Robo de Sumo- Equipe 8 Noturno

Componentes: Pedro Ferreira Pereira, João Gabriel, Iuri Bibiano, Filipe Souza, Lais Santana e Josadaque Souza

1.Objetivo:

Este projeto faz parte de uma iniciativa do SENAI-BA, cujo principal objetivo é melhorar o desempenho dos estudantes nas disciplinas de Eletrônica, além de despertar o interesse dos mesmos por projetos nessa área. Para isso, foram unidas duas turmas do curso de Desenvolvimento de Sistemas, ambas do turno noturno: uma do primeiro semestre do curso de Eletrônica e outra do terceiro semestre, que está cursando a disciplina de IoT (Internet das Coisas).

Com base nessa integração, os alunos foram organizados em grupos formados por membros das duas turmas, com a proposta de incentivar a comunicação e a troca de experiências entre os estudantes.

No escopo do projeto, foram disponibilizadas duas opções para desenvolvimento: um carro seguidor de linha e um robô de sumô. Nossa equipe optou pela segunda opção. O projeto consiste em um robô controlado via Bluetooth por um aplicativo de celular, cujo principal objetivo é empurrar o robô adversário para fora da plataforma.

2. Visão geral do projeto:

Este projeto consiste em um robô controlado via Bluetooth, por meio de um aplicativo de celular. Através desse aplicativo, é possível se comunicar com o ESP 32, enviando comandos que são lidos pela placa e convertidos em instruções para os motores, permitindo o controle livre do carrinho.

3.Componentes:

- **1. Chassi:**Componente responsável por fornecer a estrutura física básica do projeto, onde todos os outros elementos serão fixados.
- **1. Placa controladora:** Atua como o “cérebro” da máquina. Essa placa será responsável por armazenar a lógica do projeto e executar os comandos com base em estímulos externos, como os sinais vindos do controle Bluetooth ou dos sensores.
- **1. Motor Shield:** Módulo responsável por intermediar a alimentação dos motores. Ele permite que a placa controladora, que normalmente

suporta até 5V, consiga acionar motores e outros componentes que exigem tensões mais altas.

- **Aplicativo de controle:** Será utilizado um aplicativo já existente para configurar e enviar os comandos ao robô via Bluetooth. Um exemplo é o **Bluetooth electronics** , que permite controlar o robô por meio de botões configuráveis diretamente no celular.
- **2. Motores:** Esses motores serão conectados a uma ponte H, que servirá como intermediária entre a lógica do ESP 32 e os motores, permitindo o controle da direção e velocidade com base nos comandos recebidos.
- **2. Baterias:** Serão utilizadas duas baterias: uma de 3,8 V para alimentar o ESP 32 e outra de 9 V para os dois motores.

4. Montagem:

1. **Monte o chassi:** Imprima o chassi em uma impressora 3D ou compre um pronto.

2. **Fixe os motores:** Encaixe os motores nas laterais do chassi. Dependendo do modelo, pode ser necessário usar cola quente ou abraçadeiras (enforca-gato) para fixá-los firmemente.

3. **Instale a ponte H e o ESP 32:** Fixe a ponte H e o ESP 32 no chassi. O ESP 32 já deve estar carregado com o código adequado antes da montagem.

4. **Conecte os motores à ponte H:** Ligue os dois fios de cada motor à ponte H — um par do lado esquerdo e outro do lado direito. Esses fios determinam o sentido de rotação dos motores (frente ou ré).

5. **Conecte a ponte H ao ESP 32:** Na frente da ponte H, conecte quatro fios de sinal.
    - Os dois pinos mais à esquerda controlam um dos motores.

    - Os dois pinos mais à direita controlam o outro motor.
      Conecte esses fios às portas digitais do ESP 32 conforme definido no seu código.

6. **Alimente a ponte H:** Pegue uma bateria de 9V:

    - Conecte o fio negativo ao pino GND da ponte H.

- Conecte o fio positivo à entrada de alimentação externa da ponte H. A ponte H agora deve estar energizada.

7. **Alimente o ESP 32:** Pegue a bateria de 3,8V:
   - Conecte o fio negativo ao GND do ESP32.
   - Conecte o fio positivo ao pino VIN (ou VCC, dependendo do modelo). O ESP 32 deve ligar.
8. **Verifique o Bluetooth do ESP 32:** Confirme se o Bluetooth está ativo e transmitindo o nome definido no seu código.
9. **Parear com o Bluetooth:** No celular, acesse as configurações de Bluetooth, localize o dispositivo gerado pelo ESP 32 e pareie com ele.
10. **Abra o aplicativo Bluetooth Electronics**
    - Vá até a aba **"Connect"** e selecione **"Bluetooth Normal"**.
    - Escolha o dispositivo Bluetooth do ESP 32 e clique em **"Conectar"**.
11. **Configure o controle no aplicativo**
    - Edite o layout e adicione um **joystick (pad)** ou botões.
    - Configure os sinais que devem ser enviados para o ESP 32 conforme o código.

