

UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

FACULDADE DE ENGENHARIA DA COMPUTAÇÃO E DE TELECOMUNICAÇÕES

DANTON ARAUJO RODRIGUES DA CUNHA (202206840011)

JEAN DEAN MONTEIRO PEREIRA (20200684002)

JOEL TAVARES MIRANDA (202206840054)

KAUAN MIRANDA TAVARES (202206840033)

MARCO ANTONIO DO ESPIRITO SANTO MAUES JUNIOR (202206840038)

**ÁBACO ELETRÔNICO COM SINTETIZADOR DE VOZ**

BELÉM

2023

1. **INTRODUÇÃO**

Este trabalho faz parte da avaliação da disciplina de Circuitos Elétricos, do curso de Engenharia da Computação, no Instituto de Tecnologia, sediado na Universidade Federal do Pará.

A proposição do trabalho é desenvolver um ábaco, do tipo ábaco aberto, com um circuito capaz de indicar a posição das “bolinhas” quando estas forem movimentadas, permitindo assim, realizar o processo de contagem decimal e a realização de operações matemáticas básicas. Em seguida, o circuito deverá processar os valores inseridos no ábaco a fim de reproduzir verbalmente o valor correspondente ao resultado numérico.

1. **METODOLOGIA**

## Compreensão do problema

## Primeiramente, foi feito um estudo sobre a proposição do trabalho, começando pelo ábaco e o seu funcionamento. O ábaco…

## [Insira aqui imagens de ábaco]

## Neste caso, utilizaremos o ábaco aberto, que utiliza pinos verticais para a inserção de argolas, onde cada pino representa uma posição do Sistema Numérico Decimal (base 10) respectivamente, da esquerda para a direita: Unidade de Milhar, Centena, Dezena e Unidade.

## Cada argola representa uma unidade no pino referente a posição inserida, onde cada pino pode receber até 10 argolas, conforme a base 10 do sistema indo-arábico. Por exemplo, os números 1234 e 47 podem ser representados respectivamente:

## Quando um pino tiver 10 argolas, elas poderão ser substituídas por uma argola a ser inserida no pino de ordem superior, à esquerda, que equivalerá a uma unidade da ordem subsequente. As operações matemáticas feitas nesse ábaco se restrigem a adição e subtração, diferentemente de outros tipos de ábaco.

## 

## Desenvolvimento do ábaco

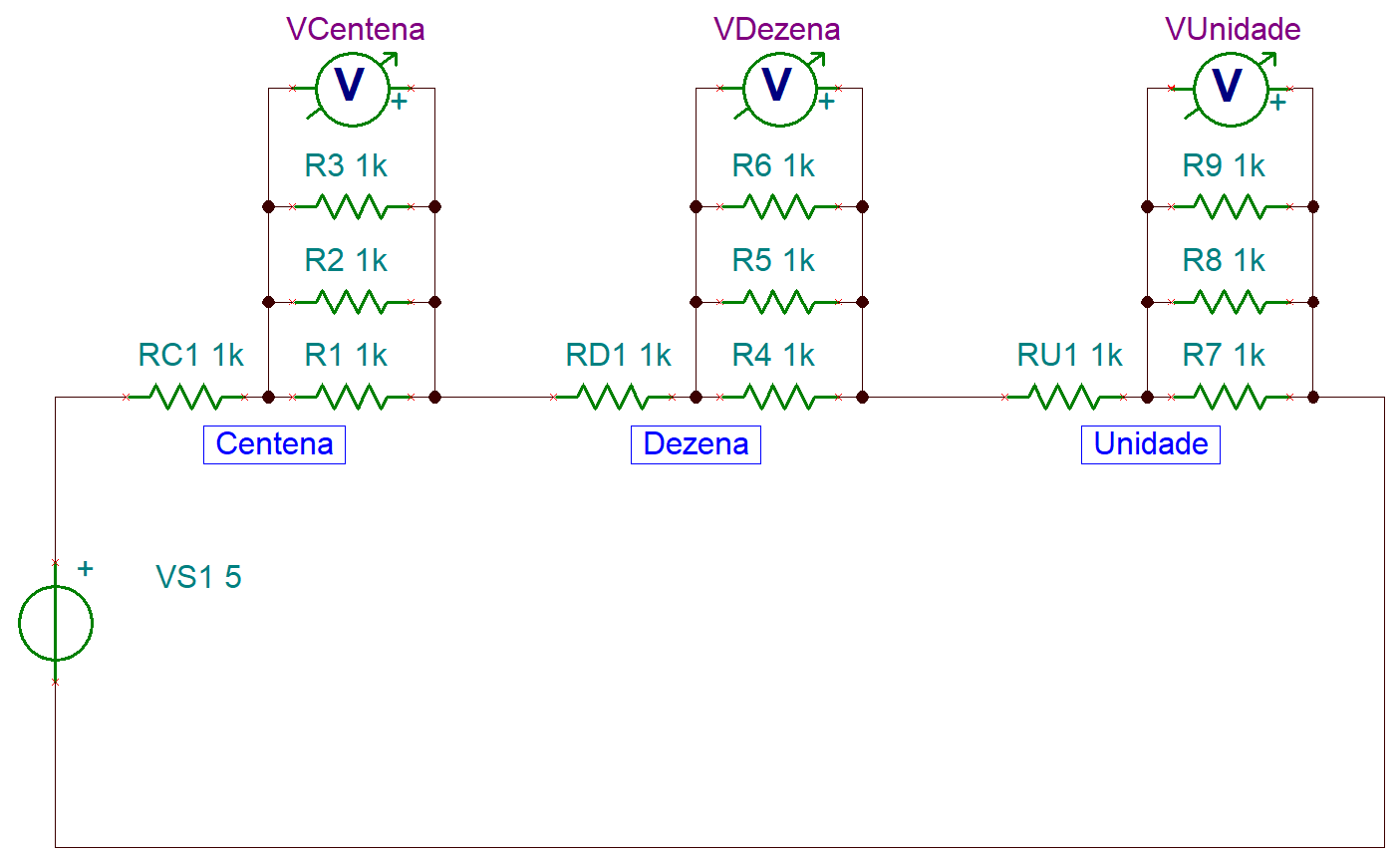
## O desenvolvimento do nosso ábaco partiu de duas problemáticas: a contagem de elementos em cada pino do ábaco e a sintetização de voz para a verbalização dos números e resultados de operações matemáticas. As possíveis opções de componentes foram ponderadas em vantagens e desvantagens, além de terem as suas funcionalidades simuladas e testadas. A partir disso, a escolha do componente responsável por processar e reproduzir esses valores foi feita. Além disso, também será feita uma placa PCB para eliminar a necessidade de protoboards.

