6 of 10

4/28/80

MIXING GRAPHICS MODES

I. GENERAL

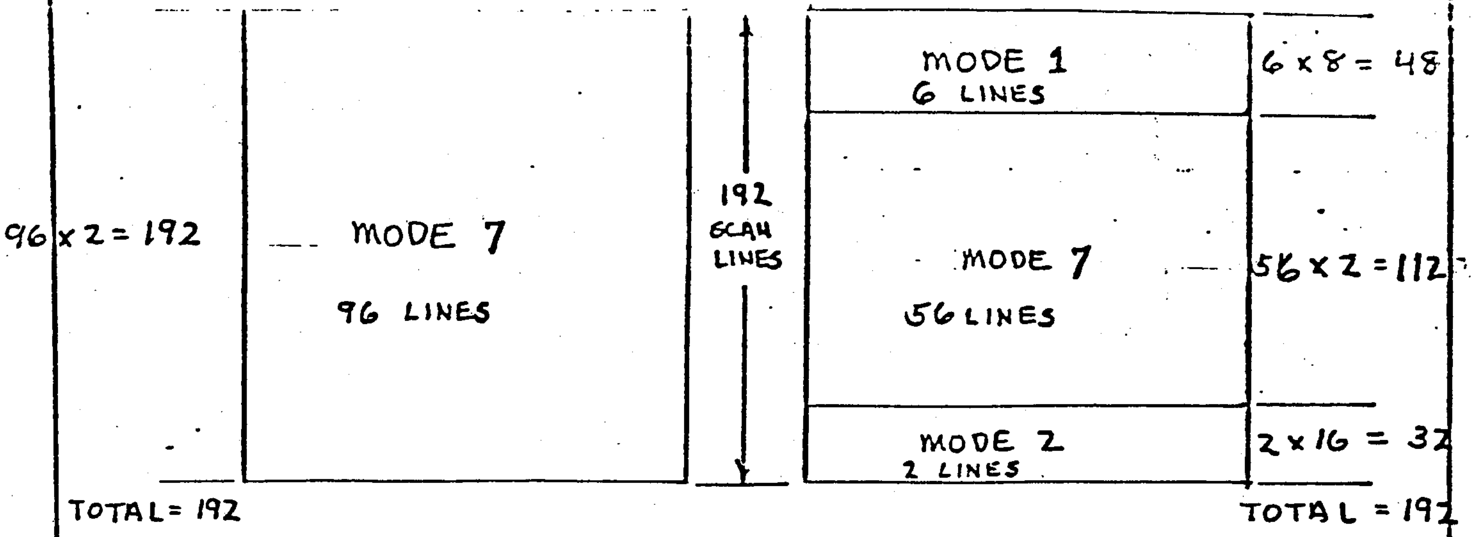
This procedure describes how to mix several graphic modes on the TV screen at the same time using BASIC commands. Each graphics mode has a different number of scan lines per "Mode Line" (one line of a graphics mode). The TV screen must consist of 192 scan lines, so when mixing modes, they must be combined in such a way as to get 192 scan lines. This is accomplished by modifying the Display List.

When a graphics mode is set on the computer, the O/S allocates RAM space for the graphics mode, then builds the display list adjacent to the graphics RAM, and sets a pointer to the beginning of the display list. Each "mode line" is constructed from a "mode byte" in the display list that determines how many scan lines in each mode line. The display list describes the screen display from top to bottom.

A Display List must be built for the "max RAM mode" (the graphics mode that requires the most RAM) then modified with POKES to mix the other modes with it. This "max RAM mode" cannot be a split screen mode (text window), therefore "max RAM mode" +16 must be used. If the max RAM mode will be at the top of the screen, then the "LMS byte" (load memory scan byte) at the top of the Display List will already be correct. If not, the "LMS byte" will have to be modified.

The Display List is modified by POKING a new mode byte for each mode line that is <u>not</u> a max RAM mode line. At the end of the display list is a JUMP instruction pointing to the top of the Display list. When the Display List is modified, the JUMP instruction must be placed immediately after the last mode byte.

Example #1 will be used throughout this procedure to illustrate each step.



