# Technique to Calculate Day of Week

*Author: Tan Beng Hai*
*Microchip Technology Inc.*

## INTRODUCTION

Basically, there are two kinds of electronic systems that come with a built-in calendar. The first kind of system is used mainly to display a calendar for a user's convenience. Examples of these systems are digital watches, computers, VCRs and TVs with on-screen display features. The second kind of system is required to know whether a given date is a weekend or weekday. Examples of such a system are multi-rate meters (such as a phone bill meter) and electronic pricing systems, where the weekend rate and the weekday rate are different.

In order to build an electronic calendar into the system, the designer needs to write a piece of software that will be able to determine the day (Sunday, Monday, ….Saturday) of the week when a date is input into the system. This routine is the basic component of an electronics calendar.

This Technical Brief provides a technique to find the exact day for a given date input.

## THEORY OF CALCULATION

The method used to calculate the day of week is a straight forward and simple one. This method makes use of 31 December 1989 as a reference point. The reason why this date was chosen as a reference point is because it was on the last day of the week (Sunday), and also on the last day of a year (this makes it easy to calculate the number of days since the next day/date will be first/first). One day after this date was Monday, and two days after this date was Tuesday, and so on.

The date given will be N days after the 31st December 1989, and if the number N is divided by the number of days in a week (7), the return will correspond to a specific day of the week. For this application, 0 corresponds to Sunday, 1 to Monday, and so on till 6 corresponds to Saturday. Therefore, when a date is given, the number of days from the date given after the 31 December can be calculated. Then the division of the number by 7 will give a remainder, which will correspond to the day of the week that the system required.

## DESCRIPTION OF SOFTWARE

This application note provides two routines for the calculation of the day of the week. One is written in ANSI C and the other is written in assembly language using a PIC16C54 microcontroller.

In these routines, the number of days after 31 December 1989 is calculated and stored in a register called `AccValue`. There are a total of three steps involved in getting the number of days.

The first step is to find out the difference in years, and convert the difference into days. A 16-bit counter, `TempYear`, is used as a temporary counter and is initialized to the year 1990. The routine will keep comparing the contents of `TempYear` to the year (`CurrentYear` in the 'C' software) that is input in the software. If the `TempYear` value is less than the year value, `TempYear` will be increased by 1 until the contents of `TempYear` match the year given.

The `AccValue` will be increased by 1 ( instead of 365, because 365MOD7 = 1) or by 2 (if `TempYear` is a leap year) for each comparison in which the `TempYear` value is less than the year value.

The second step involves calculating the number of days that have elapsed between the first day (inclusive) of that year and the first day of the month given. The number of days elapsed is calculated and pre-stored in a table. This value is retrieved with respect to the input month and added to the `AccValue`.

The way to calculate the pre-stored value is as follows; for January there is less than one month elapsed, therefore the value stored is 0. For February, the month passed is only 1 (January) and the value stored is 31MOD7, which is 3. For March, 2 months have passed, (January and February), so the value stored is (28+31)MOD7, which is 3.

