# Barometric Pressure Measurement Using Semiconductor Pressure Sensors

by: **Chris Winkler and Jeff Baum**
**Discrete Applications Engineering**

## ABSTRACT

The most recent advances in silicon micromachining technology have given rise to a variety of low-cost pressure sensor applications and solutions. Certain applications had previously been hindered by the high-cost, large size, and overall reliability limitations of electromechanical pressure sensing devices. Furthermore, the integration of on-chip temperature compensation and calibration has allowed a significant improvement in the accuracy and temperature stability of the sensor output signal. This technology allows for the development of both analog and microcomputer-based systems that can accurately resolve the small pressure changes encountered in many applications. One particular application of interest is the combination of a silicon pressure sensor and a microcontroller interface in the design of a digital barometer. The focus of the following documentation is to present a low-cost, simple approach to designing a digital barometer system.

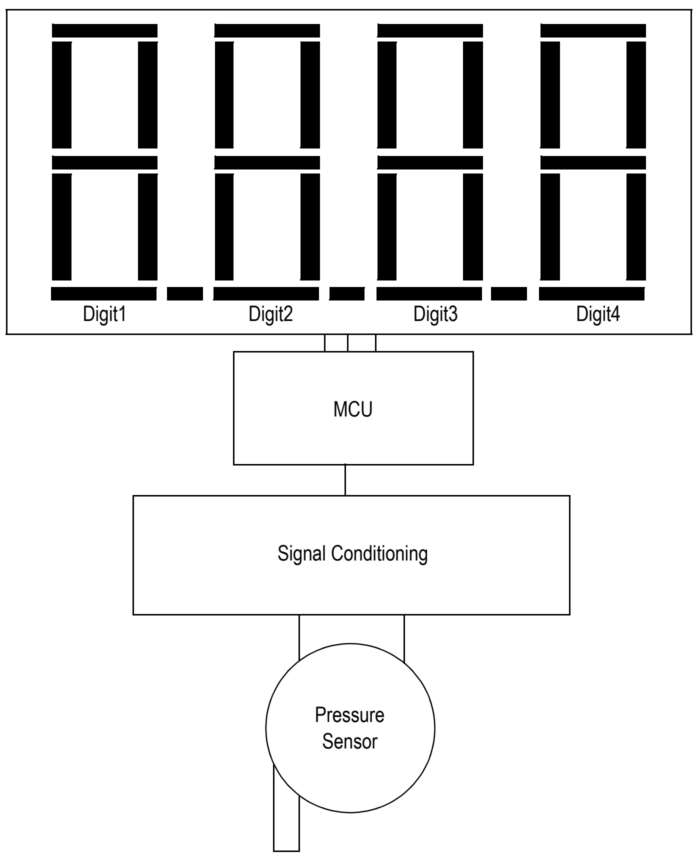


**Figure 1. Barometer System**

## INTRODUCTION

Figure 1 shows the overall system architecture chosen for this application. This system serves as a building block, from which more advanced systems can be developed. Enhanced accuracy, resolution, and additional features can be integrated in a more complex design.

There are some preliminary concerns regarding the measurement of barometric pressure which directly affect the design considerations for this system. Barometric pressure refers to the air pressure existing at any point within the earth's atmosphere. This pressure can be measured as an absolute pressure, (with reference to absolute vacuum) or can be referenced to some other value or scale. The meteorology and avionics industries traditionally measure the absolute pressure, and then reference it to a sea level pressure value. This complicated process is used in generating maps of weather systems. The atmospheric pressure at any altitude varies due to changing weather conditions over time. Therefore, it can be difficult to determine the significance of a particular pressure measurement without additional information. However, once the pressure at a particular location and elevation is determined, the pressure can be calculated at any other altitude. Mathematically, atmospheric pressure is exponentially related to altitude. This particular system is designed to track variations in barometric pressure once it is calibrated to a known pressure reference at a given altitude.

For simplification, the standard atmospheric pressure at sea level is assumed to be 29.9 in-Hg. "Standard" barometric pressure is measured at particular altitude at the average weather conditions for that altitude over time. The system described in this text is specified to accurately measure barometric pressure variations up to altitudes of 15,000 ft. This altitude corresponds to a standard pressure of approximately 15.0 in-Hg. As a result of changing weather conditions, the standard pressure at a given altitude can fluctuate approximately ±1 in-Hg. in either direction. Table 1 indicates standard barometric pressures at several altitudes of interest.

