PROJETO JUPITER

Instrumentação e Controle de Válvulas do Sistema de Alimentação de um Foguete Híbrido

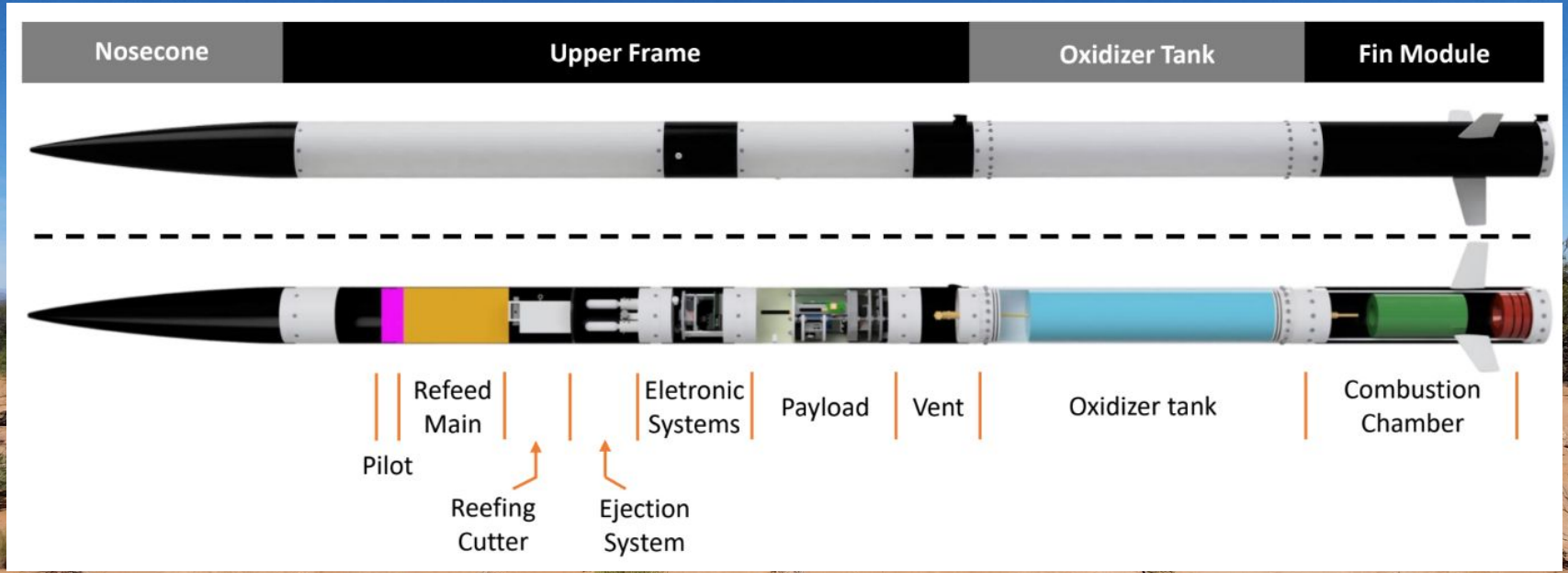
Caio Eduardo e Eduardo Seiji

Roteiro

- Contextualização
- Objetivos
- Visão geral
- Microcomputador
- Instrumentação
  - Termistor
  - Transdutor de Pressão
  - Microcontrolador
- Atuação
  - Válvulas
  - Desengate rápido
- Resultados

