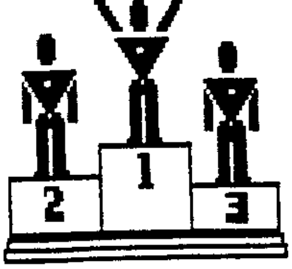
SLAVES AND OTIUG NEWSLETTER MARCH 1992

HAVE A PINCHY SAINT PATRICKS DAY

FEST WEST 92
by
Mel Bragg



On the morning of 14 FEB. 1992, at 4:30 am Dave Mischler picked me (Mel Bragg) up, and headed for Dave DeHeers house to get him loaded, then we headed for Richard Scotts house, arrived there at about 5:15 am. Got Richard loaded, then headed on our journey for Phoenix Arizona.

After a long and tiring drive or ride, we arrived at the the Camelback Hotel at 8:15 pm the 14 Feb.. We got checked into our room threw our suit cases in and went up to the Hospitality room. When we got to the room we walked in and no one was there to introduce us, so we went around and introduced our selves. I guess that is the best way to get started to talking to some of the people that was there.

Bud Mills was there, Gary Bowser (OPA) was there, Berry Miller was there, Mike Maksimik was there, there was several people from the VAST USER GROUP, I don't remember all of there names, I guess if I were taking notes I would have all of their names. While we were in the room we got talking to Rich Gilbertson about his Extended Basic Software, two men came in dressed in suits, one was from Germany, the other was from Holland, I don't remember their names.

Anyway back to Rich Gilbertson, he got talking to us about what he has been doing with Extended Basic telling us what he has added to it. Then he asked us if we would like to see it in use. We said sure we would like to see what he has been doing, He said lets go to his room, So we went up to his room and helped him set up his computer, got it working. He loaded in the ExBasic, it is very powerful in what he has done. Unfortunately you have to have a GRAM KRAKER, a GRAMULATOR, or PGRAM card to use it for now.

Dave Mischler has the P-GRAM card so on Sunday he bought his program. Dave will have to give us a review on it when he learns how to use it..

Finally we got back to our room at 12:45 am. Boy Rich Gilbertson can sure talk your hind leg off.

Dave DeHeer and myself was up at 4:30 am out looking for a cup of coffee. Every thing was closed at that time of the morning, So we came back to the room and woke Dave Mischler up and got his keys and went and found us a Dennys that was open. we drank coffee until it was time for the Hotel cafe was open went back and got up the other guys and went and had breakfast.

Then it was time for FEST WEST to start. We got to the entrance got registered and proceded to browse around and talk to the vendors, and see what was new. And wait for the lectures to start. we all bought raffel tickets for the drawings.

Then we ran into Harold and Helen Hilburn they are from the Ogden TI Users Group, they flew there own airplane down. The OGDEN TI USERS GROUP was very well represented at the faire.

10:00 am the first speaker was Jack Mathis (SW99er's) Which kicked off FEST WEST 92.

11:15 the next speaker was Bill Nelson (Asgard) He spoke on how to make a card. He asked if anyone had a Birthday or Anniversary or what ever I rose my hand and said Dave Mischler has a Birthday on Sunday so Bill made Dave a Birthday card for his demo.

1:00 the next speaker was REGENA (BASIC) she spoke on all of her BASIC programs that she has done for MicroPendium from January thru December. She is a very interesting lady to listen to. And she sure was pushing the Calendars that Dave Mischler and Myself put together for a fund raiser for our User Group I thank her very much for doing that.

2:15 the next speaker was Don O'Neil. He spoke on the Accellerator which is not done yet. And he spoke on the 4MB memory card, He did not give any date when the Accelerator will be done. He is hoping for Chicago Faire.

3:30 Berry Miller (9640) I missed his speech..

That takes care of the speakers on Saturday.

Now we have about a half hour to shop around and listen for our numbers to be called for the drawings.

We were all getting pretty hungry, So we located a RED LOBSTER in the phone book, and went and found it. We got there gave them our name, and waited. It took a little over an hour to get in and get our order in. But it was well worth the wait.

SUNDAY 16 FEB.

Same old routine, up early looking for a place for coffee. The doors opened at 8:00 am so I started to buy all the software that i didn't have. I really wanted the ESD Harddrive but it too was not ready, all Shane Truffel was taking was our addresses to recieve any info when it is ready.

9:30 Don Shorock (Education) I missed his speech also. But I did see his demos at his table. He has some very good programs for learning different languages.

10:45 Rodger Merritt (comprodine) I missed his speech too, and I wanted to watch it

12:00 Chris Taylor (99 Connection) He demoed what he has been doing with forth and it was spectacular. He is looking for input from the TI people as to what type of programs that they would like to use, and he will program them.

He is a very talented person. He was using Forth, a ramdisk, a rambo. And it was truly amazing.

1:15 Ken Gilliland (Notung Software) I missed his speech also to busy buying software and other goodies.

