# AN597

# Implementing Ultrasonic Ranging

Author:     Robert Schreiber
            Microchip Technology Inc.

## INTRODUCTION

Object ranging is essential in many types of systems. One of the most popular ranging techniques is ultrasonic ranging. Ultrasonic ranging is used in a wide variety of applications including:
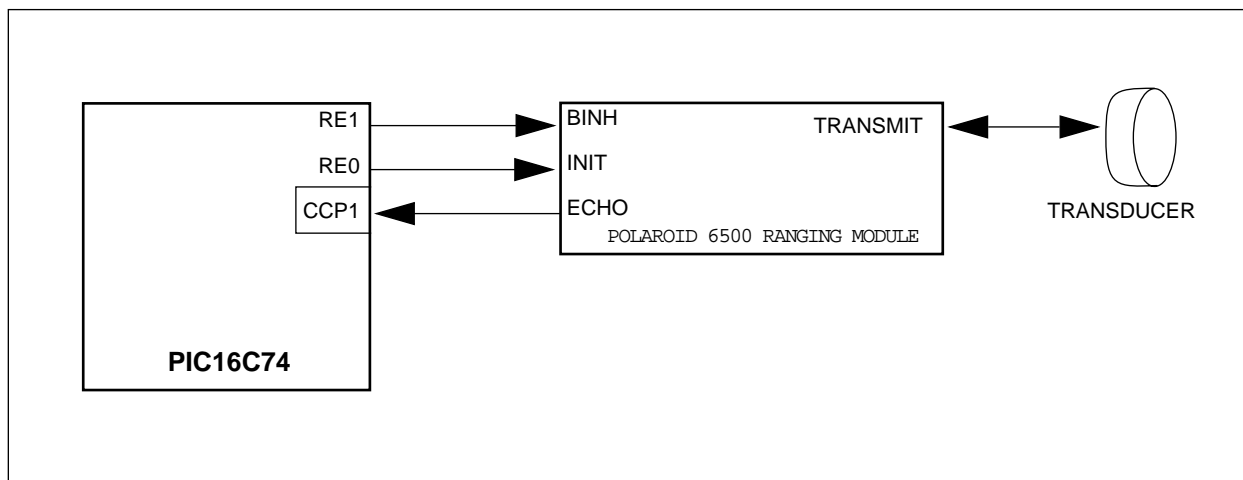
- Autofocus cameras
- Motion detection
- Robotics guidance
- Proximity sensing
- Object ranging

This application note describes a method of interfacing PIC16CXXX microcontrollers to the Polaroid 6500 Ranging Module. This implementation uses a minimum of microcontroller resources, a CCP module and two I/O pins. The two major components of the system are:

- Microcontroller
- Polaroid 6500 Ranging Module

The microcontroller performs the intelligence and arithmetic functions for ultrasonic ranging, while the Polaroid 6500 Ranging Module performs the ultrasonic signal transmissions and echo detection.
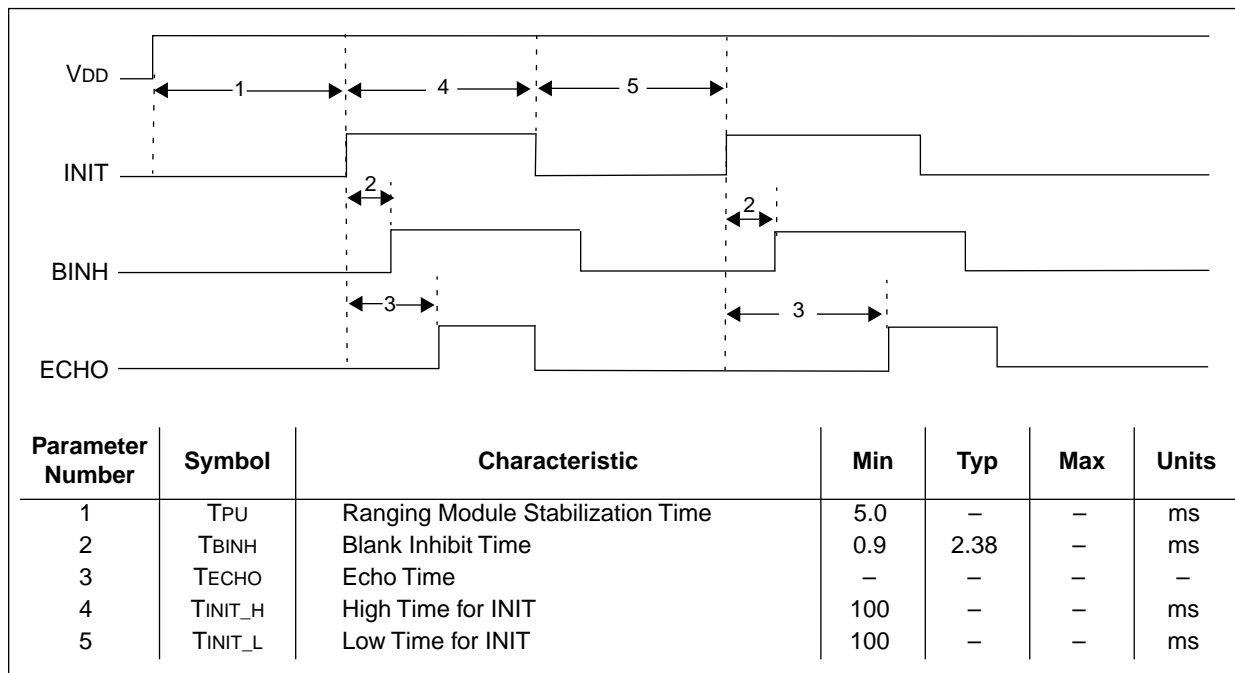
## THEORY OF OPERATION

Ultrasonic ranging entails transmitting a sound wave and measuring the time that it takes for the sound wave to reflect off of an object and back to the origin. The reflection time is proportional to the distance that the object is from the source. In this implementation, the sound wave is transmitted and received from the same transducer. Therefore, a blanking interval is required between signal transmission and reception to eliminate false echoes (i.e., a transmitted signal being detected as its own echo).

## CIRCUIT CONFIGURATION

In this implementation, a PIC16C74 is connected to the ranging module as shown in Figure 1. The RE0 and RE1 I/O pins are configured as digital outputs and are tied to INIT and BINH, respectively. The CCP1 pin is configured as a digital input and is tied to ECHO through a pull-up resistor. The pull-up resistor is needed since the ECHO signal is an open-collector output. The CCP1 pin is configured for capture mode (CCP1CON). Figure 2 shows the timing relationship for $V_{DD}$ and the three signal lines (INIT, BINH, and ECHO).

> **Note:** The ranging module requires 5.0 milliseconds to stabilize during power-up.