**Table 1. Altitude versus Pressure Data**

| Altitude (Ft.) | Pressure (in-Hg) |
|---|---|
| 0 | 29.92 |
| 500 | 29.38 |
| 1,000 | 28.85 |
| 6,000 | 23.97 |
| 10,000 | 20.57 |
| 15,000 | 16.86 |

## SYSTEM OVERVIEW

In order to measure and display the correct barometric pressure, this system must perform several tasks. The measurement strategy is outlined below in Figure 2. First, pressure is applied to the sensor. This produces a proportional differential output voltage in the millivolt range. This signal must then be amplified and level-shifted to a single-ended, microcontroller (MCU) compatible level (0.5 – 4.5 V) by a signal conditioning circuit. The MCU will then sample the voltage at the analog-to-digital converter (A/D) channel input, convert the digital measurement value to inches of mercury, and then display the correct pressure via the LCD interface. This process is repeated continuously.

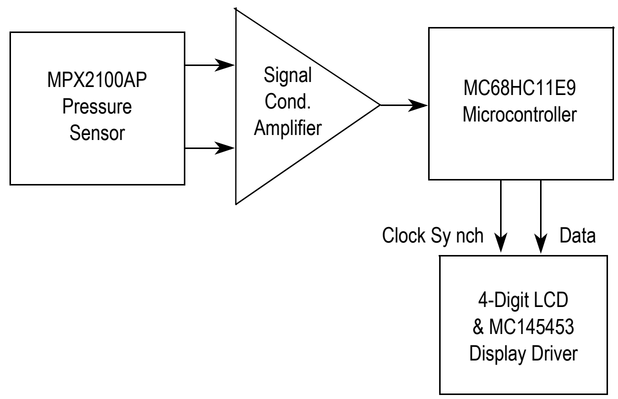


**Figure 2. Barometer System Block Diagram**

There are several significant performance features implemented into this system design. First, the system will digitally display barometric pressure in inches of mercury, with a resolution of approximately one-tenth of an inch of mercury. In order to allow for operation over a wide altitude range (0 – 15,000 ft.), the system is designed to display barometric pressures ranging from 30.5 in-Hg. to a minimum of 15.0 in-Hg. The display will read "lo" if the pressure measured is below 30.5 in-Hg. These pressures allow for the system to operate with the desired resolution in the range from sea-level to approximately 15,000 ft. An overview of these features is shown in Table 2.

**Table 2. System Features Overview**

| Display Units | in-Hg |
|---|---|
| Resolution | 0.1 in-Hg. |
| System Range | 15.0 – 30.5 in-Hg. |
| Altitude Range | 0 – 15,000 ft. |

## DESIGN OVERVIEW

The following sections are included to detail the system design. The overall system will be described by considering the subsystems depicted in the system block diagram, Figure 2. The design of each subsystem and its function in the overall system will be presented.

**Table 3. MPX2100AP Electrical Characteristics**

| Characteristic | Symbol | Minimum | Typical | Max | Unit |
|---|---|---|---|---|---|
| Pressure Range | POP | 0 | — | 100 | kPa |
| Supply Voltage | VS | — | 10 | 16 | Vdc |
| Full Scale Span | VFSS | 38.5 | 40 | 41.5 | mV |
| Zero Pressure Offset | Voff | — | — | ±1.0 | mV |
| Sensitivity | S | — | 0.4 | — | mv/kPa |
| Linearity | — | — | 0.05 | — | %FSS |
| Temperature Effect on Span | — | — | 0.5 | — | %FSS |
| Temperature Effect on Offset | — | — | 0.2 | — | %FSS |

## Pressure Sensor

The first and most important subsystem is the pressure transducer. This device converts the applied pressure into a proportional, differential voltage signal. This output signal will vary linearly with pressure. Since the applied pressure in this application will approach a maximum level of 30.5 in-Hg. (100 kPa) at sea level, the sensor output must have a linear output response over this pressure range. Also, the applied pressure must be measured with respect to a known reference pressure, preferably absolute zero pressure (vacuum). The device should also produce a stable output over the entire operating temperature range.

