Universidade Federal de Campina Grande - UFCG
João Victor de Carvalho Araujo
Relatório TrainIEEE 2023 - RAS

I-CYBIE ARDUINO

Relatório de Atividades

Para início das atividades, foi necessária a análise da estrutura e funcionamento do I-Cybie, com base nas informações presentes em seu manual.

Primeiramente, verifiquei os sensores, atuadores e demais componentes presentes na estrutura como um todo, sendo estes:

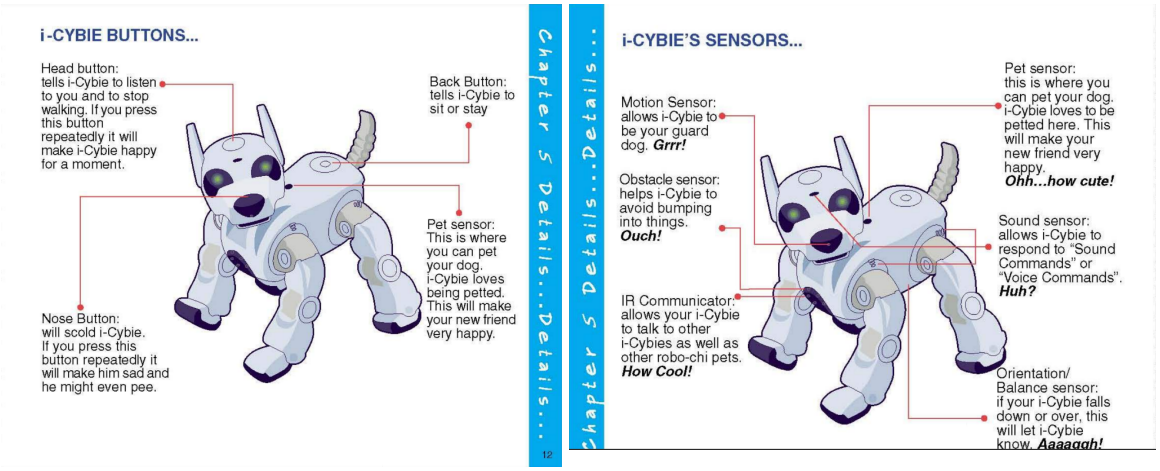
Botões:

- **Cabeça** - faz o I-Cybie receber comandos de voz e parar de se movimentar, além de interagir com o usuário ficando “feliz”* se for apertado repetidamente;
- **Nariz** - Repreende o I-Cybie, além de interagir com o usuário, ficando “triste”*, caso apertado repetidamente;
- **Costas -** Faz o I-Cybie receber comandos de ficar ou sentar.

*O produto possui uma programação baseada em humores, que alteram de acordo com a interação do usuário.

Sensores:

- **Sensor de Movimento -** O I-Cybie utiliza esse sensor para identificar movimentos próximos, interagindo com eles de acordo com o movimento;
- **Sensor de Obstáculo -** sensor digital composto por um emissor e um receptor de infravermelho, utilizado para evitar que o I-Cybie entre em colisão com outros objetos;
- **Comunicador Infravermelho -** Funcionando de maneira similar a da interação de uma Televisão com seu controle, faz com que o I-Cybie interaja com outros do mesmo modelo;
- **Sensor de Afeto -** Tradução mais adequada que achei para o sensor, creio que seja um sensor de toque, que faz o I-Cybie interagir com o usuário simulando que está recebendo carinho;
- **Sensor Sonoro -** Faz com que o I-Cybie receba comandos de voz, e interaja com o usuário, de acordo com uma voz pré definida e salva em sua memória;
- **Sensor de Orientação/Equilíbrio -** Detecta se o I-Cybie tombou ou caiu de algum lugar, interagindo com o usuário caso isso venha a acontecer.



Em resumo, no manual, identifiquei então que: O i-Cybie é um robô cão programável. Ele pode responder a comandos de voz de um usuário específico e possui reconhecimento de fala "dependente do falante". O microcontrolador RSC 300/364 pode processar comandos em qualquer idioma e possui autenticação biométrica. Há 8 comandos de voz listados no manual

para acionar 8 categorias diferentes de comportamento. Além da voz, o i-Cybie pode ouvir sons agudos, contar aplausos e até detectar quando cai e se levantar automaticamente. Também pode perceber espaços confinados e obstáculos, "ver" movimento usando sensores de luz em seu nariz, além de detectar níveis de luz e movimento. O i-Cybie possui várias formas de andar, pode executar diversas ações e até procurar seu carregador para recarregar sua bateria quando está fraca. Resumidamente, o i-Cybie é um robô cão versátil com capacidades de reconhecimento de voz, detecção de ambiente e movimento, e autossuficiência na recarga de sua bateria.

