2-Way Multiplexed LCD Drive and Low Cost A/D Converter Using V/F Techniques with COP8 Microcontrollers

National Semiconductor
Application Note 673
Volker Soffel
April 1990

ABSTRACT

This application note is intended to show a general solution for implementing a low cost A/D and a 2-way multiplexed LCD drive using National Semiconductor's COP840C 8-bit microcontroller. The implementation is demonstrated by means of a digital personal scale. Details and function of the weight sensor itself are not covered in this note. Also the algorithms used to calculate the weight from the measured frequency are not included, as they are too specific and depend on the kind of sensor used.

Typical Applications

- Weighing scales
- Sensors with voltage output
- Capacitive or resistive sensors
- All kinds of measuring equipment
- Automotive test and control systems

Features

- 2-way multiplexed LCD drive capability up to 30 segments (4 digit and 2 dot points)
- Precision frequency measurement
- Low current consumption
- Current saving HALT mode
- Additional computing power for application specific tasks

INTRODUCTION

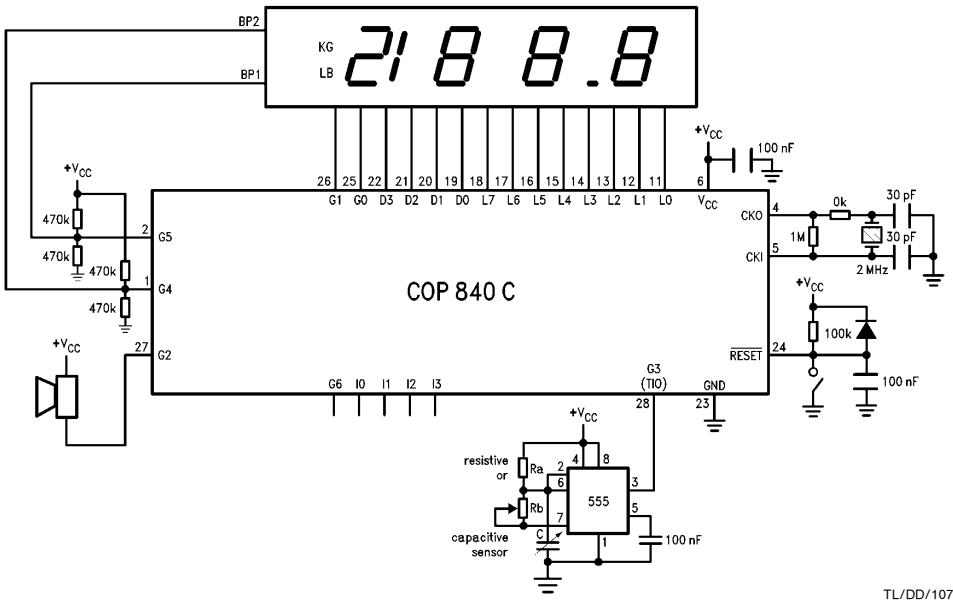
Today's most popular digital scales all have the following characteristics:

They are battery powered and use a LCD to display the weight. Instead of using a discrete A/D-converter, in many cases a V/F converter is used, which converts an output voltage change of the weight sensor to a frequency change. This frequency is measured by a microcontroller and is used to calculate the weight. The advantages of a V/F over an A/D converter are multifold. Only one line from the V/F to the microcontroller is needed, whereas a parallel A/D needs at least 8 lines or even more (National also offers A/Ds with serial output). A V/F can be constructed very simply using National Semiconductor's low cost, precision voltage to frequency converters LM331 or LM331A. Other possibilities are using Op-amps or a 555-timer in astable mode.

V/F-CONVERSION

Hardware

The basic configuration of the scale described in this application note is shown in *Figure 1*.



TL/DD/10788–1

FIGURE 1. System Diagram

TRI-STATE® is a registered trademark of National Semiconductor Corporation.

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

A capacitive or resistive sensor's weight related capacitance or resistance change is transformed by a 555 timer (in astable mode) to a change of frequency. The output frequency f is determined by the formula:

$$f = 1.44/((Ra + 2Rb) *C)$$

The output high time is given by:

$$t1 = 0.693* (Ra + Rb) *C$$

The output low time is given by:

$$t2 = 0.693* Rb* C$$

This frequency is measured using the COP800 16-bit timer in the ''input capture'' mode. After calculation, the weight is displayed on a 2-way multiplexed LCD. Using this configuration a complete scale can be built using only two ICs and a few external passive components.

For more information on V/F converters generally used with voltage output sensors, refer to the literature listed in the reference section.

Frequency Measurement

The COP 16-bit timer is ideally suited for precise frequency measurements with minimum software overhead. This timer has three programmable operating modes, of which the ''input capture'' mode is used for the frequency measurement. Allocated with the timer is a 16-bit ''autoload/capture register''. The G3-I/O-pin serves as the timer capture input (TIO). In the ''input capture'' mode the timer is decremented with the instruction cycle frequency (tc). Each positive going edge at TIO (also neg. edge programmable) causes the timer value to be copied automatically to the autoload/capture register without stopping the timer or destroying its contents. The ''timer pending'' flag (TPND) in the PSW-register is set to indicate a capture has occurred, and if the timer-interrupt is enabled, an interrupt is generated. The frequency measurement routine listed below executes the following operations (refer to the RAM/register definition file listed at the beginning for symbolic names used in the routines):

The timer is preset with FFFF Hex and is started by setting the TRUN bit, after which the software checks the TPND-flag in a loop (timer interrupt is disabled). When the TPND flag is set the first time, the contents of the capture register is saved in RAM locations STALO and STAHI (start value). The TPND pending flag now must be reset by the software. Then, another 255 positive going edges are counted (equal to 255 pulses) before the capture register is saved in RAM locations ENDLO, ENDHI (end value). The shortest time period that can be measured depends on the number of instruction cycles needed to save the capture register, because with the next positive going edge on TIO the contents of the capture register is overwritten (worst case is 18 instruction cycles, which equals a max. frequency of 55.5 kHz at tc = 1 μs).