Dave DeHeer had his number called, He won a Brand New PeBox with just the interface card. So he started to pickup RS232 card a Disk Controller Card, 32K Memory Card. He now has a complete system for about $50.00. Bill Nelson won the color printer. I don't remember who won the ESD card and Harddrive when available. It was a $300.00 certificate

They raffeled off a lot of things.

Now it is time for FEST WEST 92 to come to a close. I have made it to two Fest Wests and this one was as good as the other one that I have been to.

They have asked us (The Ogden TI Users Group) if we would be interested in Hosting FEST WEST. We are hashing it out with the Salt Lake City (SLAVES) Group. We will let BJ Mathis know by mid March. We are hoping that we can.

Sunday Evening FEST WEST is over and now we are hungry again. So we look in the phone book for the BLACK ANGUS and we found one. Was that ever good eating, too. Now we got our Bellies full we went back to our room to get rested for our long journey back to UTAH. We left at 4:30 am Monday 17.

We arrived at home at about 8:00 pm Monday night.

Now to get rested up, to get back in the swing of things. We all enjoyed our selves very much. A trip well worth taking.

TIGERCUB PRINTALL

VERSION 1.6

by Jim Peterson

****

This program will print your text in a choice of 1 to 5 columns, and gives you complete choice of fonts, left and right margins, spacing between columns, lines per page, etc., etc. I think the prompts are self-explanatory.

NOTE: Some folks have thought that this program didn't work because they expected it to reformat text into the desired column width. Use Reformatter+ or the FUNLWEB Formatter to do that.

It takes some time to read in text and format it into multiple columns, so if you need to print more than two copies, or will need more copies in future, it will pay you to print back to the disk. To do this, at the printer prompt type over the PIO.LF default with DSK. and a drive number and file name. The text will then be formatted and printed to a D/V 254 file.

The next prompt is for the record length, which will be the default of 80 if the text was prepared with TI-Writer or whatever. *However, if you enter 254 you will be prompted for an input file name of a file printed to disk by this program, and for the number of copies wanted, which will then be printed immediately.*

If you have Triton's Super Extended Basic module, you can LIST an XBasic program to disk in 28-column format by LIST "DSK1.filename":28:1-32766 . The result will be a D/V28 file. With this program you can print the listing in 5 columns by selecting 28 record length, elite condensed, 5 columns, 28 column width.

This version has been modified slightly so that it will allow the use of "Control U" codes input by FUNLWEB, to underline, emphasize, double-strike, etc. an individual word or line. Note that if you are printing in more than one column **you must turn off the codes at the end of the line**, or they will also affect the same line in all subsequent columns. You must also remember that the control codes will be deleted in printing, which will affect the format.

If the file has a Tab setting, first enter T to get to the tab line, place a period to replace the R, then go to the end of the tab line and place an R.

For instance, if you want to underline a word, press CTRL O to get the open cursor fixed mode. Position the cursor on the first letter of the word, type FCTN 2 and then space bar 3 times to open up 3 spaces, backspace to the first of these, and type CTRL U, FCTN R, CTRL U, -, CTRL U, CTRL A, CTRL U. Move the cursor to the first space beyond the word, type FCTN 2, space 3 times, backspace 3, type CTRL U, FCTN R, CTRL U, -, CTRL U, CTRL @, CTRL U.

If the word is at the end of a line or you are underlining a complete line, and the line is not completely filled with characters, go to the end of the line first and put the "turn-off" codes starting in the space just after where the last character would be. For instance, if the column width is 40, start the codes in column 41.

With this method, you can print individual lines or words in *italics*, double-struck, underlined, ^{superscript}, emphasized, or in *different NLQ fonts* or different colors. However, do not use any CTRL U codes for a feature that you plan to select from Printall, or you will turn it off for the rest of the text.

NOTE: When a line contains CTRL U codes, the program will **NOT** warn you or truncate a line which is more than the selected column width.

Although this program is intended primarily for multiple-column printing, it has other uses. If your letter turns out to be 70 lines long and you would like to print it on one page, use this program and select 70 lines. If you need a double-spaced manuscrpt, select 30 lines. If you need a tiny list, such as a list of the songs to put in the case of a music cassette, select elite condensed $_{subscript}$ and 120 lines per page.

Since the TI-99/4A has limited memory, you may get a MEMORY FULL error if you try to format much more than 60 lines of condensed print per page. You can increase this limit considerably by entering CALL FILES(1) and then NEW before loading this program.

