

Application Note

QUICK CS6420 SPEAKERPHONE INTERFACE

This application note describes a quick interface for the Crystal Semiconductor CS6420 speakerphone device to an existing analog speakerphone. The electrical connections to the speakerphone take advantage of standard signals present in all analog speakerphones. Connections are made to the Tip and Ring of an analog telephone line and the CS6420 uses the speakerphone's speaker and microphone for its acoustic interface. Figure 1 illustrates the system connections.

Minor modifications are made to the speakerphone to gain access to certain signals and wires. Six signals are connected between the target speaker-

phone and the CS6420 circuit card. The phone's speaker and microphone wires are disconnected from the speakerphone's circuitry and brought to the CS6420 circuit card. This provides the CS6420's acoustic interface and accounts for four of the six signals (SPKR+, SPKR-, MIC+ and MIC-). With the CS6420 connected to the speaker and microphone, hands-free full duplex operation and speaker volume control are available. The Tip and Ring signals in the speakerphone are not disconnected from the speakerphone's circuitry. The connections are made in parallel keeping the speakerphone's keypad functional.

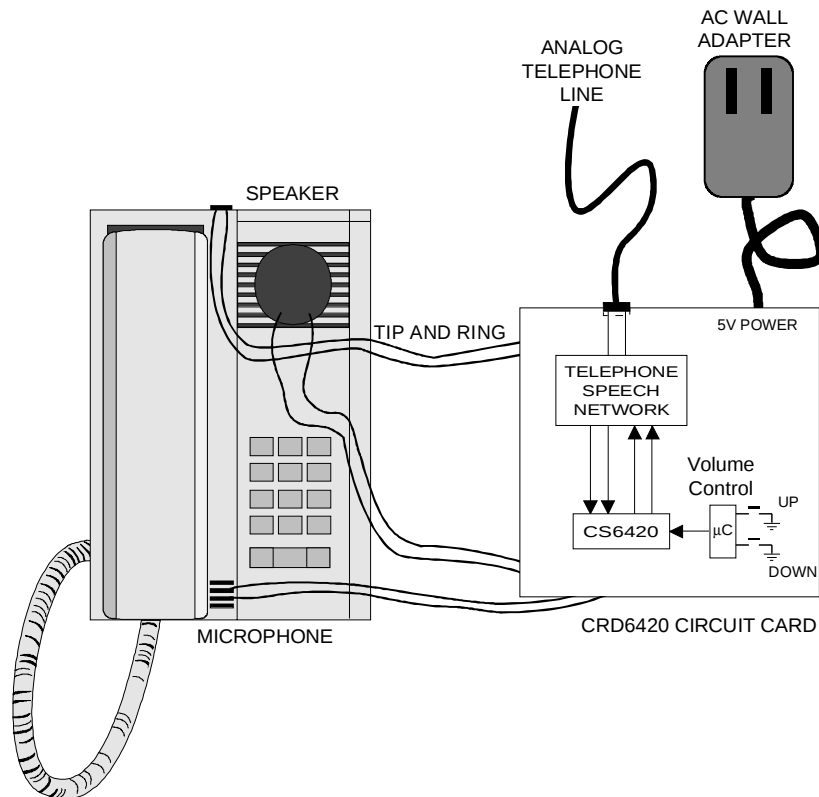


Figure 1. Speakerphone System Connections

This configuration permits easy evaluation of hands-free communication with a full duplex speakerphone. The speaker and microphone are housed in an actual chassis, providing a realistic environment. Various speakerphone designs can be quickly tested since standard analog signals are passed between the speakerphone and CS6420 circuit card. This method also uses the speakerphone's (tone) dialer.

Note that the performance of this system is dependent on the speakerphone used. In general the design criteria for full duplex speakerphones is much more stringent than for half duplex phones. In particular, distortion performance and acoustic coupling control are critical to good full duplex design. It is relatively easy to get the modified half-duplex speakerphone to work in full-duplex; however, its performance may suffer, particularly in near-end speaker volume and in idle channel noise in the transmit direction. Additionally, any distortion introduced by the speaker or the phone housing will result in uncanceled echo perceived at the far end.

