LCD Direct Drive Using HPC

National Semiconductor
Application Note 786
Santanu Roy
November 1991

INTRODUCTION

Liquid Crystal Displays (LCD) are used in a wide variety of applications. They are extremely popular because of their low power consumption. Manufacturers of Automobiles to Measuring Equipment have taken advantage of these low power displays. Driving LCDs has always been done with dedicated driver chips which not only increase the system cost, but also increase the chip count and board space. This note is developed to demonstrate a low cost solution using the HPC to directly drive LCDs without any driver interface in applications involving LCD display control. A customized 2-way multiplexed LCD (I3420) is being used to illustrate the above capability of HPC microcontrollers in the form of a simple decimal counter.

DRIVING AN LCD

An LCD consists of a backplane and any number of segments which will be used to form the image being displayed. Applying a voltage (nominally 4V–5V) between any segment and the backplane causes the segment to darken. The only catch is that the polarity of the applied voltage has to be periodically reversed, or else a chemical reaction takes place in the LCD which causes deterioration and eventual failure of the liquid crystal. (DC components higher than 100 mV can cause electrochemical reactions in LCDs). To prevent this from happening, the backplane and all the segments are driven with an AC signal, which is derived from a rectangular waveform. To turn a segment *OFF*, it is driven by the same waveform as the backplane. Thus it is always at backplane potential. If a segment is to be *ON*, it is driven with a waveform that is the inverse of the backplane waveform. Thus it has periodically changing polarity between it and the backplane.

MULTIPLEXED LCDs

Today a wide variety of LCDs ranging from static to multiplex rates of 1:64 are available on the market. The *MULTIPLEX* rate of an LCD is determined by the number of backplanes. The higher the multiplex rate the more individual segments can be controlled using only one line e.g., a static LCD has only one backplane and hence only one segment can be controlled using one line. A two way multiplexed LCD has two backplanes and two segments can be controlled with one line. In general if the multiplex ratio of the LCD is N and the number of available outputs is M, the number of segments that can be driven is:

$$S = (M - N)*N$$

i.e., N lines out of M outputs will be used to drive N backplanes, the rest (M − N) outputs are available for segment control. Each line can control N segments, so (M − N) lines can drive (M − N)*N segments. So the maximum number of segments in a 2-way MUX LCD that can be driven with an HPC (if all outputs—16 PortA, 16 PortB, and 4 PortP are used) is:

$$S = (36 - 2)*2 = 68$$

The number of backplanes in the LCD also determines the number of levels to be generated for their control signals, e.g., three different voltage levels V, 1/2V, and 0 are to be generated for a 1:2 LCD device (V = operating voltage of the LCD). A *Refresh Cycle* of LCDs (also known as *"Scan Frequency"*) is the time period during which all backplanes and segments have to be updated. Typically this is between 39 Hz–208 Hz. During each half of the refresh cycle (*Frame Time*), the polarities of the voltages driving the backplanes and the segments are reversed because of the reason stated above. The current consumption of typical LCDs is in the range of 3 μA–4 μA (at V = 4.5, refresh rate 60 Hz) per square centimeter of activated area. Thus the backplane and segment terminals can be treated as Hi-Z loads. At high refresh rates the current consumption of LCDs increases dramatically, a reason why many LCD manufacturers recommend not to exceed a refresh rate of 60 Hz.

LCD CONTROL AND HPC

Figure 1 shows the schematic of the system. With the HPC, each I/O pin can be set individually to TRI-STATE®, "HI" or "LO". Here, in this application, B4 and B5 on the HPC's PortB are selected for backplane control of a 1:2 multiplexed customized LCD–I3420. The three different voltage levels viz. V, V/2, and 0 required for backplane control are achieved through an external voltage divider circuit. The procedure is to set B4 and B5 to "LO" for 0, Hi-Z (configuring them as inputs) for 0.5V, and "HI" for V at the backplane electrodes. For segment control: 8 PortA lines (A0–A7), 4 PortP lines (P0–P3) and 3 PortB lines (B0–B2) are used. All are used as outputs to drive individual segments of the LCD. The HPC in this application is used in single-chip mode to maximize the I/O pin count for LCD control.