## Contagem de elementos no ábaco

## Contagem de elementos no ábaco

## A contagem de argolas no pino de cada posição será feita a partir do cálculo da condutância em cada pino do circuito, utilizando um resistor de valor conhecido em série com um conjunto de resistores de valores desconhecido, com essa configuração desconhecida sendo uma associação de resistores em paralelo, que funcionariam como as argolas do ábaco, para assim, ser possível contabilizar o número de resistores.

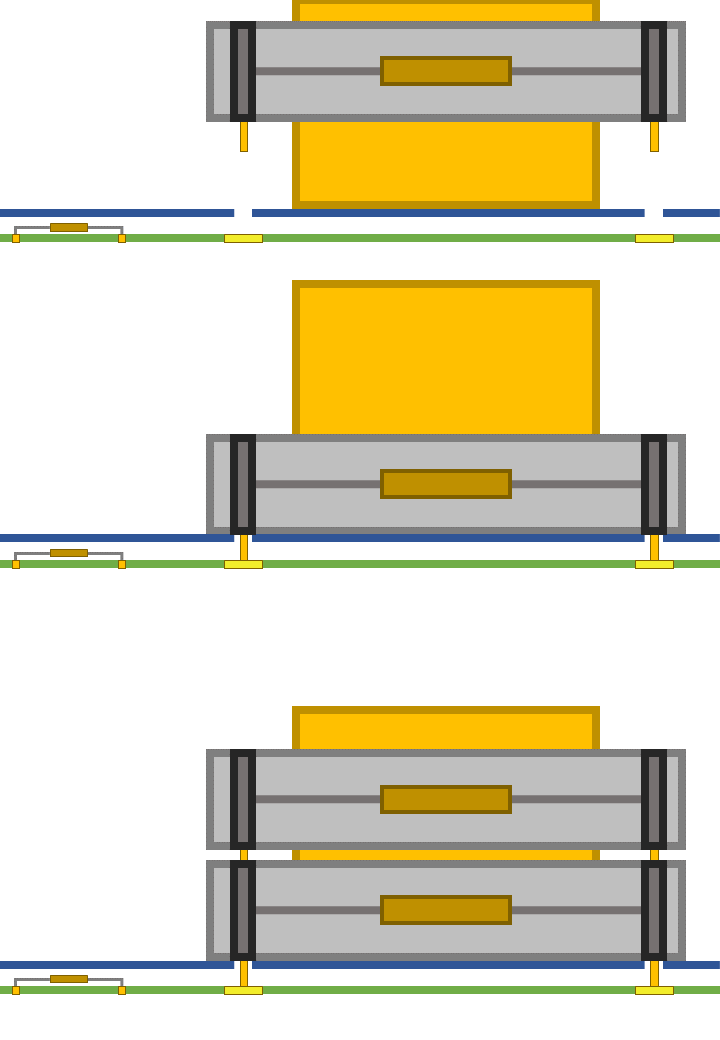
Imagem 1 - Esquema de circuito.



Fonte: TINA-TI.

## A implementação a nível de estrutura não fugiria da ideia proposta para o circuito, com a contagem de argolas no pino de cada posição será feita a partir do “empilhamento” de resistores, que ao se encaixarem, formariam uma associação de resistores em paralelo. Todas as argolas teriam resistores, de mesmo valor, encapsulados na sua construção interna e conectores, que permitiriam encaixar no circuito ou em outras argolas.

Imagem 2 - Representação do funcionamento do circuito (vista frontal).