The CS6420 circuit card interfaces to the telephone network via the Tip and Ring signals of the analog telephone line. The Tip and Ring signals are routed to a telephone speech network which converts the 2-wire analog telephone signal to 4-wire transmit and receive signals. The receive and transmit signals are interfaced to the CS6420's Network In (NI) and Network Out (NO) ports, respectively. The circuit for the POTS (Plain Old Telephone Service) analog telephone line interface is shown in Figure 2.

The 2-to-4 wire telephone hybrid is implemented with a Motorola MC34014. The MC34014 (U5) is a standard speech network meant for connection to the Tip and Ring lines through a polarity bridge (D6-D9). The discrete components surrounding U5 are as suggested by the MC34014 data sheet. The components which set the dc voltage characteristics have been omitted since this is established by

the speech network in the speakerphone. (Recall that the speakerphone's Tip and Ring signals remain connected to the POTS line in order to utilize the phone's keypad.) The omitted components include a resistor connected to LR (U5-13) and a capacitor connected to LC (U5-12).

Connection to the analog telephone line is made through J1, which is an RJ11 receptacle. This receptacle is a standard telephone connector that connects to the typical telephone cable provided with telephones. A DPDT switch (S1) is used to switch between ON HOOK and OFF HOOK.

The transmit and receive signals from the MC34014 are buffered and AC coupled using a Motorola op-amp MC33078 (U3). The gain of the op-amp circuit is set such that the NI amplitude is within the CS6420's limits of 1 V_{rms} and the CS6420's NO signal level, also 1 V_{rms}, adjusted to meet the MC34014's TX requirements. R9 and R10 create a voltage divider that establishes a voltage to bias the op-amp's inputs at 2.5 volts.

The schematic for the CS6420 (U1) circuitry is shown in Figure 3. The circuitry is identical to the typical connections described in the CS6420 data sheet and incorporated on the CDB6420 evaluation board.

Figures 4 and 5 are schematics for the speaker driver and microphone bias respectively. The Motorola MC34119 (U2) provides the capability to drive speaker loads down to 8 ohms. The microphone bias circuitry provides the necessary constant current source for an electret microphone. Both of these circuits are used on the CDB6420 evaluation board and are described in the CDB6420 data sheet.

A microcontroller is interfaced to the CS6420 to adjust the default settings and control speaker volume. Figure 6 contains a schematic using the Microchip PIC16C84 (U4) microcontroller. U4 writes control words to the CS6420's four control registers using a 3-wire interface ($\overline{\text{RDY}}$, STROBE,

