Mix Masters

Remote Controlled KitchenAid Mixer for the Clients at ADEC

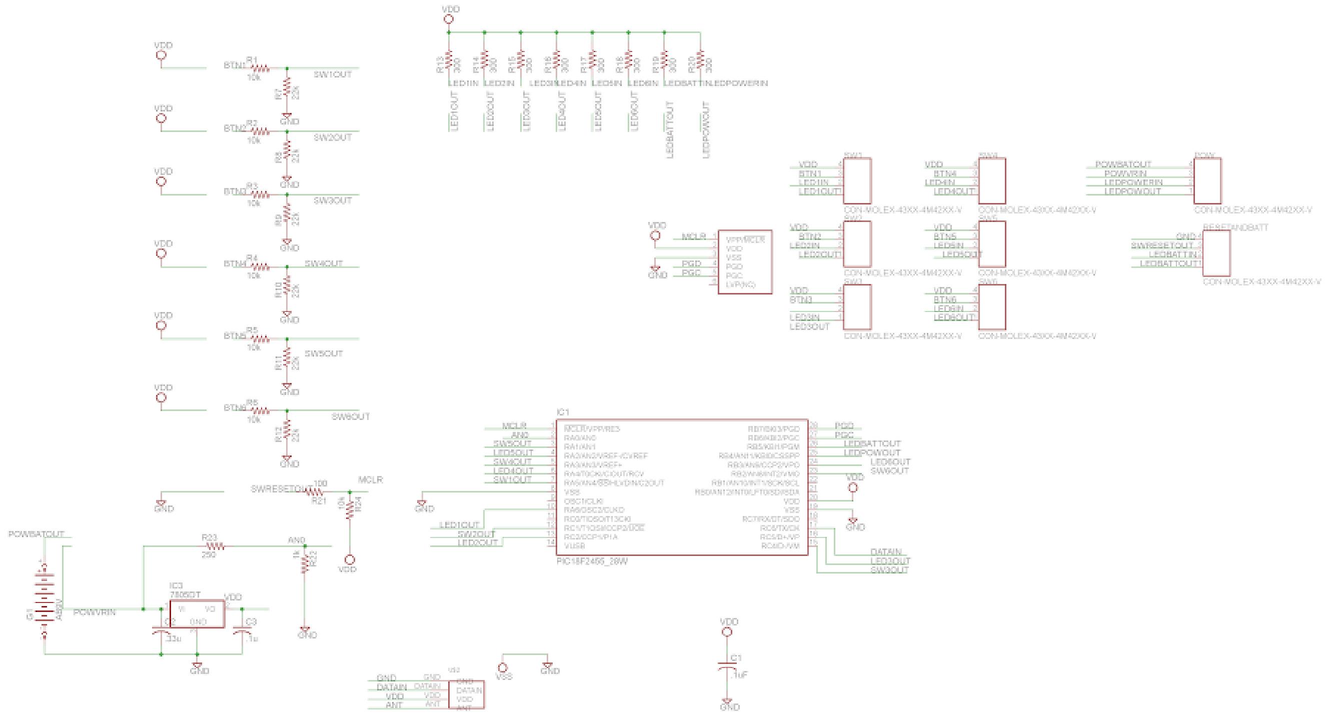
Appendices

Arnaud Bacye
Karina Dubé
Justin Erman
Matthew Martin