TIMING CONSIDERATIONS

Figure 2 shows the backplane and segment waveforms of a typical 1:2 multiplexed LCD. One Refresh Cycle T_{scan} is subdivided into four equally spaced time slots ta, tb, tc and td during which the backplane and segment terminals have to be updated in order to switch a specific segment "ON" or "OFF". The voltage waveform during BP− is the mirror image of the waveform during BP+ which satisfies polarity reversal every T_{frame}. Considering a refresh frequency of 50 Hz i.e., T_{scan} = 20 ms: ta, tb, tc and td are each equal to 5 ms. The timer T2 is used to mark off one time phase (¼ of T_{scan}) of the driving voltage waveform. The timer and autoreload value to get 5 ms time-out is 4999 (decimal) at an operating frequency of 16.0 MHz.

SEGMENT CONTROL

In *Figure 2a*, BP1 and BP2 are the typical backplane waveform of a 2-way multiplexed LCD. During BP+ time, backplane outputs are *ON* for driving voltage level V and *OFF* for the level ½V. Again for BP− frame time, backplane outputs are *ON* for "0" and *OFF* for "½V". Voltage at a particular LCD segment is the resultant of the backplane output and voltage at the line driving that segment. *Figure 2(b)* shows the waveform at an LCD segment. *Figure 2(c)* and *2(d)* are the resultant waveforms with respect to BP1 and BP2 obtained by subtracting the segment waveform in *Figure 2(b)* from the backplane waveforms BP1 and BP2 respectively.

RRD-B30M75/Printed in U. S. A.

Figure 3 shows the four different waveforms which must be generated at the segments to meet all possible combinations *ON* and *OFF* sequence viz. *OFF-OFF*, *ON-ON*, *ON-OFF,* and *OFF-ON*. A segment is *ON* if the resultant voltage across it periodically oscillates between +V and −V and is *OFF* if the swing is between +V/2 and −V/2. The result of the combination is showed in form of white and black circles, representing *OFF* and *ON* segments respectively e.g., a waveform pattern ''1'' will always turn a segment *OFF* with respect to both the backplanes. However, the waveform ''2'' will keep it *ON* with respect to BP1 and BP2. *Figurea 4a* and *Figure 4b* show the resultant voltage waveforms at an LCD segment for the above possible combinations and the status of the segment during display operation. *Figures 5* and *6* shows the internal segment and backplane connections for a typical 2-way LCD. *Figure 7* gives the details of the LCD used in this application.

