

Offset Calibration of Gauge Pressure Sensor Using Parallel I/O Ports

Prepared by: C.S. Chua
 Sensor Application Engineering
 Singapore, A/P

INTRODUCTION

This application note describes the concept of calibrating the offset of a gauge pressure sensor using the parallel i/o ports of a microcontroller which includes the hardware/software, a liquid crystal display and some switches. External stresses and mounting position can affect the zero pressure output reading of a gauge pressure sensor especially low

pressure sensor (high sensitivity), it is advisable to perform offset calibration after installation. In general, the offset value of the sensor can be stored in the ROM (read only memory), the EEPROM (electrical erasable programmable ROM) or the RAM (random access memory) of the microcontroller. The type of storage is restricted by the cost and applications of the system.

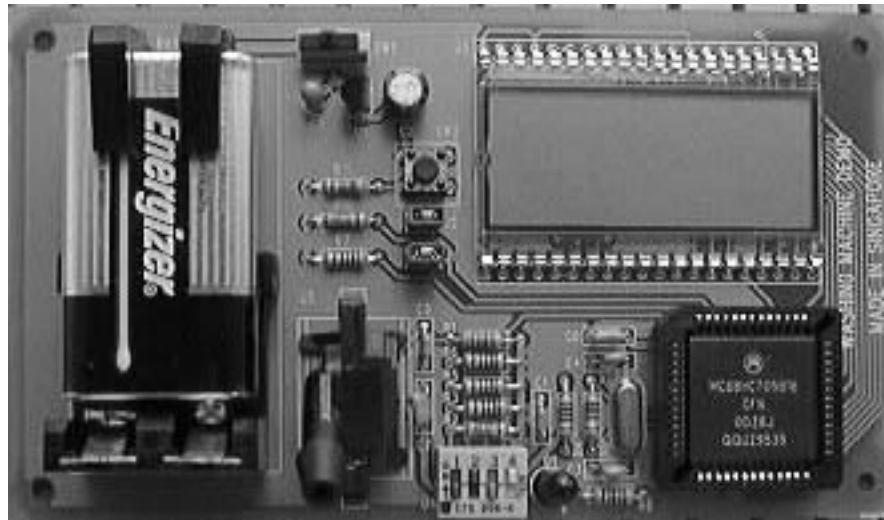


Figure 1. Offset Compensation Evaluation Board

CONCEPT OF OFFSET CALIBRATION USING PARALLEL I/O PORTS

In order to achieve better accuracy in a gauge pressure sensor, the output of the sensor is sampled at zero pressure (atmospheric pressure). This offset value will then be used to derive the actual pressure reading. Of all the methods used, the easiest way is to store the nominal offset value (given by the supplier) into the ROM. In other words, this value cannot be changed after the program code is implemented. Hence, the accuracy is affected by the part-to-part offset variation and temperature coefficient of offset (TCO).

The second method calibrates the offset value at the production and stores it in the EEPROM of the microcontroller. It eliminates the part-to-part offset variation. However, this method becomes expensive if the original system does not have any EEPROM.

The third method calibrates the offset value every time the system is power-up. The offset is then saved into the RAM of the microcontroller. This method is simple, low cost, and it can also compensate for the part-to-part variation and TCO of the sensor. It is an ideal method for offset calibration but it may not be applicable in certain systems. For instance, in a digital blood pressure monitor, the system experiences a zero pressure condition before the user starts pumping the bulb. In this case, the system can execute a command to save the offset value into the RAM. However, in a washing machine, the system may or may not experience zero pressure during start-up. This may occur when some users may start pouring water into the tub (for top loading washing machine) even before they switch on the power. However, for front loading washing machine, the method of offset calibration is possible because it is impossible for the user to pour water into the drum.

REV 1

During offset calibration, 1-byte or 8-bit of the data is used to represent the offset value for a 8-bit analog-to-digital converter (ADC). Since the offset voltage is very close to the lower rail of the supply, the 8-bit data string is not fully utilised i.e. only the lower 4 to 5 bits are used. The upper bits are all zeros.

Let's take the MPXT5006D as an example. The offset value ranges from a minimum of 0.1 V to a maximum of 0.43 V with the nominal value at 0.225 V. Hence, the variation of offset from the nominal value is -0.125 V and +0.205 V. Therefore, the offset is varied from the nominal value by -6 steps to +10 steps based on the 19.6 mV/step of ADC. In this application note, we defined this variation by a 4-bit data which is shown the Figure 2.

By closing the switch, the corresponding i/o pin will indicate a logic "1" and vice-versa. For a cheaper solution, the four switches are replaced by 4 jumper wires which represents "1111". To store a logical "0" during offset calibration, the corresponding wire will be removed or cut.

$$\text{Offset (steps)} = \text{Nominal} \pm \text{Data (3-bit)}$$

where Nominal = 11 steps
 Sign = plus if bit 4 is "0"
 Sign = minus if bit 4 is "1"

The DIL switch is "1011", offset = 11 - 3 = 9 steps or 0.176 V
 The DIL switch is "0100", offset = 11 + 4 = 15 steps or 0.294 V

Therefore, the higher the number of i/o pins used, the wider it can calibrate for the spread in sensor's offset. However, it has to depend on the availability of i/o ports in the system.

Besides performing offset calibration using i/o ports, the system also allows calibration through RAM and EEPROM. The type of storage mode for the offset value is determined by the two jumper settings i.e. J1 and J2.