The desired sensor for this application is a temperature compensated and calibrated, semiconductor pressure transducer, such as the MPXM2102A series sensor family. The MPX2000 series sensors are available in full-scale pressure ranges from 10 kPa (1.5 psi) to 200 kPa (30 psi). Furthermore, they are available in a variety of pressure configurations (gauge, differential, and absolute) and porting options. Because of the pressure ranges involved with barometric pressure measurement, this system will employ an MPXM2102AS (absolute with single port). This device will produce a linear voltage output in the pressure range of 0 to 100 kPa. The ambient pressure applied to the single port will be measured with respect to an evacuated cavity (vacuum reference). The electrical characteristics for this device are summarized in Table 3.

As indicated in Table 3, the sensor can be operated at different supply voltages. The full-scale output of the sensor, which is specified at 40 mV nominally for a supply voltage of 10 Vdc, changes linearly with supply voltage. All non-digital circuitry is operated at a regulated supply voltage of 8 Vdc. Therefore, the full-scale sensor output (also the output of the sensor at sea level) will be approximately 32 mV.

$$\left(\frac{8}{10} \times 40\ \text{mV}\right)$$

The sensor output voltage at the systems minimum range (15 in-Hg.) is approximately 16.2 mV. Thus, the sensor output over the intended range of operations is expected to vary from 32 to 16.2 mV. These values can vary slightly for each sensor as the offset voltage and full-scale span tolerances indicate.

## Signal Conditioning Circuitry

In order to convert the small-signal differential output signal of the sensor to MCU compatible levels, the next subsystem includes signal conditioning circuitry. The operational amplifier circuit is designed to amplify, level-shift, and ground reference the output signal. The signal is converted to a single-ended, 0.5 – 4.5 Vdc range. The schematic for this amplifier is shown in Figure 3.

This particular circuit is based on classic instrumentation amplifier design criteria. The differential output signal of the sensor is inverted, amplified, and then level-shifted by an adjustable offset voltage (through $R_{offset1}$). The offset voltage is adjusted to produce 0.5 volts at the maximum barometric pressure (30.5 in-Hg.). The output voltage will increase for decreasing pressure. If the output exceeds 5.1 V, a zener protection diode will clamp the output. This feature is included to protect the A/D channel input of the MCU. Using the transfer function for this circuit, the offset voltage and gain can be determined to provide 0.1 in-Hg of system resolution and the desired output voltage level. The calculation of these parameters is illustrated below.

In determining the amplifier gain and range of the trimmable offset voltage, it is necessary to calculate the number of steps used in the A/D conversion process to resolve 0.1 in-Hg.

$$(30.5 - 15.0)\text{in-Hg} * 10\ \frac{\text{steps}}{\text{Hg}} = 155\ \text{steps}$$

The span voltage can now be determined. The resolution provided by an 8-bit A/D converter with low and high voltage references of zero and five volts, respectively, will detect 19.5 mV of change per step.

$V_{RH}$ = 5 V, $V_{RL}$ = 0 V

Sensor Output at 30.5 in-Hg = 32.44 mV
Sensor Output at 15.0 in-Hg = 16.26 mV
ΔSensor Output = ΔSO = 16.18 mV

$$\text{Gain} = \frac{3.04\ \text{V}}{\Delta\text{SO}} = 187$$

**Note:** 30.5 in-Hg and 15.0 in-Hg are the assumed maximum and minimum absolute pressures, respectively.

This gain is then used to determine the appropriate resistor values and offset voltage for the amplifier circuit defined by the transfer function shown below.

$$V_{out} = -\left[\frac{R_2}{R_1} + 1\right] * \Delta V + V_{off}$$

ΔV is the differential output of the sensor.

