**MOTOROLA**
SEMICONDUCTOR APPLICATION NOTE

**Order this document
by AN1552/D**

**AN1552**

# MPX7100AP: The Sensor at the Heart of Solid-State Altimeter Applications

**By Chris Winkler and Jeff Baum
Sensor Systems Engineering Group
Motorola Semiconductor Products Sector
Phoenix, Arizona**

## INTRODUCTION

The advent of piezoresistive, bulk micromachined silicon pressure sensing devices has revolutionized sensor technology. These sensors have significantly lowered system costs and opened doors to sensing solutions never before considered possible. In addition, the integration of on–chip temperature compensation, calibration, and signal–conditioning circuitry (resulting in a microprocessor–compatible output) onto a single monolithic sensing device has eliminated many of the cost and reliability issues encountered with systems incorporating electromechanical sensors. Now, add these developments to the decreasing cost of microcontrollers (MCU) for embedded control systems, and the sensing solution possibilities seem endless. This document outlines a system approach that is designed to measure and display altitude over a specified range. Although this system can be implemented equally well with an MPX2100AP, we have chosen to design our system around the low–power version of the MPX2100AP—the MPX7100AP. There are many altimeter applications where it is desirable to run the electronic system from battery power. In such cases, the MPX7100AP has an additional benefit—it has an input impedance that is approximately five times greater than its MPX2000 series counterpart. Thus, the power consumption will be five times less than that of the MPX2100AP.

The system presented here demonstrates how the Motorola MPX7100AP (100 kPa absolute ported high–impedance) pressure transducer may be utilized for altimeter applications. This simple design serves as a building block from which more sophisticated systems can be developed. Additional circuitry, software routines, and higher precision component selection can allow for increased accuracy, additional resolution, and a variety of altimeter system features (including variometers, travel memory, audio warning signals, barographs, and maximum/minimum tracking). Table 1 is a representation of the system in terms of its general operating specifications.

**Table 1. General System Specifications**

| Display Units | Altitude in ft. |
|---|---|
| Resolution (Maximum) | 80 ft. |
| Temperature Range | –10 to 40°C |
| Altitude Range | 0 – 9999 ft. |

## ALTITUDE MEASUREMENT

An altimeter is a device that converts absolute atmospheric pressure measurements into an altitude approximation. This absolute pressure is typically measured with respect to absolute zero pressure (i.e., a vacuum–sealed reference). Atmospheric pressure is inversely related to altitude; therefore, lower atmospheric pressure is encountered at higher altitudes. However, the pressure at a particular altitude fluctuates significantly as a result of changing weather conditions. Consequently, a calibration scheme must be frequently employed to control the accuracy of the approximation.

For these reasons, the most crucial device in this system is the silicon pressure sensor. This device produces an analog voltage signal (mV range) for an applied atmospheric pressure. The pressure corresponding to the sensor output signal is mathematically converted into altitude via a microcontroller. The pressure transducer used in this system is sensitive enough to detect pressure variations corresponding to altitude changes of one foot. Since the sensor output signal is typically less than 50 mV, it must be amplified and signal–conditioned for proper interfacing to a microcontroller.

The system resolution is determined by the resolution of the analog–to–digital (A/D) converter. The overall accuracy is governed by the transducer accuracy, the nonlinear relationship between pressure and altitude, and the signal–conditioning circuitry. Hence, there are independent factors that determine system accuracy and resolution. Accuracy, resolution, and range cannot be optimized simultaneously. The design presented here is intended to display altitudes of up to 9,999 feet accurately, with a maximum resolution of 80 feet.

For discussion purposes, the standardized absolute atmospheric pressure at sea level is designated as 29.92 inches of mercury (″ Hg). The use of a standard pressure scale allows for reference between pressure and altitude under normalized climate conditions at a particular altitude. This standardized pressure decreases to 20.57″ Hg at an altitude of approximately 10,000 feet. As mentioned before, the atmospheric pressure can differ by ± 1″ Hg due to climate changes. The standard pressures at several altitudes are displayed in Table 2. Over the intended operating altitude range, the absolute atmospheric pressure is expected to fluctuate from approximately 30.5″ Hg to a minimum of 20″ Hg. This information is critical to the overall design.