	J1	J2
ROM	ON	ON
RAM	ON	OFF
EEPROM	OFF	ON
I/O Ports	OFF	OFF

The output of the integrated pressure sensor is ratiometric to the voltage applied to it. The sensor and the reference voltages are connected to a common supply; this yields a system that is ratiometric. By nature of this ratiometric system, variations in the voltage of the power supplied to the system will have no effect on the system accuracy.

The liquid crystal display (LCD) is directly driven from I/O ports A, B, and C on the microcontroller. The operation of a LCD requires that the data and backplane (BP) pins must be driven by an alternating signal. This function is provided by a software routine that toggles the data and backplane at approximately a 30 Hz rate. Other than the LCD, one light emitting diode (LED) is connected to the pulse length converter (PLM) of the microcontroller. This LED will light up whenever the pressure sensor's reading exceeded 600 mm of water level.

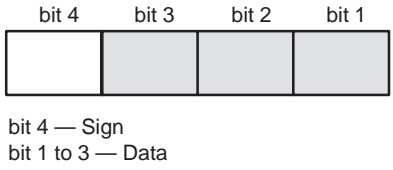


Figure 2. Data Format

In this way, the 4-bit data string can calibrate the offset value with variation of ±0.137 V from the nominal. This 4-bit data will be implemented using hardware (DIL switch is used in this case) through the parallel i/o ports of the microcontroller.

HARDWARE DESCRIPTION AND OPERATION

In this example, we use a 4-way DIL switch to represent the 4-bit data string. In an actual case, jumper wires should be used for lower cost. Figure 3 shows the interface between the DIL switch and microcontroller i/o ports. In this design, port D is used and is configured as standard input-only pins (can be read via port D data register) whenever the program reads the status of the DIL switch.

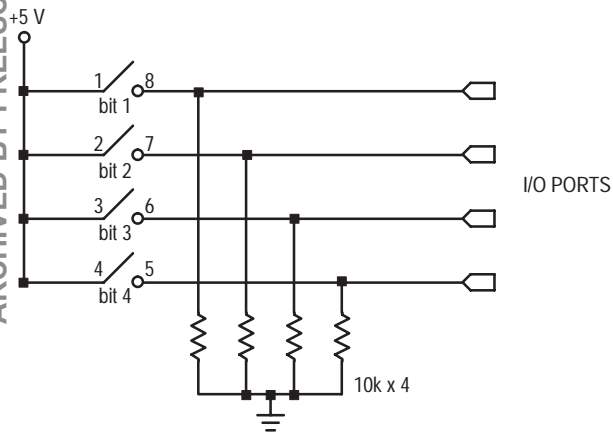


Figure 3. Interface of DIL Switch to Microcontroller

Freescale Semiconductor, Inc. ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005

