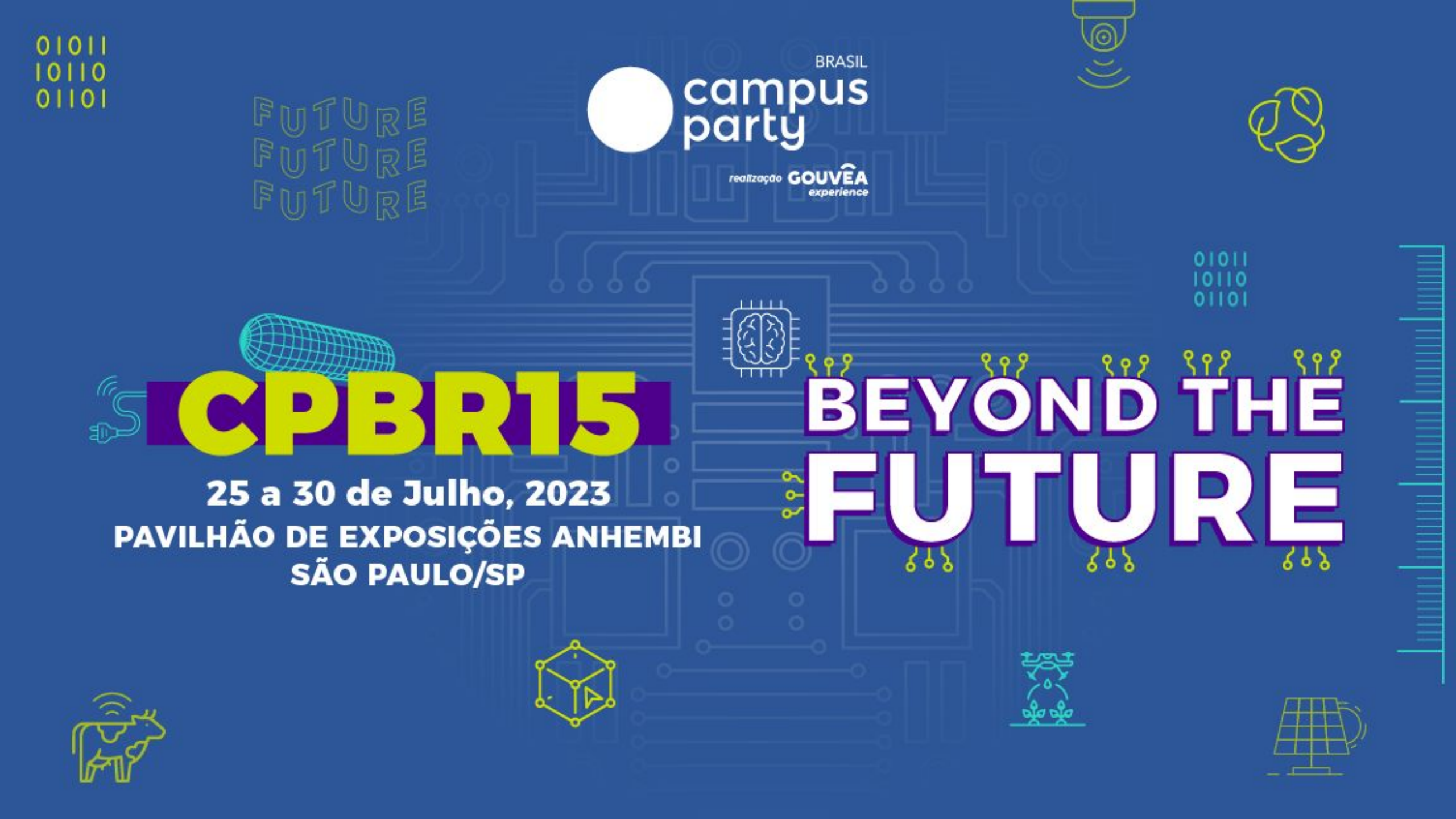

BRASIL
campus party
realização GOUVÊA experience
CPBR15
25 a 30 de Julho, 2023
PAVILHÃO DE EXPOSIÇÕES ANHEMBI
SÃO PAULO/SP
BEYOND THE FUTURE

Construindo um Robô Trekking versão 2.0

Pedro Igor Borçatti da Silva - pibscontato@gmail.com - @pedro_ibs

Qual o Objetivo?

O que eu quero com esse robô?

Como fazer?

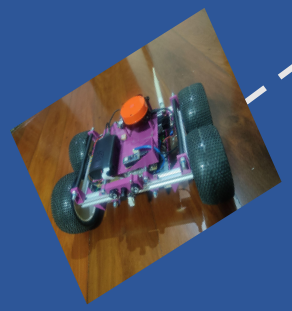
Quem vai fazer?

Para quem não sabe para onde vai qualquer caminho serve!!

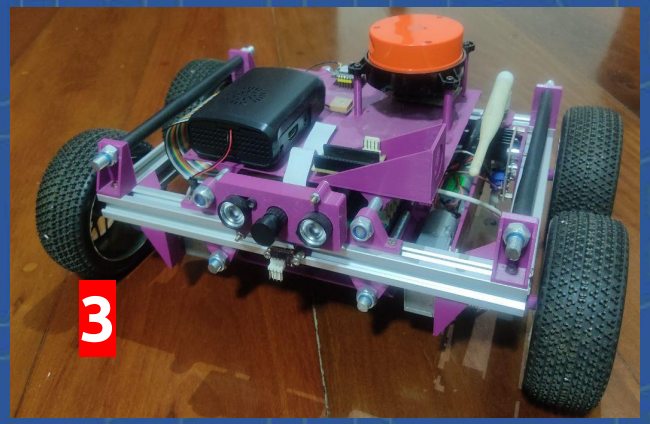
O que o robô deve fazer?

Em quanto tempo?

Ir de um ponto a outro sem a minha interferência!



