

TRENT
TITLE
TITLE

TLINE
TTY LINE COLLECTOR
XTEXT STUFF

LIST X
OPNAMES XTTEXT
TITLE MACROS

*
*
*
X J MACRO LOC
+ VFD 12/713/R,18/LOC,30/1
SP7 #+1
JP FRR
ENDM

*
*
X JR MACRO LOC,RTNAUTH
+ VFD 12/713/R,18/LOC,12/1,18/2
VFD 68/RTNAUTH
SP7 #+1
JP FRR
ENDM

*
*
*
*
X JF MACRO LOC,FGO
+ VFD 12/713/R,18/LOC,30/1
JP FGO
ENDM

*
*
CALL MACRO LOC
+ SP7 #-1
JP LOC
ENDM

*
*
MCAP MACRO NAME
+ BSS @
NAME EOJ #-
VFD 1/1,20/C,NAME,30/CX,MAST
ENDM

*
*
MXCAP MACRO NAME
+ VFD 1/1,20/C,NAME,30/CX,MAST
ENDM

*
*
ITEMS MACRO A,B,C,D,E,F,G,H
+ VFD 68/A
TEC NE,\$R\$
VFD 68/R
TEC NE,\$FC\$
VFD 68/C
TEC NE,\$DN\$
VFD 68/D
TEC NE,\$FP\$

* MAP FF.LINESF,0,0,0,SCRSZ,0
MAP FF.LINECD,1,SCRSZ,SCRSZ,FL,1

* DATA =6

* LOC 0

* CX.CODEC PARCAP FF.LINECD
CX.TTYFI PARCAP TTYFILE
PARCAP TTYREQ
PARCAP TTRESP
CX.MAST PARCAP TMASTER
CX.READ PARCAP READ
CX.READ PARCAP I.READ
CX.WRITE PARCAP I.WRITE
CX.SENDS PARCAP I.SENDS
CX.GETS PARCAP I.GETS
CX.BDCS PARCAP CO.READDR
CX.LINCL PARCAP CC.LINE

* CLSTSZ RSS 0

* DATA =6

* LOC #0

EJECT

* PARAMSZ EQU 15

* AREA TO RECIEVE PARAMETERS ON CALL

* ORG1 RSS 0

* ORG2 RSS 6

* PARAM0 RSS 1 FIXED PARAM TO CONTROL MODE

* PARAM1 RSS 1 TYPE OF CALL

* RSS 1 COUNT OF PARAM ?

* PARAM2 RSS PARAMSZ USUALLY CONTAINS SYSTEXT

* ORG2 RSS 0

* TEST ORG1+ORG2

ORG ORG1

ENDIF

IFLE ORG1+ORG2

ORG ORG2

ENDIF

EJECT

* GENERAL DATA AREA

TTYBUFF RSSZ 160 (USED BY GRAYCODE)

*
*
REGAREG RSSZ 208 (HOLDS USERS REGISTERS DURING CALL)

*
*
*
FAKEDESC ITEMS PARAM2+1
FAKEDESC RSSZ 1

*
*
*
* DATA ERROR DURING ERROR CALL HANDLING

*
* EREGS RSSZ 208 REGISTER SAVE AREA DURING ERROR CALL

*
* STKBUF RSSZ 3 TOP OF STACK DURING ERROR CALL
EJECT

*
*
* END OF SCRATCH AREA

*
* SCRSZ RSS 0

*
*
*
*
* FIXED IP LISTS

*
* RDSELFI MXCAP READ
ITEMS CX.CODEF
ITEMS 6+0+REGAREA

*
* RDSELFI MXCAP READ
ITEMS CX.CODEF
ITEMS REGAREA+20B,REGAREA+20B,SCRSZ=REGAREA=203

*
*
* RETURNDO MXCAP RETURN

*
*
* SRETURN MXCAP RETRAN
VFD 34A72.B4/PARAM2
VFD 367A.B0/0
TITLE MAIN CODE

*
*
* THIS IS MY INTERFACE TO JIM GRAYS GRAYCODE

*
*
* THE CALLS ON GRAYCODE WERE CONSTRUCTED
* TO MIMIC THE OLD ASCII XTEXT FILE PROVIDED
* BY JIM GRAY

*
*
* THE EDIT CALL WAS MODELED ON CODE IN THE OLD BEAD

RESZ 5 ALTERNATE ENTRY POINTS

* JP ERRC ERROR CALL ENTRY POINT

* MAIN XJ SAVE
XJ SETMASK
SA1 PARAM₁ CHECK HIDDEN PARAM
SP1 XJ
SR2 SWTCH1CN
GF B1,B2,FAIL
LT B1,B0,FAIL
JP SWTCH1+B1

SWTCH1 JP INITIAL INITIALIZING CALL FROM BUILDER
JP ASCIIFM ASCII INTERFACE CALL

* BSS 0
SWTCH1CN EOU #-SWTCH1

* * * FAIL CALL ERR
EJECT

* * * INITIALIZING CALL FROM BUILDER

* * * INITIAL XJ RDSELF1
XJ RDSELF2
SP1 TTYUFF
SR2 CX.TTYFL
SY RETURN #+? JP .TTY.ON
XJ RETURNUP

* * * * THIS CODE USED FOR EXIT WITH NO RETURNED PARAMS

* * * FINIT XJ RESTORE
XJ RETURNUP
EJECT

* * * ASCII INTERFACE CALL

* ASCIIFM SA1 PARAM₁
SR1 XJ
SR2 SWTCH2CN
GF B1,B2,E.2.1
LT B1,B0,E.2.0
JP SWTCH2+B1

* SWTCH2 JP FINIT 0 NO OP
JP TYPELIN 1 INPUT A LINE
JP TYPELOUT 2 OUTPUT A LINE

* JP TYPEOUT 3
JP TTYOUT 4 OUTPUT A CHARACTER
EDIT A LINE

*
SWTCH2CR RSS 0
EQU *-SWTCH2

*
*
E.2.0 XJ E.2.0.X

*
E.2.1 XJ E.2.1.X

*
*
E.2.0.A MXCAP USRER
MFD 64/2
MFD 42/T.18/0

*
E.2.1.A MXCAP USRER
MFD 64/2
MFD 42/T.18/1
EJECT

*
*
* THIS CODE HANDLES GETTING A LINE FROM TTY

*
TYPELIN SX6 B6
S6 S6 PARAMP
S6 S6 PARAMPSZ-2

*
TYPELT#1 S6 A6+7 ZERO OUT BUFFER
S6 S6 B6-1
S6 S6 B6+BD, TYPELIN1

*
S6 S6 PARAMP1 MAKE JIM GREY STORE LINE IN MY BUFFER
S6 S6 TTYBUFF+NEW
S6 S6 TTYBUFF
S6 RETURN #+1
EN MORE

*
S6 S6 TTYBUFF+NEW+1 GET COUNT
S6 XJ XJ
S6 S6 PARAMP

*
XJ RESTORE
XJ SOFTURN
EJECT

*
* THIS CODE HANDLES SENDING A LINE TO TTY

*
TYPELOUT SP.TTY TTYBUFF
S6 S6 PARAMP
S6 XJ
S6 S6 FAKEDESC
SA.LENGTH A6
S6 RETURN FTNT1

EQ PPUT
EJECT

THIS CODE SENDS A SINGLE CHAR TO TTY

TYPEOUT SAI PARAM²
RX-CHAR X1
SR-TTY TTYBUFF
SR-RETURN FTNT1
FO PPUTCTTYT
EJECT

THIS CODE ADAPTED FROM THE OLD EDIT CODE
(PROBABLY WRITTEN BY JIM GRAY)

THE FOLLOWING COMMENTS ARE QUOTED FROM THE
HEADER OF THE EDIT CODE IN THE OLD READ

THE LINE IS 1ST CHECKED FOR VALIDITY
IT IS TRUNCATED AT 1ST NON GRAPHIC
CHARACTER OR AT 88 CHARACTERS, WHICH EVER
COMES FIRST.

