
**Interfacing Microchip's MCP3201 Analog/Digital (A/D) Converter to
MC68HC11E9-Based Microcontroller**

*Author: Richard L. Fischer
Microchip Technology Inc.*

INTRODUCTION

Many of the embedded control systems designed today require some flavor of Analog-to-Digital (A/D) Converter. Embedded system applications such as Data Acquisition, Sensor Monitoring, and Instrumentation and Control all have varying A/D Converter requirements.

For the most part, these A/D Converter requirements are a combination of performance, cost, package size basis and availability. In some applications a microcontroller with an on-chip A/D Converter may meet the design requirements. Typically, these on-chip A/D Converter modules fit well into embedded applications which require a 10-35ksp/s A/D Converter. In other applications, a stand-alone A/D Converter is required for various performance reasons.

For those applications which require higher performance or remote sense capability, the Microchip MCP3201 12-bit A/D Converter fits very nicely.

The Microchip Technology Inc. MCP3201 employs a classic SAR architecture. The device uses an internal sample and hold capacitor to store the analog input while the conversion is taking place. Conversion rates of 100ksp/s are possible on the MCP3201. Minimum clock speed (10 kHz or 625sp/s, assuming 16 clocks) is a function of the capacitors used for the sample and hold.

The MCP3201 has a single pseudo-differential input. The (IN-) input is limited to $\pm 100\text{mV}$. This can be used to cancel small noise signals present on both the (IN+) and (IN-) inputs. This provides a means of rejecting noise when the (IN-) input is used to sense a remote signal ground. The (IN+) input can range from the (IN-) input to V_{REF} .

The reference voltage for the MCP3201 is applied to V_{REF} pin. V_{REF} determines the analog input voltage range and the LSB size, i.e.:

$$LSB \text{ size} = \frac{V_{REF}}{2^{12}}$$

As the reference input is reduced, the LSB size is reduced accordingly.

Communication with the MCP3201 is accomplished using a standard SPI™ compatible serial interface. This interface allows direct connection to the serial ports of microcontrollers and digital signal processors.

The MCP3201 is suitable for use with a wide variety of microcontrollers from Microchip and others. This application note describes how to interface the MCP3201 with a Motorola MC684C11 microcontroller. Application Note, AN702 covers microcontrollers based on the Intel 8051 architecture.

Figure 1 shows the hardware schematic for this interface. Appendix A contains a listing of the source code.

CIRCUIT DESCRIPTION

The serial interface of the Microchip MCP3201 A/D Converter has three wires, a serial clock input (DCLK), the serial data output (D_{OUT}) and the chip select input signal ($\overline{CS}/SHDN$). For this simple circuit interface, the Motorola MC68HC11E9 SPI port is used. PORTD:<4> is configured for the serial clock and PORTD:<2> is the data input to the microcontroller. The SPI clock rate for this application is set at 1 MHz.

The MC68HC11 is configured in the master mode with its CPOL and CPHA bits set to logic one (default setting on power-up).

A conversion is initiated with the high to low transition of $\overline{CS}/SHDN$ (active low). The chip select is generated by PORTD:<5> of the microcontroller. The device will sample the analog input from the rising edge of the first clock after \overline{CS} goes low for 1.5 clock cycles. On the falling edge of the second clock, the device will output a low null bit. With the next 12 clocks, the MCP3201 will output the result of the conversion with the MSB first (See Figure 2 and Figure 3). Data is always output from the device on the falling edge of the clock. If the device continues to receive clocks while $\overline{CS}/SHDN$ is low, the device will output the conversion LSB first. If more clocks are provided to the device while $\overline{CS}/SHDN$ is still low (after the LSB first data has been transmitted), the device will clock out zeros indefinitely.