The end-value is subtracted from the start-value and the result is restored in RAM locations STALO, STAHI. This value can then be used to calculate the time period of the frequency applied to TIO (G3) by multiplying it with the tc-time and dividing the result by the number of pulses measured (N = 255).

$$T = (\text{startvalue} - \text{endvalue}) *tc/N$$

```
;THE FOLLOWING "INCLUDE FILE" IS USED
;AS PART OF THE DEFINITION- AND INITIALIZATION PHASE
;IN COP800 PROGRAMS.
;REGISTER NAMES,CONTROL BITS ETC ARE NAMED IN THE
;SAME WAY IN THE COP800 DATA-SHEETS.


;        --- COP800  MEMORY MAPPED ---


; ****************************************************
; * PORT  -, CONFIGURATION  -  AND CONTROL REGISTERS *
; ****************************************************


      PORTLD     =     0D0          ; L-PORT DATA REGISTER
      PORTLC     =     0D1          ; L-PORT CONFIGURATION
```

TL/DD/10788–2

```
      PORTLP    =    0D2         ; L-PORT INPUT REGISTER

      PORTGD    =    0D4         ; G-PORT DATA REGISTER
      PORTGC    =    0D5         ; G-PORT CONFIGURATION
      PORTGP    =    0D6         ; G-PORT INPUT REGISTER

      PORTD     =    0DC         ; D-PORT (OUTPUT)
      PORTI     =    0D7         ; I-PORT (INPUT)


      SIOR      =    0E9         ; MWIRE SHIFT REGISTER
      TMRLO     =    0EA         ; TIMER LOW-BYTE
      TMRHI     =    0EB         ; TIMER HIGH-BYTE
      TAULO     =    0EC         ; T.-AUTO REG.LOW BYTE
      TAUHI     =    0ED         ; T.-AUTO REG.HIGH BYTE

      CNTRL     =    0EE         ; CONTROL REGISTER
      PSW       =    0EF         ; PSW-REGISTER
                     .FORM
;               **********************
;               *  CONSTANT DECLARE  *
;               **********************


;          ---  CONTROL REGISTER BITS ---


      S0         =     00      ; MICROWIRE CLOCK DIVIDE BY
                               ;      --- BIT 0 ---
      S1         =     01      ; MICROWIRE CLOCK DIVIDE BY
                               ;      --- BIT 1 ---
      IEDG       =     02      ; EXTERNAL INTERRUPT EDGE
                               ; POLARITY SELECT (0=RISING
                               ; EDGE,1=FALLING EDGE)
      MSEL       =     03      ; ENABLE MICROWIRE FUNCTION
                               ;     --- SO AND SK ---
      TRUN       =     04      ; START/STOP THE TIM/COUNT.
                               ;     (1=RUN;0=STOP)
      TEDG       =     05      ; TIMER INPUT EDGE POL.SEL.
                               ; (0=RIS. EDGE;1=FAL. EDGE)
      CSEL       =     06      ; SELECTS THE CAPTURE MODE
                               ;
      TSEL       =     07      ; SELECTS THE TIMER MODE


;           ---  P S W    REGISTER ---


      GIE        =     00      ; GLOBAL INTERRUPT ENABLE
```

TL/DD/10788–3

```
     ENI        =     01     ; EXTERNAL INTERRUPT ENABLE
     BUSY       =     02     ; MICROWIRE BUSY SHIFTING
     IPND       =     03     ; EXTERNAL INTERR. PENDING
     ENTI       =     04     ; TIMER INTERRUPT ENABLE
     TPND       =     05     ; TIMER INTERRUPT PENDING
     C          =     06     ; CARRY FLAG
     HC         =     07     ; HALF CARRY FLAG

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


        BCDLO    = 000  ;CALCULATED WEIGHT IN BCD
                        ;LOW BYTE
        BCDHI    = 001  ;CALCULATED WEIGHT IN BCD
                        ;HIGH BYTE
        MWBUF0   = 003  ;7SEGMENT DATA FOR LCD DISPL
                        ;L-PORT
        MWBUF1   = 004  ;D-PORT
        MWBUF2   = 005  ;G-PORT
        OFF1     = 006  ;OFFSET REGISTERS FOR
        OFF2     = 007  ;7 SEGMENT CODE TABLE
        OFF3     = 008  ;

        STALO    = 009  ;START VALUE,LOW BYTE
        STAHI    = 00A  ;START VALUE,HIGH BYTE
        ENDLO    = 00B  ;END VALUE LOW BYTE
        ENDHI    = 00C  ;END VALUE HIGH BYTE

        DIV0     = 00D  ;DIVISOR FOR DINBI248 ROUTINE

       ;022.. 02F RESERVED FOR STACK WITH COP820
       ;062..06F  RESERVED FOR STACK WITH COP840


;****           REGISTER DEFINITIONS      ****

        COUNT  = 0F0
        COUNT2 = 0F1
        COUNT3 = 0F2
        FLAG   = 0FF      ;FLAG REGISTER


;****           BIT DEFINITIONS FLAG REGISTER  ****

        POUND = 04      ;POUND=1:DISPLAY POUND SEGMENT
                        ;POUND=0:DISPLAY kg SEGMENT

;*****       G-PORT BIT DEFINITIONS         *****

        BP1   = 05      ;BACKPLANE 1
```

