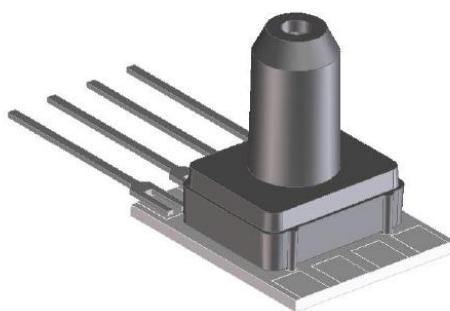


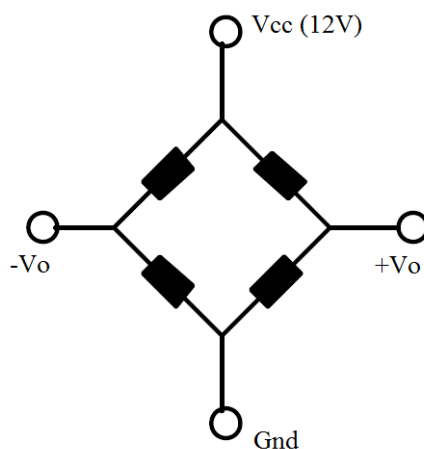
Dispositivo de Controle de Sopro

Muitos usuários de cadeiras de rodas motorizadas não possuem movimentos nas mãos, ou seja, não podem controlar a cadeira com um joystick manualmente. Uma das soluções para esse problema é o acionamento da cadeira através de sopros. O dispositivo de controle de sopro é baseado em um sensor de pressão que emite um valor de tensão elétrica proporcional a um determinado valor de pressão introduzido no bocal de entrada.

O desenvolvimento do dispositivo de controle de sopro iniciou-se com as medições de tensão no sinal de saída de um sensor de pressão Hgrade-Mini de 5psi mostrado na figura abaixo.

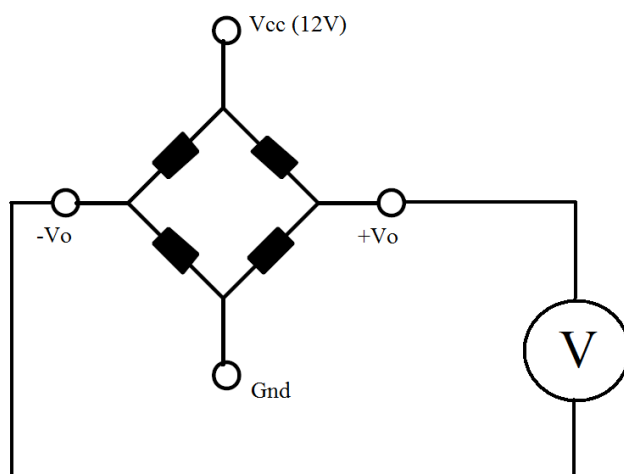


O circuito do sensor é equivalente a uma ponte de sensores piezelétricos como mostra a figura:



O sensor deve ser alimentado por 12V na entrada (Vcc) sendo que o terra deve ser ligado ao (Gnd). Dessa forma, ao ser inserida uma pressão de ar no bocal de entrada, surgirá uma tensão da ordem de até 60mV nos terminais ($-V_o$) e ($+V_o$).

Segundo as definições que constam nos requisitos do projeto quanto ao dispositivo de controle de sopro, o mesmo deverá ter a sensibilidade para definir um sopro forte, sopro fraco, sucção forte e sucção fraca. Para validar o funcionamento do sensor com essas definições, foi realizado um teste com uma alimentação de 5V e depois 12V na entrada (Vcc) e um multímetro nos terminais ($-V_o$) e ($+V_o$) como mostra o diagrama abaixo:



Os dados obtidos estão na tabela abaixo:

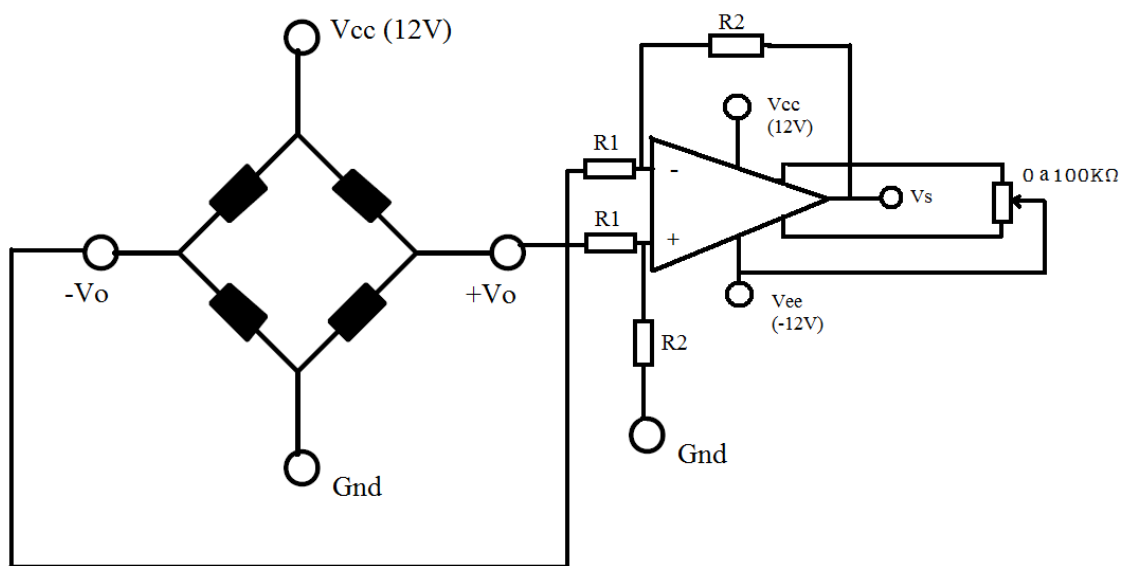
Vcc	Tensão em Pressão Mínima		Tensão em Pressão Máxima	
	Soprar	Sugar	Soprar	Sugar
5V	0,2 mV	-0,5 mV	3 mV	-5 mV
12V	1 mV	-2 mV	6 mV	-13 mV

Os padrões de entrada dos módulos de controle dispõem de uma alimentação de 12V, o que nos permite utilizar a tensão nominal do sensor, além do que, tensão na saída tem um valor muito baixo com 5V de alimentação.

O dispositivo de controle de sopro deve passar ao módulo de controle uma informação semelhante ao dispositivo de controle de chaves uma vez que será utilizada a mesma entrada. Com isso deve-se utilizar um microcontrolador para transformar o sinal analógico gerado pelo sensor em sinal digital de modo a diferenciar um sopro forte de um sopro fraco e uma sucção forte de uma sucção fraca.