Em segunda instância, após leitura do manual para entendimento dos princípios do I-Cybie, realizei a análise da pata do I-Cybie desmontado presente no laboratório para estudo de sua estrutura e tentativa de compreensão de seu princípio de de funcionamento e compreensão de acionamento dos motores.

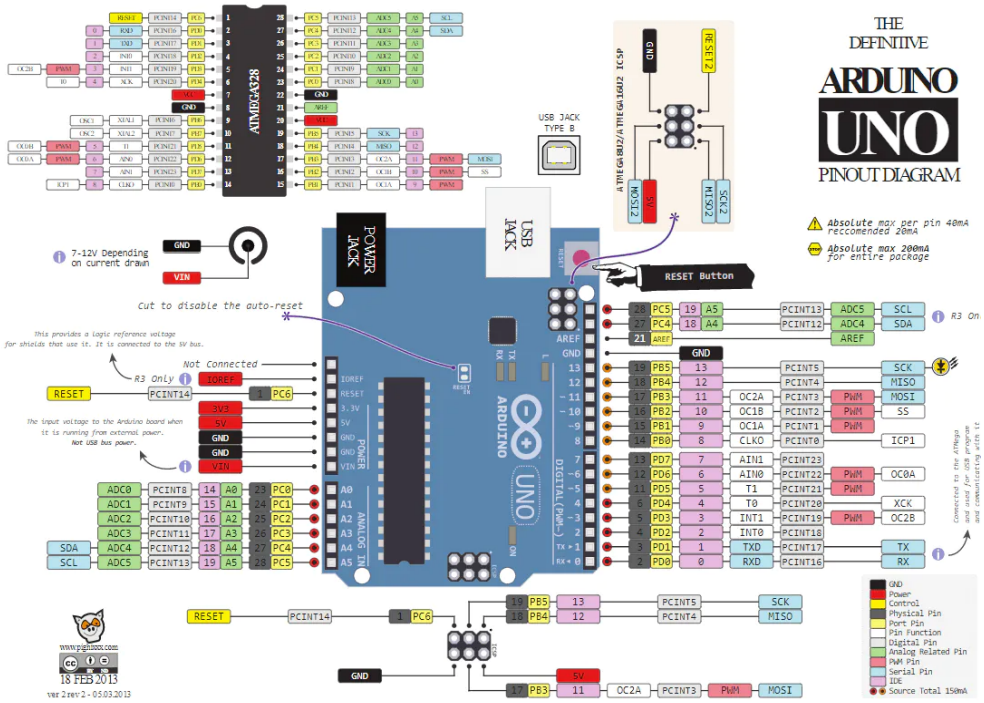
- **Estrutura das Patas**: Cada pata do i-Cybie consiste em articulações que permitem movimentos flexíveis. Elas são projetadas para se assemelhar às patas de um cão real, proporcionando estabilidade e mobilidade ao robô.
- **Detecção de Posição e Pressão**: Sensores estão incorporados nas patas para monitorar a posição da pata e a pressão exercida sobre o chão. Isso permite ao i-Cybie adaptar seu movimento de acordo com a superfície em que está caminhando e manter o equilíbrio.
- **Ativação**: As patas do i-Cybie são ativadas quando o robô está em funcionamento. Elas respondem aos comandos do usuário ou aos algoritmos de controle que determinam como o robô deve se mover. Por exemplo, quando o usuário ordena ao i-Cybie para andar para frente, os motores nas patas são acionados para gerar os movimentos necessários.
- **Princípios de Funcionamento**: As patas do i-Cybie funcionam com base em princípios de cinemática e controle. Os motores nas articulações das patas são controlados de forma a gerar movimentos coordenados para caminhar, correr, virar ou realizar outras ações, dependendo dos comandos ou do modo de operação selecionado pelo usuário. Os sensores de posição e pressão nas patas ajudam a ajustar o movimento e a manter o equilíbrio do robô.


Desse modo, com a análise das patas, verifiquei que ela possui 3 servos que são regulados por potenciômetros, dos quais foram fundamentais nos estudos da próxima

etapa na tentativa de tentar replicar parcialmente esta funcionalidade nas simulações de arduino.

Primeiramente, nessa próxima etapa, para realizar meu primeiro contato com a placa de arduino, esquematizei alguns tópicos para compreensão de funcionamento.