TL/DD/10788–4

```
        BP2  = 04       ;BACKPLANE 2

;TIME OF 255 PULSES, USING TIMER INPUT CAPTURE MODE

FMEAS:
                        ;PERIOD TIME=
                        ;(START-ENDVALUE)*tc/255
                        ;DIFFERENCE START-ENDVALUE
                        ;IS STORED IN ENDLO,ENDHI
        LD      COUNT,#000  ;LOAD PULSE COUNTER (255 PULSES)
        LD      X,#TAULO    ;POINT TO AUTO REG. LOW B.
        LD      B,#TMRLO    ;PRESET TIMER
        LD      [B+],#0FF   ;REG. WITH FFFFh
        LD      [B],#0FF
        LD      B,#CNTRL
        LD      [B+],#0D0   ;CNTRL-REG.: TIMER CAPTURE
                            ;MODE,TIO POS. TRIGGERED,
                            ;START TIMER

        RBIT    #TPND,[B]   ;RESET TIMER PENDING FLAG
L1:     IFBIT   #TPND,[B]
        JP      SSTORE
        JP      L1
SSTORE:                     ;STORE START VALUE
        RBIT    #TPND,[B]
        LD      A,[X+]      ;LOAD TIMER CAPTURE REG.
                            ;LOW BYTE
        X       A,STALO     ;STORE IN RAM
        LD      A,[X-]      ;LOAD HIGH BYTE CAPTURE,
                            ;POINT TO LOW BYTE CAPTURE
        X       A,STAHI     ;STORE IN RAM
        LD      B,#PSW
L256:
        IFBIT   #TPND,[B]
        JP      DCOU
        JP      L256
DCOU:   RBIT    #TPND,[B]   ;RESET TIMER PENDING FLAG
        DRSZ    COUNT       ;DECREMENT PULSE COUNTER
                            ;COUNTER = 0 ?
        JP      L256        ;NO,LOOP 'TIL 255 PULSES
                            ;HAVE BEEN MEASURED

ESTORE:                     ;STORE END VALUE

        LD      CNTRL,#00   ;STOP TIMER
        LD      B,#STALO    ;POINT TO START VALUE LOW BYTE
        LD      A,[X+]      ;LOAD END VALUE LOW BYTE
        X       A,[B]       ;LOAD ACCU WITH STARTVALUE LOW BYTE
                            ;& STALO WITH END VALUE LOW BYTE
```

TL/DD/10788–5

```
        SC
        SUBC    A,[B]       ;SUBTRACT ENDVALUE LOW BYTE
                            ;FROM STARTVALUE LOW BYTE
        X       A,[B+]      ;STORE RESULT IN STALO,
                            ;POINT TO STAHI
        LD      A,[X]       ;LOAD ACCU WITH ENDVALUE HIGH BYTE
        X       A,[B]       ;LOAD ACCU WITH STARTVALUE HIGH BYTE
                            ;& STAHI WITH ENDVALUE HIGH BYTE
        SUBC    A,[B]       ;SUBTRACT ENDVALUE HIGH BYTE FROM
                            ;STARTVALUE HIGH BYTE
        X       A,[B]       ;STORE RESULT IN STAHI
        RET
        .END
```

TL/DD/10788–6

2-WAY MULTIPLEXED LCD DRIVE

Today a wide variety of LCDs, ranging from static to multiplex rates of 1:64 are available on the market. The multiplex rate of a LCD can be determined by the number of its backplanes (segment-common plate). The higher the multiplex rate the more individual segments can be controlled using only one line. e.g. a static LCD only has one backplane; only one segment can be controlled with one line. A two-way multiplexed LCD has two backplanes and two segments can be controlled with one line, etc.

Common to all LCDs is the fact that the drive voltage applied to the backplane(s) and segments has to be alternating. DC-components higher than 100 mV can cause electrochemical reactions (refer to manufacturer's spec), which reduce reliability and lifetime of the display.

If the multiplex ratio of the LCD is N and the amount of available outputs is M, the number of segments that can be driven is:

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

So the maximum number of a 2-way mux LCD's segments that can be driven with a COP800 in 28-pin package (if all outputs can be used to drive the LCD) is:

$$S = (18 - 2) * 2 = 32$$

During one LCD refresh cycle tx (typical values for $1/tx = fx$ are in the range 30 Hz . . . 60 Hz), three different voltages levels: Vop, 0.5*Vop and 0V have to be generated. The ''off'' voltage across a segment is not 0V as with static LCDs and also the ''on'' voltage is not Vop, but only a fraction of it. The ratio of ''on'' to ''off'' r.m.s.-voltage (discrimination) is determined by the multiplex ratio and the number of voltage levels involved. The most desirable discrimination ratio is one that maximizes the ratio of V_{ON} to V_{OFF}, allowing the maximum voltage difference between activated and non-activated states. In general the maximum achievable ratio for any particular value of N is given by:

$$(V_{ON}/V_{OFF})\ \max = SQR\ ((SQR(N) + 1)/(SQR(N) - 1))$$

$$SQR = \text{square root}$$

Using this formula the maximum achievable discrimination ratio for a 2-way multiplex LCD is 2.41, however, it is also possible to order a customized display with a smaller ratio. For ease of operation, most LCD drivers use equal voltage steps (0V, 0.5 *Vop, Vop). Thus a discrimination ratio of 2.24 is achieved. When using the COP800 to drive a 2-way multiplexed LCD the only external hardware required to achieve the three voltage steps are 4 equal resistors that form two voltage dividers—one for each backplane *(Figure 1)*. The procedure is to set G4 and G5 to ''0'' for 0V, to HI-Z (TRI-STATE®) for 0.5*Vop and to ''1'' in order to establish Vop at the backplane electrodes.

With the COP800 each I/O pin can be set individually to TRI-STATE, ''1'' or ''0'', so this procedure can be implemented very easily.

The current consumption of typical LCDs is in the range of 3 μA to 4 μA (at Vop = 4.5V, 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 LCD's current consumption increases dramatically, which is the reason why many LCD manufacturers recommend not using a refresh frequency higher than 60 Hz.

Timing Considerations

As shown in *Figures 2* and *3*, one LCD refresh cycle tx is subdivided into four equally distant time sections 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. Considering a refresh frequency of 50 Hz (tx = 20 ms) ta, tb, tc and td are equal to 5 ms; a COP800 running from an external clock of 2 MHz has an internal instruction cycle time of 5 μs and a typical current consumption of less than 350 μA (at V_{CC} = 3V and room temperature), thus meeting both the requirements of low current consumption and additional computing power between LCD refreshes.