**FIGURE 1:    RANGING MODULE INTERFACE**

**FIGURE 2: TIMING DIAGRAM OF RANGING MODULE CONTROL LINES**



| Parameter Number | Symbol | Characteristic | Min | Typ | Max | Units |
|---|---|---|---|---|---|---|
| 1 | TPU | Ranging Module Stabilization Time | 5.0 | – | – | ms |
| 2 | TBINH | Blank Inhibit Time | 0.9 | 2.38 | – | ms |
| 3 | TECHO | Echo Time | – | – | – | – |
| 4 | TINIT_H | High Time for INIT | 100 | – | – | ms |
| 5 | TINIT_L | Low Time for INIT | 100 | – | – | ms |

The PIC16C74 is configured to use one of its internal timers, Timer1, in capture mode to measure the time between signal transmission and echo detection. The resolution of the timer is determined by the microcontroller clock frequency. For this application, a 4 MHz external oscillator was used, giving a resolution of 1 ms per bit. The PIC16C74 initiates a ranging cycle by first clearing Timer1. Timer1 is then enabled and INIT is immediately asserted on the ranging module. When INIT is asserted, the ranging module transmits a series of 16 pulses on the transducer at 49.4 kHz. The transmitted pulses reflect off the object and are received back at the transducer.

The transducer is used for both transmitting and receiving sound waves. A blanking interval is needed to ensure that the transmitted signal has decayed on the transducer, in order not to receive false echoes. In normal operation, the ranging module has a blanking interval of 2.38 milliseconds, which corresponds to a minimum detection distance of approximately 17 inches. However, the BINH (blank inhibit) signal can be manipulated to reduce the blanking time on the transducer to allow for object ranging as close as 6 inches.

In this implementation, the PIC16C74 asserts the BINH signal approximately 0.9 milliseconds after signal transmission. This enables the transducer to receive reflections off objects at a distance of 6 inches. The ranging module asserts the ECHO signal when a valid reflection has been detected. The PIC16C74 uses the ECHO signal to trigger a capture of the Timer1 value. The capture register contains the 16-bit value

representing the elapsed time between signal transmission and echo detection. The PIC16C74 then calculates object distance based on the Timer1 value, microcontroller clock speed, and the velocity of sound in the atmosphere. The basic equation for calculating distance is given below:

Distance (inches) = TECHO time / 147.9 microseconds

> **Note:** The minimum high and low time for INIT is 100 milliseconds, as seen in Figure 2.

## DESIGN CONSIDERATIONS

There are several design considerations which must be taken into account and are listed below.

The absolute measuring distance supported by the ranging module is 6 inches to 35 feet with an accuracy of +/- 1%.

The distance output from the ranging module can be averaged over time to filter distance calculations.

In some applications, the gain of the receiver amplifier may be too low or too high and may need to be adjusted. For example, if the transducer is mounted in a cylinder, the gain may need to be lowered to reduce false echoes within the cylinder. In this case, R1 (refer to the Polariod Ultrasonic Ranging System manual) may be replaced with a 20 k$\Omega$ potentiometer to tweak the gain of the receiver amplifier to reduce false echoes.

In order for the Polaroid 6500 ranging module to operate properly, the power supply must be capable of handling high current transients (2.5 A) during the

transmit pulse. The instantaneous drain on the power supply can be mitigated by installing a storage capacitor across the power lines at the ranging module. A value of 500 microfarads is recommended.

A 200 millisecond interval is recommended between ranging cycles (Figure 2) to allow the transducer to clear.

The ECHO line requires a pull-up resistor (4.7 kΩ was used in this application).

There must be a common ground between the PIC16C74 circuitry and the ranging module.

Some applications may not need the resources of the higher end PIC16CXXX devices. It is still possible to do this application using a device that does not contain a CCP module (for ECHO timing). The capture function can be implemented in firmware. The effect of a firmware implementation is that the resolution of the ECHO time would be 3 $T_{CY}$ cycles versus 1 $T_{CY}$ cycle for the CCP module. Also, the firmware implementation would not allow other tasks to be performed while the capture function was occurring.

Refer to Appendix A for general ranging module specifications.

## APPENDIX A: POLAROID MODULE SPECIFICATIONS

> **Note:** This appendix contains general specifications from the Polaroid Ultrasonic Ranging System Manual. Please refer to the current Polaroid Ultrasonic Ranging System Manual for current information regarding ranging module design considerations.

## DESIGN CONSIDERATIONS IN ULTRASONICS

**Range:** (with user custom designed processing electronics)

Farther

a) Use an acoustic horn to "focus" the sound (narrowing the beamwidth).
b) Use two transducers – 1 receiver and 1 transmitter – facing each other.
c) Lower the transmitting frequency (which will decrease the attenuation in air).

Closer

a) Use a shorter transmit signal (such as four cycles).
a) Use two transducers – one to transmit, one to receive (eliminates waiting for damping time).

Resolution

a) Above all, know the target and range well, and design a system with them in mind.
b) Use a higher transmit frequency.
c) Look at phase differences of a given cycle of the transmitted signal and received echo (as opposed to using and integration technique).
d) Increase the clock frequency of the timer.

**Accuracy:** (again, you must have a well defined target)

Temperature Compensate

a) Use a second small target, as a reference, at a known distance in the ranging path (such as a 1/4" rod several feet away), process both echoes, then normalize the second distance with respect to the first, since t1/d1 = t2/d2.
b) Incorporate a temperature sensing integrated circuit to drive a VCO to do the distance interval clocking.
c) To increase sensitivity of detection circuit change the value of C4 from 3300 pF to 1000 pF on the 6500 Series Ranging Module.

**Beam Width:**

Increase

a) Use an acoustic lens (to disperse the signal).
b) Decrease the transmitting frequency.
c) Use several transducers to span an area.

Decrease

a) Use an acoustic horn (to focus the sound).
b) Increase the transmitting frequency.

