**INTRODUCTION**

Microchip’s PIC16CXXX microcontroller family are ideally suited to directly interface to a keypad. The high 4-bits of PORTB (RB7:RB4) have internal pull-ups and can trigger a “change on state” interrupt. This interrupt, if enabled, will wake the microcontroller from SLEEP. In most battery powered applications, a microcontroller is exercised when a key is pressed (e.g., in a remote keyless entry system). The life of the battery can be extended by using PIC16CXXX microcontrollers. This is done by putting the PIC16CXXX microcontroller into SLEEP mode for most of the time and wake-up only when a key is pressed.

**IMPLEMENTATION**

Figure 1 depicts an application where four keys are connected to RB7:RB4. Internal pull-ups are used to maintain a high level on these inputs. In this example, LEDs are connected to RB3:RB0. When SW1 is pressed, LED1 is turned on and when SW2 is pressed, LED2 is turned on and so on. The PIC16CXXX is normally in SLEEP mode with the “change on state” interrupt enabled. When SW1 is pressed, RB4 goes low and triggers an interrupt. Since the PIC16CXXX is in SLEEP, it first wakes up and starts executing code at the interrupt vector. Note that if the global interrupt is enabled, the program execution after an interrupt is at the interrupt vector, if the global interrupt is not enabled, the program starts executing the first line of code right after the SLEEP instruction.

After waking up, a 20 - 40 ms de-bounce delay is executed which checks the port for a key hit and, depending on which key is hit, its associated LED is turned on. The LEDs are used purely for demonstration purposes. In a remote keyless entry application, the remote code would be transmitted when the appropriate key is hit.

Figure 2 depicts a 4x4 keypad interface to a PIC16CXXX microcontroller. When using the PIC16CXXX in a keypad application, the internal pull-ups on RB7:RB4 can be enabled, eliminating the need for external pull-up resistors. The series 100Ω resistors are used for Electrostatic Discharge (ESD) protection, and are recommended in keypad interface applications.

**SUMMARY**

The PIC16CXXX is ideally suited to interface directly to a keypad application. Built in pull-up resistors and very low current consumption during sleep make it a very good candidate for battery powered remote operations and applications. Appendix A provides an example of the code.

**Performance:**

- Code Size - 64 words
- RAM Used - 0 bytes

**FIGURE 1:** 4 KEY INTERFACE TO PIC16CXXX

**FIGURE 2:** 4x4 KEYPAD INTERFACE TO PIC16CXXX
APPENDIX A : WAKUP.ASM

MPASM 01.40 Released            WAKUP.ASM   1-16-1997  16:04:19         PAGE  1

LOC  OBJECT CODE     LINE SOURCE TEXT
 VALUE

00001 ;This program demonstrates the wake-up on Keystroke feature of
00002 ;the PIC16C71. Port B pins RB4 - RB7 can be configured as inputs with
00003 ;internal pull up resistors, also the interrupt associated with the
00004 ;change on input on RB4 - RB7 can be set up to wake the chip from
00005 ;sleep. If the global interrupt is enabled just before sleep, the
00006 ;program will vector to the interrupt vector (0004). If not, the chip
00007 ;will continue execution just after the next instruction following
00008 ;sleep. In this example code, port B is initialized to input 4
00009 ;push-buttons at RB4 - RB7. RB0 - RB3 are configured to drive LEDs
00010 ;corresponding to which pushbutton is hit (LED on RB0 when RB4 is
00011 ;hit and so on). Sleep is executed. When any key is hit, the
00012 ;processor wakes up and jumps to the interrupt vector. The
00013 ;corresponding LED is turned on and after the key is released, the
00014 ;whole process is repeated.
00015 ;
00016 ;       Program:          WAKEUP.ASM
00017 ;       Revision Date:          1-16-97         Compatibility with MPASMWIN 1.40
00018 ;
00019 ;************************************************************************
00020 ;************************************************************************
00021 ;
00022 LIST P=16C71
00023 ;
00000002 00024 z       equ     2
00000007 00025 RBPU    equ     7
00000010 00026 temp    equ     10h
00000011 00027 OptionReg equ     1h
00000001 00028 F       EQU     1
00029 ;
00030 include "p16c71.inc"
00001 LIST
00002 ;P16C71.INC Standard Header File, Version 1.00 Microchip Technology
00142 LIST
00031 ;
0000 000032 org     0
0000 2805 00033 goto    start
00034 ;
0004 00035 org     4
0004 2808 00036 goto    ServiceInterrupt
00037 ;
00038 ;
0005 00039 start
0005 2024 00040 call    InitPortB ;initialize port B
0006 00041 loop
00042 ;sleep till key is hit
0006 0000 00043 nop
0007 2806 00044 goto    loop
00045 ;
0008 00046 ServiceInterrupt
0008 180B 00047 btfsc   INTCON,RBIF ;change on rb int?
0009 280D 00048 goto    ServiceWakeup ;yes then service
000A 128B 00049 bcf      INTCON,T0IE ;clear TMR0 int mask
000B 110B 00050 bcf      INTCON,T0IF ;clear flag
000C 0008 00051 return

Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address:
This routine checks which key is hit and lights up the corresponding LED associated with it. eg. RB0's LED when RB4's key is pressed. Finally it waits till all keys have been released before returning form the service routine.

ServiceWakup

bcf INTCON,RBIE ;clear mask
comf PORTB,W ;read PORTB
bcf INTCON,RBIF ;clear flag
call delay16 ;do de-bounce for 16mSecs
comf PORTB,W ;read port B again
andlw B'11110000' ;mask outputs
movwf temp ;save in temp
swapf temp,W ;switch low and high
movwf PORTB ;send as outputs.
call KeyRelease ;check for key release

This sub-routine, waits till all key have been released
In order to save power, the chip is in sleep mode till all keys are released.

KeyRelease

call delay16 ;do debounce
comf PORTB,W ;read PORTB
bcf INTCON,RBIF ;clear flag
bsf INTCON,RBIE ;enable mask
andlw B'11110000' ;clear outputs
btfsc STATUS,z ;key still pressed?
return ;no then return
sleep ;else save power
bcf INTCON,RBIE ;on wake up clear mask
bcf INTCON,RBIE ;clear flag
goto KeyRelease ;try again

delay16 waits for approx 16.4mSecs using TMR0 interrupts
fosc speed is 4Mhz.

InitPortB

bsf STATUS,RP0 ;select bank1
movlw B'00000011' ;Port_A digital I/O
movwf ADCON1 ;
movlw 0 ;
movwf PORTA ;set port a as outputs
movlw B'11110000' ;RB0-RB3 outputs
movwf PORTB ;RB4-RB7 inputs
bcf OptionReg,RBPU ;enable pull up
bcf STATUS,R0 ;select page 0
clrf PORTB ;init port B
clrf PORTA ;make port a all low
bsf PORTA,0 ;make first bit high
bcf INTCON,RBIE ;disable mask
movf PORTB,W ;read port
bcf INTCON,RBIF ;clear flag
bsf INTCON,RBIE ;enable mask
retie ;enable global and return

delay16 waits for approx 16.4mSecs using TMR0 interrupts
fosc speed is 4Mhz.

InitPortB

bsf STATUS,R0 ;select Bank1
movlw B'00000111' ;Port_A digital I/O
movwf PORTA ;set port a as outputs
movlw B'11110000' ;RB0-RB3 outputs
movwf PORTB ;RB4-RB7 inputs
bcf OptionReg,RBPU ;enable pull up
bcf STATUS,R0 ;select page 0
clrf PORTB ;init port B
clrf PORTA ;make port a all low
bsf PORTA,0 ;make first bit high
bcf INTCON,RBIE ;disable mask
movf PORTB,W ;read port
bcf INTCON,RBIF ;clear flag
bsf INTCON,RBIE ;enable mask
retie ;enable global and return

delay16 waits for approx 16.4mSecs using TMR0 interrupts
fosc speed is 4Mhz.
003B 168B 00117 bsf INTCON,T0IE ;enable mask
003C 00118 CheckAgain
003C 1D0B 00119 btfss INTCON,T0IF ;timer overflowed?
003D 283C 00120 goto CheckAgain ;no check again
003E 128B 00121 bcf INTCON,T0IE ;else clear mask
003F 110B 00122 bcf INTCON,T0IF ;clear flag
0040 0008 00123 return
00124 ;
00125 end

MEMORY USAGE MAP ('X' = Used, ' ' = Unused)

0000 : X---XXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : X--------------- ---------------- ---------------- ----------------

All other memory blocks unused.

Program Memory Words Used: 62
Program Memory Words Free: 962

Errors : 0
Warnings : 0 reported, 0 suppressed
Messages : 1 reported, 0 suppressed
Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products 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 PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be 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 product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

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.

Trademarks

The Microchip name and logo, the Microchip logo, FilterLab, KEELOQ, microID, MPLAB, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

dspIC, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, MXDEV, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A.

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.

© 2002, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.