NEXT PAGE

```
100 DIM M$(600),F$(50)
110 GOTO 160
120 K,ST,SET,S,P$,P,CL,DW$,S
S$,I$,D$,E$,NC,CW,TC,TA,TX,A
V,CS,S$,LT,A$,LSP,LP,RM,OK$,
QQ$,X,F$(),SL,F,IP,M$(),T$,F
LAG,J,PP,LT$,Q$,F,RL,N
130 EV$,COMP,MAXL
140 CALL CLEAR :: CALL KEY :
: CALL COLOR :: CALL SCREEN
:: CALL SOUND
150 !@P-
160 CALL CLEAR :: CALL KEY(3
,X,ST):: ON WARNING NEXT
170 FOR SET=0 TO 14 :: CALL
COLOR(SET,2,8):: NEXT SET ::
 CALL SCREEN(5)
180 DISPLAY AT(3,6):"TIGERCU
B PRINTALL" :: DISPLAY AT(5,
11):"V.1.6.1":"":" for the N
X1020R and other  Epson-com
patible printers"
190 DISPLAY AT(10,1):"Progra
mmed by Jim Peterson"
200 DISPLAY AT(18,7):"TURN P
RINTER ON!":;:"Set top of fo
rm half inch  below perfora
tions"
210 DISPLAY AT(23,8):"PRESS
ANY KEY" :: DISPLAY AT(23,8)
:"press any key" :: CALL KEY
(0,X,S):: IF S=0 THEN 210 EL
SE CALL CLEAR
220 DISPLAY AT(12,1):"Printe
r designation?" :: DISPLAY A
T(14,1):"PIO.LF" :: ACCEPT A
T(14,1)SIZE(-28)BEEP:P$ :: I
F POS(P$,"DSK",1)<>0 THEN 24
0
230 IF POS(P$,".LF",1)=0 THE
N P$=P$&".LF"
240 OPEN #1:P$,VARIABLE 254
:: PRINT #1:CHR$(27)&"@";::
CALL CLEAR
250 DISPLAY AT(12,1)ERASE AL
L:"Input record length? 80"
:: ACCEPT AT(12,22)VALIDATE(
DIGIT)SIZE(-3)BEEP:RL :: IF
RL<>254 THEN 320
260 DISPLAY AT(12,1)ERASE AL
L:"Filename? DSK" :: ACCEPT
AT(12,14)BEEP:F$
270 OPEN #2:"DSK"&F$,VARIABL
E 254,INPUT
280 DISPLAY AT(14,1):"How ma
ny copies? 1" :: ACCEPT AT(1
4,18)BEEP:N
290 FOR J=1 TO N
300 LINPUT #2:M$ :: PRINT #1
:M$ :: IF EOF(2)<>1 THEN 300
310 RESTORE #2 :: NEXT J ::
CLOSE #2 :: GOTO 220
320 DISPLAY AT(12,1):"Print
size?": :"  (1) Pica":"  (2)
 Elite":"  (3) Condensed":"
 (4) Elite condensed"
330 ACCEPT AT(12,13)VALIDATE
("1234")SIZE(1)BEEP:P :: IF
P=2 THEN PRINT #1:CHR$(27)&C
HR$(77);ELSE IF P=3 THEN PRI
NT #1:CHR$(15);ELSE PRINT #1
:CHR$(27)&CHR$(77)&CHR$(15);
340 CL=(P=1)*80+(P=2)*96+(P=
3)*136+(P=4)*160 :: CL=ABS(C
L)
350 DISPLAY AT(12,1)ERASE AL
L:"NLQ characters? Y" :: ACC
EPT AT(12,17)VALIDATE("YN")S
IZE(-1)BEEP:Q$ :: IF Q$="N"
THEN 380
360 DISPLAY AT(12,1):"Font?
1":"":"(1) Courier":"(2) San
serif":"(3) Script":"(4) Ora
tor"
370 ACCEPT AT(12,7)VALIDATE(
"1234")SIZE(-1)BEEP:F :: F=(
F=1)*0+ABS(F=2)+(F=3)*-4+(F=
4)*-7 :: PRINT #1:CHR$(40)&C
HR$(40)&CHR$(70)&CHR$(41)&CH
R$(41)&CHR$(F)
380 DISPLAY AT(12,1)ERASE AL
L:"Use color? N" :: ACCEPT A
T(12,12)VALIDATE("YN")SIZE(-
1)BEEP:Q$ :: IF Q$="N" THEN
410
390 DISPLAY AT(12,1):"Color?
 1":"(1) Black":"(2) Red":"(
3) Blue":"(4) Violet":"(5) Y
ellow":"(6) Orange":"(7) Gre
en"
400 ACCEPT AT(12,8)VALIDATE(