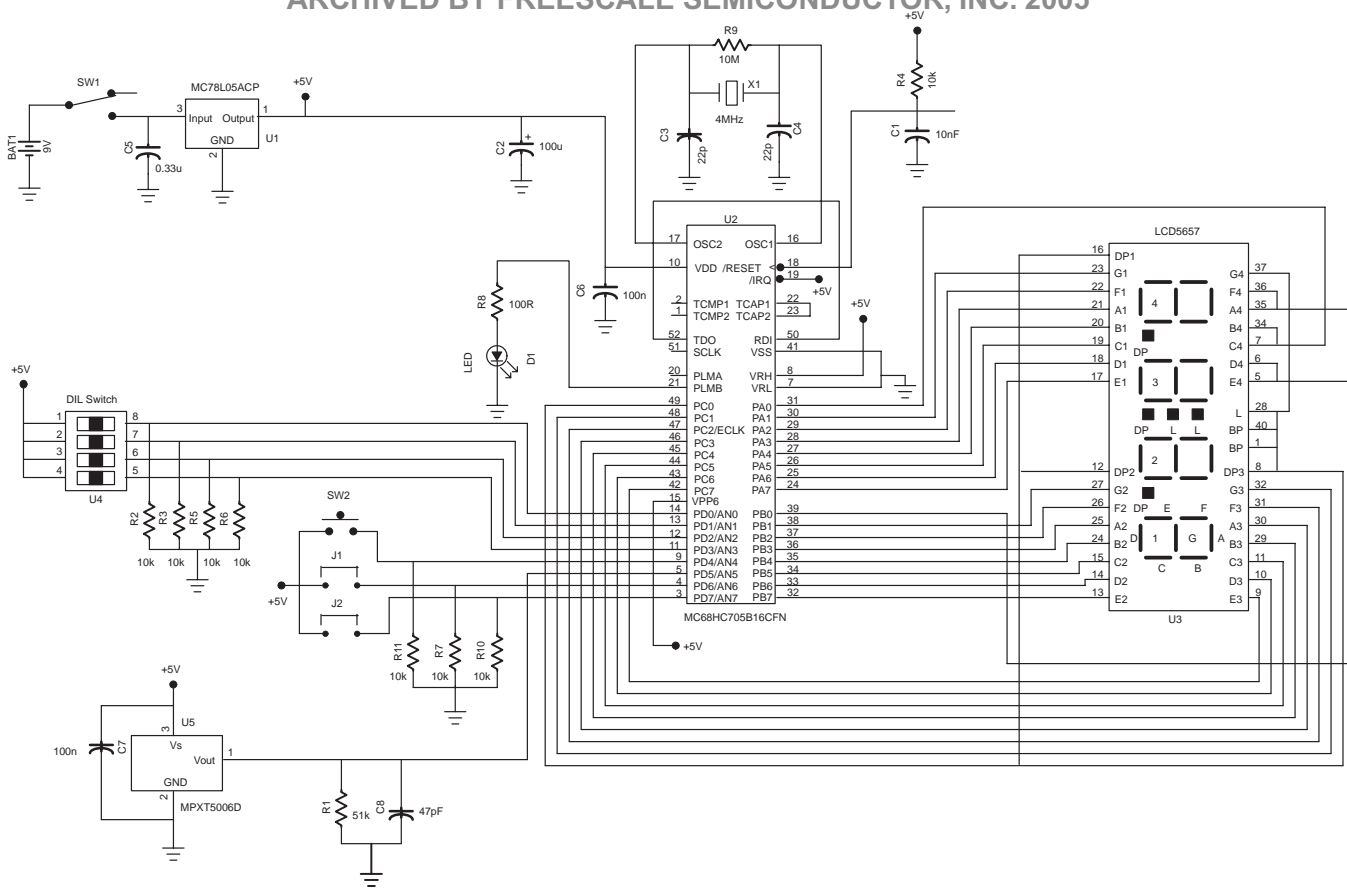


Figure 4. Offset Calibration Schematic Drawing

SOFTWARE DESCRIPTION

Upon power-up of the system, the microcontroller will sample the sensor's output and calculate the average value. Depending on the jumper settings, the calculation will retrieve the offset value from the respective location (RAM, ROM etc.) and convert the pressure to millimetre of water according to the equation below.

$$\text{Pressure (mmH}_2\text{O)} = (V_{\text{OUT}} - V_{\text{OFFSET}}) \times \text{Resolution}$$

where V_{OUT} and V_{OFFSET} is in number of steps
 Resolution is 2.6081 mm/step

This water level reading is then shown on the LCD. To calibrate the offset value, expose the sensor to atmospheric pressure and press the button SW2 after configured the

desired jumper settings. The LCD will display "CAL" for 2 seconds before the microcontroller starts to sample and calculate average value for zero offset voltage. This value will then be saved in the respective location depending of the jumper settings. One point to note is that the sensor should remain at zero pressure during the calibration process.

For calibration using i/o ports, the program will determine the required compensation and display a 4-bit data on the LCD for 2 seconds after "CAL". You will need to change the configuration of the DIL switch from number 1 to number 4 according to display on the LCD from bit 0 to bit 3 respectively.

The LCD will automatically switch back to display water level immediately after the calibration.

Figure 5 is a flowchart for the program that controls the system.