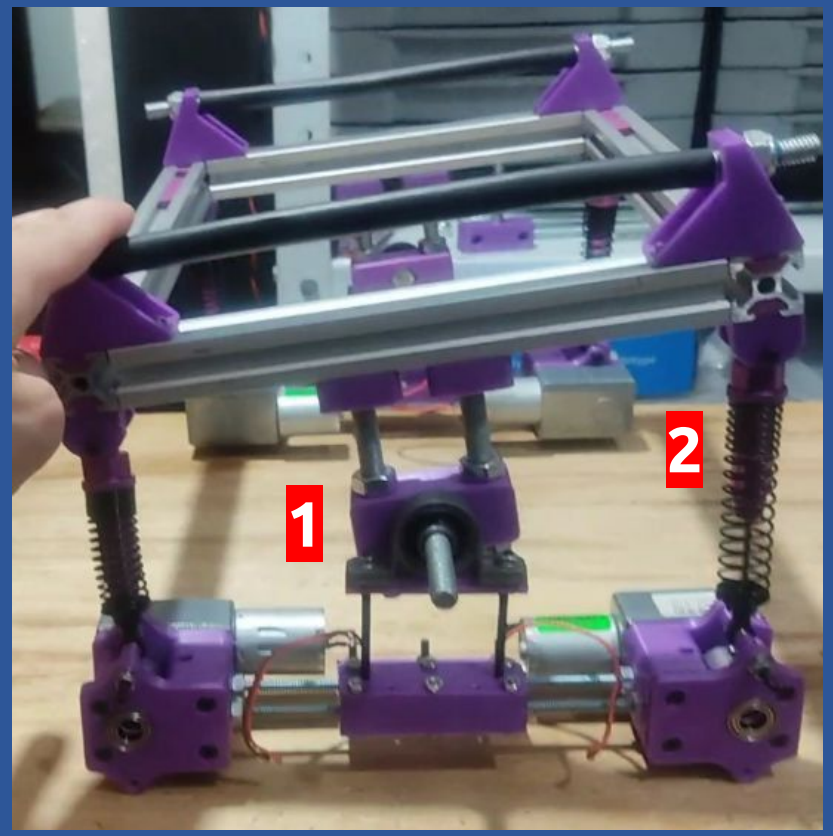
Um Breve Resumo

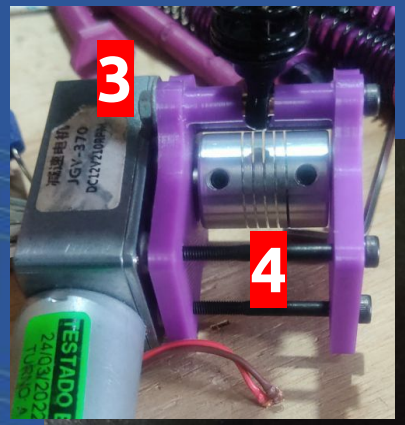


Tudo começa com um computador na CPBR 14

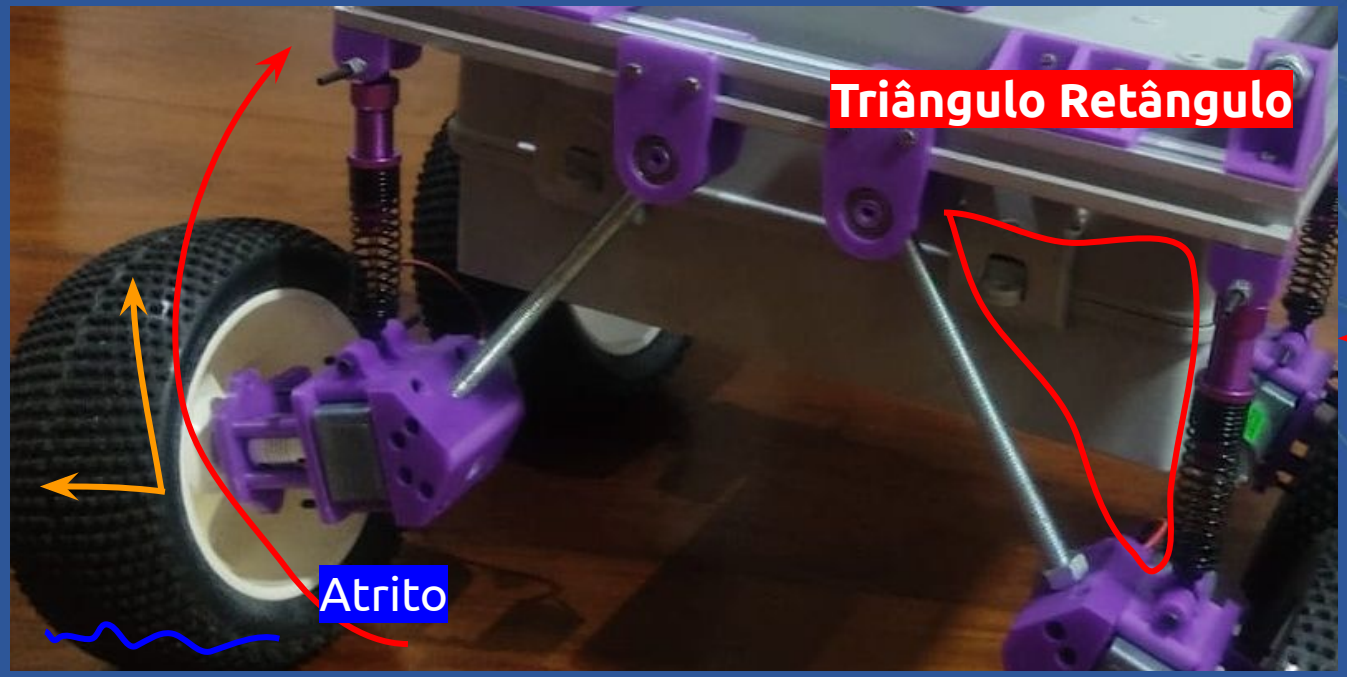
- Partes da eletrônica;
- Sistema de Supervisão;
- Suspenção;
- Motores.