MODE 7 DISPLAY LIST

	MODE LINE #	(HEX)	DEC	SCAN LINE #
	1	(4D)	77	2
	—	LO		—
	—	HI		—
MODE 7 96 LINES	2	(ØD)	13	4
	3	(ØD)	13	6
	4	(ØD)	13	8
	≈	≈		≈
	94	(ØD)	13	188
	95	(ØD)	13	190
	96	(ØD)	13	192
	JUMP→	(41)	65	
		LO		
		HI		

← (LMS BYTE) →

← "START" →

MODIFIED MODE 7 DISPLAY LIST

	MODE LINE #	(HEX)	DEC	SCAN LINE #
	1	(46)	7Ø	8
	—	LO		—
	—	HI		—
MODE 1 6 LINES	2	(Ø6)	6	16
	3	(Ø6)	6	24
	4	(Ø6)	6	32
	5	(Ø6)	6	40
	6	(Ø6)	6	48
MODE 7 56 LINES	7	(ØD)	13	5Ø
	8	(ØD)	13	52
	≈	≈		≈
	61	(ØD)	13	158
	62	(ØD)	13	160
MODE 2 2 LINES	63	(Ø7)	7	176
	64	(Ø7)	7	192
	JUMP→	(41)	65	
		LO		
		HI		

EXAMPLE #1

II. PROCEDURE TO SET UP SCREEN IN MIXED MODES:

1. Select modes desired, then look up which mode is the max RAM mode from table #2.

   example: modes selected - mode 1, mode 7, mode 2

   mode 7 = max RAM mode

2. Use table #1 to calculate the number of mode lines such that the total number of scan lines = 192.

   example:

mode	# mode line	scan lines per mode line	scan lines
1	6	8	48
7	56	2	112
2	2	16	32
			192 TOTAL

3. If the max RAM mode is at the top of the screen, then skip this step: Calculate the LMS byte by setting the left nibble to 4, then use table #1 to find the right nibble for the graphics mode at the top of the screen.

   example: 1. left nibble = 4
   2. right nibble for mode 1 = 6
   3. LMS byte = 46 (HEX)

4. Calculate the mode byte for each mode. Set the left nibble to 0, use table #1 to find the right nibble for each mode.

   example:

Mode	Left Nibble	Right Nibble	Mode Byte (HEX)
1	0	6	06
7	0	D	0D
2	0	7	07

II. PROCEDURE TO SET UP SCREEN IN MIXED MODES, cont.:

5. Convert all bytes to decimal.

example:

Byte	(HEX)	DEC
LMS	46	70
Mode 1	06	6
Mode 7	0D	13
Mode 2	07	7

6. Execute a graphics call on the computer using the max RAM mode (+16).

   example: GRAPHICS 7 + 16

7. PEEK the Display List pointer and use it to calculate a variable labelled "START".

   example: START = PEEK(560) + PEEK(561) * 256 + 4

8. If the max RAM mode is at the top of the screen, then skip this step: Poke the LMS byte to location START-1.

   example: POKE START-1,70

9. Every mode line requires a mode byte in the Display List in the same order as the mode lines appear on the screen. The mode bytes must be POKED into the Display List at location START + offset, where offset = mode line #.

Example:	MODE LINE #	POKE INSTRUCTION
MODE 1	2	POKE START + 2,6
	3	POKE START + 3,6
	4	POKE START + 4,6
	5	POKE START + 5,6
	6	POKE START + 6,6
MODE 7	see note for mode 7 (max RAM mode)	
MODE 2	63	POKE START + 63,7
	64	POKE START + 64,7

NOTE: The Display List will already be correct for the max RAM mode, therefore its mode bytes do not need to be POKEd.

II. PROCEDURE TO SET UP SCREEN IN MIXED MODES, cont.:

10. POKE the JUMP instruction followed by the LO byte, then the HI byte into the Display List. The offset for the JUMP POKE is the last mode line # + 1, for LO byte it is + 2, for HI byte it is + 3.

example: (last mode line # was 64)

REMARK	POKE INSTRUCTION
JUMP	POKE START + 65,65
LO BYTE	POKE START + 66, PEEK(560)
HI BYTE	POKE START + 67, PEEK(561)

III. PROCEDURE TO PRINT AND PLOT IN MIXED MODES

1. If the mode line #'s of a mode on the screen fall within the range of that mode's normal mode line #'s then use the following procedure:
   a. POKE 87 with the mode #
   b. Determine the Y coordinate by counting the # of mode lines from the top of the screen to the current position.
   c. Determine the X position in the normal manner for that mode.
   d. Depending on the mode, either PLOT and DRAWTO, or POSITION and PRINT.
   e. These steps must be done for each mode on the screen that meets the condition in step 1.

example:	MODE 1	POKE 87,1 POSITION 2,1:PRINT #6;"TEXT"
	MODE 7	POKE 87,7 COLOR 1:PLOT 20,20:DRAWTO 30,30
	MODE 2	See step 2

III. PROCEDURE TO PRINT AND PLOT IN MIXED MODES, cont.

2. Some modes may have mode line #'s outside of their normal range.

   example: Mode 2 normally has mode line #'s 1 through 12 (full screen). These are modified to #63 and #64 in example #1.