**TABLE 1: RECOMMENDED OPERATING CONDITIONS**

| | | Min. | Max. | Unit |
|---|---|---|---|---|
| Supply Voltage, $V_{CC}$ | | 4.5 | 6.8 | V |
| High-level input voltage, $V_{IH}$ | BINH, INIT | 2.1 | — | V |
| Low-level input voltage, $V_{IL}$ | BINH, INIT | — | 0.6 | V |
| ECHO and OSC output voltage | | — | 6.8 | V |
| Delay time, power up to INIT high | | 5 | — | ms |
| Recycle period | | 80 | — | ms |
| Operating free-air temperature, $T_A$ | | 0 | 40 | °C |

**TABLE 2: ELECTRICAL CHARACTERISTICS OVER RECOMMENDED RANGES OF SUPPLY VOLTAGE AND OPERATING FREE-AIR TEMPERATURE (UNLESS OTHERWISE NOTED)**

| Parameter | | Test Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Input current | BINH, INIT | V1 = 2.1V | — | — | 1 | mA |
| High-level output current, $I_{OH}$ | ECHO, OSC | $V_{OH}$ = 5.5V | — | — | 100 | µA |
| Low-level output voltage, $V_{OL}$ | ECHO, OSC | $I_{OL}$ = 1.6 mA | — | — | 0.4 | V |
| Transducer bias voltage | | $T_A$ = 25°C | — | 200 | | V |
| Transducer output voltage (peak-to-peak) | | $T_A$ = 25°C | — | 400 | | V |
| Number of cycles for XDCR output to reach 400V | | C= 500 pF | — | — | 7 | |
| Internal blanking interval | | | — | 2.38* | — | ms |
| Frequency during 16-pulse transmit period | OSC output | | — | 49.4* | — | kHz |
| | XMIT output | | — | 49.4* | — | |
| Frequency after 16 pulse transmit period | OSC output | | — | 93.3* | — | kHz |
| | XMIT output | | — | 0 | — | |
| Supply current, $I_{CC}$ | During transmit period | | — | — | 2000 | mA |
| | After transmit period | | — | — | 100 | |

\* These typical values apply for a 420 kHz ceramic resonator.

