Advertiser / Advertizer Module

In some words the Americans use a Z and the English use an S- computerise / computerize - but Advertiser (the word) is always an S in both the USA and the UK. Perhaps a European named this module....

Many thanks for the module photographs to Flottmann1.

Screen grabs are by me.

List of CALLS in alphabetic order.

This page contains documentation for a very early TI module developed around 1979 for the Texas Instruments TI-99/4 Home Computer. The module was unusual in NOT appearing on the computer menu screen as its sole purpose in life was to add new commands for use in TI Basic. Unlike the TI-99/4a the 99/4 did not have lower case, hence some special commands in Advertizer.
Written in GPL, examples are only known with EPROM contents (either 2x4k or 8k) and only the German language version is available to this writer. Versions of the module binary are available for the TI emulators in PC99 and Mess formats, for which it may have the unofficial identity of PHM3063. The module shown here is dated 1981, but we have Basic programs for the module originating from TI in France dated February 1980. This is another module which dislikes the liberties taken with the F18A Video board which has to use VDP ram differently to the normal VDP chip. Any modification changing VDP mapping or using VDP differently will cause problems with Advertizer.

The intent of the module was to make powerful GPL display subroutines available by simple CALLs in TI Basic programs, in a more complex format than normal TI Basic. The intended purpose was for advertising displays and interactive shop displays. Regular TI Basic programmers will find a double height font and the ability to have two different coloured fonts at the same time.

The module was written before a PAL computer console was available, and timing commands are based on 60Hz rather than the European 50Hz. As a consequence when running on a European console or an emulated European console, times will be a little slow.

Some elements originally baffled me! Official documentation then came to hand and have been used to update and extend this page- many thanks to KL and friends...]

Introduction and initialisation

Insert module.

Go to menu- you will only see "TI BASIC", this is as it should be (some emulators may show you "REVIEW MODULE LIBRARY" also).

select TI BASIC
enter CALL FILES(9)
enter NEW

[Upon power up, the module checks for 16k free VDP ram and if it finds it then creates an isolated area for its own use, safe from Basic. If however you have a disk controller attached (including emulators!) this is insufficient and you must increase the "safe" area with a CALL FILES(9) followed by NEW]

Now write your BASIC program or load a disk file eg DSK1.DEMO

NB You have a reduced memory for your basic program- about 9k.

---

note- **INITIALISATION IS REQUIRED** before using these calls.
all programs must start **CALL AD**
which can also be used in the middle of a program.
CALL AD will also clear the screen
AD CALL AD
Sets up memory- copies graphics to safe area of VDP etc, and MUST be first line of program.
Also resets VDP memory/built in graphics and fonts in a running program.

```
100 CALL AD
```

The command will fail INITIALISATION REQUIRED if VDP memory has not been prepared first (with a disk controller, using CALL FILES(9))

AUDIO:

CSOFF CALL CSOFF
CSON CALL CSON
Unable to test but probably meant to control an audio cassette remote. These calls are not mentioned in the official docs now to hand.

CAS CALL CAS
Places a message at the screen bottom, scrolling the display 3 lines,
"Press Play" "Then press ENTER"
These three CALLs take no parameters.

MUSIC CALL MUSIC(3)-
values from 1 to 12.
1 is Beep, 2 is Honk (or bloop), 3 C Major, 4 start, 5 End, 7 is from Yahtzee and 6 and 8 similar, 9 listing, 10 and 11 mistake. 12 is silent.

```
100 CALL AD
110 FOR C=1 TO 11
120 CALL MUSIC(C)
130 CALL PAUSE(5)
140 NEXT C
```

IMPORTANT NOTE ON GRAPHICS USAGE:
Advertiser makes use of three areas of memory which we will call the text area (uses Basic chars 32-95), the graphics area (uses basic chars 96-159), and a private area that can store ONE defined character for Single, Double (2x2) and Logo (4x4)- dealt with in more detail as they come up below. (Refer to DEFS, DEFD and DEFL below). There are also predefined characters (see CHARSET and GRAPSET)

Note that in many cases the 24 row screen x 28 columns (Or 32 graphics columns) is not fully used, noted against each relevant command below. When a command accepts a maximum of row of 20 or column of 26, if you exceed that number eg row 21 will be at screen top (row 1) and column=27 is at screen left (column 1).

D CALL D(M,R,C,L,M$)

Places normal text on screen.

M=mode: 1 put text on screen 2 text with chord, 3 text flashes 5 secs, 4 text flashes until key pressed, 5 letters in ripple, 6, 7 one letter at a time with note, 8 letters ripple and chord.

R row 1-20, C column 1-26

Note row 1 and column 1 are in the normal HCHAR positions, but if you use Row=24 the display will be at row 4, and if you use column 32 the display will be at column 6.

L Length to clear: if less than text the text will be curtailed, if longer than text the screen after the text will be cleared to the excess length. If negative will just write over screen positions for nominated length.

M$ is the message text.

A number, numeric formula (eg 4*4) or numeric variable is ok here.

note: The length of a number includes its sign but + is repressed- so that 2 is length 2 but displays as 2 (with a space to the left) while -2 is also length 2 and displays as -2.
If length is set too short for the number the left most part of the number is shown so that with length 2 the number 123 shows as 1 and with length 3 the number 435 shows as 43.
With length 4, number 12/10000 is displayed as .00 - the + sign and the decimal are included in the length.

The length parameter allows negative numbers and will not clear area of screen before placing text or number.

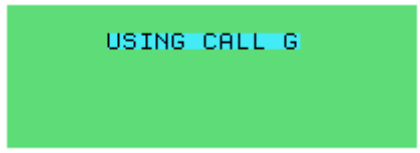
NOTE that CALL D uses the characters in the text definition area, that is Basic chars 32-95. However if you use a string with chars 96-156 and use CALL D to display them, the default definitions of chars 96-156 (eg GRAPSET(5) will be used.

```
100 CALL AD
110 CALL D(1,4,2,11,"HELLO WORLD")
120 CALL D(6,8,3,7,"MESSAGE")
130 CALL D(5,2,2,4,"LONG MESSAGE")
130 CALL HCHAR(10,1,42,256)
140 CALL D(1,10,3,17,"NO BLANK TO RIGHT")
150 CALL D(1,11,3,15,"BLANK TO RIGHT")
160 CALL D(7,18,4,2,52)
170 CALL PAUSE(10)
```

G CALL G(M,R,C,LEN(L$),L$)

Similar to CALL D but uses a separate character set making banners possible- differing fonts or colours.

