New CMOS Vacuum Fluorescent Drivers Enable Three Chip System to Provide Intelligent Control of Dot Matrix VF Display

INTRODUCTION

Vacuum Fluorescent (VF) displays are becoming more and more common in a variety of applications. Manufacturers of everything from Automobiles to Video Recorders have taken advantage of these easy to read displays. VF displays are available in a wide variety of configurations; clock displays, calculator displays, multi-segment, and dot matrix displays are readily available at a low cost. This application note develops and covers in some detail a small CMOS system consisting of a single chip microcontroller and two display drivers which control a 20 character, 5 x 7 dot matrix VF display.

Figure 1 shows the schematic of the system. The microcontroller, a COPS™ 424C, receives a character in ASCII form from the host system, stores the ASCII value of the character in its onboard RAM, converts the ASCII value to a 5 byte data word suitable for the display drivers and displays it on the VF display. The COPS also refreshes the display continuously while performing character update, much like a dumb terminal. Not including the address decoding logic, this application requires only the onboard RAM and ROM of the COPS424C, and National's MM58341 and MM58348 VF display drivers. If a steady message or a scrolling sentence is desired, only small changes in the COPS software are re-

quired. In this case the messages could be stored in the ROM of the COPS and the need for a host system would be eliminated.

VF DISPLAY AND VF DISPLAY DRIVER REQUIREMENTS

The display used in this application was an Itron #DC205G2. This 20 segment, 5 x 7 dot matrix, multiplexed display required a filament voltage of 5.7 Vac and a filament current of 37 mAac. The anode and grid voltages were supplied by the display drivers. The voltage and current requirements vary considerably for different displays depending on the size and number of characters, and the configuration (dot matrix, 7 segment, 14 segment, etc.). To determine the voltage requirements for a particular display, a simple calculation can be made. If maximum possible brightness of the display is desired, the following equation must be true:

$E_t \geq E_b + E_k + (I_b) (R_{on})$ where:

E_t is the total Voltage of the display drvier or $|V_{dis}| + V_{dd}$

E_k is the display Cathode Bias Voltage

$E_b = E_c$ is the typical Anode or Grid Voltage (V_{p-p})

I_b is the typical anode current (mAp-p)

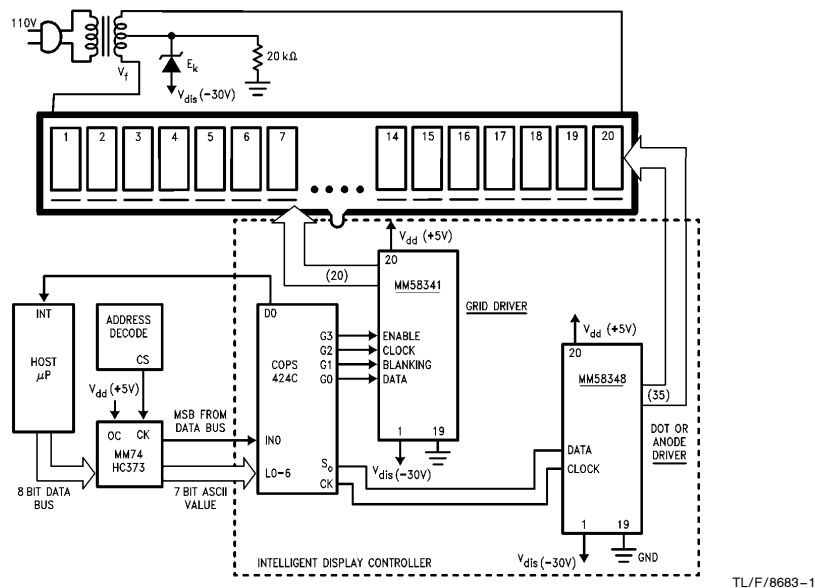R_{on} is the display driver output impedance (Ω)

FIGURE 1. System Diagram Showing the Basic 3-Chip Display Controller and the Interface to a Microprocessor System

TL/F/8683–1

COPS™ is a trademark of National Semiconductor Corp.

If the maximum brightness is not desired, the following equation can be used: $(E_t)(1.2) \geq E_b + E_k + (I_b)(R_{on})$. In this application, the calculated E_t was 42.25V, however, the display was legible under normal lighting conditions, with an E_t as low as 25V. If your display requires more than the 35V output of the MM58341 and MM58348, pin for pin compatible 60V VF Display Drivers (MM58241, MM58248) are available.

Figure 2 shows the relationship between the required VF display voltages. The cut-off voltage (E_k) is set by the Zener diode on the center tap of the filament transformer. This value is given in the VF display data sheet.

