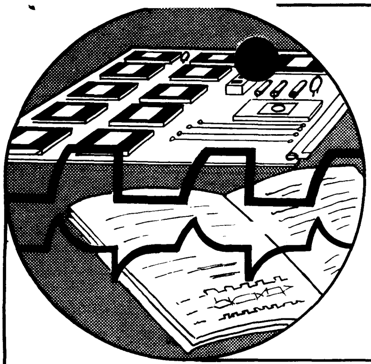


APPLICATION NOTE

AUGUST 1981

NA-006A

Semi-graphic display unit using EF9340 and EF9341

Augustin GIADIN
Laboratoire d'applications

THOMSON-EFCIS

THOMSON-EFCIS integrated circuits

CONTENTS

The aim of this application note is to aid EF9340 (VIN) and EF9341 (GEN) users in building a complete display unit for a videotex terminal, or a general purpose 25 x 40 character display unit. The first part of this document gives a design example, then programming informations beyond those included in the data sheet will be provided.

1 – HARDWARE DESIGN

A complete display unit has been designed to operate with a main CPU board. Schematic diagram of the display board is given in figure 6 and a block diagram in figure 1. It can be divided into three main functionnal blocks :

- The basic display unit
- The extended character generator
- The video interface.

1.1 – Basic display unit

The basic display unit performs all the display functions and consists of :

- GEN (EF9341)
- VIN (EF9340)
- The page memory.

GEN contains a character generator, a character code register and two 8 bit registers – TA and TB – which provide a buffered interface with a microprocessor data bus.
VIN contains a timing generator, an access automaton and a display automaton.
The page memory is made of two 1 K x 8 static RAM devices.

1.2 – Microprocessor interface

The display unit interface with the microprocessor through the transfer registers of GEN. Data and command transfers are performed through the two 8 bit registers – TA and TB –, under the control of the Enable (E) and Read/Write (R/$\overline{W}$) signals provided by the MPU board. The R/$\overline{W}$ signal controls the direction of data or command transfer.
A "low" state on the R/$\overline{W}$ line enables the input buffers and data is transferred to GEN on the falling edge of the E signal if the device has been selected ($\overline{CS}$ = 0). A high state on the Read/Write line sets up GEN for a data transfer to the microprocessor data bus when $\overline{CS}$ = 0. The GEN output buffers are enabled when the proper address and the Enable pulse are present.
The C/$\overline{T}$ input line specifies whether data (C/$\overline{T}$ = 0) or command (C/$\overline{T}$ = 1) is to be transferred from/ to GEN transfer registers.
The D0-D7 lines of GEN are connected to the microprocessor bus through a 74LS640 Bidirectionnal Driver (the MPU board data bus is inverted).
Address decoding with a 74LS04 and a 74LS30 devices selects the Bidirectionnal Driver and GEN when the hexadecimal value of the 8 MSB on the microprocessor address bus is EC and when the VUA line is high. The B/$\overline{A}$ and C/$\overline{T}$ inputs of GEN are respectively connected to the A0 and A1 lines of the address bus in order to have access to the GEN registers through four consecutive addresses :

- EC00 for TRA } data transfer registers
- EC01 for TRB }
- EC02 for CRA } command transfer registers
- EC03 for CRB }

Therefore, the whole display unit is accessed from the microprocessor through only four addresses.

1.3 – Clock generation

The half-dot frequency clock (3,5 MHz) is implemented with a Voltage-Controlled Oscillator 74LS124 and a 3,5 MHz crystal. The VCO's output is connected to the CLK input of VIN.

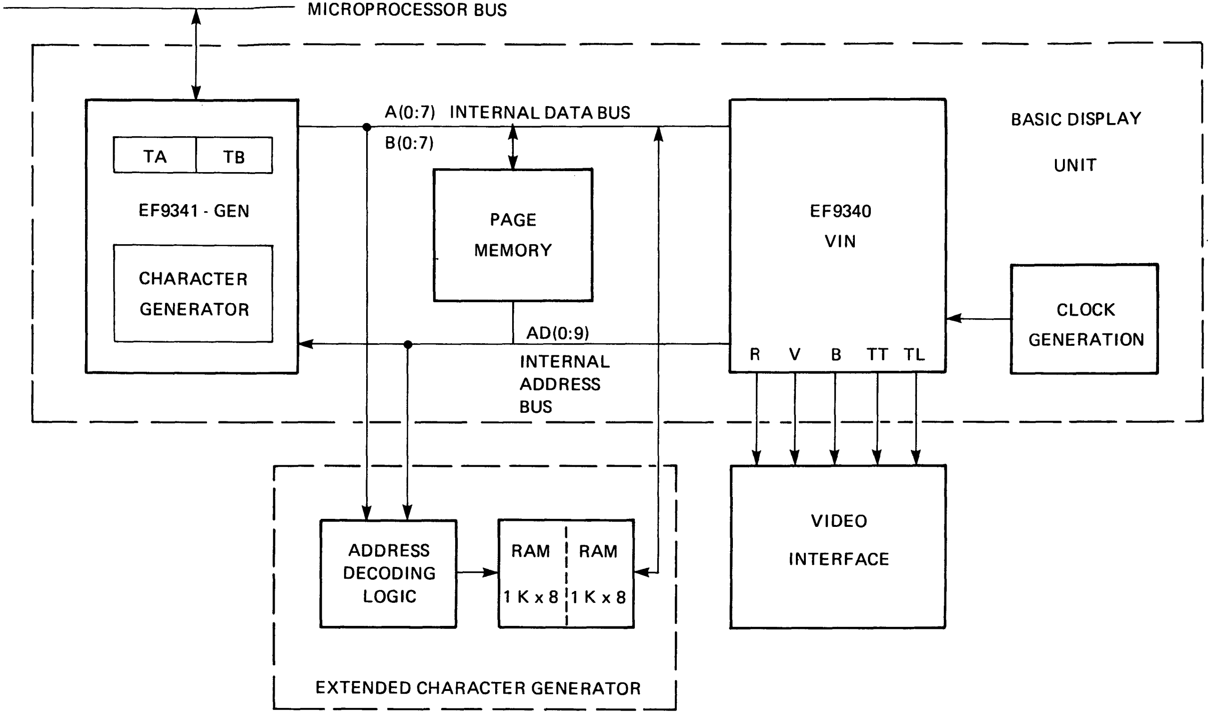


FIGURE 1 – DISPLAY BOARD BLOCK DIAGRAM

The aim of this application note is to aid EF9340 (VIN) and EF9341 (GEN) users in building a complete display unit for a semi-graphic terminal, or a general purpose 25 x 40 character display unit. The first part of this document gives a design example, then programming informations beyond those included in the data sheet will be provided.

1 – HARDWARE DESIGN

A complete display unit has been designed to operate with a main CPU board. Schematic diagram of the display board is given in figure 6 and a block diagram in figure 1. It can be divided into three main tunctionnal blocks :

- The basic display unit
- The extended character generator
- The video interface.

1.1 – Basic display unit

The basic display unit performs all the display functions and consists of :

- GEN (EF9341)
- VIN (EF9340)
- The page memory.

GEN contains a character generator, a character code register and two 8 bit registers – TA and TB – which provide a buffered interface with a microprocessor data bus.
VIN contains a timing generator, an access automaton and a display automaton.
The page memory is made of two 1 K x 8 static RAM devices.