Use CALL COLORG(B) to set the background colour of the banner.



M=mode: 1 put text on screen 2 text with chord, 3 text flashes 5 secs, 4 text flashes until key pressed, 5 letters in ripple, 6, 7 one letter at a time with note, 8 letters ripple and chord.

R row 1-20, C column 1-26

Note row 1 and column 1 are in the normal HCHAR positions, but if you use Row=24 the display will be at row 4, and if you use column 32 the display will be at column 6.

L Length to clear: if less than text the text will be curtailed, if longer than text the screen after the text will be cleared to the excess length. If negative will just write over screen positions for nominated length.

M$ is the message text.

A number, numeric formula (eg 4*4) or numeric variable is ok here.

Refer to CALL D.

NOTE that CALL G uses the characters in the graphics definition area, that is Basic chars 96-159. By default these will have default character definitions from GRAPSET(5). Use GRAPSET to set the characters to readable text.

```
100 CALL AD
110 CALL GRAPSET(1)
120 CALL COLORG(4)
130 CALL G(6,2,7,11,"HELLO WORLD")
140 CALL PAUSE(10)
```

DX2 CALL DX2(M,R,C,L,"TEXT HERE")

As D above BUT the text is duplicated using first the characters defined in chars 32-95, then below using chars 95-159.

allowing different colors and different FLASHing and double height text (top half defined in 32-95, bottom half defined in 96-159).

```
100 CALL AD
110 CALL CHARSET(3)
120 CALL GRAPSET(1)
130 CALL COLORG(4)
140 CALL DX2(1,4,2,11,"HELLO WORLD")
150 CALL FLASH(1,3,4,1)
160 CALL FLASH(1,3,5,2)
170 CALL FLASH(1,3,4,3)
180 CALL PAUSE(10)
```

110 changes normal text to lower case (top text using DX2)

120 makes graphics text upper case (bottom text using DX2)

130 changes background colour of graphics text light blue
140 displays HELLO WORLD twice, from row 1 col 4 and also row 2 col 4
150 flashes the normal text- top text using DX2
160 flashes the graphics text - bottom row using DX2
170 flashes both normal and graphics text together

and

```
100 CALL AD
110 CALL GRAPSET(3)
120 CALL COLORG(3)
130 CALL DX2(2,5,3,5,"SEVEN")
140 CALL PAUSE(3)
150 CALL DX2(1,9,3,7,"LETTERS")
160 CALL PAUSE(3)
170 CALL COLORG(4)
180 CALL PAUSE(3)
190 CALL GRAPSET(1)
200 CALL PAUSE(2)
210 CALL CHARSET(3)
200 CALL PAUSE(4)
```

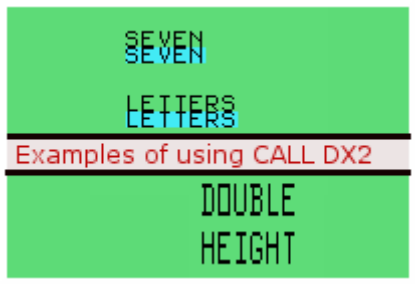
110 changes graphics text (lower text using DX2) to lower case
120 changes graphics text back ground colour to green
170 changes graphics text background to light blue
190 changes graphics text (lower text using DX2) to upper case
210 changes normal text (upper text using DX2) to lower case

also

Double Height Text

If you use CALL DX2 in this manner (double height):
- you CAN use it several times on a page.
- you cannot also use CALL TEXT
- it will corrupt the characters used for normal text (eg CALL D) and
- it will corrupt the characters used for graphics text (eg CALL G).

```
100 CALL AD
110 CALL GRAPSET(4)
120 CALL COLORG(3)
130 CALL DX2(1,5,5,6,"DOUBLE")
140 CALL DX2(1,8,8,6,"HEIGHT")
150 CALL DX2(1,11,10,7,"LETTERS")
160 CALL PAUSE(5)
```



SEE ALSO CALL FLASH; CALL GRAPSET ; CALL CHARSET ; CALL COLORG

TEXT CALL TEXT(18,8,L$) or CALL TEXT(4,2,"TEST")

eg CALL TEXT(row,col,"text")

Rows 1 to 22, columns 1 to 28 - not the same limits as CALL D and CALL G..

Displays DOUBLE HEIGHT normal text (eg two rows high), but unlike DX2 above, will only use redefined chars in the text area (chars 32-95) leaving the graphics set available for CALL G. Any characters 32-95 visible on the screen will be changed.

As only one call is permitted per page, line breaks are inserted into the string with the hash (#) character eg "THIS IS#TWO LINES". Maximum 31 different chars other than space (Alphabet=26!) - excess will give STRING OVERFLOW.

May not be used to display numbers or numeric variables which must be converted to a string using STR$(N).
Restricted to one use per screen
NOTE: Redefines characters (32-95) used for text in CALL HCHAR, PRINT, CALL D and any other use on the same screen of CALL TEXT.

```
100 CALL AD
110 CALL TEXT(5,1,"HELLO WORLD")
120 CALL PAUSE(4)
130 PRINT:"TI ADVERTISER MODULE"
140 CALL PAUSE(10)
```

OR

```
100 CALL AD
110 CALL TEXT(5,1,"HELLO WORLD")
120 CALL PAUSE(4)
130 CALL CLEAR
130 CALL TEXT(6,2,"HELLO AGAIN")
140 CALL PAUSE(10)
```

note: You can also use CALL DX2 to display double height text using the text and graphics areas- see CALL DX2. CALL DX2 can be used several times on one screen but will redefine all characters 32-159 when used for the double height font.

DISPLAY SUMMARY
CALL D- place normal text
CALL G- place "graphics" text
CALL DX2- one command to place both normal and graphics text OR place double height text multiple times. Double height will corrupt single height text on screen.
CALL TEXT- place double height text ONCE per screen, can be used with CALL G.
CALL HCHAR and PRINT also still available.

SCROLL CALL SCROLL(LINES)

Scrolls the display upwards by LINES (1-32) number of screen lines.

```
100 CALL AD
110 PRINT "HELLO WORLD"
120 CALL PAUSE(5)
130 CALL SCROLL(5)
140 CALL PAUSE(10)
150 CALL SCROLL(12)
160 CALL PAUSE(10)
```

Defining graphics:

You can define characters using prestored character sets which are switched using CHARSET for characters in the text area (Chars 32-95), GRAPSET for characters in the graphics area (chars 96-159) and PREDEFS (char 96), PREDEFD (chars 96-99) and PREDEFL(chars 96-111). Refer to these below. Note that redefining some characters for one use will prevent your using that area of graphics characters for other purposes.