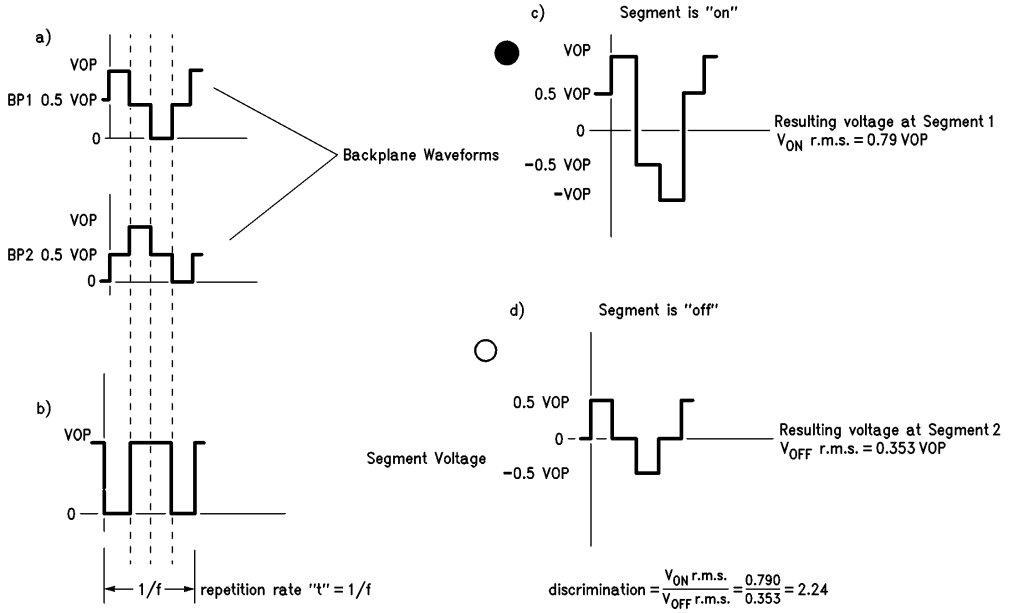
The timing is done using the COP800's 16-bit timer in the PWM autoload mode. The timer and the assigned 16-bit autoload register are preset with proper values. By setting the TRUN-flag in the CNTRL-register the timer is decremented each instruction cycle. A flag (TPND) is set at underflow and the timer is automatically reloaded with the value stored in the autoload-register. Timer underflow can also be programmed to generate an interrupt.

Segment Control

Figure 2 shows the voltage-waveforms applied to the two backplane-electrodes (a) and the waveform at a segement-electrode (b), which is needed to switch segment A on and segment B off. The resulting voltage over the segments (c and d) is achieved by subtracting waveform (b) from BP1 (segment A) and waveform (b) from BP2 (segment B).

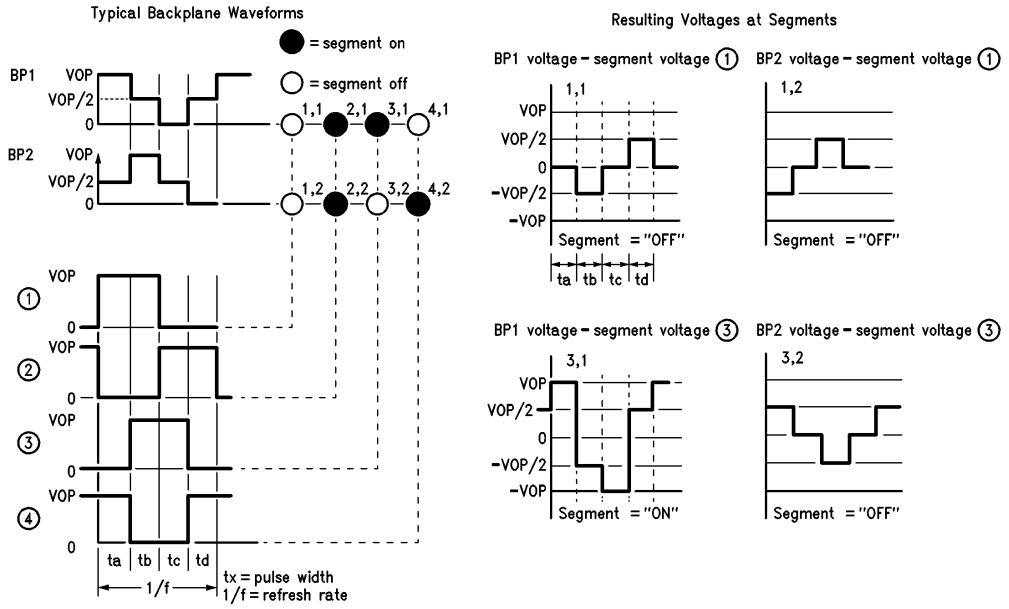
Figure 3 shows the four different waveforms which must be generated to meet all possible combinations of two segments connected to the same driving terminal (off-off, on-off, off-on, on-on).

Figure 4 shows the internal segment and backplane connections for a typical 2-way mux LCD.



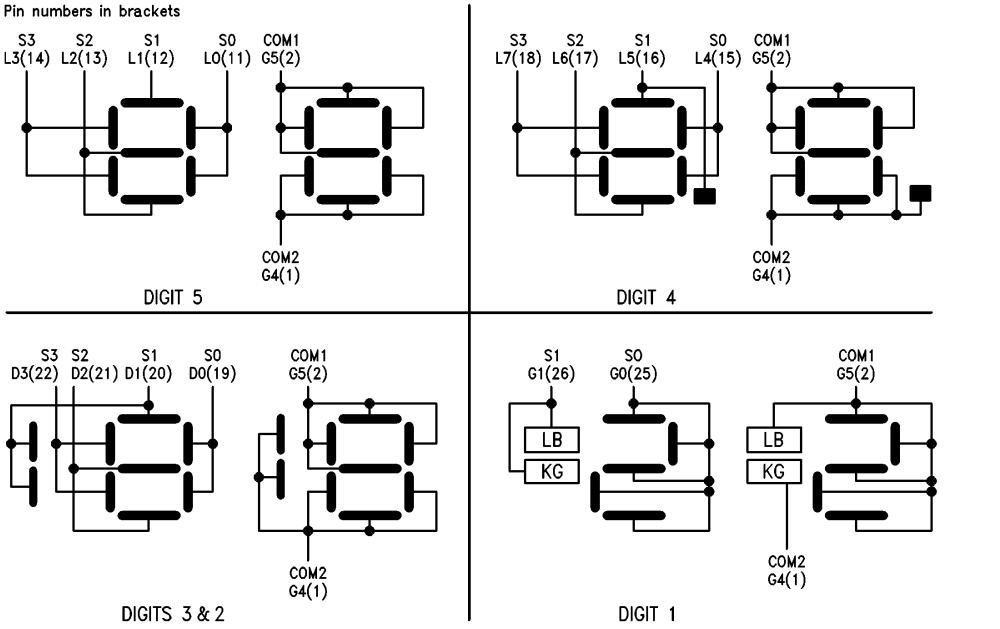
TL/DD/10788–7

FIGURE 2. LCD Waveforms



TL/DD/10788–8

FIGURE 3. Backplane and Segment Voltage Scheme for 1:2 Mux LCD-Drive



TL/DD/10788–9

FIGURE 4. Customized LCD Display (Backplane and Segment Organization)