Avoiding Flicker and Pulsing

There are two different conditions which may cause the display to appear to flicker. The first is the refresh rate. This is particularly a problem on displays where the micro-controller must up-date more than 25 characters. Since the human eye begins to notice flicker at about 40 Hz, a display with a refresh rate less than that will appear to be flashing on and off.

The second type of flicker occurs when the refresh rate is between 40 Hz and 90 Hz. In this case, the display will appear to be rolling rather than flashing. This condition occurs when the refresh rate and the filament frequency are close together. If a character is only on during the time when the filament voltage is negative, it will appear to be slightly brighter than the character next to it which may only be on during the positive cycle of the filament voltage. If this is the case, as it was in this application, the simplest solution is to increase the frequency of the filament. A DC oscillator circuit, such as the one shown in *Figure 3*, can be used to replace the AC voltage source. The filament frequency can be easily adjusted to eliminate this condition.
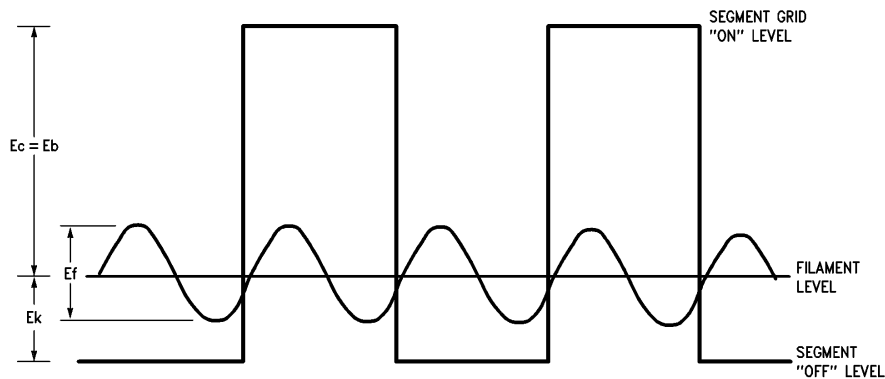

TL/F/8683–2

FIGURE 2. Voltage Levels for VF Display

TL/F/8683–3

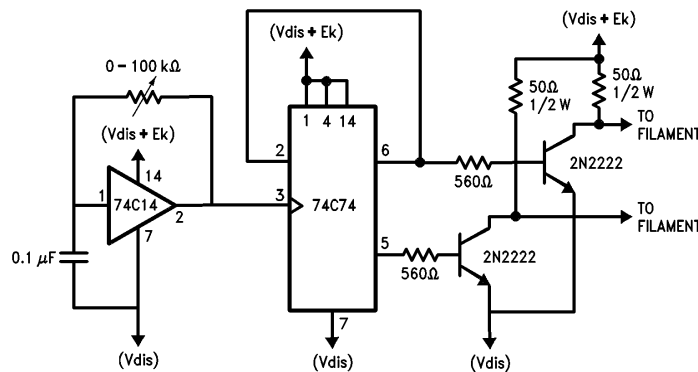**FIGURE 3. Filament Oscillator Circuit**

2

VF Display Drivers

Two high voltage display drivers were needed to control the VF display. A MM58341, was used to control the grids and a MM58348 was used to control the individual pixels or anodes. Both of these drivers receive serial information and output 32 and 35 segments of data respectively.

The MM58341 has three control pins which make it ideal for controlling the grids of a VF display. The blanking control pin will turn off all segments of the display when a logic '1' is applied to this pin. This is particularly important for reducing ghosting, and controlling brightness. Ghosting is a condition where the last characters shadow appears behind the character being displayed. The enable pin acts as an envelope for the input signal. Only while it is at a logic '1' level will the circuit accept clock inputs. When the pin goes low, all the data is latched and displayed. A data out pin is also provided for cascading. If the display has more than 32 grids, a second grid driver can be cascaded by connecting the data out pin to the input data for the second grid driver.

The MM58348 is a 35 bit shift register and latch which is used to control each pixel or dot. When a leading 1, followed by 35 bits of data, is received, the data is latched and displayed. The chip is automatically reset upon power up.