O microcontrolador dispõe de entradas nas quais recebem valores analógicos apenas em uma polaridade de sinal, o que significa que os sinais de tensão negativos não podem ser utilizados, além do mais, os níveis de tensão ainda são muito baixos e devem ser amplificados para que o microcontrolador possa receber o sinal. Para resolver esses 2 problemas deve se utilizar um Amplificador Operacional com ajuste de Offset de modo que o sinal se estabilize em 2,5V. Ao “soprar forte” a tensão chegaria em 5V e ao “sugar forte” a tensão chegaria a 0V.

Alguns testes foram realizados com o Amplificador Operacional LM741, porém não se levou em consideração o ganho exato para que os valores de tensão se aproximassem de 5V de diferença entre os extremos. Os valores dos resistores foram obtidos experimentalmente obedecendo a equação de um circuito subtrator com um potenciômetro entre os pinos 1 e 5 do amplificador operacional para fazer o ajuste fino do offset. O circuito do sensor e amplificador operacional ficou seguinte configuração:



Sendo:

$$V_s = [(+V_o) - (-V_o)] \times \frac{R_2}{R_1}$$

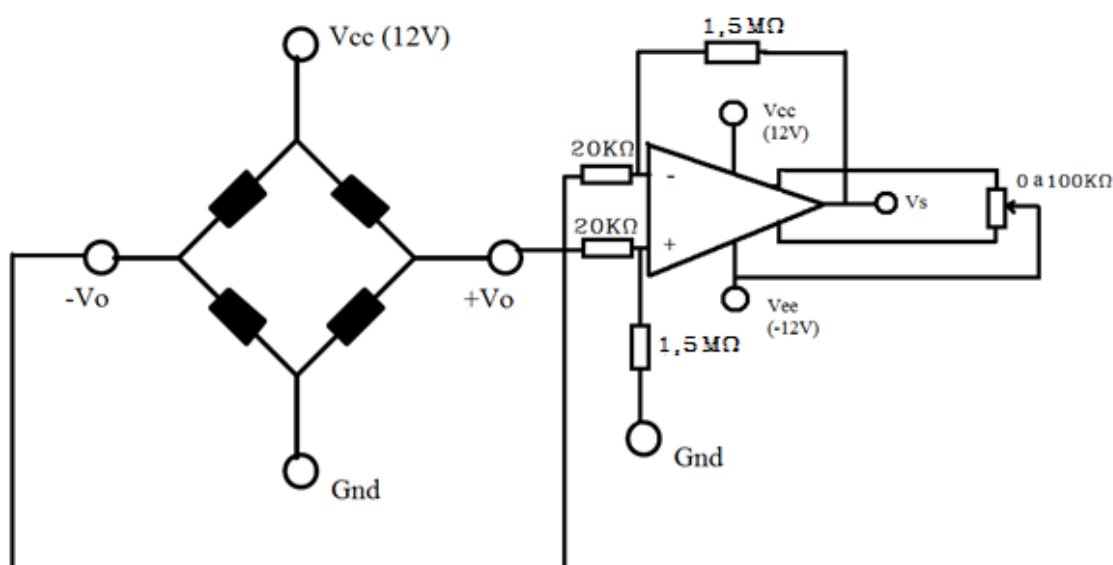
Inicialmente foram atribuídos os seguintes valores para os resistores: $R_1=20K\Omega$ e $R_2=1,5M\Omega$. Os valores de V_s foram medidos conforme a pressão introduzida no sensor e os resultados obtidos estão na tabela abaixo:

Offset	Tensão em Pressão Mínima		Tensão em Pressão Máxima	
	Soprar	Sugar	Soprar	Sugar
0V	2,2V	-1,3V	9,9V	-11,6V
2,5V	4V	-1,2V	11,5V	-10,7V

Pode-se perceber que o ganho do circuito foi bem alto, porém o ajuste de offset não regulou bem os extremos. Os valores se mantinham praticamente os mesmos, além do que foram realizados outros testes com outros valores de resistores para diminuir o ganho e observou o mesmo problema, ou seja, os extremos diminuía com o ganho, mas não se alteravam com o ajuste de offset. O offset altera apenas o valor da tensão quando não é introduzida uma pressão na entrada do sensor, ou seja, apenas o ponto de neutro.

Com diversas pesquisas realizadas, observou-se que o melhor amplificador operacional para se ter um bom ajuste de offset é o TL081. Com a pinagem idêntica ao LM741, o circuito foi mantido, apenas foi substituído o circuito integrado do amplificador operacional. Os resultados foram bem significativos, o problema do ajuste de offset foi resolvido.

Os valores dos resistores para que se mantivesse uma tensão V_s que variasse de 0V a 5V estão mostrado no diagrama abaixo:

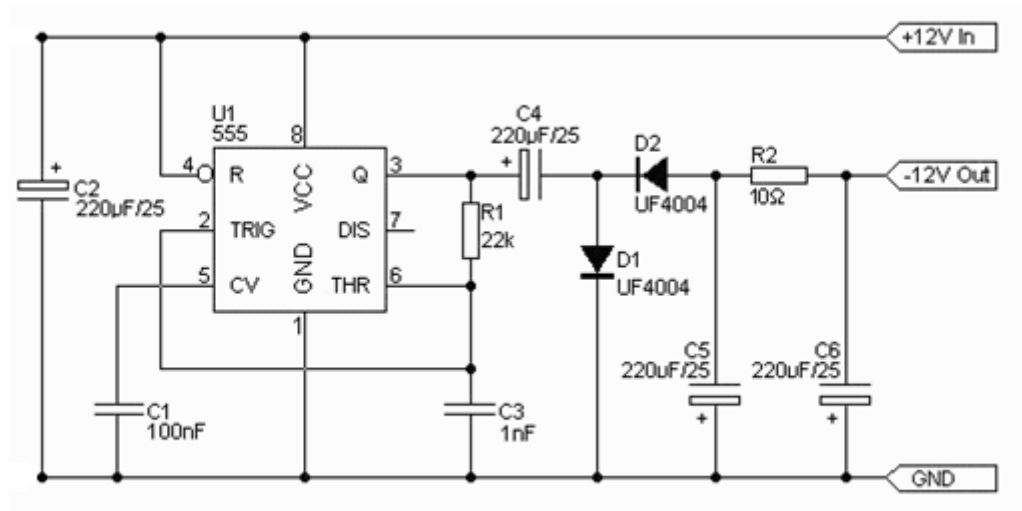


Em um primeiro teste o offset foi ajustado para a tensão de saída se mantivesse em 0V sem pressão introduzida e depois, no segundo teste, foi feito o ajuste