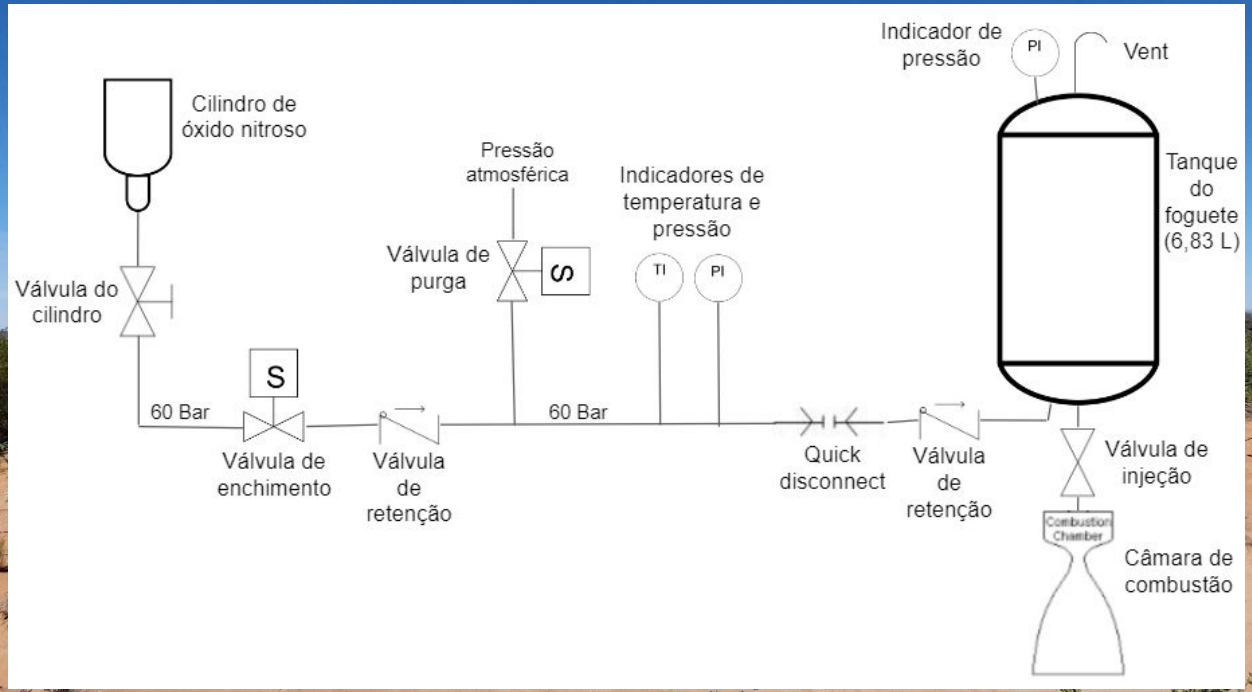
Contextualização

Esquema geral do foguete



Contextualização

P&ID (Pipe Instrumentation Design)