# AN597

## APPENDIX B: FIRMWARE LISTING

```
MPASM 01.40 Released              XDCR.ASM   1-22-1997  10:55:26         PAGE  1


LOC   OBJECT CODE      LINE SOURCE TEXT
  VALUE

                 00001 ; XDCR.ASM
                 00002 ;
                 00003 ; This routine continually executes ranging cycles in the
                 00004 ; following order:
                 00005 ;
                 00006 ;        1) Timers and Flags are cleared
                 00007 ;        2) Ranging Cycle Executes
                 00008 ;        3) Distance is Calculated (to 0.5 inch)
                 00009 ;        4) HW is re-initialized for next cycle
                 00010 ;
                 00011 ; The processor uses a 4MHz oscillator, so all timing
                 00012 ; calculations are referenced to that.  The calculated
                 00013 ; distance is a 16-bit result in the ACCbHI:ACCbLO registers.
                 00014 ;
                 00015 ;        Program:          XDCR.ASM
                 00016 ;        Revision Date:
                 00017 ;                          1-22-97      Compatibility with MPASMWIN 1.40
                 00018 ;
                 00019 ;
                 00020
                 00021          LIST P=16C74
                 00022 ;
                 00001          LIST
                 00002 ; P16C74.INC  Standard Header File, Version 1.00  Microchip Technology, Inc.
                 00318          LIST
                 00025
  00000030       00026 TEMP     equ     0x30            ;Temporary storage location
  00000031       00027 TEMP1    equ     0x31            ;Temporary storage location
  00000032       00028 TEMP2    equ     0x32            ;Temporary storage location
  00000033       00029 TEMP3    equ     0x33            ;Temporary storage location
  00000034       00030 COUNT1   equ     0x34            ;Temporary count register
  00000035       00031 COUNT2   equ     0x35            ;Temporary count register
                 00032 ;
                 00033 #DEFINE      XDCR               ; Flag for conditional assemble of test code
                 00034                                 ; in file DBL_DIVF.ASM. END directive MUST be
                 00035                                 ; commented out in file DBL_DIVF.ASM
                 00036 ;
                 00037 ;****************************************************************
                 00038          LIST
                 00039
                 00040 ;******************
                 00041 ; Bank 0 Registers
                 00042 ;******************
                 00043 ;
                 00044 ; TMR1 is off, Prescaler is 1 for a capture timeout of 65 msec
0000 0190        00045          clrf    T1CON
                 00046 ; Set to capture on every rising edge
0001 3005        00047          movlw   0x05
0002 0097        00048          movwf   CCP1CON
                 00049 ; Clear the Ports
0003 0185        00050          clrf    PORTA
0004 0186        00051          clrf    PORTB
0005 0187        00052          clrf    PORTC
0006 0188        00053          clrf    PORTD
```

```
0007 0189      00054        clrf    PORTE
               00055 ;
               00056 ;******************
               00057 ; Bank 1 Registers
               00058 ;******************
               00059 ;
0008 1683      00060        bsf     STATUS,RP0      ; Set Bank1
               00061 ; Port A is Digital, Port E is Digital
0009 3007      00062        movlw   0x07
Message[302]: Register in operand not in bank 0.  Ensure that bank bits are correct.
000A 009F      00063        movwf   ADCON1
               00064 ; Configure CCP1 (RC2) as an input, and all other ports
               00065 ; as Outputs, (RE0 = INIT, RE1 = BINH)
Message[302]: Register in operand not in bank 0.  Ensure that bank bits are correct.
000B 0185      00066        clrf    TRISA
Message[302]: Register in operand not in bank 0.  Ensure that bank bits are correct.
000C 0186      00067        clrf    TRISB
000D 3004      00068        movlw   0x04
Message[302]: Register in operand not in bank 0.  Ensure that bank bits are correct.
000E 0087      00069        movwf   TRISC
Message[302]: Register in operand not in bank 0.  Ensure that bank bits are correct.
000F 0188      00070        clrf    TRISD
Message[302]: Register in operand not in bank 0.  Ensure that bank bits are correct.
0010 0189      00071        clrf    TRISE
0011 1283      00072        bcf     STATUS,RP0      ; Clear RP0
0012          00073 Xdcr
               00074 ;
               00075 ; Initialize Timers and Flags
               00076 ;
0012 1010      00077        bcf     T1CON,0 ; Disable TMR1
0013 018C      00078        clrf    PIR1    ; Clear Timer1 Overflow Flag & Timer1 Capture Flag
0014 018E      00079        clrf    TMR1L   ; Clear TMR1L
0015 018F      00080        clrf    TMR1H   ; Clear TMR1H
0016 0195      00081        clrf    CCPR1L  ; Clear CCPR1L
0017 0196      00082        clrf    CCPR1H  ; Clear CCPR1H
0018 1409      00083        bsf     PORTE,0 ; Set INIT High on Ranging Module
0019 1410      00084        bsf     T1CON,0 ; Enable TMR1
001A 21F3      00085        call    DEL_9   ; Delay 0.9 msec for transducer to stabilize
001B 1489      00086        bsf     PORTE,1 ; Enable Transducer to Receive (BINH)
001C          00087 chk_t1
001C 190C      00088        btfsc   PIR1,2          ; Check for Capture
001D 2822      00089        goto    chk_done        ; Jump if Capture
001E 1C0C      00090        btfss   PIR1,0          ; Check for TMR1 Overflow
001F 281C      00091        goto    chk_t1          ; Loop if nothing happened
0020 1010      00092        bcf     T1CON,0         ; Turn off TMR1
0021 2833      00093        goto    ovr_flo         ; Capture event did not occur
0022          00094 chk_done
               00095 ;
               00096 ; Calculate distance to 0.5 inch resolution
               00097 ;
0022 1010      00098        bcf     T1CON,0         ; Turn off TMR1
0023 0815      00099        movf    CCPR1L,W        ; Move LSB into W
0024 00A2      00100        movwf   ACCbLO          ; Move LSB into ACCbLO
0025 0816      00101        movf    CCPR1H,W        ; Move MSB into W
0026 00A3      00102        movwf   ACCbHI          ; Move MSB into ACCbHI
0027 304A      00103        movlw   0x4A            ; Move 75usec/0.50in into W
0028 00A0      00104        movwf   ACCaLO          ; Move LSB into ACCaLO
0029 01A1      00105        clrf    ACCaHI          ; Clear MSB (ACCaHI)
002A 208F      00106        call    D_divF          ; Call 16-bit/8-bit routine
               00107                                ; which is described in
               00108                                ; Application Note 544
002B 3025      00109        movlw   0x25            ; Check remainder to see if
002C 0224      00110        subwf   ACCcLO,W        ; we should round up...
002D 1803      00111        btfsc   STATUS,C        ; If Remainder < (0.5 * Divisor), skip
002E 0AA2      00112        incf    ACCbLO,F        ; Round up
002F 1903      00113        btfsc   STATUS,Z        ; Check low byte for wrap around
0030 0AA3      00114        incf    ACCbHI,F        ; If LSB wrapped, increment high byte
```

```
0031 1D03      00115          btfss   STATUS,Z         ; Check high byte for wrap around
0032 2835      00116          goto    done             ; High byte didn't wrap
0033          00117 ovr_flo
0033 01A2      00118          clrf    ACCbLO
0034 01A3      00119          clrf    ACCbHI
0035          00120 done
0035 21FD      00121          call    DEL_100          ; Wait 100 msec before clearing HW.
0036 1009      00122          bcf     PORTE,0          ; Disable INIT
0037 1089      00123          bcf     PORTE,1          ; Disable BINH
0038 21FD      00124          call    DEL_100          ; Wait 100 msec before enabling HW.
0039 2812      00125          goto    Xdcr
              00126
              00319          LIST
              00320
              00636          LIST
              00044
  00000001    00045 TRUE     equ     1h
  00000000    00046 FALSE    equ     0h
              00047
0080          00048          org     0x080
              00049 ;****************************************************************
  00000000    00050 SIGNED   equ     FALSE            ; Set This To 'TRUE' if the routines
              00051 ;                                 ; for Multiplication & Division needs
              00052 ;                                 ; to be assembled as Signed Integer
              00053 ;                                 ; Routines. If 'FALSE' the above two
              00054 ;                                 ; routines ( D_mpy & D_div ) use
              00055 ;                                 ; unsigned arithmetic.
              00056 ;****************************************************************;
              00057 ;        division macro
              00058 ;
              00059 divMac   MACRO
              00060          LOCAL   NOCHK
              00061          LOCAL   NOGO
              00062 ;
              00063          bcf     STATUS,C
              00064          rlf     ACCdLO, F
              00065          rlf     ACCdHI, F
              00066          rlf     ACCcLO, F
              00067          rlf     ACCcHI, F
              00068          movf    ACCaHI,W
              00069          subwf   ACCcHI,W         ;check if a>c
              00070          btfss   STATUS,Z
              00071          goto    NOCHK
              00072          movf    ACCaLO,W
              00073          subwf   ACCcLO,W         ;if msb equal then check lsb