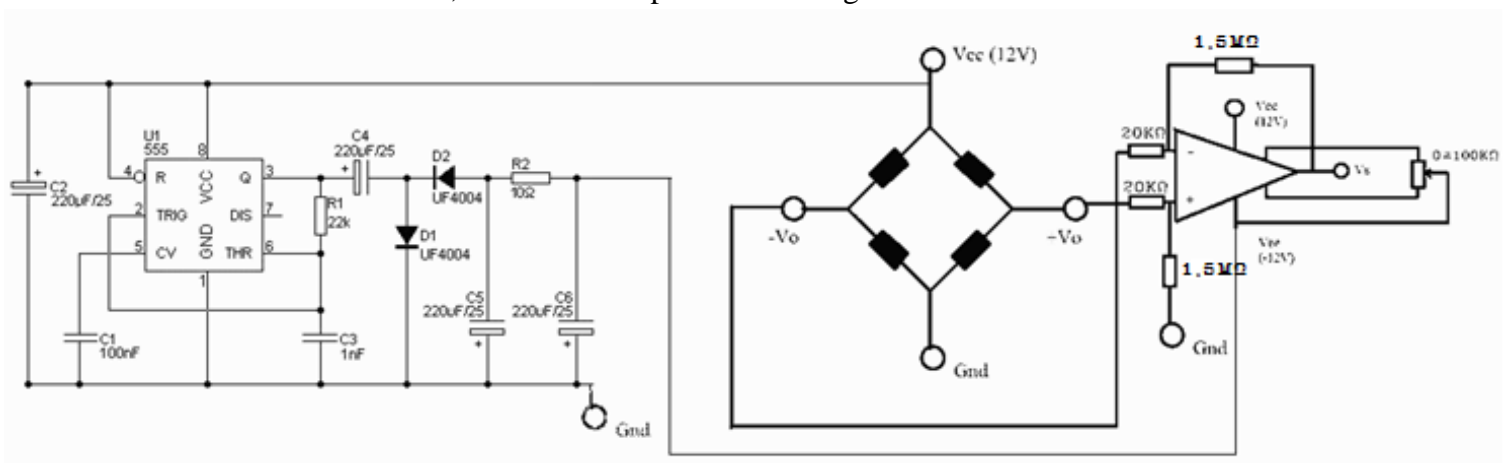
do offset para que a tensão mínima não fosse inferior a 0V e a tensão máxima não superior a 5V e os resultados dos testes estão na tabela abaixo:

Offset	Tensão em Pressão Mínima		Tensão em Pressão Máxima	
	Soprar	Sugar	Soprar	Sugar
0V	1,2V	-1,3V	2,2V	-2,3V
2,5V	3,7V	1,2V	4,7V	-0,01V

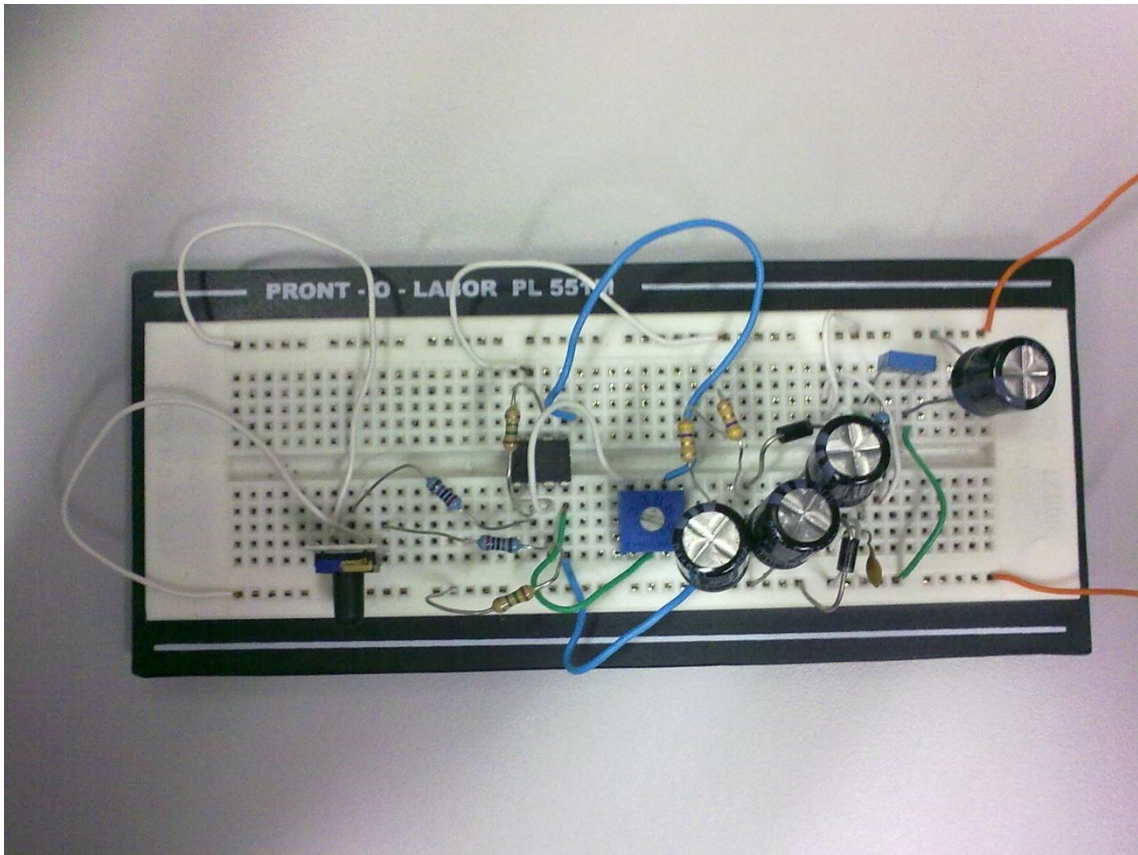
Com estes resultados pode-se definir esse como o circuito de entrada do microcontrolador, porém há outro problema que ainda não foi mencionado que é o fato de ter que utilizar uma fonte de -12V na entrada Vee do amplificador operacional. Em testes de laboratório é possível obter essa tensão facilmente, mas para um módulo na qual a tensão de alimentação fornecida é de apenas 12V será necessário um outro circuito capaz de gerar essa tensão de -12V. O diagrama abaixo mostra um circuito com um NE555 capaz de gerar a tensão negativa para a entrada Vee do Amplificador Operacional.