CHARSET CALL CHARSET(V)

Amends font for "normal" (CALL D) text (Chars 32-95) all at once.(Note special case for set 4).

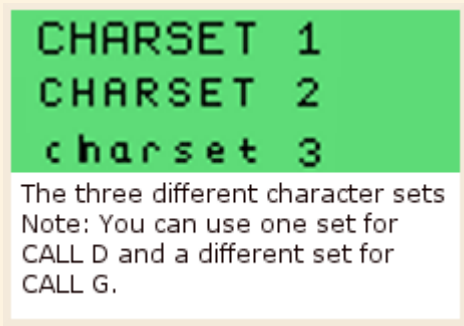
V=1 larger all caps text

V=2 smaller all caps text

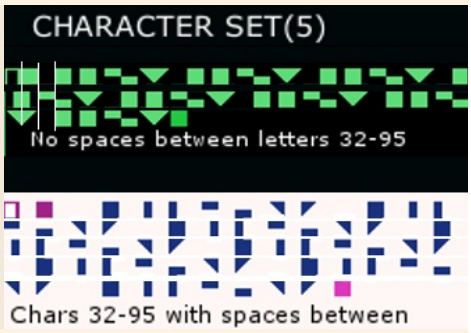
V=3 displays text in lowercase

V=4 displays text in double size (when using CALL DX2)

V=5 and V=6 install predefined graphics (2 different sets).



The three different character sets
Note: You can use one set for CALL D and a different set for CALL G.

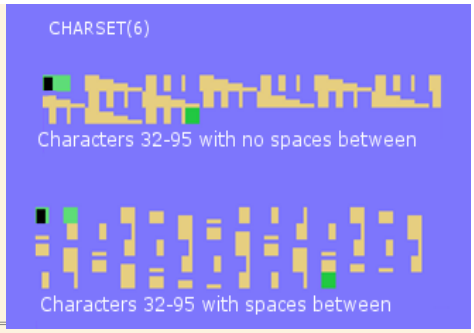



```
100 CALL AD
110 CALL D(1,1,1,10,"FIRST LINE")
120 FOR T=1 TO 3
130 A$=CHARSET "&STR$(T) 140 CALL CHARSET(T)
150 CALL D(1,3*T,3,9,A$)
160 NEXT T
170 CALL PAUSE(10)
```

Note that if you do not define a character set to use the default is the V=1 8x8 large characters.

CALL CHARSET(4) is for use with CALL DX2, **it redefines all characters 32 to 156**

```
100 CALL AD
110 CALL CHARSET(4)
120 CALL DX2(1,3,1,6,"DOUBLE")
130 CALL DX2(5,4,1,6,"HEIGHT")
140 CALL PAUSE(5)
```



GRAPSET CALL GRAPSET(M)

Defines the graphics font, eg use with CALL G. (chars 96-156) using predefined sets.

M=1 uses a capital font,

M=2 small capitals font,

M=3 sets lower case.

Set 4 is not for use.

M=5 or 6 loads one of two graphics sets.

(illustrated under CHARSET above)

NOTE: Default charset for chars 96-156 is that of GRAPSET(5)

```
100 CALL AD
110 CALL COLORG(3)
120 CALL G(1,1,1,10,"FIRST LINE")
130 FOR T=1 TO 3
140 A$=GRAPSET "&STR$(T)
150 CALL GRAPSET(T)
160 CALL G(1,3*T,3,9,A$)
170 NEXT T
180 CALL PAUSE(10)
```

LIST CALL LIST(W,R,C,CH,A1$,A2$,...A9$,A10$)

Slowly lists the strings on screen, double line spacing, and if time period >1 then waits for ENTER or timeout.

W is time in seconds (MAX 99) to wait before continuing, curtailed by pressing ENTER.

R and C is starting row (1 to 20) and column (1 to 26) - display is left aligned to column C.

CH is the character to use for the bullet point (32 to 159)

Strings can be array elements, variables or literals.

LIST uses the characters 32-95. There is a special control character, hash # which will make the display for the listed entry continue on next line (double line spacing used), eg "ITEM FIVE#IS ON TWO LINES"- text after the # do not receive a bullet point.

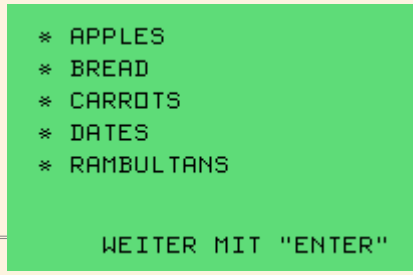
If time is more than 1 then final line of screen will show "WEITER MIT ENTER" and waits for W seconds OR the enter key if earlier, then proceeds to next command or more items. If

time=1 the program will continue, and if there are more items than will fit, the display will scroll.

As each item is placed on screen, a little bit of music is played.

Maximum displayed items per screen is restricted to 10 due to screen size. If you start at a lower row or have more items, the LIST will fit as many as as possible, invite you to press enter if time>1, list8then blank the screen and continue the list. NB The bullet point on the top item of page 2 may not show.

```
100 CALL AD
110 CALL LIST(30,2,2,42,"FIRST ITEM","SECOND
ITEM","THIRD ITEM")
120 CALL D(1,18,3,13,"LIST FINISHED")
130 CALL PAUSE(10)
```



Go to top of page

INPUT:

A CALL A(W,R,C,L,S,N) or CALL A(W,R,C,D,S,N$)

W = wait in seconds (max 99) before giving up and moving on, when S becomes value=2

R (1-20) and C (1-26) = row and column for input

L = maximum number of digits to enter for numeric input or number of characters for string input.

If L is positive number, it clears position and looks for number key to be pressed followed by ENTER within the wait time.

if L is negative, it will pick up what is on screen at that location if ENTER is pressed, OR waits for number key(s) followed by ENTER..

S=status,

S=1 if during the Wait no number key is pressed just ENTER (eg nul input)

S=0 if during the wait a number key is pressed followed by enter

S=2 if no key is pressed during wait period.

N or N$ (return variable) = Variable keyed in followed by ENTER during the wait period.