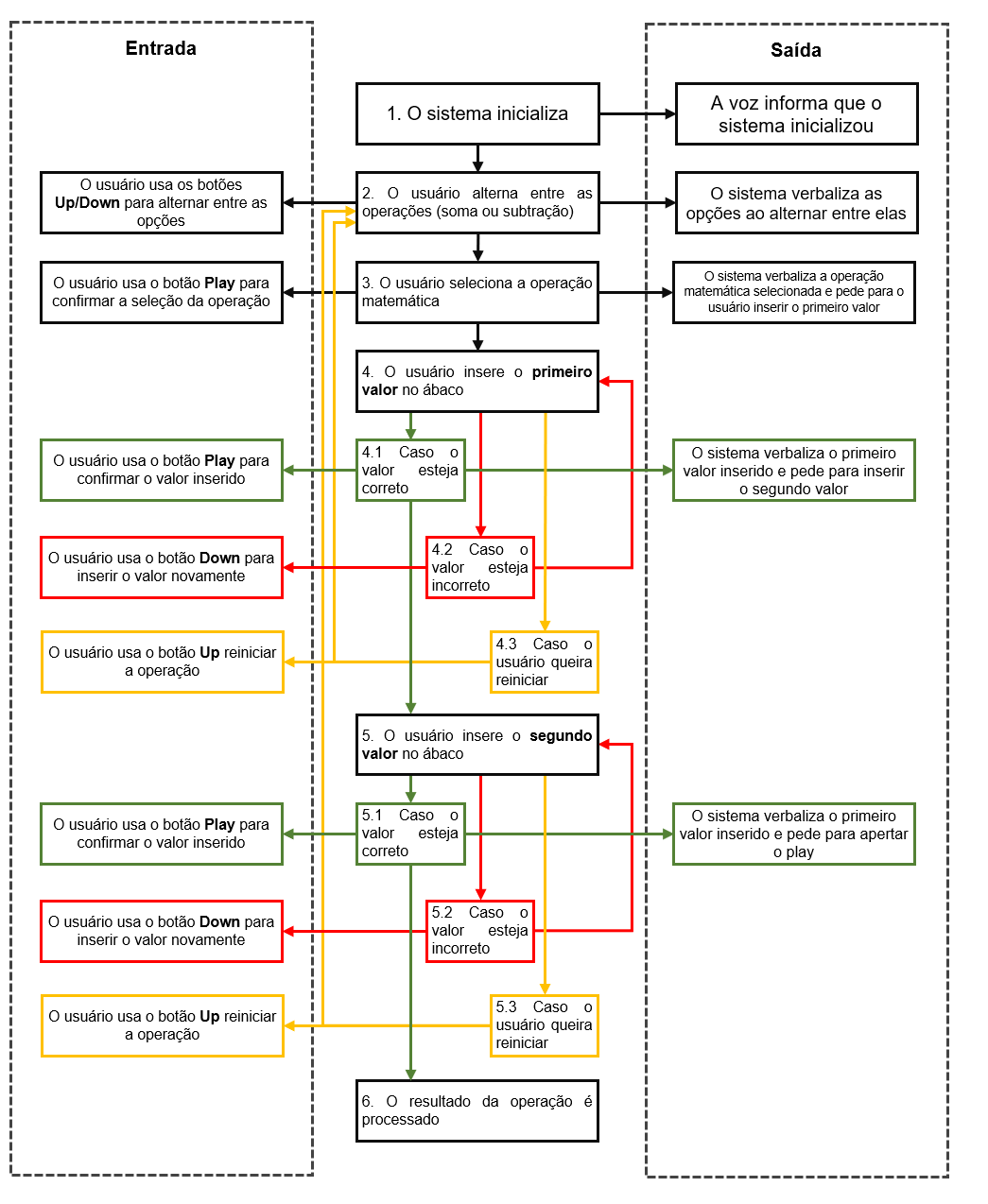


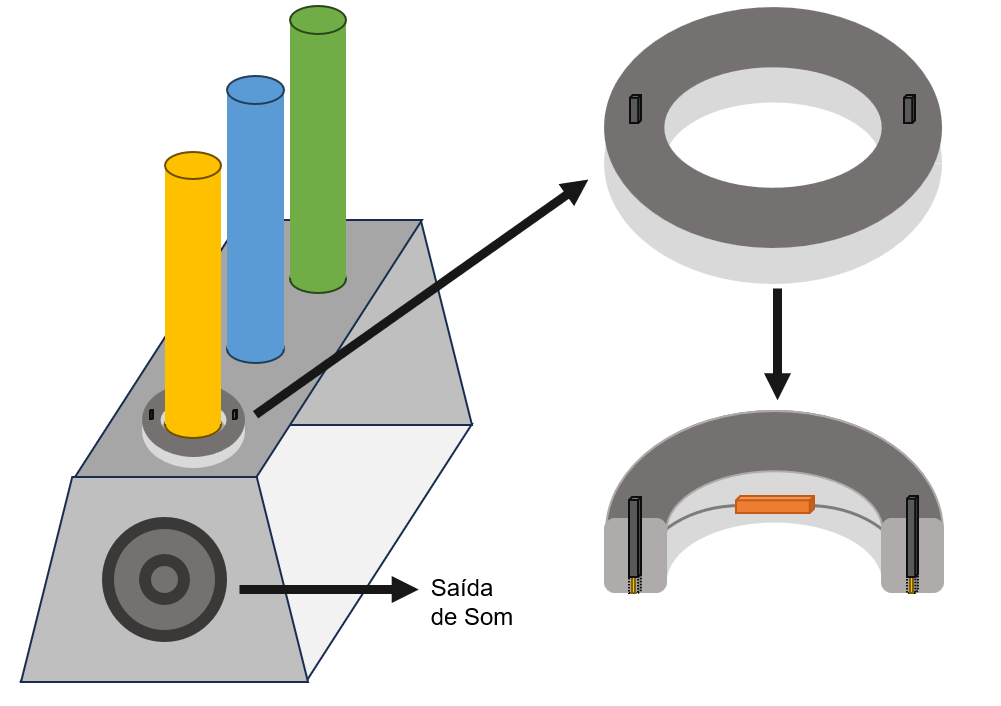
Fonte: Autor.