UNHAPPILY IT DOSENT DO BLANK COMPRESSION

THE FOLLOWING COMMENT WAS ALSO THERE, BUT I DONT
KNOW IT'S SIGNIFICANCE

UNLIKE GETLINE, IT UPDATES THE STRING
DESCRIPTOR IN THE CALLERS SPACE SO THAT
IT IS CORRECT.

TTYEDT SAI PARAM²
RX6 X1
SA5 TTYBUFF+OLD+3+3
SP6 1 ..SET R,STEP
SP3 16

SA1 A1+R6
RX6 X1
SA6 A6+R6
SP7 B3-R6
GT B3,+-1
GET READY TO CALL GETC7
SA4 TTYBUFF+OLD+1 SET A.LENGTH
SX4 B6 NOTE THE FINAL PUT OF A CR WILL SET LENGTH
SR3 TOOLONG SET B.ERROR
SP7 #+1 SET B.RRETURN
JP GETC7 GET A CHARACTER
SY1 X1-96 IS IT GRAPHIC
NG X1,GETC7 IF SO, GET ANOTHER
LINE TOO LONG OR NON GRAPHIC ENCOUNTERED
TOOLONG SX4 X4-1 STEP BACK ONE

SX1 CR PASS A CR
 SP7 #+1 RETURN BELOW
 JP PUTC7 PUR THE CR AND SET LENGTH
 *
 JP TYPELTH
 TITLE ERROR CALL HANDLING
 *
 * ERROR CALL AREA
 *
 *
 E_PANIC EQU 12
 *
 ERRCODE EQU 6
 ERRNUM EQU ERRCODE+1
 *
 *
 ERRC X,I ESAVE SAVE REGISTERS
 X,I SETEMSK RE SET ERROR SELECTION MASK
 SAI ERRCODE CHECK ERROR CLASS FOR PANIC
 SX1 X1=E_PANIC
 ZQ X1,TNTEPIUT
 X,I RDSTK READ TOP OF STACK (IN CASE UNWANTED ERROR)
 X,I ERESTOR RESTORE REGISTERS
 X,I FRTN NOW CONVERT ERROR TO E RETURN
 *
 *
 ESAVE MXCAP SAVE
 ITEMS FREGS
 *
 *
 ERESTOR MXCAP RESTOR
 ITEMS EREGS
 *
 *
 SETEMSK MXCAP ESMILDC
 ITEMS #+1
 DATA =6
 *
 *
 RDSTK MXCAP DISSEN
 ITEMS STKAUF
 DATA 1
 *
 *
 FRTN MXCAP FRETUR
 *
 *
 * WE HAVE BEEN INTERRUPTED
 *
 *
 INTERRUPT X,I FIXPC SETS P COUNTER TO INTERRUPT CODE
 Y,I ERESTOR
 X,I RETURNOP RETURN TO INTERRUPTED VERSION
 *

```

* INTCODE SAI ERRNUM
  ZP X1,INTCODE1 MILD PANIC
  XJ BEADS MAJOR PANIC
*
* INTCODE1 XJ INTERR MILD PANIC
*
*
* FIXPC MXCAP MDPDC
ITEMS CX,RDOS
ITEMS 1
ITEMS INTCODE
*
*
INTERR MXCAP USRER ( MILD PANIC )
ITEMS E_PANTC
ITEMS 0
*
*
BEADS ITEMS CX,RDOS ( MAJOR PANIC )
TITLE MISG SUBROUTINES
*
*
ERR XJ SAVE
SR6 4
XJF READ+STOP
JP DIE
*
*
DEBUG XJ SAVE
SR6 4
XJF READ+STOP
XJ RESTORE
JP RT
*
*
RESTORE MXCAP RESTOP
ITEMS REGAREA
*
*
BEAD ITEMS CX,READ
*
SAVE MXCAP SAVE
ITEMS REGAREA
*
*
DIE XJ RETHENUP
*
*
STOP PS
EJECT
TITLE GrayCode
*
```

*
* WHAT FOLLOWS IS COPIED VIA THE EDITOR FROM
* GRAYCODE, XTEXT ON OCT 22, 1970
*
*

* FIRST WE FIX UP XJ FOR GRAYCODE
*
*

XJ MACRO LOC
+ WFO 1277130B,18/LOC
ENDM

* NOW WE FIX UP COLUMNS FOR NOMPASS
*
*

	COL	36
	EJECT	
PRINT	EQU	2
FUNY	EQU	0
LOOK	EQU	0
TTY	EQU	1
CHAR	EQU	1
ERROR	EQU	3
COUNT	EQU	4
LENGTH	EQU	4
SCRATCH	EQU	5
STEP	EQU	6
OUT	EQU	6
RETURN	EQU	7
BULEN	EQU	17
INLENGTH	EQU	8
OUTLEN	EQU	8
RS		
DIFF	EQU	#0-#1
BASEPT	EQU	#0
RS		
OUTFIRST	RS	1
OUTGET	RS	1
OUTLTM	RS	1
INFIRST	RS	1
INPUT	RS	1
INLIM	RS	1
OUTPUT	RS	1
INSET	RS	1
RS	OUTLEN	
RS	INLENGTH	
SOFTECH	RS	1
NEEDEDM	RS	1
FORCE	RS	1
SENDMENT	RS	2
ACTUAL	RS	1
HANGIN	RS	2
RINPUT	RS	5
WRITETR	RS	5
RTNBUF	RS	5

WRITEOUT	ESS	3
OUTBASE	ESS	1
OUTCOUNT	ESS	1
WOUTPUT	ESS	5
ROUTGET	ESS	5
BREAKTAB	ESS	3
	ESS	1
ICFLAG	ESS	1
TABS	ESS	2
OLD	ESS	3
NEW	ESS	3
	IF	=ABS.BASEPT,1
USE		*
	IF	ARS.BASEPT,1
	ORG	BASEPT
	LOC	#0-0FFF
MAX	EOH	87
BLANK	EOU	8
SKIPCHAR	EOH	4B
ESCCCHAR	EOH	6A
LEFTTAB	EOH	34B
RIGHTTAB	EOH	34B
BACKCHAR	EOH	77B
BELLCHAR	EOH	147B
LF	EOH	142B
CR	EOH	155B
ESCC	EOH	173B
ESCC	EOH	173B

PP AND CP SIGNALS

ONE	EOH	3
READ	EOH	CX.READ . MODIFIED
WRITE	EOH	CX.WRITE . FROM
SEND	EOH	CX.SENDE . GRAYCODE
HANG	EOH	CX.GETE . ABSOLUTE

***** FILE CAPABILITY INDEX COMES IN IN R2

***** THE INDEX OF THE CP TO PP EVENT CHANNEL IS B2+1

***** THE INDEX OF THE PP TO CP CHANNEL IS B2+2

MOREOUT	EOH	16000B	MORE OUTPUT
MOREIN	EOH	26000B	MORE INPUT
PURGE	EOH	26000B	
RESUME	EOH	36000B	RESUME HARD ECHO
TABLE0	EOH	46000B	NO BREAK
TABLE1	EOH	46001B	ALL BREAK
TABLE2	EOH	46002B	BREAK ON NON GRAPHIC
TABLE3	EOH	46003B	BREAK ON NON ALPHA MERIC
TABLE4	EOH	46004B	BREAK ON NON NUMERIC
MOREROOM	EOH	66000B	MORE ROOM IN INPUT BUFFER