**Table 2. Altitude vs. Pressure Data**

| Altitude, ft. | Pressure, ″ Hg |
|---|---|
| 0 | 29.92 |
| 500 | 29.38 |
| 1,000 | 28.85 |
| 6,000 | 23.97 |
| 10,000 | 20.57 |
| 15,000 | 16.86 |

## DESIGN OVERVIEW

The remaining sections of this text describe the system design. The design is separated into subsystems as depicted in Figure 1. Each module accomplishes a specific task critical to the altitude translation process. The encountered atmospheric pressure is first converted into a differential voltage signal. This signal is then conditioned for compatibility with the analog–to–digital converter inputs of the microcontroller. A digital representation of this analog signal is mathematically converted into a pressure measurement. The microcontroller then implements a “look–up” procedure to convert this measurement into a corresponding altitude approximation. Finally, the appropriate data is transmitted serially to the LCD interface to display the altitude in feet. A discussion of each subsystem details the important design considerations.

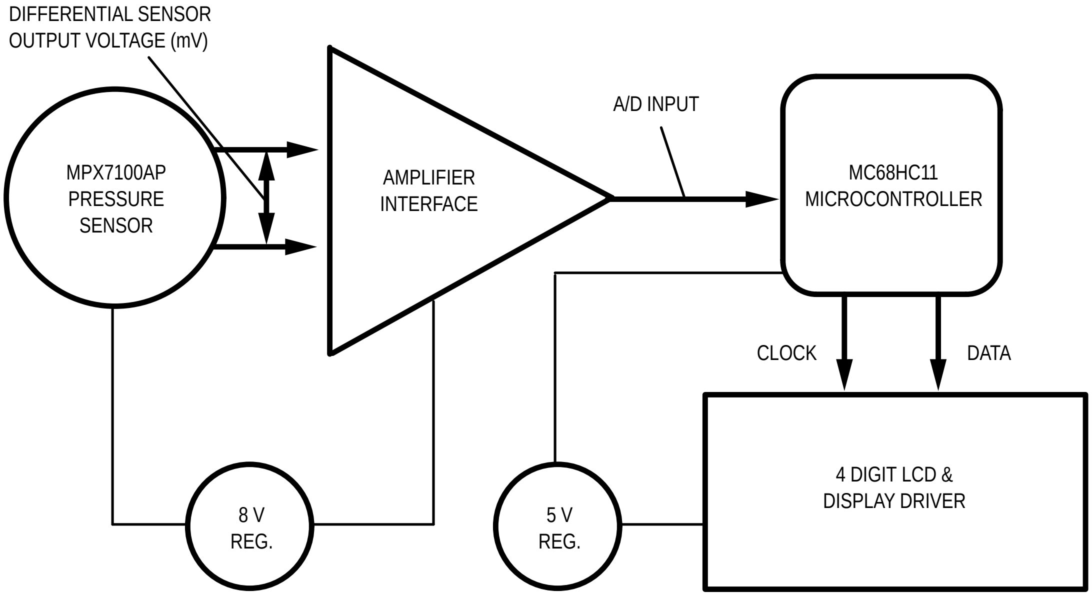

**Figure 1. Altimeter Block Diagram**

## PRESSURE SENSOR

The most critical element of this design is the pressure transducer. This device typically produces a differential output voltage linearly proportional to an applied pressure. The linearity and temperature stability of the sensor output signal over the intended operating conditions must not introduce significant errors. Otherwise, additional circuitry or software algorithms must be incorporated to compensate for the error created by the sensor. As indicated in the background information provided earlier, the output signal of the sensor must have a linear response function over the expected pressure range of 20–30.5″ Hg. Therefore, the required full–scale pressure rating for this device is approximately 30″ Hg (100 kPa). In order to measure absolute atmospheric pressure variations, the device must measure applied pressures with respect to a known, fixed pressure reference. Absolute zero (vacuum) is the easiest fixed reference to provide, with respect to manufacturability, and exhibits the best stability over temperature variations.

Based on these criteria, the design employs a Motorola MPX7100AP silicon pressure transducer. The MPX7000 series pressure transducers include on–chip temperature compensation and calibration circuitry. The zero–pressure offset voltage and full–scale span are trimmed to specified tolerances. These devices are available in full–scale pressure ranges from 10 kPa (1.5 PSI) to 700 kPa (100 PSI). To allow for use in different applications, the sensors are available in several configurations (gauge, differential, and absolute sensing) along with a variety of packaging/porting options. The MPX7100AP is an absolute sensing device that is designed to measure pressure applied to a single port with respect to absolute zero pressure sealed in an evacuated reference cavity. This feature of the transducer is critical to the design, as absolute barometric pressure must be measured with reference to a fixed pressure that is immune to ambient pressure fluctuations. The electrical characteristics for this device are summarized in Table 3. As suggested in the introduction, the MPX2100AP can provide the same functionality as the MPX7100AP. The major parametric difference between devices is that the MPX7100 is better suited for portable applications, as a result of its five times greater input resistance.

