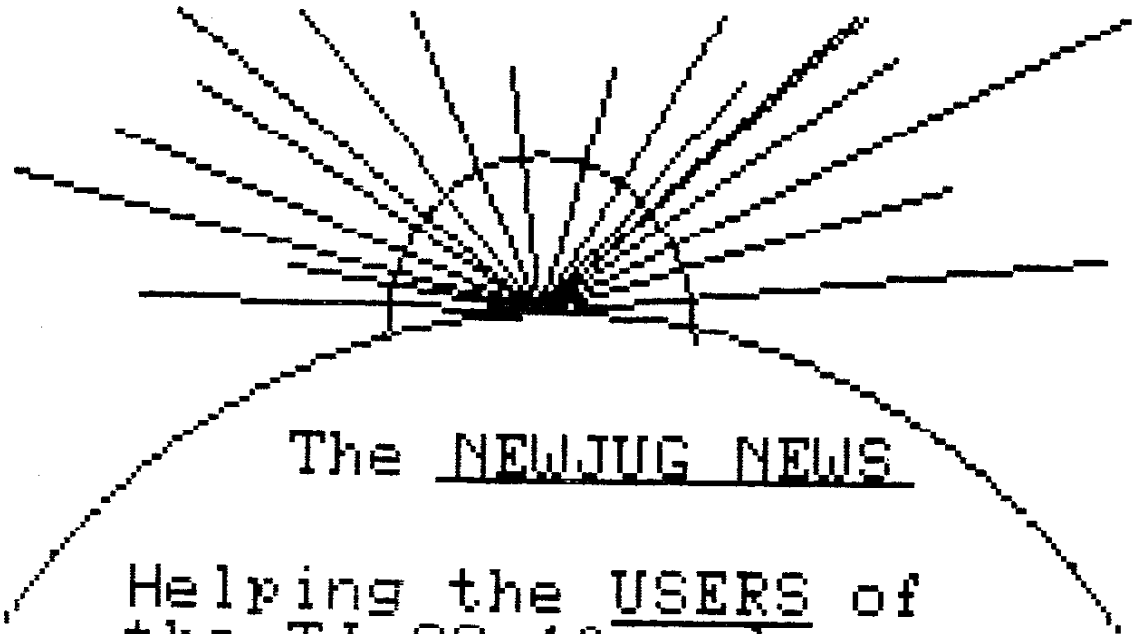
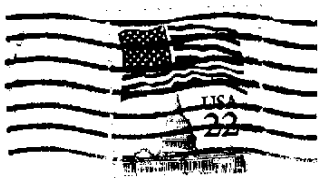


The NEWJUG NEWS

Helping the USERS of the TI-99/4A and Myarc Geneve 9640 into the dawn of a new decade.

NEXT MEETING: NOVEMBER 10, 1987

NEWJUG NEWS
c/o Bret J. Musser
60 Broadway Road
Warren, NJ 07060



Dallas TI Home Computer Group 9/99
P.O. BOX 29863
Dallas, TX 75229

NEWJUG MEETING SCHEDULE
=======================

1987	
NOVEMBER 10	In the Iselin Public Library.
DECEMBER 1	Begins at 6:45. We must vacate the
1988	premises BY 8:45!
JANUARY 5	
FEBRUARY 2	Meetings are open to the public.
MARCH 1	
APRIL 5	
MAY 3	
JUNE 7	

NEWJUG Club Officers

President: Bill Reiss.....679-3067..(BBS:679-0549)
Vice President: Dave Green.....463-9133
Jay Holovacs.....356-3150
John Molinelli....545-5690
Secretary: Johan Nykvist.....727-6217
Treasurer: Mac Rochan.....463-1918
Software Librarian: Dave Green.....463-9133
Newsletter Editor: Bret Musser.....647-1437

* * * * * * * * * * * * * * * * * *

Please send all mail to:

NEWJUG NEWS
c/o Bret Musser
60 Broadway Rd.
Warren, NJ 07060

Membership dues: $15 for individual
$20 for family membership
[All paid members receive the NewJUG News]

Send all dues to: Mac Rochan
6 Sunburst Lane
Piscataway, NJ 08854

If possible, pay your dues at the meeting.

****

November, 1987: The Editor Babbles On...

Well, another month is upon us. Just think--in two months is 1988! Lord help us... Thus I must again request donations for the newsletter. Before January or February I must have at least five college applications filled out, on top of my usual complement of schoolwork. What I'm saying is that now is the perfect time to get involved with the NewJUG News.

On that line, this month's issue won't be as splendid as last month's. I just ran out of time (thank God for Daylight Savings Time--the extra hour this weekend really helped). But, there are still a few things to highlight--Dan Gazsy's c99 tutorial, part 2, the BASIC column, and the Geneve column. I don't want to say too much now -- I'll just let you be suprised as you read.

******CORRECTION******

Last month I stated that TICOFF is not on...I was wrong! >>>>>>TICOFF IS ON! TICOFF IS ON! TICOFF IS ON! TICOFF IS ON!<<<<<< My apologies to Bob Guellentz and the rest of the TICOFF organizers for my foul up.

Also, there is now a new policy with subscription to the NewJUG News--no more free issues. Only paid members will get the issue by mail and if you want the issue when you come to the meeting, then you must see Mac Rochon first. My apologies to the club for the mishaps regarding this. [This note is of course to those who are reading it and who have not subscribed.]