SYSCALL	MACRO	ARG
*	S8,SCRATCH	B,TTY+ARG
*	AJ	-SCRATCH
*	VFN	30/1-60/0
	ENDM	SYSCALL
SENDVENT	MACRO	PARAM
	Sy.0UT	PARAM

SA_OUT
SYSCALL
SENDM
SENDNT
EJECT

B_ITY+ACTHAL
SENDVENT
SENDVENT

*
*
* SIGNALS THE CP SENDS TO THE PP

EOM SET 4001B END OF WORD AND OF MESSAGE
SET OUTPUT ACTIVE OFF IF PO
EDN SET 4002B END OF WORD, NEXT CHARACTER
IS IN NEXT WORD

*
*
* SIGNALS THE PP SENDS THE CP

DEFERRED SET 4003B ECHOS DEFERRED BEYOND THIS
NULLC SET 4004B NULL CHARACTER
LOST SET 4005B LOST SOME CHARACTERS HERE
WENTIN SET 4006B REENTERED HARD ECHO ON REQU
WENTOUT SET 4007B WENT OUT OF ECHO MODE ON RE
PURGEN SET 4010B PURGED CM INPUT BUFFER HERE

TITLE INITIALIZER
FD 60/400200100040020010008

•TTY_ONI BX_ONE
SX_OUT BX_ONE
SA_OUT B1+SOFTECHO
SX_OUT B6
SA_OUT B1+OUTCOUNT INITIALIZE OUT COUNT
SA_OUT B1+NEEDEOM
SA_OUT B1+BREAKTR
SA_OUT B1+ICFLAG
SX_OUT B_ONE
SA_OUT B1+FORCE
SA_OUT B1+RINPUT+4
SA_OUT B1+MPRITEIN+4
SA_OUT B1+ROUTGET+2
SA_OUT B1+ROUTGET+4
SA_OUT B1+OUTBUFR1+4
SA_OUT B1+OLD+1
SA_OUT B1+OLD+2
SA_SCRATCH .TTY_ON=1
SX_OUT X_SCRATCH
SA_OUT B1+TARS
SA_OUT A_OUT+B_ONE
SX_OUT B1+OLD+3+3
SA_OUT B1+OLD
SX_7 X7
LX_7 40
SA_7 X6
SX_7 X6
SX_OUT B1+NEW+3+11
SA_OUT B1+NEW
SX_OUT B6
SA_OUT A7-B_ONE
SX_OUT 36
BX_OUT -X_OUT

~~S_A.OUT~~ B₁*BREAKTAB+1
~~S_X.OUT~~ B₂
~~S_A.OUT~~ S_A.OUT+B₀.ONE
~~S_X.OUT~~ READ
~~S_A.OUT~~ B₁*RINPUT
~~S_A.OUT~~ B₁*RINRUF
~~S_A.OUT~~ B₁*ROUTGET
~~S_X.OUT~~ WRITE
~~S_A.OUT~~ B₁*WRITEIN
~~S_A.OUT~~ B₁*WRITEOUT
~~S_A.OUT~~ B₁*WOUTPUT
~~S_X.OUT~~ B₂
~~S_A.OUT~~ B₁*RINPUT+1
~~S_A.OUT~~ A₀.OUT+5
~~S_A.OUT~~ A₀.OUT+5
~~S_A.OUT~~ A₀.OUT+5
~~S_A.OUT~~ A₀.OUT+5
~~S_X.OUT~~ INPUT
~~S_A.OUT~~ B₁*RTNPUT+2
~~S_X.OUT~~ X₀.OUT+B₁
~~S_A.OUT~~ A₀.OUT+B₀.ONE
~~S_A.OUT~~ B₀.OUT+5
~~S_X.OUT~~ INGET
~~S_A.OUT~~ B₁*WRITEIN+2
~~S_X.OUT~~ Z₀.OUT+LEN
~~S_A.OUT~~ B₁*RINRUF+2
~~S_X.OUT~~ X₀.OUT+B₁
~~S_A.OUT~~ A₀.OUT+B₀.ONE
~~S_X.OUT~~ INLENGTH
~~S_A.OUT~~ A₀.OUT+1
~~S_X.OUT~~ OUTPUT
~~S_A.OUT~~ B₁*WOUTPUT+2
~~S_X.OUT~~ X₀.OUT+B₁
~~S_A.OUT~~ A₀.OUT+B₀.ONE
~~S_X.OUT~~ OUTGET
~~S_A.OUT~~ B₁*ROUTGET+2
~~S_X.OUT~~ B₁*X₀.OUT
~~S_A.OUT~~ A₀.OUT+B₀.ONE
~~S_X.OUT~~ SEND
~~S_A.OUT~~ B₁*SENDVENT
~~S_X.OUT~~ B₂+1
~~S_A.OUT~~ A₀.OUT+B₀.ONE
~~S_X.OUT~~ HANG
~~S_A.OUT~~ B₁*HANGIN
~~S_X.OUT~~ B₂+2
~~S_A.OUT~~ A₀.OUT+B₀.ONE
~~S_X.OUT~~ READ
~~S_A.OUT~~ B₁
~~S_X.OUT~~ B₂
~~S_A.OUT~~ B₁+1
~~S_X.OUT~~ B₃
~~S_A.OUT~~ B₁+2
~~S_X.OUT~~ B₁
~~S_A.OUT~~ B₁+3
~~S_X.OUT~~ B₂

SA.OUT	B1+4		
AJ	-1		
FD	35/1,60/0		
SENVENT PURGE	TURN TTY ON		
SENVENT	TABLE?		
SYSCALL	WRITEIN		
SA.SCRATCH	B.TTY+INPUT	THROW AWAY JUNK IN INPUT BUFFER	
RX.OUT	X.SCRATCH		
SA.OUT	R.TTY+INGET		
SENVENT	MOREROOM		
SENVENT	RESUME		
SR.RETURN	*+1		
JR	FLUSH3		
SA.SCRATCH	B.TTY+OUTPUT		
X.OUT	66		
SA.OUT	X.SCRATCH+B.TTY		
SX.OUT	X.SCRATCH		
SA.OUT	B.TTY+OUTBASE=1		
SX.OUT	X.OUT+B.TTY		
SA.OUT	B.TTY+OUTBASE		
JR	POP	RETURN	
TITLE		STACK MANITULATORS	
EJECT			
PUSH	ACRO	ARG	PUSH ARG INT CONTROL STACK
LX.RETURN	18		
SX.SCRATCH	B.RETURN		
SX.RETURN	X.RETURN+X.SCRATCH		
SR.RETURN	ARG		
ENDM	PUSH		
POP	SR.RETURN	X.RETURN	
	AX.RETURN	18	
EXIT	JR	B.RETURN	RETURN
FAULT	TITLE	B.ERROR	
*			PUTS A 7 BIT CHARACTER ON THE END OF A STRI
*			CONCATENATES THE 7 BIT CHARACTER FOUNG
*			CONCATENATES THE 7 BIT CHARACTER IN X.CHAR
*			STRING STARTING AT C(A.LENGTH-1) AT THE X.L
*			POSITION (COUNTING FROM ZERO)
*			THE DESCRIPTOR IS UPDATED IN THE PROCESS TO
*			TO THE CONCATENATED CHARACTER
*			RETURNS THRU B.RETURN
*			UPDATES STRING DESCRIPTOR, X.LENGTH,B.COUNT
*			CLOBBERS X.SCRATCH,B.SCRATCH,A.SCRATCH,A.OU
PUTC7	Sx.LENGTH	X.LENGTH+1	
SX.OUT	X.LENGTH		
SA.OUT	A.LENGTH		
SX.SCRATCH	X.LENGTH-1		
SX.OUT	7		
DX.OUT	X.OUT*X.SCRATCH		
AX.SCRATCH	3		
SR.SCRATCH	X.SCRATCH		
SA.SCRATCH	A.LENGTH-1		
SA.SCRATCH	X.SCRATCH+B.SCRATCH		
SR.SCRATCH	X.OUT+49		
LX.OUT	3		
DX.OUT	-X.OUT		

```

SR•SCRATCH B•SCRATCH+X•OUT
SX•OUT 177B
LX•OUT X•OUT+R•SCRATCH
SX•OUT -X•OUT+X•SCRATCH
LX•SCRATCH X•CHAR+B•SCRATCH
SX•OUT X•OUT+X•SCRATCH
SA•OUT A•SCRATCH
JP B•RETURN
TITLE GET A 7 BIT CHARACTER FROM A STRING
EJECT

