The Boston Computer Society TI-99/ 4A User Group Meeting Newsletter August 1988

Edited by J. Peter Hoddie

The September Meeting

The September meeting of the Boston Computer Society TI-99/4A User Group will be held on September 21 at 7:30 PM at the Massachusetts College of Art on Huntington Avenue in Boston. It will probably be in a new room, so watch for signs.

The meeting topic this month will focus on data management applications. This includes things like database programs (FirstBase, TI-Base, PR-Base), sort programs and programs that help you to manage your disk collections. If you have a favorite program that could fit into one of these catagories and would like to see it demoed or would like to demo it, just contact me a few days before the meeting.

The October Meeting

The October meeting will also be held at Mass Art. Since I got this memo from the BCS saying that October was something like National Computer Education Month, I figure that maybe this would be a good time to review some of the now classic education software available on the 99/4A. In particular we will look at the extensive Plato system, and some other materials.

The November Meeting

Traditionally in November we have focused on the really new products, mostly because the biggest TI show of the year is in the beginning of November - the Chicago TI Faire. This year the plan is the same. There are some exciting products being readied for release including disk copiers, word processors, at least one game, a probably a few surprises. This should be an interesting meeting, as usual. The reports from the show will be first hand as several BCS members are planning to attend.

MYARC News

For those of you following the continuing saga of MYARC, the latest rumors, news, products, and demos will be presented at BCS meetings this fall as they have been for the past year or so. For those of you who are sick of hearing MYARC news, oh well. I do take requests....

And in General

We meet on the third Wednesday of the month. Always. No summer off like many BCS groups do. If you have a question about a meeting or some TI related question in general I can be reached during the academic year at 375-6003 (area code 617). If you have a question about a meeting date call the office at 367-8080.

Bitmap supplement

Last mont this newsletter included a supplement - a list of nearly every TI book ever printed with capsule reviews by Mike Wright (with some support from Barry Traver). Next month we will feature another supplement from Mike Wright.

Next month this newsletter will feature a tutorial on bitmap mode, which can only be accessed through assembly language (or Forth). This mode allows you access to each pixel on the 256x192 TI screen.

The article was written for and supplied as part of a presentation at the 1987 Boston Computer Fayuh. It has since been corrected and improved.

The accompanying demonstration program will make some pretty patterns, and should ensure that you get at least one program running in bitmap mode.

T-pourrI

by Mike Wright

Graphics applications are among the most popular software packages on the TI-99/4A. Programs such as TI-Artist, Graphx, CSGD and Joypaint enjoy widespread use.

However, nearly all of these programs seem to be missing a vital element. Although they allow you to create an image on the screen using 16 colors, they are only able to print it out in stodgy black and white.

This state of affairs has been addressed by the Ninety-Niners of the Vancouver Area (NOVA) in Washington, who claim to be the first TI user group to publish a full color page in a newsletter. Their president, Dan Lisson, wrote in the club's June 1988 issue:

"I personally enjoy printing color — especially for graphics: the problem, as far as I know, is that none of the popular graphics programs for our computer includes a color print driver. I know of only one exception — that of Paint 'N' Print printing to a Seikosha GP-700M printer. But how many of you out there own that printer? For that matter, how many of you own/use Paint 'N' Print for your graphics work or play? I'm not putting down this program, just commenting on what I see and hear from the four corners of the 99er world. This fine program has the best magnify mode of any program I've seen!

"From what I've seen, most of you use TI-Artist or Graphx (or both). With color printers becoming more and more common and considerably less expensive as time goes by, I suspect that many of you are acquiring these machines as you add printers to your systems, or as you replace older printers (trading up). By the way, the Star NX-1000 Rainbow is commonly available for under $300.

"What might be nice to see, in my opinion, is a color printer choice for at least TI-Artist that would support printing to the Epson JX, or even the preferred EX, standard. Actually, it would be nice to see the same for Graphx and Paint 'N' Print, and perhaps a few others. If color is so nice on a screen, why shouldn't it be enjoyed equally in print.

"It seemed that a color page would call attention to the enjoyment to be gained from printing in color. If you like the idea, feel free to let someone know: like NOVA, or your favorite graphics software author.