1.2 – Microprocessor interface

The display unit interface with the microprocessor through the transfer registers of GEN. Data and command transfers are performed through the two 8 bit registers – TA and TB –, under the control of the Enable (E) and Read/Write (R/$\overline{W}$) signals provided by the MPU board. The R/$\overline{W}$ signal controls the direction of data or command transfer.
A "low" state on the R/$\overline{W}$ line enables the input buffers and data is transferred to GEN on the falling edge of the E signal if the device has been selected ($\overline{CS}$ = 0). A high state on the Read/Write line sets up GEN for a data transfer to the microprocessor data bus when $\overline{CS}$ = 0. The GEN output buffers are enabled when the proper address and the Enable pulse are present.
The C/$\overline{T}$ input line specifies whether data (C/$\overline{T}$ = 0) or command (C/$\overline{T}$ = 1) is to be transferred from/to GEN transfer registers.
The D0-D7 lines of GEN are connected to the microprocessor bus through a 74LS640 Bidirectionnal Driver (the MPU board data bus is inverted).
Address decoding with a 74LS04 and a 74LS30 devices selects the Bidirectionnal Driver and GEN when the hexadecimal value of the 8 MSB on the microprocessor address bus is EC and when the VUA line is high. The B/$\overline{A}$ and C/$\overline{T}$ inputs of GEN are respectively connected to the A0 and A1 lines of the address bus in order to have access to the GEN registers through four consecutive addresses :

- EC00 for TRA }
- EC01 for TRB } data transfer registers
- EC02 for CRA }
- EC03 for CRB } command transfer registers

Therefore, the whole display unit is accessed from the microprocessor through only four addresses.

1.3 – Clock generation

The half-dot frequency clock (3,5 MHz) is implemented with a Voltage-Controlled Oscillator 74LS124 and a 3,5 MHz crystal. The VCO's output is connected to the CLK input of VIN.

1.6 — Extended character generators

One of the most interesting features of the display unit built around VIN and GEN is to allow the user to define one or two more 96 character sets. Adding these user character sets may be done with either RAM or ROM devices. Using RAM devices allows to define dynamically different character sets (dynamically redefinable character sets).
As à character is displayed within a 10 x 8 dot-matrix, it is defined by ten eight-bit slices in a character generator. Therefore, each character set is defined by 960 eight-bit slices.
The application board has been designed with two 1 K x 8 static RAM (MK-4118) as extended character generators. The data input/output pins I/01-I/08 of the RAM devices are connected to the A(0:7) internal data bus lines, and access to each device is controlled by the internal read/write signal (R/$\overline{\text{WI}}$) and separate chip select lines.

Addressing the extended character generators

A 8-bit slice address is given by :
- an eight-bit character code
- a four-bit slice number

Access to a character generator is performed by VIN in two cycles :
- during the first cycle, the character code is present on the B(0:8) and A7 internal data bus lines.
- during the second cycle, VIN brings out the slice number on the AD(0:3) internal address bus lines, and AD4 = 1 when an extended character generator is addressed.

The character code is latched during the first cycle into a 74LS374 and a 74LS74 devices, on the leading edge of $\overline{\text{SM}}$. A quad 2 to 1 multiplexer (74LS257) is used to transcode the 11 bit logical address - seven bit character code and four-bit slice number - into a 10 bit physical address.
Address decoding includes a 74LS138 one-of-four decoder, which is enabled when AD4 is "high" and $\overline{\text{SG}}$ "low". The decoder Y2 and Y3 outputs are applied to the RAM device chip select inputs ($\overline{\text{CS}}$).

AD3 = 0	CCE6	CCE5	CCE4	CCE3	CCE2	CCE1	CCE0	AD2	AD1	AD0
AD3 = 1	0	0	CCE4	CCE3	CCE2	CCE1	CCE0	CCE6	CCE5	AD0

Address transcoding table.

NOTA : CCE : Latch output   AD3 : Multiplexer select line

1.7 — Video interface

VIN provides signals which allow simple interface with a CRT display with a minimum of external hardware :

- TL : horizontal synchronization signal
- TT : vertical synchronization signal
- R,G,B : red, green and blue signals
- I : boxing signal

The TL pulse duration is programmable through bit 5 of the operating mode R register. When R5 = 0, the TL output is low during 4.5 μs.
When R5 = 1, the TL output is high for 18.3 μs (see figure 3).
The TT vertical synchronization signal may be programmed through bit 6 of the R register to define a 312 line (R6 = 1) or 262 line (R6 = 0) frame period (see figure 4).
The I boxing signal delimits the display zone on the screen, and may be used to insert "character boxes" into a television video as a caption or subtitle.
R, G, B signals are internally shifted out by VIN. These signals may be applied to the R, G, B inputs of a display device, or mixed with TT and TL for composite signal generation. Schematic diagram in figure 6 gives an example for video signal generation.

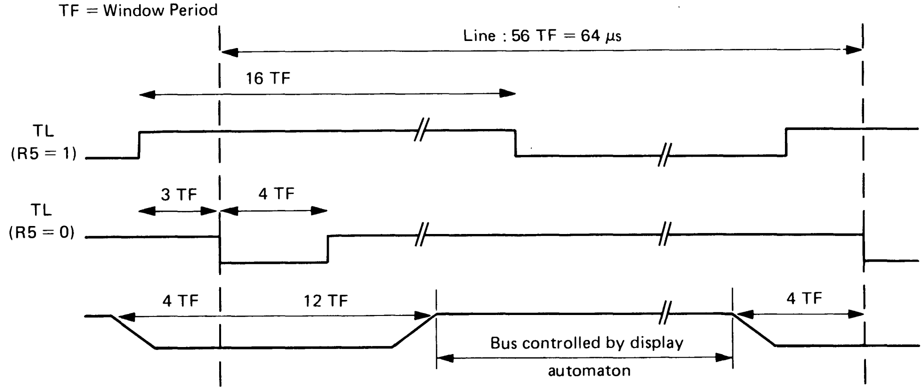
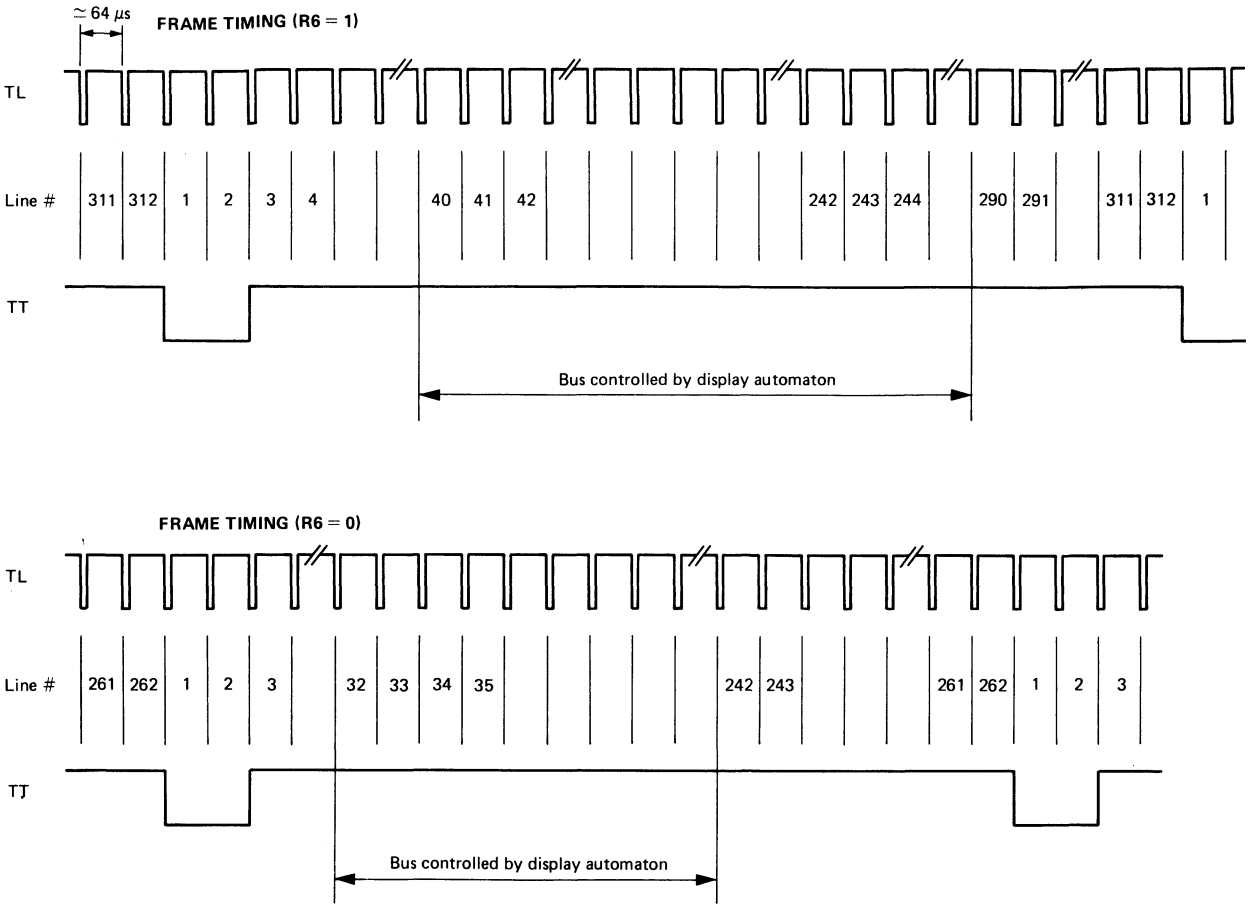