```

* GETS THE X.LENGTH=1 CHARACTER FROM THE STRI
 * STARTING AT C(A.LENGTH-1) AND RETURNS IT RI
 * ADJUSTED WITH LEADING ZEROS IN X.CHAR
 * INCREMENTS B.COUNT AND X.LENGTH BY R.STEP
 * RETURNS THRU B.RETURN
 * IF X.LENGTH IS GREATER THAN MAX OR LESS THA
 * THEN RETURN IS THRU B.ERROR AND NO CHARACTE
 * IS RETURNED

* UPDATES - X.CHAR X.LENGTHB.COUNT
 * DESTROYS X•SCRATCH,A•SCRATCH,B•SCRATCHA•OUT

```

GETC7, SR•COUNT R•COUNT+B•STEP
SX•LENGTH X•LENGTH+B•STEP
SX•CHAR X•LENGTH-1
JP GETC71

```

```

GETC7 SX•LENGTH X•LENGTH+B•STEP
SR•COUNT R•COUNT+B•STEP
SX•CHAR X•LENGTH-1
G X•CHAR.FAULT
SX•SCRATCH X•LENGTH-MAX
PL X•SCRATCH.FAULT

```

```

GETC71 SA•SCRATCH A•LENGTH-1
SX•OUT 7
SX•OUT X•CHAR#X•OUT
SX•CHAR ?
SR•SCRATCH X•CHAR
SA•SCRATCH X•SCRATCH+B•SCRATCH
SR•SCRATCH X•OUT-11
SR•SCRATCH -R•SCRATCH
LY•OUT ?
SR•SCRATCH X•OUT+B•SCRATCH
LX•SCRATCH X•SCRATCH,B•SCRATCH
SX•CHAR 177B
EX•CHAR X•CHAR#X•SCRATCH
JP B•RETURN

```

TITLE PUT A CHARACTER INTO THE TTY BUFFER
 EJECT

X.CHAR THE R BIT RIGHT ADJUSTED CHARACTER TO BE
 B.TTY A POINTER TO THE BASE OF THE TTY BUFFER

* PUT HANGS WHEN THE BUFFER IS FULL
 * PUT MAKES UP THE PPU IF THE BUFFER WAS FORMERLY EMPTY
 * ALL THIS IS ACTUALLY DONE BY FLUSH WHICH IS AN INTEGR
 * OF PUT
 * FOUR OUT OF FIVE TIMES PUT TAKES TEN MICROSECONDS
 * UNLESS FLUSHING IS REQUIRED PUT TAKES 33 MICROSECONDS
 * PUT THE LAST CHARACTER OF EACH BUFFER WORD

* IF FLUSHING IS REQUIRED IT MAY TAKE FOR EVER

* PUT DESTROYS X SCRATCH X OUT
A SCRATCH A OUT

* THE ONLY BIT PATTERN THAT CANNOT BE SENT IS 77778
INITIALIZATION AND FLUSHING MUST FORCE THE IMAGE OF T
PUT POINTER TO -027777777777777777777777777777778

* PUTCTTY RETURNS THRU B RETURN

PUTCTTY	X OUT	53	TRANSLATE FROM INTERNAL TO
	SA OUT	R TTY+NEEDEOM	
	SX SCRATCH	X CHAR+40B	
	BX OUT	=X OUT*X SCRATCH	
	CX SCRATCH	X OUT	CONSTRUCT EVEN PARITY
	LX SCRATCH	59	
*	PI	X SCRATCH+*1	
	SX OUT	X OUT+200B	
	CX SCRATCH	4P	
	RX OUT	X OUT*X SCRATCH	
	JO	PUTCTTYI	
PUTCTTY	HX OUT	28	
	RA OUT	R TTY+NEEDEOM	
	RX OUT	X CHAR+X OUT	
PUTCTTY	SA SCRATCH	R TTY+OUTPUT	PICK UP THE BUFFER WORD
	SA SCRATCH	X SCRATCH+R TTY	
	LX SCRATCH	12	MAKE ROOM FOR NEXT CHARACTER
	RX OUT	X SCRATCH*X OUT	
	SA OUT	A SCRATCH	
	LX OUT	7	RESTORE THE BUFFER WORD
	NG	X OUT, EXIT	IF THERE ARE MORE SLOTS RET
	CA SCRATCH	R TTY+OUTCOUNT	
	SX OUT	X SCRATCH+1	
	SA OUT	A SCRATCH	
	CA SCRATCH	R TTY+OUTPUT	
	SX OUT	X SCRATCH+1	
	SA SCRATCH	R TTY+OUTLIM	
	LX SCRATCH	X OUT-X SCRATCH	
	PC	X SCRATCH, SKIPA	STEP OUTCOUNT FOR SWAPPING
	SYSCALL	WRITEOUT	
	SX OUT	0	
	SA OUT	R TTY+OUTCOUNT	
	SA SCRATCH	R TTY+OUTFIRST	
	SX OUT	X SCRATCH+R TTY	
	SA OUT	R TTY+OUTBASE	
	RX OUT	X SCRATCH	
	SA OUT	R TTY+OUTBASE=1	
SKIPA	SA OUT	R TTY+OUTPUT	
	SA OUT	X OUT+R TTY	
	MX OUT	63	
	SA OUT	A OUT	
	SA SCRATCH	R TTY+OUTPUT	
	SX OUT	X SCRATCH+1	
	SA SCRATCH	R TTY+OUTLIM	
	LX SCRATCH	X OUT-X SCRATCH	
	NG	X SCRATCH, SKIPB	
	SA SCRATCH	R TTY+OUTFIRST	

```

SKIPE    BX. OUT      Y. SCRATCH
        SA. SCRATCH   R. TTY+OUTGET
        TX. SCRATCH   X. SCRATCH=X. OUT
        NZ          Y. SCRATCH, EXIT
        JP          FLUSH?
TITLE    FLUSH THE OUTPUT BUFFER
EJECT   CLEAR OUT THE OUTPUT BUFFER
*       ENTERED WITH R.TTY POINTING TO THE BASE OF A VIRTUAL
*       TTY BUFFER, RETURNS THRU R.RETURN
*       DESTROYS X.SCRATCH,A.SCRATCH,B.SCRATCH,A.OUT,X.OUT,X.
FLUSH   Sy. CHAR     EOM
        PUSH      FLUSH1
        JP          PUTTTY
FLUSHI  SA. SCRATCH   R. TTY+OUTPUT
        SA. SCRATCH   X. SCRATCH+R.TTY
        NZ          X. SCRATCH,PUTTTY
        Sy. OUT      Bp
        SA. OUT      R. TTY+NEEDEOM
        Sa. RETURN   X. RETURN
        SX. RETURN   TR
        EOM      *
*       WRITE USING OUTCOUNT
*       SEND MORE OUTPUT EVENT
READ_OUTGET
*       FULL YES BLOCK THEN READ
*       NO
*       RESET SHAP COUNT
*       RETURN
        SYSCALL     WRITEOUT
        SYSCALL     WOUTPUT
SENDVENT
        Sy. OUT      R. OUT
        SA. OUT      R. TTY+OUTCOUNT
        Sa. SCRATCH   R. TTY+OUTPUT
        Sy. OUT      X. SCRATCH
        SA. OUT      R. TTY+OUTBASE-1
        Sy. OUT      X. SCRATCH+B.TTY
        Sa. OUT      R. TTY+OUTBASE
        SA. SCRATCH   R. TTY+OUTPUT
        Sy. OUT      X. SCRATCH+1
        Sa. SCRATCH   R. TTY+OUTGET
        Ix. OUT      X. OUT-X. SCRATCH
        Zq          X. OUT,FLUSH3
        Sy. OUT      X. OUT-OUTLEN
        NZ          X. OUT, EXIT
FLUSH3  SYSCALL     ROUTGET
        SA. SCRATCH   R. TTY+OUTPUT
        BX. OUT      X. SCRATCH+1
        SA. SCRATCH   R. TTY+OUTLIM
        IX. SCRATCH   X. OUT-X. SCRATCH
        NZ          X. SCRATCH,**1
        SX. OUT      X. OUT-OUTLEN
        SA. SCRATCH   R. TTY+OUTGET
        IX. SCRATCH   X. OUT-X. SCRATCH
        Zq          X. SCRATCH,FLUSH4

