          DATA BITY         ;BIT(
0155 00E8 0000          DATA CRBY         ;CRB(
0156 00EA 0000          DATA CRFY         ;CRF(
0157 00EC 0000          DATA MEMY         ;MEM(
0158 00EE 0000          DATA MWDY         ;MWD(
0159              *
0160 00F0 4A00  C4A00   DATA >4A00
0161 00F2   20  B20     BYTE >20
0162 00F3   3F  B3F     BYTE >3F
0163 00F4   08  NXTXB   BYTE NXTB
0164 00F5   07  DATXB   BYTE DATB
0165 00F6               EVEN
0166                    END
NO ERRORS
```

LABEL		VALUE	DEFN	REFERENCES
$		00F6'		0040 0083 0122 0124
B20	D	00F2'	0161	0007
B3F	D	00F3'	0162	0007
BASY	R	00C6'	0015	0137
BAUD	R	00D4'	0012	0144
BITY	R	00E6'	0015	0154
BUS	R	0082'	0023	0097
C4A00	D	00F0'	0160	0007
CLLY	R	00B2'	0015	0127
CLRV	D	0000'	0035	0006 0058
CLRY		002C'	0057	
CRBY	R	00E8'	0015	0155
CRFY	R	00EA'	0016	0156
CRLF	R	0094'	0025	0107
DATB		0007	0122	0164
DATXB	D	00F5'	0164	0008
DDM	R	001A'	0014	0045
DEFY	R	00E0'	0016	0150
DIMY	R	00D8'	0016	0146
DLC	R	0016'	0014	0044
DLIM	R	0052'	0023	0074 0076
ELSF	R	0098'	0024	0111
ELSY		0096'	0111	0117
ERR37		0048'	0070	
ERROR		2F80	0030	0031
ERROR2		2FA0	0031	0070
ERRY	R	00AE'	0017	0125
ESCY	R	00C8'	0017	0138
FORY	R	00A6'	0017	0119
GETC	R	0070'	0025	0089
GOS1	R		0012	
GOSY	R	00A0'	0018	0116
GOTY	R	009E'	0012	0115
GSC	R	0008'	0014	0036
GSS	R	0006'	0010	0036
IFY	R	00DE'	0018	0149
IMKY	R	00CE'	0011	0141
INPY	R	00B6'	0018	0129
INTFLG	R		0012	
INTSB	R		0013	
IRTY	R	00D0'	0011	0142
ITAB	R		0013	
LDPY	R	00B4'	0018	0128
LEC	R	007C'	0023	0094
LETY	R	00DA'	0019	0120 0147
LINE	D	0056'	0081	0003 0059 0068 0113 0118 0121
LINE0	D	006A'	0088	0003
LINE1		007E'	0096	0090
LINE1A		007A'	0094	0091
LINE2	D	0084'	0099	0003 0064 0066 0112
LINE2B		0088'	0102	
LINE3		0092'	0107	0082
LINE4		0076'	0092	0087
LINE5	D	005C'	0083	0003
MEMY	R	00EC'	0019	0157
MODE	R	0058'	0023	0050 0081
MWDY	R	00EE'	0028	0158
MXLST		009E'	0115	0104 0122 0124
NEWY	R	00E2'	0019	0151
NLIN	D	0050'	0076	0004

NLIN0	D	004C'	0074	0004								
NLIN1		0036'	0063	0077								
NOEY	R	00CA'	0019	0139								
NVS	R	0002'	0010	0035								
NXTB		0008	0124	0163								
NXTXB	D	00F4'	0163	0008								
NXTY	R	00AC'	0020	0123								
OCWNDO	R	0086'	0026	0099								
ONY	R	00DC'	0020	0148								
PLC	R	006C'	0024	0053	0084	0088						
POPY	R	00D6'	0013	0145								
PRTY	R	00B0'	0020	0126								
R0		0000		0063	0065	0067	0076	0081	0094	0111		
R1		0001		0038	0039	0057						
R11		000B		0104	0105							
R2		0002		0102	0103							
R8		0008		0074	0084	0085	0086	0088	0096	0096	0097	0102
R9		0009		0035								
RANDS	R	000E'	0011	0039								
RANY	R	00CC'	0021	0140								
RDDY	R	00B8'	0021	0130								
REST	D	0012'	0044	0006								
RNWY	R	00BA'	0021	0131								
RTNY	R	00BC'	0021	0132								
RUNP	D	001E'	0050	0005								
RUNP1	D	0026'	0053	0005								
SAVY	R	00C4'	0022	0136								
SLT	R	0066'	0024	0086								
STPY	R	00E4'	0022	0092	0152							
STPYA	R	00BE'	0027	0133								
TIMY	R	00C2'	0022	0135								
TRPY	R	00D2'	0011	0143								
TYPC	R	0024'	0025	0051								
UNTY	R	00C0'	0022	0134								
VDT	R	002E'	0010	0057								
VNT	R	0028'	0010	0044	0053							

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.ON
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.ON
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.ON
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```



0001

```
                IDT  'ON'
0002                 *
0003                 * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *
0006                 * ROUTINE LIST:
0007                 *
0008                 *       ONY               ;ON COMMAND
0009                 *
0010                 * COPY: (NONE)
0011                 *
0012                 * MACRO: (NONE)
0013                 *
0014                 * EXTERNAL ROUTINES:
0015                 *
0016                       REF GOSON          ;GOSUB ENTRY
0017                       REF LINE           ;EXIT ENTRY TO MULTIPLEXOR
0018                       DXOP EVFIX,11      ;EVALUATE AND FIX
0019          2F80   ERROR  EQU >2F80                  ;XOP XX,14  (ERROR CALL)
0020                 *
0021                 * EXTERNAL DATA:
0022                 *
0023                       REF B3F            ;>3F
0024                 *
0025                 * MODULE EQUATES:  (NONE)
0026                 *
0027                 * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0030            * ABSTRACT:
0031            *
0032            *       ON COMMAND
0033            *
0034            * CALLING SEQUENCE:
0035            *
0036            *       B @ONY
0037            *
0038            *       EXIT TO LINE OR GOSON
0039            *
0040            * EXCEPTIONS AND CONDITIONS:
0041            *
0042            *       SYNTAX ERROR
0043            *       EVALUATION ERRORS
0044            *
0045            * EXTERNAL ROUTINE LIST:
0046            *
0047            *      (SEE EXTERNAL ROUTINES)
0048            *
0049            * LOCAL DATA:
0050            *
0051            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0053                    * ENTRY POINT:
0054                    *
0055                          DEF ONY
0056                    *
0057 0000 2EC1  ONY       EVFIX R1          ;EVALUATE & FIX
0058 0002 0280            CI R0,>3B00       ;THEN?
     0004 3B00
0059 0006 1611              JNE ERR1        ;N, SYNTAX ERROR
0060                    *
0061 0008 D0B8  ON1       MOVB *R8+,R2      ;GET TYPE
0062 000A 0972            SRL R2,7          ;READY
0063 000C 0282            CI R2,4           ;GOTO OR GOSUB?
     000E 0004
0064 0010 150C              JGT ERR1        ;N, ERROR
0065 0012 04C6            CLR R6            ;Y, TERMINATE LINE
0066                    *
0067 0014 0601  ON2       DEC R1            ;COUNT
0068 0016 1105              JLT ON3         ;DONE
0069 0018 1306              JEQ ON4         ;DO GOTO
0070 001A 05C8            INCT R8           ;MOVE TO NEXT
0071 001C 9838            CB *R8+,@B3F      ;,?
     001E 0000
0072 0020 13F9              JEQ ON2         ;Y, CONTINUE
0073 0022 0460  ON3       B @LINE           ;DROP THRU
     0024 0000
0074                    *
0075 0026 0460  ON4       B @GOSON
     0028 0000
0076                    *
0077 002A 2F81  ERR1      DATA ERROR+1      ;SYNTAX ERROR
0078                          END
NO ERRORS,      NO WARNINGS
```

B3F	R	001E'	0023	0071		
ERR1		002A'	0077	0059	0064	
ERROR		2F80	0019	0077		
EVFIX			0018			
GOSON	R	0028'	0016	0075		
LINE	R	0024'	0017	0073		
ON1		0008'	0061			
ON2		0014'	0067	0072		
ON3		0022'	0073	0068		
ON4		0026'	0075	0069		
ONY	D	0000'	0057	0055		
R0		0000		0058		
R1		0001		0057	0067	
R2		0002		0061	0062	0063
R6		0006		0065		
R8		0008		0061	0070	0071

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.LET
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.LET
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.LET
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT 'LET'
0002            *
0003            * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004            *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005            *           30 MAY 79          ;COPY AND UPDATE TO RELS
0006            * ROUTINE LIST:
0007            *
0008            *       LETY              ;LET COMMAND
0009            *
0010            * COPY: (NONE)
0011            *
0012            * MACRO: (NONE)
0013            *
0014            * EXTERNAL ROUTINES:
0015            *
0016                    REF EVARZ         ;EVALUATE VARIABLE
0017                    REF EVERZ         ;EVALUATE EXPRESSION
0018                    REF EVSDZ         ;EVALUATE STRING
0019                    REF CKEX          ;CHECK FOR EXPRESSION
0020                    REF CVDIFZ        ;CONVERT DECIMAL TO INTEGER/FP
0021                    REF NLIN          ;EXIT ENTRY TO MULTIPLEXOR
0022                    DXOP LOADF,0      ;LOAD FPAC
0023                    DXOP STORE,1      ;STORE FPAC
0024                    DXOP SCALE,6      ;SCALE FPAC
0025                    DXOP NORMAL,7     ;NORMALIZE FPAC
0026                    DXOP EVFIX,11     ;EVALUATE AND FIX
0027                    DXOP OUTFP,12     ;OUT FLOATING POINT #
0028                    DXOP OUTINT,13    ;OUT INTEGER
0029    2F80    ERROR   EQU >2F80         ;XOP XX,14  (ERROR CALL)
0030            *
0031            * EXTERNAL DATA:
0032            *
0033                    REF FFLG          ;FORMATTING FLAG
0034                    REF IOB           ;I/O BUFFER PTR
0035            *
0036            * MODULE EQUATES:  (NONE)
0037            *
0038            * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0041            * ABSTRACT:
0042            *
0043            *       THE LET COMMAND IS THE ASSIGNMENT STATEMENT
0044            *       AND CAN HAVE ANY OF THE FOLLOWING FORMS:
0045            *
0046            *       A[-]=B[-]                   NUMERIC ASSIGNMENT
0047            *       $A[-]=$B[-]                 ASSIGNMENT
0048            *       $A[-]=$B[-],N               PICK
0049            *       $A[-]=$B[-];N               REPLACE
0050            *       $A[-]=$B[-]+$C[0]           CONCATENATE
0051            *       $A[-]=/N                    DELETE
0052            *       $A[-]=/$B[-]                INSERT
0053            *       $A[-]=N                     CONVERT # TO ASCII
0054            *       $A[-]=#$B[-],N              CONVERT # TO ASCII FORMATTED
0055            *       $A[-]=%N                    CONVERT BYTE
0056            *       N=$A[-],E                   CONVERT ASCII TO #
0057            *
0058            * CALLING SEQUENCE:
0059            *
0060            *       B @LETY
0061            *
0062            *       EXIT TO NLIN
0063            *
0064            * EXCEPTIONS AND CONDITIONS:
0065            *
0066            *       MISSING ASSIGNMENT OPERATOR
0067            *       EVALUATION ERRORS
0068            *
0069            * EXTERNAL ROUTINE LIST:
0070            *
0071            *      (SEE EXTERNAL ROUTINES)
0072            *
0073            * LOCAL DATA:
0074            *
0075            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0077                    * ENTRY POINT:
0078                    *
0079                          DEF LETY
0080                    *
0081 0000 0420  LETY      BLWP @EVSDZ
     0002 0000
0082 0004 2F81              DATA ERROR+1    ;"
0083 0006 101A              JMP LET3        ;$
0084 0008 0420            BLWP @EVARZ       ;EVALUATE VARIABLE
     000A 0000
0085 000C 0280            CI R0,>5600       ;=?
     000E 5600
0086 0010 1618              JNE ERR36       ;N, ERROR
0087 0012 C0C2            MOV R2,R3
0088 0014 06A0            BL @CKEX          ;CHECK FOR EXPRESSION
     0016 0000
0089 0018 1007              JMP LET1        ;PROBLEM
0090 001A 0420            BLWP @EVERZ       ;EVALUATE
     001C 0000
0091                    *
0092                    *  [VAR]=[EXP]
0093                    *
0094 001E CCF2            MOV *R2+,*R3+     ;STORE RESULTS
0095 0020 CCF2            MOV *R2+,*R3+
0096 0022 C4D2            MOV *R2,*R3
0097 0024 0460  LETR      B @NLIN           ;GOTO NEXT LINE
     0026 0000
0098                    *
0099                    *  [VAR]=←
0100                    *
0101 0028 0420  LET1      BLWP @EVSDZ       ;LOOK FOR STRING
     002A 0002'
0102 002C 1002              JMP LET2        ;"
0103 002E 1001              JMP LET2        ;$
0104 0030 2F81            DATA ERROR+1      ;SYNTAX ERROR
0105                    *
0106                    *  [VAR]=[$VAR]←
0107                    *
0108 0032 C102  LET2      MOV R2,R4         ;SAVE ADR
0109 0034 0280            CI R0,>3F00       ;,?
     0036 3F00
0110 0038 1352              JEQ LET6        ;Y, CONVERT ASCII TO BINARY
0111                    *
0112 003A 2F87  ERR7      DATA ERROR+7      ;N, EXPECTING OPERATER
0113                    *
0114                    *  $←            STRING ENTRY
0115                    *
0116 003C 0280  LET3      CI R0,>5600       ;=?
     003E 5600
0117 0040 1309              JEQ LET3A       ;Y
0118 0042 2FA0  ERR36     DATA ERROR+>20    ;MISSING ASSIGNMENT OPERATER
0119 0044 0024              DATA 36
```

```
0121                    *BYTE ASSIGNMENT
0122                    *
0123                    *       $A[-]=%N...
0124                    *
0125 0046 2EC1  LET11   EVFIX R1        ;GET BYTE
0126 0048 06C1          SWPB R1
0127                    *
0128 004A DDC1  LET11A  MOVB R1,*R7+    ;SAVE #
0129 004C 0280          CI R0,>4200     ;%?
     004E 4200
0130 0050 13FA            JEQ LET11     ;Y
0131 0052 10E8          JMP LETR        ;RETURN
0132                    *
0133                    *MULTIPLEXOR FOR STRING ASSIGNMENTS
0134                    *  $[VAR]=+
0135                    *
0136 0054 C1C2  LET3A   MOV R2,R7
0137 0056 0420  LET3E   BLWP @EVSDZ     ;GET STRING OR $VAR
     0058 002A'
0138 005A 101E            JMP LET4      ;"
0139 005C 101D            JMP LET4      ;$
0140 005E D038          MOVB *R8+,R0    ;GET BYTE
0141 0060 0280          CI R0,>4200     ;%?
     0062 4200
0142 0064 13F0            JEQ LET11     ;Y
0143 0066 0280          CI R0,>5E00     ;/?
     0068 5E00
0144 006A 134A            JEQ LET12     ;Y, INSERT
0145 006C 0280          CI R0,>3E00     ;#?
     006E 3E00
0146 0070 160B            JNE LET3C     ;N
0147                    *
0148                    *CONVERT NUMBER TO ASCII
0149                    *
0150                    *       $A[-]=N
0151                    *       $A[-]=#$B[-],N
0152                    *
0153 0072 0420          BLWP @EVSDZ     ;Y, GET FORMAT
     0074 0058'
0154 0076 1002            JMP LET3B
0155 0078 1001            JMP LET3B
0156 007A 2F8E  ERR14   DATA ERROR+14   ;ERROR, EXPECTING STRING
0157                    *
0158 007C 0280  LET3B   CI R0,>3F00     ;,?
     007E 3F00
0159 0080 16DC            JNE ERR7      ;N, ERROR
0160 0082 CA42          MOV R2,@FFLG(9) ;Y, SET TO FORMAT
     0084 0000
0161 0086 1001          JMP LET3D
0162                    *
0163 0088 0608  LET3C   DEC R8          ;N, BACKUP
0164                    *
0165 008A 0420  LET3D   BLWP @EVERZ     ;EVALUATE EXPRESSION
     008C 001C'
0166 008E 2F12          OUTFP *R2       ;CONVERT #
0167 0090 04E9          CLR @FFLG(9)    ;CLEAR FORMATTING
     0092 0084'
0168 0094 75D7          SB *R7,*R7      ;END STRING
0169 0096 10C6          JMP LETR
```

```
0171                    *ASSIGN,PICK,REPLACE,CONCATENATE
0172                    *
0173                    *       $A[-]=$B[-]
0174                    *       $A[-]=$B[-],N
0175                    *       $A[-]=$B[-];N
0176                    *       $A[-]=$B[-]+$C[-]
0177                    *
0178 0098 0705  LET4    SETO R5         ;NO NULL
0179 009A 0280          CI R0,>4000     ;;?
     009C 4000
0180 009E 1305            JEQ LET4A     ;Y, REPLACEMENT
0181 00A0 04C5          CLR R5          ;SET TO NULL
0182 00A2 0280          CI R0,>3F00     ;,?
     00A4 3F00
0183 00A6 1605            JNE LET4B     ;N, DIRECT ASSIGNMENT
0184 00A8 0585          INC R5
0185                    *
0186 00AA 2EC1  LET4A   EVFIX R1        ;EVALUATE COUNT
0187 00AC C041          MOV   R1,R1     ;CHECK CONDITION
0188 00AE 1504          JGT   LET4C     ; >0 THEN DO PICK
0189 00B0 10B9          JMP   LETR      ;ELSE IGNORE
0190                    *
0191 00B2 C059  LET4B   MOV *R9,R1      ;GET MAX MOVE
0192 00B4 6047          S R7,R1
0193 00B6 0601          DEC  R1
0194                    *
0195 00B8 C0C7  LET4C   MOV R7,R3       ;MARK
0196                    *
0197 00BA D132  LET4D   MOVB *R2+,R4    ;MOVE
0198 00BC 1602            JNE LET4I     ;CHARACTER
0199 00BE C145          MOV R5,R5       ;NULL, ASSIGNMENT?
0200 00C0 1307            JEQ LET4F     ;Y
0201 00C2 DDC4  LET4I   MOVB R4,*R7+
0202 00C4 81C2          C R2,R7         ;IS S<D?
0203 00C6 1402            JHE LET4E     ;N
0204 00C8 80C2          C R2,R3         ;Y, IS S>=D0?
0205 00CA 1402            JHE LET4F     ;Y, ABORT
0206 00CC 0601  LET4E   DEC R1          ;DONE?
0207 00CE 16F5            JNE LET4D     ;N
0208                    *
0209 00D0 0915  LET4F   SRL R5,1        ;DONE, NEED NULL?
0210 00D2 1601            JNE LET4G     ;N
0211 00D4 75D7          SB *R7,*R7      ;Y
0212                    *
0213 00D6 0280  LET4G   CI R0,>5D00     ;+?
     00D8 5D00
0214 00DA 1632            JNE LETRR     ;N, RETURN
0215 00DC 10BC          JMP LET3E       ;Y, GET STRING
```

```
0217                    *ASCII TO DECIMAL
0218                    *
0219                    *       N=$A[-],E
0220                    *
0221 00DE 0420  LET6    BLWP @EVARZ     ;GET VARIABLE ADR
     00E0 000A'
0222 00E2 04D2          CLR *R2
0223 00E4 C1C4          MOV R4,R7
0224 00E6 0420          BLWP @CVDIFZ    ;CONVERT TO #
     00E8 0000
0225 00EA 1007            JMP LET6A
0226 00EC 04C1            CLR R1        ;NOTHING
0227 00EE 1000            NOP
0228 00F0 D480          MOVB R0,*R2     ;SAVE DELIMITER
0229 00F2 04F3          CLR *R3+        ;MOVE IN RESULTS
0230 00F4 CCC1          MOV R1,*R3+
0231 00F6 04D3          CLR *R3
0232 00F8 1023          JMP LETRR
0233                    *
0234 00FA D480  LET6A   MOVB R0,*R2     ;SAVE DELIMITER
0235 00FC 2C53          STORE *R3       ;STORE RESULTS
0236 00FE 1020          JMP LETRR       ;RETURN
```

```
0238                    *INSERT STRING
0239                    *
0240                    *       $A[-]=/$B[-]
0241                    *       (R7)=DESTINATION
0242                    *
0243 0100 0420  LET12   BLWP @EVSDZ      ;LOOK AT NEXT STRING
     0102 007
0244 0104 100             JMP LET13A
0245 0106 100             JMP LET13A
0246 0108 2EC           EVFIX R1        ;GET #
0247                    *
0248                    *DELETE CHARACTERS
0249                    *
0250                    *       $A[-]=/N
0251                    *       (R2)=(R7)=STRING
0252                    *       R1=COUNT
0253                    *
0254 010A 0601  LET12A  DEC R1          ;MOVE BY HOLE
0255 010C 1103            JLT LET12B    ;DONE
0256 010E D032          MOVB *R2+,R0    ;SKIP 1 CHARACTER
0257 0110 16FC            JNE LET12A    ;CONTINUE
0258 0112 0602          DEC R2          ;EOL
0259                    *
0260 0114 DDF2  LET12B  MOVB *R2+,*R7+  ;MOVE REST OF STRING
0261 0116 16FE            JNE LET12B
0262 0118 1013          JMP LETRR
0263                    *
0264                    *INSERT STRING
0265                    *
0266 011A C0C2  LET13A  MOV R2,R3       ;COUNT INSERT STRING
0267 011C 0701          SETO R1
0268                    *
0269 011E 0581  LET13B  INC R1          ;COUNT
0270 0120 D033          MOVB *R3+,R0
0271 0122 16FD            JNE LET13B
0272 0124 04C3          CLR R3          ;MOVE AND COUNT TO END OF DES STRING
0273                    *
0274 0126 0583  LET13C  INC R3          ;COUNT
0275 0128 D037          MOVB *R7+,R0
0276 012A 16FD            JNE LET13C
0277                    *
0278 012C C107          MOV R7,R4       ;MOVE DES STRING
0279 012E A101          A R1,R4
0280                    *
0281 0130 0607  LET13D  DEC R7
0282 0132 0604          DEC R4
0283 0134 D517          MOVB *R7,*R4    ;MOVE CHARACTERS
0284 0136 0603          DEC R3          ;TO HOLE?
0285 0138 15FB            JGT LET13D    ;N
0286                    *
0287 013A DDF2  LET13E  MOVB *R2+,*R7+  ;Y, INSERT STRING
0288 013C 0601          DEC R1          ;DONE?
0289 013E 15FD            JGT LET13E    ;N
0290 0140 0460  LETRR   B @NLIN         ;Y
     0142 0026'
0291                        END
NO ERRORS,     NO WARNINGS
```

```
CKEX    R  0016'   0019  0088
CVDIFZ  R  00E8'   0020  0224
ERR14      007A'   0156
ERR36      0042'   0118  0086
ERR7       003A'   0112  0159
ERROR      2F80    0029  0082  0104  0112  0118  0156
EVARZ   R  00E0'   0016  0084  0221
EVERZ   R  008C'   0017  0090  0165
EVFIX              0026
EVSDZ   R  0102'   0018  0081  0101  0137  0153  0243
FFLG    R  0092'   0033  0160  0167
IOB     R          0034
LET1       0028'   0101  0089
LET11      0046'   0125  0130  0142
LET11A     004A'   0128
LET12      0100'   0243  0144
LET12A     010A'   0254  0257
LET12B     0114'   0260  0255  0261
LET13A     011A'   0266  0244  0245
LET13B     011E'   0269  0271
LET13C     0126'   0274  0276
LET13D     0130'   0281  0285
LET13E     013A'   0287  0289
LET2       0032'   0108  0102  0103
LET3       003C'   0116  0083
LET3A      0054'   0136  0117
LET3B      007C'   0158  0154  0155
LET3C      0088'   0163  0146
LET3D      008A'   0165  0161
LET3E      0056'   0137  0215
LET4       0098'   0178  0138  0139
LET4A      00AA'   0186  0180
LET4B      00B2'   0191  0183
LET4C      00B8'   0195  0188
LET4D      00BA'   0197  0207
LET4E      00CC'   0206  0203
LET4F      00D0'   0209  0200  0205
LET4G      00D6'   0213  0210
LET4I      00C2'   0201  0198
LET6       00DE'   0221  0110
LET6A      00FA'   0234  0225
LETR       0024'   0097  0131  0169  0189
LETRR      0140'   0290  0214  0232  0236  0262
LETY    D  0000'   0081  0079
LOADF              0022
NLIN    R  0142'   0021  0097  0290
NORMAL             0025
OUTFP              0027
OUTINT             0028
R0         0000    0085  0109  0116  0129  0140  0141  0143  0145  0158
                   0179  0182  0213  0228  0234  0256  0270  0275
R1         0001    0125  0126  0128  0186  0187  0187  0191  0192  0193
                   0206  0226  0230  0246  0254  0267  0269  0279  0288
R2         0002    0087  0094  0095  0096  0108  0136  0160  0166  0197
                   0202  0204  0222  0228  0234  0256  0258  0260  0266
                   0287
R3         0003    0087  0094  0095  0096  0195  0204  0229  0230  0231
                   0235  0266  0270  0272  0274  0284
R4         0004    0108  0197  0201  0223  0278  0279  0282  0283
R5         0005    0178  0181  0184  0199  0199  0209
```

R7	0007		0128 0136 0168 0168 0192 0195 0201 0202 0211 0211 0223 0260 0275 0278 0281 0283 0287
R8	0008		0140 0163
R9	0009		0191
SCALE		0024	
STORE		0023	

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.IF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.IF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.IF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
      IDT 'IF'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       IFY                 ;IF COMMAND
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MACRO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:
0015              *
0016                      REF EVERZ           ;EVALUATE EXPRESSION
0017                      REF EVSDZ           ;EVALUATE STRING
0018                      REF LINE,NLIN       ;MULTIPLEXOR ENTRY POINTS
0019                      DXOP EVFIX,11       ;EVALUATE AND FIX
0020    2F80  ERROR       EQU >2F80           ;XOP XX,14  (ERROR CALL)
0021    2FA0  ERROR2      EQU ERROR+>20
0022              *
0023              * EXTERNAL DATA:
0024              *
0025                      REF ELSF            ;ELSE FLAG
0026              *
0027              * MODULE EQUATES:  (NONE)
0028              *
0029              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0032            * ABSTRACT:
0033            *
0034            *       EXECUTE THE IF COMMAND.  THE FORM IS:
0035            *
0036            *       IF <EXP> THEN <STATEMENT>
0037            *       IF <STRING> THEN <STATEMENT>
0038            *       IF <STRING> <RELATION> <STRING> THEN <STATEMENT>
0039            *       IF <STING> <RELATION> <STRING> , <EXP>
0040            *               THEN <STATEMENT>
0041            *
0042            * CALLING SEQUENCE:
0043            *
0044            *       B @IFY
0045            *
0046            *       EXIT TO LINE OR NLIN
0047            *
0048            * EXCEPTIONS AND CONDITIONS:
0049            *
0050            *       ILLEGAL DELIMITER, SYNTAX ERROR
0051            *
0052            * EXTERNAL ROUTINE LIST:
0053            *
0054            *      (SEE EXTERNAL ROUTINES)
0055            *
0056            * LOCAL DATA:
0057            *
0058            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0060                    * ENTRY POINT:
0061                    *
0062                          DEF IFY
0063                    *
0064 0000 04E9  IFY       CLR @ELSF(9)      ;CLEAR ELSE FLAG
     0002 0000
0065 0004 0420            BLWP @EVSDZ       ;CHECK FOR " OR $
     0006 0000
0066 0008 1008              JMP IF2
0067 000A 1007              JMP IF2
0068 000C 0420            BLWP @EVERZ       ;#
     000E 0000
0069 0010 C072            MOV *R2+,R1
0070 0012 161A              JNE IFRT
0071 0014 C052            MOV *R2,R1
0072 0016 1618              JNE IFRT
0073 0018 102F            JMP IF13
0074                    *
0075 001A 0280  IF2       CI R0,>3B00       ;THEN?
     001C 3B00
0076 001E 1603              JNE IF4         ;N
0077                    *
0078 0020 D052  IF3       MOVB *R2,R1       ;Y, STRING?
0079 0022 1612              JNE IFRT        ;Y, CONTINUE
0080 0024 1029            JMP IF13                ;N, GOTO LINE
0081                    *
0082 0026 06C0  IF4       SWPB R0
0083 0028 0220            AI R0,->55        ;LEGAL?
     002A FFAB
0084 002C 1502              JGT IF5         ;Y
0085 002E 2FA0  IFE37     DATA ERROR2,37    ;N
     0030 0025
0086                    *
0087 0032 0280  IF5       CI R0,>6
     0034 0006
0088 0036 15FB              JGT IFE37       ;N
0089 0038 C180            MOV R0,R6
0090 003A C0C2            MOV R2,R3         ;STRINGS
0091 003C 0705            SETO R5
0092 003E 0420            BLWP @EVSDZ       ;GET SECOND OPERAND
     0040 0006'
0093 0042 100B              JMP IF7         ;"
0094 0044 100A              JMP IF7         ;$
0095 0046 2F8E            DATA ERROR+14     ;EXPECTING STRING
0096                    *
0097 0048 0460  IFRT      B @NLIN
     004A 0000
0098                    *
0099 004C D072  IF6       MOVB *R2+,R1      ;LOAD BYTE FROM SECOND STRING
0100 004E 1601              JNE $+4         ;OK
0101 0050 0701              SETO R1         ;NULL
0102 0052 7044            SB R4,R1          ;SAME BYTE?
0103 0054 1608              JNE IF10        ;N
0104 0056 0605            DEC R5            ;Y, DONE?
0105 0058 1308              JEQ IF11        ;Y
0106                    *
0107 005A 0280  IF7       CI R0,>3F00       ;,?
     005C 3F00
0108 005E 1601              JNE IF8         ;N
0109 0060 2EC5            EVFIX R5          ;Y, GET COUNT
```

```
 0110                    *
 0111 0062 0200   IF8    LI R0,4             ;GET MATCH
      0064 0004
 0112 0066 D133          MOVB *R3+,R4        ;SOURCE NULL?
 0113 0068 16F1            JNE IF6           ;N
 0114                    *
 0115 006A D072   IF9    MOVB *R2+,R1        ;LOAD BYTE FROM SECOND STRING
 0116 006C 1503   IF10     JGT IF12          ;POSITIVE
 0117 006E 1601            JNE $+4           ;NEGATIVE
 0118 0070 0600   IF11   DEC R0              ;ZERO
 0119 0072 0640          DECT R0
 0120                    *
 0121 0074 2180   IF12   COC R0,R6           ;CONDITION MET?
 0122 0076 13E8            JEQ IFRT          ;Y
 0123 0078 0729   IF13   SETO @ELSF(9)       ;N
      007A 0002'
 0124 007C 0460          B @LINE
      007E 0000
 0125                  END
NO ERRORS,      NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES								
$		0080'		0100	0117							
ELSF	R	007A'	0025	0064	0123							
ERROR		2F80	0020	0021	0095							
ERROR2		2FA0	0021	0085								
EVERZ	R	000E'	0016	0068								
EVFIX			0019									
EVSDZ	R	0040'	0017	0065	0092							
IF10		006C'	0116	0103								
IF11		0070'	0118	0105								
IF12		0074'	0121	0116								
IF13		0078'	0123	0073	0080							
IF2		001A'	0075	0066	0067							
IF3		0020'	0078									
IF4		0026'	0082	0076								
IF5		0032'	0087	0084								
IF6		004C'	0099	0113								
IF7		005A'	0107	0093	0094							
IF8		0062'	0111	0108								
IF9		006A'	0115									
IFE37		002E'	0085	0088								
IFRT		0048'	0097	0070	0072	0079	0122					
IFY	D	0000'	0064	0062								
LINE	R	007E'	0018	0124								
NLIN	R	004A'	0018	0097								
R0		0000		0075	0082	0083	0087	0089	0107	0111	0118	0119
				0121								
R1		0001		0069	0071	0078	0099	0101	0102	0115		
R2		0002		0069	0071	0078	0090	0099	0115			
R3		0003		0090	0112							
R4		0004		0102	0112							
R5		0005		0091	0104	0109						
R6		0006		0089	0121							

*

```
SOURCE ACCESS NAME=      DBASIC.REL4.BASIC.SOURCE.PRINT
OBJECT ACCESS NAME=      DBASIC.REL4.BASIC.OBJECT.PRINT
LISTING ACCESS NAME=     DBASIC.REL4.BASIC.LIST.PRINT
ERROR ACCESS NAME=
OPTIONS=                 X
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'PRINT'
0002                    DEF PRTY,HOUT
0003                *     24 APRIL 1979
0004                    REF B20,B3F
0005                    REF BMVE,CCNT,FFLG
0006                    REF EVERZ,EVSDZ
0007                    REF LNSZ,MODE
0008                    REF OUTL1,PRTEND,JMPR0A
0009                    DXOP EVFIX,11
0010                    DXOP OUTFP,12
0011                *
0012                *PRINT COMMAND
0013                *
0014 0000 020E      PRTY    LI R14,LNSZ     ;GET LINE SIZE
     0002 0000
0015 0004 C029              MOV @MODE(9),R0 ;LOOK AT MODE
     0006 0000
0016 0008 1601                JNE PRT0      ;RUN
0017 000A 093E              SRL R14,3       ;IDLE, LOAD COMMA SIZE
0018                *
0019 000C C1D9      PRT0    MOV *R9,R7      ;SET R7  (WSBC)
0020                *
0021 000E 070F      PRT1    SETO R15        ;SET FOR CRLF
0022                *
0023 0010 058F      PRT2    INC R15
0024 0012 06A0              BL @JMPR0A      ;SWITCH BOARD
     0014 0000
0025 0016   17      PRTTB     BYTE PRTE-PRTTB/2,>00   ;NULL
     0017   00
0026 0018   17                BYTE PRTE-PRTTB/2,>3C   ;:
     0019   3C
0027 001A   17                BYTE PRTE-PRTTB/2,>47   ;I
     001B   47
0028 001C   19                BYTE PRTF-PRTTB/2,>3E   ;#
     001D   3E
0029 001E   47                BYTE PRTC-PRTTB/2,>3F   ;,
     001F   3F
0030 0020   4F                BYTE PRTSC-PRTTB/2,>40  ;;
     0021   40
0031 0022   56                BYTE PRTAB-PRTTB/2,>39  ;TAB
     0023   39
0032 0024 0000                DATA 0
0033 0026 0608              DEC R8          ;TRY STRINGS
0034 0028 0420              BLWP @EVSDZ
     002A 0000
0035 002C 102D                JMP PRTQ      ;"
0036 002E 102C                JMP PRTQ      ;$
0037 0030 0420              BLWP @EVERZ     ;NEITHER, TRY NUMBER
     0032 0000
0038 0034 C129              MOV @FFLG(9),R4 ;FORMATTING?
     0036 0000
0039 0038 1602                JNE $+6       ;Y, IGNOR SPACE
0040 003A DDE0              MOVB @B20,*R7+  ;N, OUT SPACE
     003C 0000
0041 003E 2F12              OUTFP *R2       ;OUTPUT #
0042                *
0043 0040 0608      PRT4    DEC R8          ;BACKUP TO DELIMITER
0044 0042 10E5              JMP PRT1
0045                *
0046 0044 0460-     PRTE    B @PRTEND        ;END PRINT
```

0046 0000

```
0048                    *PRINT FORMATTING
0049                    *
0050 0048 0420  PRTF    BLWP @EVSDZ         ;EVALUATE STRING
     004A 002A'
0051 004C 101A            JMP PRTF1         ;"
0052 004E 1019            JMP PRTF1         ;$
0053 0050 0202          LI R2,5             ;MAYBE HEX
     0052 0005
0054 0054 9818          CB *R8,@B3F         ;#,?
     0056 0000
0055 0058 1603            JNE PRTH1         ;N
0056 005A 0588          INC R8
0057 005C 2EC1          EVFIX R1            ;Y, GET #
0058 005E 1007          JMP PRTH2
0059                    *
0060 0060 9818  PRTH1   CB *R8,@B40         ;#,?
     0062 0128'
0061 0064 1608            JNE PRTH3         ;N
0062 0066 0588          INC R8
0063 0068 2EC1          EVFIX R1            ;Y, GET #
0064 006A 0642          DECT R2
0065 006C 0A81          SLA R1,8            ;LEFT JUSTIFY
0066                    *
0067 006E 06A0  PRTH2   BL @HOUTE           ;OUT #
     0070 010C'
0068 0072 0607          DEC R7              ;BACKUP ON "H
0069 0074 10E5          JMP PRT4
0070                    *
0071 0076 2EC1  PRTH3   EVFIX R1            ;FORMAT FREE
0072 0078 DDE0          MOVB @B20,*R7+      ;OUT SPACE
     007A 003C'
0073 007C 06A0          BL @HOUT            ;OUT #
     007E 00F2'
0074 0080 10DF          JMP PRT4
0075                    *
0076 0082 CA42  PRTF1   MOV R2,@FFLG(9)     ;SET FLAG
     0084 0036'
0077 0086 10DC          JMP PRT4
0078                    *
0079 0088 C14E  PRTQ    MOV R14,R5          ;GET NUMBER OF BYTES AVAILABLE
0080 008A A159          A *R9,R5
0081 008C 6147          S R7,R5
0082 008E 06A0          BL @BMVE            ;MOVE INTO OUTPUT STRING
     0090 0000
0083 0092 1005            JMP PRTQ1         ;OK
0084 0094 06A0          BL @OUTL1           ;LIST LINE
     0096 0000
0085 0098 AA43          A R3,@CCNT(9)       ;UPDATE BACKSPACES
     009A 0000
0086 009C 10F5          JMP PRTQ
0087                    *
0088 009E AA43  PRTQ1   A R3,@CCNT(9)       ;UPDATE COLUMN COUNT
     00A0 009A'
0089 00A2 10CE          JMP PRT4
0090                    *
0091                    *COMMA
0092                    *
0093 00A4 06A0  PRTC    BL @PRTCK           ;GET CURRENT # OF BYTES
     00A6 00E8'
0094 00A8 0240          ANDI R0,>F          ;MOD 16
```

```
      00AA 000F
0095  00AC 0220          AI R0,-16        ;GET -# OF BLANKS
      00AE FFF0
0096  00B0 06A0          BL @PRTSP        ;OUT SPACES
      00B2 00DE'
0097                     *
0098                     *SEMI-COLON OPERATER
0099                     *
0100  00B4 06A0  PRTSC   BL @PRTCK        ;CHECK BUFFER OVERFLOW
      00B6 00E8'
0101  00B8 600E          S R14,R0         ;EXCEEDED?
0102  00BA 11AA            JLT PRT2       ;N
0103  00BC 06A0          BL @OUTL1        ;Y, PRINT LINE
      00BE 0096'
0104  00C0 10A7          JMP PRT2
```

```
0106                *TAB FUNCTION
0107                *
0108 00C2 2EC1      PRTAB   EVFIX R1          ;GET TAB
0109 00C4 0608              DEC R8            ;BACKUP OVER DELIMITER
0110 00C6 0241              ANDI R1,>7F       ;LIMIT TABS TO 127
     00C8 007F
0111 00CA 06A0              BL @PRTCK         ;GET CURRENT CHARACTER POSITION
     00CC 00E8'
0112 00CE 6001              S R1,R0           ;NEED SPACES?
0113 00D0 159F                JGT PRT2        ;N, PAST POINT
0114                ********************************************************
0115                *     CHANGED FOR SPR 26  24 APRIL 1979
0116 00D2 C069              MOV  @MODE(R9),R1     ;CHECK MODE
     00D4 0006'
0117 00D6 139C              JEQ  PRT2             ;KEYBOARD MODE IGNORE
0118                ********************************************************
0119 00D8 06A0              BL @PRTSP         ;Y, OUT SPACES
     00DA 00DE'
0120 00DC 1099              JMP PRT2
0121                *
0122                *OUT -R0 SPACES
0123                *
0124 00DE 0580      PRTSP   INC R0            ;DONE?
0125 00E0 1507                JGT PRTCK1      ;Y
0126 00E2 DDE0              MOVB @B20,*R7+    ;N, OUT SPACE
     00E4 007A'
0127 00E6 10FB              JMP PRTSP
0128                *
0129 00E8 C029      PRTCK   MOV @CCNT(9),R0 ;GET LINE SIZE
     00EA 00A0'
0130 00EC A007              A R7,R0
0131 00EE 6019              S *R9,R0          ;GET CURRENT CHARACTER POSITION
0132 00F0 045B      PRTCK1  RT
```

```
 0134                    *PRINT HEX
 0135                    *
 0136 00F2 DDE0  HOUT    MOVB @HTXT,*R7+ ;OUT "0
      00F4 0118'
 0137 00F6 04C0          CLR R0          ;RESET ZERO FLAG
 0138 00F8 0202          LI R2,4         ;DO 4 DIGITS
      00FA 0004
 0139                    *
 0140 00FC C101  HOUT1   MOV R1,R4
 0141 00FE 0A41          SLA R1,4        ;ISOLATE DIGIT
 0142 0100 09C4          SRL R4,12
 0143 0102 1602            JNE HOUT2     ;NON-ZERO
 0144 0104 C000          MOV R0,R0       ;FLAG SET?
 0145 0106 1303            JEQ HOUT3     ;N
 0146 0108 DDE4  HOUT2   MOVB @HTXT(4),*R7+
      010A 0118'
 0147                    *
 0148 010C 0700  HOUTE   SETO R0         ;SET FLAG
 0149                    *
 0150 010E 0602  HOUT3   DEC R2          ;DONE?
 0151 0110 15F5            JGT HOUT1     ;N
 0152 0112 DDE0          MOVB @B48,*R7+  ;Y, OUT "H
      0114 0129'
 0153 0116 045B          RT
 0154                    *
 0155 0118   30  HTXT    TEXT '0123456789ABCDEF'
      0119   31
      011A   32
      011B   33
      011C   34
      011D   35
      011E   36
      011F   37
      0120   38
      0121   39
      0122   41
      0123   42
      0124   43
      0125   44
      0126   45
      0127   46
 0156 0128   40  B40     BYTE >40
 0157 0129   48  B48     BYTE >48
 0158 012A               EVEN
 0159                   END
NO ERRORS
```

```
$          012A'          0039
B20     R  00E4'  0004    0040  0072  0126
B3F     R  0056'  0004    0054
B40        0128'  0156    0060
B48        0129'  0157    0152
BMVE    R  0090'  0005    0082
CCNT    R  00EA'  0005    0085  0088  0129
EVERZ   R  0032'  0006    0037
EVFIX             0009
EVSDZ   R  004A'  0006    0034  0050
FFLG    R  0084'  0005    0038  0076
HOUT    D  00F2'  0136    0002  0073
HOUT1      00FC'  0140    0151
HOUT2      0108'  0146    0143
HOUT3      010E'  0150    0145
HOUTE      010C'  0148    0067
HTXT       0118'  0155    0136  0146
JMPR0A  R  0014'  0008    0024
LNSZ    R  0002'  0007    0014
MODE    R  00D4'  0007    0015  0116
OUTFP             0010
OUTL1   R  00BE'  0008    0084  0103
PRT0       000C'  0019    0016
PRT1       000E'  0021    0044
PRT2       0010'  0023    0102  0104  0113  0117  0120
PRT4       0040'  0043    0069  0074  0077  0089
PRTAB      00C2'  0108    0031
PRTC       00A4'  0093    0029
PRTCK      00E8'  0129    0093  0100  0111
PRTCK1     00F0'  0132    0125
PRTE       0044'  0046    0025  0026  0027
PRTEND  R  0046'  0008    0046
PRTF       0048'  0050    0028
PRTF1      0082'  0076    0051  0052
PRTH1      0060'  0060    0055
PRTH2      006E'  0067    0058
PRTH3      0076'  0071    0061
PRTQ       0088'  0079    0035  0036  0086
PRTQ1      009E'  0088    0083
PRTSC      00B4'  0100    0030
PRTSP      00DE'  0124    0096  0119  0127
PRTTB      0016'  0025    0025  0026  0027  0028  0029  0030  0031
PRTY    D  0000'  0014    0002
R0         0000           0015  0094  0095  0101  0112  0124  0129  0130  0131
                          0137  0144  0144  0148
R1         0001           0057  0063  0065  0071  0108  0110  0112  0116  0140
                          0141
R14        000E           0014  0017  0079  0101
R15        000F           0021  0023
R2         0002           0041  0053  0064  0076  0138  0150
R3         0003           0085  0088
R4         0004           0038  0140  0142
R5         0005           0079  0080  0081
R7         0007           0019  0040  0068  0072  0081  0126  0130  0136  0146
                          0152
R8         0008           0033  0043  0054  0056  0060  0062  0109
R9         0009           0019  0080  0116  0131
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.MOVE
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.MOVE
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.MOVE
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'MOVE'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       BMVE              ;BYTE MOVE
0009          *
0010          * COPY: (NONE)
0011          *
0012          * MACRO: (NONE)
0013          *
0014          * EXTERNAL ROUTINES:  (NONE)
0015          *
0016          * EXTERNAL DATA:  (NONE)
0017          *
0018          * MODULE EQUATES:  (NONE)
0019          *
0020          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0023            * ABSTRACT:
0024            *
0025            *       MOVE BYTES FROM (R2) TO (R7) WATCHING FOR
0026            *       BACKSPACES AND HEX CHARACTERS IN
0027            *       ANGLE BRACKETS.  ALSO WATCH FOR
0028            *       BUFFER OVERFLOW.
0029            *
0030            * CALLING SEQUENCE:
0031            *
0032            *       BL @BMVE
0033            *         MOVE OK
0034            *       OVERFLOW
0035            *
0036            *       IN  R2 = SOURCE
0037            *           R5 = MAX COUNT
0038            *           R7 = DESTINATION
0039            *       OUT R3 = -(# OF <10>)*2
0040            *
0041            * EXCEPTIONS AND CONDITIONS:  (NONE)
0042            *
0043            * EXTERNAL ROUTINE LIST:
0044            *
0045            *      (SEE EXTERNAL ROUTINES)
0046            *
0047            * LOCAL DATA:
0048            *
0049            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0051                    * ENTRY POINT:
0052                    *
0053                          DEF BMVE
0054                    *
0055 0000 04C3  BMVE    CLR R3              ;RESET BS COUNT
0056                    *
0057 0002 D5F2  BMVE0   MOVB *R2+,*R7       ;MOVE BYTE
0058 0004 130C            JEQ BMVE2         ;DONE
0059 0006 9817          CB *R7,@B3C         ;"<?
     0008 005F'
0060 000A 130A            JEQ BMVE3         ;Y
0061 000C 0587  BMVE1   INC R7              ;N
0062 000E C299          MOV *R9,R10         ;GET IOB
0063 0010 060A          DEC R10             ;BACKUP TO LAST BYTE
0064 0012 8287          C R7,R10            ;ABOUT TO STORE IN LAST BYTE?
0065 0014 1303            JEQ BMVE6         ;Y, QUIT
0066 0016 0605          DEC R5              ;MORE ROOM?
0067 0018 15F4            JGT BMVE0         ;Y
0068 001A 05CB          INCT R11            ;N, RETURN 2(11)
0069                    *
0070 001C 75C7  BMVE6   SB *R7,*R7          ;MARK
0071                    *
0072 001E 045B  BMVE2   RT
0073                    *
0074 0020 04C1  BMVE3   CLR R1              ;BUILD # IN R1
0075 0022 C282          MOV R2,R10          ;SAVE R2
0076                    *
0077 0024 04C0  BMVE4   CLR R0
0078 0026 D03A          MOVB *R10+,R0       ;GET NEXT CHARACTER
0079 0028 0220          AI R0,->3000
     002A D000
0080 002C 110E            JLT GHEX2
0081 002E 028           CI R0,>0900
     0030 090
0082 0032 120             JLE GHEX1
0083 0034 0220          AI R0,->0700
     0036 F900
0084 0038 0280          CI R0,>0A00
     003A 0A00
0085 003C 1106            JLT GHEX2
0086 003E 0280          CI R0,>0F00
     0040 0F00
0087 0042 1B03            JH GHEX2
0088 0044 0A41  GHEX1   SLA R1,4            ;SHIFT R1
0089 0046 B040          AB R0,R1            ;ADD NEW DIGIT
0090 0048 10ED          JMP BMVE4
0091                    *
0092      004A' GHEX2   EQU $
0093 004A 0280  BMVE5   CI R0,>3E00->3700        ;">?
     004C 0700
0094 004E 16DE            JNE BMVE1         ;N, DISREGUARD #
0095 0050 C08A          MOV R10,R2          ;Y, SET R2
0096 0052 D5C1          MOVB R1,*R7         ;MOVE INTO STRING
0097 0054 9801          CB R1,@B08          ;BS?
     0056 005E'
0098 0058 16D9            JNE BMVE1         ;N
0099 005A 0643          DECT R3             ;Y, COUNT
0100 005C 10D7          JMP BMVE1
0101 005E   08  B08     BYTE >08
0102 005F   3C  B3C     BYTE >3C
```

```
 0103 0060                EVEN
 0104                     END
NO ERRORS,      NO WARNINGS
```

$	0060'		0092								
B08	005E'	0101	0097								
B3C	005F'	0102	0059								
BMVE D	0000'	0055	0053								
BMVE0	0002'	057	0067								
BMVE1	000C'	0061	0094	0098	0100						
BMVE2	001E'	0072	0058								
BMVE3	0020'	0074	0060								
BMVE4	0024'	0077	0090								
BMVE5	004A'	0093									
BMVE6	001C'	0070	0065								
GHEX1	0044'	0088	0082								
GHEX2	004A'	0092	0080	0085	0087						
R0	0000		0077	0078	0079	0081	0083	0084	0086	0089	0093
R1	0001		0074	0088	0089	0096	0097				
R10	000A		0062	0063	0064	0075	0078	0095			
R11	000B		0068								
R2	000		0057	0075	0095						
R3	0003		0055	0099							
R5	000		0066								
R7	0007		0057	0059	0061	0064	0070	0070	0096		
R9	000		0062								

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.DIM
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.DIM
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.DIM
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT 'DIM'
0002                    *
0003                    * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                    *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                    *
0006                    * ROUTINE LIST:
0007                    *
0008                    *       DIMY                ;DIMENSION COMMAND
0009                    *
0010                    * COPY: (NONE)
0011                    *
0012                    * MARCO: (NONE)
0013                    *
0014                    * EXTERNAL ROUTINES:
0015                    *
0016                            REF EVAL,EVALS2 ;RECURSIVE EVALUATOR
0017                            REF EVARZ       ;EVALUATE VARIABLE
0018                            REF FIX         ;FIX ARGUMENT
0019                            REF NLIN        ;EXIT TO MULTIPLEXOR
0020    2F80    ERROR   EQU >2F80       ;XOP XX,14  (ERROR CALL)
0021    2FA0    ERROR2  EQU ERROR+>20
0022                    *
0023                    * EXTERNAL DATA:
0024                    *
0025                            REF VDT         ;VARIABLE DEFINITION TABLE INDEX
0026                            REF NVS         ;NEXT VARIABLE STORAGE
0027                            REF NVD         ;NEXT VARIABLE DEFINITION
0028                            REF DLIM        ;PROGRAM DELIMITER
0029                            REF C6          ;CONSTANT 6
0030                    *
0031                    * MODULE EQUATES:
0032                    *
0033    0004    NRV     EQU 4           ;NUMBER OF RESERVED WORDS
0034                    *
0035                    * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0038            * ABSTRACT:
0039            *
0040            *       DIMENSION A VARIABLE.  A DIMENSIONED
0041            *       VARIABLE IS INDICATED IN THE VDT BY A
0042            *       NEGATIVE VALUE.  THE VARIABLE
0043            *       DEFINITION POINT TO THE INFORMATION
0044            *       VECTOR.
0045            *
0046            *INFORMATION VECTOR FORMAT:
0047            *
0048            *       D1,X1,D2,X2,...DN,-1,...DATA...
0049            *
0050            *WHERE  D1,D2,...DN = DIMENSIONS
0051            *       X1,X2,...   = MULTIPLIERS
0052            *
0053            *       X1 = D2+1 X D3+1 X ... DN+1
0054            *       X2 = D3+1 X D4+1 X ... DN+1
0055            *       ...
0056            *       XN = 1
0057            *
0058            * CALLING SEQUENCE:
0059            *
0060            *       B @DIMY
0061            *
0062            * EXCEPTIONS AND CONDITIONS:
0063            *
0064            *       EVALUATION ERRORS, STORAGE OVERFLOW
0065            *       EXPECTING DIMENSIONED VARIABLE
0066            *       INVALID DELIMITER
0067            *
0068            * EXTERNAL ROUTINE LIST:
0069            *
0070            *      (SEE EXTERNAL ROUTINES)
0071            *
0072            * LOCAL DATA:
0073            *
0074            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0076                    * ENTRY POINT:
0077                    *
0078                          DEF DIMY
0079                    *
0080 0000 04C0  DIMY      CLR R0            ;GET VARIABLE NAME
0081 0002 D038            MOVB *R8+,R0
0082 0004 0280            CI R0,NRV*>100+>7000    ;VARIABLE?
     0006 7400
0083 0008 1A56              JL ERR16        ;N
0084 000A C3A9            MOV @VDT(9),R14 ;Y, GET SYMBOL ADR
     000C 0000
0085 000E 0970            SRL R0,7
0086 0010 A380            A R0,R14          ;INDEX
0087 0012 022E            AI R14,->70*2     ;ELIMINATE DISPLACEMENT
     0014 FF20
0088 0016 C05E            MOV *R14,R1       ;DEFINED?
0089 0018 161E              JNE DIM3        ;Y, SEE IF EXCEEDS PREVIOUS
0090                    *
0091 001A 0705            SETO R5           ;DO FULL EVALUATIONS
0092 001C 06A0            BL @EVALS2        ;DO INITIAL EVAL
     001E 0000
0093 0020 04F6            CLR *R6+          ;ZERO COUNT
0094 0022 1002            JMP DIM2
0095                    *
0096 0024 06A0  DIM1      BL @EVAL          ;EVALUATE NEXT SUBSCRIPT
     0026 0000
0097                    *
0098 0028 06A0  DIM2      BL @FIX           ;FIX PARAMETER (R2)
     002A 0000
0099 002C 0646            DECT R6           ;POP COUNT
0100 002E C0D6            MOV *R6,R3
0101 0030 0583            INC R3            ;COUNT DIMENSION
0102 0032 CD81            MOV R1,*R6+       ;PUSH DIMENSION
0103 0034 0736            SETO *R6+         ;LEAVE SPACE FOR DEL MULTIPLIER
0104 0036 CD83            MOV R3,*R6+       ;PUSH COUNT AGAIN
0105 0038 0280            CI R0,>3F00       ;,?
     003A 3F00
0106 003C 13F3              JEQ DIM1        ;Y
0107 003E 0280            CI R0,>4B00       ;)?
     0040 4B00
0108 0042 1607              JNE DIME37      ;N, ERROR
0109 0044 0646            DECT R6           ;RETRIEVE COUNT
0110 0046 C016            MOV *R6,R0
0111 0048 C280            MOV R0,R10        ;GET INFORMATION VECTOR LENGTH
0112 004A 0A2A            SLA R10,2         ;COUNT X 4
0113 004C 0201            LI R1,1           ;SET DEL(N) TO 1
     004E 0001
0114 0050 1011            JMP DIM5
0115                    *
0116 0052 2FA0  DIME37 DATA ERROR2,37
     0054 0025
```

```
0118 0056 0608  DIM3    DEC R8              ;BACKUP TO VARIABLE
0119 0058 0420          BLWP @EVARZ         ;EVALUATE
     005A 0000
0120 005C 1024          JMP DIM7
0121              *
0122 005E 0226  DIM4    AI R6,-4            ;DO ANOTHER DIMENSION
     0060 FFFC
0123 0062 C096          MOV *R6,R2          ;GET DIMENSION
0124 0064 0582          INC R2              ;INCREMENT
0125 0066 3881          MPY R1,R2           ;GET PRODUCT
0126 0068 C082          MOV R2,R2           ;OVERFLOW?
0127 006A 1624            JNE DIME10        ;Y
0128 006C C043          MOV R3,R1           ;SET R1
0129 006E 1122            JLT DIME10
0130 0070 C981          MOV R1,@-2(6)       ;STORE DEL IN INFORMATION VECTOR
     0072 FFFE
0131              *
0132 0074 0600  DIM5    DEC R0              ;DONE?
0133 0076 16F3            JNE DIM4          ;N
0134 0078 0226          AI R6,-4            ;FINISHED, CALCULATE VECTOR
     007A FFFC
0135 007C C016          MOV *R6,R0
0136 007E 0580          INC R0              ;1ST DIMENSION + 1
0137 0080 3840          MPY R0,R1           ;GET FINAL DIMENSION SIZE
0138 0082 38A0          MPY @C6,R2          ;X 6 BYTES
     0084 0000
0139 0086 A0CA          A R10,R3            ;ADD INFORMATION VECTOR LENGTH
0140 0088 C0A9          MOV @NVS(9),R2      ;DEFINE
     008A 0000
0141 008C 6083          S R3,R2
0142 008E 8A42          C R2,@NVD(9)        ;OK?
     0090 0000
0143 0092 1A10            JL DIME10         ;N, STORAGE OVERFLOW
0144 0094 C782          MOV R2,*R14         ;Y, SET IN SYMBOL TABLE
0145 0096 CA42          MOV R2,@NVS(9)      ;UPDATE NVS
     0098 008A'
0146              *
0147 009A CCB6  DIM6    MOV *R6+,*R2+       ;MOVE INFORMATION VECTOR
0148 009C 05A6          INC @-2(6)          ;DONE?
     009E FFFE
0149 00A0 16FC            JNE DIM6          ;N
0150              *
0151 00A2 04C0          CLR R0
0152 00A4 D038          MOVB *R8+,R0        ;GET NEXT BYTE
0153              *
0154 00A6 0280  DIM7    CI R0,>3F00         ;",?
     00A8 3F00
0155 00AA 13AA            JEQ DIMY          ;Y, DO NEXT VARIABLE
0156 00AC CA40          MOV R0,@DLIM(9)     ;N, SAVE IN DLIM
     00AE 0000
0157 00B0 0460  DIM8    B @NLIN             ;RETURN
     00B2 0000
0158              *
0159 00B4 2F8A  DIME10  DATA ERROR+10       ;STORAGE OVERFLOW
0160              *
0161 00B6 2F90  ERR16   DATA ERROR+16       ;EXPECTING DIMENSIONED VARIABLE
0162                    END
NO ERRORS,      NO WARNINGS
```

C6	R	0084'	0029	0138								
DIM1		0024'	0096	0106								
DIM2		0028'	0098	0094								
DIM3		0056'	0118	0089								
DIM4		005E'	0122	0133								
DIM5		0074'	0132	0114								
DIM6		009A'	0147	0149								
DIM7		00A6'	0154	0120								
DIM8		00B0'	0157									
DIME10		00B4'	0159	0127	0129	0143						
DIME37		0052'	0116	0108								
DIMY	D	0000'	0080	0078	0155							
DLIM	R	00AE'	0028	0156								
ERR16		00B6'	0161	0083								
ERROR		2F80	0020	0021	0159	0161						
ERROR2		2FA0	0021	0116								
EVAL	R	0026'	0016	0096								
EVALS2	R	001E'	0016	0092								
EVARZ	R	005A'	0017	0119								
FIX	R	002A'	0018	0098								
NLIN	R	00B2'	0019	0157								
NRV		0004	0033	0082								
NVD	R	0090'	0027	0142								
NVS	R	0098'	0026	0140	0145							
R0		0000		0080	0081	0082	0085	0086	0105	0107	0110	0111
				0132	0135	0136	0137	0151	0152	0154	0156	
R1		0001		0088	0102	0113	0125	0128	0130	0137		
R10		000A		0111	0112	0139						
R14		000E		0084	0086	0087	0088	0144				
R2		0002		0123	0124	0125	0126	0126	0138	0140	0141	0142
				0144	0145	0147						
R3		0003		0100	0101	0104	0128	0139	0141			
R5		0005		0091								
R6		0006		0093	0099	0100	0102	0103	0104	0109	0110	0122
				0123	0134	0135	0147					
R8		0008		0081	0118	0152						
VDT	R	000C'	0025	0084								

*

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.FOR
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.FOR
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.FOR
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'FOR'
0002            *
0003            * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004            *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005            *
0006            * ROUTINE LIST:
0007            *
0008            *       FORY              ;FOR COMMAND
0009            *       NXTY              ;NEXT COMMAND
0010            *       FOR2              ;DELETE FROM FOR/NEXT STACK
0011            *       GSIM              ;GET SIMPLE VERIABLE
0012            *
0013            * COPY: (NONE)
0014            *
0015            * MARCO: (NONE)
0016            *
0017            * EXTERNAL ROUTINES:
0018            *
0019                    REF ADDF          ;ADD
0020                    REF SUBF          ;SUBTRACT
0021                    REF EVERZ         ;EVALUATE EXPRESSION
0022                    REF NLIN,NLIN0    ;EXIT TO MULTIPLEXOR
0023    2F80    ERROR   EQU >2F80         ;XOP XX,14  (ERROR CALL)
0024    2FA0    ERROR2  EQU ERROR+>20
0025            *
0026            * EXTERNAL DATA:
0027            *
0028                    REF BUS           ;BEGINNING USER STORAGE
0029                    REF EUS           ;END USER STORAGE
0030                    REF FNS           ;FOR/NEXT STACK
0031                    REF NVD           ;NEXT VARIABLE DEFINITION
0032                    REF NVS           ;NEXT VARIABLE STORAGE
0033                    REF PLC           ;PROGRAM LINE COUNTER
0034                    REF SLT           ;STATEMENT LINE TABLE
0035                    REF VDT           ;VARIABLE DEFINITION TABLE
0036                    REF VNT           ;VARIABLE NAME TABLE
0037            *
0038            * MODULE EQUATES:
0039            *
0040    0004    NRV     EQU 4             ;NUMBER OF RESERVED WORDS
0041                    REF NXTXB         ;NEXT COMMAND BYTE
0042            *
0043            * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0046            * ABSTRACT:
0047            *
0048            *       EXECUTE FOR COMMAND.  WHEN A FOR
0049            *       STATEMENT IS EXECUTED, THE FOR/NEXT
0050            *       STACK (FNS) IS SEARCHED FOR A ZERO
0051            *       ENTRY.  DURING THE SEARCH, IF
0052            *       AN IDENTICAL ENTRY IS FOUND, IT IS
0053            *       DELETED AND THE STACK IS ROLLED DOWN.
0054            *
0055            *       A PRE-TEST IS MADE TO SEE IF THE
0056            *       CONDITION IS MET IN WHICH CASE, THE
0057            *       FOR COMMAND WILL SEARCH FOR A MATCHING
0058            *       NEXT.
0059            *
0060            *       THE FOR/NEXT STACK FORMAT IS:
0061            *
0062            *       !    '    '    '    '    '    '    '    '    !
0063            *         VAR -----STEP----- ------TO------ PBC  PLC
0064            *
0065            *       VAR = 0 INDICATES EMPTY SLOT
0066            *
0067            * CALLING SEQUENCE:
0068            *
0069            *       B @FORY
0070            *
0071            *       EXIT TO NLIN
0072            * EXCEPTIONS AND CONDITIONS:
0073            *
0074            *       FOR W/O NEXT, STACK OVERFLOW
0075            *       EVALUATION ERRORS
0076            *       STORAGE OVERFLOW, ILLEGAL DELIMITER
0077            *       EXPECTING SIMPLE VARIABLE
0078            *
0079            * EXTERNAL ROUTINE LIST:
0080            *
0081            *      (SEE EXTERNAL ROUTINES)
0082            *
0083            * LOCAL DATA:
0084            *
0085            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0087                    * ENTRY POINT:
0088                    *
0089                            DEF FORY,FOR2
0090                    *
0091 0000 06A0  FORY    BL @GSIM          ;GET SIMPLE VARIABLE
     0002 015C'
0092 0004 C0D4          MOV *R4,R3        ;DEFINED?
0093 0006 160A            JNE FOR0        ;Y
0094 0008 C0E9          MOV @NVS(9),R3    ;N, DEFINE
     000A 0000
0095 000C 0223          AI R3,-6
     000E FFFA
0096 0010 8A43          C R3,@NVD(9)      ;OK?
     0012 0000
0097 0014 1A6F            JL FORE10       ;N
0098 0016 C503          MOV R3,*R4        ;Y, DEFINE
0099 0018 CA43          MOV R3,@NVS(9)    ;UPDATE NVS
     001A 000A'
0100                    *
0101 001C C129  FOR0    MOV @FNS(9),R4    ;GET F/N STACK ADR
     001E 0000
0102 0020 9838          CB *R8+,@B56      ;=?
     0022 0180'
0103 0024 1663            JNE FORE36      ;N, PROBLEM
0104                    *
0105 0026 C014  FOR1    MOV *R4,R0        ;DONE?
0106 0028 1314            JEQ FOR4        ;Y
0107 002A 8001          C R1,R0           ;N, SAME VARIABLE?
0108 002C 1306            JEQ FOR1A       ;Y
0109 002E 0224          AI R4,18          ;N, MOVE TO NEXT
     0030 0012
0110 0032 8A44          C R4,@EUS(9)      ;ANYMORE?
     0034 0000
0111 0036 1AF7            JL FOR1         ;Y
0112 0038 2F8B          DATA ERROR+11     ;N, STACK OVERFLOW
0113                    *
0114 003A 06A0  FOR1A   BL @FOR2          ;DELETE
     003C 0040'
0115 003E 10F3          JMP FOR1
0116                    *
0117 0040 C144  FOR2    MOV R4,R5         ;DELETE FROM STACK
0118 0042 C004          MOV R4,R0
0119 0044 0220          AI R0,18
     0046 0012
0120                    *
0121 0048 CD70  FOR3    MOV *R0+,*R5+     ;MOVE UP
0122 004A 8A40          C R0,@EUS(9)      ;DONE?
     004C 0034'
0123 004E 12FC            JLE FOR3        ;N
0124 0050 045B          RT                ;Y
```

```
0126 0052 CD01  FOR4    MOV R1,*R4+     ;INSERT VAR NAME
0127 0054 0420          BLWP @EVERZ     ;GET INITIAL VALUE
     0056 0000
0128 0058 0280          CI R0,>3800     ;TO?
     005A 3800
0129 005C 1649            JNE FORE37    ;N, ERROR
0130 005E C183          MOV R3,R6       ;SAVE FOR PRE-TEST
0131 0060 CCF2          MOV *R2+,*R3+   ;MOVE IN INITIAL VALUE
0132 0062 CCF2          MOV *R2+,*R3+
0133 0064 C4D2          MOV *R2,*R3
0134                *
0135 0066 04F4          CLR *R4+        ;SET DEFAULT STEP TO 1
0136 0068 04D4          CLR *R4
0137 006A 05B4          INC *R4+
0138 006C 04F4          CLR *R4+
0139                *
0140 006E 0420          BLWP @EVERZ     ;GET 'TO' VALUE
     0070 0056'
0141 0072 C1C4          MOV R4,R7       ;SAVE FOR PRE-TEST
0142 0074 CD32          MOV *R2+,*R4+   ;MOVE IN TERMINATING VALUE
0143 0076 CD32          MOV *R2+,*R4+
0144 0078 CD12          MOV *R2,*R4+
0145 007A 04C5          CLR R5          ;SET DEFAULT SIGN
0146 007C 0280          CI R0,>3A00     ;'STEP'?
     007E 3A00
0147 0080 160C            JNE FOR6      ;N
0148 0082 0420          BLWP @EVERZ     ;Y, GET STEP
     0084 0070'
0149 0086 C044          MOV R4,R1
0150 0088 0221          AI R1,-12       ;MOVE BACK TO STEP
     008A FFF4
0151 008C C152          MOV *R2,R5      ;GET DIRECTION
0152 008E 1602            JNE $+6       ;FP
0153 0090 C162          MOV @2(2),R5    ;INTEGER
     0092 0002
0154                *
0155 0094 CC72          MOV *R2+,*R1+   ;MOVE INTO STEP
0156 0096 CC72          MOV *R2+,*R1+
0157 0098 C452          MOV *R2,*R1
0158                *
0159 009A C508  FOR6    MOV R8,*R4      ;MOVE IN PBC,PLC
0160 009C 0634          DEC *R4+        ;BACKUP OVER DLIM
0161 009E C529          MOV @PLC(9),*R4
     00A0 0000
0162                *
0163 00A2 C086          MOV R6,R2       ;DO PRE-TEST
0164 00A4 C047          MOV R7,R1
0165 00A6 06A0          BL @SUBF        ;R2=R2-R1  (VAR-TERM)
     00A8 0000
0166 00AA C072          MOV *R2+,R1     ;LOOK AT RESULT, INTEGER?
0167 00AC 1602            JNE $+6       ;N
0168 00AE C052          MOV *R2,R1      ;Y
0169 00B0 131B            JEQ FOR9      ;= IMPLIES NOT DONE
0170                *
0171 00B2 2845          XOR R5,R1       ;'EXCLUSIVE OR' SIGNS
0172 00B4 1119            JLT FOR9      ;- INPLIES LOOP NOT COMPLETE
```

```
0174 00B6 C164  FOR7    MOV @-16(4),R5  ;LOOK FOR NEXT
     00B8 FFF0
0175 00BA 04E4          CLR @-16(4)     ;CLEAR FROM STACK
     00BC FFF0
0176 00BE C1A9          MOV @PLC(9),R6  ;GET PLC
     00C0 00A0'
0177                *
0178 00C2 0226  FOR8    AI R6,-4
     00C4 FFFC
0179 00C6 8A46          C R6,@SLT(9)    ;ANY MORE STATEMENTS?
     00C8 0000
0180 00CA 1A15            JL ERR31      ;N
0181 00CC C216          MOV *R6,R8      ;GET PBC
0182 00CE A229          A   @BUS(9),R8  ;GET ABSOLUTE ADDRESS
     00D0 0000
0183 00D2 9838          CB *R8+,@NXTXB  ;NEXT?
     00D4 0000
0184 00D6 16F5            JNE FOR8      ;N
0185 00D8 06A0          BL @GSIM        ;Y, GET SIMPLE VARIABLE
     00DA 015C'
0186 00DC 8141          C R1,R5         ;SAME?
0187 00DE 16F1            JNE FOR8      ;N, CONTINUE
0188 00E0 CA46          MOV R6,@PLC(9)  ;Y, UPDATE PLC
     00E2 00C0'
0189 00E4 0460          B @NLIN0
     00E6 0000
0190                *
0191                *FOR9    EQU DIM8
0192 00E8 0460  FOR9    B @NLIN
     00EA 0000
0193                *
0194 00EC 2FA0  FORE36  DATA ERROR2,36  ;MISSING "=
     00EE 0024
0195                *
0196 00F0 2FA0  FORE37  DATA ERROR2,37  ;ILLEGAL DELIMITER
     00F2 0025
0197                *
0198 00F4 2F8A  FORE10  DATA ERROR+10   ;STORAGE OVERFLOW
0199                *
0200 00F6 2F9F  ERR31   DATA ERROR+31   ;FOR W/O NEXT
0201                *
0202 00F8 2F94  ERR20   DATA ERROR+20   ;EXPECTING SIMPLE VARIABLE
```

```
0205            * ABSTRACT:
0206            *
0207            *       PROCESS THE FOOT OF A FOR/NEXT LOOP.
0208            *
0209            *       NEXT SEARCHS THE FOR/NEXT STACK (FNS)
0210            *       FOR A MATCHING SIMPLE VARIABLE.  THE
0211            *       STEP IS ADDED TO THE VARIABLE AND
0212            *       A COMPLETION CHECK IS MADE.  IF LESS
0213            *       THAN OR EQUAL, IT LOOPS BACK.  OTHERWISE,
0214            *       THE STACK VARIBLE IS SET TO ZERO,
0215            *       AND PROGRAM EXECUTION CONTINUES AFTER
0216            *       THE NEXT STATEMENT.
0217            *
0218            * CALLING SEQUENCE:
0219            *
0220            *       B @NXTY
0221            *
0222            *       EXIT TO NLIN
0223            *
0224            * EXCEPTIONS AND CONDITIONS:
0225            *
0226            *       EVALUATION ERRORS
0227            *       NEXT W/O FOR
0228            *
0229            * EXTERNAL ROUTINES:
0230            *
0231            *       (SEE EXTERNAL ROUTINE LIST)
0232            *
0233            * LOCAL DATA:  (NONE)
```

```
0235                    * ENTRY POINT
0236                    *
0237                            DEF NXTY
0238                    *
0239 00FA 06A0  NXTY    BL @GSIM          ;GET SIMPLE VARIABLE
     00FC 015C'
0240 00FE C194          MOV *R4,R6        ;GET ADR
0241 0100 1309            JEQ ERR32       ;NOT DEFINED
0242 0102 C129          MOV @FNS(9),R4    ;GET F/N STACK ADR
     0104 001E'
0243                    *
0244 0106 8D01  NXT1    C R1,*R4+         ;SAME?
0245 0108 1307            JEQ NXT2        ;Y
0246 010A 0224          AI R4,16          ;N, MOVE TO NEXT
     010C 0010
0247 010E 8A44          C R4,@EUS(9)      ;MORE?
     0110 004C'
0248 0112 1AF9            JL NXT1         ;Y
0249                    *
0250 0114 2FA0  ERR32   DATA ERROR+>20    ;NEXT W/O FOR
0251 0116 0020            DATA 32
0252                    *
0253 0118 C154  NXT2    MOV *R4,R5        ;GET SIGN
0254 011A 1602            JNE $+6         ;FP
0255 011C C164          MOV @2(4),R5      ;INTEGER
     011E 0002
0256                    *
0257 0120 C046          MOV R6,R1         ;READY R2,R1
0258 0122 C084          MOV R4,R2
0259 0124 06A0          BL @ADDF          ;ADD STEP
     0126 0000
0260 0128 C042          MOV R2,R1         ;MOVE IN NEW VARIABLE
0261 012A CDB1          MOV *R1+,*R6+
0262 012C CDB1          MOV *R1+,*R6+
0263 012E C591          MOV *R1,*R6
0264 0130 C044          MOV R4,R1
0265 0132 0221          AI R1,6           ;MOVE TO 'TO'
     0134 0006
0266 0136 06A0          BL @SUBF          ;R2=R2-R1  (VAR-TO)
     0138 00A8'
0267 013A C072          MOV *R2+,R1       ;LOOK AT RESULT, INTEGER?
0268 013C 1602            JNE $+6         ;N
0269 013E C052          MOV *R2,R1        ;Y
0270 0140 1306            JEQ NXT3        ;0, NOT DONE
0271                    *
0272 0142 2845          XOR R5,R1         ;EXCLUSIVE OR WITH STEP
0273 0144 1104            JLT NXT3        ;NOT DONE
0274 0146 04E4          CLR @-2(4)        ;DONE, SET TOP OF STACK
     0148 FFFE
0275 014A 0460          B @NLIN0
     014C 00E6'
0276                    *
0277 014E 0224  NXT3    AI R4,12          ;LOOP NOT COMPLETE
     0150 000C
0278 0152 C234          MOV *R4+,R8       ;RESTORE PBC & PLC
0279 0154 CA54          MOV *R4,@PLC(9)
     0156 00E2'
0280 0158 0460          B @NLIN0
     015A 014C'
```

```
0283                * ABSTRACT:
0284                *
0285                *       GSIM WILL GET NEXT ITEM AND CHECK TO
0286                *       SEE IF IT IS A SIMPLE VARIABLE.
0287                *
0288                * CALLING SEQUENCE:
0289                *
0290                *       BL @GSIM
0291                *
0292                *       OUT (R4) = VARIABLE DEFINITION
0293                *
0294                * EXCEPTIONS AND CONDITIONS:
0295                *
0296                *       EXPECTING SIMPLE VARIABLE
0297                *
0298                * EXTERNAL ROUTINES:  (NONE)
0299                *
0300                * LOCAL DATA:  (NONE)
0301                *
0302                * ENTRY POINT:
0303                *
0304                        DEF GSIM
0305                *
0306 015C 04C0   GSIM   CLR R0          ;GET VARIABLE
0307 015E D038          MOVB *R8+,R0
0308 0160 0280          CI R0,NRV*>100+>7000
     0162 7400
0309 0164 1AC9            JL ERR20      ;N, EXPECTING SIMPLE VARIABLE
0310 0166 C129          MOV @VNT(9),R4  ;GET SYMBOL TABLE ADR
     0168 0000
0311 016A 0970          SRL R0,7        ;GET INDEX
0312 016C A100          A R0,R4
0313 016E C064          MOV @->70*2(4),R1
     0170 FF20
0314 0172 11C2            JLT ERR20     ;DIMENSIONED, ERROR
0315 0174 C129          MOV @VDT(9),R4
     0176 0000
0316 0178 A100          A R0,R4
0317 017A 0224          AI R4,->70*2
     017C FF20
0318 017E 045B          RT
0319                *
0320 0180   56  856     BYTE >56
0321 0182               EVEN
0322                     END
NO ERRORS,     NO WARNINGS
```

```
$          0182'          0152  0167  0254  0268
ADDF    R  0126'  0019    0259
B56        0180'  0320    0102
BUS     R  00D0'  0028    0182
ERR20      00F8'  0202    0309  0314
ERR31      00F6'  0200    0180
ERR32      0114'  0250    0241
ERROR      2F80   0023    0024  0112  0198  0200  0202  0250
ERROR2     2FA0   0024    0194  0196
EUS     R  0110'  0029    0110  0122  0247
EVERZ   R  0084'  0021    0127  0140  0148
FNS     R  0104'  0030    0101  0242
FOR0       001C'  0101    0093
FOR1       0026'  0105    0111  0115
FOR1A      003A'  0114    0108
FOR2    D  0040'  0117    0089  0114
FOR3       0048'  0121    0123
FOR4       0052'  0126    0106
FOR6       009A'  0159    0147
FOR7       00B6'  0174
FOR8       00C2'  0178    0184  0187
FOR9       00E8'  0192    0169  0172
FORE10     00F4'  0198    0097
FORE36     00EC'  0194    0103
FORE37     00F0'  0196    0129
FORY    D  0000'  0091    0089
GSIM    D  015C'  0306    0091  0185  0239  0304
NEIN    R  00EA'  0022    0192
NLIN0   R  015A'  0022    0189  0275  0280
NRV        0004   0040    0308
NVD     R  0012'  0031    0096
NVS     R  001A'  0032    0094  0099
NXT1       0106'  0244    0248
NXT2       0118'  0253    0245
NXT3       014E'  0277    0270  0273
NXTXB   R  00D4'  0041    0183
NXTY    D  00FA'  0239    0237
PLC     R  0156'  0033    0161  0176  0188  0279
R0         0000           0105  0107  0118  0119  0121  0122  0128  0146  0306
                          0307  0308  0311  0312  0316
R1         0001           0107  0126  0149  0150  0155  0156  0157  0164  0166
                          0168  0171  0186  0244  0257  0260  0261  0262  0263
                          0264  0265  0267  0269  0272  0313
R2         0002           0131  0132  0133  0142  0143  0144  0151  0155  0156
                          0157  0163  0166  0168  0258  0260  0267  0269
R3         0003           0092  0094  0095  0096  0098  0099  0130  0131  0132
                          0133
R4         0004           0092  0098  0101  0105  0109  0110  0117  0118  0126
                          0135  0136  0137  0138  0141  0142  0143  0144  0149
                          0159  0160  0161  0240  0242  0244  0246  0247  0253
                          0258  0264  0277  0278  0279  0310  0312  0315  0316
                          0317
R5         0005           0117  0121  0145  0151  0153  0171  0174  0186  0253
                          0255  0272
R6         0006           0130  0163  0176  0178  0179  0181  0188  0240  0257
                          0261  0262  0263
R7         0007           0141  0164
R8         0008           0102  0159  0181  0182  0183  0278  0307
SLT     R  00C8'  0034    0179
SUBF    R  0138'  0020    0165  0266
```

```
VDT     R  0176'    0035  0315
VNT     R  0168'    0036  0310
*
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.DEF
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.DEF
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.DEF
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
        IDT 'DEF'
0002            *
0003            * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004            *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005            *
0006            * ROUTINE LIST:
0007            *
0008            *       DEFY              ;DEF COMMAND
0009            *
0010            * COPY: (NONE)
0011            *
0012            * MARCO: (NONE)
0013            *
0014            * EXTERNAL ROUTINES:
0015            *
0016                  REF LINE            ;MULTIPLEXOR
0017            *
0018            * EXTERNAL DATA:
0019            *
0020                  REF UFT             ;USER FUNCTION TABLE
0021                  REF MODE            ;MODE FLAG
0022            *
0023            * MODULE EQUATES:  (NONE)
0024            *
0025            * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0028                * ABSTRACT:
0029                *
0030                *       PLACE THE USER FUNCTION PROGRAM-
0031                *       BYTE-COUNTER AND NUMBER OF ARGUMENTS
0032                *       IN USER-FUNCTION-TABLE (UFT) THUS
0033                *       DEFINING THE FUNCTION.
0034                *
0035                * CALLING SEQUENCE:
0036                *
0037                *       B @DEFY
0038                *
0039                * EXCEPTIONS AND CONDITIONS:  (NONE)
0040                *
0041                * EXTERNAL ROUTINE LIST:
0042                *
0043                *      (SEE EXTERNAL ROUTINES)
0044                *
0045                * LOCAL DATA:
0046                *
0047                *      (SEE MODULE VARIABLES AND CONSTANTS)
0048                *
0049                * ENTRY POINT:
0050                *
0051                      DEF DEFY
0052                *
0053                *DEF COMMAND
0054                *
0055                *USER FUNCTION TABLE:
0056                *
0057                *       PBC
0058                *       # OF ARGUMENTS
0059                *       ...
0060                *
0061 0000 C029     DEFY     MOV @MODE(9),R0 ;RUNNING?
     0002 0000
0062 0004 130D               JEQ DEF2       ;N, DON'T DEFINE
0063 0006 04C0               CLR R0
0064 0008 D038               MOVB *R8+,R0   ;GET TYPE
0065 000A 0701               SETO R1        ;START COUNT
0066                *
0067 000C 0581     DEF1      INC R1         ;COUNT ARGUMENTS
0068 000E 9838               CB *R8+,@B56   ;=?
     0010 0024'
0069 0012 16FC               JNE DEF1       ;N
0070 0014 C0E9               MOV @UFT(9),R3 ;Y, DEFINE FUNCTION
     0016 0000
0071 0018 0960               SRL R0,6       ;GET INDEX
0072 001A A0C0               A R0,R3        ;INDEX
0073 001C CCC8               MOV R8,*R3+    ;SAVE ADR
0074 001E C4C1               MOV R1,*R3     ;SAVE COUNT
0075 0020 0460     DEF2      B @LINE        ;GOTO NEXT LINE
     0022 0000
0076 0024   56     B56       BYTE >56
0077 0026                    EVEN
0078                         END
```