The sensor supply voltage can be varied as indicated in Table 3. The full–scale output signal, nominally rated at 40 mV for a supply voltage of 10 Vdc, is linearly proportional to the sensor supply voltage. In this system, all nondigital circuit elements are operated with an 8–Vdc regulated supply. At this supply voltage, the sensor will produce a differential output signal of 32 mV at the full–scale pressure of 100 kPa.

Zero–pressure offset and full–scale span variations for this device are trimmed to the tolerances shown in Table 3. To simplify the design, no additional circuitry or software is included to compensate or calibrate the sensor output signal beyond the tolerances shown. The combined effects of temperature variations and nonlinearity introduce a maximum error of 4% into the sensor output signal. Considering the linear response of the sensor and the operating pressure range, the sensor output signal is calculated to change from 32 mV to 16.2 mV over the entire altitude range. These nominal values are subject to the tolerances previously discussed.

## SIGNAL–CONDITIONING CIRCUITRY

The low–level differential output signal of the pressure sensor must be amplified to an analog voltage level compatible with the A/D inputs of the microcontroller. An instrumentation amplifier circuit converts the sensor output to a ground–referenced, 0.5 to 4.5 Vdc analog voltage. The dual operational amplifier circuit shown in Figure 2 provides the amplification, level–shifting, and stability to produce the desired analog signal. The MC33272 dual operational amplifier IC provides high output stability, low input offset voltage, low offset voltage thermal drift, and single–ended supply capability at low cost. An 8–volt regulator (MC78L08ACP) is included to limit supply voltage variations that can affect the accuracy of the circuit output.

This circuit will invert, amplify, and level–shift the differential input signal (difference between $S^+$ and $S^-$ sensor outputs). The transfer function for this circuit will have a linearly decreasing output for increasing differential input. Consequently, as altitude increases, the circuit output voltage will increase. This feature simplifies the mathematical processing in the software and eliminates the need for a negative supply voltage to provide the necessary level–shift of the amplified offset pedestal reference. The offset voltage is trimmable through resistor $R_{offset1}$ to obtain a 0.5–V output at sea level. This feature allows for system calibration at a known altitude reference.

**Table 3. MPX7100AP Operating Characteristics**
(Supply Voltage = 10 Vdc, TA=25°C unless otherwise noted)

| Characteristic | Symbol | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| Pressure Range | $P_{OP}$ | 0 | – | 100 | kPa |
| Supply Voltage | $V_S$ | – | 10 | 16 | Vdc |
| Supply Current | Io | – | 1.0 | – | mAdc |
| Full–Scale Span | $V_{FSS}$ | 38.5 | 40 | 41.5 | mV |
| Zero–Pressure Offset | $V_{off}$ | –2.0 | – | 2.0 | mV |
| Sensitivity | ΔV/ΔP | – | 0.4 | – | mV/kPa |
| Linearity | – | –1.0 | – | 1.0 | %$V_{FSS}$ |
| Temperature Effect on Span | $TCV_{FSS}$ | –1.0 | – | 1.0 | %$V_{FSS}$ |
| Temperature Effect on Offset | $TCV_{off}$ | –1.0 | – | 1.0 | mV |