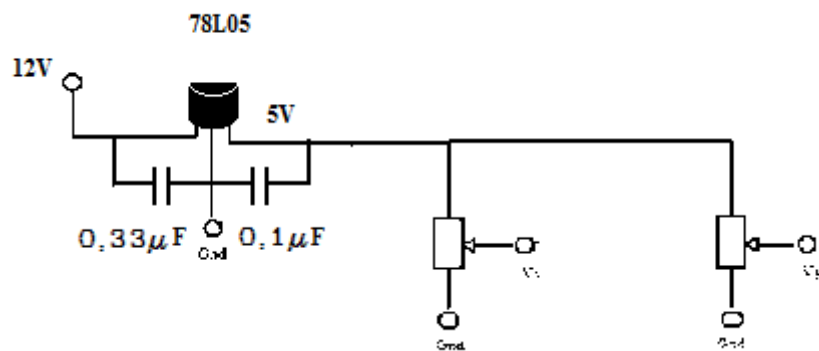
Assim, o circuito completo fica da seguinte forma:



A imagem a seguir mostra o circuito montado em um Proto Board.

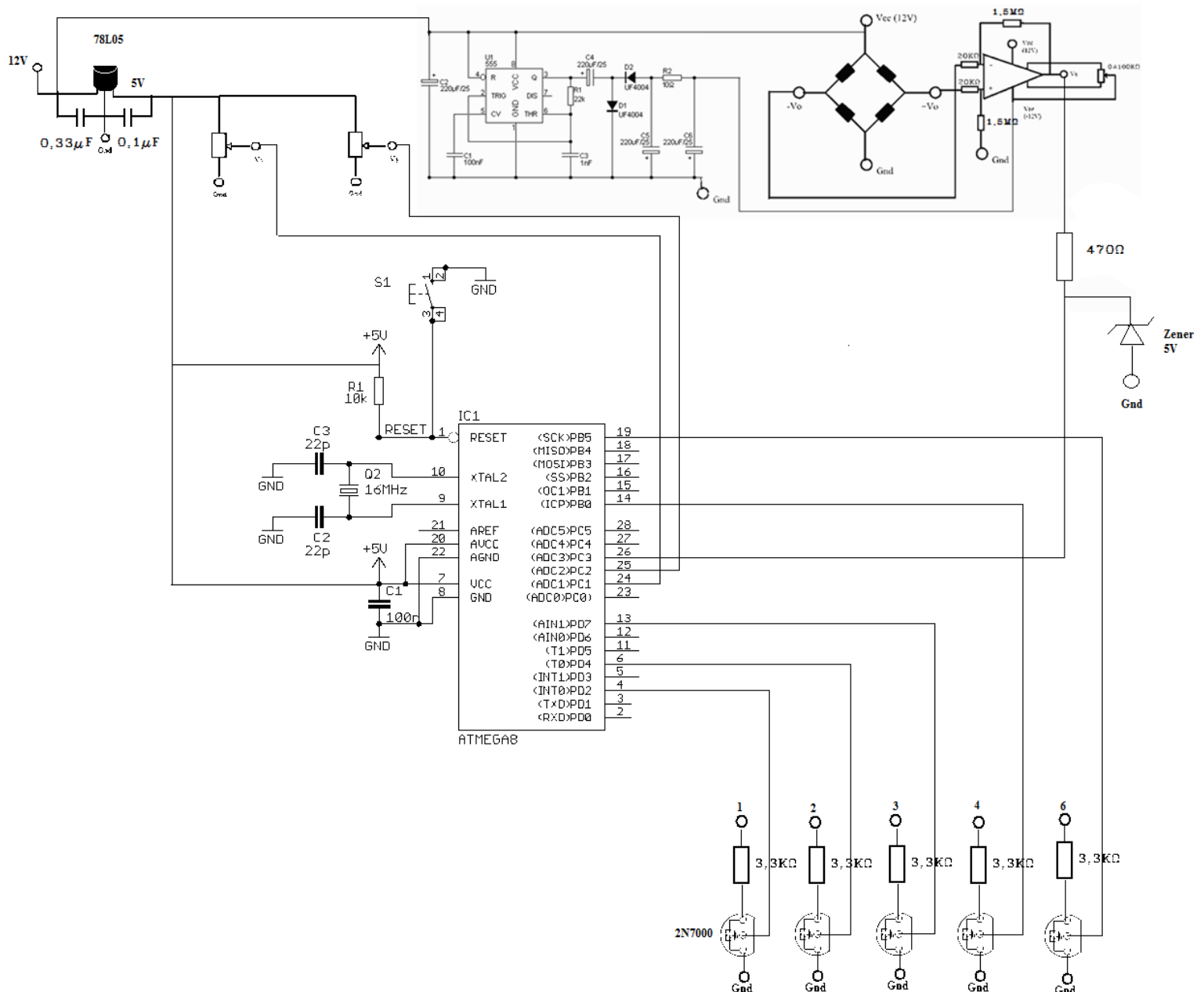


Outro circuito que deve ser implementado é o de calibração no qual o usuário poderá ajustar o ponto de transição entre sopro forte e sopro fraco bem como a sucção forte e sucção fraca. Para isso pensou-se na utilização de dois potenciômetros que fornecem níveis de tensão variados para o microcontrolador. Um potenciômetro ajusta o sopro e o outro ajusta a sucção. O circuito de calibração fica da seguinte forma:

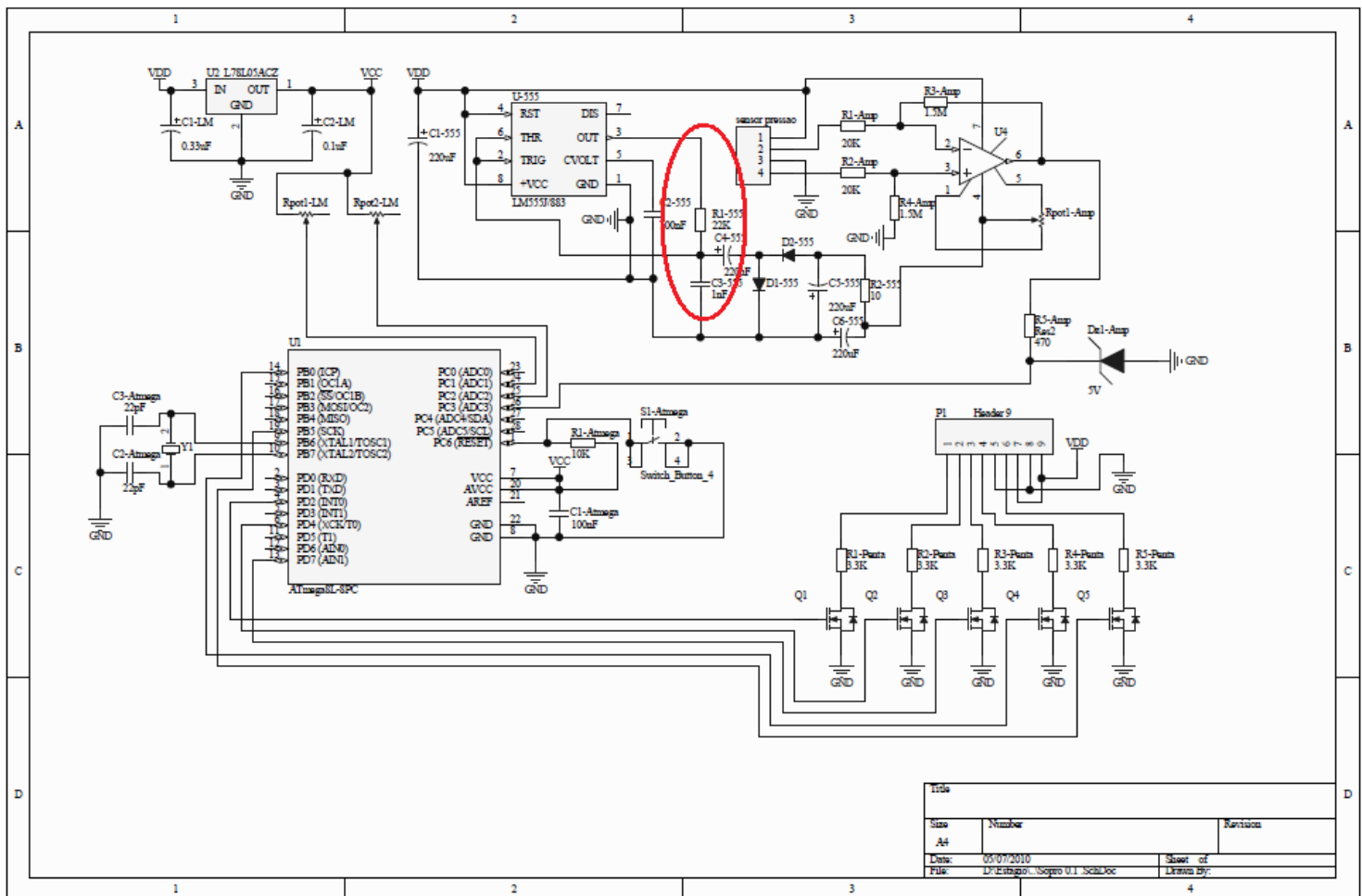


Como um protótipo do dispositivo de controle de sopro o microcontrolador que deverá ser utilizado é um ATMEGA8. Inicialmente para fins de testes de programação, o circuito para utilizar o microcontrolador é um Arduino no qual foram definidas entradas analógicas e saídas digitais para serem aplicadas ao circuito do sensor.

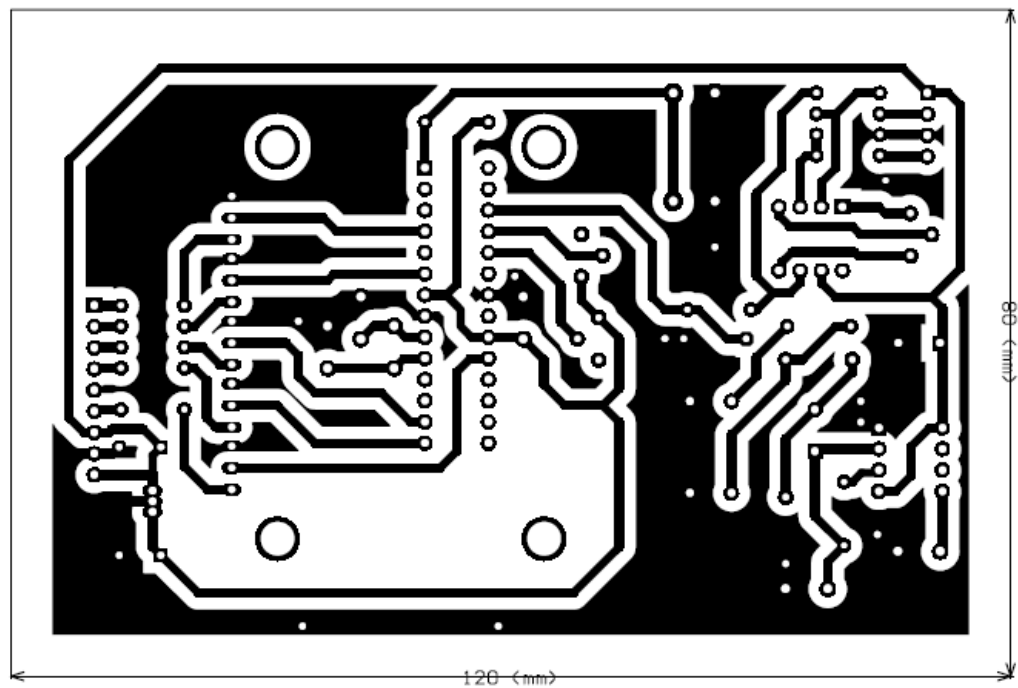
Assim sendo, o circuito completo com junto com o microcontrolador e o circuito de calibração ficou da seguinte forma:



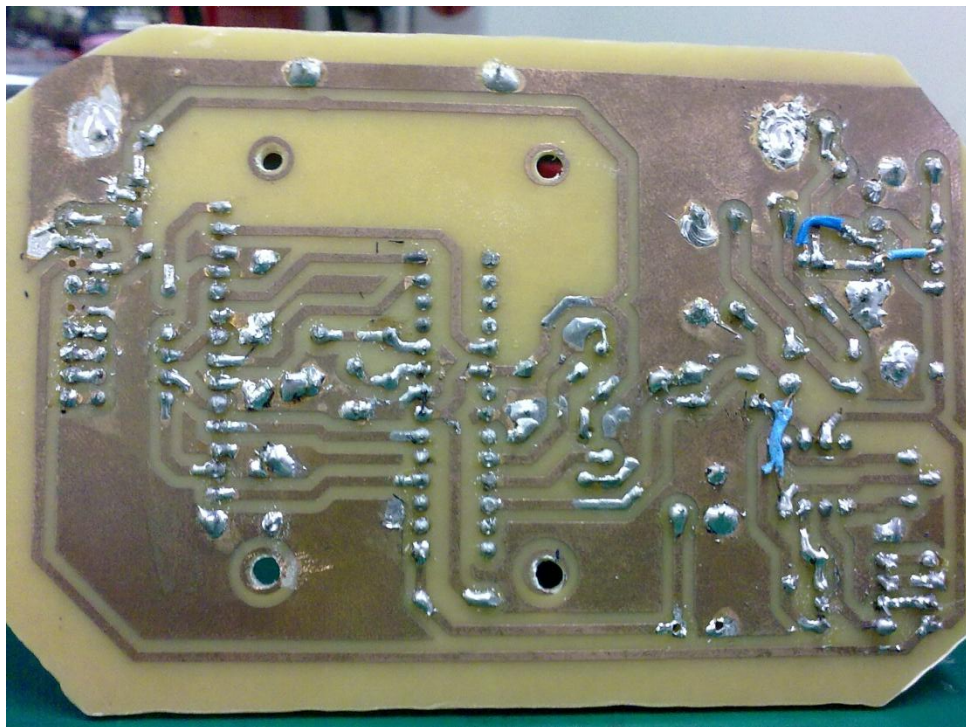
Este mesmo circuito foi recriado em um software de design para placa de circuito impresso. A figura seguinte mostra o circuito feito pelo Software.