- **Placa de prototipagem Arduino:** Arduino é uma plataforma de eletrônica de código aberto baseada em hardware e software fáceis de usar. As placas Arduino são capazes de ler entradas, sendo estas a mais comuns como luz em um sensor, um dedo em um botão e transformá-la em uma saída, ativando um motor ou acendendo um LED. Existem vários tipos de placas Arduino, como o Arduino Uno e o Arduino Mega, cada uma com suas próprias especificações e pinos;
- **Configuração dos Pinos Gerais de Entrada e Saída (GPIO):** Os pinos do Arduino podem ser configurados como entradas ou saídas. Quando configurados como entradas, eles trazem informações para a placa a partir de um dispositivo de entrada para o processador na placa do microcontrolador. Quando servem como pinos de saída, os dados da placa são transferidos para um dispositivo de saída a partir do processador;
- **Leitura e escrita digital**: Para esta função, temos os seguintes comandos - A função digitalRead() lê o valor de um pino digital especificado, que pode ser HIGH ou LOW. A função digitalWrite() é usada para escrever um valor HIGH ou LOW em um pino digital;
- **Modulação por Largura de Pulso (PWM)**: A Modulação por Largura de Pulso, ou PWM, é uma técnica para obter resultados analógicos com meios digitais. O controle digital é usado para criar uma onda quadrada, um sinal alternado entre ligado e desligado. Isso pode simular tensões entre a tensão total da placa (por exemplo, 5 V no Uno, 3.3 V em uma placa MKR) e desligado (0 Volts) alterando a parte do tempo que o sinal passa ligado versus o tempo que o sinal passa desligado.



Fonte: [The Full Arduino Uno Pinout Guide [including diagram] (circuito.io)](The Full Arduino Uno Pinout Guide [including diagram] (circuito.io))

Logo em seguida, após compreensão da placa de arduino em si. Foi dado início às simulações e programação com a placa. Para realização desta etapa, utilizei do TinkerCAD, com a placa Arduino UNO R3. dando ênfase em testes com Servos* e Potenciômetro**

* são usados em várias aplicações quando se deseja movimentar algo de forma precisa e controlada. Sua característica mais marcante é a sua capacidade de movimentar os seu braço até uma posição e mantê-lo, mesmo quando sofre uma força em outra direção.

** aplicado em ajuste de volume de áudio, seleção de temperatura, ajuste de iluminação, controle de movimento de robôs e ajustes de sinal em módulos eletrônicos.

Para o primeiro teste de simulação e compreensão, realizei o controle de dois Servos que se movimentam de acordo com indicação do usuário no terminal.

***Observação: os códigos para cada uma destas simulações estarão anexados ao fim do relatório.**

Sendo montado nesta esquemática:

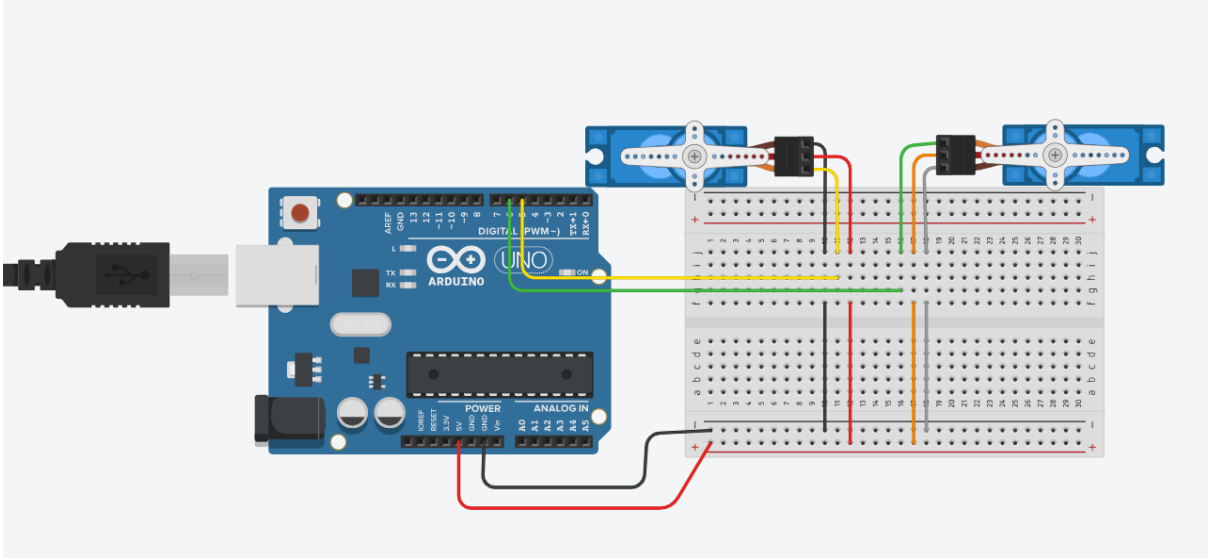


Figura 1.