Suspensão

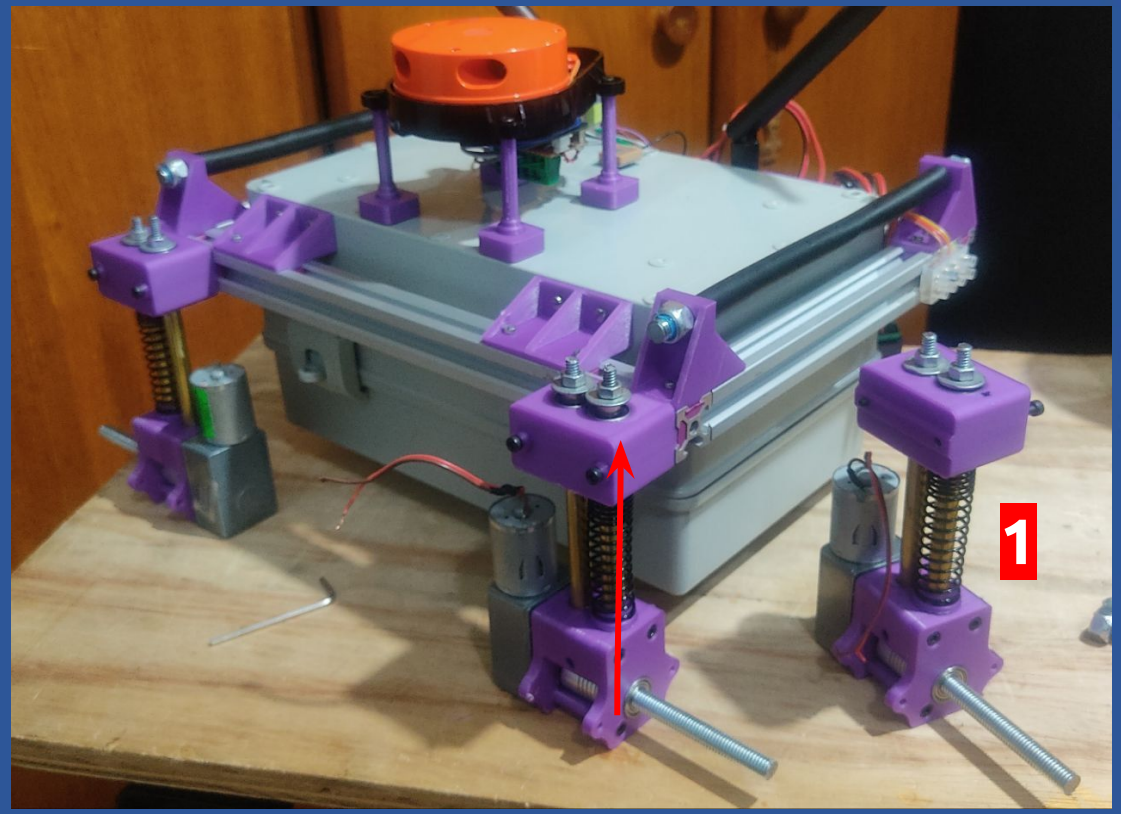


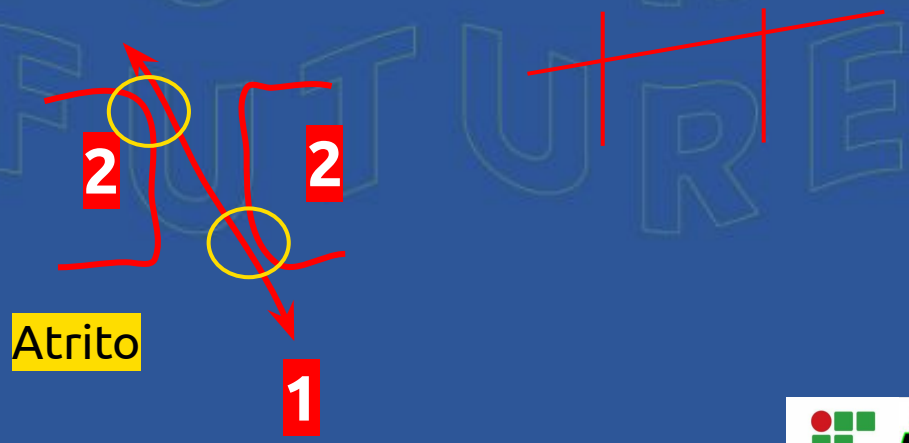


Suspensão

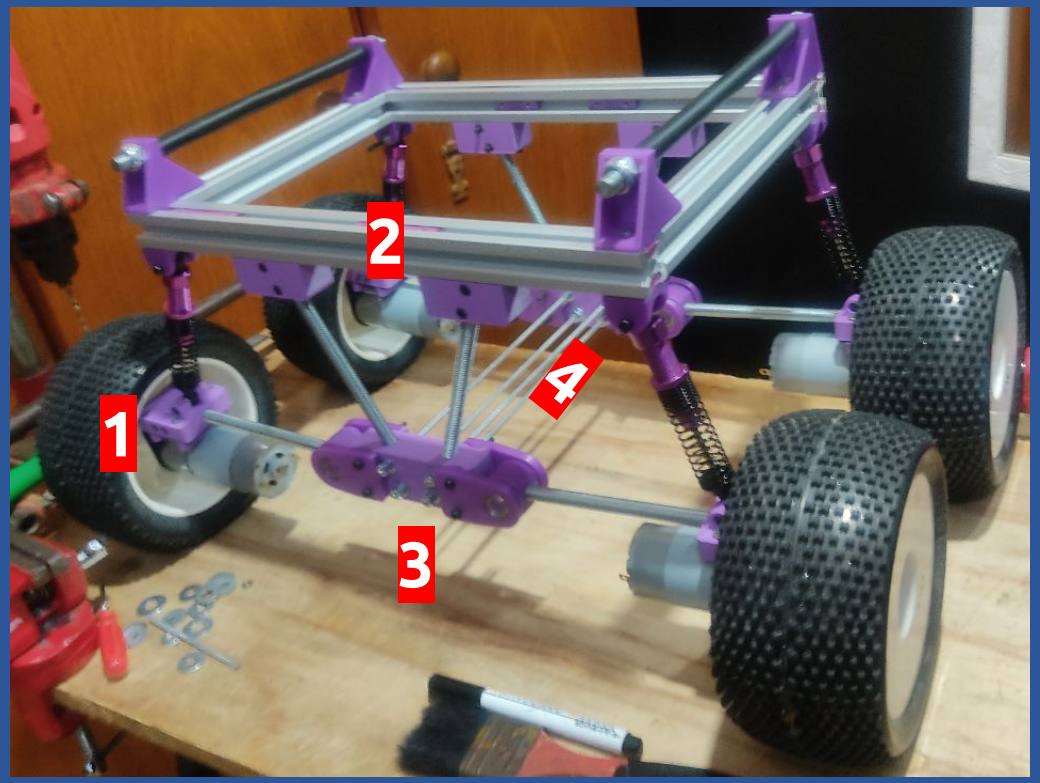


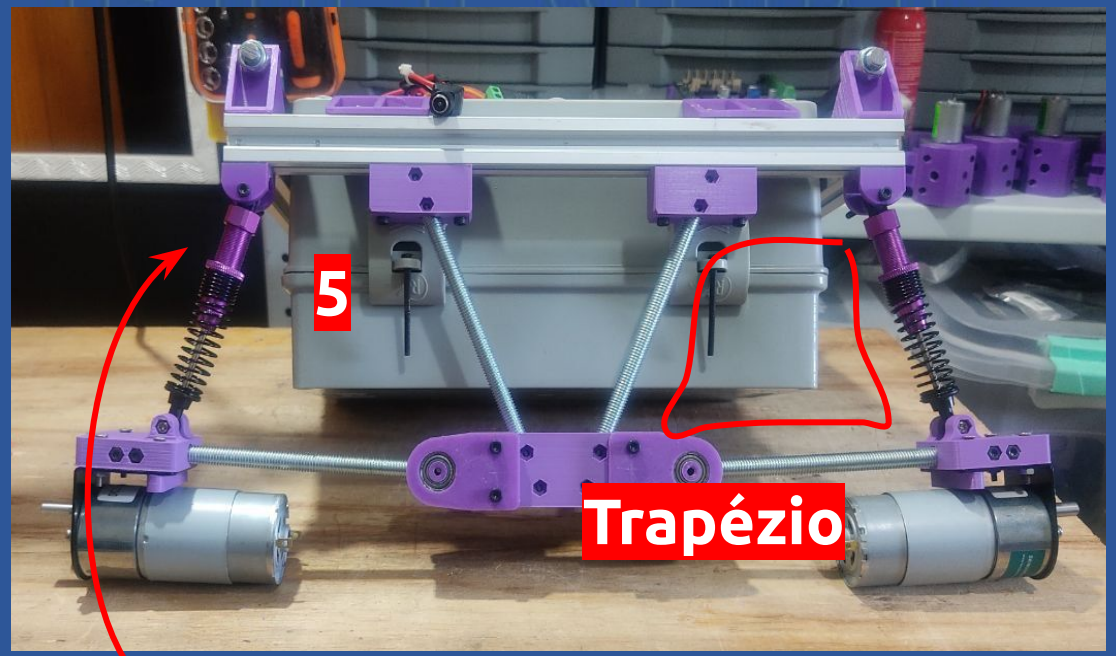
Suspensão



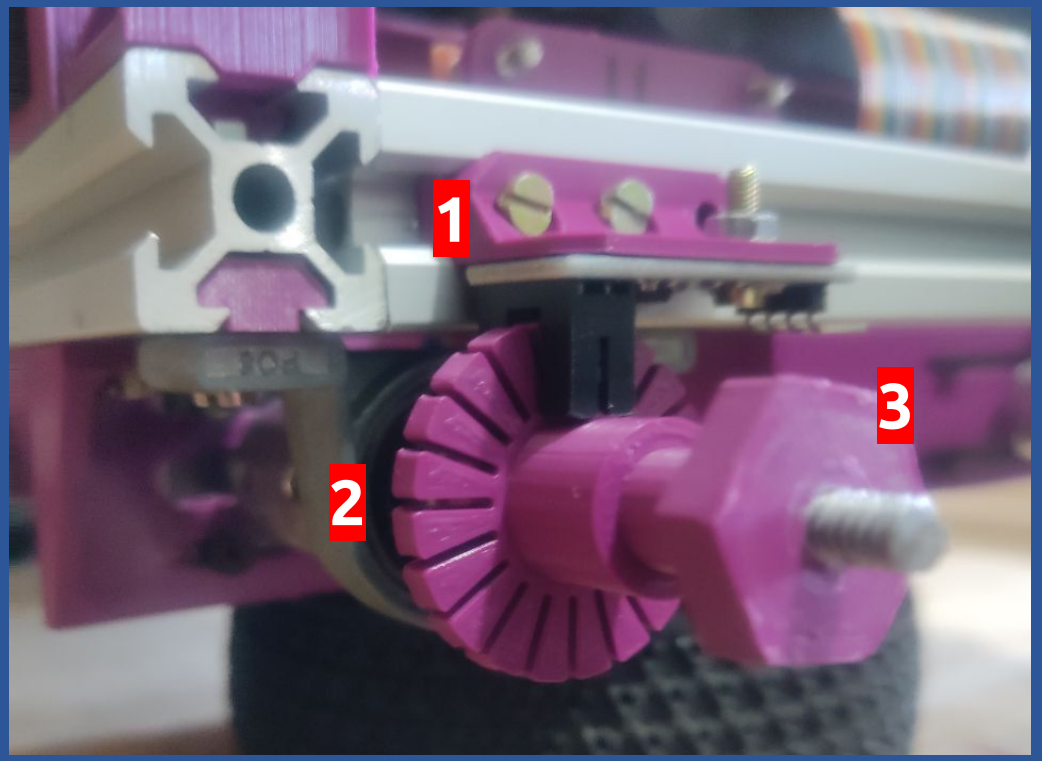


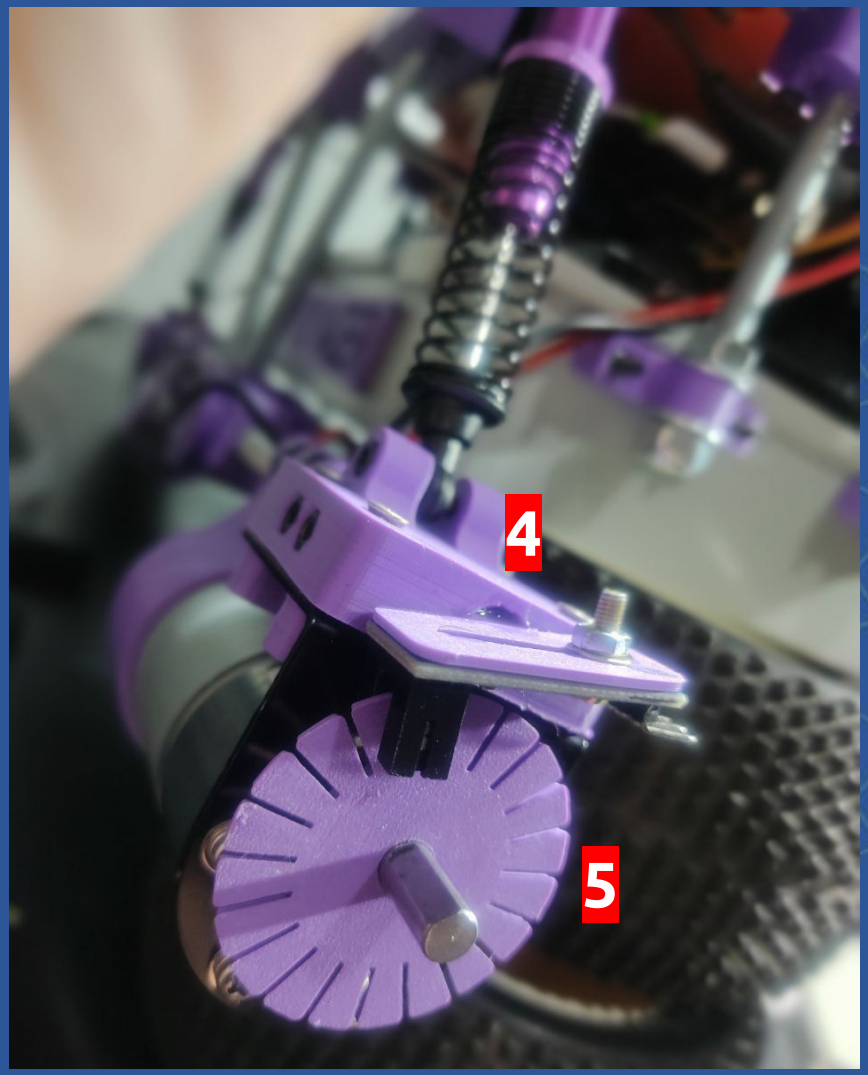
Suspensão



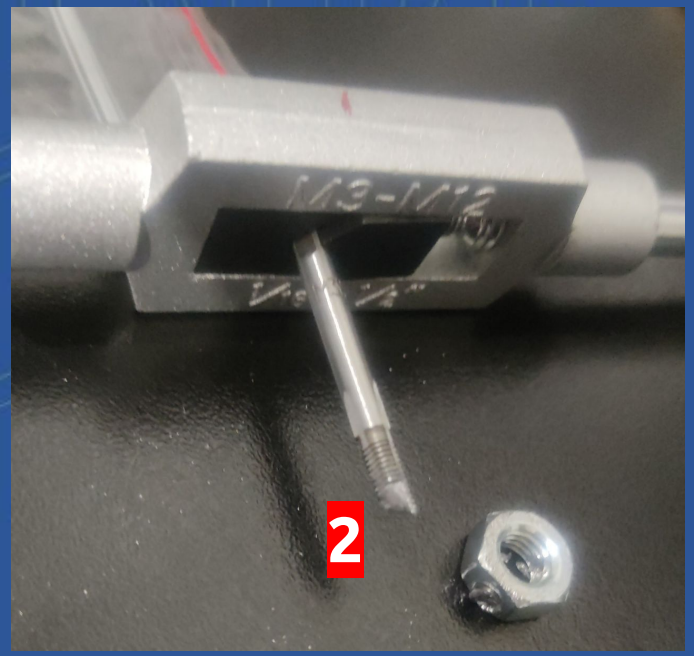


Encoder

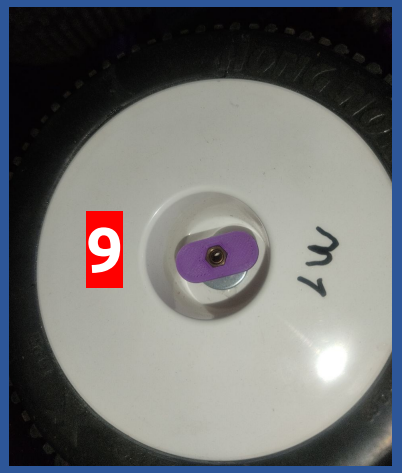




