WIRELESS REMOTE VOLUME CONTROL

This project is a wireless remote volume control that's connected between a computer or MP3 player and an amplified speaker system. The volume can be controlled from up to 50 feet away and through walls. This RF remote volume control would be useful even if your sound system already has an IR remote control, since it doesn't require range-limiting “line-of-sight” contact with the receiver. This project also features a mute function and a default, or preset volume setting. The default volume setting is used as a reference for adjusting other volume controls in the sound system. It's also great for quickly returning to normal volume when the volume is turned up loud to be heard in another room. The default volume level is 12db below the maximum volume level.

The project uses two Dallas Semiconductor DS1809 digital pots. The remote control that was hacked up to make this project is a Woods model 32555, available at Amazon.com. This particular control is perfectly suited to this hack because it’s cheap and its RF circuit is a separate module that can be removed and used in the project. The rest of the receiver can be scrapped or used to build an automatic power switch for the speaker system. The project uses two bottom-end PIC microcontrollers. One PIC decodes the remote control signals and the other PIC controls the digital pots. A discarded 5 volt cellphone charger was used to power the author’s system, but any 5V, 50mA (or larger) regulated power supply will do.

The ASM code for the pot controller PIC has the following functionality:
At power-up, the pots are clocked down to zero as if they were at the maximum (63) count. The pots are then clocked up with 1.5mS pulses (1.5mS apart) to a default count of 16. The VOLUME register is also set at 16. The volume register is clocked up and down with the pots to keep track of their position. The volume register is necessary to return to the previous volume when the device is unmuted. The mute function is activated by pressing the UP button and then pressing the DN button within 1 second. Once muted, pressing either button will unmute it. The volume can be restored to the default value by pressing the DN button and then pressing the UP button within 1 second. For optimum response from the control, the other volume controls in the sound system should be set for a normal listening volume with the remote control at the default volume setting.

A simulated logarithmic response is created in software to produce a noticeable change in volume with each single-step of the volume control, like using an audio-taper pot. The control goes from zero to maximum volume in 18 steps. Holding the UP or DN button down for 0.5 sec will cause the pots to be pulsed up or down at a rate of 10 times per second. There is no logarithmic function in this “scan” mode. The volume can be single-stepped up or down as rapidly as desired, but pressing the opposite button within 1 second (if not in scan mode) will either activate the mute function or return the device to the default volume setting.

This circuit also has an audio output for an optional audio-operated power switch that would turn the amplified speaker system on or off with the presence or absence of an input signal. This is great when using an MP3 player or an iPod as the signal source, but not so great when a computer is the signal source. A master/slave or USB operated switch would be better to use when a computer is the signal source since the speakers would otherwise switch on and off every time Windows made a brief sound. The author will make available at the reader’s request a schematic of a working master/slave and/or audio-operated switch. The author will also make available, at reasonable cost, pre-programmed PICs to those who want to build the project but do not have the means to program a PIC.

The PICs were programmed with a PICSTART PLUS, using MPLAB IDE Version 8.70. If you are new to programming the PIC10F200 or the PIC10F202, be aware that pin 1 of the PIC10F20x is plugged into pin 9 of the PICSTART PLUS ZIF socket.

If you receive a Woods model 32555 remote control with the antenna wire neatly curled into a tight coil, straighten it out. The antenna wire should be as straight as possible (no sharp turns) and looped one time around the interior of the enclosure and the excess wire cut off. This will significantly improve the operating range of the remote.

The author would like to hear from readers regarding their experiences with this project and any improvements they may have made.
Figure 1. Schematic of the main circuit board of the remote volume control.

Publishing the project made it necessary to find a suitable remote control that is currently available. Two separate circuit boards were used because the remote originally used in the project did not have a removable RF receiver module. Also, the decoder chip in the original remote had both the outputs needed for this project. The decoder chip in the new remote did not have the momentary or “strobe” output. This required the programming of a second PIC to emulate the decoder chip and add the strobe output. One circuit board can be used if desired. There are no critical layout issues. This project was hand-wired on Protoboard, so no PCB layout is available.
Figure 2. Receiver circuit board containing the RF module removed from the Woods model 32555 remote control.

NOTE:
C7 is only needed if U6 is on a separate circuit board from U1.