Posteriormente, a simulação seguinte para estudo dos mecanismos do Arduino e compreensão de funcionamento do I-Cybie, foi o controle de um servo, no qual sua movimentação era ajustada de acordo com o controle de um potenciômetro.

No qual obteve a seguinte aparência:

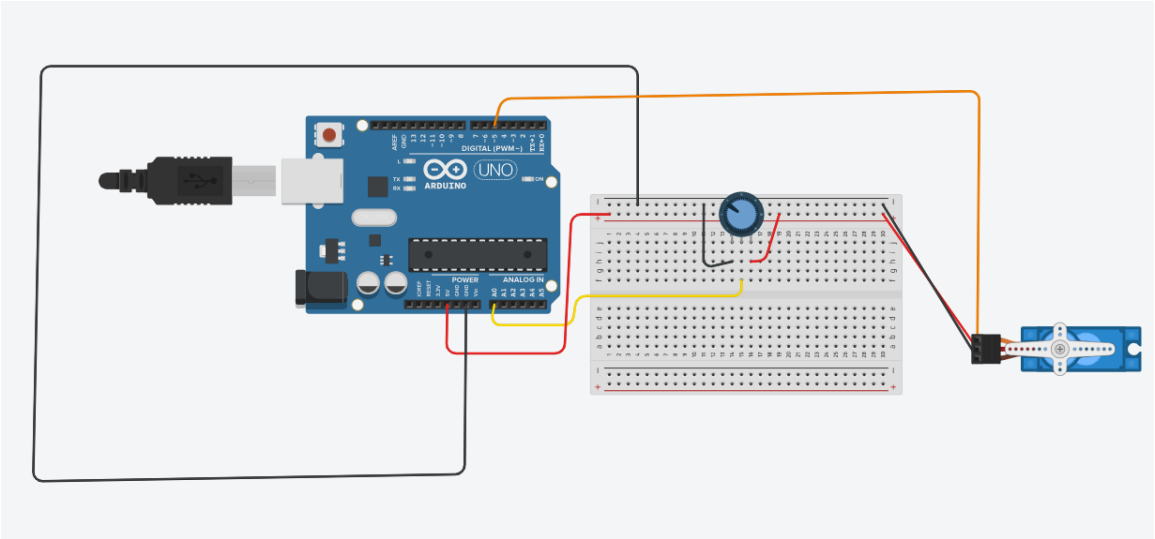


Figura 2.

A próxima simulação foi realizada sem a utilização do potenciômetro, tendo apenas o controle de dois servos que variam a sua movimentação em 180 graus, de forma automática, ou seja, indo e voltando quando chegam na posição 180 e na posição 0.

No qual foi organizado da seguinte forma:

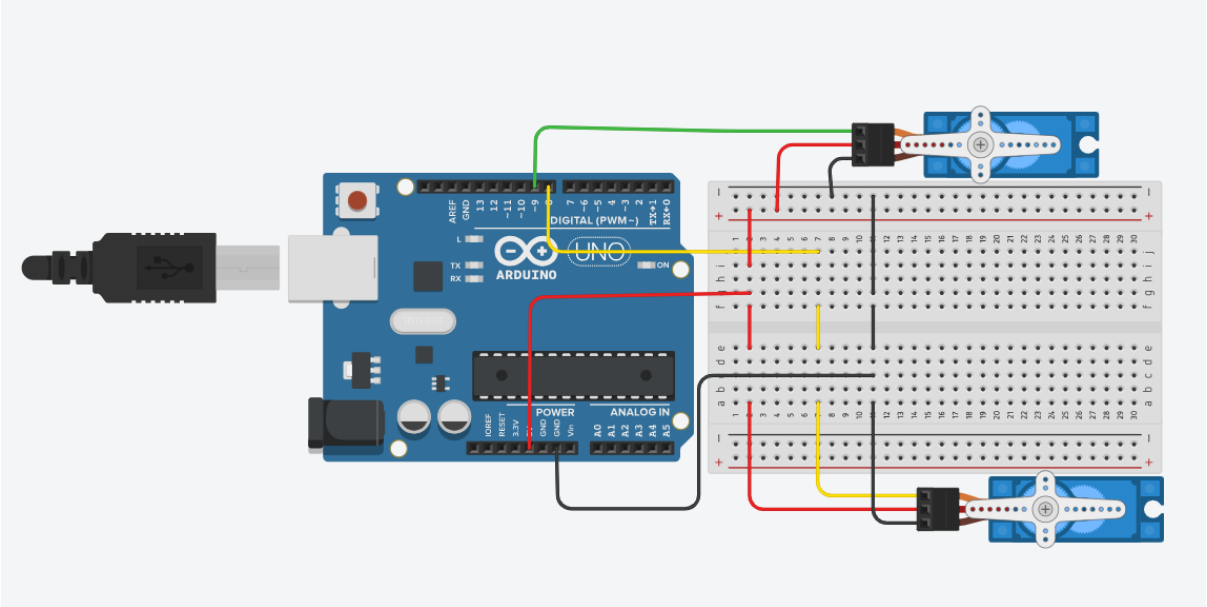


Figura 3.

A próxima simulação foi a mais próxima que cheguei na tentativa de replicar o controle das patas do I-Cybie. Utilizando de 3 Servos que foram alimentados por 4 pilhas AAA de 1.5V, sendo cada um deles controlado por Potenciômetros que estavam ligados diretamente na placa. Dessa forma, cada um dos Servos poderia ter uma movimentação diferente conforme ajustado em seu respectivo potenciômetro, replicando assim as articulaçoes do joelho e da parte superior da pata do I-Cybie

Na qual, foi organizada da seguinte maneira:

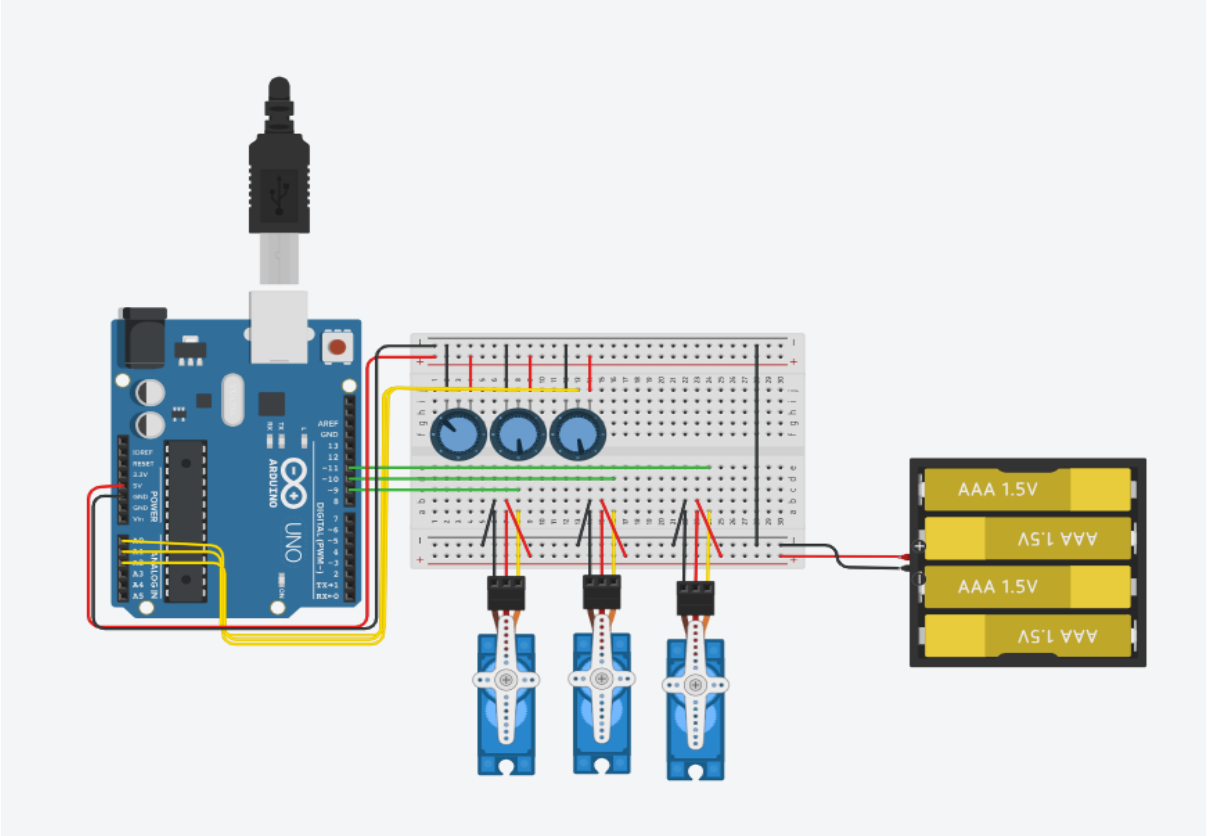


Figura 4.

