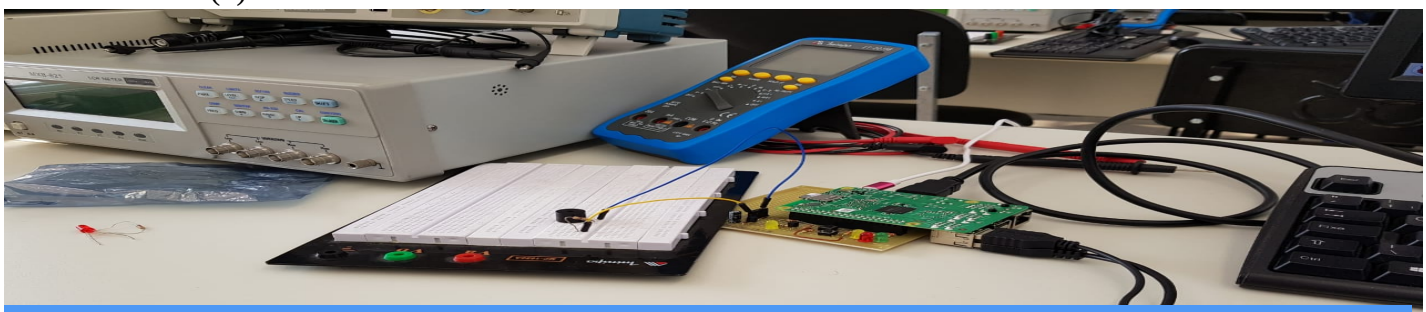
Relatório 5
Nome: Gustavo Canuto Wang RA: 11201722685

Para o relatório (1):

```
pi@ESZB026-17-5:~/sist_embarcados_git $ ls
lab02  lab03  lab04  lab05  README.md
pi@ESZB026-17-5:~/sist_embarcados_git $ cd lab05
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ ls
pwm_hw_c  pwm_hw_c.c  sons.py
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ gcc pwm_hw_c.c -o banana -lwiringPi
pwm_hw_c.c: In function 'main':
pwm_hw_c.c:28:10: warning: implicit declaration of function 'usleep' [-Wimplicit-function-declaration]
         usleep(10000);
         ^~~~~~
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ ls
banana  pwm_hw_c  pwm_hw_c.c  sons.py
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ ./banana
pinMode PWM: Unable to do this when using /dev/gpiomem. Try sudo?
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ sudo ./banana
Iniciando...
Fim.
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ sudo ./banana
Iniciando...
Fim.
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ sudo ./banana
Iniciando...
Fim.
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $
```

Para o relatório (2):



```
pi@ESZB026-
Arquivo  Editar  Abas  Ajuda
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ echo 18 > /sys/class/gpio/export
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ echo out > /sys/class/gpio/gpio18/direction
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ echo 1 > /sys/class/gpio/gpio18/value
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ echo 0 > sys/class/gpio/gpio18/value
bash: sys/class/gpio/gpio18/value: Arquivo ou diretório não encontrado
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ echo 0 > /sys/class/gpio/gpio18/value
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $ echo 18 > /sys/class/gpio/unexport
pi@ESZB026-17-5:~/sist_embarcados_git/lab05 $
```

Sim, ouvi um som. Possui um circuito oscilador e de amplificação, pois o sinal DC mesmo que seja constante, precisa dele para que a folha produza vibração em determinada frequência e amplitude de onda, fazendo assim com que o ser humano possa ser capaz de detectar o som.