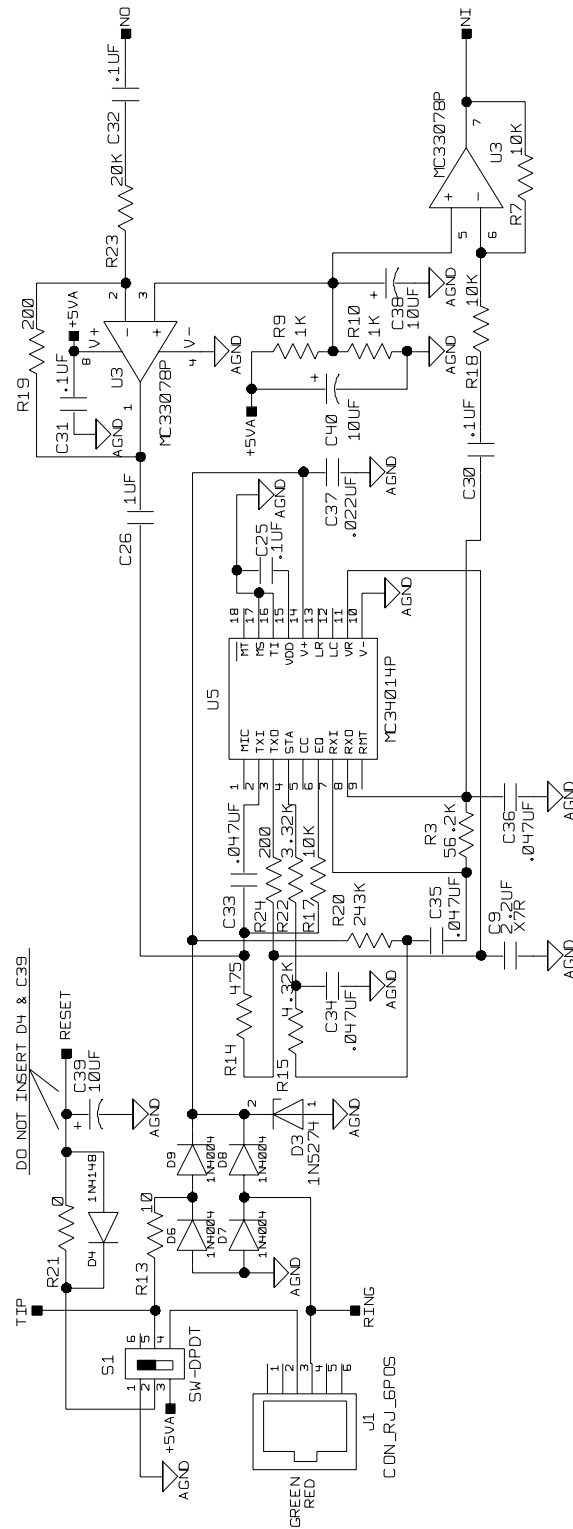


Figure 2. Interface to the Tip and Ring Signals of a POTS Line

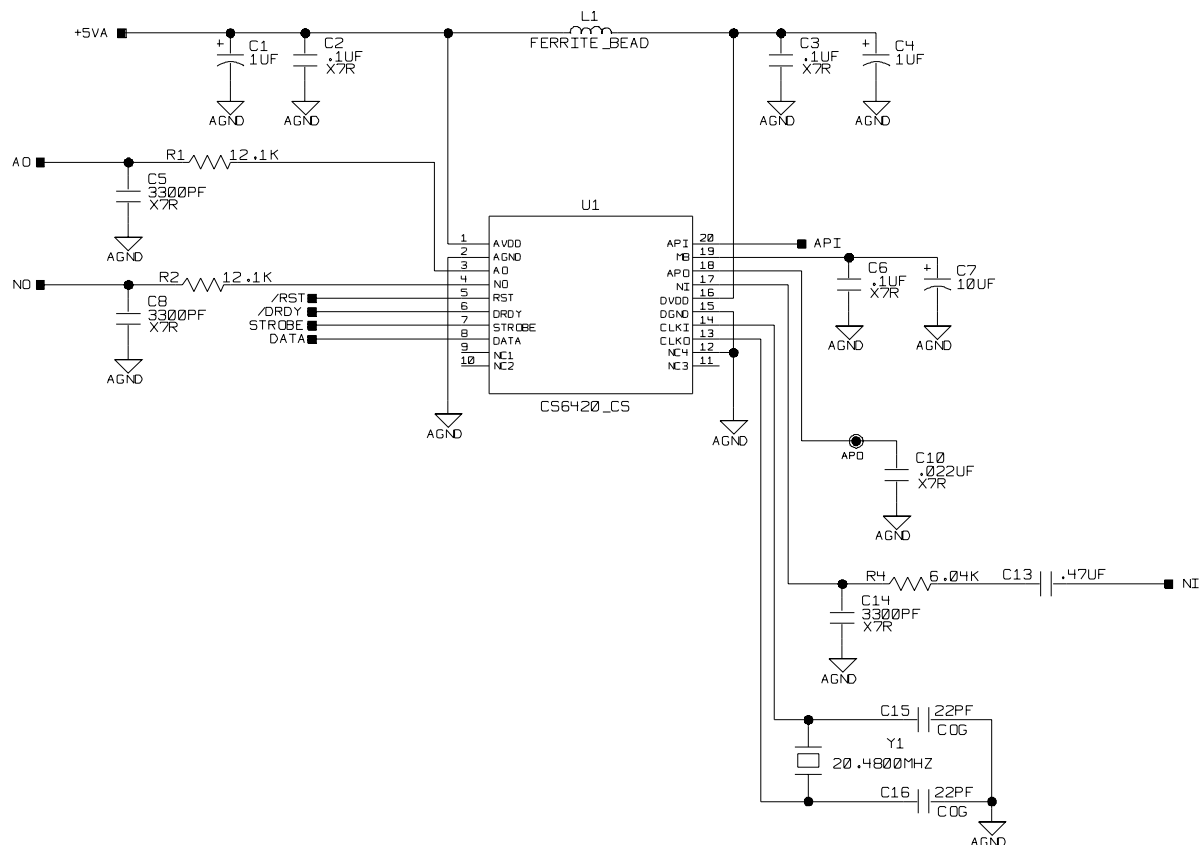


Figure 3. CS6420 Connection Diagram

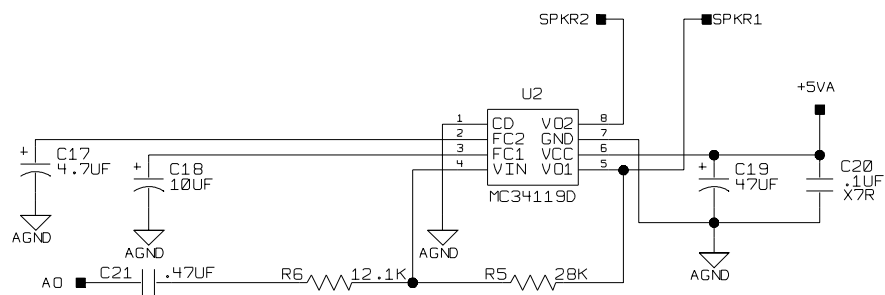
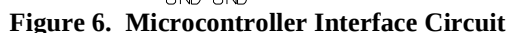


Figure 4. Speaker Driver Circuit

Figure 7 illustrates the power supply circuitry. The circuitry is designed to use a common +6 V DC power supply. The supply used should be able to source a minimum of 200mA. A diode (D5) provides a diode drop to reduce the voltage to around 5 volts. Z1 is a zener diode added to provide over-voltage and polarity inversion protection. For noise performance, the 5 volt digital voltage (+5 VD) is filtered using a ferrite bead.