The next meeting is November 10, delayed one week because of elections. The DEADLINE FOR THE NEXT NEWSLETTER IS NOVEMBER 16. Yes, this is a pretty quick turn around -- the December meeting falls on the first of the month. What I'm really looking for as in contributions is reviews of software or hardware, or even a person's first reactions on some item. If you like teaching, then I'm sure many of us, or at least myself, do not know how to use Multiplan efficiently and do not have the time/desire/will to look through the Multiplan manual, so this would be a nice topic to cover. If you're a programmer, how about some reactions to the two new programming languages for the TI -- Fortran IV and Turbo-Pasc '99. I'm planning to get Turbo-Pasc '99 if it conforms to Standard Pascal, but I'm not planning to get this version of Fortran -- Fortran IV, by its name, sounds like an old implementation of Fortran, or at least a version not up to par with Fortran-77 or Fortran-83 (or whatever the latest version is). Of course, I could be wrong, and I would just love for someone to point me the way...

Sorry for the lateness of this issue -- it really came down to the wire waiting for promised artices, which didn't come in by the last possible deadline (a week before the meeting is not much time to get out a newsletter). But, through the galliant efforts of Bruce Barlow, it did get out -- he copied and stapled and mailed the newsletters for me, saving two complicated steps for me.

Anyhow, enjoy this month and keep an open mind.

Happy Computing.

-Bret

GENEVE COLUMN: MDOS, and updates on news...
by Bret Nusser

Well, I've been using different versions of MDOS for a month now. As I write this (mid-October) MDOS is at version 0.99b, with only one or two commands missing. The most recognizable of which are the Print Screen functions. BATCH files, such as AUTOEXEC, are fully operational, yet there is one resounding downpoint to this entire releasal of DOS--nothing but the GPL interpreter and My-Art will run under 9640 (Geneve) mode. Soon though, My-Word will be converted to run under DOS and the p-code system should also be out soon (give it two months or so for safe date keeping. ie, by Christmas). Also, Advanced BASIC should come around the same time the p-system comes. [Update: BASIC will probably not come out when the p-system does. It has been very delayed and reportedly has had major portions rewritten.] BASIC was delayed as Myarc pulled the programmers from BASIC to finish the final phases of DOS.

So, what's with DOS, you ask? Well, its almost complete. I've been testing it for Myarc for a little while and the only bugs I've found have been in my hardware. For instance, even though my RS232 card has worked fairly well with the Geneve up to now, it has been messing up DOS. I tried an original TI RS232 and it worked perfectly. Here's the kicker: I own a Myarc RS232 card! Oh, well. Maybe in a few paragraphs I'll write in an update concerning the situation (I don't write the whole article in one sitting). But with the TI card, operation has been flawless. [Update: it is the system software that is messing things up with the combination of the Myarc RS232 and TI controller. By next month, it should be fixed.]

I do have a few complaints, but some of those can be fixed in the AUTOEXEC file, such as booting up into 80 columns. Also speaking of batch files, the CONFIGSYS file has been removed and will not be recognized. The reason for this is that the system automatically recognizes the hardware attached through DSRLNKs in the ROMs of the hardware. In MS-DOS, the CONFIG.SYS file would configure the system, but you would have to tell it what is in your machine. Another complaint I have is in some of their operations, such as DISKCOMP (disk compare). It reads *in the entire disk* before comparing! In the long run, I'm sure its more efficient, but not if you only want the first, let's say, 20 sectors compared. You must wait over 30 seconds to read in one disk, and then have it compare it with the second disk. Of course, if you have a single drive system, then this is a Godsend. Speaking of single drive systems, that is where many of the inadequacies of the Geneve lie (software wise, that is). All of the Geneve software must be written to support the Lowest Commond Denominator (LCD). In this case the LCD is the old single side, single density, TI disk controller disk system. Myarc could have made ALL of the software MUCH better if it were not for this factor: if everyone had DSDD, then they would have done things differently. Also, if software authors on the TI had done things differently, life would have been made easier. For example, with the SSSD system, 32 sectors were enough to hold the directory of the entire disk--really no more than 30 files could fit on a SSSD disk, unless they were all very small. Some software authors wrote this directly into their code, thus fixing it forever at 30 files. When the new DSDD came along, 30 sectors for file directories was not enough, but because some authors had encoded this figure directly, it had to be upheld. So, now you get your directories, when you have over 30 files, spread out over the entire disk--a real mess, and performance thus drops. The case has been similar with the design of the Geneve: Myarc had to keep to a very low LCD, because some people still do have SSSD, one drive systems. Sigh.

Anyhow, notes on DOS. First off, some fixes from my little demo at our October meeting. There we tried the ASSIGN command, but it didn't work. The proper format is now, contrary to the manual (written over a year ago):

ASSIGN C=DSK1:

This statement would make all access to drive C: goto drive A:. Also, last month I recommened not using punctuation marks in your files. Well, you can use them now. Just do the following in DOS: any name with something such as a "/", or whatever, enclose in quotes. For example:

RENAME A:"E/A" A:EA

Redirection to a printer now works. How? Well, two ways. First off, you can press CRTL-P and all from that moment on is echoed to the printer, as well as the screen. Also, as in MS/PC-DOS, you can do the following:

DIR > PRN

This causes the directory to output on the printer only. printer. Secondly, as in MS-DOS, you can do the following: "[function and parameters] > PRN" or "...> RS232", etc. This just redirects the output from the screen to the specified device.