LCD Drive Subroutine

The LCD drive subroutine DISPL converts a 16-bit binary value to a 24-bit BCD-value for easier display data fetch. The drive subroutine itself is built up of a main routine doing the backplane refresh and 7 subroutines (SEG0, SEG1, SEG2, SEG3, SEGOUT, TTPND, DISPD). The subroutines SEG0 to SEG4 are used to get the LCD segment data from a look-up table in ROM for time phases ta, tb, tc and td respectively. Subroutine SEGOUT writes the segment data for each time phase to the corresponding output ports. One time phase takes 5 ms, giving a total refresh cycle time of 20 ms (50 Hz). The exact timing is done by using the COP800 16-bit timer in the PWM autoload mode. In that mode the timer is reloaded with the value stored in the autoload register on every timer underflow. At the same time the timer pending flag is set. The subroutine TTPND checks this flag in a loop. If the timer pending flag is set, this subroutine resets it and returns to the calling program. Thus a 5 ms time delay is created before the segment and backplane data for the next time phase is written to the output ports. Finally the subroutine DISPD switches off the LCD by setting the backplane and segment connections to ''0''. In this digital scale application a frequency measurement is made while the LCD is off. Then the weight is calculated from this frequency and is displayed for 10s. After this 10s the LCD is switched off again and the COP800 is programmed to enter the current saving HALT mode ($I_{DD} < 10\ \mu A$). A new weight cycle on the digital scale is initiated by pressing a push button, which causes a reset of the microcontroller.

CONCLUSIONS

National Semiconductor's COP800 Microcontroller family is ideally suited for use with V/F converters and 2-way multiplexed LCDs, as they offer features, which are essential for these types of applications. The high resolution, 3-mode programmable 16-bit timer allows precise frequency measurement in the input capture mode with minimum software overhead. The timer's PWM autoreload mode offers an easy way to implement a precise timebase for the LCD refresh. The COP800's programmable I/O ports provide flexibility in driving 2-way multiplexed LCDs directly. The COP800 family, fabricated using M2CMOS technology, offers both low voltage (min V_{CC} of 2.5V) and low current drain.

REFERENCES

1. National Semiconductor, ''Linear Databook 2, Rev. 1'' LM331, LM331A datasheets pages 3–285 ff.
2. National Semiconductor, ''Linear Applications Databook, 1986'', ''Versatile monolithic V/Fs can compute as well as convert with high accuracy'', pages 1213 ff.
3. National Semiconductor, ''Microcontrollers Databook, Rev. 1'', COP820C/COP840C datasheets pages 2–7 ff.
4. U. Tietze, Ch. Schenk, ''Halbleiter-Schaltungstechnik'' 8.Auflage 1986, Springer Verlag, ISBN 0-387-16720-X, ''Funktionsgeneratoren mit steuerbarer Frequenz'', pages 465 ff, ''Multivibratoren'', pages 183 ff.
5. Lucid Displays, ''LCD design guide'', English Electric Valve Company Ltd., Chelmsford, Essex, Great Britain.

APPENDIX—Software Routines

```
;LOOKUP TABLE FOR  CUSTOMIZED 2-WAY MULIPLEX LCD

. = X'200      ;START LOOK-UP TABLE AT ROM ADRESS 200
;TIMEPHASE Ta 7 SEGMENT DATA
.BYTE   004     ;"0" AND ".0"
.BYTE   00E     ;"1" AND ".1"
.BYTE   008     ;"2" AND ".2"
.BYTE   008     ;"3" AND ".3"
.BYTE   002     ;"4" AND ".4"
.BYTE   001     ;"5" AND ".5"
.BYTE   001     ;"6" AND ".6"
.BYTE   00C     ;"7" AND ".7"
.BYTE   000     ;"8" AND ".8"
.BYTE   000     ;"9" AND ".9"
.BYTE   00F     ;" " AND ". "

;SPECIAL SEGMENTS TIMPHASE Ta
.BYTE   001     ;"LB"
.BYTE   000     ;"LB 2"
.BYTE   003     ;"KG"
.BYTE   002     ;"KG 2"

. = .+ 1
;TIMEPHASE Tb 7 SEGMENT DATA
.BYTE   002     ;"0"
.BYTE   00E     ;"1"
.BYTE   003     ;"2"
.BYTE   00A     ;"3"
.BYTE   00E     ;"4"
.BYTE   00A     ;"5"
.BYTE   002     ;"6"
.BYTE   00E     ;"7"
```

TL/DD/10788–10

```
        .BYTE   002     ;"8"
        .BYTE   00A     ;"9"
        .BYTE   00F     ;" "
        .BYTE   000     ;".0"
        .BYTE   00C     ;".1"
        .BYTE   001     ;".2"
        .BYTE   008     ;".3"
        .BYTE   00C     ;".4"
        .BYTE   008     ;".5"
        .BYTE   000     ;".6"
        .BYTE   00C     ;".7"
        .BYTE   000     ;".8"
        .BYTE   008     ;".9"
        .BYTE   00D     ;". "

        .LOCAL
TTPND:
        LD      B,#PSW
$LOOP:
        IFBIT   #TPND,[B]
        JP      $END
        JP      $LOOP
$END:
        RBIT    #TPND,[B]
        LD      B,#PORTGD
        RET
        .LOCAL

        . = .+1
        ;TIMEPHASE Tc 7 SEGMENT DATA
        .BYTE   00B     ;"0" AND ".0"
        .BYTE   001     ;"1" AND ".1"
        .BYTE   007     ;"2" AND ".2"
        .BYTE   007     ;"3" AND ".3"
        .BYTE   00D     ;"4" AND ".4"
        .BYTE   00E     ;"5" AND ".5"
        .BYTE   00E     ;"6" AND ".6"
        .BYTE   003     ;"7" AND ".7"
        .BYTE   00F     ;"8" AND ".8"
        .BYTE   00F     ;"9" AND ".9"
        .BYTE   000     ;" " AND ". "

COPY:                       ;COPY 2BYTES POINTED TO
                            ;BY B AND B+1 TO RAM
                            ;POINTED TO BY X AND X+1
        LD      A,[B+]
        X       A,[X+]
        LD      A,[B+]
        X       A,[X+]
        RET
        .LOCAL
```