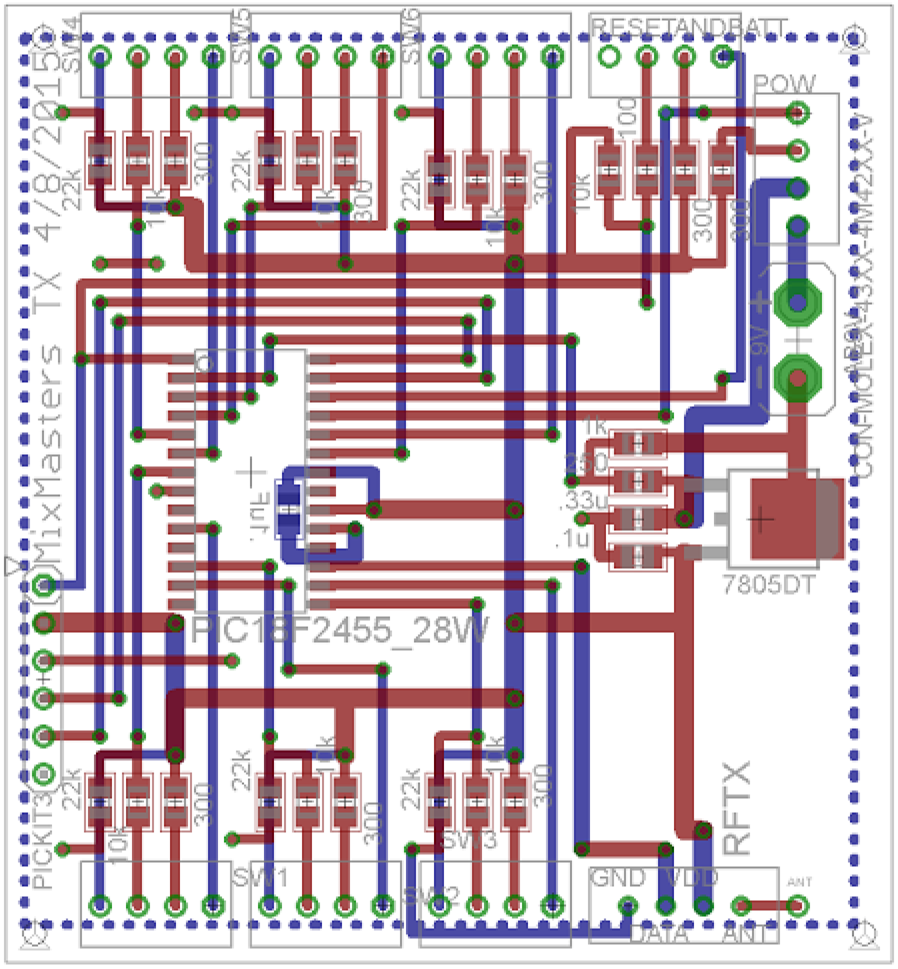
Table of Contents

Appendix A: Hardware Schematics and Boards

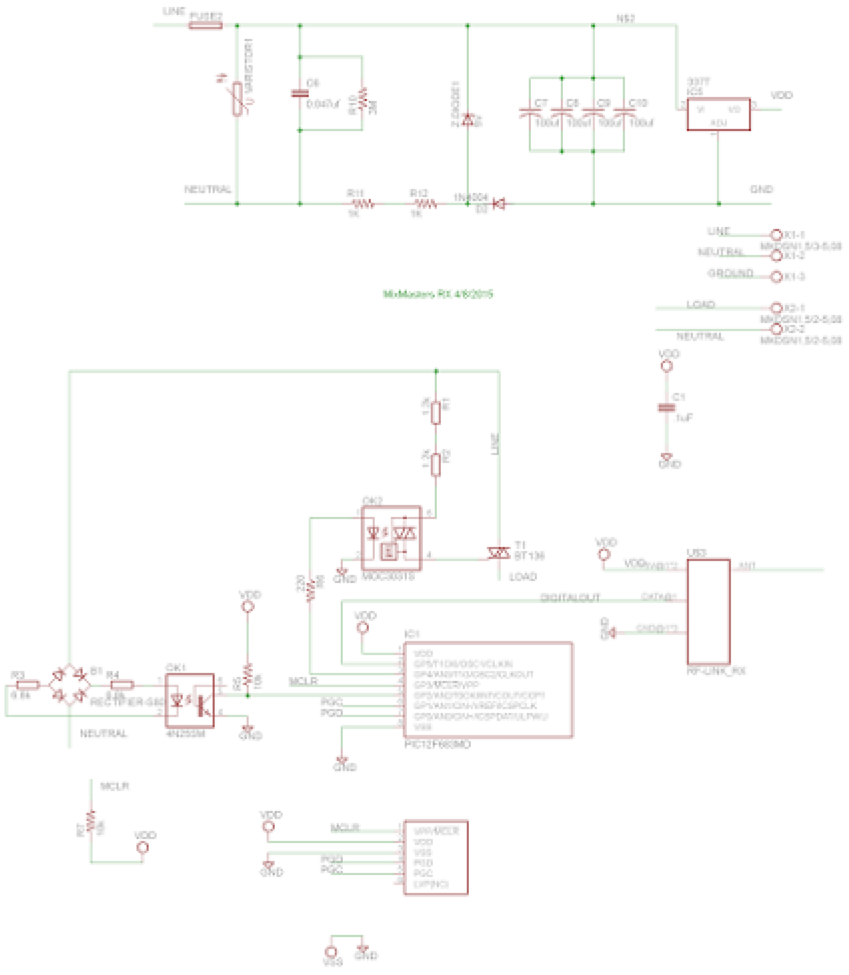
A.1 Original Schematics and Boards