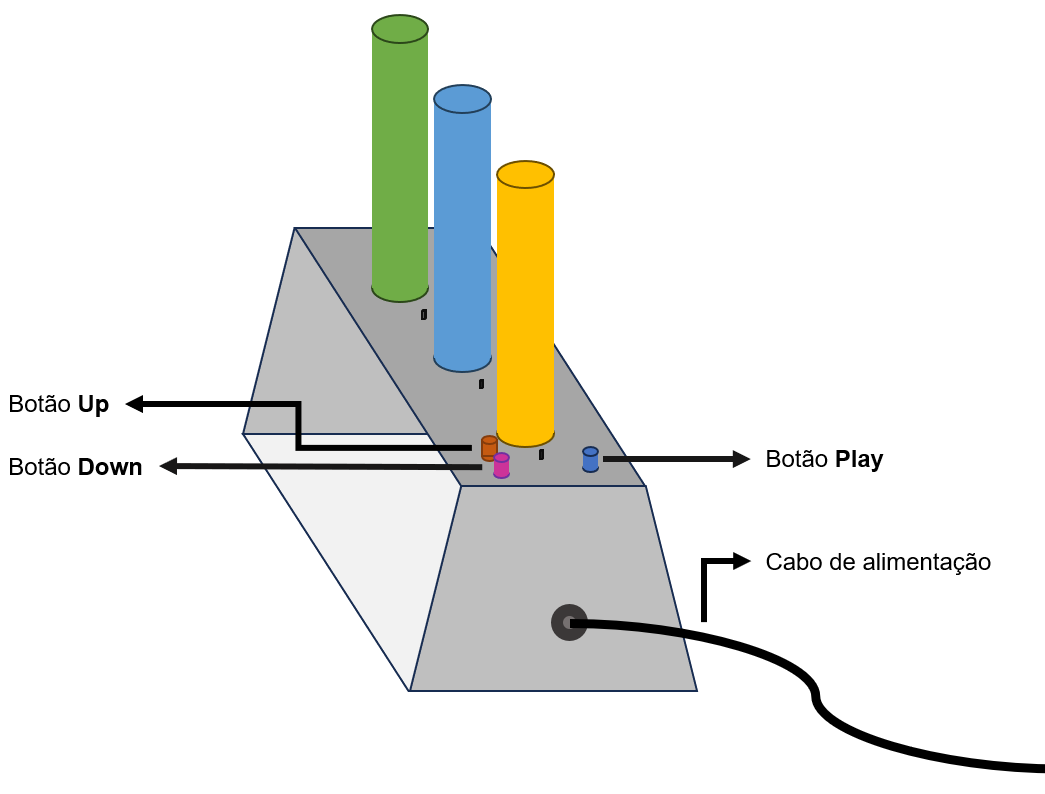
## Verbalização dos números e resultados

## O principal elemento que exigiria um cuidado, seria a sintetização de voz para a verbalização dos números e resultados de operações matemáticas. Com

## Foi feito um fluxograma para facilitar a compreensão e a elaboração do circuito, a partir das interações do usuário, bem como a entrada e a saída do circuito.



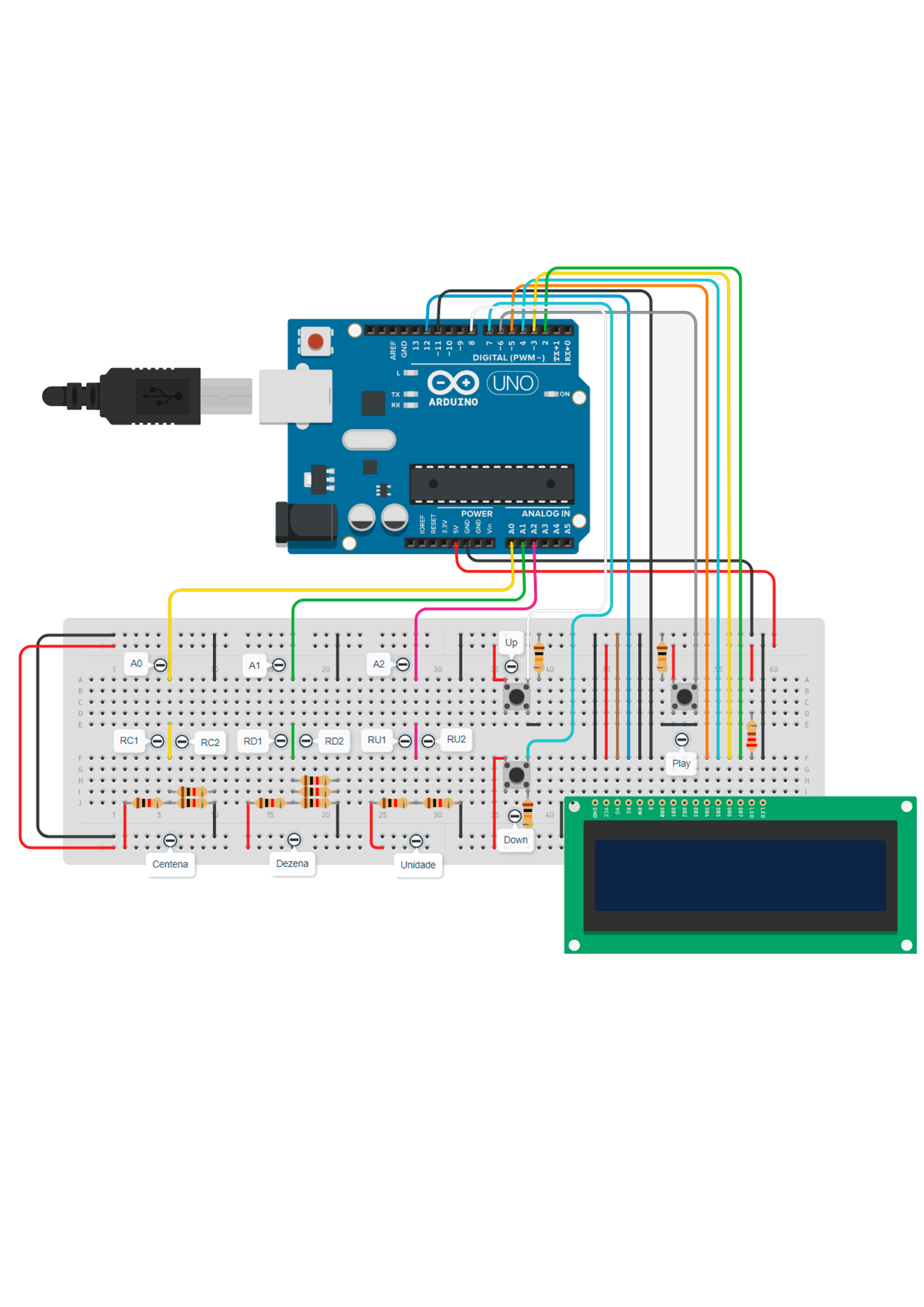




Funcionamento do acoplamento das argolas

## Simulação no TinkerCAD

## Primeiramente, foi feita uma simulação do circuito utilizando um display LCD 16x2, a fim de facilitar a visualização dos valores inseridos e verificar a viabilidade do circuito.