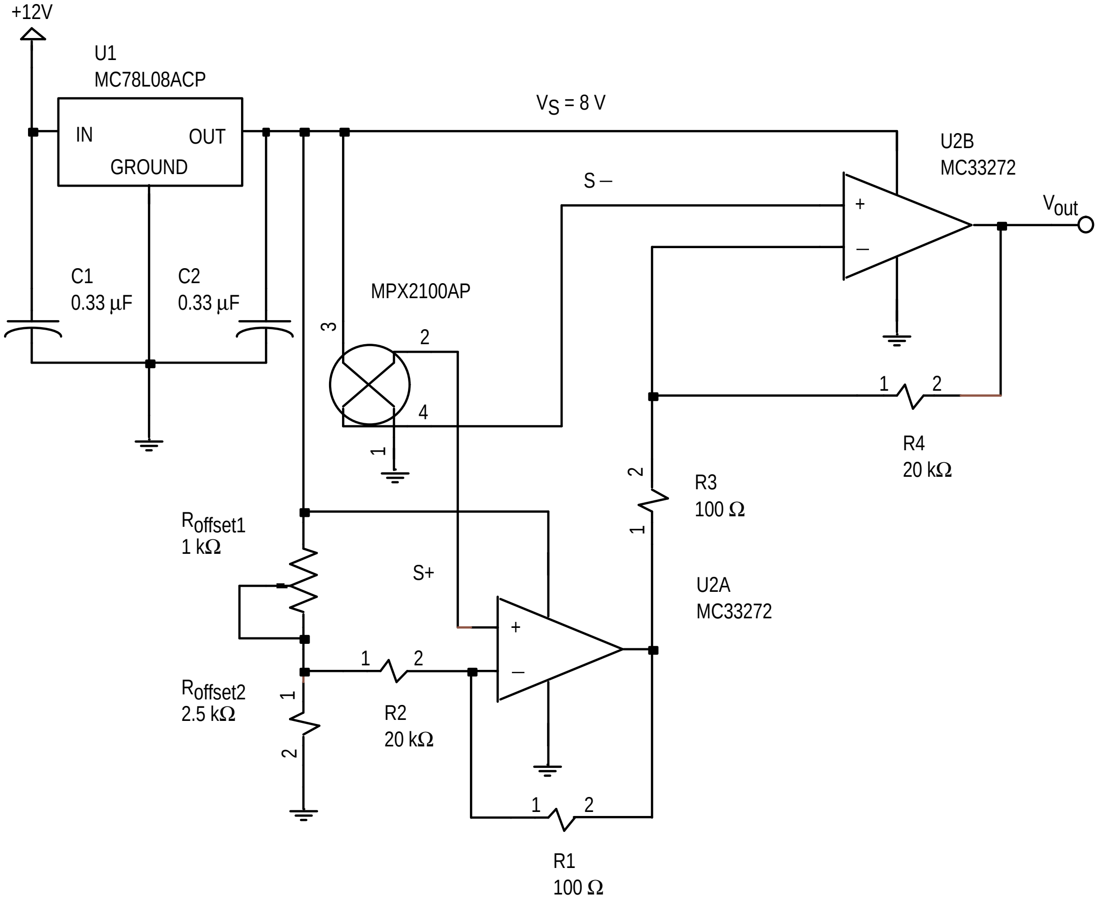

**Figure 2. Signal–Conditioning Circuit**

Using the transfer function for this circuit, the offset voltage and gain can be determined to provide maximum resolution and the desired output voltage. The calculation of these parameters for this circuit is illustrated as follows:

In determining the amplifier gain, the maximum offset voltage across $R_{offset2}$ is selected as approximately 7.0 Vdc. This value allows some room for adjustment and maximum signal gain. The amplified sensor signal at sea level cannot contribute more than the theoretical value of –6.5 Vdc, while allowing for the desired circuit output of 0.5 V.

Sensor Output at Sea Level (30.5″ Hg) ≈ 32.4 mV
Required Gain = –6.5 V / 32.4 mV ≈ –201
Sensor Output at 9,999 ft. (20.5 ″ Hg) ≈ 22.2 mV
Amplifier Output = (22.2 mV × –201) + 7.0 V = 2.53 V

The span voltage is approximately 2.0 V. The 8–bit (255 steps) resolution provided by the A/D converter on–board the MCU will detect 19.6 mV of change per step. The number of steps used in this process is determinable.

VRH = 5.0 V, VRL = 0.0 V

therefore, 5.0 V/255 steps ≈ 19.6 mV/step

Steps Required = 2.0 V/19.6 mV = 104 steps

The calculated gain is then used to determine the appropriate resistor values for the amplifier circuit defined by the transfer function shown in the following. The values for the offset resistors were selected to trim the offset voltage over the range of 8.0 – 6.6 Vdc.

$$Vout = -[R2/R1 + 1] \times V_{diff} + V_{off}\ ,\ V_{diff} = 32.4\ mV$$
$$R1 \approx R3 = 100\ \Omega\ ,\ R2 \approx R4 = 20\ k\Omega$$

Using 1% tolerance resistors, a common mode rejection ratio (CMRR) of approximately 75 dB can be obtained. The potentiometer is used to vary the offset voltage and allow for frequent calibration of the system as required to track the effects of changing weather conditions at a reference altitude.