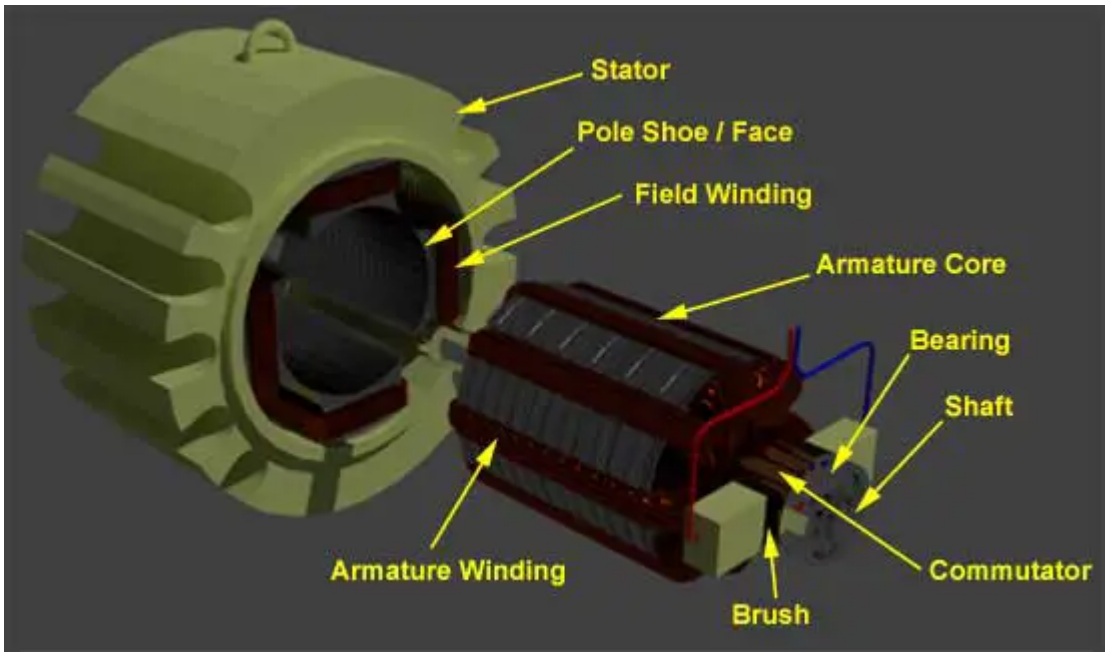
Para melhor compreensão da programação de arduino e orientação para realizar cada uma dessas simulações, foi utilizado como base, o livro ***Arduino Basics - Michael McRoberts.***

Após realização das simulações e compreensão do Arduino, foi necessário a compreensão do funcionamento de Motores DC e circuitos de acionamento do mesmo, como por exemplo, a ponte H, desse modo, investigando como o controlador possa controlar de forma eficaz os motores.