## Funcionamento:

1. Quando o Arduino liga, é exibida uma mensagem "Sistema Ok".

2. O Arduino exibe uma mensagem na tela, onde aparece na primeira linha a mensagem "Operacao:" e na segunda linha, a seguinte mensagem "1.(+) 2.(-)".

3. Usando dois botões, chamados Up e Down, a opção poderia ser alternada, por exemplo, ao apertar o botão Down, a opção selecionada indicaria "1.(+) 2.(-)\*". Ao aperta a opção Up, o display mostraria "1.(+)\* 2.(-)".

4. Ao apertar um terceiro botão Play, seria selecionada a opção desejada, onde "1.(+)" é a operação de soma e "2.(-)" é a operação de subtração.

5. Depois, o Arduino exibiria a mensagem "Insira o primeiro valor". E o Arduino aguardaria o botão Play ser pressionado para que, em seguida, o arduino exibisse na segunda linha a mensagem "Valor1: {valorLido}".

6. Depois do primeiro valor, o Arduino exibiria a mensagem "Insira o segundo valor". E o arduino aguardaria o botão Play ser pressionado para que, em seguida, o arduino exibisse na segunda linha a mensagem "Valor2: {valorLido}". Os valores dos resistores associados são lidos pelos pinos A0, A1 e A2 e unificados como um único valor inteiro.

7. Por fim, o Arduino faria a operação matemática selecionada (soma ou subtração), exibindo no display, na primeira linha a mensagem "Resultado:" e na segunda linha, o resultado da operação matemática.

#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // Configuração dos pinos do LCD

const int numPins = 3; // Número de pinos analógicos

int analogPins[numPins] = {0, 1, 2}; // Define os pinos analógicos a serem lidos

char pinLabels[numPins][3] = {"C:", "D:", "U:"}; // Etiquetas para os pinos analógicos

int raw[numPins];

int Vin = 5;

float Vout[numPins];

float R1 = 1000;

float R2[numPins];

float buffer[numPins];

int valorLido = 0; // Variável para armazenar o valor lido de C, D e U

const int botaoUpPin = 8; // Pino do botão Up

const int botaoDownPin = 7; // Pino do botão Down

const int botaoPlayPin = 6; // Pino do botão Play

int estadoBotaoUpAnterior = HIGH; // Estado anterior do botão Up

int estadoBotaoDownAnterior = HIGH; // Estado anterior do botão Down

int estadoBotaoPlayAnterior = HIGH; // Estado anterior do botão Play

int opcaoSelecionada = 1; // Inicialmente, a opção 1 é selecionada (soma)

bool operacaoSelecionada = false; // Indica se a operação foi selecionada

bool primeiroValorInserido = false; // Indica se o primeiro valor foi inserido

void setup() {

lcd.begin(16, 2); // Inicializa o display LCD com 16 colunas e 2 linhas

Serial.begin(9600);

pinMode(botaoUpPin, INPUT);

pinMode(botaoDownPin, INPUT);

pinMode(botaoPlayPin, INPUT);

// Mensagem de inicialização

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Sistema Ok");

Serial.println("Sistema Ok");

delay(2000);

// Exibe a opção selecionada

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Operacao:");

Serial.println("Operacao:");

lcd.setCursor(0, 1);

Serial.println("1.(+) 2.(-)");

lcd.print("1.(+) 2.(-)");

}

void loop() {

estadoBotaoUpAnterior = digitalRead(botaoUpPin);

estadoBotaoDownAnterior = digitalRead(botaoDownPin);

estadoBotaoPlayAnterior = digitalRead(botaoPlayPin);

// Seleciona a opção com os botões Up e Down

if (!operacaoSelecionada) {

if (digitalRead(botaoUpPin) == HIGH) {

lcd.setCursor(0, 1);

lcd.print("1.(+)\* 2.(-) ");

Serial.println("1.(+)\* 2.(-) ");

opcaoSelecionada = 1;

} else if (digitalRead(botaoDownPin) == HIGH) {

lcd.setCursor(0, 1);

Serial.println("1.(+) 2.(-)\*");

opcaoSelecionada = 2;

}

}

// Aguarda o botão Play para prosseguir

if (!operacaoSelecionada && digitalRead(botaoPlayPin) == HIGH && estadoBotaoPlayAnterior == HIGH) {

operacaoSelecionada = true;

lcd.clear();

lcd.setCursor(0, 0);

if (opcaoSelecionada == 1) {

lcd.print("Operacao: +");

Serial.println("Operacao: +");

} else {

lcd.print("Operacao: -");

Serial.println("Operacao: -");

}

delay(2000);

}

if (operacaoSelecionada) {

if (!primeiroValorInserido) {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Insira o valor 1");

Serial.println("Insira o valor 1");

while (digitalRead(botaoPlayPin) == LOW) {

// Aguarda o botão Play ser pressionado

}

valorLido = lerValor();

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Valor1: ");

Serial.println("Valor1: ");

lcd.print(valorLido);