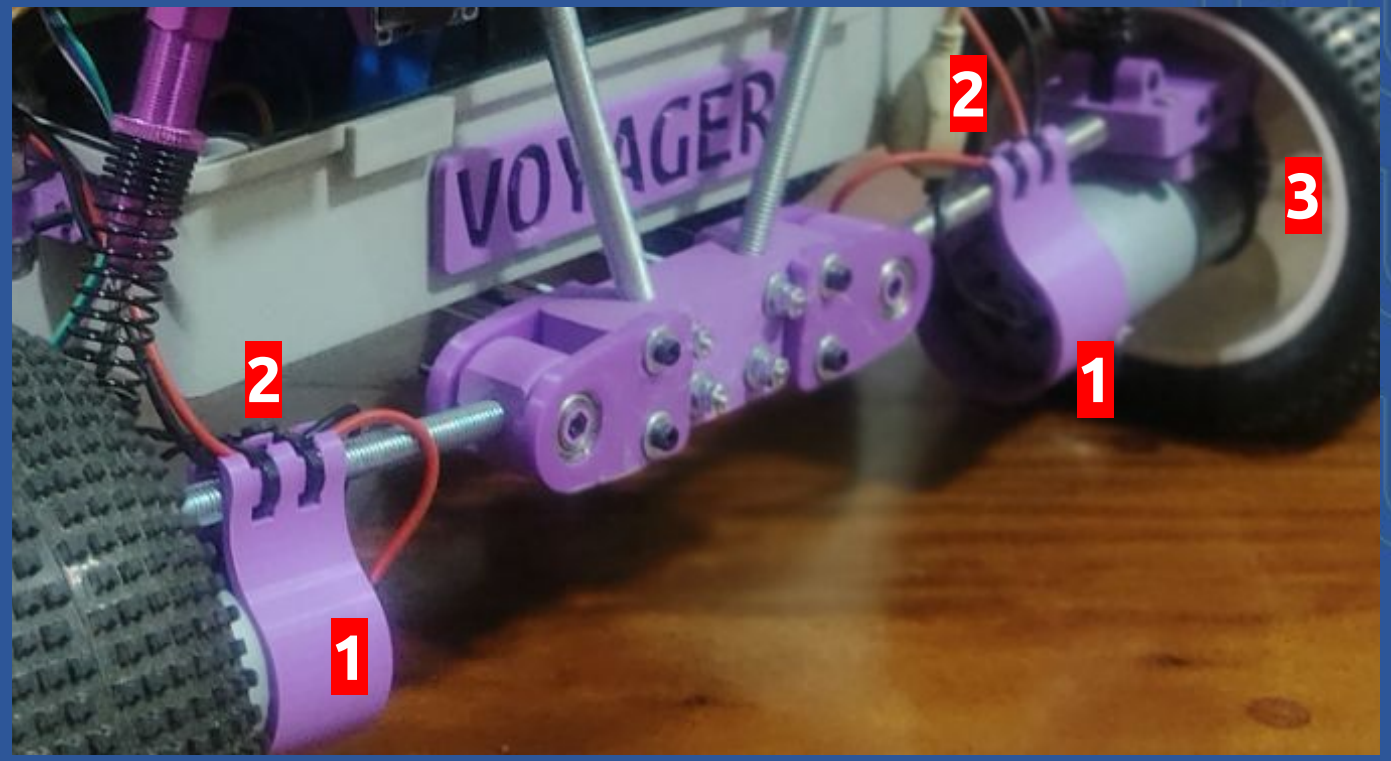
Motor - Eixo



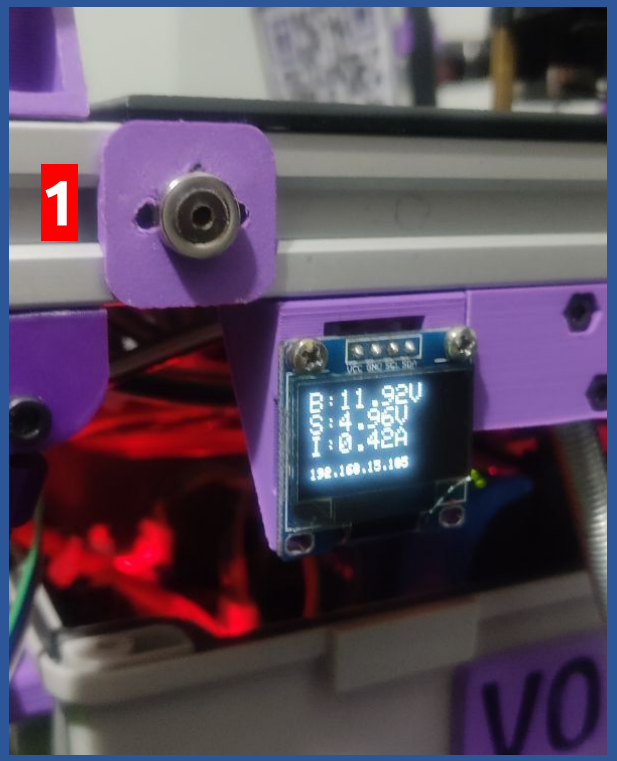
Motor - Eixo

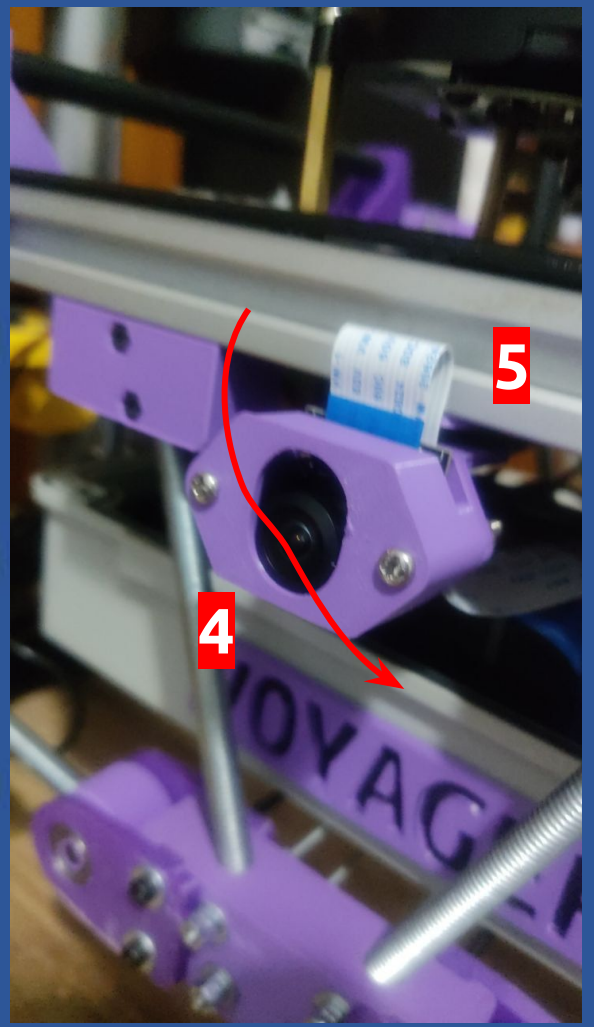


Motor - Escudo

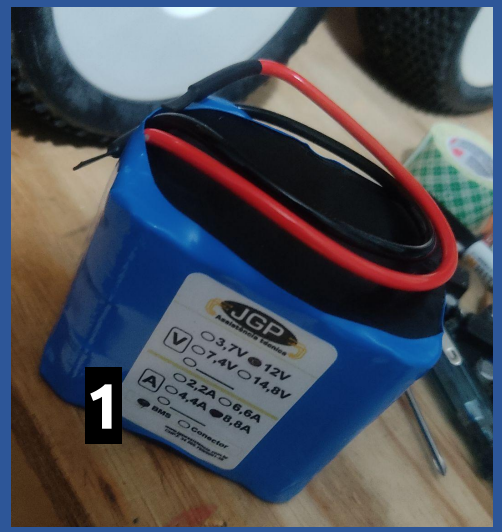


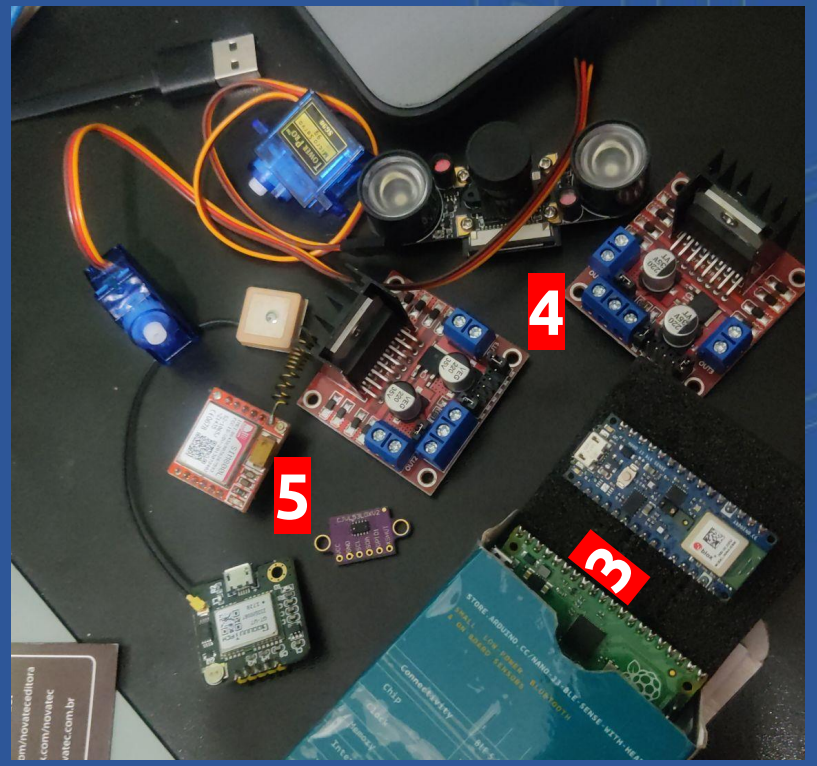
Engate Magnetico

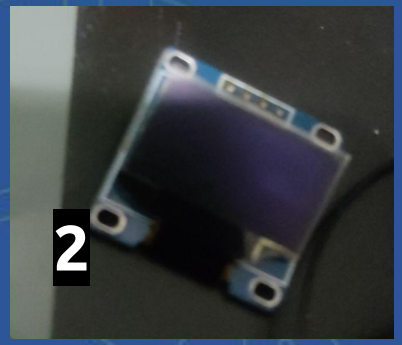




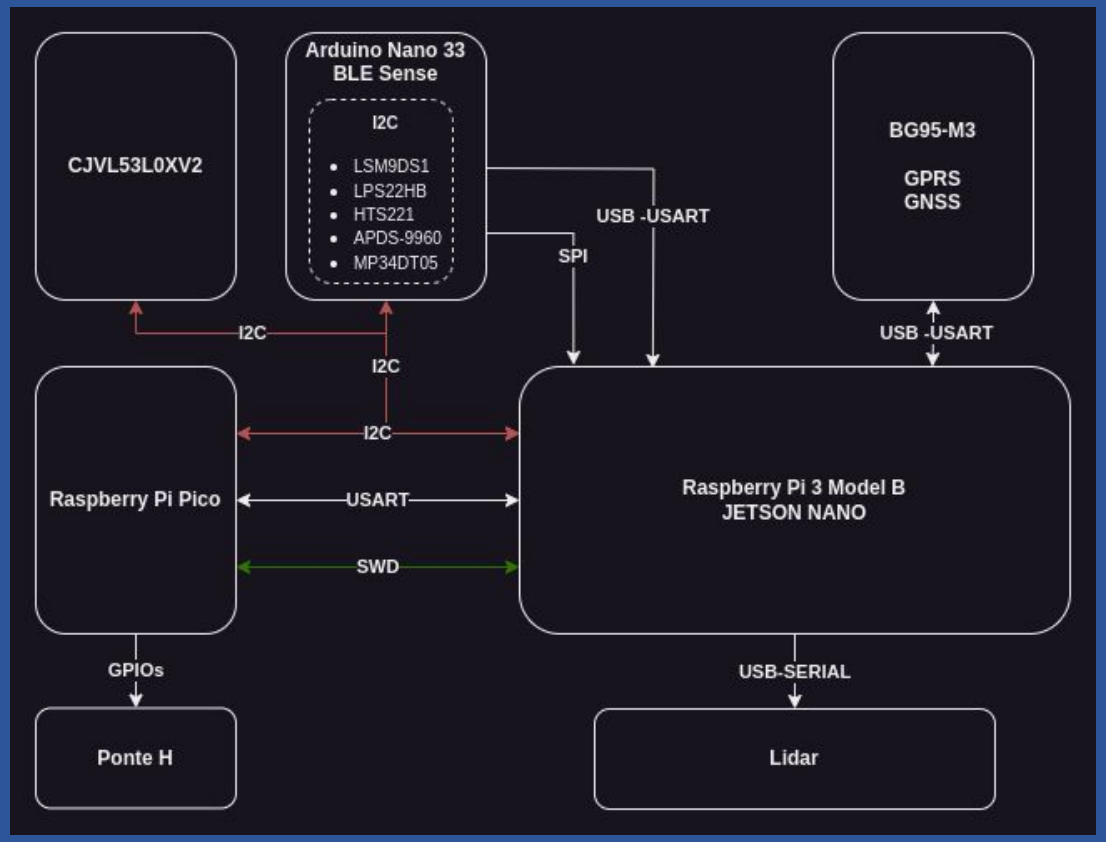