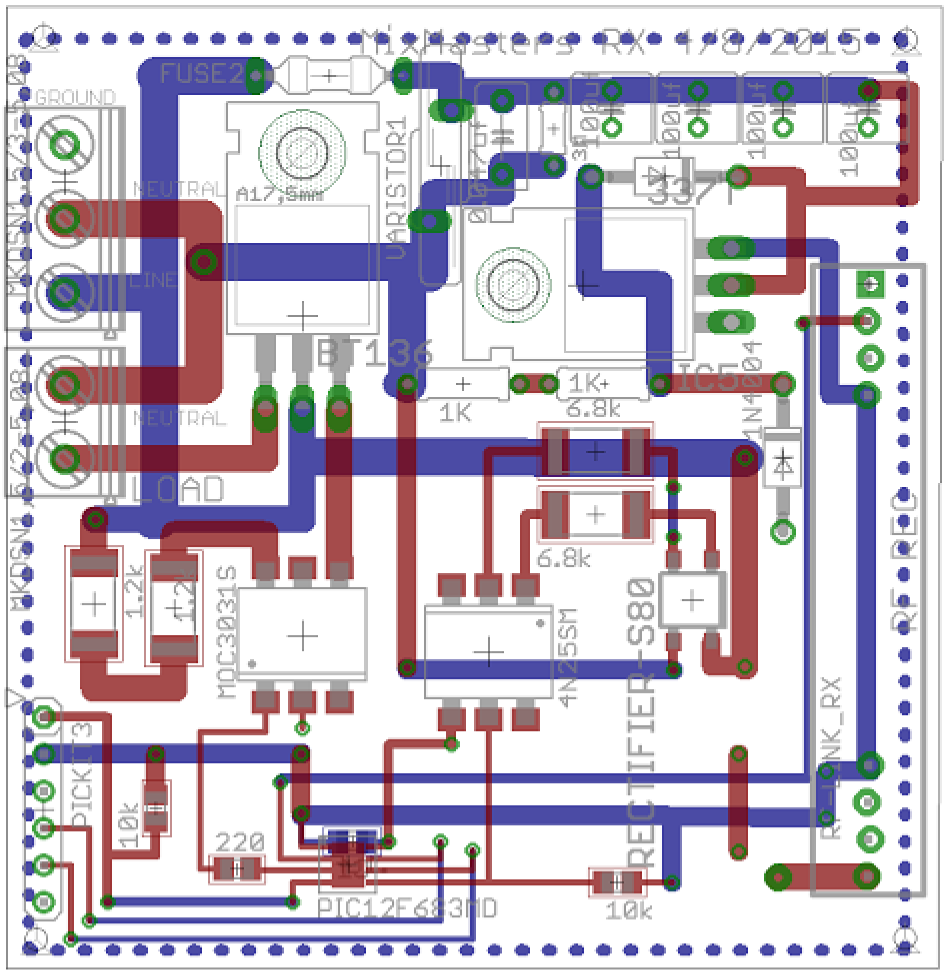
A.1.1: Remote Side Schematic



A.1.2 : Remote Side Board

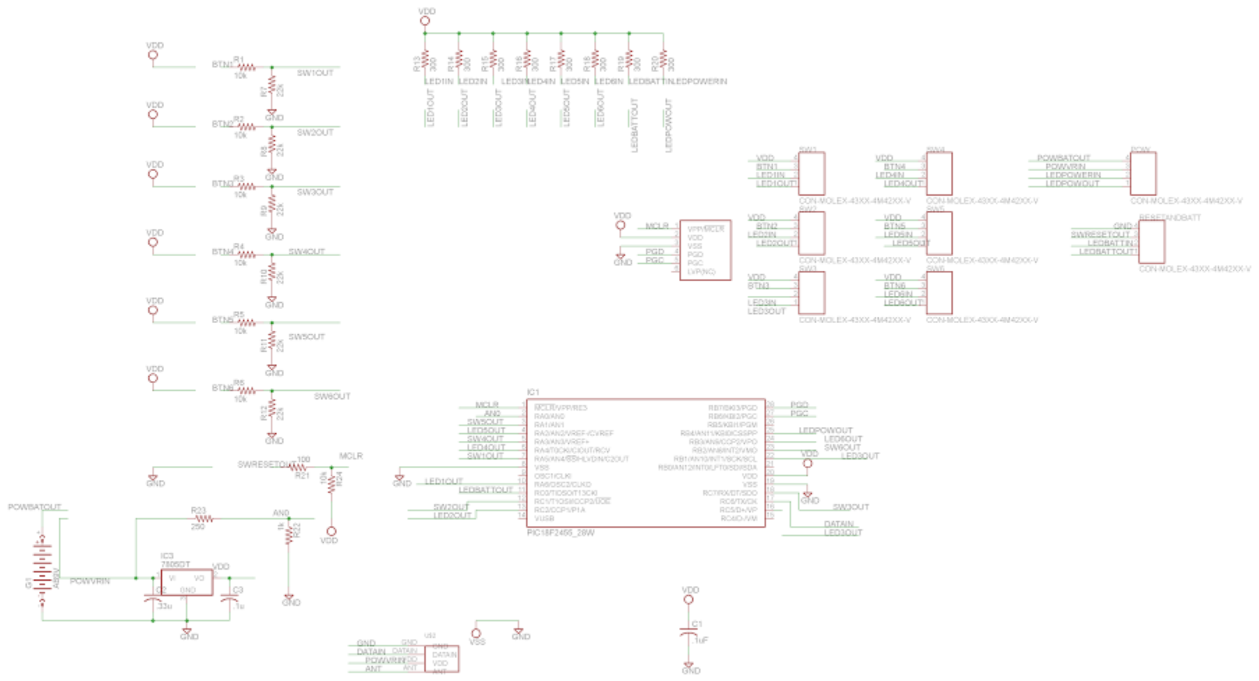


A.1.3 : Mixer Side Schematics

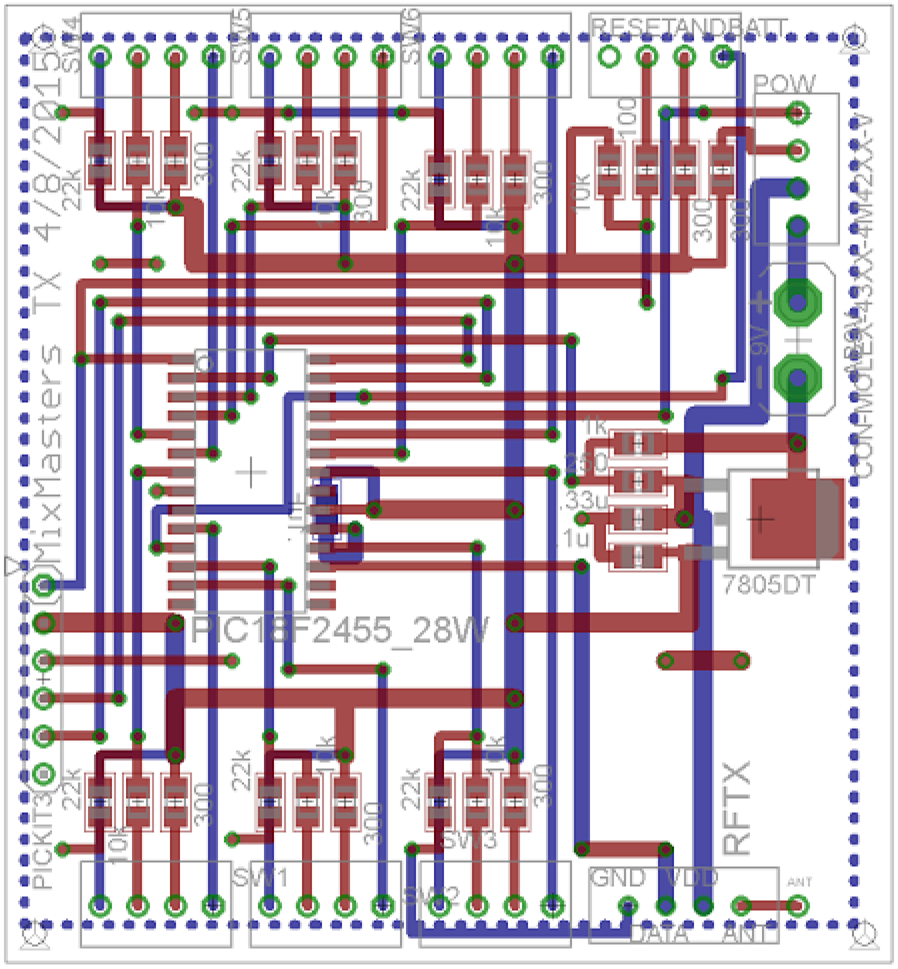