NOTE the time out and use the status if required to redo the command and avoid false input.

note: If no entry is made, the variable will RETAIN any previous value - it will only be zero or an empty string if that was its condition before the CALL A. This allows a "drop through" value to be set if ENTER is not pressed and a timeout occurs.
Please read the final note carefully.

```
100 CALL AD
110 CALL D(1,1,2,14,"ENTER A NUMBER")
120 CALL A(30,2,14,2,S,N)
130 IF S<>0 THEN 120
140 CALL D(1,5,6,15,"ENTER YOUR NAME")
150 CALL A(30,6,12,12,S,N$)
160 IF S<>0 THEN 130
170 A$=N$&" ENTERED "&STR$(N)
180 CALL D(1,8,2,22,A$)
190 CALL D(1,10,1,24,"NOW JUST PRESS ENTER AND")
200 CALL D(1,11,1,20,"COMPUTER WILL READ 4")
210 CALL HCHAR(13,1,52,256)
220 N=2
230 CALL A(8,14,1,-1,S,N)
240 CALL HCHAR(13,1,32,256)
250 A$="NUMBER ENTERED:"&STR$(N)
260 CALL D(1,18,3,18,A$
270 CALL PAUSE(12)
```

Lines 210-230: If you press enter, the number 4 will be picked up from the screen. If you press nothing, N will retain a value of 2. If you press a number but NOT enter, the value of N will not change. If you press a number and enter, the variable will take your value.

NOTE: The official documentation refers to CALL ACCEPT however the module image in circulation uses CALL A.

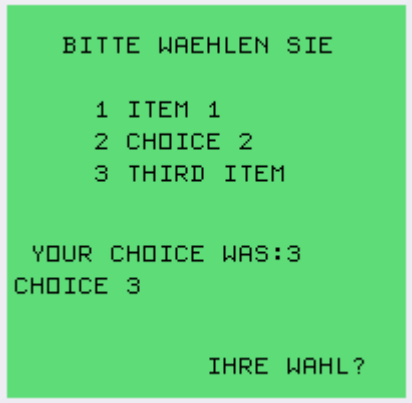
Usual editing keys on the TI: DEL=FCTN 1; INS=FCTN 2; CLR=FCTN 4; move left=FNCT S; move right=FNCT D.

MENU CALL MENU(V,D,R,C,M1$,M2$....M9$)

V = Must be a variable used to return the users choice from the menu.
D = Delay in seconds (max 99) before program execution passes on.
You may need to check the returned variable V is valid before processing.
R (max 20), C (max 26) = start Row / Column to display "Press For" (BITTE WAEHLEN SIE)
Row 1 Column one is the same as CALL HCHAR(1,1...
M1$-M9$ - Up to 9 strings. Hash # inserts a line feed eg "ONE#TWO".

The display for this command uses double line feeds. This command places "Please choose" at the top of the screen, then slowly, with tones, adds the menu items with a number to their left. The entries are placed every other line. At the end of the screen "Your choice" appears.

The command then waits up to D seconds for you to press a number key. If there are 4 menu items the computer will ignore numbers other than 1-4.



NOTE: If no choice is made the variable V will be set to zero.

At the bottom of the screen the command places the words "Your choice?". You only need to press a number key- enter is not required.

If you use more items than will fit on the screen, as the program runs, you will receive a "PAGE OVERFLOW" error.

```
100 CALL AD
110 CALL D(1,20,4,15,"SELECT 1 2 OR 3")
120 CALL MENU(V,12,3,4,"FIRST CHOICE","CHOICE 2","THIRD ITEM")
130 IF V<>0 THEN 160
140 CALL HCHAR(8,1,32,20*32)
150 GOTO 120
160 A$="YOUR CHOICE WAS:"&STR$(V)
170 CALL D(1,16,2,18,A$)
180 ON V GOTO 190,210,230
190 PRINT "FIRST CHOICE"
200 GOTO 200
210 PRINT "CHOICE 2"
220 GOTO 220
230 CALL D(1,18,1,8,"CHOICE 3")
240 CALL PAUSE(10)
```

ENTER CALL ENTER(SECS) Maximum time is 99 seconds.

Prints at bottom of screen 'Weiter Mit "ENTER"

' Translation: CONTINUE WITH ENTER

Waits for (SECS) seconds and then continues with next command OR

if you press ENTER continues immediately.

The time variable must be between 1 and 99. A number or numeric variable can be used.

CALL PAUSE is only for small periods (1-16 sec), but does not write to the screen and several CALL PAUSE commands can be used one after the other.

```
100 CALL AD
110 PRINT "THE COMPUTER WILL NOW WAIT 20 SECONDS": "OR PRESS ENTER
NOW"
120 CALL ENTER(20)
130 PRINT "YOU PRESSED ENTER OR 20 SECONDS PASSED"
140 CALL PAUSE(10)
```

PAUSE CALL PAUSE(T)

Waits for time in seconds from 1 to 16 maximum.

```
100 CALL AD
110 CALL PAUSE(10)
120 PRINT "10 SECS"
130 CALL PAUSE(6)
```

FLASH CALL FLASH(R,C,N,T)

Flashes text (normal text, graphics text or both together)

R = Flash Speed in range 1 to 5, 1 is fast

C = Control range 1-3.

C=1 flash until ENTER is pressed

C=2 flash for given number of flashes N and move on

C=3 Flash for given number of flashes N OR until space is pressed.

N = Number of flashes (seconds- max 120)

T = Type of text to flash:

T=1 flash normal text (eg CALL D, chars 32-95)

T=2 flash graphics text (eg CALL G, chars 96-156)

T=3 flash both normal and graphics text together.(chars 32-156)

```
100 CALL AD
110 CALL CHARSET(3)
120 CALL GRAPSET(1)
130 CALL COLORG(4)
140 CALL DX2(1,4,2,11,"HELLO WORLD")
150 CALL FLASH(1,3,4,1)
160 CALL FLASH(1,3,5,2)
170 CALL FLASH(1,3,4,3)
180 CALL PAUSE(10)
```

110 changes normal text to lower case (top text using DX2)

120 makes graphics text upper case (bottom text using DX2)

130 changes background colour of graphics text light blue

use CALL COLORG(3) for both texts with same background colour.

140 displays HELLO WORLD twice, from row 1 col 4 and also row 2 col 4

150 flashes the normal text- top text using DX2

160 flashes the graphics text - bottom row using DX2

170 flashes both normal and graphics text together

COLORC CALL COLORC(T) or CALL COLORC(T,V2,V3)

CALL COLORC immediately changes the colour of ALL characters 32-95. In some cases it can change the colours of all characters 32-156. Some uses take one parameter and some uses use three parameters.