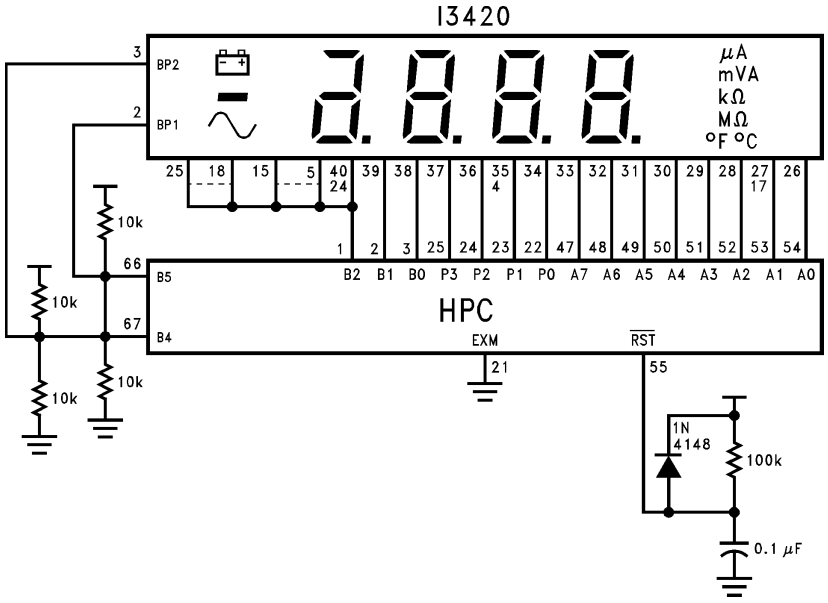
LCD DRIVE SUBROUTINE

The software for the LCD drive is provided at the end of this application note. The drive subroutine **DISPL** converts a 16-bit binary value to a 20-bit BCD value for easier display data fetch. This subroutine itself is comprised of a main routine for backplane refresh and seven subroutines (**SEGTA, SEGTB, SEGTC, SEGTD, SEGOUT, TMPND, and DISPD**). The subroutines **SEGTA** through **SEGTD** are used to fetch LCD segment data from a lookup table in ROM for time phases ta, tb, tc, and td respectively. In the table, the subroutine **SEGOUT** writes these data for each time phase to the respective ports of the HPC connected to the LCD device. For a refresh cycle of 50 Hz (20 ms), each time phase (¼ of T_{scan}) is equal to 5.0 ms. This time base is generated by the HPC timer T2 with the associated autoreload register R2. The polling routine **TMPND** checks for timer underflow flag at the end of each time phase. If the flag is set, it is reset and the program returns to the calling routine. This way a 5 ms time delay is created before the segment and backplane data for the next time phase is updated. The **DISPD** subroutine switches the LCD *OFF* by driving the segment and backplane ports to logic ''LO''. In this application, the display is initialized with ''399.9'' (which uses all LCD segments) for a BCD down counter. Each count is displayed for a fixed period of time (here a present time of 100 ms is chosen) which is user programmable. The special segments e.g., ''m'', ''A'', ''V'' . . . etc. which are not used are all connected together to a common port pin (B2) of the HPC and kept turned *OFF* throughout the display. It is mandatory to drive any unused segment lines to the *OFF* state rather than leaving them open or grounded which might result in ghost images.

Note: Selecting the resistors for the voltage divider circuits on B4 and B5 will depend on the type of LCD used.

TYPICAL APPLICATIONS

- Automotive test and control systems
- Weighing scales
- Control Panel
- Microwave
- Clocks and watches etc.



TL/DD/11250–1

FIGURE 1

LCD Waveforms

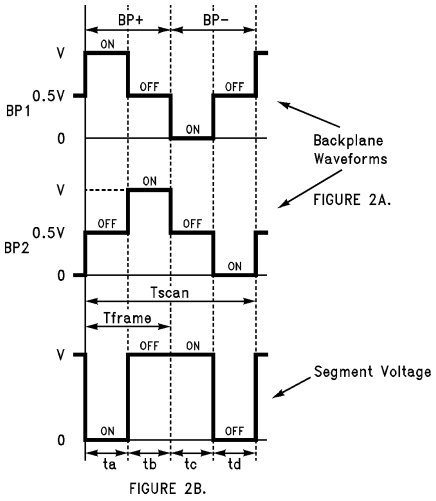


FIGURE 2B.

TL/DD/11250–2

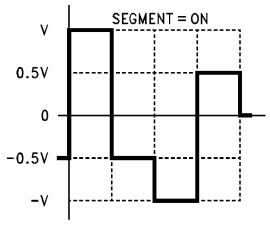


FIGURE 2C.

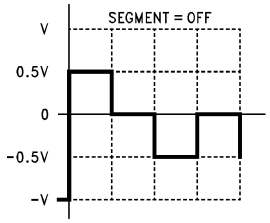
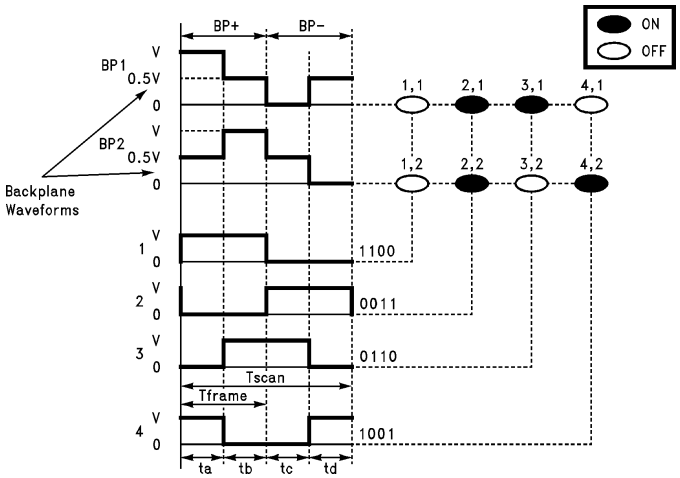


FIGURE 2D.