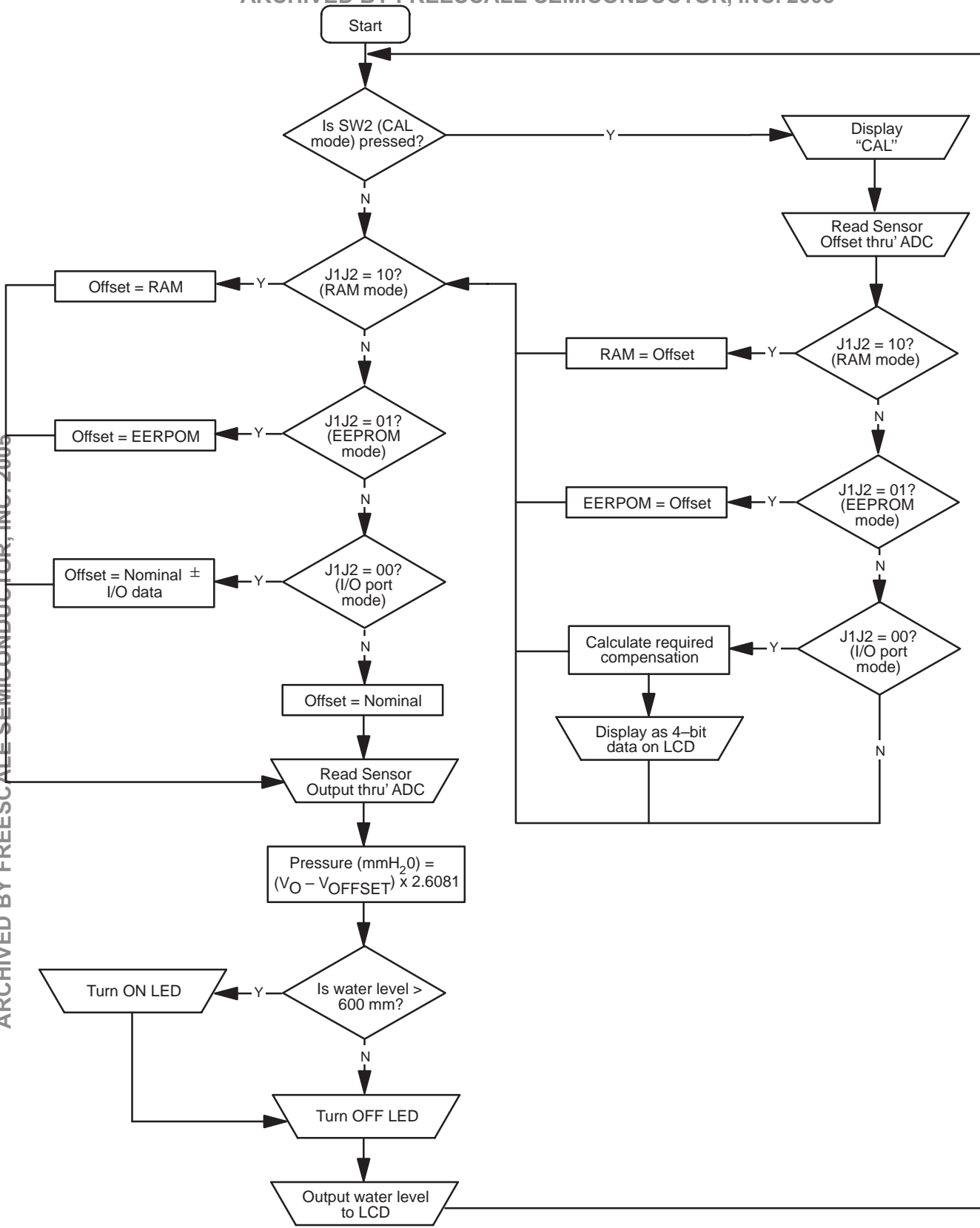


Figure 5. Main Program Flowchart



Offset calibration of a gauge pressure sensor can improve the accuracy of a system significantly. However, for a system that may not experience an atmospheric pressure during the start-up, the offset should be calibrated on the production floor. In this case, the offset value will have to be stored as non-volatile memory, either in the EEPROM or the i/o ports whichever is available.

1. Ador Reodique, "Implementing Auto Zero for Integrated Pressure Sensors", Motorola Application Note AN1636.
2. Bill Lucas, "An Evaluation System for Direct Interface of the MPX5100 Pressure Sensor with a Microprocessor", Motorola Application Note AN1305.

```
*****
*
* Offset Calibration Version 1.0
*
* The following code is written for MC68HC705B16 using MMDS05 software
* Version 1.01
* CASM05 - Command line assembler Version 3.04
* P & E Microcomputer Systems, Inc.
*
* Written by : C.S. Chua
* 21 October 1997
*
* Copyright Motorola Electronics Pte Ltd 1997
* All rights Reserved
*
*****
*
* Software Description
*
* This software is used to demonstrate offset calibration of low pressure
* sensor through RAM, ROM, EEPROM or i/o ports
* In addition, it displays the water level in millimetre.
*
*****
*
* Initialisation
*
*****
PORTA EQU $00 ; Last digit
PORTB EQU $01 ; Second digit (and negative sign)
PORTC EQU $02 ; First digit (and decimal point)
PORTD EQU $03 ; 7 digit inputs and 1 analog input
EESTAT EQU $07 ; EEPROM control register
ADDATA EQU $08 ; ADC Data
ADSTAT EQU $09 ; ADC Status
PLMB EQU $0B ; Pulse Length Modulator (Output to LED)
MISC EQU $0C ; Miscellaneous Register (slow/fast mode)
TCONTROL EQU $12 ; Timer control register
TSTATUS EQU $13 ; Timer Status Register
OCMPHI1 EQU $16 ; Output Compare Register 1 High Byte
OCMPL01 EQU $17 ; Output Compare Register 1 Low Byte
TCNTHI EQU $18 ; Timer Count Register High Byte
TCNTLO EQU $19 ; Timer Count Register Low Byte
OCMPHI2 EQU $1E ; Output Compare Register 2 High Byte
OCMPL02 EQU $1F ; Output Compare Register 2 Low Byte
*****
*
* User-defined RAM
*
*****
ORG $50
ADCOUNTER RMB 1 ; Sampling Counter
TEMPCNTHI RMB 1 ; Temp storage for Timer count register
TEMPCNTLO RMB 1 ; Temp storage for Timer count register
STACK RMB 5 ; Push acc and x-register during interrupt
RSHIFT RMB 1 ; No of shifting required for division
ADVALUE RMB 2 ; Analog voltage
OFFSET RMB 1 ; Offset value
RAM RMB 1 ; Offset stored in RAM
IO_OFFSET RMB 1 ; Offset stored in i/o
OFCNT RMB 1 ; Timer overflow counter
LCD_STATUS RMB 1 ; LCD status
NUMBER RMB 4 ; 4 byte number to be converted to decimal
; number
ADDEND RMB 4 ; Variables used in Addition subroutine
AUGEND RMB 4
SUM RMB 4
MINUE RMB 4 ; Variables used in Subtraction subroutine
SUBTRA RMB 4
DIFF RMB 4
MTEMP RMB 4 ; Variables used in Multiplication
MULCAN RMB 4 ; subroutine
MULTP RMB 4
CNT RMB 1 ; Variables used in the Division subroutine
DVSOR RMB 4
DVDND RMB 4
QUO RMB 4
```