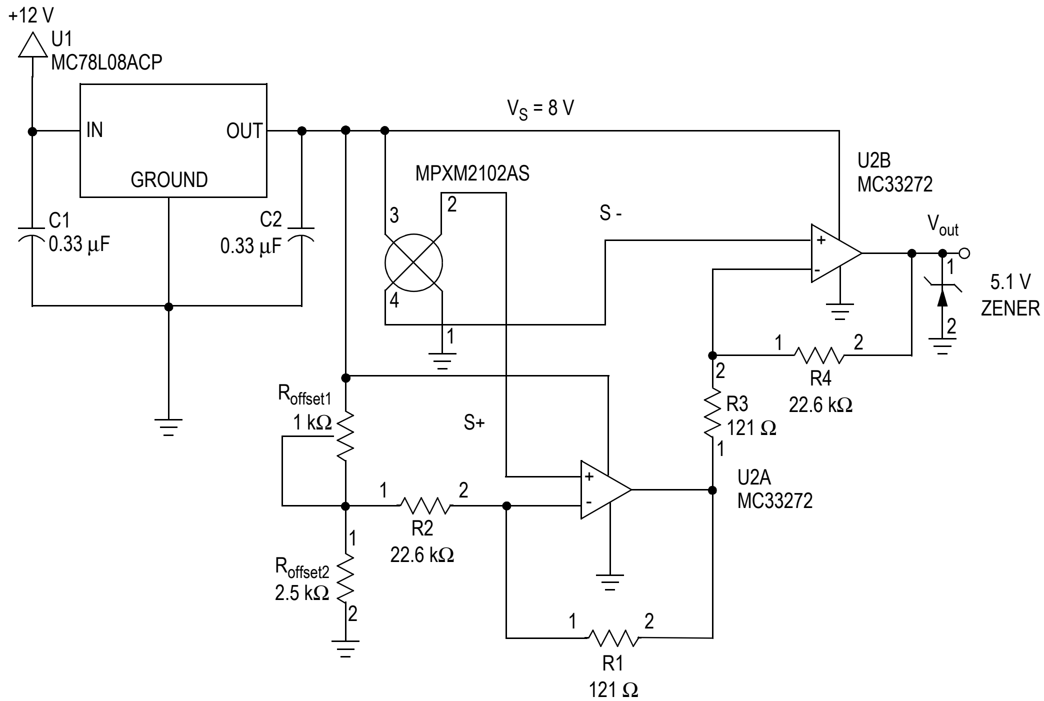


**Figure 3. Signal Conditioning Circuit**

The gain of 187 can be implemented with:

$R_1 \approx R_3 = 121\ \Omega$
$R_2 \approx R_4 = 22.6\ k\Omega$.

Choosing $R_{offset1}$ to be 1 kΩ and $R_{offset2}$ to be 2.5 kΩ, $V_{out}$ is 0.5 V at the presumed maximum barometric pressure of 30.5 in-Hg. The maximum pressure output voltage can be trimmed to a value other than 0.5 V, if desired via $R_{offset1}$. In addition, the trimmable offset resistor is incorporated to provide offset calibration if significant offset drift results from large weather fluctuations.

The circuit shown in Figure 3 employs an MC33272 (low-cost, low-drift) dual operational amplifier IC. In order to control large supply voltage fluctuations, an 8 Vdc regulator, MC78L08ACP, is used. This design permits use of a battery for excitation.

## Microcontroller Interface

The low cost of MCU devices has allowed for their use as a signal processing tool in many applications. The MCU used in this application, the MC68HC11, demonstrates the power of incorporating intelligence into such systems. The on-chip resources of the MC68HC11 include: an 8 channel, 8-bit A/D, a 16-bit timer, an SPI (Serial Peripheral Interface – synchronous), and SCI (Serial Communications Interface – asynchronous), and a maximum of 40 I/O lines. This device is available in several package configurations and product variations which include additional RAM, EEPROM, and/or I/O capability. The software used in this application was developed using the MC68HC11 EVB development system.

The following software algorithm outlines the steps used to perform the desired digital processing. This system will convert the voltage at the A/D input into a digital value, convert this measurement into inches of mercury, and output this data serially to an LCD display interface (through the on-board SPI). This process is outlined in greater detail below:

1. Set up and enable A/D converter and SPI interface.
2. Initialize memory locations, initialize variables.
3. Make A/D conversion, store result.
4. Convert digital value to inches of mercury.
5. Determine if conversion is in system range.

6a. Convert pressure into decimal display digits.
6b. Otherwise, display range error message.

7. Output result via SPI to LCD driver device.

The signal conditioned sensor output signal is connected to pin PE5 (Port E-A/D Input pin). The MCU communicates to the LCD display interface via the SPI protocol. A listing of the assembly language source code to implement these tasks is included in the appendix. In addition, the software can be downloaded directly from the Freescale MCU Freeware Bulletin Board (in the MCU directory). Further information is included at the beginning of the appendix.

## LCD Interface

In order to digitally display the barometric pressure conversion, a serial LCD interface was developed to communicate with the MCU. This system includes an MC145453 CMOS serial interface/LCD driver, and a 4-digit, non-multiplexed LCD. In order for the MCU to communicate correctly with the interface, it must serially transmit six bytes for each conversion. This includes a start byte, a byte for each of the four decimal display digits, and a stop byte. For formatting purposes, decimal points and blank digits can be displayed through appropriate bit patterns. The control of display digits and data transmission is executed in the source code through subroutines BCDCONV, LOOKUP, SP12LCD, and TRANSFER. A block diagram of this interface is included below.