```

SX.SCRATCH X.SCRATCH-OUTLEN
 17 X.SCRATCH, EXIT
 FLUSH4 RX.SCRATCH X.RETURN
 SYSCALL HANGIN
 RX.RETURN X.SCRATCH
 JP FLUSH3
 TITLE GET A CHARACTER FROM THE TTY
 EJECT
 * X.CHAR THE 12 BIT CHARACTER RIGHT ADJUSTED WITH
 * LEADING ZEROS TO BE GOTTEN
 * R.TTY A POINTER TO THE BASE OF THE VIRTUAL TTY BU
 * X.SCRATCH WORKING REGISTERS
 * X.OUT
 * IF THE VIRTUAL BUFFER IS EMPTY INFORCER IS CALLED WHO
 * TRIES TO FORCE THE PP BACK INTO HARD ECHO. THEN THE VI
 * TTY BUFFER IS REFRESHED FROM THE ACTUAL TTY BUFFER BY
 * REFRESH HANGS UNTIL THE ACTUAL BUFFER IS NON EMPTY AND
 * FLUSHES THE ACTUAL BUFFER INTO THE VIRTUAL BUFFER
 * GET DOES NOT INTERRUPT ANY OF THE CHARACTERS
 *
 * GET DESTROYS A.CHAR A.SCRATCH A.OUT
 * X.CHAR X.SCRATCH X.OUT
 *
 GETCTTY SA.CHAR R.TTY+INGET SHOULD I CALL INFORCER AND
 SA.SCRATCH R.TTY+INPUT
 TX.SCRATCH X.SCRATCH=X.CHAR
 17 X.SCRATCH,GETCTTY1
 * FORCE THE PP BACK INTO HARD ECHO
 * FLUSH OUTPUT
 * READ INPUT
 * AND RETURN THRU R.RETURN
 *
 PUSH INFORC?
 SA.SCRATCH R.TTY+NEEDEOM
 17 X.SCRATCH,FLUSH
 INFORC? SR.RETURN X.RETURN
 RX.RETURN 18
 READIN SYSCALL R.INPUT GET A NON-EMPTY INPUT
 SA.SCRATCH R.TTY+INGET
 RX.OUT X.SCRATCH
 SA.SCRATCH R.TTY+INPUT
 TX.SCRATCH X.SCRATCH=X.OUT
 17 X.SCRATCH,READIN?
 SA.SCRATCH R.TTY+SOFTECHO
 17 X.SCRATCH,READIN?
 SENDVENT RESUME
 READINI RX.SCRATCH X.RETURN
 SYSCALL HANGIN
 RX.RETURN X.SCRATCH
 JP READIN
 READINP SYSCALL R.INRUF
 SYSCALL WRITEIN
 SENDVENT MDRFROM
 GETCTTY7 SA.CHAR R.TTY+INGET
 SA.SCRATCH X.CHAR+B.TTY
 LX.SCRATCH 12
 SX.OUT 7777B
 PICK UP THE ACTIVE WORD IF
 SHIFT CHARACTER INTO LOWER
 12 BITS AND SET UP MASK

RX.CHAR	X.SCRATCH#X.OUT	REPLACE CHARACTER IN ACTIVE
RX.OUT	Y.SCRATCH+X.OUT	7777B
SA.OUT	A.SCRATCH	RESTORE ACTIVE WORD
NZ	Y.OUT,GETOUT	UNLESS THE ACTIVE WORD IS -
PL	X.OUT,GETOUT	TE EMPTY RETURN
SA.SCRATCH	R.TTY+INGET	INCREMENT THE IN GET POINTE
SX.OUT	X.SCRATCH+1	
SA.SCRATCH	R.TTY+INLIM	
TX.SCRATCH	X.OUT-X.SCRATCH	
NG	Y.SCRATCH,SKIP10	STEP TTY INGET POINTER
SA.SCRATCH	R.TTY+INFIRST	
RX.OUT	Y.SCRATCH	
SKIP1	SA.OUT	NOW THE POINTER IS UPDATED
GETOUT	RSSZ	
SP.SCRATCH	X.CHAR=EOW	
GT	P.SCRATCH,SIGNAL	INTERPRT SIGNALS
Sy.SCRATCH	177R	
Sy.CHAR	X.CHAR#X.SCRATCH	
JP	FEXIT	

THE SWITCHING TABLE TO HANDLE SIGNALS

SIGNAL	JP	*R.SCRATCH	
	SX.OUT	I	IF FORCED OUT OF ECHO MODE
	JP	SETSOFT	SET SOFT ECHO ON
	JP	GETCTTY	IGNORE NULL CHARACTERS
	SX.OUT	I	IF LOST CHARACTERS TURN ON
	JP	SETSOFT	SOFT ECHO
	SX.OUT	I	IF WENT INTO ECHO MODE ON R
	JP	SETSOFT	TURN OFF SOFTECHO
	Sy.OUT	I	CRYPTIC
	JP	SETSOFT	REMARK
	JP	GETCTTY	IGNORE PURGES
SETSOFT	S1.OUT	SOFTECHO+R.TTY	
	JP	GETCTTY	
	TITLE	GETS A STRING FROM THE TELETYPE	
	EJECT		

GETS COLLECTS CHARACTERS FROM A TELETYPE UP TO AND INCLUDING THE NEXT BREAK CHARACTER AND CONCATENATES THE STRING STARTING AT C(A.LENGTH-1) AND OF LENGTH C(A.LENGTH)

GETS TRY'S TO KEEP THE PP HAPPY. IT READS AND INTERPRTS SIGNALS, ECHOS CHARACTERS WHEN NECESSARY AND WHENEVER POSSABLE IT TRY'S TO FORCE THE PP BACK INTO ECHO MODE.