   To prevent the computer from giving a "cursor out of range" error message the following procedure can be used:

   a. Set a variable labelled "MEMST" to be the display memory start pointer.
      MEMST = PEEK(START) + PEEK(START + 1) * 256

   b. Set a variable labelled CHRPOS to position characters to be printed on the target line.

      $\text{CHRPOS} = \text{MEMST} + [(M_1-1)*R-M_2*(R-20)-M_3*(R-10)]+X$

      Where:

      X = horizontal position of character on the target line.
      R = the RAM per line of the Max RAM Mode (table #1).
      M_1 = the Mode Line # of the target line.
      M_2 = the number of mode lines of 20 bytes of RAM per line <u>above</u> the target line.
      M_3 = the number of mode lines of 10 bytes of RAM per line <u>above</u> the target line.

      Example: calculate CHRPOS for Mode Line #64 (the last line of the Mode 2 area) at horizontal position 5.

      X = 5
      R = 40
      M_1 = 64
      M_2 = 7 (6 from Mode 1 area, 1 from Mode 2 area).
      M_3 = 0

      CHRPOS = MEMST + [(64-1)*40-7*(40-20)-0*(40-10)]+5
      CHRPOS = MEMST + [(63)*40-7*(20)-0*(30)]+5
      CHRPOS = MEMST + [2520 - 140] + 5
      CHRPOS = MEMST + [2380] + 5
      CHRPOS = MEMST + 2385

III. PROCEDURE TO PRINT AND PLOT IN MIXED MODES, cont.

2. cont.

c. If few characters will be printed, then each character's internal value may be looked up in the Internal Character Set Table (Table 9.6), in the new BASIC Reference Manual. This value is then POKED into CHRPOS.

d. If strings are to be output, and if the ATASCII values of all the characters lie within one of the ranges shown in the table below, then do the following:

1) Obtain the appropriate ATASCII value range for the characters

2) Do the OPERATION the table indicates on the ATASCII value of each character.

3) POKE this value into CHRPOS.

ATASCII VALUE RANGE	OPERATION
0-31	Value + 64
32-95	Value - 32
96-127	NONE
128-159	Value + 64
160-223	Value - 32
224-255	NONE

Example: 1) assume we want to print the word "TEXT" in the mode 2 area of example #1 using the CHRPOS calculated previously.

2) these characters are in the ATASCII VALUE RANGE of "32 - 95".

3) the OPERATION for this range is "Value-32", so 32 must be subtracted from each ATASCII value.

4) the program statements would now look like this:

```
T$(1,4) = "TEXT"
CHRPOS  = MEMST + 2385

FOR X = 1 TO LEN(T$)
POKE CHRPOS + X - 1, ASC[T$(X,X)] - 32
                        ↑             ↑
NEXT X
                (OPERATION: value  -  32)
```

5) the FOR/NEXT loop POKEs the first character of T$, ASC[T$(X,X)]-32, into CHRPOS + 0.

6) the next iteration POKEs the next character of T$ into the next CHRPOS, and so on.

Mixing Graphics Modes, cont.———

TABLE #1

MODE BYTE

REMARK	LEFT NIBBLE (HEX)		RIGHT NIBBLE (HEX)	C.C. PER PIXEL	SCAN LINES PER MODE LINE	# COLORS	MODE	RAM PER LINE
①	4	CHAR MODES	2	½	8	1½ ③	0	40
			3	½	10	1½	–	40
			4	1	8	4	–	40
②	0		5	1	16	4	–	40
			6	1	8	5	1	20
			7	1	16	5	2	20
①	4	GRAPHIC MODES	8	4	8	4	3	10
			9	2	4	2	4	10
			A	2	4	4	5	20
			B	1	2	2	6	20
②	0		C	1	1	2	–	20
			D	1	2	4	7	40
			E	1	1	4	–	40
			F	½	1	1½	8	40
BLANK	0-7	④	0	BLANK	–	–	–	–
JUMP	4	SPECIAL	1	JUMP	–	–	–	–

1. When the max RAM mode is not at the top of the screen, the left nibble of the LMS byte must be changed to a 4.

2. Left nibble for all mode bytes after the LMS byte.

3. Color & Lum for the field is controlled by Setcolor 2, and Lum for charactors or graphics from Setcolor 1.

4. JUMP - used to end the display list and return to the beginning.

   BLANK - to output selected number of background lines.

