Telephony Solutions: Ring Detection with SX Microcontroller

Appication Note 4
Stephen Holland
February 1999

Introduction

This application note outlines the hardware and software needed to provide telephone ring detection. This software may be used alone or combined with other telephony modules as required.

Hardware

Certain basic hardware is required to properly interface to the telephone network. Figure 1 shows a typical circuit for ring detection - there are many possible variations in requirements based on area and telephone network providers, so check with your network provider first.

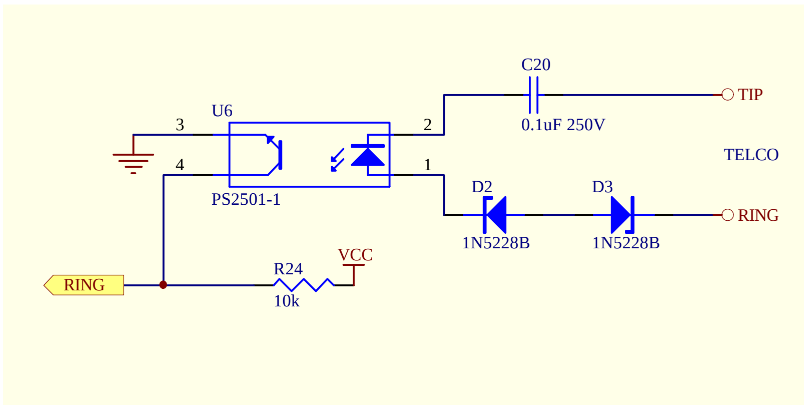

Figure 1.

Software

For software, the requirements are quite simple:

1. Ignore off-hook glitches or line noise
2. Provide notification of ring event

In the example circuit above, the RING input will go low when a ring event occurs. Noise or an off-hook can also cause this input to briefly go low, so to avoid this, while the RING input is low, the 16-bit ring_count register is incremented and the ring_lo_det flag is set. As soon as the RING input goes high again, if the ring_lo_det flag is set the ring_count register is tested to be above a certain count. This count represents the time duration that the RING input was low, with each count representing 1/RTCC seconds. If the count was below a specified duration of time, it must be noise or an off-hook, and the ring_count registers and ring_lo_det flag are cleared. If the count was above the specified duration of time, it must be a ring event, the ringing flag is set, and the ring_count registers and ring_lo_det flag are cleared to look for the next ring event.