FIGURE 3 — TL SIGNAL PROGRAMMING


FRAME TIMING (R6 = 1)

FRAME TIMING (R6 = 0)

FIGURE 4 — TT SIGNAL PROGRAMMING

2 – PROGRAMMING THE DISPLAY UNIT

2.1 – Transfer registers

The complete display unit interface with the microprocessor through 4 registers :

– 2 data transfer registers : TRA and TRB
– 2 command transfer registers : CRA and CRB.

Figure 5 summarizes GEN transfer register addressing with $C/\overline{T}$, $B/\overline{A}$ and $R/\overline{W}$ signals.
Whenever a data or command transfer is made between the display unit and the microprocessor, access to CRB or TRB sets the GEN Busy flip-flop. The Busy flip-flop is reset as soon as VIN accedes to the transfer registers. The Busy flip-flop state can be read in bit 7 of CRA. Therefore, before any access to the transfer registers, the user must verify that bit 7 of CRA is "0".

ADDRESS		ADDRESSED REGISTER		
$C/\overline{T}$	$B/\overline{A}$	$R/\overline{W} = 1$ Read	$R/\overline{W} = 0$ Write	Comments
0	0	Read TRA	Write TRA	A 16 bit data is read or written from/into the mail box
0	1	Read TRB (1)	Write TRB (1)	
1	0	Read CRA (2)	Write CRA	A 16 bit command is written into the mail box or the Busy flip-flop is read
1	1	ILLEGAL	Write CRB (1)	

FIGURE 5 – TRANSFER REGISTER ADDRESSING

(1) : Sets the GEN Busy flip-flop
(2) : Busy flip-flop is read in CRA bit 7.

2.2 – Data transfer

Data transfer is always performed through TRA and TRB data transfer registers. Data transfer within the display unit is defined by the M transfer mode register contents (see M Register programming).
A data transfer operation is not performed by VIN until TRB is accessed.

2.3 – Command transfer

The microprocessor sends a command to the display unit by writing into CRA **then CRB** registers. As for data transfer, writing into CRB sets the Busy flip-flop, which is reset as soon as VIN gets the command. The different commands to the display unit are :

– cursor programming
– operating mode initialization (R register)
– roll-up, roll-down and zoom mode programming (Y0 register)
– data transfer mode programming (M register)

Table 1 summarizes the command set.

2.4 – Cursor programming

The cursor position is defined by two counters :

- The X counter (0:5) points to a column, from 00 to 39.
- The Y counter (0:4) points to a row, from 00 to 23. The Y counter value for the service row is 31.

Whenever the cursor is incremented, the X counter is incremented. The Y counter is automatically incremented when the X counter overflows from 39 to 00. The Y counter overflows from 23 or 31 to 00.

Four commands allow the cursor programming :

BEGIN ROW : The cursor is set at the beginning of the row whose value is given by the CRA register contents. The binary value to be loaded into CRB for this command is : 000X XXXX (X = don't care).

LOAD Y : The CRA contents is loaded into the Y counter. The binary value to be loaded into CRB for this command is : 001X XXXX.

LOAD X : The CRA contents is loaded into the X counter. The binary value to be loaded into CRB for this command is : 010X XXXX.

INC C : The cursor is incremented. The binary value to be loaded into CRB for this command is : 011X XXXX.

NOTA : In addition to these four commands, the cursor position may be automatically incremented when data is read or written into the page memory (see M register programming).

2.5 – R register programming

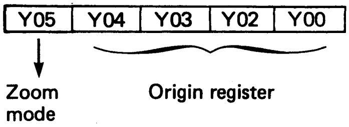
This 8 bit register of VIN defines the display unit operating mode (see EF9340 and EF9341 specifications). The "LOAD R" command loads the R register with the CRA contents when the binary value 101X XXXX is written into the CRB register.

2.6 – Y0 register programming

This 6 bit register of VIN allows roll-up, roll-down and zoom programming. The "LOAD Y0" command loads the Y0 register with the CRA contents when the binary value 110X XXXX is written into CRB.

ORIGIN REGISTER : The 5 LSB of Y0 define the origin register, i.e. the first row to be displayed on the top of the screen, after the service row. Therefore, roll-up or roll-down is made by incrementing or decrementing the origin register.

ZOOM MODE : When bit 5 of the Y0 register is set, the first 12 rows from the one defined by the origin register Y0 (0:4) are displayed in double height. Characters defined with double height attribute are displayed in quadruple height. Since programming the zoom mode is independent of the origin register, roll-up or roll-down operation may be performed in the zoom mode as well as in the normal mode.



NOTA : The service row is never affected by Y0 programming and depends only on bit 3 of the R register (service row on/off).

TABLE 1 – COMMANDS

COMMAND CODE																NAME	OPERATION
B7	B6	B5	B4	B3	B2	B1	B0	A7	A6	A5	A4	A3	A2	A1	A0		
0	0	0									K (0 : 4)					Begin Row	X (0 : 5) ← 0 Y (0 : 4) ← K (0 : 4)
0	0	1								K (0 : 4)						Load Y	Y (0 : 4) ← K (0 : 4)
0	1	0								K (0 : 5)						Load X	X (0 : 5) ← K (0 : 5)
0	1	1														INC C	C ← C+ 1
1	0	0						K (0 : 7)								Load M	M (0 : 7) ← K (0 : 7)
1	0	1						K (0 : 7)								Load R	R (0 : 7) ← K (0 : 7)
1	1	0								K (0 : 5)						Load Y0	Y0 (0 : 5) ← K (0 : 5)
1	1	1														ILLEGAL	

TABLE 2 – ACCESS MODES : M REGISTER CONTENT

ACCESS MODE REG.								ACCESS MODE	SUBSEQUENT DATA TRANSFER
M7	M6	M5	M4	M3	M2	M1	M0		
0	0	0						Write	MP (C) ← T ; C ← C+ 1
0	0	1						Read	T ← MP (C) ; C ← C+ 1
0	1	0						Write without INC	MP (C) ← T
0	1	1						Read without INC	T ← MP (C)
1	0	0		N T				Write slice	GC (MP(C), NT) ← T ; NT ← NT+ 1
1	0	1		N T				Read slice	T ← GC (MP(C), NT) ; NT ← NT+ 1

NT : Slice Number T : Mail Box C : Cursor MP : Page Memory GC : Character Generator

2.7 — M register programming

This 8 bit register of VIN defines a data transfer mode on the internal data bus. The "LOAD M" command loads the M register with CRA contents when the binary value 100X XXXX is written into CRB. There are two main data transfer modes within the display unit :

- read or write into the page memory
- read or write into a character generator.