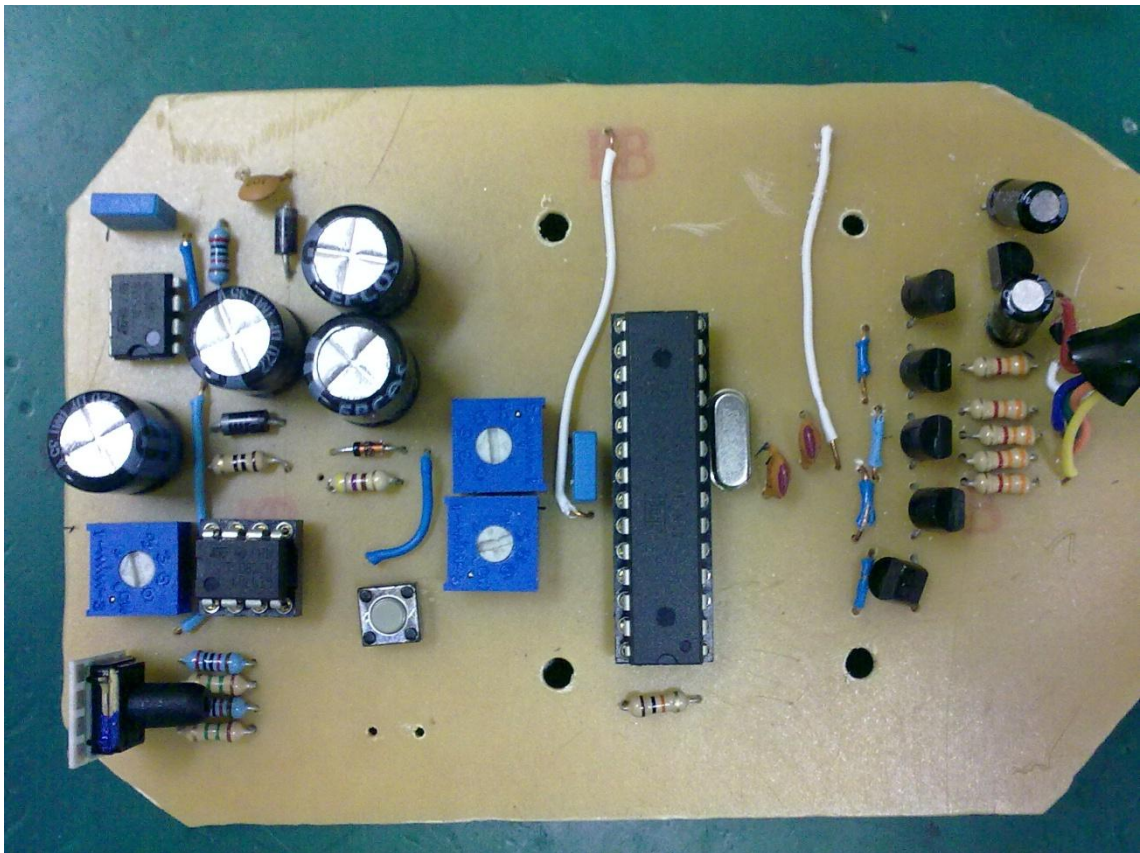
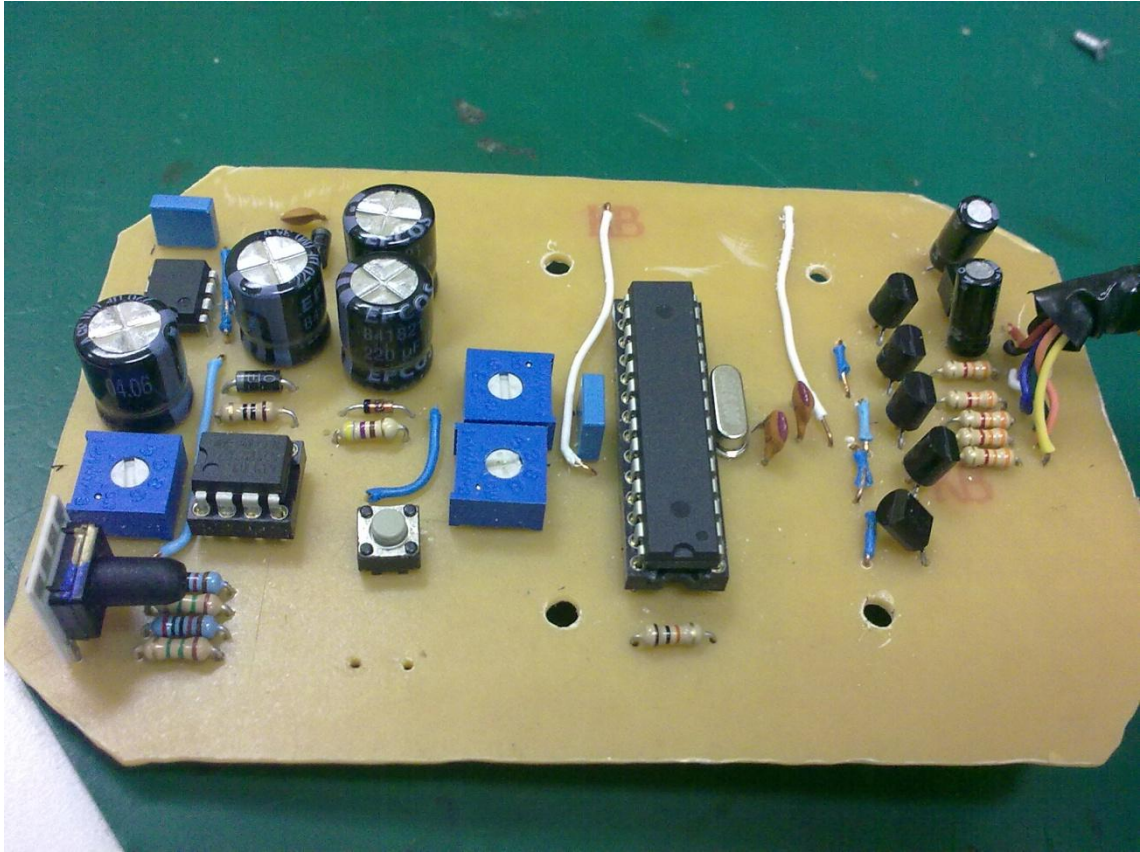