"Now that you know one reason that we've undertaken the color page project, it's only fair that we tell you how it was done. We did not cheat by printing it from another computer (it wouldn't be kosher!). We used a TI-99/4A, and printed from Paint 'N' Print to a Seikosha GP-700M. If you are looking at an original of NOVA's newsletter for June 1988, then you also have an original printed color page (not photocopied, because it's too expensive).

"If you are not familiar with Paint 'N' Print, it is essentially compatible with Graphx (in one direction). Graphx stores its entire graphics information in a memory image file format 54 sectors long, while Paint 'N' Print does the same in 57 sectors. This means that you can load a Graphx format file into Paint 'N' Print (missing, however, the color information for about the bottom sixth of the screen), but not the other way around. That is why you see the popular RLE picture WarGirl, which we colored with Paint 'N' Print.

If you would like to take this idea further, why not write to: NOVA, PO Box 508, Vancouver, WA 98666.

The book list

So far there has been one addition to the book list published last month discovered by BCS member Neil MacDonald:

TI-99/4A BASIC Guide by Giarratano. Computext, 1983. An average introduction to various aspects of the TI and how they are applied.

Hello...hello...hello

Bulletin boards suffer from what is called distributed processing. For example, if you call the BCS board in Boston, the people in Seattle and Miami will never know about it. However, it has the advantage that you probably only have to make a local call, so it ends up costing nothing.

One way out of this is centralized processing. In this scheme everybody calls one big bulletin board, such as CompuServe or The Source. The disadvantage here is that you need expensive equipment to maintain a large database and since it has to be in one place, you end up paying long distance charges one way or the other. If they offer Telenet, then they get you on connect charges.

Some time ago in the PC world someone came up with the bright idea of getting one bulletin board to call another and trade messages. This was done when the rates were cheap or by using PC Pursuit. I believe this was called Fidonet. This allowed you to ask a question in Boston and, within a few days, have it migrate throughout the land.

This concept has now spread to the TI world and is called TI-Echo. The aim of TI-Echo is have at least one participating BBS in all of the cities serviced by PC Pursuit. There are currently four numbers that you may care to try:

```
605 338 7050 Dakota InfoNet
919 851 8460 NC Central (PCP)
503 692 7024 Orcgon Opus (PCP)
919 833 3412 TI Raleigh (PCP)
```

All are run by sysops that still own TI-99/4As. As they pick up more and more BBSs, they will post the information in the TI-Echo area and also on the major network services.

And best of all — it costs absolutely nothing. It's yet another example of dedicated TIers at work to the benefit of the entire community.

More efficient Basic

Many times you come across tips on ways to make your programs more efficient. Most times they promise to save you a byte here and there and are often so arcane that I tend to stick with what I know. However, I recently came across the following piece of elegance.

In a game program you will often want to compare the final score to the high score and keep whichever is higher. The usual way is:

```
2800 IF FINAL_SCORE > HIGH_SCORE THE
HIGH_SCORE = FINAL_SCORE
```

But a better way is:

```
2800 HIGH_SCORE = MAX (HIGH_SCORE,
FINAL_SCORE)
```

Let your fingers to the typing

Almost all volunteer clubs have those that do and those that don't — contribute to the newsletter that is.

In a bid to reverse this universal trend, Steve DeGeare, editor of the Kansas City Connection, has proposed the following:

"As you will find with any editor of a computer club newsletter, there is always the need for fresh and new material to be published. Where is this source of material to be published. It come from you, the club member. Now I am sure that the members do not always like to see reprints from other newsletters covering the pages of their own newsletter. So what can be done?

"This editor has come across an idea which a few other newsletter editors have tried or implemented. The following suggestion or proposal is given so that you as members may be encouraged to participate in your newsletter.

"KC 99er CONNECTION will publish any ORIGINAL articles dealing with computers (such as program reviews, tricks and tips, etc). It must be ORIGINAL material to qualify for what is about to be suggested.

"In return for your submissions to our newsletter (that is, with each article given) you will be allowed to withdraw 3 programs from out KC TI-99/4A exchange library. Now where else could you get 3 wonderful programs for an ORIGINAL article to keep your newsletter fresh and creative?

[If you're still not convinced, think of it this way. By making a contribution you can help to cut out some of this sort of guff.]

Runner editor

BBC Radio in London, England has a long-lived program called "Desert island discs". The program host gets hold a celebrity who must pretend that he is about to be stranded on a desert island. However, he has the luxury of being able to choose a dozen records (discs in England) to be stranded with. Thoughtfully, the question of turntable and stylus service and electric power are never raised.