Figure 3. Interconnection of the circuit boards.
J4 is actually a barrel type of power connector. The schematic program did not have a suitable library component or the tools to easily make one. The connector specified in the BOM is the mating connector for the specified AC adapter. Substitute whatever connector mates with your AC adapter or use no connector at all. The author cut the cellphone connector off the adapter he used and soldered the leads directly to the circuit board. The (+) lead from the 5 volt power supply is connected to the (+2.5V) power input and the (-) lead from the power supply goes to the (-2.5V) power input. U4, Q1, Q2 and the surrounding components provide the ground. When selecting an AC adapter, bear in mind that the maximum allowable power supply voltage for the PICs is 5.5V. In the unaltered remote receiver, the RF module is powered with 5.6 volts, but it works just fine on the 5.1V from the lightly loaded cellphone charger.
Figure 4. Layout of the hand-wired main circuit board.

Note that the PIC (U1) is placed close to the top edge of the circuit board to facilitate removal and reinstallation for programming. “AG” on this sketch means “analog ground”. It is called “GND” in figure 3. The grounds of the input and output jacks are connected together and tied to the circuit board at one point for convenience. Due to the high signal levels and the low system gain, this does not cause any ground issues. On the receiver connections, “S” means “STRB” and “D” means “DATA”. “V” is the same as “+2.5V” and “G” is the same as “-2.5V” in figure 3. On the output side, “S” means “switch”, the same as “SOUT” in figure 3. “R” means “right” and “L” means “left” for both the input and output. Some of the reference designators were intentionally omitted to avoid obstructing the view of the interconnecting wires. C5 and C6 were soldered onto the bottom of the circuit board when oscillation was discovered.
Remote volume control

Written by:
Andrew R. Morris
morr3763@bellsouth.net

This code was written for a PIC10F200 or PIC10F202
This code is for a wireless remote volume control using two
Dallas Semiconductor DS1809 digital pots. The remote control
device to be hacked up and used here must have two outputs.
One output is an on/off or “latched” output that is herein called
DATA and the other is a momentary output that is herein called STRB
(for “strobe”). STRB must go high when either button is pressed.
STRB is necessary in order to know when a button has been pressed
and DATA to know which button it was.

LIST  P=10F202
#include P10F202.INC

errorlevel -302 ; suppress message 302 from list file

__CONFIG  _CP_OFF & _MCLRE_OFF & _WDT_OFF & _IntRC_OSC

;Register equates
LCVR1  EQU  10    ;Loop count variable register 1
LCVR2  EQU  11    ;Loop count variable register 2
LCVR3  EQU  12    ;Loop count variable register 3
LCVR4  EQU  13    ;Loop count variable register 4
STAT   EQU  14    ;Status flag register
VOLUME EQU  15    ;Current volume setting (0-63)
PM_VOL EQU  16    ;Pre-mute volume

;Initialization equates
INIT_GP  EQU  B'00000110' ;Initialize GPIO
CFG_GP  EQU  B'00001001' ;Configure GPIO
OPTINI  EQU  B'10000000' ;Initialize option register

;STAT register equates
MUTED  EQU  0     ;This status bit is high when the device is muted.

;Parameter equates
INIT_VOL EQU  D'16' ;Default volume level

;I/O pin equates
;GPIO
STRB_IN EQU 0 ; Strobe input from remote receiver
DATA_IN EQU 3 ; Data (on/off) input from remote receiver
UP_OUT EQU 1 ; UP output to pots
DN_OUT EQU 2 ; Down output to pots

ORG 0 ; Reset vector rolls over to here
GOTO Boot ; Go to beginning of program

; Subroutines

VOL_INI

MOVLW D'68' ; Ensures pots are at zero by
MOVVWF LCVR4 ; pulsing pots downward 68 counts.

VRS

BCF GPIO,DN_OUT ; 63 counts plus a few extra to ensure
CALL DLY1 ; synchronization of pots to each other
BSF GPIO,DN_OUT ; and the VOLUME register.
CALL DLY1
DECFSZ LCVR4,F
GOTO VRS

MOVLW INIT_VOL ; Load default value into VOLUME register
MOVVWF VOLUME ; and loop counter.
MOVVWF LCVR4

VIN

BCF GPIO,UP_OUT ; Clock pots up to default volume level
CALL DLY1
BSF GPIO,UP_OUT
CALL DLY1
DECFSZ LCVR4,F
GOTO VIN
RETLW 0

DLY1

MOVLW D'149' ; Sets width of UP_OUT and DN_OUT pulses
MOVVWF LCVR1 ; and spacing between them in MUTE and