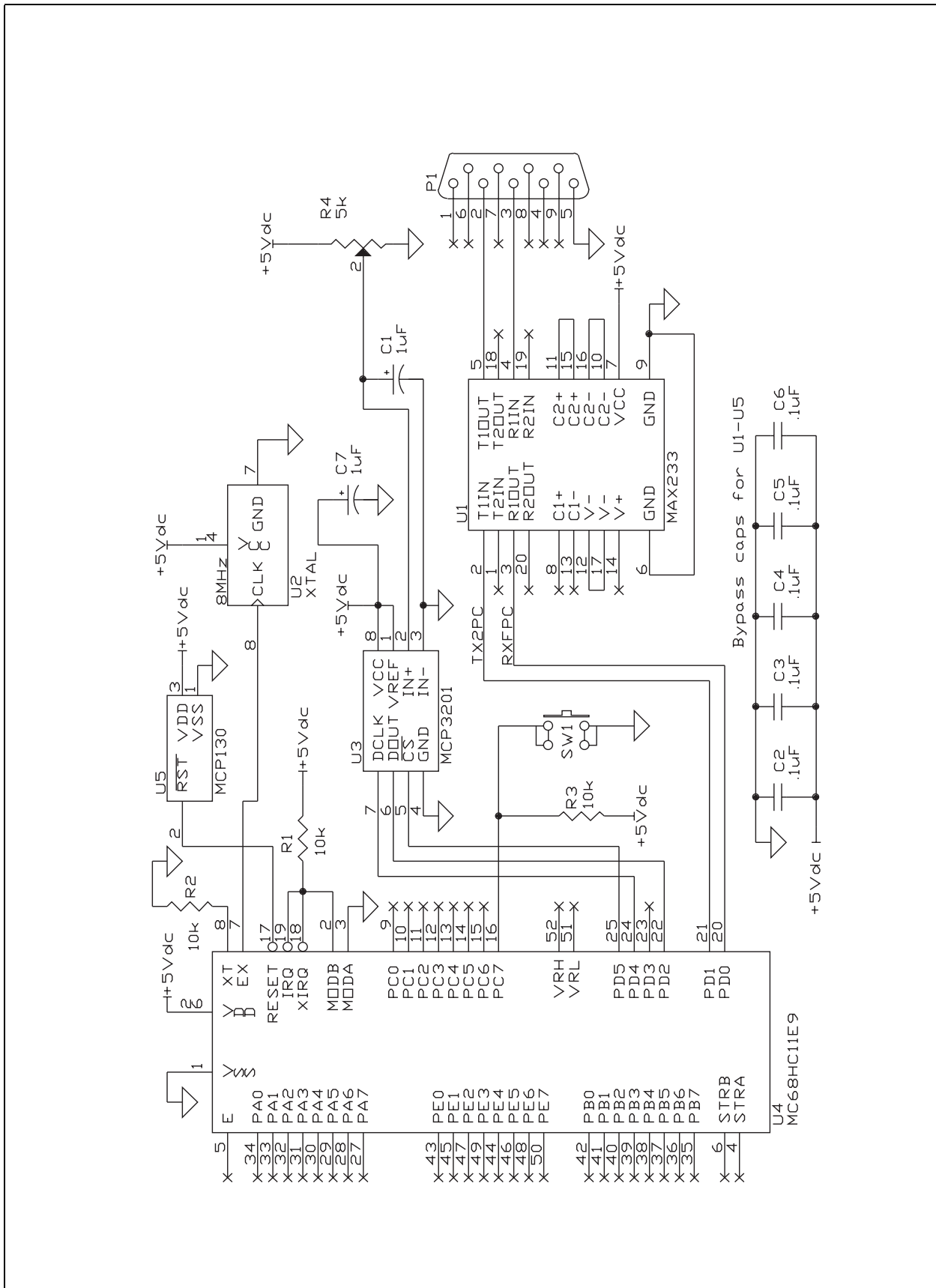


FIGURE 1: Hardware Schematic

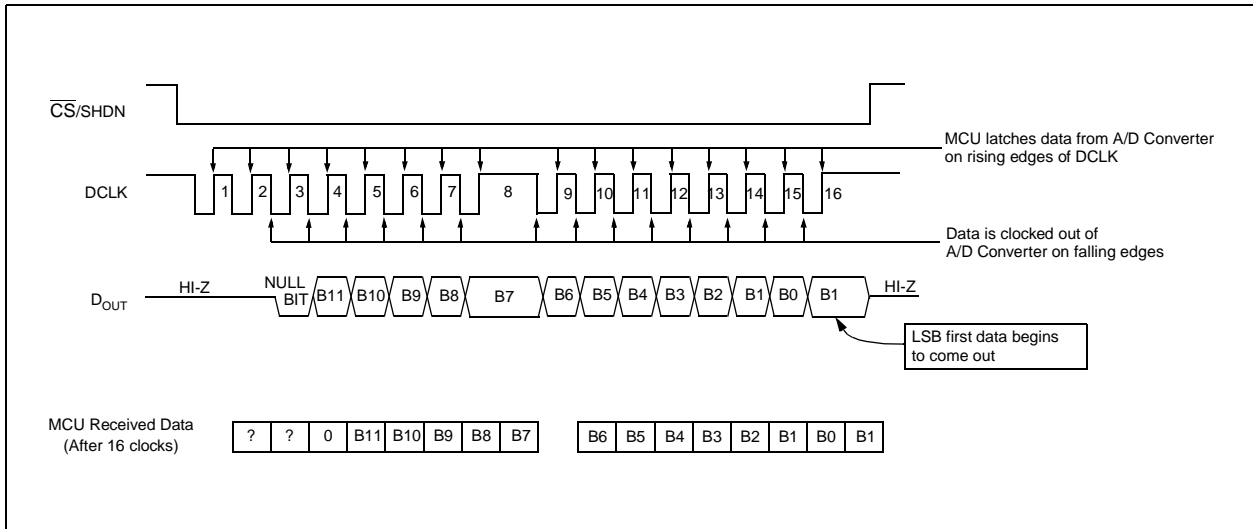


FIGURE 2: SPI Communication using 8-bit segments (Mode 1,1: SCLK idles high).