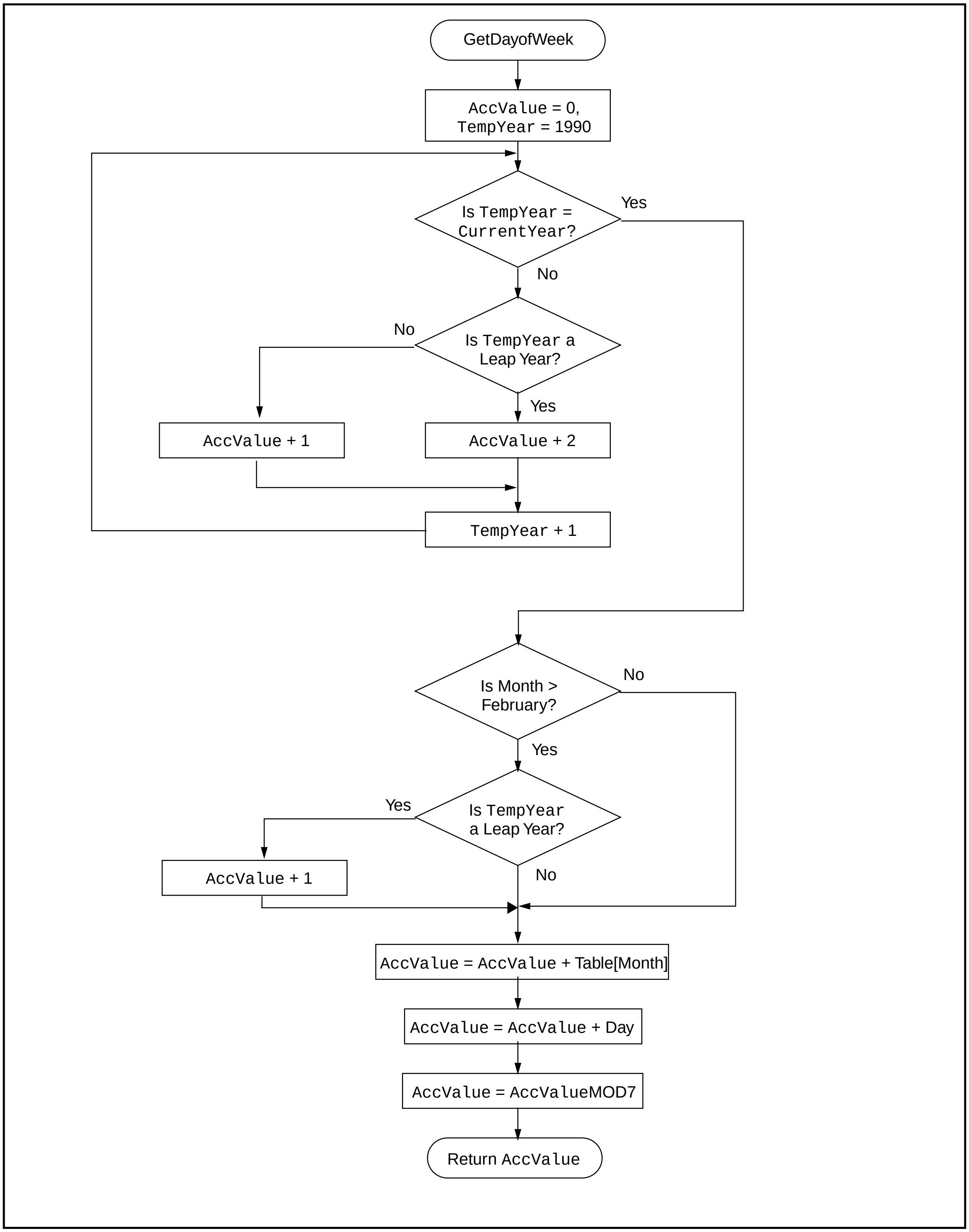
The third step involves adding the day given to the `AccValue`. The `AccValue` is then divided by 7, and the remainder gives the result corresponding to the day of week.

Figure 1 is a flowchart of the software routine.

The software here will only work if the input given ranges from 1 January 1990 to 31 December 2099.

The software does not check for the use of errorneous dates such as 29 February 1998, or 32 March 1999.