NO ERRORS, NO WARNINGS

```
B56          0024'   0076  0068
DEF1         000C'   0067  0069
DEF2         0020'   0075  0062
DEFY    D    0000'   0061  0051
LINE    R    0022'   0016  0075
MODE    R    0002'   0021  0061
R0           0000            0061  0063  0064  0071  0072
R1           0001            0065  0067  0074
R3           0003            0070  0072  0073  0074
R8           0008            0064  0068  0073
UFT     R    0016'   0020    0070
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.UNIT
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.UNIT
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.UNIT
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT  'UNIT'
0002                 *
0003                 * REVISION: 12/31/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *
0006                 * ROUTINE LIST:
0007                 *
0008                 *       UNIT              ;SET OUTPUT UNIT (DEVICE SELECT)
0009                 *
0010                 * COPY: (NONE)
0011                 *
0012                 * MACRO: (NONE)
0013                 *
0014                 * EXTERNAL ROUTINES:
0015                 *
0016                       REF NLIN            ;RETURN TO MULTIPLEXOR
0017                       DXOP EVFIX,11       ;EVALUATE AND FIX
0018           2F80  ERROR  EQU >2F80           ;XOP XX,14  (ERROR CALL)
0019                 *
0020                 * EXTERNAL DATA:
0021                 *
0022                       REF UNIT            ;INDEX TO UNIT
0023                       REF UMSK            ;UNIT MASK (ALLOWABLE DEVICES)
0024                 *
0025                 * MODULE EQUATES: (NONE)
0026                 *
0027                 * MODULE VARIABLES AND CONSTANTS: (NONE)
0028                 *
```

```
0031            * ABSTRACT:
0032            *
0033            *       THE UNIT COMMAND IS USED TO SELECT OUTPUT DEVICES
0034            *       ACCORDING TO BIT POSITIONS WITHIN @UNIT(9).
0035            *       THE UNIT MASK (UMSK) DETERMINES VALID OUTPUT
0036            *       DEVICES.
0037            *
0038            * CALLING SEQUENCE:
0039            *
0040            *       B @UNTY
0041            *
0042            *       NORMAL EXIT TO NLIN
0043            *
0044            * EXCEPTIONS AND CONDITIONS:
0045            *
0046            *       INVALID OUTPUT DEVICE RESULTS IN ERROR 21
0047            *
0048            * EXTERNAL ROUTINE LIST:
0049            *
0050            *      (SEE EXTERNAL ROUTINES)
0051            *
0052            * LOCAL DATA:
0053            *
0054            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0056                * ENTRY POINT:
0057                *
0058                        DEF UNTY
0059                *
0060 0000 2EC1  UNTY    EVFIX R1          ;GET UNIT
0061 0002 0202          LI R2,UMSK        ;GET MASK
     0004 0000
0062 0006 2081          COC R1,R2         ;VALID UNIT(S)?
0063 0008 1604            JNE ERR21       ;N, ERROR
0064 000A CA41          MOV R1,@UNIT(9)
     000C 0000
0065 000E 0460          B @NLIN
     0010 0000
0066                *
0067 0012 2F95  ERR21   DATA ERROR+21
0068                    END
NO ERRORS,      NO WARNINGS
```

```
ERR21       0012'   0067  0063
ERROR       2F80    0018  0067
EVFIX               0017
NLIN    R   0010'   0016  0065
R1          0001          0060  0062  0064
R2          0002          0061  0062
UMSK    R   0004'   0023  0061
UNIT    R   000C'   0022  0064
UNTY    D   0000'   0060  0058
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.RANDOM
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.RANDOM
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.RANDOM
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'RANDOM'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       RANY              ;RANDOM COMMAND
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MARCO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:
0015              *
0016                    REF NLIN            ;EXIT TO MULTIPLEXOR
0017                    DXOP EVFIX,11       ;EVALUATE AND FIX
0018              *
0019              * EXTERNAL DATA:
0020              *
0021                    REF RANDS           ;RANDOM NUMBER SEED
0022              *
0023              * MODULE EQUATES:  (NONE)
0024              *
0025              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0028                * ABSTRACT:
0029                *
0030                *       THE RANDOM COMMAND WILL SET THE RANDOM
0031                *       SEED.
0032                *
0033                * CALLING SEQUENCE:
0034                *
0035                *       B @RANY
0036                *
0037                *       EXIT TO NLIN
0038                *
0039                * EXCEPTIONS AND CONDITIONS:  (NONE)
0040                *
0041                * EXTERNAL ROUTINE LIST:
0042                *
0043                *      (SEE EXTERNAL ROUTINES)
0044                *
0045                * LOCAL DATA:
0046                *
0047                *      (SEE MODULE VARIABLES AND CONSTANTS)
0048                *
0049                * ENTRY POINT:
0050                *
0051                      DEF RANY
0052                *
0053 0000 2EE0     RANY    EVFIX @RANDS    ;GET RANDOM SEED
     0002 0000
0054 0004 0460             B @NLIN
     0006 0000
0055                       END
NO ERRORS,     NO WARNINGS
```

```
EVFIX              0017
NLIN    R  0006'   0016  0054
RANDS   R  0002'   0021  0053
RANY    D  0000'   0053  0051
*
```

```
SOURCE ACCESS NAME=      DBASIC.REL0.BASIC.SOURCE.ERROR
OBJECT ACCESS NAME=      DBASIC.REL0.BASIC.OBJECT.ERROR
LISTING ACCESS NAME=     DBASIC.REL0.BASIC.LIST.ERROR
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
 0001                    IDT  'ERROR'
 0002              *     08/11/78
 0003              *
 0004                    DEF ERRY
 0005                    REF EFLG
 0006                    REF NLIN
 0007                    DXOP EVFIX,11
 0008              *
 0009              *ERROR COMMAND
 0010              *
 0011 0000 2EE9    ERRY   EVFIX @EFLG(9)  ;GET LINE NUMBER
      0002 0000
 0012 0004 0460           B @NLIN
      0006 0000
 0013                    END
NO ERRORS
```

```
EFLG    R  0002'   0005  0011
ERRY    D  0000'   0011  0004
EVFIX              0007
NLIN    R  0006'   0006  0012
*
```

```
SOURCE ACCESS NAME=      DBASIC.REL0.BASIC.SOURCE.ESCAPE
OBJECT ACCESS NAME=      DBASIC.REL0.BASIC.OBJECT.ESCAPE
LISTING ACCESS NAME=     DBASIC.REL0.BASIC.LIST.ESCAPE
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'ESCAPE'
0002              *     08/11/78
0003              *
0004                    DEF ESCY
0005                    DEF NOEY
0006                    REF MODE
0007                    REF NLIN0
0008              *
0009              *ESCAPE ENABLE COMMAND
0010              *
0011 0000 C029    ESCY  MOV @MODE(9),R0 ;IDLE?
     0002 0000
0012 0004 1302          JEQ ESCR      ;Y, RETURN
0013 0006 0729          SETO @MODE(9) ;N, ENABLE ESCAPES
     0008 0002'
0014 000A 0460    ESCR  B @NLIN0
     000C 0000
0015              *
0016              *
0017              *ESCAPE DISABLE COMMAND
0018              *
0019 000E 0769    NOEY  ABS @MODE(9)  ;CLEAR UPPER BIT
     0010 0008'
0020 0012 10FB          JMP ESCR
0021                    END
NO ERRORS
```

```
ESCR       000A'   0014  0012  0020
ESCY    D  0000'   0011  0004
MODE    R  0010'   0006  0011  0013  0019
NLINO   R  000C'   0007  0014
NOEY    D  000E'   0019  0005
R0         0000           0011
*
```

```
SOURCE ACCESS NAME=      DBASIC.REL0.BASIC.SOURCE.CALL
OBJECT ACCESS NAME=      DBASIC.REL0.BASIC.OBJECT.CALL
LISTING ACCESS NAME=     DBASIC.REL0.BASIC.LIST.CALL
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                        IDT  'CALL'
0002              *         09/20/78
0003              *
0004                        DEF CLLY
0005                        REF EVERZ
0006                        REF EVSDZ
0007                        REF NLIN
0008                        REF WPR103
0009                        REF WPR108
0010       2F80   ERROR     EQU >2F80
0011                        DXOP EVFIX,11
0012              *CALL COMMAND
0013              *
0014 0000 0420    CLLY      BLWP @EVSDZ       ;CHECK FOR "STRING"
     0002 0000
0015 0004 1002              JMP CLL1          ;IS "STRING"
0016 0006 1000              NOP               ;IS $VAR - ERROR 1
0017 0008 2F81              DATA ERROR+1      ;NOT EITHER - ERROR 1
0018              *
0019 000A 020C    CLL1      LI R12,WPR103     ;GET (R3)
     000C 0000
0020              *
0021 000E 2EFC    CLL2      EVFIX *R12+       ;GET ADDRESS
0022              *
0023 0010 0280    CLL3      CI R0,>3F00       ;,?
     0012 3F00
0024 0014 160A                JNE CLL4        ;N
0025 0016 028C              CI R12,WPR108     ;(R8)?
     0018 0000
0026 001A 1407                JHE CLL4        ;Y
0027 001C 9818              CB *R8,@B4C       ;(?
     001E 0030'
0028 0020 16F6                JNE CLL2        ;N
0029 0022 0420              BLWP @EVERZ       ;Y, GET ADDRESS
     0024 0000
0030 0026 CF02              MOV R2,*R12+      ;STORE
0031 0028 10F3              JMP CLL3
0032              *
0033 002A 0693    CLL4      BL *R3            ;BRANCH AND LINK TO ROUTINE
0034 002C 0460              B @NLIN
     002E 0000
0035 0030   4C    B4C       BYTE >4C
0036 0032                   EVEN
0037                        END
NO ERRORS
```

LABEL		VALUE	DEFN	REFERENCES			
B4C		0030'	0035	0027			
CLL1		000A'	0019	0015			
CLL2		000E'	0021	0028			
CLL3		0010'	0023	0031			
CLL4		002A'	0033	0024	0026		
CLLY	D	0000'	0014	0004			
ERROR		2F80	0010	0017			
EVERZ	R	0024'	0005	0029			
EVFIX			0011				
EVSDZ	R	0002'	0006	0014			
NLIN	R	002E'	0007	0034			
R0		0000		0023			
R12		000C		0019	0021	0025	0030
R2		0002		0030			
R3		0003		0033			
R8		0008		0027			
WPR103	R	000C'	0008	0019			
WPR108	R	0018'	0009	0025			

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.BASE
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.BASE
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.BASE
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'BASE'
0002          *
0003          * REVISION: 12/01/78 1.01     ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00     ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       BASY              ;BASE COMMAND
0009          *
0010          * COPY: (NONE)
0011          *
0012          * MARCO: (NONE)
0013          *
0014          * EXTERNAL ROUTINES:
0015          *
0016                REF NLIN            ;EXIT TO MULTIPLEXOR
0017                DXOP EVFIX,11       ;EVALUATE AND FIX
0018          *
0019          * EXTERNAL DATA:
0020          *
0021                REF BCRU            ;BASE CRU INDEX
0022          *
0023          * MODULE EQUATES:  (NONE)
0024          *
0025          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0028                * ABSTRACT:
0029                *
0030                *       THE BASE COMMAND WILL SET A CRU BASE
0031                *       THAT IS USED BY CRM AND CRF
0032                *       FUNCTIONS:
0033                *
0034                * CALLING SEQUENCE:
0035                *
0036                *       B @BASY
0037                *
0038                *       EXIT TO NLIN
0039                *
0040                * EXCEPTIONS AND CONDITIONS:
0041                *
0042                *       EVALUATION ERRORS
0043                *
0044                * EXTERNAL ROUTINE LIST:
0045                *
0046                *      (SEE EXTERNAL ROUTINES)
0047                *
0048                * LOCAL DATA:
0049                *
0050                *      (SEE MODULE VARIABLES AND CONSTANTS)
0051                *
0052                * ENTRY POINT:
0053                *
0054                        DEF BASY
0055                *
0056 0000 2EE9  BASY    EVFIX @BCRU(9)  ;SAVE
     0002 0000
0057 0004 0460          B @NLIN
     0006 0000
0058                    END
NO ERRORS,     NO WARNINGS
```

```
BASY    D  0000'   0056  0054
BCRU    R  0002'   0021  0056
EVFIX              0017
NLIN    R  0006'   0016  0057
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL3.BASIC.SOURCE.TIME
OBJECT ACCESS NAME=      BD1.DBASIC.REL3.BASIC.OBJECT.TIME
LISTING ACCESS NAME=     BD1.DBASIC.REL3.BASIC.LIST.TIME
ERROR ACCESS NAME=
OPTIONS=
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'TIME'
0002                *   28 MARCH 1979
0003                *
0004                    DEF TIMY
0005                    REF B20
0006                    REF CKEX
0007                    REF CLKADR
0008                    REF EVSDZ
0009                    REF NLIN
0010                    REF TYPB
0011                    REF FPWP
0012                    REF OWNDO          ;HJC -27 MARCH 1979
0013                    REF CWNDO          ;HJC -27 MARCH 1979
0014                    REF NB             ;HJC -27 MARCH 1979
0015                    DXOP EVFIX,11
0016                *TIME COMMAND
0017                *
0018 0000 0203  TIMY    LI R3,CLKADR       ;GET CLOCK ADDRESS
     0002 0000
0019 0004 020C          LI R12,2*>80       ;SELECT 9901
     0006 0100
0020                *
0021 0008 0420          BLWP @EVSDZ        ;LOOK FOR STRING
     000A 0000
0022 000C 103B            JMP TIM5         ;PROBLEM
0023 000E 102B            JMP TIM3         ;$
0024 0010 06A0          BL @CKEX           ;LOOK FOR EXPRESSION
     0012 0000
0025 0014 1022            JMP TIM2         ;NONE
0026                *
0027 0016 2EF3  TIM0    EVFIX *R3+         ;STORE RESULT
0028 0018 0280          CI R0,>3F00        ;,?
     001A 3F00
0029 001C 13FC            JEQ TIM0         ;Y, CONTINUE
0030                *
0031 001E 33E0  TIM1    LDCR @TIMC,15      ;LOAD CLOCK FOR 9375 = 200 MS
     0020 00C4'
0032 0022 1E00          SBZ 0              ;SELECT INTERRUPT MODE
0033 0024 1D03          SBO 3              ;ENABLE INTERRUPT
0034 0026 02C1          STST R1
0035 0028 0241          ANDI R1,>F
     002A 000F
0036 002C 8801          C  R1,@CONST3      ;MASK>=3?
     002E 00C6'
0037 0030 142B            JHE TIMR         ;Y, LEAVE
0038 0032 0300          LIMI >3            ;ENABLE CLOCK INTERRUPT
     0034 0003
0039                ************************************************************
0040                *              CHANGES TO SET OPEN WINDOW TO GREATER OF 3 OR
0041                *              LEVEL OF OPEN WINDOW -HJC 27 MARCH 79
0042                ************************************************************
0043 0036 8820          C   @OWNDO,@CONST3
     0038 0000
     003A 00C6'
0044 003C 1503          JGT TIM1A
0045 003E C820          MOV @CONST3,@OWNDO
     0040 00C6'
     0042 0038'
0046                ************************************************************
0047                *              SET CLOSE WINDOW TO LESSER OF 3 OR NB-HJC 27 M
```

```
0048                  ************************************************************
0049 0044 C820  TIM1A   MOV @CONST3,@CWNDO
     0046 00C6'
     0048 0000
0050 004A 8820          C   @CWNDO,@NB
     004C 0048'
     004E 0000
0051 0050 1103          JLT TIM1B
0052 0052 C820          MOV @NB,@CWNDO
     0054 004E'
     0056 004C'
0053      0058' TIM1B   EQU $
0054                  *
0055 0058 1017          JMP TIMR          ;RETURN
0056                  *
0057 005A 0588  TIM2    INC R8            ;INCREMENT OVER DLIM
0058 005C C099          MOV *R9,R2        ;GET IOB ADDRESS
0059 005E DCA0          MOVB @B20,*R2+    ;OUT SPACE
     0060 0000
0060 0062 0706          SETO R6           ;SET TO PRINT
0061 0064 1001          JMP TIM4
0062                  *
0063 0066 04C6  TIM3    CLR R6            ;STORE IN VARIABLE
0064                  *
0065 0068 0420  TIM4    BLWP @TIMVEC      ;GET TIME
     006A 00AC'
0066 006C 06A0          BL @TIM6          ; DO HOURS
     006E 008C'
0067 0070 C004          MOV R4,R0
0068 0072 06A0          BL @TIM6          ;DO MINUTES
     0074 008C'
0069 0076 C005          MOV R5,R0
0070 0078 06A0          BL @TIM6          ;DO SECONDS
     007A 008C'
0071 007C 0602          DEC R2            ;END STRING
0072 007E 7492          SB *R2,*R2
0073 0080 C186          MOV R6,R6         ;TYPE?
0074 0082 1302            JEQ TIMR        ;N, RETURN
0075                  *
0076 0084 06A0  TIM5    BL @TYPB          ;Y, OUTPUT BUFFER
     0086 0000
0077 0088 0460  TIMR    B @NLIN
     008A 0000
0078                  *
0079                  *FIX R0 TO XX:
0080                  *
0081 008C 04C1  TIM6    CLR R1            ;CLEAR COUNTER
0082                  *
0083 008E 0220  TIM7    AI R0,-10         ;NEGATIVE
     0090 FFF6
0084 0092 1102            JLT TIM8        ;Y
0085 0094 0581          INC R1            ;N, COUNT
0086 0096 10FB          JMP TIM7
0087                  *
0088 0098 06C1  TIM8    SWPB R1
0089 009A D001          MOVB R1,R0        ;APPEND
0090 009C 0220          AI R0,>2F3A       ;ADD ASCII CODES
     009E 2F3A
0091 00A0 DC80          MOVB R0,*R2+      ;MOVE INTO OUTPUT STREAM
0092 00A2 06E0          SWPB R0
```

```
0093 00A4 DC80          MOVB R0,*R2+
0094 00A6 DCA0          MOVB @B3A,*R2+   ;OUT ":
     00A8 00C8'
0095 00AA 0458          RT
0096              *
0097      00AC' TIMVEC  EQU $
0098 00AC 0000          DATA FPWP,TIM9
     00AE 00B0'
0099              *
0100      00B0' TIM9    EQU  $
0101 00B0 0300          LIMI 0                    MASK INTRPTS
     00B2 0000
0102 00B4 0203          LI   R3,@CLKADR           GET SOURCE
     00B6 0002'
0103 00B8 C773          MOV  *R3+,*R13            SET HOURS
0104 00BA CB73          MOV  *R3+,@8(R13)         SET MIN
     00BC 0008
0105 00BE CB53          MOV  *R3,@10(R13)         SET SECONDS
     00C0 000A
0106 00C2 0380          RTWP                      AND RETURN
0107              *
0108 00C4 0EA7  TIMC    DATA 1875*2+1   1875 = 40 MS
0109 00C6 0003  CONST3  DATA >03
0110 00C8   3A  B3A     BYTE >3A
0111 00CA               EVEN
0112                    END
NO ERRORS,      NO WARNINGS
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.EVAL
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.EVAL
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.EVAL
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT  'EVAL'
0002                 *
0003                 * REVISION: 12/31/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *           30 MAY 79          ;COPY AND UPDATE TO REL5
0006                 * ROUTINE LIST:
0007                 *
0008                 *       EVARZ              ;EVALUATE VARIABLE
0009                 *       EVERZ              ;EVALUATE EXPRESSION
0010                 *       EVSDZ              ;EVALUATE STRING
0011                 *       EVFX               ;EVALUATE AND FIX
0012                 *       CKEX               ;CHECK FOR EXPRESSION
0013                 *
0014                 *       EVAL               ;RECURSIZE EVALUATOR
0015                 *       EVALS2             ;2ND ENTRY IN RECURSIVE EVALUATOR
0016                 *       ADDF               ;ADD TWO VARIABLES
0017                 *       SUBF               ;SUBTRACT TWO VARIABLES
0018                 *
0019                 * COPY: (NONE)
0020                 *
0021                 * MARCO: (NONE)
0022                 *
0023                 * EXTERNAL ROUTINES:
0024                 *
0025                       DXOP LOADF,0     ;LOAD FPAC
0026                       DXOP STORE,1     ;STORE FPAC
0027                       DXOP FADD,2              ;ADD TO FPAC
0028                       DXOP FSUB,3              ;SUBTRACT FROM FPAC
0029                       DXOP FMUL,4      ;MULTIPLY FPAC
0030                       DXOP FDIV,5      ;DIVIDE FPAC
0031                       DXOP SCALE,6     ;SCALE FPAC
0032                       DXOP NORMAL,7    ;NORMALIZE FPAC
0033                       DXOP CLEAR,8     ;CLEAR FPAC
0034                       DXOP NEGATE,9    ;NEGATE FPAC
0035                       DXOP FLOATF,10   ;FLOAT FPAC
0036                       DXOP EVFIX,11    ;EVALUATE AND FIX
0037                       DXOP OUTFP,12    ;OUT FLOATING POINT #
0038                       DXOP OUTINT,13   ;OUT INTEGER
0039    2F80   ERROR       EQU >2F80        ;XOP XX,14  (ERROR CALL)
0040    2FA0   ERROR2      EQU ERROR+>20
0041                 *
0042                       REF ABSF         ;ABSOLUTE VALUE FUNCTION
0043                       REF ASCF         ;CONVERT ASCII FUNCTION
0044                       REF ATNF         ;ARC-TANGENT FUNCTION
0045                       REF COSF         ;COSINE FUNCTION
0046                       REF EXPF         ;EXPONENTIAL FUNCTION
0047                       REF INPF         ;INTEGER PART FUNCTION
0048                       REF LOGF         ;LOG FUNCTION
0049                       REF NKYF         ;KEY FUNCTION
0050                       REF SINF         ;SINE FUNCTION
0051                       REF SQRF         ;SQUARE ROOT FUNCTION
0052                       REF SYSF         ;SYSTEM FUNCTION
0053                       REF TICF         ;TIC FUNCTION
0054                       REF BITF         ;BIT FUNCTION
0055                       REF CRBF         ;CRB FUNCTION
0056                       REF CRFF         ;CRF FUNCTION
0057                       REF MEMF         ;MEM FUNCTION
0058                       REF MWDF         ;MEMORY WORD FUNCTION
0059                       REF LENF         ;STRING LENGTH FUNCTION
0060                       REF MCHF         ;STRING MATCH FUNCTION
```

```
0061                    REF SRHF          ;STRING SEARCH FUNCTION
0062                    REF POWF          ;POWER FUNCTION
0063                    REF LORF          ;LOGICAL OR FUNCTION
0064                    REF LXORF         ;LOGICAL EXCLUSIVE OR FUNCTION
0065                    REF LANDF         ;LOGICAL AND FUNCTION
0066                    REF ANDF          ;AND FUNCTION
0067                    REF ORF           ;OR FUNCTION
0068                    REF NOTF          ;NOT FUNCTION
0069                    REF LNOTF         ;LOGICAL NOT FUNCTION
0070                *
0071                    REF FIX           ;FIX ROUTINE
0072                    REF FLOAT         ;FLOAT ROUTINE
0073                    REF RANDZ         ;GET RANDOM NUMBER ROUTINE
0074                *
0075                * EXTERNAL DATA:
0076                *
0077                    REF EVSKB,EVSKE   ;EVALUATION STACK
0078                    REF FPAC,FPAC2    ;FLOATING POINT ACCUMULATOR
0079                    REF TEMP          ;TEMP FLOATING POINT REGISTER
0080                    REF TEMP2,TEMP4
0081                    REF FUZZ          ;FUZZ VALUE
0082                    REF UFT           ;USER-FUNCTION-TABLE INDEX
0083                    REF VDT           ;VARIABLE-DEFINITION-TABLE INDEX
0084                    REF VNT           ;NVARIABLE-NAME-TABLE INDEX
0085                    REF IOB           ;I/O-BUFFER INDEX
0086                    REF DLIM          ;DELIMITER INDEX
0087                    REF NVD           ;NEXT-VARIABLE-DEFINITION INDEX
0088                    REF NVS           ;NEXT-VARIABLE-STORAGE INDEX
0089                    REF WPR2          ;SECONDARY WORKSPACE
0090                *
0091                * MODULE EXTERNAL DEFINITIONS:
0092                *
0093                    DEF EVSFR         ;RETURN FOR FUNCTIONS
0094                    DEF EVOP3A        ;RETURN FOR OPERATORS
0095                    DEF B01,B05       ;BYTES
0096                    DEF C1,C4,C6      ;WORDS
0097                *
0098                * MODULE EQUATES:  (NONE)
0099                *
0100                * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0103            * ABSTRACT:
0104            *
0105            *       EVSD EVALUATES FOR A STRING CONSTANT OR VARIABLE
0106            *       BEGINNING AT THE PBC (R8).  MULTIPLY RETURNS ARE
0107            *       USED ACCORDING TO WHAT IS FOUND.  IF NO STRING
0108            *       OR VARIABLE IS FOUND, THE PBC (PROGRAM BYTE COUNTER)
0109            *       IS LEFT UNCHANGED.
0110            *
0111            * CALLING SEQUENCE:
0112            *
0113            *       BLWP @EVSDZ
0114            *         STRING CONSTANT RETURN ("---")
0115            *         STRING VARIABLE RETURN ($VAR)
0116            *       NEITHER
0117            *
0118            *       IN  R8 = PBC (PROGRAM BYTE COUNTER)
0119            *           R9 = PTRS
0120            *       OUT R0 = DLIM (DELIMITER
0121            *           R2 = PTR TO STRING
0122            *           R8 = PBC UPDATED
0123            *
0124            * EXCEPTIONS AND CONDITIONS:
0125            *
0126            *       ERRORS INCLUDE EXPECTING VARIABLE, SYNTAX ERROR,
0127            *       UNDEFINED FUNCTION, ILLEGAL DELIMITER, STORAGE
0128            *       OVERFLOW, UNDEFINED VARIABLE, EXPRESSION TOO
0129            *       COMPLEX, SUBSCRIPT ERROR, TOO MANY SUBSCRIPTS,
0130            *       UNDIMENSIONED VARIABLE, AND ALL ARITHMETIC ERRORS.
0131            *
0132            * EXTERNAL ROUTINE LIST:
0133            *
0134            *      (SEE EXTERNAL ROUTINES)
0135            *
0136            * LOCAL DATA:
0137            *
0138            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0140                    * ENTRY POINT:
0141                    *
0142                            DEF EVSDZ
0143                    *
0144 0000 0000  EVSDZ   DATA WPR2,EVSD
     0002 0004'
0145 0004 C22D  EVSD    MOV @16(13),R8  ;GET PBC
     0006 0010
0146 0008 9818          CB *R8,@B43     ;$?
     000A 053B'
0147 000C 130E            JEQ EVSD2     ;Y
0148 000E 1A0B            JL EVSD1      ;NOT STRING
0149 0010 9838          CB *R8+,@B45    ;" OR '?
     0012 053C'
0150 0014 1B08            JH EVSD1      ;N
0151 0016 C088          MOV R8,R2       ;Y
0152                    *
0153 0018 D038          MOVB *R8+,R0    ;LOOK FOR "0
0154 001A 16FE            JNE $-2
0155                    *
0156 001C 04C0          CLR R0
0157 001E D038          MOVB *R8+,R0    ;RETURN DELIMITER
0158 0020 C26D          MOV @18(13),R9
     0022 0012
0159 0024 101D          JMP EVER1
0160                    *
0161 0026 8FBE  EVSD1   C *R14+,*R14+   ;NOT STRING, RETURN 4(14)
0162 0028 0380          RTWP
0163                    *
0164 002A 05CE  EVSD2   INCT R14
0165 002C 05AD          INC @16(13)     ;SKIP $
     002E 0010
0166                    *
0167                    *  THIS SHOULD FALL INTO EVAR
```

```
0170            * ABSTRACT:
0171            *
0172            *       EVAR CALLS THE EXPRESSION EVALUATOR FOR
0173            *       ONE ITEM ONLY, NAMELY A VARIABLE.  WITH
0174            *       R5 ZEROED, THE EVALUATOR WILL EVALUATE
0175            *       UP TO THE FIRST OPERATER OR DELIMITER.
0176            *       UPON RETURNING, EVAR WILL CHECK TO SEE
0177            *       IF THE RESULT IS IN THE EXPRESSION STACK
0178            *       IN WHICH CASE AN EXCEPTION WOULD OCCUR.
0179            *
0180            * CALLING SEQUENCE:
0181            *
0182            *       BLWP @EVARZ
0183            *       NORMAL RETURN
0184            *
0185            *       IN  R8 = PBC
0186            *           R9 = PTRS
0187            *       OUT R0 = DLIM
0188            *           R2 = ADR
0189            *           R8 = PBC UPDATED
0190            *
0191            * EXCEPTIONS AND CONDITIONS:
0192            *
0193            *       SAME AS EVSD
0194            *
0195            * EXTERNAL ROUTINE LIST:
0196            *
0197            *       (SEE EXTERNAL ROUTINES)
0198            *
0199            * LOCAL DATA:
0200            *
0201            *       (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0203                    * ENTRY POINT:
0204                    *
0205                            DEF EVARZ
0206                    *
0207 0030 04C5  EVAR    CLR R5          ;SET FOR VARIABLE ONLY
0208 0032 06A0          BL @EVALS1      ;EVALUATE
     0034 01B8'
0209 0036 0282          CI R2,EVSKB     ;# IN STACK?
     0038 0000
0210 003A 1A12            JL EVER1      ;N, RETURN
0211                    *
0212 003C 2F96  ERR22   DATA ERROR+22   ;EXPECTING VARIABLE
0213 003E 0000' EVARZ   DATA WPR2,EVAR
     0040 0030'
```

```
0216                    * ABSTRACT:
0217                    *
0218                    *       CKEX IS CALLED TO SEE IF AN EXPRESSION
0219                    *       COULD OCCUR NEXT.  THE EXCEPTION WOULD
0220                    *       OCCUR IF A DELIMITER WAS THE NEXT ITEM.
0221                    *
0222                    * CALLING SEQUENCE:
0223                    *
0224                    *       BL @CKEX
0225                    *         NO EXPRESSION
0226                    *       EXPRESSION
0227                    *
0228                    *       IN  R8 = PBC
0229                    *
0230                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0231                    *
0232                    * EXTERNAL ROUTINE LIST:  (NONE)
0233                    *
0234                    * LOCAL DATA:  (NONE)
0235                    *
0236                    * ENTRY POINT:
0237                    *
0238                            DEF CKEX
0239                    *
0240 0042 DA58  CKEX    MOVB *R8,@DLIM(9)
     0044 0000
0241 0046 1307            JEQ CKEX2       ;EOL
0242 0048 9818          CB *R8,@B38       ;FUNCTION?
     004A 053A'
0243 004C 1A03            JL CKEX1        ;Y
0244 004E 9818          CB *R8,@B4C       ;OPERATOR OR VARIABLE?
     0050 053E'
0245 0052 1A01            JL CKEX2        ;N, TAKE ERROR RETURN
0246                    *
0247 0054 05CB  CKEX1   INCT R11
0248 0056 045B  CKEX2   RT
```

```
0251                    * ABSTRACT:
0252                    *
0253                    *       EVER WILL EVALUATE A FULL EXPRESSION AND
0254                    *       WILL EXIT THRU THE ERROR CALL
0255                    *       WITH ANY EXCEPTION.
0256                    *
0257                    * CALL SEQUENCE:
0258                    *
0259                    *       BLWP @EVERZ
0260                    *
0261                    *       IN  R8 = PBC
0262                    *           R9 = PTRS
0263                    *       OUT R0 = DLIM
0264                    *           R2 = ADR
0265                    *           R8 = PBC UPDATED
0266                    *
0267                    * EXCEPTIONS AND CONDITIONS:
0268                    *
0269                    *       SAME AS EVSD
0270                    *
0271                    * EXTERNAL ROUTINE LIST:
0272                    *
0273                    *       (SEE EXTERNAL ROUTINES:)
0274                    *
0275                    * LOCAL DATA:
0276                    *
0277                    *       (SEE MODULE VARIABLES AND CONSTANTS)
0278                    *
0279                    * ENTRY POINT:
0280                    *
0281                            DEF EVERZ
0282                    *
0283 0058 003E' EVERZ   DATA WPR2,EVER
     005A 005C'
0284 005C 06A0  EVER    BL @EVALS       ;EVALUATE
     005E 01B6'
0285 0060 CB42  EVER1   MOV R2,@4(13)   ;RETURN PARAMETERS
     0062 0004
0286 0064 CA40  EVER2   MOV R0,@DLIM(9) ;RETURN DELIMITER
     0066 0044'
0287 0068 C740          MOV R0,*R13
0288 006A CB48          MOV R8,@16(13)  ;UPDATE PBC
     006C 0010
0289 006E 0380          RTWP
```

```
0292                * ABSTRACT:
0293                *
0294                *       EVFX USES AN XOP CALL SINCE IT IS
0295                *       USED SO OFTEN AND RETURNS A 1 WORD,
0296                *       2'S COMPLEMENT INTEGER IN THE OPERAND
0297                *       OF THE EXOP.  EVFX DOES A FULL
0298                *       EVALUATION (R5 <> 0) AND RETURNS
0299                *       R0, R8, AND THE RESULT.
0300                *
0301                *       SINCE R0 IS ALTERED, EVFX R0 WOULD
0302                *       BE ILLEGAL.
0303                *
0304                * CALLING SEQUENCE:
0305                *
0306                *       XOP XX,11
0307                *
0308                *       IN  R8 = PBC
0309                *           R9 = PTRS
0310                *       OUT R0 = DLIM
0311                *           R8 = PBC UPDATED
0312                *         *R11 = #
0313                *
0314                * EXCEPTIONS AND CONDITIONS:
0315                *
0316                *       SAME AS EVSD WITH THE ADDITION OF
0317                *       A FIX ERROR POSSIBILITY.
0318                *
0319                *       R0,R8 CANNOT BE USED AS THE RETURNING
0320                *       FIELD.
0321                *
0322                * EXTERNAL ROUTINE LIST:
0323                *
0324                *       (SEE EXTERNAL ROUTINES)
0325                *
0326                * LOCAL DATA:
0327                *
0328                *       (SEE MODULE VARIABLES AND CONSTANTS)
0329                *
0330                * ENTRY POINT:
0331                *
0332                *       DXOP EVFIX,11
0333                        DEF EVFX          ;*** NOT USED ***
0334                *
0335 0070 C30B     EVFX     MOV R11,R12       ;SAVE ADDRESS
0336 0072 06A0              BL @EVALS         ;EVALUATE
     0074 01B6'
0337 0076 06A0              BL @FIX           ;FIX RESULT
     0078 0000
0338 007A C701              MOV R1,*R12       ;RETURN #
0339 007C 10F3              JMP EVER2
```

```
0342            * ABSTRACT:
0343            *
0344            *       EVAL IS A RECURSIVE STACK EXPRESSION EVALUATOR.
0345            *       R6 POINTS TO THE BOTTOM OF THE STACK WHERE
0346            *       OPERANDS AND OTHER DATA ARE STACKED
0347            *       WHILE R7 POINTS TO THE TOP OF THE STACK WHERE
0348            *       OPERATORS ARE STACKED.  BOTH ENDS OF THE STACK
0349            *       ARE MARKED WITH NULLS TO CHECK FOR PROPER
0350            *       EXPRESSION TERMINATION.  THE ROUTINE CONSISTS
0351            *       MAINLY OF A MULTIPLEXOR AND A PRECEDENCE ORIENTED
0352            *       PROCESSOR.
0353            *
0354            *       VARIABLES ARE STORED ON A STACK AS AN ADDRESS,
0355            *       CONSTANTS GO DIRECTLY ON THE STACK FOLLOWED BY
0356            *       A -1 TO INDICATE A CONSTANT VALUE.  IE:
0357            *
0358            *               * (PTR) * ----- /XXXX XXXX XXXX/
0359            *               * 4110  *
0360            *               * 0000  *
0361            *               * 0000  *
0362            *               *  -1   *
0363            *
0364            *       USER FUNCTIONS, SYSTEM FUNCTIONS, AND DIMENSIONED
0365            *       VARIABLES ALL RECURSE ON THE EVALUATOR TO
0366            *       OBTAIN THE ARGUMENTS.
0367            *
0368            *       THE LEFT BYTE OF R5 CONTROLS WHETHER A OPERATOR
0369            *       IS UNARY OR NOT.  WHEN NULL AND AN OPERATOR IS
0370            *       ENCOUNTERED, IT MUST BE UNARY!  UPON ENCOUNTERING
0371            *       A VARIABLE, THIS LEFT BYTE IS SET INDICATING
0372            *       THE NEXT OPERATOR WILL NOT BE UNARY.
0373            *
0374            *       THE RIGHT BYTE OF R5 CONTROLS HOW FAR THE
0375            *       EVALUATION IS TO GO.  IE, ONE ONLY WANTS THE
0376            *       FIRST VARIABLE FOR AN ASSIGNMENT STATEMENT OR
0377            *       ANYWHERE ONE WANTS TO STORE A VALUE.  THUS,
0378            *       WHEN THE RIGHT BYTE OF R5 IS ZERO, EVALUATION
0379            *       WILL TERMINATE AT THE FIRST OPERATOR.
0380            *
0381            *       WHEN A DELIMITER IS ENCOUNTED, THE RIGHT BYTE
0382            *       OF R5 IS SET TO ZERO THUS TERMINATING THE
0383            *       EVALUATION OF THE EXPRESSION.
0384            *
0385            *       THE LEAST SIGNIFICANT BIT OF THE OPERATOR
0386            *       ADDRESS IS USED TO INDICATE A UNARY OPERATION.
0387            *
0388            *       USER FUNCTIONS USE THE STACK QUITE EXTENSIVELY
0389            *       PUSHING FUNCTION ADDRESS, DUMMY ARGUMENTS, PLC,
0390            *       R5, AND A RETORE-STACK-POINTER (R4) FOR EACH
0391            *       LEVEL OF FUNCTIONS.
0392            *
0393            * CALLING SEQUENCE:
0394            *
0395            *       BL @EVALS       ;START EVALUATION
0396            *
0397            *       IN (R9) = POINTERS
0398            *           R5  = EXPRESSION FLAG
0399            *
0400            *         (FALL THRU TO EVAL)
0401            *
```

```
0402            *       BL @EVAL        ;RECURSIVE EVALUATOR
0403            *
0404            *           R5  = EXPRESSION FLAG
0405            *           R6  = BOTTOM OF STACK
0406            *           R7  = TOP OF STACK
0407            *           R8  = PLC
0408            *          (R9) = POINTERS
0409            *
0410            * EXCEPTIONS AND CONDITIONS:
0411            *
0412            *       IF R6 >= R7, THEN AN EXPRESSION TOO
0413            *          COMPLEX OCCURS.
0414            *
0415            * EXTERNAL ROUTINE LIST:
0416            *
0417            *       (SEE EXTERNAL ROUTINES)
0418            *
0419            * LOCAL DATA:
0420            *
0421            *       (SEE MODLULE VARIABLES AND CONSTANTS
```

```
0423                    *USER FUNCTION EVALUATION
0424                    *
0425                    *STACK: R5
0426                    *       FUNC ADR
0427                    *       DUMMY #1
0428                    *       DUMMY #2
0429                    *       DUMMY #3
0430                    *       . . .
0431                    *       R8
0432                    *       R4 = STACK+2
0433                    *
0434 007E 2FA0          ERR38   DATA ERROR2,38  ;UNDEFINED FUNCTION
     0080 0026
0435                    *
0436 0082 CD85          EVAFN   MOV R5,*R6+     ;SAVE R5
0437 0084 C106                  MOV R6,R4       ;MARK
0438 0086 C0E9                  MOV @UFT(9),R3  ;GET TABLE ADR
     0088 0000
0439 008A 0960                  SRL R0,6        ;GET INDEX
0440 008C A0C0                  A R0,R3         ;INDEX
0441 008E CDB3                  MOV *R3+,*R6+   ;STACK ADR
0442 0090 13F6                    JEQ ERR38     ;NOT DEFINED, ERROR
0443 0092 C053                  MOV *R3,R1      ;GET COUNT
0444 0094 132C                    JEQ EVAFN5    ;NO ARGUMENTS
0445 0096 C0E9                  MOV @VDT(9),R3
     0098 0000
0446 009A CDB3                  MOV *R3+,*R6+   ;SAVE OLD DUMMIES
0447 009C CDB3                  MOV *R3+,*R6+
0448 009E CD93                  MOV *R3,*R6+
0449 00A0 0223                  AI R3,-4
     00A2 FFFC
0450 00A4 0705                  SETO R5         ;DO FULL EVAL IF NOT (
0451 00A6 9838                  CB *R8+,@B4A    ;(?
     00A8 053D'
0452 00AA 1303                    JEQ EVAFN1    ;Y
0453 00AC 0608                  DEC R8          ;N
0454 00AE 04C5                  CLR R5
0455 00B0 0701                  SETO R1         ;SET COUNT TO -1
0456                    *
0457 00B2 CD81          EVAFN1  MOV R1,*R6+     ;PUSH COUNT
0458 00B4 CD83                  MOV R3,*R6+
0459 00B6 CD84                  MOV R4,*R6+
0460 00B8 06A0                  BL @EVAL        ;EVALUATE NEXT PARAMETER
     00BA 01C8'
0461 00BC 0646                  DECT R6
0462 00BE C116                  MOV *R6,R4      ;POP R4
0463 00C0 0646                  DECT R6
0464 00C2 C0D6                  MOV *R6,R3      ;POP SYMBOL TABLE PTR
0465 00C4 0646                  DECT R6
0466 00C6 C056                  MOV *R6,R1      ;POP COUNT
0467 00C8 CCC6                  MOV R6,*R3+     ;LOAD NEW DUMMY ADR
0468 00CA CDB2                  MOV *R2+,*R6+   ;STACK
0469 00CC CDB2                  MOV *R2+,*R6+
0470 00CE CD92                  MOV *R2,*R6+
0471 00D0 0581                  INC R1          ;LOOKING FOR VARIABLE ONLY?
0472 00D2 1308                    JEQ EVAFN2    ;Y
0473 00D4 0641                  DECT R1         ;N, DONE?
0474 00D6 1308                    JEQ EVAFN4    ;Y
0475 00D8 0705                  SETO R5         ;N, DO FULL
0476 00DA 0280                  CI R0,>3F00     ;,?
```

```
      00DC 3F00
0477  00DE 13E9              JEQ EVAFN1      ;Y
0478  00E0 2FA0  EVER37      DATA ERROR2,37  ;ILLEGAL DELIMITER
      00E2 0025
0479               *
0480  00E4 0608  EVAFN2      DEC R8          ;BACKUP OVER DELIMITER
0481  00E6 1003              JMP EVAFN5
0482               *
0483  00E8 0280  EVAFN4      CI R0,>4B00     ;)?
      00EA 4B00
0484  00EC 16F9              JNE EVER37      ;N, ERROR
0485               *
0486  00EE CD84  EVAFN5      MOV R4,*R6+     ;PUSH R4
0487  00F0 CD88              MOV R8,*R6+     ;PUSH R8
0488  00F2 C214              MOV *R4,R8      ;GET DEF ADR
0489  00F4 0705              SETO R5
0490  00F6 06A0              BL @EVAL        ;EVALUATE
      00F8 01C8'
0491  00FA 0646              DECT R6
0492  00FC C216              MOV *R6,R8      ;POP R8
0493  00FE 0646              DECT R6
0494  0100 C196              MOV *R6,R6      ;POP R6 (R4)
0495  0102 C106              MOV R6,R4
0496  0104 05C4              INCT R4         ;MOVE TO DUMMY STORAGE
0497  0106 C0E9              MOV @VDT(9),R3
      0108 0098'
0498  010A CCF4              MOV *R4+,*R3+   ;RESTORE OLD VARIABLE ADDRESSES
0499  010C CCF4              MOV *R4+,*R3+
0500  010E C4D4              MOV *R4,*R3
0501  0110 1047              JMP EVSFR       ;RETURN
```

```
0503                    *DO SYSTEM STRING OPERAND
0504                    *
0505 0112 9818  EVSF0   CB *R8,@B43     ;$?
     0114 053B'
0506 0116 130B            JEQ EVSF02    ;Y
0507 0118 1A09            JL EVSF01
0508 011A 9818          CB *R8,@B45     ;" OR '?
     011C 053C'
0509 011E 1B06            JH EVSF01     ;N
0510 0120 0588          INC R8          ;Y
0511 0122 C088          MOV R8,R2       ;SAVE ADR
0512                    *
0513 0124 D038          MOVB *R8+,R0    ;MOVE TO END OF STRING
0514 0126 16FE            JNE $-2
0515 0128 D038          MOVB *R8+,R0    ;GET DELIMITER
0516 012A 05CB          INCT R11
0517                    *
0518 012C 045B  EVSF01  RT
0519                    *
0520 012E 0588  EVSF02  INC R8          ;MOVE TO VARIABLE
0521 0130 05CB          INCT R11        ;SET RETURN
0522 0132 04C5          CLR R5
0523 0134 0460          B @EVAL         ;DO EVALUATION
     0136 01C8'
```

```
0525                    *SYSTEM FUNCTION EVALUATION
0526                    *
0527 0138 CD85  EVSF    MOV R5,*R6+       ;STACK R5,R0
0528 013A CD80          MOV R0,*R6+
0529 013C 0201          LI R1,1           ;DEFAULT SECOND PARAMETER TO 1
     013E 0001
0530 0140 9838          CB *R8+,@B4A      ;[?
     0142 053D'
0531 0144 1306            JEQ EVSF1       ;Y
0532 0146 04C5          CLR R5            ;N, DO VARIABLE ONLY
0533 0148 0608          DEC R8
0534 014A 06A0          BL @EVAL
     014C 01C8'
0535 014E 0608          DEC R8
0536 0150 1020          JMP EVSF2
0537                    *
0538 0152 06A0  EVSF1   BL @EVSF0         ;CHECK FOR STRING
     0154 0112'
0539 0156 1009            JMP EVSF1A      ;N
0540 0158 CD82          MOV R2,*R6+       ;STACK ADR
0541 015A 0280          CI R0,>3F00       ;,?
     015C 3F00
0542 015E 1614            JNE EVSF1B      ;N
0543 0160 06A0          BL @EVSF0         ;GET SECOND STRING
     0162 0112'
0544 0164 10BD            JMP EVER37
0545 0166 C042          MOV R2,R1
0546 0168 100F          JMP EVSF1B
0547                    *
0548 016A 0705  EVSF1A  SETO R5           ;DO FULL EVALUATION
0549 016C 06A0          BL @EVAL
     016E 01C8'
0550 0170 0280          CI R0,>4B00       ;]?
     0172 4B00
0551 0174 130E            JEQ EVSF2       ;Y
0552 0176 0280          CI R0,>3F00       ;,?
     0178 3F00
0553 017A 16B2            JNE EVER37      ;N, ERROR
0554 017C CD82          MOV R2,*R6+       ;STACK ADR
0555 017E 0705          SETO R5
0556 0180 06A0          BL @EVAL          ;GET SECOND PARAMETER
     0182 01C8'
0557 0184 06A0          BL @FIX
     0186 0078'
0558                    *
0559 0188 0646  EVSF1B  DECT R6           ;POP 1ST OPERAND
0560 018A C096          MOV *R6,R2
0561 018C 0280          CI R0,>4B00       ;]?
     018E 4B00
0562 0190 16A7            JNE EVER37      ;N, ERROR
0563                    *
0564 0192 0646  EVSF2   DECT R6           ;POP ID
0565 0194 C0D6          MOV *R6,R3
0566 0196 0973          SRL R3,7          ;GET INDEX
0567 0198 C2E3          MOV @SFUNP->36(3),R11
     019A 03C2'
0568 019C 2E00          CLEAR 0           ;CLEAR FPAC
0569 019E 069B          BL *R11           ;GOTO ROUTINE
0570                    *
0571 01A0 0646  EVSFR   DECT R6           ;POP R5
```

```
0572 01A2 C156          MOV *R6,R5
0573                    *
0574 01A4 CDB2  EVSF4   MOV *R2+,*R6+   ;MOVE RESULT ON STACK
0575 01A6 CDB2          MOV *R2+,*R6+
0576 01A8 CD92          MOV *R2,*R6+
0577 01AA 1066          JMP EVAL6       ;CONTINUE
0578                    *
0579                    *GET RANDOM NUMBER
0580                    *
0581 01AC 0420  EVRND   BLWP @RANDZ     ;GET RANDOM # IN FPAC
     01AE 0000
0582 01B0 0202          LI R2,@FPAC
     01B2 0000
0583 01B4 10F7          JMP EVSF4
```

```
0585                    *ENTRY POINT:
0586                    *
0587                            DEF EVAL                ;FULL EXPRESSION EVALUATION
0588                            DEF EVALS2      ;PARTIAL EXPRESSION EVALUATION
0589                    *
0590 01B6 0705  EVALS   SETO R5         ;FULL EXPRESSION
0591                    *
0592 01B8 C22D  EVALS1  MOV @16(13),R8  ;GET PBC
     01BA 0010
0593 01BC C26D          MOV @18(13),R9  ;GET POINTERS
     01BE 0012
0594                    *
0595 01C0 0206  EVALS2  LI R6,EVSKB     ;GET BEGINNING STACK PTR
     01C2 0038'
0596 01C4 0207          LI R7,EVSKE     ;GET END STACK PTR
     01C6 0000
0597                    *
0598                    *EVALUATE SIMPLE EXPRESSION
0599                    *
0600 01C8 CD8B  EVAL    MOV R11,*R6+    ;STACK RETURN
0601 01CA 04F6          CLR *R6+        ;MARK OPERAND STACK
0602 01CC 04C0          CLR R0          ;MARK OPERATOR STACK
0603 01CE D140          MOVB R0,R5      ;IF OPERATOR: MUST BE UNARY
0604                    *
0605 01D0 0607  EVAL1   DEC R7          ;PUSH ON OPERATOR STACK
0606 01D2 D5C0          MOVB R0,*R7
0607 01D4 81C6          C R6,R7         ;STACK OVERFLOW?
0608 01D6 1457            JHE EVER27    ;Y, ERROR
0609                    *
0610 01D8 D038  EVAL2   MOVB *R8+,R0    ;PARSE
0611 01DA 1376            JEQ EVADL     ;DONE
0612 01DC 0280          CI R0,>6F00     ;VARIABLE?
     01DE 6F00
0613 01E0 1B22            JH EVAV       ;Y
0614 01E2 130F            JEQ EVFP      ;N, FP #
0615 01E4 0280          CI R0,>6200     ;CONSTANT?
     01E6 6200
0616 01E8 1413            JHE EVAL3     ;Y
0617 01EA 0280          CI R0,>4C00     ;"( OR OPERATER?
     01EC 4C00
0618 01EE 1B64            JH EVOP       ;OPERATER
0619 01F0 13EF            JEQ EVAL1     ;"(, PUSH
0620 01F2 0280          CI R0,>3800     ;DELIMITER?
     01F4 3800
0621 01F6 1468            JHE EVADL     ;Y
0622 01F8 0280          CI R0,>1B00     ;SYSTEM FUNCTION?
     01FA 1B00
0623 01FC 149D            JHE EVSF      ;Y
0624 01FE 0460          B @EVAFN        ;FUNCTION
     0200 0082'
0625                    *
0626 0202 DDB8  EVFP    MOVB *R8+,*R6+  ;FP CONSTANT
0627 0204 DDB8          MOVB *R8+,*R6+
0628 0206 DDB8          MOVB *R8+,*R6+
0629 0208 DDB8          MOVB *R8+,*R6+
0630 020A DDB8          MOVB *R8+,*R6+
0631 020C DDB8          MOVB *R8+,*R6+
0632 020E 1034          JMP EVAL6       ;MARK
0633                    *
0634 0210 0280  EVAL3   CI R0,>6D00      ;1 WORD INTEGER?
```

```
      0212 6D00
0635  0214 142A            JHE EVAL4      ;Y
0636  0216 06C0           SWPB R0
0637  0218 0220           AI R0,->63      ;N, -1 THRU 9
      021A FF9D
0638  021C 102A           JMP EVAL5       ;STORE #
0639                 *
0640  021E 2F8A  EVER10    DATA ERROR+10   ;STORAGE OVERFLOW
```

```
0642                    *UNARY MINUS
0643                    *
0644 0220 0200  EVAL9   LI R0,>6100      ;STACK -
     0222 6100
0645 0224 10D5  EVAL91  JMP EVAL1
0646                    *
0647                    *VARIABLE
0648                    *
0649 0226 0280  EVAV    CI R0,>7300      ;RND?
     0228 7300
0650 022A 13C0            JEQ EVRND      ;Y
0651 022C C069          MOV @VNT(9),R1   ;GET VARIABLE TABLE
     022E 0000
0652 0230 0970          SRL R0,7         ;SWAP & X 2
0653 0232 A040          A R0,R1          ;INDEX
0654 0234 C061          MOV @>70*2(1),R1        ;DIMENSIONED
     0236 FF20
0655 0238 117F            JLT EVADV      ;Y
0656 023A C069          MOV @VDT(9),R1
     023C 0108'
0657 023E A040          A R0,R1
0658 0240 C021          MOV @>70*2(1),R0        ;DEFINED?
     0242 FF20
0659 0244 161A            JNE EVAL7      ;Y
0660 0246 06C5          SWPB R5          ;POSITION EVALUATION TYPE LJ IN R5
0661 0248 D145          MOVB R5,R5       ;FULL OR VARIABLE EVALUATION ?
0662 024A 06C5          SWPB R5          ;RESTORE EVALUATION TYPE TO ORIGINAL
0663 024C 160C            JNE EVER40     ;NE - FULL EVAL - DO NOT DEFINE - ER
0664 024E C029          MOV @NVS(9),R0   ;VARIABLE EVALUATION ONLY - DEFINE I
     0250 0000
0665 0252 0220          AI R0,-6
     0254 FFFA
0666 0256 8A40          C R0,@NVD(9)     ;OK?
     0258 0000
0667 025A 1AE1            JL EVER10      ;N, STORAGE OVERFLOW
0668 025C C840          MOV R0,@>70*2(1)
     025E FF20
0669 0260 CA40          MOV R0,@NVS(9)   ;UPDATA NVS
     0262 0250'
0670 0264 100A          JMP EVAL7
0671      0266' EVER40  EQU $
0672 0266 2FA0          DATA ERROR2,40   ;UNDEFINED VARIABLE
     0268 0028
0673                    *
0674                    *CONSTANTS
0675                    *
0676 026A D038  EVAL4   MOVB *R8+,R0     ;1 WRD CONSTANT
0677 026C 06C0          SWPB R0
0678 026E D038          MOVB *R8+,R0
0679 0270 06C0          SWPB R0
0680                    *
0681 0272 04F6  EVAL5   CLR *R6+         ;MOVE INTO STACK
0682 0274 CD80          MOV R0,*R6+      ;INSERT 0,#,0
0683 0276 04F6          CLR *R6+
0684                    *
0685 0278 0700  EVAL6   SETO R0          ;MARK AS CONSTANT
0686                    *
0687 027A CD80  EVAL7   MOV R0,*R6+      ;MARK STACK
0688 027C 04C0          CLR R0
0689                    *
```

```
0690 027E D160  EVAL8   MOVB @BFF,R5     ;IF OPERATOR: MUST NOT BE UNARY
     0280 053F'
0691 0282 81C6          C R6,R7          ;STACK OVERFLOW?
0692 0284 1AA9            JL EVAL2       ;N, OK
0693                 *
0694 0286 2F9B  EVER27  DATA ERROR+27    ;Y, EXPRESSION TOO COMPLEX
```

```
0696                    *OPERATOR
0697                    *
0698                    *HANDLE UNARY OPERATORS
0699                    *
0700 0288 0280  EVOPA   CI R0,>5D00      ;"+?
     028A 5D00
0701 028C 13A5            JEQ EVAL2     ;Y, IGNOR
0702 028E 0280          CI R0,>5C00     ;N, "-?
     0290 5C00
0703 0292 13C6            JEQ EVAL9     ;Y, STACK UNARY -
0704 0294 0280          CI R0,>5200     ;NOT?
     0296 5200
0705 0298 139B            JEQ EVAL1     ;Y, STACK
0706 029A 0280          CI R0,>5300     ;LNOT?
     029C 5300
0707 029E 1398            JEQ EVAL1     ;Y, STACK
0708 02A0 2F81  EVER0   DATA ERROR+1    ;N, SYNTAX ERROR
0709                    *
0710                    *HANDLE OPENING AND CLOSING PAREN'S
0711                    *
0712 02A2 0280  EVOP0   CI R0,>4D00      ;")?
     02A4 4D00
0713 02A6 1612            JNE EVOP1     ;N
0714 02A8 9817  EVOP0A  CB *R7,@B4C     ;(?
     02AA 053E'
0715 02AC 160F            JNE EVOP1     ;N
0716 02AE 0587          INC R7          ;Y, POP
0717 02B0 1093          JMP EVAL2
0718                    *
0719 02B2 D057  EVOP0B  MOVB *R7,R1     ;NULL?
0720 02B4 16F9            JNE EVOP0A    ;N
0721 02B6 1008          JMP EVADL       ;Y, TERMINATE
0722                    *
0723                    *OPERATER ENTRY
0724                    *
0725 02B8 D045  EVOP    MOVB R5,R1      ;NEED UNARY OPERATER?
0726 02BA 13E6            JEQ EVOPA     ;Y
0727 02BC 0280          CI R0,>4D00     ;)?
     02BE 4D00
0728 02C0 13F8            JEQ EVOP0B    ;Y
0729 02C2 0245          ANDI R5,>FF     ;N, ALLOW UNARY ONLY TO FOLLOW
     02C4 00FF
0730 02C6 1602            JNE EVOP1     ;FULL EXPRESSION
0731                    *
0732                    *DELIMITER ENTRY
0733                    *
0734 02C8 04C5  EVADL   CLR R5          ;SET TO TERMINATE
0735 02CA 04C0          CLR R0          ;DO CR
0736                    *
0737 02CC D057  EVOP1   MOVB *R7,R1     ;LOOK AT TOP OPERATOR
0738 02CE C100          MOV R0,R4
0739 02D0 0244          ANDI R4,>FE00   ;LAP OFF LOW BIT
     02D2 FE00
0740 02D4 9044          CB R4,R1        ;HIGHER PRECEDENCE?
0741 02D6 18A6            JH EVAL91     ;Y, STACK
0742 02D8 D0F7          MOVB *R7+,R3    ;N, POP OPERATER
0743 02DA 131C            JEQ EVOP4     ;DONE
0744 02DC 0973          SRL R3,7        ;SWAP & X 2
0745 02DE C2E3          MOV @EVATB->9C(3),R11
     02E0 0326'
```

```
0746                    *
0747 02E2 0646          DECT R6             ;POP VARIABLE
0748 02E4 C056          MOV *R6,R1          ;EMPTY?
0749 02E6 13DC            JEQ EVER0         ;Y
0750 02E8 0596          INC *R6             ;CONSTANT?
0751 02EA 1603            JNE EVOP2         ;N
0752 02EC 0226          AI R6,-6            ;Y
     02EE FFFA
0753 02F0 C046          MOV R6,R1
0754                    *
0755 02F2 22E0  EVOP2   COC @C1,R11         ;UNARY OPERATION?
     02F4 03EC'
0756 02F6 1308            JEQ EVOP3         ;Y
0757 02F8 0646          DECT R6             ;N, GET SECOND OPERAND
0758 02FA C096          MOV *R6,R2          ;EMPTY?
0759 02FC 13D1            JEQ EVER0         ;Y
0760 02FE 0596          INC *R6             ;CONSTANT?
0761 0300 1603            JNE EVOP3         ;N
0762 0302 0226          AI R6,-6            ;Y
     0304 FFFA
0763 0306 C086          MOV R6,R2
0764                    *
0765 0308 069B  EVOP3   BL *R11             ;DO OPERATION
0766                    *
0767 030A CDB2  EVOP3A  MOV *R2+,*R6+       ;MOVE RESULTS ON STACK
0768 030C CDB2          MOV *R2+,*R6+
0769 030E CD92          MOV *R2,*R6+
0770 0310 0736          SETO *R6+           ;MARK AS CONSTANT
0771 0312 10C7          JMP EVOP0           ;CONTINUE
0772                    *
0773 0314 D028  EVOP4   MOVB @-1(8),R0      ;GET DELIMITER
     0316 FFFF
0774 0318 0646          DECT R6             ;POP VARIABLE
0775 031A C096          MOV *R6,R2          ;EMPTY?
0776 031C 13C1            JEQ EVER0         ;Y, ERROR
0777 031E 0596          INC *R6             ;CONSTANT?
0778 0320 1603            JNE EVOP5         ;N
0779 0322 0226          AI R6,-6            ;Y
     0324 FFFA
0780 0326 C086          MOV R6,R2
0781                    *
0782 0328 0226  EVOP5   AI R6,-4            ;DONE, POP STACK
     032A FFFC
0783 032C C2D6          MOV *R6,R11
0784 032E C066          MOV @2(6),R1        ;STACK EMPTIED?
     0330 0002
0785 0332 16B6            JNE EVER0         ;N, ERROR
0786 0334 045B          RT
```

```
0788 0336 10A3  EVAL8P  JMP EVAL8
0789            *
0790            *DIMENSIONED VARIABLE
0791            *
0792 0338 C069  EVADV   MOV @VDT(9),R1
     033A 023C'
0793 033C A040          A R0,R1
0794 033E C121          MOV @->70*2(1),R4       ;DEFINED?
     0340 FF20
0795 0342 133B            JEQ ERR39     ;N, UNDIMENSIONED VARIABLE
0796 0344 CD85          MOV R5,*R6+     ;STACK R5
0797 0346 04F6          CLR *R6+        ;ZERO INDEX & STACK
0798            *
0799 0348 CD84  EVADV1  MOV R4,*R6+     ;STACK R4
0800 034A 0705          SETO R5         ;DO FULL EXPRESSION
0801 034C 06A0          BL @EVAL        ;RECURSE
     034E 01C8'
0802 0350 06A0          BL @FIX         ;FIX #
     0352 0186'
0803 0354 0646          DECT R6         ;POP BASE ADR
0804 0356 C116          MOV *R6,R4
0805 0358 0646          DECT R6         ;READY FOR INDEX POP
0806 035A 8D01          C R1,*R4+       ;EXCEED DIMENSION?
0807 035C 182C            JH ERR17      ;Y, SUBSCRIPT ERROR
0808 035E C174          MOV *R4+,R5     ;GET DEL MULTIPLIER
0809 0360 0285          CI R5,-1        ;DONE?
     0362 FFFF
0810 0364 1306            JEQ EVADV2    ;Y
0811 0366 3845          MPY R5,R1       ;MULTIPLY
0812 0368 AD82          A R2,*R6+       ;ADD TO INDEX & STACK
0813 036A 0280          CI R0,>3F00     ;,?
     036C 3F00
0814 036E 13EC            JEQ EVADV1    ;Y, DO AGAIN
0815            *
0816 0370 2F92  ERR18   DATA ERROR+18   ;N, TOO FEW SUBSCRIPTS
0817            *
0818 0372 A056  EVADV2  A *R6,R1        ;ADD FINAL DIMENSION
0819 0374 3860          MPY @C6,R1      ;MULTIPLY BY 6
     0376 03F6'
0820 0378 A084          A R4,R2         ;ADD BASE
0821 037A 0646          DECT R6         ;POP R5
0822 037C C156          MOV *R6,R5
0823 037E CD82          MOV R2,*R6+     ;STACK ADDRESS
0824 0380 0280          CI R0,>4B00     ;]?
     0382 4B00
0825 0384 13D8            JEQ EVAL8P    ;Y
0826 0386 0280          CI R0,>4000     ;;?
     0388 4000
0827 038A 1616            JNE ERR19     ;N, TOO MANY SUBSCRIPTS
0828 038C CD85          MOV R5,*R6+     ;Y, STACK R5
0829 038E 0705          SETO R5         ;DO FULL EVAL
0830 0390 06A0          BL @EVAL        ;RECURSE
     0392 01C8'
0831 0394 06A0          BL @FIX         ;GET INTEGER
     0396 0352'
0832 0398 0280          CI R0,>4B00     ;]?
     039A 4B00
0833 039C 1610            JNE EVADVE    ;N, ERROR
0834 039E 0646          DECT R6         ;Y, POP R5
0835 03A0 C156          MOV *R6,R5
```

```
0836 03A2 0646          DECT R6          ;POP ADR
0837 03A4 C016          MOV *R6,R0
0838 03A6 0601          DEC R1           ;INDEX=COUNT-1
0839 03A8 1106            JLT ERR17      ;ERROR
0840 03AA A001          A R1,R0          ;INDEX
0841 03AC 8A40          C R0,@IOB(9)     ;EXCEED MEMORY?
     03AE 0000
0842 03B0 1402            JHE ERR17      ;Y, ERROR
0843 03B2 0460          B @EVAL7         ;N, CONTINUE
     03B4 027A'
0844                    *
0845 03B6 2F91  ERR17   DATA ERROR+17    ;SUBSCRIPT ERROR
0846                    *
0847 03B8 2F93  ERR19   DATA ERROR+19    ;TOO MANY SUBSCRIPTS
0848                    *
0849 03BA 2FA0  ERR39   DATA ERROR2,39   ;UNDIMENSIONED VARIABLE
     03BC 0027
0850                    *
0851 03BE 2FA0  EVADVE  DATA ERROR2,37   ;INVALID DELIMITER
     03C0 0025
0852                    *
0853 03C2 0000  EVATB   DATA ORF,LORF
     03C4 0000
0854 03C6 0000          DATA ANDF,LANDF
     03C8 0000
0855 03CA 0000          DATA NOTF,LNOTF
     03CC 0000
0856 03CE 0000          DATA LXORF
0857 03D0 04E8'         DATA FUZF
0858 03D2 04E8'         DATA EQUF,GTHF
     03D4 04E8'
0859 03D6 04E8'         DATA GEQF,LTHF
     03D8 04E8'
0860 03DA 04E8'         DATA LEQF,NEQUF
     03DC 04E8'
0861 03DE 046A'         DATA SUBF,ADDF
     03E0 0442'
0862 03E2 04AC'         DATA DIVF,MULF
     03E4 0478'
0863 03E6 0000          DATA POWF,UMNF+1
     03E8 0525'
0864                    *
0865      0028  EVPR    EQU $-EVATB
0866 03EA 0000          DATA 0
0867      03ED' B01     EQU $+1
0868 03EC 0001  C1      DATA 1
0869 03EE 0004  C4      DATA 4
0870      03F1' B05     EQU $+1
0871 03F0 0005          DATA 5
0872 03F2 0002          DATA 2
0873 03F4 0003          DATA 3
0874 03F6 0006  C6      DATA 6
0875                    *
0876 03F8 0000  SFUNP   DATA ABSF
0877 03FA 0000          DATA ASCF
0878 03FC 0000          DATA ATNF
0879 03FE 0000          DATA COSF
0880 0400 0000          DATA EXPF
0881 0402 0000          DATA INPF
0882 0404 0000          DATA LOGF
```

```
0883 0406 0000          DATA NKYF
0884 0408 0000          DATA SINF
0885 040A 0000          DATA SQRF
0886 040C 0000          DATA SYSF
0887 040E 0000          DATA TICF
0888 0410 0000          DATA BITF
0889 0412 0000          DATA CRBF
0890 0414 0000          DATA CRFF
0891 0416 0000          DATA MEMF
0892 0418 0000          DATA MWDF
0893 041A 0000          DATA LENF
0894 041C 0000          DATA MCHF
0895 041E 0000          DATA SRHF
```

```
0898            * ABSTRACT:
0899            *
0900            *       ARITHMETIC OPERATIONS SUCH AS ADD, SUBTRACT,
0901            *       MULTIPLY, DIVIDE, UNARY MINUS, AND RELATIONAL
0902            *       OPERATORS SUCH AS FUZZ, EQUAL, NOT EQUAL,
0903            *       LESS THAN, GREATER THAN, LESS THAN OR EQUAL,
0904            *       GREAT THAN OR EQUAL, AND NOT EQUAL USE
0905            *       OPERANDS POINTED TO BY R1,R2 AND RETURN A
0906            *       RESULT POINTED TO BY R2.
0907            *
0908            *       PREP DETERMINES IF BOTH OPERANDS ARE INTEGER
0909            *       VALUES (IN WHICH CASE AN ATTEMPT IS MADE TO
0910            *       DO AN INTERATION OPERATION) OR IF 1 OR BOTH
0911            *       OF THE OPERANDS IS A FLOATING POINT NUMBER
0912            *       (IN WHICH CASE BOTH ARE FLOATED AND A FLOATING
0913            *       POINT OPERATION IS PERFORMED.)
0914            *
0915            *       RELATIONAL OPERATORS ARE GIVEN A VALUE CORRE-
0916            *       SPONDING TO THE COMPARISON TO BE MADE.  BIT 0
0917            *       IS THE EQUAL BIT, BIT 1 IS THE LESS-THAN BIT,
0918            *       AND BIT 2 IS THE GREATER-THAN BIT.  HENCE:
0919            *
0920            *         0000  =  FUZZ
0921            *         0001  =  EQUAL
0922            *         0002  =  LESS-THAN
0923            *         0003  =  LESS-THAN OR EQUAL
0924            *         0004  =  GREATER-THAN
0925            *         0005  =  GREATER-THAN OR EQUAL
0926            *         0006  =  NOT EQUAL (LESS-THAN, GREATER-THAN)
0927            *
0928            * CALLING SEQUENCE:
0929            *
0930            *       BL @ADDF
0931            *       BL @SUBF
0932            *       BL @MULF
0933            *       BL @DIVF
0934            *
0935            *       IN (R1) = OPERAND 1
0936            *          (R2) = OPERAND 2
0937            *
0938            *       OUT (R2) = RESULT
0939            *
0940            *
0941            *       B @FUZF          ;==
0942            *       B @EQUF          ;=
0943            *       B @LTHF          ;<
0944            *       B @LEQF          ;<=
0945            *       B @GTHF          ;>
0946            *       B @GEQF          ;>=
0947            *       B @NEQUF         ;<>
0948            *
0949            *       IN (R1) = OPERAND 1
0950            *          (R2) = OPERAND 2
0951            *
0952            *       OUT (R2) = RESULT
0953            *       EXITS TO EVOP3A
0954            *
0955            *       B @UMNF          ;- (UNARY MINUS)
0956            *
0957            *       IN (R1) = OPERAND
```

```
0958            *
0959            *       OUT (R2) = RESULT
0960            *       EXISTS TO EVOP3A
0961            *
0962            * EXCEPTIONS AND CONDITIONS:
0963            *
0964            * EXTERNAL ROUTINE LIST:
0965            *
0966            *       (SEE EXTERNAL ROUTINES)
0967            *
0968            * LOCAL DATA:
0969            *
0970            *       (SEE MODULE VARIABLES AND CONSTANTS
```

```
0972                *ARITHMETIC PREPARATION
0973                *       BL @PREP
0974                *         VECTOR
0975                *       BOTH INTEGERS
0976                *
0977                *       IN      R1 = SOURCE 1
0978                *               R2 = SOURCE 2
0979                *
0980                *       INT OUT R3 = S1
0981                *               R1 = S2
0982                *
0983 0420 C811     PREP     MOV *R1,@TEMP   ;LOAD TEMP
     0422 0000
0984 0424 C821              MOV @2(1),@TEMP2
     0426 0002
     0428 0000
0985 042A C821              MOV @4(1),@TEMP4
     042C 0004
     042E 0000
0986 0430 2C12              LOADF *R2       ;LOAD FPAC
0987 0432 C0F1              MOV *R1+,R3     ;INTEGER?
0988 0434 160C                JNE PREP4     ;N
0989 0436 C0F2              MOV *R2+,R3     ;INTEGER?
0990 0438 160A                JNE PREP4     ;N
0991 043A C0D1              MOV *R1,R3      ;BOTH INTEGERS
0992 043C C052              MOV *R2,R1      ;GET INTEGERS
0993 043E 046B              B @2(11)
     0440 0002
```

```
0995                    * ENTRY POINT:
0996                    *
0997                            DEF ADDF
0998                    *
0999 0442 C28B  ADDF    MOV R11,R10     ;SAVE RETURN
1000 0444 06A0          BL @PREP        ;PREPARE OPEANDS
     0446 0420'
1001 0448 2CA0            DATA 2*>40+>2C20
1002 044A A043          A R3,R1         ;BOTH INTEGERS
1003 044C 1907            JNO PREP5A    ;NO OVERFLOW
1004 044E 0420  PREP4   BLWP @FLTMZ     ;FLOAT TEMP
     0450 0466'
1005 0452 2E80          FLOATF 0        ;FLOAT FPAC
1006 0454 049B          X *R11          ;DO FP OPERATION
1007 0456 0422'           DATA TEMP
1008 0458 1003          JMP PREP5B      ;RETURN
1009                    *
1010 045A 0501  PREP5   NEG R1          ;NEGATE RESULT
1011                    *
1012 045C C801  PREP5A  MOV R1,@FPAC2   ;SAVE RESULT
     045E 0000
1013                    *
1014 0460 0202  PREP5B  LI R2,FPAC      ;GET ADDRESS OF RESULT
     0462 0182'
1015 0464 045A          B *R10          ;RETURN
1016                    *
1017 0466 0456' FLTMZ   DATA TEMP,FLOAT
     0468 0000
1018                    *
1019                    * ENTRY POINT:
1020                    *
1021                            DEF SUBF
1022                    *
1023 046A C28B  SUBF    MOV R11,R10     ;SAVE RETURN
1024 046C 06A0          BL @PREP
     046E 0420'
1025 0470 2CE0            DATA 3*>40+>2C20
1026 0472 6043          S R3,R1         ;INTEGERS
1027 0474 19F3            JNO PREP5A
1028 0476 10EB          JMP PREP4       ;OVERFLOW
```

```
1030                    *MULTIPLY
1031                    *
1032 0478 C28B  MULF    MOV R11,R10     ;SAVE RETURN
1033 047A 06A0          BL @PREP
     047C 0420'
1034 047E 2D20            DATA 4*>40+>2C20
1035 0480 C0C3          MOV R3,R3       ;R3<0?
1036 0482 1108            JLT MULF5     ;Y
1037 0484 C041          MOV R1,R1       ;R1<0?
1038 0486 110B            JLT MULF6     ;Y
1039                    *
1040 0488 3843  MULF2   MPY R3,R1       ;R1,R2 = R3 X R1
1041 048A C041          MOV R1,R1       ;OVERFLOW?
1042 048C 16E0            JNE PREP4     ;Y
1043 048E C042          MOV R2,R1       ;OVERFLOW?
1044 0490 11DE            JLT PREP4     ;Y
1045 0492 10E4          JMP PREP5A      ;N
1046                    *
1047 0494 0503  MULF5   NEG R3          ;R3<0
1048 0496 C041          MOV R1,R1       ;R1<0
1049 0498 1503            JGT MULF7     ;N
1050 049A 0501          NEG R1          ;Y
1051 049C 10F5          JMP MULF2       ;Y
1052                    *
1053 049E 0501  MULF6   NEG R1          ;SIGNS DIFFERENT
1054 04A0 3843  MULF7   MPY R3,R1       ;R1,R2 = R3 X R1
1055 04A2 C041          MOV R1,R1       ;OVERFLOW?
1056 04A4 16D4            JNE PREP4     ;Y
1057 04A6 C042          MOV R2,R1       ;OVERFLOW?
1058 04A8 11D2            JLT PREP4     ;Y
1059 04AA 10D7          JMP PREP5       ;RETURN
```

```
1061                *DIVIDE
1062                *
1063 04AC C28B   DIVF     MOV R11,R10      ;SAVE RETURN
1064 04AE 06A0           BL @PREP
     04B0 0420'
1065 04B2 2D60             DATA 5*>40+>2C20
1066 04B4 C0C3           MOV R3,R3        ;SOURCE<0?
1067 04B6 110A             JLT DIVF5      ;Y
1068 04B8 1316             JEQ ERR28      ;DIVISION BY ZERO
1069 04BA C081           MOV R1,R2        ;DESTINATION<0?
1070 04BC 110C             JLT DIVF6      ;Y
1071                *
1072 04BE 04C1   DIVF2    CLR R1           ;POSITIVE RESULT
1073 04C0 3C43           DIV R3,R1        ;DIVIDE
1074 04C2 C082           MOV R2,R2        ;REMAINDER?
1075 04C4 16C4             JNE PREP4      ;Y, FLOAT
1076 04C6 C041           MOV R1,R1        ;OVERFLOW?
1077 04C8 11C2             JLT PREP4      ;Y, FLOAT
1078 04CA 10C8           JMP PREP5A       ;N, STORE RESULTS
1079                *
1080 04CC 0503   DIVF5    NEG R3           ;SOURCE<0
1081 04CE C081           MOV R1,R2        ;DESTINATION<0?
1082 04D0 1503             JGT DIVF7      ;N
1083 04D2 0502           NEG R2           ;Y, POSITIVE RESULT
1084 04D4 10F4           JMP DIVF2
1085                *
1086 04D6 0502   DIVF6    NEG R2           ;SIGNS DIFFERENT
1087 04D8 04C1   DIVF7    CLR R1
1088 04DA 3C43           DIV R3,R1        ;DIVIDE
1089 04DC C082           MOV R2,R2        ;REMAINDER?
1090 04DE 16B7             JNE PREP4      ;Y, OVERFLOW
1091 04E0 C041           MOV R1,R1        ;OVERFLOW?
1092 04E2 11B5             JLT PREP4      ;Y
1093 04E4 10BA           JMP PREP5        ;RETURN
1094                *
1095 04E6 2F9C   ERR28    DATA ERROR+28    ;DIVISION BY ZERO
```

```
1097                *RELATIONAL OPERATERS
1098                *
1099        04E8'   FUZF    EQU $
1100        04E8'   EQUF    EQU $
1101        04E8'   LTHF    EQU $
1102        04E8'   LEQF    EQU $
1103        04E8'   GTHF    EQU $
1104        04E8'   GEQF    EQU $
1105        04E8'   NEQUF   EQU $
1106 04E8  C123             MOV @EVATB->AA+EVPR(3),R4
     04EA  0340'
1107 04EC  06A0             BL @SUBF
     04EE  046A'
1108 04F0  0203             LI R3,4             ;GET MATCH CONSTANT
     04F2  0004
1109 04F4  C104             MOV R4,R4           ;FUZZ?
1110 04F6  1609               JNE EQUF1         ;N
1111 04F8  0584             INC R4              ;Y, R4=0001
1112 04FA  C052             MOV *R2,R1          ;GET EXPONENT
1113 04FC  1306               JEQ EQUF1         ;INTEGER
1114 04FE  0241             ANDI R1,>7F00       ;ISOLATE
     0500  7F00
1115 0502  0281             CI R1,FUZZ          ;LESS THAN FUZZ?
     0504  0000
1116 0506  1A06               JL EQUF2A         ;Y, EQUAL
1117 0508  1007             JMP EQUF3           ;N, UNEQUAL, SET R3 TO -2
1118                *
1119 050A  C072  EQUF1      MOV *R2+,R1         ;GET SIGN OF RESULT
1120 050C  1601               JNE EQUF2
1121 050E  C052             MOV *R2,R1
1122 0510  1503  EQUF2        JGT EQUF3
1123 0512  1601               JNE $+4
1124 0514  0603  EQUF2A     DEC R3
1125 0516  0643             DECT R3
1126                *
1127 0518  2E00  EQUF3      CLEAR 0             ;CLEAR FPAC
1128 051A  2103             COC R3,R4           ;CORRESPONDING BITS SET?
1129 051C  160A               JNE UMNF3
1130 051E  05A0             INC @FPAC2          ;RETURN 1
     0520  045E'
1131 0522  1007             JMP UMNF3
```

