INTRODUCTION

This application note describes a technique used to calibrate the internal oscillator on the PIC12F6XX devices. This technique allows the internal oscillator to be calibrated within ±1%. See the test circuit shown in Figure 1.

This application note is useful in the following applications:

1. High volume production environment.
2. Battery applications using on-board calibration to recalibrate the internal RC oscillator as the battery voltage drops.
3. Applications that are exposed to a varying voltage and temperature range could have intelligent on-board recalibration.

FIGURE 1: TEST CIRCUIT

THEORY OF OPERATION

Equipment Used

- HP54645D Oscilloscope with digital probes
- AGILENT 33120A Function Generator
- HPE3620A Power Supply
- Digital multimeter with frequency counter

Definitions

- **Tolerance** - A firmware specified value indicating the allowed deviation in the measured period of a 5 kHz square wave output from a fixed frequency source.
- **Calibration Counter** - A firmware specified value indicating how many times to perform a calibration.

Author: Mike Rylee
Microchip Technology Inc.
Testing

To test the PIC12F6XX device, the firmware first determines if a calibration was performed. If a calibration was performed, the firmware loads the new value from EEPROM into the OSCCAL register. Otherwise, the original factory value is loaded into the OSCCAL register. If the device is calibrated, a 5 kHz square wave is generated by the PIC12F6XX and output on an I/O pin. By hooking up an oscilloscope or frequency counter to this pin, one can determine how precise the internal oscillator is calibrated by measuring the deviation of the frequency, or period of the square wave being output when compared to 5 kHz. See Figure 2 and Figure 3.

FIGURE 2: MAIN ROUTINE

FIGURE 3: TEST ROUTINE
Calibration

To calibrate the PIC12F6XX device, a 5 kHz 50% duty cycle square wave is injected into an I/O pin on the PIC12F6XX device. The firmware measures the period of the 5 kHz square wave and checks to see if it is within a specified tolerance. If the firmware determines that the PIC12F6XX internal oscillator is running faster than the specified tolerance, then it will decrement the value in the OSCCAL register by 1. If the firmware determines that the PIC12F6XX internal oscillator is running slower than the specified tolerance, then it will increment the value in the OSCCAL register by 1. The PIC12F6XX firmware will repeat the above procedure until the calibration counter reaches 0. After the firmware is finished calibrating the internal oscillator, the new calibration value is stored in EEPROM. See Figure 4 and Figure 5.

**FIGURE 4: CALIBRATE ROUTINE**

```
Start CALIBRATE()

Reset Calibration Counter

Measure 200 µs Period from Function Generator

CALL CHECKPERIOD()

No

Decrement Calibration Counter is Calibration Counter = 0?

Yes

Update EEPROM Flag & Calibration Value

End CALIBRATE()
```

**CONCLUSION**

This application note shows a method for auto-calibration of the PIC12F6XX internal RC oscillator. In particular, using a fixed frequency source allowed calibration of the PIC12F6XX internal RC oscillator to be within ± 1%.
APPENDIX A: PIC12F6XX AUTO-CALIBRATION & TEST PROGRAM

;************************************************************************************************
;PIC12F6XX Autocalibration & Test Program
;
;Version: 1.0
;Date: 10/21/02
;Author: Mike Rylee
;Description: This program calibrates the internal RC oscillator on the PIC12F6XX to the tolerance
;specified in the #define TOLERANCE measurement parameter below. The oscillator is
;calibrated using a 5kHz 50% square wave signal input on GP0 from a function generator.
;The calibration is started by pushing the pushbutton connected to GP2.
;After calibration a 5kHz test signal can be output on GP5 by the pushbutton connected
to GP1 to check the calibration.
;************************************************************************************************

list    p-12f675
#include <p12f675.inc>

__CONFIG  _CP_OFF & _CPD_OFF & _BODEN_OFF & _MCLRE_OFF & _WDT_OFF & _PWRTE_ON &
INTRC_OSC_NOCLKOUT

;GENERAL PURPOSE REGISTERS
COUNTER EQU 0X20

;MEASUREMENT PARAMETERS
#define REFERENCEPERIOD .200 ;Reference Period In Microseconds
#define TOLERANCE .1 ;Tolerance Can Be Tweaked For Oscillator Accuracy
#define CALIBRATIONTIME .10 ;Number Of Times To Measure Signal During Calibration