Overall, it seems to be a pretty good system. If you have a knowledge of assembly programming then DOS is a pleasure to program under. For instance, to display to the screen, you would just preform an XOP and be done with it. To access disk, just preform an XOP. Also, for assembly programmers (the only way to program for 9640 mode), there is support for multitasking, but it isn't quite what you think: one program must load another, which must load another, etc. then each will execute simultaneously. But still, the promise is there. I just wish that I had stuck with assembly two summers ago, but then I reasoned "Why, with c99 and FORTH?" 20-20 Hindsight...

Well, another day, another dollar... to the government, that is. But that shouldn't be too much now--I have Multiplan to help me on my way. Now I can do my taxes on it, calculate a projection of the increase in the Federal Deficit (although that is probably too large for Multiplan), or keep a list of colleges to apply to along with other pertinent information about each one.

So, with the Multiplan on the Geneve, are there any improvements? Well, I'm not quite sure. I never used Multiplan on the TI, nor on an IBM, nor anywhere! Just by looking at the screen, all the difference seems to be physical--the display is now 80 columns by 26 rows, you have the date or time at the bottom of the screen, along with the formula of the current cell, etc. This all seems to be quite similar to the TI version. What is improved? Well, now you can have spreadsheets up to 41K or so (that is all that is probably allowed by original Microsoft programmers, just like TI only allowed a possible max of 56K for TI-Writer). Of course, when you run at speed 5, its much faster, except for one point--saving data. According to the Myarc Addendums, there is a "new verification process" that insures your data is saved correctly. Why no improvements? Well, simply put, Microsoft owns the Copyright to Multiplan, and Myarc could not go around improving upon Microsoft's programs. On TI-Writer/My-Word, TI-Writer was basically put into the public domain, especially with all the copies of Funlwriter, BA-Writer, QS-Writer, etc. floating around. (You're really just buying the manual when you purchase it--the disks can be had anywhere.) TI just kinda gave it up; Microsoft is still around to whip you.

Many people have been confused by the highest resolution mode of the Geneve--the 512 x 424, where the 424 vertical resolution is attained by *interlacing* the video. So, for those who are confused by the term "interlace", here is a short explanation. Usually, the screen is display or updated 60 times per second, just like the frequency of household current. In interlace mode, two images are projected to the screen instead of one, and the second image in *interlaced* with the first, that is, it is shifted down one line on the screen. This creates a display with very nice text, except for a little flicker in the display. It flickers because two screens are being displayed in the time one was before. Thus, each screen is display only 30 times per second, just slow enough were the human eye can detect a litte flicker. Of course, the easiest way to understand it all is to actually see it as someone explains it.

Non-interlaced screen	Interlaced screen
AAAAAAAAAAAAAAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
	BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
AAAAAAAAAAAAAAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
	BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
AAAAAAAAAAAAAAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
	BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

As you can see, the interlaced mode fills the entire screen as one complete image using two sections of video ram: A and B. It first displays A, drops a line, the displays B. This, of course, uses twice as much video ram, but with 128 K of VDP RAM, you won't care about that. On the left side, you see non-interlaced mode. This displays only one "screen" from VDP RAM and thus does it 60 times a second. But then you are left with the empty lines between the text. Some people like the interlace (like myself), but others are very sensitive to that little flicker. If you want the good text, but no flicker, then buy a longer phosphorus RGB analog monitor. I'm using an Amiga 1080 RGB analog monitor, and I can see the flicker, but I believe the Sony RGB analog monitor will lessen the flicker (I'm not sure of the model number), but at a higher cost.

Even more user notes: in MyWord, use CRTL-SHFT-ALT to adjust your screen (interlace/non-interlace, to display the info of the bottom of the screen/to not display it).

There are still a few misconceptions about hardware compatibility with the Geneve. You CAN use ANY disk controller card (I'm using a TI controller). Also, you can use ANY RS232 card. About ramdisks, you can of course use a Myarc ramdisk, and you can use a Horizon ramdisk(I'm not sure if you need a new EPROM for it or not, but you can use it). I am not sure about the Corcomp ramdisk, but I would imagine that you could use it at least as a ramdisk, if not as true memory expansion. You can use a TV/RF modulator, composite color, monochrome, or RGB analog to display your picture.

I must be honest with you about the system software that I've been using on the Geneve. The truth is that the software has never inspired confidence in me. That is, I have never, and I doubt that I will ever, trust the software, especially (only) in disk access. Recently I downloaded two beautiful files from a BBS--Tom Bentley's Windows and his updated TCIO library. I now have neither one. Why? Well, because, probably due to Myarc Basic 2.11 (the interim BASIC, not the final version at all), they were destroyed by the opening of another file on the same disk. In some later experimentation, I noted that my BASIC disk/files were kind of old, and I have since updated them, but I have not tried using BASIC 2.11 again, nor will I in the future, except in circumstances where I'm not going to use any disk files. The moral is: I don't, nor will I ever, trust Myarc BASIC 2.11 (the interim BASIC, mind you, NOT the final BASIC 3.0).