When acceding to a character generator, the 4 LSB of the M register define a slice number.
Table 2 summarizes the data transfer modes defined by the M register.

PAGE MEMORY READ

In the page memory read mode, a two byte character code addressed by the cursor is loaded into the TRA and TRB data transfer registers.
Then the cursor position is incremented or not, according to the M register contents :
– M = 001X XXXX for page memory read with cursor incrementation
– M = 011X XXXX for page memory read without cursor incrementation.

PAGE MEMORY WRITE

The TRA and TRB register contents is loaded into the page memory at the location addressed by the cursor. Then the cursor is incremented or not, according to the M register contents :
– M = 000X XXXX for page memory write with cursor incrementation.
– M = 010X XXXX for page memory write without cursor incrementation.

WRITING INTO AN EXTENDED CHARACTER GENERATOR

This transfer mode allows the user to load a RAM memory used as an extended character generator. The address of the data to be written is given by :
- NT : a character slice number.
- MP(C) : a character code in page memory pointed by the cursor, which should be an extended character code : B7 = 1 and B5 + B6 = 1. (see table 3).

The initial slice number is given by the 4 LSB of the M register :

M7							M0
1	0	0	X				

Initial slice value

After the M register is loaded with the initial slice number value, whenever TRA then TRB is written by the microprocessor, the TRA register contents is loaded into the extended character generator location addressed by (MP(C), NT). Then NT is automatically incremented modulo 10.

READING A CHARACTER GENERATOR

This data transfer mode is used to read slices from a character generator. The value to be loaded by a command transfer into the M register to set this mode is :

M7							M0
1	0	1	X				

Initial slice number NT

As in the previous mode, the slice address is given by (MP(C), NT) and the initial slice number is given by the 4 LSB of the M register. The slice number NT is automatically incremented modulo 10 after a read operation.

REMARKS :

- Programming the M register by a "LOAD M" command sets only a data transfer mode within the display unit. Subsequent data transfers are not performed until access either to TRA then TRB, or to TRB. Specially, when programming the M register in a read mode from the page memory or a character generator, a first access to TRB is necessary to load GEN data transfer registers with the correct data to be read.

- When reading or writing into a character generator, a page memory word must be initialized with a character code which is used to address the character generator. This initialization may be done by writing into page memory without cursor incrementation.

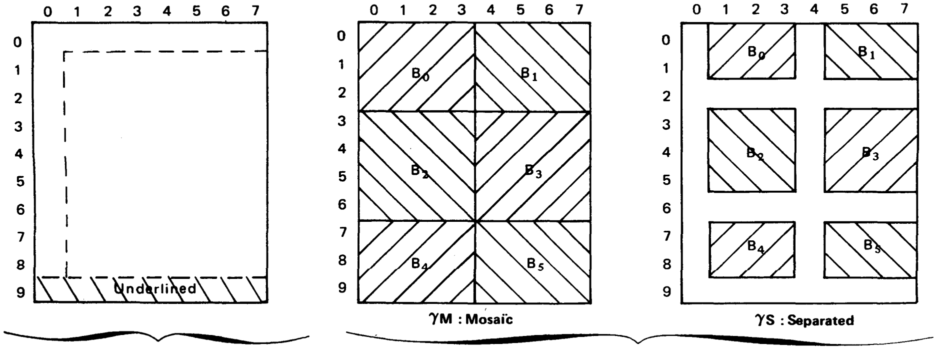
TABLE 3 – CHARACTER CODES

WINDOW CODE IN PAGE MEMORY																COMMENTS			
Type and character code field									Attribute field							Type	Implicite Attribute	Ad - hoc Serial Attribute	Remark
B7	B6	B5	B4	B3	B2	B1	B0	A7	A6	A5	A4	A3	A2	A1	A0				
0	X	X	X	X	X	X	X	0	N	L	H	S	B_1	G_1	R_1	α_0 (128)		Underlined, C_0	in
1	0	0	–	–	s	i	m	0	B_0	G_0	R_0	–	B_1	G_1	R_1	DEL			GEN
1 1 1	0 1 1	1 0 1	X	X	X	X	X	0	N	L	H	S	B_1	G_1	R_1	α_1 (96)		C_0	EXTENSION
0	0	X	X	X	X	X	X	1	B_0	G_0	R_0	S	B_1	G_1	R_1	γ_S (64)	Normal size,		in
0	1	X	X	X	X	X	X	1	B_0	G_0	R_0	S	B_1	G_1	R_1	γ_M (64)	positive		GEN
1	0	0	—	—	—	—	—	1	—	—	—	—	—	—	—	ILLEGAL			
1 1 1	0 1 1	1 0 1	X	X	X	X	X	1	B_0	G_0	R_0	S	B_1	G_1	R_1	γ_1 (96)	Normal size		EXTENSION

Nota : Extension for α 1 and γ 1 may be mapped in only one 1 K x 8 RAM or ROM.

GLOSSARY :

α	Alphanumeric	C_0 (R_0, G_0, B_0)	Background colour
γ	Semi-graphic	C_1 (R_1, G_1, B_1)	Foreground colour
γS	Separated semi-graphic	H	Double height
γM	Mosaïc semi-graphic	L	Double width
DEL	Delimitor	N	Reverse video (negative)
m	Blanking	S	Stable (non blinking)
i	Boxing	X	Character code
s	Underlining	–	Don't care