"1234567")SIZE(-1)BEEP:J ::
PRINT #1:CHR$(27)&CHR$(114)&
CHR$(J-1);
410 DISPLAY AT(12,1)ERASE AL
L:"Double-width? N" :: ACCEP
T AT(12,15)SIZE(-1)VALIDATE(
"YN")BEEP:DW$ :: IF DW$="Y"
THEN PRINT #1:CHR$(27);"W";C
HR$(1);:: CL=CL/2
420 DISPLAY AT(12,1)ERASE AL
L:"Superscript? N" :: ACCEPT
 AT(12,14)SIZE(-1)VALIDATE("
YN")BEEP:SS$ :: IF SS$="Y" T
HEN PRINT #1:CHR$(27);"S";CH
R$(0);
430 DISPLAY AT(12,1)ERASE AL
L:"Italics? N" :: ACCEPT AT(
12,10)VALIDATE("YN")SIZE(-1)
BEEP:I$ :: IF I$="Y" THEN PR
INT #1:CHR$(27);"4";
440 DISPLAY AT(12,1)ERASE AL
L:"Double-strike? Y" :: ACCE
PT AT(12,16)VALIDATE("YN")SI
ZE(-1)BEEP:D$ :: IF D$="Y" T
HEN PRINT #1:CHR$(27);"G";
450 IF P<3 AND SS$<>"Y" THEN
 DISPLAY AT(12,1):"Emphasize
d? Y" :: ACCEPT AT(12,13)VAL
IDATE("YN")SIZE(-1)BEEP:E$ :
: IF E$="Y" THEN PRINT #1:CH
R$(27);"E";
460 DISPLAY AT(12,1)ERASE AL
L:"Number of columns? (1-5)"
 :: ACCEPT AT(12,26)VALIDATE
("12345")SIZE(1)BEEP:NC
470 DISPLAY AT(12,1):"Column
 width (number of": :"charac
ters?" :: ACCEPT AT(14,13)VA
LIDATE(DIGIT)BEEP:CW
480 TC=NC*CW :: TA=CL-TC ::
TX=TC+NC*2-2
490 IF TX<=CL THEN 510 :: DI
SPLAY AT(18,1):STR$(NC)&" co
lumns of "&STR$(CW)&" charac
ters":"plus 2-column spacing
 equals"
500 DISPLAY AT(20,1):STR$(TC
)&" characters; maximum":"av
ailable in print size":"sele
cted is "&STR$(CL)&".":"****
Please reselect****" :: GOTO
 320
510 IF NC=1 THEN 530 :: AV=I
NT(TA/(NC-1)):: DISPLAY AT(1
2,1)ERASE ALL:"Column separa
tion?":"minimum 2":"maximum
"&STR$(AV)&" available ":"2"
520 ACCEPT AT(15,1)VALIDATE(
DIGIT)SIZE(-2)BEEP:CS :: IF
CS<2 OR CS>AV THEN 520 ELSE
S$=RPT$(" ",CS)
530 TA=TA-CS*(NC-1):: IF TA<
2 THEN 570
540 DISPLAY AT(12,1)ERASE AL
L:"Left margin width?": :"ma
ximum "&STR$(TA)&" available
" :: ACCEPT AT(12,20)VALIDAT
E(DIGIT)BEEP:LT :: IF LT>TA
THEN 540
550 DISPLAY AT(12,1):"Altern
ating left/right": :"margins
 (for pages to be":"later re
produced on both":"sides) N"
560 ACCEPT AT(16,8)VALIDATE(
"YN")SIZE(-1)BEEP:A$
570 LSP=12 :: DISPLAY AT(10,
1):" ":" ":"Lines per page?
60":" ":" ":" ":" " :: ACCEP
T AT(12,17)VALIDATE(DIGIT)SI
ZE(-3)BEEP:LP
580 LSP=72/(LP/10):: PRINT #
1:CHR$(27);"A";CHR$(LSP)
590 RM=TA-LT
600 DISPLAY AT(12,1)ERASE AL
L:STR$(NC)&" columns of":STR
$(CW)&"-character width":"le
ft margin of "&STR$(LT)&" sp
aces"
610 DISPLAY AT(15,1):STR$(LP
)&" lines per page":"with "&
STR$(INT(LSP))&"/72 line spa
cing"
620 DISPLAY AT(17,1):STR$(CS
)&" spaces between columns":
"right margin of "&STR$(RM)&
" spaces": :"OK? Y"
630 ACCEPT AT(20,5)VALIDATE(
"YN")SIZE(-1)BEEP:OK$ :: IF