The same method could also be used for detection of distinctive ring patterns by incrementing another count register instead of just setting the ringing flag.

```
; Filename: Ring_detect.src
; Author:   Stephen Holland
;                       Applications Engineers
;                       Scenix Semiconductor Inc.
; Revision: 1.0
; Date:                 Jan. 14, 1999
; Part:                 SX28AC rev. 2.5
; Freq:                 50Mhz
; Compiled using Parallax SX-Key software v1.01
;
; Ring detection routine for Scenix SX Microcontrollers. This code is
; designed to be used in the interrupt service routine for passive detection of
; a ring event.  A ring detection is indicated by the setting of the ’ringing’
; flag.
;
;**************************************************************
; Device
;**************************************************************
            device      pins28,pages4,banks8,oschs
            device      turbo,stackx,optionx
            id          ’Ring_Det’
            reset       reset_entry
            freq        50_000_000
;
; Watches
;
            watch       ring,1,ubin
            watch       ringing,1,ubin

            watch       ring_count,16,uhex

;**************************************************************
; Variables
;**************************************************************
;**************************************************************
; Global variables
;**************************************************************
            org         8
temp        ds          1
flags       ds          1
;
timer_flag  =           flags.0     ;Indicates timer expired
ring_det_en =           flags.1     ;Enables ring detection
ringing     =           flags.2     ;Indicates a (confirmed) ring is happening
ring_lo_det =           flags.3     ;Indicates that the ring line has been low recently

;**************************************************************
; Bank 0 variables
;**************************************************************
            org         $10

timers      =           $
timer_accl  ds          1
timer_acch  ds          1

ring_bank   =           $
ring_count  ds          2
```

```
;*************************************************************
; Bank 1 variables
;*************************************************************
            org         $30
;*************************************************************
; Bank 2 variables
;*************************************************************
            org         $50
;*************************************************************
; Declarations
;*************************************************************

int_period  =           163         ;period between interrupts

; Pin assignments
led_pin     =           rb.0
ring        =           rb.3

;*************************************************************
; Interrupt routine - virtual peripherals
;*************************************************************
            org         0
interrupt                           ;3 it takes 3 cycles to get an interrupt

;*************************************************************
; Timers
;*************************************************************
; Timer 1
timer       bank        timers                  ;1
            add         timer_accl,#1           ;2 add timer_accl+carry(=1)
            sc
            jmp         :timer_out

            add         timer_acch,#1
            sc                                  ;1
            jmp         :timer_out
            setb        timer_flag              ;1
:timer_out                                      ;=7
;*************************************************************
; Ring Detection
;*************************************************************
;           jnb         ring_det_en,ring_det_out
            jb          ring,:ring_high

:ring_low   setb        ring_lo_det             ;Set ring_lo_det to indicate that a
                                                ;ring event has started
            inc         ring_count+0            ;Increment 16-bit ring_count register
            snz
            inc         ring_count+1
            jmp         ring_det_out            ; exit

:ring_high  ;After a ring has been high for a specified amount of time,
            ; check to see if ring_count is above a specified count.
            ; This is to resist the detection of noise or off-hook glitches.
            jnb         ring_lo_det,ring_det_out
            cjb         ring_count+1,#$50,ring_det_out
            setb        ringing
            clrb        ring_lo_det             ;Reset ring_lo_det
            clr         ring_count              ;Reset ring_count
            clr         ring_count+1
ring_det_out
;*************************************************************
interrupt_out
            mov     w,#-int_period     ;1;interrupt every 'int_period' clocks
            retiw                      ;3;exit interrupt
*************************************************************
; Reset entry
;*************************************************************
reset_entry mov         m,#$0f
            mov          ra,#%0110              ;init ra
            mov         !ra,#%0010              ;ra0-1 = input, ra2-3 = output
            mov          rb,#%00000000          ;init rb
            mov         !rb,#%00001110          ;rb1-3 = input, rb0,rb4-7 = output
            mov          rc,#%00000000          ;init rc
            mov         !rc,#%01111101          ;rc0,rc2-7 = input, rc1 = output
            mov         m,#$0f                  ;Point MODE register back to ports
            clr         fsr                     ;reset all ram banks

:loop       setb        fsr.4
            clr         ind
            ijnz        fsr,:loop

            clr         flags                   ;Clear flags registers

            mov         !option,#%00011111;enable wreg and rtcc interrupt

            jmp         @main                   ;Jump to main code

;*************************************************************
; Subroutines
;*************************************************************
            org         $200
;*************************************************************
;*************************************************************
            org         $400
;*************************************************************
;*************************************************************
            org         $600
;*************************************************************
;*************************************************************
; Main
;*************************************************************
main
            bank        ring_bank
            clr         ring_count
            clr         ring_count+1
            clrb        ring_lo_det
            clrb        ringing
; Main loop
main_loop
```

```
;**************************************************************
get_ring    bank        ring_bank
            jb          ringing,:send_ring
            jmp         get_ring_done

:send_ring  ;This is where ring event would be announced
            ;In this example, we just flash the LED to indicate which ring pattern
            ;was detected

            mov         temp,#10
:again      setb        led_pin
            bank        timers
            mov         timer_accl,#$00         ;200mS
            mov         timer_acch,#$85         ;--//--
            clrb        timer_flag
            jnb         timer_flag,$
            clrb        led_pin
            bank        timers
            mov         timer_accl,#$00         ;200mS
            mov         timer_acch,#$85         ;--//--
            clrb        timer_flag
            jnb         timer_flag,$
            djnz        temp,:again

:send_done  clrb        ringing
get_ring_done

            jmp         main_loop
;**************************************************************
; End
```