```
 1133                    *UNARY MINUS
 1134                    *
 1135 0524 2C11  UMNF    LOADF *R1           ;LOAD FPAC
 1136 0526 C051          MOV *R1,R1          ;INTEGER
 1137 0528 1302            JEQ UMNF1         ;Y
 1138 052A 2E40          NEGATE 0            ;N, NEGATE FPAC
 1139 052C 1002          JMP UMNF3
 1140                    *
 1141 052E 0520  UMNF1   NEG @FPAC2
      0530 0520'
 1142                    *
 1143 0532 0202  UMNF3   LI R2,FPAC          ;RETURN
      0534 0462'
 1144 0536 0460  UMNF4   B @EVOP3A
      0538 030A'
 1145 053A   38  B38     BYTE >38
 1146 053B   43  B43     BYTE >43
 1147 053C   45  B45     BYTE >45
 1148 053D   4A  B4A     BYTE >4A
 1149 053E   4C  B4C     BYTE >4C
 1150 053F   FF  BFF     BYTE >FF
 1151 0540               EVEN
 1152                    END
NO ERRORS,      NO WARNINGS
```

```
$          0540'          0154  0514  0671  0865  0867  0870  1099  1100  1101
                          1102  1103  1104  1105  1123
ABSF   R   03F8'   0042   0876
ADDF   D   0442'   0999   0861  0997
ANDF   R   03C6'   0066   0854
ASCF   R   03FA'   0043   0877
ATNF   R   03FC'   0044   0878
B01    D   03ED'   0867   0095
B05    D   03F1'   0870   0095
B38        053A'   1145   0242
B43        053B'   1146   0146  0505
B45        053C'   1147   0149  0508
B4A        053D'   1148   0451  0530
B4C        053E'   1149   0244  0714
BFF        053F'   1150   0690
BITF   R   0410'   0054   0888
C1     D   03EC'   0868   0096  0755
C4     D   03EE'   0869   0096
C6     D   03F6'   0874   0096  0819
CKEX   D   0042'   0240   0238
CKEX1      0054'   0247   0243
CKEX2      0056'   0248   0241  0245
CLEAR          0033
COSF   R   03FE'   0045   0879
CRBF   R   0412'   0055   0889
CRFF   R   0414'   0056   0890
DIVF       04AC'   1063   0862
DIVF2      04BE'   1072   1084
DIVF5      04CC'   1080   1067
DIVF6      04D6'   1086   1070
DIVF7      04D8'   1087   1082
DLIM   R   0066'   0086   0240  0286
EQUF       04E8'   1100   0858
EQUF1      050A'   1119   1110  1113
EQUF2      0510'   1122   1120
EQUF2A     0514'   1124   1116
EQUF3      0518'   1127   1117  1122
ERR17      03B6'   0845   0807  0839  0842
ERR18      0370'   0816
ERR19      03B8'   0847   0827
ERR22      003C'   0212
ERR28      04E6'   1095   1068
ERR38      007E'   0434   0442
ERR39      03BA'   0849   0795
ERROR      2F80    0039   0040  0212  0640  0694  0708  0816  0845  0847  1095
ERROR2     2FA0    0040   0434  0478  0672  0849  0851
EVADL      02C8'   0734   0611  0621  0721
EVADV      0338'   0792   0655
EVADV1     0348'   0799   0814
EVADV2     0372'   0818   0810
EVADVE     03BE'   0851   0833
EVAFN      0082'   0436   0624
EVAFN1     00B2'   0457   0452  0477
EVAFN2     00E4'   0480   0472
EVAFN4     00E8'   0483   0474
EVAFN5     00EE'   0486   0444  0481
EVAL   D   01C8'   0600   0460  0490  0523  0534  0549  0556  0587  0801  0830
EVAL1      01D0'   0605   0619  0645  0705  0707
EVAL2      01D8'   0610   0692  0701  0717
EVAL3      0210'   0634   0616
```

```
EVAL4       026A'   0676  0635
EVAL5       0272'   0681  0638
EVAL6       0278'   0685  0577  0632
EVAL7       027A'   0687  0659  0670  0843
EVAL8       027E'   0690  0788
EVAL8P      0336'   0788  0825
EVAL9       0220'   0644  0703
EVAL91      0224'   0645  0741
EVALS       01B6'   0590  0284  0336
EVALS1      01B8'   0592  0208
EVALS2  D   01C0'   0595  0588
EVAR        0030'   0207  0213
EVARZ   D   003E'   0213  0205
EVATB       03C2'   0853  0745  0865  1106
EVAV        0226'   0649  0613
EVER        005C'   0284  0283
EVER0       02A0'   0708  0749  0759  0776  0785
EVER1       0060'   0285  0159  0210
EVER10      021E'   0640  0667
EVER2       0064'   0286  0339
EVER27      0286'   0694  0608
EVER37      00E0'   0478  0484  0544  0553  0562
EVER40      0266'   0671  0663
EVERZ   D   0058'   0283  0281
EVFIX               0036
EVFP        0202'   0626  0614
EVFX    D   0070'   0335  0333
EVOP        02B8'   0725  0618
EVOP0       02A2'   0712  0771
EVOP0A      02A8'   0714  0720
EVOP0B      02B2'   0719  0728
EVOP1       02CC'   0737  0713  0715  0730
EVOP2       02F2'   0755  0751
EVOP3       0308'   0765  0756  0761
EVOP3A  D   030A'   0767  0094  1144
EVOP4       0314'   0773  0743
EVOP5       0328'   0782  0778
EVOPA       0288'   0700  0726
EVPR        0028    0865  1106
EVRND       01AC'   0581  0650
EVSD        0004'   0145  0144
EVSD1       0026'   0161  0148  0150
EVSD2       002A'   0164  0147
EVSDZ   D   0000'   0144  0142
EVSF        0138'   0527  0623
EVSF1       0152'   0538  0531
EVSF1A      016A'   0548  0539
EVSF1B      0188'   0559  0542  0546
EVSF2       0192'   0564  0536  0551
EVSF4       01A4'   0574  0583
EVSF0       0112'   0505  0538  0543
EVSF01      012C'   0518  0507  0509
EVSF02      012E'   0520  0506
EVSFR   D   01A0'   0571  0093  0501
EVSKB   R   01C2'   0077  0209  0595
EVSKE   R   01C6'   0077  0596
EXPF    R   0400'   0046  0880
FADD                0027
FDIV                0030
FIX     R   0396'   0071  0337  0557  0802  0831
```

```
FLOAT   R  0468'   0072  1017
FLOATF             0035
FLTMZ      0466'   1017  1004
FMUL               0029
FPAC    R  0534'   0078  0582  1014  1143
FPAC2   R  0530'   0078  1012  1130  1141
FSUB               0028
FUZF       04E8'   1099  0857
FUZZ    R  0504'   0081  1115
GEQF       04E8'   1104  0859
GTHF       04E8'   1103  0858
INPF    R  0402'   0047  0881
IOB     R  03AE'   0085  0841
LANDF   R  03C8'   0065  0854
LENF    R  041A'   0059  0893
LEQF       04E8'   1102  0860
LNOTF   R  03CC'   0069  0855
LOADF              0025
LOGF    R  0404'   0048  0882
LORF    R  03C4'   0063  0853
LTHF       04E8'   1101  0859
LXORF   R  03CE'   0064  0856
MCHF    R  041C'   0060  0894
MEMF    R  0416'   0057  0891
MULF       0478'   1032  0862
MULF2      0488'   1040  1051
MULF5      0494'   1047  1036
MULF6      049E'   1053  1038
MULF7      04A0'   1054  1049
MWDF    R  0418'   0058  0892
NEGATE             0034
NEQUF      04E8'   1105  0860
NKYF    R  0406'   0049  0883
NORMAL             0032
NOTF    R  03CA'   0068  0855
NVD     R  0258'   0087  0666
NVS     R  0262'   0088  0664  0669
ORF     R  03C2'   0067  0853
OUTFP              0037
OUTINT             0038
POWF    R  03E6'   0062  0863
PREP       0420'   0983  1000  1024  1033  1064
PREP4      044E'   1004  0988  0990  1028  1042  1044  1056  1058  1075  1077
                         1090  1092
PREP5      045A'   1010  1059  1093
PREP5A     045C'   1012  1003  1027  1045  1078
PREP5B     0460'   1014  1008
R0         0000         0153  0156  0157  0286  0287  0439  0440  0476  0483
                        0513  0515  0528  0541  0550  0552  0561  0602  0603
                        0606  0610  0612  0615  0617  0620  0622  0634  0636
                        0637  0644  0649  0652  0653  0657  0658  0664  0665
                        0666  0668  0669  0676  0677  0678  0679  0682  0685
                        0687  0688  0700  0702  0704  0706  0712  0727  0735
                        0738  0773  0793  0813  0824  0826  0832  0837  0840
                        0841
R1         0001         0338  0443  0455  0457  0466  0471  0473  0529  0545
                        0651  0653  0654  0656  0657  0719  0725  0737  0740
                        0748  0753  0784  0792  0793  0806  0811  0818  0819
                        0838  0840  0983  0987  0991  0992  1002  1010  1012
                        1026  1037  1037  1040  1041  1041  1043  1048  1048
```

				1050	1053	1054	1055	1055	1057	1069	1072	1073
				1076	1076	1081	1087	1088	1091	1091	1112	1114
				1115	1119	1121	1135	1136	1136			
R10		000A		0999	1015	1023	1032	1063				
R11		000B		0247	0335	0516	0521	0567	0569	0600	0745	0755
				0765	0783	0999	1006	1023	1032	1063		
R12		000C		0335	0338							
R13		000D		0287								
R14		000E		0161	0161	0164						
R2		0002		0151	0209	0285	0468	0469	0470	0511	0540	0545
				0554	0560	0574	0575	0576	0582	0758	0763	0767
				0768	0769	0775	0780	0812	0820	0823	0986	0989
				0992	1014	1043	1057	1069	1074	1074	1081	1083
				1086	1089	1089	1112	1119	1121	1143		
R3		0003		0438	0440	0441	0443	0445	0446	0447	0448	0449
				0458	0464	0467	0497	0498	0499	0500	0565	0566
				0742	0744	0987	0989	0991	1002	1026	1035	1035
				1040	1047	1054	1066	1066	1073	1080	1088	1108
				1124	1125	1128						
R4		0004		0437	0459	0462	0486	0488	0495	0496	0498	0499
				0500	0738	0739	0740	0794	0799	0804	0806	0808
				0820	1106	1109	1109	1111	1128			
R5		0005		0207	0436	0450	0454	0475	0489	0522	0527	0532
				0548	0555	0572	0590	0603	0660	0661	0661	0662
				0690	0725	0729	0734	0796	0800	0808	0809	0811
				0822	0828	0829	0835					
R6		0006		0436	0437	0441	0446	0447	0448	0457	0458	0459
				0461	0462	0463	0464	0465	0466	0467	0468	0469
				0470	0486	0487	0491	0492	0493	0494	0494	0495
				0527	0528	0540	0554	0559	0560	0564	0565	0571
				0572	0574	0575	0576	0595	0600	0601	0607	0626
				0627	0628	0629	0630	0631	0681	0682	0683	0687
				0691	0747	0748	0750	0752	0753	0757	0758	0760
				0762	0763	0767	0768	0769	0770	0774	0775	0777
				0779	0780	0782	0783	0796	0797	0799	0803	0804
				0805	0812	0818	0821	0822	0823	0828	0834	0835
				0836	0837							
R7		0007		0596	0605	0606	0607	0691	0714	0716	0719	0737
				0742								
R8		0008		0145	0146	0149	0151	0153	0157	0240	0242	0244
				0288	0451	0453	0480	0487	0488	0492	0505	0508
				0510	0511	0513	0515	0520	0530	0533	0535	0592
				0610	0626	0627	0628	0629	0630	0631	0676	0678
R9		0009		0158	0593							
RANDZ	R	01AE'	0073	0581								
SCALE			0031									
SFUNP		03F8'	0876	0567								
SINF	R	0408'	0050	0884								
SQRF	R	040A'	0051	0885								
SRHF	R	041E'	0061	0895								
STORE			0026									
SUBF	D	046A'	1023	0861	1021	1107						
SYSF	R	040C'	0052	0886								
TEMP	R	0466'	0079	0983	1007	1017						
TEMP2	R	0428'	0080	0984								
TEMP4	R	042E'	0080	0985								
TICF	R	040E'	0053	0887								
UFT	R	0088'	0082	0438								
UMNF		0524'	1135	0863								
UMNF1		052E'	1141	1137								

```
UMNF3       0532'   1143  1129  1131  1139
UMNF4       0536'   1144
VDT     R   033A'   0083  0445  0497  0656  0792
VNT     R   022E'   0084  0651
WPR2    R   0058'   0089  0144  0213  0283
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.IPCOM
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.IPCOM
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.IPCOM
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'IPCOM'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       OUTL1              ;LIST LINE
0009          *       INPEND             ;END INPUT
0010          *       PRTEND             ;END PRINT
0011          *       UCCNT              ;UPDATE CCNT
0012          *
0013          * COPY: (NONE)
0014          *
0015          * MACRO: (NONE)
0016          *
0017          * EXTERNAL ROUTINES:
0018          *
0019                REF TYPB             ;OUTPUT BUFFER
0020                REF TYPC             ;OUTPUT CRLF
0021                REF NLIN             ;EXIT TO MULTIPLEXOR
0022          *
0023          * EXTERNAL DATA:
0024          *
0025                REF CCNT             ;COLUMN COUNTER
0026                REF FFLG             ;FORMATTING FLAG
0027          *
0028          * MODULE EQUATES:  (NONE)
0029          *
0030          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0033            * ABSTRACT:
0034            *
0035            *       OUTL1 - UPDATES CCNT AND PRINTS BUFFER.
0036            *
0037            *       PRTEND - UPDATES CCNT, PRINTS BUFFER, AND
0038            *                AND ENDS WITH A CRLF IF R15=0
0039            *                BEFORE EXITING TO NLIN.
0040            *
0041            *       INPEND - PRINTS A CRLF IF R15=0 AND
0042            *                EXITS TO NLIN.
0043            *
0044            *       UCCNT - UPDATES CCNT SUCH THAT IF REFLECTS
0045            *                WHAT HAS BEEN PRINTED PLUS WHAT
0046            *                IS IN THE BUFFER AND TERMINATES
0047            *                BUFFER WITH NULL.
0048            *
0049            * CALLING SEQUENCE:
0050            *
0051            *       BL @OUTL1
0052            *
0053            *       IN  R7 = BUFFER PTR
0054            *       OUT R7 = BUFFER ADR
0055            *
0056            *
0057            *       B @PRTEND
0058            *
0059            *       IN  R7 = BUFFER PTR
0060            *          R15 = CRLF FLAG
0061            *       EXITS TO NLIN
0062            *
0063            *
0064            *       B @INPEND
0065            *
0066            *       IN R15 = CRLF FLAG
0067            *       EXITS TO NLIN
0068            *
0069            *
0070            *       BL @UCCNT
0071            *
0072            *       IN R7 = BUFFER PTR
0073            *
0074            * EXCEPTIONS AND CONDITIONS:  (NONE)
0075            *
0076            * EXTERNAL ROUTINE LIST:
0077            *
0078            *      (SEE EXTERNAL ROUTINES)
0079            *
0080            * LOCAL DATA:
0081            *
0082            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0084                    * ENTRY POINT:
0085                    *
0086                            DEF OUTL1
0087                    *
0088 0000 C34B  OUTL1   MOV R11,R13     ;SAVE RETURN
0089 0002 06A0          BL @UCCNT       ;UPDATE CCNT
     0004 0026'
0090 0006 06A0          BL @TYPB        ;OUT BUFFER
     0008 0000
0091 000A C1D9          MOV *R9,R7
0092 000C 045D          B *R13          ;RETURN
0093                    *
0094                    *END PRINT LINE
0095                    *
0096                    * ENTRY POINT:
0097                    *
0098                            DEF PRTEND
0099                    *
0100 000E 06A0  PRTEND  BL @UCCNT       ;UPDATE CCNT
     0010 0026'
0101 0012 06A0          BL @TYPB        ;PRINT BUFFER
     0014 0008'
0102                    *
0103                    *END INPUT LINE
0104                    *
0105                    * ENTRY POINT:
0106                    *
0107                            DEF INPEND
0108                    *
0109 0016 04E9  INPEND  CLR @FFLG(9)    ;CLEAR FORMATTING FLAG
     0018 0000
0110 001A C3CF          MOV R15,R15     ;NEED CRLF?
0111 001C 1602            JNE INPE1     ;N
0112 001E 06A0          BL @TYPC        ;Y, OUT CRLF
     0020 0000
0113 0022 0460  INPE1   B @NLIN         ;CONTINUE
     0024 0000
0114                    *
0115                    *UPDATE CCNT
0116                    *
0117                    * ENTRY POINT:
0118                    *
0119                            DEF UCCNT
0120                    *
0121 0026 C029  UCCNT   MOV @CCNT(9),R0 ;GET COUNT
     0028 0000
0122 002A A007          A R7,R0         ;ADD CURRENT PTR
0123 002C 6019          S *R9,R0        ;SUBTRACT IOB
0124 002E 0240          ANDI R0,>7F     ;MOD 128
     0030 007F
0125 0032 CA40          MOV R0,@CCNT(9) ;SAVE RESULT
     0034 0028'
0126 0036 D5C0          MOVB R0,*R7     ;TERMINATE STRING
0127 0038 045B          RT
0128                    END
NO ERRORS,     NO WARNINGS
```

CCNT	R	0034'	0025	0121	0125				
FFLG	R	0018'	0026	0109					
INPE1		0022'	0113	0111					
INPEND	D	0016'	0109	0107					
NLIN	R	0024'	0021	0113					
OUTL1	D	0000'	0088	0086					
PRTEND	D	000E'	0100	0098					
R0		0000		0121	0122	0123	0124	0125	0126
R11		000B		0088					
R13		000D		0088	0092				
R15		000F		0110	0110				
R7		0007		0091	0122	0126			
R9		0009		0091	0123				
TYPB	R	0014'	0019	0090	0101				
TYPC	R	0020'	0020	0112					
UCCNT	D	0026'	0121	0089	0100	0119			

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.POWF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.POWF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.POWF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'POWF'
0002          *
0003          * REVISION: 12/01/78 1.01     ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00     ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       POWF              ;POWER FUNCTION
0009          *
0010          * COPY: (NONE)
0011          *
0012          * MARCO: (NONE)
0013          *
0014          * EXTERNAL ROUTINES:
0015          *
0016                REF LOGF            ;LOG FUNCTION
0017                REF EXPF            ;EXP FUNCTION
0018                REF EVOP3A          ;EXIT TO EVAL
0019                DXOP LOADF,0        ;LOAD FPAC
0020                DXOP STORE,1        ;STORE FPAC
0021                DXOP FMUL,4                 ;MULTIPLY FPAC
0022                DXOP FLOATF,10      ;FLOAT FPAC
0023          *
0024          * EXTERNAL DATA:
0025          *
0026                REF TEMP,TEMP6      ;TEMPORARY STORAGE
0027          *
0028          * MODULE EQUATES:  (NONE)
0029          *
0030          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0033                * ABSTRACT:
0034                *
0035                *       RAISE (R1) TO THE (R2) POWER USING
0036                *       LOG AND EXPONENTIAL FUNCTIONS.
0037                *
0038                *               R1 ↑ R2 = EXP( R1 * LOG(R2))
0039                *
0040                * CALLING SEQUENCE:
0041                *
0042                *       B @POWF
0043                *
0044                *       IN  (R1) = ARG1
0045                *           (R2) = ARG2
0046                *       OUT (R2) = RESULT
0047                *
0048                *       EXIT TO EVOP3A
0049                *
0050                * EXCEPTIONS AND CONDITIONS:
0051                *
0052                *       PRESERVE R0
0053                *       EXP,LOG, AND FP ERRORS
0054                *
0055                * EXTERNAL ROUTINE LIST:
0056                *
0057                *      (SEE EXTERNAL ROUTINES)
0058                *
0059                * LOCAL DATA:
0060                *
0061                *      (SEE MODULE VARIABLES AND CONSTANTS)
0062                *
0063                * ENTRY POINT:
0064                *
0065                      DEF POWF
0066                *
0067 0000 C101     POWF   MOV R1,R4        ;PERSERVE R0,R1
0068 0002 C800            MOV R0,@TEMP6
     0004 0000
0069 0006 06A0            BL @LOGF        ;GET LOG(R2)
     0008 0000
0070 000A 2C60            STORE @TEMP     ;MOVE TO TEMP
     000C 0000
0071 000E 2C14            LOADF *R4       ;GET B
0072 0010 2E80            FLOATF 0        ;FLOAT IF NECESSARY
0073 0012 2D20            FMUL @TEMP      ;MULTIPLY BY TEMP
     0014 000C'
0074 0016 06A0            BL @EXPF        ;GET EXP(R1*LOG(R2))
     0018 0000
0075 001A C020            MOV @TEMP6,R0   ;RESTORE R0
     001C 0004'
0076 001E 0460            B @EVOP3A
     0020 0000
0077                      END
NO ERRORS,     NO WARNINGS
```

```
EVOP3A  R  0020'   0018  0076
EXPF    R  0018'   0017  0074
FLOATF             0022
FMUL               0021
LOADF              0019
LOGF    R  0008'   0016  0069
POWF    D  0000'   0067  0065
R0         0000          0068  0075
R1         0001          0067
R4         0004          0067  0071
STORE              0020
TEMP    R  0014'   0026  0070  0073
TEMP6   R  001C'   0026  0068  0075
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.LOGICOPS
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.LOGICOPS
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.LOGICOPS
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'LOGICOPS'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       LNOTF             ;LOGICAL NOT
0009          *       LORF              ;LOGICAL OR
0010          *       LXORF             ;LOGICAL EXCLUSIVE OR
0011          *       LANDF             ;LOGICAL AND
0012          *
0013          * COPY: (NONE)
0014          *
0015          * MACRO: (NONE)
0016          *
0017          * EXTERNAL ROUTINES:
0018          *
0019                REF FIX             ;FIX ARGUMENT
0020                REF EVOP3A          ;EXIT ENTRY TO EVALUATOR
0021                DXOP CLEAR,8        ;CLEAR FPAC
0022          *
0023          * EXTERNAL DATA:
0024          *
0025                REF FPAC,FPAC2      ;FLOATING POINT ACCUMULATOR
0026          *
0027          * MODULE EQUATES:  (NONE)
0028          *
0029          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0032            * ABSTRACT:
0033            *
0034            *       PERFORM LOGICAL OPERATIONS OF
0035            *
0036            *       LOR     = LOGICAL 'OR'
0037            *       LNOT    = LOGICAL 'NOT'
0038            *       LXOR    = LOGICAL EXCLUSIVE 'OR'
0039            *       LAND    = LOGICAL 'AND'
0040            *
0041            * CALLING SEQUENCE:
0042            *
0043            *       B @LNOTF
0044            *       B @LORF
0045            *       B @LXORF
0046            *       B @LANDF
0047            *
0048            *       IN  (R1) = ARG1
0049            *           (R2) = ARG2
0050            *       OUT (R2) = RESULT (1 OR 0)
0051            *       EXIT TO EVOP3A
0052            *
0053            * EXCEPTIONS AND CONDITIONS:  (NONE)
0054            *
0055            * EXTERNAL ROUTINE LIST:
0056            *
0057            *      (SEE EXTERNAL ROUTINES)
0058            *
0059            * LOCAL DATA:
0060            *
0061            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0063                    *
0064                    *LOGICAL OPERATERS
0065                    *
0066                    *       R1 = XXI R4,..
0067                    *       R2 = DATA
0068                    *       R3 = B *R10
0069                    *
0070 0000 C28B  LFIX    MOV R11,R10
0071 0002 2E00          CLEAR 0
0072 0004 C101          MOV R1,R4
0073 0006 06A0          BL @FIX
     0008 0000
0074 000A C084          MOV R4,R2
0075 000C C101          MOV R1,R4       ;STORE OBJECT
0076 000E 06A0          BL @FIX
     0010 0008'
0077 0012 C081          MOV R1,R2
0078 0014 C07A          MOV *R10+,R1
0079 0016 0203          LI R3,>045A     ;B *R10
     0018 045A
0080 001A 0441          B R1
0081                    *
0082                    * ENTRY POINT:
0083                    *
0084                    DEF LORF
0085                    *
0086 001C 06A0  LORF    BL @LFIX
     001E 0000'
0087 0020 0264            DATA >0264    ;ORI R4,..
0088 0022 1008          JMP LANDF1
0089                    *
0090                    * ENTRY POINT:
0091                    *
0092                    DEF LXORF
0093                    *
0094 0024 06A0  LXORF   BL @LFIX
     0026 0000'
0095 0028 045A            DATA >045A    ;B *R10
0096 002A 2902          XOR R2,R4
0097 002C 1003          JMP LANDF1
0098                    *
0099                    * ENTRY POINT:
0100                    *
0101                    DEF LANDF
0102                    *
0103 002E 06A0  LANDF   BL @LFIX
     0030 0000'
0104 0032 0244            DATA >0244    ;ANDI R4,..
0105                    *
0106 0034 C804  LANDF1  MOV R4,@FPAC2
     0036 0000
0107                    *
0108 0038 0202  LANDF2  LI R2,FPAC
     003A 0000
0109 003C 0460          B @EVOP3A
     003E 0000
0110                    *
0111                    * ENTRY POINT:
0112                    *
0113                    DEF LNOTF
```

```
0114                *
0115                *LOGICAL NOT
0116                *
0117 0040 2E00  LNOT    CLEAR 0          *CLEAR FPAC
0118 0042 C081          MOV R1,R2
0119 0044 06A0          BL @FIX
     0046 0010'
0120 0048 0541          INV R1
0121 004A C801          MOV R1,@FPAC2
     004C 0036'
0122 004E 10F4          JMP LANDF2
0123                *
0124      0041' LNOTF   EQU LNOT+1
0125                    END
NO ERRORS,     NO WARNINGS
```

CLEAR			0021								
EVOP3A	R	003E'	0020	0109							
FIX	R	0046'	0019	0073	0076	0119					
FPAC	R	003A'	0025	0108							
FPAC2	R	004C'	0025	0106	0121						
LANDF	D	002E'	0103	0101							
LANDF1		0034'	0106	0088	0097						
LANDF2		0038'	0108	0122							
LFIX		0000'	0070	0086	0094	0103					
LNOT		0040'	0117	0124							
LNOTF	D	0041'	0124	0113							
LORF	D	001C'	0086	0084							
LXORF	D	0024'	0094	0092							
R1		0001		0072	0075	0077	0078	0080	0118	0120	0121
R10		000A		0070	0078						
R11		0008		0070							
R2		0002		0074	0077	0096	0108	0118			
R3		0003		0079							
R4		0004		0072	0074	0075	0096	0106			

*

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.RELATOPS
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.RELATOPS
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.RELATOPS
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'RELATOPS'
0002          *
0003          * REVISION: 12/31/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       ANDF              ;AND FUNCTION
0009          *       ORF               ;OR FUNCTION
0010          *       NOTF              ;NOT FUNCTION
0011          *
0012          * COPY: (NONE)
0013          *
0014          * MACRO: (NONE)
0015          *
0016          * EXTERNAL ROUTINES:
0017          *
0018                DXOP CLEAR,8              ;CLEAR FPAC
0019          *
0020          * EXTERNAL DATA:
0021          *
0022                REF FPAC,FPAC2                ;FLOATING POINT ACCUMULA
0023          *
0024          * MODULE EQUATES: (NONE)
0025          *
0026          *
0027          * MODULE VARIABLES AND CONSTANTS: (NONE)
0028          *
```

```
0031            * ABSTRACT:
0032            *
0033            *       PERFORM 'AND' FUNCTION ON TWO OPERANDS
0034            *       AND RETURN A VALUE OF 1 OR 0 ACCORDINGLY.
0035            *
0036            *
0037            * CALLING SEQUENCE:
0038            *
0039            *       BL @ANDF
0040            *
0041            *       IN (R1) = OPERAND 1
0042            *          (R2) = OPERAND 2
0043            *
0044            *       OUT (R2) = 1 OR 0 IN FPAC
0045            *
0046            *      NORMAL EXIT - RETURN
0047            *
0048            * EXCEPTIONS AND CONDITIONS:  (NONE)
0049            *
0050            * EXTERNAL ROUTINE LIST:
0051            *
0052            *      (SEE EXTERNAL ROUTINES)
0053            *
0054            * LOCAL DATA:
0055            *
0056            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0058                    * ENTRY POINT:
0059                    *
0060                            DEF ANDF
0061                    *
0062                    *AND OPERATOR
0063                    *
0064 0000 2E00  ANDF    CLEAR 0          ;CLEAR FPAC
0065 0002 C0F1          MOV *R1+,R3      ;INTEGER?
0066 0004 1602            JNE ANDF1      ;N
0067 0006 C0D1          MOV *R1,R3       ;Y, GET INTEGER
0068 0008 1306            JEQ ANDF3      ;0, RESULT=0
0069                    *
0070 000A C0F2  ANDF1   MOV *R2+,R3      ;INTEGER?
0071 000C 1602            JNE ANDF2      ;RESULT=1
0072 000E C0D2          MOV *R2,R3       ;Y
0073 0010 1302            JEQ ANDF3      ;0, RESULT=0
0074                    *
0075 0012 05A0  ANDF2   INC @FPAC2       ;RESULT=1
     0014 0000
0076                    *
0077 0016 0202  ANDF3   LI R2,@FPAC
     0018 0000
0078 001A 0458          RT
```

```
0081            * ABSTRACT:
0082            *
0083            *       PERFORM AN 'OR' OPERATION ON THE
0084            *       2 OPERANDS AND RETURN A VALUE OF
0085            *       1 FOR TRUE AND 0 FOR FALSE.
0086            *
0087            * CALLING SEQUENCE:
0088            *
0089            *       BL @ORF
0090            *
0091            *       IN (R1) = OPERAND 1
0092            *          (R2) = OPERAND 2
0093            *
0094            *       OUT (R2) = 1 OR 0 IN FPAC
0095            *
0096            *      NORMAL EXIT - RETURN
0097            *
0098            * EXCEPTIONS AND CONDITIONS:  (NONE)
0099            *
0100            * EXTERNAL ROUTINE LIST:
0101            *
0102            *      (SEE EXTERNAL ROUTINES)
0103            *
0104            * LOCAL DATA:
0105            *
0106            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0108                    * ENTRY POINT:
0109                    *
0110                            DEF ORF
0111                    *
0112                    *OR OPERATOR
0113                    *
0114 001C 2E00  ORF     CLEAR 0          ;CLEAR FPAC
0115 001E C0F1          MOV *R1+,R3
0116 0020 16F8            JNE ANDF2      ;RESULT=1
0117 0022 C0D1          MOV *R1,R3
0118 0024 16F6            JNE ANDF2
0119 0026 C0F2          MOV *R2+,R3
0120 0028 16F4            JNE ANDF2      ;RESULT=1
0121 002A C0D2          MOV *R2,R3
0122 002C 16F2            JNE ANDF2
0123 002E 10F3          JMP ANDF3        ;RESULT=0
```

```
0126              * ABSTRACT:
0127              *
0128              *       PERFORM A 'NOT' FUNCTION ON THE OPERAND
0129              *
0130              * CALLING SEQUENCE:
0131              *
0132              *       BL @NOTF
0133              *
0134              *       IN (R1) = OPERAND
0135              *
0136              *       OUT (R2) = 1 OR 0 IN FPAC
0137              *
0138              *
0139              * EXCEPTIONS AND CONDITIONS:  (NONE)
0140              *
0141              * EXTERNAL ROUTINE LIST:
0142              *
0143              *      (SEE EXTERNAL ROUTINES)
0144              *
0145              * LOCAL DATA:
0146              *
0147      0031' NOTF    EQU     NOT+1   #INDICATE AS UNARY OPERATION
```

```
 0149                   * ENTRY POINT:
 0150                   *
 0151                           DEF NOTF
 0152                   *
 0153 0030 2E00  NOT     CLEAR 0          ;CLEAR FPAC
 0154 0032 C0F1          MOV *R1+,R3
 0155 0034 16F0            JNE ANDF3
 0156 0036 C0D1          MOV *R1,R3
 0157 0038 13EC            JEQ ANDF2
 0158 003A 10ED          JMP ANDF3
 0159                   *
 0160                         END
NO ERRORS,      NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES								
ANDF	D	0000'	0064	0060								
ANDF1		000A'	0070	0066								
ANDF2		0012'	0075	0071	0116	0118	0120	0122	0157			
ANDF3		0016'	0077	0068	0073	0123	0155	0158				
CLEAR			0018									
FPAC	R	0018'	0022	0077								
FPAC2	R	0014'	0022	0075								
NOT		0030'	0153	0147								
NOTF	D	0031'	0147	0151								
ORF	D	001C'	0114	0110								
R1		0001		0065	0067	0115	0117	0154	0156			
R2		0002		0070	0072	0077	0119	0121				
R3		0003		0065	0067	0070	0072	0115	0117	0119	0121	0154
				0156								

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.GETP2
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.GETP2
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.GETP2
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'GETP2'
0002            *
0003            * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004            *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005            *
0006            * ROUTINE LIST:
0007            *
0008            *       GETP2             ;GET POWER OF 2
0009            *
0010            * COPY: (NONE)
0011            *
0012            * MARCO: (NONE)
0013            *
0014            * EXTERNAL ROUTINES:
0015            *
0016                    DXOP FMUL,4       ;MULTIPLY FPAC
0017            *
0018            * EXTERNAL DATA:
0019            *
0020                    REF CVCH          ;VARIABLE HOLDER
0021            *
0022            * MODULE EQUATES:  (NONE)
0023            *
0024            *
0025            * MODULE VARIABLES AND CONSTANTS:  (NONE)
0027            *
0028                  TITL
```

```
0030                * ABSTRACT:
0031                *
0032                *       GETP2 WILL RETURN A 3 WORD FLOATING
0033                *       POINT POWER OF 2 AS SPECIFIED BY R2.
0034                *
0035                * CALLING SEQUENCE:
0036                *
0037                *       BL @GETP2
0038                *
0039                *       IN R2    = 1,2 OR 3
0040                *       OUT CVCH = #
0041                *
0042                *
0043                * EXCEPTIONS AND CONDITIONS:  (NONE)
0044                *
0045                *
0046                * EXTERNAL ROUTINE LIST:
0047                *
0048                *      (SEE EXTERNAL ROUTINES)
0049                *
0050                * LOCAL DATA:
0051                *
0052                *      (SEE MODULE VARIABLES AND CONSTANTS)
0053                *
0054                * ENTRY POINT:
0055                *
0056                        DEF GETP2
0057                *
0058 0000 0202     GETP2    LI R2,GETPC-2
     0002 0012'
0059 0004 0812              SRA R2,1            ;/2
0060 0006 A083              A R3,R2             ;INDEX
0061 0008 0A12              SLA R2,1            ;MAKE WORD INDEX
0062 000A C812              MOV *R2,@CVCH       ;MOVE INTO CONSTANT
     000C 0000
0063 000E 2D20              FMUL @CVCH          ;MULTIPLY FPAC
     0010 000C'
0064 0012 045B              RT
0065                *
0066 0014 4120     GETPC    DATA >4120,>4140,>4180  POWERS OF 2
     0016 4140
     0018 4180
0067                        END
NO ERRORS,     NO WARNINGS
```

```
CVCH    R  0010'   0020  0062  0063
FMUL               0016
GETP2   D  0000'   0058  0056
GETPC      0014'   0066  0058
R2         0002          0058  0059  0060  0061  0062
R3         0003          0060
*
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.CVGC
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.CVGC
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.CVGC
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'CVGC'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       CVGCN             ;GET CONSTANT
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MACRO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:  (NONE)
0015              *
0016              * EXTERNAL DATA:  (NONE)
0017              *
0018              * MODULE EQUATES:  (NONE)
0019              *
0020              * MODULE VARIABLES AND CONSTANTS:
0021              *
0022                     DEF CVC10        ;FLOATING POINT CONSTANT 10.
```

```
0025                    * ABSTRACT:
0026                    *
0027                    *       CVGCN WILL GET THE POWER OF TEN CLOSEST
0028                    *       TO THE POWER OF 16 IN R8.  IE:
0029                    *
0030                    *               16↑1  >  10↑1
0031                    *               16↑2  >  10↑2
0032                    *               16↑3  >  10↑3
0033                    *               16↑4  >  10↑4
0034                    *               16↑5  >  10↑6
0035                    *               16↑6  >  10↑7
0036                    *               ....     ....
0037                    *
0038                    * CALLING SEQUENCE:
0039                    *
0040                    *       BL @CVGCN
0041                    *
0042                    *       IN    R8 = POWER OF 16
0043                    *       OUT (R0) = FP CONSTANT
0044                    *
0045                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0046                    *
0047                    * EXTERNAL ROUTINE LIST:  (NONE)
0048                    *
0049                    * LOCAL DATA:  (NONE)
0050                    *
0051                    * ENTRY POINT:
0052                    *
0053                            DEF CVGCN,CVGCN1
0054                    *
0055 0000 0288  CVGCN   CI R8,5
     0002 0005
0056 0004 1A06            JL CVGCN2
0057 0006 0588          INC R8              EXP CHANGE >=5
0058 0008 0288  CVGCN1  CI R8,10
     000A 000A
0059 000C 1A02            JL CVGCN2
0060 000E 0208          LI R8,9             EXP CHANGE >9, USE 9
     0010 0009
0061                    *
0062 0012 04BB  CVGCN2  X *R11+             ADJUST DECIMAL COUNTER
0063 0014 0200          LI R0,CVTB1         GET 10'S TABLE ADR
     0016 0022'
0064 0018 0A18          SLA R8,1            MAKE WORD INDEX
0065 001A A008          A R8,R0             INDEX
0066 001C 0A18          SLA R8,1            R8 X 2
0067 001E A008          A R8,R0             R3=R3+3*R8
0068 0020 045B          RT                  RETURN
0069                    *
0070 0022 4110  CVTB1   DATA >4110,>0000,>0000  10↑0
     0024 0000
     0026 0000
0071 0028 41A0  CVC10   DATA >41A0,>0000,>0000  10↑1
     002A 0000
     002C 0000
0072 002E 4264          DATA >4264,>0000,>0000  10↑2
     0030 0000
     0032 0000
0073 0034 433E          DATA >433E,>8000,>0000  10↑3
     0036 8000
```

```
      0038 0000
 0074 003A 4427           DATA >4427,>1000,>0000  10↑4
      003C 1000
      003E 0000
 0075 0040 4518           DATA >4518,>6A00,>0000  10↑5
      0042 6A00
      0044 0000
 0076 0046 45F4           DATA >45F4,>2400,>0000  10↑6
      0048 2400
      004A 0000
 0077 004C 4698           DATA >4698,>9680,>0000  10↑7
      004E 9680
      0050 0000
 0078 0052 475F           DATA >475F,>5E10,>0000  10↑8
      0054 5E10
      0056 0000
 0079 0058 483B           DATA >483B,>9ACA,>0000  10↑9
      005A 9ACA
      005C 0000
 0080                     END
NO ERRORS,     NO WARNINGS
```

```
CVC10   D  0028'   0071  0022
CVGCN   D  0000'   0055  0053
CVGCN1  D  0008'   0058  0053
CVGCN2     0012'   0062  0056  0059
CVTB1      0022'   0070  0063
R0         0000           0063  0065  0067
R11        000B           0062
R8         0008           0055  0057  0058  0060  0064  0065  0066  0067
*
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL3.BASIC.SOURCE.INTR
OBJECT ACCESS NAME=     BD1.DBASIC.REL3.BASIC.OBJECT.INTR
LISTING ACCESS NAME=    BD1.DBASIC.REL3.BASIC.LIST.INTR
ERROR ACCESS NAME=
OPTIONS=
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'INTR'
0002            *       28 MAR 1979
0003            *
0004                    DEF IMKY,TRPY,IRTY
0005                    REF WPR2,RTNY
0006                    REF ITAB,NLIN,INTSP,INTSB
0007                    REF NB                  !HJC-27 MAR 1979
0008                    REF OWNDO               !HJC-27 MAR 1979
0009                    REF CWNDO               !HJC-27 MAR 1979
0010                    DXOP EVFIX,11
0011    2F80    ERROR   EQU >2F80
0012            *
0013            *TRAP COMMAND
0014            *
0015 0000 2EC1  TRPY    EVFIX R1                ;GET INTERRUPT LEVEL
0016 0002 0241          ANDI R1,>F              ;MASK
     0004 000F
0017            *  CHECK N ,TRAP LEVEL GT NB -HJC 27 MAR 1979
0018 0006 8801          C   R1,@NB
     0008 0000
0019 000A 1207          JLE ERR1
0020 000C 0A11          SLA R1,1                ;X 2
0021 000E 0280          CI R0,>3800             ;TO?
     0010 3800
0022 0012 1603            JNE ERR1              ;N, ERROR
0023 0014 2EE1          EVFIX @ITAB(1)          ;STORE IN TABLE
     0016 0000
0024 0018 100F          JMP IMKY1
0025            *
0026 001A 2F81  ERR1    DATA ERROR+1
0027            *
0028            *SET INTERRUPT MASK
0029            *
0030 001C 2EC1  IMKY    EVFIX R1                ;GET MASK
0031 001E C801          MOV  R1,@OWNDO          ;SET OPEN WINDOW
     0020 0000
0032 0022 8801          C    R1,@NB
     0024 0008'
0033 0026 1102          JLT  IMKY0
0034 0028 C060          MOV  @NB,R1
     002A 0024'
0035 002C C801  IMKY0   MOV  R1,@CWNDO
     002E 0000
0036 0030 C060          MOV  @OWNDO,R1          ;RESTORE R1 IN CASE CORRUPTING IT G
     0032 0020'
0037 0034 0420          BLWP @IMK1Z             REPLACED BY 2 NOPS - PATCH 13/12/79
     0036 003C'                                                    SOURCE MOD 3/8/80
0038 0038 0460  IMKY1   B @NLIN
     003A 0000                                  INTERRUPT MASK BEING SET BEFORE RETURNING
0039            *
0040 003C 0000  IMK1Z   DATA WPR2,IMK1          PENDING INTERRUPTS .'. BEING RECOGNISED
     003E 0040'
0041            *                              SHOULD BE CIMASK VALUE AND GETTING CO
0042 0040 C3ED  IMK1    MOV @2(13),R15
     0042 0002
0043 0044 0380          RTWP
0044            *
0045            *RETURN FROM INTERRUPT
0046            *
0047 0046 0420  IRTY    BLWP @IRT1Z
```

```
Y,IRTY
Y
N,INTSP,INTSB
        !HJC-27 MAR 1979
        !HJC-27 MAR 1979
        !HJC-27 MAR 1979
1



        ;GET INTERRUPT LEVEL
        ;MASK

VEL GT NB -HJC 27 MAR 1979



        ;X 2
        ;TO?

        ;N, ERROR
.)      ;STORE IN TABLE




        ;GET MASK
)0      ;SET OPEN WINDOW




0

1       ;RESTORE R1 IN CASE CORRUPTING IT G
        REPLACED BY 2 NOPS- PATCH   13/12/79          FOUND BY TIMEX DEC 79
                            SOURCE MOD  3/1/80  JGHGPV
        INTERRUPT MASK BEING SET BEFORE RETURNING TO NWIN-
1       PENDING INTERRUPTS .'. BEING RECOGNISED BEFORE THEY
        SHOULD BE (IMASK VALUE ALSO GETTING CORRUPTED).
5


PT
```

```
      0048 004E'
 0048 004A 0460          B @RTNY
      004C 0000
 0049              *
 0050 004E 003C' IRT1Z   DATA WPR2,IRT1
      0050 0052'
 0051              *
 0052 0052 0300  IRT1    LIMI 0              ;DISABLE INTERRUPTS
      0054 0000
 0053 0056 C060          MOV @INTSP,R1
      0058 0000
 0054 005A 0281          CI R1,INTSB         ;ANY TO POP?
      005C 0000
 0055 005E 1206            JLE ERR12
 0056 0060 0641          DECT R1             ;Y, POP
 0057 0062 C811          MOV *R1,@OWNDO      ;TAKE VALUE OFF OWNDO STACK,PUT IN
      0064 0032'
 0058 0066 C801          MOV R1,@INTSP       ;UPDATE STACK PTR
      0068 0058'
 0059 006A 0380          RTWP
 0060              *
 0061 006C 2F8C  ERR12   DATA ERROR+12
 0062                    END
NO ERRORS,      NO WARNINGS
*
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.FP
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.FP
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.FP
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'FP'
0002            *
0003            * REVISION: 12/31/78 1.01      ;SOURCE STANDARDIZATION
0004            *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005            *
0006            * ROUTINE LIST:
0007            *
0008            *       FAD               ;ADD TO FPAC
0009            *       FSD               ;SUBTRACT FROM FPAC
0010            *       FMD               ;MULTIPLY FPAC
0011            *       FDD               ;DIVIDE FPAC
0012            *       FLDD              ;LOAD FPAC
0013            *       FSRD              ;STORE FPAC
0014            *       FNEG              ;NEGATE FPAC
0015            *       FCLR              ;CLEAR FPAC
0016            *       FNRM              ;NORMALIZE FPAC
0017            *       FSCL              ;SCALE FPAC
0018            *       FLOAT             ;FLOAT FPAC
0019            *
0020            *       FADDI             ;3 WRD ADDITION
0021            *       FSUBI             ;3 WRD SUBTRACTION
0022            *
0023            *       THE FLOATING POINT ACCUMULATOR IS THE FIRST 3
0024            *       WORDS OF THE FLOATING POINT REGISTERS  (R0,R1,R2).
0025            *
0026            *       ALL FLOATING POINT OPERATIONS ASSUME NORMALIZED
0027            *       NUMBERS AS INPUTS AND ALL RESULTS ARE NORMALIZED.
0028            *
0029            *       THE FORM OF A FLOATING POINT NUMBER IS AS FOLLOWS:
0030            *
0031            *       1ST WORD          SCCC CCCC MMMM MMMM
0032            *       2ND WORD          MMMM MMMM MMMM MMMM
0033            *       3RD WORD          MMMM MMMM MMMM MMMM
0034            *
0035            *       WHERE  S = SIGN BIT
0036            *              C = 7 BIT, EXCESS >40 CHARACTERISTIC
0037            *              M = 40 BIT UNSIGNED MAGNITUDE MANTISSA
0038            *                  0 <= M < 1
0039            *
0040            *       A NUMBER IS NORMALIZED WHEN THE 1ST HEX DIGIT
0041            *       OF THE MANTISSA IS NON-ZERO.
0042            *
0043            *       A TRUE ZERO (ALL ZERO'S) IS USED FOR ZERO.
0044            *
0045            * COPY: (NONE)
0046            *
0047            * MACRO: (NONE)
0048            *
0049            * EXTERNAL ROUTINES:
0050            *
0051    2F80    ERROR   EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0052    2FA0    ERROR2  EQU ERROR+>20
0053            *
0054            * EXTERNAL DATA:  (NONE)
0055            *
0056            * MODULE EQUATES:  (NONE)
0057            *
0058            * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0061                    * ABSTRACT:
0062                    *
0063                    *       LOAD INTO FPAC A 3 WRD FLOATING POINT #.
0064                    *
0065                    * CALLING SEQUENCE:
0066                    *
0067                    *       XOP XX,1
0068                    *
0069                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0070                    *
0071                    * EXTERNAL ROUTINE LIST:  (NONE)
0072                    *
0073                    * LOCAL DATA:  (NONE)
0074                    *
0075                    * ENTRY POINT:
0076                    *
0077                            DEF FSRD
0078                    *
0079 0000 CEC0  FSRD    MOV R0,*R11+
0080 0002 CEC1          MOV R1,*R11+
0081 0004 C6C2          MOV R2,*R11
0082 0006 0380          RTWP
```

```
0085                    * ABSTRACT:
0086                    *
0087                    *       LOAD FLOATING POINT ACCUMULATOR WITH 3RD FLOATING
0088                    *       POINT NUMBER.
0089                    *
0090                    * CALL SEQUENCE:
0091                    *
0092                    *       XOP XX,0
0093                    *
0094                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0095                    *
0096                    * EXTERNAL ROUTINE LIST:  (NONE)
0097                    *
0098                    * LOCAL DATA:  (NONE)
0099                    *
0100                    * ENTRY POINT:
0101                    *
0102                            DEF FLDD
0103                    *
0104 0008 C03B  FLDD    MOV *R11+,R0    ;LOAD FPAC
0105 000A C07B  FLDD1   MOV *R11+,R1
0106 000C C09B          MOV *R11,R2
0107 000E 0380  FLDD2   RTWP
```

```
0110                    * ABSTRACT:
0111                    *
0112                    *       ADD OR SUBTRACT FROM FPAC A 3 WORD FLOATING
0113                    *       POINT NUMBER.
0114                    *
0115                    * CALL SEQUENCE:
0116                    *
0117                    *       XOP XX,2        ;ADD TO FPAC
0118                    *
0119                    *       XOP XX,3        ;SUBTRACT FROM FPAC
0120                    *
0121                    * EXCEPTIONS AND CONDITIONS:
0122                    *
0123                    *       ERROR 29 CAN RESULT FROM OVERFLOW
0124                    *
0125                    * EXTERNAL ROUTINE LIST:
0126                    *
0127                    *       FLDD            ;LOAD FPAC
0128                    *       FARS            ;SHIFT FPAC RIGHT 1 HEX DIGIT
0129                    *       FADDI           ;ADD R4,R5,R6 TO R0,R1,R2
0130                    *
0131                    * LOCAL DATA:  (NONE)
0132                    *
0133                    *ENTRY POINT:
0134                    *
0135                            DEF FSD         ;SUBTRACT FROM FPAC
0136                    *
0137                    *       SUBTRACTION IS MADE SIMPLY BY TOGGLING SIGN BIT
0138                    *
0139 0010 C13B  FSD     MOV *R11+,R4    ;GET FIRST WORD, ZERO?
0140 0012 13FD            JEQ FLDD2     ;Y, ZERO, RETURN
0141 0014 0224          AI R4,>8000     ;N, TOGGLE SIGN BIT
     0016 8000
0142 0018 1002          JMP FAD0
0143                    *
0144                    * ENTRY POINT:
0145                    *
0146                            DEF FAD
0147                    *
0148 001A C13B  FAD     MOV *R11+,R4    ;GET FIRST WORD, ZERO?
0149 001C 13F8            JEQ FLDD2     ;Y, RETURN
0150 001E C000  FAD0    MOV R0,R0       ;N, FPAC ZERO?
0151 0020 1602            JNE FAD0A     ;N
0152 0022 C004          MOV R4,R0       ;Y, MOVE TEMP TO FPAC
0153 0024 10F2          JMP FLDD1
0154                    *
0155 0026 C17B  FAD0A   MOV *R11+,R5    ;LOAD R4,R5,R6
0156 0028 C19B          MOV *R11,R6
0157 002A D0C0          MOVB R0,R3      GET EXPONENTS
0158 002C D1C4          MOVB R4,R7
0159 002E 7000          SB R0,R0        ISOLATE
0160 0030 7104          SB R4,R4
0161 0032 0A13          SLA R3,1        REMOVE SIGN
0162 0034 1702            JNC FAD1      NEGATIVE?
0163 0036 06A0          BL @FADN        Y, NEGATE
     0038 0088'
0164                    *
0165 003A 0993  FAD1    SRL R3,9
0166 003C 0A17          SLA R7,1        REMOVE SIGN
0167 003E 1708            JNC FAD2      NEGATIVE?
```

```
0168 0040 0506           NEG R6          Y, NEGATE
0169 0042 1604             JNE FAD1A
0170 0044 0505           NEG R5
0171 0046 1603             JNE FAD1B
0172 0048 0504           NEG R4
0173 004A 1002             JMP FAD2
0174 004C 0545  FAD1A    INV R5
0175 004E 0544  FAD1B    INV R4
0176                     *
0177 0050 020A  FAD2     LI R10,10       10 SHIFTS GIVE ZERO RESULT
     0052 000A
0178 0054 0997           SRL R7,9        POSITION EXPONENT
0179                     *
0180 0056 81C3  FAD2A    C R3,R7         COMPARE EXPONENTS
0181 0058 131D             JEQ FAD4      SAME, DO ADD
0182 005A 1509             JGT FAD3      SHIFT TEMP (IS R3>R7?)
0183 005C 06A0           BL @FARS        N, R3<R7, SHIFT FPAC
     005E 02D2'
0184 0060 060A           DEC R10         COUNT SHIFTS
0185 0062 16F9             JNE FAD2A     OK
0186 0064 C004           MOV R4,R0       FPAC=0, MOVE TEMP TO FPAC
0187 0066 C045           MOV R5,R1
0188 0068 C086           MOV R6,R2
0189 006A C0C7           MOV R7,R3       MOVE EXPONENT
0190 006C 100C           JMP FAD3A
0191                     *
0192 006E 0946  FAD3     SRL R6,4        SHIFT TEMP 1 HEX DIGIT RIGHT
0193 0070 C245           MOV R5,R9
0194 0072 0AC9           SLA R9,12
0195 0074 A189           A R9,R6         MOVE HEX DIGIT ACROSS
0196 0076 0945           SRL R5,4
0197 0078 C244           MOV R4,R9
0198 007A 0AC9           SLA R9,12
0199 007C A149           A R9,R5         MOVE HEX DIGIT ACROSS
0200 007E 0844           SRA R4,4
0201 0080 0587           INC R7          DONE, INCREMENT EXPONENT
0202 0082 060A           DEC R10         COUNT SHIFTS
0203 0084 16E8             JNE FAD2A     OK
0204                     *
0205 0086 C000  FAD3A    MOV R0,R0       TEMP=0, NEED NEGATION?
0206 0088 1553             JGT FNRM3     N, FPAC OK
0207 008A 06A0           BL @FADN        Y, NEGATE FPAC
     008C 00B8'
0208 008E 0223           AI R3,>80       ADD SIGN TO EXPONENT
     0090 0080
0209 0092 104E           JMP FNRM3
0210                     *
0211 0094 06A0  FAD4     BL @FADDI       ADD
     0096 02F2'
0212 0098 1106             JLT FAD5      NEGATIVE
0213 009A 1509             JGT FAD6      POSITIVE
0214 009C C041           MOV R1,R1
0215 009E 1607             JNE FAD6      NON-ZERO
0216 00A0 C082           MOV R2,R2
0217 00A2 1605             JNE FAD6      NON-ZERO
0218 00A4 1047           JMP FNRM4       ZERO, RTWP
0219                     *
0220 00A6 06A0  FAD5     BL @FADN        NEGATE FPAC
     00A8 00B8'
0221 00AA 0263           ORI R3,>80      ADD SIGN BIT
```

```
      00AC 0080
0222                  *
0223 00AE D000  FAD6    MOVB R0,R0        CHECK ADDITION OVERFLOW
0224 00B0 132E            JEQ FNRM2       NUMBER OK
0225 00B2 06A0            BL @FARS        SHIFT NUMBER
     00B4 02D2'
0226 00B6 102B            JMP FNRM2       NORMALIZE
0227                  *
0228 00B8 0502  FADN    NEG R2            NEGATE FPAC
0229 00BA 1604            JNE FADN1
0230 00BC 0501          NEG R1
0231 00BE 1603            JNE FADN2
0232 00C0 0500          NEG R0
0233 00C2 045B          RT
0234                  *
0235 00C4 0541  FADN1   INV R1
0236 00C6 0540  FADN2   INV R0
0237 00C8 045B          RT                RETURN
```

```
0240                    * ABSTRACT:
0241                    *
0242                    *       GET ABSOLUTE VALUE OF FPAC.
0243                    *       NEGATE FPAC.
0244                    *
0245                    * CALL SEQUENCE:
0246                    *
0247                    *       XOP XX,9        ;NEGATE FPAC
0248                    *
0249                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0250                    *
0251                    * LOCAL DATA:  (NONE)
0252                    *
0253                    * ENTRY POINT:
0254                    *
0255                    *       DEF FABS
0256                    *
0257                    *FABS   ANDI R0,>7FFF   CLEAR SIGN BIT
0258                    *       RTWP            RETURN
0259                    *
0260                    * ENTRY POINT:
0261                    *
0262                            DEF FNEG
0263                    *
0264 00CA C000  FNEG    MOV R0,R0       ZERO?
0265 00CC 1302            JEQ FNEG1     Y, LEAVE TRUE ZERO
0266 00CE 0220          AI R0,>8000     N, COMPLEMENT SIGN BIT
     00D0 8000
0267 00D2 0380  FNEG1   RTWP            RETURN
```

```
0270                    * ABSTRACT:
0271                    *
0272                    *       IF FPAC IS NOT NORMALIZED, AN ATTEMPT WILL BE
0273                    *       MADE TO NORMALIZE.  IN GENERAL, FLOAT WORKS ON
0274                    *       R0,R1,R2 AND HENCE IT CAN BE USED TO FLOAT
0275                    *       OTHER SETS OF NUMBERS THAN FPAC.
0276                    *
0277                    * CALLING SEQUENCE:
0278                    *
0279                    *       XOP XX,10       WHERE XX IS NOT USED
0280                    *
0281                    * EXCEPTIONS AND CONDITIONS:
0282                    *
0283                    *       IF R0 IS NON-ZERO, THE ROUTINE WILL ABORT.
0284                    *
0285                    * EXTERNAL ROUTING LIST:
0286                    *
0287                    *       FNRM            ;NORMALIZE
0288                    *       FCLR            ;CLEAR FPAC
0289                    *
0290                    * LOCAL DATA:  (NONE)
0291                    *
0292                    * ENTRY POINT:
0293                    *
0294                            DEF FLOAT
0295                    *
0296 00D4 C000  FLOAT   MOV R0,R0       ALREADY FLOATING?
0297 00D6 162E            JNE FNRM4     Y, RTWP
0298 00D8 0203          LI R3,>44       N, GET EXPONENT
     00DA 0044
0299 00DC 04C2          CLR R2          CLEAR LOW
0300 00DE C041          MOV R1,R1       CHECK SIGN
0301 00E0 1504            JGT FLOAT1    POSITIVE
0302 00E2 132C            JEQ FCLRP     ZERO
0303 00E4 0501          NEG R1          NEGATIVE
0304 00E6 0203          LI R3,>C4       NEGATE EXPONENT
     00E8 00C4
0305                    *
0306 00EA D001  FLOAT1  MOVB R1,R0       SHIFT 1 BYTE LEFT
0307 00EC 1303            JEQ FLOAT2    SHIFT 2 BYTES
0308 00EE 06C0          SWPB R0
0309 00F0 0A81          SLA R1,8
0310 00F2 100D          JMP FNRM2
0311                    *
0312 00F4 C001  FLOAT2  MOV R1,R0       SHIFT 2 BYTES
0313 00F6 04C1          CLR R1
0314 00F8 0643          DECT R3         ADJUST EXPONENT
0315 00FA 1009          JMP FNRM2
```

```
0318                    * ABSTRACT:
0319                    *
0320                    *       FPAC IS NORMALIZED SUCH THAT THERE EXISTS
0321                    *       A NON-ZERO HEX DIGIT IN BITS 8-11 OF THE 1ST
0322                    *       WORD OF FPAC.
0323                    *
0324                    * CALL SEQUENCE:
0325                    *
0326                    *       XOP XX,7        WHERE XX IS NOT USED
0327                    *
0328                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0329                    *
0330                    * EXTERNAL ROUTINE LIST:
0331                    *
0332                    *       FCLR            ;CLEAR FPAC
0333                    *
0334                    * LOCAL DATA:  (NONE)
0335                    *
0336                    * ENTRY POINT:
0337                    *
0338                            DEF FNRM
0339                    *
0340 00FC 04C3  FNRM    CLR R3          CLEAR R3
0341 00FE D0C0          MOVB R0,R3      GET EXPONENT & SIGN
0342 0100 6003          S R3,R0         REMOVE FROM NUMBER
0343 0102 1604            JNE FNRM1     LOOK FOR ZERO
0344 0104 C041          MOV R1,R1
0345 0106 1602            JNE FNRM1
0346 0108 C082          MOV R2,R2
0347 010A 1314            JEQ FNRM4     ZERO, RETURN
0348                    *
0349 010C 06C3  FNRM1   SWPB R3         READY FOR DECREMENTING
0350                    *
0351 010E 2420  FNRM2   CZC @CF0,R0     NORMALIZED?
     0110 0344'
0352 0112 160E            JNE FNRM3     Y
0353 0114 24E0          CZC @C7F,R3     N, EXPONENT=ZERO?
     0116 0342'
0354 0118 1311            JEQ FCLRP     Y, CANNOT NORMALIZE
0355 011A 0603          DEC R3          N, OK TO DECREMENT
0356 011C 0A40          SLA R0,4        SHIFT R0,R1,R2 4 BITS LEFT
0357 011E C241          MOV R1,R9
0358 0120 09C9          SRL R9,12
0359 0122 A009          A R9,R0         MOVE 1ST HEX DIGIT ACROSS
0360 0124 0A41          SLA R1,4
0361 0126 C242          MOV R2,R9
0362 0128 09C9          SRL R9,12
0363 012A A049          A R9,R1         MOVE 2ND HEX DIGIT ACROSS
0364 012C 0A42          SLA R2,4
0365 012E 10EF          JMP FNRM2
0366                    *
0367 0130 06C3  FNRM3   SWPB R3         READY EXPONENT
0368 0132 D003          MOVB R3,R0      MOVE INTO FPAC
0369 0134 0380  FNRM4   RTWP            RETURN
```

```
0372                    * ABSTRACT:
0373                    *
0374                    *       MULTIPLY FPAC BY FLOATING POINT NUMBER
0375                    *       POINTED TO BY R11.
0376                    *
0377                    * CALL SEQUENCE:
0378                    *
0379                    *       XOP XX,4
0380                    *
0381                    * EXCEPTIONS AND CONDITIONS:
0382                    *
0383                    *       ERROR 29 CAN RESULT FROM OVERFLOW
0384                    *
0385                    * EXTERNAL ROUTINE LIST:
0386                    *
0387                    *       FCLR            ;CLEAR FPAC
0388                    *       FPFX            ;FIX EXPONENTS
0389                    *       FAD6            ;ADD AND NORMALIZE
0390                    *
0391                    * LOCAL DATA:  (NONE)
0392                    *
0393                    * ENTRY POINT:
0394                    *
0395                            DEF FMD
0396                    *
0397 0136 C000  FMD     MOV R0,R0       ;FPAC=0?
0398 0138 1374            JEQ FCLR      ;Y
0399 013A C138          MOV *R11+,R4    ;FPAC X 0?
0400 013C 1372  FCLRP     JEQ FCLR      ;Y
0401 013E C17B          MOV *R11+,R5    ;N, LOAD R4,R5,R6
0402 0140 C19B          MOV *R11,R6
0403 0142 06A0          BL @FPFX        FIX EXPONENTS
     0144 022A'
0404 0146 A0C7            A R7,R3
0405 0148 FFC0            DATA >FFC0
0406                    *
0407                    *DO MULTIPLICATION
0408                    *
0409                    *                       R4,R5,R6
0410                    *                       R0,R1,R2
0411                    *                       =======
0412                    *                          XX XX        A       R2 X R6
0413                    *                       XX XX           B       R2 X R5
0414                    *                    XX XX              C       R2 X R4
0415                    *                       XX XX           D       R1 X R6
0416                    *                    XX XX              E       R1 X R5
0417                    *                 XX XX                 F       R1 X R4
0418                    *                    XX XX              G       R0 X R6
0419                    *                 XX XX                 H       R0 X R5
0420                    *              XX XX                    I       R0 X R4
0421                    *              ----------------
0422                    *              R7,R8,R9,R10
0423                    *
0424 014A C286          MOV R6,R10
0425 014C 3A82          MPY R2,R10      R10,R11 (A)
0426 014E C206          MOV R6,R8
0427 0150 3A01          MPY R1,R8       R8,R9 (D)
0428 0152 3980          MPY R0,R6       R6,R7 (G)
0429                    *
0430 0154 A289          A R9,R10        SUM PARTIAL PRODUCTS (A,D,G)
```

```
0431 0156 1701            JNC $+4
0432 0158 0588            INC R8
0433 015A A207          A R7,R8
0434 015C 1701            JNC $+4
0435 015E 0586            INC R6
0436 0160 C248          MOV R8,R9
0437 0162 C206          MOV R6,R8       R8,R9,R10 = A+D+G
0438              *
0439 0164 C185          MOV R5,R6
0440 0166 3982          MPY R2,R6       R6,R7 (B)
0441 0168 A287          A R7,R10        SUM
0442 016A 1701            JNC $+4
0443 016C 0586            INC R6
0444 016E A246          A R6,R9
0445 0170 1701            JNC $+4
0446 0172 0588            INC R8        (NO CARRY OUT)
0447 0174 C289          MOV R9,R10
0448 0176 C248          MOV R8,R9
0449 0178 C1C5          MOV R5,R7
0450 017A 39C1          MPY R1,R7       R7,R8 (E)
0451 017C 3940          MPY R0,R5       R5,R6 (H)
0452              *
0453 017E A288          A R8,R10        FINISH PARTIAL SUMS B,E,H
0454 0180 04C8          CLR R8
0455 0182 1701            JNC $+4
0456 0184 0587            INC R7
0457 0186 A247          A R7,R9
0458 0188 1701            JNC $+4
0459 018A 0588            INC R8
0460 018C A246          A R6,R9
0461 018E 1701            JNC $+4
0462 0190 0588            INC R8
0463 0192 04C7          CLR R7
0464 0194 A205          A R5,R8
0465 0196 1701            JNC $+4
0466 0198 0587            INC R7        R7,R8,R9,R10 = A+B+D+E+G+H
0467 019A C144          MOV R4,R5
0468 019C 3942          MPY R2,R5       R5,R6 (C)
0469 019E A286          A R6,R10        SUM
0470 01A0 1705            JNC FMD4
0471 01A2 0589            INC R9
0472 01A4 1703            JNC FMD4
0473 01A6 0588            INC R8
0474 01A8 1701            JNC FMD4
0475 01AA 0587            INC R7
0476 01AC A245  FMD4    A R5,R9
0477 01AE 1703            JNC FMD5
0478 01B0 0588            INC R8
0479 01B2 1701            JNC FMD5
0480 01B4 0587            INC R7
0481              *
0482 01B6 C144  FMD5    MOV R4,R5
0483 01B8 3941          MPY R1,R5       R5,R6 (F)
0484 01BA A246          A R6,R9         SUM
0485 01BC 1703            JNC FMD6
0486 01BE 0588            INC R8
0487 01C0 1701            JNC FMD6
0488 01C2 0587            INC R7
0489 01C4 A205  FMD6    A R5,R8
0490 01C6 1701            JNC $+4
```

```
0491 01C8 0587            INC R7
0492 01CA 3900          MPY R0,R4       R4,R5 (I)
0493 01CC A205          A R5,R8
0494 01CE 1701            JNC $+4
0495 01D0 0587            INC R7
0496 01D2 A1C4          A R4,R7         PRODUCT COMPLETE
0497 01D4 D1C8          MOVB R8,R7      SHIFT 1 BYTE LEFT
0498 01D6 06C7          SWPB R7
0499 01D8 D209          MOVB R9,R8
0500 01DA 06C8          SWPB R8
0501 01DC D24A          MOVB R10,R9
0502 01DE 06C9          SWPB R9
0503 01E0 06CA          SWPB R10
```

```
0505                    *MULTIPLY AND DIVIDE ENTRY
0506                    *
0507                    *     R10 HAS GUARD BITS FOR NORMALIZING & ROUNDING
0508                    *
0509 01E2 C007  FMD7      MOV R7,R0         MOVE INTO FPAC
0510 01E4 C048            MOV R8,R1
0511 01E6 C089            MOV R9,R2
0512                    *
0513 01E8 2420            CZC @CF0,R0       NORMALIZED?
     01EA 0344'
0514 01EC 1611              JNE FMD9        Y, ROUND
0515 01EE 24E0            CZC @C7F,R3       N, EXP=0?
     01F0 0342'
0516 01F2 1317              JEQ FCLR        Y, CANNOT NORMALIZE
0517 01F4 0603            DEC R3            N, OK TO DECREMENT
0518 01F6 0A40            SLA R0,4          SHIFT R0,R1,R2,R10 LEFT 4 BITS
0519 01F8 C241            MOV R1,R9
0520 01FA 09C9            SRL R9,12
0521 01FC A009            A R9,R0           MOVE 1ST HEX DIGIT ACROSS
0522 01FE 0A41            SLA R1,4
0523 0200 C242            MOV R2,R9
0524 0202 09C9            SRL R9,12
0525 0204 A049            A R9,R1           MOVE 2ND HEX DIGIT ACROSS
0526 0206 0A42            SLA R2,4
0527 0208 C24A            MOV R10,R9
0528 020A 09C9            SRL R9,12
0529 020C A089            A R9,R2           MOVE 3RD HEX DIGIT ACROSS
0530 020E 0A4A            SLA R10,4
0531                    *
0532                    *DONE, ROUND USING R10
0533                    *
0534 0210 0A1A  FMD9      SLA R10,1         GET GUARD BIT
0535 0212 178E              JNC FNRM3       IF ZERO, DONE
0536 0214 0582            INC R2            ELSE ROUND
0537 0216 178C              JNC FNRM3
0538 0218 0581            INC R1
0539 021A 178A              JNC FNRM3
0540 021C 0580            INC R0            IF THIS FAR, POSSIBILITY
0541 021E 0460            B @FAD6             OF NORMALIZATION
     0220 00AE'
```

```
0544                * ABSTRACT:
0545                *
0546                *       SET FPAC TO TRUE ZERO.
0547                *
0548                * CALLING SEQUENCE:
0549                *
0550                *       XOP XX,8        WHERE XX IS NOT USED
0551                *
0552                * EXCEPTIONS AND CONDITIONS:  (NONE)
0553                *
0554                * EXTERNAL ROUTINE LIST:  (NONE)
0555                *
0556                * LOCAL DATA:  (NONE)
0557                *
0558                * ENTRY POINT:
0559                *
0560                        DEF FCLR
0561                *
0562 0222 04C0   FCLR   CLR R0          CLEAR FPAC
0563 0224 04C1          CLR R1
0564 0226 04C2          CLR R2
0565 0228 0380          RTWP
```

```
0568            * ABSTRACT:
0569            *
0570            *       DIVIDE FPAC BY THE FLOATING POINT NUMBER POINTED
0571            *       TO BY R11.
0572            *
0573            *       FPFX REMOVES EXPONENTS FROM R0,R1,R2 AND
0574            *       R4,R5,R6, UNBIASES THEM, PERFORMS MULTIPLICATION
0575            *       OR DIVISION (ADD OR SUBTRACT) AND THEN
0576            *       RE-BIASES THEM.
0577            *
0578            * CALL SEQUENCE:
0579            *
0580            *       XOP XX,5
0581            *       BL @FPFX
0582            *
0583            * EXCEPTIONS AND CONDITIONS:
0584            *
0585            *       ERROR 28 WILL RESULT FROM DIVISION BY ZERO
0586            *       ERROR 29 WILL RESULT FROM OVERFLOW
0587            *
0588            * EXTERNAL ROUTINE LIST:
0589            *
0590            *       FCLR            ;CLEAR
0591            *       FMD7            ;FINISH BY NORMALIZING
0592            *
0593            * LOCAL DATA:  (NONE)
```

```
0595                    *FIX EXPONENTS
0596                    *
0597                    *       BL @FPFX
0598                    *         BIAS INSTRUCTION
0599                    *         REBIAS #
0600                    *
0601 022A D0C0  FPFX    MOVB R0,R3      GET EXPONENTS
0602 022C D1C4          MOVB R4,R7
0603 022E 7000          SB R0,R0        ;REMOVE FROM NUMBERS
0604 0230 7104          SB R4,R4
0605 0232 C207          MOV R7,R8       GET SIGN
0606 0234 2A03          XOR R3,R8       +-,-+ = NEG  ++,-- = POS
0607 0236 06C3          SWPB R3         REMOV SIGN BITS
0608 0238 0243          ANDI R3,>7F
     023A 007F
0609 023C 06C7          SWPB R7
0610 023E 0247          ANDI R7,>7F
     0240 007F
0611 0242 04BB          X *R11+         DO ADD FOR MUL, SUB FOR DIV
0612 0244 A0FB          A *R11+,R3      BIAS
0613 0246 24E0          CZC @CFF80,R3   CHECK FOR OVERFLOW
     0248 0346'
0614 024A 1642            JNE ERR29     PROBLEM
0615 024C 0A18          SLA R8,1        ADD SIGN
0616 024E 1702            JNC $+6
0617 0250 0223            AI R3,>80     NEGATIVE
     0252 0080
0618 0254 045B          RT              RETURN
0619                    *
0620                    *FLOATING POINT DIVIDE
0621                    *
0622                    * ENTRY POINT:
0623                    *
0624                          DEF FDD
0625                    *
0626 0256 C000  FDD     MOV R0,R0       ;FPAC=ZERO?
0627 0258 13E4            JEQ FCLR      ;Y
0628 025A C13B          MOV *R11+,R4    ;FPAC / 0?
0629 025C 1338            JEQ FDDER     ;Y, DIVISION BY ZERO
0630 025E C17B          MOV *R11+,R5    ;N, LOAD R4,R5,R6
0631 0260 C19B          MOV *R11,R6
0632 0262 06A0          BL @FPFX        FIX EXPONENTS
     0264 022A'
0633 0266 60C7            S R7,R3       SUBTRACT EXPONENTS
0634 0268 0040            DATA >40      ADD >40 TO BIAS
0635 026A 8100          C R0,R4         CHECK FOR PROPER FRACTION
0636 026C 110A            JLT FDD2      OK
0637 026E 0583          INC R3          IMPROPER, INCREMENT EXPONENT
0638 0270 0A44          SLA R4,4        SHIFT R4,R5,R6 LEFT 4 BITS
0639 0272 C245          MOV R5,R9
0640 0274 09C9          SRL R9,12
0641 0276 A109          A R9,R4         MOVE 1ST HEX DIGIT ACROSS
0642 0278 0A45          SLA R5,4
0643 027A C246          MOV R6,R9
0644 027C 09C9          SRL R9,12
0645 027E A149          A R9,R5         MOVE 2ND HEX DIGIT ACROSS
0646 0280 0A46          SLA R6,4
0647                    *
0648 0282 04C7  FDD2    CLR R7          Y, CLEAR QUOTIENT
0649 0284 04C8          CLR R8
```

```
0650 0286 04C9          CLR R9
0651 0288 020A          LI R10,40       DO 40 TIMES
     028A 0028
0652              *
0653 028C 0A10  FDD3    SLA R0,1        SHIFT LEFT R0,R1,R2,R7,R8,R9
0654 028E 0A11          SLA R1,1
0655 0290 1701            JNC $+4
0656 0292 0580            INC R0
0657 0294 0A12          SLA R2,1
0658 0296 1701            JNC $+4
0659 0298 0581            INC R1
0660 029A 0A17          SLA R7,1
0661 029C 0A18          SLA R8,1
0662 029E 1701            JNC $+4
0663 02A0 0587            INC R7
0664 02A2 0A19          SLA R9,1
0665 02A4 1701            JNC $+4
0666 02A6 0588            INC R8
0667 02A8 8100          C R0,R4         IS R0,R1,R2<=R4,R5,R6?
0668 02AA 1A09            JL FDD5       Y
0669 02AC 1B05            JH FDD4       N
0670 02AE 8141          C R1,R5         MAYBE
0671 02B0 1A06            JL FDD5       Y
0672 02B2 1B02            JH FDD4       N
0673 02B4 8182          C R2,R6         MAYBE
0674 02B6 1A03            JL FDD5       Y
0675 02B8 06A0  FDD4    BL @FSUBI       N, R0,R1,R2=R0,R1,R2-R4,R5,R6
     02BA 0306'
0676 02BC 0589          INC R9          ENTER BIT
0677              *
0678 02BE 060A  FDD5    DEC R10         DONE?
0679 02C0 16E5            JNE FDD3      N, LOOP AGAIN
0680 02C2 0A10          SLA R0,1        Y, CHECK FOR ROUNDING
0681 02C4 8100          C R0,R4         2*REMAINDER < DIVISOR?
0682 02C6 1A8D            JL FMD7       Y, NO NEED TO ROUND
0683 02C8 020A          LI R10,>8000    ROUND
     02CA 8000
0684 02CC 108A          JMP FMD7        DONE
0685              *
0686 02CE 2F9C  FDDER   DATA ERROR+28   ;DIVISION BY ZERO
0687              *
0688 02D0 2F9D  ERR29   DATA ERROR+29   ;FP ERROR
```

```
0691                * ABSTRACT:
0692                *
0693                *       SHIFT R0,R1,R2 RIGHT 1 HEX DIGIT WHILE
0694                *       UPDATEING EXPONENT IN R3
0695                *
0696                * CALLING SEQUENCE:
0697                *
0698                *       BL @FARS
0699                *
0700                * EXCEPTIONS AND CONDITIONS:
0701                *
0702                *       ERROR 29 WILL RESULT ON OVERFLOW
0703                *
0704                * EXTERNAL ROUTINE LIST:
0705                *
0706                *       ERR29           ;ERROR 29
0707                *
0708                * LOCAL DATA:  (NONE)
0709                *
0710                * ENTRY POINT:
0711                *
0712 02D2 0942   FARS    SRL R2,4          SHIFT R0,R1,R2 4 BITS RIGHT
0713 02D4 C241           MOV R1,R9
0714 02D6 0AC9           SLA R9,12
0715 02D8 A089           A R9,R2           MOVE 1ST HEX DIGIT ACROSS
0716 02DA 0941           SRL R1,4
0717 02DC C240           MOV R0,R9
0718 02DE 0AC9           SLA R9,12
0719 02E0 A049           A R9,R1           MOVE 2ND HEX DIGIT ACROSS
0720 02E2 0840           SRA R0,4
0721 02E4 C243           MOV R3,R9
0722 02E6 0583           INC R3            INCREMENT EXPONENT
0723 02E8 2A43           XOR R3,R9         WATCH FOR SIGN CHANGE
0724 02EA 0249           ANDI R9,>80       SIGN CHANGE?
     02EC 0080
0725 02EE 16F0             JNE ERR29       Y, OVERFLOW
0726 02F0 045B           RT
```

```
0729                    * ABSTRACT:
0730                    *
0731                    *       ADD OR SUBTRACT R4,R5,R6 FROM R0,R1,R2.
0732                    *
0733                    * CALL SEQUENCE:
0734                    *
0735                    *       BL @FADDI
0736                    *       BL @FSUBI
0737                    *
0738                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0739                    *
0740                    * EXTERNAL ROUTINE LIST:  (NONE)
0741                    *
0742                    * LOCAL DATA:  (NONE)
0743                    *
0744                    * ENTRY POINT:
0745                    *
0746                            DEF FADDI
0747                    *
0748 02F2 A086  FADDI   A R6,R2         R0,R1,R2=R0,R1,R2+R4,R5,R6
0749 02F4 1703            JNC $+8
0750 02F6 0581            INC R1
0751 02F8 1701            JNC $+4
0752 02FA 0580            INC R0
0753 02FC A045          A R5,R1
0754 02FE 1701            JNC $+4
0755 0300 0580            INC R0
0756 0302 A004          A R4,R0         ADDITION COMPLETE
0757 0304 0458          RT
0758                    *
0759                    * ENTRY POINT:
0760                    *
0761                            DEF FSUBI
0762                    *
0763 0306 6086  FSUBI    S R6,R2         R0,R1,R2=R0,R1,R2-R4,R5,R6
0764 0308 1803            JOC $+8
0765 030A 0601            DEC R1
0766 030C 1801            JOC $+4
0767 030E 0600            DEC R0
0768 0310 6045          S R5,R1
0769 0312 1801            JOC $+4
0770 0314 0600            DEC R0
0771 0316 6004          S R4,R0         SUBTRACTION COMPLETE
0772 0318 0458          RT
```

```
0775                    * ABSTRACT:
0776                    *
0777                    *       FPAC WILL BE SHIFTED LEFT OR RIGHT IN
0778                    *       ORDER TO MAKE THE EXPONENT AGREE WITH THE
0779                    *       SCALE FACTOR POINTED TO BY R11.  THIS
0780                    *       OPERATION IS ESSENTIALLY THE OPPOSITE
0781                    *       OF NORMALIZATION.
0782                    *
0783                    *       SCALING TO >4600 PLACES THE DECIMAL POINT
0784                    *       BETWEEN THE SECOND AND THIRD WORD OF THE FLOATING
0785                    *       POINT NUMBER.  SCALING TO >4A00 PLACES THE DECIMAL
0786                    *       POINT AFTER THE THIRD WORD.
0787                    *
0788                    *       HENCE, TO INTEGERIZE A FLOATING POINT NUMBER IN FPAC
0789                    *       ONE SCALES TO >4A00 AND THEN NORMALIZES.
0790                    *
0791                    * CALLING SEQUENCE:
0792                    *
0793                    *       XOP XX,6
0794                    *
0795                    * EXCEPTIONS AND CONDITIONS:
0796                    *
0797                    *       ERROR 29 WILL RESULT FROM A LEFT SHIFT
0798                    *
0799                    * EXTERNAL ROUTINE LIST:
0800                    *
0801                    *       ERR29           ;ERROR 29
0802                    *
0803                    * LOCAL DATA:  (NONE)
0804                    *
0805                    * ENTRY POINT:
0806                    *
0807                            DEF FSCL
0808                    *
0809                    *
0810 031A C11B     FSCL     MOV *R11,R4     GET SCALE FACTOR
0811 031C 04C3              CLR R3
0812 031E C200              MOV R0,R8       SAVE SIGN
0813 0320 D003              MOVB R3,R0      ZERO EXPONENT
0814 0322 D1C8              MOVB R8,R7      GET EXPONENT
0815 0324 0247              ANDI R7,>7F00   GET EXPONENT
     0326 7F00
0816 0328 61C4              S R4,R7         GET DIFFERENCE
0817 032A 130A                JEQ FSCL2     ALREADY SCALED
0818 032C 0887              SRA R7,8        RIGHT JUSTIFY
0819 032E 15D0                JGT ERR29     SHIFT LEFT, FP ERROR?
0820                    *
0821 0330 06A0     FSCL1    BL @FARS        SHIFT RIGHT
     0332 02D2'
0822 0334 0587              INC R7          DONE?
0823 0336 16FC                JNE FSCL1     N
0824 0338 D008              MOVB R8,R0      RESTORE SIGN BIT
0825 033A 0240              ANDI R0,>80FF   ;MASK EXPONENT
     033C 80FF
0826 033E B004              AB R4,R0
0827 0340 0380     FSCL2    RTWP
0828                    *
0829 0342 007F     C7F      DATA >007F
0830 0344 00F0     CF0      DATA >00F0
0831 0346 FF80     CFF80    DATA >FF80
```