OK$="N" THEN 320
640 DISPLAY AT(12,1)ERASE AL
L:"Pause at end of page? N"
:: ACCEPT AT(12,23)VALIDATE(
"YN")SIZE(-1)BEEP:QQ$ :: IF
NC=1 THEN 660
650 DISPLAY AT(12,1)ERASE AL
L:"Print last page in even":
"columns? Y" :: ACCEPT AT(13
,10)VALIDATE("YN")SIZE(-1)BE
EP:EV$
660 DISPLAY AT(1,1)ERASE ALL
:"Input filenames to be":"pr
inted.":"Press Enter when do
ne."
670 X=X+1 :: DISPLAY AT(X+3,
1):"Filename  DSK" :: ACCEPT
 AT(X+3,14)SIZE(-12)BEEP:F$(
X)
680 IF F$(X)="" THEN X=X-1 :
: GOTO 710 ELSE F$(X)="DSK"&
F$(X)
690 ON ERROR 700 :: OPEN #2:
F$(X),INPUT ,VARIABLE RL ::
CLOSE #2 :: GOTO 670
700 ON ERROR STOP :: CALL SO
UND(1000,110,0,-4,0):: DISPL
AY AT(20,1):"CANNOT OPEN "&F
$(X):: X=X-1 :: RETURN 670
710 ON ERROR STOP
720 SL=1 :: IF NC>1 THEN F=0
 :: GOTO 790
725 K=0 :: PP=1 :: LT$=RPT$(
" ",LT):: FOR J=1 TO X :: OP
EN #2:F$(J),INPUT
730 LINPUT #2:Q$ :: IF ASC(Q
$)=128 THEN 770 :: K=K+1 ::
PRINT #1:LT$&Q$&CHR$(10):: I
F K<LP THEN 770
740 IF QQ$="N" THEN 760
750 DISPLAY AT(24,7):"PRESS
ANY KEY" :: DISPLAY AT(24,7)
:"press any key" :: CALL KEY
(0,X,S):: IF S=0 THEN 750 EL
SE DISPLAY AT(24,7):""
760 PRINT #1:CHR$(12):: K=0
:: PP=PP+1 :: IF PP/2=INT(PP
/2)AND A$="Y" THEN LT$=RPT$(
" ",RM)ELSE LT$=RPT$(" ",LT)
770 IF EOF(2)<>1 THEN 730
780 CLOSE #2 :: NEXT J :: PR
INT #1:CHR$(12):: STOP
790 F=F+1 :: IF F>X THEN 890
 :: ON ERROR 800 :: OPEN #2:
F$(F),INPUT ,VARIABLE RL ::
DISPLAY AT(22,1):"Reading ";
F$(F):: ON ERROR STOP :: GOT
O 810
800 CALL SOUND(1000,110,0,-4
,0):: DISPLAY AT(20,1):"COUL
D NOT OPEN "&F$(F):: STOP
810 FOR IP=SL TO LP*NC :: LI
NPUT #2:M$(IP):: DISPLAY AT(
24,12):IP :: IF LEN(M$(IP))=
0 THEN 860 :: IF NC>1 AND PO
S(M$(IP),CHR$(13),1)<>0 THEN
 M$(IP)=SEG$(M$(IP),1,LEN(M$
(IP))-1)
815 IF LEN(M$(IP))=0 THEN M$
(IP)=RPT$(" ",CW)
820 IF ASC(M$(IP))=128 THEN
IP=IP-1 :: GOTO 870
830 IF ASC(M$(IP))<32 OR POS
(M$(IP),CHR$(27),1)<>0 OR AS
C(SEG$(M$(IP),LEN(M$(IP)),1)
)=32 THEN 860
840 IF LEN(M$(IP))<=CW THEN
860 :: T$=SEG$(M$(IP),1,CW):
: CALL SOUND(1000,110,0,-4,0
):: DISPLAY AT(12,1):M$(IP);
" over";CW;"characters":"tru
ncated to ";T$:"OK?"
850 CALL KEY(3,K,S):: IF S=0
 THEN 850 ELSE IF K<>89 THEN
 STOP ELSE M$(IP)=T$
860 IF LEN(M$(IP))<CW THEN M
$(IP)=M$(IP)&RPT$(" ",CW-LEN
(M$(IP)))
870 IF EOF(2)=1 THEN CLOSE #
2 :: SL=IP+1 :: GOTO 790
880 NEXT IP :: IF EOF(2)=1 T
HEN CLOSE #2 :: GOTO 910 ELS
E GOTO 910
890 FLAG=1 :: FOR J=IP+1 TO
NC*LP :: M$(J)="" :: NEXT J
:: GOTO 910
910 PP=PP+1 :: IF PP/2=INT(P
P/2)AND A$="Y" THEN LT$=RPT$
(" ",RM)ELSE LT$=RPT$(" ",LT
)
```