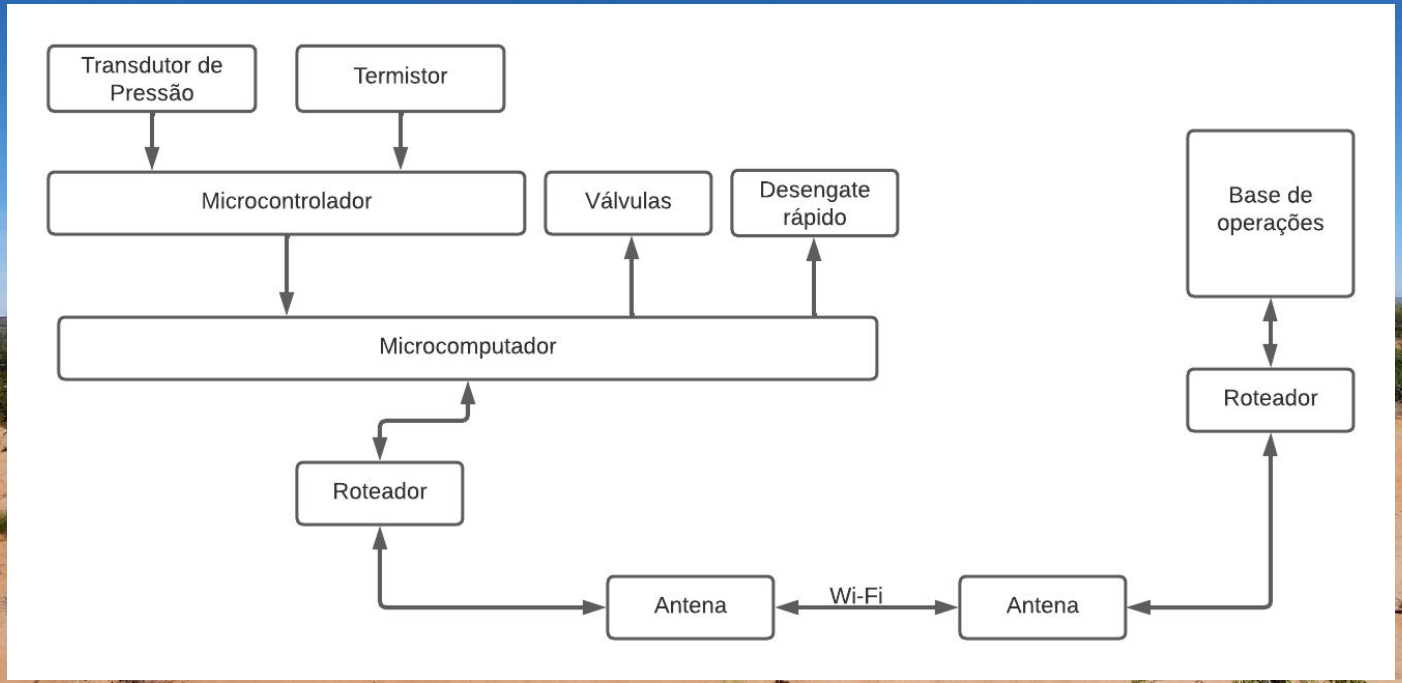


Objetivos

- Instrumentação do sistema de alimentação
  - Leitura de temperatura
  - Leitura de pressão
- Atuação
  - Atuação das válvulas de controle de abastecimento e purga
  - Atuação do sistema de desengate rápido
- Telemetria
  - Comunicação com a base de operações

Visão geral

Esquema simplificado do sistema (final)



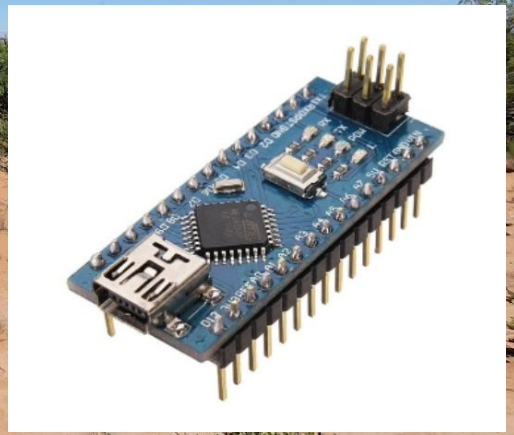
Microcomputador

Raspberry Pi

- Alto poder de processamento e capacidade de realizar múltiplas tarefas simultaneamente;
- Conexões: Ethernet, GPIO, USB

Instrumentação

- Termistor
- Transdutor de Pressão
- Microcontrolador (com conversor analógico-digital)