A.1.4 : Mixer Side Board

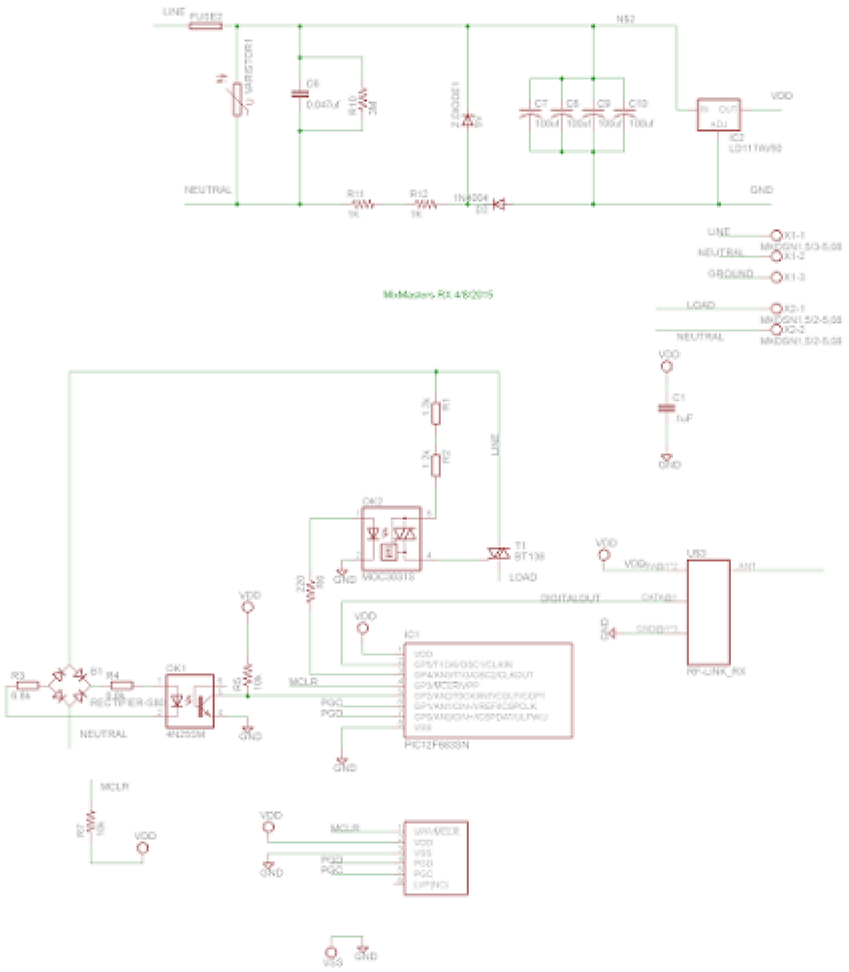
A.2 Amended Schematics and Boards



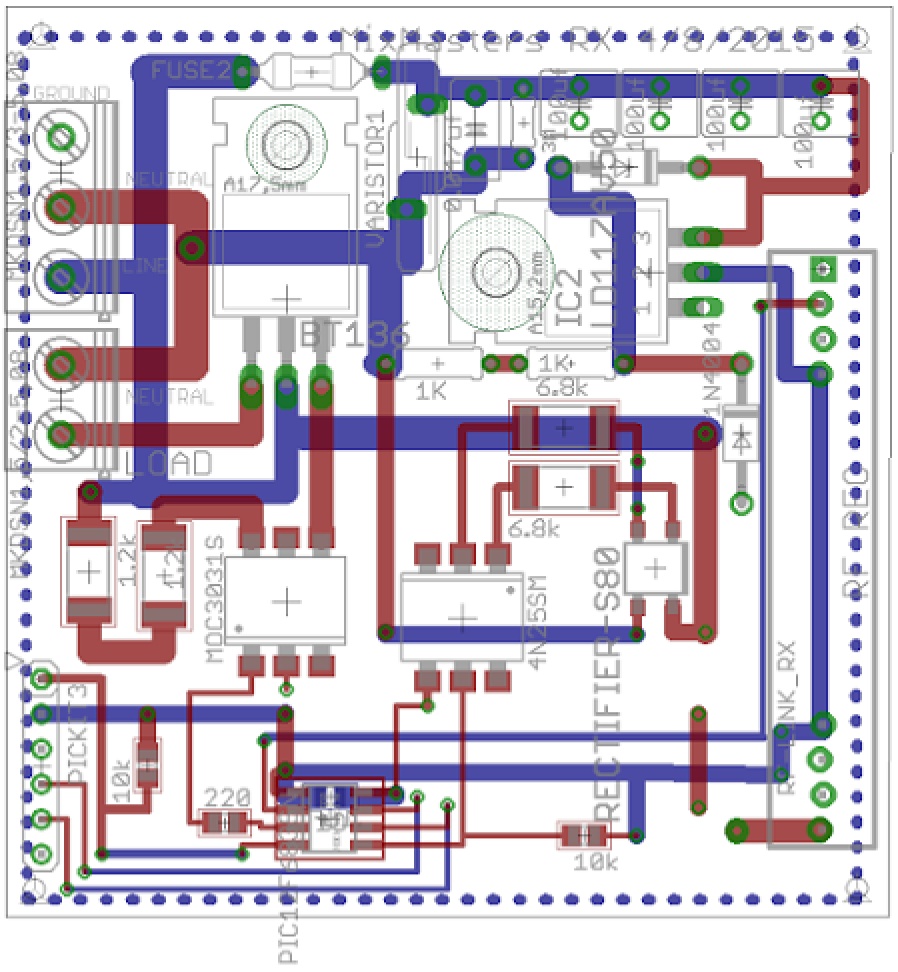
A.2.1: Remote Side Schematic (amended)