Freescale Semiconductor, Inc. ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005



```

*****
*
*           User-defined EEPROM
*
*****
      ORG     $120
EEPROM   RMB     1           ; Offset stored in EEPROM
EEREG    EQU     $100       ; Options register
      ORG     $300       ; ROM space 0300 to 3DFE (15,104 bytes)
      DB     $FC           ; Display "0"
      DB     $30           ; Display "1"
      DB     $DA           ; Display "2"
      DB     $7A           ; Display "3"
      DB     $36           ; Display "4"
      DB     $6E           ; Display "5"
      DB     $EE           ; Display "6"
      DB     $38           ; Display "7"
      DB     $FE           ; Display "8"
      DB     $7E           ; Display "9"
NOMINAL  DB     !11       ; Offset nominal value
*****
*
*   Program starts here upon hard reset
*
*****
RESET    CLR     PORTC       ; Port C = 0
          CLR     PORTB       ; Port B = 0
          CLR     PORTA       ; Port A = 0
          LDA     #$FF
          STA     $06         ; Port C as output
          STA     $05         ; Port B as output
          STA     $04         ; Port A as output
          LDA     TSTATUS     ; Dummy read the timer status register
          CLR     OCMPHI2     ; so as to clear the OCF
          CLR     OCMPHI1
          LDA     OCMPLQ2
          JSR     COMPRGT
          LDA     #$60         ; Enable the output compare interrupt
          STA     TCONTROL
          CLI
          CLR     OFFSET     ; Interrupt begins here
          CLR     RAM
ALWAYS   BCLR    6,ADSTAT     ; Switch off ADC
          BCLR    5,ADSTAT
          BRCLR   4,PORTD,NO_CAL ; Check if SW2 is pressed
NO_CAL   JSR     CALIBRATE
          BCLR    6,ADSTAT     ; Switch off ADC
          BCLR    5,ADSTAT
          LDA     PORTD
          AND     %%11000000   ; Check J1 J2 status
          EOR     %%01000000
          BNE     MEMORY1
          LDA     RAM         ; Used RAM value for offset
          STA     OFFSET
MEMORY1  JMP     READ_SENSOR
          BCLR    6,ADSTAT     ; Switch off ADC
          BCLR    5,ADSTAT
          LDA     PORTD
          AND     %%11000000   ; Check J1 J2 status
          EOR     %%10000000
          BNE     MEMORY2
          LDA     EEPROM     ; Use EPROM value for offset
          STA     OFFSET
MEMORY2  JMP     READ_SENSOR
          BCLR    6,ADSTAT     ; Switch off ADC
          BCLR    5,ADSTAT
          LDA     PORTD
          AND     %%11000000   ; Check J1 J2 status
          EOR     %%00000000
          BNE     MEMORY3
          LDA     PORTD     ; Use i/o value for offset
          AND     %%00000111
          STA     OFFSET
          BRSET   3,PORTD,MINUS ; If DIL switch 4 is OFF
          LDA     NOMINAL     ; Offset = Nominal + Offset
          ADD     OFFSET
          STA     OFFSET
          JMP     READ_SENSOR
MINUS    LDA     NOMINAL     ; If DIL switch 4 is ON
          SUB     OFFSET     ; Offset = Nominal - Offset
          STA     OFFSET
          JMP     READ_SENSOR
MEMORY3  LDA     NOMINAL     ; Use ROM/nominal value
          STA     OFFSET     ; for offset
READ_SENSOR JSR    READAD    ; Read Vo of sensor
          LDA     ADVALUE+1

```

Freescale Semiconductor, Inc. ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005