TL/DD/11250–3

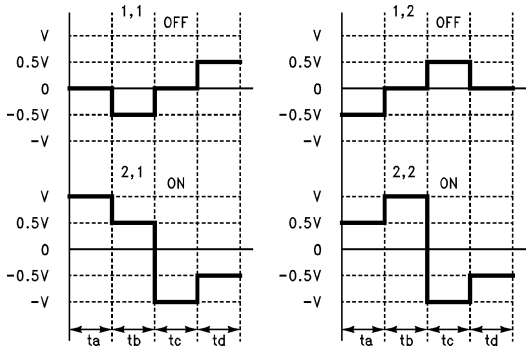
Segment and Backplane Waveforms



TL/DD/11250–4

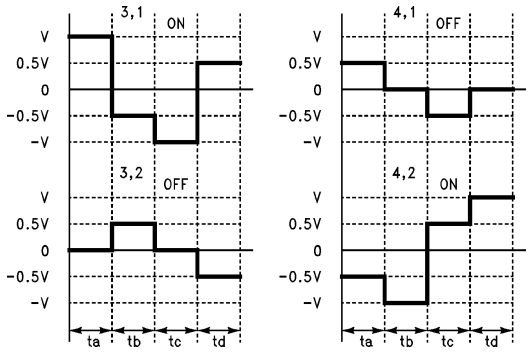
FIGURE 3. Segment Waveforms

Resultant Waveforms at Segments



TL/DD/11250–5

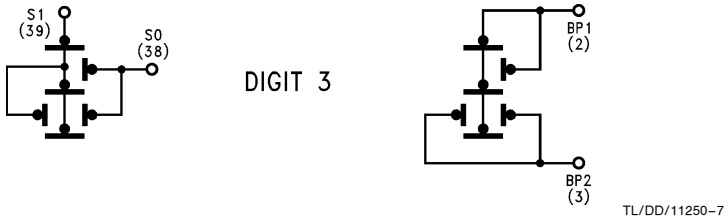
FIGURE 4a



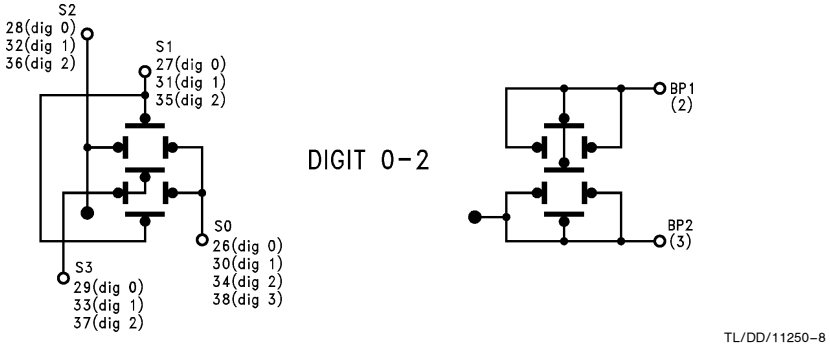
TL/DD/11250–6

FIGURE 4b

Segment and Backplane Distribution



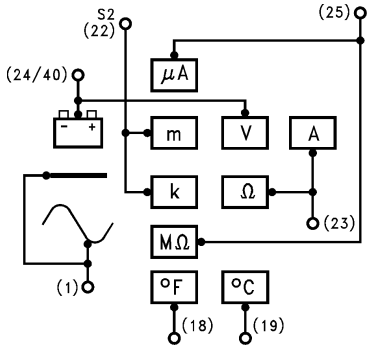
TL/DD/11250–7



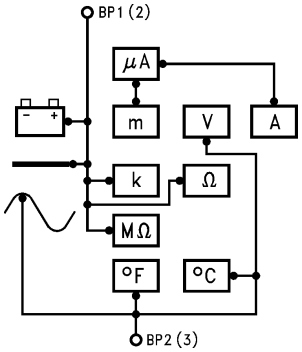
TL/DD/11250–8

FIGURE 5

Special Segments
Segment and Backplane Distribution



TL/DD/11250–9



TL/DD/11250–10

FIGURE 6

```
;MEMAP.INC

;This is the memory map of different RAM areas used in the
;LCD program.


;*********************** RAM DEFINITIONS ******************

BCDLO   = 02:b              ;Measured period in BCD (lo byte)
BCDHI   = 03:b              ;High Byte

MWBUF0 = 05:b               ;A-port data (7-segment)
MWBUF1 = 06:w               ;P-port data
MWBUF2 = 08:b               ;B-port data

OFF1    = 0a:b              ;offset reg. for 7-seg code table
OFF2    = 0b:b              ;
OFF3    = 0c:b              ;

EVAL    = 0e:w              ;end value lo-byte (period)
SVAL    = 010:w             ;hi-byte

COUNT   = 020:b             ;counter #1
COUNT2 = 021:b              ;counter #2
BCNT    = 022:w
;********************* BIT MAP *****************************

BP1     = 05                ;Backplane 1
BP2     = 04                ;Backplane 2
```

TL/DD/11250–12

```
;File Name:CNTR.ASM

;Function: Counter displayed on a 2-way muxed LCD display
;directly driven by the HPC16083.


.incld reg16083.inc             ;HPC register def. file
                                ;Chip - HPC16083

.incld memap.inc

.extrn DISPL,DISPD,COPY


.sect cntr,rom16

BEGIN:
          ld   sp,#01c0         ;set the stack pointer
BINIT:
          jsr  DISPD            ;define port config,
                                ;switch diplay OFF
OK1:
          ld   BCNT,#0f9f       ;set counter to 3999 decimal
BLOOP:
          ld   b,#BCNT          ;copy the decimal value
          ld   x,#EVAL          ;to the location which
          jsr  COPY             ;undergoes conversion
          ld   COUNT2,#01       ;display time=100 ms
          jsr  DISPL            ;Display 399.9 first

BLOOP2:
          decsz BCNT            ;display till
          jp   BLOOP            ;counter=0
          ld   b,#BCNT          ;display "0" also
          ld   x,#EVAL          ;and then restart
          jsr  COPY             ;the session
          ld   COUNT2,#01       ;
          jsr  DISPL            ;go back and start
          jp   BINIT

.endsect
.end BEGIN
```

TL/DD/11250–13

```
;Title: DISPL.ASM

;COUNT2 = Contains display time in seconds e.g if "1" ->
;display time is 1 second.

;SEGTA: Gets LCD segment data for time phase Ta
;SEGTB: .... Time phase Tb
;SEGTC: .... Time phase Tc
;SEGTD: .... Time phase Td

;OFF1:  .... Offset register for DIGIT 0+1
;OFF2:  .... Offset register for DIGIT 2
;OFF3:  .... Offset register for DIGIT 3

.incld reg16083.inc
.incld memap.inc

.extrn TMPND,TBL,BINBCD
.public DISPL,DISPD

.sect drive,rom16


SEGTA:
          ld    OFF1.w,#0          ;clear OFF1 and OFF2
          ld    a,#042             ;point to DIG3 data

$APORT:
          st    a,OFF3             ;put it in OFF3 reg.
          ld    x,#BCDLO           ;point to BCDLO byte
          ld    b,#MWBUF0          ;point to MWBUF0
          ld    a,[x].b            ;get the bcd lo byte
          and   a,#0f              ;get low nibble
          add   a,OFF1             ;add to the offset reg1
          ld    a,TBL[a].b         ;get the 7-seg code
          st    a,[b].b            ;save the data in MWBUF0
          ld    a,[x+].b           ;x reg points to BCDLO+1
          and   a,#0f0             ;upper nibble of lower
          swap  a                  ;byte of BCDLO
          and   a,#00f             ;clear other bits
          add   a,OFF2             ;add to the OFF2 reg
          ld    a,TBL[a].b         ;
          swap  a                  ;position upper nibble
          and   a,#0f0             ;clear all other bits
          or    MWBUF0,a           ;data (+ dec. point)

$PPORT:
          ld    b,#MWBUF1          ;point to MWBUF1
          ld    a,[x].b            ;get BCDLO+1 data
          and   a,#0f              ;get the lower nibble
          add   a,OFF2             ;add the reqd. offset
          ld    a,TBL[a].b         ;get the 7-seg data
          st    a,[b].w            ;
          ifbit 1,[b].b            ;rearrange as PORTP
```

