MICROCHIP

AN552

Implementing Wake-up on Key Stroke

INTRODUCTION

Microchip's PIC16CXX family of microcontrollers are ideally suited to directly interface to a keypad. The high 4-bits of PortB (RB4 - RB7) have internal pull-ups and can trigger a "change on port 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 PIC16CXX microcontrollers. This can be done by putting the PIC16CXX 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 RB4 - RB7. Internal pull-ups are used to maintain a high level on these inputs. In this example, LEDs are connected to RB0 - RB3. When SW1 is pressed, LED1 is turned on and when SW2 is pressed, LED2 is turned on and so on. The PIC16CXX is normally in sleep mode with the "change on port state" interrupt enabled. When SW1 is pressed, RB4 goes low and triggers an interrupt. Since the PIC16CXX 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 right after the sleep instruction.

After waking up, a 20 - 40 msec. 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 the PIC16CXX. When using the PIC16CXX in a keypad application, the internal pull-ups on RB4 - RB7 can be enabled eliminating the need for external pull-up resistors. The series 100Ω resistors are used for ESD protection, and are recommended in keypad interface applications.

Author: Stan D'Souza, Logic Products Division

SUMMARY

The PIC16CXX is ideally suited to interface directly to a Keypad application. Built in pull-up resistors and very low sleep current make it a very good candidate for battery powered remote operations and applications.

Performance:

Code Size: 64 words

RAM Locations Used: 0 bytes

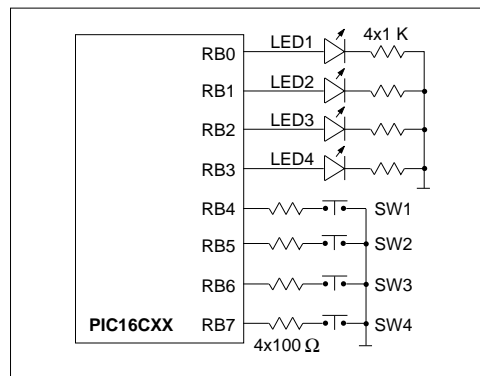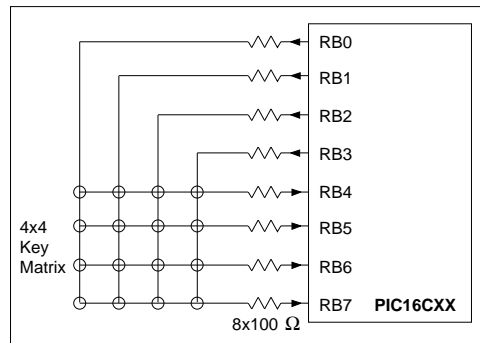**3**