Serial.println(valorLido);

lcd.setCursor(0, 1);

lcd.print("Aguarde...");

Serial.println("Aguarde...");

delay(4000); // Aguarda 2 segundos

primeiroValorInserido = true;

} else {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Insira o valor 2");

Serial.println("Insira o valor 2");

while (digitalRead(botaoPlayPin) == LOW) {

// Aguarda o botão Play ser pressionado

}

int segundoValor = lerValor();

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Valor2: ");

Serial.println("Valor2: ");

lcd.print(segundoValor);

Serial.println(segundoValor);

lcd.setCursor(0, 1);

lcd.print("Aguarde...");

Serial.println("Aguarde...");

delay(4000); // Aguarda 2 segundos

valorLido += (opcaoSelecionada == 1) ? segundoValor : -segundoValor;

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Resultado:");

Serial.println("Resultado:");

lcd.setCursor(0, 1);

lcd.print(valorLido);

Serial.println(valorLido);

delay(5000);

operacaoSelecionada = false; // Reinicia o processo

primeiroValorInserido = false;

}

}

delay(100);

}

int lerValor() {

int valor = 0;

for (int i = 0; i < numPins; i++) {

raw[i] = analogRead(analogPins[i]);

if (raw[i]) {

buffer[i] = raw[i] \* Vin;

Vout[i] = (buffer[i]) / 1024.0;

buffer[i] = (Vin / Vout[i]) - 1;

R2[i] = R1 \* buffer[i];

valor = valor \* 10 + (int)(round(R2[i] / 1000));

lcd.setCursor(i \* 4, 1);

lcd.print(" "); // Limpa o espaço anterior

Serial.println(" "); // Limpa o espaço anterior

lcd.setCursor(i \* 4, 1);

lcd.print((int)(round(R2[i] / 1000)));

Serial.println((int)(round(R2[i] / 1000)));

}

}

return valor;

}

Versão 1.2

const int numPins = 3; // Número de pinos analógicos

int analogPins[numPins] = {0, 1, 2}; // Define os pinos analógicos a serem lidos

char pinLabels[numPins][3] = {"C:", "D:", "U:"}; // Etiquetas para os pinos analógicos

int raw[numPins];

int Vin = 5;

float Vout[numPins];

float R1 = 1000;

float R2[numPins];

float buffer[numPins];

int valorLido = 0; // Variável para armazenar o valor lido de C, D e U

const int botaoUpPin = 8; // Pino do botão Up

const int botaoDownPin = 7; // Pino do botão Down

const int botaoPlayPin = 6; // Pino do botão Play

int estadoBotaoUpAnterior = HIGH; // Estado anterior do botão Up

int estadoBotaoDownAnterior = HIGH; // Estado anterior do botão Down

int estadoBotaoPlayAnterior = HIGH; // Estado anterior do botão Play

int opcaoSelecionada = 1; // Inicialmente, a opção 1 é selecionada (soma)

long primeiroValor;

long segundoValor;

long resultado;

bool operacaoSelecionada = false; // Indica se a operação foi selecionada

bool primeiroValorInserido = false; // Indica se o primeiro valor foi inserido

int lerValor() {

int valor = 0;

for (int i = 0; i < numPins; i++) {

raw[i] = analogRead(analogPins[i]);

if (raw[i]) {

buffer[i] = raw[i] \* Vin;

Vout[i] = (buffer[i]) / 1024.0;

buffer[i] = (Vin / Vout[i]) - 1;

R2[i] = R1 \* buffer[i];

valor = valor \* 10 + (int)(round(R2[i] / 1000));

// Serial.println(" "); // Limpa o espaço anterior

//Serial.print((int)(round(R2[i] / 1000)));

}

}

return valor;

}

void setup() {

Serial.begin(9600);

pinMode(botaoUpPin, INPUT);

pinMode(botaoDownPin, INPUT);

pinMode(botaoPlayPin, INPUT);

// Mensagem de inicialização

Serial.println("Sistema Ok");

delay(1000);

// Exibe a opção selecionada

Serial.println("1.(+) | 2.(-) | 3.(\*) | 4.(/)");

Serial.println("Operacao: +");

}

void loop() {

estadoBotaoUpAnterior = digitalRead(botaoUpPin);

estadoBotaoDownAnterior = digitalRead(botaoDownPin);

estadoBotaoPlayAnterior = digitalRead(botaoPlayPin);

// Seleciona a opção com os botões Up e Down

if (!operacaoSelecionada) {

// Verifica o botão Up

if (digitalRead(botaoUpPin) == HIGH) {

opcaoSelecionada++;

if (opcaoSelecionada == 1) {

Serial.println("Operacao: +");

}

else if (opcaoSelecionada == 2){

Serial.println("Operacao: -");

}

else if (opcaoSelecionada == 3){

Serial.println("Operacao: \*");

}

else if (opcaoSelecionada == 4){

Serial.println("Operacao: /");

}

if (opcaoSelecionada > 4) { // Define o limite máximo de opções

opcaoSelecionada = 1;

Serial.println("Operacao: +");

}

delay(250); // Adiciona um pequeno atraso para evitar leituras repetidas

}

// Verifica o botão Down

if (digitalRead(botaoDownPin) == HIGH) {

opcaoSelecionada--;

if (opcaoSelecionada == 1) {

Serial.println("Operacao: +");

}

else if (opcaoSelecionada == 2){

Serial.println("Operacao: -");

}

else if (opcaoSelecionada == 3){

Serial.println("Operacao: \*");

}

else if (opcaoSelecionada == 4){

Serial.println("Operacao: /");

}

if (opcaoSelecionada < 1) { // Define o limite mínimo de opções

opcaoSelecionada = 4;

Serial.println("Operacao: /");