Returning to something a bit more optimistic (I've calmed down in the past few days), I have done some serious thinking about purchasing the Myarc Hard Disk controller (the one that controls up to 3 140 Meg HDs, and 4-5 DSQD floppies, remember?). Although it has had the usual delays, and although I have a hate-like relationship with my Geneve, the Geneve problems are software related -- ie. they CAN be fixed, I still want one. Why? Well, for a number of reasons. One, I'll have over 30 megabytes of storage (if I get a 30 Meg. HD) to store files on. Just think--no more putting floppies into the machine to boot up, to load My-Word, to load anything! Just think--when I'm on a BBS, no more "NO DISK SPACE" errors (at least not for a month or two!). Just think--all of this loading and saving of files at super-fast speed! Why else? Well, one more main reason--DMA (Direct Memory Access). This means that the controller card can place programs/data into memory without microprocessor intervention (notice how your machine stops when accessing disk drives?). This will allow a whole universe of MULTI-TASKING applications to come in! With the Geneve running over twice as fast as the old TI, I could run two programs at the same speed as a TI at the same time. Why do this? Well, one big application I though of while waiting half an hour on the modem--downloading files. With multitasking, you could download a file using a terminal program and be editing a text file on My-Word or using Multiplan at the same time! Neither process would be interrupted when the other was accessing the disk, so work would go undisturbed. Downpoints about the system? Price. $260 for the controller, at LEAST $400 for a decent hard disk, and another $100 for a power supply/box for the hard disk (I'll put it outside my PEB). So, for the system that I would want, with 30 Megs, it would cost around $800. Eeek!

Well, by the next issue I should have my first assembly program that runs under DOS written. I received the documentation (quite sparse -- all it is really is a list of the OP codes) for using DOS commands in your programs (such as opening files, printing to the screen, etc). These docs don't make good bedtime reading, but they are quite interesting. I have a lot to learn about XOPs, their use, and how to stick them into my program. What program am I trying to write? The simple Sieve Of Eratosthenes, the classical benchmark. It seems like a pretty simple program to write (so I can concentrate on the interface with DOS), and an assembly version will provide an interesting time comparison with a c99 version, and (if I ever get it) a Turbo-Pasc '99 version. Who knows, maybe this will start a spark in me to always write in assembly? Just make sure that you reserve the first half dozen registers for passing parameters to the XOPs. For example, to write a character to the screen:

```
* First set the video mode: text mode 2
 XOP @six,00
* where R1: Which video mode you want
* Now, to set the cursor position
 XOP @six,0Z
* R1: page number, R2: row number, R3: column number
* Now to write a character
 XOP @six,0A
* R1: ASCII character to write, R2: foregnd.color, R3: bkgnd color
* R4: Number of times to write character
```

Now, at least to me, this is a whole lot easier than VSBW, etc. I don't have to worry as much about if I want VSBW or VMBW or how to initialize text mode, or bit-map mode, or whatever. DOS does all the nitty-gritty details for you!

Happy computing (I hope). :-)

[Cool feature just noted: In MY-Word, press F7 in the formatter--you get a disk directory!]

BASIC COLUMN: Subroutines

BY BRET MUSSER

According to Computer Scientists, the following is a list of requirements for a "theoretically complete" computer language:

1) Sequential execution
2) Branching (IF-THEN, etc)
3) Looping
4) Subroutines or procedures

Thus,BASIC, by this defination, is a theoretically complete programming language. Most books that I have read covering the BASIC programming language covered the first three of those requirements quite well--they are common to every BASIC implementation. But TI Extended BASIC also has true subroutines, something missing in many other BASICs. By a true subroutine, I do not mean the use of GOSUBs, or the equivalent, but the use of CALL XXXX(), where the user writes the subroutine XXXX.

You're all familar with TI BASICs built in subroutines--CALL CLEAR, CALL SPRITE, CALL CHAR, etc. Well, you can write your own subroutines that you can access by CALLing it. Read on and see:

Sample program #1:

```
 10 CALL CLEAR
 20 PRINT "HELLO"
 30 CALL GETDATA(A,B,C)
  .
  .
  .
400 STOP
1000 SUB GETDATA(A,B,C)
1010 INPUT "PLEASE ENTER VALUE FOR A:":A
1020 INPUT "PLEASE ENTER VALUE FOR B:":B
1030 INPUT "PLEASE ENTER VALUE FOR C:":C
1040 SUBEND
```

In sample program #1, I've introduced you to a very simple subroutine--to get user input. Instead of having the INPUTs in the main program, I have separated them into a special area denoted by the SUB and SUBEND statements. SUB and SUBEND mark the beginning and ending of a subroutine. On the SUB line you also type two more things: 1) the name of your subroutine. There are really only one restriction on the name and that is that it cannot already be defined by the computer (eg. CALL CLEAR, PRINT, etc. ie. no reserved words allowed). Also on the SUB line (sublime?) is the parameter list--this is where you pass values back and forth from your program to your subroutine. Why pass values back and forth? Well, if you want to use anything (variables) from the main program then you must hand them to the subroutine this way. Inside the subroutine, the variables are called "local" because they can only be accessed in the subroutine, and likevise the variables in the main program cannot be shared with the subroutine (unless they are, of course, passed through the parameter list). This explains my having put (A,B,C) in my simple program above.

I'm sure that was alot to digest (I know it was alot to type), so let me back up a step or two. In my example, why bother with a subroutine for getting the data when it is only three INPUTs? Well, let's say that you won't always be INPUTting data from the keyboard (eg. you're using keyboard input for debugging purposes only) and then you switch to have the input come from disk. Will you rewrite all that code inside the main program (can you squeeze it all in between lines 30 and 60?), or is it simpler just to rewrite the subroutine and not have to disturb any main program code (which might cause more bugs). The point is this: with a subroutine, the main program doesn't care how the data gets back, just that it does. The details of data retrieval are left for a seperate part of the program and does not influence the operation of the main program.