α : ALPHANUMERIC

γ : SEMI-GRAPHIC

```
PAGE 001  APPLI   .SA:0

00001                         OPT    LLE=110
00002                  *
00003                  ************************************************
00004                  *  THIS PROGRAMMING EXAMPLE SHOWS TYPICAL DATA
00005                  *  AND COMMAND TRANSFERS BETWEEN THE DISPLAY
00006                  *  UNIT AND THE MICROPROCESSOR.
00007                  ************************************************
00008                  *
00009                  *
00010                  * REGISTER ADDRESS DEFINITION
00011                  *
00012          EC00  A TRA    EQU    $EC00    DATA TRANSFER REGISTERS.
00013          EC01  A TRB    EQU    $EC01
00014          EC02  A CRA    EQU    $EC02    COMMAND TRANSFER REGISTERS.
00015          EC03  A CRB    EQU    $EC03
00016                  *
00017                  * CONSTANT DEFINITION
00018                  *
00019                  *
00020          0020  A LDY    EQU    $20      "LOAD Y" COMMAND
00021          0040  A LDX    EQU    $40      "LOAD X" COMMAND
00022          0000  A BEGROW EQU    $00      "BEGIN ROW" COMMAND
00023          0060  A INCCUR EQU    $60      "INC C" COMMAND
00024          0080  A LDM    EQU    $80      "LOAD M" COMMAND
00025          00A0  A LDR    EQU    $A0      "LOAD R" COMMAND
00026          00C0  A LDY0   EQU    $C0      "LOAD Y0" COMMAND
00027                  *
00028A 4000                   ORG    $4000
00029                  *
00030                  * CHARACTER STRING DEFINITION.
00031                  *
00032A 4000    45    A MESS1  FCC    /EF9340 AND EF9341/
00033          4011  A ENDM1  EQU    *
00034A 4011    41    A MESS2  FCC    /APPLICATION/
00035          401C  A ENDM2  EQU    *
00036                  *
00037                  * EXTENDED CHARACTER DEFINITION
00038                  *
00039A 401C    20    A CAR1   FCB    $20,$38,$3C,$3E,$3F,$3F,$1F,$1F,$0F,$0F
00040A 4026    04    A CAR2   FCB    $04,$1C,$3C,$7C,$FC,$FC,$F8,$F8,$F0,$F0
00041A 4030    07    A CAR3   FCB    $07,$C7,$E3,$F3,$F9,$FC,$FC,$F8,$E0,$80
00042A 403A    E0    A CAR4   FCB    $E0,$E3,$C7,$CF,$9F,$3F,$3F,$1F,$07,$01
00043                  *
00044          5000  A STACK  EQU    $5000    STACK INITIALIZATION.
00045                  *
00046                  *  MEMORY RESERVATION
00047                  *
00048A 4044    0002  A SAV    RMB    2
00049A 4046    0050  A IBUF   RMB    80
```

```
PAGE  002  APPLI   .SA:0

00051                       *
00052                       *  MAIN PROGRAM :
00053                       *




00055A 4096 8E 5000  A DEB    LDS    #STACK   STACK INITIALIZATION

00057                       ***************************************************
00058                       * R REGISTER INITIALIZATION :
00059                       *   R7=1  BLINKING IS ENABLED.
00060                       *   R6=1  50 HZ OPERATION.
00061                       *   R5=0  TL IS LOW FOR 4 WINDOW PERIODS.
00062                       *   R4=1  THE CURSOR POSITION IS DISPLAYED.
00063                       *   R3=1  THE SERVICE ROW IS DISPLAYED.
00064                       *   R2=1  THE CONCEAL ATTRIBUTE IS ENABLED.
00065                       *   R1=0  THE BOXING ATTRIBUTE IS DISABLED.
00066                       *   R0=1  DISPLAY ON.
00067                       ***************************************************

00069A 4099 86 DD    A        LDAA   #%11011101
00070A 409B BD 41D8  A        JSR    LOADR    INIT. OPERATING MODE REGISTER R.

00072                       ***************************************************
00073                       * PAGE MEMORY INITIALIZATION :
00074                       * ALL THE PAGE MEMORY IS FILLED WITH THE ALPHANUMERIC
00075                       * CHARACTER $7F,IN BLACK COLOR
00076                       ***************************************************

00078A 409E C6 1F    A        LDAB   #31      SET CURSOR AT THE BEGINNING OF
00079A 40A0 BD 419E  A        JSR    ROW      THE SERVICE ROW
00080A 40A3 4F                CLRA            SET M REGISTER INTO "WRITE PAGE
00081A 40A4 BD 41E6  A        JSR    LOADM    MEMORY WITH   INC." MODE.
00082A 40A7 CE 03E8  A        LDX    #1000    INIT. LOOP COUNTER
00083A 40AA 86 08    A        LDAA   #$08     COLOR AND ATTRIBUTE DEFINITION.
00084A 40AC C6 7F    A        LDAB   #$7F

00086A 40AE BD 4194  A LOOP1  JSR    BUSY     TEST GEN BUSY FLIP-FLOP.
00087A 40B1 B7 EC00  A        STAA   TRA      WRITE A 2 BYTE CHARACTER CODE
00088A 40B4 F7 EC01  A        STAB   TRB      INTO PAGE MEMORY.
00089A 40B7 09                DEX             DECREMENT LOOP COUNTER.
00090A 40B8 26 F4 40AE        BNE    LOOP1

00092                       *************************************************
00093                       * WRITE MESS1 AND MESS2 STRINGS INTO PAGE MEMORY
00094                       * IN WHITE COLOR
00095                       *************************************************

00097A 40BA 86 0C    A        LDAA   #12      SET CURSOR AT LOCATION
00098A 40BC C6 0A    A        LDAB   #10      X=12 AND Y=10
00099A 40BE BD 41C2  A        JSR    LOADXY
00100A 40C1 CE 4000  A        LDX    #MESS1
00101A 40C4 86 0F    A        LDAA   #$0F     ATTRIBUTE AND COLOR DEFINITION.
```

```
PAGE  003  APPLI   .SA:0

00103A 40C6 E6 00     A LOOP2  LDAB    0,X
00104A 40C8 BD 4194   A        JSR     BUSY
00105A 40CB B7 EC00   A        STAA    TRA       WRITE A 2 BYTE CHARACTER CODE.
00106A 40CE F7 EC01   A        STAB    TRB
00107A 40D1 08                 INX
00108A 40D2 8C 4011   A        CPX     #ENDM1
00109A 40D5 26 EF 40C6         BNE     LOOP2

00111A 40D7 86 0F     A        LDAA    #15       SET CURSOR AT LOCATION X=15 AND
00112A 40D9 C6 0C     A        LDAB    #12       Y=12
00113A 40DB BD 41C2   A        JSR     LOADXY
00114A 40DE CE 4011   A        LDX     #MESS2
00115A 40E1 86 0F     A        LDAA    #$0F

00117A 40E3 E6 00     A LOOP3  LDAB    0,X
00118A 40E5 BD 4194   A        JSR     BUSY
00119A 40E8 B7 EC00   A        STAA    TRA       WRITE A 2 BYTE CHARACTER CODE.
00120A 40EB F7 EC01   A        STAB    TRB
00121A 40EE 08                 INX
00122A 40EF 8C 401C   A        CPX     #ENDM2
00123A 40F2 26 EF 40E3         BNE     LOOP3

00125                   *************************************************
00126                   * READ 17 CHARACTER CODES FROM PAGE MEMORY INTO
00127                   * A BUFFER IBUF.THE FIRST CHARACTER CODE TO BE
00128                   * READ IS AT LOCATION (X=12,Y=10).
00129                   * EACH CHARACTER CODE CONSISTS OF TWO BYTES.
00130                   *************************************************

00132A 40F4 86 0C     A        LDAA    #12       SET CURSOR AT LOCATION
00133A 40F6 C6 0A     A        LDAB    #10       X=12 AND Y=10
00134A 40F8 BD 41C2   A        JSR     LOADXY
00135A 40FB 86 20     A        LDAA    #$20      SET M REGISTER INTO "READ FROM
00136A 40FD BD 41E6   A        JSR     LOADM     PAGE MEMORY" MODE.
00137A 4100 C6 11     A        LDAB    #17       INIT. LOOP COUNTER
00138A 4102 CE 4046   A        LDX     #IBUF
00139A 4105 BD 4194   A        JSR     BUSY      TEST BUSY FLIP-FLOP
00140A 4108 B6 EC01   A        LDAA    TRB       FIRST READ TO LOAD THE TRANSFER
                                                 REGISTE
00142A 410B BD 4194   A RDCAR0 JSR     BUSY
00143A 410E B6 EC00   A        LDAA    TRA       READ FIRST BYTE OF CHAR. CODE
00144A 4111 A7 00     A        STAA    0,X       STORE INTO IBUF.
00145A 4113 08                 INX
00146A 4114 B6 EC01   A        LDAA    TRB       READ 2ND. BYTE OF CHAR. CODE
00147A 4117 A7 00     A        STAA    0,X
00148A 4119 08                 INX
00149A 411A 5A                 DECB              DECREMENT LOOP COUNTER.
00150A 411B 26 EE 410B         BNE     RDCAR0
```

```
PAGE  004  APPLI   .SA:0

00152                         *************************************************
00153                         * WRITING INTO AN EXTENDED CHARACTER GENERATOR.
00154                         * FOUR EXTENDED CHARACTER, A0 THROUGH A3 ARE DEFINED
00155                         * THEN WILL BE WRITTEN INTO THE PAGE MEMORY.
00156                         *************************************************

00158A 411D CE 401C  A              LDX     #CAR1
00159A 4120 4F                      CLRA
00160A 4121 C6 A0    A              LDAB    #$A0
00161A 4123 BD 4256  A LOOP5        JSR     WRGEN
00162A 4126 5C                      INCB
00163A 4127 C1 A3    A              CMPB    #$A3
00164A 4129 23 F8 4123              BLS     LOOP5
00165A 412B 4F                      CLRA              SET M REGISTER INTO "WRITE PAGE
00166A 412C BD 41E6  A              JSR     LOADM     MEMORY WITHOUT INC." MODE.
00167A 412F 86 13    A              LDAA    #19       SET CURSOR AT LOCATION
00168A 4131 C6 11    A              LDAB    #17       X=19,Y=17.
00169A 4133 BD 41C2  A              JSR     LOADXY
00170A 4136 86 0F    A              LDAA    #$0F      COLOR AND ATTRIBUTE DEFINITION.
00171A 4138 C6 A0    A              LDAB    #$A0      FIRST CHARACTER CODE
00172A 413A BD 4194  A              JSR     BUSY
00173A 413D B7 EC00  A              STAA    TRA       WRITE INTO PAGE MEMORY.
00174A 4140 F7 EC01  A              STAB    TRB
00175A 4143 5C                      INCB
00176A 4144 BD 4194  A              JSR     BUSY
00177A 4147 B7 EC00  A              STAA    TRA
00178A 414A F7 EC01  A              STAB    TRB
00179A 414D 86 13    A              LDAA    #19
00180A 414F C6 12    A              LDAB    #18       CURSOR AT X=19,Y=18
00181A 4151 BD 41C2  A              JSR     LOADXY
00182A 4154 86 0F    A              LDAA    #$0F
00183A 4156 C6 A2    A              LDAB    #$A2
00184A 4158 BD 4194  A              JSR     BUSY
00185A 415B B7 EC00  A              STAA    TRA
00186A 415E F7 EC01  A              STAB    TRB
00187A 4161 5C                      INCB
00188A 4162 BD 4194  A              JSR     BUSY
00189A 4165 B7 EC00  A              STAA    TRA
00190A 4168 F7 EC01  A              STAB    TRB

00192                         *************************************************
00193                         * ROLLUP AND ROLL-DOWN EXAMPLE
00194                         *************************************************

00196A 416B 4F                      CLRA
00197A 416C 4C                ROLLUP INCA
00198A 416D BD 41CA  A              JSR     LOADY0    LOAD A CONTENTS INTO Y0 REGISTER.
00199A 4170 BD 42B6  A              JSR     TEMPO     WAIT
00200A 4173 81 17    A              CMPA    #23       23 ROWS ROLLED UP?
00201A 4175 26 F5 416C              BNE     ROLLUP
00202A 4177 4A                ROLLDO DECA
00203A 4178 BD 41CA  A              JSR     LOADY0
00204A 417B BD 42B6  A              JSR     TEMPO
00205A 417E 4D                      TSTA
00206A 417F 26 F6 4177              BNE     ROLLDO
```