LP1

MOVLW D'2' ; UNMUTE. 1.5mS (1mS min per pot datasheet)
MOVVWF LCVR2

LP2

DECFSZ LCVR2,F
GOTO LP2
DECFSZ LCVR1,F
GOTO LP1
RETLW 0

DLY2

MOVLW D'150' ; Sets time between UP_OUT or DN_OUT pulses
MOVVWF LCVR1 ; during scan. 100mS (6.5sec full range)

LP3

MOVLW D'220' ;
MOVVWF LCVR2

LP4

DECFSZ LCVR2,F
GOTO LP4
DECFSZ LCVR1,F
GOTO LP3
RETLW 0

DLY3
MOVLW D'190' ;15mS delay to ensure pots are ready to
MOVVWF LCVR1 ;accept data.(10mS min per datasheet)
LP5
MOVLW D'25'
MOVVWF LCVR2
LP6
DECFSZ LCVR2,F
GOTO LP6
DECFSZ LCVR1,F
GOTO LP5
RETLW 0

VOL_UP ;Adjusts volume upward. (Pseudo-log response)

MOVF VOLUME,W
MOVVWF PM_VOL ;Save previous volume level for unmute.
BTFSC VOLUME,5 ;Volume rate of change doubles as each upper
GOTO UP8 ;bit goes high (excluding bits 0, 1 and 3).
BTFSC VOLUME,4
GOTO UP4
BTFSC VOLUME,2
GOTO UP2

UP1
MOVLW D'1'
GOTO $+6
UP2
MOVLW D'2'
GOTO $+4
UP4
MOVLW D'4'
GOTO $+2
UP8
MOVLW D'8'
MOVVWF LCVR3

UPLP
CALL DLY1 ;Max count of VOLUME register held at 63.
BCF GPIO,UP_OUT ;Further pulsing of pots OK for synchronization.
CALL DLY1
BSF GPIO,UP_OUT ;Pots will ignore further pulsing at max or min.
INCF VOLUME,F
BTFSC VOLUME,6 ;Test if count reached 64
DECF VOLUME,F ;Decrement back to 63
DECFSZ LCVR3,F
GOTO UPLP
RETLW 0

VOL_DN ;Adjusts volume downward. (Pseudo-log response)

BTFSC VOLUME,5 ;Same process as VOL_UP, except no volume save.
GOTO DN8
BTFSC VOLUME,4
GOTO DN4
BTFSC VOLUME,2
GOTO DN2
DN1
MOVLW D'1'
GOTO $+6
DN2 MOVLW D'2'
GOTO $+4
DN4 MOVLW D'4'
GOTO $+2
DN8 MOVLW D'8'
MOVWF LCVR3
DNLP CALL DLY1
BCF GPIO,DN_OUT ;Min count of VOLUME register held at 0.
CALL DLY1 ;Further pulsing of pots OK for synchronization.
BSF GPIO,DN_OUT
DECF VOLUME,F
BTFS C VOLUME,7 ;Test if count reached -1 (all 1's)
INCF VOLUME,F ;Increment back to zero
DECFSZ LCVR3,F
GOTO DNLP
RETLW 0
SCN_UP ;Scans (moves rapidly) volume upward
BTFSS GPIO,STRB_IN ;Stop scanning if button is released
GOTO START
CALL UP1 ;Bump volume up one count
CALL DLY2 ;Wait 100mS
GOTO SCN_UP ;Repeat
SCN_DN ;Scans (moves rapidly) volume downward
BTFSS GPIO,STRB_IN ;Stop scanning if button is released
GOTO START
CALL DN1 ;Bump volume down one count
CALL DLY2 ;Wait 100mS
GOTO SCN_DN ;Repeat
MUTE ;Adjusts volume down past zero
 ;to ensure both pots are in sync.
MOVLW D'68'
MOVWF LCVR4
MLP BCF GPIO,DN_OUT
CALL DLY1
BSF GPIO,DN_OUT
CALL DLY1
DECFSZ LCVR4,F
GOTO MLP
BSF STAT,MUTED ;Set MUTED flag
GOTO START
UNMUTE ;Adjusts volume back to previous level
MOVF PM_VOL,W ;Put previous volume level into VOLUME register
MOVWF VOLUME ;and loop counter because volume was changed
MOVWF LCVR4 ;when the UP button was pressed to mute the device.
UMLP BCF GPIO,UP_OUT
CALL DLY1
BSF GPIO,UP_OUT
CALL DLY1
DECFSZ LCVR4,F
GOTO UMLP
BCF STAT,MUTED ;Clear MUTED flag
GOTO START