;MAIN I/O
#define INPUT0 GPIO,0 ;Input Reference Waveform Of 5KHz
#define INPUT1 GPIO,1 ;Test Button
#define INPUT2 GPIO,2 ;Calibrate Button
#define OUTPUT GPIO,5 ;Outputs 5KHz Calibrated Waveform

;EEPROM Definitions
#define CALFLAGADR 0x7E ;Value of 0xA5 => Calibration Has Not Been Performed
#define CALVALADR 0x7F ;Value of 0x5A => Calibration Has Been Performed

;DEBUG
#define LED GPIO,4 ;Debug LED

;Reset Vector

ORG 0x000
call 0x3FF ; Retrieve Factory Calibration Value
BANKSEL OSCCAL ; BANK1
movwf OSCCAL ; Load OSCCAL
goto INIT

;************************************************************************************************
;Initialization
;************************************************************************************************
INIT
movlw b'00001111' ;GP0-Input, GP1-Input, GP2-Input, GP4-Output, GP5-Output
movwf TRISIO
clrf ANSEL
clrf VRCON ;Turn Off VREF
BANKSEL GPIO ;BANK 0
movlw .7 ;Turn Off comparator
movwf CMCON

BANKSEL OPTION_REG ;BANK 1
movlw b'01001000' ;Pull Ups Enabled, Rising Edge, Assigned to WDT, Prescaler is 1:1 WDT
movwf OPTION_REG
bsf WPU,2 ;GP2 - Pullup Enabled
bsf WPU,1 ;GP1 - Pullup Enabled
BANKSEL GPIO ;BANK0
clrf GPIO

;************************************************************************************************
;Main Program - This routine watches button INPUT1 & INPUT2
; - Calls CALIBRATE() or Calls TEST()
;************************************************************************************************
MAIN
btfss INPUT1 ;Check Test Button
goto ONE
btfsc INPUT2 ;Check Calibration Button
goto MAIN
TWO
bsf LED
call CALIBRATE
bcf LED
goto MAIN
ONE
bsf LED
call TEST
bcf LED
goto MAIN

;************************************************************************************************
;Subroutines
;************************************************************************************************
;TEST() - This routine is used to test the OSCCAL value in the PIC12F6XX
; - Checks If Calibration Was Performed
; - Updates OSCCAL Value If Calibration Was Performed
; - Outputs A 5 kHz 50% Square Wave On OUTPUT Until Pushbutton Is Released
;************************************************************************************************
TEST
movlw CALFLAGADR
call EEREAD
sublw 0x5A
btfss STATUS,Z ;Was Calibration Flag Set?
goto STARTTEST ;No Don't Change OscCal
movlw CALVALADR
call EEREAD ;Yes Change The OscCal
BANKSEL OSCCAL ;BANK1
movwf OSCCAL
BANKSEL GPIO ;BANK0

STARTTEST ;The instructions below make a 5kHz 50% Square Wave
Output On GP1 if the device is calibrated
bsf OUTPUT ;1 us
movlw .31 ;Delay 99 us
call DELAYUS ;99+1 = 100 us
bcf OUTPUT ;1us
movlw .30 ;Delay 96 us
call DELAYUS
btfss INPUT1 ;1 us
goto STARTTEST ;2 us ->1+96+1+2 = 100 us
return

;************************************************************************************************
;Calibrate() - Measures A Period From The Input(GP0) Reference Signal
; - Updates Oscal Value
; - Updates E^2
;************************************************************************************************
CALIBRATE
movlw CALIBRATIONTIME
movwf COUNTER ;Calibration Counter

LOW0
btfsc INPUT0 ;Wait To Sample Low Edge #0 (Makes Sure We Are
;Synchronized First Time Through)
goto LOW0

HIGH1
btfss INPUT0 ;Wait To Sample High Edge #1
goto HIGH1
clrf TMR0 ;Start Timer (Timer Will Be Behind By 5us After This
;Instruction)

LOW1
btfsc INPUT0 ;Wait To Sample Low Edge #1
goto LOW1

HIGH2
btfss INPUT0
goto HIGH2
movf TMR0,W ;Stop Timer (Timer Will Be Stopped 3us Late)
addlw .2 ;Timer Is Behind By 2us Total From Start To Stop
call CHECKPERIOD ;See If Oscal Needs To Be Adjusted

LOW2
btfsc INPUT0 ;Wait To Sample Low Edge #2
goto LOW2
decfsz COUNTER,F ;Decrement The Calibration Counter
goto HIGH1
call UPDATE_EE ;Update E^2
return