}

delay(250); // Adiciona um pequeno atraso para evitar leituras repetidas

}

}

// Aguarda o botão Play para prosseguir

if (!operacaoSelecionada && digitalRead(botaoPlayPin) == HIGH && estadoBotaoPlayAnterior == HIGH) {

operacaoSelecionada = true;

if (opcaoSelecionada == 1) {

Serial.println("Operacao selecionada: +");

}

else if (opcaoSelecionada == 2){

Serial.println("Operacao selecionada: -");

}

else if (opcaoSelecionada == 3){

Serial.println("Operacao selecionada: \*");

}

else if (opcaoSelecionada == 4){

Serial.println("Operacao selecionada: /");

}

delay(1000);

}

if (operacaoSelecionada) {

if (!primeiroValorInserido) {

Serial.println("Insira o valor 1");

while (digitalRead(botaoPlayPin) == LOW) {

// Aguarda o botão Play ser pressionado

}

primeiroValor = lerValor();

Serial.println("Valor1: ");

Serial.println(primeiroValor);

Serial.println("Aguarde...");

delay(1000); // Aguarda 2 segundos

primeiroValorInserido = true;

}

else {

Serial.println("Insira o valor 2");

while (digitalRead(botaoPlayPin) == LOW) {

// Aguarda o botão Play ser pressionado

}

segundoValor = lerValor();

Serial.println("Valor2: ");

Serial.println(segundoValor);

Serial.println("Aguarde...");

delay(1000); // Aguarda 2 segundos

if (opcaoSelecionada == 1){

// Soma

resultado = primeiroValor + segundoValor;

Serial.println("Resultado:");

Serial.println(resultado);

} else if (opcaoSelecionada == 2){

// Subtração

resultado = primeiroValor - segundoValor;

Serial.println("Resultado:");

Serial.println(resultado);

} else if (opcaoSelecionada == 3){

// Multiplicação

resultado = primeiroValor \* segundoValor;

Serial.println("Resultado:");

Serial.println(resultado);

} else if (opcaoSelecionada == 4){

// Divisão

if (segundoValor == 0){

Serial.println("Impossível dividir por zero!");

} else {

resultado = (float) primeiroValor / segundoValor;

Serial.println("Resultado:");

Serial.println(resultado);

}

}

delay(1000);

operacaoSelecionada = false; // Reinicia o processo

primeiroValorInserido = false;

}

}

delay(100);

}

**Versão 1.3**

const int numPins = 3;

int analogPins[numPins] = {0, 1, 2};

char pinLabels[numPins][3] = {"C:", "D:", "U:"};

float Vout[numPins];

float R1 = 1000;

float R2[numPins];

const int botaoUpPin = 8;

const int botaoDownPin = 7;

const int botaoPlayPin = 6;

int opcaoSelecionada = 1;

long primeiroValor;

long segundoValor;

float resultado;

bool operacaoSelecionada = false;

int lerValor() {

float Vout[numPins];

int valor = 0;

for (int i = 0; i < numPins; i++) {

int raw = analogRead(analogPins[i]);

float Vout = (raw \* 5.0) / 1024.0;

R2[i] = R1 \* (5.0 / Vout - 1) ; // Simplified R2 calculation

valor = valor \* 10 + int(round(R2[i] / 1000.0));

}

return valor;

}

void exibirOperacaoSelecionada() {

switch (opcaoSelecionada) {

case 1:

Serial.println("Operacao: +");

break;

case 2:

Serial.println("Operacao: -");

break;

case 3:

Serial.println("Operacao: \*");

break;

case 4:

Serial.println("Operacao: /");

break;

}

}

void setup() {

Serial.begin(9600);

pinMode(botaoUpPin, INPUT);

pinMode(botaoDownPin, INPUT);

pinMode(botaoPlayPin, INPUT);

Serial.println("Sistema Ok");

delay(1000);

Serial.println("Escolha uma operação:");

Serial.println("1.(+) | 2.(-) | 3.(\*) | 4.(/)");

exibirOperacaoSelecionada();

}

// ... (seu código existente)

bool correcaoPrimeiroValor = false;

bool correcaoSegundoValor = false;

bool primeiroValorInserido = false;

bool segundoValorInserido = false;

void loop() {

int estadoBotaoUp = digitalRead(botaoUpPin);

int estadoBotaoDown = digitalRead(botaoDownPin);

int estadoBotaoPlay = digitalRead(botaoPlayPin);

if (!operacaoSelecionada) {

if (estadoBotaoUp == HIGH) {

opcaoSelecionada = (opcaoSelecionada % 4) + 1;

exibirOperacaoSelecionada();

delay(250);

}

if (estadoBotaoDown == HIGH) {

opcaoSelecionada = (opcaoSelecionada - 2 + 4) % 4 + 1;

exibirOperacaoSelecionada();

delay(250);

}

}

if (!operacaoSelecionada && estadoBotaoPlay == HIGH) {

operacaoSelecionada = true;

Serial.print("Operacao selecionada: ");

Serial.println(opcaoSelecionada);

delay(1000);

}

if (operacaoSelecionada) {

if (!primeiroValorInserido || correcaoPrimeiroValor) {

Serial.println("Insira o valor 1 ou pressione Up para reiniciar");

while (digitalRead(botaoPlayPin) == LOW) {

if (digitalRead(botaoUpPin) == HIGH) {

// Se o botão Up for pressionado, reinicie o loop

operacaoSelecionada = false;

primeiroValorInserido = false;

segundoValorInserido = false;

correcaoPrimeiroValor = false;

correcaoSegundoValor = false;

opcaoSelecionada = 0;

return;

}

}

correcaoPrimeiroValor = false;

primeiroValor = lerValor();

Serial.print("Valor1: ");

Serial.println(primeiroValor);

delay(1000);