VOL_TST ;Tests to see if volume is to go up or down
BTFSC GPIO,DATA_IN
GOTO VUP

CALL VOL_DN
MOVLW D'10' ;Go to default volume level if UP button is
MOVWF LCVR1 ;pressed within 1 sec. of pressing DN.
LP7 MOVLW D'91'
MOVWF LCVR2
LP8 MOVLW D'220'
MOVWF LCVR3
LP9 BTFSC GPIO,STRB_IN ;Leave 1 sec timer if a button has been pushed.
GOTO DNBR ;Go to branch routine to determine which
DECFSZ LCVR3,F ;button was pushed.
GOTO LP9
DECFSZ LCVR2,F
GOTO LP8
DECFSZ LCVR1,F
GOTO LP7
GOTO START

DNBR BTFSC GPIO,DATA_IN ;Reset volume to default value.
GOTO RST
CALL VOL_DN ;Lower volume another step.
GOTO START

VUP CALL VOL_UP
MOVLW D'10' ;Go to MUTE if DN button is pressed
MOVWF LCVR1 ;within 1 sec. of pressing UP.
LP10 MOVLW D'91'
MOVWF LCVR2
LP11 MOVLW D'220'
MOVWF LCVR3
LP12 BTFSC GPIO,STRB_IN ;Leave 1 sec timer if a button has been pushed.
GOTO UPBR ;Go to branch routine to determine which
DECFSZ LCVR3,F ;button was pushed.
GOTO LP12
DECFSZ LCVR2,F
GOTO LP11
DECFSZ LCVR1,F
GOTO LP10
GOTO START

UPBR BTFSS GPIO,DATA_IN ;Mute volume.
GOTO MUTE
CALL VOL_UP ;Raise volume another step.
GOTO START

SCN_TST ;Tests to see if volume is to be scanned up or down
BTFSC GPIO,DATA_IN
GOTO SCN_UP
GOTO SCN_DN

Boot
MOVWF OSCCAL ;Load the factory Internal oscillator calibration value

; Initialize I/O ports
MOVLW CFG_GP ;Select direction of GPIO bits
TRIS GPIO ;Write selection to GPIO data
 ;direction register
MOVLW INIT_GP ;Initialize GPIO
MOVWF GPIO
CLRF STATUS ;Clear status flag register.
CLRF STAT
CLRF TMR0 ;Reset TMR0 and prescaler
CLRWDT ;Reset watchdog timer
MOVLW OPTINI ;Initialize option register
OPTION CALL DLY3 ;Wait 15mS for pots to be able to accept data.
RST CALL VOL_INI ;Set initial volume setting

START ;The program actually starts here.
BTFSC GPIO,STRB_IN ;Make sure no button is still pushed.
GOTO §-1
BTFSS GPIO,STRB_IN ;Wait until a button is pushed.
GOTO §-1
BTFSC STAT,MUTED ;Are we in MUTED mode?
GOTO UNMUTE

MOVLW D'5' ;Go to scan routine if the UP or DN button
MOVWF LCVR1 ;is pressed longer than 0.5 sec. Otherwise
LP13 MOVLW D'91' ;adjust volume.
MOVWF LCVR2
LP14 MOVLW D'220'
MOVWF LCVR3
LP15 BTFSS GPIO,STRB_IN
GOTO VOL_TST
DECFSZ LCVR3,F
GOTO LP15
DECFSZ LCVR2,F
The purpose of this program is to emulate the decoder chip in the Woods model 32555 wireless remote control, adding a strobe output which the original circuit lacks. The strobe output goes high when either button on the remote is pressed. The strobe output is necessary when using the remote for anything but ON/OFF applications. The remote and the receiver both have 4 PCB bridges which are severed to select a digital "channel". In the receiver, they are labeled "CH A, B, C, and D". Severing the bridge causes the corresponding bit to go high. Because the bottom-end PIC used here doesn't have enough I/O pins, channel selection must be done during programming by changing the CHNL parameter.