## CONCLUSION

This digital barometer system described herein is an excellent example of a sensing system using solid state components and software to accurately measure barometric pressure. This system serves as a foundation from which more complex systems can be developed. The MPXM2102A series pressure sensors provide the calibration and temperature compensation necessary to achieve the desired accuracy and interface simplicity for barometric pressure sensing applications.

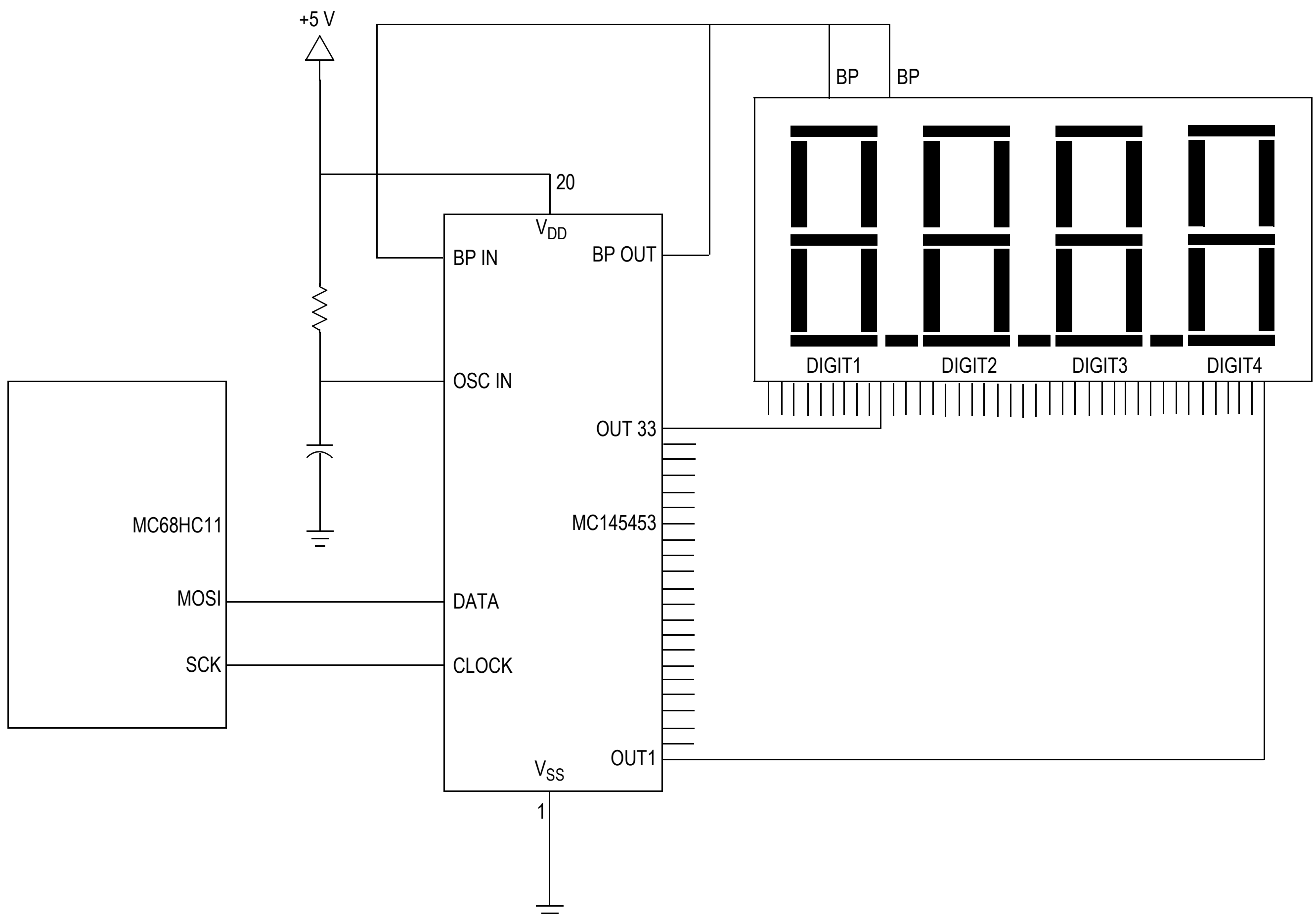


**Figure 4. LCD Display Interface Diagram**

# APPENDIX

MC68HC11 Barometer Software Available on:
Freescale Electronic Bulletin Board
MCU Freeware Line
8-bit, no parity, 1 stop bit
1200/300 baud
(512) 891-FREE (3733)

```
* BAROMETER APPLICATIONS PROJECT - Chris Winkler
* Developed: October 1st, 1992   - Freescale Discrete Applications
* This code will be used to implement an MC68HC11 Micro-Controller
* as a processing unit for a simple barometer system.
* The HC11 will interface with an MPX2100AP to monitor,store
* and display measured Barometric pressure via the 8-bit A/D channel
* The sensor output (32mv max) will be amplified to .5 - 2.5 V dc
* The processor will interface with a 4-digit LCD (FE202) via
* a Freescale LCD driver (MC145453) to display the pressure
* within +/- one tenth of an inch of mercury.
* The systems range is 15.0 - 30.5 in-Hg

*        A/D  & CPU Register Assignment
*        This code will use index addressing to access the
*        important control registers.  All addressing will be
*        indexed off of REGBASE, the base address for these registers.

REGBASE  EQU      $1000            * register base of control register
ADCTL             EQU      $30              * offset of A/D control register
ADR2              EQU      $32              * offset of A/D results register
ADOPT             EQU      $39              * offset for A/D option register location
PORTB             EQU      $04              * Location of PORTB used for conversion
PORTD             EQU      $08              * PORTD Data Register Index
DDRD              EQU      $09              * offset of Data Direction Reg.
SPCR              EQU      $28              * offset of SPI Control Reg.
SPSR              EQU      $29              * offset of SPI Status Reg.
SPDR              EQU      $2A              * offset of SPI Data Reg.

*        User Variables
*        The following locations are used to store important measurements
*        and calculations used in determining the altitude.  They
*        are located in the lower 256 bytes of user RAM

DIGIT1   EQU      $0001            * BCD blank digit (not used)
DIGIT2   EQU      $0002            * BCD tens digit for pressure
DIGIT3   EQU      $0003            * BCD tenths digit for pressure
DIGIT4   EQU      $0004            * BCD ones digit for pressure
COUNTER  EQU      $0005            * Variable to send 5 dummy bytes
POFFSET  EQU      $0010            * Storage Location for max pressure offset
SENSOUT  EQU      $0012            * Storage location for previous conversion
RESULT   EQU      $0014            * Storage of Pressure(in Hg) in hex format
FLAG              EQU      $0016            * Determines if measurement is within range

*        MAIN PROGRAM
*        The conversion process involves the following steps:
*
*                          1.               Set-Up SPI device-                 SPI_CNFG
*                          2.               Set-Up A/D, Constants              SET_UP
*                          3.               Read A/D, store sample             ADCONV
*                          4.               Convert into in-Hg                 IN_HG
*                          5.               Determine FLAG conditionIN_HG
*                                  a.       Display error                      ERROR
*                                  b.       Continue Conversion                INRANGE