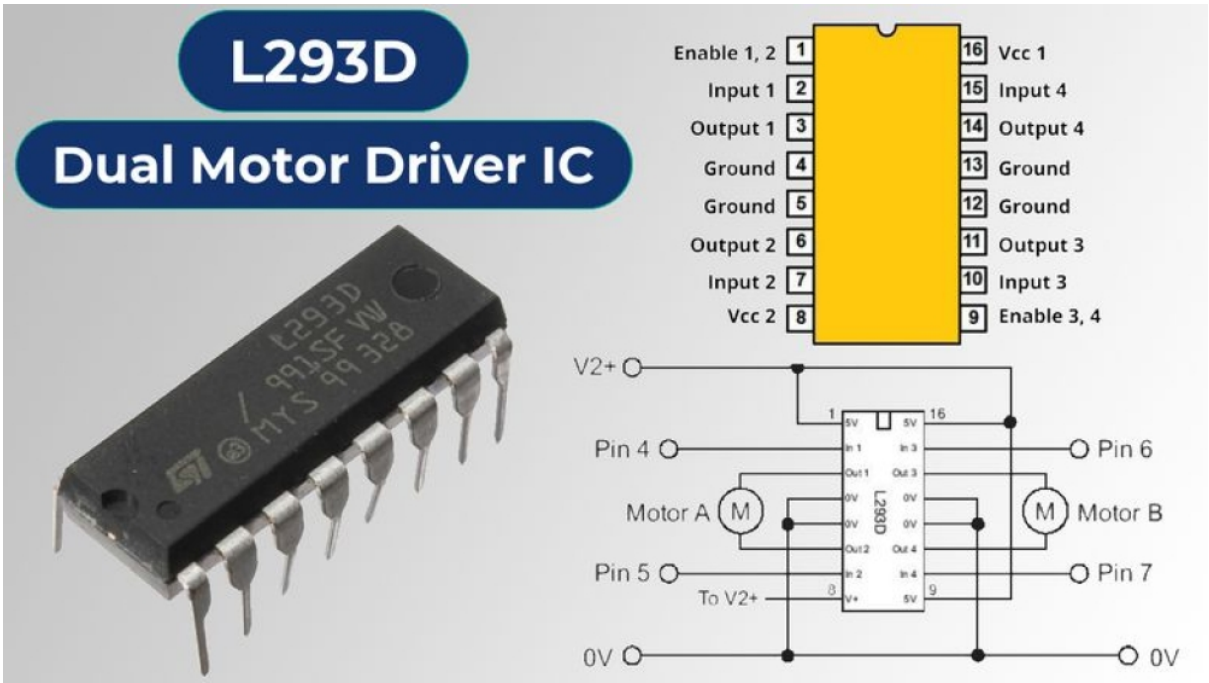
Acionamento e funcionalidades de Motores DC:

- **Circuitos de Acionamento de Motores DC:** Os circuitos de acionamento de motores DC são usados para controlar a velocidade e a direção dos motores DC. Eles podem ser integrados em pequenos módulos ou construídos a partir de componentes discretos, especialmente quando são necessárias potências mais altas.
- **Ponte H:** A Ponte H é um circuito eletrônico que alterna a polaridade de uma tensão aplicada a uma carga. Este circuito é frequentemente usado em robótica e outras aplicações para permitir que os motores DC funcionem para frente ou para trás. A Ponte H é construída com quatro chaves (transistores de estado sólido ou mecânicos). Quando as chaves S1 e S4 são fechadas (e S2 e S3 estão abertas), uma tensão positiva é aplicada ao motor. Ao abrir as chaves S1 e S4 e fechar as chaves S2 e S3, essa tensão é invertida, permitindo a operação reversa do motor
- **Microcontrolador Controlando Motores DC:** Em resumo, os microcontroladores são uma ótima maneira de automatizar dispositivos motorizados. Com uma ampla gama de drivers de motor, você

pode usar um microcontrolador como um Arduino para controlar praticamente qualquer tamanho de motor DC. A seleção do driver do motor pode ser bastante importante. Se o único requisito for ligar e desligar um motor sem considerar o controle de velocidade ou direção, então um relé ou MOSFET seria suficiente. Mas na maioria das aplicações, você vai querer ser capaz de ajustar os parâmetros do seu motor, e para isso, você vai precisar de um driver de motor.



Componentes de um motor DC:
Fonte: Construction of DC Motor (Parts & Images) | Electrical4U



Ponte H módulo L293D.
Fonte: L293D Dual H-Bridge Motor Driver IC Pins, Circuit, Working (how2electronics.com)

Não foram realizados testes com este módulo em mãos, entretanto, como este é um modelo disponível no Laboratório, aqui está um estudo feito a respeito de seus componentes:

Configuração e componentes de uma Ponte H L293D:

- **L293D IC:** O L293D é um IC de driver de motor muito popular. Internamente, ele contém duas Pontes H que permitem que você controle individualmente 2 motores bidirecionalmente com até 36V a 0.6A cada (1A para surtos curtos). Isso equivale a 21.6 Watts (36 Watts para surtos curtos). Ele também contém dois pinos de habilitação, um para cada lado.
- **Configuração do L293D IC:** O L293D IC funciona no princípio básico da Ponte H, este circuito de controle do motor permite que a tensão flua em qualquer direção. Como sabemos, a tensão deve mudar de direção para ser capaz de girar o motor DC em ambas as direções. Portanto, os circuitos da Ponte H usando ICs L293D são perfeitos para acionar um motor.
- **Pinagem do L293D IC:** O L293D consiste em dois circuitos de Ponte H. Existem 2 pinos de SAÍDA, 2 pinos de ENTRADA e 1 pino de HABILITAÇÃO para acionar cada motor.