              00074 NOCHK    btfss   STATUS,C         ;carry set if c>a
              00075          goto    NOGO
              00076          movf    ACCaLO,W         ;c-a into c
              00077          subwf   ACCcLO, F
              00078          btfss   STATUS,C
              00079          decf    ACCcHI, F
              00080          movf    ACCaHI,W
              00081          subwf   ACCcHI, F
              00082          bsf     STATUS,C         ;shift a 1 into b (result)
              00083 NOGO     rlf     ACCbLO, F
              00084          rlf     ACCbHI, F
              00085 ;
              00086          ENDM
              00087 ;
              00088 ;****************************************************************
              00089 ;        Double Precision Divide ( 16/16 -> 16 )
              00090 ;
              00091 ; ( ACCb/ACCa -> ACCb with remainder in ACCc ) : 16 bit output
              00092 ; with Quotiont in ACCb (ACCbHI,ACCbLO) and Remainder in ACCc
              00093 ; (ACCcHI,ACCcLO).
              00094 ;   NOTE  :  Before calling this routine, the user should make sure that
```

```
                 00095 ;           the Numerator(ACCb) is greater than Denominator(ACCa). If
                 00096 ;           the case is not true, the user should scale either Numerator
                 00097 ;           or Denominator or both such that Numerator is greater than
                 00098 ;           the Denominator.
                 00099 ;
                 00100 ;
0080 3010        00101 setup   movlw   .16              ; for 16 shifts
0081 00A8        00102         movwf   temp
0082 0823        00103         movf    ACCbHI,W         ;move ACCb to ACCd
0083 00A7        00104         movwf   ACCdHI
0084 0822        00105         movf    ACCbLO,W
0085 00A6        00106         movwf   ACCdLO
0086 01A3        00107         clrf    ACCbHI
0087 01A2        00108         clrf    ACCbLO
0088 3400        00109         retlw   0
                 00110 ;
                 00111 ;********************************************************************
                 00112 ;
0089 09A0        00113 neg_A   comf    ACCaLO, F        ; negate ACCa ( -ACCa -> ACCa )
008A 0AA0        00114         incf    ACCaLO, F
008B 1903        00115         btfsc   STATUS,Z
008C 03A1        00116         decf    ACCaHI, F
008D 09A1        00117         comf    ACCaHI, F
008E 3400        00118         retlw   0
                 00119 ;
                 00120 ;********************************************************************
                 00121
                 00122 ;
008F             00123 D_divF
                 00124 ;
                 00125     IF   SIGNED
                 00126     CALL    S_SIGN
                 00127     ENDIF
                 00128 ;
008F 2080        00129         call    setup
0090 01A5        00130         clrf    ACCcHI
0091 01A4        00131         clrf    ACCcLO
                 00132 ;
                 00133 ; use the mulMac macro 16 times
                 00134 ;
                 00135         divMac
  0000           M       LOCAL   NOCHK
  0000           M       LOCAL   NOGO
                 M ;
0092 1003        M       bcf     STATUS,C
0093 0DA6        M       rlf     ACCdLO, F
0094 0DA7        M       rlf     ACCdHI, F
0095 0DA4        M       rlf     ACCcLO, F
0096 0DA5        M       rlf     ACCcHI, F
0097 0821        M       movf    ACCaHI,W
0098 0225        M       subwf   ACCcHI,W         ;check if a>c
0099 1D03        M       btfss   STATUS,Z
009A 289D        M       goto    NOCHK
009B 0820        M       movf    ACCaLO,W
009C 0224        M       subwf   ACCcLO,W         ;if msb equal then check lsb
009D 1C03        M NOCHK btfss   STATUS,C         ;carry set if c>a
009E 28A6        M       goto    NOGO
009F 0820        M       movf    ACCaLO,W         ;c-a into c
00A0 02A4        M       subwf   ACCcLO, F
00A1 1C03        M       btfss   STATUS,C
00A2 03A5        M       decf    ACCcHI, F
00A3 0821        M       movf    ACCaHI,W
00A4 02A5        M       subwf   ACCcHI, F
00A5 1403        M       bsf     STATUS,C         ;shift a 1 into b (result)
00A6 0DA2        M NOGO  rlf     ACCbLO, F
00A7 0DA3        M       rlf     ACCbHI, F
                 M ;
```

```
                        00136           divMac
    0000                M               LOCAL   NOCHK
    0000                M               LOCAL   NOGO
                        M ;
00A8 1003               M               bcf     STATUS,C
00A9 0DA6               M               rlf     ACCdLO, F
00AA 0DA7               M               rlf     ACCdHI, F
00AB 0DA4               M               rlf     ACCcLO, F
00AC 0DA5               M               rlf     ACCcHI, F
00AD 0821               M               movf    ACCaHI,W
00AE 0225               M               subwf   ACCcHI,W        ;check if a>c
00AF 1D03               M               btfss   STATUS,Z
00B0 28B3               M               goto    NOCHK
00B1 0820               M               movf    ACCaLO,W
00B2 0224               M               subwf   ACCcLO,W        ;if msb equal then check lsb
00B3 1C03               M NOCHK         btfss   STATUS,C        ;carry set if c>a
00B4 28BC               M               goto    NOGO
00B5 0820               M               movf    ACCaLO,W        ;c-a into c
00B6 02A4               M               subwf   ACCcLO, F
00B7 1C03               M               btfss   STATUS,C
00B8 03A5               M               decf    ACCcHI, F
00B9 0821               M               movf    ACCaHI,W
00BA 02A5               M               subwf   ACCcHI, F
00BB 1403               M               bsf     STATUS,C        ;shift a 1 into b (result)
00BC 0DA2               M NOGO          rlf     ACCbLO, F
00BD 0DA3               M               rlf     ACCbHI, F
                        M ;
                        00137           divMac
    0000                M               LOCAL   NOCHK
    0000                M               LOCAL   NOGO
                        M ;
00BE 1003               M               bcf     STATUS,C
00BF 0DA6               M               rlf     ACCdLO, F
00C0 0DA7               M               rlf     ACCdHI, F
00C1 0DA4               M               rlf     ACCcLO, F
00C2 0DA5               M               rlf     ACCcHI, F
00C3 0821               M               movf    ACCaHI,W
00C4 0225               M               subwf   ACCcHI,W        ;check if a>c
00C5 1D03               M               btfss   STATUS,Z
00C6 28C9               M               goto    NOCHK
00C7 0820               M               movf    ACCaLO,W
00C8 0224               M               subwf   ACCcLO,W        ;if msb equal then check lsb
00C9 1C03               M NOCHK         btfss   STATUS,C        ;carry set if c>a
00CA 28D2               M               goto    NOGO
00CB 0820               M               movf    ACCaLO,W        ;c-a into c
00CC 02A4               M               subwf   ACCcLO, F
00CD 1C03               M               btfss   STATUS,C
00CE 03A5               M               decf    ACCcHI, F
00CF 0821               M               movf    ACCaHI,W
00D0 02A5               M               subwf   ACCcHI, F
00D1 1403               M               bsf     STATUS,C        ;shift a 1 into b (result)
00D2 0DA2               M NOGO          rlf     ACCbLO, F
00D3 0DA3               M               rlf     ACCbHI, F
                        M ;
                        00138           divMac
    0000                M               LOCAL   NOCHK
    0000                M               LOCAL   NOGO
                        M ;
00D4 1003               M               bcf     STATUS,C
00D5 0DA6               M               rlf     ACCdLO, F
00D6 0DA7               M               rlf     ACCdHI, F
00D7 0DA4               M               rlf     ACCcLO, F
00D8 0DA5               M               rlf     ACCcHI, F
00D9 0821               M               movf    ACCaHI,W
00DA 0225               M               subwf   ACCcHI,W        ;check if a>c
00DB 1D03               M               btfss   STATUS,Z
00DC 28DF               M               goto    NOCHK
```

```
00DD 0820              M          movf    ACCaLO,W
00DE 0224              M          subwf   ACCcLO,W         ;if msb equal then check lsb
00DF 1C03              M NOCHK     btfss   STATUS,C         ;carry set if c>a
00E0 28E8              M          goto    NOGO
00E1 0820              M          movf    ACCaLO,W         ;c-a into c
00E2 02A4              M          subwf   ACCcLO, F
00E3 1C03              M          btfss   STATUS,C
00E4 03A5              M          decf    ACCcHI, F
00E5 0821              M          movf    ACCaHI,W
00E6 02A5              M          subwf   ACCcHI, F
00E7 1403              M          bsf     STATUS,C         ;shift a 1 into b (result)
00E8 0DA2              M NOGO      rlf     ACCbLO, F
00E9 0DA3              M          rlf     ACCbHI, F
                      M ;
              00139    M          divMac
   0000               M          LOCAL   NOCHK
   0000               M          LOCAL   NOGO
                      M ;
00EA 1003              M          bcf     STATUS,C
00EB 0DA6              M          rlf     ACCdLO, F
00EC 0DA7              M          rlf     ACCdHI, F
00ED 0DA4              M          rlf     ACCcLO, F
00EE 0DA5              M          rlf     ACCcHI, F
00EF 0821              M          movf    ACCaHI,W
00F0 0225              M          subwf   ACCcHI,W         ;check if a>c
00F1 1D03              M          btfss   STATUS,Z