*                          6.               Convert hex to BCD formatBCDCONV
*                          7.               Convert LCD display digits         LOOKUP
*                          8.               Output via SPI to LCD              SPI2LCD

*        This process is continually repeated as the loop CONVERT
*        runs unconditionally through BRA (the BRANCH ALWAYS statement)
*        Repeats to step 3 indefinitely.
                  ORG      $C000                    * DESIGNATES START OF MEMORY MAP FOR USER CODE
                  LDX      #REGBASE         * Location of base register for indirect adr
                  BSR      SPI_CNFG         * Set-up SPI Module for data X-mit to LCD
                  BSR      SET_UP           * Power-Up A/D, initialize constants
CONVERT  BSR      ADCONV           * Calls subroutine to make an A/D conversion
                  BSR      DELAY                    * Delay routine to prevent LCD flickering
```

```
                BSR     IN_HG                   * Converts hex format to in of Hg

*       The value of FLAG passed from IN_HG is used to determine
*       If a range error has occurred.  The following logical
*       statements are used to either allow further conversion or jump
*       to a routine to display a range error message.

                LDAB    FLAG                    * Determines if an range Error has ocurred
                CMPB    #$80                    * If No Error detected (FLAG=$80) then
                BEQ     INRANGE         *   system will continue conversion process
                BSR     ERROR                   * If error occurs (FLAG<>80), branch to ERROR
                BRA     OUTPUT          * Branches to output ERROR code to display

*       No Error Detected, Conversion Process Continues

INRANGE JSR     BCDCONV         * Converts Hex Result to BCD
                JSR     LOOKUP          * Uses Look-Up Table for BCD-Decimal

OUTPUT  JSR     SPI2LCD         * Output transmission to LCD
                BRA     CONVERT         * Continually converts using Branch Always

*       Subroutine SPI_CNFG
*               Purpose is to initialize SPI for transmission
*               and clear the display before conversion.

SPI_CNFG BSET   PORTD,X #$20    * Set SPI SS Line High to prevent glitch
                LDAA    #$38                    * Initializing Data Direction for Port D
                STAA    DDRD,X          * Selecting SS, MOSI, SCK as outputs only

                LDAA    #$5D                    * Initialize SPI-Control Register
                STAA    SPCR,X          * selecting SPE,MSTR,CPOL,CPHA,CPRO

                LDAA    #$5                     * sets counter to X-mit 5 blank bytes
                STAA    COUNTER

                LDAA    SPSR,X          * Must read SPSR to clear SPIF Flag

                CLRA                            * Transmission of Blank Bytes to LCD

ERASELCD JSR    TRANSFER        * Calls subroutine to transmit
                DEC     COUNTER
                BNE     ERASELCD

                RTS

*       Subroutine SET_UP
*               Purpose is to initialize constants and to power-up A/D
*               and to initialize POFFSET used in conversion purposes.
SET_UP  LDAA    #$90                    * selects ADPU bit in OPTION register
                STAA    ADOPT,X         * Power-Up of A/D complete
                LDD     #$0131+$001A    * Initialize POFFSET
                STD     POFFSET         * POFFSET = 305 - 25 in hex
                LDAA    #$00                    * or Pmax + offset voltage (5 V)
                RTS

*       Subroutine DELAY
*               Purpose is to delay the conversion process
*               to minimize LCD flickering.

DELAY           LDA     #$FF                    * Loop for delay of display
OUTLOOP LDB     #$FF                    * Delay = clk/255*255
INLOOP  DECB
                BNE     INLOOP
                DECA