One parameter:
1 = random fg and bg colours for chars 32-95
2 = random fg and bg colours for chars 32-156
3 = all fg and bg colours to Character set 1 values
4 = Foreground to black, background to cyan
5 = rotates colour values:
  If all colour sets are the same, does nothing. If colour sets vary, each call will rotate the values. Very rapid flicker if you don't also insert delay loops.
6 = rotates colour values for chars 32-156
7 = Sets all BG to same as for colour set 1
8 = predefined colour set (see below)
9 = Special use for CALL LOGO- ("making all even F/B the same as the odd F/B preceeding")
10= Special use requires three parameters,, see below

Note: LOGO is 4x4 characters, thus it will always use two contiguous colour sets, which each have 8 characters. Thus using parameter 9 ensures that all parts of a LOGO will be the same two colours.

CALL COLORC(8) will set the 8 colour sets for chars 32-95 as follows-
charset/foreground/background:
1:3/4 2:6/11 3:9/8 4:14/12 5:10/3 6: 5/15 7:15/9 8:9/15

Three parameters:
CALL COLORC(10,F,B) - immediately set chars 32-95 for specified foreground and background colours.
Colours: 1=transparent 2=black 3=medium green 4=light green 5=dark blue 6=light blue 7=dark red 8=cyan 9=medium red 10=light red 11=dark yellow 12=light yellow 13=dark green 14=magenta 15=grey 16=white

You can continue to use CALL COLOR in TI Basic to set individual colour sets.
Reminder of colour sets - set number and (character numbers in the set):
1 (32-39); 2 (40-47); 3 (48-55); 4 (56-63); 5 (64-71); 6 (72-79); 7 (80-87); 8 (88-95)
Reminder: In TI Basic you cannot amend Colour Set 0, the edge and cursor characters.

COLORG CALL COLORG(B) or CALL COLORG(V1,V2,V3)

Sets the colours for characters 96-156. Some commands will affect chars 32-156. One or three parameters are used- see below.

One parameter:

1 = random fg and bg colours for chars 96-156

2 = random fg and bg colours for chars 32-156

3 = all fg and bg colours to Character set 1 values

4 = Foreground to black, background to cyan

5 = rotates colour values:

If all colour sets are the same, does nothing. If colour sets vary, each call will rotate the values. Very rapid flicker if you don't also insert delay loops.

6 = rotates colour values for chars 32-156

7 = Sets all BG to same as for colour set 1

8 = predefined colour set (see below)

9 = Special use for CALL LOGO ("making all even F/B the same as the odd F/B preceeding")

10= Special use requires three parameters,, see below

Note: LOGO is 4x4 characters, thus it will always use two contiguous colour sets, which each have 8 characters. Thus using parameter 9 ensures that all parts of a LOGO will be the same two colours.

CALL COLORG(8) will set the 8 colour sets for chars 32-95 as follows-
charset/foreground/background:
1:3/4 2:6/11 3:9/8 4:14/12 5:10/3 6: 5/15 7:15/9 8:9/15

Three parameters:

CALL COLORG(10,F,B) - immediately set chars 32-95 for specified foreground and background colours.

Colours: 1=transparent 2=black 3=medium green 4=light green 5=dark blue 6=light blue 7=dark red 8=cyan 9=medium red 10=light red 11=dark yellow 12=light yellow 13=dark green 14=magenta 15=grey 16=white

You can continue to use CALL COLOR in TI Basic to set individual colour sets. Reminder of colour sets - set number and (character numbers in the set):

1 (32-39); 2 (40-47); 3 (48-55); 4 (56-63); 5 (64-71); 6 (72-79); 7 (80-87); 8 (88-95)

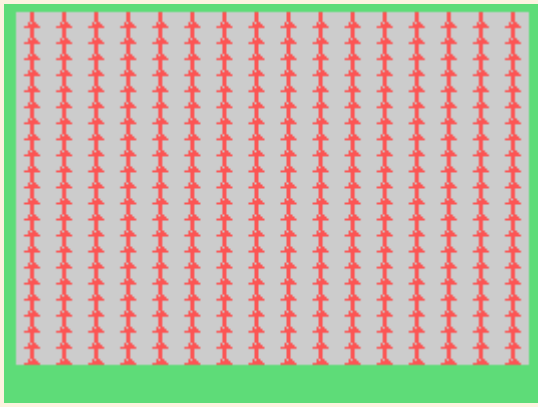
S CALL S(M,S,A)

S for screensaver? Exits with ENTER or timeout.

The graphic used is from the graphics set or the stored LOGO (see below and CALL LOGO, CALL DEFL)

Important: to use stored characters single, double, logo eg in CALL S, REFER to LOADS, LOADD, LOADL.

Fills the screen with a graphic pattern using animation style A (1-10)
M = Mode
M= 1 until FIRE or ENTER is pressed
M= 2 for period S
M= 3 for the period S, or pressing ENTER to end.
S=time in seconds (Max 120) (when M=1, if S=1. unending until key press)
A=Animation type (1 to 10):
1-flashing lights 2-flashing lights and music 3-LOGO stripes (diagonal top left to botto right) 4-as 3 with music 5-Random logo stripes(diagonal top right to bottom left) 6-as 5 with music 7-Logo curtain down 8-as 7 with music 9-Logo strip up 10-as 9 with music.



ALSO SEE CALL PREDEFL, CALL LOADL, CALL COLORC(10... and CALL COLORG. You can define the character used by using CALL CHAR with chars 96-111.

NOTE: If you use CALL S without prior use of LOGO or storing a LOGO character, the default used for the LOGO used by CALL S is taken from the graphic character set obtained with CALL GRAPSET(5). Using CALL S like this does not redefine the character set used with CALL G.
If you use CALL LOGO then CALL S without further defining characters, the LOGO used for CALL S will be the default LOGO of a STAR.

```
100 CALL AD
110 FOR T=3 TO 7
120 CALL S(1,6,T+2)
130 CALL CLEAR
140 NEXT T
150 CALL PAUSE(10)
```

and (next page...)

```
100 CALL AD
110 CALL CHAR(66,"1010101010101010")
120 CALL CHAR(67,"FF00FF00FF00FF00")
130 CALL DEFL(66,66,66,66,66,66,66,66,67,67,67,67,67,67,67,67)
140 CALL LOADL(1)
150 CALL S(1,6,6)
160 CALL PAUSE(4)
```

In the above code we define chars 66 and 67 (could be any characters), then store these definitions to make up a 4x4 character LOGO using DEFL. This does NOT at this stage amend chars 96-111. As CALL S uses the definitions of characters 96-111, we then copy the stored LOGO definition from private storage to characters 96-111 by using CALL LOADL.