5 .Código:

1. Inclusão de Biblioteca

```
#include "BluetoothSerial.h"
```

Inclui a biblioteca necessária para comunicação Bluetooth serial no ESP 32, permitindo o envio e recebimento de dados com dispositivos externos, como um smartphone.

---

2. Declaração de Objetos e Variáveis

```
BluetoothSerial SerialBT;
char comando;
```

- `SerialBT`: objeto que gerencia a comunicação Bluetooth.
- `comando`: variável do tipo `char` que armazena o caractere recebido via Bluetooth (representando uma ação de controle).

---

3. Mapeamento dos Pinos

```
const int motor_direita_f = 14;
const int motor_direita_t = 12;
const int motor_esquer_f = 25;
const int motor_esquer_t = 33;
```

Define os pinos digitais do ESP32 responsáveis pelo controle dos motores por meio da ponte H:

- `f` refere-se ao sentido horário (frente).
- `t` refere-se ao sentido anti-horário (trás).

---

4. Função `setup()`

```
Serial.begin(9600);
```

Inicializa a comunicação serial padrão (via cabo USB) com baud rate de 9600 para fins de depuração.

```
pinMode(motor_direita_f, OUTPUT);
pinMode(motor_direita_t, OUTPUT);
pinMode(motor_esquer_f, OUTPUT);
pinMode(motor_esquer_t, OUTPUT);
```

Configura os pinos ligados à ponte H como saídas digitais.

```
SerialBT.begin("Carrinho Equipe 8");
```

Inicializa o Bluetooth com o nome visível **"Carrinho Equipe 8"**, permitindo o pareamento com o celular.

```
Serial.println("Fim Setup");
```

Envia mensagem de confirmação ao terminal indicando que a inicialização foi concluída.

```
digitalWrite(motor_direita_f, LOW);
digitalWrite(motor_direita_t, LOW);
digitalWrite(motor_esquer_f, LOW);
digitalWrite(motor_esquer_t, LOW);
```

Garante que todos os motores estejam desligados no momento da inicialização, evitando acionamento indesejado.

---

5. Função `loop()`

```
if (SerialBT.available()) {
```

Verifica se há dados disponíveis para leitura via Bluetooth.

```
comando = SerialBT.read();
```

Lê um único caractere do buffer Bluetooth e armazena em `comando`.

```
Serial.print("Recebi ASCII: ");
Serial.println((int)comando);
```

Converte o caractere para seu valor ASCII e imprime no monitor serial para fins de depuração.

---

6. Interpretação dos Comandos

A seguir, cada valor de `comando` é mapeado para um movimento específico do robô:

Frente ('1')

```
if (comando == '1') {
  digitalWrite(motor_direita_f, HIGH);
  digitalWrite(motor_direita_t, LOW);
  digitalWrite(motor_esquer_f, HIGH);
  digitalWrite(motor_esquer_t, LOW);

  Serial.println("Frente");
```

- Ambos os motores giram para frente.

Trás ('2')

```
} else if (comando == '2') {
  digitalWrite(motor_direita_f, LOW);
  digitalWrite(motor_direita_t, HIGH);
  digitalWrite(motor_esquer_f, LOW);
  digitalWrite(motor_esquer_t, HIGH);
  Serial.println("Trás");
```

- Ambos os motores giram para trás.

Esquerda ('3')

```
} else if (comando == '3') {
  digitalWrite(motor_direita_f, HIGH);
  digitalWrite(motor_direita_t, LOW);
  digitalWrite(motor_esquer_f, LOW);
  digitalWrite(motor_esquer_t, LOW);
  Serial.println("Esquerda");
```

- Apenas o motor direito é acionado para frente, provocando rotação à esquerda.

Direita ('4')

```
} else if (comando == '4') {
  digitalWrite(motor_direita_f, LOW);
  digitalWrite(motor_direita_t, LOW);
  digitalWrite(motor_esquer_f, HIGH);
  digitalWrite(motor_esquer_t, LOW);
  Serial.println("Direita");
```

- Apenas o motor esquerdo é acionado para frente, provocando rotação à direita.

Parado (qualquer outro caractere)

```
} else {
  digitalWrite(motor_direita_f, LOW);
  digitalWrite(motor_direita_t, LOW);
  digitalWrite(motor_esquer_f, LOW);
  digitalWrite(motor_esquer_t, LOW);
  Serial.println("parado");
}
```

- Todos os pinos de controle são desativados, interrompendo o movimento.