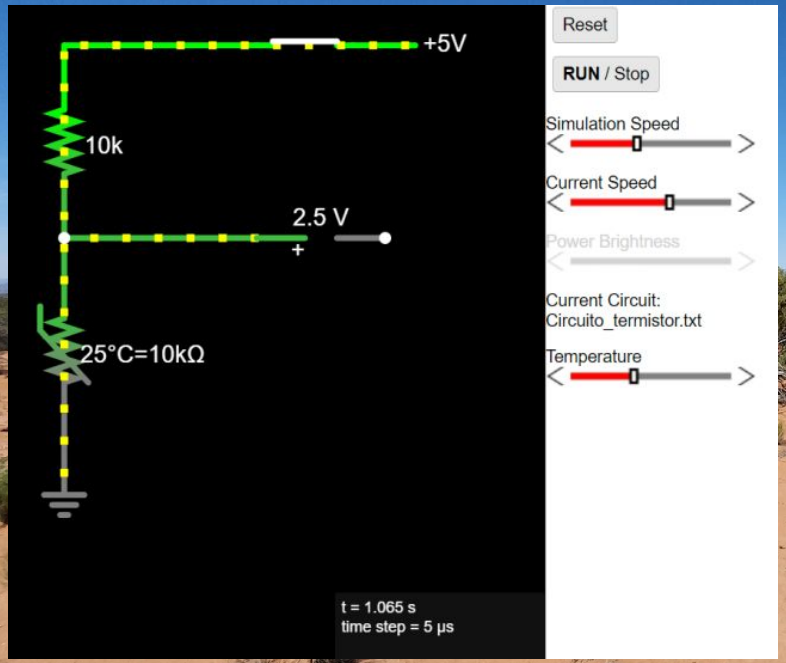
Termistor

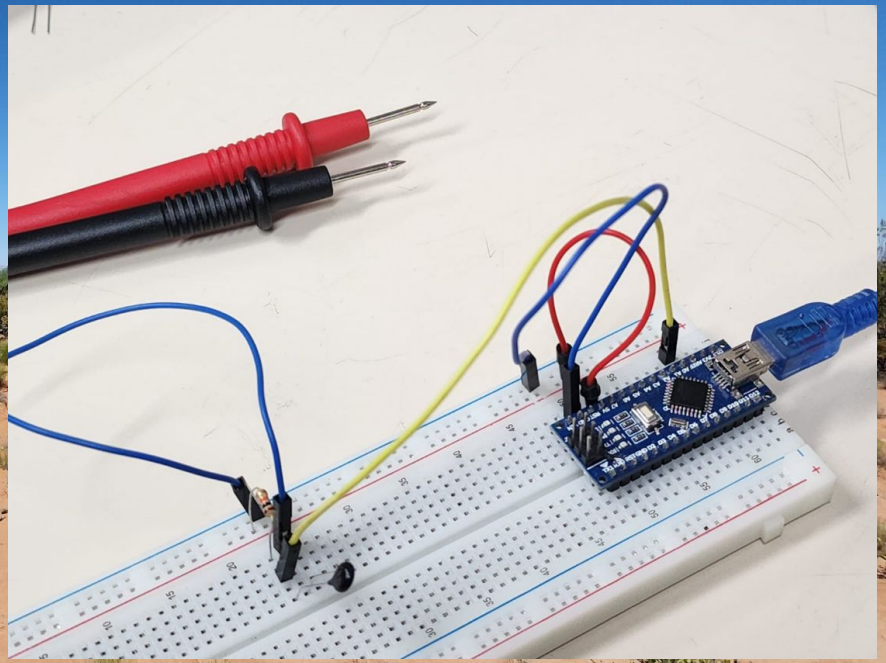
Cantherm NTC MF52 10K

- Faixa de medição: -55 a 125 °C
- Tolerância: 1%
- Conexão: medição será feita externamente à tubulação (aço, 0,89 mm de espessura), com pasta térmica

Termistor

Circuito de leitura





Termistor

Código de leitura

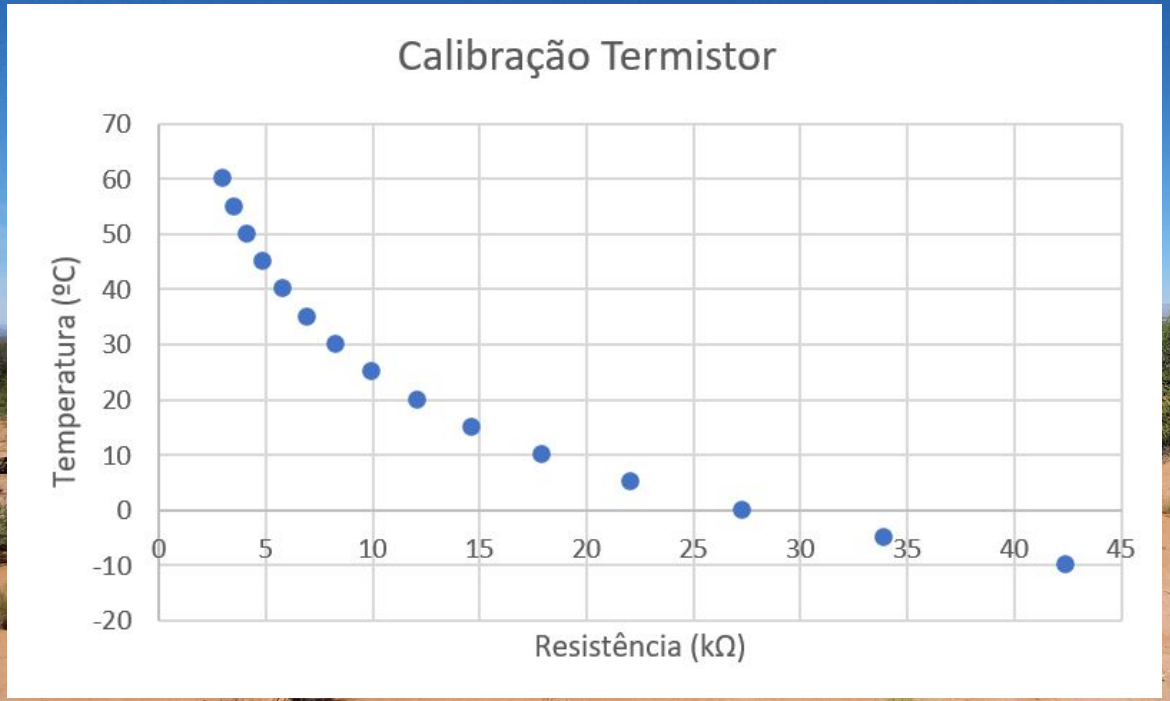
Leitura_termistor.ino

```
1
2   int sensorPin = A0;
3
4   float resistance = 10000;
5
6   void setup() {
7     // put your setup code here, to run once:
8     // initialize serial communication at 9600 bits per second:
9     Serial.begin(9600);
10  }
11
12  double linearInterpolate(double y, double data[][2])
13  {
14    double x0, x1, y0, y1;
15    for (int i = 0; i < sizeof(data) / (sizeof(data[0][0]) * 2); i++)
16    {
17      if (y > data[i][1] && y < data[i + 1][1])
18      {
19        y0 = data[i][1];  //lower bound
20        y1 = data[i + 1][1]; //upper bound
21        x0 = data[i][0];
22        x1 = data[i + 1][0];
23        return (x0 + ((x1 - x0) * ((y - y0) / (y1 - y0))));
24      }
25    }
26  }
27
28  double temperature_vs_resistance[][2] = {
29    {-30  111.3},
30    {-20  67.74},
31    {-15  53.39},
32    {-10  42.45},
33    {-5 33.89}
```

```
39  void loop() {
40    // put your main code here, to run repeatedly:
41
42    // read the input on analog pin 0:
43    int sensorValue = analogRead(sensorPin);
44
45    // remap the output from the range [0,1023] to [0,5]
46    // float voltage = map(sensorValue, 0, 1023, 0.0, 5.0); nope, only works for int values
47    float voltage = 5.0*sensorValue/1023.0;
48
49    //current
50    float termistor_resistance = 5.0*resistance/(5.0-voltage) - resistance;
51
52    // print out the value:
53    Serial.println(termistor_resistance/1000.0);
54
55    // transform resistance into temperature, using the calibration table
56    float temperature = linearInterpolate(resistance, temperature_vs_resistance;
57
58    // print out the value:
59    Serial.println(temperature);
60
61    // delay in between reads for stability
62    delay(1000);
63
64  }
```