Para o relatório (4):
Para a construção dos alarmes, dado que a frequência sonora dada corresponde a 261,6Hz, calculamos então as váriáveis range e divisor de acordo com a fórmula da frequência do PWM por hardware. Os valores obtidos foram 245 e 300, que multiplicados resultam em 261,22Hz, valor próximo do esperado. Para o de baixa prioridade, o tempos utilizado foi 200000ms. Para o de alta prioridade, foi utilizado 200000ms, 2*200000ms e 2000000ms, respectivamente.

```
// Ajustando o PWM por HARDWARE na Raspberry Pi

#include <stdio.h>
#include <wiringPi.h>

#define pino_PWM0 18                    // o PWM sera acionado na GPIO18

int main() {                            // este programa deve ser rodado com 'sudo'
  int dc, ciclos;
  wiringPiSetupGpio();                  // usa a numeracao da GPIO
  pinMode(pino_PWM0, PWM_OUTPUT);       // configura a GPIO18 com o PWM por hardware

  // Ajustando a frequencia do PWM em 10kHz com 245 passos de duty cycle
  // frequencia PWM = 19,2 MHz / (divisor * range)
  // 10000 = 19200000 / (divisor * 245) => divisor = 300
  pwmSetMode(PWM_MODE_MS);
  pwmSetRange(245);
  pwmSetClock(300);
  printf("Iniciando...\n");
  dc=122;
  int tempo = 200000;
  pwmWrite(pino_PWM0, dc);
  usleep(tempo);
  pwmWrite(pino_PWM0, 0);
  usleep(tempo);
  pwmWrite(pino_PWM0, dc);
  usleep(tempo);
  pwmWrite(pino_PWM0, 0);
  usleep(tempo);
  pwmWrite(pino_PWM0, dc);
  usleep(tempo);
  pwmWrite(pino_PWM0, 0);



  printf("Fim.\n");
  return 0;                             // a saida PWM permanece ligada apos o termino do programa
}
```

```
#include <stdio.h>
#include <wiringPi.h>

#define pino_PWM0 18                    // o PWM sera acionado na GPIO18

int main() {                            // este programa deve ser rodado com 'sudo'
  int dc, ciclos;
  wiringPiSetupGpio();                  // usa a numeracao da GPIO
  pinMode(pino_PWM0, PWM_OUTPUT);       // configura a GPIO18 com o PWM por hardware

  // Ajustando a frequencia do PWM em 10kHz com 128 passos de duty cycle
  // frequencia PWM = 19,2 MHz / (divisor * range)
  // 261 = 19200000 / (divisor * 245) => divisor = 300
  pwmSetMode(PWM_MODE_MS);              // usando frequencia fixa
  pwmSetRange(245);                     // passos do duty cycle (max=4096)
  pwmSetClock(300);                     // fornece uma frequencia de 10kHz (max=4096)
  printf("Iniciando...\n");
  dc=122;
  int tempo = 2000000;
  int duracao = 200000;
  pwmWrite(pino_PWM0, dc); // 1o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao);
  pwmWrite(pino_PWM0, dc); // 2o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao);
  pwmWrite(pino_PWM0, dc); // 3o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao*2);
  pwmWrite(pino_PWM0, dc); // 4o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao);
  pwmWrite(pino_PWM0, dc); // 5o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(tempo);          //espera 2seg
  pwmWrite(pino_PWM0, dc); // 1o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao);
  pwmWrite(pino_PWM0, dc); // 2o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao);
  pwmWrite(pino_PWM0, dc); // 3o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
```

```
  usleep(duracao*2);
  pwmWrite(pino_PWM0, dc); // 4o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);
  usleep(duracao);
  pwmWrite(pino_PWM0, dc); // 5o toque
  usleep(duracao);
  pwmWrite(pino_PWM0, 0);




  printf("Fim.\n");
  return 0;                    // a saida PWM permanece ligada apos o termino do programa
}
```