NEXT PAGE

OBJECT FILES
by Norm Sellers

When I mention how I always, if possible, edit object files instead of re-assembling a program, I get varied responses from amazement to astonishment. I get the idea many people are afraid of object files probably because of some lack of understanding. Therefore, I would like to try to describe the characteristics of object files and how editing, carefully done, is not only possible, but often a preferred approach.

If the TI99/4A and GENEVE worlds , object files are always 'TAGGED'. This means that TI has a TAG or CODE or every word (2 bytes) of the program being loaded that tells the loader what is being loaded and how to load it.

Something to be aware of is that there is some versatility an how object files may be written as long as certain rules are adhered to. A have not seen a formal list of these rules but I will try to list the rules which are deductions of my observations.

The first rule is all OBJECT files are FIXED record length of 80. Each record contains a tag followed by the data to be loaded according to the preceeding tag. with these pairs repeated as many times as possible saving 4 or 5 bytes for a sequence number in the last 4 positions of the record. After all tag/data combinations on a record, the 'F' tag is given with no data following it. This tag indicates an END OF RECORD and the loader ignores the remainder of the record. The last four bytes (pos 77 thru 80) are reserved for an optional 4 digit decimal sequence number.

There are two distinct types of object files: UNCOMPRESSED and COMPRESSED. Uncompressed files are ASCII files (those that may be written and edited with TI Writer) and of course are highly preferred when debugging a new program. Compressed files are faster loading and require less disk space. They are hybrid fixed 80 files containing exactly the same tags and data that the uncompressed files contain, except that the two bytes of data between each ASCII tag are in hex and can not be edited with the TI Writer. The only programs I have found that will edit compressed files are sector editors which show data in both ASCII and HEX. The compressed files may be used when the program is complete, if it is to be used with the EA Cartridge. Extended Basic can not use compressed files.

Certain tags are for RELOCATABLE programs while gther tags ar The ABSOLUTE loaded programs. The relocatable programs contain both absolute and relocatable tags however the absolute programs contain NO relocatable tags.

An object file consists of two portions: The executable code with initialized constants or variables, and following this is the section containing all DEF's, REF's, and an Autostart entry if any.

A sample of object code that is generated with the TI assembler may be found in the TI EA manual on page 245. Page 415 shows a quick reference description of the tags with an indication of whach tags are used with which loaders. Page 309 shows a list of the tags with more detailed information about each tag.

Back to the characteristics of object files. For quick reference now, I'll list the tags with the brief description of each:

TAG		
ABSOLUTE	RELATIVE	FUNCTION
>01	>01	Compressed file
0		Followed by 0000
	0	Module length
1	2	AUTOSTART address
	4	External REF addr.
5	6	External DEF addr.
7	7	Checksum for line
8	8	Ignore Checksum
9	A	Load address
B	C	Load data or code
D	E	Load bias
F	F	End of record
I	I	Program ID
M	E	Data/Common Seg.
:	:	End of file

The first byte of the first record contains either a hex 01 for compressed files and an ASCII '0' if the file is not compressed.

Uncompressed object files created with the TI assembler use the tag '7' to checksum each line. If you change a line, it is either necessary to erase

cont. p7

the '7' checksum tag with its 2 byte checksue at the end of the changed line, gr change the '7' tag to '8' an the changed line to ignore the incorrect checksum.

The simplest type of changes that can be edited into object files are changes that require the same amount of memory, especially that are of the same type as before: things like changing DATA or TEXT statements, changing statements from MOV to MOVB, or changing JEQ to JNE, or LI to AI etc. These simple changes can usually be determined using pages 434 thru 439 of the EA Manual which show the Op codes for each statement. What they do not say is that thas 2 byte Op code also contains information indicating the type of addressing used in the two arguments to the statement. For example,

```
>C107            MOV R7,R6
>C147            MOV R7,R5
>C807 8300       MOV R7,>8300
>C820 A000 B000  MOV >A000 >B000
```

If you need to change registers in a MOV statement, either the register number is contained in the Op code last digit or it increments the 3rd digit by 4 for each register number increase (compare the first 2 moves above). All Op codes from >4000 to >F000 behave exactly alike with respect to how the arguments change the Op code. See the following table which shows the groups of Op codes that behave alike:

From	To
>0200	>02E0
>0340	>03E0
>0400	>0740
>0800	>1F00
>2000	>3C00
>4000	>F000

(Note this table does not contain the add-ons as a result of the argument addressing.

The jump statements, >1000 to >1C00 count words to jump starting with >00 for the word following the jump. Therefore, to jump (any kind) forward skipping 2 words, we would have:

```
>1502            JGT  $+6
>C807 8300       MOV R7,>8300
* NEXT STATEMENT GOES HERE.
```

while jumping back 3 words would be:

```
* NEXT STATEMENT IS THE FOLLOWING MOV
>C807 8300       MOV R7$>8300
>15FD            JGT  $-4       (FD=-3)
```

The important points to remember here are that $ represents the address of the current statement, $+2 is the address of the next statement which normally would be executed (without a jump) and bytes are added to or subtracted from the $ to get the address in the statement to assemble. However, words, not bytes from the following statement are counted an the Op code.

To completely describe how the arguments affect the Op codes would take a full volume of text which is beyond the scope of this news letter, so I must move on to the second section of the object code.

The data for the tags in this section are slightly different. The REF and DEF tags are followed by a 4 hex digit (ASCII if uncompressed, or 2 byte hex if compressed) address followed by 6 ASCII bytes (both compressed and uncompressed) for the name of the entry. If the name of the entry is less than 6, the remaining bytes are blank.

DEF's may easily be added to object files. Simply lock at the address of the entry or variable and add a tag of 5 or 6 according to whether the code is relocatable or absolute respectively, followed by the address followed by the 6 byte name. Simply move the F tag after the entry or add a new line if necessary. The sequence number should be edited to account for the added line but it isn't necessary. I have had to add entries to object files (of 40 page listing assemblies) when I find later that another routine that I am linking with needs a variable from the first assembly.

I have also had to add REF's to assemblies (again 40 page listings) when I assemble to find certain variables or entries undefined because I forgot to indicate they were REF's. In this case the assembler assembles the statements using the variable correctly except >0000 is put in the code for each use of the REF variable.