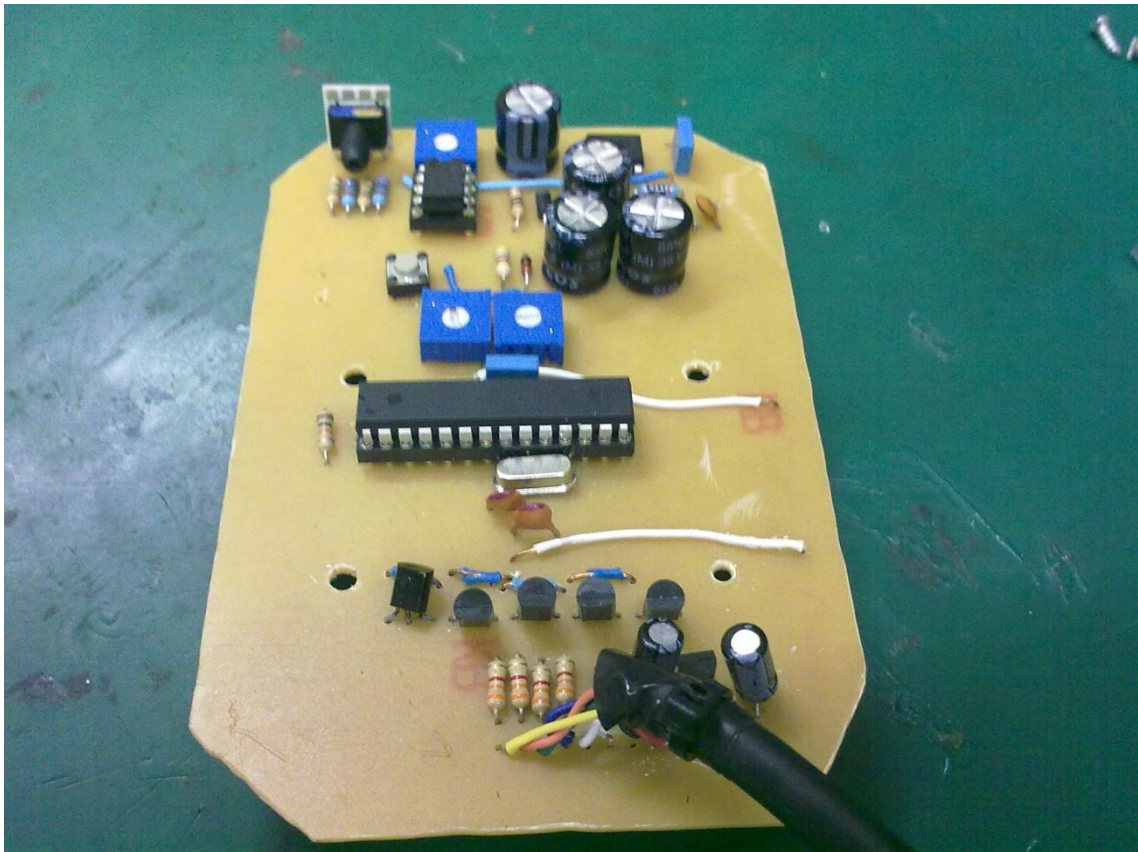
O circuito foi refeito, porém com um pequeno erro na região circulada da figura, a ligação do capacitor e do resistor está na posição incorreta. O erro foi corrigido posteriormente na própria placa, porém, o circuito foi roteado com esse erro. A figura seguinte mostra o circuito roteado.



Após a corrosão e soldagem dos componentes a placa ficou do modo como mostram as figuras seguintes.







Para fechar a placa foi utilizada uma caixa e uma conexão para mangueira. As margens da placa foram lixadas para encaixar a mesma na caixa. Uma mangueira foi utilizada para conectar o sensor de pressão à conexão.





A caixa com a tampa e os parafusos ficou da seguinte forma:



Na parte superior foi colocada uma etiqueta indicativa com o nome “Interface de Sopro” e o logotipo do LSI-TEC.



Assim, o dispositivo de Interface de Sopro fica com a seguinte característica.



Código do Microcontrolador

O código para programação do microcontrolador foi feito para que o mesmo servisse para acionar um controle de chaves. Cada intensidade do sopro é proporcional a uma tensão que alimenta uma das entradas analógicas do microcontrolador. As outras entradas analógicas foram utilizadas para receber os níveis de tensão fornecidos pelos potenciômetros de calibração, de modo que a tensão em cada entrada corresponde ao que se pode chamar de “limite” entre sopro fraco e sopro forte bem como sucção fraca e sucção forte.

As saídas do microcontrolador são digitais e fornecem níveis altos de tensão para cada mosfet ligado ao módulo de controle. Cada saída corresponde a direção desejada da cadeira. A configuração das direções em relação a intensidade do sopro ficou da seguinte forma: sopro forte faz a cadeira seguir para frente, sopro fraco faz a cadeira girar para a direita, sucção fraca faz a cadeira girar para a esquerda e sucção forte faz a cadeira andar para traz.

Outra função importante é o quinto botão do controle de chaves padrão que é o acionamento do menu principal para configuração da cadeira. Para ativar o quinto botão foi atribuída a ação de sugar uma vez rapidamente.

Assim o código da programação do microcontrolador feito no Arduino está disponível nas páginas seguintes.

/*

Geração de sinais esquema penta para interface de sopro.
O programa gera quatro sinais: FRONT, BACK, LEFT, RIGHT.
Cada um dos sinais representa uma direção no esquema 'penta'.

Ao contrário do joystick analógico, a interface de sopro recebe apenas um sinal e gera quatro sinais de saída.

Esta versão permite calibração através de 2 potenciômetros; cada um deles é responsável por controlar os níveis de entrada para o sopro e para a sucção.