Eletrônica - Componentes

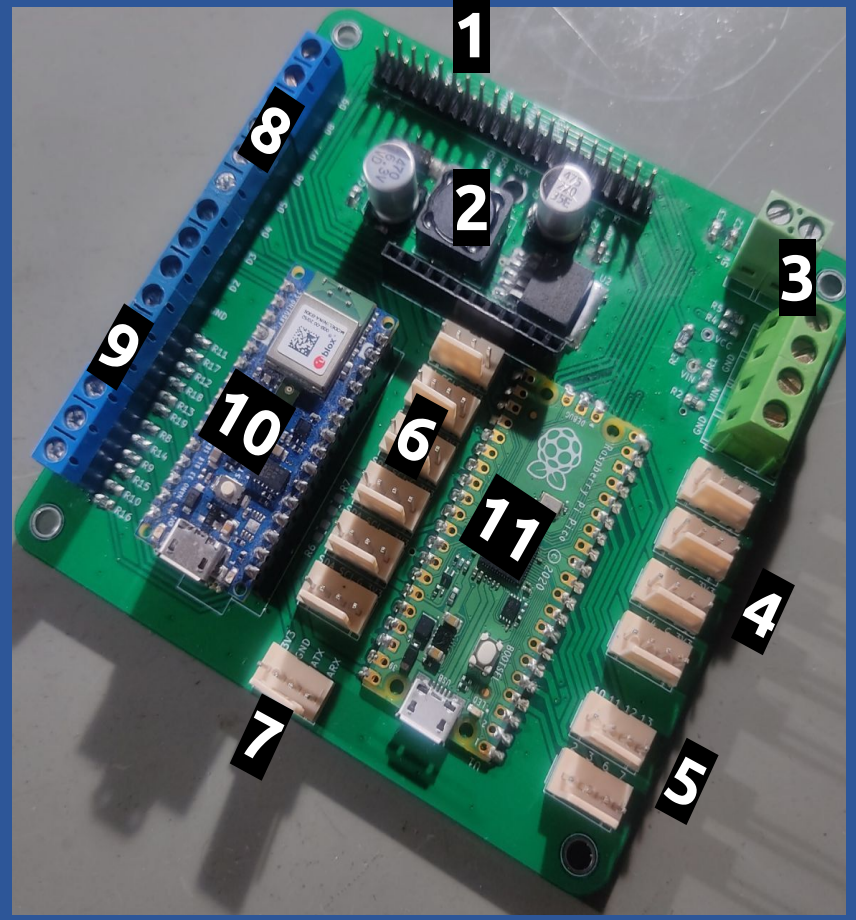




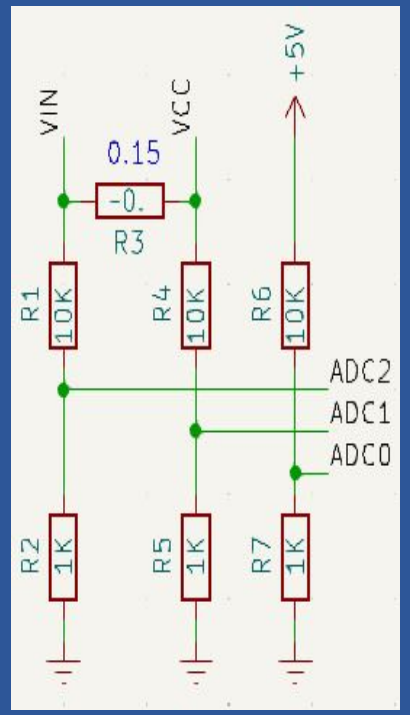


Eletrônica - Arquitetura



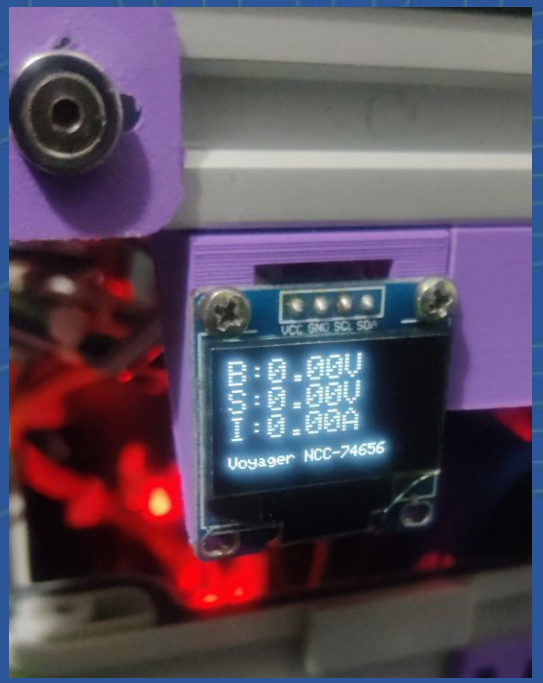


Eletrônica - Multímetro



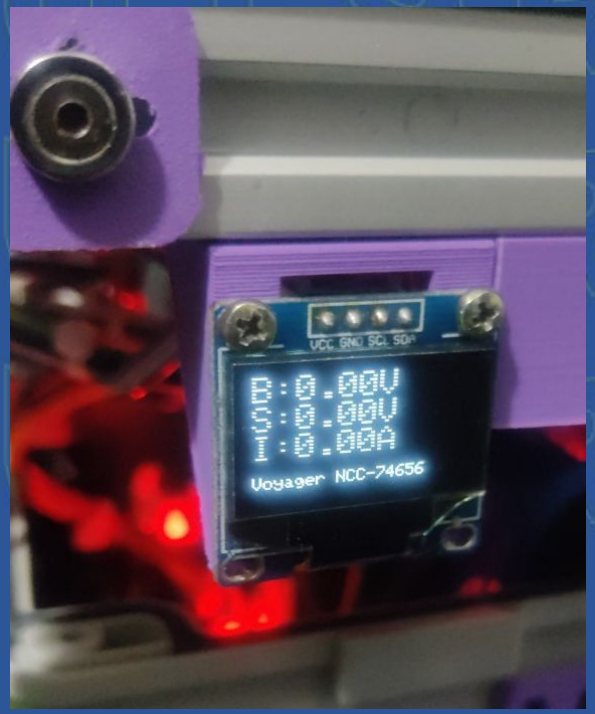

```
/* ************************************************************************
/* Configurações gerais da aplicação
/* ************************************************************************
#define CONFIG_QUEUE_ELEMENTS           ( 1                     )
#define CONFIG_DELAY_LOOP_MS            ( 40                    )
#define CONFIG_MOVING_AVERAGE           ( 24                    )
#define CONFIG_POWER_SUPPLY_OFFSET      ( 12.00-11.7            )
#define CONFIG_POWER_SYSTEM_OFFSET      ( 5.00 - 4.9            )
```

```
/* Setings ---------------------------------------------------------
/* Function prototype ----------------------------------------------
#define TO_VOLTAGE( BIT )     ( ( (float)BIT / 4096.0 ) * 36.3 )
/* Setup -----------------------------------------------------------
/* ----------------------------------------------------------------
```