FIGURE 1 - 4 KEY INTERFACE TO PIC16CXX

FIGURE 2 - 4X4 KEYPAD INTERFACE TO PIC16CXX

Implementing Wake-up on Key Stroke

```
LOC   OBJECT CODE    LINE SOURCE TEXT

                     0001 ;This program demonstrates the wake-up on Keystroke feature of the
                     0002 ;PIC16C71. Port B pins RB4 - RB7 can be configured as inputs with
internal
                     0003 ;pull up resistors, also the interrupt associated with the change on
input
                     0004 ;on RB4 - RB7 can be set up to wake the chip from sleep. If the
                     0005 ;global interrupt is enabled just before sleep, the program will vector
to
                     0006 ;the interrupt vector (0004). If not the chip will continue execution
                     0007 ;just after the next instruction following sleep.
                     0008 ;In this example code, the port B is initalized to input 4 push-buttons
at
                     0009 ;RB4 - RB7. RB0 - RB3 are configured to drive LEDs corresponding to
                     0010 ;which push-button is hit (LED on RB0 when RB4 is hit and so on).
                     0011 ;Sleep is executed. When any keys is hit, the processor wakes
                     0012 ;up and jumps to the interrupt vector. The corresponding LED is
                     0013 ;turned on and after the key is released, the whole process is re-
peated.
                     0014 ;
                     0015       LIST P=16C71, F=INHX8M
                     0016 ;
0002                 0017 z       equ     2
0007                 0018 RBPU    equ     7
0010                 0019 temp    equ     10h
0001                 0020 OptionReg  equ    1h
                     0021    include "picreg.equ"
                     0083
                     0084
                     0021
                     0022 ;
                     0023       org     0
0000 2805            0024       goto    start
                     0025 ;
                     0026       org     4
0004 2808            0027       goto    ServiceInterrupt
                     0028 ;
                     0029 ;
                     0030 start
0005 2024            0031       call    InitPortB       ;initalize port B
                     0032 loop
                     0033 ;      sleep                   ;sleep till key is hit
0006 0000            0034       nop
0007 2806            0035       goto    loop
                     0036 ;
                     0037 ServiceInterrupt
0008 180B            0038       btfsc   INTCON,RBIF     ;change on rb int?
0009 280D            0039       goto    ServiceWakup    ;yes then service
000A 128B            0040       bcf     INTCON,RTIE     ;clear RTCC int mask
000B 110B            0041       bcf     INTCON,RTIF     ;clear flag
000C 0008            0042       return
                     0043 ;
                     0044 ;This routine checks which keys is hit and lights up the
                     0045 ;corresponding LED associated with it. eg. RB0's LED when
                     0046 ;RB4's key is pressed. Finally it waits till all keys have
                     0047 ;been released before returning form the service routine.
                     0048 ServiceWakup
000D 118B            0049       bcf     INTCON,RBIE     ;clear mask
000E 0906            0050       comf    PORT_B,w        ;read PORT_B

000F 100B            0051       bcf     INTCON,RBIF     ;clear flag
0010 2035            0052       call    delay16         ;do de-bounce for 16mSecs
0011 0906            0053       comf    PORT_B,w        ;read port B again
0012 39F0            0054       andlw   B'11110000'     ;mask outputs
0013 0090            0055       movwf   temp            ;save in temp
```

```
0014 0E10          0056          swapf    temp,w            ;switch low and high
0015 0086          0057          movwf    PORT_B            ;send as outputs.
0016 2018          0058          call     KeyRelease        ;check for key release
0017 0009          0059          retfie
                   0060 ;
                   0061 ;This sub-routine, waits till all key have been released
                   0062 ;In order to save power, the chip is in sleep mode till
                   0063 ;all keys are released.
                   0064 KeyRelease
0018 2035          0065          call     delay16           ;do debounce
0019 0906          0066          comf     PORT_B,w          ;read PORT_B
001A 100B          0067          bcf      INTCON,RBIF       ;clear flag
001B 158B          0068          bsf      INTCON,RBIE       ;enable mask
001C 39F0          0069          andlw    B'11110000'       ;clear outputs
001D 1903          0070          btfsc    STATUS,z          ;key still pressed?
001E 0008          0071          return                     ;no then return
001F 0063          0072          sleep                      ;else save power
0020 118B          0073          bcf      INTCON,RBIE       ;on wake up clear mask
0021 0906          0074          comf     PORT_B,w
0022 100B          0075          bcf      INTCON,RBIF       ;clear flag
0023 2818          0076          goto     KeyRelease        ;try again
                   0077 ;
                   0078 ;
                   0079 ;This sub-routine, initializes PortB.
                   0080 InitPortB
0024 1683          0081          bsf      STATUS,RP0        ;select bank 1
0025 3003          0082          movlw    B'00000011'       ;Port_A digital I/O
0026 0088          0083          movwf    ADCON1            ;                 /
0027 3000          0084          movlw    0                 ;
0028 0085          0085          movwf    PORT_A            ;set port a as outputs
0029 30F0          0086          movlw    B'11110000'       ;RB0-RB3 outputs
002A 0086          0087          movwf    PORT_B            ;RB4-RB7 inputs
002B 1381          0088          bcf      OptionReg,RBPU    ;enable pull up
002C 1283          0089          bcf      STATUS,RP0        ;select page 0
002D 0186          0090          clrf     PORT_B            ;init port B
002E 0185          0091          clrf     PORT_A            ;make port a all low
002F 1405          0092          bsf      PORT_A,0          ;make first bit high
0030 118B          0093          bcf      INTCON,RBIE       ;disable mask
0031 0806          0094          movf     PORT_B,w          ;read port
0032 100B          0095          bcf      INTCON,RBIF       ;clear flag
0033 158B          0096          bsf      INTCON,RBIE       ;enable mask
0034 0009          0097          retfie                     ;enable global and return
                   0098 ;
                   0099 ;delay16 waits for approx 16.4mSecs using RTCC interrupts
                   0100 ;fosc speed is 4Mhz.
                   0101 delay16
0035 1683          0102          bsf      STATUS,RP0        ;select page 1
0036 3007          0103          movlw    B'00000111'       ;fosc/256 -> RTCC
0037 0081          0104          movwf    OptionReg         ;                 /
0038 1283          0105          bcf      STATUS,RP0        ;select page 0
0039 0181          0106          clrf     RTCC
003A 110B          0107          bcf      INTCON,RTIF       ;clear flag
003B 168B          0108          bsf      INTCON,RTIE       ;enable mask
                   0109 CheckAgain
003C 1D0B          0110          btfss    INTCON,RTIF       ;timer overflowed?
003D 283C          0111          goto     CheckAgain        ;no check again
003E 128B          0112          bcf      INTCON,RTIE       ;else clear mask
003F 110B          0113          bcf      INTCON,RTIF       ;clear flag
0040 0008          0114          return
                   0115 ;
                   0116          end
                   0117
                   0118
                   0119
                   0120
                   0121
                   0122
                   0123
                   0124
```

3

Implementing Wake-up on Key Stroke

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

0000 : X—XXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : X——————— ———————— ———————— ————————

All other memory blocks unused.


Errors   :   0
Warnings :   0
```

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.mchip.com/microhip

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: 708 285-0071 Fax: 708 285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
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

AMERICAS (continued)

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

ASIA/PACIFIC

Hong Kong
Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

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

Singapore
Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan
Microchip Technology
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 0 1628 851077 Fax: 44 0 1628 850259

France
Arizona Microchip Technology SARL
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 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

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

9/22/95