## MICROCONTROLLER INTERFACE

Embedded microcontrollers are extremely useful devices in data acquisition systems. The decreasing costs have made widespread use feasible. The microcontroller employed in this system, the Motorola MC68HC11, is representative of a state–of–the–art 8–bit MCU. The MC68HC11 incorporates an 8–channel A/D with 8–bit resolution, a 16–bit enhanced timer (including a COP watchdog), an SPI (Serial Peripheral Interface: Synchronous), an SCI (Serial Communications Interface: Asynchronous), and a maximum of 40 parallel I/O lines. The HC11 is available in several package configurations and product variations allowing for additional RAM, EEPROM, and

I/O capability. The software was developed using the M68HC11EVB development system. This evaluation system is driven by the BUFFALO monitor program that can be operated from a host computer to develop and debug code.

In developing the system code, the following processing flow was used to outline the steps necessary for the conversion process. The system will convert the analog voltage into an 8–bit digital approximation, convert the pressure variation into altitude, and output this value to an LCD display interface serially (through the on–board SPI). Following is an outline of the process in greater detail:

1. Initialize and enable the A/D converter, SPI interface.
2. Initialize memory locations. Initialize variables.
3. Make A/D conversion of amplified sensor output. Store result.
4. "Look–Up" to determine altitude from pressure measurement.
5. Determine if conversion is in system range.
6. Convert pressure into decimal display digits or error message.
7. Output result via SPI to LCD interface.

The system is designed to display an error message if the conversion is out of the specified operating altitude range (0–9,999 ft.)   A "hi" is displayed if the conversion result is above 9,999 feet, and a "lo" is displayed for a conversion below sea level. A listing of the assembly source code to implement these tasks is included in the Appendix. The altitude "look–up" table was generated using a mathematical function which models the relationship between altitude and barometric pressure. The signal–conditioned sensor output signal is connected to pin PE5 (Port E–A/D Input pin). The HC11 communicates to the LCD display interface via the SPI synchronous clock signal SCLK and the serial data line MOSI (Master Out / Slave In).

## LCD DISPLAY INTERFACE

To display the altitude digitally in feet, a serial LCD interface was used to receive data from the HC11. The devices required are an MC145453 CMOS serial interface/LCD driver, and an FE202 four–digit LCD. To display the four decimal digits correctly, the HC11 serially transmits six bytes for each conversion. This includes a start byte, a byte for each of the four decimal display digits, and a stop byte. These decimal digits are used to transmit the numbers 0–9,999. The control of display digits and data transmission is executed in the source code through subroutines BCD_CONV, LOOKUP, SPI2LCD, and TRANSFER. A block diagram of this interface is included in Figure 3.