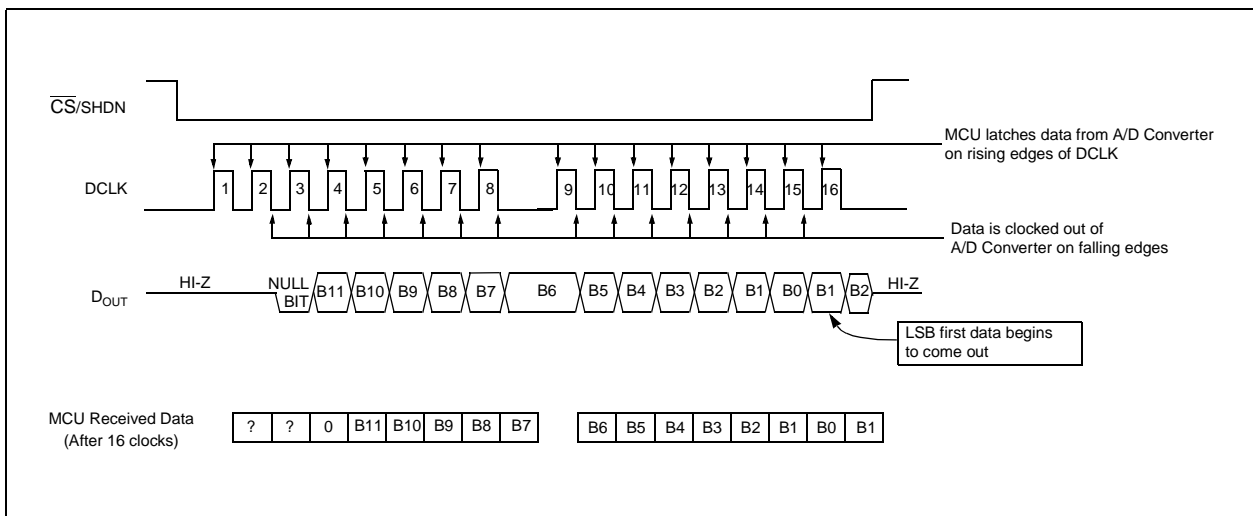


FIGURE 3: SPI Communication using 8-bit segments (Mode 0,0: SCLK idles low).

As the analog input signal is applied to the IN+ and IN- inputs, it is ratioed to the V_{REF} input for conversion scaling.

$$\text{Digital output code} = \frac{V_{IN} \times F.S.}{V_{REF}}$$

Where:

V_{IN} = analog input voltage $V(IN+) - V(IN-)$

V_{REF} = reference voltage

F.S. = full scale = 4096

V_{REF} can be sourced directly from V_{DD} or can be supplied by an external reference. In either configuration, the V_{REF} source must be evaluated for noise contributions during the conversion. The voltage reference input of the MCP3201 ranges from 250mV to $5V_{DC}$, which approximately translates to a corresponding LSB size from 61 μ V to 1.22mV per bit.

$$1.22mV = \frac{5V_{DC}}{2^{12} \text{ bits}}$$

For this simple application, the MCP3201 voltage reference input is tied to $5V_{DC}$. This translates to a 1.22mV/bit resolution for the A/D Converter module. The voltage input to the MCP3201 is implemented with a multi-turn potentiometer. The output voltage range of this passive driver is approximately $0V_{DC}$ to $5V_{DC}$.

Finally, a simple RS-232 interface is implemented using the USART peripheral of the microcontroller and a MAX233 transceiver IC. The USART transmits the captured A/D Converter binary value, both in ASCII and Decimal, to the PC terminal at 9600 baud.

As with all applications which require moderate to high performance A/D Converter operation, proper grounding and layout techniques are essential in achieving optimal performance. Proper power supply decoupling and input signal and V_{REF} parameters must be considered for noise contributions.

SOURCE CODE DESCRIPTION

The code written for this simple application performs five main functions:

1. Controller Initialization.
2. A/D Converter Conversion.
3. Conversion to ASCII.
4. Conversion to Decimal.
5. Transmit ASCII and Decimal to PC for display.

Upon power-up the microcontroller Port pins, USART peripheral and SSP module are initialized. The default microcontroller SPI bus mode is 1,1. The bus mode can be changed on power-up via SW1. If SW1 is depressed on power-up then PORTC:<7> is pulled low. The initialization code checks this pin state and if a logic low SPI bus mode 0,0 is selected. Likewise if PORTC:<7> is a logic high, (SW1 is not depressed), then mode 1,1 is the operational bus state. Once the initialization code is executed, the main code loop is entered and executed continuously.