MULTIPLEXED DISPLAY REFRESH TIMING

Considering first the digit driver (MM58341), it becomes clear that the digits must be enabled or refreshed sequentially and that this process must be continuous regardless if the display data has changed. The data for the MM58341 is simply a 1 followed by 19 zeroes where the 1 is shifted through the internal registers of the MM58341. As each digit is enabled, the corresponding segment data is displayed. To insure that no ghosting effects are seen during the transition between digits, the blanking control is activiated just before the data is latched into the dot or anode driver and deactivated just after the data has been latched. During this time when the blanking control is activated, the grid driver is clocked shifting the 1 to the next location. *Figure 4* shows the micro-controller waveforms and the resultant display waveforms for the 20 character display.
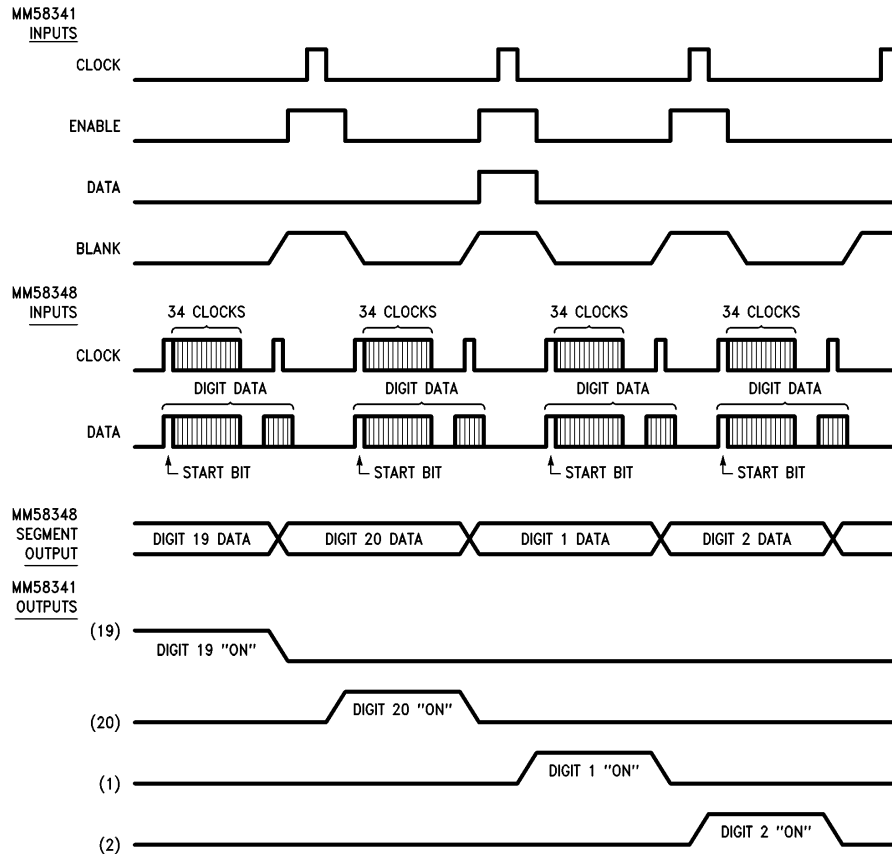

TL/F/8683–4

FIGURE 4. Timing Diagram

3

In between digit strobes, the segment data is updated. The first 34 bits of segment data are set up in the dot driver and the blanking signal is activated to disable all 20 digits. The 35th bit of data is clocked in, updating the segments. Since the MM58348 resets its internal shift register each time the data is latched, it can accept all but the final data bit while still displaying the previous digit. The digit driver is then clocked, shifting the digit strobe to the next position. The enable is then brought low, enabling the next digit. Finally blanking control is deactivated and the data displayed.

During the time which the blanking control is high, the order in which the segments or the digits are updated is not critical. Since this occurs while the display is blank. The digit driver may be clocked first, or the segments could be changed first. In general, the philosophy for the driving this VF multiplexed display is outlined in *Figure 5*.

HOST INTERFACE AND PROGRAMMING

With a minimal amount of address decoding and an eight bit latch, COPS can be interfaced with a common microprocessor bus. When a character has been input into the host to be displayed, the ASCII value of that character is latched into the eight bit latch (MM74HC373) and is read on the L port (L0–6) of the COPS. The MSB of the ASCII value must be a logic 1. This MSB is the signal to the COPS that a new character is being presented. Once the character has been stored, an interrupt is sent from the COPS to the host through the D-0 port. The COPS checks for a new character being input every 200 μs. If a character is being sent, 1 ms is required to store that character in the RAM of the COPS. With the COPS controlling the display, the host micro-processor is not being tied down with character look-up and display refresh. A simple flowchart of the host requirements is shown in *Figure 6*.

COPS SOFTWARE

There are four main sections of the COPS software. The first section, the initialization of the RAM, sets up the RAM as shown in *Figure 7*. A '0' is stored in all of the LSB positions and a '2' is stored in all of the MSB positions. Since the COPS is in a constant display loop, this is necessary to insure a blank display. 20H is the ASCII value of a space. With the RAM set up in this way, a maximum of 28 characters can be stored in RAM. Since the display in this application is only 20 characters long, RAM locations M1,4 to M1,11 and M3,4 to M3,11 are not used. RAM locations 1,12 to 1,15 and 3,12 to 3,15 are used as temporary storage throughout the program and cannot be used for character storage.