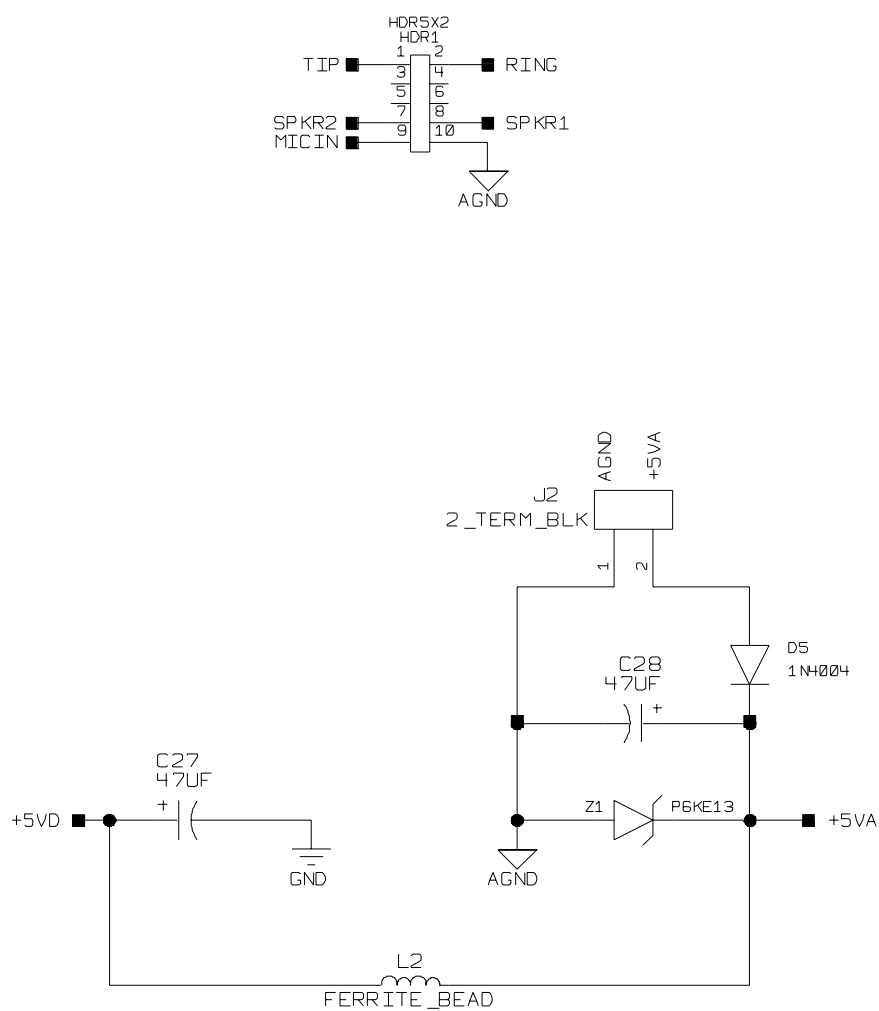


Figure 7. Power Supply Circuitry and Phone Connection Pinout

PIC16C84 ASSEMBLY CODE

```

;*****
;
;*
;* File:          crd6420.asm
;* Date:          May 12, 1997
;*
;*
;* PIC16C84
;*
;*
;* Program entry point is at routine "Main". The entry point
;* is address 0x05
;*****
;
;*
;* Program is designed to provide simple volume control for the CS6420
;* in a CDB6420-based demo. Two buttons, one to increase volume and
;* one to decrease volume are connected to PortB of the PIC16C84. Four
;* PortA pins are used to control the CS6420's Microcontroller Interface.
;*
;*****
;***** Memory Map Equates
INDF          equ      0x00          ; Indirect Address Register
STATUS       equ      0x03          ; STATUS register equate
RP0          equ      0x05          ; Register Bank Select Bit
ZERO         equ      0x02          ; Represents the Zero Bit in Status Reg
CARRY        equ      0x00          ; Represents the Carry Bit in Status Reg
FSR          equ      0x04          ; File Select Register
PORTA        equ      0x05          ; General Purpose I/O Port
PORTB        equ      0x06          ; General Purpose I/O Port
INTCON       equ      0x0B          ; INTCON register equate
GIE          equ      0x07          ; Represents the GIE bit in IntCon Reg
RBIE         equ      0x03          ; Represents the RBIE bit in IntCon Reg
RBIF         equ      0x00          ; Represents the RBIF bit in IntCon Reg
OPT_REG      equ      0x81          ; OPTION register equate
RBPU         equ      0x07          ; Represents the RBPU bit in Option Reg
TRISA        equ      0x85          ; Data Direction Control For Port A
TRISB        equ      0x86          ; Data Direction Control For Port B
DATAIN       equ      0x00          ; Port A bit 0
STROBE       equ      0x01          ; Port A bit 1
DRDY         equ      0x02          ; Port A bit 2
RST          equ      0x03          ; Port A bit 3
INC          equ      0x04          ; Port B bit 4
DEC          equ      0x05          ; Port B bit 5

;***** Ram Memory Equates
Reg0Hi       equ      0x0C          ; Upper 8 bits of Reg0
Reg1Hi       equ      0x0D          ; Upper 8 bits of Reg1
Reg2Hi       equ      0x0E          ; Upper 8 bits of Reg2
Reg3Hi       equ      0x0F          ; Upper 8 bits of Reg3
Reg0Lo       equ      0x10          ; Lower 8 bits of Reg0
Reg1Lo       equ      0x11          ; Lower 8 bits of Reg1
Reg2Lo       equ      0x12          ; Lower 8 bits of Reg2
Reg3Lo       equ      0x13          ; Lower 8 bits of Reg3
RegHi        equ      0x14          ; Upper 8 bits of Transmit Reg
RegLo        equ      0x15          ; Lower 8 bits of Transmit Reg
Temp         equ      0x16          ; Temporary byte
RVol         equ      0x17          ; Volume control variable
Count        equ      0x18          ; Loop counter
Port         equ      0x19          ; PortB temporary value
DelayTime    equ      0x1a          ; Delay counter
DelayMore    equ      0x1b          ; Delay counter