                BNE     OUTLOOP
                RTS

*       Subroutine ADCONV
*               Purpose is to read the A/D input, store the conversion into
*               SENSOUT. For conversion purposes later.
ADCONV  LDX     #REGBASE        * loads base register for indirect addressing
                LDAA    #$25
                STAA    ADCTL,X         * initializes A/D cont. register SCAN=1,MULT=0

WTCONV  BRCLR   ADCTL,X  #$80 WTCONV    * Wait for completion of conversion flag
                LDAB    ADR2,X          * Loads conversion result into Accumulator
                CLRA
                STD     SENSOUT         * Stores conversion as SENSOUT
                RTS
```

```
*        Subroutine IN_HG
*                 Purpose is to convert the measured pressure SENSOUT, into
*                 units of in-Hg, represented by a hex value of 305-150
*                 This represents the range 30.5 - 15.0 in-Hg
IN_HG             LDD      POFFSET          * Loads maximum offset for subtraction
                  SUBD     SENSOUT          * RESULT = POFFSET-SENSOUT in hex format
                  STD      RESULT           * Stores hex result for P, in Hg
                  CMPD     #305
                  BHI      TOHIGH

                  CMPD     #150
                  BLO      TOLOW

                  LDAB     #$80
                  STAB     FLAG
                  BRA      END_CONV

TOHIGH   LDAB     #$FF
                  STAB     FLAG
                  BRA      END_CONV

TOLOW             LDAB     #$00
                  STAB     FLAG
END_CONV RTS

*        Subroutine ERROR
*                          This subroutine sets the display digits to output
*                          an error message having detected an out of range
*                          measurement in the main program from FLAG

ERROR             LDAB     #$00                     * Initialize digits 1,4 to blanks
                  STAB     DIGIT1
                  STAB     DIGIT4

                  LDAB     FLAG                     * FLAG is used to determine
                  CMPB     #$00                     * if above or below range.
                  BNE      SET_HI           * If above range GOTO SET_HI

                  LDAB     #$0E                     * ELSE display LO on display
                  STAB     DIGIT2           * Set DIGIT2=L,DIGIT3=O
                  LDAB     #$7E
                  STAB     DIGIT3
                  BRA      END_ERR          * GOTO exit of subroutine

SET_HI   LDAB     #$37                      * Set DIGIT2=H,DIGIT3=1
                  STAB     DIGIT2
                  LDAB     #$30
                  STAB     DIGIT3

END_ERR  RTS

*        Subroutine BCDCONV
*                          Purpose is to convert ALTITUDE from hex to BCD
*                          uses standard HEX-BCD conversion scheme
*                          Divide HEX/10 store Remainder, swap Q & R, repeat
*                          process until remainder = 0.

BCDCONV  LDAA     #$00                      * Default Digits 2,3,4 to 0
                  STAA     DIGIT2
                  STAA     DIGIT3
                  STAA     DIGIT4
                  LDY      #DIGIT4          * Conversion starts with lowest digit
                  LDD      RESULT           * Load voltage to be converted
CONVLP   LDX      #$A                       * Divide hex digit by 10
                  IDIV                              * Quotient in X, Remainder in D
                  STAB     0,Y                      * stores 8 LSB's of remainder as BCD digit
                  DEY
                  CPX      #$0                      * Determines if last digit stored
                  XGDX                              * Exchanges remainder & quotient
                  BNE      CONVLP
                  LDX      #REGBASE         * Reloads BASE into main program
                  RTS

*        Subroutine LOOKUP