The second part of the program, stores the new characters sent by the host CPU in RAM. Once a character has been sent, this section of the program checks the ASCII value of that character to see if it is a control character or a display character. If it is a display character, the character is stored in RAM and an interrupt is sent to the host. There are three control characters which the COPS program will recognize. Cursor forward (ASCII value 08H) moves the cursor forward without destroying the data, cursor backwards (ASCII value 0CH) moves the cursor backwards without destroying the data, and return (ASCII value 0DH) will clear the display and put the cursor at the beginning of the display. To recognize and store a character, 1 ms is required.

The third part of the program, the display loop, is the heart of the program. Unless a new character has been detected, the program is always in this loop. This section does the
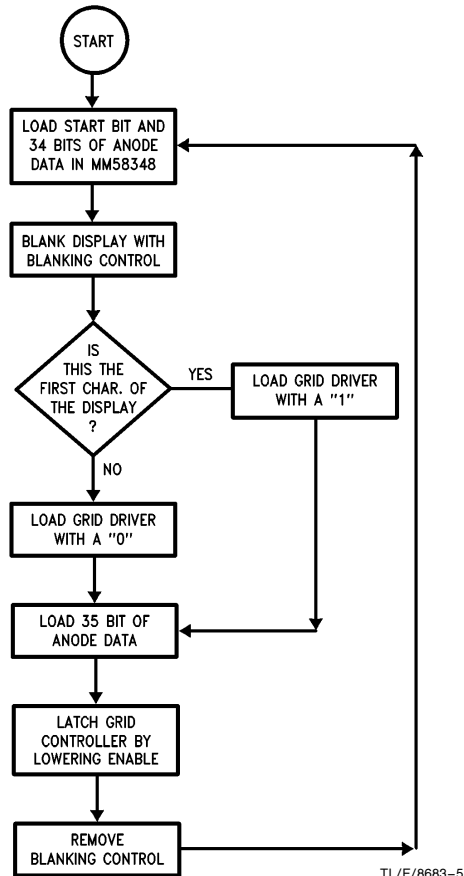

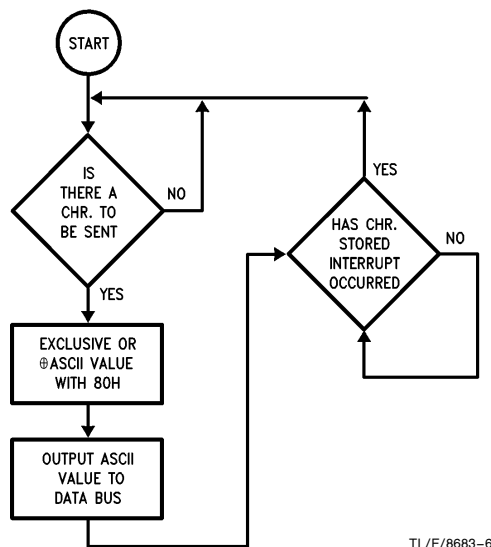FIGURE 5. Flowchart for Display Drivers

TL/F/8683–5

FIGURE 6. Host System Flowchart

TL/F/8683–6

4

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
LSB Chr 1	LSB Chr 2	LSB Chr 3	LSB Chr 4	LSB Chr 5	LSB Chr 6	LSB Chr 7	LSB Chr 8	LSB Chr 9	LSB Chr 10	LSB Chr 11	LSB Chr 12	LSB Chr 13	LSB Chr 14	LSB Chr 15	LSB Chr 16	M0
MSB Pointer	LSB Pointer	Temp. ASCII STORAGE										LSB Chr 17	LSB Chr 18	LSB Chr 19	LSB Chr 20	M1
MSB Chr 1	MSB Chr 2	MSB Chr 3	MSB Chr 4	MSB Chr 5	MSB Chr 6	MSB Chr 7	MSB Chr 8	MSB Chr 9	MSB Chr 10	MSB Chr 11	MSB Chr 12	MSB Chr 13	MSB Chr 14	MSB Chr 15	MSB Chr 16	M2
Temp. Storage of Pointer												MSB Chr 17	MSB Chr 18	MSB Chr 19	MSB Chr 20	M3

FIGURE 7. COPS RAM Map

Matrix	PAD	Column 1	Column 2	Column 3	Column 4	Column 5	PAD
Binary	0 0 0 1	0 0 1 1 1 1 1 0	1 0 1 0 0 0 1 0	0 1 0 0 0 0 1 0	1 0 0 0 0 0 1 1	1 1 1 0	
Hex.	13	EA	24	28	3E	

FIGURE 8

character font look-up, shifts the character data out the COPS serial port to the MM58348, and controls the MM58341 through the four bit parallel port (G0–4). Because the most significant nibble of the program counter is used as part of some COPS instructions, it is important that parts of the program are located at specific locations in ROM.