// Pergunte se o usuário deseja corrigir o valor

Serial.println("Pressione Down para corrigir o Valor1");

// Aguarde até que o botão Down seja pressionado ou o botão Play seja pressionado para continuar

while (true) {

if (digitalRead(botaoDownPin) == HIGH) {

correcaoPrimeiroValor = true;

break;

}

if (digitalRead(botaoPlayPin) == HIGH) {

Serial.println("Valor1 inserido. Aguarde...");

delay(1000);

break;

}

}

if (correcaoPrimeiroValor) {

// Se a correção estiver ativa, permita que o usuário insira um novo Valor1

primeiroValorInserido = false;

} else {

primeiroValorInserido = true;

}

} else if (primeiroValorInserido && (!segundoValorInserido || correcaoSegundoValor)) {

Serial.println("Insira o valor 2 ou pressione Up para reiniciar");

while (digitalRead(botaoPlayPin) == LOW) {

if (digitalRead(botaoUpPin) == HIGH) {

// Se o botão Up for pressionado, reinicie o loop

operacaoSelecionada = false;

primeiroValorInserido = false;

segundoValorInserido = false;

correcaoPrimeiroValor = false;

correcaoSegundoValor = false;

opcaoSelecionada = 0;

return;

}

}

correcaoSegundoValor = false;

segundoValor = lerValor();

Serial.print("Valor2: ");

Serial.println(segundoValor);

delay(1000);

// Pergunte se o usuário deseja corrigir o valor

Serial.println("Pressione Down para corrigir o Valor2");

// Aguarde até que o botão Down seja pressionado ou o botão Play seja pressionado para continuar

while (true) {

if (digitalRead(botaoDownPin) == HIGH) {

correcaoSegundoValor = true;

break;

}

if (digitalRead(botaoPlayPin) == HIGH) {

Serial.println("Valor2 inserido. Aguarde...");

delay(1000);

break;

}

}

if (correcaoSegundoValor) {

// Se a correção estiver ativa, permita que o usuário insira um novo Valor2

segundoValorInserido = false;

} else {

segundoValorInserido = true;

}

} else {

switch (opcaoSelecionada) {

case 1:

resultado = primeiroValor + segundoValor;

Serial.println("Resultado:");

Serial.println((int) resultado);

break;

case 2:

resultado = primeiroValor - segundoValor;

Serial.println("Resultado:");

Serial.println((int) resultado);

break;

case 3:

resultado = primeiroValor \* segundoValor;

Serial.println("Resultado:");

Serial.println((long) resultado);

break;

case 4:

if (segundoValor == 0) {

Serial.println("Impossível dividir por zero!");

} else {

resultado = float(primeiroValor) / segundoValor;

Serial.println("Resultado:");

Serial.println(resultado);

}

break;

}

delay(1000);

Serial.println("Reiniciando loop...");

Serial.println("Escolha uma operação:");

Serial.println("1.(+) | 2.(-) | 3.(\*) | 4.(/)");

Serial.println("Operacao: +");

opcaoSelecionada = 1;

operacaoSelecionada = false;

primeiroValorInserido = false;

segundoValorInserido = false;

correcaoPrimeiroValor = false;

correcaoSegundoValor = false;

}

}

delay(100);

}

**Adições:**

- Otimização do código

- Retirada de funções referentes ao display LED

- Mudança do tipo de variável int para float

- Adição de outras duas operações matemáticas: multiplicação e divisão

- Lista vertical de opções para as operações matemáticas

- Redução de tempo após algumas ações

- Lista vertical de opções para as operações matemáticas

- Condição para o caso de divisão por zero

- Arredondamento para duas casas decimais para o caso de divisão

- Permite corrigir o valor antes de prosseguir

- Permite reiniciar a operação antes de inserir os valores

## Portanto, foi possível realizar a contagem de resistores associados em parelelo utilizando o Arduino. Com isso, atestada a viabilidade, foi dado início aos testes utilizando o Raspeberry Pi Zero 2W.

Link para o circuito: <https://www.tinkercad.com/things/lQFfcyubIAD?sharecode=RrI1T1UnC0oJmcN4KmV9MNEeHQ4PolZKbdAiFhXOXw8>

## Desenvolvimento utilizando o Raspberry Pi Zero 2W

## Depois de verificar a viabilidade de realizar a contagem do resistores utilizando o Arduino simulado no TinkerCAD, foi iniciado o desenvolvimento do código para o Raspberry Pi Zero 2W.

## Desenvolvimento no KiCAD

## Depois de verificar a viabilidade de realizar a contagem do resistores utilizando o Arduino simulado no TinkerCAD, foi iniciado o desenvolvimento do código para o Raspberry Pi Zero 2W.

**REFERÊNCIAS BIBLIOGRÁFICAS**

RAMALHO, Leonardo Lira. **Projeto de Fonte AC-DC com Diodos**., 2023. Disponível em: https://docs.google.com/document/d/1BWIprbYlGRbey0U-X\_30LiKOUhgmVbQQOVBLm2aINB4/edit. Acesso em: 01 jun. 2023.

**ANEXOS**

| **Anexo** | **Componente** |
| --- | --- |
| Anexo I | Datasheet do Transformador 3FS-XXX & PSS3-XX |
| Anexo II | Diodo 1N4001 |
| Anexo III | Datasheet dos capacitores 3,3µF / 10µF / 820µF |
| Anexo IV | Resistor 50kΩ |
| Anexo V | Resistor 1,8kΩ e 180Ω |
| Anexo VI | LED SMD LN1251C |
| Anexo VII | Diodo Zener 1N5239B |
| Anexo VIII | Resistor 400Ω |
| Anexo IX | Borne de Conexão KRE KF128-V2 |