A.2.2 : Remote Side Board (amended)



A.2.3 : Mixer Side Schematic (amended)



A.2.4: Mixer Side Board (amended)

Appendix B : Full Software Listing

B.1 Remote Side Code (PIC18F2450)

```
/*
 * File:   remoteSide.c
 * Author: kdube
 *
 * Created on March 18, 2015, 2:30 PM
 */

#include <stdio.h>
#include <stdlib.h>
#include <adc.h>
#include <xc.h> // system header file
//#include <pic.h>
//#include <pic18f2450.h>
//#include "configbitsrev2014vC.h" // config bits
#include "SDlibMM.h" // library header file
//#include <sys/attribs.h>
#define _XTAL_FREQ 31000
#pragma config FOSC = INTOSCIO_EC
#pragma config WDT = OFF
#pragma config MCLRE = ON



/* This is the remote side code for MixMasters : EE 2015 Senior
Design Project
 * This will contain:
 * 1) Button interface to microcontroller
 * 2) Indicator lights for last pressed button
 * 3) Send commands corresponding to different speeds on receive side
 * 4) Change batteries indicator
 */

/*Button interface:
 * PORTB
 */
//void countUp(void);
//void countDown(void);
void serial_inittx(int);
void setLights(unsigned char);
//void delay_msMM(int);
int clockspeed=31000;
unsigned char counter=0;
unsigned char onLight=0;
unsigned char data=0x17;
int count=0;
int count2=0;
int check=0;
unsigned char belowseven;
unsigned char below;
int main(int argc, char** argv) {
    //CLFR PORTA;
    //MOVLW 0Fh;
    //MOVWF ADCON1;
    //MOVLW 0CFh;
    //MOVWF TRISA;
    UCONbits.USBEN=0;
  //  TBLWT
    //MY Initialize
    OSCCONbits.SCS=3; //UNSURE
    ADCON1bits.PCFG=0xE;
    //TRISA = 0b10101011; //Buttons and LEDs 1, 4, 5
    TRISAbits.TRISA0=1; //ADC for batt life
    TRISAbits.TRISA1=1; //button 5
    TRISAbits.TRISA2=0; //led 5
    LATAbits.LATA2=1; //set led 5 off
    TRISAbits.TRISA3=1; //button 4
    TRISAbits.TRISA4=0; //led 4
    LATAbits.LATA4=1; //set led 4 off
    TRISAbits.TRISA5=1; //button 1
    TRISAbits.TRISA6=0; //led 1
    LATAbits.LA6=0; //set led 1 off

    //TRISC = 0b11011011;

    TRISCbits.TRISC1=1; //button 2
    TRISCbits.TRISC2=0; //led 2
    LATCbits.LATC2=1; //set led 2 off
    //TRISCbits.TRISC4=1; //unused, always an input
    //TRISCbits.TRISC5=0; //unused, input only
    TRISCbits.TRISC6=0; //RFTX
    TRISCbits.TRISC7=1; ///button 3
    TRISBbits.RB1 = 0;  //LED 3
    LATBbits.LATB1=1; //set led 3 off
    TRISBbits.RB2 = 1;  //Button 6
    TRISBbits.RB3 = 0;  //LED 6
    LATBbits.LATB3=1; //set led 6 off
    //TRISBbits.RB4 = 0;  //LED for Power On, set as Output
    //LATBbits.LATB4= 0; //turn on power LED
    //TRISBbits.RB5 = 1;  // LED for Low Battery
    //LATBbits.LATB5 = 0; //turn off low battery indicator initially
    TRISCbits.TRISC0=0; //LED Battery
    LATCbits.LC0=1;


    int baud=2583;
    serial_inittx(baud);
//Arnaud's Initialize
    float adc_result;

    unsigned short count=0;
  //OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD,ADC_CH0 &
ADC_INT_OFF, 0); //open adc port for reading
    ADCON2bits.ADFM=1; //Right justified
    ADCON2bits.ACQT=0x5; //A/D Aquisition Time Select Bits 12TAD
    ADCON2bits.ADCS=0x2; //A/D Conversion Clock Select Bits
    ADCON1bits.PCFG = 0b1110;
    ADCON1bits.VCFG=0x0;  //Vref+ = VDD and Vref- = VSS


    //LATE = ~(data);
    while(1){
        if (PORTCbits.RC1){ //Button 2
            //Send 'Speed 2' Signal
            //countDown();
            onLight=2;
            data=0x27;
            setLights(onLight);
            count=0;
        }
        if (PORTCbits.RC7){ //Button 3
            //Send 'Speed 3' Signal
            onLight=3;
            data=0x36;
```

```
                setLights(onLight);
                count=0;
            }
            if (PORTAbits.RA3){ //Button 4
                onLight=4;
                data=0x4B;
                setLights(onLight);
                count=0;
                //Send 'Speed 4' Signal
            }
            if (PORTAbits.RA1){ //Button 5
                //Send 'Speed 5' Signal
                onLight=5;
                data=0x5A;
                setLights(onLight);
                count=0;
            }
            if (PORTBbits.RB2){ //Button 6
                //Send 'Speed 6' Signal
                onLight=6;
                data=0x69;
                setLights(onLight);
                count=0;
            }
            if (PORTAbits.RA5){ //Button 1
                //Send 'Stop' Signal
                //countUp();
                onLight=1;
                data=0x17;
                setLights(onLight);
                count=0;
            }
//          TXREG=0x55;
//          TXSTAbits.TXEN=1;
//          while(!TXSTAbits.TRMT);
            TXREG=data;
            TXSTAbits.TXEN=1;
            while(!TXSTAbits.TRMT);