The final part of the program is the data. Each character is represented by a 5 byte data word. Each byte of the data word is stored at a different location in ROM. Fonts for characters with the ASCII values from 20H–5AH have already been stored in ROM. These characters can be changed or more characters can be added. The only limitation to the number of characters is the amount of available ROM.

CREATING THE 5 BYTE DATA WORD

Any number or combination of pixels or dots can be turned on at a time. To create a new character, it is easiest to first create a binary string which represents the character. A '1' in the binary string will turn on the pixel, a '0' will turn it off. To create this string, start in the upper left corner of the matrix and go down the columns.

The letter 'A' (*Figure 9*) would have a binary string shown in *Figure 8*. The data must be padded to make it an even 5 bytes in length. The pad at the beginning of the data (0001) is used as the leading 1 for the MM58348. The one bit pad at the end of the binary string must be a 0. If a 1 were sent as the pad, it would be used as the start bit for the next character.

The 5 byte data word that would be stored in ROM and represent the letter 'A' would then be 13EA24283E.

STORING THE DATA IN ROM

The 5 bytes of data are stored in 5 different locations in ROM. The first byte of data will be stored, LSB first, at location 200H plus the ASCII value of the character. For example, the ASCII value of the letter 'A' is 41H. The first byte of data for the letter 'A' would be stored, least significant bit first, at 241H. The second byte of data is stored at the location of the first data byte plus 60H or in this case at 2A1H. The location of the third byte is 40H plus the location of the

second byte. In this case, the third byte of data would be stored at 2E1H. The fourth byte of data is stored at 300H plus the ASCII value of the character or at 341H for the letter 'A'. The final byte of data is stored 40H from the fourth byte or at 381H. Remember the LSB of each byte is stored first. Table I shows the locations in ROM and the values stored in them for the letter 'A'.

This application shows a VF display controller designed with a minimum number of IC's. If additional information about VF displays or VF display drivers is required, refer to Application Note AN-371 (The MM58348/342/341/248/242/241 direct drive Vacuum Fluorescent (VF) Displays.

TABLE I. Character Data of 'A' and Its Locations in ROM

Address In ROM	Data Stored
0241H	31
02A1H	AE
02E1H	42
0341H	82
0381H	E3

TL/F/8683–7

FIGURE 9. 5 x 7 Character as Stored in ROM

Section 1 of COPS Software

```
          .CHIP 424C             ;DEFINES COPS CHIP
;THIS SECTION INITIALIZES THE RAM IN THE COPS BY LOADING A
;2 IN THE MSB AND A 0 IN THE LSB LOCATIONS OF EACH CHARACTER.
;IT ALSO STOPS THE CLOCK AND SETS THE POINTER AT THE FIRST
;CHARACTER OF THE DISPLAY.


RESET:          CLRA
                LBI 3,15       ;LOADS A 2 IN ALL
                JSR CLEAR2     ;MSB LOCATIONS
                LBI 2,15       ;LOADS A 2 IN ALL
                JSR CLEAR2     ;MSB LOCATIONS
                LBI 1,15       ;LOADS A 0 IN ALL
                JSR CLEAR      ;LSB LOCATIONS
                LBI 0,15       ;LOADS A 0 IN ALL
                JSR CLEAR      ;LSB LOCATIONS

                CLRA           ;LOADS POINTER IN RAM
                XAD 1,15       ;MSB IN 1,0F
                CLRA
                AISC 15        ;LSB IN 1,0E
                XAD 1,14

                RC             ;RESETS CARRY TO
                XAS            ;    STOP CLOCK
                JMP START

CLEAR:          CLRA           ;CLEARS REGISTORS
                XDS 0
                JMP CLEAR
                RET

CLEAR2:         CLRA           ;PUTS A 2 IN REGISTORS
                AISC 02
                XDS 0
                JMP CLEAR2
                RET
```