GET IS ENTERED WITH R.TTY POINTING TO THE BASE OF A V TTY BUFFER. A.LENGTH POINTING TO THE SECOND WORD OF A DESCRIPTOR

IT UPDATES X.LENGTH, AND RETURNS THE LAST CHARACTER IN X.CHAR. IF THE STRING OVERFLOWS (IE X.LENGTH REACHES BEFOR A BREAK IS FOUND GETS RETURNS THE COMPLEMENT OF LAST CHARACTER IN X.CHAR

GETS RETURNS THRU X.RETURN

GETS DESTROYS A.CHAR,X.CHAR,R.SCRATCH,A.SCRATCH,X.SCRATCH,A.OUT

```

*----- R-COUNT AND R-STEP -----
*
*----- ECHO STRATEGY. -----
*----- ECHO ANY NONBREAK CHARACTER WHEN SOFT ECHO IS ON -----
*----- NEVER ECHO A BREAK CHARACTER -----
*----- THE CALLING PROGRAM MAY ECHO A BREAK CHARACTER OR ECH -----
*----- A TRANSLITERATION OF IT. -----
*----- IN ANY CASE BEFOR GETTING A CHARACTER SEE IF THE TTY -----
*----- BUFFER IS EMPTY. IF SO AND IF THE SOFTECHO FLAG IS UP -----
*----- CALL THE TNFORCER. -----
*----- THE TNFORCER DOES THE FOLLOWING THINGS -----
*----- (A) IF NEEDEOM THEN PPUTCTTY(EOM) -----
*----- (B) IF TTYOUT NONEMPTY THEN FLUSH -----
*----- (C) SEND A RESUME ECHO EVENT TO THE PP -----
*----- (D) RETURN -----
*
*----- ENTERED THRU XRETURN WITH ALLENGTH SET -----
GETS    Sb.RETURN      BREAK0
       Sb.COUNT      C
LOOP    Sx.SCRATCH     X.LENGTH=MAX
       RL              X.SCRATCH,GETS1
       Sb.RETURN      BREAK01
       JP              GETCTTY
BREAK01  Xx.OUT        Sa
       Sx.CHAR      X.CHAR+14DB
       Sx.CHAR      =X.OUT#X.CHAR
       Sb.RETURN      BREAK0
       JP              PPUTC7
BREAK0  Sb.SCRATCH     X.CHAR
       Sa.SCRATCH     R.TTY+BREAKTAB=1   TEST FOR BREAK CHARACTER
BREAK1  Sa.SCRATCH     A=SCRATCH+1
       [X.SCRATCH     X.SCRATCH,R.SCRATCH
       Sb.SCRATCH     R.SCRATCH=60
       GE              R.SCRATCH,BREAK1
       MG              X.SCRATCH,POP
       Sb.COUNT      R.COUNT+1
       SA.SCRATCH     R.TTY+SOFTECHO
       PPUTCTTYT      IF SOFTECHO IS ON ECHO IT
       X.SCRATCH,LOOP
       LOOP
       Sb.RETURN      IP
       PPUTCTTYT
GETST   Sx.CHAR      =X.CHAR
       JP              POP
ITLE    PPUT A STRING TO THE TELETYPE
SJECT   PPUT THE STRING WITH DESCRIPTOR C(A.LENGTH=1) C(A.LENGTH) TO THE
       POINTED TO BY B.TTY IF THE STRING ENDS IN A CR THE AD
       LF. IF THE LENGTH OF THE STRING EXCEEDS MAX TRUNCATE
       AT THAT POINT. PPUT FLUSHES THE OUTPUT BUFFER
RETURNS THRU X.RETURN
CLOBBERS A.CHAR,X.CHAR,X.LENGTH,B.SCRATCH,A.SCRATCH,X
R.ERROR,R.COUNT,A.OUT,R.OUT
PUT HANDLES BOTH SQUOZE AND UNSQUOZE TEXT
IF FORCE NON-ZERO THEN LINE IS FLUSHED
PUTL   Sx.FUNY      0
       Sb.COUNT      X.LENGTH
       Sb.COUNT      -R.COUNT

```

	SX.LENGTH	P
	SQ.STEP	I
	SX.CHAR	S
PUTLT	SX.OUT	X.CHAR-CR
	I7	X.OUT,PUTLB
	SX.CHAR	LF
	SQ.RETURN	PUTL4
	JP	PUTCTTYT
PUTL2	SQ.RETURN	PUTL2
	LT	R.COUNT,GETC7.
	JP	PUTL4
PUTL2	I7	X.FINY,SPECIAL
	SX.SCRATCH	X.CHAR-ESCC
	LP	X.SCRATCH,ESCAP
	SQ.RETURN	PUTL1
	JP	PUTCTTYT
PUTL4	SQ.RETURN	X.RETURN
	4X.RETURN	I8
	SA.SCRATCH	B.TTY+FORCE
	I7	X.SCRATCH,FLUSH
	JP	P.RETURN

SPECIAL BX. FUNY D
***** BUTLERS FUNNIES HERE
***** NOW IT ONLY HANDLES BLANK COMPRESSION

SY.FUNY	X.CHAR
SY.CHAR	24JP
SR.RETURN	BLANK WITH PARITY
BLANKS	BLANKS
SY.FUNY	X.FUNY=1
PL	X.FUNY,PUTCTTY
SY.FUNY	QZ
DE	PHTL1
ESCAP	Sy.FUNY
	1
JP	PHTL1

	EFECT	
LINK	SOU	3
MORE.	SY.LINK	X.RETURN
MORE.	SA.SCRATCH	R.TTY+OLD+1
	SY.OUT	X.SCRATCH
	SA.OUT	R.TTY+OLD+2
	SY.OUT	R
	SA.OUT	A.SCRATCH
	SA.OUT	R.TTY+NEW+1
	SY.OUT	1
	SA.OUT	R.TTY+ICFLAG

MORE	BSZ	R
	SR,STEP	I
	SA,LENGTH	B,TTY+NEW+1
	SX,RETURN	BROKEN
	SR,ERROR	BELL
	JP	GETS
BROKEN	NG	X,CHAR,NOROOM
	SX,OUT	X,LENGTH-1
	SA,OUT	A,LENGTH
	SA,SCRATCH	B,TTY+ICFLAG
	NG	X,SCRATCH,BROKE
	SA,SCRATCH	B,TTY+OLD+1