After asserting PORTD:<5> the SPDR register is written to for initiating a SPI bus cycle. When the SPI cycle is complete, (SPIF flag is set to logic 1), the received data is read from the SPDR register and written to the RAM variable `RESULT_MSB`. The SPDR register is again written to, which initiates a SPI bus cycle, and the second 8-bits are received and written to the RAM variable `RESULT_LSB`. Here the composite result, located in variables `RESULT_MSB` and `RESULT_LSB` is right adjusted one bit location. The $\overline{CS}/SHDN$ is then negated and the MCP3201 enters into the shutdown mode.

Next the `hex_to_ascii` and `hex_to_decimal` routines are called and executed. Then the `display_conversion` routine is executed which sends the data to the USART for transmission to the PC for display.

REFERENCES

Williams, Jim, "Analog Circuit Design," Butterworth-Heinemann

Baker, Bonnie, "Layout Tips for 12-bit A/D Converter Applications," AN688, Microchip Technology Inc.

MCP3201 12-bit A/D Converter with SPI Serial Interface, Microchip Technology, Document DS21290B, 1999.

APPENDIX A: SOURCE CODE

MCP3201.SRC Assembled with IASM 07/01/1999 11:30 PAGE 1
 Interfacing Microchip MCP3201 ADC to Motorola MC68HC11E9-Based Microcontroller

```

1
2
3 *****
4 *
5 *   Filename:      MCP3201.src
6 *   Date:         07/01/99
7 *
8 *   File Version:  1.00
9 *
10 *   Author:       Richard L. Fischer
11 *                Microchip Technology Inc.
12 *
13 *****
14 *
15 *
16 * This code demonstrates how the Microchip MCP3201 Analog-to-Digital
17 * Converter (ADC) is interfaced to the Synchronous Serial Peripheral
18 * (SSP) of the MC68HC11E9 microcontroller. The interface uses two
19 * Serial Peripheral Interface (SPI) lines (SCK, MISO) on the
20 * 68HC11E9 Microcontroller for the clock (SCK) and data in (MISO).
21 * A chip select (CS) to the MCP3201 is generated with a general
22 * purpose port line (PD5). The MC68HC11E9 is placed into the master
23 * mode which allows use of the port line PD5 for the CS control
24 * signal. The simple application uses Mode 00 or Mode 11 to the
25 * define bus clock polarity and phase.
26 *
27 * SPI bus mode 1,1 is the default mode of operation upon power-up.
28 * If SPI bus Mode 0,0 is desired, cycle off power, depress and
29 * hold SW1, cycle on power then release SW1. SPI bus Mode 0,0 is
30 * now the operational mode.
31 *
32 * For this application, the SPI data rate is set to one eighth of
33 * the microcontroller clock frequency. The MC68HC11E9 device clock
34 * frequency used for this application is 8MHz. This translates to
35 * an ADC throughput of 62.5kHz. In order to obtain the maximum
36 * throughput (100kHz) from the MCP3201 ADC the M68HC11E9 must be
37 * clocked at 12.8Mhz.
38 *
39 *
40 *****
41
42
43 *****
44 *           MICROCONTROLLER RELATED EQUATES           *
45 *****
46
0000 47 REGBASE EQU $1000 ; Register Base Address
0000 48 PACTL EQU $26 ; PortA bit7 control
0000 49 PORTA EQU $00 ; PortA Address
0000 50 PORTB EQU $04 ; PortB Address
0000 51 DDRC EQU $07 ; PortD Data Direction Register
0000 52 PORTC EQU $03 ; PortC Address
0000 53 DDRD EQU $09 ; PortD Data Direction Register
0000 54 PORTD EQU $08 ; PortD Address
55
0000 56 SPCR EQU $28 ; SPI Control Register
0000 57 SPSR EQU $29 ; SPI Status Register
0000 58 SPDR EQU $2A ; SPI Data Register
59
0000 60 BAUD EQU $2B ; Baud Rate Register