REF's are a little more tricky since they chain backwards thru the code. This means the '3' or '4' tag you add (like for DEF's) contains the address of the last reference to the variable in the listing (this address often is not

cont. p.8

the address of the statement containing it). For example, if the following code was assembled with the external reference, RF, which was forgotten,

```
>0100 >C021 0000      MOV RF,R1
>0104 >C807 0000      MOV R7,RF
>0108 >C820 0000 0000 MOV >A000 >B000
>010E >1306           JEQ AGN
```

These statements produce 4 unresolved errors and put in place of the unresolved addresses, 0000. To correct this error, just edit the object code adding at the end (before the ':' line, assuming relocatable code here):

```
'3010CRF '
```

and edit in the object code for the statements incorrectly assembled the backward chain of addresses with the first reference left at 0000. I will show these corrections in the listing as above but of course they are made in the object file:

```
>0100 >C021 0000      MOV RF,R1
>0104 >C807 0100      MOV R7,RF
>0108 >C820 0106 010A MOV >A000 >B000
>010E >1306           JEQ AGN
```

The tags in these corrections will be 'C' for relocatable code and 'B' for absolute except for the first 0000 reference always has the 'B' tag.

It sounds more difficult than it really is and a little practice makes it quite easy, just tedious

I would suggest keeping handy a small program with a listing of its object file illustrating how the REF's are correctly done so when you need to edit one into an object file, you have a quick reminder of the process.

Lastly, I would like to discuss adding and deleting AUTOSTARTs. The tag is '1' or '2' for absolute and relocatable code respectively. This tag is followed by 4 hex digits, like the DEF tag which give the address of the first executable statement of the program. If you want to temporarily remove an AUTOSTART, just move it after the 'F' tag of some line (or new line) for the time being.

It would be wise to backup the object file before you edit but after you gain some experience and confidence with it, you will probably get careless with the backups too.

REPRINTED FROM
SUG NEWS 1/92

PROGRAM OF THE MONTH

by Bob August

The program this month will read and list any TI-WRITER program to the screen or printer. The program can be any size and it will still read it. It will also read and/or print any DV80 file, like data files. You need to check line 210 to make sure that the printer name is the same as yours. It is set for "PIO".

To stop the screen from scrolling, so you can read the screen, press the enter key. To start the scrolling again, press the space bar.

Hope you enjoy.

```
100 ! <READ TI-WRITER >
110 ! IN EXTENDED BASIC
120 ! BY R.W. AUGUST
130 CALL CLEAR :: CALL SCREE
N(5):: FOR CS=0 TO 12 :: CAL
L COLOR(CS,16,1):: NEXT CS
140 DISPLAY AT(6,6):"< READ
TT-WRITER >": : : :"DSK DRI
VE [1-9]:[1]"
150 ACCEPT AT(10,19)VALIDATE
(NUMERIC,"123456789")SIZE(-1
)BEEP:N :: N$=STR$(N)
160 DK$="DSK"&N$&"."
170 DISPLAY AT(12,1):"ENTER'
filename'": :DK$
180 ACCEPT AT(14,6)SIZE(10)B
EEP:F$ :: FILE$=DK$&F$
190 DISPLAY AT(16,1):"TO Pri
nter [P]":"or  Screen [S]:[P
]" :: ACCEPT AT(17,18)VALIDA
TE("PS")SIZE(-1)BEEP:P$ :: C
ALL CLEAR
200 OPEN #1:FILE$ :: IF P$<>
"P" THEN 230 ELSE 210
210 DISPLAY AT(12,1):"PRINTI
NG FILE < ";F$;" >" :: OPEN
#2:"PIO",OUTPUT
220 LINPUT #1:A$ :: PRINT #2
:A$ :: IF EOF(1)THEN 320 ELS
E 220
230 PRINT "Press:": :"[ENTER
 KEY] to stop listing.": :"[
SPACE BAR] to continue.": :
240 FOR DELAY=1 TO 800 :: NE
XT DELAY
250 LINPUT #1:A$ :: PRINT A$
 :: IF EOF(1)THEN 290
260 CALL KEY(0,K,S):: IF S<>
1 THEN 250 ELSE 280
270 IF S<>1 THEN 330 ELSE 28
0
```

Cont. P. 9

I WISH by Jim Peterson

I wish that someone would write a rapid disk copier, like Rediskit, that would allow me to specify the drive I wanted to copy to each time, so that I could be able to load a disk into one drive while the disk in another drive was being written to.

Even better, I wish someone would write an even faster disk copier that would read in as much as possible and then write it to two or more drives.

I wish that someone would figure out how to put four more tone generators under the hood of the TI-99/4A, with software to access them.

I wish that someone would write a LINK to assembly to store strings in the expansion memory; the limitation to console memory in Extended Basic is one of the worst, although least-known, faults of the TI.

I wish that someone would write a library of links to assembly, to do the many things that Extended Basic can't do or can't do fast enough.

I wish that the anonymous genius who created the Ernie and Bert program would share his secret with us. He seems to have achieved better sound quality, in far less memory, than Sound F/X.

I wish that someone would remap the keyboard to simulate the Dvorak typewriter.