	<i>Sx</i> .OUT	<i>x</i> .SCRATCH+B.COUNT
	<i>SA</i> .OUT	<i>A</i> .SCRATCH
	<i>Sx</i> .PRINT	<i>B</i>
SPKOE	<i>SR</i> .SCRATCH	<i>x</i> .CHAR=140B
	<i>SR</i> .COUNT	<i>B</i>
	<i>SA</i> .LENGTH	<i>B</i> .TTY+OLD+1
	<i>JP</i>	<i>B</i> .SCRATCH+SWITCH+1
NORMON	<i>Sx</i> .OUT	MAX=1
	<i>SA</i> .OUT	<i>A</i> .LENGTH
	<i>JP</i>	RELL
SWITCH	<i>SR</i> .COUNT	1
	<i>JP</i>	COPY
+	<i>Sx</i> .RETURN	SKIP
	<i>JP</i>	TAB
+	<i>Sx</i> .RETURN	SKIP
	<i>JP</i>	RIGHTUNI
+	<i>Sx</i> .RETURN	COPY
	<i>JP</i>	RIGHTUNI
+	<i>SR</i> .COUNT	1
	<i>JP</i>	LEFTUNI
+	<i>Sx</i> .RETURN	COPY
	<i>JP</i>	SCAN
+	<i>Sx</i> .RETURN	COPY
	<i>JP</i>	TAB
+	<i>SR</i> .RETURN	COPY
	<i>JP</i>	REDGE
+	<i>Sx</i> .RETURN	TABB
	<i>JP</i>	TAB
	<i>JP</i>	RELL
+	<i>SA</i> .SCRATCH	TABCHG
	<i>JP</i>	<i>B</i> .TTY+ICFLAG
	<i>SA</i> .SCRATCH	ICSWITCH
	<i>JP</i>	<i>B</i> .TTY+NEW+1
+	<i>SR</i> .RETURN	ACCEPT
	<i>JP</i>	SKIP
+	<i>SR</i> .RETURN	REDGE
	<i>JP</i>	CONCAT
	<i>Sx</i> .PRINT	1
	<i>JP</i>	CONCAT
	<i>SR</i> .COUNT	-1
	<i>JP</i>	BACK
+	<i>JP</i>	LEFTTUT
+	<i>Sx</i> .RETURN	COPY
	<i>JP</i>	WORD
+	<i>Sx</i> .RETURN	BACK
	<i>JP</i>	TABL
	<i>JP</i>	RELL
+	<i>Sx</i> .RETURN	SKIP
	<i>JP</i>	SCAN
+	<i>Sx</i> .RETURN	BACK
	<i>JP</i>	ENDWORD
	<i>Sx</i> .RETURN	SKIP
	<i>JP</i>	WORD
+	<i>JP</i>	REDGE
	<i>SR</i> .COUNT	1
	<i>JP</i>	SKIP

		IP	TYPESTAT	
+ * *		SX_PRINT	3	
	JP	CONCAT		
+ *	JP	BELL	CONTROL SHIFT M	35B
+ *	JP	BELL	CONTROL SHIFT N	36B
+ *	JP	BELL	CONTROL SHIFT O	37B
ACCEPT1	SA_SCRATCH	R_TTY+NEW+1		
ACCEPT	Sy_OUT	X_SCRATCH+B_STEP		
	Sa_OUT	A_SCRATCH		
	JP	FINE		
CONCAT	Sy_PRINT	=X_PRINT		
	Sa_SCRATCH	A_LENGTH+B_STEP		
	Tx_SCRATCH	X_SCRATCH=X_LENGTH		
	SR_COUNT	X_SCRATCH=1		
	SR_ERROR	CONCAT1		
	BT	R_COUNT,COPY		
CONCAT1	SS	H		
	SA_LENGTH	R_TTY+NEW+1		
	SR_RETURN	FINE		
	Sy_CHAR	CR		
	JP	PUTC7		
FINE	Sa_CHAR	R_TTY+NEW+1		
	Sy_OUT	X_CHAR+0		
	Sa_OUT	R_TTY+OLD+1		
	Ax_CHAR	3		
	SR_SCRATCH	X_CHAR		
	Sa_CHAR	R_TTY+NEW		
	SA_LENGTH	R_TTY+OLD		
FINET	SA_SCRATCH	X_CHAR+B_SCRATCH		
	Sy_OUT	X_SCRATCH		
	Sa_OUT	X_LENGTH+B_SCRATCH		
	SR_SCRATCH	R_SCRATCH=B_STEP		
	RE	R_SCRATCH,FINE1		
	SA_LENGTH	NEWLINE+1		
	SX_RETURN	MORE.1		
	SX_PRINT	X_PRINT+?		
	NG	X_PRINT,PUTL		
	Sy_PRINT	A_PRINT=1		
	Sy_RETURN	X_LINK		
	NG	X_PRINT,MORE.		
	FL	X_LINK,PUTL		
	Sy_RETURN	=X_RETURN		
	Sa_SCRATCH	X_RETURN		
	Ax_RETURN	18		
	7	X_RETURN,PUTL		
	SX_RETURN	R_SCRATCH		
	JP	PUTL		
TYRSTAT	SA_LENGTH	NEWLINE+1		
	Sy_RETURN	TYRSTAT1		
	JP	PUTL		
TYRSTAT1	SR_COUNT -120			
	SA_LENGTH	R_TTY+OLD+1		
	SX_CHAR 40B			
	SR_RETURN	BLANKS		
	Sy_RETURN	TYPS1		
	SA_SCRATCH	R_TTY+NEW+1		

Sx.FUNY	X.SCRATCH	
SA.SCRATCH	B.TTY+OLD+2	
Lx.OUT	X.LENGTH-X.SCRATCH	
NG	X.OUT,BLANKS	
Sx.LENGTH	X.SCRATCH=1	
Sx.FUNY	RA	
JP	PUTL1	
TYPST	SA.LENGTH	B.TTY+NEW+1
	Sx.RETURN	MORE
	JP	PUTL
TABCHG	SA.LENGTH	B.TTY+NEW+1 PREVENT ERASING TABS AT END OF LI
ZP	X.LENGTH,BELL	
Sx.SCRATCH	X.LENGTH=MAX	
PL	X.SCRATCH,BELL	
PR.COUNT	X.LENGTH=60	
SA.SCRATCH	B.TTY+TABS	
LT	B.COUNT,*+2	
*	SA.SCRATCH	B.TTY+TABS+1
*	SR.COUNT	B.COUNT=60
*	SR.SCRATCH	B.COUNT+60
Lx.OUT	X.SCRATCH,B.SCRATCH	
X.SCRATCH	1	
RX.OUT	X.OUT-X.SCRATCH	
SP.SCRATCH	=B.COUNT	
Lx.OUT	X.OUT,B.SCRATCH	
SA.OUT	A.SCRATCH	
JP	MORE	
TABR	SA.LENGTH	B.TTY+NEW+1
Sx.OUT	X.LENGTH*B.COUNT	
Sx.OUT	X.OUT=MAX	
PL	X.OUT,BELL	
SA.SCRATCH	P.TTY+TCFLAG	
NG	X.SCRATCH,TABR1	
SA.SCRATCH	B.TTY+OLD+1	
Sx.OUT	X.SCRATCH+B.COUNT	
SA.OUT	A.SCRATCH	
TABR1	Sx.CHAR	BLANK
ZC	B.COUNT,MORE	
SR.RETURN	TABR2	
SR.COUNT	B.COUNT=1	
JP	PUT07	
TABR2	SR.RETURN	TABR1
JP	PUTCTTYT	
TABL	SR.STEP	=1
TAB	SA.LENGTH	B.TTY+NEW+1
	Sx.LENGTH	X.LENGTH*B.STEP
	SP.COUNT	B.STEP
	SA.SCRATCH	B.TTY+TABS=1
	SR.SCRATCH	X.LENGTH
*	SR.SCRATCH	B.SCRATCH=60
	SA.SCRATCH	A.SCRATCH+1
	SE	B.SCRATCH,*-1
	SR.SCRATCH	B.SCRATCH+60
	Lx.SCRATCH	X.SCRATCH,B.SCRATCH
	SR.RETURN	B.STEP+60
	LT	B.STEP,TAB1