```

AN704

```
0000          61  SCCR2    EQU  $2D    ; SCI Control Register 2
0000          62  SCSR     EQU  $2E    ; SCI Status Register
0000          63  SCDR     EQU  $2F    ; SCI Data Register
          64
          65
0000          66  MASKCS   EQU  $20    ; Mask for Chip Select Bit
          67
          68
          69  *****
          70  *          INTERNAL RAM USAGE          *
          71  *****
          72
0000          73          ORG  $0000    ; Internal RAM
          74
0000          75  RESULT_MSB RMB  1    ; Converted MSB
0001          76  RESULT_LSB RMB  1    ; Converted LSB
          77
0002          78  THOUS     RMB  1    ; Variable for ASCII thousands
0003          79  HUNDS     RMB  1    ; Variable for ASCII hundreds
0004          80  TENS      RMB  1    ; Variable for ASCII tens
0005          81  ONES      RMB  1    ; Variable for ASCII ones
          82
0006          83  DISPLAY_BUFF RMB 1F  ; Buffer string for display
          84
          85
          86
          87  *****
          88  *          INTERNAL MEMORY EQUATES          *
          89  *****
          90
0025          91  EEPMBEG   EQU  $B600  ; EEPROM begin
0025          92  EEPMEND   EQU  $B7FF  ; EEPROM end
          93
0025          94  EPRMBEG   EQU  $D000  ; EPROM begin
0025          95  EPRMEND   EQU  $FFFF  ; EPROM end
          96
0025          97  RAMBEG    EQU  $0000  ; RAM begin
0025          98  RAMEND    EQU  $01FF  ; RAM end
          99
          100
          101
          102  *****
          103
D000          104          ORG  EPRMBEG    ; Beginning of code
          105
D000 [03] 8E01FF          106  start  LDS  #RAMEND    ; Initialize Stack Pointer
D003 [06] 7F0000          107          CLR  RESULT_MSB  ; Clear ram variables
D006 [06] 7F0001          108          CLR  RESULT_LSB  ; Clear ram variables
          109
D009 [03] CE0006          110          LDX  #DISPLAY_BUFF ; Initialize X pointer to RAM
D00C [04] 18CED12D        111          LDY  #DISPLAY_IMAGE ; Initialize Y pointer to ROM
D010 [05] 18A600          112  copy  LDAA 0,Y        ; Read from ROM
D013 [04] A700            113          STAA 0,X        ; Save into RAM
D015 [03] 08              114          INX          ; Point to next RAM location
D016 [04] 1808            115          INY          ; Point to next ROM location
D018 [04] 8C0026          116          CPX  #DISPLAY_BUFF+20 ; Copy display string complete?
D01B [03] 26F3            117          BNE  copy      ; No, so copy some more
          118
          119
D01D [03] CE1000          120          LDX  #REGBASE    ; Initialize X Index pointer
          121
D020 [07] 1D00FF          122          BCLR  PORTA,X,$FF  ; Set PortA outputs low
D023 [07] 1C2680          123          BSET  PACTL,X,$80  ; Set PortA bit7 to output
D026 [07] 1D04FF          124          BCLR  PORTB,X,$FF  ; Set PortB outputs low
D029 [07] 1C077F          125          BSET  DDRC,X,$7F  ; Set PortC for outputs
          126          ; except for pin7
```

```

D02C [07] 1D03FF    127          BCLR  PORTC,X,$FF      ; Set PortC outputs low
D02F [07] 1C0937    128          BSET  DDRD,X,$37      ; Make all PortD pins outputs
                                129          ; except PD3
D032 [02] 8620      130          LDAA  #$20            ;
D034 [04] A708      131          STAA  PORTD,X         ; Set CS high and all other
                                132
D036 [07] 1C2D08    133          BSET  SCCR2,X,$08     ; Enable SCI TX
D039 [02] 8630      134          LDAA  #$30            ; Initialize SCI for 9600
D03B [04] A72B      135          STAA  BAUD,X         ; baud at 8Mhz
                                136
D03D [07] 1E038006  137          BRSET PORTC,X,$80,m11 ; Branch to Mode 11 if pin high
D041 [02] 8650      138          LDAA  #$50            ; Else, Mode 00
D043 [04] A728      139          STAA  SPCR,X         ; Initialize SPI control reg
D045 [03] 2004      140          BRA   start_conversion ; Go start conversion
                                141
D047 [02] 865C      142          m11 LDAA  #$5C            ; Mode 11
D049 [04] A728      143          STAA  SPCR,X         ; Initialize SPI control reg
                                144
                                145
                                146
                                147          *****
                                148          *
                                149          * The routines to follow perform 3 repetitive functions:
                                150          *
                                151          * 1. Initiate an ADC conversion sequence
                                152          * 2. Convert acquired data for display
                                153          * 2. Transmit converted data to PC
                                154          *
                                155          *****
                                156
                                157          start_conversion
                                158
D04B [07] 1D0820    159          BCLR  PORTD,X,MASKCS  ; Assert CS to ADC
                                160
D04E [06] 6F2A      161          CLR   SPDR,X         ; Initiate SPI bus cycle
D050 [07] 1F2980FC  162          rd1 BRCLR SPSR,X,$80,rd1 ; Wait until cycle completes
D054 [04] A62A      163          LDAA  SPDR,X         ; Read data and clear flag (SPIF)
D056 [03] 9700      164          STAA  RESULT_MSB     ; Save off upper bits
                                165
D058 [06] 6F2A      166          CLR   SPDR,X         ; Initiate SPI bus cycle
D05A [07] 1F2980FC  167          rd2 BRCLR SPSR,X,$80,rd2 ; Wait until cycle completes
                                168
D05E [07] 1C0820    169          BSET  PORTD,X,MASKCS  ; Negate CS, place ADC in shutdown
D061 [04] A62A      170          LDAA  SPDR,X         ; Read data and clear flag (SPIF)
D063 [03] 9701      171          STAA  RESULT_LSB     ; Save off lower bits
                                172
D065 [06] 760000    173          ROR   RESULT_MSB     ; Move bit 0 into carry
D068 [06] 760001    174          ROR   RESULT_LSB     ; Rotate MSB bit0 into LSB bit7
D06B [06] 1500F0    175          BCLR  RESULT_MSB,$F0 ; Mask out upper bits of MSB
                                176
D06E [06] BDD093    177          JSR   hex_to_ascii   ; Convert to ASCII for display
D071 [06] BDD0D3    178          JSR   hex_to_decimal ; Convert to decimal for display
                                179
                                180
                                181
                                182          ***** ROUTINE FOR DISPLAYING CONVERTED DATA *****
                                183
                                184          display_conversion
                                185
D074 [04] 18CE0006  186          tx_loop LDY   #DISPLAY_BUFF ; Initialize Y pointer
D078 [05] 18A600    187          LDAA  0,Y           ; Retrieve MSB upper nibble
                                188
D07B [04] A72F      189          STAA  SCDR,X         ; Initiate SCI Transmit
D07D [07] 1F2E40FC  190          wtx BRCLR SCSR,X,$40,wtx ; Wait until cycle completes
D081 [04] 1808      191          INY                    ; Increment Y pointer
D083 [05] 188C0026  192          CPY   #DISPLAY_BUFF+20 ; All characters sent?