```
 0832                    END
NO ERRORS,     NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES								
$		0348'		0431	0434	0442	0445	0455	0458	0461	0465	0490
				0494	0616	0655	0658	0662	0665	0749	0751	0754
				0764	0766	0769						
C7F		0342'	0829	0353	0515							
CF0		0344'	0830	0351	0513							
CFF80		0346'	0831	0613								
ERR29		02D0'	0688	0614	0725	0819						
ERROR		2F80	0051	0052	0686	0688						
ERROR2		2FA0	0052									
FAD	D	001A'	0148	0146								
FAD0		001E'	0150	0142								
FAD0A		0026'	0155	0151								
FAD1		003A'	0165	0162								
FAD1A		004C'	0174	0169								
FAD1B		004E'	0175	0171								
FAD2		0050'	0177	0167	0173							
FAD2A		0056'	0180	0185	0203							
FAD3		006E'	0192	0182								
FAD3A		0086'	0205	0190								
FAD4		0094'	0211	0181								
FAD5		00A6'	0220	0212								
FAD6		00AE'	0223	0213	0215	0217	0541					
FADDI	D	02F2'	0748	0211	0746							
FADN		00B8'	0228	0163	0207	0220						
FADN1		00C4'	0235	0229								
FADN2		00C6'	0236	0231								
FARS		02D2'	0712	0183	0225	0821						
FCLR	D	0222'	0562	0398	0400	0516	0560	0627				
FCLRP		013C'	0400	0302	0354							
FDD	D	0256'	0626	0624								
FDD2		0282'	0648	0636								
FDD3		028C'	0653	0679								
FDD4		02B8'	0675	0669	0672							
FDD5		02BE'	0678	0668	0671	0674						
FDDER		02CE'	0686	0629								
FLDD	D	0008'	0104	0102								
FLDD1		000A'	0105	0153								
FLDD2		000E'	0107	0140	0149							
FLOAT	D	00D4'	0296	0294								
FLOAT1		00EA'	0306	0301								
FLOAT2		00F4'	0312	0307								
FMD	D	0136'	0397	0395								
FMD4		01AC'	0476	0470	0472	0474						
FMD5		01B6'	0482	0477	0479							
FMD6		01C4'	0489	0485	0487							
FMD7		01E2'	0509	0682	0684							
FMD9		0210'	0534	0514								
FNEG	D	00CA'	0264	0262								
FNEG1		00D2'	0267	0265								
FNRM	D	00FC'	0340	0338								
FNRM1		010C'	0349	0343	0345							
FNRM2		010E'	0351	0224	0226	0310	0315	0365				
FNRM3		0130'	0367	0206	0209	0352	0535	0537	0539			
FNRM4		0134'	0369	0218	0297	0347						
FPFX		022A'	0601	0403	0632							
FSCL	D	031A'	0810	0807								
FSCL1		0330'	0821	0823								
FSCL2		0340'	0827	0817								
FSD	D	0010'	0139	0135								
FSRD	D	0000'	0079	0077								

LABEL		VALUE	DEFN	REFERENCES								
FSUBI	D	0306'	0763	0675	0761							
R0		0000		0079	0104	0150	0150	0152	0157	0159	0159	0186
				0205	0205	0223	0223	0232	0236	0264	0264	0266
				0296	0296	0306	0308	0312	0341	0342	0351	0356
				0359	0368	0397	0397	0428	0451	0492	0509	0513
				0518	0521	0540	0562	0601	0603	0603	0626	0626
				0635	0653	0656	0667	0680	0681	0717	0720	0752
				0755	0756	0767	0770	0771	0812	0813	0824	0825
				0826								
R1		0001		0080	0105	0187	0214	0214	0230	0235	0300	0300
				0303	0306	0309	0312	0313	0344	0344	0357	0360
				0363	0427	0450	0483	0510	0519	0522	0525	0538
				0563	0654	0659	0670	0713	0716	0719	0750	0753
				0765	0768							
R10		000A		0177	0184	0202	0424	0425	0430	0441	0447	0453
				0469	0501	0503	0527	0530	0534	0651	0678	0683
R11		000B		0079	0080	0081	0104	0105	0106	0139	0148	0155
				0156	0399	0401	0402	0611	0612	0628	0630	0631
				0810								
R2		0002		0081	0106	0188	0216	0216	0228	0299	0346	0346
				0361	0364	0425	0440	0468	0511	0523	0526	0529
				0536	0564	0657	0673	0712	0715	0748	0763	
R3		0003		0157	0161	0165	0180	0189	0208	0221	0298	0304
				0314	0340	0341	0342	0349	0353	0355	0367	0368
				0404	0515	0517	0601	0606	0607	0608	0612	0613
				0617	0633	0637	0721	0722	0723	0811	0813	
R4		0004		0139	0141	0148	0152	0158	0160	0160	0172	0175
				0186	0197	0200	0399	0467	0482	0492	0496	0602
				0604	0604	0628	0635	0638	0641	0667	0681	0756
				0771	0810	0816	0826					
R5		0005		0155	0170	0174	0187	0193	0196	0199	0401	0439
				0449	0451	0464	0467	0468	0476	0482	0483	0489
				0493	0630	0639	0642	0645	0670	0753	0768	
R6		0006		0156	0168	0188	0192	0195	0402	0424	0426	0428
				0435	0437	0439	0440	0443	0444	0460	0469	0484
				0631	0643	0646	0673	0748	0763			
R7		0007		0158	0166	0178	0180	0189	0201	0404	0433	0441
				0449	0450	0456	0457	0463	0466	0475	0480	0488
				0491	0495	0496	0497	0498	0509	0602	0605	0609
				0610	0633	0648	0660	0663	0814	0815	0816	0818
				0822								
R8		0008		0426	0427	0432	0433	0436	0437	0446	0448	0453
				0454	0459	0462	0464	0473	0478	0486	0489	0493
				0497	0499	0500	0510	0605	0606	0615	0649	0661
				0666	0812	0814	0824					
R9		0009		0193	0194	0195	0197	0198	0199	0357	0358	0359
				0361	0362	0363	0430	0436	0444	0447	0448	0457
				0460	0471	0476	0484	0499	0501	0502	0511	0519
				0520	0521	0523	0524	0525	0527	0528	0529	0639
				0640	0641	0643	0644	0645	0650	0664	0676	0713
				0714	0715	0717	0718	0719	0721	0723	0724	

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.RNDF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.RNDF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.RNDF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'RNDF'
0002          *
0003          * REVISION: 12/31/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       RANDG             ;RANDOM NUMBER GENERATOR
0009          *       RAND              ;GET FLOATING POINT RANDOM NUMBER
0010          *
0011          *
0012          * COPY: (NONE)
0013          *
0014          * MACRO: (NONE)
0015          *
0016          * EXTERNAL ROUTINES:
0017          *
0018                REF FNRM                ;FIXES ARGUMENT
0019                REF RANDS                 ;RANDOM SEED
0020          *
0021          *
0022          * EXTERNAL DATA:
0023          *
0024                REF FPWP                  ;FLOATING POINT WORKSPACE PO
0025          *
0026          * MODULE EQUATES (NONE)
0027          *
0028          *
0029          * MODULE VARIABLES AND CONSTANTS:
0030          *
```

```
0033                * ABSTRACT:
0034                *
0035                *       RETURNS A 16 BIT RANDOM NUMBER.  THE SEED
0036                *       IS MAINTAINED IN RANDS AND IS UPDATED
0037                *       ON EACH CALL.
0038                *
0039                * GENERATE PSEUDO RANDOM SEQUENCE OF INTEGERS
0040                *
0041                *       LINEAR CONGRUENTIAL SEQUENCE:
0042                *       X[N+1] = (X[N] * A + C) MOD 2↑16
0043                *
0044                * USE MOST SIGNIFICANT K-BITS IF <16 ARE REQUIRED
0045                *
0046                *
0047                * CALLING SEQUENCE:
0048                *
0049                *       BL @RANDG
0050                *
0051                *       OUT R0 = 16 BIT RANDOM NUMBER
0052                *
0053                * EXCEPTIONS AND CONDITIONS:  (NONE)
0054                *
0055                * EXTERNAL ROUTINE LIST:
0056                *
0057                *      (SEE EXTERNAL ROUTINES)
0058                *
0059                * LOCAL DATA:
0060                *
0061                *      (SEE MODULE VARIABLES AND CONSTANTS)
0062                * ENTRY POINT:
0063                *
0064                *       DEF RANDG
0065                *
0066 0000 C020     RANDG    MOV @RANDS,R0    ;GET RANDOM SEED
     0002 0000
0067 0004 C0C0              MOV R0,R3
0068 0006 0AB3              SLA R3,11        ;N*2↑11
0069 0008 A0C0              A R0,R3          ;N*2↑11+N
0070 000A 0A23              SLA R3,2
0071 000C A003              A R3,R0          ;N*2↑11+5N
0072 000E 0220              AI R0,13849      ;(N*A+13849.)  MOD 2↑16
     0010 3619
0073 0012 C800              MOV R0,@RANDS    ;STORE NEW SEED
     0014 0002'
0074 0016 045B              RT
```

```
0077                * ABSTRACT:
0078                *
0079                *       RETURNS A FLOATING POINT RANDOM NUMBER IN FPAC.
0080                *       TWO 16-BIT RANDOM NUMBERS ARE USED TO GENERATE
0081                *       THE THREE WORD FLOATING POINT NUMBER.
0082                *
0083                * CALLING SEQUENCE:
0084                *
0085                *       BLWP @RANDZ
0086                *
0087                *       OUT FPAC = FLOATING POINT RANDOM NUMBER
0088                *
0089                * EXCEPTIONS AND CONDITIONS:  (NONE)
0090                *
0091                * EXTERNAL ROUTINE LIST:
0092                *
0093                *       FNRM            ;NORMALIZE NUMBER
0094                *
0095                * LOCAL DATA:
0096                *
0097                *      (SEE MODULE VARIABLES AND CONSTANTS)
0098                *
0099                * ENTRY POINT:
0100                *
0101                        DEF RANDZ
0102                *
0103 0018 0000     RANDZ    DATA FPWP,RAND
     001A 001C'
0104                *
0105 001C 06A0     RAND     BL @RANDG       ;GET RANDOM #
     001E 0000'
0106 0020 C040              MOV R0,R1
0107 0022 06A0              BL @RANDG
     0024 0000'
0108 0026 C080              MOV R0,R2       ;LOAD LOW
0109 0028 0200              LI R0,>4200     ;ADD EXPONENT
     002A 4200
0110 002C 0460              B @FNRM         ;NORMALIZE
     002E 0000
0111                        END
NO ERRORS,     NO WARNINGS
```

```
FNRM    R  002E'   0018  0110
FPWP    R  0018'   0024  0103
R0         0000          0066  0067  0069  0071  0072  0073  0106  0108  0109
R1         0001          0106
R2         0002          0108
R3         0003          0067  0068  0069  0070  0071
RAND       001C'  0105   0103
RANDG      0000'  0066   0105  0107
RANDS   R  0014'  0019   0066  0073
RANDZ   D  0018'  0103   0101
*
```

```
SOURCE ACCESS NAME=      DBASIC.REL0.BASIC.SOURCE.CVBD
OBJECT ACCESS NAME=      DBASIC.REL0.BASIC.OBJECT.CVBD
LISTING ACCESS NAME=     DBASIC.REL0.BASIC.LIST.CVBD
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'CVBD'
0002          *         08/09/78
0003          *
0004                    DEF CVBD,CVBI,CVBFR
0005                    REF FPAC,FPAC2
0006                    REF CVHD,CVHD01,CVHD15
0007                    REF FFLG
0008                    REF FADDI,FSUBI
0009                    REF CVGCN,B01,B05
0010                    REF CK302,CVBFP
0011          *
0012          *  DEFINE FLOATING POINT XOPS FOR THIS MODULE
0013          *
0014                    DXOP LOADF,0            ;LOAD FPAC
0015                    DXOP STORE,1            ;STORE FPAC
0016                    DXOP FADD,2             ;ADD TO FPAC
0017                    DXOP FSUB,3             ;SUBTRACT FROM FPAC
0018                    DXOP FMUL,4             ;MULTIPLY FPAC
0019                    DXOP FDIV,5             ;DIVIDE FPAC
0020                    DXOP SCALE,6            ;SCALE PFAC
0021                    DXOP NORMAL,7           ;NORMALIZE FPAC
0022                    DXOP CLEAR,8            ;CLEAR FPAC
0023                    DXOP NEGATE,9           ;NEGATE FPAC
0024                    DXOP FLOATF,10          ;FLOAT FPAC
0025          *
0026                    DXOP EVFIX,11           ;EVALUATE AND FIX
0027                    DXOP OUTFP,12           ;OUT FLOATING POINT #
0028                    DXOP OUTINT,13          ;OUT INTEGER
0029    2F80  ERROR     EQU  >2F80              ;XOP XX,14  (ERROR CALL)
0030    2FA0  ERROR2    EQU  ERROR+>20
```

```
0032 0000 2FA0  ERR34   DATA ERROR2,34  ;UNNORMALIZED #
     0002 0022
0033            *
0034            *OUTPUT INTEGER
0035            *
0036 0004 2E00  CVBI    CLEAR 0         ;CLEAR FPAC
0037 0006 C81B          MOV *R11,@FPAC2
     0008 0000
0038 000A 1008          JMP CVBD0
0039            *
0040            *OUTPUT FLOATING POINT #
0041            *
0042 000C 020A  CVBD    LI R10,FPAC     ;GET FPAC ADDRESS
     000E 0000
0043 0010 DEBB          MOVB *R11+,*R10+
0044 0012 DEBB          MOVB *R11+,*R10+
0045 0014 DEBB          MOVB *R11+,*R10+
0046 0016 DEBB          MOVB *R11+,*R10+
0047 0018 DEBB          MOVB *R11+,*R10+
0048 001A D69B          MOVB *R11,*R10
0049            *
0050 001C 2E80  CVBD0   FLOATF 0        ;FLOAT FPAC IF NECESSARY
0051 001E 04CC          CLR R12         ;CLEAR SIGN FLAG
0052 0020 0207          LI R7,CVHD      ;GET HOLD ADR
     0022 0000
0053 0024 7DD7          SB *R7,*R7+     ;CLEAR FIRST BYTE
0054 0026 C060          MOV @FPAC,R1    ;CHECK FOR ZERO
     0028 000E'
0055 002A 1106            JLT CVBD1     ;NEGATIVE
0056 002C 1507            JGT CVBD2     ;POSITIVE
0057 002E DDE0          MOVB @B30,*R7+  ;ZERO, OUT "0
     0030 0211'
0058 0032 020A          LI R10,12       ;SET DIGIT COUNT
     0034 000C
0059 0036 103B          JMP CVBD12
0060            *
0061 0038 2E40  CVBD1   NEGATE 0        ;NEGATE FPAC
0062 003A 070C          SETO R12        ;SET FLAG
0063            *
0064 003C 0241  CVBD2   ANDI R1,>00F0   ;NORMALIZED?
     003E 00F0
0065 0040 13DF            JEQ ERR34     ;N
0066 0042 04CA          CLR R10         ;CLEAR DECIMAL ADJUST COUNTER
0067            *
0068 0044 D060  CVBD3   MOVB @FPAC,R1   GET EXPONENT
     0046 0028'
0069 0048 0981          SRL R1,8
0070 004A 0208          LI R8,>4A       GET EXP=4A (16↑10)
     004C 004A
0071 004E 6201          S R1,R8
0072 0050 1310            JEQ CVBD6     EXP=4A
0073 0052 1105            JLT CVBD4     EXP<4A    (MUL)
0074            *
0075 0054 06A0          BL @CVGCN       FIX
     0056 0000
0076 0058 A288            A R8,R10      UPDATE DECIMAL ADJUSTOR
0077 005A 2D10          FMUL *R0        MULTIPLY
0078 005C 10F3          JMP CVBD3
0079            *
0080 005E 0508  CVBD4   NEG R8          R8=-R8
```

```
0081 0060 06A0          BL @CVGCN
     0062 0056'
0082 0064 6288            S R8,R10      UPDATE DECIMAL ADJUSTOR
0083 0066 2D50          FDIV *R0        DIVIDE
0084 0068 10ED          JMP CVBD3
0085              *
0086 006A 06A0  CVBD5   BL @FADDI       WENT NEG, ADD BACK
     006C 0000
0087 006E 058A          INC R10
0088 0070 1007          JMP CVBD7
0089              *
0090 0072 2C40  CVBD6   STORE R0        LOAD FPAC IN R0,R1,R2
0091 0074 0240          ANDI R0,>00FF   CLEAR EXPONENT
     0076 00FF
0092 0078 0203          LI R3,CVTB0     GET TABLE ADR
     007A 01C0'
0093 007C 0209          LI R9,>30       "0
     007E 0030
0094              *
0095 0080 C133  CVBD7   MOV *R3+,R4     GET 1ST #
0096 0082 C173          MOV *R3+,R5
0097 0084 C1B3          MOV *R3+,R6
0098 0086 06A0          BL @FSUBI       R0,R1,R2=R0,R1,R2-R4,R5,R6
     0088 0000
0099 008A 11EF            JLT CVBD5     WENT NEG, ADD BACK
0100              *
0101 008C 0589  CVBD8   INC R9          COUNT
0102 008E 06A0          BL @FSUBI       ;R0,R1,R2=R0,R1,R2-R4,R5,R6
     0090 0088'
0103 0092 13FC            JEQ CVBD8
0104 0094 15FB            JGT CVBD8
0105 0096 06A0          BL @FADDI       WENT NEGATIVE, ADD BACK
     0098 006C'
0106 009A 06C9          SWPB R9         READY BYTE FOR STORING
0107 009C DDC9          MOVB R9,*R7+    MOVE INTO BUFFER
0108 009E 0209          LI R9,>2F       RELOAD R9
     00A0 002F
0109 00A2 C133          MOV *R3+,R4     GET NEXT CONSTANT
0110 00A4 C173          MOV *R3+,R5
0111 00A6 C1B3          MOV *R3+,R6
0112 00A8 0283          CI R3,CVTB0E    DONE?
     00AA 020E'
0113 00AC 12EF            JLE CVBD8     N
```

```
0115                *CONVERSION DONE, ROUND AND SUPPRESS TRAILING ZERO'S
0116                *
0117                *       R10 = F = .001
0118                *             E = .01
0119                *             D = .1
0120                *             C = 1
0121                *             B = 10
0122                *             A = 100
0123                *
0124 00AE 7DD7      CVBD12  SB *R7,*R7+     OUT NULL
0125 00B0 0287              CI R7,CVHD15    DONE?
     00B2 0000
0126 00B4 1AFC                JL CVBD12     N
0127                *
0128 00B6 C1ED              MOV @14(13),R7  GET OUTPUT BUFFER PTR
     00B8 000E
0129 00BA C26D              MOV @18(13),R9  GET POINTERS
     00BC 0012
0130 00BE 0200              LI R0,8         GET ROUNDING DIGIT COUNT
     00C0 0008
0131 00C2 C269              MOV @FFLG(9),R9 FORMATTING?
     00C4 0000
0132 00C6 160A                JNE CVBF      Y
0133 00C8 06A0              BL @CVBFR       N, ROUND
     00CA 0178'
0134 00CC 060A                DEC R10       NEW DIGIT
0135 00CE 04C4              CLR R4          CLEAR TRAILING ZEROES
0136                *
0137 00D0 0605  CVBD16      DEC R5          BACKUP 1 DIGIT
0138 00D2 9560              CB @B30,*R5     "0?
     00D4 0211'
0139 00D6 1606                JNE CVBD17    N
0140 00D8 D544              MOVB R4,*R5     Y, MAKE NULL
0141 00DA 10FA              JMP CVBD16
0142                *
0143                *  CHECK IF EXPANSION PRESENT
0144                *
0145      00DC' CVBF    EQU   $
0146 00DC 06A0              BL    @CK302    IS 302 EXPANSION EPROM PRESENT
     00DE 0000
0147 00E0 0460              B     @CVBFP    YES, DO FORMATTING
     00E2 0000
0148                *
0149                *SET SIGN AND GET INITIAL PTRS
0150                *
0151 00E4 C30C  CVBD17      MOV R12,R12     NEGATIVE?
0152 00E6 1302                JEQ $+6       N
0153 00E8 DDE0              MOVB @B2D,*R7+  Y, OUT "-
     00EA 020F'
```

```
0155                    *PROCESS FORMAT FREE #
0156                    *
0157 00EC 064A                  DECT R10            >E11?
0158 00EE 1127                    JLT CVBD27        Y
0159 00F0 028A                  CI R10,16           N, <E05?
     00F2 0010
0160 00F4 1524                    JGT CVBD27        Y
0161 00F6 022A                  AI R10,-11          N, "0.?
     00F8 FFF5
0162 00FA 1510                    JGT CVBD22        Y
0163 00FC 130F                    JEQ CVBD22        Y
0164                    *
0165 00FE DDF3  CVBD19  MOVB *R3+,*R7+  N, MOVE NUMBER
0166 0100 1307                    JEQ CVBD20        DONE
0167 0102 058A                  INC R10             TIME FOR ".?
0168 0104 11FC                    JLT CVBD19        N
0169 0106 D013                  MOVB *R3,R0         Y, ". NEEDED?
0170 0108 1315                    JEQ CVBD25        N, DONE
0171 010A DDE0                  MOVB @B2E,*R7+      Y, OUT ".
     010C 0210'
0172 010E 100F                  JMP CVBD24
0173                    *
0174 0110 0607  CVBD20  DEC R7
0175                    *
0176 0112 DDE0                  MOVB @B30,*R7+      OUT "0 UNTIL R10=0
     0114 0211'
0177 0116 058A                  INC R10             DONE?
0178 0118 11FC                    JLT $-6           N
0179 011A 100C                  JMP CVBD25          Y
0180                    *
0181 011C DDE0  CVBD22  MOVB @B30,*R7+  OUT "0
     011E 0211'
0182 0120 DDE0                  MOVB @B2E,*R7+      OUT ".
     0122 0210'
0183                    *
0184 0124 060A  CVBD23  DEC R10             TIME FOR DIGIT?
0185 0126 1103                    JLT CVBD24        Y
0186 0128 DDE0                  MOVB @B30,*R7+      N, OUT "0
     012A 0211'
0187 012C 10FB                  JMP CVBD23
0188                    *
0189 012E DDF3  CVBD24  MOVB *R3+,*R7+  MOVE REST OF STRING
0190 0130 16FE                    JNE $-2
0191 0132 0607                  DEC R7              BACKUP OVER NULL
0192                    *
0193 0134 C847  CVBD25  MOV R7,@14(13)  RETURN UPDATE PTR
     0136 000E
0194 0138 0380                  RTWP                DONE
0195                    *
0196 013A C1C6  CVBD26  MOV R6,R7           FORMATTING OVERFLOW
0197 013C 10FB                  JMP CVBD25
```

```
0199                    *EXPONENTIAL FORM
0200                    *
0201 013E DDF3  CVBD27  MOVB *R3+,*R7+  OUT FIRST DIGIT
0202 0140 D113          MOVB *R3,R4     ANOTHER DIGIT?
0203 0142 1302            JEQ $+6       N, NO ".
0204 0144 DDE0          MOVB @B2E,*R7+  Y, ".
     0146 0210'
0205                    *
0206 0148 DDF3          MOVB *R3+,*R7+  MOVE #
0207 014A 16FE            JNE $-2
0208 014C 0607          DEC R7
0209 014E DDE0          MOVB @B45,*R7+  "E
     0150 0214'
0210 0152 022A          AI R10,-10
     0154 FFF6
0211 0156 050A          NEG R10
0212 0158 1503            JGT CVBD28    POSITIVE
0213 015A DDE0          MOVB @B2D,*R7+  -, OUT "-
     015C 020F'
0214 015E 050A          NEG R10
0215                    *
0216 0160 04C9  CVBD28  CLR R9          ;CLEAR UPPER PART
0217 0162 3E60          DIV @C000A,R9   ;R9,R10/10
     0164 0206'
0218 0166 0229          AI R9,>30       ;ADD BITS
     0168 0030
0219 016A 022A          AI R10,>30
     016C 0030
0220 016E 06C9          SWPB R9         ;POSITION
0221 0170 06CA          SWPB R10
0222 0172 DDC9          MOVB R9,*R7+    ;MOVE INTO STREAM
0223 0174 DDCA          MOVB R10,*R7+
0224 0176 10DE          JMP CVBD25      ;RETURN
```

```
0226                    *
0227                    *ROUND CHARACTER STRING TO R0 TH POSITION
0228                    *
0229 0178 0206  CVBFR   LI R6,CVHD01     GET STRING ADR
     017A 0000
0230 017C A180          A R0,R6          INDEX TO ROUNDING POSITION
0231 017E D000          MOVB R0,R0       ABLE TO ROUND?
0232 0180 1618            JNE CVBFR3     N
0233 0182 C146          MOV R6,R5        Y, MARK
0234 0184 0586          INC R6
0235                    *
0236 0186 D0D6          MOVB *R6,R3      GET ROUNDING DIGIT
0237 0188 1314            JEQ CVBFR3     DONE
0238 018A D580          MOVB R0,*R6      SET TO NULL
0239 018C B0E0          AB @B05,R3       ADD 5
     018E 0000
0240 0190 90E0          CB @B3A,R3       CARRY?
     0192 0213'
0241 0194 1B0E            JH CVBFR3      N, DONE
0242                    *
0243 0196 0606  CVBFR1  DEC R6           Y, HANDLE CARRY
0244 0198 D016          MOVB *R6,R0      EOS?
0245 019A 1308            JEQ CVBFR2     Y, INSERT NEW DIGIT
0246 019C B5A0          AB @B01,*R6      N, ADD CARRY
     019E 0000
0247 01A0 95A0          CB @B3A,*R6      CARRY?
     01A2 0213'
0248 01A4 1B06            JH CVBFR3      N, DONE
0249 01A6 D5A0          MOVB @B30,*R6    Y, INSERT "0
     01A8 0211'
0250 01AA 10F5          JMP CVBFR1       CONTINUE
0251                    *
0252 01AC 049B  CVBFR2  X *R11           NEW DIGIT, ADJUST R10
0253 01AE D5A0          MOVB @B31,*R6    INSERT "1
     01B0 0212'
0254                    *
0255 01B2 0203  CVBFR3  LI R3,CVHD       GET INITIAL PTR
     01B4 0022'
0256 01B6 D1B3          MOVB *R3+,R6     NULL?
0257 01B8 1301            JEQ CVBFR4     Y
0258 01BA 0603            DEC R3         N, MOVE BACK
0259 01BC 046B  CVBFR4  B @2(11)         RETURN
     01BE 0002
0260                    *
0261                    *
0262 01C0 00E8  CVTB0   DATA >00E8,>D4A5,>1000
     01C2 D4A5
     01C4 1000
0263 01C6 0017          DATA >0017,>4876,>E800
     01C8 4876
     01CA E800
0264 01CC 0002          DATA >0002,>540B,>E400
     01CE 540B
     01D0 E400
0265 01D2 0000          DATA >0000,>3B9A,>CA00
     01D4 3B9A
     01D6 CA00
0266 01D8 0000          DATA >0000,>05F5,>E100
     01DA 05F5
     01DC E100
```

```
 0267 01DE 0000          DATA >0000,>0098,>9680
      01E0 0098
      01E2 9680
 0268 01E4 0000          DATA >0000,>000F,>4240
      01E6 000F
      01E8 4240
 0269 01EA 0000          DATA >0000,>0001,>86A0
      01EC 0001
      01EE 86A0
 0270 01F0 0000          DATA >0000,>0000,>2710
      01F2 0000
      01F4 2710
 0271 01F6 0000          DATA >0000,>0000,>03E8
      01F8 0000
      01FA 03E8
 0272 01FC 0000          DATA >0000,>0000,>0064
      01FE 0000
      0200 0064
 0273      0206' C000A   EQU $+4
 0274 0202 0000          DATA >0000,>0000,>000A
      0204 0000
      0206 000A
 0275 0208 0000          DATA >0000,>0000,>0001
      020A 0000
      020C 0001
 0276      020E' CVTB0E  EQU $
 0277            *
 0278 020E   2A  B2A     BYTE >2A
 0279 020F   2D  B2D     BYTE >2D
 0280 0210   2E  B2E     BYTE >2E
 0281 0211   30  B30     BYTE >30
 0282 0212   31  B31     BYTE >31
 0283 0213   3A  B3A     BYTE >3A
 0284 0214   45  B45     BYTE >45
 0285 0216               EVEN
 0286                    END
NO ERRORS
```

```
$          0216'          0145  0152  0178  0190  0203  0207  0273  0276
B01     R  019E'  0009    0246
B05     R  018E'  0009    0239
B2A        020E'  0278
B2D        020F'  0279    0153  0213
B2E        0210'  0280    0171  0182  0204
B30        0211'  0281    0057  0138  0176  0181  0186  0249
B31        0212'  0282    0253
B3A        0213'  0283    0240  0247
B45        0214'  0284    0209
C000A      0206'  0273    0217
CK302   R  00DE'  0010    0146
CLEAR             0022
CVBD    D  000C'  0042    0004
CVBD0      001C'  0050    0038
CVBD1      0038'  0061    0055
CVBD12     00AE'  0124    0059  0126
CVBD16     00D0'  0137    0141
CVBD17     00E4'  0151    0139
CVBD19     00FE'  0165    0168
CVBD2      003C'  0064    0056
CVBD20     0110'  0174    0166
CVBD22     011C'  0181    0162  0163
CVBD23     0124'  0184    0187
CVBD24     012E'  0189    0172  0185
CVBD25     0134'  0193    0170  0179  0197  0224
CVBD26     013A'  0196
CVBD27     013E'  0201    0158  0160
CVBD28     0160'  0216    0212
CVBD3      0044'  0068    0078  0084
CVBD4      005E'  0080    0073
CVBD5      006A'  0086    0099
CVBD6      0072'  0090    0072
CVBD7      0080'  0095    0088
CVBD8      008C'  0101    0103  0104  0113
CVBF       00DC'  0145    0132
CVBFP   R  00E2'  0010    0147
CVBFR   D  0178'  0229    0004  0133
CVBFR1     0196'  0243    0250
CVBFR2     01AC'  0252    0245
CVBFR3     01B2'  0255    0232  0237  0241  0248
CVBFR4     01BC'  0259    0257
CVBI    D  0004'  0036    0004
CVGCN   R  0062'  0009    0075  0081
CVHD    R  0184'  0006    0052  0255
CVHD01  R  017A'  0006    0229
CVHD15  R  00B2'  0006    0125
CVTB0      01C0'  0262    0092
CVTB0E     020E'  0276    0112
ERR34      0000'  0032    0065
ERROR      2F80   0029    0030
ERROR2     2FA0   0030    0032
EVFIX             0026
FADD              0016
FADDI   R  0098'  0008    0086  0105
FDIV              0019
FFLG    R  00C4'  0007    0131
FLOATF            0024
FMUL              0018
FPAC    R  0046'  0005    0042  0054  0068
```

```
FPAC2   R  0008'    0005  0037
FSUB               0017
FSUBI   R  0090'    0008  0098  0102
LOADF              0014
NEGATE             0023
NORMAL             0021
OUTFP              0027
OUTINT             0028
R0         0000          0077  0083  0090  0091  0130  0169  0230  0231  0231
                         0238  0244
R1         0001          0054  0064  0068  0069  0071
R10        000A          0042  0043  0044  0045  0046  0047  0048  0058  0066
                         0076  0082  0087  0134  0157  0159  0161  0167  0177
                         0184  0210  0211  0214  0219  0221  0223
R11        000B          0037  0043  0044  0045  0046  0047  0048  0252
R12        000C          0051  0062  0151  0151
R3         0003          0092  0095  0096  0097  0109  0110  0111  0112  0165
                         0169  0189  0201  0202  0206  0236  0239  0240  0255
                         0256  0258
R4         0004          0095  0109  0135  0140  0202
R5         0005          0096  0110  0137  0138  0140  0233
R6         0006          0097  0111  0196  0229  0230  0233  0234  0236  0238
                         0243  0244  0246  0247  0249  0253  0256
R7         0007          0052  0053  0053  0057  0107  0124  0124  0125  0128
                         0153  0165  0171  0174  0176  0181  0182  0186  0189
                         0191  0193  0196  0201  0204  0206  0208  0209  0213
                         0222  0223
R8         0008          0070  0071  0076  0080  0082
R9         0009          0093  0101  0106  0107  0108  0129  0131  0216  0217
                         0218  0220  0222
SCALE              0020
STORE              0015
*
```

```
SOURCE ACCESS NAME=      DBASIC.REL0.BASIC.SOURCE.FUNC
OBJECT ACCESS NAME=      DBASIC.REL0.BASIC.OBJECT.FUNC
LISTING ACCESS NAME=     DBASIC.REL0.BASIC.LIST.FUNC
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'FUNC'
0002          *         08/16/78
0003          *
0004                    DEF SYSF,ABSF,NKYF
0005                    DEF INPF
0006                    REF FPAC,FPAC2
0007                    REF FIX,EVSFR
0008                    REF HFLG,ENUM,ELNM,LEC
0009                    REF C4A00
0010    2F80  ERROR     EQU >2F80
0011    2FA0  ERROR2    EQU ERROR+>20
0012          *
0013          *  DEFINE FLOATING POINT XOPS FOR THIS MODULE
0014          *
0015                    DXOP LOADF,0            ;LOAD FPAC
0016                    DXOP SCALE,6            ;SCALE PFAC
0017                    DXOP NORMAL,7           ;NORMALIZE FPAC
0018                    DXOP CLEAR,8            ;CLEAR FPAC
0019                    DXOP NEGATE,9           ;NEGATE FPAC
0020                    DXOP FLOATF,10          ;FLOAT FPAC
0021          *
0022          *SYSTEM FUNCTION
0023          *
0024 0000 06A0  SYSF    BL @FIX                 ;GET PARAMETER
     0002 0000
0025 0004 C041          MOV R1,R1               ;0?
0026 0006 1305            JEQ SYSF0             ;Y
0027 0008 0911          SRL R1,1
0028 000A 1606            JNE SYSF2
0029 000C C069  SYSF1   MOV @ENUM(9),R1 ;RETURN LAST ERROR #
     000E 0000
0030 0010 100D          JMP INPF1
0031          *
0032 0012 C069  SYSF0   MOV @HFLG(9),R1 ;RETURN HELP FLAG
     0014 0000
0033 0016 100A          JMP INPF1
0034          *
0035 0018 C069  SYSF2   MOV @ELNM(9),R1 ;GET LAST ERROR LINE #
     001A 0000
0036 001C 1007          JMP INPF1
```

```
0038                    *INTEGER PART FUNCTION
0039                    *
0040 001E C052  INPF    MOV *R2,R1      ;INTEGER ALREADY?
0041 0020 130A            JEQ INPF3     ;Y
0042 0022 2C12          LOADF *R2       ;N, LOAD FPAC
0043 0024 2DA0          SCALE @C4A00    ;SCALE OFF FRACTION
     0026 0000
0044 0028 2DC0          NORMALIZE 0
0045 002A 1003          JMP INPF2
0046                    *
0047 002C 2E00  INPF1   CLEAR 0         ;SAVE #
0048 002E C801          MOV R1,@FPAC2
     0030 0000
0049 0032 0202  INPF2   LI R2,FPAC      ;RETURN ADDRESS
     0034 0000
0050 0036 0460  INPF3   B @EVSFR        ;RETURN
     0038 0000
0051                    *
0052                    *ABSOLUTE FUNCTION
0053                    *
0054 003A 2C12  ABSF    LOADF *R2       ;LOAD FPAC
0055 003C C052          MOV *R2,R1      ;CHECK SIGN
0056 003E 1303            JEQ ABSF1     ;INTEGER
0057 0040 15FA            JGT INPF3     ;+, RT
0058 0042 2E40          NEGATE 0        ;-, NEGATE
0059 0044 10F6          JMP INPF2       ;RETURN
0060                    *
0061 0046 0760  ABSF1   ABS @FPAC2      ;INTEGER, TAKE ABS
     0048 0030'
0062 004A 19F3            JNO INPF2     ;NO OVERFLOW
0063 004C 0200          LI R0,>4480     ;OVERFLOW, LOAD 32768 (FP)
     004E 4480
0064 0050 04C2          CLR R2
0065 0052 2C00          LOADF R0        ;LOAD R0,R1,R2
0066 0054 10EE          JMP INPF2
0067                    *
0068                    *NKY FUNCTION
0069                    *
0070 0056 C029  NKYF    MOV @LEC(9),R0  ;CHARACTER?
     0058 0000
0071 005A 13EB            JEQ INPF2     ;N, RETURN 0
0072 005C 06C0          SWPB R0
0073 005E 06A0          BL @FIX         ;FIX ARGUMENT
     0060 0002'
0074 0062 C041          MOV R1,R1       ;ARG=0?
0075 0064 1302            JEQ NKYF1     ;Y, RETURN LEC
0076 0066 6040          S R0,R1         ;N, ARG=LEC?
0077 0068 16E4            JNE INPF2     ;N, RETURN 0
0078 006A CA41  NKYF1   MOV R1,@LEC(9)  ;CLEAR LEC
     006C 0058'
0079 006E C040          MOV R0,R1
0080 0070 10DD          JMP INPF1
0081                    END
NO ERRORS
```

```
*  IF   NUMBER >= SCALING FACTOR
*                 GET ERROR 24 (FLOATING PT OVERFLOW).
*INEI  R1 > 7F00
C      R1 =
JHE    INPF2                    >= SCALING FACTOR?
```

IN

2768 (FP)

LABEL		VALUE	DEFN	REFERENCES								
ABSF	D	003A'	0054	0004								
ABSF1		0046'	0061	0056								
C4A00	R	0026'	0009	0043								
CLEAR			0018									
ELNM	R	001A'	0008	0035								
ENUM	R	000E'	0008	0029								
ERROR		2F80	0010	0011								
ERROR2		2FA0	0011									
EVSFR	R	0038'	0007	0050								
FIX	R	0060'	0007	0024	0073							
FLOATF			0020									
FPAC	R	0034'	0006	0049								
FPAC2	R	0048'	0006	0048	0061							
HFLG	R	0014'	0008	0032								
INPF	D	001E'	0040	0005								
INPF1		002C'	0047	0030	0033	0036	0080					
INPF2		0032'	0049	0045	0059	0062	0066	0071	0077			
INPF3		0036'	0050	0041	0057							
LEC	R	006C'	0008	0070	0078							
LOADF			0015									
NEGATE			0019									
NKYF	D	0056'	0070	0004								
NKYF1		006A'	0078	0075								
NORMAL			0017									
R0		0000		0063	0065	0070	0072	0076	0079			
R1		0001		0025	0025	0027	0029	0032	0035	0040	0048	0055
				0074	0074	0076	0078	0079				
R2		0002		0040	0042	0049	0054	0055	0064			
SCALE			0016									
SYSF	D	0000'	0024	0004								
SYSF0		0012'	0032	0026								
SYSF1		000C'	0029									
SYSF2		0018'	0035	0028								

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.FIX
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.FIX
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.FIX
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT  'FIX'
0002                 *
0003                 * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *
0006                 * ROUTINE LIST:
0007                 *
0008                 *       FIX      ;FIX 2'S COMPLEMENT INTEGER
0009                 *       PFIX     ;FIX 16-BIT POSITIVE INTEGER
0010                 *
0011                 * COPY: (NONE)
0012                 *
0013                 * MACRO: (NONE)
0014                 *
0015                 * EXTERNAL ROUTINES:
0016                 *
0017                       DXOP LOADF,0              ;LOAD FPAC
0018                       DXOP SCALE,6              ;SCALE FPAC
0019                       DXOP CLEAR,8              ;CLEAR FPAC
0020      2F80       ERROR  EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0021                 *
0022                 * EXTERNAL DATA:
0023                 *
0024                      REF FPAC,FPAC2                ;FLOATING POINT ACCUMULA
0025                 *
0026                 * MODULE EQUATES:       (NONE)
0027                 *
0028                 * MODULE VARIABLES AND CONSTANTS:
0029                 *
0030                      TITL "FIX INTEGER"
```

```
0032            * ABSTRACT:
0033            *
0034            *       FIX RETURNS THE 2'S COMPLEMENT 16-BIT INTEGER VALUE
0035            *       THE 3 WORD NUMBER POINTED TO BY R2.  IF THE NUMBER I
0036            *       AN INTEGER (1ST WORD = 0 ) THEN IT SIMPLY RETURNS TH
0037            *       IF THE VALUE IS A FLOATING POINT NUMBER (1ST WORD <>
0038            *       NUMBER IS LOADED INTO THE FLOATING POINT ACCUMULATOR
0039            *       THE DECIMAL POINT IS AFTER THE SECOND WORD, AD THEN
0040            *       2ND WORD OF FPAC IS RETURNED.  CARE IS TAKE TO ZERO
0041            *       TO NEGATE THE RESULT DEPENDING UPON THE SIGN BIT.
0042            *
0043            *       PFIX RETURNS A 16-BIT RESULT RATHER THAN THE 2'S
0044            *       COMPLEMENT VALUE.
0045            *
0046            * CALLING SEQUENCE:
0047            *
0048            *       BL @FIX
0049            *         OR
0050            *       BL @PFIX
0051            *
0052            *       IN - (R2) = 3 WORD PBASIC NUMBER.
0053            *       OUT - R1 = RESULT.
0054            *
0055            *      NORMAL EXIT - RETURN
0056            *
0057            * EXCEPTIONS AND CONDITIONS:
0058            *
0059            *       ERROR 30 RESULTS IF THE ABSOLUTE VALUE OF THE NUMBER
0060            *       GREATER THAN 2↑15
0061            *
0062            *       R0 IS PRESERVED
0063            *
0064            * EXTERNAL ROUTINE LIST:
0065            *
0066            *    (SEE EXTERNAL ROUTINES)
0067            *
0068            * LOCAL DATA:
0069            *
0070            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0072                    * ENTRY POINT:
0073                    *
0074                            DEF FIX
0075                    *
0076 0000 C072  FIX     MOV *R2+,R1     ;INTEGER?
0077 0002 1602            JNE FIX1      ;N, DO INTF
0078 0004 C052          MOV *R2,R1      ;Y, RETURN #
0079 0006 045B          RT
0080                    *
0081 0008 2C22  FIX1    LOADF @-2(2)    ;LOAD FPAC
     000A FFFE
0082                    *
0083 000C C081  FIX2    MOV R1,R2       ;SAVE
0084 000E 0241          ANDI R1,>7F80   ;MASK TO EXPONENT + 1 BIT
     0010 7F80
0085 0012 0281          CI R1,>4400     ;TOO LARGE?
     0014 4400
0086 0016 1509            JGT ERR30     ;Y
0087 0018 2DA0          SCALE @C4600    ;N, GET INTEGER
     001A 003C'
0088 001C C060          MOV @FPAC2,R1
     001E 0000
0089 0020 2E00          CLEAR 0         ;LEAVE FPAC ZERO
0090 0022 0A12          SLA R2,1        ;NEGATIVE?
0091 0024 1701            JNC FIX3      ;N
0092 0026 0501          NEG R1          ;Y
0093 0028 045B  FIX3    RT
0094                    *
0095 002A 2F9E  ERR30   DATA ERROR+30   ;FIX ERROR
0096                    *
0097                    *
0098                    *       ENTRY POINT:
0099                    *
0100                            DEF PFIX
0101                    *
0102 002C 2C12  PFIX    LOADF *R2       ;LOAD FPAC
0103 002E C052          MOV *R2,R1      ;INTEGER?
0104 0030 1302            JEQ PFIX1     ;Y
0105 0032 2DA0          SCALE @C4600    ;N, SCALE
     0034 003C'
0106                    *
0107 0036 C060  PFIX1   MOV @FPAC2,R1   ;GET RESULT
     0038 001E'
0108 003A 045B          RT
0109 003C 4600  C4600   DATA >4600
0110                    END
NO ERRORS,      NO WARNINGS
```

C4600		003C'	0109	0087	0105						
CLEAR			0019								
ERR30		002A'	0095	0086							
ERROR		2F80	0020	0095							
FIX	D	0000'	0076	0074							
FIX1		0008'	0081	0077							
FIX2		000C'	0083								
FIX3		0028'	0093	0091							
FPAC	R		0024								
FPAC2	R	0038'	0024	0088	0107						
LOADF			0017								
PFIX	D	002C'	0102	0100							
PFIX1		0036'	0107	0104							
R1		0001		0076	0078	0083	0084	0085	0088	0092	0103	0107
R2		0002		0076	0078	0083	0090	0102	0103			
SCALE			0018								

*

SOURCE ACCESS NAME= BD1.DBASIC.REL5.BASIC.SOURCE.POLY
OBJECT ACCESS NAME= BD1.DBASIC.REL5.BASIC.OBJECT.POLY
LISTING ACCESS NAME= BD1.DBASIC.REL5.BASIC.LIST.POLY
ERROR ACCESS NAME=
OPTIONS= XREF
MACRO LIBRARY PATHNAME=

0001

```
                IDT  'POLY'
0002                 *
0003                 * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *
0006                 * ROUTINE LIST:
0007                 *
0008                 *       FUNBK               ;BREAK FPAC INTO INTEGER & FRACTION
0009                 *       FUNFX               ;FIX FPAC INTO EXPONENT AND ABSOLUTE
0010                 *       PLYX                ;EVALUATE P(X) WITH X IN TEMP
0011                 *       PLYXX               ;EVALUATE P(X*X) WITH X IN FPAC
0012                 *
0013                 * COPY: (NONE)
0014                 *
0015                 * MARCO: (NONE)
0016                 *
0017                 * EXTERNAL ROUTINES:
0018                 *
0019                       DXOP LOADF,0      ;LOAD FPAC
0020                       DXOP STORE,1      ;STORE FPAC
0021                       DXOP FADD,2       ;ADD TO FPAC
0022                       DXOP FSUB,3       ;SUBTRACT FROM FPAC
0023                       DXOP FMUL,4       ;MULTIPLY FPAC
0024                       DXOP FDIV,5       ;DIVIDE FPAC
0025                       DXOP SCALE,6      ;SCALE FPAC
0026                       DXOP NORMAL,7     ;NORMALIZE FPAC
0027                       DXOP CLEAR,8      ;CLEAR FPAC
0028                       DXOP NEGATE,9     ;NEGATE FPAC
0029                       DXOP FLOATF,10    ;FLOAT FPAC
0030          2F80  ERROR  EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0031                 *
0032                 * EXTERNAL DATA:
0033                 *
0034                       REF FPAC,FPAC4     ;FLOATING POINT ACCUMULATOR
0035                       REF TEMP          ;3 WRD TEMPORARY STORAGE
0036                       REF C4A00         ;>4A00
0037                 *
0038                 * MODULE EQUATES:  (NONE)
0039                 *
0040                 * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0043                    * ABSTRACT:
0044                    *
0045                    *       BREAK FPAC INTO INTEGER IN R1 AND
0046                    *       LEAVE FRACTIONAL PART IN FPAC.
0047                    *
0048                    * CALLING SEQUENCE:
0049                    *
0050                    *       BL @FUNBK
0051                    *
0052                    *       IN FPAC = #
0053                    *       OUT  R1 = INTEGER PART
0054                    *          FPAC = FRACTIONAL PART
0055                    *
0056                    * EXCEPTIONS AND CONDITIONS:
0057                    *
0058                    *       ERROR 30 = FIX ERROR
0059                    *
0060                    * EXTERNAL ROUTINE LIST:
0061                    *
0062                    *      (SEE EXTERNAL ROUTINES)
0063                    *
0064                    * LOCAL DATA:
0065                    *
0066                    *      (SEE MODULE VARIABLES AND CONSTANTS)
0067                    *
0068                    * ENTRY POINT:
0069                    *
0070                            DEF FUNBK
0071                    *
0072 0000 C060  FUNBK   MOV @FPAC,R1    GET HIGH WORD
     0002 0000
0073 0004 0A11          SLA R1,1        REMOVE SIGN
0074 0006 0281          CI R1,>8900     TOO BIG?
     0008 8900
0075 000A 1B0F            JH ERR30      Y, ERROR
0076 000C 2C60          STORE @TEMP     ;N, MOVE TO TEMP
     000E 0000
0077 0010 2DA0          SCALE @C4A00    ;SCALE
     0012 0000
0078 0014 C060          MOV @FPAC4,R1   LOAD LOW
     0016 0000
0079 0018 C0A0          MOV @FPAC,R2    <0?
     001A 0002'
0080 001C 1501            JGT FUNBK1    N
0081 001E 0501          NEG R1          Y, GET 2'S COMPLEMENT
0082                    *
0083 0020 2DC0  FUNBK1  NORMALIZE 0     NORMALIZE
0084 0022 2E40          NEGATE 0        NEGATE
0085 0024 2CA0          FADD @TEMP      ADD TEMP TO FPAC
     0026 000E'
0086 0028 045B          RT              RETURN
0087                    *
0088 002A 2F9E  ERR30   DATA ERROR+30
```

```
0091                * ABSTRACT:
0092                *
0093                *       FIX FPAC INTO EXPONENT IN R1, SIGN BIT
0094                *       IN R3, AND ABSOLUTE VALUE IN FPAC
0095                *
0096                * CALLING SEQUENCE:
0097                *
0098                *       BL @FUNFX
0099                *
0100                *       IN  R2 = #
0101                *       OUT R1 = EXPONENT
0102                *         (R2) = ABS(FPAC)
0103                *           R3 = SIGN BIT
0104                *
0105                * EXCEPTIONS AND CONDITIONS:  NONE
0106                *
0107                * EXTERNAL ROUTINE:
0108                *
0109                *       (SEE EXTERNAL ROUTINE LIST)
0110                *
0111                * LOCAL DATA:  NONE
0112                *
0113                * ENTRY POINT:
0114                *
0115                        DEF FUNFX
0116                *
0117 002C 2C12     FUNFX    LOADF *R2          ;LOAD FPAC
0118 002E 2E80              FLOATF 0           ;FLOAT IF NECESSARY
0119 0030 0202              LI R2,FPAC         GET ADR
     0032 001A'
0120 0034 C0D2              MOV *R2,R3         GET SIGN
0121 0036 1306                JEQ FUNFX2       ZERO
0122 0038 09F3              SRL R3,15          GET SIGN BIT
0123 003A 1301                JEQ FUNFX1       + OR 0
0124 003C 2E40              NEGATE 0           -, NEGATE
0125                *
0126 003E 05CB     FUNFX1   INCT R11           RETURN @2(11)
0127 0040 04C1              CLR R1
0128 0042 D052              MOVB *R2,R1        GET EXPONENT
0129 0044 045B     FUNFX2   RT
```

```
0132                    * ABSTRACT:
0133                    *
0134                    *       EVALUATE POLYNOMIAL P(X) OR P(X*X) WITH
0135                    *       X IN TEMP OR FPAC RESPECTIVELY.  THE
0136                    *       POLYNOMIAL IS GIVEN BY:
0137                    *
0138                    *               (R0) = N,C1,C2,...,CN
0139                    *
0140                    *       WHERE   N = # OF COEFFICIENTS
0141                    *               C = COEFFICIENTS
0142                    *
0143                    * CALLING SEQUENCE:
0144                    *
0145                    *       BL @PLYXX
0146                    *       BL @PLYX
0147                    *
0148                    *       IN (R0) = N,C1,C2,...,CN
0149                    *       OUT FPAC = RESULT
0150                    *
0151                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0152                    *
0153                    * EXTERNAL ROUTINES:
0154                    *
0155                    *       (SEE EXTERNAL ROUTINE LIST)
0156                    *
0157                    * LOCAL DATA:  (NONE)
0158                    *
0159                    * ENTRY POINTS:
0160                    *
0161                            DEF PLYX,PLYXX
0162                    *
0163                    *EVALUATE P(X*X) WITH X IN FPAC
0164                    *
0165 0046 2C60  PLYXX   STORE @TEMP      MOVE X TO TEMP
     0048 0026'
0166 004A 2D20          FMUL @TEMP      FPAC=FPAC*FPAC
     004C 0048'
0167 004E 2C60          STORE @TEMP     MOVE X*X INTO TEMP
     0050 004C'
0168                    *
0169                    *EVALUATE P(X) WITH X IN TEMP
0170                    *
0171 0052 C03B  PLYX    MOV *R11+,R0    GET CONSTANTS ADR
0172 0054 C070          MOV *R0+,R1     GET COUNT
0173 0056 2C10          LOADF *R0       LOAD 1ST CONSTANT
0174                    *
0175 0058 2D20  PLYXA   FMUL @TEMP      FPAC=FPAC*TEMP
     005A 0050'
0176 005C 0220          AI R0,6         MOVE TO NEXT CONSTANT
     005E 0006
0177 0060 2C90          FADD *R0        ADD
0178 0062 0601          DEC R1          DONE?
0179 0064 16F9            JNE PLYXA     N
0180 0066 045B          RT              Y
0181                    END
NO ERRORS,     NO WARNINGS
```

C4A00	R	0012'	0036	0077							
CLEAR			0027								
ERR30		002A'	0088	0075							
ERROR		2F80	0030	0088							
FADD			0021								
FDIV			0024								
FLOATF			0029								
FMUL			0023								
FPAC	R	0032'	0034	0072	0079	0119					
FPAC4	R	0016'	0034	0078							
FSUB			0022								
FUNBK	D	0000'	0072	0070							
FUNBK1		0020'	0083	0080							
FUNFX	D	002C'	0117	0115							
FUNFX1		003E'	0126	0123							
FUNFX2		0044'	0129	0121							
LOADF			0019								
NEGATE			0028								
NORMAL			0026								
PLYX	D	0052'	0171	0161							
PLYXA		0058'	0175	0179							
PLYXX	D	0046'	0165	0161							
R0		0000		0171	0172	0173	0176	0177			
R1		0001		0072	0073	0074	0078	0081	0127	0128	0172	0178
R11		0008		0126	0171						
R2		0002		0079	0117	0119	0120	0128			
R3		0003		0120	0122						
SCALE			0025								
STORE			0020								
TEMP	R	005A'	0035	0076	0085	0165	0166	0167	0175		

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.LOGF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.LOGF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.LOGF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'LOGF'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       LOGF              ;NATURAL LOG FUNCTION
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MACRO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:
0015              *
0016                      REF FUNFX         ;FIX ARGUMENT
0017                      REF PLYX,PLYXX    ;EVALUATE POLYNOMIAL
0018                      REF GETP2         ;GET POWER OF 2
0019                      DXOP LOADF,0      ;LOAD FPAC
0020                      DXOP STORE,1      ;STORE FPAC
0021                      DXOP FADD,2       ;ADD TO FPAC
0022                      DXOP FSUB,3       ;SUBTRACT FROM FPAC
0023                      DXOP FMUL,4       ;MULTIPLY FPAC
0024                      DXOP FDIV,5       ;DIVIDE FPAC
0025                      DXOP NORMAL,7     ;NORMALIZE FPAC
0026    2F80  ERROR       EQU >2F80         ;XOP XX,14  (ERROR CALL)
0027              *
0028              * EXTERNAL DATA:  (NONE)
0029                    REF FPAC            ;FLOATING POINT ACCUMULATOR
0030                    REF TEMP            ;3 WRD TEMPORARY STORAGE
0031                    REF DS,DS1          ;3 WRD TEMPORARY STORAGE
0032              *
0033              * MODULE EQUATES:  (NONE)
0034              *
0035              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0038            * ABSTRACT:
0039            *
0040            *       COMPUTE THE NATURAL LOG OF (R2)
0041            *
0042            * CALLING SEQUENCE:
0043            *
0044            *       BL @LOGF
0045            *
0046            *       IN  (R2) = ARG
0047            *       OUT (R2) = RESULT
0048            *
0049            * EXCEPTIONS AND CONDITIONS:
0050            *
0051            *       LOG OF NON-POSITIVE NUMBER
0052            *
0053            * EXTERNAL ROUTINE LIST:
0054            *
0055            *      (SEE EXTERNAL ROUTINES)
0056            *
0057            * LOCAL DATA:
0058            *
0059            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0061                    * ENTRY POINT:
0062                    *
0063                          DEF LOGF
0064                    *
0065 0000 2F9A  ERR26   DATA ERROR+26    ;LOG OF NON-POSITIVE NUMBER
0066                    *
0067 0002 C28B  LOGF    MOV R11,R10      ;SAVE RETURN
0068 0004 06A0          BL @FUNFX        ;FIX
     0006 0000
0069 0008 10FB            JMP ERR26      ;LOG 0
0070 000A C0C3          MOV R3,R3        ;NEGATIVE?
0071 000C 16F9            JNE ERR26      ;Y, ERROR
0072 000E D820          MOVB @B40,@FPAC  ;LAOD EXPONENT
     0010 00EA'
     0012 0000
0073 0014 0961          SRL R1,6         ;SWAP & X 4
0074 0016 0221          AI R1,->100      ;UNBIAS
     0018 FF00
0075 001A C092          MOV *R2,R2       ;GET FPAC
0076 001C 0200          LI R0,>80
     001E 0080
0077 0020 2080          COC R0,R2        ;HIGH BIT SET?
0078 0022 1308            JEQ LOGF2      ;Y
0079 0024 04C3          CLR R3           ;N, FIND POWER OF 2
0080                    *
0081 0026 0583  LOGF1   INC R3           ;COUNT
0082 0028 0910          SRL R0,1
0083 002A 2080          COC R0,R2        ;BIT SET?
0084 002C 16FC            JNE LOGF1      ;N, CONTINUE TO COUNT
0085 002E 6043          S R3,R1
0086 0030 06A0          BL @GETP2        ;GET POWER OF 2 & MULTIPLY
     0032 0000
0087                    *
0088 0034 C081  LOGF2   MOV R1,R2
0089 0036 2C60          STORE @TEMP      ;MOVE TO TEMP
     0038 0000
0090 003A 2CE0          FSUB @LOGC0
     003C 00AA'
0091 003E 0200          LI R0,FP1        ;GET FP1
     0040 00B8'
0092 0042 C060          MOV @FPAC,R1     ;CHECK SIGN
     0044 0012'
0093 0046 1504            JGT LOGF3
0094 0048 1303            JEQ LOGF3
0095 004A 0200          LI R0,LOGC1
     004C 00B0'
0096 004E 0602          DEC R2
0097                    *
0098 0050 2C20  LOGF3   LOADF @TEMP      ;MOVE TEMP TO FPAC
     0052 0038'
0099 0054 2C90          FADD *R0
0100 0056 2C60          STORE @DS        ;SAVE IN DS
     0058 0000
0101 005A 2C20          LOADF @TEMP      ;RELOAD TEMP
     005C 0052'
0102 005E 2CD0          FSUB *R0
0103 0060 2D60          FDIV @DS         ;DIVIDE BY DS
     0062 0058'
0104 0064 2C60          STORE @DS        ;SAVE IN DS
     0066 0062'
```

```
0105 0068 06A0          BL @PLYXX
     006A 0000
0106 006C 00B6'           DATA LOGC2
0107 006E 2C60          STORE @DS1      ;STORE IN DS1
     0070 0000
0108 0072 06A0          BL @PLYX
     0074 0000
0109 0076 00D0'           DATA LOGC3
0110 0078 2D60          FDIV @DS1       ;DIVIDE BY DS1
     007A 0070'
0111 007C 2D20          FMUL @DS        ;MULTIPLY BY DS
     007E 0066'
0112 0080 2C60          STORE @TEMP     ;STORE IN TEMP
     0082 005C'
0113 0084 0200          LI R0,>8C00
     0086 8C00
0114 0088 C082          MOV R2,R2
0115 008A 1102            JLT LOGF4
0116 008C 0910          SRL R0,1
0117 008E 1002          JMP LOGF5
0118              *
0119 0090 0502  LOGF4   NEG R2
0120 0092 0810          SRA R0,1
0121              *
0122 0094 C042  LOGF5   MOV R2,R1
0123 0096 04C2          CLR R2
0124 0098 2C00          LOADF R0        ;LOAD FPAC
0125 009A 2DC0          NORMALIZE 0
0126 009C 2D20          FMUL @LOGC4
     009E 00E4'
0127 00A0 2CA0          FADD @TEMP
     00A2 0082'
0128              *
0129 00A4 0202  LOGF6   LI R2,FPAC
     00A6 0044'
0130 00A8 045A          B *R10
0131              *
0132 00AA 40B5  LOGC0   DATA >40B5,>04F3,>33FA
     00AC 04F3
     00AE 33FA
0133 00B0 4080  LOGC1   DATA >4080,>0000,>0000  1/2
     00B2 0000
     00B4 0000
0134 00B6 0003  LOGC2   DATA 3
0135 00B8 4110  FP1     DATA >4110,>0000,>0000
     00BA 0000
     00BC 0000
0136 00BE C214          DATA >C214,>BBC5,>DCDB
     00C0 BBC5
     00C2 DCDB
0137 00C4 423D          DATA >423D,>C2D5,>31F0
     00C6 C2D5
     00C8 31F0
0138 00CA C22D          DATA >C22D,>165C,>4BE0
     00CC 165C
     00CE 4BE0
0139 00D0 0002  LOGC3   DATA 2
0140 00D2 C212          DATA >C212,>53EF,>500E
     00D4 53EF
     00D6 500E
```

```
 0141 00D8 425D           DATA >425D,>76C2,>314A
      00DA 76C2
      00DC 314A
 0142 00DE C25A           DATA >C25A,>2CB8,>97BF
      00E0 2CB8
      00E2 97BF
 0143 00E4 40B1  LOGC4    DATA >40B1,>7217,>F7D2
      00E6 7217
      00E8 F7D2
 0144 00EA   40  B40      BYTE >40
 0145 00EC                EVEN
 0146                     END
NO ERRORS,      NO WARNINGS
```

B40		00EA'	0144	0072								
DS	R	007E'	0031	0100	0103	0104	0111					
DS1	R	007A'	0031	0107	0110							
ERR26		0000'	0065	0069	0071							
ERROR		2F80	0026	0065								
FADD			0021									
FDIV			0024									
FMUL			0023									
FP1		00B8'	0135	0091								
FPAC	R	00A6'	0029	0072	0092	0129						
FSUB			0022									
FUNFX	R	0006'	0016	0068								
GETP2	R	0032'	0018	0086								
LOADF			0019									
LOGC0		00AA'	0132	0090								
LOGC1		00B0'	0133	0095								
LOGC2		00B6'	0134	0106								
LOGC3		00D0'	0139	0109								
LOGC4		00E4'	0143	0126								
LOGF	D	0002'	0067	0063								
LOGF1		0026'	0081	0084								
LOGF2		0034'	0088	0078								
LOGF3		0050'	0098	0093	0094							
LOGF4		0090'	0119	0115								
LOGF5		0094'	0122	0117								
LOGF6		00A4'	0129									
NORMAL			0025									
PLYX	R	0074'	0017	0108								
PLYXX	R	006A'	0017	0105								
R0		0000		0076	0077	0082	0083	0091	0095	0099	0102	0113
				0116	0120	0124						
R1		0001		0073	0074	0085	0088	0092	0122			
R10		000A		0067	0130							
R11		000B		0067								
R2		0002		0075	0075	0077	0083	0088	0096	0114	0114	0119
				0122	0123	0129						
R3		0003		0070	0070	0079	0081	0085				
STORE			0020									
TEMP	R	00A2'	0030	0089	0098	0101	0112	0127				

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.EXPF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.EXPF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.EXPF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT  'EXPF'
0002                 *
0003                 * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *
0006                 * ROUTINE LIST:
0007                 *
0008                 *       EXPF              ;EXPONENTIAL FUNCTION
0009                 *
0010                 * COPY: (NONE)
0011                 *
0012                 * MARCO: (NONE)
0013                 *
0014                 * EXTERNAL ROUTINES:
0015                 *
0016                       REF FUNFX                ;FIXES ARGUMENT
0017                       REF PLYX,PLYXX            ;EVALUATE POLYNOMIALS
0018                       REF FUNBK                 ;BREAK FP #
0019                       REF GETP2                 ;GET POWER OF 2
0020                 *
0021                       DXOP LOADF,0              ;LOAD FPAC
0022                       DXOP STORE,1              ;STORE FPAC
0023                       DXOP FADD,2               ;ADD TO FPAC
0024                       DXOP FSUB,3               ;SUBTRACT FROM FPAC
0025                       DXOP FMUL,4               ;MULTIPLY FPAC
0026                       DXOP FDIV,5               ;DIVIDE FPAC
0027      2F80   ERROR    EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0028      2FA0   ERROR2   EQU ERROR+>20
0029                 *
0030                 * EXTERNAL DATA:
0031                 *
0032                       REF FPAC                 ;FLOATING POINT ACCUMULATOR
0033                       REF TEMP                 ;3 WRD TEMPORARY STORAGE
0034                       REF DS1                  ;3 WRD TEMPORARY STORAGE
0035                       REF DS                   ;3 WRD TEMPORARY STORAGE
0036                 *
0037                 * MODULE EQUATES:  (NONE)
0038                 *
0039                 * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0042            * ABSTRACT:
0043            *
0044            *       CALCULATE THE EXPONENTIAL VALUE OF
0045            *       E RAISED TO (ARG)
0046            *
0047            * CALLING SEQUENCE:
0048            *
0049            *       B @EXPF
0050            *
0051            *       IN  (R2) = ARG
0052            *       OUT (R2) = E ↑ ARG
0053            *
0054            * EXCEPTIONS AND CONDITIONS:
0055            *
0056            *       ERROR 33 = EXP ARGUMENT TOO LARGE
0057            *       FLOATING POINT ERRORS
0058            *
0059            * EXTERNAL ROUTINE LIST:
0060            *
0061            *      (SEE EXTERNAL ROUTINES)
0062            *
0063            * LOCAL DATA:
0064            *
0065            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0067                    * ENTRY POINT:
0068                    *
0069                          DEF EXPF
0070                    *
0071 0000 2FA0  ERR33   DATA ERROR2,33  ;EXP ARG TOO LARGE
     0002 0021
0072                    *
0073 0004 C28B  EXPF    MOV R11,R10
0074 0006 06A0          BL @FUNFX       ;FIX SIGN & FPAC
     0008 0000
0075 000A 1339            JEQ EXPF2     ;EXP(0)=1
0076 000C 2D20          FMUL @EXPC0     ;F=F*LN 2
     000E 0088'
0077 0010 06A0          BL @FUNBK       ;BREAK TO INTEGER AND FRACTION
     0012 0000
0078 0014 0281          CI R1,>7D       ;ARGUMENT TOO LARGE?
     0016 007D
0079 0018 15F3            JGT ERR33     ;Y, ERROR
0080 001A C081          MOV R1,R2       ;N, SAVE
0081 001C 2CE0          FSUB @EXPC1     ;F=(F*LN 2)-C1
     001E 008E'
0082 0020 2C60          STORE @DS       ;STORE IN DS
     0022 0000
0083 0024 06A0          BL @PLYXX       ;EVALUATE
     0026 0000
0084 0028 0094'           DATA EXPC2
0085 002A 2D20          FMUL @DS        ;* DS
     002C 0022'
0086 002E 2C60          STORE @DS       ;STORE IN DS AGAIN
     0030 002C'
0087 0032 06A0          BL @PLYX        ;EVALUATE
     0034 0000
0088 0036 00A8'           DATA EXPC3
0089 0038 2C60          STORE @TEMP     ;MOVE TO TEMP
     003A 0000
0090 003C 2CE0          FSUB @DS        ;-DS
     003E 0030'
0091 0040 2C60          STORE @DS1      ;SAVE IN DS1
     0042 0000
0092 0044 2C20          LOADF @TEMP     ;MOVE TEMP TO FPAC
     0046 003A'
0093 0048 2CA0          FADD @DS        ;ADD DS
     004A 003E'
0094 004C 2D60          FDIV @DS1       ;/ DS1
     004E 0042'
0095 0050 2D20          FMUL @EXPC4     ;* C4
     0052 00C2'
0096 0054 C042          MOV R2,R1       ;FIX EXPONENT
0097 0056 0921          SRL R1,2
0098 0058 06C1          SWPB R1
0099 005A A801          A R1,@FPAC      ;ADD TO EXPONENT
     005C 0000
0100 005E C043          MOV R3,R1
0101 0060 0242          ANDI R2,>3      ;NEED POWER OF 2?
     0062 0003
0102 0064 1303            JEQ EXPF1     ;N
0103 0066 C0C2          MOV R2,R3
0104 0068 06A0          BL @GETP2       ;MULTIPY BY POWER OF 2
     006A 0000
0105                    *
```

```
0106 006C C041  EXPF1   MOV R1,R1        ;NEGATIVE?
0107 006E 1309            JEQ EXPF3      ;N, RETURN
0108 0070 2C60          STORE @TEMP      ;Y, MOVE TO TEMP
     0072 0046'
0109 0074 2C20          LOADF @FP1       ;GET INVERSE, LOAD FPAC
     0076 00AA'
0110 0078 2D60          FDIV @TEMP
     007A 0072'
0111 007C 1002          JMP EXPF3        ;DONE
0112              *
0113 007E 2C20  EXPF2   LOADF @FP1       ;EXP(0)=1
     0080 00AA'
0114              *
0115 0082 0202  EXPF3   LI R2,FPAC
     0084 005C'
0116 0086 045A          B *R10
0117              *
0118 0088 4117  EXPC0   DATA >4117,>1547,>652C  1.442695  LOG 2 (E)
     008A 1547
     008C 652C
0119 008E 4080  EXPC1   DATA >4080,>0000,>0000  1/2
     0090 0000
     0092 0000
0120 0094 0002  EXPC2   DATA 2
0121 0096 423C          DATA >423C,>9D67,>06A2  60.614853
     0098 9D67
     009A 06A2
0122 009C 4476          DATA >4476,>4EF8,>C12A
     009E 4EF8
     00A0 C12A
0123 00A2 461F          DATA >461F,>BE80,>58C1
     00A4 BE80
     00A6 58C1
0124 00A8 0003  EXPC3   DATA 3
0125 00AA 4110  FP1     DATA >4110,>0000,>0000
     00AC 0000
     00AE 0000
0126 00B0 436D          DATA >436D,>549A,>5FE1
     00B2 549A
     00B4 5FE1
0127 00B6 4550          DATA >4550,>02D2,>6DCF
     00B8 02D2
     00BA 6DCF
0128 00BC 465B          DATA >465B,>9820,>5C39
     00BE 9820
     00C0 5C39
0129 00C2 4116  EXPC4   DATA >4116,>A09E,>667F  1.4142135
     00C4 A09E
     00C6 667F
0130                    END
NO ERRORS,     NO WARNINGS
```

DS	R	004A'	0035	0082	0085	0086	0090	0093				
DS1	R	004E'	0034	0091	0094							
ERR33		0000'	0071	0079								
ERROR		2F80	0027	0028								
ERROR2		2FA0	0028	0071								
EXPC0		0088'	0118	0076								
EXPC1		008E'	0119	0081								
EXPC2		0094'	0120	0084								
EXPC3		00A8'	0124	0088								
EXPC4		00C2'	0129	0095								
EXPF	D	0004'	0073	0069								
EXPF1		006C'	0106	0102								
EXPF2		007E'	0113	0075								
EXPF3		0082'	0115	0107	0111							
FADD			0023									
FDIV			0026									
FMUL			0025									
FP1		00AA'	0125	0109	0113							
FPAC	R	0084'	0032	0099	0115							
FSUB			0024									
FUNBK	R	0012'	0018	0077								
FUNFX	R	0008'	0016	0074								
GETP2	R	006A'	0019	0104								
LOADF			0021									
PLYX	R	0034'	0017	0087								
PLYXX	R	0026'	0017	0083								
R1		0001		0078	0080	0096	0097	0098	0099	0100	0106	0106
R10		000A		0073	0116							
R11		000B		0073								
R2		0002		0080	0096	0101	0103	0115				
R3		0003		0100	0103							
STORE			0022									
TEMP	R	007A'	0033	0089	0092	0108	0110					

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.SINF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.SINF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.SINF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'SINF'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       COSF              ;COSINE FUNCTION
0009          *       SINF              ;SINE FUNCTION
0010          *
0011          * COPY: (NONE)
0012          *
0013          * MARCO: (NONE)
0014          *
0015          * EXTERNAL ROUTINES:
0016          *
0017                REF FUNFX           ;FIX ARGUMENTS
0018                REF FUNBK           ;BREAK ARGUMENTS
0019                REF PLYX,PLYXX      ;EVALUATE POLYNOMIAL
0020          *
0021                DXOP LOADF,0        ;LOAD FPAC
0022                DXOP STORE,1        ;STORE FPAC
0023                DXOP FADD,2         ;ADD TO FPAC
0024                DXOP FSUB,3         ;SUBTRACT FROM FPAC
0025                DXOP FMUL,4         ;MULTIPLY FPAC
0026                DXOP FDIV,5         ;DIVIDE FPAC
0027                DXOP NEGATE,9       ;NEGATE FPAC
0028          *
0029          * EXTERNAL DATA:
0030          *
0031                REF FPAC            ;FLOATING POINT ACCUMULATOR
0032                REF TEMP            ;3 WRD TEMPORARY STORAGE
0033                REF DS,DS1          ;3 WRD TEMPORARY STORAGE
0034          *
0035          * MODULE EQUATES:  (NONE)
0036          *
0037          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0040            * ABSTRACT:
0041            *
0042            *       CALCULATE SINE/COSINE FUNCTIONS OF (ARG).
0043            *
0044            * CALLING SEQUENCE:
0045            *
0046            *       BL @SINF
0047            *       BL @COSF
0048            *
0049            *       IN  (R2) = ARG
0050            *       OUT (R2) = RESULT
0051            *
0052            * EXCEPTIONS AND CONDITIONS:
0053            *
0054            *       FLOATING POINT ERRORS
0055            *
0056            * EXTERNAL ROUTINE LIST:
0057            *
0058            *      (SEE EXTERNAL ROUTINES)
0059            *
0060            * LOCAL DATA:
0061            *
0062            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0064                    * ENTRY POINT:
0065                    *
0066                            DEF SINF,COSF
0067                    *
0068 0000 C28B  COSF    MOV R11,R10
0069 0002 06A0          BL @FUNFX       FIX SIGN
     0004 0000
0070 0006 1003            JMP COSF1     0, RETURN 1
0071 0008 0203          LI R3,1         START WITH 1
     000A 0001
0072 000C 1008          JMP COSF2
0073                    *
0074 000E 2C20  COSF1   LOADF @COSC2    COS 0 = 1
     0010 0068'
0075 0012 1023          JMP COSF5       RETURN
0076                    *
0077 0014 C28B  SINF    MOV R11,R10
0078 0016 06A0          BL @FUNFX       FIX EXPONENT
     0018 0004'
0079 001A 101F            JMP COSF5     ;0, RETURN 0
0080 001C 0A13          SLA R3,1        IF +: 0, ELSE 2
0081                    *
0082 001E 2D20  COSF2   FMUL @COSC0     FPAC=FPAC*C1
     0020 0060'
0083 0022 06A0          BL @FUNBK       BREAK INTO FRAC AND INT
     0024 0000
0084 0026 A043          A R3,R1
0085 0028 0911          SRL R1,1        QUADRANT 2 OR 4?
0086 002A 1706            JNC COSF3     N
0087 002C 2C60          STORE @TEMP     Y, GET 1-FRACTION
     002E 0000
0088 0030 2C20          LOADF @COSC2    LOAD FPAC WITH 1
     0032 0068'
0089 0034 2CE0          FSUB @TEMP      SUBTRACT FRACTION
     0036 002E'
0090                    *
0091 0038 0911  COSF3   SRL R1,1        EFFECTIVE QUAD 3 OR 4?
0092 003A 1701            JNC COSF4     N
0093 003C 2E40          NEGATE 0        Y, NEGATE
0094                    *
0095 003E 2C60  COSF4   STORE @DS       STORE IN DS
     0040 0000
0096 0042 06A0          BL @PLYXX       EVALUATE POLYNOMIAL X*X
     0044 0000
0097 0046 0066'           DATA COSC1
0098 0048 2C60          STORE @DS1      SAVE IN DS1
     004A 0000
0099 004C 06A0          BL @PLYX        EVALUATE POLYNOMIAL X
     004E 0000
0100 0050 007A'           DATA COSC3
0101 0052 2D60          FDIV @DS1       FPAC=FPAC/DS1
     0054 004A'
0102 0056 2D20          FMUL @DS        FPAC=(FPAC/DS1)*DS
     0058 0040'
0103                    *
0104 005A 0202  COSF5   LI R2,FPAC      RETURN
     005C 0000
0105 005E 045A          B *R10
0106                    *
0107 0060 40A2  COSC0   DATA >40A2,>F983,>6E4E
```