Section 2 of COPS Software

```
;THIS SECTION OF CODE IS ONLY EXECUTED WHEN A NEW
;CHARACTER HAS BEEN ENTERED. IF THE CHARACTER IS
;A CONTROL CHARACTER, THE CURSOR IS MOVED ACCORDINGLY,
;OTHERWISE THE CHARACTER IS STORED IN THE RAM OF THE COPS.


          ;NEW CHARACTER HAS BEEN ENTERED
NEW:            LBI 1,0C       ;DUMMY POINTER
                INL            ;READS ASCII FROM
                XIS  0         ;DATA BUS
                X 0
                LDD 1,0D
                RC             ;CHAR.  MSB=0 THEN YES
                AISC 15        ;MSB<>0 THEN NO
                JMP SPECIAL
                AISC 01
                LDD 1,0E       ;STORE ASCII IN RAM
                CAB
                LDD 1,0F
                XABR
                LDD 1,0C       ;MSB IN 1,0C
                X 2
                LDD 1,0D       ;LSB IN 1, 0D
                X 0
```

```
            JSR CURFOR
            LBI 0,01        ;SENDS INTERRUPT TO
            OBD             ;  HOST.  CHAR. IS
            LBI 0,0         ;  STORED IN RAM
            OBD
            JMP START


        ;SPECIAL CHARS. (CR, LF, CLEAR DISPLAY)


CURFOR:     LDD 1,0E        ;MOVES CURSOR FORWARD ONE
            COMP            ;SPACE.  IF CURSOR IS
            AISC 01         ;MOVED BEYOND THE END OF
            JMP OK          ;DISPLAY, IT WRAPS AROUND
            AISC 0F         ;TO THE OTHER END.  DATA IS
            XAD 1,0E        ;NOT DESTROYED BY MOVING
            CLRA            ;CURSOR
            AISC 01
            LBI 1,0F
            XOR
            JMP SKIP
OK:         COMP
SKIP:       LBI 1,0E
            X 0
            RET
CURBAC:     LDD 1,0F        ;MOVES CURSOR BACK ONE
            AISC 01         ;CHARACTER.  DOES NOT
            JMP GOOD        ;DESTROY DATA AS IT IS MOVED
            LBI 1,0E        ;IF MOVED BEYOND THE
            CLRA            ;END OF THE DISPLAY IT
            AISC 01         ;WRAPS AROUND TO THE OTHER
            XOR             ;END
            X 0
            JMP START
GOOD:       XAD 1,0F
            JMP START


SPECIAL:    LDD 1,0C        ;CONTROL CHAR. HAS BEEN
            AISC 03         ;DETECTED
            JMP NOTRET
            JMP RESET       ;RETURN CLEARS DISPLAY,STARTS
                            ;PROGRAM OVER
NOTRET:     AISC 01         ;NOT RETURN, CHECK FOR CURSOR
            JMP CFOR        ;FORWARD
            JMP CURBAC      ;BY DEFAULT, CURSOR BACKWARDS

CFOR:       JSR CURFOR
            JMP START


        ;DISPLAY LOOP
```

```
;THIS IS THE DISPLAY LOOP OF THE PROGRAM. UNLESS A NEW CHARACTER
;HAS BEEN ENTERED AND IS BEING STORED, THE PROGRAM IS ALWAYS IN
;THIS DISPLAY LOOP. IT LOOKS UP THE CHARACTER FONT, SHIFTS THE
;CHARACTER DATA OUT THE SERIAL PORT AND CONTROLS THE GRID DRIVER.


START:          LBI 2,15        ;DISPLAY LOOP POINTER
                JSR HERE        ;GOTO DISPLAY LOOP
                LBI 3,03        ;SECOND DISPLAY LOOP POINTER
                JSR HERE        ;GOTO DISPLAY LOOP
                OGI 09          ;LOADS A 1 IN GRID DRIVER
                OGI 0D
                OGI 09

                JMP START

           ;CHECKS FOR NEW CHAR


HERE:           RC
                ININ
                AISC 15
                JMP OLDCHR
                JMP NEW

           ;DISPLAY LOOP FOR OLD CHAR AND
           ;  LOOK UP

OLDCHR:         LD 2            ;LOOKS UP FIRST BYTE OF CHR.FONT
                JSR DATA4       ;   200H+ASCII VALUE
                AISC 06         ;ADDS 06H TO MSB OF ASCII
                JSR DATA2       ;LOOKS UP SECOND BYTE OF CHR FONT
                AISC 0A         ;ADDS 0AH TO MSB OF ASCII
                JSR DATA2       ;LOOKS UP THIRD BYTE OF CHR. FONT
                JSR DATA3       ;LOOKS UP THIRD BYTE OF CHR. FONT
                                ;   AT 300H+ASCII VALUE
                AISC 06         ;ADDS 06H TO MSB OF ASCII VALUE
                OGI 02          ;TURNS ON BLANKING CONTROL
                JSR DATA3       ;LOOKS UP LAST BYTE OF CHR. FONT
           ;CLOCKS A 0 IN GRID DRIVER

                OGI 0A          ;ENABLE,BLANKING CONTROL
                OGI 0E          ;ENABLE,BLANKING CONTROL,CLOCK
                OGI 0A          ;ENABLE,BLANKING CONTROL
                OGI 00          ;A 0 SHIFTED IN

                LD 0
                XDS 2
                JMP HERE
                RET

RIGHT:          LBI 3,15
                CQMA
                JSR SHIFT       ;OUTPUTS A
                X 0             ;NEW DATA
                JSR SHIFT       ;OUTPUTS A
                LEI 01          ;COUNTER MODE
                LDD 3,14        ;1,0 IN A
                XABR            ;A IN BR
                LDD 3,13        ;1,1 IN A
                CAB             ;A IN BD
                LD 2
                RET
```

```
POINTER:        LEI 01          ;COUNTER MODE
                XAS             ;A IN SIO
                XABR            ;BR IN A
                AISC 02         ;ADD 2
                XAD 3,14        ;A IN 1,0
                CBA             ;BD IN A
                XAD 3,13        ;A IN 1,1
                LBI 3,15
                XAS             ;SIO IN A
                LEI 08          ;SERIAL MODE
                JMP RIGHT
        ;SHIFTS OUT SERIAL PORT


SHIFT:          LEI 08          ;THIS ROUTINE SHIFTS THE DATA
                SC              ;FROM THE SI/0 REGISTER OUT
                XAS             ;THE SERIAL PORT WITH EACH
                NOP             ;CLOCK CYCLE
                NOP
                RC
                XAS
                RET


        .=0200
DATA3:          LQID
                JMP RIGHT
DATA4:          LQID
                JMP POINTER

        .=0300
DATA3:          LQID
                JMP RIGHT
```