            TXREG=0xFF-data;
            //TXSTAbits.TXEN=1;
            while(!TXSTAbits.TRMT);
            TXREG=data;
            TXSTAbits.TXEN=1;
```

```
        while(!TXSTAbits.TRMT);
        TXREG=0xFF-data;
        //TXSTAbits.TXEN=1;
        while(!TXSTAbits.TRMT);
        TXSTAbits.TXEN=0;
        //delay_msMM(30);
        __delay_ms(30);
        check=check+1;
        if (check>=60){
        ADCON0bits.CHS=0;
        ADCON0bits.GO_DONE=1;
        ADCON0bits.ADON=1;

        while(ADCON0bits.GO_DONE !=0);
        ADCON0bits.GO_NOT_DONE=1;
        adc_result = (((unsigned int)ADRESH)<<8)|(ADRESL);


        if (adc_result<0x028B){ //if less than 3 volts (battery at
6V)
            count2=count2+1;
            if (count2<10){
                LATCbits.LATC0 = 0;
            }
            else if ((count2>=10)&&(count2<20)){
                LATCbits.LATC0 = 1;
            }
            else if  (count2>=20){
                count2=0;
            }
        }
        else if (adc_result<0x02A8){ //if less than 4 volts (battery
at 8V) //VALUE IS TOO HIGH
           LATCbits.LATC0 = 0;
        }
        else{
            LATCbits.LATC0 = 1;
        }
        check=0;
        }
        count=count+1; //below:disable mixer if 10 minutes has passed
        if (count>=15000){//10 minutes? 600 seconds/(30ms delay+~10ms
data)
```

```
            onLight=1;
            data=0x17;
            setLights(onLight);
            count=0;
          }
      }

    return; //(EXIT_SUCCESS);
}

//void countUp(void){
//    delay_ms(250);
//
//    counter=counter+1;// increasing count value
//    if (counter == 64){
//        counter = 0;
//    }
//
//    onLight = 128;
//}
//void countDown(void){
//    delay_ms(250);
//
//    if (counter == 0){
//        counter = 64;
//    }
//    counter=counter-1;// increasing count value
//
//    onLight = 64;
//}

void serial_inittx(int baud){
/* setting bits for the U6MODE register */
//U3MODE=0; //Clear all the bits in the register to 0.
    TXSTAbits.SYNC=0;
    TXSTAbits.BRGH=1;
    BAUDCONbits.BRG16=1; //Set the UART to high speed mode
    RCSTAbits.SPEN=1; //Is this right?  Enables serial port.

BAUDCONbits.TXCKP=1; //TX data is inverted
/*Setting bits for the U6STA register*/
//U3STA=0; //Clear all the bits in the register to 0.
//U3STAbits.URXEN=1; //Enable the Receive
```

```
//U3STAbits.UTXEN=1; // Enable the Transmit

/*Get the Peripheral Bus Clock Frequency*/
unsigned long fpb;
//fpb=get_pb_clock();
fpb=clockspeed;
/*Set up the baud rate register*/
unsigned long SPB=(fpb/(4*baud))-1; //floor function rounds down to a
whole number
    SPBRG=SPB&0xFF;
//SPBRG=0x33;
    SPBRGH=SPB>>8; //floor function rounds down to a whole number
//SPBRGH=0x00;
//U3MODEbits.ON=1; //Enable the UART
}

void setLights(unsigned char onLight){

    switch (onLight){
        case 1:
            LATAbits.LA2 = 1; //LED5
            LATAbits.LA4 = 1; //LED4
            LATCbits.LC2 = 1; //LED2
            LATBbits.LB1 = 1; //LED3
            LATBbits.LB3 = 1; //LED6
            LATAbits.LA6 = 0; //LED1
            break;
        case 2:
            LATAbits.LA2 = 1; //LED5
            LATAbits.LA4 = 1; //LED4
            LATAbits.LA6 = 1; //LED1
            LATBbits.LB1 = 1; //LED3
            LATBbits.LB3 = 1; //LED6
            LATCbits.LC2 = 0; //LED2
            break;
        case 3:
            LATAbits.LA2 = 1; //LED5
            LATAbits.LA4 = 1; //LED4
            LATAbits.LA6 = 1; //LED1
            LATCbits.LC2 = 1; //LED2
            LATBbits.LB3 = 1; //LED6
            LATBbits.LB1 = 0; //LED3
            break;
        case 4:
            LATAbits.LA2 = 1; //LED5
            LATAbits.LA6 = 1; //LED1
            LATCbits.LC2 = 1; //LED2
            LATBbits.LB1 = 1; //LED3
            LATBbits.LB3 = 1; //LED6
            LATAbits.LA4 = 0; //LED4
            break;
        case 5:
            LATAbits.LA4 = 1; //LED4
            LATAbits.LA6 = 1; //LED1
            LATCbits.LC2 = 1; //LED2
            LATBbits.LB1 = 1; //LED3
            LATBbits.LB3 = 1; //LED6
            LATAbits.LA2 = 0; //LED5
            break;
        case 6:
            LATAbits.LA2 = 1; //LED5
            LATAbits.LA4 = 1; //LED4
            LATAbits.LA6 = 1; //LED1
            LATCbits.LC2 = 1; //LED2
            LATBbits.LB1 = 1; //LED3
            LATBbits.LB3 = 0; //LED6
            break;
    }
}

//void delay_msMM(int ms){
//    T2CONbits.TMR2ON=0;
//
//    //can only get factors of 2
//    //T1GCONbits.TMR1GE=0; //used in pic12f
//    PR2= ms*10^-3*clockspeed;
//    TMR2=0;
//    T2CONbits.TMR2ON=1;
//    while (TMR2<PR2); //TMR1=[TMR1H:TMR1L]
//    T2CONbits.TMR2ON=0;
//    TMR2=0;
//    }
```