```
      0062 F983
      0064 6E4E
 0108 0066 0002  COSC1   DATA 2
 0109 0068 4110  COSC2   DATA >4110,>0000,>0000
      006A 0000
      006C 0000
 0110 006E 4273          DATA >4273,>4DCA,>815D
      0070 4DCA
      0072 815D
 0111 0074 4411          DATA >4411,>7825,>5584
      0076 7825
      0078 5584
 0112 007A 0005  COSC3   DATA 5
 0113 007C BE95          DATA >BE95,>3606,>2DEE
      007E 3606
      0080 2DEE
 0114 0082 4041          DATA >4041,>E3F5,>31B8
      0084 E3F5
      0086 31B8
 0115 0088 C1C6          DATA >C1C6,>5036,>51D0
      008A 5036
      008C 51D0
 0116 008E 4311          DATA >4311,>B7C5,>5A8F
      0090 B7C5
      0092 5A8F
 0117 0094 C3A9          DATA >C3A9,>3BA0,>C828
      0096 3BA0
      0098 C828
 0118 009A 441B          DATA >441B,>70D4,>8BB5
      009C 70D4
      009E 8BB5
 0119                    END
NO ERRORS,     NO WARNINGS
```

COSC0		0060'	0107	0082		
COSC1		0066'	0108	0097		
COSC2		0068'	0109	0074	0088	
COSC3		007A'	0112	0100		
COSF	D	0000'	0068	0066		
COSF1		000E'	0074	0070		
COSF2		001E'	0082	0072		
COSF3		0038'	0091	0086		
COSF4		003E'	0095	0092		
COSF5		005A'	0104	0075	0079	
DS	R	0058'	0033	0095	0102	
DS1	R	0054'	0033	0098	0101	
FADD			0023			
FDIV			0026			
FMUL			0025			
FPAC	R	005C'	0031	0104		
FSUB			0024			
FUNBK	R	0024'	0018	0083		
FUNFX	R	0018'	0017	0069	0078	
LOADF			0021			
NEGATE			0027			
PLYX	R	004E'	0019	0099		
PLYXX	R	0044'	0019	0096		
R1		0001		0084	0085	0091
R10		000A		0068	0077	0105
R11		000B		0068	0077	
R2		0002		0104		
R3		0003		0071	0080	0084
SINF	D	0014'	0077	0066		
STORE			0022			
TEMP	R	0036'	0032	0087	0089	

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.SQRF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.SQRF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.SQRF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'SQRF'            ;SQUARE ROOT FUNCTION
0002               *
0003               * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004               *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005               *
0006               * ROUTINE LIST:
0007               *
0008               *       SQRF                   ;SQUARE ROOT FUNCTION
0009               *
0010               * COPY: (NONE)
0011               *
0012               * MACRO: (NONE)
0013               *
0014               * EXTERNAL ROUTINES:
0015               *
0016                     REF FUNFX                ;FIXES ARGUMENT
0017                     DXOP LOADF,0             ;LOAD FPAC
0018                     DXOP STORE,1             ;STORE FPAC
0019                     DXOP FADD,2              ;ADD TO FPAC
0020                     DXOP FMUL,4              ;MULTIPLY FPAC
0021                     DXOP FDIV,5              ;DIVIDE FPAC
0022      2F80   ERROR  EQU >2F80                 ;XOP XX,14 (ERROR CALL)
0023               *
0024               * EXTERNAL DATA:
0025               *
0026                     REF FPAC                 ;FLOATING POINT ACCUMULATOR
0027                     REF TEMP                 ;3 WRD TEMPORARY STORAGE
0028                     REF DS                   ;3 WRD TEMPORARY STORAGE
0029                     REF EVSFR                ;EXIT ADDRESS
0030               *
0031               * MODULE EQUATES:
0032               *
0033      0004   SQRI   EQU 4                     ;# OF NEWTON ITERATIONS
0034               *
0035               * MODULE VARIABLES AND CONSTANTS:
0036               *
```

```
0039            * ABSTRACT:
0040            *
0041            *       COMPUTE THE SQUARE ROOT OF N (R2) USING A
0042            *       NEWTON ITERATION DEFINED AS FOLLOWS:
0043            *
0044            *            X(I+1) = [X(I) + N / X(I)] / 2
0045            *
0046            *       ITERATE FOR SQRI TIMES.  3 ITERATIONS =
0047            *       6 DIGITS, 4 ITERATION = 11 DIGITS.
0048            *
0049            * CALLING SEQUENCE:
0050            *
0051            *       B @SQRF
0052            *
0053            *       IN  (R0) = 3 WRD FLOATING POINT #
0054            *       OUT (R2) = 3 WRD FLOATING POINT ROOT OF #
0055            *
0056            *       NORMAL EXIT IS TO @EVSFR
0057            *       ERROR EXIT IS THRU XOP XX.14
0058            *
0059            * EXCEPTIONS AND CONDITIONS:
0060            *
0061            *       USES R0-R4,R11
0062            *
0063            *       ERROR 25 = SQUARE ROOT OF NEGATIVE NUMBER
0064            *
0065            * EXTERNAL ROUTINE LIST:
0066            *
0067            *       (SEE EXTERNAL ROUTINES)
0068            *
0069            * LOCAL DATA:
0070            *
0071            *       (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0073                    * ENTRY POINT:
0074                    *
0075        0000'       SQRF    EQU $
0076                            DEF SQRF                ;ENTRY POINT
0077 0000 06A0                  BL @FUNFX               ;MOVE INTO FPAC
     0002 0000
0078 0004 1022                    JMP SQRF3             ;ZERO, DONE
0079 0006 C0C3                  MOV R3,R3               ;NEGATIVE?
0080 0008 1624                    JNE ERR25             ;Y, ERROR
0081                    *******
0082                    * NEWTON'S ITERATION HAS QUADRADIC CONVERGENCE GIVEN
0083                    * A GOOD FIRST GUESS.  PUT FIRST GUESS IN FPAC.
0084                    *******
0085 000A 2C60                  STORE @DS               ;MOVE INTO DS
     000C 0000
0086 000E 0200                  LI R0,>4000             ;GET 16↑0 EXPONENT
     0010 4000
0087 0012 6040                  S R0,R1                 ;UNBIAS OLD EXPONENT
0088 0014 D800                  MOVB R0,@FPAC           ;LOAD EXPONENT
     0016 0000
0089 0018 2D20                  FMUL @SQRC1             ;FPAC=N*C1
     001A 0054'
0090 001C 2CA0                  FADD @SQRC2
     001E 005A'
0091 0020 0991                  SRL R1,9                ;ODD EXPONENT?
0092 0022 1702                    JNC SQRF1             ;N
0093 0024 2D20                  FMUL @FP4               ;Y, MULTIPLY BY 4
     0026 0066'
0094                    *
0095        0028'       SQRF1   EQU $
0096 0028 06C1                  SWPB R1                 ;POSITION EXPONENT
0097 002A B801                  AB R1,@FPAC             ;ADJUST EXPONENT
     002C 0016'
0098 002E 0201                  LI R1,SQRI              ;DO SQRI ITERATIONS
     0030 0004
0099                    *
0100                    *******
0101                    * ITERATION LOOP FOLLOWS.
0102                    * DO X(I+1) = [X(I) + N / X(I)] / 2
0103                    *******
0104        0032'       SQRF2   EQU $
0105 0032 2C60                  STORE @TEMP             ;MOVE FPAC TO TEMP
     0034 0000
0106 0036 2C20                  LOADF @DS
     0038 000C'
0107 003A 2D60                  FDIV @TEMP              ;DIVIDE..N/X(I)
     003C 0034'
0108 003E 2CA0                  FADD @TEMP              ;ADD TEMP..X(I)+N/X(I)
     0040 003C'
0109 0042 2D20                  FMUL @SQRC3             ;X 0.5
     0044 0060'
0110 0046 0601                  DEC R1                  ;SQRI TIMES?
0111 0048 16F4                    JNE SQRF2             ;N
0112                    *******
0113                    * GET ADDRESS OF RESULT AND RETURN TO EVAL
0114                    *******
0115        004A'       SQRF3   EQU $
0116 004A 0202                  LI R2,FPAC              ;Y, RETURN
     004C 002C'
0117 004E 0460                  B @EVSFR
```

```
NTRY POINT
OVE INTO FPAC

ERO, DONE
EGATIVE?
 ERROR

IC CONVERGENCE GIVEN
 GUESS IN FPAC.

OVE INTO DS

IT 16↑0 EXPONENT

BIAS OLD EXPONENT
AD EXPONENT

PAC=N*C1


D EXPONENT?

 MULTIPLY BY 4


SITION EXPONENT
JUST EXPONENT

 SQRI ITERATIONS


 2

VE FPAC TO TEMP


VIDE..N/X(I)

D TEMP..X(I)+N/X(I)

0.5

RI TIMES?


N TO EVAL

 RETURN
```

← S2C @ 58500, *F3

```
      0050 0000
 0118              *
 0119 0052 2F99   ERR25  DATA ERROR+25           ;SQUARE ROOT OF NEGATIVE NUMBE
 0120              *
 0121 0054 40E4   SQRC1  DATA >40E4,>F92A,>F9A8   ;0.894 427
      0056 F92A
      0058 F9A8
 0122 005A 4039   SQRC2  DATA >4039,>3E4E,>F028   ;0.223 607
      005C 3E4E
      005E F028
 0123 0060 4080   SQRC3  DATA >4080,>0000,>0000   ;0.5
      0062 0000
      0064 0000
 0124 0066 4140   FP4    DATA >4140,>0000,>0000   ;4.0
      0068 0000
      006A 0000
 0125                    END
NO ERRORS,     NO WARNINGS
```

$		006C'		0075	0095	0104	0115		
DS	R	0038'	0028	0085	0106				
ERR25		0052'	0119	0080					
ERROR		2F80	0022	0119					
EVSFR	R	0050'	0029	0117					
FADD			0019						
FDIV			0021						
FMUL			0020						
FP4		0066'	0124	0093					
FPAC	R	004C'	0026	0088	0097	0116			
FUNFX	R	0002'	0016	0077					
LOADF			0017						
R0		0000		0086	0087	0088			
R1		0001		0087	0091	0096	0097	0098	0110
R2		0002		0116					
R3		0003		0079	0079				
SQRC1		0054'	0121	0089					
SQRC2		005A'	0122	0090					
SQRC3		0060'	0123	0109					
SQRF	D	0000'	0075	0076					
SQRF1		0028'	0095	0092					
SQRF2		0032'	0104	0111					
SQRF3		004A'	0115	0078					
SQRI		0004	0033	0098					
STORE			0018						
TEMP	R	0040'	0027	0105	0107	0108			

*

```
SOURCE ACCESS NAME=      DBASIC.REL4.BASIC.SOURCE.ATNF
OBJECT ACCESS NAME=      DBASIC.REL4.BASIC.OBJECT.ATNF
LISTING ACCESS NAME=     DBASIC.REL4.BASIC.LIST.ATNF
ERROR ACCESS NAME=
OPTIONS=                 X
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'ATNF'
0002               *    1 MAY 1979
0003               *  EXTERNAL REFERENCES AND DEFINITIONS
0004               *
0005                    DEF  ATNF
0006                    REF  CK302,ATNFE
0007               *
```

```
 0009                *ARC TANGENT FUNCTION
 0010                *
 0011        0000'   ATNF    EQU  $
 0012                *    CHECK TO SEE IF 302 PROMS ARE MOUNTED
 0013  0000 06A0            BL   @CK302
       0002 0000
 0014  0004 0460            B    @ATNFE
       0006 0000
 0015                       END
NO ERRORS
```

```
$        0008'          0011
ATNF   D 0000'   0011   0005
ATNFE  R 0006'   0006   0014
CK302  R 0002'   0006   0013
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.TICF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.TICF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.TICF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'TICF'
0002            *
0003            * REVISION: 12/31/78 1.01      ;SOURCE STANDARDIZATION
0004            *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005            *
0006            * ROUTINE LIST:
0007            *
0008            *       TICF              ;GET CLOCK DELTAS
0009            *       GTIC              ;GET TICKS IN FPAC
0010            *
0011            * COPY: (NONE)
0012            *
0013            * MACRO: (NONE)
0014            *
0015            * EXTERNAL ROUTINES:
0016            *
0017                    REF FNRM                ;FIXES ARGUMENT
0018            *
0019                    DXOP LOADF,0              ;LOAD FPAC
0020                    DXOP STORE,1              ;STORE FPAC
0021                    DXOP FADD,2               ;ADD TO FPAC
0022                    DXOP FSUB,3               ;SUBTRACT FROM FPAC
0023                    DXOP FMUL,4               ;MULTIPLY FPAC
0024                    DXOP FDIV,5               ;DIVIDE FPAC
0025                    DXOP SCALE,6              ;SCALE FPAC
0026                    DXOP NORMAL,7             ;NORMALIZE FPAC
0027                    DXOP CLEAR,8              ;CLEAR FPAC
0028                    DXOP NEGATE,9             ;NEGATE FPAC
0029                    DXOP FLOATF,10            ;FLOAT FPAC
0030                    DXOP EVFIX,11             ;EVALUATE AND FIX
0031                    DXOP OUTFP,12             ;OUT FLOATING POINT #
0032                    DXOP OUTINT,13            ;OUT INTEGER
0033            *
0034            * EXTERNAL DATA:
0035            *
0036                    REF FPWP                  ;FLOATING POINT WORKSPACE PO
0037                    REF FPAC                ;FLOATING POINT ACCUMULATOR
0038                    REF TEMP                ;3 WRD TEMPORARY STORAGE
0039                    REF CLKT01
0040                    REF CLKT02
0041                    REF EVSFR               ;EXIT ADDRESS
0042            *
0043            * MODULE EQUATES (NONE)
0044            *
0045            *
0046            * MODULE VARIABLES AND CONSTANTS:
0047            *
```

```
0050                    * ABSTRACT:
0051                    *
0052                    *       GET AND FLOAT THE NUMBER OF ELAPSED
0053                    *       TICS OF THE CLOCK IN FPAC.
0054                    *
0055                    * CALLING SEQUENCE:
0056                    *
0057                    *       BLWP @GTICZ
0058                    *
0059                    *       OUT FPAC = CLOCK TICS
0060                    *
0061                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0062                    *
0063                    * EXTERNAL ROUTINE LIST:
0064                    *
0065                    *      (SEE EXTERNAL ROUTINES)
0066                    *
0067                    * LOCAL DATA:
0068                    *
0069                    *      (SEE MODULE VARIABLES AND CONSTANTS)
0070                    *
0071                    * ENTRY POINT:
0072                    *
0073                            DEF GTICZ
0074                    *
0075 0000 0000  GTICZ   DATA FPWP,GTIC  ;ACCESS VECTOR
     0002 0004'
0076                    *
0077 0004 0300  GTIC    LIMI 0          ;DIABLE INTERRUPTS
     0006 0000
0078 0008 C060          MOV @CLKT01,R1
     000A 0000
0079 000C C0A0          MOV @CLKT02,R2
     000E 0000
0080 0010 0200          LI R0,>4A00     ;GET EXPONENT
     0012 4A00
0081 0014 0460          B @FNRM         ;NORMALIZE
     0016 0000
```

```
0084            * ABSTRACT:
0085            *
0086            *       GET DELTA TICS.  (SUBTRACT FROM THE
0087            *       FUNCTION ARGUMENT THE NUMBER OF ELAPSED
0088            *       TICS OF THE CLOCK.)
0089            *
0090            * CALLING SEQUENCE:
0091            *
0092            *       B @TICF
0093            *
0094            *       IN (R2) = ARGUMENT
0095            *
0096            *       OUT (R2) = ARG - TICS
0097            *
0098            *       NORMAL EXIT TO EVSFR
0099            *
0100            * EXCEPTIONS AND CONDITIONS (NONE)
0101            *
0102            * EXTERNAL ROUTINE LIST:
0103            *
0104            *       (SEE EXTERNAL ROUTINES)
0105            *
0106            * LOCAL DATA:
0107            *
0108            *       (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0110                    * ENTRY POINT:
0111                    *
0112                            DEF TICF
0113                    *
0114 0018 2C12  TICF    LOADF *R2       ;LOAD FPAC
0115 001A 2E80          FLOATF 0        ;FLOAT IF NECESSARY
0116 001C 2C60          STORE @TEMP     ;STORE IN TEMP
     001E 0000
0117 0020 0420          BLWP @GTICZ     ;GET TICS IN FPAC
     0022 0000'
0118 0024 2CE0          FSUB @TEMP      ;FPAC=FPAC-TEMP
     0026 001E'
0119 0028 0202          LI R2,FPAC      ;RETURN ADDRESS
     002A 0000
0120 002C 0460          B @EVSFR        ;RETURN
     002E 0000
0121                            END
NO ERRORS,      NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES		
CLEAR			0027			
CLKT01	R	000A'	0039	0078		
CLKT02	R	000E'	0040	0079		
EVFIX			0030			
EVSFR	R	002E'	0041	0120		
FADD			0021			
FDIV			0024			
FLOATF			0029			
FMUL			0023			
FNRM	R	0016'	0017	0081		
FPAC	R	002A'	0037	0119		
FPWP	R	0000'	0036	0075		
FSUB			0022			
GTIC		0004'	0077	0075		
GTICZ	D	0000'	0075	0073	0117	
LOADF			0019			
NEGATE			0028			
NORMAL			0026			
OUTFP			0031			
OUTINT			0032			
R0		0000		0080		
R1		0001		0078		
R2		0002		0079	0114	0119
SCALE			0025			
STORE			0020			
TEMP	R	0026'	0038	0116	0118	
TICF	D	0018'	0114	0112		

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.MEMF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.MEMF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.MEMF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'MEMF'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       MEMY              ;MEM COMMAND
0009              *       MEMF              ;MEM FUNCTION
0010              *       ASCF              ;ASC FUNCTION
0011              *
0012              * COPY: (NONE)
0013              *
0014              * MACRO: (NONE)
0015              *
0016              * EXTERNAL ROUTINES:
0017              *
0018                      REF EVERZ         ;EVALUATE EXPRESSION
0019                      REF PFIX          ;FIX 16 BIT NUMBER
0020                      REF GPRM2         ;GET PARAMETERS
0021                      REF NLIN          ;EXIT ENTRY TO MULTIPLEXOR
0022                      REF EVSFR         ;EXIT ENTRY TO EVALUATOR
0023                      DXOP STORE,1      ;STORE FPAC
0024                      DXOP CLEAR,8      ;CLEAR FPAC
0025                      DXOP EVFIX,11     ;EVALUATE AND FIX
0026                      DXOP OUTFP,12     ;OUT FLOATING POINT #
0027    2F80  ERROR       EQU >2F80                 ;XOP XX,14  (ERROR CALL)
0028              *
0029              * EXTERNAL DATA:
0030              *
0031                      REF FPAC,FPAC2    ;FLOATING POINT ACCUMULATOR
0032                      REF B4A           ;>4A
0033              *
0034              * MODULE EQUATES:  (NONE)
0035              *
0036              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0039            * ABSTRACT:
0040            *
0041            *       THE MEM COMMAND ALLOWS PBASIC TO ALTER
0042            *       A BYTE OF MEMORY WHILE THE MEM FUNCTION
0043            *       ALLOWS PBASIC TO READ A BYTE OF
0044            *       MEMORY.
0045            *
0046            * CALLING SEQUENCE:
0047            *
0048            *       B @MEMY
0049            *
0050            *       EXIT TO NLIN
0051            *
0052            *
0053            *       B @MEMF
0054            *
0055            *       IN  (R2) = ADDRESS
0056            *       OUT (R2) = RESULT
0057            *       EXIT TO EVSFR
0058            *
0059            * EXCEPTIONS AND CONDITIONS:
0060            *
0061            *       EVALUATION ERRORS
0062            *       SYNTAX ERROR
0063            *
0064            * EXTERNAL ROUTINE LIST:
0065            *
0066            *      (SEE EXTERNAL ROUTINES)
0067            *
0068            * LOCAL DATA:
0069            *
0070            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0072                    * ENTRY POINT:
0073                    *
0074                            DEF MEMY
0075                    *
0076 0000 9838  MEMY    CB *R8+,@B4A    ;LEFT BRACKET?
     0002 0000
0077 0004 160A            JNE ERR1      ;N, ERROR
0078 0006 0420          BLWP @EVERZ     ;GET ADDRESS
     0008 0000
0079 000A 06A0          BL @PFIX        ;GET INTEGER
     000C 0000
0080 000E 06A0          BL @GPRM2       ;CHECK PARAMETERS
     0010 0000
0081 0012 06C3          SWPB R3         ;LEFT JUSTIFY
0082 0014 D443          MOVB R3,*R1     ;STORE BYTE IN MEMORY
0083 0016 0460          B @NLIN
     0018 0000
0084                    *
0085 001A 2F81  ERR1    DATA ERROR+1
0086                    *
0087                    *READ MEMORY
0088                    *
0089                    * ENTRY POINT:
0090                    *
0091                            DEF MEMF
0092                    *
0093 001C 06A0  MEMF    BL @PFIX        ;GET PARAMETER
     001E 000C'
0094 0020 D051          MOVB *R1,R1     ;GET BYTE
0095 0022 0981  MEMF1   SRL R1,8        ;RIGHT JUSTIFY
0096 0024 2E00          CLEAR 0         ;CLEAR FPAC
0097 0026 C801          MOV R1,@FPAC2   ;SAVE
     0028 0000
0098 002A 0202          LI R2,FPAC      ;GET ADDRESS
     002C 0000
0099 002E 0460          B @EVSFR
     0030 0000
```

```
0102                * ABSTRACT:
0103                *
0104                *       ASC RETURNS THE DECIMAL INTEGER VALUE
0105                *       OF THE FIRST BYTE OF THE ARGUMENT.
0106                *
0107                * CALLING SEQUENCE:
0108                *
0109                *       B @ASCF
0110                *
0111                *       IN  (R2) = ADDRESS
0112                *       OUT (R2) = RESULT
0113                *       EXIT TO EVSFR
0114                *
0115                * EXCEPTIONS AND CONDITIONS:  (NONE)
0116                *
0117                * EXTERNAL ROUTINES:
0118                *
0119                *       REF MEMF1       ;USE MEMF EXIT
0120                *
0121                * LOCAL DATA:  (NONE)
0122                *
0123                * ENTRY POINT:
0124                *
0125                        DEF ASCF
0126                *
0127      0032' ASCF    EQU  $
0128 0032 D052          MOVB *R2,R1
0129 0034 10F6          JMP  MEMF1
0130                     END
NO ERRORS,     NO WARNINGS
```

$		0036'		0127					
ASCF	D	0032'	0127	0125					
B4A	R	0002'	0032	0076					
CLEAR			0024						
ERR1		001A'	0085	0077					
ERROR		2F80	0027	0085					
EVERZ	R	0008'	0018	0078					
EVFIX			0025						
EVSFR	R	0030'	0022	0099					
FPAC	R	002C'	0031	0098					
FPAC2	R	0028'	0031	0097					
GPRM2	R	0010'	0020	0080					
MEMF	D	001C'	0093	0091					
MEMF1		0022'	0095	0129					
MEMY	D	0000'	0076	0074					
NLIN	R	0018'	0021	0083					
OUTFP			0026						
PFIX	R	001E'	0019	0079	0093				
R1		0001		0082	0094	0094	0095	0097	0128
R2		0002		0098	0128				
R3		0003		0081	0082				
R8		0008		0076					
STORE			0023						

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.BITF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.BITF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.BITF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
                IDT  'BITF'
0002                 *
0003                 * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                 *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                 *
0006                 * ROUTINE LIST:
0007                 *
0008                 *       BITF              ;BIT FUNCTION
0009                 *       BITY              ;BIT COMMAND
0010                 *
0011                 * COPY: (NONE)
0012                 *
0013                 * MARCO: (NONE)
0014                 *
0015                 * EXTERNAL ROUTINES:
0016                 *
0017                       REF EVSFR           ;EXIT TO EVALUATOR
0018                       REF NLIN            ;EXIT TO MULTIPLEXOR
0019                       REF EVERZ           ;EVALUATE EXPRESSION
0020                       REF GPRM1           ;GET-PARAMETER ENTRY
0021                       DXOP STORE,1        ;STORE FPAC
0022                       DXOP EVFIX,11       ;EVALUATE AND FIX
0023                       DXOP OUTFP,12       ;OUT FLOATING POINT #
0024          2F80  ERROR  EQU >2F80           ;XOP XX,14  (ERROR CALL)
0025                 *
0026                 * EXTERNAL DATA:
0027                 *
0028                       REF FPAC,FPAC2 ;FLOATING POINT ACCUMULATOR
0029                 *
0030                 * MODULE EQUATES:  (NONE)
0031                 *
0032                 * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0035                * ABSTRACT:
0036                *
0037                *       THE BIT COMMAND ALLOWS A PBASIC STATEMENT
0038                *       TO ALTER ANY BIT WITHIN A PBASIC
0039                *       VARIABLE.  THE FORM IS:
0040                *
0041                *       BIT[ <VAR> , <EXP1> ] = <EXP2>
0042                *
0043                *       WHERE:  <VAR>  = VARIABLE TO BE ALTERED
0044                *               <EXP1> = BIT POSITION WITHIN VARIABLE
0045                *               <EXP2> = 0 OR <>0 TO RESET OR SET
0046                *
0047                * CALLING SEQUENCE:
0048                *
0049                *       B @BITY
0050                *
0051                *       EXIT TO NLIN
0052                *
0053                * EXCEPTIONS AND CONDITIONS:
0054                *
0055                *       EVALUATION ERRORS
0056                *       ERROR 1 IF BRACKETS MISSING OR NO ",
0057                *
0058                * EXTERNAL ROUTINE LIST:
0059                *
0060                *       NLIN            ;EXIT TO MULTIPLEXOR
0061                *
0062                * LOCAL DATA:
0063                *
0064                *     (SEE MODULE VARIABLES AND CONSTANTS)
0065                *
0066                * ENTRY POINT:
0067                *
0068                        DEF BITY
0069                *
0070 0000 9838  BITY    CB *R8+,@B4A    ;RIGHT BRACKET?
     0002 0030'
0071 0004 1614            JNE ERR1      ;N, ERROR
0072 0006 0420          BLWP @EVERZ      ;GET ADR
     0008 0000
0073 000A 0280          CI R0,>3F00       ;,?
     000C 3F00
0074 000E 160F            JNE ERR1        ;N
0075 0010 06A0          BL @GPRM1       ;GET PARAMETERS
     0012 0000
0076 0014 0601          DEC R1
0077 0016 C001          MOV R1,R0       ;GET WORD INDEX
0078 0018 0931          SRL R1,3
0079 001A A081          A R1,R2         ;INDEX
0080 001C 0201          LI R1,>8000
     001E 8000
0081 0020 0B01          SRC R1,0        ;POSITION MASK
0082 0022 4481          SZC R1,*R2      ;SET BIT TO ZERO
0083 0024 C0C3          MOV R3,R3       ;NEED 1?
0084 0026 1301            JEQ BITY1     ;N
0085 0028 E481          SOC R1,*R2      ;Y, SET BIT TO ONE
0086 002A 0460  BITY1   B @NLIN         ;RETURN
     002C 0000
0087 002E 2F81  ERR1    DATA ERROR+1
0088 0030   4A  B4A     BYTE >4A
```

0089 0032 EVEN

```
0092                    * ABSTRACT:
0093                    *
0094                    *       THE BIT FUNCTION WILL DISPLAY ANY
0095                    *       BIT VALUE WITHIN A PBASIC VARIABLE.
0096                    *
0097                    *       BIT[ <EXP1> , <EXP> ]
0098                    *
0099                    *       WHERE   <EXP1> = PBASIC VARIABLE
0100                    *               <EXP2> = BIT POSITION
0101                    *
0102                    * CALLING SEQUENCE:
0103                    *
0104                    *       B @BITF
0105                    *
0106                    *       EXIT TO EVSFR
0107                    *
0108                    * EXCEPTIONS AND CONDITIONS:  (NONE)
0109                    *
0110                    * EXTERNAL ROUTINE LIST:
0111                    *
0112                    * LOCAL DATA:
0113                    *
0114                    * ENTRY POINT:
0115                    *
0116                            DEF BITF
0117                    *
0118 0032 0601          BITF    DEC R1
0119 0034 C001                  MOV R1,R0
0120 0036 0931                  SRL R1,3            ;GET WORD INDEX
0121 0038 A081                  A R1,R2             ;INDEX
0122 003A 0201                  LI R1,>8000         ;GET INITIAL MASK
     003C 8000
0123 003E 0B01                  SRC R1,0            ;SHIFT
0124 0040 C092                  MOV *R2,R2          ;GET WORD
0125 0042 2081                  COC R1,R2           ;BIT SET?
0126 0044 1602                    JNE BITF1         ;N, RETURN 0
0127 0046 05A0                  INC @FPAC2          ;Y, RETURN 1
     0048 0000
0128                    *
0129 004A 0202          BITF1   LI R2,FPAC          ;RETURN ADR
     004C 0000
0130 004E 0460                  B @EVSFR
     0050 0000
0131                            END
NO ERRORS,      NO WARNINGS
```

B4A		0030'	0088	0070							
BITF	D	0032'	0118	0116							
BITF1		004A'	0129	0126							
BITY	D	0000'	0070	0068							
BITY1		002A'	0086	0084							
ERR1		002E'	0087	0071	0074						
ERROR		2F80	0024	0087							
EVERZ	R	0008'	0019	0072							
EVFIX			0022								
EVSFR	R	0050'	0017	0130							
FPAC	R	004C'	0028	0129							
FPAC2	R	0048'	0028	0127							
GPRM1	R	0012'	0020	0075							
NLIN	R	002C'	0018	0086							
OUTFP			0023								
R0		0000		0073	0077	0119					
R1		0001		0076	0077	0078	0079	0080	0081	0082	0085	0118
				0119	0120	0121	0122	0123	0125			
R2		0002		0079	0082	0085	0121	0124	0124	0125	0129	
R3		0003		0083	0083							
R8		0008		0070								
STORE			0021									

*

SOURCE ACCESS NAME= BD1.DBASIC.REL5.BASIC.SOURCE.CRUF
OBJECT ACCESS NAME= BD1.DBASIC.REL5.BASIC.OBJECT.CRUF
LISTING ACCESS NAME= BD1.DBASIC.REL5.BASIC.LIST.CRUF
ERROR ACCESS NAME=
OPTIONS= XREF
MACRO LIBRARY PATHNAME=

0001

```
                IDT 'CRUF'
0002                    *
0003                    * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                    *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                    *
0006                    * ROUTINE LIST:
0007                    *
0008                    *       CRBY              ;CRB COMMAND
0009                    *       CRBF              ;CRB FUNCTION
0010                    *       CRFY              ;CRF COMMAND
0011                    *       CRFF              ;CRF FUNCTION
0012                    *
0013                    * COPY: (NONE)
0014                    *
0015                    * MARCO: (NONE)
0016                    *
0017                    * EXTERNAL ROUTINES:
0018                    *
0019                            REF FIX           ;INTEGER FIX
0020                            REF EVSFR         ;ENTRY EXIT TO EVALUATOR
0021                            REF NLIN          ;ENTRY EXIT TO MULTIPLEXOR
0022                            REF GPRM          ;GET PARAMETER
0023                            DXOP OUTINT,13            ;OUT INTEGER
0024            2F80    ERROR   EQU >2F80         ;XOP XX,14  (ERROR CALL)
0025                    *
0026                    * EXTERNAL DATA:
0027                    *
0028                            REF FPAC,FPAC2    ;FLOATING POINT ACCUMULATOR
0029                            REF BCRU          ;BASE CRU INDEX
0030                    *
0031                    * MODULE EQUATES:  (NONE)
0032                    *
0033                    * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0036                    * ABSTRACT:
0037                    *
0038                    *       STORE MULTIPLY BIT TO CRU LINES AT
0039                    *       CRU BASE AS SELECTED BY BASE COMMAND.
0040                    *
0041                    * CALLING SEQUENCE:
0042                    *
0043                    *       B @CRFY
0044                    *
0045                    *       EXIT TO NLIN
0046                    *
0047                    * EXCEPTIONS AND CONDITIONS:
0048                    *
0049                    *       PARAMETER AND EVALUATION ERRORS
0050                    *
0051                    * EXTERNAL ROUTINE LIST:
0052                    *
0053                    *      (SEE EXTERNAL ROUTINES)
0054                    *
0055                    * LOCAL DATA:
0056                    *
0057                    *      (SEE MODULE VARIABLES AND CONSTANTS)
0058                    *
0059                    * ENTRY POINT:
0060                    *
0061                            DEF CRFY
0062                    *
0063 0000 06A0  CRFY    BL @GPRM        ;GET PARAMETERS
     0002 0000
0064 0004 0241          ANDI R1,>F      ;MASK COUNT
     0006 000F
0065 0008 1304            JEQ CRF1      ;0=16 BITS
0066 000A 0281          CI R1,8         ;BYTE?
     000C 0008
0067 000E 1501            JGT CRF1      ;N
0068 0010 06C3          SWPB R3         ;Y
0069                    *
0070 0012 0A61  CRF1    SLA R1,6        ;POSITION
0071 0014 0221          AI R1,>3003     ;MAKE 'LDCR R3,X'
     0016 3003
0072 0018 100A          JMP CRBY1
```

```
0075                    * ABSTRACT:
0076                    *
0077                    *       STORE SINGLE CRU BIT AS A DISPLACEMENT
0078                    *       FROM CRU BASE AS SPECIFIED BY
0079                    *       BASE COMMAND.
0080                    *
0081                    * CALLING SEQUENCE:
0082                    *
0083                    *       B @CRBY
0084                    *
0085                    *       EXIT TO NLIN
0086                    *
0087                    * EXCEPTIONS AND CONDITIONS:
0088                    *
0089                    *       EVALUATION AND PARAMETER ERRORS
0090                    *
0091                    * EXTERNAL ROUTINE LIST:
0092                    *
0093                    *       (SEE EXTERNAL ROUTINES)
0094                    *
0095                    * LOCAL DATA:  (NONE)
0096                    *
0097                    * ENTRY POINT
0098                    *
0099                            DEF CRBY
0100                    *
0101 001A 06A0  CRBY    BL @GPRM          ;GET DISPLACEMENT
     001C 0002'
0102 001E 0241          ANDI R1,>FF       ;MASK
     0020 00FF
0103 0022 0221          AI R1,>1D00       ;GET SBO
     0024 1D00
0104 0026 C0C3          MOV R3,R3         ;SET TO 1?
0105 0028 1602            JNE CRBY1       ;N
0106 002A 0221          AI R1,>0100       ;Y, CHANGE TO SBZ
     002C 0100
0107                    *
0108 002E C329  CRBY1   MOV @BCRU(9),R12
     0030 0000
0109 0032 0481          X R1              ;DO CRU INSTRUCTION
0110 0034 0460          B @NLIN           ;RETURN
     0036 0000
```

```
0113                * ABSTRACT:
0114                *
0115                *       READ FROM CRU LINES THE ARGUMENT NUMBER OF
0116                *       BITS.
0117                *
0118                * CALLING SEQUENCE:
0119                *
0120                *       B @CRFF
0121                *
0122                *       EXIT TO EVSFR
0123                *
0124                * EXCEPTIONS AND CONDITIONS:
0125                *
0126                *       FIX ERROR
0127                *
0128                * EXTERNAL ROUTINE LIST:
0129                *
0130                *       (SEE EXTERNAL ROUTINES)
0131                *
0132                * LOCAL DATA:  (NONE)
0133                *
0134                * ENTRY POINT:
0135                *
0136                        DEF CRFF
0137                *
0138 0038 06A0  CRFF    BL @FIX             ;GET PARAMETER
     003A 0000
  39 003C 0241          ANDI R1,>F          ;MASK
     003E 000F
0 40 0040 C001          MOV R1,R0           ;SAVE
0 41 0042 0A61          SLA R1,6            ;POSITION
0  2 0044 0221          AI R1,>3401         ;MAKE 'STCR R1,X'
     0046 3401
0143 0048 C329          MOV @BCRU(9),R12    ;GET BASE
     004A 0030'
014  004C 0481          X R1                ;DO STCR
014  004E C000          MOV R0,R0           ;16?
0146 0050 1304            JEQ CRFF1         ;Y
0147 0052 0280          CI R0,8             ;N, BYTE?
     0054 0008
0148  056 1501            JGT CRFF1         ;N
0149 0058 0981          SRL R1,8            ;Y, POSITION RESULTS
0150                *
0151 005A C801  CRFF1   MOV R1,@FPAC2
     005C 0000
0152 005E 0202  CRFF2   LI R2,FPAC          ;RETURN ADR
     0060 0000
0153 0062 0460          B @EVSFR
     0064 0000
```

GUMENT NUMBER OF

METER

CR R1,X'

Routine called via EVAL module
(if EVFIX, R12 contains address of EVFIX's parameter)

← MOV R12,R2

ION RESULTS

OR

```
CEFF2   MOV   R2,R12.     see above
        LI    R3,FFAC
        B     @EVSFR
```

```
0156                * ABSTRACT:
0157                *
0158                *       READ 1 CRU BIT GIVEN BY THE CRU BASE
0159                *       AND THE ARGUMENT DISPLACEMENT.
0160                *
0161                * CALLING SEQUENCE:
0162                *
0163                *       B @CRBF
0164                *
0165                *       EXIT TO EVSFR
0166                *
0167                * EXCEPTIONS AND CONDITIONS:
0168                *
0169                *       FIX ERROR
0170                *
0171                * EXTERNAL ROUTINE LIST:
0172                *
0173                *       (SEE EXTERNAL ROUTINES)
0174                *
0175                * LOCAL DATA:  (NONE)
0176                *
0177                * ENTRY POINT:
0178                *
0179                        DEF CRBF
0180                *
0181 0066 06A0  CRBF    BL @FIX             ;GET DISPLACEMENT
     0068 003A'
0182 006A 0241          ANDI R1,>FF         ;MASK
     006C 00FF
0183 006E 0221          AI R1,>1F00         ;MAKE 'TB XX'
     0070 1F00
0184 0072 C329          MOV @BCRU(9),R12    ;GET BASE
     0074 004A'
0185 0076 0481          X R1                ;DO TEST
0186 0078 16F2            JNE CRFF2         ;0, RT
0187 007A 05A0          INC @FPAC2          ;1
     007C 005C'
0188 007E 10EF          JMP CRFF2
0189                    END
NO ERRORS,     NO WARNINGS
```

CRU BASE
NT.

ACEMENT

XX'

MOV R12, R2. see comment over page.

LABEL		VALUE	DEFN	REFERENCES								
BCRU	R	0074'	0029	0108	0143	0184						
CRBF	D	0066'	0181	0179								
CRBY	D	001A'	0101	0099								
CRBY1		002E'	0108	0072	0105							
CRF1		0012'	0070	0065	0067							
CRFF	D	0038'	0138	0136								
CRFF1		005A'	0151	0146	0148							
CRFF2		005E'	0152	0186	0188							
CRFY	D	0000'	0063	0061								
ERROR		2F80	0024									
EVSFR	R	0064'	0020	0153								
FIX	R	0068'	0019	0138	0181							
FPAC	R	0060'	0028	0152								
FPAC2	R	007C'	0028	0151	0187							
GPRM	R	001C'	0022	0063	0101							
NLIN	R	0036'	0021	0110								
OUTINT			0023									
R0		0000		0140	0145	0145	0147					
R1		0001		0064	0066	0070	0071	0102	0103	0106	0109	0139
				0140	0141	0142	0144	0149	0151	0182	0183	0185
R12		000C		0108	0143	0184						
R2		0002		0152								
R3		0003		0068	0104	0104						

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.GETPARM
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.GETPARM
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.GETPARM
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'GETPARM'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       GPRM,GPRM1,GPRM2         ;GET PARAMETER
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MARCO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:
0015              *
0016                      DXOP EVFIX,11    ;EVALUATE AND FIX
0017     2F80     ERROR    EQU >2F80        ;XOP XX,14  (ERROR CALL)
0018     2FA0     ERROR2   EQU ERROR+>20
0019              *
0020              * EXTERNAL DATA:  (NONE)
0021              *
0022              * MODULE EXTERNALS:
0023              *
0024                      DEF B4A          ;>4A
0025              *
0026              * MODULE EQUATES:  (NONE)
0027              *
0028              *
0029              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0032                * ABSTRACT:
0033                *
0034                *       FIX PARAMETERS SUCH THAT:
0035                *
0036                *         XXX[ (R2) , R1 ] = R3
0037                *
0038                *       DIFFERENT ENTRY POINTS WILL PICK UP
0039                *       THE EVALUATION AND CHECKING AT
0040                *       DIFFERENT POINTS.
0041                *
0042                * CALLING SEQUENCE:
0043                *
0044                *       BL @GPRM
0045                *       BL @GPRM1
0046                *       BL @GPRM2
0047                *
0048                *       IN (R2) = VARIABLE
0049                *       OUT R1  = INDEX
0050                *           R3  = ASSIGNMENT FIXED
0051                *
0052                * EXCEPTIONS AND CONDITIONS:
0053                *
0054                *       EVALUATION ERRORS, FIX ERRORS, AND
0055                *       EXPECTING OPERATOR.
0056                *
0057                * EXTERNAL ROUTINE LIST:
0058                *
0059                *      (SEE EXTERNAL ROUTINES)
0060                *
0061                * LOCAL DATA:
0062                *
0063                *      (SEE MODULE VARIABLES AND CONSTANTS)
0064                *
0065                * ENTRY POINT:
0066                *
0067                       DEF GPRM,GPRM1,GPRM2
0068                *
0069 0000 9838  GPRM    CB *R8+,@B4A    ;RIGHT BRACKET?
     0002 001E'
0070 0004 1609          JNE ERR1        ;N, ERROR
0071                *
0072 0006 2EC1  GPRM1   EVFIX R1        ;GET INDEX
0073                *
0074 0008 0280  GPRM2   CI R0,>4B00     ;]?
     000A 4B00
0075 000C 1605          JNE ERR1        ;N
0076 000E 9E20          CB @B56,*R8+    ;=?
     0010 001F'
0077 0012 1603          JNE ERR36       ;N
0078 0014 2EC3          EVFIX R3        ;GET RESULT
0079 0016 045B          RT
0080                *
0081 0018 2F81  ERR1    DATA ERROR+1
0082 001A 2FA0  ERR36   DATA ERROR2,36
     001C 0024
0083 001E   4A  B4A     BYTE >4A
0084 001F   56  B56     BYTE >56
0085                    END
NO ERRORS,      NO WARNINGS
```

```
B4A     D  001E'   0083  0024  0069
B56        001F'   0084  0076
ERR1       0018'   0081  0070  0075
ERR36      001A'   0082  0077
ERROR      2F80    0017  0018  0081
ERROR2     2FA0    0018  0082
EVFIX              0016
GPRM    D  0000'   0069  0067
GPRM1   D  0006'   0072  0067
GPRM2   D  0008'   0074  0067
R0         0000           0074
R1         0001           0072
R3         0003           0078
R8         0008           0069  0076
*
```

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.CHRF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.CHRF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.CHRF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'CHRF'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       SRHF              ;STRING SEARCH FUNCTION
0009              *       MCHF              ;STRING MATCH FUNCTION
0010              *       LENF              ;STRING LENGTH FUNCTION
0011              *
0012              * COPY: (NONE)
0013              *
0014              * MARCO: (NONE)
0015              *
0016              * EXTERNAL ROUTINES:
0017              *
0018                    REF EVSFR         ;EXIT ENTRY TO EVALUATOR
0019              *
0020              * EXTERNAL DATA:
0021              *
0022                    REF FPAC,FPAC2    ;FLOATING POINT ACCUMULATOR
0023              *
0024              * MODULE EQUATES:  (NONE)
0025              *
0026              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0029                * ABSTRACT:
0030                *
0031                *       SRH WILL SEARCH FOR THE FIRST STRING
0032                *       IN THE SECOND STRING AND RETURN THE
0033                *       STARTING CHARACTER POSITION OF THE
0034                *       FIRST MATCH IF FOUND.  OTHERWISE RETURN
0035                *       A 0.
0036                *
0037                * CALLING SEQUENCE:
0038                *
0039                *       B @SRHF
0040                *
0041                *       EXIT TO EVSFR
0042                *
0043                * EXCEPTIONS AND CONDITIONS:  (NONE)
0044                *
0045                * EXTERNAL ROUTINE LIST:
0046                *
0047                *      (SEE EXTERNAL ROUTINES)
0048                *
0049                * LOCAL DATA:
0050                *
0051                *      (SEE MODULE VARIABLES AND CONSTANTS)
0052                *
0053                * ENTRY POINT:
0054                *
0055                        DEF SRHF
0056                *
0057 0000 05A0     SRHF     INC @FPAC2          ;COUNT
     0002 0000
0058 0004 D031             MOVB *R1+,R0        ;GET FIRST BYTE
0059 0006 130A               JEQ SRHF2         ;NOT FOUND, RETURN 0
0060 0008 9012             CB *R2,R0           ;SAME?
0061 000A 16FA               JNE SRHF          ;N, CONTINUE LOOKING
0062 000C C0C2             MOV R2,R3           ;Y, LOOK FURTHER
0063 000E 0583             INC R3
0064 0010 C101             MOV R1,R4
0065                *
0066 0012 D033     SRHF1    MOVB *R3+,R0        ;MATCH?
0067 0014 1305               JEQ SRHF3         ;Y, RETURN FPAC
0068 0016 9D00             CB R0,*R4+          ;SAME?
0069 0018 13FC               JEQ SRHF1         ;Y, CONTINUE
0070 001A 10F2             JMP SRHF            ;N, START AGAIN
0071                *
0072 001C 04E0     SRHF2    CLR @FPAC2
     001E 0002'
0073 0020 0202     SRHF3    LI R2,FPAC
     0022 0000
0074 0024 0460             B @EVSFR
     0026 0000
```

```
0077                *  ABSTRACT:
0078                *
0079                *        MATCH STRING 1 INTO STRING 2 AND RETURN
0080                *        THE NUMBER OF CHARACTERS TO WHICH THEY
0081                *        AGREE.
0082                *
0083                *  CALLING SEQUENCE:
0084                *
0085                *        B @MCHF
0086                *
0087                *        EXIT TO EVSFR
0088                *
0089                *  EXCEPTIONS AND CONDITIONS:  (NONE)
0090                *
0091                *  EXTERNAL FUNCTION LIST:  (NONE)
0092                *
0093                *  LOCAL DATA:  (NONE)
0094                *
0095                *  ENTRY POINT:
0096                *
0097                         DEF MCHF
0098                *
0099                *
0100 0028 9C72  MCHF     CB *R2+,*R1+     ;SAME?
0101 002A 16FA             JNE SRHF3      ;N
0102 002C 05A0           INC @FPAC2       ;Y, COUNT
     002E 001E'
0103 0030 10FB           JMP MCHF
```

AND RETURN
HICH THEY

CB *R3, *R1+
JNE
~~JMP~~ MOV *R2, ~~*R2+~~
JEQ ~~ERR~~ SRHF3

```
0106                * ABSTRACT:
0107                *
0108                *       RETURN THE STRING LENGTH (THE FIRST
0109                *       NULL BYTE.
0110                *
0111                * CALLING SEQUENCE:
0112                *
0113                *       B @LENF
0114                *
0115                *       EXIT TO EVSFR
0116                *
0117                * EXCEPTIONS AND CONDITIONS:  (NONE)
0118                *
0119                * EXTERNAL ROUTINE LIST:  (NONE)
0120                *
0121                * LOCAL DATA: (NONE)
0122                *
0123                * ENTRY POINT:
0124                *
0125                        DEF LENF
0126                *
0127                *
0128 0032 D072  LENF    MOVB *R2+,R1
0129 0034 13F5            JEQ SRHF3     ;DONE
0130 0036 05A0          INC @FPAC2      ;COUNT
     0038 002E'
0131 003A 10FB          JMP LENF
0132                *
0133                    END
NO ERRORS,     NO WARNINGS
```

EVSFR	R	0026'	0018	0074				
FPAC	R	0022'	0022	0073				
FPAC2	R	0038'	0022	0057	0072	0102	0130	
LENF	D	0032'	0128	0125	0131			
MCHF	D	0028'	0100	0097	0103			
R0		0000		0058	0060	0066	0068	
R1		0001		0058	0064	0100	0128	
R2		0002		0060	0062	0073	0100	0128
R3		0003		0062	0063	0066		
R4		0004		0064	0068			
SRHF	D	0000'	0057	0055	0061	0070		
SRHF1		0012'	0066	0069				
SRHF2		001C'	0072	0059				
SRHF3		0020'	0073	0067	0101	0129		

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.PUTB
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.PUTB
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.PUTB
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT  'PUTB'
0002          *
0003          * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004          *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005          *
0006          * ROUTINE LIST:
0007          *
0008          *       PUTB              ;PUT BYTE
0009          *
0010          * COPY: (NONE)
0011          *
0012          * MARCO: (NONE)
0013          *
0014          * EXTERNAL ROUTINES:  (NONE)
0015          *
0016          * EXTERNAL DATA:
0017          *
0018                REF UFT             ;USER FUNCTION TABLE
0019                REF PLF             ;PROGRAM LOAD FLAG
0020                REF CCNT            ;COLUMN COUNTER
0021          *
0022          * MODULE EQUATES:  (NONE)
0023          *
0024          * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0027            * ABSTRACT:
0028            *
0029            *       PUTB TRANSFERS LEFT BYTE IN R0 INTO
0030            *       BUFFER (R7).  A CHECK IS MADE FOR
0031            *       CARRIAGE RETURN AND CONTROL CHARACTERS.
0032            *
0033            * CALLING SEQUENCE:
0034            *
0035            *       BL @PUTB
0036            *         CARRIAGE RETURN
0037            *         CONTROL CHARACTER
0038            *       CHARACTER
0039            *
0040            *       IN   (R7) = BUFFER
0041            *            R0   = CHARACTER
0042            *       OUT  R0   = CHARACTER OR BACKSPACE OR BELL
0043            *
0044            * EXCEPTIONS AND CONDITIONS:  (NONE)
0045            *
0046            * EXTERNAL ROUTINE LIST:
0047            *
0048            *      (SEE EXTERNAL ROUTINES)
0049            *
0050            * LOCAL DATA:
0051            *
0052            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0054                    * ENTRY POINT:
0055                    *
0056                          DEF PUTB
0057                    *
0058 0000 0240  PUTB    ANDI R0,>7F00   ;REMOVE PARITY
     0002 7F00
0059 0004 0280          CI R0,>0D00     ;CR?
     0006 0D00
0060 0008 1310            JEQ PUTB1     ;Y
0061 000A 0280          CI R0,>2000     ;<BLK?
     000C 2000
0062 000E 1A1D            JL PUTB6      ;Y, IGNOR
0063 0010 0280          CI R0,>7F00     ;RUBOUT?
     0012 7F00
0064 0014 130C            JEQ PUTB2     ;Y
0065 0016 8A47          C R7,@UFT(9)    ;BUFFER FULL?
     0018 0000
0066 001A 140C            JHE PUTB3     ;Y
0067 001C C069          MOV @PLF(9),R1  ;N, LOADING?
     001E 0000
0068 0020 1602            JNE PUTB0     ;Y
0069 0022 05A9          INC @CCNT(9)    ;N, INCREMENT COLUMN COUNTER
     0024 0000
0070                    *
0071 0026 DDC0  PUTB0   MOVB R0,*R7+    ;STORE BYTE
0072 0028 100E          JMP PUTB5       ;B @4(11)
0073                    *
0074 002A 75D7  PUTB1   SB *R7,*R7      ;CR
0075 002C 045B          RT
0076                    *
0077 002E 0584  PUTB2   INC R4          ;ALLOW 1 MORE CHARACTER
0078 0030 8647          C R7,*R9        ;RUBOUT, EMPTY?
0079 0032 1B03            JH PUTB4      ;N
0080 0034 0200  PUTB3   LI R0,>0700     ;Y, OUT BELL
     0036 0700
0081 0038 1006          JMP PUTB5
0082                    *
0083 003A 0607  PUTB4   DEC R7          ;BACKUP
0084 003C 0584          INC R4          ;ALLOW CHARACTER
0085 003E 0629          DEC @CCNT(9)
     0040 0024'
0086 0042 0200          LI R0,>0800     ;OUT BS
     0044 0800
0087 0046 046B  PUTB5   B @4(11)
     0048 0004
0088                    *
0089 004A 046B  PUTB6   B @2(11)
     004C 0002
0090                    END
NO ERRORS,     NO WARNINGS
```

```
LABEL     VALUE   DEFN  REFERENCES

CCNT   R  0040'   0020  0069  0085
PLF    R  001E'   0019  0067
PUTB   D  0000'   0058  0056
PUTB0     0026'   0071  0068
PUTB1     002A'   0074  0060
PUTB2     002E'   0077  0064
PUTB3     0034'   0080  0066
PUTB4     003A'   0083  0079
PUTB5     0046'   0087  0072  0081
PUTB6     004A'   0089  0062
R0        0000          0058  0059  0061  0063  0071  0080  0086
R1        0001          0067
R4        0004          0077  0084
R7        0007          0065  0071  0074  0074  0078  0083
R9        0009          0078
UFT    R  0018'   0018  0065
*
```

SOURCE ACCESS NAME= DBASIC.REL4.BASIC.SOURCE.GETLINE
OBJECT ACCESS NAME= DBASIC.REL4.BASIC.OBJECT.GETLINE
LISTING ACCESS NAME= DBASIC.REL4.BASIC.LIST.GETLINE
ERROR ACCESS NAME=
OPTIONS=
MACRO LIBRARY PATHNAME=

```
0001                    IDT 'GETLINE'
0002              *     23 APRIL 1979
0003              *
0004                    DEF GTLN
0005                    DEF EDER
0006                    REF AINC
0007                    REF B20
0008                    REF CRLF1
0009                    REF CVDIZ
0010                    REF DCNT
0011                    REF ERM1
0012                    REF GETCR
0013                    REF ID302
0014                    REF JMPR0
0015                    REF LDP5
0016                    REF LNUM
0017                    REF LSTL
0018                    REF NIC
0019                    REF PLF
0020                    REF SFSN
0021                    REF TYP0
0022                    REF TYP11
0023                    REF TYPB
0024                    REF TYPC
0025                    REF TYPE
0026                    REF ERRLS2
0027      2F80  ERROR   EQU >2F80
0028                    DXOP OUTINT,13
0029              *      08/10/78
0030              *↑E
0031              *
0032 0000 C1D9  GTCE    MOV *R9,R7        ;GET BUFFER ADR
0033 0002 0420          BLWP @CVDIZ       ;CONVERT #
     0004 0000
0034 0006 1000            NOP
0035 0008 2F8D            DATA ERROR+13   ;NO SUCH LINE
0036 000A 06A0          BL @SFSN          ;SEARCH FOR STATEMENT #
     000C 0000
0037 000E C1D9          MOV *R9,R7        ;RESET BUFFER ADR
0038 0010 0729          SETO @DCNT(9)     ;RESET INDENT COUNTER
     0012 0000
0039 0014 06A0          BL @LSTL          ;LIST LINE
     0016 0000
0040 0018 06A0          BL @TYPC          ;OUT CRLF
     001A 0000
0041 001C 1013          JMP GTCE2
0042              *
0043              *GET INPUT LINE
0044              *       R6 = MAX # OF CHARACTERS
0045              *       R7 = I/O POINTER
0046              *       R8 = RETURN ADR
0047              *
0048 001E 0206  GTLN    LI R6,NIC         ;MAXIMUM OF NIC CHARACTERS
     0020 0000
0049 0022 C1D9          MOV *R9,R7        ;GET IOB PTR
0050 0024 C0C7          MOV R7,R3
0051 0026 C046          MOV R6,R1
0052              *
0053 0028 04F3          CLR *R3+          ;CLEAR BUFFER
0054 002A 0641          DECT R1           ;DONE?
```

```
0055 002C 15FD            JGT $-4          ;N
0056                 *
0057 002E 0769            ABS @AINC(9)     ;AUTO-INCREMENT?
     0030 0000
0058 0032 150E              JGT GTLLP      ;N
0059 0034 130D              JEQ GTLLP      ;N
0060 0036 C069            MOV @LNUM(9),R1  ;ADD LINE #
     0038 0000
0061 003A 0221            AI R1,10
     003C 000A
0062 003E 2F41            OUTINT R1        ;CONVERT
0063 0040 DDE0            MOVB @B20,*R7+   ;SPACE
     0042 0000
0064                 *
0065 0044 06A0  GTCE2     BL @TYPB         ;TYPE
     0046 0000
0066 0048 0206            LI R6,NIC        ;GET NEW COUNT
     004A 0020'
0067 004C A199            A *R9,R6         ;ADD BUFFER ADR
0068 004E 6187            S R7,R6          ;SUB CURRENT ADR
```

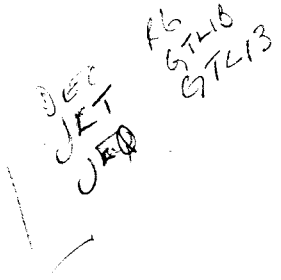
```
0070 0050 06A0  GTLLP   BL @GETCR        ;LOOP
     0052 0000
0071 0054 06A0          BL @JMPR0        ;DO JUMP ON R0
     0056 0000
0072 0058   51  GTJMP     BYTE GTLD-GTJMP/2,>04  ↑D
     0059   04
0073 005A   D4            BYTE GTCE-GTJMP/2,>05  ↑E
     005B   05
0074 005C   2C            BYTE GTCF-GTJMP/2,>06  ↑F
     005D   06
0075 005E   1C            BYTE GTBS-GTJMP/2,>08  ↑H
     005F   08
0076 0060   30            BYTE GTLI-GTJMP/2,>09  ↑I
     0061   09
0077 0062   18            BYTE GTLF-GTJMP/2,>0A  LF
     0063   0A
0078 0064   1A            BYTE GTCR-GTJMP/2,>0D  CR
     0065   0D
0079 0066   20            BYTE GTRB-GTJMP/2,>7F  RUBOUT
     0067   7F
0080 0068 0000            DATA 0
0081 006A 0280          CI R0,>2000      ;<BLK?
     006C 2000
0082 006E 1A07            JL GTLB        ;Y, OUT BELL
0083 0070 D157          MOVB *R7,R5      ;EXPANDING BUFFER?
0084 0072 1602            JNE GTLLP1     ;N
0085 0074 0606          DEC R6           ;Y, ROOM?
0086 0076 1102            JLT GTLLP2     ;N, OUT BELL
0087 0078 DDC0  GTLLP1  MOVB R0,*R7+     ;Y, STORE BYTE
0088 007A 1003          JMP GTLLP3
0089              *
0090 007C 0586  GTLLP2  INC R6           ;RESTORE R6
0091 007E 0200  GTLB    LI R0,>0700      ;OUT BELL
     0080 0700
0092 0082 06A0  GTLLP3  BL @TYP0         ;ECHO CHARACTER
     0084 0000
0093 0086 10E4          JMP GTLLP
0094              *
0095 0088 0729  GTLF    SETO @AINC(9)    ;SET AUTO-INC
     008A 0030'
0096 008C 0460  GTCR    B @CRLF1         ;RETURN
     008E 0000
```

```
0098                    *BACKSPACE
0099                    *
0100 0090 81D9  GTBS    C *R9,R7          ;BEGINNING OF BUFFER?
0101 0092 13F5            JEQ GTLB        ;Y, OUT BELL
0102 0094 0607          DEC R7            ;N, BACKUP
0103 0096 1008          JMP GTRB1         ;OUT BACKSPACE
0104                    *
0105                    *RUBOUT
0106                    *
0107 0098 81D9  GTRB    C *R9,R7          ;BEGINNING OF BUFFER?
0108 009A 13F1            JEQ GTLB        ;Y, OUT BELL
0109 009C 0607          DEC R7            ;N, BACKUP
0110 009E D5E0          MOVB @B20,*R7     ;STORE BLANK
     00A0 0042'
0111 00A2 06A0          BL @TYP11         ;OUT BKSP,BLK
     00A4 0000
0112 00A6 0820            DATA >0820
0113                    *
0114 00A8 06A0  GTRB1   BL @TYP11         ;OUT BKSP
     00AA 00A4'
0115 00AC 0800            DATA >0800
0116 00AE 10D0          JMP GTLLP         ;GOTO LOOP
0117                    *
0118                    *FORWARD SPACE (↑F)
0119                    *
0120 00B0 D037  GTCF    MOVB *R7+,R0      ;GET CHARACTER
0121 00B2 16E7            JNE GTLB+4      ;OK TO SEND
0122 00B4 0607          DEC R7            ;TOO FAR
0123 00B6 10E3          JMP GTLB          ;OUT BELL
```

```
0125                    *INSERT BLANKS
0126                    *
0127 00B8 06A0  GTLI    BL @GETCR           ;GET INSERT #
     00BA 0052'
0128 00BC 0240          ANDI R0,>0F00       ;MASK
     00BE 0F00
0129 00C0 06C0          SWPB R0             ;RIGHT JUSTIFY
0130 00C2 C147          MOV R7,R5           ;SAVE CBP (CURRENT BUFFER PTR)
0131                    *
0132 00C4 0606  GTLI1   DEC R6              ;ROOM?
0133 00C6 130A            JEQ GTLI3         ;N
0134 00C8 C1C5          MOV R5,R7           ;Y, RESTORE R7
0135 00CA D120          MOVB @B20,R4        ;INSERT BLK AS 1ST CHAR
     00CC 00A0'
0136                    *
0137 00CE D0D7  GTLI2   MOVB *R7,R3         ;GET NEW
0138 00D0 DDC4          MOVB R4,*R7+        ;INSERT OLD
0139 00D2 D103          MOVB R3,R4          ;OLD=NEW
0140 00D4 16FC            JNE GTLI2         ;MORE
0141 00D6 D5C4          MOVB R4,*R7         ;CLEAR BUFFER
0142 00D8 0600          DEC R0              ;DONE?
0143 00DA 15F4            JGT GTLI1         ;N
0144                    *
0145                    *INSERT,DELETE,EDER ENTRY
0146                    *
0147 00DC 06A0  GTLI3   BL @TYPC            ;OUT CRLF
     00DE 001A'
0148 00E0 06A0          BL @TYPB            ;OUT BUFFER
     00E2 0046'
0149 00E4 06A0          BL @TYP11           ;OUT CR ONLY
     00E6 00AA'
0150 00E8 0D00            DATA >0D00
0151 00EA C1D9          MOV *R9,R7
0152                    *
0153 00EC 8147  GTLI4   C R7,R5             ;POSITIONED?
0154 00EE 1480            JHE GTLLP         ;Y, GOTO LOOP
0155 00F0 04C0          CLR R0
0156 00F2 D037          MOVB *R7+,R0        ;N, OUT CHARACTER
0157 00F4 06A0          BL @TYP0
     00F6 0084'
0158 00F8 10F9          JMP GTLI4           ;DO AGAIN
0159                    *
0160                    *DELETE CHARACTERS
0161                    *
0162 00FA 06A0  GTLD    BL @GETCR           ;GET #
     00FC 00BA'
0163 00FE 0240          ANDI R0,>0F00       ;MASK
     0100 0F00
0164 0102 06C0          SWPB R0             ;RIGHT JUSTIFY
0165                    *
0166 0104 C147  GTLD1   MOV R7,R5           ;LOAD PTRS
0167 0106 C087          MOV R7,R2
0168 0108 D072          MOVB *R2+,R1        ;LOOK AT 1ST BYTE,23 APRIL79 HJC
0169 010A 13E8            JEQ GTLI3         ;ZERO
0170 010C 0586          INC R6              ;SOMETHING TO DELETE
0171                    *
0172 010E DD72          MOVB *R2+,*R5+      ;MOVE CHARACTER BACK
0173 0110 16FE            JNE $-2           ;MORE TO MOVE
0174 0112 0600          DEC R0              ;DONE?
0175 0114 15F7            JGT GTLD1         ;N
```

NT BUFFER PTR)

ST CHAR



R

E,23 APRIL79 HJC

LETE

BACK

```
0176                    *
0177 0116 C147  GTLD2   MOV R7,R5       ;Y, MARK
0178 0118 10E1          JMP GTLI3
```

```
 0180                   *EDIT ERROR RECOVERY ROUTINE
 0181                   *
 0182 011A C00B  EDER   MOV *R11,R3      ;PICKUP ERROR CHARACTERS
 0183 011C 0607         DEC R7
 0184 011E 8820         C   @ID302,@>3000  ;IS >A55A AT LOCATION >3000
      0120 0000
      0122 3000
 0185 0124 1603         JNE EDER01       ;NO, OUTPUT ERROR NUMBER
 0186 0126 06A0         BL  @ERRLS2      ;OUTPUT ERROR MESSAGE
      0128 0000
 0187 012A 1007         JMP EDER02       ;CONTINUE
 0188 012C 0201  EDER01 LI R1,ERM1       ;OUT 'ERR='
      012E 0000
 0189 0130 06A0         BL @TYPE
      0132 0000
 0190 0134 C003         MOV R3,R0        ;READY ERROR CHARACTERS
 0191 0136 06A0         BL @TYP0         ;OUT
      0138 00F6'
 0192 013A C18E  EDER02 MOV R14,R6       ;RESTORE LINE COUNT
 0193 013C C029         MOV @PLF(9),R0   ;LOADING?
      013E 0000                                  <- INC &0.
 0194 0140 13EA           JEQ GTLD2      ;N
 0195 0142 06A0         BL @TYPC
      0144 00DE'
 0196 0146 06A0         BL @TYPB
      0148 00E2'
 0197 014A 0460         B @LDP5          ;Y, CONTINUE LOADING
      014C 0000
 0198                   END
NO ERRORS
*
```

CHARACTERS

AT LOCATION >3000

RROR NUMBER
MESSAGE

CHARACTERS

COUNT

INC &ϕ

LOADING

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL4.BASIC.SOURCE.EDIT
OBJECT ACCESS NAME=      BD1.DBASIC.REL4.BASIC.OBJECT.EDIT
LISTING ACCESS NAME=     BD1.DBASIC.REL4.BASIC.LIST.EDIT
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'EDIT'
0002              *     09/26/78
0003              *
0004                    DEF EDIT,LSTL,LDPS,LDPY
0005                    DEF SAVY,LDSTRT,CK302
0006                    DEF LCN1,LCN2,LCN3,LCN4
0007                    DEF LNXT,LDEF
0008              *
0009                    REF CSOPZ,CSRDZ,CSWTZ,CSCLZ,PROGM
0010                    REF B20,B01,CRCASS,STRTC,RUNP1
0011                    REF BUS,EORBUS,DDM,DLC,EBP
0012                    REF CKEX,CVDB20
0013                    REF CVDIFZ,CVDIZ,EDER
0014                    REF FNS,GSC,GSS,LNUM
0015                    REF FOR2,JMPR0
0016                    REF FPAC,SLT,SLN
0017                    REF GETC,GETCR,LINE2,NLIN
0018                    REF MODE,NVD,NVS,PLF
0019                    REF PUTB,RUNP,SIZE
0020                    REF STCK,ID302
0021                    REF SSP,UFT,VDT,VNT
0022                    REF TYP11,TYPC
0023                    REF TYPB,TYPE
0024                    REF STCK
0025                    REF DCNT,GOS2
0026                    REF CRLF
0027                    REF HOUT
0028                    DXOP EVFIX,11
0029                    DXOP OUTFP,12
0030                    DXOP OUTINT,13
0031              *
0032       2F80   ERROR  EQU >2F80
0033       2FA0   ERROR2 EQU ERROR+>20
0034              *
0035              *VERIFY PRESENCE OF 4TH PAIR OF ROMS (302 BOARD)
0036              *
0037 0000 8820    CK302  C @ID302,@>3000   ;>A55A SHOULD BE AT LOC >3000
     0002 0000
     0004 3000
0038 0006 1601           JNE ERR41
0039 0008 0458           RT
0040 000A 2FA0    ERR41  DATA ERROR2,41
     000C 0029
0041              *
0042              *PROGRAM COMMAND
0043              *
0044 000E 06A0    PRGM   BL @CK302         ;302?
     0010 0000'
0045 0012 0460           B @PROGM         ;Y
     0014 0000
0046              *
0047              *CONTINUE COMMAND
0048              *
0049 0016 0729    CONT   SETO @MODE(9)     ;SET TO RUN
     0018 0000
0050 001A 06A0           BL @TYPC         ;OUT CRLF
     001C 0000
0051 001E C069           MOV @SLN(9),R1   ;GET LINE #
     0020 0000
0052 0022 0460           B @GOS2
```

0024 0000

```
0054                    *LOAD PROGRAM
0055                    *       PROGRAM LOAD FLAG INDICATES AUDIO OR DIGITAL CASSETTE
0056                    *            PLF=1      LOAD UNIT 1 OR 2 AUDIO CAS.
0057                    *            PLF=-1     LOAD DIGITAL CASSETTE
0058                    *
0059 0026 0729  LDPY    SETO @PLF(9)        ;SET FLAG
     0028 0000
0060 002A 06A0          BL @CKEX            ;CHECK FOR EXPRESSION
     002C 0000
0061 002E 104E            JMP LDP5          ;NONE
0062 0030 2EC2          EVFIX R2            ;GET ADDRESS
0063 0032 0282          CI R2,2
     0034 0002
0064 0036 121E            JLE LDP01
0065                    *
0066 0038 05C2  LDSTRT  INCT R2             ;MOVE OVER RUN ID
0067 003A CA72          MOV *R2+,@SLT(9) ;GET SLT DISPLACEMENT
     003C 0000
0068 003E CA72          MOV *R2+,@VNT(9) ;GET VNT DISPLACEMENT
     0040 0000
0069 0042 C069          MOV @EORBUS(9),R1   ;GET OLD BUS
     0044 0000
0070 0046 CA41          MOV R1,@VDT(9)
     0048 0000
0071 004A CA41          MOV R1,@NVD(9)
     004C 0000
0072 004E AA72          A *R2+,@NVD(9)   ;ADD VARIABLE SPACE
     0050 004C'
0073 0052 CA42          MOV R2,@BUS(9)   ;GET NEW BUS
     0054 0000
0074 0056 AA42          A   R2,@SLT(9)   ;GET ABS ADDR OF SLT
     0058 003C'
0075 005A AA42          A   R2,@VNT(9)   ;GET ABS ADDR OF VNT
     005C 0040'
0076 005E 0222          AI  R2,-8        ;POINT TO RUN/NORUN FLAG
     0060 FFF8
0077 0062 84A0          C   @STRTC,*R2   ;AUTO RUN REQUESTED?
     0064 0000
0078 0066 1602          JNE LDP0         ;NO - MANUAL START
0079 0068 0460          B   @RUNP1        ;YES - FORCE RUN
     006A 0000
0080 006C 0460  LDP0    B   @CRLF
     006E 0000
0081                    *
0082 0070 2FA0  ERR44   DATA ERROR2,44
     0072 002C
0083                    *
0084 0074 C002  LDP01   MOV R2,R0
0085 0076 0600          DEC R0
0086 0078 1129            JLT LDP5       ;0, TREAT AS LOAD
0087 007A 06A0          BL @CK302        ;302 THERE?
     007C 0000'
0088 007E 0420          BLWP @CSOPZ      ;Y - OPEN FILE FOR READ
     0080 0000
0089 0082 0000          DATA 0           ;OPEN FOR READ FLAG
0090 0084 05E9          INCT @PLF(9)     ;ADJUST FLAG
     0086 0028'
0091                    *
0092      0088' LDP02   EQU $
0093 0088 0205          LI  R5,EBP       ;GET EDIT BUFFER AS INPUT BUFFER
```

```
      008A 0000
0094  008C C005          MOV R5,R0          ;PASS BUFFER ADDRESS
0095  008E 0420          BLWP @CSRDZ        ;GET RECORD FROM AUDIO CASSETTE
      0090 0000
0096  0092 1014            JMP LDP4         ;EOT RETURN
0097  0094 10ED            JMP ERR44        ;TAPE READ ERROR RETURN
0098                 *
0099  0096 C1D9   LDP1    MOV *R9,R7         ;GET BUFFER ADR
0100                 *
0101  0098 C0E9   LDP2    MOV @PLF(9),R3     ;CHECK FOR TYPE OF LOAD
      009A 0086'
0102  009C 1503            JGT LDP2A        ;FROM AUDIO CASETTE
0103  009E 06A0          BL @GETCR          ;CASSETTE
      00A0 0000
0104  00A2 1001          JMP LDP2B
0105                 *
0106  00A4 D035   LDP2A   MOVB *R5+,R0
0107                 *
0108  00A6 06A0   LDP2B   BL @PUTB           ;STORE
      00A8 0000
0109  00AA 1002            JMP LDP3         ;CR
0110  00AC 10F5            JMP LDP2         ;CONTROL CHARACTER
0111  00AE 10F4          JMP LDP2           ;CHARACTER
0112                 *
0113  00B0 06A0   LDP3    BL @EDIT           ;EDIT LINE
      00B2 00DC'
0114  00B4 06A0          BL @GETC           ;CHECK FOR ESCAPE
      00B6 0000
0115  00B8 1009            JMP LDP5         ;NOTHING
0116  00BA 1008            JMP LDP5         ;CHARACTER
0117                 *
0118  00BC 04E9   LDP4    CLR @PLF(9)        ;EOF OR ESC, CLEAR FLAG
      00BE 009A'
0119                 *
0120                 *  STOP AUDIO CASSETTE UPON ENTRY OF ESCAPE KEY
0121                 *
0122  00C0 020C          LI  R12,CRCASS     ;GET AUDIO CASSETTE CRU BASE
      00C2 0000
0123  00C4 1E00          SBZ 0              ;STOP CASSETTE DRIVE #1
0124  00C6 1E01          SBZ 1              ;STOP CASSETTE DRIVE #2
0125                 *
0126  00C8 0460          B @NLIN            ;DONE
      00CA 0000
0127  00CC C0E9   LDP5    MOV @PLF(9),R3     ;CHECK FOR AUDIO CASS.
      00CE 00BE'
0128  00D0 15DB            JGT LDP02        ;YES  GET NEW BUFFER
0129  00D2 0201          LI R1,@BLKFWD      ;OUTPUT BLOCK FORWARD
      00D4 0AA9'
0130  00D6 06A0          BL @TYPE           ;TO DIGITAL CASSETTE
      00D8 0000
0131  00DA 10DD          JMP LDP1           ;AND GET CHARACTERS
```

```
0133                *EDIT INPUT STRING
0134                *
0135                *       R6  = SUBSCRIPT STACK PTR
0136                *       R7  = INPUT BYTE PTR
0137                *       R8  = OUTPUT BYTE PTR
0138                *       R9  = SYSTEM PTRS
0139                *       R10 = RETURN ADR
0140                *       R12 = BEGINNING OF COMMAND PTR
0141                *       R14 = SAVE FOR LINE COUNTER
0142                *
0143 00DC C28B  EDIT    MOV R11,R10     ;SAVE RETURN
0144                *       MOV R6,R14      ;SAVE LINE COUNT FOR EDER ROUTINE
0145 00DE C1D9          MOV *R9,R7      ;GET I/O PTR
0146 00E0 0206          LI R6,SSP       ;GET SUBSCRIPT STACK PTR
     00E2 0000
0147 00E4 0716          SETO *R6        ;MARK SUBSCRIPT STACK
0148 00E6 0208          LI R8,EBP       ;GET EDIT BUFFER PTR
     00E8 008A'
0149 00EA C169          MOV @VNT(9),R5  ;SET UP RESERVED WORDS
     00EC 005C'
0150 00EE 04F5          CLR *R5+        ;CLEAR DUMMY VARIABLES
0151 00F0 04F5          CLR *R5+
0152 00F2 04F5          CLR *R5+
0153 00F4 0200          LI R0,>11D2
     00F6 11D2
0154 00F8 C540          MOV R0,*R5
0155                *
0156                *       GET LINE NUMBER
0157                *
0158 00FA 0420          BLWP @CVDIZ     ;CONVERT LINE NUMBER
     00FC 0000
0159 00FE 103C            JMP EDER3     ;OVERFLOW
0160 0100 04C1            CLR R1        ;NO NUMBER
0161 0102 CA41          MOV R1,@LNUM(9) ;STORE LINE NUMBER
     0104 0000
0162 0106 1138            JLT EDER3
0163 0108 1301            JEQ $+4       ;LINE #
0164 010A 0607          DEC R7          ;Y, BACKUP OVER DELIMITER
0165 010C C000          MOV R0,R0       ;DONE?
0166 010E 1602            JNE ED3       ;N
0167 0110 0460          B @EMV0         ;Y, DELETE LINE
     0112 0478'
0168                *
0169                *DECODE COMMAND WORD
0170                *
0171 0114 06A0  ED3     BL @EDGLS       ;GET FIRST LETTER
     0116 0610'
0172 0118 1038            JMP EDSCI     ;LET OR ";
0173 011A C0C7          MOV R7,R3       ;MARK
0174 011C C001          MOV R1,R0
0175 011E 06A0          BL @EDGL        ;GET SECOND LETTER
     0120 0618'
0176 0122 1014            JMP EDLET
0177 0124 A001          A R1,R0
0178                *
0179                *CHECK FOR TWO LETTER COMMANDS HERE
0180                *
0181 0126 0280          CI R0,>1920     ;IF?
     0128 1920
0182 012A 1329            JEQ EDIF      ;Y
```

```
0183 012C 0280          CI R0,>39E0     ;ON?
     012E 39E0
0184 0130 1329            JEQ EDON      ;Y
0185 0132 06A0          BL @EDGL        ;N, GET 3RD LETTER
     0134 0618'
0186 0136 100A            JMP EDLET     ;LET
0187 0138 A001          A R1,R0
```

```
0189                    *SEARCH COMMAND LIST
0190                    *
0191 013A 0204                  LI R4,EDSCL     ;Y, GET COMMAND LIST PTR
     013C 0680'
0192                    *
0193 013E C074  ED5     MOV *R4+,R1     ;GET FIRST COMMAND
0194 0140 0911          SRL R1,1        ;MOVE 2ND WORD INDICATOR INTO CARRY
0195 0142 8040          C R0,R1         ;FOUND?
0196 0144 1307            JEQ ED6       ;MAYBE
0197 0146 0284          CI R4,EDCLE     ;N, MORE COMMANDS?
     0148 06DC'
0198 014A 1AF9            JL ED5        ;Y
0199                    *
0200 014C C1C3  EDLET   MOV R3,R7       ;RESTORE R7
0201                    *
0202 014E DE20  EDLETA  MOVB @LETB,*R8+ ;INSERT >6 (IMPLIED LET)
     0150 0888'
0203 0152 103D          JMP ED13
0204                    *
0205 0154 C080  ED6     MOV R0,R2       ;SAVE IN CASE NOT FOUND
0206 0156 C3C7          MOV R7,R15
0207 0158 C024          MOV @EDCS(4),R0 ;GET SECOND WORD
     015A 005A
0208 015C 171C            JNC ED9       ;FOUND
0209                    *
0210 015E C140  ED7     MOV R0,R5
0211 0160 0245          ANDI R5,>3E     ;MASK CHARACTER
     0162 003E
0212 0164 1318            JEQ ED9       ;FOUND
0213 0166 06A0          BL @EDGL        ;GET LETTER
     0168 0618'
0214 016A 1003            JMP ED8       ;PROBLEM
0215 016C 0991          SRL R1,9        ;POSITION
0216 016E 8045          C R5,R1         ;LETTER SAME?
0217 0170 13F6            JEQ ED7       ;Y
0218                    *
0219 0172 C002  ED8     MOV R2,R0       ;PROBLEM, CONTINUE TO LOOK
0220 0174 C1CF          MOV R15,R7
0221 0176 10E3          JMP ED5
0222                    *
0223 0178 06A0  EDER3   BL @EDER        ;INVALID LINE NUMBER
     017A 0000
0224 017C   30            TEXT '03'
     017D   33
0225                    *
0226 017E 0204  EDIF    LI R4,IFB       ;IF ENTRY
     0180 0042
0227 0182 100E          JMP ED10
0228                    *
0229 0184 0204  EDON    LI R4,ONB       ;ON ENTRY
     0186 0040
0230 0188 100B          JMP ED10
0231                    *
0232 018A 9837  EDSCI   CB *R7+,@B3B    ;";?
     018C 07A9'
0233 018E 16DF            JNE EDLETA    ;N
0234 0190 0204          LI R4,PTRB      ;"; ENTRY
     0192 0014
0235 0194 1005          JMP ED10
```

```
0237                    *COMMAND TYPE DONE
0238                    *CHECK FOR GOTO OR GOSUB
0239                    *
0240 0196 0201  ED9     LI R1,EDCL          ;AI R4,-EDCL
     0198 068A'
0241 019A 6101          S R1,R4             ;SYSTEM COMMAND?
0242 019C 1501            JGT ED10          ;N
0243 019E 0450          B *R0               ;Y
0244                    *
0245 01A0 0A74  ED10    SLA R4,7            ;POSITION BYTE
0246 01A2 DE04          MOVB R4,*R8+        ;STORE IN STREAM
0247 01A4 0284          CI R4,>300          ;CHECK SPECIAL TYPES
     01A6 0300
0248 01A8 1108            JLT ED12          ;GOTO,GOSUB
0249 01AA 13B4            JEQ ED3           ;ELSE
0250 01AC 0284          CI R4,>400          ;REM?
     01AE 0400
0251 01B0 160F            JNE ED14          ;N
0252 01B2 0460          B @EDREM            ;Y
     01B4 03C0'
0253                    *
0254 01B6 DE20  ED11    MOVB @B3F,*R8+      ;STORE ,
     01B8 07AA'
0255                    *
0256 01BA 0420  ED12    BLWP @CVDIZ         ;GET INTEGER
     01BC 00FC'
0257 01BE 1000            NOP
0258 01C0 10DB            JMP EDER3
0259 01C2 DE01          MOVB R1,*R8+        ;STORE INTEGER
0260 01C4 06C1          SWPB R1
0261 01C6 DE01          MOVB R1,*R8+
0262 01C8 0280          CI R0,>2C00         ;",?
     01CA 2C00
0263 01CC 13F4            JEQ ED11          ;Y
0264                    *
0265 01CE 0607  ED13    DEC R7              ;FALL THRU TO ED20
0266                    *
0267 01D0 C308  ED14    MOV R8,R12          ;MARK
```

```
0269                    *PROCESS RESET OF LIST
0270                    *
0271 01D2 0420  ED20    BLWP @CVDIFZ        ;LOOK FOR NUMBER
     01D4 0000
0272 01D6 1014            JMP ED23          ;OVERFLOW, FP
0273 01D8 101D            JMP ED30          ;NO NUMBER
0274 01DA 1009            JMP ED20H         ;HEX
0275 01DC 0281          CI R1,-1            ;<-1?
     01DE FFFF
0276 01E0 1109            JLT ED21          ;Y
0277 01E2 0281          CI R1,9             ;>9?
     01E4 0009
0278 01E6 1506            JGT ED21          ;N
0279 01E8 0221          AI R1,>63           ;GET CODE (62-6C)
     01EA 0063
0280 01EC 1006          JMP ED22            ;INSERT
0281                    *
0282 01EE DE20  ED20H   MOVB @B6E,*R8+      ;MARK AS HEX
     01F0 0A03'
0283 01F2 1002          JMP ED21A
0284                    *
0285 01F4 DE20  ED21    MOVB @B6D,*R8+      ;2 BYTE INTEGER, INSERT >6D
     01F6 07B8'
0286 01F8 DE01  ED21A   MOVB R1,*R8+
0287 01FA 06C1  ED22    SWPB R1
0288 01FC DE01          MOVB R1,*R8+
0289 01FE 1009          JMP ED24            ;LOOK FOR OPERATOR
0290                    *
0291 0200 DE20  ED23    MOVB @B6F,*R8+      ;FP #, INSERT >6F
     0202 07BA'
0292 0204 0203          LI R3,6
     0206 0006
0293 0208 0204          LI R4,FPAC          ;GET FPAC ADR
     020A 0000
0294                    *
0295 020C DE34          MOVB *R4+,*R8+      ;MOVE IN NUMBER
0296 020E 0603          DEC R3              ;DONE?
0297 0210 16FD            JNE $-4           ;N
0298                    *
0299 0212 0607  ED24    DEC R7              ;Y, BACKUP OVER DELIMITER
0300                    *
0301                    *VARIABLE OR KEY WORD
0302                    *GET 3 CHARACTERS - EXIT TO EDOP IF NON-LETTER
0303                    *
0304 0214 06A0  ED30    BL @EDGWO           ;CHECK FOR WORD OPERATER
     0216 0630'
0305 0218 06A0          BL @EDGLS           ;PROCESS COMMAND LIST
     021A 0610'
0306 021C 105E            JMP EDOP          ;NOT LETTER, OPERATOR
0307 021E C001          MOV R1,R0
0308 0220 06A0          BL @EDGL            ;GET NEXT LETTER
     0222 0618'
0309 0224 101D            JMP EDV1          ;NOT LETTER, 1 CHARACTER VARIABLE
0310 0226 A001          A R1,R0             ;ADD NEW LETTER
0311 0228 0280          CI R0,>38C0         ;'FN'?
     022A 38C0
0312 022C 1602            JNE ED31          ;N
0313 022E 0460          B @EDFN             ;Y, FUNCTION
     0230 0406'
0314                    *
```

```
0315 0232 06A0  ED31    BL @EDGL          ;N, GET 3RD LETTER
     0234 0618'
0316 0236 1025            JMP EDV2        ;NOT LETTER, 2 LETTER VARIABLE
0317 0238 A001            A R1,R0
```