FIGURE 1: APPLICATION FLOWCHART



## APPENDIX A: PIC16C54 ASSEMBLY CODE

```
      ListP=16C54
;
;**********************************************************
;
PCL          equ    02h
STATUS       equu   03h
;
#define      Z      STATUS,2
#define      C      STATUS,0
;
YearHi       equ    10h             ;Store the Upper byte of Year
YearLw       equ    11h             ;Store the Lower byte of year
Month        equ    12h             ;Store the Month(1 for January, 2 for February and so on)
Day          equ    13h             ;Store the Day
AccValue     equ    14h
TempA        equ    15h
TempB        equ    16h
TempYearHi   equ    15h
TempYearLw   equ    16h
Lbyte        equ    17h
Hbyte        equ    18h
Ltemp        equ    19h
Htemp        equ    1Ah
Temp         equ    1Bh
;
      org 0x00
;**********************************************************
; Test program for GetDayofWeek
; The End Result willbe  stored in AccValue
;**********************************************************
;
main         movlw  19h             ;Set Date as 21 September 1998
             movwf  TempA
             movlw  98h
             movwf  TempB
             movlw  9
             movwf  Month
             movlw  D'21'
             movwf  Day
;
             call   BCDtoBin        ;Convert the 4-digit BCD Year to 16bit int value
;
             movfw  Lbyte
             movwf  YearLw
;
             movfw  Hbyte
             movwf  YearHi
;
             call   GetDayofWeek    ;Calculate Day
;
Loop  gotoLoop
;
;**********************************************************
; Accumulated Value for Month
; This routine return the remainder value when
; number of days summed up and divided by 7
;**********************************************************
;
GetMonthValueaddwfPCL,1
             nop
             retlw  0               ;January
             retlw  3               ;February, %(31/7) = 3
             retlw  3               ;March, Remaider for (3+28)/7
             retlw  6               ;April, (3+31)/7
             retlw  1               ;May, (6+30)/7
```

```
                 retlw  4              ;June, (1+31)/7
                 retlw  6              ;July,%(4+30)/7= 6
                 retlw  2              ;August, %(6+31/7) = 2
                 retlw  5              ;September, %(2+31)/7=4
                 retlw  0              ;October, %(5+30)/7=6
                 retlw  3              ;November, (0+31)/7=3
                 retlw  5              ;December, (3+30)/7=5
;
;**********************************************************
;                Routine : GetDayofWeek
;**********************************************************
;
GetDayofWeekclrfAccValue
                 call   CheckValidInput
                 btfss  Z              ;Is input Valid ?
                 retlw  08
;
                 movlw  0C6h           ;Set the Temp counter to 1990(Decimal)
                 movwf  TempYearLw
                 movlw  07
                 movwf  TempYearHi
;
GetDay_0         call   CompYear       ;Check is Year > Temp counter
                 btfsc  Z
                 goto   GetDay_1       ;Year = Temp
;
;Year is > Temp Year
;
                 incf   AccValue,1
                 call   IsLeapYear     ;Check is TempYear = Leap Year
                 btfsc  Z              ;Is Leap Year ?
                 incf   AccValue,1     ;Yes !
;
                 call   IncTemp
                 goto   GetDay_0
;
GetDay_1         movlw.3
                 subwf  Month,W        ;Check is Month > February ?
                 btfss  C
                 goto   GetDay_2       ;No !
;
                 call   IsLeapYear     ;Check for Leap year
                 btfss  Z
                 goto   GetDay_2       ;Not Leap Year!
                 incf   AccValue,1
;
GetDay_2         movf   Month,w
                 call   GetMonthValue
                 addwf  AccValue,1     ;Sum the AccValue with Month Value
                 movf   Day,w   ;
                 addwf  AccValue,1     ;Sum the AccValue with Day Value
                 call   Modula_7       ;AccValue%7
                 retlw  0
;
;**********************************************************
;                Routine: Check Valid Input for 1990-2099
;                (equivalent to 07C6h - 0833h)
;                If Input Valid, return Valid = 1
;                Else Valid = 0
;**********************************************************
;
CheckValidInput  movlw  07h
                 subwf  YearHi,w
                 btfss  C
                 goto   NotValid       ;YearHi is < 07h
;
```

```
             btfss  Z
             goto   CheckValid_0   ;YearHi not equal to 07h
;
;YearHi is = 07h Check For Year low
;
             movlw  0C6h
             subwf  YearLw,w
             btfss  C
             goto   NotValid       ;YearLw is <90
;
ValidYear    bsf    Z
             retlw  0              ;Year is valid
;
;YearHi is greater 07h, check for YearHi=08h
;
CheckValid_0 movlw  08h
             subwf  YearHi,w
             btfss  Z
             goto   NotValid       ;YearHi is > 20
;
;YearHi is 20, check for YearLw < or = 33
;
             movlw  34h
             subwf  YearLw,w
             btfss  C              ;is YearLw end with Hex value ?
             goto   ValidYear      ;Not a valid value
;
NotValid     bcf    Z
             retlw  0
;
;**********************************************************
;            Routine: Modula_7
;            Register: ACC
;            Output: Remaider of Acc/7
;**********************************************************
;
Modula_7     movlw  .7
             subwf  AccValue,w
             btfss  C
             goto   Modula_70
;
;Contents of Acc > 7
;
             movlw  .7
             subwf  AccValue,1
             goto   Modula_7
;
Modula_70    movf   AccValue,w
             movwf  Temp
             retlw  0
;
;**********************************************************
;            Routine : CompYear
;            ReturnZ=1 if Year > TempYear
;            Else return 0
;**********************************************************
;
CompYear     movf   YearHi,w
             subwf  TempYearHi,w
             btfss  Z
             retlw  0              ;YearHi > TempYearHi
;
             movfw  YearLw
             subwf  TempYearLw,w
             retlw  0
;
```

```
;**********************************************************
;                Routine: IsLeapYear
;                Return Z=1 if TempYear is Leap year
;                Else Return 0
;**********************************************************
;
IsLeapYear       btfsc  TempYearLw,0
                 goto   NotLeapYear
;
                 btfsc  TempYearLw,1
                 goto   NotLeapYear
;
                 bsf    Z
                 retlw  0
;
NotLeapYear      bcf    Z
                 retlw  0
;
;**********************************************************
;                Routine: IncTemp
;                Increment The Temporary Counter
;**********************************************************
;
IncTemp          incfsz TempYearLw,1
                 retlw  0
;
                 incf   TempYearHi,1
                 retlw  0
;
;**********************************************************
;                Routine : BCDtoBin
;                This routine convert 4-Digit BCD value(D3D2D1D0)
;                into 16Bit Binary code
;                input : 2 digit High Byte is stored in TempA
;                      2 digit Low byte is store in TempB
;                Output: Hbyte:Lbyte
;                For more on the BCD to Bin conversion please refer
;                to AN544
;**********************************************************
;
BCDtoBin         clrf   Htemp
                 clrf   Ltemp
                 clrf   Lbyte
                 clrf   Hbyte
;
                 swapf  TempA,w        ;D3*10
                 call   Mpy10
;
                 movfw  TempA          ;[(D3*10)+D2]*10
                 call   Mpy10
;
                 swapf  TempB,w        ;{[(D3*10)+D2]*10+D1}*10
                 call   Mpy10
;
                 movfw  TempB          ;
                 andlw  0x0f
;
                 addwf  Lbyte,1        ;{[(D3*10)+D2]*10+D1}*10+D4
                 btfsc  C
                 incfH  byte,1
                 retlw  0
;
```

```
;**********************************************************
;            Routine : Mpy10
;            This routine multiply the value store in W register by 10
;            Theory : Let say the input is N,
;            1st,  Store the product of 2*N into Temporary register
;            2nd,  Multiply the value N by 8, (8*N) and store in Hbyte and Lbyte
;            3nd,  Sum up the value obtained in 1st and 2nd steps
;            The whole process is equivalent to 2*N+8*N = N(2+8) = 10*N
;**********************************************************
;
Mpy10        andlw  0x0f
             addwf  Lbyte,1        ;2*N and store the product in temp
             btfsc  C
             incf   Hbyte,1
;
             bcf    C
             rlf    Lbyte,w
             movwf  Ltemp
             rlf    Hbyte,W
             movwf  Htemp
;
             bcf    C              ;8*N
             rlf    Lbyte,1
             rlf    Hbyte,1
;
             bcf    C
             rlf    Lbyte,1
             rlf    Hbyte,1
;
             bcf    C
             rlf    Lbyte,1
             rlf    Hbyte,1
;
             movfw  Ltemp          ;8*N+2*N =10*N
             addwf  Lbyte,1
             movfw  Htemp
             addwf  Hbyte,1
             retlw  0
;
;**********************************************************
;
     END
```