TL/DD/10788–11

```
        ;TIMEPHASE Td 7 SEGMENT DATA
        .BYTE   00D     ;"0"
        .BYTE   001     ;"1"
        .BYTE   00C     ;"2"
        .BYTE   005     ;"3"
        .BYTE   001     ;"4"
        .BYTE   005     ;"5"
        .BYTE   00D     ;"6"
        .BYTE   001     ;"7"
        .BYTE   00D     ;"8"
        .BYTE   005     ;"9"
        .BYTE   000     ;" "
        .BYTE   00F     ;".0"
        .BYTE   003     ;".1"
        .BYTE   00E     ;".2"
        .BYTE   007     ;".3"
        .BYTE   003     ;".4"
        .BYTE   007     ;".5"
        .BYTE   00F     ;".6"
        .BYTE   003     ;".7"
        .BYTE   00F     ;".8"
        .BYTE   007     ;".9"
        .BYTE   002     ;". "

        ;SPECIAL SEGMENTS TIMEPHASE Tb
        .BYTE   003     ;"LB"
        .BYTE   003     ;"LB 2 "
        .BYTE   001     ;"KG"
        .BYTE   001     ;"KG 2"

        ;SPECIAL SEGMENTS TIMPHASE Tc
        .BYTE   002     ;"LB"
        .BYTE   003     ;"LB 2"
        .BYTE   000     ;"KG"
        .BYTE   001     ;"KG 2"

        ;SPECIAL SEGMENTS TIMEPHASE Td
        .BYTE   000     ;"LB"
        .BYTE   000     ;"LB 2"
        .BYTE   002     ;"KG"
        .BYTE   002     ;"KG 2"

        .END
;DISPL:
;INPUT PARAMETER: COUNT2 =RAM REGISTER, WHICH CONTAINS
;THE DISPLAY TIME IN SEC.
;EXAMPLE COUNT2= 1-> DISPLAY TIME IS 1SEC.

;LCD DRIVE ROUTINE FOR CUSTOMIZED 2 WAY MULTIPLEX
;LCD
```

TL/DD/10788–12

```
;ROUTINE CONVERTS BCD DATA STORED IN RAM LOCATIONS
;BCDLO, BCDHI INTO LCD OUTPUT DATA STORED AT
;MWBUF0 = LPORT DATA
;MWBUF1 = DPORT DATA
;MWBUF2 = G-PORT DATA (G0,G1 ONLY, OTHER BITS
;                      STAY UNCHANGED)
;SUBROUTINES INCLUDED:
;SEG0: GETS LCD SEGMENT DATA FOR TIMEPHASE TA
;SEG1: GETS LCD SEGMENT DATA FOR TIMEPHASE TB
;SEG2: GETS LCD SEGMENT DATA FOR TIMEPHASE TC
;SEG3: GETS LCD SEGMENT DATA FOR TIMEPHASE TD
;DISPD:  SWITCHES THE DISPLAY OFF AND
;        CONFIGURES G-,L- AND D-PORTS
;TTPND:  CHECKS TIMER PENDING FLAG (REFRESH
;        RATE GENERATION)
;SEGOUT: OUTPUTS LCD SEGMENT AND BACKPLANE DATA
;SUBROUTINES  SEG0...  SEG1  MUST FOLLOW DIRECTLY  AFTER  LOOK-UP
;TABLE, BECAUSE OF THE USE OF THE LAID-INSTRUCTION

        .LOCAL
SEG0:
        LD      B,#OFF1  ;POINT TO OFFSET 1 REG.
        LD      [B+],#000
        LD      [B+],#000
        LD      A,#00B
$TWO:
        IFBIT   #05,BCDHI ;WEIGHT >= 200 POUNDS?
        INCA              ;YES DISPLAY DIGIT5 ("2")

$POUND:
        IFBIT   #POUND,FLAG
        JP      $LPORT
        ADD     A,#002
$LPORT:
        X       A,[B]
        LD      X,#BCDLO
        LD      B,#MWBUF0
        LD      A,[X]
        AND     A,#00F  ;ELIMINATE  DIGIT1 BITS
        ADD     A,OFF2
        LAID            ;GET DIGIT1 DATA
        X       A,[B]   ;SAVE DIGIT1 DATA
                        ;IN MWBUF0
        LD      A,[X+]
        AND     A,#0F0  ;ELIMINATE DIGIT1 BITS
        SWAP    A
        ADD     A,OFF1  ;ALWAYS DISPLAY DECIMAL POINT
        LAID            ;GET DIGIT1 DATA
        SWAP    A
        OR      A,[B]   ;STORE DIGIT1 AND
        X       A,[B+]  ;DIGIT2 DATA IN MWBUF0
```

TL/DD/10788–13

```
$DPORT:
        LD      A,[X]
        IFBIT   #04,BCDHI
        JP      $ADD1
        AND     A,#00F
        ADD     A,OFF2  ;DISPLAY NO LEADING ZERO
        JP      $GET
$ADD1:
        AND     A,#00F
        ADD     A,OFF1  ;DISPLAY "1" (DIGIT4)
$GET:
        LAID            ;GET DIGIT3 DATA
        X       A,[B+]  ;STORE DIGIT3 DATA IN
                        ;MWBUF1
$GPORT:
        LD      A,OFF3
        LAID            ;GET DIGIT5 ("2") AND SPECIAL
                        ;SEGMENT DATA
        OR      A,#0FC  ;SET BITS 2...7 TO 1
        X       A,[B]   ;SAVE  DATA IN MWBUF2
        RET
SEG1:
        LD      B,#OFF1
        LD      [B+],#01B
        LD      [B+],#010
        LD      A,#056
        JP      $TWO

SEG2:
        LD      B,#OFF1
        LD      [B+],#030
        LD      [B+],#030
        LD      A,#05A
        JP      $TWO

SEG3:
        LD      B,#OFF1
        LD      [B+],#04B
        LD      [B+],#040
        LD      A,#05E
        JP      $TWO
        .LOCAL


DISPL:
        IFBIT   #POUND,FLAG
        JP      MULT2
        JP      LDT
MULT2:                      ;CALCULATE WEIGHT IN POUNDS
        LD      B,#BUF12LO  ;(Multiplication of kg *2.2)
        LD      [B+],#22
```