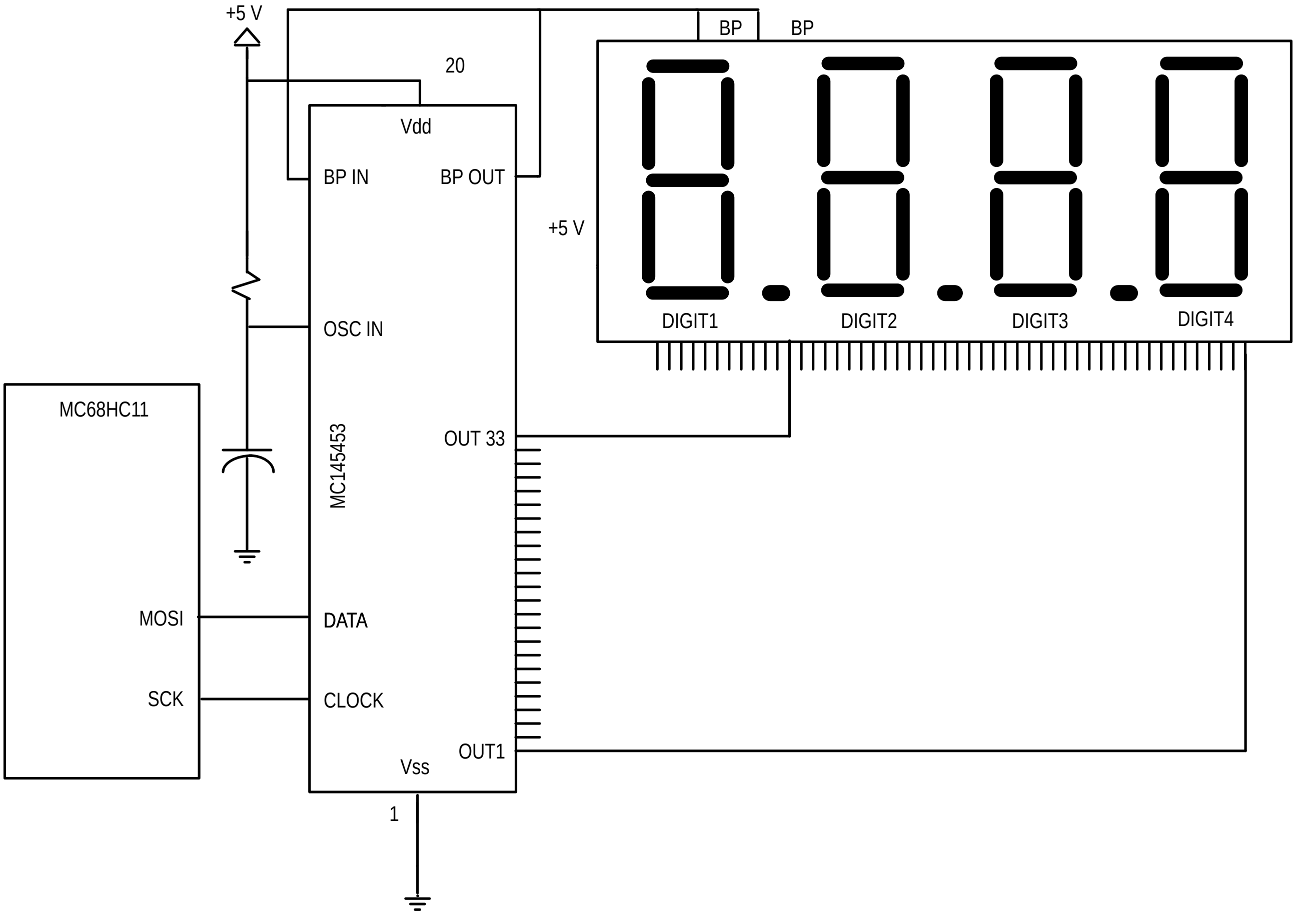

**Figure 3. LCD Display Interface Diagram**

## CONCLUSION

Silicon pressure sensors can provide the reliability and accuracy required in many applications at a relatively low cost. The altimeter design reviewed in this article is indicative of the system simplification achieved. Having outlined a simplified means of measuring altitude and the design issues involved, more sophisticated systems incorporating additional software and circuitry into this basic design are possible. The MPX7100AP, compensated and calibrated pressure sensor, is well suited for achieving the desired accuracy and design simplicity for common altimeter applications. Besides the integration of laser–trimmed resistive networks to tighten the device parameter tolerances, the MPX7100AP also offers the power–reducing advantage of low current–drain (due to its high input resistance). These features, combined with the absolute pressure reference sealed within the device's micromachined cavity, make the MPX7100AP an ideal choice for developing low–power, low–cost altimeter systems.