```

AN704

```
D087 [03] 26EF      193          BNE   tx_loop          ; No, so send more characters
                    194
                    195
D089 [04] 18CE0000  196          LDY   #$0000          ; Approximately 485mS delay
D08D [04] 1809      197  decy    DEY          ; Decrement counter
D08F [03] 26FC      198          BNE   decy           ; Stay in loop until Y=0
                    199
D091 [03] 20B8      200          BRA   start_conversion ; Stay in main loop
                    201
                    202
                    203
                    204  ****          ROUTINE FOR CONVERTING TO ASCII  ****
                    205
                    206  hex_to_ascii
                    207
D093 [04] 3C        208          PSHX          ; Save off X Index register
D094 [03] CE0000    209          LDX   #RESULT_MSB   ; Initialize X
D097 [04] 18CE000E  210          LDY   #DISPLAY_BUFF+8 ; Initialize Y
D09B [04] A600      211  next    LDAA  0,X          ; Get Result_MSB data
D09D [03] 36        212          PSHA          ; Save ACCA
D09E [02] 84F0      213          ANDA  #$F0          ; Mask out lower nibble
D0A0 [02] 44        214          LSRA          ; Shift upper nibble
D0A1 [02] 44        215          LSRA          ; into lower nibble
D0A2 [02] 44        216          LSRA          ;
D0A3 [02] 44        217          LSRA          ;
D0A4 [02] 8109      218          CMPA  #$09          ; Is value a number ?
D0A6 [03] 2F07      219          BLE   numb_1        ; Yes, make it 0-9 ($30 - $39)
D0A8 [02] 8B37      220          ADDA  #$37          ; No, make it a letter (A - F)
D0AA [05] 18A700    221          STAA  0,Y          ; Save ASCII letter
D0AD [03] 2005      222          BRA   do_lsb        ; Convert lower nibble
D0AF [02] 8B30      223  numb_1  ADDA  #$30          ;
D0B1 [05] 18A700    224          STAA  0,Y          ; Save ASCII number
                    225
D0B4 [04] 1808      226  do_lsb  INY          ; Increment Y
D0B6 [04] 32        227          PULA          ; Restore ACCA
D0B7 [02] 840F      228          ANDA  #$0F          ; Mask out upper nibble
D0B9 [02] 8109      229          CMPA  #$09          ; Is value a number ?
D0BB [03] 2F07      230          BLE   numb_2        ; Yes, make it 0-9 ($30 - $39)
D0BD [02] 8B37      231          ADDA  #$37          ; No, make it a letter (A - F)
D0BF [05] 18A700    232          STAA  0,Y          ; Save ASCII letter
D0C2 [03] 2005      233          BRA   next1         ;
                    234
D0C4 [02] 8B30      235  numb_2  ADDA  #$30          ;
D0C6 [05] 18A700    236          STAA  0,Y          ; Save ASCII number
                    237
D0C9 [03] 08        238  next1   INX          ; Increment X
D0CA [04] 1808      239          INY          ; Increment Y
D0CC [04] 8C0002    240          CPX   #RESULT_MSB+2 ; Converted all bytes?
D0CF [03] 26CA      241          BNE   next          ; No, so do some more
D0D1 [05] 38        242          PULX          ; Else yes so restore X
D0D2 [05] 39        243          RTS          ; Return from subroutine
                    244
                    245
                    246
                    247  ****          ROUTINE FOR CONVERTING TO DECIMAL  ****
                    248
                    249  hex_to_decimal
                    250
D0D3 [04] DC00      251          LDD   RESULT_MSB     ; Initialize ACCA and ACCB
D0D5 [06] 7F0002    252          CLR   THOUS          ; Zero out THOUSANDS temp location
D0D8 [06] 7F0003    253          CLR   HUNDS         ; Zero out HUNDREDS temp location
D0DB [06] 7F0004    254          CLR   TENS          ; Zero out TENS temp location
D0DE [06] 7F0005    255          CLR   ONES          ; Zero out ONES temp location
                    256
D0E1 [05] 1A8303E7  257  ck_thous CPD   #$03E7     ; ACCD >= 1000 or more?