TL/DD/10788–14

```
        LD      X,#STALO
        JSR     MULBI168
        LD      B,#BUF12LO
        JSR     COPY
        LD      STAHI+1,#00
        LD      DIV0,#10
        JSR     DIVBI248
LDT:
        JSR     BINBCD16    ;CONVERT BINARY TO BCD WEIGHT
        LD      COUNT,#50   ;REPEAT DISPLAY LOOP 50 TIMES
                            ;(=1 SEC DISPLAY TIME)
        LD      B,#TMRLO
        LD      [B+],#0E8 ;LOAD TIMER WITH 1000(03E8h)
        LD      [B+],#003 ;(=50 Hz LCD REFRESH AT tc=5us)
        LD      [B+],#0E8 ;LOAD AUTOREG. WITH 1000
        LD      [B+],#003
        LD      [B+],#090 ;CNTRL-REG.:"TIMER WITH AUTO-
                          ;LOAD"- MODE,START TIMER
        LD      [B+],#010 ;PSW-REG.:RESET TPND FLAG
DISP1:
        JSR     SEG0      ;GET 7-SEGM. DATA FOR REFRESH
                          ;TIMEPHASE Ta
        JSR     TTPND     ;TEST TIMER PENDING FLAG
TP0:                      ;BACKPLANE REFRESH Ta
        SBIT    #BP1,[B]
        LD      A,[B+]    ;POINT TO G-CONFIG.-REG.
        RBIT    #BP2,[B]
        SBIT    #BP1,[B]
        LD      A,[B-]    ;POINT TO G-DATA REG.
        RBIT    #BP2,[B]
        JSR     SEGOUT    ;SEGMENT DATA OUT
        JSR     SEG1      ;GET 7-SEG. DATA FOR Tb
        JSR     TTPND
TP1:
        SBIT    #BP2,[B]
        LD      A,[B+]    ;POINT TO G-CONF.-REG.
        RBIT    #BP1,[B]
        SBIT    #BP2,[B]
        LD      A,[B-]    ;POINT TO G-DATA REG.
        RBIT    #BP1,[B]
        JSR     SEGOUT
        JSR     SEG2      ;GET 7-SEGM. DATA FOR Tc
        JSR     TTPND
TP2:
        RBIT    #BP1,[B]
        LD      A,[B+]    ;POINT TO G-CONFIG.-REG.
        RBIT    #BP2,[B]
        SBIT    #BP1,[B]
        LD      A,[B-]    ;POINT TO G-DATA-REG.
        RBIT    #BP2,[B]
        JSR     SEGOUT
```

TL/DD/10788–15

```
        JSR     SEG3
        JSR     TTPND
TP3:
        RBIT    #BP1,[B]
        RBIT    #BP2,[B]
        LD      A,[B+]
        RBIT    #BP1,[B]
        SBIT    #BP2,[B]
        JSR     SEGOUT
        DRSZ    COUNT
        JP      DISP1
        LD      COUNT,#50
        DRSZ    COUNT2      ;10SEC OVER?
        JP      DISP1       ;NO, DISPLAY WEIGHT
        JSR     DISPD
        RET                 ;YES ROUTINE FINISHED


DISPD:                      ;SWITCH DISPLAY OFF
        LD      B,#PORTLD
        LD      [B+],#000   ;OUTPUT 0 TO L PORT
        LD      [B+],#0FF   ;L-PORT = OUTPUT PORT
        LD      B,#PORTGD
        LD      [B+],#000   ;OUTPUT 0 TO G OUTPUTS
        LD      [B+],#037   ;G0..G2,G4,G5=OUTPUTS
        LD      PORTD,#000 ;OUTPUT 0 TO D-PORT
        RET

SEGOUT:
        LD      B,#MWBUF0
        LD      A,[B+]     ;POINT TO MWBUF1
        X       A,PORTLD   ;OUTPUT 7 SEG. DATA IN
                           ;MWBUF0 TO L-PORT
        LD      A,[B+]     ;POINT TO MWBUF2
        X       A,PORTD    ;OUTPUT MWBUF1 TO D-PORT
        LD      X,#PORTGD
        LD      A,[X]
        AND     A,[B]      ;AND MWBUF2 WITH PORTGD
                           ;LEAVE BITS 2...7 UNCHANGED
        X       A,[B]      ;STORE RESULT (A')IN
                           ;MWBUF2,LOAD A WITH
                           ;ORIGINAL MWBUF2 VALUE
        AND     A,#003     ;AND 007 WITH ORIGINAL
                           ;MWBUF2 (A''),SET BITS 0,1 TO
                           ;CORRECT VALUE
        OR      A,[B]      ;OR  A' WITH A'',RESTORE ORIGINAL
                           ;G2...G7 BITS
        X       A,[X]      ;OUTPUT RESULT TO G-PORT
        RET
```