00F2 28F5              M          goto    NOCHK
00F3 0820              M          movf    ACCaLO,W
00F4 0224              M          subwf   ACCcLO,W         ;if msb equal then check lsb
00F5 1C03              M NOCHK     btfss   STATUS,C         ;carry set if c>a
00F6 28FE              M          goto    NOGO
00F7 0820              M          movf    ACCaLO,W         ;c-a into c
00F8 02A4              M          subwf   ACCcLO, F
00F9 1C03              M          btfss   STATUS,C
00FA 03A5              M          decf    ACCcHI, F
00FB 0821              M          movf    ACCaHI,W
00FC 02A5              M          subwf   ACCcHI, F
00FD 1403              M          bsf     STATUS,C         ;shift a 1 into b (result)
00FE 0DA2              M NOGO      rlf     ACCbLO, F
00FF 0DA3              M          rlf     ACCbHI, F
                      M ;
              00140    M          divMac
   0000               M          LOCAL   NOCHK
   0000               M          LOCAL   NOGO
                      M ;
0100 1003              M          bcf     STATUS,C
0101 0DA6              M          rlf     ACCdLO, F
0102 0DA7              M          rlf     ACCdHI, F
0103 0DA4              M          rlf     ACCcLO, F
0104 0DA5              M          rlf     ACCcHI, F
0105 0821              M          movf    ACCaHI,W
0106 0225              M          subwf   ACCcHI,W         ;check if a>c
0107 1D03              M          btfss   STATUS,Z
0108 290B              M          goto    NOCHK
0109 0820              M          movf    ACCaLO,W
010A 0224              M          subwf   ACCcLO,W         ;if msb equal then check lsb
010B 1C03              M NOCHK     btfss   STATUS,C         ;carry set if c>a
010C 2914              M          goto    NOGO
010D 0820              M          movf    ACCaLO,W         ;c-a into c
010E 02A4              M          subwf   ACCcLO, F
010F 1C03              M          btfss   STATUS,C
0110 03A5              M          decf    ACCcHI, F
0111 0821              M          movf    ACCaHI,W
0112 02A5              M          subwf   ACCcHI, F
0113 1403              M          bsf     STATUS,C         ;shift a 1 into b (result)
0114 0DA2              M NOGO      rlf     ACCbLO, F
0115 0DA3              M          rlf     ACCbHI, F
```

```
                    M ;
                  00141          divMac
    0000          M              LOCAL    NOCHK
    0000          M              LOCAL    NOGO
                  M ;
 0116 1003        M              bcf      STATUS,C
 0117 0DA6        M              rlf      ACCdLO, F
 0118 0DA7        M              rlf      ACCdHI, F
 0119 0DA4        M              rlf      ACCcLO, F
 011A 0DA5        M              rlf      ACCcHI, F
 011B 0821        M              movf     ACCaHI,W
 011C 0225        M              subwf    ACCcHI,W        ;check if a>c
 011D 1D03        M              btfss    STATUS,Z
 011E 2921        M              goto     NOCHK
 011F 0820        M              movf     ACCaLO,W
 0120 0224        M              subwf    ACCcLO,W        ;if msb equal then check lsb
 0121 1C03        M NOCHK        btfss    STATUS,C        ;carry set if c>a
 0122 292A        M              goto     NOGO
 0123 0820        M              movf     ACCaLO,W        ;c-a into c
 0124 02A4        M              subwf    ACCcLO, F
 0125 1C03        M              btfss    STATUS,C
 0126 03A5        M              decf     ACCcHI, F
 0127 0821        M              movf     ACCaHI,W
 0128 02A5        M              subwf    ACCcHI, F
 0129 1403        M              bsf      STATUS,C        ;shift a 1 into b (result)
 012A 0DA2        M NOGO         rlf      ACCbLO, F
 012B 0DA3        M              rlf      ACCbHI, F
                  M ;
                  00142          divMac
    0000          M              LOCAL    NOCHK
    0000          M              LOCAL    NOGO
                  M ;
 012C 1003        M              bcf      STATUS,C
 012D 0DA6        M              rlf      ACCdLO, F
 012E 0DA7        M              rlf      ACCdHI, F
 012F 0DA4        M              rlf      ACCcLO, F
 0130 0DA5        M              rlf      ACCcHI, F
 0131 0821        M              movf     ACCaHI,W
 0132 0225        M              subwf    ACCcHI,W        ;check if a>c
 0133 1D03        M              btfss    STATUS,Z
 0134 2937        M              goto     NOCHK
 0135 0820        M              movf     ACCaLO,W
 0136 0224        M              subwf    ACCcLO,W        ;if msb equal then check lsb
 0137 1C03        M NOCHK        btfss    STATUS,C        ;carry set if c>a
 0138 2940        M              goto     NOGO
 0139 0820        M              movf     ACCaLO,W        ;c-a into c
 013A 02A4        M              subwf    ACCcLO, F
 013B 1C03        M              btfss    STATUS,C
 013C 03A5        M              decf     ACCcHI, F
 013D 0821        M              movf     ACCaHI,W
 013E 02A5        M              subwf    ACCcHI, F
 013F 1403        M              bsf      STATUS,C        ;shift a 1 into b (result)
 0140 0DA2        M NOGO         rlf      ACCbLO, F
 0141 0DA3        M              rlf      ACCbHI, F
                  M ;
                  00143          divMac
    0000          M              LOCAL    NOCHK
    0000          M              LOCAL    NOGO
                  M ;
 0142 1003        M              bcf      STATUS,C
 0143 0DA6        M              rlf      ACCdLO, F
 0144 0DA7        M              rlf      ACCdHI, F
 0145 0DA4        M              rlf      ACCcLO, F
 0146 0DA5        M              rlf      ACCcHI, F
 0147 0821        M              movf     ACCaHI,W
 0148 0225        M              subwf    ACCcHI,W        ;check if a>c
 0149 1D03        M              btfss    STATUS,Z
```

```
014A 294D            M           goto    NOCHK
014B 0820            M           movf    ACCaLO,W
014C 0224            M           subwf   ACCcLO,W        ;if msb equal then check lsb
014D 1C03            M NOCHK      btfss   STATUS,C        ;carry set if c>a
014E 2956            M           goto    NOGO
014F 0820            M           movf    ACCaLO,W        ;c-a into c
0150 02A4            M           subwf   ACCcLO, F
0151 1C03            M           btfss   STATUS,C
0152 03A5            M           decf    ACCcHI, F
0153 0821            M           movf    ACCaHI,W
0154 02A5            M           subwf   ACCcHI, F
0155 1403            M           bsf     STATUS,C        ;shift a 1 into b (result)
0156 0DA2            M NOGO       rlf     ACCbLO, F
0157 0DA3            M           rlf     ACCbHI, F
                     M ;
                  00144           divMac
  0000               M           LOCAL   NOCHK
  0000               M           LOCAL   NOGO
                     M ;
0158 1003            M           bcf     STATUS,C
0159 0DA6            M           rlf     ACCdLO, F
015A 0DA7            M           rlf     ACCdHI, F
015B 0DA4            M           rlf     ACCcLO, F
015C 0DA5            M           rlf     ACCcHI, F
015D 0821            M           movf    ACCaHI,W
015E 0225            M           subwf   ACCcHI,W        ;check if a>c
015F 1D03            M           btfss   STATUS,Z
0160 2963            M           goto    NOCHK
0161 0820            M           movf    ACCaLO,W
0162 0224            M           subwf   ACCcLO,W        ;if msb equal then check lsb
0163 1C03            M NOCHK      btfss   STATUS,C        ;carry set if c>a
0164 296C            M           goto    NOGO
0165 0820            M           movf    ACCaLO,W        ;c-a into c
0166 02A4            M           subwf   ACCcLO, F
0167 1C03            M           btfss   STATUS,C
0168 03A5            M           decf    ACCcHI, F
0169 0821            M           movf    ACCaHI,W
016A 02A5            M           subwf   ACCcHI, F
016B 1403            M           bsf     STATUS,C        ;shift a 1 into b (result)
016C 0DA2            M NOGO       rlf     ACCbLO, F
016D 0DA3            M           rlf     ACCbHI, F
                     M ;
                  00145           divMac
  0000               M           LOCAL   NOCHK
  0000               M           LOCAL   NOGO
                     M ;
016E 1003            M           bcf     STATUS,C
016F 0DA6            M           rlf     ACCdLO, F
0170 0DA7            M           rlf     ACCdHI, F
0171 0DA4            M           rlf     ACCcLO, F
0172 0DA5            M           rlf     ACCcHI, F
0173 0821            M           movf    ACCaHI,W
0174 0225            M           subwf   ACCcHI,W        ;check if a>c
0175 1D03            M           btfss   STATUS,Z
0176 2979            M           goto    NOCHK
0177 0820            M           movf    ACCaLO,W
0178 0224            M           subwf   ACCcLO,W        ;if msb equal then check lsb
0179 1C03            M NOCHK      btfss   STATUS,C        ;carry set if c>a
017A 2982            M           goto    NOGO
017B 0820            M           movf    ACCaLO,W        ;c-a into c
017C 02A4            M           subwf   ACCcLO, F
017D 1C03            M           btfss   STATUS,C
017E 03A5            M           decf    ACCcHI, F
017F 0821            M           movf    ACCaHI,W
0180 02A5            M           subwf   ACCcHI, F
0181 1403            M           bsf     STATUS,C        ;shift a 1 into b (result)
0182 0DA2            M NOGO       rlf     ACCbLO, F
```

```
0183 0DA3            M          rlf     ACCbHI, F
                    M ;
                    00146      divMac
  0000              M          LOCAL   NOCHK
  0000              M          LOCAL   NOGO
                    M ;