Em resumo, os princípios de acionamento e funcionamento de um motor DC são:

1. **Princípio de Funcionamento do Motor DC:** O princípio de funcionamento de um motor DC baseia-se no fato de que quando um condutor que transporta corrente é colocado em um campo magnético, o condutor experimenta uma força mecânica. A direção e a magnitude desta força são dadas pela regra da mão esquerda de Fleming
2. **Circuitos de Acionamento de Motores DC:** Os circuitos de acionamento de motores; DC são usados para controlar a velocidade e a direção dos motores DC3. Eles podem ser integrados em pequenos módulos ou construídos a partir de componentes discretos, especialmente quando são necessárias potências mais altas.

Conclusão:

Dado o prazo de duas semanas, foram estas as atividades realizadas para compreensão de funcionamento do I-Cybie e de sua estrutura, visando buscar uma base concisa para introdução ao projeto e buscando dar uma continuidade com base no que foi feito e observado até então. Abordando os seguintes tópicos:

1. Planejamento Inicial: Análise das funcionalidades do I-Cybie e estrutura, com o objetivo de compreender os sensores e atuadores disponíveis.
2. Programação e Microcontrolador: Investigação a placa de prototipagem Arduino, compreendendo como funciona sua programação;
3. Circuitos de Acionamento de Motores DC: Compreensão do funcionamento dos circuitos de acionamento de motores DC, como a Ponte H, para garantir que o microcontrolador possa controlar eficazmente os motores.
4. Princípios subjacentes a esses circuitos

Campina Grande, 20/10/2023

Anexo:

Aqui estarão disponíveis os códigos utilizados nas simulações realizadas no TinkerCAD com a placa de Arduino.

Figura 1:

```
void loop() {
  if (Serial.available() > 0) {
    int index = 0;
    delay(100);
    int numChar = Serial.available();
    if (numChar>10) {
      numChar=10;
    }
    while (numChar--) {
      buffer[index++] = Serial.read();
    }
    splitString(buffer);
  }
}
void splitString(char* data) {
  Serial.print("Data entered: ");
  Serial.println(data);
  char* parameter;
  parameter = strtok (data, " ,");
  while (parameter != NULL) {
    setServo(parameter);
    parameter = strtok (NULL, " ,");
  }
  for (int x=0; x<9; x++) {
    buffer[x]='\0';
  }
```

```
  Serial.flush();
 }

 void setServo(char* data) {
  if ((data[0] == 'L') || (data[0] == 'l')) {
    int firstVal = strtol(data+1, NULL, 10);
    firstVal = constrain(firstVal,0,180);
    servo1.write(firstVal);
    Serial.print("Servo1 is set to: ");
    Serial.println(firstVal);
  }
  if ((data[0] == 'R') || (data[0] == 'r')) {
    int secondVal = strtol(data+1, NULL, 10);
    secondVal = constrain(secondVal,0,255);
    servo2.write(secondVal);
    Serial.print("Servo2 is set to: ");
    Serial.println(secondVal);
  }
 }
```

Figura 2:

```
#include <Servo.h>

Servo servo1;

void setup() {
  servo1.attach(5);
}

void loop() {
  int angle = analogRead(0);
```

```
  angle=map(angle, 0, 1023, 0, 180);
  servo1.write(angle);
  delay(15);
}
```

Figura 3:

```
#include <Servo.h>

int pos = 0;

Servo servo_9;
Servo servo_8;

void setup()
{
  servo_9.attach(9, 500, 2500);
  servo_8.attach(8, 500, 2500);
}

void loop()
{

  for (pos = 0; pos <= 180; pos += 1) {
    servo_9.write(pos);
    servo_8.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    servo_9.write(pos);
    servo_8.write(pos);
    delay(15);
```

```
  }
}
```

Figura 4:

```
#include <Servo.h>
Servo servo3;
Servo servo5;
Servo servo6;

int potpin = 0;
int potpin2 = 1;
int potpin3 = 2;

int val = 0;
int val2 = 0;
int val3 = 0;

void setup(){
  servo3.attach(9);
  servo5.attach(10);
  servo6.attach(11);
}

void loop(){
  val = analogRead(potpin);
  val = map(val, 3, 1023, 0, 176);
  servo3.write(val);
  delay(25);

  val2 = analogRead(potpin2);
  val2 = map(val2, 3, 1023, 0, 176);
```

```
  servo5.write(val2);
  delay(25);

  val3 = analogRead(potpin3);
  val3 = map(val3, 3, 1023, 0, 175);
  servo6.write(val3);
  delay(25);
}
```