Now transpose the scene to the TI world. If there was one game that I would choose for my desert island sojourn it would be TI Runner, by EB Software.

But just in case I ever managed to get through all 50 screens and started to get bored, I would take along the TI Runner editor. The game is constructed so that it reads in each screen from disk. The editor allows you to create and store your own screens. It is available from: EB Software, 905 West Middlefield Rd #953, Mountain View, CA 94043. Cost is $19.95 and the program requires 32K memory, disk drive and Xbasic.

Another expansion box

At last year's Chicago Fair Ryte-Data was showing an expansion box that was based on an IBM box. Unfortunately, Ryte-Data has been going through a bad patch and has proven very difficult to contact.

At last month's BCS meeting, Corson Wyman, of Genial Computerware and the MUNCH user group, let on that he was close to bring another expansion box to market. For more details write to: Genial Computerware, PO Box 183, Grafton, MA 01519.

c.COLUM

by Donald L.Mahler

This month we will continue to look at tips for using Clint Pulley's c99 files for the 9640. The program at the end, however, can be compiled with either the 9640 or the 99/4A.

1) To include stdio_h, the I/O file, you must write:

```
#include "dskx.stdio_h"
```

If you use:

```
#include dskx.stdio_h (no quotes)
```

the file name will be truncated, and you get an error message.

2) You must be in 80 column mode to use QDA; if you try in 40 column mode screen is filled with garbage!

3) To make the whole process automatic
— editing, compiling, and assembling
— set up a batch file similar to this (DSK5. is my 9640 RD)

```
B:QDE %1
B:C99C %1 DSK5.SFILE
B:
QDA DSK5.SFILE %2 /C
```

Note that since we are in MDOS, we can call drives A:,B:, etc. If you call the file e.g. CLINT, you set up this way:

```
CLINT A:FILE_C  A:FILE O
```

FILE_C on drive 1 will be read by QDE on drive 2 and appear on screen. Once edited and saved (by ESC,ESC), the compiler will be loaded, and the source code stored on the RD as SFILE. Then QDA loads, and the object file ends up on drive 1 in compressed mode, ready to "load and run", most easily using QDL!

The program this month is my modification of Kirkwood's program which appeared in the April MICROoendium. I am fond of using SCANF; it always feels like a bargain when I can pick up multiple variable values with a single command!

```
/*FILE I/O PROGRAM*/
#include"dsk2.stdio_h"  /*library I/O file */
extern fprintf();
extern printf();
extern scanf();
main()
{char buf[81],*mode;     /* *mode is a pointer
*/
int in,out,col;
int b,o;  char m;
col=80;
printf("FILE I/O PROGRAM by CHARLES KIRKWOOD
\n\n");
printf("Modified by dlm \n\n");
printf("This program will copy a file and
also \n");
printf(" append one file to another \n \n");
printf("The INPUT dsk and file is the file
\n");
printf("to be copied or appended. \n \n");
printf("Input dsk  and filename:   ");
in=fopen(gets(buf),"r");   /*opens input file
```

```
as "read"-note compression */
puts('\n');
printf("Output dsk and filename,mode of
output (w/a):  ");
scanf("%s %c", o,&m);  /*picks up filename
and mode */
while((m != 'a')&&(m != 'w'))
{  printf("\n Try again.Must be lower case
'a' or 'w'!  ");
   scanf("%s%c",o,&m);
}
/* filters out incorrect mode values */
if(m=='a')     mode="APPEND";
if(m=='w')     mode="WRITE";
printf("File is %s and mode is %s",o,mode);
if(m=='a')
   out=fopen(o,"a");  /*opens output file */
 else
   out=fopen(o,"w");
b=fgets(buf,col,in);
while(b)
 {fprintf(out,"%s\n",b);
 b=fgets(buf,col,in); }
putchar('\n');
fclose(out);
fclose(in); }
```

If you are running this using QDL, the "command file" is:

```
B:CSUP
B:CFIO
B:PRINTF
C:FPRINTF
B:SCANF
A:PROG_O
```

(Program is on dsk1, new c99c files for 9640 are on dsk2, and old library is on dsk3. Of course, you can run using E/A3 and loading in each of files manually; program name is as usual START)

Intro to the UCSD P-System

By Ron Williams

This month I will cover the complete number of steps needed to edit, compile and execute a Pascal program. First boot up the p-system by putting the p-system disks in each drive and turning on the computer. The first thing that should be done after booting is to press the "F" key to go into the filer and then press the "D" key to set the system date. This way all files and programs created will have the correct date next to them when doing a directory of the disk. The other reason that you would want the date changed is any updates to a file or program will have a new date, any program updates can be checked by the date created. The way to get out of the filer program after setting the correct date is to press the "Q" key and this will put you at the main command menu.