D0E5 [03] 2508      258          BLO   ck_hunds      ; No, so go check hundreds
```



```

D0E7 [04] 8303E8    259          SUBD  #$03E8      ; Subtract 1000 from ACCD
D0EA [06] 7C0002    260          INC   THOUS      ; Increment THOUS
D0ED [03] 20F2      261          BRA   ck_thous   ; Go check for more
                262
D0EF [05] 1A830063  263  ck_hunds  CPD   #$0063      ; ACCD >= 100 OR more?
D0F3 [03] 2508      264          BLO   ck_tens    ; No, so go check tens
D0F5 [04] 830064    265          SUBD  #$0064      ; Subtract 100 from ACCD
D0F8 [06] 7C0003    266          INC   HUNDS     ; Increment HUND
D0FB [03] 20F2      267          BRA   ck_hunds   ; Go check for more
                268
D0FD [02] C10A      269  ck_tens  CMPB  #$0A      ; No, ACCB >= 10 or more?
D0FF [03] 2507      270          BLO   do_ones    ; No, finish up with ONES
D101 [02] C00A      271          SUBB  #$0A      ; Subtract another 10 from B
D103 [06] 7C0004    272          INC   TENS      ; Bump "TENS"
D106 [03] 20F5      273          BRA   ck_tens    ; Loop again
                274
D108 [02] CB30      275  do_ones  ADDB  #$30      ; Convert ONES to ASCII
D10A [04] 18CE001E  276          LDY   #DISPLAY_BUFF+18 ; Initialize Y pointer into buffer
D10E [03] 9602      277          LDAA  THOUS     ;
D110 [02] 8B30      278          ADDA  #$30      ; Convert THOUS to ASCII
D112 [05] 18A700    279          STAA  0,Y      ; Save it
D115 [04] 1808      280          INY   ; Increment Y
D117 [03] 9603      281          LDAA  HUNDS     ;
D119 [02] 8B30      282          ADDA  #$30      ; Convert HUNDS to ASCII
D11B [05] 18A700    283          STAA  0,Y      ; Save it
D11E [04] 1808      284          INY   ; Increment Y
D120 [03] 9604      285          LDAA  TENS      ;
D122 [02] 8B30      286          ADDA  #$30      ; Convert TENS to ASCII
D124 [05] 18A700    287          STAA  0,Y      ; Save it
D127 [04] 1808      288          INY   ; Increment Y
D129 [05] 18E700    289          STAB  0,Y      ; Save conversion into string
D12C [05] 39        290          RTS           ; Return from subroutine
                291
                292
                293
D12D      4865782D  294  DISPLAY_IMAGE  DB   `Hex-> 0x      : Decimal->      `,0D,0A
                3E203078
                20202020
                203A2044
                6563696D
                616C2D3E
                20202020
                200D0A
                295
                296
                297 *****
298 *           INTERRUPT VECTORS           *
299 *****
300
FFD6      301          ORG   $FFD6      ; VECTORS
                302
FFD6      D000      303          DW   start      ; SCI Serial System - RIE, TIE, TCIE, ILIE
FFD8      D000      304          DW   start      ; SPI Serial Transfer Complete - SPIE
FFDA      D000      305          DW   start      ; Pulse Accumalator Input Edge - PAII
FFDC      D000      306          DW   start      ; Pulse Accumalator Overflow - PAOVI
FFDE      D000      307          DW   start      ; Timer Overflow - TOI
FFE0      D000      308          DW   start      ; Timer Input Capture 4/Output Compare 5 - I4/O5I
FFE2      D000      309          DW   start      ; Timer Output Compare 4 - OC4I
FFE4      D000      310          DW   start      ; Timer Output Compare 3 - OC3I
FFE6      D000      311          DW   start      ; Timer Output Compare 2 - OC2I
FFE8      D000      312          DW   start      ; Timer Output Compare 1 - OC1I
FFEA      D000      313          DW   start      ; Timer Input Capture 3 - IC3I
FFEC      D000      314          DW   start      ; Timer Input Capture 2 - IC2I
FFEE      D000      315          DW   start      ; Timer Input Capture 1 -IC1I
FFF0      D000      316          DW   start      ; Real Time Interrupt - RTII
FFF2      D000      317          DW   start      ; IRQ (External Pin)