```
0319                    *LOOK FOR 3 LETTER KEY WORDS
0320                    *
0321 023A 0204          LI R4,EDIL      ;GET LIST PTR
     023C 071E'
0322                    *
0323 023E C074  ED32    MOV *R4+,R1     ;GET CHARACTERS
0324 0240 0911          SRL R1,1        ;REMOVE TERMINATOR
0325 0242 8040          C R0,R1         ;SAME?
0326 0244 1609          JNE ED34        ;N
0327 0246 020B          LI R11,EDIL
     0248 071E'
0328 024A 0224          AI R4,>1A*2
     024C 0034
0329 024E 610B          S R11,R4
0330 0250 0914          SRL R4,1        ;SYSTEM FUNCTION
0331                    *
0332 0252 06C4  ED33    SWPB R4         ;POSITION
0333 0254 DE04          MOVB R4,*R8+    ;INSERT
0334 0256 10BD          JMP ED20
0335                    *
0336 0258 0284  ED34    CI R4,EDILE     ;N, LIST DONE?
     025A 0746'
0337 025C 1AF0          JL ED32         ;N
0338 025E 1011          JMP ED35
0339                    *
0340                    *1 CHARACTER VARIABLE
0341                    *
0342 0260 C0C0  EDV1    MOV R0,R3       ;LOOK FOR #
0343 0262 06A0          BL @CVDB20      ;CHECK FOR DIGIT
     0264 0000
0344 0266 100B          JMP EDV1A       ;N
0345 0268 0607          DEC R7          ;Y, BACKUP
0346 026A 0420          BLWP @CVDIZ     ;CONVERT
     026C 01BC'
0347 026E 101D          JMP EDER4       ;FP #, ILLEGAL VARIABLE NAME
0348 0270 1007          JMP EDV1B       ;NO NUMBER
0349 0272 2460          CZC @CFF80,R1   ;TOO LARGE?
     0274 06EE'
0350 0276 1619          JNE EDER4       ;Y, ILLEGAL NAME
0351 0278 A0C1          A R1,R3         ;COMBINE
0352 027A 0263          ORI R3,>0380    ;INDICATE AS NUMBER
     027C 0380
0353      027C' C380    EQU $-2
0354                    *
0355 027E 0607  EDV1A   DEC R7          ;BACKUP OVER DELIMITER
0356                    *
0357 0280 C003  EDV1B   MOV R3,R0       ;RESTORE R0
```

```
0359                *2 OR 3 CHARACTER VARIABLE
0360                *
0361        0282'   EDV2    EQU $
0362  0282  06A0    ED35    BL @EDGP          ;LOOK FOR DIMENSION
      0284  0666'
0363  0286  1002              JMP ED35A       ;Y, DIMENSIONED
0364  0288  0607              DEC R7          ;BACKUP
0365  028A  1003              JMP ED36
0366                *
0367  028C  0500    ED35A    NEG R0          ;SET DIMENSION
0368  028E  05C6              INCT R6         ;STACK -1
0369  0290  0716              SETO *R6
0370                *
0371                *SAVE VARIABLE NAME
0372                *
0373  0292  0204    ED36     LI R4,>FF00+>70 ;-# OF VARIABLES
      0294  FF70
0374  0296  C169              MOV @VNT(9),R5  ;GET VARIABLE TABLE ADR
      0298  00EC'
0375                *
0376  029A  8A45    ED37     C R5,@VDT(9)    ;DONE?
      029C  0048'
0377  029E  1409              JHE ED38        ;Y, MAKE NEW VARIABLE
0378  02A0  8D40              C R0,*R5+       ;VARIABLE SAME?
0379  02A2  1317              JEQ ED39        ;Y, FOUND
0380  02A4  0584              INC R4          ;STILL ROOM?
0381  02A6  16F9              JNE ED37        ;Y
0382                *
0383  02A8  2F85    ERR5     DATA ERROR+5    ;N, TOO MANY VARIABLES
0384                *
0385  02AA  06A0    EDER4    BL @EDER        ;ILLEGAL VARIABLE NAME
      02AC  017A'
0386  02AE    30             TEXT '04'
      02AF    34
0387                *
0388  02B0  2F8A    ERR10    DATA ERROR+10   ;STORAGE OVERFLOW
```

```
0390                    *DEFINE NEW VARIABLE
0391                    *
0392 02B2 C0A9  ED38    MOV @NVD(9),R2  ;GET END OF TABLE
     02B4 0050'
0393 02B6 8CB2          C *R2+,*R2+     ;ADD 4
0394 02B8 8A42          C R2,@NVS(9)    ;ROOM FOR NEW VARIABLE?
     02BA 0000
0395 02BC 14F9            JHE ERR10     ;N
0396 02BE CA42          MOV R2,@NVD(9)  ;Y, UPDATE NVD
     02C0 02B4'
0397                    *
0398 02C2 0222  ED38A   AI R2,-4        ;MOVE POINTERS UP 2 BYTES
     02C4 FFFC
0399 02C6 C4B2          MOV *R2+,*R2
0400 02C8 8142          C R2,R5         ;DONE?
0401 02CA 18FB            JH ED38A      ;N
0402 02CC C540          MOV R0,*R5      ;Y, STORE NEW NAME
0403 02CE 05E9          INCT @VDT(9)    ;UPDATE VDT
     02D0 029C'
0404                    *
0405 02D2 C000  ED39    MOV R0,R0       ;DIMENSIONED?
0406 02D4 11BE            JLT ED33      ;Y
0407 02D6 06C4          SWPB R4         ;N, INSERT VARIABLE CODE
0408 02D8 DE04          MOVB R4,*R8+
0409                    *
0410                    *       TRANSLATE CHARACTER INTO CODE
0411                    *
0412 02DA 04C0  EDOP    CLR R0          ;GET OPERATER
0413 02DC D037          MOVB *R7+,R0    ;GET BYTE
0414 02DE 1372            JEQ ED90      ;DONE
0415 02E0 C040          MOV R0,R1
0416 02E2 06C1          SWPB R1
0417 02E4 0280          CI R0,>4100     ;<"A?
     02E6 4100
0418 02E8 1A08            JL EDOP1      ;Y
0419 02EA 0280          CI R0,>5B00     ;>"Z?
     02EC 5B00
0420 02EE 1A47            JL ED49       ;N, LETTER, TRY WORD OPERATER
0421 02F0 0221          AI R1,-26       ;SKIP ALPHABET
     02F2 FFE6
0422 02F4 0280          CI R0,>5E00     ;>"↑?
     02F6 5E00
0423 02F8 183F            JH EDER6      ;Y, INVALID CHARACTER
0424                    *
0425 02FA D061  EDOP1   MOVB @EDSL->21(1),R1  ;GET CODE
     02FC 0725'
0426 02FE 133C            JEQ EDER6     ;INVALID CHARACTER
0427 0300 D601          MOVB R1,*R8     ;STORE BYTE
0428 0302 06A0          BL @JMPR0       ;SWITCHBOARD ON SPECIAL CHARACTERS
     0304 0000
0429 0306   EA  EDOPTB    BYTE EDOP-EDOPTB/2,>20  SP
     0307   20
0430 0308   42            BYTE ED50-EDOPTB/2,>28  "(
     0309   28
0431 030A   42            BYTE ED50-EDOPTB/2,>5B  "[
     030B   5B
0432 030C   52            BYTE ED51-EDOPTB/2,>29  ")
     030D   29
0433 030E   52            BYTE ED51-EDOPTB/2,>5D  "]
     030F   5D
```

```
0434 0310   0F            BYTE ED42-EDOPTB/2,>22  ""
     0311   22
0435 0312   0F            BYTE ED42-EDOPTB/2,>27  "'
     0313   27
0436 0314   30            BYTE ED47-EDOPTB/2,>3A  ":
     0315   3A
0437 0316   5C            BYTE ED54-EDOPTB/2,>21  "!
     0317   21
0438 0318   16            BYTE ED43-EDOPTB/2,>3D  "=
     0319   3D
0439 031A   27            BYTE ED45-EDOPTB/2,>3E  ">
     031B   3E
0440 031C 0000            DATA 0
0441 031E 0588  ED40      INC R8              ;MOVE OVER CODE
0442 0320 0460  ED41      B @ED20             ;CONTINUE PARSE
     0322 01D2'
0443                      *
0444 0324 0588  ED42      INC R8              ;PROCESS " OR '
0445 0326 D637            MOVB *R7+,*R8       ;GET NEXT BYTE
0446 0328 136B              JEQ EDER2         ;PROBLEM
0447 032A 9600            CB R0,*R8           ;CLOSE?
0448 032C 16FB              JNE ED42          ;N
0449 032E 7E18            SB *R8,*R8+         ;Y, TERMINATE WITH NULL
0450 0330 10F7            JMP ED41
0451                      *
0452 0332 0608  ED43      DEC R8              ;PROCESS "=
0453 0334 9818            CB *R8,@B57         ;>=?
     0336 0763'
0454 0338 130A              JEQ ED44          ;Y
0455 033A 9818            CB *R8,@B59         ;<=?
     033C 0761'
0456 033E 1307              JEQ ED44          ;Y
0457 0340 9838            CB *R8+,@B56        ;==?
     0342 0762'
0458 0344 16EC              JNE ED40
0459 0346 7A20            SB @B01,@-1(8)      ;Y, MAKE ==
     0348 0000
     034A FFFF
0460 034C 10E9            JMP ED41
0461                      *
0462 034E BE20  ED44      AB @B01,*R8+        ;MODIFY CODE
     0350 0348'
0463 0352 10E6            JMP ED41            ;GOTO ED20
0464                      *
0465 0354 9828  ED45      CB @-1(8),@B59      ;PROCESS ">
     0356 FFFF
     0358 0761'
0466 035A 16E1              JNE ED40
0467 035C BA20            AB @B02,@-1(8)      ;MAKE <>
     035E 06F3'
     0360 FFFF
0468 0362 10DE  ED41P     JMP ED41
0469                      *
0470 0364 0587  ED46      INC R7              ;PROCESS :,THEN
0471 0366 9817  ED47      CB *R7,@B20         ;SPACE
     0368 0000
0472 036A 13FC              JEQ ED46          ;Y, IGNOR
0473 036C 9817            CB *R7,@B3A         ;ANOTHER COLON?
     036E 07A5'
0474 0370 13F9              JEQ ED46          ;Y
```

```
0475 0372 0588          INC R8          ;N, PROCESS NEW COMMAND
0476 0374 0460          B @ED3
     0376 0114'
0477                *
0478 0378 06A0  EDER6   BL @EDER        ;ILLEGAL CHARACTER
     037A 02AC'
0479 037C   30            TEXT '06'
     037D   36
0480                *
0481                *LOOK FOR WORD OPERATER
0482                *
0483 037E 0607  ED49    DEC R7
0484 0380 06A0          BL @EDGWO
     0382 0630'
0485 0384 06A0          BL @EDER        ;EXPECTING OPERATER
     0386 037A'
0486 0388   30            TEXT '07'
     0389   37
```

```
0488                *( OR [ ENTRY
0489                *
0490 038A 8308  ED50    C R8,R12          ;COMMAND BYTE?
0491 038C 1804            JH ED50A        ;N
0492 038E 9828            CB @-1(8),@BCPB ;MEM,CRB,CRF,BIT?
     0390 FFFF
     0392 06EC'
0493 0394 1A08            JL ED50B        ;N, LEAVE (
0494 0396 9828  ED50A   CB @-1(8),@B38    ;FUNCTION ARGUMENTS?
     0398 FFFF
     039A 078B'
0495 039C 1404            JHE ED50B       ;N, LEAVE (
0496 039E 05C6            INCT R6         ;Y, MAKE [
0497 03A0 04D6            CLR *R6         ;INSERT -1 INTO STACK
0498 03A2 7620            SB @B02,*R8     ;*R8=>4A
     03A4 06F3'
0499                *
0500 03A6 0616  ED50B   DEC *R6           ;INCREMENT TOP ITEM ON STACK
0501 03A8 10BA            JMP ED40        ;LEAVE
0502                *
0503                *) OR ] ENTRY
0504                *
0505 03AA 0596  ED51    INC *R6           ;NEED ]?
0506 03AC 1106            JLT ED52        ;N, LEAVE >4D
0507 03AE 0646            DECT R6         ;Y, DECREMENT STACK
0508 03B0 0286            CI R6,SSP       ;UNMATCHED?
     03B2 00E2'
0509 03B4 1A25            JL EDER2        ;Y
0510 03B6 7620            SB @B02,*R8     ;N, INSERT ] CODE (>4B)
     03B8 06F3'
0511                *
0512 03BA 0588  ED52    INC R8            ;LEAVE CODE
0513 03BC 108E            JMP EDOP        ;LOOK FOR OPERATER
```

?

BIT?

UMENTS?

```
MOVB  @-1(R8),R1   last char string terminator (>)?
JEQ   EDSUB
CB    R1,@R22
```

TO STACK

P ITEM ON STACK

STACK

CODE (>4B)

RATER

```
0515                *PROCESS REMARKS
0516                *
0517 03BE 0588  ED54    INC R8              ;PROCESS TAIL REMARK
0518                *
0519 03C0 DE37  EDREM   MOVB *R7+,*R8+      ;MOVE REMARK
0520 03C2 16FE            JNE EDREM
0521                *
0522                *END PARSE, CHECK FINAL ERRORS
0523                *
0524 03C4 0286  ED90    CI R6,SSP           ;SUBSCRIPT ERROR?
     03C6 03B2'
0525 03C8 161B            JNE EDER2         ;Y
0526 03CA 0596          INC *R6             ;PAREN ERROR?
0527 03CC 1619            JNE EDER2         ;Y
0528 03CE 7E18          SB *R8,*R8+         ;N, MARK OUTPUT
0529 03D0 7618          SB *R8,*R8          ;DOUBLE NULL
0530 03D2 0700          SETO R0             ;SET TO INSERT
0531 03D4 C069          MOV @LNUM(9),R1     ;GET LINE NUMBER
     03D6 0104'
0532 03D8 164F            JNE EMV0          ;INSERT OR CHANGE
0533 03DA C1D9          MOV *R9,R7          ;NO LINE NUMBER
0534 03DC 0227          AI R7,30            ;MOVE INTO IOB
     03DE 001E
0535 03E0 C007          MOV R7,R0           ;MARK
0536 03E2 0203          LI R3,EBP           ;**NOTE** ALL PTRS ON WRD BOUNDARIES
     03E4 00E8'
0537                *
0538 03E6 CDF3          MOV *R3+,*R7+       ;MOVE
0539 03E8 8203          C R3,R8             ;DONE?
0540 03EA 1AFD            JL $-4            ;N
0541                *
0542 03EC C200          MOV R0,R8           ;SET R8
0543 03EE 9818          CB *R8,@B01         ;CHECK TYPE, GOTO?
     03F0 0350'
0544 03F2 1B04            JH ED91           ;N
0545 03F4 0729          SETO @MODE(9)       ;Y, SET TO RUN MODE
     03F6 0018'
0546 03F8 06A0          BL @TYPC            ;OUT CRLF
     03FA 001C'
0547                *
0548 03FC 0460  ED91    B @LINE2
     03FE 0000
0549                *
0550 0400 06A0  EDER2   BL @EDER            ;UNMATCHED PARENTHESIS
     0402 0386'
0551 0404   30            TEXT '02'
     0405   32
```

```
0553                    *PROCESS FN-
0554                    *
0555 0406 06A0  EDFN    BL @EDGLS           ;GET LETTER
     0408 0610'
0556 040A 102F            JMP EDER8         ;NO LETTER
0557 040C 0921          SRL R1,2            ;POSITION
0558 040E DE01          MOVB R1,*R8+        ;INSERT INLINE
0559 0410 C0C8          MOV R8,R3           ;CHECK PREVIOUS BYTE
0560 0412 0643          DECT R3             ;LOOK FOR OPERATOR
0561 0414 0283          CI R3,EBP           ;BEGINNING OF BUFFER?
     0416 03E4'
0562 0418 1683            JNE ED41          ;N, DISREGUARD
0563 041A 9813          CB *R3,@DEFXB       ;'DEF'?
     041C 07BC'
0564 041E 1680            JNE ED41          ;N, CONTINUE
0565 0420 06A0          BL @EDGP            ;Y, LOOK FOR ( OR [
     0422 0666'
0566 0424 1002            JMP EDFN0         ;"( OR "[, OK
0567 0426 0607          DEC R7              ;NEITHER, LOOK FOR "=
0568 0428 1017          JMP EDFN3
0569                    *
0570 042A 0203  EDFN0   LI R3,3             ;Y, ALLOW 3 DUMMY VARIABLES
     042C 0003
0571 042E C129          MOV @VNT(9),R4      ;GET STORAGE ADR
     0430 0298'
0572                    *
0573 0432 06A0  EDFN1   BL @EDGLS           ;GET DUMMY VARIABLE
     0434 0610'
0574 0436 101C            JMP EDER9         ;PROBLEM
0575 0438 DE01          MOVB R1,*R8+        ;INSERT IN CODE
0576 043A CD01          MOV R1,*R4+         ;STORE CODE
0577 043C 06A0          BL @EDGP            ;GET NEXT BYTE
     043E 0666'
0578 0440 1017            JMP EDER9
0579 0442 0603          DEC R3              ;ROOM FOR MORE?
0580 0444 1303            JEQ EDFN2         ;N, SEE IF ")
0581 0446 0281          CI R1,>2C00         ;Y, DELIMITER ",?
     0448 2C00
0582 044A 13F3            JEQ EDFN1         ;Y, LOOP
0583                    *
0584 044C 0281  EDFN2   CI R1,>2900         ;N, CLOSING PAREN?
     044E 2900
0585 0450 1303            JEQ EDFN3         ;Y, LOOK FOR "=
0586 0452 0281          CI R1,>5D00         ;N, "]?
     0454 5D00
0587 0456 16D4            JNE EDER2         ;N, PROBLEM
0588 0458 95E0  EDFN3   CB @B3D,*R7         ;Y, "=?
     045A 07BF'
0589 045C 1382            JEQ ED41P         ;Y
0590 045E 9DE0          CB @B20,*R7+        ;SPACE?
     0460 0368'
0591 0462 13FA            JEQ EDFN3         ;Y, TRY AGAIN
0592 0464 06A0          BL @EDER            ;MISSING ASSIGNMENT
     0466 0402'
0593 0468   33              TEXT '36'
     0469   36
0594                    *
0595 046A 06A0  EDER8   BL @EDER            ;ILLEGAL FUNCTION NAME
     046C 0466'
0596 046E   30              TEXT '08'
```

```
      046F   38
0597                *
0598  0470 06A0  EDER9   BL @EDER          ;ILLEGAL FUNCTION ARGUMENT
      0472 046C'
0599  0474   30              TEXT '09'
      0475   39
```

```
0601                    *FINISH EDIT PROCESS
0602                    *       BL @EMV
0603                    *
0604                    *       IN  R0 = 0 FOR DELETE, <>0 FOR CHANGE OR INSERT
0605                    *           R1 = LINE NUMBER
0606                    *
0607 0476 C28B  EMV     MOV R11,R10     ;SAVE RETURN
0608                    *
0609 0478 C041  EMV0    MOV R1,R1       ;DELETE 0?
0610 047A 136F            JEQ EMVR      ;Y, ACTION COMPLETE
0611                    *
0612 047C 0202          LI R2,EBP
     047E 0416'
0613 0480 6202          S R2,R8         ;GET # OF BYTES
0614 0482 0588          INC R8          ;GET # OF WORDS (NEXT HIGHEST)
0615 0484 0818          SRA R8,1
0616 0486 C088          MOV R8,R2       ;GET # OF BYTES
0617 0488 0A12          SLA R2,1
0618                    *
0619 048A C1A9          MOV @SLT(9),R6  ;GET START OF STATEMENT TABLE
     048C 0058'
0620 048E C1C6          MOV R6,R7
0621 0490 61E9          S @BUS(9),R7    ;R7=POINT OF QUESTION IN PSEUDO SOUR
     0492 0054'
0622                    *
0623                    *UPON EXIT OF EMV1, R7 DISPLACES INTO PSEUDO SOURCE
0624                    *  AND R6 POINTS INTO SLT
0625                    *
0626 0494 C116  EMV1    MOV *R6,R4      ;DONE?
0627 0496 1307            JEQ EMV2      ;Y
0628 0498 8581          C R1,*R6        ;N, FOUND?
0629 049A 1360            JEQ EMV5      ;Y, CHANGE OR DELETE
0630 049C 1504            JGT EMV2      ;Y, NEW LINE
0631 049E 05C6          INCT R6
0632 04A0 C1F6          MOV *R6+,R7     ;GET NEW POINT OF QUESTION
0633 04A2 10F8          JMP EMV1
0634                    *
0635 04A4 2F8D  ERR13   DATA ERROR+13   ;NO SUCH LINE NUMBER
0636                    *
0637 04A6 C000  EMV2    MOV R0,R0       ;NEW #, DELETE?
0638 04A8 13FD            JEQ ERR13     ;Y, PROBLEM
0639                    *
0640                    *INSERT NEW LINE ENTRY
0641                    *
0642 04AA C0C6          MOV R6,R3       ;GET SOURCE
0643 04AC 06A0          BL @EMVA        ;ADJUST
     04AE 0592'
0644 04B0 0004            DATA 4        ;INSERT 4 BYTES IN SLT
0645 04B2 CD81          MOV R1,*R6+     ;INSERT NEW LINE #
0646 04B4 C587          MOV R7,*R6      ;INSERT DISPLACEMENT
0647                    *
0648 04B6 0204  EMV3    LI R4,EBP       ;MOVE IN SOURCE LINE
     04B8 047E'
0649 04BA C196          MOV *R6,R6      ;GET ADR
0650 04BC A1A9          A @BUS(9),R6    ;MAKE DISPLACEMENT, POINTER
     04BE 0492'
0651 04C0 C1C6          MOV R6,R7       ;MOVE IN STRING
0652                    *
0653 04C2 CDF4          MOV *R4+,*R7+   ;MOVE
0654 04C4 0608          DEC R8          ;DONE?
```

```
0655 04C6 16FD          JNE $-4         ;N
```

```
0657                    *ADJUST GOSUB
0658                    *
0659                    *       IN  R2 = PBC ADJUSTMENT
0660                    *           R3 = PLC ADJUSTMENT
0661                    *           R6 = START
0662                    *           R7 = END
0663                    *
0664 04C8 C129  EMV4    MOV @GSC(9),R4  ;GET GOSUB STACK PTR
     04CA 0000
0665                    *
0666 04CC 8A44  EMV4A   C R4,@GSS(9)    ;DONE?
     04CE 0000
0667 04D0 1214            JLE EMV4E     ;Y
0668 04D2 0224          AI R4,-4        ;N, BACKUP
     04D4 FFFC
0669 04D6 8506          C R6,*R4        ;LESS THAN INSERTED LINE?
0670 04D8 1BF9            JH EMV4A      ;Y
0671 04DA 8507          C R7,*R4        ;GREATER THAN?
0672 04DC 120A            JLE EMV4C     ;Y
0673 04DE C144          MOV R4,R5       ;N, DELETE ENTRY
0674                    *
0675 04E0 CD65  EMV4B   MOV @4(5),*R5+  ;DELETE ENTRY
     04E2 0004
0676 04E4 8A45          C R5,@GSC(9)    ;DONE?
     04E6 04CA'
0677 04E8 1AFB            JL EMV4B
0678 04EA AA60          A @CM4,@GSC(9)  ;BACKUP PTR
     04EC 0572'
     04EE 04E6'
0679 04F0 10ED          JMP EMV4A
0680                    *
0681 04F2 A502  EMV4C   A R2,*R4        ;Y, ADJUST PBC
0682 04F4 A903          A R3,@2(4)      ;ADJUST PLC
     04F6 0002
0683 04F8 10E9          JMP EMV4A
0684                    *
0685                    *ADJUST FOR/NEXT STACK
0686                    *
0687 04FA C129  EMV4E   MOV @FNS(9),R4  ;DO FOR/NEXT STACK
     04FC 0000
0688                    *
0689 04FE C014  EMV4F   MOV *R4,R0      ;DONE?
0690 0500 130F            JEQ EMV4J     ;Y
0691 0502 C144          MOV R4,R5       ;N
0692 0504 0225          AI R5,14
     0506 000E
0693 0508 8546          C R6,*R5        ;LESS THAN INSERTED LINE?
0694 050A 1B07            JH EMV4I      ;Y
0695 050C 8547          C R7,*R5        ;>=?
0696 050E 1203            JLE EMV4H     ;Y
0697 0510 06A0          BL @FOR2        ;DELETE ENTRY
     0512 0000
0698 0514 10F4          JMP EMV4F       ;LOOK AGAIN
0699                    *
0700 0516 AD42  EMV4H   A R2,*R5+       ;Y, ADJUST PBC
0701 0518 A543          A R3,*R5        ;ADJUST PLC
0702 051A 0224  EMV4I   AI R4,18        ;MOVE TO NEXT
     051C 0012
0703 051E 10EF          JMP EMV4F
```

```
0705                    *ADJUST DATA POINTERS
0706                    *
0707 0520 C109  EMV4J   MOV R9,R4          ;GET DELIMITER PTR
0708 0522 0224          AI R4,DDM          ;DISPLACE
     0524 0000
0709 0526 8506          C R6,*R4           ;CHECK DATA PTRS
0710 0528 1809            JH EMV4L         ;OK
0711 052A 8507          C R7,*R4           ;DELETED OR CHANGED?
0712 052C 1202            JLE EMV4K        ;N, INSERTED
0713 052E 04D4          CLR *R4            ;Y, SET TO LOOK FURTHER
0714 0530 1003          JMP EMV4KA
0715                    *
0716 0532 C014  EMV4K   MOV *R4,R0         ;DEFINED?
0717 0534 1303            JEQ EMV4L        ;N, DON'T WORRY ABOUT IT
0718 0536 A502          A R2,*R4           ;Y, ADJUST PBC
0719                    *
0720 0538 AA43  EMV4KA  A R3,@DLC(9)       ;ADJUST PLC
     053A 0000
0721                    *
0722                    *ADJUST FUNCTION DEFINITION STACK
0723                    *
0724 053C C129  EMV4L   MOV @UFT(9),R4     ;GET POINTER
     053E 0000
0725                    *
0726 0540 C014  EMV4M   MOV *R4,R0         ;DEFINED?
0727 0542 1307            JEQ EMV40        ;N
0728 0544 8506          C R6,*R4           ;LESS THAN CHANGED LINE?
0729 0546 1805            JH EMV40         ;Y
0730 0548 8507          C R7,*R4
0731 054A 1202            JLE EMV4N        ;N
0732 054C 04D4          CLR *R4            ;UNDEFINE
0733 054E 1001          JMP EMV40
0734                    *
0735 0550 A502  EMV4N   A R2,*R4           ;ADJUST PBC
0736 0552 8D34  EMV40   C *R4+,*R4+        ;MOVE TO NEXT
0737 0554 8A44          C R4,@GSS(9)       ;DONE?
     0556 04CE'
0738 0558 1AF3            JL EMV4M         ;N
0739                    *
0740 055A 045A  EMVR    B *R10             ;RETURN
```

```
0742                *DELETE OR CHANGE LINE
0743                *
0744 055C C2E6  EMV5    MOV @2(6),R11    ;GET BOL
     055E 0002
0745 0560 62C7          S R7,R11         ;GET -LINE LENGTH
0746 0562 C000          MOV R0,R0        ;DELETE LINE?
0747 0564 1608            JNE EMV6       ;N
0748                *
0749                *DELETE LINE
0750                *
0751 0566 C08B          MOV R11,R2       ;Y
0752 0568 C0C6          MOV R6,R3
0753 056A 0223          AI R3,4          ;GET SLT SOURCE
     056C 0004
0754 056E 06A0          BL @EMVA         ;ADJUST
     0570 0592'
0755 0572 FFFC  CM4       DATA -4        ;DELETE 4 BYTES FROM SLT
0756 0574 10A9          JMP EMV4
0757                *
0758                *CHANGE LINE
0759                *
0760 0576 A08B  EMV6    A R11,R2         ;GET DELTA CHANGE
0761 0578 C029          MOV @NVD(9),R0   ;CHECK STORAGE
     057A 02C0'
0762 057C A002          A R2,R0
0763 057E 8A40          C R0,@NVS(9)     ;ROOM?
     0580 02BA'
0764 0582 1406            JHE EMVE10     ;N
0765 0584 020D          LI R13,EMV6A     ;GET RETURN ADR
     0586 058A'
0766 0588 1017          JMP EMVA1
0767                *
0768 058A 0000  EMV6A     DATA 0         ;LEAVE PLC ALONE
0769 058C 05C6          INCT R6          ;MOVE TO LINE ADR
0770 058E 1093          JMP EMV3
0771                *
0772 0590 2F8A  EMVE10  DATA ERROR+10    ;STORAGE OVERFLOW
```

```
0774                *ALTER SOURCE CODE
0775                *       BL @EMVA
0776                *         DATA (SLT ADJUSTMENT)
0777                *         R3=SLT SOURCE PTR
0778                *
0779 0592 C348  EMVA    MOV R11,R13      ;SAVE RETURN
0780 0594 C129          MOV @NVD(9),R4   ;GET THRU POINTER (END OF VARIABLE D
     0596 057A'
0781 0598 C143          MOV R3,R5        ;GET DESTINATION
0782 059A A15D          A *R13,R5        ; (4=INSERT,-4=DELETE,0=CHANGE)
0783 059C C004          MOV R4,R0        ;CHECK FOR SIZE
0784 059E A01D          A *R13,R0
0785 05A0 A002          A R2,R0          ;ADD LINE
0786 05A2 8A40          C R0,@NVS(9)
     05A4 0580'
0787 05A6 14F4            JHE EMVE10     ;OVERFLOW
0788 05A8 AA5D          A *R13,@VNT(9)   ;ADJUST POINTERS
     05AA 0430'
0789 05AC AA5D          A *R13,@VDT(9)
     05AE 02D0'
0790 05B0 AA5D          A *R13,@NVD(9)
     05B2 0596'
0791 05B4 06A0          BL @MOVE         ;DO FIRST MOVE
     05B6 05F0'
0792                *
0793 05B8 C0C7  EMVA1   MOV R7,R3        ;MAKE HOLE IN PSEUDO SOURCE
0794 05BA A0E9          A @BUS(9),R3     ;MAKE DISPLACEMENT INTO POINTER
     05BC 04BE'
0795 05BE C129          MOV @NVD(9),R4   ;THRU
     05C0 05B2'
0796 05C2 C143          MOV R3,R5
0797 05C4 A142          A R2,R5          ;DESTINATION
0798 05C6 A182          A R2,R6
0799 05C8 AA42          A R2,@SLT(9)     ;ADJUST POINTERS
     05CA 048C'
0800 05CC AA42          A R2,@VNT(9)
     05CE 05AA'
0801 05D0 AA42          A R2,@VDT(9)
     05D2 05AE'
0802 05D4 AA42          A R2,@NVD(9)
     05D6 05C0'
0803 05D8 06A0          BL @MOVE         ;DO SECOND MOVE
     05DA 05F0'
0804                *
0805                *ADJUST SLT
0806                *
0807 05DC C0C6          MOV R6,R3        ;GET POINTER
0808                *
0809 05DE 8A43  EMVA2   C R3,@SLT(9)     ;DONE?
     05E0 05CA'
0810 05E2 1204            JLE EMVA3      ;Y
0811 05E4 0643          DECT R3          ;N
0812 05E6 A4C2          A R2,*R3         ;ADJUST PTR
0813 05E8 0643          DECT R3
0814 05EA 10F9          JMP EMVA2
0815                *
0816 05EC C0FD  EMVA3   MOV *R13+,R3     ;GET PLC ADJUSTMENT
0817 05EE 045D          B *R13           ;RETURN
```

```
0819                    *MOVE
0820                    *       BL @MOVE
0821                    *
0822                    *       IN  R3 = SOURCE
0823                    *           R4 = SOURCE END
0824                    *           R5 = DESTINATION
0825                    *
0826 05F0 8143  MOVE    C R3,R5         ;THERE?
0827 05F2 131D            JEQ MOVE4     ;Y
0828 05F4 1A04            JL MOVE2      ;N, S<D
0829                    *
0830 05F6 8103  MOVE1   C R3,R4         ;S>D, DONE?
0831 05F8 1B1A            JH MOVE4      ;Y
0832 05FA CD73          MOV *R3+,*R5+   ;MOVE DATA
0833 05FC 10FC          JMP MOVE1
0834                    *
0835 05FE C004  MOVE2   MOV R4,R0       ;S<D
0836 0600 6003          S R3,R0         ;GET # OF WORDS
0837 0602 A140          A R0,R5
0838                    *
0839 0604 C554  MOVE3   MOV *R4,*R5     ;MOVE DATA
0840 0606 8103          C R3,R4         ;DONE?
0841 0608 1312            JEQ MOVE4     ;Y
0842 060A 0644          DECT R4         ;N, BACKUP
0843 060C 0645          DECT R5
0844 060E 10FA          JMP MOVE3
```

```
0846                *GET LETTER
0847                *       BL @EDGL
0848                *         NO LETTER, R7 UNCHANGED
0849                *       LETTER, R7 UPDATED
0850                *
0851 0610 9DE0  EDGLS   CB @B20,*R7+    ;GET LETTER, SPACE?
     0612 0460'
0852 0614 13FD            JEQ EDGLS     ;Y
0853 0616 0607          DEC R7          ;N, BACKUP
0854                *
0855 0618 9817  EDGL    CB *R7,@B40     ;<"A?
     061A 07A6'
0856 061C 1208            JLE EDGL1     ;Y
0857 061E 9817          CB *R7,@B5B     ;>"Z?
     0620 07B3'
0858 0622 1405            JHE EDGL1     ;Y
0859 0624 04C1          CLR R1          ;LETTER
0860 0626 D077          MOVB *R7+,R1
0861 0628 0A21          SLA R1,2        ;REMOVE UPPER BITS
0862 062A 0950          SRL R0,5        ;ADJUST R0
0863 062C 05CB          INCT R11
0864 062E 045B  EDGL1   RT
0865                *
0866      062E' MOVE4   EQU EDGL1
0867                *
0868                *GET WORD OPERATER
0869                *
0870 0630 C14B  EDGWO   MOV R11,R5
0871 0632 06A0          BL @EDGLS       ;GET LETTER
     0634 0610'
0872 0636 0455            B *R5         ;NO LETTER, RETURN
0873 0638 0607          DEC R7          ;OK, PROCESS
0874 063A 0204          LI R4,EDGWOL    ;GET WORD OPERATER LIST
     063C 076A'
0875                *
0876 063E C0C7  EDGWO1  MOV R7,R3       ;MARK
0877                *
0878 0640 D074  EDGWO2  MOVB *R4+,R1    ;GET CHARACTER, FOUND?
0879 0642 1308            JEQ EDGWO4    ;Y
0880 0644 9073          CB *R3+,R1      ;N, SAVE LETTER?
0881 0646 13FC            JEQ EDGWO2    ;Y
0882 0648 D074          MOVB *R4+,R1    ;N, MOVE TO NEXT
0883 064A 16FE            JNE $-2
0884 064C 0584          INC R4          ;MOVE OVER CODE
0885 064E D054          MOVB *R4,R1     ;DONE?
0886 0650 16F6            JNE EDGWO1    ;N, KEEP TRYING
0887 0652 0455          B *R5           ;Y, RETURN
0888                *
0889 0654 D614  EDGWO4  MOVB *R4,*R8    ;RESERVED WORD, GET CODE
0890 0656 C1C3          MOV R3,R7       ;UPDATE R7
0891 0658 9818          CB *R8,@B3B     ;THEN?
     065A 07A9'
0892 065C 1302            JEQ EDGWO5    ;Y, PROCESS :
0893 065E 0460          B @ED40         ;N, INSERT OPERATER
     0660 031E'
0894                *
0895 0662 0460  EDGWO5  B @ED47
     0664 0366'
```

```
0897                    *GET CHARACTER
0898                    *
0899                    *       BL @EDGP
0900                    *         "( OR "[
0901                    *       OTHER
0902                    *
0903 0666 04C1  EDGP    CLR R1
0904 0668 D077          MOVB *R7+,R1    ;GET CHARACTER
0905 066A 0281          CI R1,>2000     ;SP?
     066C 2000
0906 066E 13FB            JEQ EDGP      ;Y
0907 0670 0281          CI R1,>2800     ;"(?
     0672 2800
0908 0674 1304            JEQ EDGP1     ;Y
0909 0676 0281          CI R1,>5B00     ;"[?
     0678 5800
0910 067A 1301            JEQ EDGP1     ;Y
0911 067C 05CB          INCT R11        ;N, RETURN 2(11)
0912 067E 045B  EDGP1   RT
```

```
0914                    *COMMAND LIST
0915                    *
0916                    *SYMBOLS STORED AS:
0917                    *  3333 3222 2211 111S
0918                    *  WHERE S=0  3 LETTERS
0919                    *        S=1  4-6 LETTERS
0920                    *
0921                    *           0 1 2 3 4 5 6 7
0922                    *        0  @ A B C D E F G
0923                    *        1  H I J K L M N O
0924                    *        2  P Q R S T U V W
0925                    *        3  X Y Z
0926                    *
0927 0680 7564  EDSCL   DATA >7564                RUN
0928 0682 D266          DATA >D266,>9A58          SIZ,LIS
     0684 9A58
0929 0686 73C6          DATA >73C6                CON
0930 0688 7CA0          DATA >7CA0                PRO(GRAM)
0931                    *
0932 068A A3CF  EDCL    DATA >A3CF,>9BCF          GOTO*,GOSUB*
     068C 9BCF
0933 068E 9B0B          DATA >9B0B,>6964          ELSE*,REM*
     0690 6964
0934 0692 93CC          DATA >93CC                FOR*
0935 0694 0000          DATA 0                    (LET*)
0936 0696 A049          DATA >A049                DATA
0937 0698 C15D  NXTX    DATA >C15D                NEXT
0938 069A 948B          DATA >948B                ERROR
0939 069C 4CA1  PRTX    DATA >4CA1                PRINT
0940 069E 6047          DATA >6047                CALL
0941 06A0 0BD9          DATA >0BD9                LOAD
0942 06A2 8393          DATA >8393                INPUT
0943 06A4 0965          DATA >0965                READ
0944 06A6 9965          DATA >9965                RESTOR
0945 06A8 A165          DATA >A165                RETURN
0946 06AA 7D27          DATA >7D27                STOP
0947 06AC 4BAB          DATA >4BAB                UNIT
0948 06AE 6A69          DATA >6A69                TIME
0949 06B0 B067          DATA >B067                SAVE
0950 06B2 9845          DATA >9845                BASE
0951 06B4 1CCB          DATA >1CCB                ESCAPE
0952 06B6 2BDD          DATA >2BDD                NOESC
0953 06B8 7065          DATA >7065                RANDOM
0954 06BA 0B53          DATA >0B53                IMASK
0955 06BC A493          DATA >A493                IRTN
0956 06BE 0CA9          DATA >0CA9                TRAP
0957 06C0 A845          DATA >A845                BAUD
0958 06C2 83E0          DATA >83E0                POP
0959 06C4 6A48          DATA >6A48                DIM
0960 06C6 A158          DATA >A158                LET
0961 06C8 73C0  ONX     DATA >73C0                ON
0962 06CA 3240  IFX     DATA >3240                IF
0963 06CC 3148  DEFX    DATA >3148                DEF
0964 06CE B95C          DATA >B95C                NEW
0965 06D0 238A          DATA >238A                END
0966      0025  BCP     EQU $-EDCL/2+1    ****** INSERTS BEFORE HERE
0967 06D2 A244          DATA >A244                BIT  (SEE ED50)
0968 06D4 1486          DATA >1486                CRB
0969 06D6 3486          DATA >3486                CRF
0970 06D8 695A          DATA >695A                MEM
```

```
0971 06DA 25DA           DATA >25DA                  MWD  2 MAY 1979 HJC
0972      06DC' EDCLE    EQU $
0973            *
0974      0042  IFB      EQU IFX-EDCL+2
0975      0040  ONB      EQU ONX-EDCL+2
0976      0014  PTRB     EQU PRTX-EDCL+2
```

```
0978                *SECOND HALF OF PRIMITIVE TABLE
0979                *
0980 06DC 0000              DATA RUNP
0981 06DE 0000              DATA SIZE,LST
     06E0 0806'
0982 06E2 0016'             DATA CONT
0983 06E4 000E'             DATA PRGM
0984                *
0985       005A     EDCS    EQU $-EDCL-2
0986 06E6 001E              DATA >001E,>00AA          GOTO,GOSUB
     06E8 00AA
0987       06EB'    B0A     EQU $+1
0988 06EA 000A              DATA >000A                ELSE*
0989 06EC   25      BCPB    BYTE BCP,0                REM*
     06ED   00
0990 06EE FF80      CFF80   DATA >FF80                FOR*
0991 06F0 0000              DATA >0000                (LET)*
0992       06F3'    B02     EQU $+1
0993 06F2 0002              DATA >0002                DATA
0994 06F4 0028              DATA >0028                NEXT
0995 06F6 049E              DATA >049E                ERROR
0996 06F8 051C              DATA >051C                PRINT
0997 06FA 0018              DATA >0018                CALL
0998 06FC 0008              DATA >0008                LOAD
0999 06FE 052A              DATA >052A                INPUT
1000 0700 0008              DATA >0008                READ
1001 0702 93E8              DATA >93E8                RESTOR
1002 0704 74AA              DATA >74AA                RETURN
1003 0706 0020              DATA >0020                STOP
1004 0708 0028              DATA >0028                UNIT
1005 070A 000A              DATA >000A                TIME
1006 070C 000A              DATA >000A                SAVE
1007 070E 000A              DATA >000A                BASE
1008 0710 2C02              DATA >2C02                ESCAPE
1009 0712 00E6              DATA >00E6                NOESC
1010 0714 6BC8              DATA >6BC8                RANDOM
1011 0716 02E6              DATA >02E6                IMASK
1012 0718 001C              DATA >001C                IRTN
1013 071A 0020              DATA >0020                TRAP
1014 071C 0008              DATA >0008                BAUD
1015                *
```

```
1017                    *SYSTEM FUNCTION TABLE
1018                    *
1019 071E 9882  EDIL    DATA >9882              ABS
1020 0720 1CC2          DATA >1CC2              ASC
1021 0722 7502          DATA >7502              ATN
1022 0724 98C6          DATA >98C6              COS
1023 0726 860A          DATA >860A              EXP
1024 0728 8392          DATA >8392              INP
1025 072A 38D8          DATA >38D8              LOG
1026 072C CADC          DATA >CADC              NKY
1027 072E 7266          DATA >7266              SIN
1028 0730 9466          DATA >9466              SQR
1029 0732 9E66          DATA >9E66              SYS
1030 0734 1A68          DATA >1A68              TIC
1031                    *
1032                    *       ASSIGNABLE FUNCTIONS
1033                    *
1034 0736 A244          DATA >A244              BIT
1035 0738 1486          DATA >1486              CRB
1036 073A 3486          DATA >3486              CRF
1037 073C 695A          DATA >695A              MEM
1038 073E 25DA          DATA >25DA              MWD   2 MAY 1979 HJC
1039                    *
1040                    *       CHARACTER FUNCTIONS
1041                    *
1042 0740 7158          DATA >7158              LEN
1043 0742 40DA          DATA >40DA              MCH
1044 0744 44A6          DATA >44A6              SRH
1045      0746' EDILE   EQU $
1046                    *
1047                    *TRANSLATION TABLE INDEXED BY ASCII CODE.
1048                    *       NULLS ARE ILLEGAL.
1049                    *
1050 0746 4744  EDSL     DATA >4744,>3E43        ! " # $
     0748 3E43
1051 074A 4248          DATA >4248,>454C        % & ' (
     074C 454C
1052 074E 4D5F          DATA >4D5F,>5D3F        ) * + ,
     0750 5D3F
1053 0752 5C00          DATA >5C00,>5E00        - . / 0
     0754 5E00
1054 0756 0000          DATA >0000,>0000        1 2 3 4
     0758 0000
1055 075A 0000          DATA >0000,>0000        5 6 7 8
     075C 0000
1056 075E 003C          DATA >003C,>4059        9 : ; <
     0760 4059
1057 0762 5657  B56      DATA >5657,>413D        = > ? @
     0764 413D
1058 0766 4C46          DATA >4C46,>4D60        [ \ ] ↑
     0768 4D60
1059                    *
1060      0761' B59     EQU EDSL+27
1061      0763' B57     EQU EDSL+29
1062      0767' B46     EQU EDSL+33
```

```
1064                    *WORD OPERATERS
1065                    *
1066          076A'     EDGWOL  EQU $
1067 076A      54       LSTO    TEXT 'TO'
     076B      4F
1068 076C      00               BYTE 0,>38
     076D      38
1069 076E      54       LSTB    TEXT 'TAB'
     076F      41
     0770      42
1070 0771      00               BYTE 0,>39
     0772      39
1071 0773      53       LSST    TEXT 'STEP'
     0774      54
     0775      45
     0776      50
1072 0777      00               BYTE 0,>3A
     0778      3A
1073 0779      54       LSTH    TEXT 'THEN'
     077A      48
     077B      45
     077C      4E
1074 077D      00               BYTE 0,>3B
     077E      3B
1075 077F      4F       LSOR    TEXT 'OR'
     0780      52
1076 0781      00               BYTE 0,>4E
     0782      4E
1077 0783      4C       LSLOR   TEXT 'LOR'
     0784      4F
     0785      52
1078 0786      00               BYTE 0,>4F
     0787      4F
1079 0788      41       LSAN    TEXT 'AND'
     0789      4E
     078A      44
1080 078B      00               BYTE 0,>50
     078C      50
1081 078D      4C       LSLAN   TEXT 'LAND'
     078E      41
     078F      4E
     0790      44
1082 0791      00               BYTE 0,>51
     0792      51
1083 0793      4E       LSNT    TEXT 'NOT'
     0794      4F
     0795      54
1084 0796      00               BYTE 0,>52
     0797      52
1085 0798      4C       LSLNT   TEXT 'LNOT'
     0799      4E
     079A      4F
     079B      54
1086 079C      00               BYTE 0,>53
     079D      53
1087 079E      4C       LSLXO   TEXT 'LXOR'
     079F      58
     07A0      4F
     07A1      52
1088 07A2      00               BYTE 0,>54,0
```

```
07A3    54
07A4    00
```

```
1090                    *LIST TRANSLATION TABLE INDEXED BY PSEUDO CODE.
1091                    *       "← ARE UNDEFINED.
1092                    *
1093          07A5'     EDLC    EQU $
1094  07A5      3A      B3A     BYTE >3A          :
1095  07A6      40      B40     BYTE >40          @
1096  07A7      23              BYTE >23          #
1097  07A8      2C      B2C     BYTE >2C          ,
1098  07A9      3B      B3B     BYTE >3B          ;
1099  07AA      3F      B3F     BYTE >3F          ?
1100  07AB      25              BYTE >25          %
1101  07AC      24              BYTE >24          $
1102  07AD      22              BYTE >22          "
1103  07AE      27              BYTE >27          '
1104  07AF      5C              BYTE >5C          \
1105  07B0      21              BYTE >21          !
1106  07B1      26              BYTE >26          &
1107  07B2      03      B03     BYTE >03          ←     (UNUSED)
1108  07B3      5B      B5B     BYTE >5B          [
1109  07B4      5D      B5D     BYTE >5D          ]
1110  07B5      28              BYTE >28          (
1111  07B6      29              BYTE >29          )
1112  07B7      1B      B1B     BYTE >1B          OR    (UNUSED)
1113  07B8      6D      B6D     BYTE >6D          LOR   (UNUSED)
1114  07B9      4E      B4E     BYTE >4E          AND   (UNUSED)
1115  07BA      6F      B6F     BYTE >6F          LAND  (UNUSED)
1116  07BB      38      B38     BYTE >38          NOT   (UNUSED)
1117  07BC      22      DEFXB   BYTE DEFX-EDCL/2+1      LNOT (UNUSED)
1118  07BD      08      NXTXB   BYTE NXTX-EDCL/2+1      LXOR (UNUSED)
1119  07BE      3D              BYTE >3D          ==
1120  07BF      3D      B3D     BYTE >3D          =
1121  07C0      3E              BYTE >3E          >
1122  07C1      3E              BYTE >3E          >=
1123  07C2      3C              BYTE >3C          <
1124  07C3      3C              BYTE >3C          <=
1125  07C4      3C              BYTE >3C          <>
1126  07C5      2D              BYTE >2D          -
1127  07C6      2B              BYTE >2B          +
1128  07C7      2F              BYTE >2F          /
1129  07C8      2A              BYTE >2A          *
1130  07C9      5E              BYTE >5E          ↑
1131                    *
1132  07CA                      EVEN
```

```
1134                    *
1135                    *PROGRAM LOAD  FLAG
1136                    *       PLF =  1            SAVE        DIGITAL CASS.
1137                    *           =  0            LIST
1138                    *           = -1            SAVE 1,2    AUDIO CASS.
1139                    *
1140        0769'       LCN1    EQU EDLC->3C
1141        0688'       LCN2    EQU EDCL-2
1142        06E8'       LCN3    EQU EDIL->36
1143        005C        LCN4    EQU EDCS+2
1144        0010        LNXT    EQU NXTX-EDCL+2
1145        0044        LDEF    EQU DEFX-EDCL+2
1146                    *
1147  07CA  C019        SAV1     MOV *R9,R0         ;GET BUFFER ADR
1148  07CC  0420                 BLWP @CSWTZ
      07CE  0000
1149  07D0  102A                 JMP LST1           ;DONE
1150                    *
1151  07D2  06A0        SAV3     BL @TYP11          ;SEND DC2 TO START RECORD
      07D4  0000
1152  07D6  1200                   DATA >1200
1153  07D8  05E9                 INCT @PLF(9)       ;SET TO 1
      07DA  00CE'
1154  07DC  1014                 JMP LST            ;LIST SOURCE
1155                    *SAVE COMMAND
1156                    *
1157  07DE  0729        SAVY     SETO @PLF(9)       ;SET FLAG
      07E0  07DA'
1158  07E2  06A0                 BL @CKEX           ;LOOK FOR EXPRESSION
      07E4  002C'
1159  07E6  10F5                   JMP SAV3         ;NONE
1160  07E8  2EC1                 EVFIX R1
1161  07EA  0281                 CI R1,2
      07EC  0002
1162  07EE  1809                   JH ERR40
1163  07F0  C001                 MOV R1,R0
1164  07F2  0600                 DEC R0
1165  07F4  11EE                   JLT SAV3         ;SAVE 0 = SAVE
1166  07F6  06A0                 BL @CK302          ;302 THERE?
      07F8  0000'
1167  07FA  0420                 BLWP @CSOPZ        ;Y - OPEN FILE FOR WRITE
      07FC  0080'
1168  07FE  FFFF                 DATA -1            ;OPEN FOR WRITE FLAG
1169  0800  1004                 JMP LST0A
1170                    *
1171  0802  2FA0        ERR40    DATA ERROR2,46
      0804  002E
```

```
1173                    *LIST COMMAND
1174                    *
1175 0806 06A0  LST     BL @TYPC          ;OUT CRLF
     0808 03FA'
1176 080A C069  LST0A   MOV @LNUM(9),R1 ;GET INITIAL START
     080C 03D6'
1177 080E C229          MOV @VNT(9),R8  ;GET TABLE ADR
     0810 05CE'
1178 0812 0648          DECT R8           ;BACKUP
1179                    *
1180 0814 8A48  LST0    C R8,@SLT(9)      ;DONE?
     0816 05E0'
1181 0818 121D            JLE LST2        ;Y
1182 081A 0228          AI R8,-4          ;MOVE TO NEXT ENTRY
     081C FFFC
1183 081E 8601          C R1,*R8          ;SAME?
1184 0820 15F9            JGT LST0
1185 0822 0729          SETO @DCNT(9)     ;RESET INDENT COUNTER
     0824 0000
1186                    *
1187 0826 8A48  LST1    C R8,@SLT(9)      ;DONE?
     0828 0816'
1188 082A 1A14            JL LST2         ;Y
1189 082C C078          MOV *R8+,R1       ;N, GET LINE NUMBER
1190 082E C1D9          MOV *R9,R7        ;GET BUFFER ADR
1191 0830 06A0          BL @LSTL          ;LIST LINE
     0832 0886'
1192 0834 0228          AI R8,-6          ;PREPARE FOR NEXT LINE
     0836 FFFA
1193 0838 DDE0          MOVB @B0D,*R7+    ;OUT CRLF
     083A 0AA6'
1194 083C DDE0          MOVB @B0A,*R7+
     083E 06EB'
1195 0840 75D7          SB *R7,*R7
1196 0842 C029          MOV @PLF(9),R0    ;SAVE ?
     0844 07E0'
1197 0846 11C1            JLT SAV1        ;Y, AUDIO CASS.
1198 0848 06A0  LST1A   BL @TYPB          ;OUTPUT LINE
     084A 0000
1199 084C 06A0          BL @GETC          ;CHECK FOR ESC
     084E 00B6'
1200 0850 10EA            JMP LST1        ;NOTHING
1201 0852 10E9            JMP LST1        ;CHARACTER
1202                    *
1203 0854 C029  LST2    MOV @PLF(9),R0    ;EOP OR ESCAPE, SAVE?
     0856 0844'
1204 0858 1106            JLT LST3        ;Y
1205 085A 1307            JEQ LST4        ;LIST
1206 085C 0201          LI R1,LSTEM       ;OUT EOT
     085E 0AA4'
1207 0860 06A0          BL @TYPE          ;LIST
     0862 00D8'
1208 0864 1002          JMP LST4
1209                    *
1210 0866 0420  LST3    BLWP @CSCLZ       ;Y, CLOSE FILE
     0868 0000
1211 086A 0460  LST4    B @CRLF           ;DONE
     086C 006E'
```

```
1213                *LIST GOTO'S AND GOSUB'S
1214                *
1215 086E DDE0     LSTG1   MOVB @B2C,*R7+  ;OUT ",
     0870 07A8'
1216                *
1217 0872 D073     LSTG    MOVB *R3+,R1    ;GET LINE NUMBER
1218 0874 06C1             SWPB R1
1219 0876 D073             MOVB *R3+,R1
1220 0878 06C1             SWPB R1
1221 087A 2F41             OUTINT R1       ;CONVERT
1222 087C 9833             CB *R3+,@B3F    ;CHECK NEXT BYTE FOR ",
     087E 07AA'
1223 0880 13F6               JEQ LSTG1     ;Y, ANOTHER GOTO, OR GOSUB
1224 0882 0603             DEC R3          ;N, PROCESS
1225 0884 1037             JMP LSTLX
```

```
1227                *LIST LINE
1228                *       BL @LSTL
1229                *
1230                *       IN  R1 = LINE #
1231                *           R7 = IOB
1232                *          *R8 = PBC
1233                *       PRESERVE R6,R8
1234                *       R15 = SP
1235                *
1236 0886 C28B     LSTL    MOV R11,R10      ;SAVE RETURN
1237 0888 C0D8             MOV *R8,R3       ;GET PBC
1238 088A A0E9             A @BUS(9),R3     ;MAKE DISPLACEMENT INTO POINTER
     088C 05BC'
1239 088E 020F             LI R15,>2000     ;GET SPACE
     0890 2000
1240 0892 DDCF             MOVB R15,*R7+    ;OUT SPACE
1241 0894 2F41             OUTINT R1        ;CONVERT LINE NUMBER
1242 0896 C0A9             MOV @DCNT(9),R2  ;GET INDENT COUNT
     0898 0824'
1243 089A 9813             CB *R3,@B03      ;ELSE?
     089C 07B2'
1244 089E 1601               JNE LSTL1      ;N
1245 08A0 0642             DECT R2          ;Y, INDENT 2 MORE SPACES
1246                *
1247 08A2 DDCF     LSTL1   MOVB R15,*R7+    ;OUT SPACE
1248 08A4 0582             INC R2           ;MORE?
1249 08A6 11FD               JLT LSTL1      ;Y
1250 08A8 9813             CB *R3,@NXTXB    ;NEXT?
     08AA 07BD'
1251 08AC 1601               JNE LSTL2      ;N
1252 08AE 0607             DEC R7           ;Y, INDENT 1 LESS
1253                *
1254 08B0 DDCF     LSTL2   MOVB R15,*R7+    ;OUT SPACE
1255                *
1256 08B2 04C0     LSTL3   CLR R0
1257 08B4 D033             MOVB *R3+,R0     ;GET TYPE
1258 08B6 0280             CI R0,>0600      ;IMPLIED LET?
     08B8 0600
1259      08B8'    LETB    EQU $-2
1260 08BA 1308               JEQ LSTL4      ;Y
1261 08BC 06A0             BL @LWRD         ;LIST COMMAND TYPE
     08BE 0A6C'
1262 08C0 0688'              DATA LCN2      ;EDCL-2
1263 08C2 DDCF             MOVB R15,*R7+    ;OUT SPACE
1264 08C4 0280             CI R0,BCP*2      ;BIT,CRB,CRF,MEM?
     08C6 004A
1265 08C8 1A01               JL LSTL4       ;N
1266 08CA 0607             DEC R7           ;Y, ELIMINATE SPACE
1267                *
1268 08CC C380     LSTL4   MOV R0,R14       ;SAVE TYPE
1269 08CE 0280             CI R0,>3*2       ;CHECK TYPE
     08D0 0006
1270 08D2 11CF               JLT LSTG       ;GOTO OR GOSUB
1271 08D4 13EE               JEQ LSTL3      ;ELSE
1272 08D6 0280             CI R0,>5*2
     08D8 000A
1273 08DA 115D               JLT LSTRM      ;LT - REMARK STATEMENT
1274 08DC 1502               JGT LSTL5      ;GT - CONTINUE LOOKING
1275 08DE 0629             DEC @DCNT(9)     ;FOR, DECREMENT COUNTER
     08E0 0898'
```

```
1276                    *
1277 08E2 0280  LSTL5   CI R0,LNXT       ;NEXT?
     08E4 0010
1278 08E6 1605            JNE LSTL6     ;N
1279 08E8 05A9          INC @DCNT(9)    ;Y, INCREMENT COUNTER
     08EA 08E0'
1280 08EC 1102            JLT LSTL6     ;COUNTER OK
1281 08EE 0729          SETO @DCNT(9)   ;RESET COUNTER
     08F0 08EA'
1282 08F2 C380  LSTL6   MOV R0,R14      ;SAVE TYPE
```

```
1284 08F4 D033  LSTLX   MOVB *R3+,R0     ;GET CODE
1285 08F6 1336            JEQ LSTLE      ;DONE
1286 08F8 9800          CB R0,@B1B       ;USER FUNCTION?
     08FA 07B7'
1287 08FC 1A5A            JL LSTFNP      ;Y
1288 08FE 9800          CB R0,@B38       ;SYSTEM FUNCTION?
     0900 07BB'
1289 0902 1A5E            JL LSTSF       ;Y
1290 0904 0980          SRL R0,8         ;READY R0,R4
1291 0906 C100          MOV R0,R4
1292 0908 0280          CI R0,>3C        ;TO, TAB, STEP, THEN?
     090A 003C
1293 090C 1A63            JL LSTTT       ;Y
1294 090E 0280          CI R0,>4E        ;< OR?
     0910 004E
1295 0912 1A03            JL LSTLX0      ;Y
1296 0914 0280          CI R0,>54        ;> LXOR?
     0916 0054
1297 0918 125B            JLE LSTTI      ;N
1298              *
1299 091A 0280  LSTLX0  CI R0,>62        ;CHARACTER?
     091C 0062
1300 091E 1A24            JL LSTCR       ;Y
1301 0920 0280          CI R0,>6F        ;CONSTANT?
     0922 006F
1302 0924 1A62            JL LSTCN       ;Y, INTEGERS
1303 0926 1374            JEQ LSTCNF     ;Y, FLOATING POINT
1304              *
1305 0928 C169          MOV @VNT(9),R5   ;N, VARIABLE
     092A 0810'
1306 092C 0A10          SLA R0,1         ;X 2
1307 092E A140          A R0,R5          ;INDEX & GET VARIABLE
1308 0930 04C2          CLR R2
1309 0932 C165          MOV @-2*>70(5),R5
     0934 FF20
1310 0936 1503            JGT LSTVN      ;DIMENSIONED?
1311 0938 0505          NEG R5           ;Y
1312      093D' B4A     EQU $+3
1313 093A 0202          LI R2,>4A        ;OUT [
     093C 004A
1314              *
1315 093E 2160  LSTVN   COC @C380,R5      ;LETTER + NUMBER?
     0940 027C'
1316 0942 1608            JNE LSTVN1     ;N
1317 0944 06A0          BL @LWRDO        ;Y, OUT CHARACTER
     0946 0A92'
1318 0948 0921            SRL R1,2
1319 094A C045          MOV R5,R1        ;OUT #
1320 094C 0241          ANDI R1,>7F      ;MASK
     094E 007F
1321 0950 2F41          OUTINT R1        ;CONVERT
1322 0952 1005          JMP LSTVN2
1323              *
1324 0954 0A15  LSTVN1  SLA R5,1          ;REGULAR VARIABLE
1325 0956 020D          LI R13,LSTVN2
     0958 095E'
1326 095A 0460          B @LWRD2         ;LIST
     095C 0A76'
1327              *
1328 095E C102  LSTVN2  MOV R2,R4        ;"[ NEEDED?
```

```
1329 0960 162C              JNE LSTCR1      ;Y
1330 0962 10C8            JMP LSTLX
1331                 *
1332 0964 75D7  LSTLE     SB *R7,*R7        ;PUT NULL ON END
1333 0966 045A            B *R10
```

```
1335 0968 DDE4  LSTCR   MOVB @LCN1(4),*R7+  ;EDLC->3C
     096A 0769'
1336 096C 06C0          SWPB R0
1337 096E 06A0          BL @JMPR0          ;LOOK FOR CODES
     0970 0304'
1338 0972   21  LSTCRT    BYTE LSTCR0-LSTCRT/2,>44       "
     0973   44
1339 0974   21            BYTE LSTCR0-LSTCRT/2,>45       '
     0975   45
1340 0976   0C            BYTE LSTTR-LSTCRT/2,>47        !
     0977   47
1341 0978 0000            DATA 0
1342 097A 06A0          BL @LSTCR2         ;LOOK FOR DOUBLES
     097C 09A0'
1343              *                  ==    >=    <=    <>    ::
1344 097E 3D55            DATA >3D55,>3D58,>3D5A,>3E5B,>3A3C,>0000
     0980 3D58
     0982 3D5A
     0984 3E5B
     0986 3A3C
     0988 0000
1345              *
1346              *TAIL REMARK
1347              *
1348 098A 0607  LSTTR   DEC R7             ;BACKUP OVER !
1349 098C DDCF          MOVB R15,*R7+      ;OUT SPACE
1350 098E DDCF          MOVB R15,*R7+
1351 0990 DDE4          MOVB @LCN1(4),*R7+
     0992 0769'
1352 0994 1001          JMP LSTRM1
1353              *
1354              *REMARK ENTRY
1355              *
1356 0996 0607  LSTRM   DEC R7             ;REM, MOVE BACK 1 CHAR
1357              *
1358 0998 DDF3  LSTRM1  MOVB *R3+,*R7+     ;MOVE INTO LINE
1359 099A 16FE            JNE $-2
1360 099C 0607          DEC R7             ;BACKUP
1361 099E 045A          B *R10             ;RETURN
```

```
1363                    *
1364 09A0 D5FB  LSTCR2  MOVB *R11+,*R7  ;MOVE CHARACTER INTO LINE
1365 09A2 13A8            JEQ LSTLX     ;NOT FOUND
1366 09A4 9EC0          CB R0,*R11+     ;FOUND?
1367 09A6 16FC            JNE LSTCR2    ;N, KEEP LOOKING
1368 09A8 9837          CB *R7+,@B3A    ;":?
     09AA 07A5'
1369 09AC 16A3            JNE LSTLX     ;N
1370 09AE 0460  LSTL2P  B @LSTL2
     09B0 08B0'
1371                    *
1372 09B2 1032  LSTFNP  JMP LSTFN
1373 09B4 DDF3  LSTCR0  MOVB *R3+,*R7+  ;MOVE IN CHARACTER STRING
1374 09B6 16FE            JNE $-2       ;LOOP UNTIL NULL
1375 09B8 0607          DEC R7          ;BACKUP OVER NULL
1376                    *
1377 09BA DDE4  LSTCR1  MOVB @LCN1(4),*R7+ ;EDLC->3C
     09BC 0769'
1378 09BE 109A          JMP LSTLX
1379                    *
1380 09C0 06A0  LSTSF   BL @LWRD        ;LIST SYSTEM FUNCTION
     09C2 0A6C'
1381 09C4 06E8'           DATA LCN3     ;EDIL->36
1382 09C6 9813  LSTSF1  CB *R3,@B4A     ;(?
     09C8 093D'
1383 09CA 1394            JEQ LSTLX     ;Y, NO SPACE
1384 09CC DDCF  LSTSP   MOVB R15,*R7+   ;OUT SPACE
1385 09CE 1092  LSTLXP  JMP LSTLX
1386                    *
1387 09D0 0224  LSTTI   AI R4,->12      ;MOVE BACK BY THEN
     09D2 FFEE
1388                    *
1389 09D4 DDCF  LSTTT   MOVB R15,*R7+   ;OUT SPACE
1390 09D6 A104          A R4,R4         ;DOUBLE INDEX
1391 09D8 C164          MOV @LSTTL->70(4),R5    38*2
     09DA 09E6'
1392                    *
1393 09DC DDF5  LSTT1   MOVB *R5+,*R7+  ;MOVE LETTER
1394 09DE D015          MOVB *R5,R0     ;CHECK END
1395 09E0 16FD            JNE LSTT1     ;LOOP
1396 09E2 0284          CI R4,>38*2     ;THEN?
     09E4 0076
1397 09E6 13E3            JEQ LSTL2P    ;Y
1398 09E8 10F1          JMP LSTSP       ;OUT SPACE
```

```
1400 09EA 0280  LSTCN   CI R0,>6D         ;-1 THRU 9?
     09EC 006D
1401 09EE 1404            JHE LSTCN1      ;N
1402 09F0 0220          AI R0,->63        ;Y
     09F2 FF9D
1403 09F4 C040          MOV R0,R1
1404 09F6 1007          JMP LSTCN2        ;CONVERT
1405              *
1406 09F8 D073  LSTCN1  MOVB *R3+,R1      ;INTEGER
1407 09FA 06C1          SWPB R1
1408 09FC D073          MOVB *R3+,R1
1409 09FE 06C1          SWPB R1
1410      0A03' B6E     EQU $+3
1411 0A00 0280          CI R0,>6E         ;HEX?
     0A02 006E
1412 0A04 1302            JEQ LSTCN3      ;Y
1413              *
1414 0A06 2F41  LSTCN2  OUTINT R1          ;N, CONVERT
1415 0A08 10E2          JMP LSTLXP
1416              *
1417 0A0A 06A0  LSTCN3  BL @HOUT           ;OUT HEX
     0A0C 0000
1418 0A0E 10DF          JMP LSTLXP
1419              *
1420 0A10 2F13  LSTCNF  OUTFP *R3          ;OUTPUT FP #
1421 0A12 0223          AI R3,6           ;INCREMENT OVER NUMBER
     0A14 0006
1422 0A16 10DB          JMP LSTLXP
1423              *
1424              *USER FUNCTION ENTRY
1425              *
1426 0A18 DDE0  LSTFN   MOVB @B46,*R7+    ;OUT "F
     0A1A 0767'
1427 0A1C DDE0          MOVB @B4E,*R7+    ;OUT "N
     0A1E 07B9'
1428 0A20 0220          AI R0,>4000
     0A22 4000
1429 0A24 DDC0          MOVB R0,*R7+      ;OUT LETTER
1430 0A26 028E          CI R14,LDEF       ;DEF?
     0A28 0044
1431 0A2A 16CD            JNE LSTSF1      ;N, CHECK FOR (
1432 0A2C 9813          CB *R3,@B56       ;Y, ARGUMENTS?
     0A2E 0762'
1433 0A30 13CE            JEQ LSTLXP      ;N
1434 0A32 DDE0          MOVB @B5B,*R7+    ;Y, OUT "(
     0A34 07B3'
1435 0A36 C3A9          MOV @VNT(9),R14   ;GET TABLE ADR & CLEAR TYPE
     0A38 092A'
1436              *
1437 0A3A D7B3  LSTFD1  MOVB *R3+,*R14    ;SAVE DUMMY NAME
1438 0A3C C07E          MOV *R14+,R1
1439 0A3E 0821          SRA R1,2
1440 0A40 06A0          BL @LWRD01        ;MAKE LETTER & STORE
     0A42 0A9C'
1441 0A44 9813          CB *R3,@B56       ;=?
     0A46 0762'
1442 0A48 1303            JEQ LSTFD2      ;Y
1443 0A4A DDE0          MOVB @B2C,*R7+    ;N, OUT ",
     0A4C 07A8'
1444 0A4E 10F5          JMP LSTFD1        ;LOOP AGAIN
```

R NUMBER

& CLEAR TYPE

E

STORE

Note: If Program in Rom
Rlk. Lets Vnt in Rom.
∴ only outputs @ Adj.
Adjustments.

```
1445                    *
1446 0A50 DDE0  LSTFD2  MOVB @B5D,*R7+  ;OUT "]
     0A52 0784'
1447 0A54 108C          JMP LSTLXP
1448                    *
1449 0A56 076A' LSTTL   DATA LSTO,LSTB
     0A58 076E'
1450 0A5A 0773'         DATA LSST,LSTH
     0A5C 0779'
1451 0A5E 077F'         DATA LSOR,LSLOR
     0A60 0783'
1452 0A62 0788'         DATA LSAN,LSLAN
     0A64 078D'
1453 0A66 0793'         DATA LSNT,LSLNT
     0A68 0798'
1454 0A6A 079E'         DATA LSLXO
```

```
 1456                    *LIST WORD
 1457                    *       BL @LWRD
 1458                    *         ADR
 1459                    *
 1460 0A6C C13B  LWRD    MOV *R11+,R4    ;GET ADR OF WORD
 1461 0A6E C34B          MOV R11,R13     ;SAVE RETURN
 1462 0A70 0870          SRA R0,7        ;SWAP AND X 2
 1463 0A72 A100          A R0,R4         ;INDEX
 1464                    *
 1465 0A74 C154  LWRD1   MOV *R4,R5      ;GET WORD
 1466 0A76 06A0  LWRD2   BL @LWRDO       ;OUT FIRST CHARACTER
      0A78 0A92'
 1467 0A7A 0A71            SLA R1,7
 1468 0A7C 06A0          BL @LWRDO       ;OUT 2ND OR 5TH CHARACTER
      0A7E 0A92'
 1469 0A80 0A21            SLA R1,2
 1470 0A82 06A0          BL @LWRDO       ;OUT 3RD OR 6TH CHARACTER
      0A84 0A92'
 1471 0A86 0931            SRL R1,3
 1472 0A88 0224          AI R4,@LCN4     ;MOVE TO NEXT HALF (EDCS+2)
      0A8A 005C
 1473 0A8C 0915          SRL R5,1        ;ANOTHER HALF?
 1474 0A8E 18F2            JOC LWRD1     ;Y
 1475 0A90 045D          B *R13          ;N, RETURN
 1476                    *
 1477 0A92 C045  LWRDO   MOV R5,R1       ;LOAD TEMP
 1478 0A94 04BB          X *R11+         ;EXECUTE SHIFT
 1479 0A96 0241          ANDI R1,>1F00   ;MASK
      0A98 1F00
 1480 0A9A 1303            JEQ LWRDO2    ;RETURN
 1481                    *
 1482 0A9C 0221  LWRDO1  AI R1,>4000     ;ADD LETTER BITS
      0A9E 4000
 1483 0AA0 DDC1          MOVB R1,*R7+    ;MOVE OUT
 1484 0AA2 045B  LWRDO2  RT              ;RETURN
 1485                    *
 1486 0AA4   13  LSTEM   BYTE >13,>1B,>0D,>14,0
      0AA5   1B
      0AA6   0D
      0AA7   14
      0AA8   00
 1487       0AA6' BOD    EQU LSTEM+2
 1488 0AA9   10  BLKFWD  BYTE >10,>37,>0
      0AAA   37
      0AAB   00
 1489 0AAC               EVEN
 1490                    END
NO ERRORS,      NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES								
$		0AAC'		0092	0163	0297	0353	0361	0540	0655	0883	0966
				0972	0985	0987	0992	1045	1066	1093	1259	1312
				1359	1374	1410						
B01	R	03F0'	0010	0459	0462	0543						
B02		06F3'	0992	0467	0498	0510						
B03		07B2'	1107	1243								
B0A		06E8'	0987	1194								
B0D		0AA6'	1487	1193								
B1B		07B7'	1112	1286								
B20	R	0612'	0010	0471	0590	0851						
B2C		07A8'	1097	1215	1443							
B38		07BB'	1116	0494	1288							
B3A		07A5'	1094	0473	1368							
B3B		07A9'	1098	0232	0891							
B3D		07BF'	1120	0588								
B3F		07AA'	1099	0254	1222							
B40		07A6'	1095	0855								
B46		0767'	1062	1426								
B4A		093D'	1312	1382								
B4E		07B9'	1114	1427								
B56		0762'	1057	0457	1432	1441						
B57		0763'	1061	0453								
B59		0761'	1060	0455	0465							
B5B		07B3'	1108	0857	1434							
B5D		07B4'	1109	1446								
B6D		07B8'	1113	0285								
B6E		0A03'	1410	0282								
B6F		07BA'	1115	0291								
BCP		0025	0966	0989	1264							
BCPB		06EC'	0989	0492								
BLKFWD		0AA9'	1488	0129								
BUS	R	088C'	0011	0073	0621	0650	0794	1238				
C380		027C'	0353	1315								
CFF80		06EE'	0990	0349								
CK302	D	0000'	0037	0005	0044	0087	1166					
CKEX	R	07E4'	0012	0060	1158							
CM4		0572'	0755	0678								
CONT		0016'	0049	0982								
CRCASS	R	00C2'	0010	0122								
CRLF	R	086C'	0026	0080	1211							
CSCLZ	R	0868'	0009	1210								
CSOPZ	R	07FC'	0009	0088	1167							
CSRDZ	R	0090'	0009	0095								
CSWTZ	R	07CE'	0009	1148								
CVDB20	R	0264'	0012	0343								
CVDIFZ	R	01D4'	0013	0271								
CVDIZ	R	026C'	0013	0158	0256	0346						
DCNT	R	08F0'	0025	1185	1242	1275	1279	1281				
DDM	R	0524'	0011	0708								
DEFX		06CC'	0963	1117	1145							
DEFXB		07BC'	1117	0563								
DLC	R	053A'	0011	0720								
EBP	R	0488'	0011	0093	0148	0536	0561	0612	0648			
ED10		01A0'	0245	0227	0230	0235	0242					
ED11		01B6'	0254	0263								
ED12		01BA'	0256	0248								
ED13		01CE'	0265	0203								
ED14		01D0'	0267	0251								
ED20		01D2'	0271	0334	0442							
ED20H		01EE'	0282	0274								

```
ED21        01F4'   0285  0276  0278
ED21A       01F8'   0286  0283
ED22        01FA'   0287  0280
ED23        0200'   0291  0272
ED24        0212'   0299  0289
ED3         0114'   0171  0166  0249  0476
ED30        0214'   0304  0273
ED31        0232'   0315  0312
ED32        023E'   0323  0337
ED33        0252'   0332  0406
ED34        0258'   0336  0326
ED35        0282'   0362  0338
ED35A       028C'   0367  0363
ED36        0292'   0373  0365
ED37        029A'   0376  0381
ED38        02B2'   0392  0377
ED38A       02C2'   0398  0401
ED39        02D2'   0405  0379
ED40        031E'   0441  0458  0466  0501  0893
ED41        0320'   0442  0450  0460  0463  0468  0562  0564
ED41P       0362'   0468  0589
ED42        0324'   0444  0434  0435  0448
ED43        0332'   0452  0438
ED44        034E'   0462  0454  0456
ED45        0354'   0465  0439
ED46        0364'   0470  0472  0474
ED47        0366'   0471  0436  0895
ED49        037E'   0483  0420
ED5         013E'   0193  0198  0221
ED50        038A'   0490  0430  0431
ED50A       0396'   0494  0491
ED50B       03A6'   0500  0493  0495
ED51        03AA'   0505  0432  0433
ED52        038A'   0512  0506
ED54        038E'   0517  0437
ED6         0154'   0205  0196
ED7         015E'   0210  0217
ED8         0172'   0219  0214
ED9         0196'   0240  0208  0212
ED90        03C4'   0524  0414
ED91        03FC'   0548  0544
EDCL        068A'   0932  0240  0966  0974  0975  0976  0985  1117  1118  1141
                          1144  1145
EDCLE       06DC'   0972  0197
EDCS        005A    0985  0207  1143
EDER    R   0472'   0013  0223  0385  0478  0485  0550  0592  0595  0598
EDER2       0400'   0550  0446  0509  0525  0527  0587
EDER3       0178'   0223  0159  0162  0258
EDER4       02AA'   0385  0347  0350
EDER6       0378'   0478  0423  0426
EDER8       046A'   0595  0556
EDER9       0470'   0598  0574  0578
EDFN        0406'   0555  0313
EDFN0       042A'   0570  0566
EDFN1       0432'   0573  0582
EDFN2       044C'   0584  0580
EDFN3       0458'   0588  0568  0585  0591
EDGL        0618'   0855  0175  0185  0213  0308  0315
EDGL1       062E'   0864  0856  0858  0866
EDGLS       0610'   0851  0171  0305  0555  0573  0852  0871
```

```
EDGP      0666'   0903  0362  0565  0577  0906
EDGP1     067E'   0912  0908  0910
EDGWO     0630'   0870  0304  0484
EDGWO1    063E'   0876  0886
EDGWO2    0640'   0878  0881
EDGWO4    0654'   0889  0879
EDGWO5    0662'   0895  0892
EDGWOL    076A'   1066  0874
EDIF      017E'   0226  0182
EDIL      071E'   1019  0321  0327  1142
EDILE     0746'   1045  0336
EDIT    D 00DC'   0143  0004  0113
EDLC      07A5'   1093  1140
EDLET     014C'   0200  0176  0186
EDLETA    014E'   0202  0233
EDON      0184'   0229  0184
EDOP      02DA'   0412  0306  0429  0513
EDOP1     02FA'   0425  0418
EDOPTB    0306'   0429  0429  0430  0431  0432  0433  0434  0435  0436  0437
                        0438  0439
EDREM     03C0'   0519  0252  0520
EDSCI     018A'   0232  0172
EDSCL     0680'   0927  0191
EDSL      0746'   1050  0425  1060  1061  1062
EDV1      0260'   0342  0309
EDV1A     027E'   0355  0344
EDV1B     0280'   0357  0348
EDV2      0282'   0361  0316
EMV       0476'   0607
EMV0      0478'   0609  0167  0532
EMV1      0494'   0626  0633
EMV2      04A6'   0637  0627  0630
EMV3      04B6'   0648  0770
EMV4      04C8'   0664  0756
EMV4A     04CC'   0666  0670  0679  0683
EMV4B     04E0'   0675  0677
EMV4C     04F2'   0681  0672
EMV4E     04FA'   0687  0667
EMV4F     04FE'   0689  0698  0703
EMV4H     0516'   0700  0696
EMV4I     051A'   0702  0694
EMV4J     0520'   0707  0690
EMV4K     0532'   0716  0712
EMV4KA    0538'   0720  0714
EMV4L     053C'   0724  0710  0717
EMV4M     0540'   0726  0738
EMV4N     0550'   0735  0731
EMV4O     0552'   0736  0727  0729  0733
EMV5      055C'   0744  0629
EMV6      0576'   0760  0747
EMV6A     058A'   0768  0765
EMVA      0592'   0779  0643  0754
EMVA1     0588'   0793  0766
EMVA2     05DE'   0809  0814
EMVA3     05EC'   0816  0810
EMVE10    0590'   0772  0764  0787
EMVR      055A'   0740  0610
EORBUS  R 0044'   0011  0069
ERR10     0280'   0388  0395
ERR13     04A4'   0635  0638
```