To start the editor press the "E" key and you will see a prompt "no workfile present, file?" press enter at this point or if you want to load a file into the editor enter its name by typing its d... number like this #4: followed by the file name like this #4:my... the .text suffix will automatically be appended to it. For ... example we will assume there is no file present so press en... and you will see "edit:" followed by a few commands the cursor will be in the upper left hand side of the screen waiting for a command. To start entering text press the "I" key to insert text.

Enter the following text as an example:

```
Program count;
```

When you get to the semi-colon press enter and the cursor will drop down to the next line if you make a mistake press fctn arrow to go over the error then re-type it. If you make a mistake and notice it after you press enter press control-c and this will take you out of insert mode and at this point all arrow keys now work, press fctn arrow to put the cursor over the mistake and at this point you may press the "X" key to type over an incorrect character, or use the insert mode to insert text on this line after finished with either mode press control-c to go back to command mode.

Type the rest of this text as follows:

```
Program count;
  var
    number : integer;
  begin
    page(output);
    for number:=1 to 10 do
      writeln(number);
    page(output);
  end.
```

This program will print out the numbers from 1 to 10 and is very simple but it will be good enough as an example program. If you need to delete a line of text or just one letter put the cursor over it with fctn arrow keys while in command mode then press the "D" key to delete one letter press the space bar or to delete a line just press enter to go out of this mode press control-c. The adjust mode is entered by pressing the "A" key all arrow keys now work. To move over a line of text put the cursor on it and use the arrow left and arrow right keys to move it. After the program is entered press "Q" while in command mode. The next screen you will see has the following prompts:

```
>QUIT:

U(pdate the workfile and leave
E(dit without updating
R(eturn to the editor without updating
W(rite to a file name and return
```

Press the "U" key to write out the file in the editor's buffer to drive #4: with a file name of system.wrk.text this is the default name used when updating. Exit will return you to command mode and all contents of the buffer will be lost. Return will just put you back in the editor use this key if you pressed "Q" by mistake. Write will let you save the buffer to any file name or drive use this key when

sh to save out a file and then return to the editor. The file written out to disk then you will be given a chance to exit rn to the editor.

r editing the program you must now compile it and execute it. To compile a program saved as system.wrk.text press the "C" key at the main command menu or press the "R" key to compile and execute the program all in one step. When compiling the screen will first go blank then at the top of the screen you will see compiling... then the version number.

The compiling process looks like this:

```
Pascal compiler - release 99/4 IV.0 cla-4
<0    >....
COUNT
<4    >....
8 lines compiled

count    .
```

If any errors are found you can directly go back to the editor or continue the compiling process. If the "R" option was used the program will begin right after compiling if not you can execute the program from the main command line by pressing "R" or pressing the "X" key and entering the file name of the program which will be for this example system.wrk you don't need to add the .code suffix when executing a program as it will be added automatically. If you did add it put a period after the "e" in code this will make the computer find the correct file and not system.wrk.code.code.

Well that's it for this month I hope this was useful to you. I have had a few people ask me about this in the past. So long until next month.

Random Ramblings

By J. Peter Hoddie

MacFlix

If you have been to a BCS TI-99/4A User Group meeting in June or July of this year, you have probably seen a demo of a program I have been working on which allows the use of MacPaint graphics on the 99/4A and 9640 computers. The working name of the program was "Mac the Ripper" but this has been changed to "MacFlix" for reasons which escape me. In any case, the program is finally ready for public release so I thought I would present some information about it here. The following paragraphs have been toned down from some advertising copy I've written for MacFlix.

Using MacFlix you can load and view MacPaint image ese images are a full 8 by 10 inches so MacFlix gives you the power to easily move around the picture. If you have an Epson compatible or ProWriter printer the image may be printed. With a keystroke MacFlix saves your image as a TI-Artist screen which can be manipulated with most TI graphics programs!