Termistor

Calibração



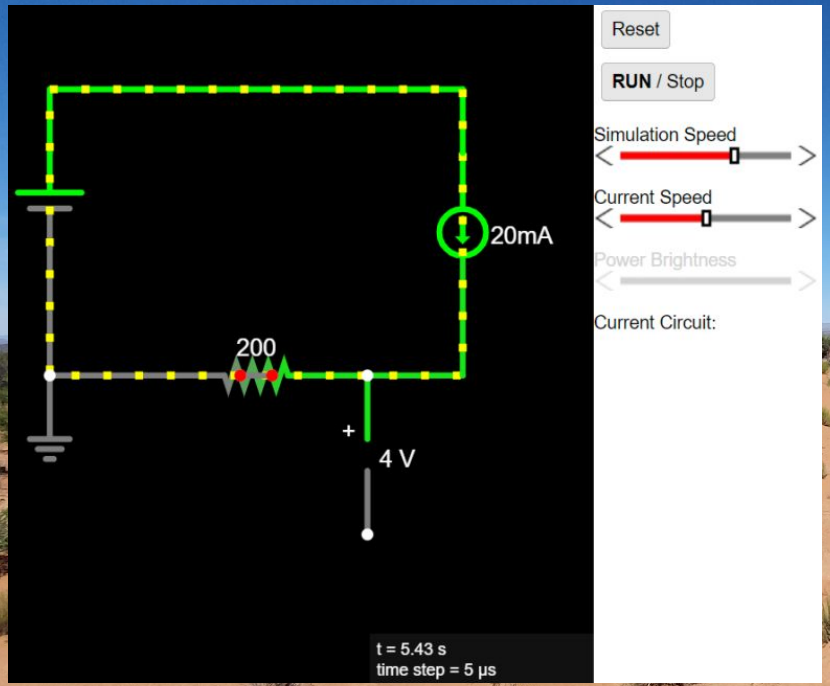
Transdutor de Pressão

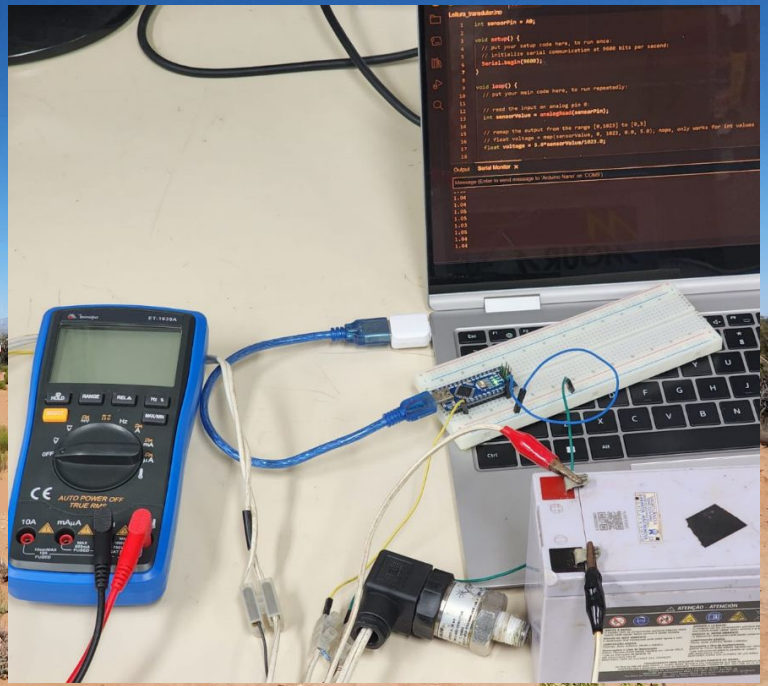
Velki VKP-011

- Faixa de medição: 0 a 200 bar
- Sensibilidade: 0,1 bar
- Material (diafragma e invólucro): aço inoxidável AISI-316L
- Corrente de saída: 4 a 20 mA
- Conexão: ¼” NPT

Transdutor de Pressão

Circuito de leitura





Transdutor de Pressão

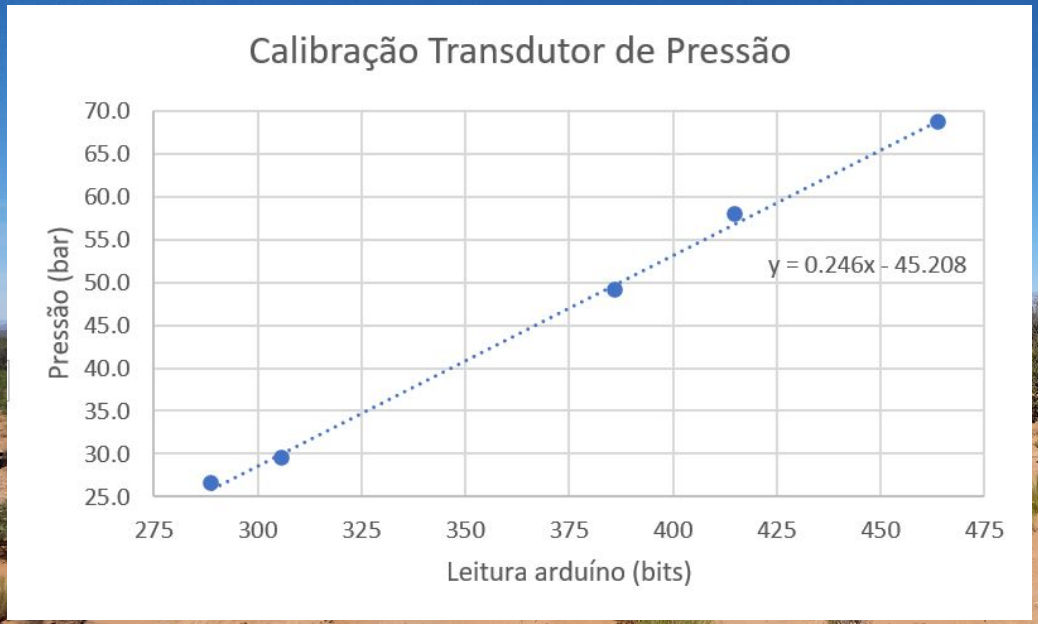
Código de leitura

Leitura_transdutor.ino

```
1  int sensorPin = A0;
2
3  void setup() {
4    // put your setup code here, to run once:
5    // initialize serial communication at 9600 bits per second:
6    Serial.begin(9600);
7  }
8
9  void loop() {
10   // put your main code here, to run repeatedly:
11
12   // read the input on analog pin 0:
13   int sensorValue = analogRead(sensorPin);
14
15   // transform analog read into temperature, using the cailbration table
16   // and function, got from comparing with a calibrated pump
17   // data: [p(kgf/cm2), bits; 27, 289; 30, 306; 50, 386; 59, 415; 70, 464]
18   float pressure = 0.246*sensorValue - 45.208;
19
20   // print out the pressure:
21   Serial.println(pressure);
22
23   // delay in between reads for stability
24   delay(1000);
25 }
```

Transdutor de Pressão

Calibração



Microcontrolador

Arduíno Nano

- Conversor analógico-digital;
- Comunicação serial com o RaspberryPi;
- Resolução 10 bits;
- Tensão de operação: 2 a 5,5 V.
- Tensão de entrada analógica: -0,3 a VDD + 0,3 V.

Atuação

- Visão geral do subsistema;
- Válvulas;
- Circuito de potência;
- Análise de resultados para o circuito de potência;
- Adaptação do sistema para acionamento do desengate rápido.