B.2 Board Side Code (PIC12F1822)

```
/*
 * File:   MixerSide12mainJE.c
 * Author: mmarti29
 *
 * Created on April 19, 2015, 1:24 PM
 */

#include <stdio.h>
#include <stdlib.h>
#include <xc.h>
#include <pic12f1822.h>
#include "SDlibMM.h"
#include "configbitsMixerSide1242515"
#define _XTAL_FREQ 500000



/*
 *
 */
unsigned char data=0x17; //state 0
void serial_initrx(int baud);
int clockspeed=500000; //Med F INT OSC 500k
int count=0;
int test = 0;
int main(int argc, char** argv) { //void main(void){
    OSCCONbits.SCS=0b11; //chooses the internal oscillator
    OSCCONbits.IRCF=0x7; //default on reset, sets internal oscillator
to 500kHz
    //Init things
    APFCONbits.RXDTSEL=1; //RX on RA5
    INTCONbits.GIE = 1; // Global enable interrupts
    INTCONbits.INTE = 1; //external enable
    INTCONbits.PEIE = 1;
    //INTCONbits.IOCIE = 1;
    OPTION_REGbits.INTEDG = 1; //rising edge
    int baud_rate=2583;
    serial_initrx(baud_rate);
    ANSELAbits.ANSELA=0x00; //set all to digital
    // RCIF is PIR1bits.RCIF
```

```
TRISAbits.TRISA2 = 1;
TRISAbits.TRISA4 = 0;
TRISAbits.TRISA5 = 1;

while(1){
        if (test == 0){
            data = 0x17;
            test = 1;
            delay_ms(100);
        }
        else if (test == 1) {
            test = 2;
            data = 0x27;
            delay_ms(200);
        }
        else if (test == 2){
            test =3;
            data = 0x36;
            delay_ms(400);
        }
        else if (test==3) {
            data = 0x4B;
            test = 4;
            delay_ms(800);
        }
        else if (test==4) {
            data = 0x5A;
            test = 5;
            delay_ms(1000);
        }
        else if (test==5) {
            data = 0x69;
            test = 0;
            delay_ms(1000);
        }
        else {
            test = 0;
        }

        /*         RCSTAbits.CREN=1;
        while (!PIR1bits.RCIF);
        PIR1bits.RCIF=0;
```

```
            unsigned char datab=RCREG;
            switch (datab){
                    case 0x17:
                        data=0x17;
                        count=0;
                        break;
                    case 0x27:
                        data=0x27;
                        count=0;
                        break;
                    case 0x36:
                        data=0x36;
                        count=0;
                        break;
                    case 0x4B:
                        data=0x4B;
                        count=0;
                        break;
                    case 0x5A:
                        data=0x5A;
                        count=0;
                        break;
                    case 0x69:
                        data=0x69;
                        count=0;
                        break;
                    case 0x78:
                        data=0x78;
                        count=0;
                        break;
                }*/

        //data=0x17;

    }
    return (EXIT_SUCCESS); //return;
}

void serial_initrx(int baud){
    RCSTAbits.SPEN=0;
        TXSTAbits.BRGH=1;
    BAUDCONbits.BRG16=1;
     TXSTAbits.SYNC=0;
    unsigned long fpb = clockspeed;
    //fpb=get_pb_clock();
    //[SPBRGH:SPBRGL]=(fpb/(64*baud_rate))-1; //floor function rounds
down to a whole number
    unsigned long SPB=(fpb/(4*baud))-1; //floor function rounds down
to a whole number
    SPBRGL=SPB&0xFF;
    SPBRGH=SPB>>8;



    //PIE1bits.RCIE = 1; //receive interrupt enable bit





/*Set up the baud rate register*/

    RCSTAbits.SPEN=1;
//RCSTAbits.CREN=1; //continous receive enable bit
//U3MODEbits.ON=1; //Enable the URCSTAbitsART
}

void interrupt ZC(void){
        while(PORTAbits.RA2);
//        IFS0bits.INT1IF=0; //clearing flag
        switch (data){
            case 0x17:
                LATAbits.LATA4 = 0;
                break;
            case 0x27:
                delay_ms(4000);
                LATAbits.LATA4 = 1; // send pulse after 1ms
                delay_us(250);
                break;
            case 0x36:
                delay_us(3500);
                LATAbits.LATA4 = 1; // send pulse after 2ms
                delay_us(250);
                break;
            case 0x4B:
                delay_us(3000);
                LATAbits.LATA4 = 1; // send pulse after 3ms
                delay_us(250);
                break;
            case 0x5A:
                delay_us(2500);
                LATAbits.LATA4 = 1; // send pulse after 3ms
                delay_us(250);
                break;
            case 0x69:
                delay_us(2000);
                LATAbits.LATA4 =  1; // send pulse after 3ms
                delay_us(250);
                break;
        }
        //count=count++;


        PORTAbits.RA4 = 0;
        INTCONbits.INTF=0; //clearing flag


}
```