# APPENDIX
## Altimeter Software Listing

```
*ALTIMETER APPLICATIONS PROJECT – Chris Winkler
*Developed: October 1, 1992 – Motorola Discrete Applications
*This code is used to implement an MC68HC11 Micro–Controller
*as master processing unit for a simple Altimeter system.
*The HC11 will interface with an MPX7100AP to monitor, store,
*and display measured altitude via the 8–bit A/D channel
*The sensor output (32 mV max) will be amplified to 0.5 – 2.5 V dc
*The HC11 will interface with a 4–digit LCD (FE202) via
*a Motorola LCD driver (MC145453) to display the 4–digit
*altitude through the SPI.

*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 assignment of control register
ADCTL        EQU         $30           * offset of A/D conversion 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 thousands digit for altitude
DIGIT2       EQU         $0002         * BCD hundreds digit for altitude
DIGIT3       EQU         $0003         * BCD tens digit for altitude
DIGIT4       EQU         $0004         * BCD ones digit for altitude
COUNTER      EQU         $0005         * Variable to send 5 dummy bytes
ZEROALT      EQU         $0010         * Storage for 0 altitude in hex
MAXALT       EQU         $0011         * Storage for system’s maximum altitude
PRESSURE     EQU         $0012         * Storage location for A/D conversion
DELTA_P      EQU         $0014         * Storage of pressure change in hex format
INDEX        EQU         $0016         * Index to retrieve entire word from table
ALTITUDE     EQU         $0018         * Storage of altitude in hex format
FLAG         EQU         $0020         * Determination of conversion 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.          Delay conversion to LCD               DELAY
*4.          Read A/D, detect error                ADCONV
*5           Error detected – Code                 ERROR
*6.          No Error – Convert Altitude           ALT_CONV
*7.          Convert hex to BCD format             BCDCONV
*8.          Convert LCD display digits            LOOKUP
*9.          Output via SPI to LCD                 SPI2LCD

*This process is continually repeated as the loop CONVERT
*runs unconditionally through BRA (the BRANCH ALWAYS statement)

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         DELAY         * Slow down conversion process for display
             BSR         ADCONV        * Calls subroutine to make A/D conversion
             LDAB        FLAG          * Determines if a range Error has occurred
             CMPB        #$80          * If No Error detected (FLAG=$80) then
             BEQ         INRANGE       * system will continue conversion process
             BSR         ERROR         * Upon detecting Error, branches to ERROR
             BRA         OUTPUT        * Branches to output ERROR code
```

```
*Continues conversion if pressure is in system’s range

INRANGE      JSR         ALT_CONV      * Convert pressure to altitude in hex
             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, LCD interface for transmission

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 initialize pressures for min. and max. altitudes.

SET_UP       LDAA        #$90          * selects ADPU bit in OPTION register
             STAA        ADOPT,X       * Power–Up of A/D complete
             LDAB        #$0019        * Initialize Hex value of sea level
             STAB        ZEROALT       * Pressure (altitude = 0)
             LDAB        #$7D          * Initialize Hex value of highest altitude
             STAB        MAXALT        * Pressure (altitude>9999ft)
                         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 implement A/D conversion, store the value
*as PRESSURE, and return a value for FLAG to determine
*if this measurement is above or below the system’s range.

ADCONV       LDX         #REGBASE      * Loads base register for indirect addressing
             LDAA        #$25
             STAA        ADCTL,X       * Initializes A/D Conv. Reg.SCAN=1,MULT=0
WTCONV       BRCLR ADCTL,X #$80 WTCONV * Wait for completion of conversion flag
             LDAB        ADR2,X        * Loads conversion result into Accumulator
             STAB        PRESSURE      * Stores A/D conversion as PRESSURE
             STAB        PORTB,X       * Aids in DEBUGGING purposes

*The purpose of the following code is to determine if the measurement
*is out of the range of the systems range.  The following logical
*statements will set FLAG to a particular value so that the main
*program can determine if the pressure applied is out of range.
*If the pressure is measured as below sea level, FLAG=$00
*If the pressure is measured as above 9999 ft, FLAG=$FF
*Otherwise flag is set to an inordinate value, FLAG=$80

             CMPB        ZEROALT       * Determines if conversion<sea level
             BLO         TOLOW         * Branches to set FLAG=$00
             CMPB        MAXALT        * Determines if conversion>9999 ft
             BHI         TOHIGH        * Branches to set FLAG=$FF

             LDAB        #$80          * Otherwise conversion is in range
             STAB        FLAG          * Set flag untrue FLAG=$80
             BRA         END_AD        * replaces GOTO RTS
```

```
TOLOW        LDAB        #$00          * Conversion out of lower range
             STAB        FLAG          * FLAG = $00
             BRA         END_AD

TOHIGH       LDAB        #$FF          * Conversion out of upper range
             STAB        FLAG          * FLAG = $FF
END_AD       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 ALT_CONV
*Purpose is to determine the pressure variation
*from the calibrated Sea Level pressure – ZEROALT
*The pressure variation,DELTA_P is then used
*as index of ALT_TABLE to look–up the altitude.

ALT_CONV     LDAB        PRESSURE      * Loads A/D sample of pressure
             SUBB        ZEROALT       * DELTA_P = PRESSURE – ALTZERO
             STAB        DELTA_P       * Stores hex result – pressure variation
             LDAA        #$02          * Altitudes are in word format
             MUL                       * Multiply B=DELTA_P by 2 (A) = INDEX
             STD         INDEX         * Allows for correct retrieval of entire word
             LDD         #ALT_TABLE    * Loads base of table
             ADDD        INDEX         * Adds INDEX to table base
             STD         INDEX         * INDEX acts as pointer for altitude
             LDY         INDEX         * loads correct location into Y–pointer
             LDD         0,Y           * uses index addr.to look–up altitude
             STD         ALTITUDE      * stores hex format of altitude into memory
             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        #$00          * Default Digits 1,2,3,4 to blanks
             STAA        DIGIT1
             STAA        DIGIT2
             STAA        DIGIT3
             STAA        DIGIT4
             LDY         #DIGIT4       * Conversion starts with lowest digit
             LDD         ALTITUDE      * Load HEX ALTITUDE to be converted
CONVLP       LDX         #$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        * returns if Quotient>0
             LDX         #REGBASE      * Reloads BASE into main program
                         RTS
```

```
*Subroutine LOOKUP
*Purpose is to implement a Look–Up conversion
*The BCD is used to implement off of TABLE
*where the appropriate hex code to display
*that digit is contained.
*DIGIT4,3,2 are converted only

LOOKUP       LDX         #DIGIT1+4     * Counter starts at 5
TABLOOP      DEX                       * Start with Digit4
             LDY         #LCDSEGS      * 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         #DIGIT1       * Loop condition complete, DIGIT1 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 display 4 decimal digits or error code.
*Sending LCD Start Byte

SPI2LCD      LDX         #REGBASE
             LDAA        SPSR,X        * Reads to clear SPIF flag
             LDAA        #$02          * 6 0’s, no colon, set start bit
             BSR         TRANSFER      * Transmit byte

*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 4 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–mission
             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 decimal conversion look–up table
*NOTE: 11 possible digits: blank, 0 – 9

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