ERR40		0802'	1171	1162				
ERR41		000A'	0040	0038				
ERR44		0070'	0082	0097				
ERR5		02A8'	0383					
ERROR		2F80	0032	0033	0383	0388	0635	0772
ERROR2		2FA0	0033	0040	0082	1171		
EVFIX			0028					
FNS	R	04FC'	0014	0687				
FOR2	R	0512'	0015	0697				
FPAC	R	020A'	0016	0293				
GETC	R	084E'	0017	0114	1199			
GETCR	R	00A0'	0017	0103				
GOS2	R	0024'	0025	0052				
GSC	R	04EE'	0014	0664	0676	0678		
GSS	R	0556'	0014	0666	0737			
HOUT	R	0A0C'	0027	1417				
ID302	R	0002'	0020	0037				
IFB		0042	0974	0226				
IFX		06CA'	0962	0974				
JMPR0	R	0970'	0015	0428	1337			
LCN1	D	0769'	1140	0006	1335	1351	1377	
LCN2	D	0688'	1141	0006	1262			
LCN3	D	06E8'	1142	0006	1381			
LCN4	D	005C	1143	0006	1472			
LDEF	D	0044	1145	0007	1430			
LDP0		006C'	0080	0078				
LDP01		0074'	0084	0064				
LDP02		0088'	0092	0128				
LDP1		0096'	0099	0131				
LDP2		0098'	0101	0110	0111			
LDP2A		00A4'	0106	0102				
LDP2B		00A6'	0108	0104				
LDP3		00B0'	0113	0109				
LDP4		00BC'	0118	0096				
LDP5	D	00CC'	0127	0004	0061	0086	0115	0116
LDPY	D	0026'	0059	0004				
LDSTRT	D	0038'	0066	0005				
LETB		08B8'	1259	0202				
LINE2	R	03FE'	0017	0548				
LNUM	R	080C'	0014	0161	0531	1176		
LNXT	D	0010	1144	0007	1277			
LSAN		0788'	1079	1452				
LSLAN		078D'	1081	1452				
LSLNT		0798'	1085	1453				
LSLOR		0783'	1077	1451				
LSLXO		079E'	1087	1454				
LSNT		0793'	1083	1453				
LSOR		077F'	1075	1451				
LSST		0773'	1071	1450				
LST		0806'	1175	0981	1154			
LST0		0814'	1180	1184				
LST0A		080A'	1176	1169				
LST1		0826'	1187	1149	1200	1201		
LST1A		0848'	1198					
LST2		0854'	1203	1181	1188			
LST3		0866'	1210	1204				
LST4		086A'	1211	1205	1208			
LSTB		076E'	1069	1449				
LSTCN		09EA'	1400	1302				
LSTCN1		09F8'	1406	1401				

```
LSTCN2      0A06'   1414   1404
LSTCN3      0A0A'   1417   1412
LSTCNF      0A10'   1420   1303
LSTCR       0968'   1335   1300
LSTCR0      09B4'   1373   1338   1339
LSTCR1      09BA'   1377   1329
LSTCR2      09A0'   1364   1342   1367
LSTCRT      0972'   1338   1338   1339   1340
LSTEM       0AA4'   1486   1206   1487
LSTFD1      0A3A'   1437   1444
LSTFD2      0A50'   1446   1442
LSTFN       0A18'   1426   1372
LSTFNP      09B2'   1372   1287
LSTG        0872'   1217   1270
LSTG1       086E'   1215   1223
LSTH        0779'   1073   1450
LSTL    D   0886'   1236   0004   1191
LSTL1       08A2'   1247   1244   1249
LSTL2       08B0'   1254   1251   1370
LSTL2P      09AE'   1370   1397
LSTL3       08B2'   1256   1271
LSTL4       08CC'   1268   1260   1265
LSTL5       08E2'   1277   1274
LSTL6       08F2'   1282   1278   1280
LSTLE       0964'   1332   1285
LSTLX       08F4'   1284   1225   1330   1365   1369   1378   1383   1385
LSTLX0      091A'   1299   1295
LSTLXP      09CE'   1385   1415   1418   1422   1433   1447
LSTO        076A'   1067   1449
LSTRM       0996'   1356   1273
LSTRM1      0998'   1358   1352
LSTSF       09C0'   1380   1289
LSTSF1      09C6'   1382   1431
LSTSP       09CC'   1384   1398
LSTT1       09DC'   1393   1395
LSTTI       09D0'   1387   1297
LSTTL       0A56'   1449   1391
LSTTR       098A'   1348   1340
LSTTT       09D4'   1389   1293
LSTVN       093E'   1315   1310
LSTVN1      0954'   1324   1316
LSTVN2      095E'   1328   1322   1325
LWRD        0A6C'   1460   1261   1380
LWRD1       0A74'   1465   1474
LWRD2       0A76'   1466   1326
LWRD0       0A92'   1477   1317   1466   1468   1470
LWRD01      0A9C'   1482   1440
LWRD02      0AA2'   1484   1480
MODE    R   03F6'   0018   0049   0545
MOVE        05F0'   0826   0791   0803
MOVE1       05F6'   0830   0833
MOVE2       05FE'   0835   0828
MOVE3       0604'   0839   0844
MOVE4       062E'   0866   0827   0831   0841
NLIN    R   00CA'   0017   0126
NVD     R   05D6'   0018   0071   0072   0392   0396   0761   0780   0790   0795   0802
NVS     R   05A4'   0018   0394   0763   0786
NXTX        0698'   0937   1118   1144
NXTXB       078D'   1118   1250
ONB         0040    0975   0229
```

```
ONX          06C8'   0961  0975
OUTFP                0029
OUTINT               0030
PLF       R  0856'   0018  0059  0090  0101  0118  0127  1153  1157  1196  1203
PRGM         000E'   0044  0983
PROGM     R  0014'   0009  0045
PRTX         069C'   0939  0976
PTRB         0014    0976  0234
PUTB      R  00A8'   0019  0108
R0           0000          0084  0085  0094  0106  0153  0154  0165  0165  0174
                           0177  0181  0183  0187  0195  0205  0207  0210  0219
                           0243  0262  0307  0310  0311  0317  0325  0342  0357
                           0367  0378  0402  0405  0405  0412  0413  0415  0417
                           0419  0422  0447  0530  0535  0542  0637  0637  0689
                           0716  0726  0746  0746  0761  0762  0763  0783  0784
                           0785  0786  0835  0836  0837  0862  1147  1163  1164
                           1196  1203  1256  1257  1258  1264  1268  1269  1272
                           1277  1282  1284  1286  1288  1290  1291  1292  1294
                           1296  1299  1301  1306  1307  1336  1366  1394  1400
                           1402  1403  1411  1428  1429  1462  1463
R1           0001          0051  0069  0070  0071  0129  0160  0161  0174  0177
                           0187  0193  0194  0195  0215  0216  0240  0241  0259
                           0260  0261  0275  0277  0279  0286  0287  0288  0307
                           0310  0317  0323  0324  0325  0349  0351  0415  0416
                           0421  0425  0427  0531  0557  0558  0575  0576  0581
                           0584  0586  0609  0609  0628  0645  0859  0860  0861
                           0878  0880  0882  0885  0903  0904  0905  0907  0909
                           1160  1161  1163  1176  1183  1189  1206  1217  1218
                           1219  1220  1221  1241  1318  1319  1320  1321  1403
                           1406  1407  1408  1409  1414  1438  1439  1467  1469
                           1471  1477  1479  1482  1483
R10          000A          0143  0607  0740  1236  1333  1361
R11          000B          0143  0327  0329  0607  0744  0745  0751  0760  0779
                           0863  0870  0911  1236  1364  1366  1460  1461  1478
R12          000C          0122  0267  0490
R13          000D          0765  0779  0782  0784  0788  0789  0790  0816  0817
                           1325  1461  1475
R14          000E          1268  1282  1430  1435  1437  1438
R15          000F          0206  0220  1239  1240  1247  1254  1263  1349  1350
                           1384  1389
R2           0002          0062  0063  0066  0067  0068  0072  0073  0074  0075
                           0076  0077  0084  0205  0219  0392  0393  0393  0394
                           0396  0398  0399  0399  0400  0612  0613  0616  0617
                           0681  0700  0718  0735  0751  0760  0762  0785  0797
                           0798  0799  0800  0801  0802  0812  1242  1245  1248
                           1308  1313  1328
R3           0003          0101  0127  0173  0200  0292  0296  0342  0351  0352
                           0357  0536  0538  0539  0559  0560  0561  0563  0570
                           0579  0642  0682  0701  0720  0752  0753  0781  0793
                           0794  0796  0807  0809  0811  0812  0813  0816  0826
                           0830  0832  0836  0840  0876  0880  0890  1217  1219
                           1222  1224  1237  1238  1243  1250  1257  1284  1358
                           1373  1382  1406  1408  1420  1421  1432  1437  1441
R4           0004          0191  0193  0197  0226  0229  0234  0241  0245  0246
                           0247  0250  0293  0295  0321  0323  0328  0329  0330
                           0332  0333  0336  0373  0380  0407  0408  0571  0576
                           0626  0648  0653  0664  0666  0668  0669  0671  0673
                           0681  0687  0689  0691  0702  0707  0708  0709  0711
                           0713  0716  0718  0724  0726  0728  0730  0732  0735
                           0736  0736  0737  0780  0783  0795  0830  0835  0839
```

				0840	0842	0874	0878	0882	0884	0885	0889	1291
				1328	1387	1390	1390	1396	1460	1463	1465	1472
R5		0005		0093	0094	0106	0149	0150	0151	0152	0154	0210
				0211	0216	0374	0376	0378	0400	0402	0673	0675
				0676	0691	0692	0693	0695	0700	0701	0781	0782
				0796	0797	0826	0832	0837	0839	0843	0870	0872
				0887	1305	1307	1309	1311	1315	1319	1324	1391
				1393	1394	1465	1473	1477				
R6		0006		0146	0147	0368	0369	0496	0497	0500	0505	0507
				0508	0524	0526	0619	0620	0626	0628	0631	0632
				0642	0645	0646	0649	0649	0650	0651	0669	0693
				0709	0728	0752	0769	0798	0807			
R7		0007		0099	0145	0164	0173	0200	0206	0220	0232	0265
				0299	0345	0355	0364	0413	0445	0470	0471	0473
				0483	0519	0533	0534	0535	0538	0567	0588	0590
				0620	0621	0632	0646	0651	0653	0671	0695	0711
				0730	0745	0793	0851	0853	0855	0857	0860	0873
				0876	0890	0904	1190	1193	1194	1195	1195	1215
				1240	1247	1252	1254	1263	1266	1332	1332	1335
				1348	1349	1350	1351	1356	1358	1360	1364	1368
				1373	1375	1377	1384	1389	1393	1426	1427	1429
				1434	1443	1446	1483					
R8		0008		0148	0202	0246	0254	0259	0261	0267	0282	0285
				0286	0288	0291	0295	0333	0408	0427	0441	0444
				0445	0447	0449	0449	0452	0453	0455	0457	0462
				0475	0490	0498	0510	0512	0517	0519	0528	0528
				0529	0529	0539	0542	0543	0558	0559	0575	0613
				0614	0615	0616	0654	0889	0891	1177	1178	1180
				1182	1183	1187	1189	1192	1237			
R9		0009		0099	0145	0533	0707	1147	1190			
RUNP	R	06DC'	0019	0980								
RUNP1	R	006A'	0010	0079								
SAV1		07CA'	1147	1197								
SAV3		07D2'	1151	1159	1165							
SAVY	D	07DE'	1157	0005								
SIZE	R	06DE'	0019	0981								
SLN	R	0020'	0016	0051								
SLT	R	0828'	0016	0067	0074	0619	0799	0809	1180	1187		
SSP	R	03C6'	0021	0146	0508	0524						
STCK	R		0020									
			0024									
STRTC	R	0064'	0010	0077								
TYP11	R	07D4'	0022	1151								
TYPB	R	084A'	0023	1198								
TYPC	R	0808'	0022	0050	0546	1175						
TYPE	R	0862'	0023	0130	1207							
UFT	R	053E'	0021	0724								
VDT	R	05D2'	0021	0070	0376	0403	0789	0801				
VNT	R	0A38'	0021	0068	0075	0149	0374	0571	0788	0800	1177	1305
				1435								

*

```
SOURCE ACCESS NAME=      DBASIC.REL4.BASIC.SOURCE.SIZE
OBJECT ACCESS NAME=      DBASIC.REL4.BASIC.OBJECT.SIZE
LISTING ACCESS NAME=     DBASIC.REL4.BASIC.OBJECT.LIST
ERROR ACCESS NAME=
OPTIONS=
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'SIZE'
0002            *       25 APRIL 1979
0003            *
0004                    DXOP OUTINT,13
0005                    DEF SIZE
0006                    DEF SIZE1
0007                    DEF MOVEB
0008                    DEF MOVEL
0009                    REF VDT,BUS,NVS,NVD,VNT
0010                    REF ERRS2
0011                    REF HOUT        ;30 APRIL 1979 HJC SPR 27
0012            *
0013            *SIZE ROUTINE
0014            *
0015 0000 06A0  SIZE    BL @MOVEB       ;OUT 'PRGM:'
     0002 007A'
0016 0004 0A0D            DATA >0A0D
0017 0006   50            TEXT 'PRGM:'
     0007   52
     0008   47
     0009   4D
     000A   3A
0018 000B   00            BYTE 0
0019 000C C069          MOV @NVD(9),R1  ;PRGM=((NVD-VDT)+VNT)-BUS
     000E 0000
0020 0010 6069          S   @VDT(9),R1
     0012 0000
0021 0014 A069          A   @VNT(9),R1
     0016 0000
0022 0018 6069          S   @BUS(9),R1
     001A 0000
0023 001C 0221          AI R1,8         ;ADD PROM OVERHEAD
     001E 0008
0024 0020 06A0          BL @HOUT        ; PUT HEX NUMBER IN BUFFER SPR 27
     0022 0000
0025 0024 06A0          BL @MOVEL
     0026 007C'
0026 0028   20            TEXT ' BYTES'
     0029   42
     002A   59
     002B   54
     002C   45
     002D   53
0027 002E   0A            BYTE >0A,>0D
     002F   0D
0028 0030   56            TEXT 'VARS:'
     0031   41
     0032   52
     0033   53
     0034   3A
0029 0035   00            BYTE 0
0030 0036 C059          MOV *R9,R1      ;VARS=IOB-NVS+NVD-VDT
0031 0038 6069          S @NVS(9),R1
     003A 0000
0032 003C A069          A @NVD(9),R1
     003E 000E'
0033 0040 6069          S @VDT(9),R1
     0042 0012'
0034 0044 0221          AI R1,-8        ;REMOVE EPROM OVERHEAD
     0046 FFF8
```

```
0035 0048 06A0          BL @HOUT        ;PUT HEX NUMBER IN BUFFER SPR 27
     004A 0022'
0036 004C 06A0          BL @MOVEL
     004E 007C'
0037 0050   20            TEXT ' BYTES'
     0051   42
     0052   59
     0053   54
     0054   45
     0055   53
0038 0056   0A            BYTE >0A,>0D
     0057   0D
0039 0058   46            TEXT 'FREE:'
     0059   52
     005A   45
     005B   45
     005C   3A
0040 005D   00            BYTE 0
0041 005E C069          MOV @NVS(9),R1  ;FREE=NVS-NVD
     0060 003A'
0042 0062 6069          S @NVD(9),R1
     0064 003E'
0043                *
0044 0066 06A0  SIZE1   BL @HOUT        ;OUTPUT HEX NUMBER SPR 27
     0068 004A'
0045 006A 06A0          BL @MOVEL
     006C 007C'
0046 006E   20            TEXT ' BYTES '
     006F   42
     0070   59
     0071   54
     0072   45
     0073   53
     0074   20
0047 0075   00            BYTE 0
0048 0076 0460          B @ERRS2
     0078 0000
0049                *
0050                *MOVE STRING ROUTINES
0051                *
0052                *       BL @MOVEB       MOVE INTO IOB
0053                *       BL @MOVEL       MOVE INTO R7
0054                *
0055                *       IN  0(11) = STRING
0056                *               R7 = OBC
0057                *       OUT     R7 = UPDATED OBC
0058                *       NOTE: STRINGS MUST BE OF EVEN LENGTH
0059                *
0060 007A C1D9  MOVEB   MOV *R9,R7      ;GET BUFFER ADR
0061 007C DDFB  MOVEL   MOVB *R11+,*R7+ ;MOVE CHARACTER
0062 007E 16FE            JNE $-2       ;UNTIL NULL
0063 0080 0607          DEC R7          ;BACKUP OVER NULL
0064 0082 045B          RT
0065                    END
NO ERRORS
*
```

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL5.BASIC.SOURCE.INPUT
OBJECT ACCESS NAME=     BD1.DBASIC.REL5.BASIC.OBJECT.INPUT
LISTING ACCESS NAME=    BD1.DBASIC.REL5.BASIC.LIST.INPUT
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'INPUT'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *
0006              * ROUTINE LIST:
0007              *
0008              *       INPY              ;INPUT ROUTINE
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MARCO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:
0015              *
0016                      REF INPEND        ;QUIT INPUT
0017                      REF OUTL1         ;OUT LINE
0018                      REF CVDIFZ        ;CONVERT DECIMAL TO INTEGER/FP
0019                      REF EVARZ         ;EVALUATE VARIABLE
0020                      REF EVSDZ         ;EVALUATE STRING
0021                      REF GOSB1         ;GOSUB ENTRY
0022                      REF BMVE          ;BYTE MOVE
0023                      REF JMPR0A        ;BYTE MULTIPLEXOR JUMP
0024                      REF PUTB          ;PUT BYTE IN BUFFER
0025                      REF GETCR         ;GET CHARACTER
0026                      REF TYP0          ;ECHO R0
0027                      DXOP STORE,1      ;STORE FPAC
0028                      DXOP EVFIX,11     ;EVALUATE AND FIX
0029              *
0030              * EXTERNAL DATA:
0031              *
0032                      REF FFLG          ;FORMATTING FLAG
0033                      REF HFLG          ;HELP FLAG
0034                      REF IFLG          ;INPUT FLAG
0035              *
0036              * MODULE EQUATES:  (NONE)
0037              *
0038              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0041            * ABSTRACT:
0042            *
0043            *       EXECUTE THE INPUT COMMAND
0044            *
0045            * CALLING SEQUENCE:
0046            *
0047            *       B @INPY
0048            *
0049            *       EXIT TO INPEND, GOSB1
0050            *
0051            *       R7 = OBC
0052            *       R8 = PBC
0053            *       R15 = CR/LF FLAG
0054            *
0055            * EXCEPTIONS AND CONDITIONS:  (NONE)
0056            *
0057            * EXTERNAL ROUTINE LIST:
0058            *
0059            *      (SEE EXTERNAL ROUTINES)
0060            *
0061            * LOCAL DATA:
0062            *
0063            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0065                    * ENTRY POINT:
0066                    *
0067                          DEF INPY
0068                    *
0069 0000 04E9  INPY    CLR @HFLG(9)     ;CLEAR HELP FLAG
     0002 0000
0070 0004 04E9          CLR @IFLG(9)     ;CLEAR INPUT FLAG
     0006 0000
0071 0008 1003          JMP INP1
0072                    *
0073                    *HELP QUERY SETUP
0074                    *
0075 000A 2EE9  INPQM   EVFIX @HFLG(9)   ;GET LINE #
     000C 0002'
0076                    *
0077 000E 0608  INP0    DEC R8           ;BACKUP TO DELIMITER
0078                    *
0079 0010 070F  INP1    SETO R15         ;SET FOR CR
0080                    *
0081 0012 058F  INPSC   INC R15          ;SET FOR NO CRLF
0082 0014 06A0          BL @JMPR0A       ;SWITCH BOARD
     0016 0000
0083 0018   49  INPTB     BYTE INPE-INPTB/2,>00   ;NULL
     0019   00
0084 001A   49            BYTE INPE-INPTB/2,>3C   ;:
     001B   3C
0085 001C   49            BYTE INPE-INPTB/2,>47   ;1
     001D   47
0086 001E   FC            BYTE INP1-INPTB/2,>3F   ;,
     001F   3F
0087 0020   FD            BYTE INPSC-INPTB/2,>40  ;;
     0021   40
0088 0022   2D            BYTE INPFM-INPTB/2,>3E  ;#
     0023   3E
0089 0024   F9            BYTE INPQM-INPTB/2,>41  ;?
     0025   41
0090 0026   28            BYTE INPFL-INPTB/2,>42  ;%
     0027   42
0091 0028 0000            DATA 0
0092 002A 0608          DEC R8           ;NONE OF ABOVE, TRY STRING
0093 002C 0420          BLWP @EVSDZ
     002E 0000
0094 0030 1023            JMP INPQ       ;STRING
0095 0032 102E            JMP INPD       ;$VAR
0096 0034 0420          BLWP @EVARZ      ;NEITHER, TRY VARIABLE
     0036 0000
0097 0038 C182          MOV R2,R6        ;SAVE
0098 003A 0202          LI R2,INM1       ;'? '
     003C 00FA'
0099                    *
0100 003E 06A0  INP3    BL @INPP         ;PROMPT AND GET STRING
     0040 00AE'
0101 0042 0420          BLWP @CVDIFZ     ;CONVERT
     0044 0000
0102 0046 1008            JMP INP4       ;FP
0103 0048 04C1            CLR R1         ;NO NUMBER
0104 004A 1000            NOP
0105 004C C000          MOV R0,R0        ;CHECK DELIMITER
0106 004E 1608            JNE INP5       ;PROBLEM
0107 0050 04F6          CLR *R6+         ;INTEGER
```

```
0108 0052 CD81          MOV R1,*R6+
0109 0054 04F6          CLR *R6+
0110 0056 10DB          JMP INP0        ;LOOK AT DELIMITER
```

```
0112                    *GET NUMERIC INPUT
0113                    *
0114 0058 C000  INP4    MOV R0,R0           ;EOL?
0115 005A 1602            JNE INP5          ;N, PROBLEM
0116 005C 2C56          STORE *R6           ;MOVE NUMBER
0117 005E 10D7          JMP INP0
0118                    *
0119                    *ERROR IN NUMERIC INPUT
0120                    *
0121 0060 0202  INP5    LI R2,INM2          ;OUT '?? '
     0062 00FD'
0122 0064 C069          MOV @HFLG(9),R1     ;HELP WANTED?
     0066 000C'
0123 0068 13EA            JEQ INP3          ;N
0124 006A 0700          SETO R0             ;YES, RETURN -1
0125 006C 1040          JMP INPP6
0126                    *
0127                    *% FORMATTING
0128                    *
0129 006E 0729  INPFL   SETO @IFLG(9)       ;SET FLAG
     0070 0006'
0130                    *
0131                    *# FORMATTING
0132                    *
0133 0072 2EE9  INPFM   EVFIX @FFLG(9)      ;GET FLAG
     0074 0000
0134 0076 10CB          JMP INP0
0135                    *
0136                    *OUT LITERAL
0137                    *
0138 0078 020B  INPQ    LI R11,INP0         ;GET RETURN ADR
     007A 000E'
0139                    *
0140                    *OUT STRING
0141                    *       R2 = STRING ADR
0142                    *
0143 007C C34B  OUTL    MOV R11,R13         ;SAVE RETURN
0144 007E C1D9          MOV *R9,R7          ;SET R7 (WSBC)
0145 0080 0205          LI R5,132           ;GET MAX
     0082 0084
0146 0084 06A0          BL @BMVE            ;MOVE TO OUTPUT BUFFER
     0086 0000
0147 0088 1000            NOP
0148 008A C2CD          MOV R13,R11
0149 008C 0460          B @OUTL1
     008E 0000
```

```
0151                *GET STRING
0152                *
0153 0090 C182      INPD    MOV R2,R6       ;SAVE ADR
0154 0092 0202              LI R2,INM3      ;GET ': '
     0094 0101'
0155 0096 06A0              BL @INPP        ;PROMPT AND GET CHARACTERS
     0098 00AE'
0156 009A C087              MOV R7,R2       ;SET FOR MOVE
0157 009C C1C6              MOV R6,R7
0158 009E 0205              LI R5,80        ;MAXIMUM OF EIGHTY CHARACTERS
     00A0 0050
0159 00A2 06A0              BL @BMVE        ;MOVE INTO VARIABLE
     00A4 0086'
0160 00A6 1083                JMP INP0      ;OK
0161 00A8 1082              JMP INP0
0162                *
0163                *END INPUT LINE
0164                *
0165 00AA 0460      INPE    B @INPEND
     00AC 0000
```

```
0167                    *PROMPT AND GET CHARACTERS
0168                    *
0169 00AE C38B  INPP    MOV R11,R14       ;SAVE RETURN
0170 00B0 C3CF          MOV R15,R15       ;SUPPRESS PROMPT?
0171 00B2 1602            JNE INPP1       ;Y
0172 00B4 06A0          BL @OUTL          ;N, LIST
     00B6 007C'
0173                    *
0174 00B8 C1D9  INPP1   MOV *R9,R7        ;GET IOB  (WSBC)
0175 00BA C129          MOV @FFLG(9),R4 ;GET INPUT COUNTER
     00BC 0074'
0176 00BE 1602            JNE INPP2       ;OK
0177 00C0 0204          LI R4,>80         ;SET MAX
     00C2 0080
0178                    *
0179 00C4 06A0  INPP2   BL @GETCR         ;GET CHARACTER
     00C6 0000
0180 00C8 06A0          BL @PUTB          ;STORE
     00CA 0000
0181 00CC 1006            JMP INPP3       ;CR
0182 00CE 100B            JMP INPP4       ;CONTROL CHARACTER
0183 00D0 06A0          BL @TYPO          ;ECHO
     00D2 0000
0184 00D4 0604          DEC R4            ;MORE CHARACTERS?
0185 00D6 16F6            JNE INPP2       ;Y
0186 00D8 1003          JMP INPPS         ;N
0187                    *
0188 00DA C029  INPP3   MOV @IFLG(9),R0 ;LOOKING FOR MAX?
     00DC 0070'
0189 00DE 16F2            JNE INPP2       ;Y
0190                    *
0191 00E0 75D7  INPPS   SB *R7,*R7        ;TERMINATE
0192 00E2 C1D9          MOV *R9,R7        ;RESET BYTE COUNTERS  (WSBC)
0193 00E4 045E          B *R14            ;RETURN
0194                    *
0195 00E6 C069  INPP4   MOV @HFLG(9),R1 ;HELP WANTED?
     00E8 0066'
0196 00EA 13EC            JEQ INPP2       ;N, CONTINUE
0197 00EC 06C0          SWPB R0           ;Y
0198                    *
0199 00EE CA40  INPP6   MOV R0,@HFLG(9) ;SAVE CONTROL CHARACTER
     00F0 00E8'
0200 00F2 04E9          CLR @FFLG(9)      ;CLEAR FORMATTING
     00F4 008C'
0201 00F6 0460          B @GOSB1          ;DO SYSTEM GOSUB
     00F8 0000
0202                    *
0203 00FA   3F  INM1    TEXT '? '
     00FB   20
0204 00FC   00          BYTE 0
0205 00FD   3F  INM2    TEXT '?? '
     00FE   3F
     00FF   20
0206 0100   00          BYTE 0
0207 0101   3A  INM3    TEXT ': '
     0102   20
0208 0103   00          BYTE 0
0209 0104               EVEN
0210                    END
NO ERRORS,      NO WARNINGS
```

LABEL		VALUE	DEFN	REFERENCES						
BMVE	R	00A4'	0022	0146	0159					
CVDIFZ	R	0044'	0018	0101						
EVARZ	R	0036'	0019	0096						
EVFIX			0028							
EVSDZ	R	002E'	0020	0093						
FFLG	R	00F4'	0032	0133	0175	0200				
GETCR	R	00C6'	0025	0179						
GOSB1	R	00F8'	0021	0201						
HFLG	R	00F0'	0033	0069	0075	0122	0195	0199		
IFLG	R	00DC'	0034	0070	0129	0188				
INM1		00FA'	0203	0098						
INM2		00FD'	0205	0121						
INM3		0101'	0207	0154						
INP0		000E'	0077	0110	0117	0134	0138	0160	0161	
INP1		0010'	0079	0071	0086					
INP3		003E'	0100	0123						
INP4		0058'	0114	0102						
INP5		0060'	0121	0106	0115					
INPD		0090'	0153	0095						
INPE		00AA'	0165	0083	0084	0085				
INPEND	R	00AC'	0016	0165						
INPFL		006E'	0129	0090						
INPFM		0072'	0133	0088						
INPP		00AE'	0169	0100	0155					
INPP1		00B8'	0174	0171						
INPP2		00C4'	0179	0176	0185	0189	0196			
INPP3		00DA'	0188	0181						
INPP4		00E6'	0195	0182						
INPP5		00E0'	0191	0186						
INPP6		00EE'	0199	0125						
INPQ		0078'	0138	0094						
INPQM		000A'	0075	0089						
INPSC		0012'	0081	0087						
INPTB		0018'	0083	0083	0084	0085	0086	0087	0088	0089	0090
INPY	D	0000'	0069	0067						
JMPR0A	R	0016'	0023	0082						
OUTL		007C'	0143	0172						
OUTL1	R	008E'	0017	0149						
PUTB	R	00CA'	0024	0180						
R0		0000		0105	0105	0114	0114	0124	0188	0197	0199
R1		0001		0103	0108	0122	0195				
R11		000B		0138	0143	0148	0169				
R13		000D		0143	0148						
R14		000E		0169	0193						
R15		000F		0079	0081	0170	0170				
R2		0002		0097	0098	0121	0153	0154	0156		
R4		0004		0175	0177	0184					
R5		0005		0145	0158						
R6		0006		0097	0107	0108	0109	0116	0153	0157	
R7		0007		0144	0156	0157	0174	0191	0191	0192	
R8		0008		0077	0092						
R9		0009		0144	0174	0192					
STORE			0027								
TYP0	R	00D2'	0026	0183							

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.CVDB
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.CVDB
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.CVDB
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

0001

```
      IDT 'CVDB'
0002              *
0003              * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004              *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005              *           30 MAY 79          ;COPY AND UPDATE TO REL5
0006              * ROUTINE LIST:
0007              *
0008              *       CVDIZ   ;CONVERT DECIMAL TO BINARY INTEGER
0009              *       CVDIFZ  ;CONVERT DECIMAL TO FLOATING POINT
0010              *       CVDB20  ;GET NEXT DIGIT
0011              *
0012              * COPY: (NONE)
0013              *
0014              * MACRO: (NONE)
0015              *
0016              * EXTERNAL ROUTINES:
0017              *
0018                      REF CVGCN1        ;GET FLOATING POINT CONSTANT
0019              *
0020                      DXOP LOADF,0      ;LOAD FPAC
0021                      DXOP STORE,1      ;STORE FPAC
0022                      DXOP FADD,2       ;ADD TO FPAC
0023                      DXOP FSUB,3       ;SUBTRACT FROM FPAC
0024                      DXOP FMUL,4       ;MULTIPLY FPAC
0025                      DXOP FDIV,5       ;DIVIDE FPAC
0026                      DXOP SCALE,6      ;SCALE FPAC
0027                      DXOP NORMAL,7     ;NORMALIZE FPAC
0028                      DXOP CLEAR,8      ;CLEAR FPAC
0029                      DXOP NEGATE,9     ;NEGATE FPAC
0030                      DXOP FLOATF,10    ;FLOAT FPAC
0031    2F80  ERROR   EQU >2F80             ;XOP XX,14  (ERROR CALL)
0032    2FA0  ERROR2  EQU ERROR+>20
0033              *
0034              * EXTERNAL DATA:
0035              *
0036                      REF TEMP          ;3 WORD TEMPORARY STORAGE
0037                      REF CVCH          ;CONVERSION HOLDING AREA
0038                      REF WPR2          ;SECONDARY WORKSPACE
0039                      REF CVC10         ;FLOATING POINT CONSTANTS
0040              *
0041              * MODULE EQUATES: (NONE)
0042              *
0043              *
0044              * MODULE VARIABLES AND CONSTANTS:
0045              *
0046                    TITL "CONVERT DECIMAL TO BINARY"
```

```
0048            * ABSTRACT:
0049            *
0050            *       CVDIZ AND CVDIFZ CONVERT A STRING OF
0051            *       ASCII DECIMAL DIGITS TO A BINARY NUMBER.
0052            *       CVDIZ IS CALLED IF ONLY AN INTEGER WILL
0053            *       BE ACCEPTED.  CVDIFZ IS CALLED IF A
0054            *       FLOATING POINT NUMBER CAN BE ALLOWED.
0055            *       CVDI IS FIRST EXECUTED TO GET AN
0056            *       INTEGER.  IF IT FAILS, THEN A RETURN
0057            *       IS MADE OR CVDIFZ IS EXECUTED DEPENDING
0058            *       ON WHICH ROUTINE WAS INITIALLY CALLED.
0059            *
0060            *
0061            * CALLING SEQUENCE:
0062            *
0063            *       BLWP @CVDIZ
0064            *         OR
0065            *       BLWP @CVDIFZ
0066            *
0067            *      IN - R7 =PTR
0068            *
0069            *      OUT -     R0 = DELIMITER
0070            *                R1 = 16-BIT 2'S COMPLEMENT INTEGER
0071            *                FPAC = FLOATING POINT NUMBER
0072            *                R7 = NEW PTR
0073            *
0074            *      NORMAL EXIT - RTWP
0075            *      ERROR EXIT
0076            *
0077            * EXCEPTIONS AND CONDITIONS:
0078            *
0079            *
0080            *
0081            * EXTERNAL ROUTINE LIST:
0082            *
0083            *      (SEE EXTERNAL ROUTINES)
0084            *
0085            * LOCAL DATA:
0086            *
0087            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0089                * ENTRY POINT:
0090                *
0091                        DEF CVDIZ
0092                *CONVERT DECIMAL TO INTEGER
0093                *       BLWP @CVDIZ
0094                *         16-BIT OVERFLOW - FP # IN FPAC
0095                *         NO NUMBER
0096                *         (HEX ON CVDIF)
0097                *       NUMBER
0098                *
0099                *  IN   R7 = PTR
0100                *  OUT  R0 = DELIMITER
0101                *       R1 = 16-BIT 2'S COMPLEMENT INTEGER
0102                *       R7 = NEW PTR
0103                *
0104 0000 0000     CVDIZ    DATA WPR2,CVDI
     0002 0008'
0105 0004 0000'    CVDIFZ   DATA WPR2,CVDIF
     0006 000C'
0106                *
0107 0008 04C8     CVDI     CLR R8           GET INTEGER ONLY
0108 000A 1001              JMP $+4
```

```
0110                    *
0111                    * ABSTRACT:
0112                    *
0113                    * ENTRY POINT
0114                    *
0115                            DEF CVDIFZ
0116 000C 0708  CVDIF   SETO R8         ALLOW EXPONENT
0117                    *
0118 000E C1ED          MOV @14(13),R7  GET PTR
     0010 000E
0119 0012 04C2          CLR R2          CLEAR RESULT
0120 0014 04C4          CLR R4          CLEAR SIGN FLAG
0121 0016 06A0          BL @CVDB25      LOOK FOR SIGN
     0018 0156'
0122 001A 103A            JMP CVDI4     NO NUMBER, RETURN
0123 001C 0704            SETO R4       NEGATIVE
0124 001E C187          MOV R7,R6
0125                    *
0126 0020 D036  CVDIH1  MOVB *R6+,R0
0127 0022 0980          SRL R0,8        ;POSITION
0128 0024 0220          AI R0,->30
     0026 FFD0
0129 0028 1114            JLT CVDIH4    ;NOT HEX
0130 002A 0280          CI R0,>09
     002C 0009
0131 002E 1208            JLE CVDIH2    ;HEX 0-9
0132 0030 0220          AI R0,->07
     0032 FFF9
0133 0034 0280          CI R0,>0A
     0036 000A
0134 0038 110C            JLT CVDIH4    ;NOT HEX
0135 003A 0280          CI R0,>0F
     003C 000F
0136 003E 1B03             JH CVDIH3    ;NOT HEX, "H MAYBE?
0137                    *
0138 0040 0A42  CVDIH2  SLA R2,4         ;ADD NEW HEX DIGIT
0139 0042 A080          A R0,R2
0140 0044 10ED          JMP CVDIH1
0141                    *
0142 0046 0220  CVDIH3  AI R0,->11       ;"H?
     0048 FFEF
0143 004A 1603            JNE CVDIH4    ;N
0144 004C D036          MOVB *R6+,R0    ;Y, GET DELIMITER
0145 004E C1C6          MOV R6,R7       ;UPDATE R7
0146 0050 1017          JMP CVDI3A
0147                    *
0148 0052 04C2  CVDIH4  CLR R2          ;NOT HEX, CLEAR R2
0149                    *
0150 0054 06A0  CVDI1   BL @CVDB20      GET DIGIT
     0056 0136'
0151 0058 100A            JMP CVDI2     N
0152 005A C042          MOV R2,R1       SET FOR MULTIPLICATION
0153 005C 3860          MPY @C000A,R1   R1,R2=R1*10
     005E 01AA'
0154 0060 C041          MOV R1,R1       OVERFLOW?
0155 0062 1618            JNE CVDB      Y
0156 0064 C082          MOV R2,R2       OVERFLOW?
0157 0066 1119            JLT CVDB      ;Y
0158 0068 A080          A R0,R2         ADD NEW DIGIT, OVERFLOW?
0159 006A 1117            JLT CVDB      Y, OVERFLOW
```

```
0160 006C 10F3           JMP CVDI1        LOOP
0161                     *
0162 006E 0280  CVDI2    CI R0,>2E00      ".?
     0070 2E00
0163 0072 1313             JEQ CVDB       Y
0164 0074 0280           CI R0,>4500      "E?
     0076 4500
0165 0078 130E             JEQ CVDI5      Y, SEE IF LEGAL
0166                     *
0167 007A C208  CVDI3    MOV R8,R8        ;SKIP HEX RETURN?
0168 007C 1301             JEQ CVDI3A     ;Y
0169 007E 05CE           INCT R14         ;N
0170                     *
0171 0080 C104  CVDI3A   MOV R4,R4        N, NEGATIVE?
0172 0082 1301             JEQ $+4        N
0173 0084 0502             NEG R2         Y, NEGATE
0174 0086 CB42           MOV R2,@2(13)    RETURN NUMBER IN R1
     0088 0002
0175 008A CB47           MOV R7,@14(13)   RETURN PTR IN R7
     008C 000E
0176 008E 05CE           INCT R14         RETURN 4(14)
0177 0090 05CE  CVDI4    INCT R14         RETURN 2(14) - NO NUMBER
0178 0092 C740           MOV R0,*R13      RETURN DELIMITER IN R0
0179 0094 0380           RTWP             RETURN 0(14) - OVERFLOW
0180                     *
0181 0096 C208  CVDI5    MOV R8,R8        ;EXPONENT ALLOWED?
0182 0098 13F0             JEQ CVDI3      ;N
0183                     *
0184                     *FALL THRU TO CVDB
```

```
0186                    *CONVERT DECIMAL TO FP
0187                    *       BLWP @CVDBZ
0188                    *         NUMBER
0189                    *       NO NUMBER
0190                    *
0191                    *  IN   R7 = STRING ADDRESS
0192                    *  OUT  R0 = DELIMITER
0193                    *       R7 = NEW PTR
0194                    *       FPAC = NUMBER
0195                    *
0196 009A 2E00  CVDB    CLEAR 0           CLEAR FPAC
0197 009C C1ED          MOV @14(13),R7    GET STRING ADR
     009E 000E
0198 00A0 C26D          MOV @18(13),R9    GET POINTER REGISTER
     00A2 0012
0199 00A4 0702          SETO R2           SET DECIMAL FLAG
0200 00A6 04C6          CLR R6            DECIMAL ADJUSTMENT=0
0201                    *
0202 00A8 04CC          CLR R12           CLEAR SIGN FLAG
0203 00AA 06A0          BL @CVDB25        PROCESS SIGN
     00AC 0156'
0204 00AE 10F0            JMP CVDI4       NO NUMBER
0205 00B0 070C            SETO R12        NEG, SET TO -1
0206                    *
0207 00B2 06A0  CVDB1   BL @CVDB20        GET CHARACTER
     00B4 0136'
0208 00B6 100E            JMP CVDB4       NOT NUMBER
0209 00B8 C082          MOV R2,R2         AFTER DECIMAL?
0210 00BA 1101            JLT $+4         N
0211 00BC 0586          INC R6            Y, COUNT
0212                    *
0213 00BE 2D20          FMUL @CVC10       ;OK, FPAC=FPAC*10
     00C0 0000
0214 00C2 0A40          SLA R0,4          00X0
0215 00C4 13F6            JEQ CVDB1       IGNOR IF ZERO
0216 00C6 0220          AI R0,>4100       41X0 (FP # NOW)
     00C8 4100
0217 00CA C800          MOV R0,@CVCH      SAVE
     00CC 0000
0218 00CE 2CA0          FADD @CVCH        ADD NEW DIGIT
     00D0 00CC'
0219 00D2 10EF          JMP CVDB1
0220                    *
0221 00D4 0280  CVDB4   CI R0,>2E00       ".?
     00D6 2E00
0222 00D8 1604            JNE CVDB5       N
0223 00DA C082          MOV R2,R2         Y, FIRST PERIOD?
0224 00DC 1302            JEQ CVDB5       N, END CONVERSION
0225 00DE 04C2          CLR R2            Y, SET FLAG
0226 00E0 10E8          JMP CVDB1
0227                    *
0228 00E2 0280  CVDB5   CI R0,>4500       "E?
     00E4 4500
0229 00E6 1612            JNE CVDB10      N
0230 00E8 0205          LI R5,>0501       Y, (NEG R1)
     00EA 0501
0231 00EC 06A0          BL @CVDB25        CHECK FOR SIGN
     00EE 0156'
0232 00F0 10CF            JMP CVDI4       PROBLEM, RETURN NO NUMBER
0233 00F2 0AC5            SLA R5,12       NEG, LOAD NOOP (1000)
```

```
ADR

 REGISTER

 FLAG
USTMENT=0

FLAG                    <- CLR  R4
N


 -1

ER


AL?


PAC*10


RO
NOW)


IT




RIOD?
ERSION




GN

URN NO NUMBER
OOP (1000)
```

```
0234 00F4 04C1            CLR R1
0235                 *
0236 00F6 06A0  CVDB6     BL @CVDB20        GET DIGIT
     00F8 0136'
0237 00FA 1006              JMP CVDB7       NO DIGIT, DONE
0238 00FC C0C1            MOV R1,R3
0239 00FE 38E0            MPY @C000A,R3     X 10
     0100 01AA'
0240 0102 A100            A R0,R4           ADD NEW DIGIT
0241 0104 C044            MOV R4,R1         RESTORE R1
0242 0106 10F7            JMP CVDB6         LOOP AGAIN
0243                 *
0244 0108 0485  CVDB7     X R5              DO EXPONENT CHANGE
0245 010A A181            A R1,R6           ADD EXPONENT ADJUSTOR
0246                 *
0247 010C C740  CVDB10    MOV R0,*R13       DONE, RETURN DELIMITER
0248 010E CB47            MOV R7,@14(13)    RETURN NEW PTR
     0110 000E
0249                 *
0250 0112 C206  CVDB11    MOV R6,R8         CHECK ADJUSTMENT
0251 0114 1106              JLT CVDB12      DO ADJUSTMENT NECESSARY
0252 0116 130B              JEQ CVDB13      "       "       "
0253 0118 06A0            BL @CVGCN1        GET CONSTANT
     011A 0000
0254 011C 6188              S R8,R6
0255 011E 2D50            FDIV *R0          DIVIDE BY 10↑R6
0256 0120 10F8            JMP CVDB11
0257                 *
0258 0122 0508  CVDB12    NEG R8
0259 0124 06A0            BL @CVGCN1        GET CONSTANT
     0126 011A'
0260 0128 A188              A R8,R6
0261 012A 2D10            FMUL *R0          MULTIPLY BY 10↑R6
0262 012C 10F2            JMP CVDB11
0263                 *
0264 012E C30C  CVDB13    MOV R12,R12       CHECK FOR NEGATIVE #
0265 0130 1301              JEQ CVDB14      POSITIVE
0266 0132 2E40            NEGATE 0
0267 0134 0380  CVDB14    RTWP
0268                 *
```

```
0270            * ABSTRACT:
0271            *
0272            *       CVDB20 GETS THE NEXT DIGIT IF THERE
0273            *       IS ONE.  LEADING BLANKS ARE SKIPPED.
0274            *       IF THE NEXT CHARACTER IS A NON-DIGIT
0275            *       THEN RETURN IS MADE TO THE FIRST
0276            *       INSTRUCTION FOLLOWING THE CALL.  IF
0277            *       THE NEXT CHARACTER IS A DIGIT, A
0278            *       RETURN IS TAKEN TO THE SECOND
0279            *       INSTRUCTION FOLLOWING THE CALL.
0280            *
0281            * CALLING SEQUENCE:
0282            *
0283            *       BL @CVDB20
0284            *
0285            *       IN -    R7 = POINTER
0286            *       OUT -   R0 = CHARACTER
0287            *
0288            *       NORMAL EXIT - RETURN + 1 ON DIGIT
0289            *                     RETURN ON NON-DIGIT
0290            *
0291            * EXCEPTIONS AND CONDITIONS:
0292            *
0293            *
0294            *
0295            * EXTERNAL ROUTINE LIST:
0296            *
0297            *       (SEE EXTERNAL ROUTINES)
0298            *
0299            * LOCAL DATA:
0300            *
0301            *       (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0303                    *
0304                    *       ENTRY POINT:
0305                    *
0306                            DEF CVDB20
0307 0136 04C0  CVDB20  CLR R0          GET DIGIT
0308 0138 D037          MOVB *R7+,R0    GET NEXT CHARACTER
0309 013A 130C            JEQ CVDB22    EOL
0310 013C 0280          CI R0,>2000     BLANK?
     013E 2000
0311 0140 13FA            JEQ CVDB20    Y
0312 0142 0280          CI R0,>3000     <"0?
     0144 3000
0313 0146 1A06            JL CVDB22     Y
0314 0148 0280          CI R0,>3900     >"9?
     014A 3900
0315 014C 1B03            JH CVDB22     Y
0316 014E 0A40          SLA R0,4
0317 0150 09C0          SRL R0,12       RIGHT JUSTIFY CHARACTER
0318 0152 05CB          INCT R11
0319 0154 045B  CVDB22  RT              RETURN
```

```
0321                    *PROCESS SIGN
0322                    *       BL @CVDB25
0323                    *         NO NUMBER
0324                    *         -NUMBER
0325                    *       NUMBER OR +NUMBER
0326                    *
0327 0156 9837  CVDB25  CB *R7+,@B20    SPACE?
     0158 01AC'
0328 015A 13FD            JEQ CVDB25    ?
0329 015C 0607          DEC R7          BACKUP
0330 015E C047          MOV R7,R1       MARK
0331 0160 C28B          MOV R11,R10
0332 0162 06A0          BL @CVDB20      GET NUMBER
     0164 0136'
0333 0166 1004            JMP CVDB28    NO NUMBER
0334 0168 05CA  CVDB26  INCT R10        NUMBER, RETURN 4(10)
0335 016A 05CA          INCT R10        -NUMBER, RETURN 2(10)
0336 016C C1C1  CVDB27  MOV R1,R7       RESTORE R7
0337 016E 045A          B *R10          RETURN
0338                    *
0339 0170 0280  CVDB28  CI R0,>2B00      "+?
     0172 2B00
0340 0174 1307            JEQ CVDB29    Y
0341 0176 0280          CI R0,>2D00     N, "-?
     0178 2D00
0342 017A 1307            JEQ CVDB30    Y
0343 017C 0280          CI R0,>2E00     ".?
     017E 2E00
0344 0180 16F5            JNE CVDB27    N, NO NUMBER
0345 0182 0607          DEC R7          Y, BACKUP OVER PERIOD
0346                    *
0347 0184 06A0  CVDB29  BL @CVDB31      PROCESS POSITIVE NUMBER
     0186 0190'
0348 0188 10EF          JMP CVDB26      OK
0349                    *
0350 018A 06A0  CVDB30  BL @CVDB31      PROCESS NEGATIVE NUMBER
     018C 0190'
0351 018E 10ED          JMP CVDB26+2    OK
0352                    *
0353 0190 C0CB  CVDB31  MOV R11,R3      SAVE RETURN
0354 0192 C047          MOV R7,R1       MARK
0355 0194 06A0          BL @CVDB20      DIGIT?
     0196 0136'
0356 0198 1001            JMP CVDB32    N, LOOK FOR PERIOD
0357 019A 0453          B *R3           Y, NUMBER
0358                    *
0359 019C 0280  CVDB32  CI R0,>2E00      ".?
     019E 2E00
0360 01A0 16E5            JNE CVDB27    N, NO NUMBER
0361 01A2 06A0          BL @CVDB20      Y, LOOK FOR DIGIT
     01A4 0136'
0362 01A6 10E2            JMP CVDB27    NO NUMBER
0363 01A8 0453          B *R3           NUMBER OK
0364                    *
0365 01AA 000A  C000A   DATA >000A
0366 01AC   20  B20     BYTE >20
0367 01AE               EVEN
0368                    END
NO ERRORS,      NO WARNINGS
```

```
$          01AE'          0108  0172  0210
B20        01AC'    0366  0327
C000A      01AA'    0365  0153  0239
CLEAR              0028
CVC10   R  00C0'    0039  0213
CVCH    R  00D0'    0037  0217  0218
CVDB       009A'    0196  0155  0157  0159  0163
CVDB1      00B2'    0207  0215  0219  0226
CVDB10     010C'    0247  0229
CVDB11     0112'    0250  0256  0262
CVDB12     0122'    0258  0251
CVDB13     012E'    0264  0252
CVDB14     0134'    0267  0265
CVDB20  D  0136'    0307  0150  0207  0236  0306  0311  0332  0355  0361
CVDB22     0154'    0319  0309  0313  0315
CVDB25     0156'    0327  0121  0203  0231  0328
CVDB26     0168'    0334  0348  0351
CVDB27     016C'    0336  0344  0360  0362
CVDB28     0170'    0339  0333
CVDB29     0184'    0347  0340
CVDB30     018A'    0350  0342
CVDB31     0190'    0353  0347  0350
CVDB32     019C'    0359  0356
CVDB4      00D4'    0221  0208
CVDB5      00E2'    0228  0222  0224
CVDB6      00F6'    0236  0242
CVDB7      0108'    0244  0237
CVDI       0008'    0107  0104
CVDI1      0054'    0150  0160
CVDI2      006E'    0162  0151
CVDI3      007A'    0167  0182
CVDI3A     0080'    0171  0146  0168
CVDI4      0090'    0177  0122  0204  0232
CVDI5      0096'    0181  0165
CVDIF      000C'    0116  0105
CVDIFZ  D  0004'    0105  0115
CVDIH1     0020'    0126  0140
CVDIH2     0040'    0138  0131
CVDIH3     0046'    0142  0136
CVDIH4     0052'    0148  0129  0134  0143
CVDIZ   D  0000'    0104  0091
CVGCN1  R  0126'    0018  0253  0259
ERROR      2F80     0031  0032
ERROR2     2FA0     0032
FADD               0022
FDIV               0025
FLOATF             0030
FMUL               0024
FSUB               0023
LOADF              0020
NEGATE             0029
NORMAL             0027
R0         0000          0126  0127  0128  0130  0132  0133  0135  0139  0142
                         0144  0158  0162  0164  0178  0214  0216  0217  0221
                         0228  0240  0247  0255  0261  0307  0308  0310  0312
                         0314  0316  0317  0339  0341  0343  0359
R1         0001          0152  0153  0154  0154  0234  0238  0241  0245  0330
                         0336  0354
R10        000A          0331  0334  0335  0337
R11        000B          0318  0331  0353
```

LABEL		VALUE	DEFN	REFERENCES								
R12		000C		0202	0205	0264	0264					
R13		000D		0178	0247							
R14		000E		0169	0176	0177						
R2		0002		0119	0138	0139	0148	0152	0156	0156	0158	0173
				0174	0199	0209	0209	0223	0223	0225		
R3		0003		0238	0239	0353	0357	0363				
R4		0004		0120	0123	0171	0171	0240	0241			
R5		0005		0230	0233	0244						
R6		0006		0124	0126	0144	0145	0200	0211	0245	0250	0254
				0260								
R7		0007		0118	0124	0145	0175	0197	0248	0308	0327	0329
				0330	0336	0345	0354					
R8		0008		0107	0116	0167	0167	0181	0181	0250	0254	0258
				0260								
R9		0009		0198								
SCALE			0026									
STORE			0021									
TEMP	R		0036									
WPR2	R	0004'	0038	0104	0105							

*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL5.BASIC.SOURCE.MWDF
OBJECT ACCESS NAME=      BD1.DBASIC.REL5.BASIC.OBJECT.MWDF
LISTING ACCESS NAME=     BD1.DBASIC.REL5.BASIC.LIST.MWDF
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT 'MWDF'
0002              *              2 MAY 79
0003              * REVISION:  22 MAY 79
0004              *
0005              * ROUTINE LIST:
0006              *
0007              *         MWDY              ;MWD COMMAND
0008              *         MWDF              ;MWD FUNCTION
0009              *
0010              * COPY: (NONE)
0011              *
0012              * MACRO: (NONE)
0013              *
0014              * EXTERNAL ROUTINES:
0015              *
0016                      REF EVERZ           ;EVALUATE EXPRESSION
0017                      REF PFIX            ;FIX 16 BIT NUMBER
0018                      REF GPRM2           ;GET PARAMETERS
0019                      REF NLIN            ;EXIT ENTRY TO MULTIPLEXOR
0020                      REF EVSFR           ;EXIT ENTRY TO EVALUATOR
0021                      DXOP STORE,1        ;STORE FPAC
0022                      DXOP CLEAR,8        ;CLEAR FPAC
0023                      DXOP EVFIX,11       ;EVALUATE AND FIX
0024                      DXOP OUTFP,12       ;OUT FLOATING POINT #
0025     2F80   ERROR    EQU >2F80                    ;XOP XX,14  (ERROR CALL)
0026              *
0027              * EXTERNAL DATA:
0028              *
0029                      REF FPAC,FPAC2      ;FLOATING POINT ACCUMULATOR
0030                      REF B4A             ;>4A
0031              *
0032              * MODULE EQUATES:  (NONE)
0033              *
0034              * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0037            * ABSTRACT:
0038            *
0039            *       THE MWD COMMAND ALLOWS PBASIC TO ALTER
0040            *       A WORD OF MEMORY WHILE THE MEM FUNCTION
0041            *       ALLOWS PBASIC TO READ A WORD OF
0042            *       MEMORY.
0043            *
0044            * CALLING SEQUENCE:
0045            *
0046            *       B @MWDY
0047            *
0048            *       EXIT TO NLIN
0049            *
0050            *
0051            *       B @MWDF
0052            *
0053            *       IN  (R2) = ADDRESS
0054            *       OUT (R2) = RESULT
0055            *       EXIT TO EVSFR
0056            *
0057            * EXCEPTIONS AND CONDITIONS:
0058            *
0059            *       EVALUATION ERRORS
0060            *       SYNTAX ERROR
0061            *
0062            * EXTERNAL ROUTINE LIST:
0063            *
0064            *      (SEE EXTERNAL ROUTINES)
0065            *
0066            * LOCAL DATA:
0067            *
0068            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0070                    * ENTRY POINT:
0071                    *
0072                            DEF MWDY
0073                    *
0074 0000 9838  MWDY    CB *R8+,@B4A    ;LEFT BRACKET?
     0002 0000
0075 0004 1609            JNE ERR1      ;N, ERROR
0076 0006 0420          BLWP @EVERZ     ;GET ADDRESS
     0008 0000
0077 000A 06A0          BL @PFIX        ;GET INTEGER
     000C 0000
0078 000E 06A0          BL @GPRM2       ;CHECK PARAMETERS
     0010 0000
0079 0012 C443          MOV  R3,*R1     ;STORE BYTE IN MEMORY
0080 0014 0460          B @NLIN
     0016 0000
0081                    *
0082 0018 2F81  ERR1    DATA ERROR+1
0083                    *
0084                    *READ MEMORY
0085                    *
0086                    * ENTRY POINT:
0087                    *
0088                            DEF MWDF
0089                    *
0090 001A 06A0  MWDF    BL @PFIX        ;GET PARAMETER
     001C 000C'
0091 001E C051          MOV  *R1,R1     ;GET BYTE
0092 0020 2E00          CLEAR 0         ;CLEAR FPAC
0093 0022 C801          MOV R1,@FPAC2   ;SAVE
     0024 0000
0094 0026 0202          LI R2,FPAC      ;GET ADDRESS
     0028 0000
0095 002A 0460          B @EVSFR
     002C 0000
0096                    END
NO ERRORS,      NO WARNINGS
```

```
LABEL     VALUE   DEFN  REFERENCES

B4A     R 0002'   0030  0074
CLEAR             0022
ERR1      0018'   0082  0075
ERROR     2F80    0025  0082
EVERZ   R 0008'   0016  0076
EVFIX             0023
EVSFR   R 002C'   0020  0095
FPAC    R 0028'   0029  0094
FPAC2   R 0024'   0029  0093
GPRM2   R 0010'   0018  0078
MWDF    D 001A'   0090  0088
MWDY    D 0000'   0074  0072
NLIN    R 0016'   0019  0080
OUTFP             0024
PFIX    R 001C'   0017  0077  0090
R1        0001          0079  0091  0091  0093
R2        0002          0094
R3        0003          0079
R8        0008          0074
STORE             0021
*
```

```
COMMAND LIST

TASK EXTENDED
NOTGLOBAL
INCLUDE DBASIC.REL2.EXTENDED.OBJECT.INTERFAC
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.CASSETTE
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.PROGRAM
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.FORMAT
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.ERRORLST
INCLUDE DBASIC.REL2.EXTENDED.OBJECT.ATNFE
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.MINUS1
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.MINUS1
INCLUDE DBASIC.REL0.EXTENDED.OBJECT.MINUS1
END
```

LINK MAP

CONTROL FILE = BD1.DBASIC.REL2.EXTENDED.LINK.CNTL

LINKED OUTPUT FILE = BD1.DBASIC.REL2.EXTENDED.LINK.OBJECT

LIST FILE = BD1.DBASIC.REL2.EXTENDED.LINK.LIST

OUTPUT FORMAT = ASCII

PHASE 0, EXTENDED ORIGIN = 0000 LENGTH = 110C

MODULE	NO	ORIGIN	LENGTH	TYPE	DATE	TIME	CREATOR
INTERFAC	1	0000	0026	INCLUDE	08/01/79	10:43:05	SDSMAC
CASSETTE	2	0026	0212	INCLUDE	10/03/78	15:19:51	SDSMAC
PROGRAM	3	0238	0214	INCLUDE	09/26/78	13:54:57	SDSMAC
FORMAT	4	044C	013E	INCLUDE	08/31/78	14:42:44	SDSMAC
ERRORLST	5	058A	049D	INCLUDE	09/18/78	17:20:35	SDSMAC
ATNFE	6	0A28	00E4	INCLUDE	05/30/79	13:28:48	SDSMAC
MINUS1	7	0B0C	0200	INCLUDE	09/18/78	18:04:43	SDSMAC
MINUS1	8	0D0C	0200	INCLUDE	09/18/78	18:04:43	SDSMAC
DUPLICATE IDT NAME *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*							
MINUS1	9	0F0C	0200	INCLUDE	09/18/78	18:04:43	SDSMAC
DUPLICATE IDT NAME *W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*W*							

D E F I N I T I O N S

NAME	VALUE	NO	NAME	VALUE	NO	NAME	VALUE	NO	NAME	VALUE	NO
ATNFE	0A28	6	BUS	000A*	1	CLKADR	FE56*	1	CSCL00	020A	2
CSOP00	0026	2	CSRD00	0064	2	CSWT00	0164	2	CVBF	044C	4
CVBFR	0090*	1	DS2	FFBC*	1	DS3	FFC2*	1	ERRLS1	059A	5
ERRLS2	05C2	5	EVSFR	0094*	1	FPAC	FF9C*	1	FUNFX	0098*	1
GETC	0080*	1	IDRUN	A5A5*	1	NVD	0014*	1	PLYX	00A0*	1
PLYXX	009C*	1	PRDY	0084*	1	PROGM	02BD	3	SLT	000E*	1
TEMP	FF94*	1	TYP0	008C*	1	TYPE	0088*	1	VDT	0012*	1
VNT	0010*	1									

2 WARNINGS

**** LINKING COMPLETED
*

```
SOURCE ACCESS NAME=      BD1.DBASIC.REL2.EXTENDED.SOURCE.INTERFAC
OBJECT ACCESS NAME=      BD1.DBASIC.REL2.EXTENDED.OBJECT.INTERFAC
LISTING ACCESS NAME=     BD1.DBASIC.REL2.EXTENDED.LIST.INTERFAC
ERROR ACCESS NAME=
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'INTERFAC'
0002                *   1 MAY 1979 HJC
0003                    DEF  GETC,PRDY,TYPE,TYP0
0004                    DEF  BUS,NVD,SLT,VNT,VDT
0005                    DEF  IDRUN,CLKADR,CVBFR
0006                    DEF  EVSFR,FUNFX,PLYXX,PLYX        ;EP DEFS  1 MAY 79
0007                    DEF  DS2,DS3,TEMP,FPAC             ;VAR DEFS 1 MAY 79
0008                    REF  ATNFE                         ;EP REFS  1 MAY 79
0009                    REF  CSOP00,CSRD00,CSWT00,CSCL00
0010                    REF  PROGM,CVBF,ERRLS1,ERRLS2
0011                *
0012                ************************WARNING**************************
0013                *     THE FOLLOWING LOCATIONS SHOULD BE
0014                *     CHANGED ONLY WITH CORRESPONDING CHANGES
0015                *     IN THE DEVELOPMENT BASIC MODULES
0016                *********************************************************
0017                *
0018        FE56    CLKADR EQU  >FE56
0019        FFBC    CSWP   EQU >FFBC
0020        FFBC    DS2    EQU CSWP
0021        FFC2    DS3    EQU CSWP+6
0022        FF94    TEMP   EQU >FF94
0023        FF9C    FPAC   EQU >FF9C
0024                *
0025  0000  A55A    ID302  DATA >A55A          ;302 EPROM ID
0026  0002  FFBC    CSOPZ  DATA CSWP,CSOP00    ;CASETTE ENTRIES
      0004  0000
0027  0006  FFBC    CSRDZ  DATA CSWP,CSRD00
      0008  0000
0028  000A  FFBC    CSWTZ  DATA CSWP,CSWT00
      000C  0000
0029  000E  FFBC    CSCLZ  DATA CSWP,CSCL00
      0010  0000
0030  0012  0460            B @PROGM            ;PROM ENTRY
      0014  0000
0031  0016  0460            B @CVBF             ;PRINT FORMATTING ENTRY
      0018  0000
0032  001A  0460            B @ERRLS1           ;ERROR LISTING #1 ENTRY POINT
      001C  0000
0033  001E  0460            B @ERRLS2           ;ERROR LISTING #2 ENTRY POINT
      0020  0000
0034  0022  0460            B @ATNFE            ;  FOR ATNF 1 MAY 1979
      0024  0000
0035                *
0036        0080    GETC   EQU >0080           ;GETC
0037        0084    PRDY   EQU >0084           ;PRDY
0038        0088    TYPE   EQU >0088           ;TYPE
0039        008C    TYP0   EQU >008C           ;TYP0
0040        0090    CVBFR  EQU >0090           ;ROUND ROUTINE
0041        0094    EVSFR  EQU >0094           ;
0042        0098    FUNFX  EQU >0098           ;  FOR   ATNF 1 MAY 1979
0043        009C    PLYXX  EQU >009C           ;
0044        00A0    PLYX   EQU >00A0           ;
0045        000A    BUS    EQU >000A           ;BEGINNING USER STORAGE
0046        000E    SLT    EQU >000E           ;STATEMENT LOCATION TABLE
0047        0010    VNT    EQU >0010           ;VARIABLE NAME TABLE
0048        0012    VDT    EQU >0012           ;VARIABLE DEFINITION TABLE
0049        0014    NVD    EQU >0014           ;NEXT VARIABLE DEFIN
0050        A5A5    IDRUN  EQU >A5A5
0051                *
```

```
 0052            **********WARNING**********
 0053            *      THE ABOVE LOCATIONS SHOULD BE
 0054            *      CHANGED ONLY WITH CORRESPONDING CHANGES
 0055            *      IN THE DEVELOPMENT BASIC MODULES
 0056            ************************************
 0057                    END
NO ERRORS,      NO WARNINGS
```

ATNFE	R	0024'	0008	0034
BUS	D	000A	0045	0004
CLKADR	D	FE56	0018	0005
CSCL00	R	0010'	0009	0029
CSCLZ		000E'	0029	
CSOP00	R	0004'	0009	0026
CSOPZ		0002'	0026	
CSRD00	R	0008'	0009	0027
CSRDZ		0006'	0027	
CSWP		FFBC	0019	0020 0021 0026 0027 0028 0029
CSWT00	R	000C'	0009	0028
CSWTZ		000A'	0028	
CVBF	R	0018'	0010	0031
CVBFR	D	0090	0040	0005
DS2	D	FFBC	0020	0007
DS3	D	FFC2	0021	0007
ERRLS1	R	001C'	0010	0032
ERRLS2	R	0020'	0010	0033
EVSFR	D	0094	0041	0006
FPAC	D	FF9C	0023	0007
FUNFX	D	0098	0042	0006
GETC	D	0080	0036	0003
ID302		0000'	0025	
IDRUN	D	A5A5	0050	0005
NVD	D	0014	0049	0004
PLYX	D	00A0	0044	0006
PLYXX	D	009C	0043	0006
PRDY	D	0084	0037	0003
PROGM	R	0014'	0010	0030
SLT	D	000E	0046	0004
TEMP	D	FF94	0022	0007
TYP0	D	008C	0039	0003
TYPE	D	0088	0038	0003
VDT	D	0012	0048	0004
VNT	D	0010	0047	0004

*

```
SOURCE ACCESS NAME=      DBASIC.REL0.EXTENDED.SOURCE.CASSETTE
OBJECT ACCESS NAME=      DBASIC.REL0.EXTENDED.OBJECT.CASSETTE
LISTING ACCESS NAME=     DBASIC.REL0.EXTENDED.LIST.CASSETTE
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0002                IDT  'CASSETTE'
0003          ********************************************************************
0004          *
0005          *      DEVELOPMENT BASIC CASSETTE DEVICE SERVICING ROUTINE
0006          *
0007          *           UPDATE 1.0:  AUGUST 7, 1978
0008          *
0009          ********************************************************************
0010          *
0011          *
0012          *   EXTERNAL REFERENCES AND DEFINITIONS
0013          *
0014                DEF  CSOP00,CSRD00,CSWT00,CSCL00
0015                REF  CSOPZ,CSRDZ,CSWTZ,CSCLZ
0016                REF  CLKADR
```

```
0018                *
0019                *  EQUATES
0020                *
0021      0100  CR9901 EQU  >100          9901 CRU ADDRESS
0022      1700  CRCASS EQU  >1700         CASSETTE DATA CRU ADDRESS
0023      0028  NUMZ   EQU  40            NUMBER OF ZEROES BEFORE SYNC
0024      0096  NUMFIL EQU  150           NUMBER OF FILLS BEFORE FIRST BLOCK
0025      0025  ZERHI  EQU  >25           OUTPUT WAIT COUNT FOR ZERO HIGH
0026      0011  ONEHI  EQU  >11             "     "    "    "  ONE  HIGH
0027      7FA5  HIZER  EQU  >7FA5          INPUT MAX. WAIT COUNT FOR HIGH CYCL
0028      7FC7  LOZER  EQU  >7FC7            "   MIN.  "     "    "    "    "
0029      5596  SYNCB  EQU  >5596          ASCII SYNC CHARACTER (READ)
0030      9600  SYNC   EQU  >9600          ASCII SYNC CHARACTER (WRITE)
0031      0004  EOT    EQU  >04            ASCII EOT CHARACTER
0032      0003  SCANCT EQU  3              SCAN COUNT FOR WAITB
0033      0032  CHKSM  EQU  50             CHECKSUM ERROR CODE
0034      0033  GARB   EQU  51             GARBAGE ERROR CODE
0035      0005  STWT   EQU  5              DETERMINES LENGTH OF WAITS FOR CASS
0036                *                          MOTION. (1 COUNT = .25 SECS.)
0037      0011  TRYCNT EQU  17             # OF ALTERNATING MATCHES AND
0038                *                          MICMATCHES EXPECTED
0039      0019  INVCNT EQU  25             # OF TRIES BEFORE INVERSION IS TRID
0040      0005  THRESH EQU  5              THRESHHOLD FOR MATCH OF TIMES
0041      0020  BADTHR EQU  32             THRESHOLD FOR A MISMATCH
```

```
0043                  ******************************************************
0044                  *                  OPEN PROCESSOR                    *
0045                  ******************************************************
0046        0000'     CSOP00 EQU  $
0047 0000 0300              LIMI 0                      CLEAR INTERUPT MASK
     0002 0000
0048 0004 04CF              CLR R15                     CLEAR INT MASK FOR RTWP
0049 0006 020C              LI   R12,CRCASS             CASSETTE CRU BASE ADDRESS
     0008 1700
0050 000A C28C              MOV  R12,R10                SAVE CASSETTE CRU BASE
0051 000C C01D              MOV  *R13,R0                GET CASSETTE DRIVE NUMBER
0052 000E 1301              JEQ  CSOP10                 EQ - CASSETTE DRIVE #1
0053 0010 05CC              INCT R12                    NE - CASSETTE DRIVE #2
0054        0012'     CSOP10 EQU  $
0055 0012 1D00              SBO  0                      START THE MOTOR
0056                  *
0057                  *  ZERO THE CLOCK
0058                  *
0059 0014 0203              LI   R3,CLKADR              GET CLOCK ADDRESS
     0016 0000
0060 0018 04F3              CLR  *R3+                   CLEAR HOURS
0061 001A 04F3              CLR  *R3+                   CLEAR MINUTES
0062 001C 04D3              CLR  *R3                    CLEAR SECONDS
0063                  *
0064                  *   CHECK FOR TYPE OF OPEN
0065                  *       *R14 =  0000      ON OPEN FOR READ
0066                  *       *R14 <> 0000      ON OPEN FOR WRITE
0067                  *
0068 001E C03E              MOV  *R14+,R0       GET OPEN TYPE
0069 0020 130D              JEQ  RTN
0070                  *
0071                  *  WAIT FOR 1.25 SECONDS ON OPEN TO PERMIT AUDIO CASSETTE
0072                  *  TO COME UP TO SPEED & FOR AGC TO GET SETUP.
0073                  *
0074 0022 020C              LI   R12,CR9901             SET THE CRU BASE TO THE 9901
     0024 0100
0075 0026 3020              LDCR @NUM8,0                LOAD THE INTERRUPT MASK
     0028 01B0'
0076 002A 0201              LI   R1,STWT                LOAD THE LOOP COUNT
     002C 0005
0077 002E 33C1              LDCR R1,15                  START CLOCK FOR SIGNAL
0078 0030 0200              LI   R0,>5BB3               LOAD THE WAIT COUNT
     0032 5BB3
0079        0034'     PAWS   EQU  $
0080 0034 06A0              BL   @SIGNAL
     0036 016E'
0081 0038 0601              DEC  R1                     COMPLETE
0082 003A 15FC              JGT  @PAWS                  NO
0083 003C 0380        RTN   RTWP                        AND RETURN
```

```
0085                ***********************************************************
0086                *                      READ PROCESSOR                     *
0087                ***********************************************************
0088       003E'    CSRD00 EQU  $
0089                *
0090                *  INTER-RECORD ENTRY, INTER-BLOCK ENTRY BELOW
0091                *
0092 003E 0700           SETO R0            SET FIRST 'HIGH'
0093 0040 0709           SETO R9            SET MAXIMUM VALUE IN R9
0094                *
0095                *  JUMPS HERE AFTER INVERSION
0096                *
0097       0042'    AFTRIV EQU  $
0098 0042 06A0           BL   @COUNT        START CLOCK FOR FIRST TIME
     0044 00C8'
0099 0046 0204           LI   R4,INVCNT     LOAD COUNT BEFORE NEXT INVERSION
     0048 0019
0100                *
0101                *  JUMPS HERE AFTER WRONG EXPECTATION
0102                *
0103       004A'    WRONG  EQU  $
0104 004A 04C6           CLR  R6            SET 'MATCH EXPECTED'
0105 004C 0205           LI   R5,TRYCNT     SET COUNT FOR NUMBER OF CORRECT EXP
     004E 0011
0106                *
0107                *  JUMPS HERE AFTER CORRECT EXPECTATION
0108                *
0109       0050'    CORRCT EQU  $
0110 0050 0605           DEC  R5            HAVE ENOUGH CORRECT'S OCCURRED?
0111 0052 1311           JEQ  BITSYN        YES
0112 0054 0604           DEC  R4            TIME FOR INVERSION?
0113 0056 130D           JEQ  INVERT        YES
0114                *
0115 0058 06A0           BL   @COUNT        GET NEXT TIME
     005A 00C8'
0116 005C 60C1           S    R1,R3         CALCULATE DIFFERENCE IN TIMES
0117 005E 0743           ABS  R3
0118 0060 0283           CI   R3,THRESH     MATCH?
     0062 0005
0119 0064 1A03           JL   MATCH         YES
0120                *
0121                *  MISMATCH OCCURRED, WAS ONE EXPECTED
0122                *
0123 0066 0546           INV  R6
0124 0068 13F3           JEQ  CORRCT        YES
0125 006A 10EF           JMP  WRONG         NO
0126                *
0127                *  MATCH OCCURRED, WAS ONE EXPECTED
0128                *
0129       006C'    MATCH  EQU  $
0130 006C 0546           INV  R6
0131 006E 16F0           JNE  CORRCT        YES
0132 0070 10EC           JMP  WRONG         NO
0133                *
0134                *  INVERT THE LEVEL
0135                *
0136       0072'    INVERT EQU  $
0137 0072 0540           INV  R0
0138 0074 10E6           JMP  AFTRIV        CONTINUE
0139                *
```

```
0140                    *  CORRECT 'HIGH' LEVEL FOUND, FIND BIT BOUNDARIES
0141                    *
0142          0076'     BITSYN EQU  $
0143 0076 C240                   MOV  R0,R9          SAVE 'HIGH'
0144                    *
0145                    *
0146          0078'     GNB    EQU  $
0147 0078 0204                   LI   R4,1                  SET COUNT FOR GETBIT
     007A 0001
0148 007C 06A0                   BL   @GETBIT               GET A BIT
     007E 00E6'
0149 0080 0A18                   SLA  R8,1                  SHIFT PREVIOUS RESULT
0150 0082 A203                   A    R3,R8                 APPEND BIT JUST READ
0151 0084 0288                   CI   R8,SYNCB              IS THIS THE SYNC BYTE?
     0086 5596
0152 0088 16F7                   JNE  @GNB                  NO
0153                    *
0154 008A C21D                   MOV  *R13,R8               GET ADDR OF PTRS IN DBASIC
0155 008C 0206                   LI   R6,GETBYT             LOAD ADDRESS OF GETBYT
     008E 00E2'
0156                    *
0157                    *  CLEAR CHECKSUM AND GET RECORD TYPE BYTE
0158                    *
0159 0090 04C7                   CLR  R7                    CLEAR CHECKSUM
0160 0092 0696                   BL   *R6                   GET THE RECORD TYPE BYTE
0161 0094 0283                   CI   R3,EOT                CHECK IF EOT RECORD
     0096 0004
0162 0098 130F                   JEQ  EOTRTN                EQ - YES- EOT RETURN
0163                    *
0164                    *  TURN ON PROGRAM ENABLE LED EACH TIME A RECORD IS
0165                    *  BEING READ. (DISABLE PROGRAMMING PULSE.)
0166                    *
0167 009A C30A                   MOV  R10,R12               SET CRU BASE TO CASSETTE
0168 009C 1E04                   SBZ  4                     TURN OFF PROGRAMMING PULSE
0169 009E 1D03                   SBO  3                     TURN ON PROGRAM ENABLE PULSE
0170                    *
0171                    *  GET DATA BYTES AND PLACE IN I/O BUFFER
0172                    *
0173          00A0'     GDAT   EQU  $
0174 00A0 0696                   BL   *R6                   GET BYTE
0175 00A2 06C3                   SWPB R3                    POSITION BYTE FOR STORAGE
0176 00A4 DE03                   MOVB R3,*R8+               STORE BYTE INTO I/O BUFFER
0177 00A6 16FC                   JNE  GDAT                  NULL TERMINATES DATA
0178                    *
0179                    *  TURN OFF PROGRAM ENABLE LED AFTER READING RECORD.
0180                    *
0181 00A8 C30A                   MOV  R10,R12               SET CRU BASE TO CASSETTE
0182 00AA 1E03                   SBZ  3                     TURN OFF PROGRAM ENABLE
0183                    *
0184                    *  GET CHECKSUM AND COMPARE
0185                    *
0186 00AC 0507                   NEG  R7                    NEGATE CHECKSUM
0187 00AE 0696                   BL   *R6                   GET CHECKSUM BYTE
0188 00B0 0247                   ANDI R7,>FF                MASK OFF CALCUALTED CHECKSUM
     00B2 00FF
0189 00B4 1304                   JEQ  NORMRT                EQ - CHECKSUM OK
```

```
0191                    *
0192                    *  RETURN - ERROR OR EOT RETURN - STOP CASSETTE & CLOCK
0193                    *
0194          00B6' ERRTN  EQU  $
0195 00B6 05CE          INCT R14                INCREMENT OVER EOT RETURN
0196          00B8' EOTRTN EQU  $                EOT RETURN
0197 00B8 06A0          BL   @CSSTOP            STOP THE CASSETTE & CLOCK
     00BA 01F4'
0198 00BC 0380          RTWP                    AND RETURN
0199                    *
0200                    *   RETURN - NORMAL   - STOP CLOCK
0201                    *
0202          00BE' NORMRT EQU  $
0203 00BE 06A0          BL   @CKSTOP            STOP THE 9901 CLOCK
     00C0 0202'
0204 00C2 05CE          INCT R14                INCREMENT OVER EOT RETURN
0205 00C4 05CE          INCT R14                INCREMENT OVER ERROR RETURN
0206 00C6 0380          RTWP                    AND RETURN
```

```
0208                    *
0209                    *
0210                    *    THIS ROUTINE IS USED IN SYNCHING UP WITH THE BIT
0211                    *    BOUNDARY OF THE LEADING SEGMENT OF THE FIRST BLOCK
0212                    *    OF A RECORD.  IT TIMES THE DURATION OF ONE CYCLE.
0213                    *
0214                    *
0215          00C8' COUNT  EQU  $
0216 00C8 C1CB          MOV  R11,R7        SAVE RETURN LINK
0217 00CA C0C1          MOV  R1,R3         SAVE PREVIOUS LINK
0218 00CC 0540          INV  R0            WAIT FOR 'LOW'
0219 00CE 06A0          BL   @WAITB
     00D0 0120'
0220 00D2 0540          INV  R0            WAIT FOR NEXT 'HIGH'
0221 00D4 06A0          BL   @WAITB
     00D6 0120'
0222 00D8 1D00          SBO  0             SET 9901 TO CLOCK MODE
0223 00DA 37C1          STCR R1,15         GET THE TIME OF THE CYCLE
0224 00DC 33C9          LDCR R9,15         RESTART TIMER
0225 00DE 1E00          SBZ  0             SET 9901 OUT OF CLOCK MODE
0226 00E0 0457          B    *R7           RETURN
```

```
0228                    *
0229                    *       THIS ROUTINE GETS A BIT/BYTE DEPENDING
0230                    *       UPON THE ENTRY POINT CHOSEN:
0231                    *            GETBIT - GET A BIT (CALLER SETS R4 TO 1)
0232                    *            GETBYT - GET A BYTE
0233                    *
0234                    *       R0 & R2 - USED BY WAITB
0235                    *       R3 - RIGHT JUSTIFIED BIT/BYTE
0236                    *       R4 - COUNTER, LEFT AT 0
0237                    *       R5 - RETURN LINK
0238                    *
0239 00E2 0204  GETBYT LI   R4,8                BYTE ENTRY
     00E4 0008
0240                    *
0241                    *
0242                    *    FOR GETBIT ENTRY, CALLER MUST SET R4 TO 1
0243                    *      (THIS ENTRY ONLY USED IN ONE PLACE)
0244                    *
0245 00E6 C14B  GETBIT MOV  R11,R5               SAVE RETURN LINK
0246 00E8 04C3         CLR  R3                   CLEAR RESULT REGISTER
0247                    *
0248 00EA 0A13  GNBIT  SLA  R3,1                 SHIFT PREVIOUS BITS
0249 00EC C009         MOV  R9,R0                SET HIGH LEVEL TO DETECT
0250 00EE 0540         INV  R0
0251 00F0 06A0         BL   @WAITB
     00F2 0120'
0252 00F4 0540         INV  R0
0253                    *
0254                    *  GET TIME FOR CYCLE
0255                    *
0256 00F6 0701         SETO R1                   SET MAX. CLOCK TIME
0257 00F8 06A0         BL   @WAITB
     00FA 0120'
0258                    *
0259                    *       START TIMER
0260                    *
0261 00FC 33C1         LDCR R1,15                LOAD CLOCK COUNT
0262 00FE 1E00         SBZ  0                    TAKE 9901 OUT OF CLOCK MODE
0263                    *
0264                    *  WAIT FOR ONE CYCLE
0265                    *
0266 0100 0540         INV  R0                   SET THE LEVEL TO BE DETECTED
0267 0102 06A0         BL   @WAITB
     0104 0120'
0268 0106 0540         INV  R0
0269 0108 06A0         BL   @WAITB
     010A 0120'
0270                    *
0271                    *       GET ELAPSED TIME
0272                    *
0273 010C 1D00         SBO  0                    SET 9901 BACK TO CLOCK MODE
0274 010E 37C1         STCR R1,15                GET THE ELAPSED TIME
```

```
0276                    *
0277                    *       ONE, ZERO, OR GARBAGE?
0278                    *
0279 0110 0281          CI      R1,@LOZER
     0112 7FC7
0280 0114 1A01          JL      @BITZER
0281                    *
0282 0116 0583          INC     R3                      ONE DETECTED
0283                    *
0284 0118 0604  BITZER  DEC     R4                      DONE?
0285 011A 15E7          JGT     @GNBIT                  NO
0286                    *
0287                    *
0288 011C A1C3          A       R3,R7                   UPDATE THE CHECKSUM
0289 011E 0455          B       *R5                     RETURN
```

```
0291                    *
0292                    *       THIS ROUTINE LOOKS FOR A GIVEN LEVEL.
0293                    *       THE LEVEL MUST BE SEEN TWICE IN SUC-
0294                    *       CESSION BEFORE CONTROL IS RETRUNED
0295                    *            ENTER:        R0 - LEVEL:
0296                    *                                0 - LOW
0297                    *                                1 - HIGH
0298                    *            EXIT:         R2 - 0
0299                    *                          R12 - 9901
0300                    *
0301 0120 C30A  WAITB   MOV   R10,R12           SET CRU BASE TO CASSETTE
0302                    *
0303 0122 0202  RLOAD   LI    R2,@SCANCT        LOAD THE # OF REPEAT PASSES
     0124 0003
0304                    *
0305 0126 1F00  INPBIT  TB    0                 CHECK INPUT BIT
0306 0128 1303          JEQ   @TSTONE           ONE DETECTED, LOOKING FOR HIGH
0307 012A C000          MOV   R0,R0             LOOKING FOR A LOW?
0308 012C 16FA          JNE   @RLOAD            NO, TRY AGAIN
0309 012E 1002          JMP   @CORLEV           CORRECT LEVEL DETECTED.
0310 0130 C000  TSTONE  MOV   R0,R0             LOOKING FOR A HIGH?
0311 0132 13F7          JEQ   @RLOAD            NO, TRY AGAIN
0312 0134 0602  CORLEV  DEC   R2                YES, SEEN TWICE?
0313 0136 15F7          JGT   @INPBIT           NO
0314                    *
0315                    *       LEVEL FOUND, RETURN
0316 0138 020C          LI    R12,CR9901        SET CRU BASE TO THE TMS9901
     013A 0100
0317 013C 0458          RT
```

```
0319                ********************************************************
0320                *                    WRITE PROCESSOR                   *
0321                ********************************************************
0322      013E'     CSWT00 EQU  $
0323                *
0324                *  WRITE THE FILL BYTES AND SYNC BYTE
0325                *
0326 013E 06A0            BL   @SUBREC           WRITE FILL BYTES AND SYNC BYTE
     0140 0180'
0327                *
0328                *  GET I/O BUFFER ADDRESS
0329                *
0330 0142 C21D            MOV  *R13,R8           GET ADDR OF PTRS IN DBASIC
0331                *
0332                *  WRITE THE RECORD TYPE
0333                *
0334 0144 0201            LI   R1,>1700          SET THE RECORD TYPE TO EOB
     0146 1700
0335                *
0336                *  WRITE THE DATA BYTES
0337                *
0338      0148'     WDATA  EQU  $
0339 0148 0696            BL   *R6               OUTPUT TYPE & DATA BYTES
0340 014A D078            MOVB *R8+,R1           GET THE DATA BYTE
0341 014C 16FD            JNE  WDATA             NULL BYTE TERMINATES BUFFER
0342 014E 0696            BL   *R6               OUTPUT NULL
0343                *
0344                *  WRITE CHECKSUM BYTE
0345                *
0346 0150 D047            MOVB R7,R1             PLACE CHECKSUM BYTE INTO BUFFR
0347 0152 0696            BL   *R6               OUTPUT THE CHECKSUM BYTE
0348                *
0349                *  WAIT FOR COMPLETION OF LAST CYCLE
0350                *                  AND
0351                *  WAIT LOOP BEFORE RETURNING TO ALLOW SUFFICIENT TIME FOR
0352                *  CURRENT RECORD INTERPRETATION ON LOADING BEFORE WRITING
0353                *  NEXT RECORD.
0354                *
0355 0154 0201            LI   R1,4              LOOP COUNT FOR 300 MSEC (3+1)
     0156 0004
0356 0158 0200            LI   R0,>1256*2+1      100 MSEC CLOCK COUNT
     015A 24AD
0357 015C 06A0  LOOP1     BL   @SIGNAL       PERFORM INITIAL WAIT & SET UP AGAIN
     015E 016E'
0358 0160 C30A            MOV  R10,R12           SET CRU BASE BACK TO THE CST
0359 0162 1D02            SBO  2                 SET FINAL LEVEL OF HIGH
0360 0164 0601            DEC  R1                DECREMENT COUNTER
0361 0166 15FA            JGT  LOOP1             NOT DONE
0362 0168 06A0            BL   @CKSTOP           STOP THE 9901 CLOCK
     016A 0202'
0363 016C 0380            RTWP                   AND RETURN
```

```
0365                    *
0366                    *     THIS ROUTINE WAITS FOR THE SIGNAL FROM THE
0367                    *     9901 AND THEN RESTARTS THE TIMER WITH COUNT
0368                    *     VALUE IN R0 AND RETURNS.
0369                    *
0370                    *           ENTRY:      R0 CONTAINS COUNT, NOT CHANGED.
0371                    *
0372          016E'     SIGNAL EQU  $
0373 016E 020C                   LI   R12,CR9901        LOAD CRU BASE TO 9901
     0170 0100
0374                    *
0375                    *  WAIT
0376                    *
0377          0172'     WAITLP EQU  $
0378 0172 1F0F                   TB   15                INTREQ?
0379 0174 16FE                   JNE  WAITLP            NO
0380                    *
0381                    *  LOAD NEW CLOCK VALUE
0382                    *
0383          0176'     LOADCR EQU  $
0384 0176 33C0                   LDCR R0,15             LOAD THE COUNT VALUE
0385 0178 1E00                   SBZ  0                 ENTER INTERRUPT MODE
0386 017A 1D03                   SBO  3                 ENABLE CLOCK INTERRUPT
0387 017C 1D00                   SBO  0                 ENTER CLOCK MODE
0388 017E 045B                   RT                     AND RETURN
0389                    *
0390                    *  THIS ROUTINE FALLS THROUGH TO ALLOW CONCURRENT ACTIONS.
0391                    *  A WAIT LOOP MUST FOLLOW BEFORE NON-CONCURRENT ACTIONS
0392                    *  MAY OCCUR.
0393                    *
```

```
0395                    *
0396                    *  ROUTINE TO WRITE FILL BYTES AND SYNC BYTE
0397                    *
0398          0180'     SUBREC EQU  $
0399 0180 C20B                   MOV  R11,R8              SAVE RETURN
0400 0182 020C                   LI   R12,CR9901          SET THE CRU BASE TO THE 9901
     0184 0100
0401 0186 3020                   LDCR @NUM8,0             LOAD THE INTERRUPT MASK
     0188 0180'
0402 018A 33E0                   LDCR @STWT,15            START THE CLOCK
     018C 0005
0403 018E 0206                   LI   R6,WBYTE            LOAD THE ADDRESS OF WBYTE
     0190 01AE'
0404 0192 0209                   LI   R9,@NUMFIL          SET # OF FILL BYTES FOR LEADER
     0194 0096
0405 0196 0229                   AI   R9,NUMZ         SET # OF FILL BYTES TO BE OUTPUT
     0198 0028
0406                    *
0407 019A 0201    WFILL  LI   R1,>5555            SET UP FILL BYTE
     019C 5555
0408 019E 0696                   BL   *R6                 OUTPUT THE BYTE
0409 01A0 0609                   DEC  R9                  DECREMENT THE COUNT
0410 01A2 15FB                   JGT  WFILL               DONE?
0411                    *
0412                    *  WRITE THE SYNC BYTE
0413                    *
0414 01A4 0201                   LI   R1,SYNC             LOAD THE SYNC BYTE
     01A6 9600
0415 01A8 0696                   BL   *R6                 WRITE THE SYNC BYTE
0416                    *
0417 01AA 04C7                   CLR  R7                  CLEAR THE CHECKSUM
0418 01AC 0458                   B    *R8                 RETURN
```

```
0420                    *
0421                    *       THIS ROUTINE WRITES A BYTE
0422                    *            R0 - WAIT COUNT (HIGH)
0423                    *            R1 - BYTE TO BE OUTPUTTED
0424                    *            R2 - COUNT
0425                    *            R3 - BIT MASKED OFF (SCRATCH)
0426                    *            R4 - WAIT COUNT (LOW)
0427                    *            R5 - RETURN LINK
0428                    *
0429 01AE 0202  WBYTE   LI   R2,8               LOAD THE COUNT
     01B0 0008
0430 01B2 C14B          MOV  R11,R5             SAVE THE RETURN LINK
0431 01B4 B1C1          AB   R1,R7              UPDATE THE CHECKSUM
0432                    *
0433                    *       SELECT CYCLE TIMES
0434                    *
0435 01B6 0200  WBIT    LI   R0,@ZERHI          # OF 9901 CYCLES FOR ZERO HIGH
     01B8 0025
0436 01BA C0C1          MOV  R1,R3              MOVE BYTE TO SCRATCH REGISTER
0437 01BC 0243          ANDI R3,>8000           MASK OUT BIT
     01BE 8000
0438 01C0 1303          JEQ  @WZERO             WRITE ZERO IF RESULT IS ZERO
0439 01C2 0200          LI   R0,@ONEHI          # OF 9901 CYCLES FOR ONE HIGH
     01C4 0011
0440 01C6 04C3          CLR  R3                 CLEAR R3 TO SIGNAL FIRST CYCLE
0441 01C8 06A0  WZERO   BL   @SIGNAL            SIGNAL START OF 'HIGH' TIME
     01CA 016E'
0442 01CC C30A          MOV  R10,R12            SET CRU BASE TO THE CASSETTE
0443 01CE 1D02          SBO  2                  CASSETTE BIT HIGH
0444 01D0 06A0          BL   @SIGNAL            START SIGNAL OF 'LOW' TIME
     01D2 016E'
0445 01D4 C30A          MOV  R10,R12            SET CRU BASE BACK TO THE CASSE
0446 01D6 1E02          SBZ  2                  CASSETTE BIT LOW
0447 01D8 0543          INV  R3                 BIT COMPLETED?
0448 01DA 16F6          JNE  @WZERO             NO
0449 01DC 0A11          SLA  R1,1               SHIFT THE BYTE
0450 01DE 0602          DEC  R2                 DECREMENT COUNT
0451 01E0 15EA          JGT  @WBIT              NOT DONE?
0452 01E2 0455          B    *R5                RETURN
0453      01B0' NUM8    EQU  WBYTE+2            POINTS AT VALUE >0008
```

```
0455                ************************************************
0456                *                 CLOSE PROCESSOR                  *
0457                ************************************************
0458      01E4' CSCL00 EQU  $
0459 01E4 06A0         BL   @SUBREC            WRITE FILL BYTES & SYNC BYTE
     01E6 0180'
0460 01E8 0201         LI   R1,>0400           LOAD EOT BYTE
     01EA 0400
0461 01EC 0696         BL   *R6                OUTPUT THE EOT BYTE
0462 01EE 06A0         BL   @CSSTOP            STOP CASSETTE
     01F0 01F4'
0463 01F2 0380         RTWP                    RETURN
0464                *
0465                *  STOP THE CASSETTE & STOP THE CLOCK
0466                *
0467      01F4' CSSTOP EQU  $
0468 01F4 C20B         MOV  R11,R8             SAVE RETURN
0469 01F6 C30A         MOV  R10,R12            GET CASSETTE CRU BASE
0470 01F8 1E00         SBZ  0                  STOP CASSETTE DRIVE #1
0471 01FA 1E01         SBZ  1                  STOP CASSETTE DRIVE #2
0472 01FC 06A0         BL   @CKSTOP            STOP THE 9901 CLOCK
     01FE 0202'
0473 0200 0458         B    *R8                RETURN
0474                *
0475                *  STOP THE CLOCK BEFORE RETURNING TO POWER BASIC
0476                *
0477      0202' CKSTOP EQU  $
0478 0202 C24B         MOV  R11,R9             SAVE RETURN
0479 0204 0200         LI   R0,1               CLOCK VALUE OF ZERO
     0206 0001
0480 0208 020C         LI   R12,CR9901         SET CRU BASE TO 9901
     020A 0100
0481 020C 06A0         BL   @LOADCR            OUTPUT COUNT VALUE
     020E 0176'
0482 0210 0459         B    *R9                AND RETURN
0483                   END
NO ERRORS
```