TL/DD/11250–14

```
            sbit  4,MWBUF1           ;data bits are 0,4,7,15
            ifbit 2,[b].b            ;locations in portP
            sbit  0,MWBUF1+1.b       ;
            ifbit 3,[b].b            ;
            sbit  4,MWBUF1+1.b       ;

$BPORT:
            ld    b,#MWBUF2          ;point to MWBUF2
            ld    a,[x].b            ;get digit3 data
            and   a,#0f0             ;get the higher nibble
            swap  a                  ;
            and   a,#0f              ;position it right
            add   a,OFF3             ;add the reqd. offset
            ld    a,TBL[a].b         ;from the table
            or    a,#0f8             ;sbit 3...7 and save
            st    a,[b].b            ;save it in MWBUF2
            ret

SEGTB:
            ld   OFF1,#016           ;with dec. pt
            ld   OFF2,#00b           ;without dec. pt
            ld   a,#046              ;
            jp   $APORT              ;

SEGTC:
            ld   OFF1,#021           ;
            ld   OFF2,#021           ;
            ld   a,#04a              ;
            jp   $APORT              ;

SEGTD:
            ld   OFF1,#037           ;
            ld   OFF2,#02c           ;
            ld   a,#04e              ;
            jp   $APORT              ;


DISPL:
            jsr  BINBCD              ;convert bin to BCD
            ld   COUNT,#05           ;50*20 ms = 1 sec
                                     ;10*1 = 10 sec display
            ld   irpd,#0             ;clear all pending bits
            ld   tmmode,#04440       ;timer ckt. initialize
            ld   pwmode,#04444       ;stop all timers
            ld   tmmode,#0ccc8       ;and acknowledge all
            ld   pwmode,#0cccc       ;interrupts
            ld   divby,#02222        ;select T2 clock=CKI/16
            ld   t2reg,#01387        ;LCD refresh rate of
            ld   r2reg,#01387        ;50 Hz (20 ms) -> 5ms
                                     ;per time slot (5000
                                     ;counts @ 16.0 Mhz)
            rbit 2,tmmodeh           ;start timer T2

DISP1:
```

TL/DD/11250–15

```
            jsr  SEGTA              ;get 7 seg. dat for
                                    ;refresh time phase Ta
            jsr  TMPND              ;test pending T2

TP0:
            sbit BP1,portbl         ;backplane refresh Ta
            rbit BP2,dirbl          ;make it i/p (Hi-z)
            sbit BP1,dirbl          ;
            rbit BP2,portbl         ;BP1=1, BP2=.5
            jsr  SEGOUT             ;
            jsr  SEGTB              ;time phase Tb
            jsr  TMPND              ;

TP1:
            sbit BP2,portbl         ;BP2 data = 1
            rbit BP1,dirbl          ;make BP1 i/p
            sbit BP2,dirbl          ;send BP2=1
            rbit BP1,portbl         ;Hi-z
            jsr  SEGOUT             ;BP1=.5, BP2=1
            jsr  SEGTC              ;
            jsr  TMPND              ;

TP2:
            rbit BP1,portbl         ;BP1 data=0
            rbit BP2,dirbl          ;BP2 i/p
            sbit BP1,dirbl          ;o/p "0" on BP1
            rbit BP2,portbl         ;BP2 = 0.5
            jsr  SEGOUT             ;
            jsr  SEGTD              ;
            jsr  TMPND              ;

TP3:
            rbit BP2,portbl         ;BP1 data=0
            rbit BP1,dirbl          ;BP2 data=0
            sbit BP2,dirbl          ;make BP1 Hi-z (0.5)
            rbit BP1,portbl         ;BP1=.5, BP2=0
            jsr  SEGOUT             ;
            decsz COUNT             ;do the loop N times
            jp   DISP1              ;
            ld   COUNT,#5           ;
            decsz COUNT2            ;COUNT2 = X*N = set time
            jp   DISP1              ;
            ret                     ;

DISPD:
            ld   portal,#00         ;switch display OFF
            ld   diral,#0ff         ;as o/p
            ld   portbl,#0          ;
            ld   dirbl,#037         ;B0-B2,B5,B4 = outputs
            ld   portp,#0           ;
            ret

SEGOUT:
            ld   porta,MWBUF0       ;portA data (DIG 4+5)
```