Instead of using a stored definition we can just use CALL CHAR to redefine all 16 characters 96-111.

LOGO CALL LOGO(ROW,COL,Q)

Displays a LOGO (4x4 chars, chars 96-111). Used with CALL LOADL, CALL PREDEFL, CALL DEFL

Places a 4x4 graphic at Row (max 20), Colum (Max 26)- where R=1, C=1 are the location of CALL HCHAR(1,1...

Parameter Q can be either 1 or 2.

Q=1 Load the stored LOGO into chars 96-111

The Default LOGO is a star shape.

Q=2 Use the existing definition of chars 96-111 -note if not previously defined, the default 4x4 character will be made up of characters from GRAPSET(5). You may use eg CALL CHAR to redefine chars 96-111 etc.

Refer to CALL PREDEFL, CALL DEFL

```
100 CALL AD
110 CALL LOGO(2,2,1)
120 CALL PAUSE(4)
```

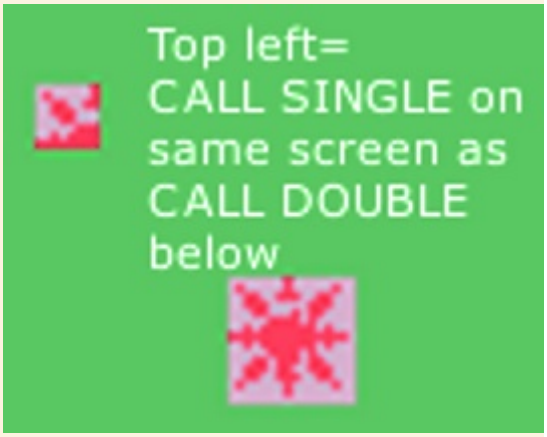
If the third parameter is set to 2, a different LOGO is placed.

```
100 CALL AD
110 FOR T=1 TO 5
120 CALL PREDEFL(T)
130 CALL LOGO(2,3,1)
140 CALL PAUSE(2)
150 NEXT T
```

AND (next page)

```
100 CALL AD
110 CALL CHAR(66,"1010101010101010")
120 CALL CHAR(67,"FF00FF00FF00FF00")
130 CALL DEFL(66,66,66,66,66,66,66,66,67,67,67,67,67,67,67,67)
140 CALL LOADL(1)
150 CALL LOGO(2,3,1)
160 CALL PAUSE(4)
```

DOUBLE CALL DOUBLE(ROW, COL, T)



Max Row=22, Max Col=26 where R=1 C=1 is the location of CALL HCHAR(1,1...
T=1 load the stored DOUBLE (2x2 chars, char 96-99) - changes definitions for chars 96-99.
Default is a star shape.
T=2 use the existing definitions for chars 96-99 . You can change the definitions using eg CALL CHAR. If not already defined the default chars are from GRAPSET(5).

Refer to CALL DEFD.

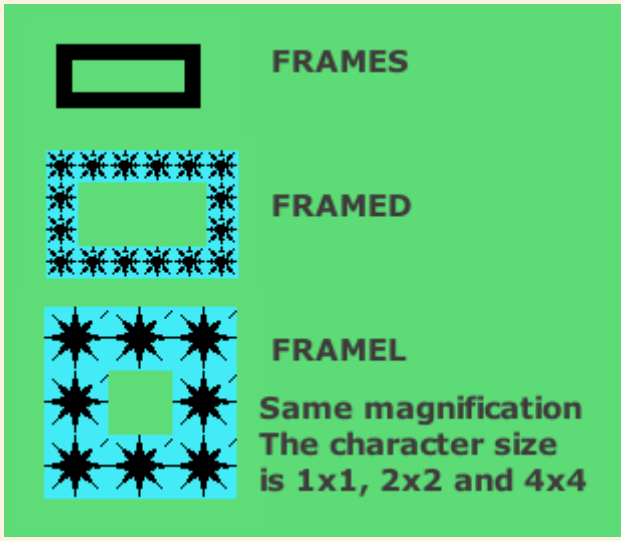
SINGLE CALL SINGLE(R,C,T)
Places a single graphics character on the screen

T=1 load the stored SINGLE (1 chars, char 96)
Default= star. This changes char 96.
T=2 use the existing definition for char 96
> Default is char 96 from GRAPSET(5) which can be redefined with CALL CHAR.
Refer to CALL DEFS.
Max Row=22, Max Col=26 where R=1 C=1 is the location of CALL HCHAR(1,1...
T may be 1 or 2.

```
100 CALL AD
110 CALL DEFS(65)
120 CALL SINGLE(6,7,1)
130 CALL PAUSE(4)
```

Draw a frame

Uses a SINGLE (char 96) (FRAMES), a DOUBLE(chars 96-99) (FRAMED) or a LOGO (chars 96-111) (FRAMEL) to draw a frame with randomly changing colour bands.



Important: to use stored special graphic single, double, logo, refer to LOADS, LOADD , LOADL.

eg FRAMES CALL FRAMES(E,S,T,R,C,W,H)

E=1 until enter is pressed

E=2 until time period ends

E=3 until timeout or enter.

S=Period in seconds max 120.

T=1 random selection

T=2 sequential selection

T=3 whole band then sequential

Row max 20, Column max 26 where r=1 c=1 is the position of CALL HCHAR(1,1...

W=width being the number of patterns horizontally (note D=2x2 and L=4x4). Max C+W=32. Overflow truncated.

H=height, number of patterns vertically. Max R+H=24.

```
100 CALL AD
110 CALL COLORG(4)
120 CALL FRAMES(1,3,3,4,4,22,22)
```

Will repeatedly draw a frame in a sequence of solid colours until space is held..

Also see CALL COLORG, CALL DEFS, CALL LOADS...

FRAMED CALL FRAMED(E,S,T,R,C,W,H)

FRAMED is twice the thickness of FRAMES

FRAMES(E,S,T,R,C,W,H)

E=1 until enter is pressed

E=2 until time period ends

E=3 until timeout or enter.

S=Period in seconds max 120.

T=1 random selection

T=2 sequential selection

T=3 whole band then sequential Row max 20, Column max 26 where r=1 c=1 is the position

of CALL HCHAR(1,1...
W=width being the number of patterns horizontally (note D=2x2 and L=4x4). Max C+W=32. Overflow truncated.
H=height, number of patterns vertically. Max R+H=24. Overflow truncated.
Refer to CALL DEFD
100 CALL AD
110 CALL FRAMED(1,2,3,4,4,22,22)
Will repeatedly draw a bicolour frame in a sequence of colours until space is held..