*/

```
#define Back 0
#define Front 1
#define Left 2
#define Right 3
#define Dead 4
#define Menu 5
```

```
// Frente: sopro forte ( >650 )
// Direita: sopro fraco ( 515~650 )
// Esquerda: sucção fraca ( 420~170 )
// Ré: sucção forte ( <170 )
// Acesso ao menu: 2 sopros seguidos
```

```
// Definição dos valores padrão para limites de sopro e sucção
#define CONST_SOPRO_FORTE 650
#define CONST_SOPRO_FRACO_MAX 650
#define CONST_SOPRO_FRACO_MIN 515
#define CONST_SUCCAO_FRACO_MAX 420
#define CONST_SUCCAO_FRACO_MIN 170
#define CONST_SUCCAO_FORTE 170
```

```
// Entradas analógicas
int analogPin = 3;           // pino de entrada para o sensor de pressão
int calibrationPin1 = 1;     // pino do potenciômetro de calibração (sopro)
int calibrationPin2 = 2;     // pino do potenciômetro de calibração (sucção)
```

```
// Saídas digitais
int backPin = 4;
int frontPin = 2;
int leftPin = 7;
int rightPin = 8;
int menuPin = 13;
```

```
// Variáveis globais
int analogVal = 0;           // armazena o valor da entrada do sensor de pressão
int sopro_forte = CONST_SOPRO_FORTE;
int sopro_frac_max = CONST_SOPRO_FRACO_MAX;
```

```

int sopra_fraco_min = CONST_SOPRO_FRACO_MIN;
int succao_fraco_max = CONST_SUCCAO_FRACO_MAX;
int succao_fraco_min = CONST_SUCCAO_FRACO_MIN;
int succao_forte = CONST_SUCCAO_FORTE;
int actual_state = Dead;
int last_state = Dead;
int enter_menu = 0;

// contadores
int pico1 = 0;
int vale1 = 0;
int menu_timeout = 0;

void setup() {
  pinMode(backPin, OUTPUT);
  pinMode(frontPin, OUTPUT);
  pinMode(leftPin, OUTPUT);
  pinMode(rightPin, OUTPUT);
  pinMode(menuPin, OUTPUT);
  Serial.begin(9600);
}

int updateCalibration(){
  int ajuste_sopro = analogRead(calibrationPin1);
  int ajuste_succao = analogRead(calibrationPin2);
  //Serial.print("ajustes: ");
  //Serial.println(ajuste_sopro, ajuste_succao);

  // ajusta a escala dos sensores para -128 a 127
  ajuste_sopro = ajuste_sopro/4 -128;
  ajuste_succao = ajuste_succao/4 - 128;
  //Serial.print("depois da escala:");
  //Serial.println(ajuste_sopro, ajuste_succao);

  sopra_forte = CONST_SOPRO_FORTE + ajuste_sopro;
  sopra_fraco_max = CONST_SOPRO_FRACO_MAX + ajuste_sopro;
  // não ajustar limite de sopra fraco
  //sopro_fraco_min = CONST_SOPRO_FRACO_MIN + ajuste_sopro;
  //Serial.print("sopro: ");
  //Serial.print(sopro_forte);
  //Serial.print(" , ");
  //Serial.print(sopro_fraco_max);
  //Serial.print(" , ");
  //Serial.println(sopro_fraco_min);

  succao_forte = CONST_SUCCAO_FORTE + ajuste_succao;
  // não ajustar limite de sucção fraca
  //succao_fraco_max = CONST_SUCCAO_FRACO_MAX + ajuste_succao;
  succao_fraco_min = CONST_SUCCAO_FRACO_MIN + ajuste_succao;
  //Serial.print("succao: ");

```



```

    //Serial.print(succao_forte);
    //Serial.print(" , ");
    //Serial.print(succao_fraco_max);
    //Serial.print(" , ");
    //Serial.println(succao_fraco_min);
}

int updateState(int input) {
    if (input < succao_forte)
        return Back;
    else if (input > sopro_forte)
        return Front;
    else if (input < succao_fraco_max && input > succao_fraco_min)
        return Left;
    else if (input < sopro_fraco_max && input > sopro_fraco_min)
        return Right;
    else
        return Dead;
}

void loop() {
    analogVal = analogRead(analogPin);    // lê o valor da pressão

    updateCalibration();

    actual_state = updateState(analogVal);

    Serial.print(analogVal);

    /***** Acinamento do Menu *****/
    // detecção do primeiro pico
    if (last_state == Dead &&
        analogVal <= CONST_SUCCAO_FRACO_MAX && enter_menu == 0)
    {
        enter_menu = 1;
        pico1 = 1;
        Serial.println("primeiro pico");
        actual_state = Dead;
    }

    // conta 5 varreduras com o usuário soprando para acionar o próximo estado
    if (enter_menu == 1 && analogVal <= CONST_SUCCAO_FRACO_MAX){
        pico1++;
        if (pico1 >= 2){
            enter_menu = 2;
            vale1 = 0;
            actual_state = Dead;
        }
    }
}

```

```

// detecção do primeiro vale
// conta 5 varreduras sem pressão no sensor para acionar o próximo estado
if (enter_menu == 2 && analogVal >= CONST_SUCCAO_FRACO_MAX){
    vale1++;
    if (vale1 >= 2){
        actual_state = Menu;
        Serial.println("primeiro vale");
        pico1 = 0;
        vale1 = 0;
        enter_menu = 0;
        menu_timeout = 0;
    }
}

```

```

// contagem de timeout para acionamento do menu

```

```

if (enter_menu != 0){
    menu_timeout++;
    if (menu_timeout >= 6) {
        Serial.println("timeout!");
        pico1 = 0;
        vale1 = 0;
        enter_menu = 0;
        menu_timeout = 0;
    }
}

```

```

/***** fim da detecção de acinamento do menu *****/

```

```

switch (actual_state) {
case Back:
    digitalWrite(backPin, HIGH);
    digitalWrite(frontPin, LOW);
    digitalWrite(leftPin, LOW);
    digitalWrite(rightPin, LOW);
    digitalWrite(menuPin, LOW);
    Serial.println("Back");
    break;

```

```

case Front:
    digitalWrite(backPin, LOW);
    digitalWrite(frontPin, HIGH);
    digitalWrite(leftPin, LOW);
    digitalWrite(rightPin, LOW);
    digitalWrite(menuPin, LOW);
    Serial.println("Front");

```

```

    break;

```

```

case Left:
    digitalWrite(backPin, LOW);

```

```

    digitalWrite(frontPin, LOW);
    digitalWrite(leftPin, HIGH);
    digitalWrite(rightPin, LOW);
    digitalWrite(menuPin, LOW);
    Serial.println("Left");

    break;

case Right:
    digitalWrite(backPin, LOW);
    digitalWrite(frontPin, LOW);
    digitalWrite(leftPin, LOW);
    digitalWrite(rightPin, HIGH);
    digitalWrite(menuPin, LOW);
    Serial.println("Right");

    break;

case Dead:
    digitalWrite(backPin, LOW);
    digitalWrite(frontPin, LOW);
    digitalWrite(leftPin, LOW);
    digitalWrite(rightPin, LOW);
    digitalWrite(menuPin, LOW);
    Serial.println("Dead");

    break;

case Menu:
    digitalWrite(backPin, LOW);
    digitalWrite(frontPin, LOW);
    digitalWrite(leftPin, LOW);
    digitalWrite(rightPin, LOW);
    digitalWrite(menuPin, HIGH);
    Serial.println("Menu");
    break;
}
delay(100);
last_state = actual_state;
}

```