A few more notes on subroutines. First, subroutines in TI XBASIC go at the END of a program, not before the main program as in Pascal and other languages. Second, once you start your first SUB, no main program lines can proceed it. You cannot do the following:

```
 10 REM MAIN PROGRAM HERE
200 SUB FIRSTSUBPROGRAM(PARAMETER LIST)
210 REM SUBROUTINE GOES HERE...
300 SUBEND
310 CALL CLEAR            <--THIS LINE IS ILLEGAL! Once you
                    SUBEND, only more SUB...SUBENDs can
                    follow!
```

Also, as I mentioned before, all variable within a subroutine are local only to that subroutine--once SUBEND arrives, those variables no longer exist. Finally, if if you wish to depart from a subroutine early, use SUBEXIT.

Well, here follows a small, fairly insignificant, do nothing special program demostrating the use of subroutines.

```
100  REM  STARTING THE MAIN PROGRAM
110  REM  PROGRAM TAKES IN AN EMPLOYEE #, AND BASED ON THAT CALCS
120  REM  A PAY FOR THAT PERSON.
130  REM
140  REM  THE VARIABLES:
150  REM       EMPNO   = EMPLOYEE #
160  REM       PAYRATE = PAY RATE
170  REM       HRS     = HOURS WORKED IN ONE WEEK
180  REM       GPAY    = GROSS PAY
190  REM       NPAY    = NET PAY
200  REM       DEDS    = DEDUCTIONS
210  REM
220  REM
230  REM  NOW TO BEGIN THE MAIN PROGRAM
240  !
250  CALL GETDATA(EMPNO,HRS)       !Get the data.
260  !    the main program doesn't care where the data comes from
270  !    as long as GETDATA returns some values
280  !
290  CALL GETPAYRATE(EMPNO,PAYRATE)     ! will return PAYRATE
300  !    based on the value of the employee number
310  !
320  CALL GROSSPAY(HRS,PAYRATE,GPAY)
330  !    Will calc GPAY from HouRS worked and PAYRATE
340  !
350  CALL DEDUCT(GPAY,DEDS)   !just like the IRS--DEDS based on
360  !                              GrossPAY.
370  !
380  CALL NET(GPAY,DEDS,NETPAY) !calc net pay.
390  !
400  CALL PRNTDATA(EMPNO,HRS,GPAY,NETPAY)    !display the data
410  END  !We're all done!
```

###SO FAR: the main program is 6 lines long! Now I'm going to write each module/subroutine to preform each individual action. As I stated before, I can change the details in a subroutine, like getting the data from disk, or printing the results to PIO printer without disturbing the flow of the main program.

```
1000 REM THE SUBROUTINES
1010 !
1020 SUB GETDATA(A,B)  ! <--THE PARAMETER VARS. DO NOT HAVE TO BE
1030 !                        THE SAME!
1040 INPUT "PLEASE ENTER EMPLOYEE #":A
1050 INPUT "PLEASE ENTER THE NUMBER OF HOURS WORKED: ":B
1060 !
1070 SUBEND
1090 !
1100 SUB GETPAYRATE(EMPNO,PAYRATE)
1105      !Of course, if you want them to be the same, they can be
1110 !
1120 IF EMPNO < 100 THEN PAYRATE = 3.50 ELSE 1130
1125 SUBEXIT !Nothing further to do, so lets get outta here!
1130 IF EMPNO < 200 THEN PAYRATE = 4.50 ELSE 1140
1135 SUBEXIT
1140 IF EMPNO < 300 THEN PAYRATE = 7.50 ELSE PAYRATE = 10.00
1150 SUBEND
1160 !    Now, if I ever changed the formula for the pay scale,
1165 !    then I'd only have to change these lines here, not the
1170 !    lines in the main program (once all the bugs in the main
1180 !    program are out, then nothing will disturb it this way).
1190 !
1200 SUB GROSSPAY(HRS,PAYRATE,GPAY)
1220 GPAY = HRS * PAYRATE
1230 IF HRS > 40 THEN GPAY = GPAY + PAYRATE * (HRS-40)
1240 !
1250 SUBEND
1270 !    Like I keep saying... I can now easily change this
1280 !    formula without disrupting any other routines
1290 !
1300 SUB DEDUCT(A,B)
1320 ! B REFERENCES TO DEDS, A TO GPAY (REFS TO MAIN PROGRAM VARS)
1325 ! Deduction based on amount of pay...(sound familar?)
1330 IF GPAY < 100 THEN B = GPAY * .20  ELSE 1340
1335 SUBEXIT
1340 IF GPAY < 200 THEN B = GPAY * .25  ELSE 1350
1345 SUBEXIT
1350 IF GPAY < 300 THEN B = GPAY * .30  ELSE B = GPAY * .50
1360 SUBEND
1370 !
1400 SUB NET(PARAMETER1,PARAMETER2,PARAMETER3)
1420 ! PARAMETER1 IS REFERENCE MAIN PROG. VARIABLE GPAY
1430 ! PARAMETER2 IS REFERENCE MAIN PROG. VARIABLE DEDS
1440 ! PARAMETER3 IS REFERENCE MAIN PROG. VARIABLE NETPAY
1460 PARAMETER3 = GPAY - DEDS
1470 SUBEND
1480 !        As I always say, if there was another equation to
1485 !    fit here, this is an easier place to put it.
1490 !
1500 SUB PRNTDATA(A,B,C,D)
1510 !
1520 PRINT "Employee #";A;"  Hours worked: ";B
1530 PRINT "Gross pay: $";C;" Net pay: $";D
1540 SUBEND
1550 ! that's all, folks!
```

FROM TI TO PC AND BACK

by John Bonito

Do you wish you could transfer files from your TI to your PC? It's easier than you think. Read on for details to transmit ASCII files from and to the TI 99 4/A and the PC-XT. Although the specifics in this article apply to the TI, with the proper cabling or using a null modem, data can be transmitted between any two computers.