This code was written for a PIC10F200 or PIC10F202

LIST  P=10F202
#include P10F202.INC

errorlevel -302 ; suppress message 302 from list file

__CONFIG  _CP_OFF & _MCLRE_OFF & _WDT_OFF & _IntRC_OSC

;Register equates

LCVR1    EQU  10     ;Loop count variable register 1
LCVR2    EQU  11     ;Loop count variable register 2
STAT     EQU  12     ;STAT register
CHANNEL  EQU  13     ;CHANNEL register

;Initialization equates

INIT_GP   EQU  B'00000000' ;Initialize GPIO
CFG_GP    EQU  B'00001000' ;Configure GPIO
OPTINI    EQU  B'10000000' ;Initialize option register

;STAT register equates

ON       EQU  0       ;1 if "ON" button was pushed
BIT       EQU  1       ;1 if incoming bit is high
;CHANNEL register equates
CHA  EQU  3
CHB  EQU  2
CHC  EQU  1
CHD  EQU  0

;Parameter equates
CHNL EQU B'00001000' ;Digital channel "A"
;Channel selection is B'xxxxABCD'
;Channel A is B'xxxx1000'
;Channel G is B'xxxx0011'

;/O pin equates

;GPIO
DATA_IN EQU 3 ;Data input from RF module
STRB_OUT EQU 1 ;Strobe output to digital pot controller
DATA_OUT EQU 2 ;Data output to digital pot controller

;-----------------------------------------------
ORG 0 ;Reset vector rolls over to here
GOTO Boot ;Go to beginning of program

;Subroutines

BITST ;Tests incoming bits for 1 or 0
BTFSS GPIO,DATA_IN ;Wait for data input to go high
GOTO $-1
BCF STAT,BIT
MOVWF LCVR1
LP1
MOVFW D'2'
MOVWF LCVR2
LP2
DECFSZ LCVR2,F
GOTO LP2
DECFSZ LCVR1,F
GOTO LP1
BTFSC GPIO,DATA_IN ;Test data bit
BSF STAT,BIT
BTFSC GPIO,DATA_IN ;Wait for data input to go low
GOTO $-1
RETLW 0

Boot
MOVWF OSCCAL ;Load the factory Internal oscillator calibration value

; Initialize I/O ports
MOVFW CFG_GP ;Select direction of GPIO bits
TRIS GPIO ;Write selection to GPIO data 
   ;direction register

MOVLW INIT_GP ;Initialize GPIO
MOVWF GPIO

CLRF STATUS

CLRF TMRO ;Reset TMRO and prescaler

CLRWDT ;Reset watchdog timer

MOVLW OPTINI ;Initialize option register

OPTION

START ;The actual program starts here.

BCF GPIO,STRB_OUT ;Reset strobe output

SYNC MOVLW D'62' ;Wait for data from RF module to stay low for 8mS.
MOVWF LCVR1 ;This is the point where the decoder is

LP3 MOVLW D'25' ;synchronized to the transmitter.
MOVWF LCVR2

LP4 BTFSC GPIO,DATA_IN
GOTO START ;Start over if high detected.

DECFSZ LCVR2,F
GOTO LP4

DECFSZ LCVR1,F
GOTO LP3

CALL BITST ;Start testing incoming bits.

BTFSC STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "1"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "1"
GOTO START

CALL BITST

BTFSC STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "1"
GOTO START

CALL BITST

BTFSC STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSC STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSS STAT,BIT ;Bit must be "0"
GOTO START

CALL BITST

BTFSC STAT,BIT ;Bit must be "0"
GOTO START
CALL BITST
BTFSC STAT,BIT ;Bit must be "0"
GOTO START
CALL BITST
BTFSC STAT,BIT ;Bit must be "0"
GOTO START
CALL BITST
BTFSC STAT,BIT ;Set ON bit if high
BSF STAT,ON
CALL BITST
BTFSC STAT,BIT ;Clear ON bit if high
BCF STAT,ON
CLRF CHANNEL
CALL BIT
BTFSC STAT,BIT ;Set channel A bit
BSF CHANNEL,CHA ;if bit is high
CALL BITST
BTFSC STAT,BIT ;Set channel B bit
BSF CHANNEL,CHB ;if bit is high
CALL BITST
BTFSC STAT,BIT ;Set channel C bit
BSF CHANNEL,CHC ;if bit is high
CALL BITST
BTFSC STAT,BIT ;Set channel D bit
BSF CHANNEL,CHD ;if bit is high
MOVLW CHNL
ANDLW B'00001111'
SUBWF CHANNEL,W ;Test if "channel" received is
BTFSS STATUS,Z ;the one desired.
GOTO START ;Start over if not.
BTFSC STAT,ON
GOTO $+3
BCF GPIO,DATA_OUT ;Set data output high or low
GOTO $+2 ;according to button pressed.
BSF GPIO,DATA_OUT
BSF GPIO,STRB_OUT ;Set strobe output.
GOTO SYNC ;Repeat without resetting strobe output.

END