## APPENDIX B: C IMPLEMENTATION

```
#include<P17C756.H>

#defineOK 1
#defineError8

charGetDayofWeek();
charCheckValidInput(unsigned int);
charIsLeapYear(int);

rom     constunsigned charTable[13]={0,0,3,3,6,1,4,6,2,5,0,3,5};

unsigned intCurrentYear;
unsigned charMonth,Day;

/*********************************************************
* Test Program for the routine GetDayofWeek()
*********************************************************/
voidmain(){

  charTemp;

  CurrentYear = 1998;/*Date : 21 September 1998*/
  Month    = 9;
  Day         = 21;

  Temp = GetDayofWeek();/*Result stored in Temp*/
  do{
  }while(1);
}

/****************************************************************
* GetDayofWeek
* This routine calculate the Day(Sunday, Monday,...Saturday) of
* week  when a Date(year, Month, Day) is given.
* Input :  Year, Month and Day which in this routine is used
* as global variable.
* Output Variable :  0 to 6(which correspond to Sunday to Saturdaday
* respectively) if the input is acceptable, else a value 8 is return
****************************************************************/
charGetDayofWeek(){

  unsigned intTempYear;
  unsigned charAccValue, Temp;

  if(CheckValidInput(CurrentYear)!=OK)/* Return Error if input not Valid*/
      return Error;

  TempYear = 1990;/*Comparation start with year 1990*/
  AccValue = 0;/*Init AccValue to 0*/


/* If TempYear is a leap year AccValue +2, else AccValue+1 */

  while(TempYear != CurrentYear){
      AccValue++;
      if(IsLeapYear(TempYear))
          AccValue++;
      TempYear++;
  }

  if(Month > 2){
      if(IsLeapYear(TempYear)==1)
              AccValue++;
  }
  AccValue += Table[Month];
  AccValue += Day;

  AccValue= AccValue%7;

  return(AccValue);
}

/*************************************************************************
* CheckValidInput
* Return a '1' if the input is within the required range. Else
* return a '0'.
*
* Input Variable : 16Bit Unsigned Int
* Output Varible  : '1' if input ranges between 1990 & 2099 inclusively
***************************************************************************/
charCheckValidInput(unsigned int  Input){

  if(Input>=1990 && Input<=2099 )
      returnOK;
  else
      return  !OK;

}

/*************************************************************************
* IsLeapYear
* Return a '1' if the input is a leap year. Else return a '0'
*
* Input Variable : 16Bit Unsigned Int
* Output Varible  : '1' if is a leap year, else '0'.
***************************************************************************/
charIsLeapYear(int Year){

  Year=Year&0x0003;
  if(Year==0)
      return 1;
  else
      return0;
}
/*************************************************************************/
```

MICROCHIP

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

## EUROPE

### United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh 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 hoj 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 - ler 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

DNV Certification, Inc.
USA
ANSI • RAB
QMS
ACCREDITED
DNV
DNV MSC
The Netherlands
Accredited by the RvA
ISO 9001 / QS-9000
REGISTERED FIRM

*Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer 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, KEELOQ® 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.*

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.