If you are using a modem with your PC you're part way there. Use the standard RS232 modem cable you are presently using without modification. It has all the wires connected in parallel to the same pin number on each connector. Only 8 of the 25 wires are used and since the TI RS232 is configured as Data Communications Equipment (DCE) it is not necessary to switch pins 2 and 3. Just plug each cable connector into the serial ports of each computer.

Boot up both computers and load a communications program into each one. Since I use ProComm on my PC the comments will apply to that program but the procedure is the same regardless which program you use. Using ProComm on the XT and either Fast-Term or P-Term on the TI, both programs will transmit and receive properly. Configure each program to use the same parameters and set to access port 1.

All files to be transmitted must have been saved in ASCII format. Default on some word processors is to save the files in their own unique format but the user can elect to save the file in ASCII. If unsure of the file format, load the file in the word processor and save it in ASCII format before transmitting. When transmitting, if you get hex values or garbage on the screen, it indicates a non-ASCII file.

If the lines overwrite each other on the screen, then line feeds are required. Use ALT F3 on ProComm or FCTN J on Fast-Term to toggle line feeds on/off.

USING P-TERM

Receiving on the XT...

1 - On the XT, press ALT F1 and enter the drive/filename to open a buffer to capture the file

2 - On the TI, press CTRL 2 and enter the drive/filename of the file to be sent, then press CTRL 3 to send line by line or CTRL 4 to send all at once. The XT should be displaying text

3 - On the XT, when transmission is complete, press ALT F1 to close the buffer and write the buffer to disk

Receiving on the TI...

1 - On the TI, press CTRL 5 to clear the buffer, if necessary

2 - On the XT, press PgUp, then choose 7 to upload ASCII files and enter the drive/filename of the file to be sent. The TI should be displaying text

3 - On the TI, when transmission is complete, press CTRL 6 to close the buffer, write the file to disk, and prompt for a new drive/filename. If you aren't sending again, press <ENTER> to return to terminal mode

USING FAST-TERM

Receiving on the XT...

1 - On the XT, press ALT F1 and enter the drive/filename to op
en
a buffer to capture the file

2 - On the TI, press FCTN N and enter the drive/filename ofthe file to be sent then press FCTN , (comma). After choosing whether to send line by line or all at once, press enter

3 - On the XT, press ALT F1 to close the buffer and write the file to disk

Receiving on the TI...

1 - On the TI, press FCTN 8 to clear the buffer, if necessary, then enter the drive/filename to open a capture buffer

2 - On the XT, press PgUp, then choose 7 to upload ASCII files and enter the drive/filename of the file to be sent

3 - On the TI, press FCTN 8 to close the buffer, write the file to disk to disk, and prompt for a new filename. If not receiving any more files, press <ENTER> to return to the terminal mode.

Note: Using Fast-Term and sending a file from the XT to the TI, the TI screen will display the first 40 columns of an 80 column line, then scroll up one line and display the remainder of the 80 column line sent. The next 80 column line transmitted will overwrite the last 40 columns displayed , scroll up one line and send the remainder of the second 80 column line. The complete file will be displayed in this manner on the screen making it unreadable but the file will be saved to disk as an 80 column file in its entirety. If you want to read the file as it is being transmitted, change the Fast-Term parameter file.

Referring to page 15 of the TI RS232 manual, column 1 indicates the transfer time for a 256 byte record from the serial port. Columns 2 and 3 indicate the transmission times for two different length files at different baud rates from the TI to the XT. Transmitting over 1200 Baud caused characters to be dropped.

	(1)	(2) 230 WORDS	(3) 1490 WORDS
BAUD	SECONDS	SECONDS	MIN:SEC
300	8.6	52.04	6:05
1200	2.5	14.02	1:34
9600	0.3	9.26	--
19200	-	--	--

You can use this method to save typing ASCII files you may want to port over to your PC. Also, you can list a basic or Xbasic file to disk (i.e. LIST "DSKn.filename") to convert the program to a D/V 80 file. Then load the file into basic on your PC and edit each line to change the code to the proper format. That's another story.

Also, you can use TI Writer and a communications program on the XT. Load a file into TI Writer, then choose PrintFile (PF) and enter RS232 for the device name. Open a buffer on the communications program to receive the file. The open buffer on the XT's terminal software will capture the data. Don't forget to close the buffer on the XT to save the file to disk. When using this method you must use 300 Baud.

This setup works for me on my PC-XT. If you are unsure about the wiring, check with another user who might be familiar with the setup.

ATTENTION ALL TI-99/4A OWNERS:

TICOFF '88 ***IS ON***

Yes, its true (contrary to what I mistakenly printed here last month)!

MARCH 26, 1988 at the Roselle Park High School, just like last year!

Come, bring your kids, your wife, yourself!

Demonstrations! Sales! Meet the "big names" of the TI community!

Let me repeat it:

TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.
TICOFF IS ON! MARCH 26, 1988 at the Roselle Park H.S.

Got it?

For more information, contact:

Bob Guellentz
Evenings: 201-382-5963

Using the OPTIMIZER and RANGE FINDER with c99

by

Dan Gazsy

Sysop: Beaver BBS

For those of you who are familiar with writing , compiling and assembling c programs on the TI99, you soon realize just how much extra assembly source code is generated by the c99 compiler. This is not a problem for most c programs, however if you attempt to write anything that uses a bunch of support functions; you soon realize that the object files generated by the assembler can be rather large. Realizing this could be a significant problem, Tom Wible of Sterling,Va wrote an assembly code optimizer and a range finder which would significantly reduce the assembly source code. The remainder of this article is dedicated to the use of both the optimizer and the range finder.