TL/DD/11250–16

```
            ld    portp,MWBUF1          ;portP data (16-bit reg)
            ld    b,#MWBUF2             ;
            ld    x,#portbl             ;read portb low byte
            ld    a,[x].b               ;and it with MWBUF2
            and   a,[b].b               ;save original MWBUF2 in
            ld    k,MWBUF2              ;K register
            st    a,[b].b               ;store MWBUF2&PORTBL in
            ld    a,k                   ;MWBUF2
            and   a,#007                ;get orig. MWBUF2 and
            or    a,[b].b               ;extract B0-B2, OR it
            st    a,portbl              ;with new MWBUF2 and
            ret                         ;send it

.endsect
```

TL/DD/11250–17

```
;Title :   BINBCD.ASM

;Function: This program takes a 16-bit binary number and
;converts into a 20-bit BCD number.
;
;************************ RAM MAP ************************

;INPUT DATA  -> BINLO+1  BINLO
;
;BCD OUTPUT  -> BCDLO+2   BCDLO+1  BCDLO

.incld memap.inc

BINLO = EVAL


.public BINBCD

.sect code,rom8

BINBCD:
          ld   COUNT,#16              ;Number of left shifts
          ld   bk,#BCDLO,#BCDLO+2     ;

$CBCD:
          clr  a                      ;clear BCD ram space
          xs   a,[b+].b               ;
          jp   $CBCD                  ;

$LSH:                                 ;left shift binary
                                      ;routine
          ld   bk,#BINLO,#BINLO+1     ;
          rc                          ;reset carry
$LSHFT:
          ld   a,[b].b                ;start shifting
          adc  a,[b].b                ;if MSB=1, set C
          xs   a,[b+].b               ;do for all 4 nibbles
          jp   $LSHFT                 ;of the Binary data

          ld   bk,#BCDLO,#BCDLO+2     ;

$BCDADD:
          ld   a,[b].b                ;get the BCD data
          dadc  a,[b].b               ;decimal add with carry
          xs   a,[b+].b               ;put it back
          jp   $BCDADD                ;loop for all 3 bytes

          decsz COUNT                 ;is shift =16?

COUNTER:
          jp   $LSH                   ;no - go back
          ret

.endsect
```

TL/DD/11250–18

```
;Lookup table for customized 2-way MUX LCD I3420
;
.incld reg16083.inc


.public TBL,TMPND,COPY

.sect table,rom8

TBL:

;Timephase Ta ---- 7 segment data

.byte     08                    ;'0' and '.0'
.byte     0e                    ;'1' and '.1'
.byte     04                    ;'2' and '.2'
.byte     04                    ;'3' and '.3'
.byte     02                    ;'4' and '.4'
.byte     01                    ;'5' and '.5'
.byte     01                    ;'6' and '.6'
.byte     0c                    ;'7' and '.7'
.byte     00                    ;'8' and '.8'
.byte     00                    ;'9' and '.9'
.byte     0f                    ;' ' and '.'


;Timephase Tb ---- 7 segment data

.byte     04                    ;'0'
.byte     0e                    ;'1'
.byte     05                    ;'2'
.byte     0c                    ;'3'
.byte     0e                    ;'4'
.byte     0c                    ;'5'
.byte     04                    ;'6'
.byte     0e                    ;'7'
.byte     04                    ;'8'
.byte     0c                    ;'9'
.byte     0f                    ;' '
```

TL/DD/11250–19

```
.byte     00                    ;'.0'
.byte     0a                    ;'.1'
.byte     01                    ;'.2'
.byte     08                    ;'.3'
.byte     0a                    ;'.4'
.byte     08                    ;'.5'
.byte     00                    ;'.6'
.byte     0a                    ;'.7'
.byte     00                    ;'.8'
.byte     08                    ;'.9'
.byte     0b                    ;'.'


;Timephase Tc ---- 7 segment data

.byte     07                    ;'0' and '.0'
.byte     01                    ;'1' and '.1'
.byte     0b                    ;'2' and '.2'
.byte     0b                    ;'3' and '.3'
.byte     0d                    ;'4' and '.4'
.byte     0e                    ;'5' and '.5'
.byte     0e                    ;'6' and '.6'
.byte     03                    ;'7' and '.7'
.byte     0f                    ;'8' and '.8'
.byte     0f                    ;'9' and '.9'
.byte     00                    ;' ' and '.'
```