Mixing Graphics Modes, cont.

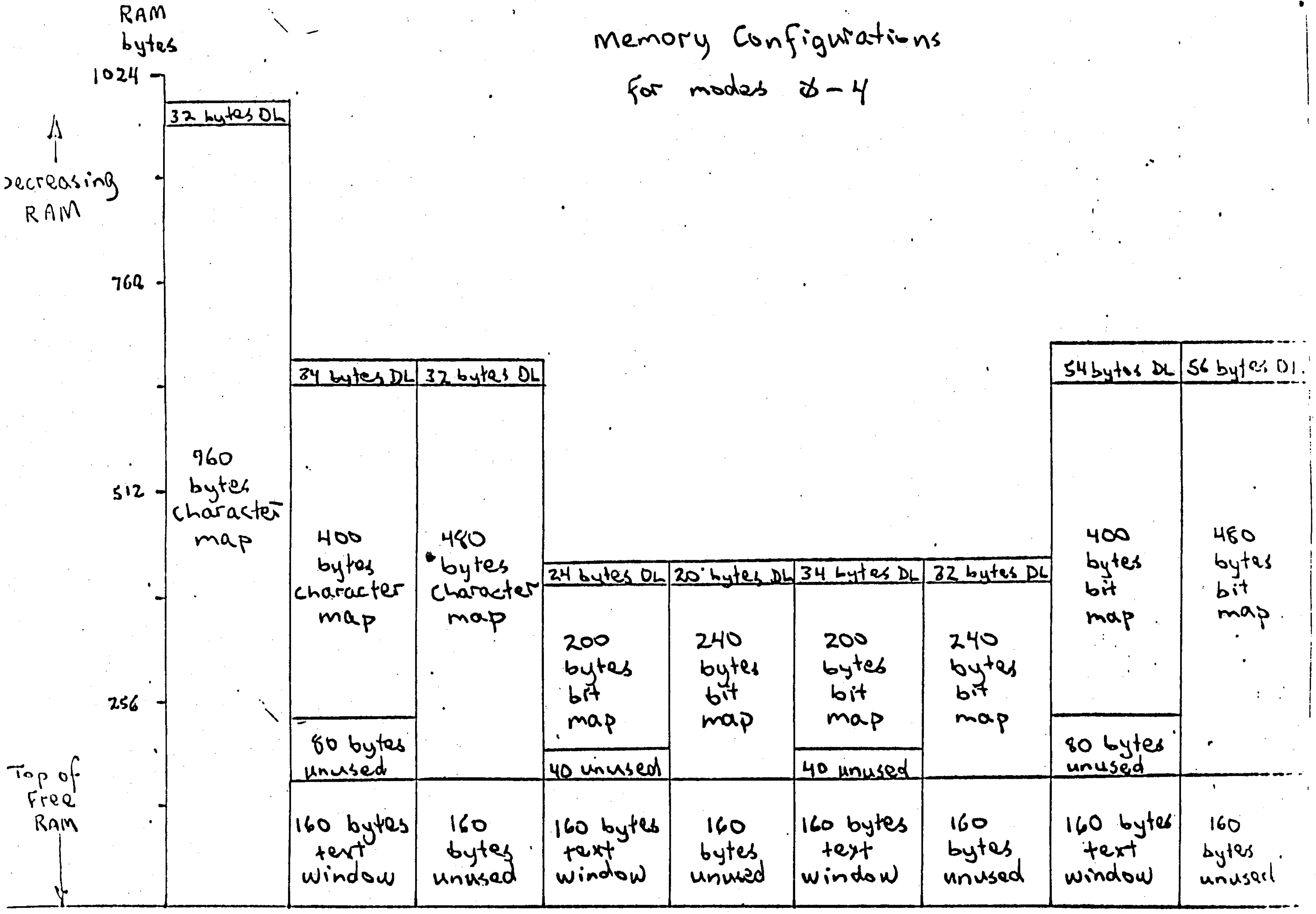
TABLE #2

GRAPHICS MODES
RAM REQUIREMENTS

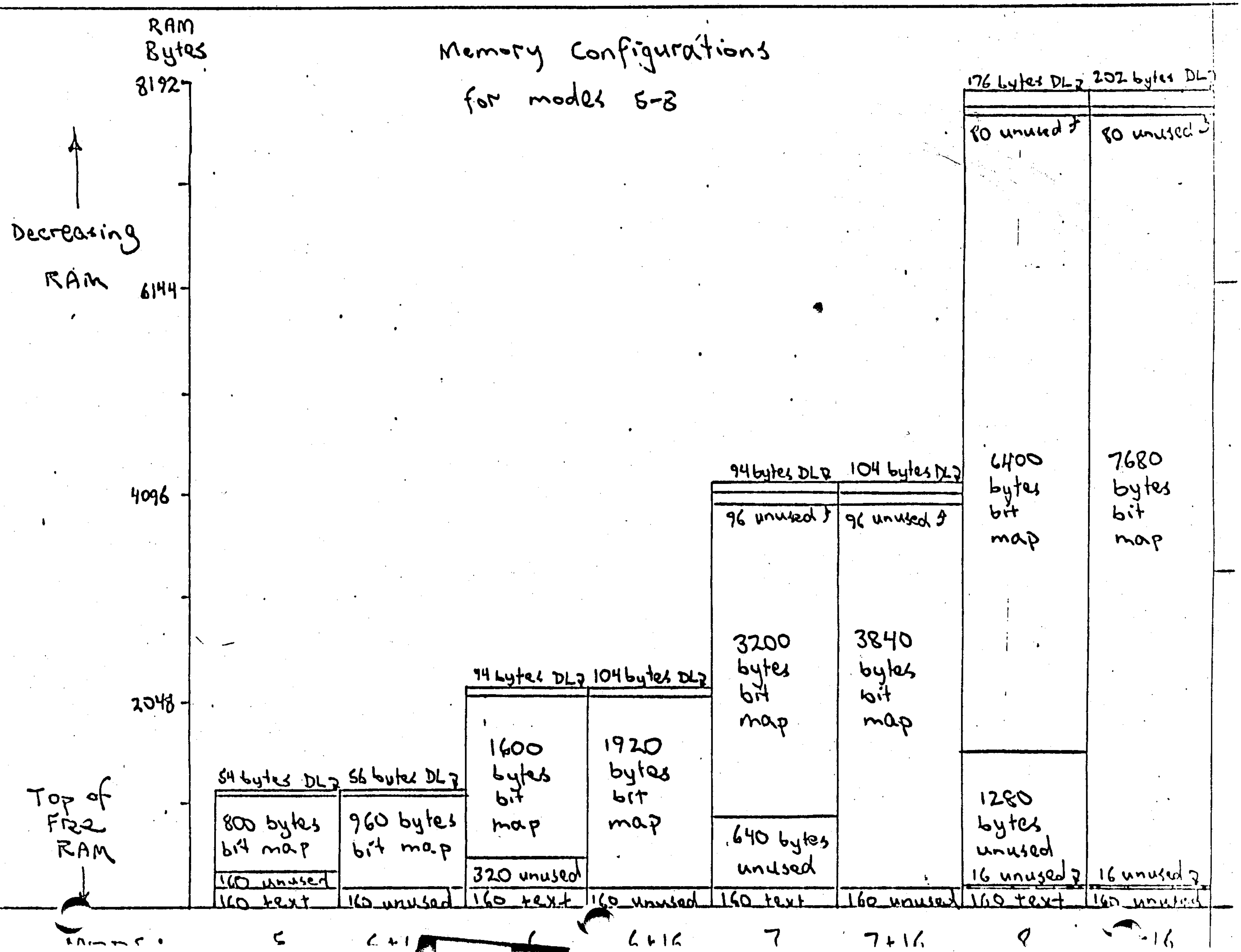
Mode	8 + 16	8138	Bytes
	8	8112	
	7 + 16	4200	
	7	4190	
	6 + 16	2184	
	6	2174	
	5 + 16	1176	
	5	1174	
	4 + 16	696	
	4	694	
	3 + 16	432	
	3	434	
	2 + 16	420	
	2	424	
	1 + 16	672	
	1	674	
	0	992	

These values include the display list and any imbedded unused memory blocks.

Memory Configurations
for modes ∅ – 4



Memory Configurations for modes 5-8


RAM Bytes
Decreasing RAM
Top of Free RAM
54 bytes DL?
800 bytes bit map
160 unused
160 text
56 bytes DL?
960 bytes bit map
160 unused
94 bytes DL?
1600 bytes bit map
320 unused
160 text
104 bytes DL?
1920 bytes bit map
160 unused
94 bytes DL?
96 unused ?
3200 bytes bit map
640 bytes unused
160 text
104 bytes DL?
96 unused ?
3840 bytes bit map
160 unused
176 bytes DL?
80 unused ?
6400 bytes bit map
1280 bytes unused
16 unused ?
160 text
202 bytes DL?
80 unused ?
7680 bytes bit map
16 unused ?
160 unused