For something a little different add
105 CALL PREDEFD(3)

OR TRY
105 CALL LOADD(3)
and stabilise it with
106 CALL COLORG(4)

Also see CALL COLORG, CALL DEFD, CALL LOADD...

FRAMEL CALL FRAMEL(E,S,T,R,C,W,H)

Larger graphic than FRAMES or FRAMED - 4x4 chars.
E=1 until enter is pressed
E=2 until time period ends
E=3 until timeout or enter.
S=Period in seconds max 120.
T=1 random selection
T=2 sequential selection
T=3 whole band then sequential Row max 20, Column max 26 where r=1 c=1 is the position of CALL HCHAR(1,1...
W=width being the number of patterns horizontally (note D=2x2 and L=4x4). Max C+W=32. Overflow truncated.
H=height, number of patterns vertically. Max R+H=24. Overflow truncated.
Refer to CALL DEFL.

100 CALL AD
110 CALL FRAMEL(1,2,3,4,4,22,22)
Will repeatedly draw a bicolour frame in a sequence of colours until space is held..
Also see CALL DEFL and CALL PREDEFL.

Draw box using a single character char 96 (BOXS) a 2x2 character chars 96-99 (BOXD) or a 4x4 character chars 96-111 (BOXL) :

BOXS CALL BOXS(S,R,C,W,H)
BOXD CALL BOXD(S,R,C,W,H)
BOXL CALL BOXL(S,R,C,W,H)

S=1 use defined SINGLE, DOUBLE, or LOGO- characters 96(S), 96-99(D) or 96-111(L).
S=2 load and use the stored SINGLE, DOUBLE or LOGO - default is a star. Refer to CALL DEFS, CALL DEFD, and CALL DEFL. Characters 96-111 can be redefined using CALL CHAR. Using S=2 will change characters 96-111.

Fills a rectangular shape on the screen
R=top row C=left hand column W=width H=height

Use CALL COLORG to set colours- you can have either a patterned box using two colours or a block box by setting background and froeground the same.

```
100 CALL AD
110 CALL BOXS(1,1,1,30,23)
120 CALL PAUSE(6)
130 CALL BOXD(2,4,4,26,19)
140 CALL PAUSE(6)
150 CALL BOXL(3,9,9,31,14)
160 CALL PAUSE(10)
```

Try adding 125 CALL LOADD(2); then try also adding 105 CALL LOADS(1) and see how the top left of the large graphic is used in BOXS when you reach LOADD.

DEFINE GRAPHICS:
Usual CALL CHAR may be used to define individual characters. Here we are talking about SINGLE, DOUBLE and LOGO characters, accessing predefinitions and storing our own definitions. Double is 2x2 chars and LOGO is 4x4 chars.

DEFS CALL DEFS(CHAR)
DEFD CALL DEFD(V1,V2,V3,V3)
DEFL CALL DEFL(char1,char2,char3,char4...char16)

These routines allow you to store special graphics characters referred to as SINGLE, DOUBLE and LOGO. We first define 1, 4 or 16 characters (any!) then use DEFS... to define the SINGLE to match those characters. The characters can be redefined but SINGLE will not change unless we use DEFS... again. Definitions stored in this way are NOT stored as

definitions for chars 32-156.
Double is 2x2 chars and LOGO is 4x4 chars.

```
100 CALL AD
110 CALL CHAR(65,"F1F1F1F1F1F1F1F1")
120 CALL DEFD(65,65,65,65)
130 CALL CHAR(65,"0000FFFF000000FF")
140 CALL DOUBLE(5,8,1)
150 CALL PAUSE(7)
```

Line 110 defines character 65.
Line 120 transfers this definition to the special character DOUBLE. (We can define 4 different characters and use say CALL DEFD(96,97,98,99)
Line 130 removes the prior definition of character 65, but leaves the old definition of DOUBLE.
Line 140 uses the stored DOUBLE and places it in row 5, column 8.

DEFL CALL DEFL(char1,char2,char3,char4...char16)

Define LOGO in graphics- 16 characters are used (can all be same character).
Use CHAR to define each char;
Also used to define the character for CALL S

```
100 CALL AD
110 CALL CHAR(66,"1010101010101010")
120 CALL CHAR(67,"FF00FF00FF00FF00")
130 CALL DEFL(66,66,66,66,66,66,66,66,67,67,67,67,67,67,67,67)
140 CALL LOGO(2,3,1)
150 CALL PAUSE(4)

and

100 CALL AD
110 CALL CHAR(66,"1010101010101010")
120 CALL CHAR(67,"FF00FF00FF00FF00")
130 CALL DEFL(66,66,66,66,66,66,66,66,67,67,67,67,67,67,67,67)
140 CALL S(1,6,6)
150 CALL PAUSE(4)
```

To transfer PRE DEFINITIONS FOR GRAPHICS:
where S=single, D=double, L=LOGO
PREDEFD CALL PREDEFD(N)
PREDEFS CALL PREDEFS(N)
PREDEFL CALL PREDEFL(3)

Valid values 1-5

Causes a graphic predefined in the module to be used for uses single, double and logo. eg CALL LOGO and CALL FRAMEL and CALL BOXL etc

```
100 CALL AD
110 FOR T=1 TO 5
120 CALL PREDEFL(T)
130 CALL LOGO(2,3,1)
140 CALL PAUSE(2)
150 NEXT T
```

You can define characters using CALL CHAR, can quickly define sets of characters with CALL GRAPSET and CALL CHARSET, but in addition there are 5 predefined graphics which you can allocate to the special graphics single, double and logo.

The five predefined graphics for single and double are: 1-star; 2-diamond; 3-Harlequin; 4-dots; 5-spots.

The five predefined graphics for logo are: 1-star; 2-diamond; 3-lady; 4-man; 5-checkers.

LOADD CALL LOADD(N)
LOADS CALL LOADS(N)
LOADL CALL LOADL(N)
N can have values 1, 2 or 3.

This command allows wider access to the stored special characters single, double, and logo, eg to use with CALL S.