B.3: Board Side Code (PIC32)

```
/*
 * File:   RFmotorMain.c
 * Author: jerman
 *
 * Created on March 26, 2015, 2:16 PM
 */

#include <stdio.h>
#include <stdlib.h>
#include <xc.h>
#include "configbitsJEv2014.h"
#include "SDlib.h"
#include <sys/attribs.h>


/*
 *
 */

void serial_init3(void);
unsigned char data=0x17;
unsigned char datab=0x17;
int main(int argc, char** argv) {

    INTCONbits.MVEC = 1;
    IEC0bits.INT1IE=1;
    IPC1bits.INT1IP=4;
    __builtin_enable_interrupts();
//     T2CON=0x0000; //Stop timer and clear control register
//    T2CONbits.TCKPS0=1; // prescaler set at 256
//    T2CONbits.TCKPS1=1;
//    T2CONbits.TCKPS2=1;
//    TMR2=0x0000; //Clear timer register
//    PR2=0x2710; //Load period register
//    T2CONSET=0x8000; // Start timer
    TRISGbits.TRISG7=1;
    TRISD &= 0xFFFF; //D is input
    TRISB &= 0xFF00; //lower 8 bits outputs
    LATB &= 0xFF00; //set to 0
    TRISE &= 0xFF00; // lower 8 bits outputs
    LATE &= 0xFF00; //set to 0
    TRISDbits.TRISD7=0;

     serial_init3();
    while(1){
        U3STAbits.URXEN=1;
        while(!U3STAbits.URXDA);
        datab=U3RXREG;
        switch (datab){
            case 0x17:
                data=0x17;
                break;
            case 0x27:
                data=0x27;
                break;
            case 0x36:
                data=0x36;
                break;
            case 0x4B:
                data=0x4B;
                break;
            case 0x5A:
                data=0x5A;
                break;
            case 0x69:
                data=0x69;
                break;
//            case 0x78:
//                data=0x78;
//                break;
        }
    if ((data!=0x00)&&(data!=0xFF)){
    LATE = (~data);
    }
    }
    return (EXIT_SUCCESS);
}

void __ISR(7, IPL4AUTO) Int1hand(void){

        while(PORTDbits.RD8); //while(!PORTDbits.RD8);
        //IFS0bits.INT1IF=0; //clearing flag
        switch (data){
            case 0x17:
                PORTDbits.RD7 = 0;
                break;
            case 0x27:
                delay_ms(5);
                PORTDbits.RD7 = 1; // send pulse after 5ms
                delay_us(250);
                break;
            case 0x36:
                delay_ms(4);
                PORTDbits.RD7 = 1; // send pulse after 4ms
                delay_us(250);
                break;
            case 0x4B:
                delay_ms(3);
                PORTDbits.RD7 = 1; // send pulse after 3ms
                delay_us(250);
                break;
            case 0x5A:
                delay_ms(2);
                PORTDbits.RD7 = 1; // send pulse after 2ms
                delay_us(250);
                break;
            case 0x69:
                delay_ms(1);
                PORTDbits.RD7 = 1; // send pulse after 1ms
                delay_us(250);
                break;
        }

        PORTDbits.RD7 = 0;
        IFS0bits.INT1IF=0; //clearing flag


}

void serial_init3(void){
/* setting bits for the U6MODE register */
U3MODE=0; //Clear all the bits in the register to 0.

U3MODEbits.BRGH=1; //Set the UART to high speed mode
U3MODEbits.RXINV=1;
/*Setting bits for the U6STA register*/
```

```
U3STA=0; //Clear all the bits in the register to 0.
//U3STAbits.URXEN=1; //Enable the Receive
//U6STAbits.UTXEN=1; // Enable the Transmit
U3STAbits.URXEN=1; //Enable the Receive
/*Get the Peripheral Bus Clock Frequency*/
unsigned long fpb;
fpb=get_pb_clock();

/*Set up the baud rate register*/
U3BRG=(fpb/(4*2400))-1; //floor function rounds down to a whole
number
U3MODEbits.ON=1; //Enable the UART
}
```

Appendix C: Relevant Datasheets

Microcontrollers

PIC18F2450: http://ww1.microchip.com/downloads/en/DeviceDoc/39760d.pdf
PIC12F1822: http://ww1.microchip.com/downloads/en/DeviceDoc/40001413E.pdf
PIC32MX795: https://www.sparkfun.com/datasheets/Components/SMD/PIC32MX.pdf

Polycase

8.5” x 5” x 3” case with gasket: http://www.polycase.com/dc-58p-gasket

RF Modules

Transmitter:
http://cdn.sparkfun.com/datasheets/Wireless/General/TWS-BS-3_433.92MHz_ASK_RF_Transmitter_Module_Data_Sheet.pdf
Receiver:
http://cdn.sparkfun.com/datasheets/Wireless/General/RWS-371-6_433.92MHz_ASK_RF_Receiver_Module_Data_Sheet.pdf