LABEL		VALUE	DEFN	REFERENCES
$		0212'		0046 0054 0079 0088 0097 0103 0109 0129 0136 0142 0146 0173 0194 0196 0202 0215 0322 0338 0372 0377 0383 0398 0458 0467 0477
AFTRIV		0042'	0097	0138
BADTHR		0020	0041	
BITSYN		0076'	0142	0111
BITZER		0118'	0284	0280
CHKSM		0032	0033	
CKSTOP		0202'	0477	0203 0362 0472
CLKADR	R	0016'	0016	0059
CORLEV		0134'	0312	0309
CORRCT		0050'	0109	0124 0131
COUNT		00C8'	0215	0098 0115
CR9901		0100	0021	0074 0316 0373 0400 0480
CRCASS		1700	0022	0049
CSCL00	D	01E4'	0458	0014
CSCLZ	R		0015	
CSOP00	D	0000'	0046	0014
CSOP10		0012'	0054	0052
CSOPZ	R		0015	
CSRD00	D	003E'	0088	0014
CSRDZ	R		0015	
CSSTOP		01F4'	0467	0197 0462
CSWT00	D	013E'	0322	0014
CSWTZ	R		0015	
EOT		0004	0031	0161
EOTRTN		00B8'	0196	0162
ERRTN		00B6'	0194	
GARB		0033	0034	
GDAT		00A0'	0173	0177
GETBIT		00E6'	0245	0148
GETBYT		00E2'	0239	0155
GNB		0078'	0146	0152
GNBIT		00EA'	0248	0285
HIZER		7FA5	0027	
INPBIT		0126'	0305	0313
INVCNT		0019	0039	0099
INVERT		0072'	0136	0113
LOADCR		0176'	0383	0481
LOOP1		015C'	0357	0361
LOZER		7FC7	0028	0279
MATCH		006C'	0129	0119
NORMRT		00BE'	0202	0189
NUM8		01B0'	0453	0075 0401
NUMFIL		0096	0024	0404
NUMZ		0028	0023	0405
ONEHI		0011	0026	0439
PAWS		0034'	0079	0082
R0		0000		0051 0068 0078 0092 0137 0143 0218 0220 0249 0250 0252 0266 0268 0307 0307 0310 0310 0356 0384 0435 0439 0479
R1		0001		0076 0077 0081 0116 0217 0223 0256 0261 0274 0279 0334 0340 0346 0355 0360 0407 0414 0431 0436 0449 0460
R10		000A		0050 0167 0181 0301 0358 0442 0445 0469
R11		000B		0216 0245 0399 0430 0468 0478
R12		000C		0049 0050 0053 0074 0167 0181 0301 0316 0358 0373 0400 0442 0445 0469 0480
R13		000D		0051 0154 0330
R14		000E		0068 0195 0204 0205

R15	000F		0048								
R2	0002		0303	0312	0429	0450					
R3	0003		0059	0060	0061	0062	0116	0117	0118	0150	0161
			0175	0176	0217	0246	0248	0282	0288	0436	0437
			0440	0447							
R4	0004		0099	0112	0147	0239	0284				
R5	0005		0105	0110	0245	0289	0430	0452			
R6	0006		0104	0123	0130	0155	0160	0174	0187	0339	0342
			0347	0403	0408	0415	0461				
R7	0007		0159	0186	0188	0216	0226	0288	0346	0417	0431
R8	0008		0149	0150	0151	0154	0176	0330	0340	0399	0418
			0468	0473							
R9	0009		0093	0143	0224	0249	0404	0405	0409	0478	0482
RLOAD	0122'	0303	0308	0311							
RTN	003C'	0083	0069								
SCANCT	0003	0032	0303								
SIGNAL	016E'	0372	0080	0357	0441	0444					
STWT	0005	0035	0076	0402							
SUBREC	0180'	0398	0326	0459							
SYNC	9600	0030	0414								
SYNCB	5596	0029	0151								
THRESH	0005	0040	0118								
TRYCNT	0011	0037	0105								
TSTONE	0130'	0310	0306								
WAITB	0120'	0301	0219	0221	0251	0257	0267	0269			
WAITLP	0172'	0377	0379								
WBIT	0186'	0435	0451								
WBYTE	01AE'	0429	0403	0453							
WDATA	0148'	0338	0341								
WFILL	019A'	0407	0410								
WRONG	004A'	0103	0125	0132							
WZERO	01C8'	0441	0438	0448							
ZERHI	0025	0025	0435								

*

```
SOURCE ACCESS NAME=      DBASIC.REL0.EXTENDED.SOURCE.PROGRAM
OBJECT ACCESS NAME=      DBASIC.REL0.EXTENDED.OBJECT.PROGRAM
LISTING ACCESS NAME=     DBASIC.REL0.EXTENDED.LIST.PROGRAM
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0002                    IDT  'PROGRAM'
0003                *
0004                *****
0005                *
0006                *  THIS IS THE INTERFACE TO THE EPROM PROGRAMMER.  IT WILL
0007                *  RELOCATE THE BASIC INTERNAL CODE AND PERFORM THE EPROM
0008                *  PROGRAMMING.
0009                *
0010                *****
0011                *
0012                *  INPUT:
0013                *     NONE
0014                *
0015                *  OUTPUT:
0016                *    NONE
0017                *
0018                *  CALLING SEQ:
0019                *     BL    *R0                WHERE R0 CONTAINS ADDR OF PROGM
0020                *
0021                *
```

```
0023                    *
0024                    *  EXTERNAL REFERENCES AND DEFINITIONS
0025                    *
0026                          DEF  PROGM
0027                    *
0028                          REF  GETC,PRDY,TYPE,TYP0
0029                          REF  BUS,NVD,SLT,VNT,VDT
0030                          REF  IDRUN
0031                    *
0032                    *  DATA BASE
0033                    *  CONSTANTS
0034                    *
0035                    *
0036                    *   PROGRAMMING LOOP CONSTANTS
0037                    *
0038          0064      PRGPAS EQU  100
0039          000A      VERPAS EQU  10
0040                    *
0041                    *  CRU BASES AND DATA
0042                    *
0043          1710      CBEPRO EQU  >1710
0044                    *
0045                    *
0046                    *  MESSAGES
0047                    *
0048          0000'     MSG01  EQU  $
0049 0000     0D               BYTE >0D,>0A
     0001     0A
0050 0002     52               TEXT 'RUN? '
     0003     55
     0004     4E
     0005     3F
     0006     20
0051 0007     00               BYTE 0
0052          0008'     MSG02  EQU  $
0053 0008     0D               BYTE >0D,>0A,>0A
     0009     0A
     000A     0A
0054 000B     4D               TEXT 'MOUNT EPROM'
     000C     4F
     000D     55
     000E     4E
     000F     54
     0010     20
     0011     45
     0012     50
     0013     52
     0014     4F
     0015     4D
0055          0016'     MSG03  EQU  $
0056 0016     0D               BYTE >0D,>0A
     0017     0A
0057 0018     45               TEXT 'EPROM READY? '
     0019     50
     001A     52
     001B     4F
     001C     4D
     001D     20
     001E     52
     001F     45
```

```
                0020   41
                0021   44
                0022   59
                0023   3F
                0024   20
0058            0025   00          BYTE >0
0059                0026' MSG04     EQU  $
0060            0026   0D          BYTE >0D,>0A
                0027   0A
0061            0028   50          TEXT 'PROGRAMMING'
                0029   52
                002A   4F
                002B   47
                002C   52
                002D   41
                002E   4D
                002F   4D
                0030   49
                0031   4E
                0032   47
0062            0033   00          BYTE 0
0063                0034' MSG05     EQU  $
0064            0034   56          TEXT 'VERIFYING'
                0035   45
                0036   52
                0037   49
                0038   46
                0039   59
                003A   49
                003B   4E
                003C   47
0065            003D   0D          BYTE >0D,>0A
                003E   0A
0066            003F   00          BYTE 0
0067                0040' MSG06     EQU  $
0068            0040   56          TEXT 'VERIFY ERROR!'
                0041   45
                0042   52
                0043   49
                0044   46
                0045   59
                0046   20
                0047   45
                0048   52
                0049   52
                004A   4F
                004B   52
                004C   21
0069            004D   0D          BYTE >0D,>0A
                004E   0A
0070            004F   00          BYTE 0
0071                0050' MSG07     EQU  $
0072            0050   0D          BYTE >0D,>0A
                0051   0A
0073            0052   4D          TEXT 'MEM BYTE = '
                0053   45
                0054   4D
                0055   20
                0056   42
                0057   59
```

```
      0058   54
      0059   45
      005A   20
      005B   3D
      005C   20
0074  005D   00          BYTE 0
0075       005E' MSG08   EQU  $
0076  005E   20          TEXT ' COMPLETE'
      005F   43
      0060   4F
      0061   4D
      0062   50
      0063   4C
      0064   45
      0065   54
      0066   45
0077  0067   0D          BYTE >0D,>0A
      0068   0A
0078  0069   00          BYTE 0
0079       006A' MSG09   EQU  $
0080  006A   20          TEXT ' TERMINATED'
      006B   54
      006C   45
      006D   52
      006E   4D
      006F   49
      0070   4E
      0071   41
      0072   54
      0073   45
      0074   44
0081  0075   0A          BYTE >0A,>0D
      0076   0D
0082  0077   00          BYTE 0
0083       0078' LOW     EQU  $
0084  0078   4C          TEXT 'LOW'
      0079   4F
      007A   57
0085  007B   00          BYTE 0
0086       007C' HIGH    EQU  $
0087  007C   48          TEXT 'HIGH '
      007D   49
      007E   47
      007F   48
      0080   20
0088  0081   00          BYTE 0
0089       0082' CRLF    EQU  $
0090  0082   0A          BYTE >0A,>0D
      0083   0D
0091  0084   00          BYTE 0
```

```
0093                    *
0094                    *  EPROM PROGRAM & VERIFY PROCESSOR
0095                    *
0096          0085'     PROGM  EQU  $              ENTRY POINT
0097                    *
0098                    *  DETERMINE RUN OR NO RUN
0099                    *
0100 0086 0201          PROM01 LI   R1,MSG01       'RUN?'
     0088 0000'
0101 008A 06A0                 BL   @TYPE
     008C 0000
0102 008E 0205                 LI   R5,IDRUN       RUN VALUE
     0090 0000
0103 0092 06A0                 BL   @YORN          GET USER RESPONSE
     0094 0134'
0104 0096 10F7                   JMP PROM01        NEITHER Y NOR N
0105 0098 1001                   JMP PROM02        Y - TAKE DEFAULT
0106 009A 0545                   INV R5            N - USE NORUN VALUE
0107                    *
0108                    *       STORE CRITICAL PARAMETERS IN 1ST ROM
0109                    *
0110                    *       0 - RUN/NORUN FLAG  ( >A5A5=RUN, >5A5A=NORUN )
0111                    *       1 - SLT  STATEMENT LOCATION TABLE
0112                    *       2 - VNT  VARIABLE NAME TABLE
0113                    *       3 - NVD-VDT  LENGTH OF VARIABLE DEF TABLE
0114                    *
0115 009C C0E9          PROM02 MOV  @BUS(9),R3     GET START OF USER AREA
     009E 0000
0116 00A0 C129                 MOV  @VDT(9),R4     GET END OF USER AREA
     00A2 0000
0117 00A4 0223                 AI   R3,-8          BACKUP FOR EPROM POINTERS
     00A6 FFF8
0118 00A8 C083                 MOV  R3,R2          SAVE R3
0119 00AA C204                 MOV  R4,R8          SAVE R4
0120 00AC CC85                 MOV  R5,*R2+        STORE RUN/NORUN FLAG
0121 00AE C4A9                 MOV  @SLT(9),*R2    STORE SLT POINTER
     00B0 0000
0122 00B2 6CA9                 S    @BUS(9),*R2+   GET SLT DISPLACEMENT
     00B4 009E'
0123 00B6 C4A9                 MOV  @VNT(9),*R2    STORE VNT POINTER
     00B8 0000
0124 00BA 6CA9                 S    @BUS(9),*R2+   GET VNT DISPLACEMENT
     00BC 00B4'
0125 00BE C4A9                 MOV  @NVD(9),*R2    CALCULATE NVD-VDT
     00C0 0000
0126 00C2 6484                 S    R4,*R2         STORE NVD-VDT
0127 00C4 C108          PROM05 MOV  R8,R4          LAST BYTE SAVED IN R8
0128 00C6 04C5          PROM10 CLR  R5             ZERO EPROM STARTING ADDR
0129                    *
0130                    *  OUTPUT PROMPT
0131                    *
0132          00C8'     PRMT00 EQU  $
0133 00C8 0201                 LI   R1,MSG02       "EPROM READY? " PROMPT
     00CA 0008'
0134 00CC 06A0          PRMT01 BL   @TYPE          OUTPUT STRING R1
     00CE 008C'
0135 00D0 06A0                 BL   @YORN          GET RESPONSE
     00D2 0134'
0136 00D4 10F9                   JMP PRMT00        NEITHER Y NOR N
0137 00D6 1001                   JMP PROM20        Y - GO AHEAD
```

```
0138 00D8 10F7            JMP PRMT00            N - WAIT TILL READY
0139                 *
0140                 *  CHECK IF TMS2716 EPROM MOUNTED
0141                 *
0142 00DA 020C  PROM20 LI   R12,@CBEPRO-8        EPROM CARD ID CRU BASE
     00DC 1708
0143 00DE 04C0         CLR  R0                   CLEAR R0 FOR CRU INPUT
0144 00E0 3500         STCR R0,4                 INPUT CARD ID
0145 00E2 0280         CI   R0,>0800             TEST IF TMS2716 EPROM
     00E4 0800
0146 00E6 1303         JEQ  SIZE                 EQ - CONTINUE
0147 00E8 0201         LI   R1,MSG03             'EPROM READY?'
     00EA 0016'
0148 00EC 10EF         JMP  PRMT01               OUTPUT & TRY AGAIN
```

```
0150                    *
0151                    *  SIZE RAM & EPROM AREA
0152                    *
0153          00EE'     SIZE   EQU  $
0154 00EE C284                 MOV  R4,R10          UPPER BOUND
0155 00F0 6283                 S    R3,R10          # OF BYTES
0156                    *
0157                    *  TEST IF SUFFICIENT EPROM AREA
0158                    *
0159 00F2 0207                 LI   R7,4096         EPROM SIZE(ACTUALLY 2048 BYTS)
     00F4 1000
0160 00F6 61C5                 S    R5,R7           REMAINING EPROM AREA
0161 00F8 81CA                 C    R10,R7          COMPARE AREAS
0162 00FA 1202                 JLE  PROGRM          SUFFICIENT AREA - CONTINUE
0163                    *
0164                    *  ADJUST POINTERS FOR EPROM SIZE
0165                    *
0166 00FC A1C3                 A    R3,R7           AMOUNT THAT WILL FIT
0167 00FE C107                 MOV  R7,R4           R4 CONTAINS UPPER BOUND+2
```

```
0169                    *
0170                    *  PROGRAM THE EPROM
0171                    *
0172          0100'     PROGRM EQU  $
0173 0100 0201                   LI   R1,MSG04          'PROGRAMMING'
     0102 0026'
0174 0104 06A0                   BL   @TYPE             OUTPUT STRING R1
     0106 00CE'
0175 0108 0201                   LI   R1,CRLF           'CR/LF'
     010A 0082'
0176 010C 06A0                   BL   @TYPE             OUTPUT STRING R1
     010E 0106'
0177 0110 0702                   SETO R2                SET PROGRAMMING FLAG (-1)
0178 0112 020D                   LI   R13,@PRGPAS       LOAD LOOP COUNT   (>64)
     0114 0064
0179 0116 020C                   LI   R12,@CBEPRO       SET CRU BASE TO EPROM
     0118 1710
0180 011A C38C                   MOV  R12,R14           SAVE EPROM CRU BASE
0181 011C 1DFB                   SBO  -5                SET PROGRAM ENABLE
0182 011E 06A0                   BL   @TOPR             BRANCH TO TOP OF THE LOOP
     0120 017C'
0183                    *
0184                    *
0185 0122 C30E                   MOV  R14,R12       SET CRU BASE TO EPROM DATA CHANNEL
0186 0124 3217                   LDCR *R7,8         LOAD BYTE FROM MEMORY.
0187 0126 1DFC                   SBO  -4            START PROGRAMMING PULSE
0188 0128 020B                   LI   R11,>A0       DELAY AT LEASE 1 MILLISEC
     012A 00A0
0189 012C 060B   LOOP1  DEC  R11                    DECREMENT AND
0190 012E 16FE                   JNE LOOP1          WAIT
0191 0130 1EFC                   SBZ  -4            STOP THE PROGRAMMING PULSE
0192 0132 102B                   JMP  BOTM
```

```
0194                    *
0195                    *  GET CHAR FROM KEYBOARD AND CHECK FOR Y OR N
0196                    *       BL @YORN
0197                    *         NEITHER Y NOR N RETURN
0198                    *         Y RETURN
0199                    *         N RETURN
0200 0134
0201         0134' YORN    EQU  $
0202 0134 C28B             MOV R11,R10
0203 0136 06A0   YORN1     BL @GETC                 GET CHAR FROM KEYBOARD
     0138 0000
0204 013A 10FD               JMP YORN1              NO CHAR
0205 013C 1002               JMP YORN2              VALID CHAR
0206 013E 0460               B @ESCAPE              ESCAPE CHAR
     0140 0206'
0207                    *
0208         0142' YORN2   EQU  $
0209 0142 0280             CI R0,>5900              Y?
     0144 5900
0210 0146 1304               JEQ YORN3
0211 0148 0280             CI R0,>4E00              N?
     014A 4E00
0212 014C 1602               JNE YORN4
0213 014E 05CA             INCT R10                 'N' DOUBLE SKIP
0214 0150 05CA   YORN3     INCT R10                 'Y' SINGLE SKIP
0215         0152' YORN4   EQU  $                   NEITHER NO SKIP
0216 0152 06A0             BL   TYPO                ECHO CHARACTER
     0154 0000
0217 0156 045A             B    *R10                RETURN
```

```
0219                    *
0220                    *  COMPARE CONTENTS OF EPROM TO THAT OF MEMORY
0221                    *
0222        0158'  VERIFY EQU  $
0223  0158  0201         LI   R1,MSG05          'VERIFYING'
      015A  0034'
0224  015C  06A0         BL   @TYPE             OUTPUT STRING R1
      015E  010E'
0225  0160  C30E         MOV  R14,R12           GET CRU BASE TO EPROM
0226  0162  1EFB         SBZ  -5                CLEAR PROGRAM ENABLE
0227  0164  04C2         CLR  R2                VERIFYING FLAG
0228  0166  020D         LI   R13,VERPAS        LOAD THE LOOP COUNT (10)
      0168  000A
0229  016A  06A0         BL   @TOPR             BRANCH TO TOP OF THE LOOP
      016C  017C'
0230                    *
0231                    *
0232  016E  C30E         MOV  R14,R12           GET CRU DATA BASE TO EPROM
0233  0170  3601         STCR R1,8              READ BYTE FROM THE EPROM.
0234  0172  9057         CB   *R7,R1            COMPARE MEMORY BYTE
0235  0174  130A         JEQ  @BOTM
0236                    *
0237                    *       MEMORY AND EPROM DO NOT MATCH - OUTPUT ERROR
0238                    *
0239  0176  0201         LI   R1,@MSG06         'VERIFY ERROR'
      0178  0040'
0240  017A  102B         JMP  OUTPUT            OUTPUT ERROR & "EPROM READY?"
```

```
0242                    *
0243                    *   LOOPS FOR TRAVERSAL OF MEMORY WITH EPROM
0244                    *
0245        017C'  TOPR    EQU  $
0246  017C  C28B          MOV  R11,R10          SAVE RETURN
0247  017E  C1C3  TOPR10   MOV  R3,R7          LOAD THE LOCAL MEMORY ADDRESS
0248  0180  C185          MOV  R5,R6            "   "     "   EPROM     "
0249        0182'  INRLP   EQU  $
0250  0182  020C          LI   R12,@CBEPRO+16 SET THE CRU BASE TO THE ADDRESS C
      0184  1720
0251  0186  33C6          LDCR R6,15          LOAD THE ADDRESS INTO THE EPROM.
0252  0188  045A          B    *R10           PERFORM THE CENTRAL TASK OF THIS FU
0253        018A'  BOTM    EQU  $
0254  018A  06A0          BL   @GETC               CHECK FOR ESCAPE
      018C  0138'
0255  018E  1002          JMP  BOTM1               NOTHING
0256  0190  1001          JMP  BOTM1               CHARACTER IN R0
0257  0192  1039          JMP  ESCAPE         ESCAPE AND RETURN
0258  0194  05C7  BOTM1    INCT R7             INCREMENT THE MEMORY ADDRESS.
0259  0196  0586          INC  R6                 "       "  EPROM     "
0260  0198  8107          C    R7,R4          IS THIS PASS COMPLETE?
0261  019A  11F3          JLT  @INRLP         NO                                1AF3
0262  019C  C082          MOV  R2,R2          PROGRAMMING?
0263  019E  130A          JEQ  CONT           NO, DON'T WAIT LOOP, CONTINUE
0264                    *
0265                    *  MUST PROGRAM ALL OF TMS2716 BEFORE PERFORMING THE NEXT
0266                    *  LOOP.  THEREFORE, DELAY AN AMOUNT EQUAL TO THE NUMBER OF
0267                    *  BITS NOT PROGRAMMED BEFORE CONTINUING.
0268                    *
0269  01A0  6185            S    R5,R6          # OF BYTES PROGRAMMED
0270  01A2  0207            LI   R7,2048        # OF BYTES IN EPROM
      01A4  0800
0271  01A6  61C6            S    R6,R7          # OF BYTES NOT PROGRAMMED
0272  01A8  020B    WAITP   LI   R11,>A0        DELAY MIN OF 1 MILLISEC
      01AA  00A0
0273  01AC  060B    WAITL   DEC  R11              PER BYTE FOR MAX # BYTES
0274  01AE  15FE              JGT  WAITL
0275  01B0  0607            DEC  R7
0276  01B2  15FA              JGT  WAITP
0277        01B4'  CONT     EQU  $
0278  01B4  060D            DEC  R13            DECREMENT THE LOOP COUNT.
0279  01B6  15E3            JGT  @TOPR10        HAVE ALL OF THE PASSES BEEN COMPLET
0280  01B8  C082            MOV  R2,R2               PROGRAMMING?
0281  01BA  11CE            JLT  VERIFY              EQ - YES - NOW VERIFY
```

WITH EPROM

RETURN
LOCAL MEMORY ADDRESS
" EPROM "

CRU BASE TO THE ADDRESS C

ADDRESS INTO THE EPROM.
HE CENTRAL TASK OF THIS FU

K FOR ESCAPE

ING
ACTER IN R0
) RETURN
THE MEMORY ADDRESS.
" EPROM "
ASS COMPLETE?

1AF3 JL

NG?
WAIT LOOP, CONTINUE

ORE PERFORMING THE NEXT
NT EQUAL TO THE NUMBER OF
INUING.

S PROGRAMMED
S IN EPROM

S NOT PROGRAMMED
OF 1 MILLISEC

E FOR MAX # BYTES

THE LOOP COUNT.
OF THE PASSES BEEN COMPLET
RAMMING?
YES - NOW VERIFY

```
0283                    *
0284                    *  EPROM FINISHED - OUTPUT MESSAGE
0285                    *
0286 01BC 0201               LI   R1,MSG07          "MEM BYTE = "
     01BE 0050'
0287 01C0 06A0               BL   @TYPE             OUTPUT STRING R1
     01C2 015E'
0288                    *
0289                    *  HIGH OR LOW BYTE
0290                    *
0291 01C4 C043               MOV  R3,R1             GET FIRST ADDR
0292 01C6 0911               SRL  R1,1              SHIFT OUT EVEN OR ODD
0293 01C8 1808               JOC  LOWBYT            CARRY - LOW (ODD) BYTE
0294                    *
0295                    *  HIGH BYTE PROGRAMMING COMPLETE
0296                    *
0297 01CA 0583               INC  R3                INC TO LOW BYTE
0298 01CC 0584               INC  R4                INC TO LOW BYTE
0299 01CE 0201               LI   R1,HIGH           "HIGH" BYTE
     01D0 007C'
0300 01D2 06A0  OUTPUT       BL   @TYPE             OUTPUT STRING R1
     01D4 01C2'
0301 01D6 0460               B    @PROM10           DO LOW ORDER BYTE
     01D8 00C6'
0302                    *
0303                    *  LOW BYTE PROGRAMMING COMPLETE
0304                    *
0305      01DA' LOWBYT   EQU  $
0306 01DA 0201               LI   R1,LOW            "LOW" BYTE
     01DC 0078'
0307 01DE 06A0               BL   @TYPE             OUTPUT STRING R1
     01E0 01D4'
0308 01E2 C0C7               MOV  R7,R3             SET NEXT MEMORY ADDR
0309 01E4 0603               DEC  R3                BACKUP TO EVEN ADDRESS
0310 01E6 8203               C    R3,R8             FINISHED?
0311 01E8 1402               JHE  RETURN            GE - FINISHED - RETURN
0312 01EA 0460               B    @PROM05           PROGRAM NEXT EPROM
     01EC 00C4'
```

```
 0314                    *
 0315                    *  RETURN
 0316                    *
 0317         01EE' RETURN EQU  $
 0318 01EE 0201          LI   R1,MSG04          'PROGRAMMING'
      01F0 0026'
 0319 01F2 06A0          BL   @TYPE             OUTPUT STRING R1
      01F4 01E0'
 0320 01F6 0201          LI   R1,MSG08          'COMPLETE
      01F8 005E'
 0321 01FA 06A0  RETN    BL   @TYPE             OUTPUT STRING R1
      01FC 01F4'
 0322 01FE C30E          MOV  R14,R12           GET EPROM DATA CHANNEL BASE
 0323 0200 1EFB          SBZ  -5                DISABLE PROGRAMMING
 0324 0202 0460          B    @PRDY
      0204 0000
 0325                    *
 0326                    *  ESCAPE RETURN
 0327                    *
 0328         0206' ESCAPE EQU  $
 0329 0206 0201          LI   R1,MSG04          'PROGRAMMING'
      0208 0026'
 0330 020A 06A0          BL   @TYPE             OUTPUT STRING R1
      020C 01FC'
 0331 020E 0201          LI   R1,MSG09          'TERMINATED'
      0210 006A'
 0332 0212 10F3          JMP  RETN              OUTPUT & RETURN
 0333                    END
NO ERRORS
```

LABEL		VALUE	DEFN	REFERENCES
$		0214'		0048 0052 0055 0059 0063 0067 0071 0075 0079 0083 0086 0089 0096 0132 0153 0172 0201 0208 0215 0222 0245 0249 0253 0277 0305 0317 0328
BOTM		018A'	0253	0192 0235
BOTM1		0194'	0258	0255 0256
BUS	R	008C'	0029	0115 0122 0124
CBEPRO		1710	0043	0142 0179 0250
CONT		0184'	0277	0263
CRLF		0082'	0089	0175
ESCAPE		0206'	0328	0206 0257
GETC	R	018C'	0028	0203 0254
HIGH		007C'	0086	0299
IDRUN	R	0090'	0030	0102
INRLP		0182'	0249	0261
LOOP1		012C'	0189	0190
LOW		0078'	0083	0306
LOWBYT		01DA'	0305	0293
MSG01		0000'	0048	0100
MSG02		0008'	0052	0133
MSG03		0016'	0055	0147
MSG04		0026'	0059	0173 0318 0329
MSG05		0034'	0063	0223
MSG06		0040'	0067	0239
MSG07		0050'	0071	0286
MSG08		005E'	0075	0320
MSG09		006A'	0079	0331
NVD	R	00C0'	0029	0125
OUTPUT		01D2'	0300	0240
PRDY	R	0204'	0028	0324
PRGPAS		0064	0038	0178
PRMT00		00C8'	0132	0136 0138
PRMT01		00CC'	0134	0148
PROGM	D	0085'	0096	0026
PROGRM		0100'	0172	0162
PROM01		0086'	0100	0104
PROM02		009C'	0115	0105
PROM05		00C4'	0127	0312
PROM10		00C6'	0128	0301
PROM20		00DA'	0142	0137
R0		0000		0143 0144 0145 0209 0211
R1		0001		0100 0133 0147 0173 0175 0223 0233 0234 0239 0286 0291 0292 0299 0306 0318 0320 0329 0331
R10		000A		0154 0155 0161 0202 0213 0214 0217 0246 0252
R11		000B		0188 0189 0202 0246 0272 0273
R12		000C		0142 0179 0180 0185 0225 0232 0250 0322
R13		000D		0178 0228 0278
R14		000E		0180 0185 0225 0232 0322
R2		0002		0118 0120 0121 0122 0123 0124 0125 0126 0177 0227 0262 0262 0280 0280
R3		0003		0115 0117 0118 0155 0166 0247 0291 0297 0308 0309 0310
R4		0004		0116 0119 0126 0127 0154 0167 0260 0298
R5		0005		0102 0106 0120 0128 0160 0248 0269
R6		0006		0248 0251 0259 0269 0271
R7		0007		0159 0160 0161 0166 0167 0186 0234 0247 0258 0260 0270 0271 0275 0308
R8		0008		0119 0127 0310
RETN		01FA'	0321	0332
RETURN		01EE'	0317	0311
SIZE		00EE'	0153	0146

LABEL		VALUE	DEFN	REFERENCES								
SLT	R	00B0'	0029	0121								
TOPR		017C'	0245	0182	0229							
TOPR10		017E'	0247	0279								
TYP0	R	0154'	0028	0216								
TYPE	R	020C'	0028	0101	0134	0174	0176	0224	0287	0300	0307	0319
				0321	0330							
VDT	R	00A2'	0029	0116								
VERIFY		0158'	0222	0281								
VERPAS		000A	0039	0228								
VNT	R	00B8'	0029	0123								
WAITL		01AC'	0273	0274								
WAITP		01A8'	0272	0276								
YORN		0134'	0201	0103	0135							
YORN1		0136'	0203	0204								
YORN2		0142'	0208	0205								
YORN3		0150'	0214	0210								
YORN4		0152'	0215	0212								

*

```
SOURCE ACCESS NAME=      DBASIC.REL0.EXTENDED.SOURCE.FORMAT
OBJECT ACCESS NAME=      DBASIC.REL0.EXTENDED.OBJECT.FORMAT
LISTING ACCESS NAME=     DBASIC.REL0.EXTENDED.LIST.FORMAT
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'FORMAT'
0002               *
0003               *  EXTERNAL DEFINITIONS AND REFERENCES
0004               *
0005                    DEF  CVBF
0006                    REF  CVBFR
```

```
0008                    *
0009                    *PRINT FORMATTING
0010                    *
0011 0000 C209  CVBF    MOV R9,R8       MARK FORMAT
0012 0002 C187          MOV R7,R6       MARK CHARACTER BUFFER
0013 0004 04C3          CLR R3          CLEAR DECIMAL FLAG
0014 0006 04C4          CLR R4          CLEAR BEFORE DECIMAL COUNT
0015 0008 04C5          CLR R5          CLEAR TOTAL DIGIT HOLDER COUNT
0016                    *
0017 000A 06A0  CVBF1   BL @CVBFT       GET TYPE
     000C 00F0'
0018 000E 100A            JMP CVBF4     NULL, DONE
0019 0010 1002            JMP CVBF2     DIGIT HOLDER
0020 0012 0703            SETO R3       DECIMAL
0021 0014 1004          JMP CVBF3       CHARACTER
0022                    *
0023 0016 C0C3  CVBF2   MOV R3,R3       BEFORE DECIMAL?
0024 0018 1601            JNE $+4       N
0025 001A 0584          INC R4          Y, COUNT
0026 001C 0585          INC R5
0027                    *
0028 001E DDA0  CVBF3   MOVB @B2A,*R6+  OUT "*
     0020 0062'
0029 0022 10F3          JMP CVBF1
0030                    *
0031 0024 022A  CVBF4   AI R10,-13      GET # OF LEADING DIGITS
     0026 FFF3
0032 0028 050A          NEG R10
0033 002A 810A          C R10,R4        GREATER THAN DIGIT HOLDERS?
0034 002C 151B            JGT CVBFR1    Y, ERROR RETURN
0035 002E 6144          S R4,R5         GET NUMBER OF DIGITS PAST DECIMAL
0036 0030 A14A          A R10,R5        INDEX TO ROUNDING DIGIT
0037 0032 0285          CI R5,12        TOO LARGE?
     0034 000C
0038 0036 1102            JLT CVBF5     N
0039 0038 0205          LI R5,12        Y, SET TO 12
     003A 000C
0040                    *
0041 003C C005  CVBF5   MOV R5,R0       SET TO ROUND
0042 003E 0600          DEC R0
0043 0040 06A0          BL @CVBFR       ROUND
     0042 0000
0044 0044 058A            INC R10       ADD NEW DIGIT
0045 0046 C209          MOV R9,R8       RESET FOR CVBFT
0046 0048 04C5          CLR R5          CLR # FLAG
0047                    *
0048 004A 06A0  CVBF6   BL @CVBFT       GET FORMAT TYPE
     004C 00F0'
0049 004E 100B  CVBD7A    JMP CVBFR2    DONE
0050 0050 100D            JMP CVBF8     DIGIT HOLDER
0051 0052 102D            JMP CVBF15    ".
0052 0054 0280          CI R0,>2C00     ",?
     0056 2C00
0053 0058 1602            JNE CVBF7     N
0054 005A C145          MOV R5,R5       Y, DIGIT OUT?
0055 005C 1321            JEQ CVBF11    N, OUT SPACE
0056 005E DDC0  CVBF7   MOVB R0,*R7+    Y, OUT CHARACTER
0057 0060 10F4          JMP CVBF6
0058 0062   2A  B2A     BYTE >2A
0059 0063   30  B30     BYTE >30
```

```
0061 0064 C1C6  CVBFR1 MOV  R6,R7        FORMATTING OVERFLOW
0062 0066 CB47  CVBFR2 MOV  R7,@14(13)   RETURN UPDATE PTR
     0068 000E
0063 006A 0380         RTWP              DONE
0064                   *
0065 006C 0604  CVBF8   DEC R4           DIGIT HOLDER
0066 006E 8284          C R4,R10         TIME FOR DIGIT?
0067 0070 1118            JLT CVBF13     Y
0068 0072 1302            JEQ CVBF8A     N, CHECK FLOATER
0069 0074 C104          MOV R4,R4        FLOATER?
0070 0076 1611            JNE CVBF10     N
0071 0078 0280  CVBF8A  CI R0,>2400       N, MAYBE FLOATER THOUGH, "$?
     007A 2400
0072 007C 13F0            JEQ CVBF7      Y, INSERT
0073 007E 0280          CI R0,>5300      "$?
     0080 5300
0074 0082 1605            JNE CVBF9      N
0075 0084 C30C          MOV R12,R12      Y, POSITIVE?
0076 0086 130C            JEQ CVBF11     Y, INSERT BLANK
0077 0088 0200          LI R0,>2D00      N, INSERT "-
     008A 2D00
0078 008C 10E8          JMP CVBF7
0079                   *
0080 008E 0280  CVBF9   CI R0,>3C00       "<?
     0090 3C00
0081 0092 1603            JNE CVBF10     N
0082 0094 C30C          MOV R12,R12      Y, POSITIVE?
0083 0096 16E3            JNE CVBF7      N, INSERT "<
0084 0098 1003          JMP CVBF11       Y, INSERT BLANK
0085                   *
0086 009A 0280  CVBF10  CI R0,>3000       "0?
     009C 3000
0087 009E 1302            JEQ CVBF12     Y
0088 00A0 0200  CVBF11  LI R0,>2000       N, INSERT BLANK
     00A2 2000
0089 00A4 DDC0  CVBF12  MOVB R0,*R7+     INSERT
0090 00A6 10D1          JMP CVBF6
0091                   *
0092 00A8 06A0  CVBF13  BL @CVBF30
     00AA 00DE'
0093 00AC 10CE          JMP CVBF6
0094                   *
0095 00AE DDC0  CVBF15  MOVB R0,*R7+     INSERT ".
0096                   *
0097 00B0 06A0  CVBF16  BL @CVBFT        GET BYTE
     00B2 00F0'
0098 00B4 10CC            JMP CVBD7A     DONE
0099 00B6 100A            JMP CVBF20     DIGIT HOLDER
0100 00B8 1000            JMP $+2
0101 00BA 0280          CI R0,>3E00      ">?
     00BC 3E00
0102 00BE 1604            JNE CVBF18     N
0103 00C0 C30C          MOV R12,R12      Y, POSITIVE?
0104 00C2 1602            JNE CVBF18     N
0105 00C4 0200          LI R0,>2000      Y, OUT SPACE
     00C6 2000
0106 00C8 DDC0  CVBF18  MOVB R0,*R7+     OUT
0107 00CA 10F2          JMP CVBF16
0108                   *
0109 00CC 0604  CVBF20  DEC R4           COUNT
```

```
0110 00CE 8284           C R4,R10          TIME FOR DIGIT?
0111 00D0 1103             JLT CVBF21      Y
0112 00D2 0200           LI R0,>3000       N, OUT "0
     00D4 3000
0113 00D6 10F8           JMP CVBF18
0114              *
0115 00D8 06A0  CVBF21   BL @CVBF30        OUT DIGIT
     00DA 00DE'
0116 00DC 10E9           JMP CVBF16
0117              *
0118 00DE 060A  CVBF30   DEC R10           COUNT DOWN
0119 00E0 0705           SETO R5           ALLOW COMMA'S
0120 00E2 DDF3           MOVB *R3+,*R7+    INSERT DIGIT
0121 00E4 1604             JNE CVBF31
0122 00E6 0607           DEC R7            NO DIGIT
0123 00E8 0603           DEC R3
0124 00EA DDE0           MOVB @B30,*R7+    INSERT "0
     00EC 0063'
0125 00EE 045B  CVBF31   RT
```

```
0127                    *CHECK FORMAT TYPE
0128                    *       BL @CVBFT
0129                    *         NULL
0130                    *         DIGIT HOLDER  < $ S 0 9
0131                    *         DECIMAL
0132                    *       CHAR ,
0133                    *
0134 00F0 04C0  CVBFT   CLR R0          GET BYTE
0135 00F2 D038          MOVB *R8+,R0
0136 00F4 1323            JEQ CVBFT3    NULL
0137 00F6 0280          CI R0,>3C00     "<?
     00F8 3C00
0138 00FA 131F            JEQ CVBFT2    Y
0139 00FC 0280          CI R0,>2400     "$?
     00FE 2400
0140 0100 131C            JEQ CVBFT2    Y
0141 0102 0280          CI R0,>5300     "S?
     0104 5300
0142 0106 1319            JEQ CVBFT2    Y
0143 0108 0280          CI R0,>3000     "0?
     010A 3000
0144 010C 1316            JEQ CVBFT2    Y
0145 010E 0280          CI R0,>3900     "9?
     0110 3900
0146 0112 1313            JEQ CVBFT2    Y
0147      0116' B2E     EQU $+2
0148 0114 0280          CI R0,>2E00     ".?
     0116 2E00
0149 0118 130F            JEQ CVBFT1    Y
0150 011A 0280          CI R0,>4500     "E?
     011C 4500
0151 011E 1606            JNE CVBFT4    N
0152 0120 0200          LI R0,>2000     Y, DEFAULT TO SPACE
     0122 2000
0153 0124 C30C          MOV R12,R12     NEGATIVE?
0154 0126 1302            JEQ $+6       N
0155 0128 0200          LI R0,>2D00     Y, OUT "-
     012A 2D00
0156                    *
0157 012C 0280  CVBFT4  CI R0,>5E00     "↑?
     012E 5E00
0158 0130 1602            JNE CVBFT0    N
0159 0132 0200          LI R0,>2E00     Y, REPLACE WITH ".
     0134 2E00
0160                    *
0161 0136 05CB  CVBFT0  INCT R11
0162 0138 05CB  CVBFT1  INCT R11
0163 013A 05CB  CVBFT2  INCT R11
0164 013C 045B  CVBFT3  RT
0165                    END
NO ERRORS
```

$	013E'		0024 0100 0147 0154
B2A	0062'	0058	0028
B2E	0116'	0147	
B30	0063'	0059	0124
CVBD7A	004E'	0049	0098
CVBF D	0000'	0011	0005
CVBF1	000A'	0017	0029
CVBF10	009A'	0086	0070 0081
CVBF11	00A0'	0088	0055 0076 0084
CVBF12	00A4'	0089	0087
CVBF13	00A8'	0092	0067
CVBF15	00AE'	0095	0051
CVBF16	00B0'	0097	0107 0116
CVBF18	00C8'	0106	0102 0104 0113
CVBF2	0016'	0023	0019
CVBF20	00CC'	0109	0099
CVBF21	00D8'	0115	0111
CVBF3	001E'	0028	0021
CVBF30	00DE'	0118	0092 0115
CVBF31	00EE'	0125	0121
CVBF4	0024'	0031	0018
CVBF5	003C'	0041	0038
CVBF6	004A'	0048	0057 0090 0093
CVBF7	005E'	0056	0053 0072 0078 0083
CVBF8	006C'	0065	0050
CVBF8A	0078'	0071	0068
CVBF9	008E'	0080	0074
CVBFR R	0042'	0006	0043
CVBFR1	0064'	0061	0034
CVBFR2	0066'	0062	0049
CVBFT	00F0'	0134	0017 0048 0097
CVBFT0	0136'	0161	0158
CVBFT1	0138'	0162	0149
CVBFT2	013A'	0163	0138 0140 0142 0144 0146
CVBFT3	013C'	0164	0136
CVBFT4	012C'	0157	0151
R0	0000		0041 0042 0052 0056 0071 0073 0077 0080 0086 0088 0089 0095 0101 0105 0106 0112 0134 0135 0137 0139 0141 0143 0145 0148 0150 0152 0155 0157 0159
R10	000A		0031 0032 0033 0036 0044 0066 0110 0118
R11	000B		0161 0162 0163
R12	000C		0075 0075 0082 0082 0103 0103 0153 0153
R3	0003		0013 0020 0023 0023 0120 0123
R4	0004		0014 0025 0033 0035 0065 0066 0069 0069 0109 0110
R5	0005		0015 0026 0035 0036 0037 0039 0041 0046 0054 0054 0119
R6	0006		0012 0028 0061
R7	0007		0012 0056 0061 0062 0089 0095 0106 0120 0122 0124
R8	0008		0011 0045 0135
R9	0009		0011 0045

*

```
SOURCE ACCESS NAME=      DBASIC.REL0.EXTENDED.SOURCE.ERRORLST
OBJECT ACCESS NAME=      DBASIC.REL0.EXTENDED.OBJECT.ERRORLST
LISTING ACCESS NAME=     DBASIC.REL0.EXTENDED.LIST.ERRORLST
ERROR ACCESS NAME=       DBASIC.ERR
OPTIONS=                 XREF
MACRO LIBRARY PATHNAME=
```

```
0001                        IDT  'ERRORLST'
0002                        OPTION TUNLST
0003                  *
0004                  *   THIS MODULE OUTPUTS THE ERROR MESSAGE IN PLACE OF
0005                  *   THE ERROR NUMBER AS TYPICALLY OUTPUT.
0006                  *
0007                        DEF  ERRLS1,ERRLS2
0008                        REF  TYPE
0009                  *
0010        0000' CRLF1 EQU  $
0011  0000    0A          BYTE >0A,>0D,>2A,>2A,>2A,0
      0001    0D
      0002    2A
      0003    2A
      0004    2A
      0005    00
0012        0006' ASTER EQU  $
0013  0006    2A          BYTE >2A,>2A,>2A,0
      0007    2A
      0008    2A
      0009    00
0014        000A' CRLF2 EQU  $
0015  000A    2A          BYTE >2A,>2A,>2A,>0A,>0D,0
      000B    2A
      000C    2A
      000D    0A
      000E    0D
      000F    00
```

```
0017       0010' ERRLS1 EQU  $
0018 0010 C28B          MOV  R11,R10              SAVE RETURN
0019 0012 C1D9          MOV  *R9,R7               GET BUFFER ADDRESS
0020 0014 0202          LI   R2,CRLF1             GET "CR/LF***"
     0016 0000'
0021 0018 06A0          BL   @MOVE                MOVE INTO BUFFER
     001A 0030'
0022 001C 0A11          SLA  R1,1                 MULTIPLY BY 2 TO GET WORD INDX
0023 001E C0A1          MOV  @ERRTBL-2(R1),R2     GET ERROR ADRESS FROM TABLE
     0020 006A'
0024 0022 06A0          BL   @MOVE                MOVE INTO BUFFER
     0024 0030'
0025 0026 0202          LI   R2,ASTER             GET "***"
     0028 0006'
0026 002A 06A0          BL   @MOVE                MOVE INTO BUFFER
     002C 0030'
0027 002E 045A          B    *R10                 AND RETURN
0028       0030' MOVE   EQU  $
0029 0030 DDF2          MOVB *R2+,*R7+            MOVE CHARACTER
0030 0032 16FE          JNE  MOVE                 NULL TERMINATES STRING
0031 0034 0607          DEC  R7                   BACKUP BUFFER POINTER
0032 0036 0458          B    *R11                 RETURN
```

```
0034       0038' ERRLS2 EQU  $
0035 0038 C28B          MOV  R11,R10            SAVE RETURN
0036 003A 0201          LI   R1,CRLF1           GET "CR/LF***"
     003C 0000'
0037 003E 06A0          BL   @TYPE              OUTPUT STRING R1
     0040 0000
0038 0042 C003          MOV  R3,R0              GET ERROR NUMBER(IN ASCII)
0039 0044 06C0          SWPB R0                 RJ TENS DIGIT
0040 0046 0240          ANDI R0,>F              MASK DIGIT
     0048 000F
0041 004A 3820          MPY  @D10,R0            CONVERT
     004C 006A'
0042 004E C2C3          MOV  R3,R11             COPY ERROR NUMBER
0043 0050 024B          ANDI R11,>F             MASK DIGIT
     0052 000F
0044 0054 A04B          A    R11,R1             ADD TO RESULT
0045 0056 0A11          SLA  R1,1               MULTIPLY BY 2 TO GET WORD INDX
0046 0058 C061          MOV  @ERRTBL-2(R1),R1   GET ERROR ADDRESS FROM TABLE
     005A 006A'
0047 005C 06A0          BL   @TYPE              OUTPUT STRING R1
     005E 0040'
0048 0060 0201          LI   R1,CRLF2           GET "***CR/LF"
     0062 000A'
0049 0064 06A0          BL   @TYPE              OUTPUT STRING R1
     0066 005E'
0050 0068 045A          B    *R10               AND RETURN
0051              *
0052 006A 000A  D10     DATA 10                 CONSTANT 10
```

```
0054        006C' ERRTBL EQU  $
0055 006C 00C8'          DATA ERR01
0056 006E 00D6'          DATA ERR02
0057 0070 00EC'          DATA ERR03
0058 0072 0100'          DATA ERR04
0059 0074 0116'          DATA ERR05
0060 0076 012A'          DATA ERR06
0061 0078 013C'          DATA ERR07
0062 007A 0150'          DATA ERR08
0063 007C 0166'          DATA ERR09
0064 007E 0180'          DATA ERR10
0065 0080 0192'          DATA ERR11
0066 0082 01A2'          DATA ERR12
0067 0084 01B2'          DATA ERR13
0068 0086 01C6'          DATA ERR14
0069 0088 01E0'          DATA ERR15
0070 008A 01F8'          DATA ERR16
0071 008C 0218'          DATA ERR17
0072 008E 0230'          DATA ERR18
0073 0090 0244'          DATA ERR19
0074 0092 0258'          DATA ERR20
0075 0094 0272'          DATA ERR21
0076 0096 0286'          DATA ERR22
0077 0098 029A'          DATA ERR23
0078 009A 02AC'          DATA ERR24
0079 009C 02CE'          DATA ERR25
0080 009E 02E8'          DATA ERR26
0081 00A0 0304'          DATA ERR27
0082 00A2 031C'          DATA ERR28
0083 00A4 032E'          DATA ERR29
0084 00A6 0346'          DATA ERR30
0085 00A8 0350'          DATA ERR31
0086 00AA 035E'          DATA ERR32
0087 00AC 036C'          DATA ERR33
0088 00AE 038E'          DATA ERR34
0089 00B0 03A2'          DATA ERR35
0090 00B2 03B2'          DATA ERR36
0091 00B4 03CE'          DATA ERR37
0092 00B6 03E0'          DATA ERR38
0093 00B8 03F4'          DATA ERR39
0094 00BA 040C'          DATA ERR40
0095 00BC 0420'          DATA ERR41
0096 00BE 043E'          DATA ERR42
0097 00C0 0452'          DATA ERR43
0098 00C2 0464'          DATA ERR44
0099 00C4 0474'          DATA ERR45
0100 00C6 0487'          DATA ERR46
```

```
0102        00C8' ERR01   EQU  $
0103 00C8     53          TEXT 'SYNTAX ERROR'
0104 00D4     00          BYTE 0
0105 00D6                 EVEN
0106        00D6' ERR02   EQU  $
0107 00D6     55          TEXT 'UNMATCHED PARENTHESIS'
0108 00EB     00          BYTE 0
0109 00EC                 EVEN
0110        00EC' ERR03   EQU  $
0111 00EC     49          TEXT 'INVALID LINE NUMBER'
0112 00FF     00          BYTE 0
0113 0100                 EVEN
0114        0100' ERR04   EQU  $
0115 0100     49          TEXT 'ILLEGAL VARIABLE NAME'
0116 0115     00          BYTE 0
0117 0116                 EVEN
0118        0116' ERR05   EQU  $
0119 0116     54          TEXT 'TOO MANY VARIABLES'
0120 0128     00          BYTE 0
0121 012A                 EVEN
0122        012A' ERR06   EQU  $
0123 012A     49          TEXT 'ILLEGAL CHARACTER'
0124 013B     00          BYTE 0
0125 013C                 EVEN
0126        013C' ERR07   EQU  $
0127 013C     45          TEXT 'EXPECTING OPERATOR'
0128 014E     00          BYTE 0
0129 0150                 EVEN
0130        0150' ERR08   EQU  $
0131 0150     49          TEXT 'ILLEGAL FUNCTION NAME'
0132 0165     00          BYTE 0
0133 0166                 EVEN
0134        0166' ERR09   EQU  $
0135 0166     49          TEXT 'ILLEGAL FUNCTION ARGUMENT'
0136 017F     00          BYTE 0
0137 0180                 EVEN
0138        0180' ERR10   EQU  $
0139 0180     53          TEXT 'STORAGE OVERFLOW'
0140 0190     00          BYTE 0
0141 0192                 EVEN
0142        0192' ERR11   EQU  $
0143 0192     53          TEXT 'STACK OVERFLOW'
0144 01A0     00          BYTE 0
0145 01A2                 EVEN
0146        01A2' ERR12   EQU  $
0147 01A2     53          TEXT 'STACK UNDERFLOW'
0148 01B1     00          BYTE 0
0149 01B2                 EVEN
0150        01B2' ERR13   EQU  $
0151 01B2     4E          TEXT 'NO SUCH LINE NUMBER'
0152 01C5     00          BYTE 0
0153 01C6                 EVEN
0154        01C6' ERR14   EQU  $
0155 01C6     45          TEXT 'EXPECTING STRING VARIABLE'
0156 01DF     00          BYTE 0
0157 01E0                 EVEN
0158        01E0' ERR15   EQU  $
0159 01E0     49          TEXT 'INVALID SCREEN COMMAND'
0160 01F6     00          BYTE 0
0161 01F8                 EVEN
```

insert two spaces!

```
0162       01F8' ERR16  EQU  $
0163 01F8   45          TEXT 'EXPECTING DIMENSIONED VARIABLE'
0164 0216   00          BYTE 0
0165 0218               EVEN
0166       0218' ERR17  EQU  $
0167 0218   53          TEXT 'SUBSCRIPT OUT OF RANGE'
0168 022E   00          BYTE 0
0169 0230               EVEN
0170       0230' ERR18  EQU  $
0171 0230   54          TEXT 'TOO FEW SUBSCRIPTS'
0172 0242   00          BYTE 0
0173 0244               EVEN
0174       0244' ERR19  EQU  $
0175 0244   54          TEXT 'TOO MANY SUBSCRIPTS'
0176 0257   00          BYTE 0
0177 0258               EVEN
0178       0258' ERR20  EQU  $
0179 0258   45          TEXT 'EXPECTING SIMPLE VARIABLE'
0180 0271   00          BYTE 0
0181 0272               EVEN
0182       0272' ERR21  EQU  $
0183 0272   44          TEXT 'DIGITS OUT OF RANGE'
0184 0285   00          BYTE 0
0185 0286               EVEN
0186       0286' ERR22  EQU  $
0187 0286   45          TEXT 'EXPECTING VARIABLE'
0188 0298   00          BYTE 0
0189 029A               EVEN
0190       029A' ERR23  EQU  $
0191 029A   52          TEXT 'READ OUT OF DATA'
0192 02AA   00          BYTE 0
0193 02AC               EVEN
0194       02AC' ERR24  EQU  $
0195 02AC   52          TEXT 'READ TYPE DIFFERS FROM DATA TYPE'
0196 02CC   00          BYTE 0
0197 02CE               EVEN
0198       02CE' ERR25  EQU  $
0199 02CE   53          TEXT 'SQUARE OF NEGATIVE NUMBER'
0200 02E7   00          BYTE 0
0201 02E8               EVEN
0202       02E8' ERR26  EQU  $
0203 02E8   4C          TEXT 'LOG OF NON-POSITIVE NUMBER'
0204 0302   00          BYTE 0
0205 0304               EVEN
0206       0304' ERR27  EQU  $
0207 0304   45          TEXT 'EXPRESSION TOO COMPLEX'
0208 031A   00          BYTE 0
0209 031C               EVEN
0210       031C' ERR28  EQU  $
0211 031C   44          TEXT 'DIVISION BY ZERO'
0212 032C   00          BYTE 0
0213 032E               EVEN
0214       032E' ERR29  EQU  $
0215 032E   46          TEXT 'FLOATING POINT OVERFLOW'
0216 0345   00          BYTE 0
0217 0346               EVEN
0218       0346' ERR30  EQU  $
0219 0346   46          TEXT 'FIX ERROR'
0220 034F   00          BYTE 0
0221 0350               EVEN
```

```
0222        0350' ERR31   EQU  $
0223 0350    46           TEXT 'FOR W/O NEXT'
0224 035C    00           BYTE 0
0225 035E                 EVEN
0226        035E' ERR32   EQU  $
0227 035E    4E           TEXT 'NEXT W/O FOR'
0228 036A    00           BYTE 0
0229 036C                 EVEN
0230        036C' ERR33   EQU  $
0231 036C    45           TEXT 'EXP FUNCTION HAS INVALID ARGUMENT'
0232 038D    00           BYTE 0
0233 038E                 EVEN
0234        038E' ERR34   EQU  $
0235 038E    55           TEXT 'UNNORMALIZED NUMBER'
0236 03A1    00           BYTE 0
0237 03A2                 EVEN
0238        03A2' ERR35   EQU  $
0239 03A2    50           TEXT 'PARAMETER ERROR'
0240 03B1    00           BYTE 0
0241 03B2                 EVEN
0242        03B2' ERR36   EQU  $
0243 03B2    4D           TEXT 'MISSING ASSIGNMENT OPERATOR'
0244 03CD    00           BYTE 0
0245 03CE                 EVEN
0246        03CE' ERR37   EQU  $
0247 03CE    49           TEXT 'ILLEGAL DELIMITER'
0248 03DF    00           BYTE 0
0249 03E0                 EVEN
0250        03E0' ERR38   EQU  $
0251 03E0    55           TEXT 'UNDEFINED FUNCTION'
0252 03F2    00           BYTE 0
0253 03F4                 EVEN
0254        03F4' ERR39   EQU  $
0255 03F4    55           TEXT 'UNDIMENSIONED VARIABLE'
0256 040A    00           BYTE 0
0257 040C                 EVEN
0258        040C' ERR40   EQU  $
0259 040C    55           TEXT 'UNDEFINED VARIABLE'
0260 041E    00           BYTE 0
0261 0420                 EVEN
0262        0420' ERR41   EQU  $
0263 0420    45           TEXT 'EXPANSION EPROM NOT INSTALLED'
0264 043D    00           BYTE 0
0265 043E                 EVEN
0266        043E' ERR42   EQU  $
0267 043E    49           TEXT 'INTERRUPT W/O TRAP'
0268 0450    00           BYTE 0
0269 0452                 EVEN
0270        0452' ERR43   EQU  $
0271 0452    49           TEXT 'INVALID BAUD RATE'
0272 0463    00           BYTE 0
0273 0464                 EVEN
0274        0464' ERR44   EQU  $
0275 0464    54           TEXT 'TAPE READ ERROR'
0276 0473    00           BYTE 0
0277 0474                 EVEN
0278        0474' ERR45   EQU  $
0279 0474    45           TEXT 'EPROM VERIFY ERROR'
0280 0486    00           BYTE 0
0281        0487' ERR46   EQU $
```

```
 0282 0487   49          TEXT 'INVALID DEVICE NUMBER'
 0283 049C   00          BYTE 0
 0284                    END
NO ERRORS
```

LABEL	VALUE	DEFN	REFERENCES
$	049D'		0010 0012 0014 0017 0028 0034 0054 0102 0106 0110 0114 0118 0122 0126 0130 0134 0138 0142 0146 0150 0154 0158 0162 0166 0170 0174 0178 0182 0186 0190 0194 0198 0202 0206 0210 0214 0218 0222 0226 0230 0234 0238 0242 0246 0250 0254 0258 0262 0266 0270 0274 0278 0281
ASTER	0006'	0012	0025
CRLF1	0000'	0010	0020 0036
CRLF2	000A'	0014	0048
D10	006A'	0052	0041
ERR01	00C8'	0102	0055
ERR02	00D6'	0106	0056
ERR03	00EC'	0110	0057
ERR04	0100'	0114	0058
ERR05	0116'	0118	0059
ERR06	012A'	0122	0060
ERR07	013C'	0126	0061
ERR08	0150'	0130	0062
ERR09	0166'	0134	0063
ERR10	0180'	0138	0064
ERR11	0192'	0142	0065
ERR12	01A2'	0146	0066
ERR13	01B2'	0150	0067
ERR14	01C6'	0154	0068
ERR15	01E0'	0158	0069
ERR16	01F8'	0162	0070
ERR17	0218'	0166	0071
ERR18	0230'	0170	0072
ERR19	0244'	0174	0073
ERR20	0258'	0178	0074
ERR21	0272'	0182	0075
ERR22	0286'	0186	0076
ERR23	029A'	0190	0077
ERR24	02AC'	0194	0078
ERR25	02CE'	0198	0079
ERR26	02E8'	0202	0080
ERR27	0304'	0206	0081
ERR28	031C'	0210	0082
ERR29	032E'	0214	0083
ERR30	0346'	0218	0084
ERR31	0350'	0222	0085
ERR32	035E'	0226	0086
ERR33	036C'	0230	0087
ERR34	038E'	0234	0088
ERR35	03A2'	0238	0089
ERR36	03B2'	0242	0090
ERR37	03CE'	0246	0091
ERR38	03E0'	0250	0092
ERR39	03F4'	0254	0093
ERR40	040C'	0258	0094
ERR41	0420'	0262	0095
ERR42	043E'	0266	0096
ERR43	0452'	0270	0097
ERR44	0464'	0274	0098
ERR45	0474'	0278	0099
ERR46	0487'	0281	0100
ERRLS1 D	0010'	0017	0007
ERRLS2 D	0038'	0034	0007
ERRTBL	006C'	0054	0023 0046
MOVE	0030'	0028	0021 0024 0026 0030

LABEL		VALUE	DEFN	REFERENCES							
R0		0000		0038	0039	0040	0041				
R1		0001		0022	0023	0036	0044	0045	0046	0046	0048
R10		000A		0018	0027	0035	0050				
R11		000B		0018	0032	0035	0042	0043	0044		
R2		0002		0020	0023	0025	0029				
R3		0003		0038	0042						
R7		0007		0019	0029	0031					
R9		0009		0019							
TYPE	R	0066'	0008	0037	0047	0049					

*

```
SOURCE ACCESS NAME=     BD1.DBASIC.REL2.EXTENDED.SOURCE.ATNFE
OBJECT ACCESS NAME=     BD1.DBASIC.REL2.EXTENDED.OBJECT.ATNFE
LISTING ACCESS NAME=    BD1.DBASIC.REL2.EXTENDED.LIST.ATNFE
ERROR ACCESS NAME=
OPTIONS=                XREF
MACRO LIBRARY PATHNAME=
```

```
0001                    IDT  'ATNFE'
0002                *
0003                * REVISION: 12/01/78 1.01      ;SOURCE STANDARDIZATION
0004                *           08/09/78 1.00      ;INITIAL RECEIPT BY TI
0005                *           30 MAY 79          ;UPDATE AND COPY FOR REL2
0006                * ROUTINE LIST:
0007                *
0008                *       ATNF              ;ARC-TANGENT FUNCTION
0009                *
0010                * COPY: (NONE)
0011                *
0012                * MARCO: (NONE)
0013                *
0014                * EXTERNAL ROUTINES:
0015                *
0016                        REF FUNFX         ;FIX ARGUMENT
0017                        REF PLYX,PLYXX    ;EVALUATE POLYMONIAL
0018                        REF EVSFR         ;EXIT TO EVALUATOR
0019                *
0020                        DXOP LOADF,0      ;LOAD FPAC
0021                        DXOP STORE,1      ;STORE FPAC
0022                        DXOP FADD,2       ;ADD TO FPAC
0023                        DXOP FSUB,3       ;SUBTRACT FROM FPAC
0024                        DXOP FMUL,4       ;MULTIPLY FPAC
0025                        DXOP FDIV,5       ;DIVIDE FPAC
0026                        DXOP SCALE,6      ;SCALE FPAC
0027                        DXOP NORMAL,7     ;NORMALIZE FPAC
0028                        DXOP CLEAR,8      ;CLEAR FPAC
0029                        DXOP NEGATE,9     ;NEGATE FPAC
0030                        DXOP FLOATF,10    ;FLOAT FPAC
0031                *
0032                * EXTERNAL DATA:
0033                *
0034                        REF FPAC          ;FLOATING POINT ACCUMULATOR
0035                        REF TEMP          ;TEMP REGISTER
0036                        REF DS2,DS3       ;DATA STORAGE
0037                *
0038                * MODULE EQUATES:  (NONE)
0039                *
0040                * MODULE VARIABLES AND CONSTANTS:  (NONE)
```

```
0043            * ABSTRACT:
0044            *
0045            *       COMPUTE ARC-TANGENT OF ARGUMENT (R2).
0046            *       RESULT IS IN RADIANS.
0047            *
0048            * CALLING SEQUENCE:
0049            *
0050            *       B @ATNF
0051            *
0052            *       EXIT TO EVSFR
0053            *
0054            *
0055            * EXCEPTIONS AND CONDITIONS:
0056            *
0057            *       FLOATING POINT ERRORS
0058            *
0059            * EXTERNAL ROUTINE LIST:
0060            *
0061            *      (SEE EXTERNAL ROUTINES)
0062            *
0063            * LOCAL DATA:
0064            *
0065            *      (SEE MODULE VARIABLES AND CONSTANTS)
```

```
0067                    * ENTRY POINT:
0068                    *
0069                          DEF  ATNFE
0070                    *
0071          0000'     ATNFE EQU  $
0072 0000 06A0                BL   @FUNFX       ;FIX
     0002 0000
0073 0004 1041                 JMP  ATNF5       ;0
0074                    *
0075 0006 04C1          ATNF0  CLR R1
0076 0008 D052                 MOVB *R2,R1
0077 000A 04C4                 CLR R4
0078 000C C0A0                 MOV @FPAC,R2
     000E 0000
0079 0010 60A0                 S @ATNC6,R2       ;<1?
     0012 00A0'
0080 0014 1107                   JLT ATNF1       ;Y
0081 0016 2C60                 STORE @TEMP       ;MOVE TO TEMP
     0018 0000
0082 001A 2C20                 LOADF @ATNC6      ;LOAD 1
     001C 00A0'
0083 001E 2D60                 FDIV @TEMP        ;GET INVERSE
     0020 0018'
0084 0022 0584                 INC R4            ;SET FLAG
0085                    *
0086 0024 2C60          ATNF1  STORE @TEMP       ;MOVE TO TEMP
     0026 0020'
0087 0028 2CE0                 FSUB @ATNC0       ;F-C0
     002A 0092'
0088 002C C0A0                 MOV @FPAC,R2      ;GET SIGN
     002E 000E'
0089 0030 2C20                 LOADF @TEMP       ;MOVE TEMP TO FPAC
     0032 0026'
0090 0034 C082                 MOV R2,R2         ;>.268...?
0091 0036 110C                   JLT ATNF2       ;N
0092 0038 2CA0                 FADD @ATNC1       ;Y
     003A 0098'
0093 003C 2C60                 STORE @DS2        ;SAVE IN DS
     003E 0000
0094 0040 2C20                 LOADF @TEMP       ;MOVE TEMP TO FPAC
     0042 0032'
0095 0044 2D20                 FMUL @ATNC1
     0046 0098'
0096 0048 2CE0                 FSUB @ATNC6       ;SUBTRACT 1
     004A 00A0'
0097 004C 2D60                 FDIV @DS2         ;F=F/DS
     004E 003E'
0098                    *
0099 0050 2C60          ATNF2  STORE @DS2        ;STORE IN DS
     0052 004E'
0100 0054 06A0                 BL @PLYXX         ;EVALUATE
     0056 0000
0101 0058 009E'                  DATA ATNC2
0102 005A 2C60                 STORE @DS3        ;SAVE IN DS1
     005C 0000
0103 005E 06A0                 BL @PLYX          ;EVALUATE
     0060 0000
0104 0062 00BE'                  DATA ATNC3
0105 0064 2D60                 FDIV @DS3         ;DIVIDE BY DS1
     0066 005C'
```

```
0106 0068 2D20          FMUL @DS2       ;MULTIPLY BY DS
     006A 0052'
0107 006C C082          MOV R2,R2       ;WAS IT >.268...?
0108 006E 1102            JLT ATNF3     ;N
0109 0070 2CA0          FADD @ATNC4     ;Y, ADD CONSTANT
     0072 00D8'
0110                *
0111 0074 0604  ATNF3   DEC R4
0112 0076 1602            JNE ATNF4
0113 0078 2CA0          FADD @ATNC5
     007A 00DE'
0114                *
0115 007C 0B13  ATNF4   SRC R3,1        ;GET SIGN BIT
0116 007E 4820          SZC @C8000,@FPAC
     0080 0090'
     0082 002E'
0117 0084 A803          A R3,@FPAC      ;ADD SIGN
     0086 0082'
0118 0088 0202  ATNF5   LI R2,FPAC      ;GET VALUE
     008A 0086'
0119 008C 0460          B @EVSFR        ;AND RETURN
     008E 0000
0120                *
0121 0090 8000  C8000   DATA >8000
0122 0092 4044  ATNC0   DATA >4044,>9851,>7A7B
     0094 9851
     0096 7A7B
0123 0098 411B  ATNC1   DATA >411B,>B67A,>E858
     009A B67A
     009C E858
0124 009E 0004  ATNC2   DATA 4
0125 00A0 4110  ATNC6   DATA >4110,>0000,>0000
     00A2 0000
     00A4 0000
0126 00A6 4225          DATA >4225,>10EB,>4200
     00A8 10EB
     00AA 4200
0127 00AC 42CF          DATA >42CF,>B153,>9710
     00AE B153
     00B0 9710
0128 00B2 4316          DATA >4316,>CA99,>3433
     00B4 CA99
     00B6 3433
0129 00B8 42C5          DATA >42C5,>33FE,>142D
     00BA 33FE
     00BC 142D
0130 00BE 0003  ATNC3   DATA 3
0131 00C0 41C9          DATA >41C9,>8867,>F42A
     00C2 8867
     00C4 F42A
0132 00C6 427D          DATA >427D,>9444,>406E
     00C8 9444
     00CA 406E
0133 00CC 4312          DATA >4312,>AED9,>3E72
     00CE AED9
     00D0 3E72
0134 00D2 42C5          DATA >42C5,>33FE,>142D
     00D4 33FE
     00D6 142D
0135 00D8 4086  ATNC4   DATA >4086,>0A91,>C16C
```

```
      00DA 0A91
      00DC C16C
 0136 00DE C119  ATNC5   DATA >C119,>21FB,>5444
      00E0 21FB
      00E2 5444
 0137                    END
NO ERRORS,      NO WARNINGS
```

$		00E4'		0071							
ATNC0		0092'	0122	0087							
ATNC1		0098'	0123	0092	0095						
ATNC2		009E'	0124	0101							
ATNC3		00BE'	0130	0104							
ATNC4		00D8'	0135	0109							
ATNC5		00DE'	0136	0113							
ATNC6		00A0'	0125	0079	0082	0096					
ATNF0		0006'	0075								
ATNF1		0024'	0086	0080							
ATNF2		0050'	0099	0091							
ATNF3		0074'	0111	0108							
ATNF4		007C'	0115	0112							
ATNF5		0088'	0118	0073							
ATNFE	D	0000'	0071	0069							
C8000		0090'	0121	0116							
CLEAR			0028								
DS2	R	006A'	0036	0093	0097	0099	0106				
DS3	R	0066'	0036	0102	0105						
EVSFR	R	008E'	0018	0119							
FADD			0022								
FDIV			0025								
FLOATF			0030								
FMUL			0024								
FPAC	R	008A'	0034	0078	0088	0116	0117	0118			
FSUB			0023								
FUNFX	R	0002'	0016	0072							
LOADF			0020								
NEGATE			0029								
NORMAL			0027								
PLYX	R	0060'	0017	0103							
PLYXX	R	0056'	0017	0100							
R1		0001		0075	0076						
R2		0002		0076	0078	0079	0088	0090	0090	0107	0107	0118
R3		0003		0115	0117						
R4		0004		0077	0084	0111					
SCALE			0026								
STORE			0021								
TEMP	R	0042'	0035	0081	0083	0086	0089	0094			

*