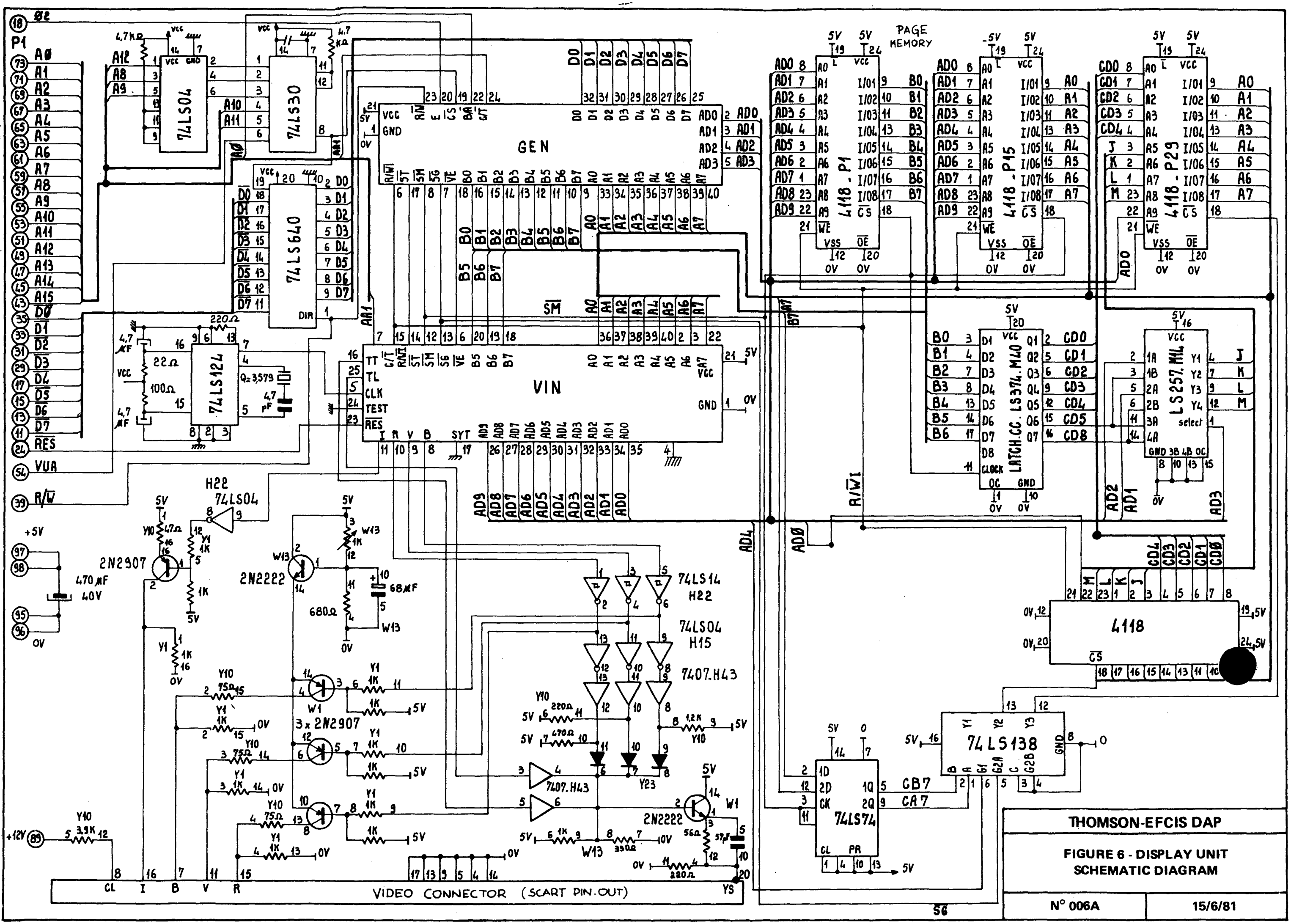
```
PAGE  005  APPLI   .SA:0

00208                  *************************************************
00209                  * READING THE GEN ALPHANUMERIC CHARACTER GENERATOR
00210                  * INTO A MEMORY BUFFER STARTING AT $6000.
00211                  *************************************************

00213A 4181 CE 6000  A        LDX    #$6000
00214A 4184 4F                CLRA
00215A 4185 5F                CLRB                INIT A AND B TO 0.
00216A 4186 36         DEBO   PSHA
00217A 4187 37                PSHB
00218A 4188 BD 41F4  A        JSR    RDGEN    READ TEN SLICES.
00219A 418B 33                PULB
00220A 418C 32                PULA
00221A 418D 5C                INCB
00222A 418E C1 80    A        CMPB   #128
00223A 4190 26 F4 4186        BNE    DEBO
00224A 4192 3F                SWI
00225A 4193    3F    A        FCB    $3F      REBOOT MDOS.
00226                  *************************************************
00227                  *  BUSY: THIS SUBROUTINE TESTS THE GEN BUSY
00228                  *        FLIP-FLOP IN CRA REGISTER BIT 7.BEFOR
00229                  *        ACCEDING TO ANY TRANSFER REGISTER OF
00230                  *        GEN,BUSY MUST BE CALLED.
00231                  *        REGISTER DESTROYED : NONE.
00232                  *************************************************
00233                  *
00234A 4194 36         BUSY   PSHA
00235A 4195 B6 EC02  A BUSYO  LDAA   CRA      READ CRA REGISTER
00236A 4198 84 80    A        ANDA   #$80     TEST CRA BIT 7
00237A 419A 26 F9 4195        BNE    BUSYO
00238A 419C 32                PULA                RESTORE ACCUMULATOR
00239A 419D 39                RTS
00240                  *
00241                  *
00242                  *
00243                  *************************************************
00244                  *  ROW   : SETS THE CURSOR AT THE BEGINNING OF
00245                  *         THE ROW WHOSE VALUE IS IN ACCUMULATOR
00246                  *         B.
00247                  *         REGISTER DESTROYED : B.
00248                  *************************************************
00249                  *
00250A 419E BD 4194  A ROW    JSR    BUSY
00251A 41A1 F7 EC02  A        STAB   CRA      WRITE B CONTENTS INTO CRA
00252A 41A4 C6 00    A        LDAB   #BEGROW
00253A 41A6 F7 EC03  A        STAB   CRB      WRITE "BEGIN ROW" COMMAND INTO CRB.
00254A 41A9 39                RTS
```

```
PAGE  006  APPLI   .SA:0

00256                        *
00257                        ***********************************************
00258                        *  LOADX: LOADS THE CURSOR X COUNTER WITH THE A
00259                        *         ACCUMULATOR CONTENTS.
00260                        *         REGISTER DESTROYED : A.
00261                        ***********************************************
00262                        *
00263A 41AA BD 4194    A LOADX   JSR    BUSY
00264A 41AD B7 EC02    A         STAA   CRA       WRITE A CONTENTS INTO CRA
00265A 41B0 86 40      A         LDAA   #LDX      LOAD "LOAD X" COMMAND INTO CRB
00266A 41B2 B7 EC03    A         STAA   CRB
00267A 41B5 39                   RTS
00268                        *
00269                        ***********************************************
00270                        *  LOADY: LOADS THE CURSOR Y COUNTER WITH THE A
00271                        *         ACCUMULATOR CONTENTS.
00272                        *         REGISTER DESTROYED : A.
00273                        ***********************************************
00274                        *
00275A 41B6 BD 4194    A LOADY   JSR    BUSY
00276A 41B9 B7 EC02    A         STAA   CRA       WRITE A CONTENTS INTO CRA
00277A 41BC 86 20      A         LDAA   #LDY      LOAD "LOAD Y" COMMAND INTO CRB
00278A 41BE B7 EC03    A         STAA   CRB
00279A 41C1 39                   RTS
00280                        *
00281                        ***********************************************
00282                        * LOADXY : SETS THE CURSOR POSITION (X,Y).
00283                        *          ENTRY PARAMETERS:
00284                        *          ACC.A=X VALUE.
00285                        *          ACC.B=Y VALUE.
00286                        *          REGISTER A IS DESTROYED.
00287                        ***********************************************
00288                        *
00289A 41C2 BD 41AA    A LOADXY  JSR    LOADX     LOAD X COUNTER.
00290A 41C5 17                   TBA
00291A 41C6 BD 41B6    A         JSR    LOADY     LOAD Y COUNTER.
00292A 41C9 39                   RTS
00293                        *
00294                        ***********************************************
00295                        *  LOADY0: THIS SUBROUTINE LOAD THE Y0 REGISTER
00296                        *          WITH THE A ACCUMULATOR CONTENTS.
00297                        *          REGISTER DESTROYED : NONE.
00298                        ***********************************************
00299                        *
00300A 41CA 36           LOADY0  PSHA
00301A 41CB BD 4194    A         JSR    BUSY