With a 9640 you can view the picture in high-res mode, with or without interlace. On a 9640 MacFlix, can actually display more than a standard Mac screen. MacFlix also saves images in both MY-Art formats.

MacFlix also has the ability to invert a picture, and convenient Catalog and Delete File routines. If you own PC-Transfer , MacFlix comes with a custom conversion routine that lets you move MacPaint pictures to and from IBM disks, a source of thousands of pictures.

Written in assembly language, MacFlix runs on a TI-99/4A computer with Extended BASIC or Editor Assembler, or Super-Cart; or on a MYARC 9640 computer.

MacFlix is available for only $15. Orders received before September 15 will get a free disk of MacPaint pictures.

I should probably mention that the Genial Computerware address is P.O. Box 183, Grafton, MA 01519.

Another Assembly Example

This summer I have been doing much more coding than I get the opportunity to do during the normal school year. This has led to an over abundance of source code sitting around on my hard drive. I have cooked up a bunch of little assembly routines that do odd things, and this newsletter seems an appropriate place to present them. The one I chose for this month is a really good example program. The routine I want to focus on is called MOVEIT. The purpose of this routine is to make text strings bounce around the screen. It can handle text bouncing horizontally, vertically, and even diagonally. It will work with 32, 40, and 80 columns.

It can handle having multiple strings [illegible] once. I originally wrote it as a test of an idea for the title screen to my new MacFlix program. I didn't like it enough to use there, although I suspect there are many places where it could be applied.

The routine is based on a simple data structure which is 10 bytes long plus the string to be bounced. You pass the routine the address of this structure following the BL @MOVEIT as illustrated in the code below. I use the DORG (Dummy ORiGin) directive to generate offset addresses for the structure. All that DORG does is tells the assembler to (1) stop generating code but (2) continue generating labels. It is a really convenient way to simulate the record types of high level languages. The same effect can be accomplished with EQUates but it is not as clear to read. For some excellent examples of using the DORG directive see the source code to Fast-Term. That's how I learned to use DORG.

To keep multiple strings moving at the same time you simply make multiple calls to MOVEIT. The example below includes both a horizontal and vertical bouncing string. For best results the delay loop should really be synched to the 60 cycle VDP interrupt, but that is beyond the scope of the example.

The routine handles any reasonable data passed to it fine. Don't pass it a string that is longer than the area it is supposed to bounce in or the results could be a mess.

This routine is presented in the interest of making TI software a bit more "fun." I mean sure this routine is essentially useless (although it is a good example to learn from, if you really understand it you're doing well). It doesn't sort your mailing list, it doesn't compress files, nothing. I just moves some letters around. On the flip side, software that has interesting effects, or clever use of graphics or sound, is more enjoyable to use. However, one should also be careful not to let such features get in the way of using the program. It's a fine line to walk. End of hypocritical speech. Here's the code:

```
*
       REF  VSBW
*
*       sample data
STRCT1 DATA 0,0,1,31
       BYTE 5
       TEXT 'HELLO'
       EVEN
STRCT2 DATA 32+10,32+10,32,32*23+10
       BYTE 6
       TEXT 'A DEMO'
       EVEN
*
*      demo program
TEST   LWPI >8300
*
TEST0  BL   @MOVEIT
       DATA STRCT1
*
       BL   @MOVEIT
       DATA STRCT2
*
       LI   R0,>2000
       LIMI 2
TEST1  DEC  R0
       JNE  TEST1
       LIMI 0
       JMP  TEST0

*--------------
*      the subroutine
*--------------
       DORG 0
MOVSTA DATA 0                      starting position
MOVLOC DATA 0                      current location
MOVINC DATA 0                      increment
MOVSTO DATA 0                      stop position
MOVLEN BYTE 0                      length
MOVTXT BYTE 0                      the text...
*
       RORG
*
MOVEIT MOV  *R11+,R5
*
       MOVB @MOVLEN(R5),R2         get length
       SRL  R2,8                    as word
       MOV  @MOVINC(R5),R0         get increment
       ABS  R0                      ...as a positive
number
       MPY  R0,R2                  r3=offset from this
run
       A    @MOVLOC(R5),R3         r3=stop position
       C    R3,@MOVSTO(R5)         too far?
       JLE  MOVE10                  nope
MOVE09 NEG  @MOVINC(R5)            change directions
       A    @MOVINC(R5),@MOVLOC(R5)
       A    @MOVINC(R5),@MOVLOC(R5)
MOVE10 C    @MOVLOC(R5),@MOVSTA(R5) are we at the
start?
       JLT  MOVE09
       MOVB @MOVLEN(R5),R2
       SRL  R2,8
       LI   R3,MOVTXT
       A    R5,R3                  r3 points to text now
       MOV  @MOVLOC(R5),R0         get       current
```

```
ion....
      MOV  @MOVINC(R5),R4
      ABS  R4                    increment to use
 writing....
      S    R4,R0                 point to previous
tion
      C    R0,@MOVSTA(R5)        before start?
      JLT  MOVE15                 yes
      C    R0,@MOVSTO(R5)        after end?
      JGT  MOVE15                 yes
      LI   R1,'  '
      BLWP @VSBW
MOVE15 A   R4,R0                 next position
      MOVB *R3+,R1               fetch a character
      BLWP @VSBW                 write a character
      DEC  R2                    done?
      JNE  MOVE15                 nope
*
      A    R4,R0
      C    R0,@MOVSTA(R5)        before start?
      JLT  MOVE25                 yes
      C    R0,@MOVSTO(R5)        after end?
      JGT  MOVE25                 yes
      LI   R1,'  '
      BLWP @VSBW
MOVE25 A   @MOVINC(R5),@MOVLOC(R5)
*
      RT
*
      END
*
```