The first thing you'll want to do is compile you c program. Everyone uses their own set of standards as far as naming conventions are concerned. Mine are as follows:

```
c source code       -   ;c extension
       c compiler output   -   ;s extension
       optimizer output    -   ;opt extension
       range finder output -   ;j2b extension
assembler output    -   ;o extension
```

After you've compiled your program it's time to run the optimizer. The optimizer is an option 5 file which can be run from the E/A environment. First question the optimizer will ask you, is the name of the input file. Reply with the compiler output file (mine has a ;s extension). After this it will give you a recommended output file name with the original input filename plus the ;opt extension. Generally, I remove the ;s in the filename to keep it within the file naming convention limits. The next thing it will ask is if you'd like to change the specific type of branch instructions and it also mentions that it may cause range errors. To this question reply Y. The next question you'll see is in reference to instructions which pass parameters to functions like printf.

What they mean is if any of your printf statements look like this

```
char *parm1,*parm2;
parm1="test"; parm2="printf";
printf("this is a %s of the %s function",parm1,parm2);
```

then answer N to this question (Values parm1 and parm2 are passed parameters). That's it as far as responding to questions for the optimizer. Now sit back and wait for it to finish running. Upon completion, we now run the Assembler supplying the filename with the ;OPT extension as the input file and the file with the ;o extension as the output file. In the event any assembly range errors are detected, they will be listed on the screen. Record these numbers, cause you're going need them as input to the range finder program. If there were no assembly range errors, then its not necessary to run the Range finder. If assembly errors other then range errors are generated; Range finder will not resolve them and you should look elsewhere for your problem (preferably in your c program). The assumption I'm making is that the assembler generated some range errors and its time to run range finder. Range finder is an option 5 file that loads from E/A environment. After you load it in, you'll be prompted for an input filename. Reply with the filename produced by the optimizer (mine have ;opt extensions). After this it will suggest an output filename consisting of the input file plus ;j2b added on. Again make sure that the filename doesn't exceed the file naming conventions. I usually shorten mine by removing the ;opt from the filename. The only thing left is to supply the range finder with the line numbers that appeared on the screen while the assembler was executing. Keep in mind that the range finder will only accept 80 characters of input; then begin to execute. For most programs, this is ample space; if you cant do the complete job in one pass you'll have to run this process multiple number of times (that includes re assembly). The next assumption I'll make is that we have managed to convert all the range errors and we are ready to recompile. Load the assembler again and this time use the filename with the ;j2b extension as the input filename and the output filename again has the ;o extension. The only other option you may want to specify is the C or compress option for the output of the assembler. If everything went well, the assembly process should execute and 0 errors generated. You have just optimized your c program and in some cases seen a savings as high as 20% to your program size.

==========

Dan Gazsy is the Sysop of the Beaver BBS, phone number 238-8170. Dan has been involved with c99 since it was first released and is one of the premier authorities on c99. We are lucky to have such a contributor to the NewJUG News!

==========

CORRECTION ON LAST MONTH'S COLUMN:

A member on our October 6 meeting brought to our attention the following point: Dan's tutorial was written using the 2.1 version of Clint Pulley's c99 compiler. The member tried using 2.0 and it did not work, crashing on the statement:

"extern ...."

This compiler directive allows the use of external functions, such as printf(). If you have version 2.0, change the extern to #asm, REF xxxx, #endasm. I'm not sure if that is the only difference, but that is the point that crashed last month.

==========

To obtain the latest version of c99, contact Dave Green, our club libarian at the meeting or contact him at 463-9133. c99 is based on Small-c by Ron Cain and is distibuted as a Fairware product. Charges by the club for copying DO NOT go to the author -- you must pay him separately.

The BBS Scene: Xmodem and updates

by Bret Musser

First, as usual, the list of BBSs serving New Jersey, with one addition and one withdraval:

Name or program	Tele #	SysOp	System notes
TURBO BBS	257-2607	Bill Wright	
BEAVER BBS	238-8170	Dan Gaszy	
OBT	679-0549	Bill Reiss	
RAMER '99 BBS	584-5373	Dave Sontos	North NJ 99ers U.G.
NNJTIBBS	472-1799	?	North NJ TI UG 300 baud

As you can see, I have added on the North NJ UG's BBS (not the same as the North NJ 99'ers). Right now the board is running at 300 baud, because their newly acquired 1200 baud modem started smoking (their board has always been at 300 baud and that modem was supposed to make it 1200 baud--they got it working for a few hours, but the modem was obviously defetive).

Also as you can see, I have removed Panhandler BBS from this month's listing. Sorry Matt, but I have tried for two months to call it up with no luck. If someone can verify a logon to his BBS, I'll put it back in the listing, but until then, its out.

Some updates: Beaver BBS, which was vicously attacked last month, is doing well. Operations have returned to normal and its running as good as ever, if not better with the new word wrap routines in place in the message base.

Last month I asked for information about PC Pursuit. Well, for some it looks quite promising, but not for me. Why not? Well, PC Pursuit uses local Telenet numbers and the problem is that none are local to me! So, I won't be getting the service, since I'd still have to pay NJ Bell about 5-15 cents per minute connect time (it does add up).

Now for my usual groveling time: if anyone has anything to say about telecommunications, please write an article and send it to me! Topics? Well, how about an ariticle on signing up and using PC Pursuit, or a review of the most popular terminal programs, or a report on the status of the II Special Interest Groups on the major services such as Compuserve, The Source, GEnie, BIX, etc.