PROJETO FINAL

Para o projeto, será montada uma estrutura de quatro células de carga do tipo strain Gauge, cada uma suportando uma massa máxima de 50kg, que ficarão sobrepostas por uma superfície plana, servindo assim como uma balança. Cada célula de carga será de meia ponte de Wheatstone. Como serão quatro, então ficaremos com duas pontes completas, não precisando assim completar a ponte com nenhum outro resistor. Nesse tipo de célula de carga, a mudança da resistência elétrica do sensor é proprocional à força ou pressão aplicada sobre ele. Como as deformações nela causadas pela aplicação de forças e pressões são mínimas, provocando variações na ordem de décimos de ohms, será/serão usado(s) amplificador(es) de sinais de modelo HX711, que será/serão conectados ao sinal de tensão de saída das células. Esse amplificador opera em uma tensão de 4,8 a 5,5 V DC, corrente de operação de 1,6mA, temperatura de operação de -20 à 85°C, interface SPI e dimensões de 29x17x4mm (sem os pinos). O módulo é produzido pela Avia Semiconductors e seu conversor AD tem precisão de 24 bits. O módulo foi projetado para interfacear diretamente com sensores em ponte para aplicações de medição de carga. O multiplexador de entrada seleciona entre dois canais diferenciais A e B. Cada canal diferencial pode ser ligado em até duas células de carga(A+ e A- e B+ e B-, cada qual ligado na saída de uma célula de carga em meia ponte). Para o padrão de cores, o fio vermelho, que normalmente é usado para VCC, é na verdade o sinal de saída (ligado ao pino A- do HX711). E o fio no qual o VCC deve ser ligado é o fio branco. Será medidas as resistências entre os fios da célula para saber qual é qual. Os fios que derem uma leitura de 2K são os que devem ser conectados à E+ e E- (geralmente, o fio branco(E+) e preto(E-)). O fio em que forem obtidas leituras de 1K é o que deve ser conectado ao A+ (geralmente é o fio vermelho). Assim, o pino A+ do HX711 será ligado ao ponto comum entre os dois resistores de 1K. O programa avaliará o peso do paciente em relação à sua altura, de acordo com o cálculo do Índice de Massa Corporal (IMC) e indicará através dele se a pessoa está acima, abaixo ou no peso ideal. Para cada classificação de intervalo do IMC será usado um led de cor diferente, que permanecerá aceso enquanto o peso do paciente estiver naquele intervalo. Será utilizado um buzzler também, que acionará ao detectar variação de peso e só será desligado após a estalibilização dele. Segue a tabela:

Classificação	IMC	O que pode acontecer
Muito abaixo do peso	16 a 16,9 kg/m2	Queda de cabelo, infertilidade, ausência menstrual
Abaixo do peso	17 a 18,4 kg/m2	Fadiga, stress, ansiedade
Peso normal	**18,5 a 24,9 kg/m2**	Menor risco de doenças cardíacas e vasculares
Acima do peso	25 a 29,9 kg/m2	Fadiga, má circulação, varizes
Obesidade Grau I	30 a 34,9 kg/m2	Diabetes, angina, infarto, aterosclerose
Obesidade Grau II	35 a 40 kg/m2	Apneia do sono, falta de ar
Obesidade Grau III	maior que 40 kg/m2	Refluxo, dificuldade para se mover, escaras, diabetes, infarto, AVC