Another noteworthy feature of this program is the use of a DATA statement following the BL @ MOVEIT to pass the address of the record structure. This saves 2 bytes over using the more traditional LI x,data form of passing data to subroutines. It also increases readability of the code, at least in my mind. Of course there are cases where you want to pass the data in a register, however I think the data directive approach is far to often over looked.

One of my favorite subroutines in programs these days is one to do a VMBW with data statements. It is as follows:

```
VMBWD   MOV   *R11+,R0
        MOV *R11+,R1
        MOV *R11+,R2
        BLWP @VMBW
        RT
```

And the calling form for this is:

```
        BL     @VMBWD
        DATA VDP ADDRESS
        DATA CPU_ADDRESS
        DATA BYTE_COUNT
```

This routine is extremely useful when putting text up on the screen, or shuffling data around for DSR acc[illegible]ther such VDP access. Similar subroutines can easily be written for other often called utility routines, as I have done for VWTR on occassion.

The JPH Assembler Hack

Have you ever been totally frustrated when using the TI Assembler program? When it gives you an error, have you ever said "Look, you know what the line looked like, can't I see it now?" Have you ever wanted to print out a listing of assembler errors rather than having to write them down? Have you ever wished you could pause the assembler output before it scrolled away? Are you sick of constantly entering the same Options over and over again? Did you ever wonder what the names of those undefined symbols were?

If you answered yes to any of these questions then have I got a deal for you! (proper intonation required) The JPH Assembler Hack will run on a 9640 or TI-99/4a with a Super-Cart. It provides convenient solutions to all the above questions without removing a single feature from the origianal TI Assembler. Once you start using this assembler you won't ever want to go back to the original.

It is being released as fairware, something I don't really believe in anymore. It just don't pay. Hoever, if you're interested, the files can be obtained from Delphi, and GEnie and I suspect soon on CompuServe as well. They are also available on BCS TI99 BBS #1 (617 331-4181) as ASSM. If you can't get it any other way send me a disk, return mailer, return postage, and if you're feeling generous the $10 fairware payment and I'll get it out to you. My address is 12 Paul Revere Road, Lexington, MA 02173. This will eventually be integrated into the BCS Software Library, but I don't have enough matching material to make a full disk yet.

Rumor Killing

Somewhere along the line I must have slipped. It seems that word got out that there was music buried somewhere in the MY-Word program. Some of the theories have been pretty interesting too. One holds that if you enter my birthday into the Reminder prompt (another mystery feature....), the program will play Happy Birthday. Others theories hold that on a particular date, say Christmas for example, that appropriate music will play.

I should make it clear that I dislike holiday type music as a general rule. And I'm not a terribly clever person. And the Reminder feature has quite a bit to do with a product that Genial Computerware markets. And about this music thing....well time will tell. And my birthday is 10/17/66.