0184 1003           M          bcf     STATUS,C
0185 0DA6           M          rlf     ACCdLO, F
0186 0DA7           M          rlf     ACCdHI, F
0187 0DA4           M          rlf     ACCcLO, F
0188 0DA5           M          rlf     ACCcHI, F
0189 0821           M          movf    ACCaHI,W
018A 0225           M          subwf   ACCcHI,W        ;check if a>c
018B 1D03           M          btfss   STATUS,Z
018C 298F           M          goto    NOCHK
018D 0820           M          movf    ACCaLO,W
018E 0224           M          subwf   ACCcLO,W        ;if msb equal then check lsb
018F 1C03           M NOCHK    btfss   STATUS,C        ;carry set if c>a
0190 2998           M          goto    NOGO
0191 0820           M          movf    ACCaLO,W        ;c-a into c
0192 02A4           M          subwf   ACCcLO, F
0193 1C03           M          btfss   STATUS,C
0194 03A5           M          decf    ACCcHI, F
0195 0821           M          movf    ACCaHI,W
0196 02A5           M          subwf   ACCcHI, F
0197 1403           M          bsf     STATUS,C        ;shift a 1 into b (result)
0198 0DA2           M NOGO     rlf     ACCbLO, F
0199 0DA3           M          rlf     ACCbHI, F
                    M ;
                    00147      divMac
  0000              M          LOCAL   NOCHK
  0000              M          LOCAL   NOGO
                    M ;
019A 1003           M          bcf     STATUS,C
019B 0DA6           M          rlf
ACCdLO, F
019C 0DA7           M          rlf     ACCdHI, F
019D 0DA4           M          rlf     ACCcLO, F
019E 0DA5           M          rlf     ACCcHI, F
019F 0821           M          movf    ACCaHI,W
01A0 0225           M          subwf   ACCHI,W         ;check if a>c
01A1 1D03           M          btfss   STAUS,Z
01A2 29A5           M          goto    NOCHK
01A3 0820           M          movf    ACCaLO,W
01A4 0224           M          subwf   ACCcLO,W        ;if msb equal then check lsb
01A5 1C03           M NOCHK    btfss   STAUS,C         ;carry set if c>a
01A6 29AE           M          goto    NOGO
01A7 0820           M          movf    ACCaLO,W        ;c-a into c
01A8 02A4           M          subwf   ACCcLO, F
01A9 1C03           M          btfss   STATUS,C
01AA 03A5           M          decf    ACCHI, F
01AB 0821           M          movf    ACCaHI,W
01AC 02A5           M          subwf   ACCHI, F
01AD 1403           M          bsf     STAUS,C         ;shift a 1 into b (result)
01AE 0DA2           M NOGO     rlf     ACCbLO, F
01AF 0DA3           M          rlf     ACCbHI, F
                    M ;
                    00148      divMac
  0000              M          LOCAL   NOCHK
  0000              M          LOCAL   NOGO
                    M ;
01B0 1003           M          bcf     STATUS,C
01B1 0DA6           M          rlf     ACCdLO, F
01B2 0DA7           M          rlf     ACCHI, F
01B3 0DA4           M          rlf     ACCcLO, F
01B4 0DA5           M          rlf     ACCcHI, F
01B5 0821           M          movf    ACCaHI,W
```

```
01B6 0225          M        subwf   ACCHI,W        ;check if a>c
01B7 1D03          M        btfss   STATUS,Z
01B8 29BB          M        goto    NOCHK
01B9 0820          M        movf    ACCaLO,W
01BA 0224          M        subwf   ACCcLO,W       ;if msb equal then check lsb
01BB 1C03          M NOCHK  btfss   STATUS,C       ;carry set if c>a
01BC 29C4          M        goto    NOGO
01BD 0820          M        movf    ACCaLO,W       ;c-a into c
01BE 02A4          M        subwf   ACCcLO, F
01BF 1C03          M        btfss   STATUS,C
01C0 03A5          M        decf    ACCcHI, F
01C1 0821          M        movf    ACCaHI,W
01C2 02A5          M        subwf   ACCcHI, F
01C3 1403          M        bsf     STATUS,C       ;shift a 1 into b (result)
01C4 0DA2          M NOGO   rlf     ACCbLO, F
01C5 0DA3          M        rlf     ACCbHI, F
                   M ;
                 00149      divMac
  0000             M        LOCAL   NOCHK
  0000             M        LOCAL   NOGO
                   M ;
01C6 1003          M        bcf     STATUS,C
01C7 0DA6          M        rlf     ACCdLO, F
01C8 0DA7          M        rlf     ACCdHI, F
01C9 0DA4          M        rlf     ACCcLO, F
01CA 0DA5          M        rlf     ACCcHI, F
01CB 0821          M        movf    ACCaHI,W
01CC 0225          M        subwf   ACCcHI,W       ;check if a>c
01CD 1D03          M        btfss   STATUS,Z
01CE 29D1          M        goto    NOCHK
01CF 0820          M        movf    ACCaLO,W
01D0 0224          M        subwf   ACCcLO,W       ;if msb equal then check lsb
01D1 1C03          M NOCHK  btfss   STATUS,C       ;carry set if c>a
01D2 29DA          M        goto    NOGO
01D3 0820          M        movf    ACCaLO,W       ;c-a into c
01D4 02A4          M        subwf   ACCcLO, F
01D5 1C03          M        btfss   STATUS,C
01D6 03A5          M        decf    ACCcHI, F
01D7 0821          M        movf    ACCaHI,W
01D8 02A5          M        subwf   ACCcHI, F
01D9 1403          M        bsf     STATUS,C       ;shift a 1 into b (result)
01DA 0DA2          M NOGO   rlf     ACCbLO, F
01DB 0DA3          M        rlf     ACCbHI, F
                   M ;
                 00150      divMac
  0000             M        LOCAL   NOCHK
  0000             M        LOCAL   NOGO
                   M ;
01DC 1003          M        bcf     STATUS,C
01DD 0DA6          M        rlf     ACCdLO, F
01DE 0DA7          M        rlf     ACCdHI, F
01DF 0DA4          M        rlf     ACCcLO, F
01E0 0DA5          M        rlf     ACCcHI, F
01E1 0821          M        movf    ACCaHI,W
01E2 0225          M        subwf   ACCcHI,W       ;check if a>c
01E3 1D03          M        btfss   STATUS,Z
01E4 29E7          M        goto    NOCHK
01E5 0820          M        movf    ACCaLO,W
01E6 0224          M        subwf   ACCcLO,W       ;if msb equal then check lsb
01E7 1C03          M NOCHK  btfss   STATUS,C       ;carry set if c>a
01E8 29F0          M        goto    NOGO
01E9 0820          M        movf    ACCaLO,W       ;c-a into c
01EA 02A4          M        subwf   ACCcLO, F
01EB 1C03          M        btfss   STATUS,C
01EC 03A5          M        decf    ACCcHI, F
01ED 0821          M        movf    ACCaHI,W
01EE 02A5          M        subwf   ACCcHI, F
```

```
01EF 1403            M        bsf     STATUS,C         ;shift a 1 into b (result)
01F0 0DA2            M NOGO   rlf     ACCbLO, F
01F1 0DA3            M        rlf     ACCbHI, F
                     M ;
               00151 ;
               00152    IF    SIGNED
               00153        btfss   sign,MSB         ; check sign if negative
               00154        retlw   0
               00155        goto    neg_B            ; negate ACCa ( -ACCa -> ACCa )
               00156    ELSE
01F2 3400      00157        retlw   0
               00158    ENDIF
               00159 ;
               00160 ;*********************************************************************
               00161 ;  Assemble this section only if Signed Arithmetic Needed
               00162 ;
               00163     IF    SIGNED
               00164 ;
               00165 S_SIGN  movf    ACCaHI,W
               00166        xorwf   ACCbHI,W
               00167        movwf   sign
               00168        btfss   ACCbHI,MSB       ; if MSB set go & negate ACCb
               00169        goto    chek_A
               00170 ;
               00171        comf    ACCbLO, F        ; negate ACCb
               00172        incf    ACCbLO, F
               00173        btfsc   STATUS,Z
               00174        decf    ACCbHI, F
               00175        comf    ACCbHI, F
               00176 ;
               00177 chek_A  btfss   ACCaHI,MSB       ; if MSB set go & negate ACCa
               00178        retlw   0
               00179        goto    neg_A
               00180 ;
               00181    ENDIF
               00182 ;
               00183 ;
               00184 ;
               00185    ifdef XDCR
               00186 ;
               00187 ; This file has been included, do not have test program.
               00188 ;
               00189    else
               00190 ;
               00191 ;*********************************************************************
               00192 ;                    Test Program
               00193 ;*********************************************************************
               00194 ;   Load constant values to ACCa & ACCb for testing
               00195 ;
               00196 main   movlw   1
               00197        movwf   ACCaHI
               00198        movlw   0FF              ; loads ACCa = 01FF
               00199        movwf   ACCaLO
               00200 ;
               00201        movlw   07F
               00202        movwf   ACCbHI
               00203        movlw   0FF              ; loads ACCb = 7FFF
               00204        movwf   ACCbLO
               00205 ;
               00206        call    D_divF           ; remainder in ACCc. Here ACCb =0040 &
               00207                                 ; ACCc=003F
               00208 self   goto    self
               00209 ;
               00210 ;      org     PIC54
               00211 ;      LIST    p=16c54
               00212 ;      goto    main
               00213 ;**************************************************************
```

```
          00214 ;
          00215
          00216     endif
          00217 ;
          00218 ;     END ; END directive MUST be commented out if this file is included.
          00129        LIST
          00130
          00158        LIST
          00159
          00160        end
```