Freescale Semiconductor, Inc.
 ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005

```

CMP      OFFSET
BHS     POSITIVE
LDA     OFFSET                ; Offset - Vo
SUB     ADVALUE+1
BSET    3,LCD_STATUS        ; Turn ON negative sign
BRA     EQUATION
POSITIVE
SUB     OFFSET                ; Vo - Offset
BCLR    3,LCD_STATUS        ; Turn OFF negative sign
EQUATION
STA     MULTP+3
CLRA
STA     MULTP+2
STA     MULTP+1
STA     MULTP
LDA     #$E1
STA     MULCAN+3
LDA     #$65
STA     MULCAN+2
CLRA
STA     MULCAN+1
STA     MULCAN
JSR     CALMUL                ; (Vo - Offset)*2.6081
LDA     MULCAN+3
STA     ADDEND+3
LDA     MULCAN+2
STA     ADDEND+2
LDA     MULCAN+1
STA     ADDEND+1
LDA     MULCAN
STA     ADDEND
LDA     #$88                ; Add 0.5000
STA     AUGEND+3
LDA     #$13
STA     AUGEND+2
CLRA
STA     AUGEND+1
STA     AUGEND
JSR     CALADD
LDA     SUM+3
STA     DVDND+3
LDA     SUM+2
STA     DVDND+2
LDA     SUM+1
STA     DVDND+1
LDA     SUM
STA     DVDND
LDA     #$10
STA     DVSOR+3
LDA     #$27
STA     DVSOR+2
CLRA
STA     DVSOR+1
STA     DVSOR
JSR     CALDIV                ; / 10000
LDA     OFCNT                ; No refresh of LCD if the
CMP     #11                  ; time elapse is too short
BLS     NO_REFRESH
CLR     OFCNT
LDA     QUO+2                ; Check if level > 600 mm
CMP     #$02
BHS     MAX_LIMIT
BRA     IN_LIMIT
MAX_LIMIT
LDA     QUO+3
CMP     #$58
BLS     IN_LIMIT
LDA     #$80                ; LED ON if limit is exceeded
STA     PLMB
BRA     SHOW_LEVEL
IN_LIMIT
CLRA                ; LED OFF if within limit
SHOW_LEVEL
STA     PLMB
LDA     QUO+3                ; Display water level
STA     NUMBER+3
LDA     QUO+2
STA     NUMBER+2
LDA     QUO+1
STA     NUMBER+1
LDA     QUO
STA     NUMBER
JSR     ADTOLCD
NO_REFRESH
JMP     ALWAYS
*****
*
*      Offset Calibration      *
*
*      Output: Offset          *
*
*****
    
```



```

CALIBRATE   LDA   #\$CC           ; Port C = 1100 1100   Letter "C"
            STA   PORTC
            LDA   #\$BE           ; Port B = 1011 1110   Letter "A"
            STA   PORTB
            LDA   #\$C4           ; Port A = 1100 0100   Letter "L"
            STA   PORTA
            LDA   #16
IDLE        JSR   DLY20           ; Idling for a while (16*0.125 = 2 sec)
            DECA          ; for the zero offset to stabilize
            BNE   IDLE           ; before perform auto-zero
            JSR   READAD
            BCLR  6,ADSTAT       ; Switch off ADC
            BCLR  5,ADSTAT
            LDA   PORTD
            AND   #11000000      ; Check J1 J2 status
            EOR   #01000000
            BNE   NON_RAM
            LDA   ADVALUE+1      ; Save the offset into RAM
            STA   RAM
            BRA   END_CAL
NON_RAM     LDA   PORTD
            AND   #11000000      ; Check J1 J2 status
            EOR   #10000000
            BNE   NON_EPROM
            JSR   LOAD_EPROM     ; Call EPROM subroutine
            BRA   END_CAL
NON_EPROM   LDA   PORTD
            AND   #11000000      ; Check J1 J2 status
            EOR   #00000000
            BNE   END_CAL
            JSR   IO_CAL         ; Call i/o CAL subroutine
END_CAL     RTS
*****
*
*           I/O CAL Subroutine
*
*****
IO_CAL      SEI
            CLRA
            STA   PORTA
            STA   PORTB
            STA   PORTC
            LDA   ADVALUE+1      ; If Voffset >= nominal
            CMP   NOMINAL
            BLO  NEGATIVE
            SUB  NOMINAL         ; Voffset - nominal
            STA   IO_OFFSET
            BRA  SHOW_DIL
NEGATIVE    LDA   NOMINAL        ; If Voffset < nominal
            SUB  ADVALUE+1       ; Nominal - Voffset
            STA   IO_OFFSET
SHOW_DIL    BSET  3,IO_OFFSET     ; Nominal > Voffset
            BRCLR 0,IO_OFFSET,BIT1_ZERO
            LDA   \$301
            STA   PORTA           ; 1st digit shows "1"
            BRA  BIT2
BIT1_ZERO   LDA   \$300           ; 1st digit shows "0"
            STA   PORTA
            BRCLR 1,IO_OFFSET,BIT2_ZERO
            LDA   \$301
            STA   PORTB           ; 2nd digit shows "1"
            BRA  BIT3
BIT2_ZERO   LDA   \$300           ; 2nd digit shows "0"
            STA   PORTB
            BRCLR 2,IO_OFFSET,BIT3_ZERO
            LDA   \$301
            STA   PORTC           ; 3rd digit shows "1"
            BRA  BIT4
BIT3_ZERO   LDA   \$300           ; 3rd digit shows "0"
            STA   PORTC
            BRCLR 3,IO_OFFSET,BIT4_ZERO
            BSET  0,PORTA         ; 4th digit shows "1"
            BRA  END_IO_CAL
BIT4_ZERO   BSET  0,PORTA         ; 4th digit shows "0"
            BSET  0,PORTB
END_IO_CAL  CLI
            LDA   #16
IDLE1      JSR   DLY20           ; Idling for a while (16*0.125 = 2 sec)
            DECA          ; for the user to read the LCD
            BNE   IDLE1         ; before switching the DIL SW
            RTS
*****
*
*           Save Offset into EEPROM
*
*****

```

```

LOAD_EPROM  CLRA
             STA  EESTAT      ; Erase EEPROM
             BSET 1,EESTAT    ; Set E1LAT bit
             BSET 2,EESTAT    ; Set E1ERA bit
             STA  EEPROM      ; Save data to location to be erased
             BSET 0,EESTAT    ; Set E1PGM bit
             JSR  DLY20       ; Wait for T(ERAl)
             BCLR 1,EESTAT    ; Reset E1LAT

             CLRA
             STA  EESTAT
             BSET 1,EESTAT
             LDA  ADVALUE+1   ; Load value into EEPROM
             STA  EEPROM
             BSET 0,EESTAT
             JSR  DLY20
             BCLR 1,EESTAT
             CLRA
             STA  EESTAT      ; Enable read operation
             RTS

*****
*
*      Delay Subroutine
*      (162 * 0.7725 ms = 0.125 sec)
*
*****
DLY20      STA  STACK
           STX  STACK+1
           LDA  #!162        ; 1 unit = 0.7725 ms
           CLRX
           INNRLP
           DECA
           BNE  OUTLP
           LDX  STACK+1
           LDA  STACK
           RTS

*****
*
*      Reading the ADC data 128 times
*      and take the average
*
*      Output: ADVALUE (Average ADC data)
*
*****
ENDREAD   LDA  #$25          ; Enable PD5 as ADC
           STA  ADSTAT
           CLR  ADVALUE      ; Clear the memory
           CLR  ADVALUE+1
           CLR  ADCOUNTER
           CLR  RSHIFT
LOOP128  LDA  ADCOUNTER
           CMP  #$80         ; If ADCOUNTER >= 128
           BHS DIVIDE       ; Branch to DIVIDE
ENDREAD  BRCLR 7,ADSTAT,ENDREAD ; Halt here till AD read is finished
           LDA  ADDATA
           ADD  ADVALUE+1    ; ADVALUE = ADVALUE + ADDATA
           STA  ADVALUE+1
           LDA  ADVALUE
           ADC  #$00
           STA  ADVALUE
           INC  ADCOUNTER    ; Increase the AD counter by 1
           BRA  LOOP128     ; Branch to Loop128
ENDDIVIDE INC  RSHIFT       ; Increase the right counter
           LSR  ADVALUE     ; Right shift the high byte
           ROR  ADVALUE+1   ; Right shift the low byte
           LDA  RSHIFT
           CMP  #$07        ; if the right shift counter >= 7
           BHS ENDDIVIDE   ; End the shifting
           JMP  DIVIDE      ; otherwise continue the shifting
ENDDIVIDE RTS

*****
*
*      Timer service interrupt
*      Alternates the Port data and
*      backplane of LCD
*
*****
TIMERCOMP STA  STACK+2      ; Push Accumulator
           COM  PORTC        ; Port C = - (Port C)
           COM  PORTB        ; Port B = - (Port B)
           COM  PORTA        ; Port A = - (Port A)
           BSR  COMPREGT     ; Branch to subroutine compare register
           LDA  STACK+2      ; Pop Accumulator
           RTI