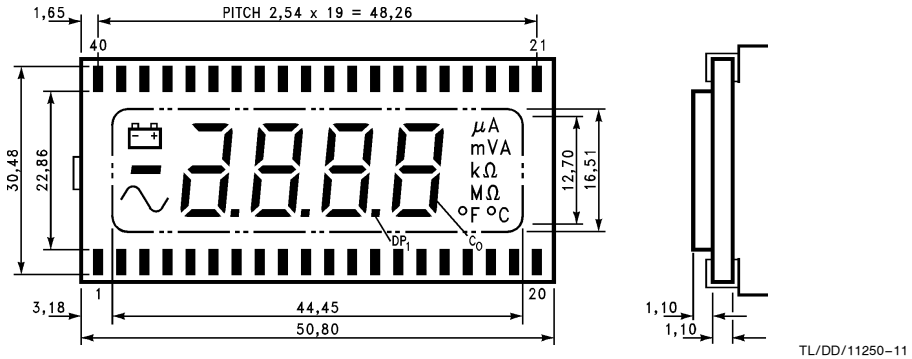
TL/DD/11250–20

```
;Timephase Td ---- 7 segment data

.byte     0b                    ;'0'
.byte     01                    ;'1'
.byte     0a                    ;'2'
.byte     03                    ;'3'
.byte     01                    ;'4'
.byte     03                    ;'5'
.byte     0b                    ;'6'
.byte     01                    ;'7'
.byte     0b                    ;'8'
.byte     03                    ;'9'
.byte     00                    ;' '

.byte     0f                    ;'.0'
.byte     05                    ;'.1'
.byte     0e                    ;'.2'
.byte     07                    ;'.3'
.byte     05                    ;'.4'
.byte     07                    ;'.5'
.byte     0f                    ;'.6'
.byte     05                    ;'.7'
.byte     0f                    ;'.8'
.byte     07                    ;'.9'
.byte     04                    ;'.'



;

;Digit '3' codes

;Time phase Ta
.byte        07                      ;''
.byte        06                      ;'1'
.byte        04                      ;'2'
.byte        04                      ;'3'

;Timephase Tb

.byte        07                      ;''
.byte        06                      ;'1'
.byte        05                      ;'2'
.byte        06                      ;'3'

;Timephase Tc

.byte        00                      ;''
.byte        01                      ;'1'
.byte        03                      ;'2'
.byte        03                      ;'3'

;Timephase Td
```

TL/DD/11250–21

```
.byte       00                  ;''
.byte       01                  ;'1'
.byte       02                  ;'2'
.byte       01                  ;'3'

TMPND:
            ld      b,#tmmodeh
$LOOP:
            ifbit   1,[b].b
            jp      $END
            jp      $LOOP
$END:
            sbit    3,[b].b
            ret
COPY:
            ld      a,[b].w
            x       a,[x].w
            ret
.endsect
```

TL/DD/11250–22



Segment	/	Backplane	Pin Assignment (Proposal)	
BP1		BP2	2 3	
Minus		∿	1	
K		m	22	
Ω		A	23	
nΩ		μA	25	The pairing and
±		V	40/24	the annunciators
B_0		C_0	26	may be slightly
A_0		D_0	27/17	rearranged by the
G_0		E_0	29	LCD manufacturer.
F_0		DP_1	28	
B_1		C_1	30	
A_1		D_1	31/16	
G_1		E_1	33	
F_1		DP_2	32	
B_2		C_2	34	
A_2		D_2	35/4	
G_2		E_2	37	
F_2		DP_3	36	
B_3		C_3	38	$ADG_3 = A_3, D_3, G_3$
ADG_3		E_3	39	(One segment)
		°C	19	
		°F	18	

FIGURE 7

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.

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor Japan Ltd.
Sumitomo Chemical
Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductores Do Brazil Ltda.
Rue Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

National Semiconductor (Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998