00302A 41CE B7 EC02    A         STAA   CRA       LOAD CRA TRANSFER REGISTER
00303A 41D1 86 C0      A         LDAA   #LDY0     LOAD "LOADY0" COMMAND INTO
00304A 41D3 B7 EC03    A         STAA   CRB       CRB TRANSFER REGISTER.
00305A 41D6 32                   PULA
00306A 41D7 39                   RTS
```

```
PAGE  007  APPLI   .SA:0

00308                     *
00309                     ******************************************************
00310                     *  LOADR : THIS SUBROUTINE LOADS THE R REGISTER WITH
00311                     *        THE A ACCUMULATOR CONTENTS.
00312                     *        REGISTER DESTROYED : NONE.
00313                     ******************************************************
00314                     *
00315A 41D8 36            LOADR  PSHA
00316A 41D9 BD 4194  A           JSR    BUSY
00317A 41DC B7 EC02  A           STAA   CRA      LOAD CRA TRANSFER REGISTER
00318A 41DF 86 A0    A           LDAA   #LDR     LOAD CRB WITH "LOADR" COMMAND.
00319A 41E1 B7 EC03  A           STAA   CRB
00320A 41E4 32                   PULA
00321A 41E5 39                   RTS
00322                     *
00323                     **************************************************
00324                     *  LOADM: LOADS THE M REGISTER WITH THE A
00325                     *         ACCUMULATOR CONTENTS.
00326                     *         REGISTER DESTROYED : NONE.
00327                     **************************************************
00328                     *
00329A 41E6 36            LOADM  PSHA
00330A 41E7 BD 4194  A           JSR    BUSY
00331A 41EA B7 EC02  A           STAA   CRA      LOAD CRA TRANSFER REGISTER.
00332A 41ED 86 80    A           LDAA   #LDM     LOAD CRB REGISTER WITH "LOAD M"
00333A 41EF B7 EC03  A           STAA   CRB      COMMAND.
00334A 41F2 32                   PULA
00335A 41F3 39                   RTS
00336                     *
00337                     ***************************************************************
00338                     *  RDGEN:THIS SUBROUTINE READS 10 CHARACTER SLICES FROM
00339                     *        A CHARACTER GENERATOR.THE PAGE MEMORY LOCATION
00340                     *        (X=0,Y=31) IS USED TO STORE THE CHARACTER CODE
00341                     *        GIVEN AS A PARAMETER IN A AND B ACCUMULATOR.
00342                     *        THIS LOCATION CONTENTS IS FIRST SAVED INTO
00343                     *        SAV AND WILL BE RESTORED AT THE END OF THE
00344                     *        ROUTINE.
00345                     *        ENTRY PARAMETERS : A AND B ACCUMULATOR CONTAIN
00346                     *        THE CHARACTER CODE WHOSE THE SLICES ARE TO BE
00347                     *        READ.
00348                     *        X = MEMORY ADDRESS OF THE BUFFER WHERE THE
00349                     *        SLICES ARE TO BE STORED.
00350                     *        EXTERNAL REFERENCES:ROW,LOADM,BUSY.
00351                     *        REGISTER DESTROYED: A,B,X.
00352                     ***************************************************************
00353                     *
00354A 41F4 36            RDGEN  PSHA            SAVE PARAMETER ON STACK
00355A 41F5 37                   PSHB
00356A 41F6 C6 1F    A           LDAB   #31      SET CURSOR AT THE BEGINNING OF ROW
00357A 41F8 BD 419E  A           JSR    ROW      NUMBER 31 (SERVICE ROW).
00358A 41FB 86 60    A           LDAA   #$60     SET M REGISTER IN "READ PAGE MEMORY
00359A 41FD BD 41E6  A           JSR    LOADM    WITHOUT CURSOR INCREMENTATION" MODE.
00360A 4200 BD 4194  A           JSR    BUSY
00361A 4203 B6 EC01  A           LDAA   TRB      ACCESS TO TRB IN ORDER TO LOAD THE
00362A 4206 BD 4194  A           JSR    BUSY     DATA TRANSFER REGISTER.
```

```
PAGE  008  APPLI   .SA:0

00364A 4209 B6 EC00  A        LDAA   TRA       READ CHARACTER CODE FROM PAGE MEMORY
00365A 420C F6 EC01  A        LDAB   TRB       AT LOCATION (0,31).
00366A 420F B7 4044  A        STAA   SAV       SAVE INTO SAV AND SAV+1.
00367A 4212 F7 4045  A        STAB   SAV+1
00368A 4215 86 40    A        LDAA   #$40      SET M REGISTER IN "WRITE INTO P.M.
00369A 4217 BD 41E6  A        JSR    LOADM     WITHOUT INCREMENTATION" MODE.
00370A 421A 33                PULB
00371A 421B 32                PULA             RESTORE INITIAL PARAMETER FROM STACK.
00372A 421C BD 4194  A        JSR    BUSY
00373A 421F B7 EC00  A        STAA   TRA       WRITE CHAR. CODE GIVEN AS PARAMETER
00374A 4222 F7 EC01  A        STAB   TRB       INTO P.M. LOCATION (0,31)
00375A 4225 86 A0    A        LDAA   #$A0      SET M REGISTER IN "READ SLICES" MODE
00376A 4227 BD 41E6  A        JSR    LOADM
00377A 422A C6 0A    A        LDAB   #10       INIT. LOOP COUNTER FOR 10 READ
00378A 422C BD 4194  A        JSR    BUSY      OPERATIONS.
00379A 422F B6 EC01  A        LDAA   TRB       READ TRB IN ORDER TO LOAD THE
00380                  *                       TRANSFER REGISTER.

00382A 4232 BD 4194  A RDGEN0 JSR    BUSY
00383A 4235 B6 EC00  A        LDAA   TRA       READ A SLICE FROM TRA.
00384A 4238 A7 00    A        STAA   0,X       STORE INTO BUFFER.