```

Freescale Semiconductor, Inc.
 ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005



```

*****
*
*      Subroutine reset
*      the timer compare register
*
*****
COMPRGT   LDA    TCNTHI           ; Read Timer count register
          STA    TEMPTCNTHI      ; and store it in the RAM
          LDA    TCNTLO
          STA    TEMPTCNTLO
          ADD    #$4C             ; Add 1D4C H = 7500 periods
          STA    TEMPTCNTLO      ; with the current timer count
          LDA    TEMPTCNTHI      ; 1 period = 2 us
          ADC    #$1D
          STA    TEMPTCNTHI      ; Save the next count to the register
          STA    OCMPHIL
          LDA    TSTATUS         ; Clear the output compare flag
          LDA    TEMPTCNTLO      ; by access the timer status register
          STA    OCMPL01         ; and then access the output compare
          RTS                    ; register

*****
*
*      This subroutine is to convert
*      the AD data to the LCD
*
*      LCD_STATUS:
*      Bit 3 - Negative Sign
*      Bit 2 - 4th Digit
*      Bit 0 - No zero suppression
*
*****
ADTOLCD   SEI                    ; Disable the Timer Interrupt !!
          BCLR   0,LCD_STATUS
          BCLR   1,LCD_STATUS
          BCLR   2,LCD_STATUS
          CLRA
          STA    PORTA           ; Clear the 3 ports to
          STA    PORTB           ; prevent the segment to turn
          STA    PORTC           ; ON partially
FORTH_DIGIT LDA    NUMBER+3
          STA    DVDND+3
          LDA    NUMBER+2
          STA    DVDND+2
          LDA    NUMBER+1
          STA    DVDND+1
          LDA    NUMBER
          STA    DVDND
          LDA    #$E8
          STA    DVSOR+3
          LDA    #$03
          STA    DVSOR+2
          CLRA
          STA    DVSOR+1
          STA    DVSOR
          JSR    CALDIV           ; Number / 1000
          LDA    QUO+3
          CMP    #1
          BHI    END_LCD         ; Exit if more than 2000
          BLO    NO_4_DIGIT
          BSET   2,LCD_STATUS     ; Turn ON bit-2 of LCD_status
          BSET   0,LCD_STATUS     ; No need to suppress zero
          BRA    THIRD_DIGIT
NO_4_DIGIT BCLR   2,LCD_STATUS   ; Turn OFF bit-2 of LCD_status
THIRD_DIGIT LDA    #$64
          STA    DVSOR+3
          CLRA
          STA    DVSOR+2
          STA    DVSOR+1
          STA    DVSOR
          JSR    CALDIV           ; Remainder / 100
          LDA    QUO+3
          CMP    #0
          BNE    NON_ZERO
          BRSET 0,LCD_STATUS, NON_ZERO
          CLRA
          STA    PORTC           ; Zero suppression
          BRA    SECOND_DIGIT     ; Clear port C
NON_ZERO  LDX    QUO+3
          LDA    $0300,X
          STA    PORTC           ; Display 3rd digit
          BSET   0,LCD_STATUS     ; No need to suppress zero
SECOND_DIGIT LDA    #$0A
          STA    DVSOR+3
          CLRA
          STA    DVSOR+2
          STA    DVSOR+1
          STA    DVSOR

```

Freescale Semiconductor, Inc. ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005

```

JSR    CALDIV                ; Remainder / 10
LDA    QUO+3
CMP    #0
BNE    NON_ZERO1
BRSET  0,LCD_STATUS,NON_ZERO1
CLRA                    ; Zero suppression
STA    PORTB              ; Clear port B
BRA    FIRST_DIGIT
NON_ZERO1  LDX    QUO+3
LDA    $0300,X
STA    PORTB              ; Display 2nd digit
BRA    FIRST_DIGIT
FIRST_DIGIT  LDX    DVDND+3
LDA    $0300,X
STA    PORTA
BRSET  2,LCD_STATUS,DIGIT4_ON ; Jump if the 4th digit is ON
BRA    END_LCD
DIGIT4_ON  BSET  0,PORTA      ; ON the forth digit
BRA    END_LCD
END_LCD  BRCLR  3,LCD_STATUS,NO_SIGN
NO_SIGN  BSET  1,PORTC      ; Turn ON negative sign
CLI                    ; Enable Interrupt again !
RTS

```

```

*****
*
*      Add two 32-bit values
*
*      ADDEND + AUGEND = SUM
*
*****