MPASM 01.40 Released          XDCR.ASM   1-22-1997  10:55:26          PAGE 15
MEMORY USAGE MAP ('X' = Used,  '-' = Unused)

```
0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXX------
0080 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
00C0 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0100 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0140 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0180 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
01C0 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0200 : XXXXXX---------- ---------------- ---------------- ----------------
```

All other memory blocks unused.

Program Memory Words Used:   448
Program Memory Words Free:  3648


```
Errors   :      0
Warnings :      0 reported,     0 suppressed
Messages :      7 reported,     0 suppressed
```

# WORLDWIDE SALES & SERVICE

## AMERICAS

**Corporate Office**
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602-786-7200 Fax: 602-786-7277
*Technical Support:* 602 786-7627
*Web:* 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.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 972-991-7177 Fax: 972-991-8588

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

**Los Angeles**
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 714-263-1888 Fax: 714-263-1338

**New York**
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516-273-5305 Fax: 516-273-5335

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

**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
RM 3801B, Tower Two
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

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

**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, Hongiao District
Shanghai, PRC 200335
Tel: 86-21-6275-5700
Fax: 86 21-6275-5060

**Singapore**
Microchip Technology Taiwan
Singapore Branch
200 Middle Road
#10-03 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 2-717-7175 Fax: 886-2-545-0139

## EUROPE

**United Kingdom**
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44-1628-851077 Fax: 44-1628-850259

**France**
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
2 Rue du Buisson aux Fraises
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 Müchen, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

**Italy**
Arizona Microchip Technology SRL
Centro Direzionale Colleone
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-39-6899939 Fax: 39-39-6899883

## JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81-4-5471- 6166 Fax: 81-4-5471-6122

5/8/97

**MICROCHIP**