00385A 423A 08                INX
00386A 423B B6 EC01  A        LDAA   TRB       READ TRB TO LOAD AGAIN A SLICE.
00387A 423E 5A                DECB             DECREMENT LOOP COUNTER.
00388A 423F 26 F1 4232        BNE    RDGEN0    IF NOT 0, LOOP AGAIN.

00390A 4241 86 40    A        LDAA   #$40      SET M REGISTER IN "WRITE INTO P.M.
00391A 4243 BD 41E6  A        JSR    LOADM     WITHOUT CURSOR INCREMENTATION" MODE.
00392A 4246 B6 4044  A        LDAA   SAV
00393A 4249 F6 4045  A        LDAB   SAV+1
00394A 424C BD 4194  A        JSR    BUSY
00395A 424F B7 EC00  A        STAA   TRA       RESTORE CHARACTER CODE INTO PAGE
00396A 4252 F7 EC01  A        STAB   TRB       MEMORY LOCATION (0,31).
00397A 4255 39                RTS
00398                  *
00399                  ************************************************************
00400                  *  WRGEN:THIS SUBROUTINE WRITES 10 CHARACTER SLICES INTO
00401                  *        A CHARACTER GENERATOR.THE PAGE MEMORY LOCATION
00402                  *        (X=0,Y=31) IS USED TO STORE THE CHARACTER CODE
00403                  *        GIVEN AS A PARAMETER IN A AND B ACCUMULATOR.
00404                  *        THIS LOCATION CONTENTS IS FIRST SAVED INTO
00405                  *        SAV AND WILL BE RESTORED AT THE END OF THE
00406                  *        ROUTINE.
00407                  *        ENTRY PARAMETERS : A AND B ACCUMULATOR CONTAIN
00408                  *        THE CHARACTER CODE WHOSE THE SLICES ARE TO BE
00409                  *        WRITTEN.
00410                  *        X = MEMORY ADDRESS OF THE BUFFER WHERE THE
00411                  *        SLICES ARE TO BE STORED.
00412                  *        EXTERNAL REFERENCES:ROW,LOADM,BUSY.
00413                  *        REGISTER DESTROYED: X.
00414                  ************************************************************
00415A 4256 36         WRGEN  PSHA             SAVE PARAMETER ON STACK
00416A 4257 37                PSHB
00417A 4258 36                PSHA             SAVE A AND B
00418A 4259 37                PSHB
00419A 425A C6 1F    A        LDAB   #31       SET CURSOR AT THE BEGINNING OF ROW
00420A 425C BD 419E  A        JSR    ROW       NUMBER 31 (SERVICE ROW).
```

```
PAGE  009  APPLI   .SA:0

00422A 425F 86 60    A        LDAA   #$60     SET M REGISTER IN "READ PAGE MEMORY
00423A 4261 BD 41E6  A        JSR    LOADM    WITHOUT CURSOR INCREMENTATION" MODE.
00424A 4264 BD 4194  A        JSR    BUSY
00425A 4267 B6 EC01  A        LDAA   TRB      ACCESS TO TRB IN ORDER TO LOAD THE
00426A 426A BD 4194  A        JSR    BUSY     TRANSFER REGISTER.
00427A 426D B6 EC00  A        LDAA   TRA      READ CHARACTER CODE FROM PAGE MEMORY
00428A 4270 F6 EC01  A        LDAB   TRB      AT LOCATION (0,31).
00429A 4273 B7 4044  A        STAA   SAV      SAVE INTO SAV AND SAV+1.
00430A 4276 F7 4045  A        STAB   SAV+1
00431A 4279 86 40    A        LDAA   #$40     SET M REGISTER IN "WRITE INTO P.M.
00432A 427B BD 41E6  A        JSR    LOADM    WITHOUT INCREMENTATION" MODE.
00433A 427E 33                PULB
00434A 427F 32                PULA            RESTORE INITIAL PARAMETER FROM STACK.
00435A 4280 BD 4194  A        JSR    BUSY
00436A 4283 B7 EC00  A        STAA   TRA      WRITE CHAR. CODE GIVEN AS PARAMETER
00437A 4286 F7 EC01  A        STAB   TRB      INTO P.M. LOACATION (0,31)
00438A 4289 86 80    A        LDAA   #$80     SET M REGISTER IN "WRITE SLICES" MODE
00439A 428B BD 41E6  A        JSR    LOADM
00440A 428E C6 0A    A        LDAB   #10      INIT. LOOP COUNTER FOR 10 WRITE .

00442A 4290 BD 4194  A WRGENO JSR    BUSY
00443A 4293 A6 00    A        LDAA   0,X      WRITE A SLICE.
00444A 4295 B7 EC00  A        STAA   TRA
00445A 4298 B7 EC01  A        STAA   TRB
00446A 429B 08                INX
00447A 429C 5A                DECB            DECREMENT LOOP COUNTER.
00448A 429D 26 F1 4290        BNE    WRGENO   IF NOT 0, LOOP AGAIN.

00450A 429F 86 40    A        LDAA   #$40     SET M REGISTER IN "WRITE INTO P.M.
00451A 42A1 BD 41E6  A        JSR    LOADM    WITHOUT CURSOR INCREMENTATION" MODE.
00452A 42A4 B6 4044  A        LDAA   SAV
00453A 42A7 F6 4045  A        LDAB   SAV+1
00454A 42AA BD 4194  A        JSR    BUSY
00455A 42AD B7 EC00  A        STAA   TRA      RESTORE CHARACTER CODE INTO PAGE
00456A 42B0 F7 EC01  A        STAB   TRB      MEMORY LOCATION (0,31).
00457A 42B3 33                PULB
00458A 42B4 32                PULA            RESTORE A AND B FROM STACK.
00459A 42B5 39                RTS

00461                  *  TEMPO : WAIT SUBROUTINE.
00462                  *        REGISTER X IS DESTROYED.

00464A 42B6 36         TEMPO  PSHA
00465A 42B7 86 05    A        LDAA   #5
00466A 42B9 CE FFFF  A TEMP1  LDX    #$FFFF
00467A 42BC 09         TEMP2  DEX
00468A 42BD 26 FD 42BC        BNE    TEMP2
00469A 42BF 4A                DECA
00470A 42C0 26 F7 42B9        BNE    TEMP1
00471A 42C2 32                PULA
00472A 42C3 39                RTS
00474           4096 A        END    DEB
TOTAL ERRORS 00000--00000
```



THOMSON-EFCIS DAP

FIGURE 6 - DISPLAY UNIT SCHEMATIC DIAGRAM

N° 006A

15/6/81