```

```

CALADD  LDA    ADDEND+3
        ADD    AUGEND+3
        STA    SUM+3
        LDA    ADDEND+2
        ADC    AUGEND+2
        STA    SUM+2
        LDA    ADDEND+1
        ADC    AUGEND+1
        STA    SUM+1
        LDA    ADDEND
        ADC    AUGEND
        STA    SUM
        RTS

```

```

*****
*
*      Subtract two 32-bit values
*
*      MINUE - SUBTRA = DIFF
*
*****

```

```

CALSUB  LDA    MINUE+3
        SUB    SUBTRA+3
        STA    DIFF+3
        LDA    MINUE+2
        SBC    SUBTRA+2
        STA    DIFF+2
        LDA    MINUE+1
        SBC    SUBTRA+1
        STA    DIFF+1
        LDA    MINUE
        SBC    SUBTRA
        STA    DIFF
        RTS

```

```

*****
*
*      Multiply 32-bit value by 32-bit value
*
*      MULTP * MULCAN = MTEMP:MULCAN
*
*****

```

```

CALMUL  LDX    #!32
        CLR    MTEMP
        CLR    MTEMP+1
        CLR    MTEMP+2
        CLR    MTEMP+3
        ROR    MULCAN
        ROR    MULCAN+1
        ROR    MULCAN+2
        ROR    MULCAN+3
MNEXT  BCC    ROTATE
        LDA    MTEMP+3
        ADD    MULTP+3
        STA    MTEMP+3
        LDA    MTEMP+2
        ADC    MULTP+2
        STA    MTEMP+2

```

Freescale Semiconductor, Inc.

ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005



```

LDA    MTEMP+1
ADC    MULTP+1
STA    MTEMP+1
LDA    MTEMP
ADC    MULTP
STA    MTEMP
ROTATE ROR    MTEMP
      ROR    MTEMP+1
      ROR    MTEMP+2
      ROR    MTEMP+3
      ROR    MULCAN
      ROR    MULCAN+1
      ROR    MULCAN+2
      ROR    MULCAN+3          ; MTEMP,MULCAN
DECX
BNE    MNEXT
RTS

```

```

*****
*
*   Divide 32-bit by 32-bit
*
*   DVDND / DVSOR = QUO + DVDND
*
*****

```

```

CALDIV CLR    QUO
      CLR    QUO+1
      CLR    QUO+2
      CLR    QUO+3
      LDA    #1
      TST   DVSOR
      BMI   DIV153
DIV151 INCA
      ASL   DVSOR+3
      ROL   DVSOR+2
      ROL   DVSOR+1
      ROL   DVSOR
      BMI   DIV153
      CMP   #!33
      BNE   DIV151
DIV153 STA    CNT
DIV163 LDA    DVDND+3
      SUB   DVSOR+3
      STA   DVDND+3
      LDA   DVDND+2
      SBC   DVSOR+2
      STA   DVDND+2
      LDA   DVDND+1
      SBC   DVSOR+1
      STA   DVDND+1
      LDA   DVDND
      SBC   DVSOR
      STA   DVDND
      BCC   DIV165
      LDA   DVDND+3
      ADD   DVSOR+3
      STA   DVDND+3
      LDA   DVDND+2
      ADC   DVSOR+2
      STA   DVDND+2
      LDA   DVDND+1
      ADC   DVSOR+1
      STA   DVDND+1
      LDA   DVDND
      ADC   DVSOR
      STA   DVDND
      CLC
      BRA   DIV167

```

```

DIV165 SEC
DIV167 ROL    QUO+3
      ROL    QUO+2
      ROL    QUO+1
      ROL    QUO
      LSR   DVSOR
      ROR   DVSOR+1
      ROR   DVSOR+2
      ROR   DVSOR+3
      DEC   CNT
      BNE   DIV163
      RTS

```

```

*****
*
*   Overflow Interrupt Service Routine
*   Counting the number of overflow
*
*****

```

```

TIMERROV STA    STACK+4

```

```

INC      OFCNT      ; Increase overflow counter
LDA      TSTATUS   ; Clear overflow flag
LDA      TCNTLO
LDA      STACK+4
RTI

```

```

*****
*
*   This subroutine provides services   *
*   for those unintended interrupts   *
*
*****


```

```

SWI      RTI      ; Software interrupt return
TIMERCAP RTI      ; Timer input capture
SCI      RTI      ; Serial communication Interface
IRQ      RTI
ORG      $3FF2    ; For 68HC05B16, the vector location
FDB      SCI      ; starts at 3FF2
FDB      TIMERROV ; For 68HC05B5, the address starts
FDB      TIMERCMP ; 1FF2
FDB      TIMERCAP
FDB      IRQ
FDB      SWI
FDB      RESET

```



Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

Mfax is a trademark of Motorola, Inc.

How to reach us:

USA/EUROPE/Locations Not Listed: Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217. 1-303-675-2140 or 1-800-441-2447

JAPAN: Nippon Motorola Ltd.; SPD, Strategic Planning Office, 141,
4-32-1 Nishi-Gotanda, Shinagawa-ku, Tokyo, Japan. 81-3-5487-8488

Customer Focus Center: 1-800-521-6274

Mfax™: RMFAX0@email.sps.mot.com – TOUCHTONE 1-602-244-6609
Motorola Fax Back System – US & Canada ONLY 1-800-774-1848
– http://sps.motorola.com/mfax/

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298

HOME PAGE: <http://motorola.com/sps/>



MOTOROLA

◇ **For More Information On This Product,
Go to: www.freescale.com**

AN1638/D