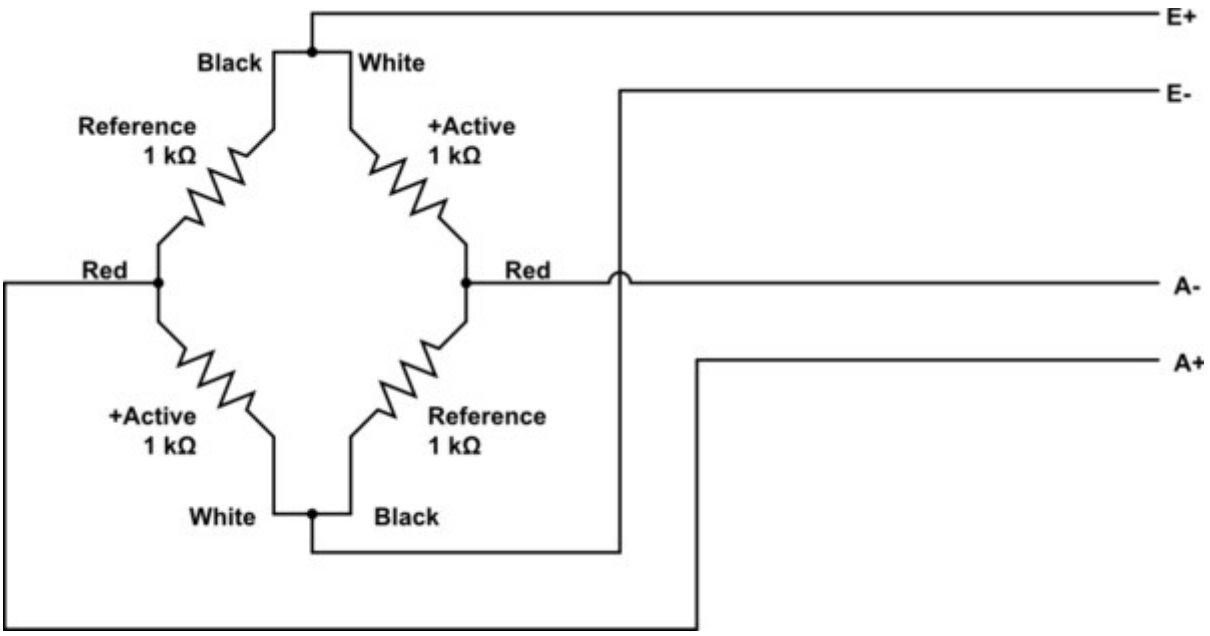
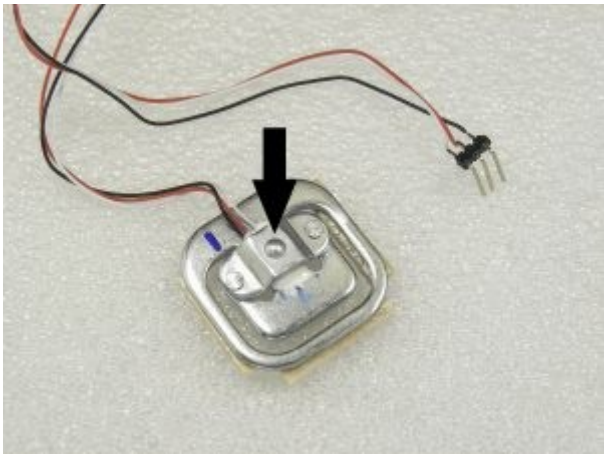
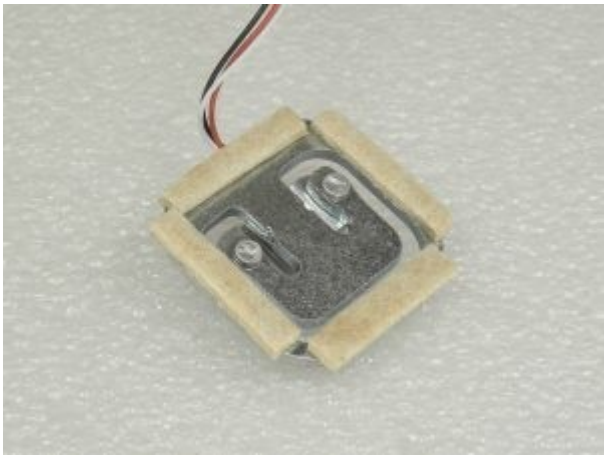


Illustration 1: Exemplo de circuito com células de carga



Pinos devem ser soldados nas extremidades dos fios da célula de carga



Parte inferior da célula de carga suspendida para os pinos não impedirem a flexão das células de carga