;************************************************************************************************
;Update_EE - This routine Updates Calibration Flag & Calibration Value
;************************************************************************************************
UPDATE_EE
BANKSEL EEDATA ;BANK1
movlw 0x5A ;Update Calibration Flag
movwf EEDATA
movlw CALFLAGADR
call EEWRITE
movf OSCCAL,W
movwf EEDATA
movlw CALVALADR
call EEWRITE ;Update Calibration Value
BANKSEL GPIO ;BANK0
return

;**********************************************************************************************
;CheckPeriod(W) - This routine computes the difference between the REFERENCEPERIOD and
;                MEASUREDPERIOD
;                - The MEASUREDPERIOD is contained in W when this routine is called
;                - The OSCCAL Is Adjusted Up or Down If The Period Is Outside The Specified
;                Tolerance
;**********************************************************************************************
CHECKPERIOD
sublw REFERENCEPERIOD
btfsc STATUS,Z ;If (ReferencePeriod - MeasuredPeriod = 0) Don't Change
                ;OSCCAL
                return
btfsc STATUS,C ;If (ReferencePeriod - MeasuredPeriod > 0) Oscillator
                ;Could Be Too Fast
                goto RUNNINGFAST ;Else Oscillator Could Be Too Slow
RUNNINGFAST
xorlw 0xFF ;Two's Complement Value
addlw .1
sublw TOLERANCE ;If (Tolerance - (ReferencePeriod - MeasuredPeriod) = 0
                ;Don't Change Osccal
                btfsc STATUS,Z
                return
                goto ADJUSTDOWN ;Else Adjust Osccal Down
RUNNINGSLOW
sublw TOLERANCE ;If (Tolerance - (ReferencePeriod - MeasuredPeriod) = 0
                ;Don't Change Osccal
                btfsc STATUS,Z
                return
                goto ADJUSTUP ;Else Adjust Osccal Up
ADJUSTDOWN
BANKSEL OSCCAL ;BANK1
movlw .4
subwf OSCCAL,F ;Adjust Osccal Down
BANKSEL GPIO ;BANK0
return
ADJUSTUP
BANKSEL OSCCAL ;BANK1
movlw .4
addwf OSCCAL,F ;Adjust Osccal Up
BANKSEL GPIO ;BANK0
return

;**********************************************************************************************
;EEREAD(W) - Address To Read Is Contained In W When This Function Is Called
;**********************************************************************************************
EEREAD
BANKSEL EEADR ;BANK1
movwf EEADR ;Bank
bsf EECON1,RD
movf EEDATA,W
BANKSEL GPIO ;BANK0
return

;**********************************************************************************************
;EEWRITE(W) - Address To Read Is Contained In W When This Function Is Called
;**********************************************************************************************
; - EEDATA Is Loaded Prior To This Function Call
; - BANK1 must be selected before this function is called
;************************************************************************************************

**EEWRITE**

```assembly
movwf EEADR
bsf EECON1, WREN
bcf INTCON, GIE
movlw 0x55
movwf EECON2
movlw 0xAA
movwf EECON2
bsf EECON1, WR

**EECOMPLETE**

btfsc EECON1, WR
goto EECOMPLETE
bcf EECON1, WREN
return
```

;************************************************************************************************

**DELAYUS(W)** - Delay Microseconds
; TotalTime(W) = [(1)+(2)+(1)+(W*3-1)+1+2] * (4/OSC) (This includes the movlw & the call)
; Max Time When W=0xFF, [ 771 Cycles * (4/OSC) ]
; Must Declare COUNTER AS GPR
; W > 0
;************************************************************************************************

**DELAYUS**

```assembly
movwf COUNTER

**LOOP1**

decfsz COUNTER, F
goto LOOP1

nop
return
```

;************************************************************************************************

**EEPROM** - CALFLAGADR - Contains Calibration Flag Value
; CALVALADR - Contains Calibration Value
;************************************************************************************************

```assembly
ORG (0X2100+CALFLAGADR)

DE 0A5H ;Initialize Calibration Flag

DE 000H ;Initialize Calibration Value

END
```
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. 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.

Trademarks

The Microchip name and logo, KEELoQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Accuron, Application Maestro, dsPIC, dsPICDEM, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICkit, PICDEM, PICDEM.net, PowerCal, PowerInfo, PowerMate, PowerTool, rFLAB, rFPIC, Select Mode, SmartSensor, SmartShunt, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2003, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, dsPIC® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.