Section 4 of COPS Software

```
;THE CHARACTER FONTS FOR THE CHARACTERS WITH ASCII VALUES BETWEEN
20H AND 5AH HAVE BEEN STORED IN THIS SECTION OF THE PROGRAM.

        ;DATA FOR FIRST 2 BYTES OF EACH
        ;   CHAR.

        .=0220
    .WORD 001, 001, 001, 021, 021, 0C1, 061, 001
    .WORD 031, 001, 041, 011, 001, 011, 001, 001
    .WORD 071, 001, 041, 081, 011, 0E1, 031, 081
    .WORD 061, 061, 001, 001, 001, 021, 001, 041
    .WORD 071, 031, 081, 071, 081, 0F1, 0F1, 071
    .WORD 0F1, 081, 081, 0F1, 0F1, 0F1, 0F1, 071
    .WORD 0F1, 071, 0F1, 061, 081, 0F1, 0F1, 0F1
    .WORD 0C1, 0C1, 081
```

```
        ;DATA FOR SECOND 2 BYTES OF EACH
        ;    CHAR.


        .=0280
.WORD 000, 000, 0C1, 0F9, 0A4, 095, 02D, 000
.WORD 088, 000, 054, 020, 000, 020, 000, 014
.WORD 01D, 082, 003, 005, 058, 045, 0AC, 001
.WORD 02D, 023, 000, 000, 020, 058, 001, 001
.WORD 00D, 0AE, 0F3, 00D, 0F3, 02F, 02F, 00D
.WORD 02E, 003, 00D, 02E, 00E, 08E, 08E, 00D
.WORD 02F, 00D, 02F, 025, 001, 00C, 008, 00C
.WORD 056, 040, 017


        ;THIRD 2 BYTES OF DATA FOR EACH CHAR.


        .=02C0
.WORD 000, 0E3, 000, 0AC, 0FB, 040, 0A5, 083
.WORD 00A, 002, 0F3, 0F1, 034, 040, 008, 040
.WORD 046, 0F7, 02E, 046, 021, 086, 046, 02E