TL/DD/10788–16

```
;16 BIT BINARY TO BCD CONVERSION
;THE MEMORY ASSIGNMEMTS ARE AS FOLLOWS:

;BINLO: RAM ADRESS BINARY LOW BYTE
;BCDLO: RAM ADRESS BCD LOW BYTE
;COUNT: RAM ADRESS SHIFT COUNTER (0F0...0FB,0FF)

;BCD NUMBER IN BCDLO,BCDLO+1,BCDLO+2
;
;MEMORY ADRESS        M(BINLO+1)  M(BINLO)
;DATA                 BINARY HB   BINARY LOW BYTE
;
;MEMORY ADRESS        M(BCDLO+2)  M(BCDLO+1)  M(BCDLO)
;DATA                 BCD HB      BCD         BCD LOW BYTE
;


        BINLO =  STALO
        .LOCAL
        $BCDT = (BCDLO + 3) & 0F
        $BINT = (BINLO + 2) & 0F
BINBCD:
        LD      COUNT,#16 ;LOAD CONTROL REGISTER WITH
                          ;NUMBER OF LEFTSHIFTS TO
                          ;EXECUTE
        LD      B,#BCDLO  ;LOAD BCD-NUMBER LOWEST BYTE
                          ;ADRESS

$CBCD:                    ;CLEAR BCD RAM-REGISTERS
        LD      [B+],#00
        IFBNE   #$BCDT
        JP      $CBCD
$LSH:                     ;LEFTSHIFT BINARY NUMBER
        LD      B,#BINLO
        RC
$LSHFT:
        LD      A,[B]
        ADC     A,[B]     ;IF MSB IS SET, SET CARRY
        X       A,[B+]
        IFBNE   #$BINT
        JP      $LSHFT
        LD      B,#BCDLO
$BCDADD:
        LD      A,[B]
        ADD     A,#066    ;ADD CORRECTION FACTOR
        ADC     A,[B]     ;LEFTSHIFT BCD NUMBER
                          ;(BCD=2**WEIGHT OF
                          ;BINARY BIT(=CARRY BIT))
        DCOR    A         ;DECIMAL CORRECT ADDITION
        X       A,[B+]
        IFBNE   #$BCDT
```

TL/DD/10788–17

```
        JP      $BCDADD
        DRSZ    COUNT     ;DECREMENT SHIFT COUNTER
        JP      $LSH
        RET
        .LOCAL
```

TL/DD/10788–18

```
;BINARY DIVIDE 24BIT BY  8BIT (Q=Y/Z)
;YL: LOW BYTE RAM ADRESS DIVIDEND
;ZL: LOW BYTE RAM ADRESS DIVISOR
;CNTR: RAM ADRESS SHIFT COUNTER (0F0...0FB,0FF)

;QUOTIENT AT RAM LOCATIONS YL..YL+2
;REMAINDER AT YL+3
;QUOTIENT IS ALL '1's IF DIVIDE BY ZERO, REMAINDER
;THEN CONTAINS YL

;THE MEMORY ASSIGNMENTS ARE AS FOLLOWS:
;
;       M(YH+1)  M(YH)          M(YL+1)  M(YL)
;       0        Y(HIGH BYTE)   Y        Y(LOW BYTE)
;      ------------------------------------------------
;       M(ZL)
;       Z
;
;ROUTINE NEEDS 1.21ms FOR EXECUTION AT tc=1us

        ZL     =  DIV0
        YL     =  STALO
        CNTR   =  COUNT
        .LOCAL
        $YH     = YL+2
        $BTY    = ($YH&00F)+2 ;PARAMETER FOR "IFBNE"-INSTR.

DIVBI248:
        LD      CNTR,#018  ;INITIALIZE SHIFT COUNTER
        LD      B,#$YH+1   ;FOR 24 COUNTS
        LD      [B],#000   ;PUT 0 IN M(YH+1)
        LD      X,#$YH+1
$LSHFT:
        LD      B,#YL   ;LEFT SHIFT DIVIDEND
        RC
$LUP:
        LD      A,[B]
        ADC     A,[B]
        X       A,[B+]
        IFBNE   #$BTY
        JP      $LUP
        LD      B,#ZL
        IFC
        JP      $SUBT
$TSUBT:
                        ;SUBTRACT AND TEST
        SC              ;SUBTRACT Z FROM M(YH+1,YH+2)
        LD      A,[X]
        SUBC    A,[B]
        IFNC
        JP      $TEST
```

TL/DD/10788–19

```
$SUBT:
                        ;SUBTRACT Z FROM M(YH+1,YH+2)
        LD      A,[X]
        SUBC    A,[B]
        X       A,[X]
        LD      B,#YL
        SBIT    #0,[B]
$TEST:
        DRSZ    CNTR    ;24 SHIFTS EXECUTED?
        JP      $LSHFT  ;NO, LEFT SHIFT DIVIDEND
        RET
        .LOCAL

;BINARY MULTIPLIES A 16BIT VALUE (X1)
;WITH A 8BIT VALUE (X2): M = X1 * X2

;X1L: RAM ADRESS X1 LOW BYTE
;X2L: RAM ADRESS X2
;COUNT RAM ADRESS SHIFT COUNTER

;M IS STORED AT RAM ADRESSES X2L...X2L+2

;THE MEMORY ASSIGNMEMTS ARE AS FOLLOWS:
;MEMORY       M(X2L+2)  M(X2L+1)       M(X2L)
;DATA         0         0              X2
;-----------------------------------------------------
;MEMORY        M(X1L+1)   M(X1L)
;DATA          X1(H.B.)   X1(LOW BYTE)

;THE EXECUTION TIME FOR THE ROUTINE AT tc=1us IS 240us
;


        .LOCAL

MULBI168:

        LD      COUNT,#9 ;PRESET SHIFT COUNTER
        LD      [B+],#00 ;PRESET X2L+1,X2L+2 WITH '0'
        LD      [B],#00
        RC
$LOOP:
        LD      A,[B]   ;RIGHT SHIFT
        RRCA
        X       A,[B-]
        LD      A,[B]
        RRCA
        X       A,[B-]
        LD      A,[B]
        RRCA
        X       A,[B+]
```

TL/DD/10788–20

```
        LD      A,[B+] ;INCREMENT B POINTER
        IFNC           ;MOST SIGN. BIT OF X2 SET?
        JP      $TEST  ;NO,TEST SHIFT COUNTER
        RC             ;YES,RESET CARRY
        LD      A,[B-] ;POINT TO 2nd HIGHEST BYTE
                       ;OF RESULT
        LD      A,[X+] ;DO WEIGHTED ADD
        ADC     A,[B]
        X       A,[B+]
        LD      A,[X-]
        ADC     A,[B]
        X       A,[B]
$TEST:
        DRSZ    COUNT  ;8 RIGHT SHIFTS EXECUTED?
        JP      $LOOP  ;NO,SHIFT
        RET            ;YES,MULIPLICATION FINISHED
        .LOCAL
        .END
```

TL/DD/10788–21

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