Visão geral do subsistema

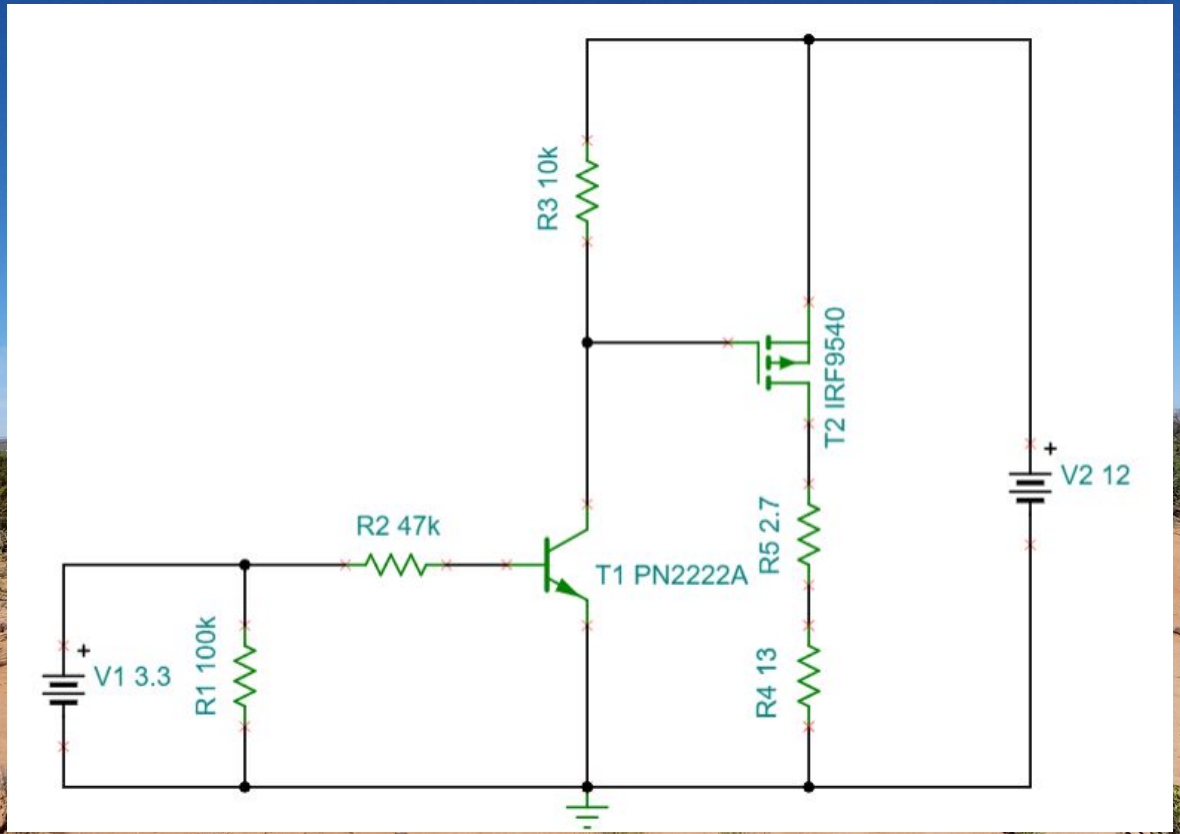
- Circuito de potência, pois as GPIOs do Raspberry Pi não fornecem energia suficiente para acionar as válvulas;
- Utilização de uma associação entre transistores do tipo BJT e MOSFET para chaveamento do circuito;
- Bateria de especificações 12 V - 7 Ah para a alimentação.