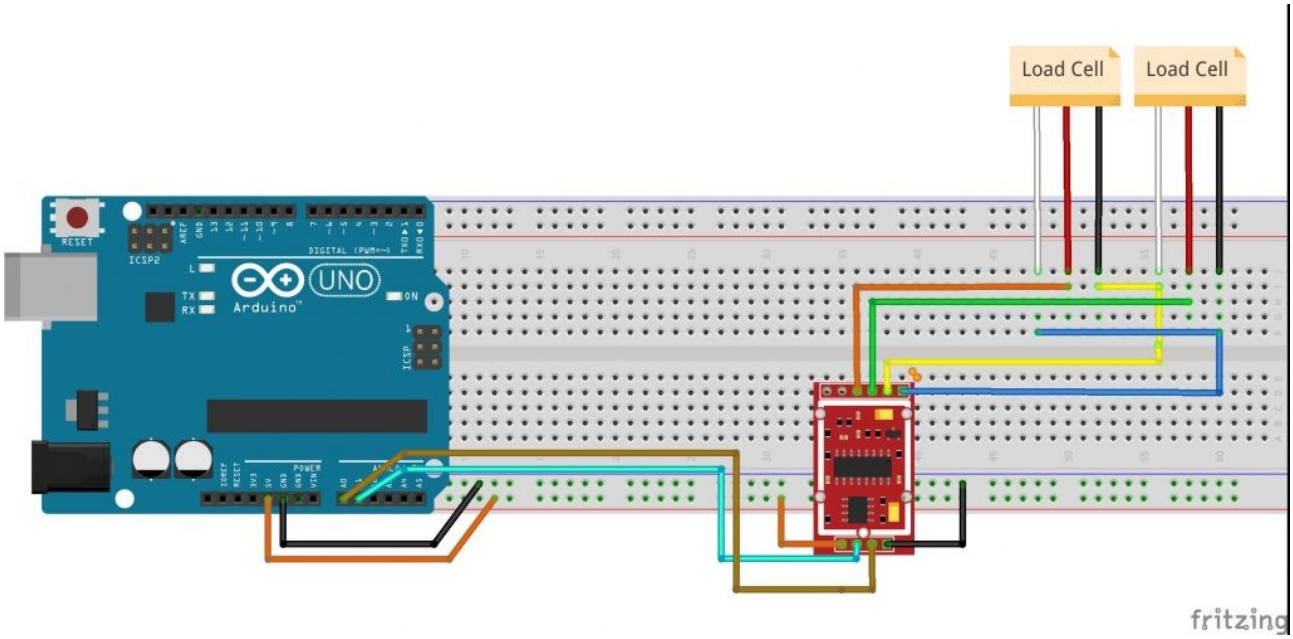


Illustration 2: Célula de carga com o Arduíno

Para os Sketches será usada a biblioteca HX711.h.

Primeiro passo é a calibração das células de carga. Usando a montagem com as duas células de 50 Kg, será rodado o programa de calibração que será colocado no GitHub. Definir um fator de calibração e inserir no programa. Após a calibração, anotar o valor aferido do Fator de Calibração para ser inserido nos programas de Balança com o HX711. **Passos para Calibração :**

1.Remova qualquer peso sobre as células de carga,

2.Após a balança for zerada pelo programa, coloque um peso conhecido sobre a célula de carga,

3.Pressione as teclas a,s,d,f para aumentar o Fator de Calibração por 10,100,1000,10000 respectivamente ou

4.Pressione as teclas z,x,c,v para diminuir Fator de Calibração por 10,100,1000,10000 respectivamente,

5.Pressione ENTER após digitar a letra,

6.Repita os passos 3 a 5, até o peso medido corresponder ao peso conhecido,

7.Remova o peso novamente, e zere a Balança (digite t + ENTER para zerar),

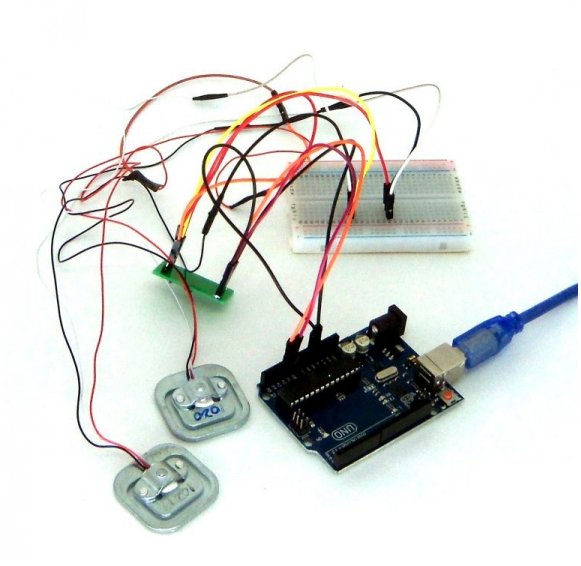
8.Coloque o peso novamente e repita os passos 3 até 6 para refazer a calibração.

Cada célula de carga poderá ter um valor diferente para o fator de calibração . Se algum peso medido estiver dando valor negativo, inverta os fios dos pinos A+ e A-.

Após a Calibração das células de Carga, poderá rodar o programa da Balança. Deve ser alterado o valor do Fator de calibração no Programa Arduino HX711 Balança (inserir o valor encontrado no procedimento de Calibração).

float calibration_factor = 42130; // fator de calibração aferido na Calibração

Remova qualquer peso sobre as células de carga, e pressione T para zerar a Balança.
Insira o peso na Balança e verifique o peso.



Circuito montado