```
*Location for FDB memory for altitude conversion look–up table
*Generated using Approximation Curve calculations simulated
*in EXCEL spreadsheet.

ALT_TABLE    FDB         0000
             FDB         0083
             FDB         0166
             FDB         0250
             FDB         0333
             FDB         0417
             FDB         0501
             FDB         0586
             FDB         0670
             FDB         0755
             FDB         0840
             FDB         0925
             FDB         1010
             FDB         1096
             FDB         1182
             FDB         1268
             FDB         1354
             FDB         1440
             FDB         1527
             FDB         1614
             FDB         1701
             FDB         1788
             FDB         1876
             FDB         1963
             FDB         2051
             FDB         2140
             FDB         2228
             FDB         2317
             FDB         2406
             FDB         2495
             FDB         2584
             FDB         2674
             FDB         2764
             FDB         2854
             FDB         2944
             FDB         3035
             FDB         3126
             FDB         3217
             FDB         3308
             FDB         3400
             FDB         3492
             FDB         3584
             FDB         3676
             FDB         3769
             FDB         3862
             FDB         3955
             FDB         4049
             FDB         4142
             FDB         4236
             FDB         4330
             FDB         4425
             FDB         4520
             FDB         4615
             FDB         4710
             FDB         4806
             FDB         4901
             FDB         4998
             FDB         5094
             FDB         5191
             FDB         5288
             FDB         5385
             FDB         5483
             FDB         5580
             FDB         5679
             FDB         5777
             FDB         5876
             FDB         5975
             FDB         6074
             FDB         6174
             FDB         6274
             FDB         6374
             FDB         6474
             FDB         6575
             FDB         6677
```

```
                  FDB          6778
                  FDB          6880
                  FDB          6982
                  FDB          7084
                  FDB          7187
                  FDB          7290
                  FDB          7394
                  FDB          7498
                  FDB          7602
                  FDB          7706
                  FDB          7811
                  FDB          7916
                  FDB          8022
                  FDB          8127
                  FDB          8233
                  FDB          8340
                  FDB          8447
                  FDB          8554
                  FDB          8662
                  FDB          8770
                  FDB          8878
                  FDB          8987
                  FDB          9096
                  FDB          9205
                  FDB          9315
                  FDB          9425
                  FDB          9536

                  END
```

**How to reach us:**
**USA / EUROPE**: Motorola Literature Distribution;
P.O. Box 20912; Phoenix, Arizona 85036. 1–800–441–2447

**MFAX**: RMFAX0@email.sps.mot.com – TOUCHTONE (602) 244–6609
**INTERNET**: http://Design–NET.com

**JAPAN**: Nippon Motorola Ltd.; Tatsumi–SPD–JLDC, Toshikatsu Otsuki,
6F Seibu–Butsuryu–Center, 3–14–2 Tatsumi Koto–Ku, Tokyo 135, Japan. 03–3521–8315

**HONG KONG**: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

MOTOROLA