```

```

*****
;
;*   Program Code
;
*****
;
        processor    16C84                ; Set Processor Type
        org          0x00                ; Reset Vector
        goto         Main                ; Start at Main

        org          0x04                ; Interrupt Vector
        goto         ISR                 ; Service the Interrupt
*****
;
;*
;
;* Routine - Main
;* Input  - none
;* Output - none
;* This is the entry point to the program
*****
;
        org    0x05

Main                                ; Start from Reset Vector

***** Initialize System and Perform SELF OFFSET Calibration
;
        CALL    Initialization        ; Initialize the system
        CALL    PowerOn               ; Powerup the CS6420
Here      GOTO    Here                ; Infinite loop

```

```

;*****
;
;*
;* Routine - Initialization
;* Input - none
;* Output - none
;* Set up default control bytes, default volume setting, port pins
;*****
Initialization
; Set up default control bytes ;This configuration has the microphone tied directly to APO.
; Reg0Hi
        MOVLW    0x20
        MOVWF    Reg0Hi
; mic preamp off, RVol = +24 dB, TGain = +12 dB

; Reg0Lo
        MOVLW    0x58
        MOVWF    Reg0Lo

; Reg1Hi
        MOVLW    0x20
        MOVWF    Reg1Hi
; TVol = +21 dB, RGain = +12 dB

; Reg1Lo
        MOVLW    0x7a
        MOVWF    Reg1Lo

; Reg2Hi
        MOVLW    0x08
        MOVWF    Reg2Hi
; RHDet = 6 dB, RSThd = 6 dB, PCSen = low

; Reg2Lo
        MOVLW    0x2c
        MOVWF    Reg2Lo

; Reg3Hi
        MOVLW    0x0a
        MOVWF    Reg3Hi
; THDet = 6 dB, TSAAtt = 24 dB

; Reg3Lo
        MOVLW    0x0b
        MOVWF    Reg3Lo

; Set up default volume setting
; RVol = 0x02 (24 dB) (max of 0x1f)
        MOVLW    0x02
        MOVWF    RVol

; Set up port pins
; 2 inputs (increase volume, decrease volume)
; 4 outputs (/RST, /DRDY, STROBE, DATAIN)
        CLRF     PORTA
; Initialize PORTA by setting output
; data latches.
        CLRF     PORTB
; Initialize PORTB by setting output
; data latches.
        BSF      STATUS, RP0
; Select Bank 1
        MOVLW    0x00
; Value used to initialize direction
        MOVWF    TRISA
; Set PortA as outputs

        MOVLW    0x30
; Value used to initialize direction
        MOVWF    TRISB
; Set PortB RB5 and RB4 as inputs
        BCF      OPT_REG, RBPU
; Enable weak pullups on PortB
        BCF      STATUS, RP0
; Select Bank 0
        MOVLW    0xff
; bring all port pins high
        MOVWF    PORTA
        MOVLW    0x0f
; bring all port pins high
        MOVWF    PORTB
        BSF      INTCON, RBIE
; Enable interrupt on PortB change
        BSF      INTCON, GIE
; Enable all interrupts
        RETURN