*                          Purpose is to implement a Look-Up conversion
*                          The BCD is used to index off of TABLE
*                          where the appropriate hex code to display
*                          that decimal digit is contained.
```

```
*                       DIGIT4,3,2 are converted only.

LOOKUP   LDX      #DIGIT1+4               * Counter starts at 5
TABLOOP  DEX                              * Start with Digit4
                  LDY      #TABLE         * Loads table base into Y-pointer
                  LDAB     0,X                    * Loads current digit into B
                  ABY                             * Adds to base to index off TABLE
                  LDAA     0,Y                    * Stores HEX segment result in A
                  STAA     0,X
                  CPX      #DIGIT2        * Loop condition complete, DIGIT2 Converted
                  BNE      TABLOOP

                  RTS

*        Subroutine SPI2LCD
*                       Purpose is to output digits to LCD via SPI
*                       The format for this is to send a start byte,
*                       four digits, and a stop byte. This system
*                       will have 3 significant digits: blank digit
*                       and three decimal digits.

*                                                 Sending LCD Start Byte

SPI2LCD  LDX      #REGBASE
                  LDAA     SPSR,X         * Reads to clear SPIF flag
                  LDAA     #$02                   * Byte, no colon, start bit
                  BSR      TRANSFER       * Transmit byte

*                                                 Initializing decimal point & blank digit
                  LDAA     DIGIT3         * Sets MSB for decimal pt.
                  ORA      #$80                   * after digit 3
                  STAA     DIGIT3

                  LDAA     #$00                   * Set 1st digit as blank
                  STAA     DIGIT1

*                                                 Sending four decimal digits

                  LDY      #DIGIT1        * Pointer set to send 4 bytes
DLOOP             LDAA     0,Y                    * Loads digit to be x-mitted
                  BSR      TRANSFER       * Transmit byte
                  INY                             * Branch until both bytes sent
                  CPY      #DIGIT4+1
                  BNE      DLOOP

*                                                 Sending LCD Stop Byte

                  LDAA     #$00                   * end byte requires all 0's
                  BSR      TRANSFER       * Transmit byte

                  RTS

*        Subroutine TRANSFER
*                 Purpose is to send data bits to SPI
*                 and wait for conversion complete flag bit to be set.

TRANSFER LDX      #REGBASE
                  BCLR     PORTD,X #$20   * Assert SS Line to start X-misssion
                  STAA     SPDR,X         * Load Data into Data Reg.,X-mit
XMIT              BRCLR    SPSR,X  #$80 XMIT * Wait for flag
                  BSET     PORTD,X #$20   * DISASSERT SS Line
                  LDAB     SPSR,X         * Read to Clear SPI Flag

                  RTS

*        Location for FCB memory for look-up table
*        There are 11 possible digits: blank, 0-9

TABLE             FCB      $7E,$30,$6D,$79,$33,$5B,$5F,$70,$7F,$73,$00
                  END
```

# NOTES

# NOTES

## *How to Reach Us:*


**Home Page:**
www.freescale.com

**Web Support:**
http://www.freescale.com/support

**USA/Europe or Locations Not Listed:**
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

**Europe, Middle East, and Africa:**
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

**Japan:**
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

**Asia/Pacific:**
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

***For Literature Requests Only:***
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com





AN1326
Rev. 3
11/2006