Well, this month I'm giving you an article I was going to produce last month--a short attempt at explaining the XMODEM transfer protocol (my attempt at keeping the interest of the experts programmers out there). Alot of the article was taken from a BBS I was logged onto, so if you can't understand it, don't fully blame me! (Gee, what an intro to an article!)

THE XMODEM PROTOCOL

XMODEM, as I just mentioned, is a "transfer protocol." In other words, it is a method by which two computers, even if they are different makes or models, can transfer files. The XMODEM protocol is how two computers "handshake" and send control codes to each other indicating errors, and other conditions (eg. "Pause so I can save"). The XMODEM standard (it is a true standard unlike "standard" BASIC, or "standard" RS232, etc) is now on almost every computer, from our humble TI-99/4A (90K floppies, 48K RAM max) to the latest in personal computer technology such as the IBM PS/2 Model 80 (with a 115 megabyte hard disk, 2 megabytes of RAM... well, you get the idea). There are other protocols by which data is transmitted (Kermit, YMODEM) and in future columns I'll be discussing those. But until then, let's take a crack at XMODEM...

Why is XMODEM so popular? Well, two reasons. One, it started with late, but popular, CP/M operating system (the closest standard U.S. that personal computers have had) and it spread throughout the entire CP/M world, then was converted for other machines. Also, it has an extremely high accuracy rate (over 99.6%!).

As you explore BBSs, you probably notice that there are two forms of XMODEM: "checksum" and "CRC." The checksum method is quite simple: the receiving computer adds up the characters and compares that one byte value to the last byte in the block. If equal, then it may proceed with the next block. As I'm sure you can deduce, the checksum can be easily fooled by switching characters (the sum will be the same, but the order of data different). Thus the need for a second method, CRC.

Xmodem/CRC uses a cyclical redundancy check, which is not easily fooled by exchanging the places of two characters because the calculations performed on each character are positional, also decreasing the chances of several characters being replaced to add up to the same CRC.

CRC Xmodem works a little different. It sends the same block prefix information and uses the initial same block of 128 bytes of data, but it does a little something different. When preparing a block to send, it takes the 128 bytes of the block and adds two CHR$(0)'s onto the end - so that the block is now 130 characters long. Then it calculates the CRC for the block - including the two 0's. It does this by executing a FOR/NEXT loop inside the assembly routine. When it is done, it has calculated a CRC Value which is comprised of a high and low bit. These two bits are then inserted into the block to send in place of the two CHR$(0)'s and the block is sent to the end user. When they are received, the 130 bytes are sent through a similar loop. If at the end the last two bytes equal 0 - or in other words if CRC Value is zero - the block is good and an ACK (acknowledge--data good) should be returned to the host. If not, a NAK (ie. resend data) should be sent back. In other words, the CRC sort of converts the two end bytes back to their CHR$(0) values (sort of)!

As you can see, the CRC is a bit more complex. But it is also much more accurate (I have yet to have bad up/download because of CRC XMODEM). When given the choice, if your software supports it, always choose CRC. You will be rewarded in error-free transmissions. I stated before that you can transfer between two different computers--the data will transfer, but you may not get readable results. For instance, on 99.9% of IBM BBSs, I find articles that I want to read, but it is stored in non-ASCII format, and is thus unreadable to me (not stored in ASCII to save space). So when transfering between two different computers, be sure to use a common "language" such as ASCII.

Sorry if this article scared away any beginners to modems. Next month, I'll have reviews of around half a dozen terminal programs, ranging from the simple TE-II to a program that supports on-line RLE graphics.

Until then, Happy Computing.

:-)

[Parts of this article downloaded from local BBSs.]

LIBRARY LISTING: OCTOBER

DISKNAME	AVAIL	USED	# PROGS
ANIMATOR	135	223	7
BEST/HYMNS	23	335	24
BEST/SONGS	0	358	18
BITROUTINE	314	44	2
BOARD&PUZL	110	248	7
C99REL1	1	357	25
CALENDAR	139	219	9
CATLIB*14	83	275	5
COMP/CRAPS	32	326	9
COMP/UNCOM	322	36	2
DIR99	44	314	12
DM10003/5	4	354	8
DMGR-1000	127	231	6
DVUGSAMPLE	4	354	24
FAST-TERM	40	318	13
FILEREADER	171	187	14
FNLWR3/5-E	573	705	33
FUNL_WRTR	6	352	17
GPL-1	8	350	18
HOMEFINAN	41	317	9
HORIZON-6	0	358	30
MASSCOPY	291	67	3
MENULOADER	156	202	11
MINIBASE99	200	158	5
NEATLIST	0	358	16
P-MANUAL	24	334	1
PILOT	0	358	3
PRBASE	630	648	16
PROG_AIDS	205	153	12
SBUG-V3/1+	0	358	7
SCRABBLE	56	302	6
SCREENDUMP	51	307	8
SIDEWAYS	78	280	16
SPEECHSAVR	78	280	20
TI-REWRITE	15	343	10
TIMP&TIWRT	67	291	12
TK-WRITER	1	357	17
TRIO+FW1A	226	132	4
TRIVIA99ER	49	309	14
UTILITES1	160	198	10
UTILITIES2	211	147	5
WEATHER	72	286	7
X-MODEM+	108	250	7
XB-IIDEMO	924	514	17
X_D	20	338	23

The Disk Of The Month was not ready by publication time. Sorry. Next month the DOM will be back and I'm told its going to be great!