N=1: transfer stored character to char set 9.
N=2: transfer stored character to all char sets 9-16
N=3: transfer stored character to char sets 11-16 - required to use in CALL FRAME.

```
100 CALL AD
110 CALL CHAR(66,"1010101010101010")
120 CALL CHAR(67,"FF00FF00FF00FF00")
130 CALL DEFL(66,66,66,66,66,66,66,66,67,67,67,67,67,67,67,67)
140 CALL LOADL(1)
150 CALL S(1,6,6)
160 CALL PAUSE(4)
```

MULTIPLE WAYS OF PLACING TEXT ON SCREEN:

```
100 CALL AD
110 CALL CHARSET(1)
120 CALL GRAPSET(1)
130 CALL COLORG(4)
140 PRINT "PRINTED TEXT"
150 CALL D(1,4,2,11,"CALL D TEXT")
160 CALL G(1,11,2,11,"CALL G TEXT")
170 FOR T= 65 TO 90
180 CALL HCHAR(18,T-62,T)
190 NEXT T
200 CALL PAUSE(5)
210 CALL TEXT(8,5,"CALL TEXT")
220 CALL PAUSE(4)
230 CALL CLEAR
240 CALL CHARSET(4)
250 CALL COLORG(3)
260 CALL DX2(1,14,2,13,"CALL DX2 TEXT")
270 CALL DX2(1,18,2,8,"COMPUTER")
280 CALL PAUSE(5)
```

Note line 210 causes "normal" text to be corrupted.

See CALL D, CALL DX2, CALL G, CALL TEXT, CALL CHARSET, CALL GRAPSET, CALL COLORG.

Downloads

NOTE: This module is VERY particular about the state of the vdp chip and does not seem to work on consoles which have been fitted with the replacement F18A video board, which does seem to cause several difficulties.

FG99 **For FinalGrom99:**
The binary has been modified for fg99 as the original module did not have either a module entry or an autostart program, one of which fg99 needed.

Place the bin file on your SDHC card in the usual manner.
- select 2 for Finalgrom
- select ADVERTISER
IF you have no disk controller, the screen will turn purple (the Advertiser module does this) and a TI Title screen with purple borders appears. (See note below re compatability) If you have a disk controller you don't get the purple.

Select 1 for TI Basic and you will get a BASIC READY.
If there is a disk controller attached - type CALL FILES(9) then NEW then CALL AD then CALL MUSIC(7)
With no disk controller- from the command line type CALL AD then CALL MUSIC(7)
The binary meets TI specifications and works fine with FG99 Vn 1.0 and 1.1 on an unmodified console. The binary REQUIRES that the VDP memory 100% conforms to the TI Original in all respect, and using a more modern video chip (eg F18a) will cause a variety of problems which may or may not affect you depending on how you use Advertizer. This is not a bug in the binary. Any variation in VDP memory mapping or usage will cause problems in this module which makes heavy use of VDP processes.

This is the change made to the original grom image to add to FG99 menu and jump to title screen- original first then new:

Adding menu entry and jump to Advertiser image:

ORIGINAL

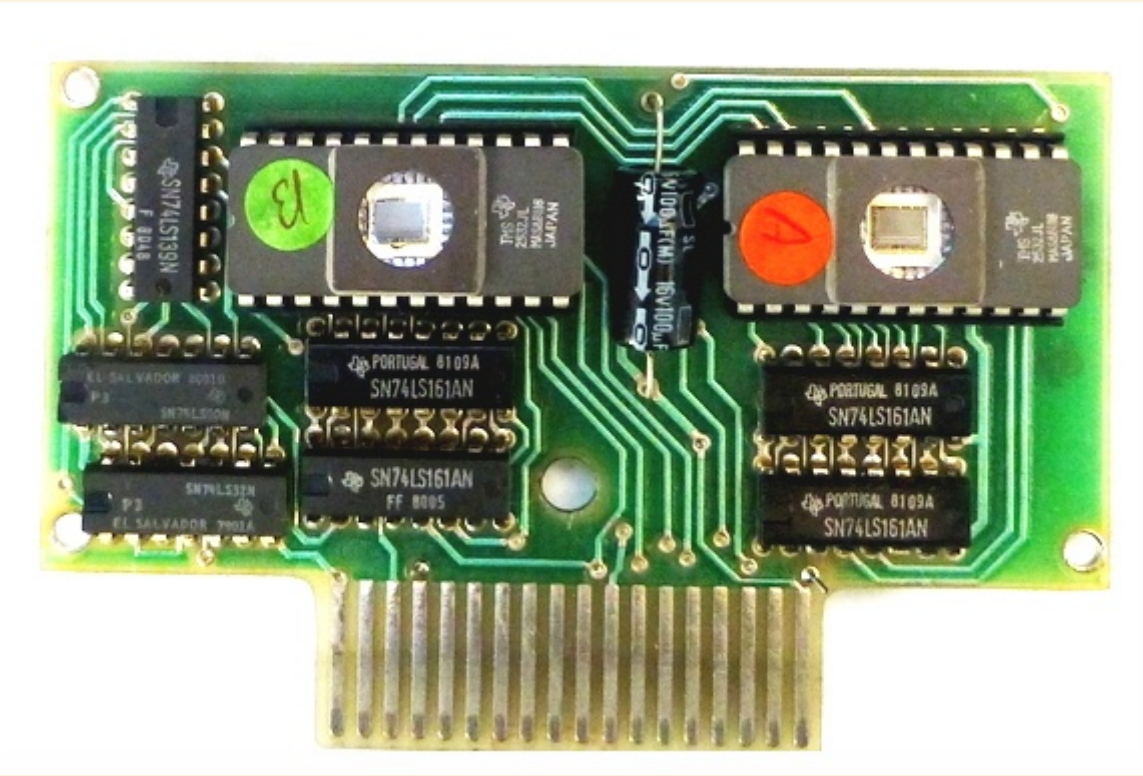
```
1ff0 | 82 33 00 01 af 33 a2 77 de 0f 00 d6 10 04 50 0b
2000 |
```

EDITED

```
1ff0 | 82 33 00 01 af 33 a2 77 de 0f 00 d6 10 04 50 0b
2000 | aa 01 01 00 00 00 80 10 00 00 00 00 00 00 00 00
2010 | 00 00 80 20 0a 41 44 56 45 52 54 49 53 45 52 00
2020 | 0b
```

Address is offset from file start, add >6000 for TI Grom address

---



For the technically minded- TI knew how to run GPL without using GROM chips from early on.

The Advertiser module is written in GPL and has NO grom chips.

The sample seen has the following chips:

2 x 4k eprom TMS2532JL,
4x synchonous 4 bit binary counters SN74LS161,
2 x quadruple 2 input positive OR gates SN74LS32N
and a dual decoder/demultiplexer SN74LS139N.

Go to top of page

Index of CALLS