```

```

;*****
;
;*
;* Routine - Power On
;* Input - none
;* Output - RegXHi,RegXLo
;* Reset CS6420 and initialize registers
;*****
PowerOn
    BCF      PORTA,RST      ; bring /RST lo
    BSF      PORTA,RST      ; bring /RST hi
                                ; Write Reg0 (RegHi=Reg0Hi, RegLo=Reg0Lo)
    CALL     Delay           ; hold time
    MOVF     Reg0Hi,0        ; move Reg0Hi to W
    MOVWF    RegHi           ; move W to RegHi
    MOVF     Reg0Lo,0        ; move Reg0Lo to W
    MOVWF    RegLo          ; move W to RegLo
    CALL     WriteReg        ; Write Reg1 (RegHi=Reg1Hi, RegLo=Reg1Lo)
                                ; move Reg1Hi to W
    MOVF     Reg1Hi,0        ; move W to RegHi
    MOVWF    RegHi           ; move W to RegHi
    MOVF     Reg1Lo,0        ; move Reg1Lo to W
    MOVWF    RegLo          ; move W to RegLo
    CALL     WriteReg        ; Write Reg2 (RegHi=Reg2Hi, RegLo=Reg2Lo)
                                ; move Reg2Hi to W
    MOVF     Reg2Hi,0        ; move W to RegHi
    MOVWF    RegHi           ; move W to RegHi
    MOVF     Reg2Lo,0        ; move Reg2Lo to W
    MOVWF    RegLo          ; move W to RegLo
    CALL     WriteReg        ; Write Reg3 (RegHi=Reg3Hi, RegLo=Reg3Lo)
                                ; move Reg3Hi to W
    MOVF     Reg3Hi,0        ; move W to RegHi
    MOVWF    RegHi           ; move W to RegHi
    MOVF     Reg3Lo,0        ; move Reg3Lo to W
    MOVWF    RegLo          ; move W to RegLo
    CALL     WriteReg
    CALL     UpdateRVol      ; Update RVol
    RETURN
;*****
;
;*
;* Routine - ISR
;* Input - none
;* Output - none
;* The ISR calls IncreaseVolume or DecreaseVolume as appropriate
;*****
ISR
    MOVF     PORTB,0         ; move contents of PortB to W
    MOVWF    Port            ; save contents to Temp for bit test
    BTFSS    Port,INC        ; if INC is high, is DEC set?
    CALL     IncreaseVolume   ; else IncreaseVolume
    BTFSS    Port,DEC        ; if DEC is high, fail
    CALL     DecreaseVolume   ; else DecreaseVolume
    BCF      INTCON, RBIF    ; clear port interrupt
    RETFIE                ; return from interrupt