I wish that someone would write a tutorial on programming in assembly in very simple language that I can understand, using three-letter words to replace such intimidating terms as "least significant byte" and "floating point accumulator".

I wish that someone would write a music composing program. A person with a good knowledge of both music theory and programming should be able to do so, because music is basically mathematical in concept - sounds must vibrate a certain number of times per second in order to be recognized as musical tones, and collections of those tones must be arranged within certain parameters, definable by music theory, in order to sound musical. It should be possible to randomly produce phrases within those parameters and allow the user to select a phrase to be further randomly developed, until a complete melody emerges.

I wish that someone would write a really complete tutorial article on the various types of disk files and the means of accessing them.

If those folks who like to brag about the lightning speed of their low level languages are actually writing programs in those languages, I wish they would share them with us.

I wish that my new Star NX1020R printer had, among its many character sets, those handy graphics characters that were accessible in ASCII 225-254 in the Star emulation of the old Gemini 10X and SG-10.

I wish that the suppliers of products for the TI, and the developers of new products, would advertise in MICROpendium so we could find out what they have to offer!

I wish that printers were made so that they would continually shift either the print head or the ribbon up and down, so that we could use up the entire width of the ribbon instead of that narrow strip down the center. Considering their outrageous prices for ribbon cartridges, they owe us that much!

I wish that the local stores would get together and set up a computer information service so I could use my modem to find out what stores have what I'm looking for, and what their price is.

I wish that manufacturers of electronic equipment would stop labeling all the controls in raised black letters on a black background.

I wish most of all that Texas Instruments had continued developing our computer, that we now had a TI-99/9Z! END

```
280 CALL KEY(0,K,S):: IF K<>
32 THEN 280 ELSE 250
290 CLOSE #1 :: PRINT :"End
of File < ";F$;">": :"Press
any Key."
300 CALL KEY(0,K,S):: IF S=0
 THEN 300 ELSE STOP
310 IF S=0 THEN 300 ELSE STO
P
320 CLOSE #2
330 DISPLAY AT(14,1):"FILE <
 ";F$;" > PRINTED" :: CLOSE
#1 :: END
```

Cont. from P. 8

```
920 IF EV$="Y" AND F>X AND I
P<LP*NC THEN LP=INT(IP/NC)+1
930 FOR J=1 TO LP :: ON NC G
OSUB 950,960,970,980,990 ::
NEXT J :: PRINT #1:CHR$(12):
: SL=1 :: IF F>X THEN STOP E
LSE IF QQ$="N" THEN 810
940 DISPLAY AT(24,1)BEEP:"Pr
ess any key to continue" ::
CALL KEY(0,K,S):: IF S=0 THE
N 940 ELSE DISPLAY AT(24,1):
"" :: GOTO 810
950 PRINT #1:LT$&M$(J)&CHR$(
10):: RETURN
960 PRINT #1:LT$&M$(J)&S$&M$
(J+LP)&CHR$(10):: RETURN
970 PRINT #1:LT$&M$(J)&S$&M$
(J+LP)&S$&M$(J+LP*2)&CHR$(10
):: RETURN
980 PRINT #1:LT$&M$(J)&S$&M$
(J+LP)&S$&M$(J+LP*2)&S$&M$(J
+LP*3)&CHR$(10):: RETURN
990 PRINT #1:LT$&M$(J)&S$&M$
(J+LP)&S$&M$(J+LP*2)&S$&M$(J
+LP*3)&S$&M$(J+LP*4)&CHR$(10
):: RETURN
```
END

Cont. FROM P.5

TI SLAVES AND OGDEN TI USERS GROUPS OFFICERS

	TI SLAVES		OGDEN TI USERS GROUP	
PRESIDENT	JOE MASARONE	966-3694	HAROLD BINGHAM	394-6382
VICE PRES	WARREN YOUNG	278-1052	DAVE DEHEER	394-6815
SEC/TREAS	RENN CRUMP	966-7144	HELEN HILBURN	773-0622
LIBRARIAN	RENN CRUMP	966-7144	ED ISLER	825-9158
ASST.LIB.			TONY LEAVITT	723-3597
NEWSLETTER EDITOR FOR BOTH GROUPS			MEL BRAGG	393-9605

MARCH 1992 NEWSLETTER

TI SLAVES
OUR NEXT MEETING IS MARCH 21 1992 AT 9:00 am WE MEET IN THE DISABLED AMERICAN VETERANS HALL AT 273 E. 800 S. PLEASE BE THERE PROMPTLY.!!

COME AND HAVE FUN.

OGDEN TI USERS GROUP
OUR NEXT MEETING IS MARCH 7th AT 9:00 am and MARCH 17th

WE MEET AT THE OGDEN MUNICIPAL AIRPORT IN THE FIRST BUILDING JUST EAST OF THE NEW TOWER.

COME HAVE FUN.

Slaves & Otiug
1396 Lincoln APT B
Ogden, Utah 84404

SALT LAKE CITY, UT 841
PM
7 MAR
1992