```

AN704

```
FFF4      D000      318      DW      start      ; XIRQ Pin
FFF6      D000      319      DW      start      ; Software Interrupt
FFF8      D000      320      DW      start      ; Illegal Opcode Trap
FFFA      D000      321      DW      start      ; COP Failure
FFFC      D000      322      DW      start      ; Clock Monitor Fail
FFFE      D000      323      DW      start      ; Reset
          324
0000          325      END
          326
          327
          328
```

Symbol Table

```
BAUD          002B
CK_HUNDS      D0EF
CK_TENS       D0FD
CK_THOUS      D0E1
COPY          D010
DDRC          0007
DDRDR        0009
DECY          D08D
DISPLAY_BUFF  0006
DISPLAY_CONVERSI D074
DISPLAY_IMAGE D12D
DO_LSB        D0B4
DO_ONES       D108
EEPMBEG       B600
EEPMBEND      B7FF
EPRMBEG       D000
EPRMBEND      FFFF
HEX_TO_ASCII  D093
HEX_TO_DECIMAL D0D3
HUNDS         0003
M11           D047
MASKCS        0020
NEXT          D09B
NEXT1         D0C9
NUMB_1        D0AF
NUMB_2        D0C4
ONES          0005
PACTL         0026
PORTA         0000
PORTB         0004
PORTC         0003
PORTD         0008
RAMBEG        0000
RAMEND        01FF
RD1           D050
RD2           D05A
REGBASE       1000
RESULT_LSB    0001
RESULT_MSB    0000
SCCR2         002D
SCDR          002F
SCSR          002E
SPCR          0028
SPDR          002A
SPSR          0029
START         D000
START_CONVERSION D04B
TENS          0004
THOUS         0002
TX_LOOP       D078
WTX           D07D
```

NOTES:



WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office

Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-786-7200 Fax: 480-786-7277
Technical Support: 480-786-7627
Web Address: <http://www.microchip.com>

Atlanta

Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Boston

Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508-480-9990 Fax: 508-480-8575

Chicago

Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas

Microchip Technology Inc.
4570 Westgrove Drive, Suite 160
Addison, TX 75248
Tel: 972-818-7423 Fax: 972-818-2924

Dayton

Microchip Technology Inc.
Two Prestige Place, Suite 150
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Detroit

Microchip Technology Inc.
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Los Angeles

Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York

Microchip Technology Inc.
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

San Jose

Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

AMERICAS (continued)

Toronto

Microchip Technology Inc.
5925 Airport Road, Suite 200
Mississauga, Ontario L4V 1W1, Canada
Tel: 905-405-6279 Fax: 905-405-6253

ASIA/PACIFIC

Hong Kong

Microchip Asia Pacific
Unit 2101, Tower 2
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

Beijing

Microchip Technology, Beijing
Unit 915, 6 Chaoyangmen Bei Dajie
Dong Erhuan Road, Dongcheng District
New China Hong Kong Manhattan Building
Beijing 100027 PRC
Tel: 86-10-85282100 Fax: 86-10-85282104

India

Microchip Technology Inc.
India Liaison Office
No. 6, Legacy, Convent Road
Bangalore 560 025, India
Tel: 91-80-229-0061 Fax: 91-80-229-0062

Japan

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa 222-0033 Japan
Tel: 81-45-471-6166 Fax: 81-45-471-6122

Korea

Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Shanghai

Microchip Technology
RM 406 Shanghai Golden Bridge Bldg.
2077 Yan'an Road West, Hong Qiao District
Shanghai, PRC 200335
Tel: 86-21-6275-5700 Fax: 86 21-6275-5060

ASIA/PACIFIC (continued)

Singapore

Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan, R.O.C

Microchip Technology Taiwan
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

United Kingdom

Arizona Microchip Technology Ltd.
505 Eskdale Road
Winkers Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5858 Fax: 44-118 921-5835

Denmark

Microchip Technology Denmark ApS
Regus Business Centre
Lautrup hof 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France

Arizona Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - 1er Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany

Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 München, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy

Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

11/15/99



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and water fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOC® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

All rights reserved. © 1999 Microchip Technology Incorporated. Printed in the USA. 11/99 Printed on recycled paper.

Information contained in this publication regarding device applications and the like is intended for suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.