Válvulas

Válvula Mal’s Nitro Solenóide de Nitro

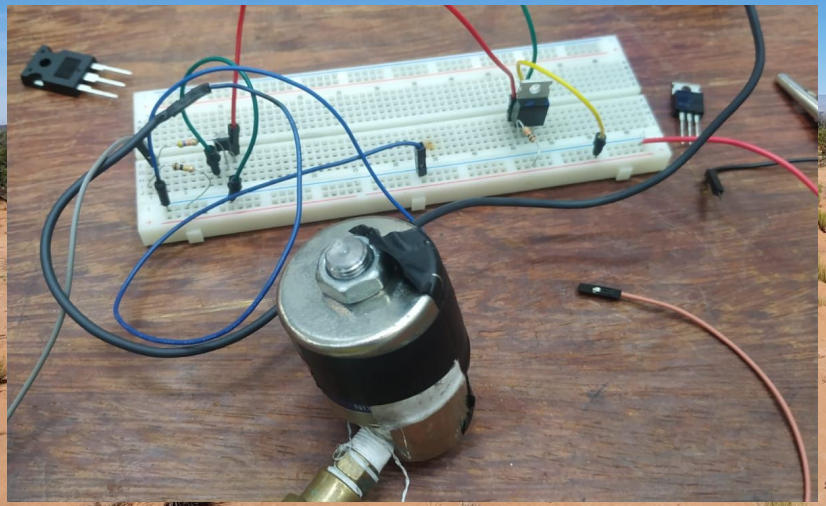
- Tipo: agulha
- Atuador: solenóide
- Tensão nominal: 12 V
- Corrente máxima: 6 A
- Pressão de trabalho: 82,74 bar
- Material: latão niquelado
- Conexões: ⅛” NPT

Circuito de Potência



Análise do Circuito de Potência

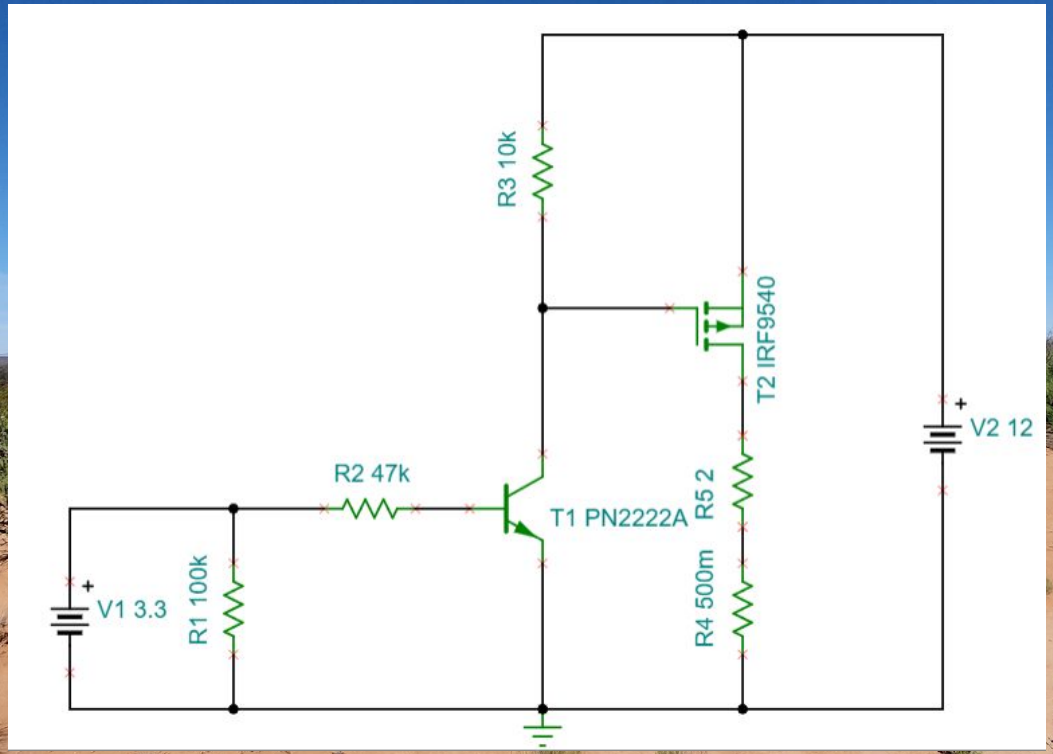
- Corrente na válvula: ~700mA;
- Potência dissipada na ativação: ~9W;
- Autonomia da bateria: ~6h



Desengate rápido

- Rompimento de um fio polimérico por aquecimento de um fio de Ni-Cr por efeito Joule;
- Temperatura de rompimento do fio polimérico: 144°C;
- Resistência do fio de Ni-Cr: 0,5Ω.

Adaptação do circuito para o desengate rápido



Análise do Circuito de Potência

- Corrente no fio de Ni-Cr: ~4,5A;
- Potência dissipada no fio: ~10W;
- Temperatura na qual o fio chega: ~160°C

Resultados

Tarefas concluídas

- Circuitos e códigos de leitura dos sensores de temperatura e pressão;
- Calibração dos sensores;
- Circuito de potência para ativação de válvulas;
- Códigos para a utilização das GPIOs do RaspberryPi para ativação das válvulas.

Próximas etapas

- Fazer a comunicação serial entre o Arduíno e o RaspberryPi;
- Fazer a comunicação por telemetria com o usuário.