.WORD 046, 046, 0A0, 0B4, 0A0, 0A0, 015, 022
.WORD 0E6, 042, 04E, 006, 00E, 046, 042, 046
.WORD 040, 0F7, 006, 0A0, 004, 080, 0E0, 006
.WORD 042, 026, 062, 046, 0F3, 004, 008, 034
.WORD 040, 070, 046


        ;FOURTH TWO BYTES OF DATA FOR EACH CHAR.


        .=0320
.WORD 000, 008, 007, 0F7, 0AA, 031, 028, 000
.WORD 008, 02A, 049, 080, 000, 080, 000, 001
.WORD 01D, 018, 09C, 09D, 0F7, 01D, 09C, 084
.WORD 09C, 0AC, 000, 000, 022, 041, 041, 08C
.WORD 0DC, 082, 09C, 01C, 01C, 09C, 084, 09C
.WORD 080, 01C, 0EF, 022, 018, 002, 020, 01C
.WORD 084, 02C, 0A4, 09C, 00C, 018, 028, 010
.WORD 041, 009, 01D


        ;LAST BYTES OF DATA FOR EACH CHAR.


        .=0380
.WORD 000, 000, 000, 082, 084, 064, 0A0, 000
.WORD 000, 083, 044, 001, 000, 001, 000, 004
.WORD 0C7, 020, 026, 0CC, 080, 0C9, 0C8, 00E
.WORD 0C6, 087, 000, 000, 028, 082, 001, 006
.WORD 027, 0E3, 0C6, 044, 0C7, 028, 008, 0C5
.WORD 0EF, 028, 008, 028, 020, 0EF, 0EF, 0C7
.WORD 006, 0A7, 026, 0C4, 008, 0CF, 08F, 0CF
.WORD 06C, 00C, 02C


    .END
```
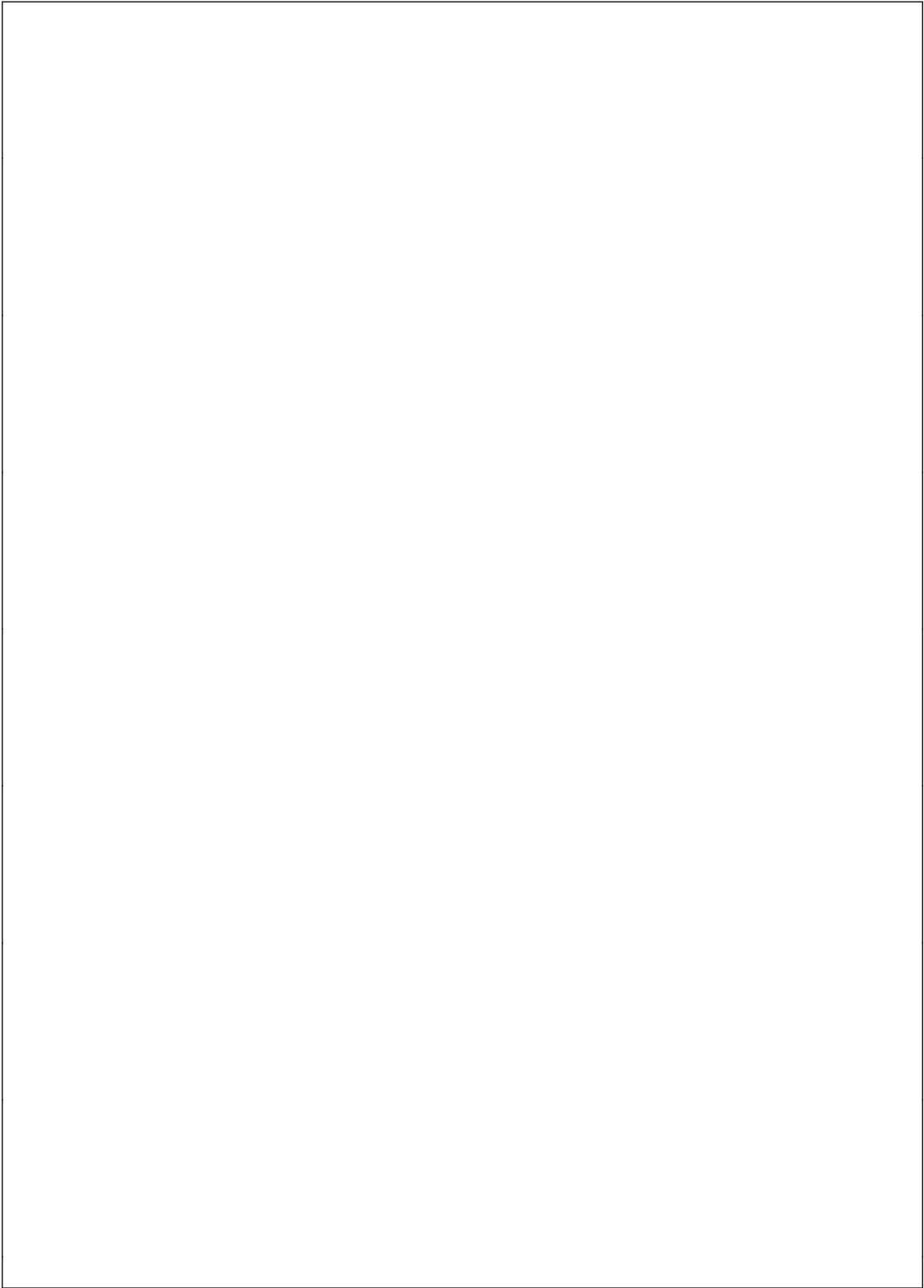
New CMOS Vacuum Fluorescent Drivers Enable Three Chip System to Provide Intelligent Control of Dot Matrix VF Display

AN-440

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.