```

```

;*****
;
;*
;* Routine - Increase Volume
;* Input - none
;* Output - RVol
;* If RVol>0x00 increment RVol and update
;*****
IncreaseVolume
    MOVF    RVol,0          ; move RVol to W
    SUBLW   0x00            ; W = 0x00 - RVol
    BTFSS   STATUS,ZERO     ; if zero bit is set, don't decrement
    DECF    RVol            ; else decrement
    CALL    UpdateRVol      ; Update RVol
    RETURN

;*****
;*
;* Routine - Decrease Volume
;* Input - none
;* Output - RVol
;* If RVol<0x1f increment RVol and update
;*****
DecreaseVolume
    MOVF    RVol,0          ; move RVol to W
    SUBLW   0x1f            ; W = 0x1f - RVol
    BTFSS   STATUS,ZERO     ; if zero bit is set, don't increment
    INCF    RVol            ; else increment
    CALL    UpdateRVol      ; Update RVol
    RETURN

;*****
;*
;* Routine - Update RVol
;* Input - Move RVol to Reg0Hi:Reg0Lo
;* Output - Reg0Hi,Reg0Lo
;* This routine takes RVol and updates Reg0 with the values
;*****
UpdateRVol
    BCF     Reg0Hi,1
    BCF     Reg0Hi,0
    BCF     Reg0Lo,7
    BCF     Reg0Lo,6
    BCF     Reg0Lo,5          ; set RVol area in Reg0 to 0s
    BTFSC   RVol,4
    BSF     Reg0Hi,1          ; set appropriate bit to 1 as needed
    BTFSC   RVol,3
    BSF     Reg0Hi,0
    BTFSC   RVol,2
    BSF     Reg0Lo,7
    BTFSC   RVol,1
    BSF     Reg0Lo,6
    BTFSC   RVol,0
    BSF     Reg0Lo,5
    MOVF    Reg0Hi,0          ; move Reg0Hi to W
    MOVWF   RegHi             ; move W to RegHi
    MOVF    Reg0Lo,0          ; move Reg0Lo to W
    MOVWF   RegLo             ; move W to RegLo
    CALL    WriteReg
    RETURN

```

```

;*****
;
;*
;*
;* Routine - WriteReg (keeps /RST hi)
;* Input - Word to be sent is in RegHi:RegLo
;* Output - none
;* This routine calls SendData twice with the real data and adds the dummy writes
;*****
WriteReg
    BCF      PORTA,STROBE    ; set STROBE lo
    BCF      PORTA,DRDY     ; set DRDY lo
    MOVF     RegHi,0        ; move RegHi to W
    MOVWF    Temp          ; move W to Temp
    CALL     SendData       ; bit-bang 8 bits (most-significant)
    MOVF     RegLo,0        ; move RegLo to W
    MOVWF    Temp          ; move W to Temp
    CALL     SendData       ; bit-bang 8 bits (least-significant)
    BSF      PORTA,DRDY     ; signal end of transaction
    MOVLW    #0x04         ; NOW send four dummy clocks to latch data
    MOVWF    Count         ; initialize Count to 4
LOOP4   BCF      PORTA,DATAIN ; set DATAIN lo
        BSF      PORTA,STROBE ; toggle STROBE (4 extra STROBE pulses)
        BCF      PORTA,STROBE
        DECFSZ   Count       ; decrement counter
        GOTO     LOOP4      ; loop 4 times
        RETURN            ; return from subroutine
;*****
;
;*
;* Routine - SendData (keeps /DRDY lo and /RST hi)
;* Input - Byte to be transmitted is passed in Temp
;* Output - none
;* This routine sends 1 byte to the CS6420 MCR
;*****
SendData
    MOVLW    #0x08
    MOVWF    Count         ; initialize Count to 8
LOOP8   RLF     Temp,1      ; rotate byte left one
        BTFSC    STATUS,CARRY ; if carry bit = 1,
        BSF      PORTA,DATAIN ; set DATAIN hi
        BTFSS    STATUS,CARRY ; if carry bit = 0,
        BCF      PORTA,DATAIN ; set DATAIN lo
        BSF      PORTA,STROBE ; clock data in
        BCF      PORTA,STROBE
        DECFSZ   Count,1    ; decrement counter
        GOTO     LOOP8      ; loop 8 times
        BCF      PORTA,DATAIN ; set DATAIN high
        RETURN
;*****
;
;*
;* Routine - Delay
;* Input - none
;* Output - none
;* This routine is a software delay
;*****
Delay
    CLRf     DelayTime     ; implement when needed
DelayLoop MOVlw    0x34      ; Count 100 or 0x34
    MOVWF    DelayMore
Delay2   DECFSZ   DelayMore,1 ; loop 100 times
        GOTO     Delay2
        DECFSZ   DelayTime,1 ; loop 255 times
        GOTO     DelayLoop
        RETURN
end

```

• Notes •

SMART
Analog™