	SR. SCRATCH	-R. SCRATCH
	SR. SCRATCH	R. SCRATCH+60
TAB1	AG	X. SCRATCH, POP
	SR. COUNT	R. COUNT+B. STEP
	Lx. SCRATCH	X. SCRATCH, B. RETURN
	SR. SCRATCH	R. SCRATCH=1
	BT	B. SCRATCH, TAB1
	SR. SCRATCH	60
	SA. SCRATCH	A. SCRATCH+B. STEP
	SE	R. STEP, TAB1
	Lx. SCRATCH	X. SCRATCH, R. RETURN
	JP	TAB1
RIGHTTAB	SR. COUNT	-1
	JP	SCAN
LEFTTAB	SR. STEP	-1
	SA. LENGTH	R. TTY+NEW+1
	SX. RETURN	BACK
	JP	SCAN
EDGE	SA. SCRATCH	B. TTY+OLD+2
	Lx. SCRATCH	X. SCRATCH=X. LENGTH
	SR. COUNT	X. SCRATCH=1
	JP	R. RETURN
SCAN	SR. RETURN	SCAN.
	SENDVENT	TABLE1
	BT	R. STEP, GETCTTY
	SX. LENGTH	X. LENGTH+1
	JP	GETCTTY
SCAN.	SENDVENT	TABLE2
	SX. CHAR	X. CHAR+140B
	NX. OUT	ER
	SX. LOOK	-X. OUT*X. CHAR
	SR. RETURN	SCANP
	JP	GETCTY
SCANP	Lx. SCRATCH	X. LOOK=X. CHAR
	Z	X. SCRATCH, GETCTY
	SX. FUNY	ER
	JP	POP
IOWITH	NX. OUT	-X. SCRATCH
	SA. OUT	A. SCRATCH
	SX. CHAR	LEFTBRA
	AG	X. OUT+ECHO
	SX. CHAR	RIGHTRA
	JP	ECHO
COPY	LF	R. COUNT, BELL
	SA. SCRATCH	B. TTY+NEW+1
	SX. OUT	X. SCRATCH+B. COUNT
	SX. SCRATCH	X. OUT-MAX+1
	PL	X. SCRATCH, FAULT
	SA. LENGTH	R. TTY+OLD+1
	SX. LENGTH	X. LENGTH+R. COUNT
	SA. SCRATCH	R. TTY+OLD+2
	Lx. SCRATCH	X. LENGTH-X. SCRATCH
	PL	X. SCRATCH, FAULT
	SX. OUT	X. LENGTH
	SA. OUT	A. LENGTH
	SR. COUNT	-R. COUNT

COPY1	OP	B.COUNT,COPY4	
	SA.LENGTH	B.TTY+OLD+1	
	SX.LENGTH	X.LENGTH+B.COUNT	
	SR.RETURN	COPY2	
	JP	GETC7	
COPY2	SR.RETURN	COPY3	
	SA.LENGTH	B.TTY+NEW+1	
	JP	PUTC7	
COPY3	SR.RETURN	COPY1	
	NZ	X.PRINT,COPY1	
	JP	PUTCTTYT	
COPY4	PL	X.PRINT,MORE	
	JP	CONCAT1	
SKIP	LE	B.COUNT,BELL	
	SA.LENGTH	B.TTY+OLD+1	
	SX.OUT	X.LENGTH+B.COUNT	
	SA.SCRATCH	B.TTY+OLD+2	
	LX.SCRATCH	X.SCRATCH=X.OUT	
	NG	X.SCRATCH,BELL	
	SA.OUT	A.LENGTH	
	SX.CHAR	SKIPCHAR	
	SR.RETURN	SKIP1	
SKP1	LE	B.COUNT,MORE	
	SR.COUNT	B.COUNT=1	
	JP	PUTCTTYT	
LWORD	SA.LENGTH	B.TTY+NEW+1	
	SX.LENGTH	X.LENGTH+1	
	SR STEP	=1	
WORD	SR.RETURN	WORD1	
	JP	GETC7	
WORD1	SR.SCRATCH	X.CHAR=17B	
	SX.SCRATCH	X.CHAR=155B	
	ZP	X.SCRATCH,BELL	SENSE CARRAGE RETURN
	SX.SCRATCH	X.CHAR=73B	
	PL	X.SCRATCH,GETC7	SENSE SPECIAL CHAR
	SA.SCRATCH	WORDRKK	GET BREAK WORD
	LX.SCRATCH	X.SCRATCH+R.SCRATCH	
	NG	X.SCRATCH,GETC7	
	SR.RETURN	WORD2	
	SX.SCRATCH	X.LENGTH=1	
+*	PL	B.STEP="#1	SKIP TEST IF FORWARD SCAN
	ZO	X.SCRATCH,POP	SENSE BEGINNING OF LINE
	SR.SCRATCH	X.CHAR=17B	
	SX.SCRATCH	X.CHAR=73B	
	PL	X.SCRATCH,WORD3	
	SA.SCRATCH	WORDRKK	
	LX.SCRATCH	X.SCRATCH+B.SCRATCH	
	PL	X.SCRATCH,GETC7	
WORD3	SR.COUNT	B.COUNT=B.STEP	
	SX.CHAR	-B.STEP	
	IX.LENGTH	X.LENGTH+X.CHAR	
	JP	POP	
WORDRKK	VFD	1/1	FOR CHARS .LT. 20B IN WORD ALGORITHM
	DUP	10.1	10 DIGITS
	VFD	1/2	
	DUP	7.1	7 INTERMEDIATE

VFD 1/1
DUP 26,1
VFD 1/1
DUP 16,1
VFD 1/1
26 LETTERS
REST OF CHARACTERS

BELL SX.CHAR BELLCHAR
ECHO SP.RETURN MORE
JP PUTCTTYT
LEDGE SA.SCRATCH B.TTY+NEW+1
SR.COUNT X.SCRATCH
SR.COUNT =R.COUNT
BACK SA.SCRATCH B.TTY+NEW+1
SX.OUT X.SCRATCH+B.COUNT
PG X.OUT,BELL
SA.OUT B.TTY+NEW+1
SA.SCRATCH B.TTY+ICFLAG
PG X.SCRATCH+BACK1
SA.SCRATCH B.TTY+OLD+1
BX.OUT X.SCRATCH+B.COUNT
SA.OUT A.SCRATCH
PL X.OUT,BACK1
SX.OUT R6
SA.OUT A.SCRATCH
BACK1 SA.SCRATCH B.TTY+NEW+1
M7 X.SCRATCH,BACK2
RACK TO THE BEGINNING
RESET ICFLAG
RESET NEW LENGTH
RESET OLD LENGTH
SX.OUT B6
SA.OUT B.TTY+OLD+1
SA.OUT B.TTY+ICFLAG
SA.LENGTH FRESH+1
SX.RETURN MORE
JP PUTL

FRESH VFD 4/0,7/76B,7/CR+42/*

BACK2 DATA 2
SX.CHAR PACKCHAR
SR.COUNT B.COUNT+1
GF B.COUNT*ECHO
SX.CHAR 76B
JP ECHO

NEWLINE VFD 4/0,7/CR+49/*

VFD 66/J
TITLE FINAL STUFF

DATA 1
FL DATA 1

END

REFL 20000

REWIND,LGO

REWIND,BINARY

REWIND,TBIN

COPYL,BINARY,LGO,TBIN