Eletrônica - Multímetro

```
64  void loop( void ){
65
66          if(Serial.available() > 1){
67                  digitalWrite(LED1, HIGH);
68
69          1       delay(100);
70                  String sData = Serial.readString();
71                  sData.trim();
72
73                  if( !deserializeJson(jsonBuffer, sData) ){
74
75          2               float fVcc      = jsonBuffer["vcc"];
76                          float fVsys     = jsonBuffer["vsys"];
77                          float fCurrent  = jsonBuffer["current"];
78                          String msg      = jsonBuffer["msg"];
79                          msg.trim();
80
81                          display_show(fVcc, fVsys, fCurrent, msg);
82
83                  }                                       3
84                  digitalWrite(LED1, LOW);
85          }
86
87          delay(1);
88  }
```



Eletrônica - Motores

```
const char *pcBuffer = uart_pcGetBuffer( _idxUart0 );

if ( textp_bFindString( pcBuffer, "\n" ) == true ) {

        if( textp_bGetLabelInfo( pcBuffer, "mr1", 0, buffer ) ){
                velocities.motorRight1  = atoi(buffer);
        }

        if( textp_bGetLabelInfo( pcBuffer, "mr2", 0, buffer ) ){
                velocities.motorRight2  = atoi(buffer);
        }

        if( textp_bGetLabelInfo( pcBuffer, "ml1", 0, buffer ) ){
                velocities.motorLeft1   = atoi(buffer);
        }

        if( textp_bGetLabelInfo( pcBuffer, "ml2", 0, buffer ) ){
                velocities.motorLeft2   = atoi(buffer);
        }

        queue_add_blocking( &sendToCore1Queue, &velocities );

        uart_vCleanBuffer( _idxUart0 );
}


if( queue_try_remove( &sendToCore0Queue, &motors ) == true ){
        textp_puCleanBlk( (uint8_t*)buffer, CONFIG_BUFFER_SIZE );
        main_showInformation( &motors, buffer );
}
```

1

2

3

```
queue_try_add(&sendToCore0Queue, &motors);

if( queue_try_remove(&sendToCore1Queue, &velocities) == true ){
        // pids.motorRight1.fSetPoint   = velocities.motorRight1;
        // pids.motorRight2.fSetPoint   = velocities.motorRight2;
        // pids.motorLeft1.fSetPoint    = velocities.motorLeft1;
        // pids.motorLeft2.fSetPoint    = velocities.motorLeft2;
}

motor_vMove(&motors.right1,     velocities.motorRight1  );
motor_vMove(&motors.right2,     velocities.motorRight2  );
motor_vMove(&motors.left1,      velocities.motorLeft1   );
motor_vMove(&motors.left2,      velocities.motorLeft2   );


sleep_ms( CONFIG_PID_FRAME_HATE_MS );
```

4

5

6

...mr1=1024;...

Controle

```
31          FindEvent1 = True
32          while FindEvent1:
33              time.sleep(1)
34              dir_list = os.listdir(path)
35              print( "find devices between: " + str(dir_list) )
36              print( "----------------------" )
37              for elem in dir_list:
38                  try:
39                      gamepad = 'null'
40                      gamepad = InputDevice( path + elem )
41                      print( "check [ " +   path + elem + " ]--> " + str( gamepad ) )
42                  except:
43                      gamepad = 'null'
44                      print( "check [ " +   path + elem + " ]--> " + str( gamepad ) )
45                      pass
46
47                  if "Microsoft X-Box 360 pad" in str(gamepad):
48                      print("gameped connected!!")
49                      event = path + elem
50                      FindEvent1 = False
51                      break
52
53          print( gamepad.capabilities( verbose = True ) )
54
55          put_event('42', '42', '42')
56
57          for event in gamepad.read_loop():
58              if event.type == ecodes.EV_ABS:
59                  absevent = categorize(event)
60                  put_event(_type=absevent.event.type, _code=absevent.event.code, _value=absevent.event.value)
61              else:
62                  put_event(_type=event.type, _code=event.code, _value=event.value)
63              time.sleep(0.02)
64
```

1

3

4

Lidar

```
while True:
        time.sleep(3)
        try:
                # setting server
                UDP_IP = '127.0.0.1'
                UDP_PORT = 42421
                sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

                # get serial port
                ydlidar.os_init();
                ports = ydlidar.lidarPortList();
                port = "/dev/ydlidar";
                for key, value in ports.items():
                        port = value;
                        print(port);

                # setting YDlidar X3
                laser = ydlidar.CYdLidar();
                laser.setlidaropt(ydlidar.LidarPropSerialPort, port);
                laser.setlidaropt(ydlidar.LidarPropSerialBaudrate, 115200);
                laser.setlidaropt(ydlidar.LidarPropLidarType, ydlidar.TYPE_TRIANGLE);
                laser.setlidaropt(ydlidar.LidarPropDeviceType, ydlidar.YDLIDAR_TYPE_SERIAL);
                laser.setlidaropt(ydlidar.LidarPropScanFrequency, 10.0);
                laser.setlidaropt(ydlidar.LidarPropSampleRate, 3);
                laser.setlidaropt(ydlidar.LidarPropSingleChannel, True);
                laser.setlidaropt(ydlidar.LidarPropMaxAngle, 180.0);
                laser.setlidaropt(ydlidar.LidarPropMinAngle, -180.0);
                laser.setlidaropt(ydlidar.LidarPropMaxRange, 8.0);
                laser.setlidaropt(ydlidar.LidarPropMinRange, 0.08);
                laser.setlidaropt(ydlidar.LidarPropIntenstiy, False);
                scan = ydlidar.LaserScan()

                ret = laser.initialize();
                if ret:
                        ret = laser.turnOn();
                        if ret:
                                while True:
                                        payload = scaning()
                                        sock.sendto(payload , (UDP_IP, UDP_PORT))
                laser.turnOff();
                laser.disconnecting();
```

1

2

3

Arquitetura de comunicação

```
[Unit]
Description=application for aquisittion of ydlidarx3.

Want=network.target
After=syslog.target network-online.target


[Service]
Type=simple
ExecStart=/usr/local/bin/ydlidarx3Server
Restart=on-failure
RestartSec=1
KillMode=process

[Install]
WantedBy=multi-user.target
```


```
[Unit]
Description=application for aquisittion of ydlidarx3.

Want=network.target
After=syslog.target network-online.target


[Service]
Type=simple
ExecStart=/usr/local/bin/ydlidarx3Server
Restart=on-failure
RestartSec=1
KillMode=process

[Install]
WantedBy=multi-user.target
```


systemd

Arquitetura de comunicação

USB-UART

Lidar

UDP-JSON

Node-RED

SENSE 33

Node-Red

USB-UART JSON

USB-RF

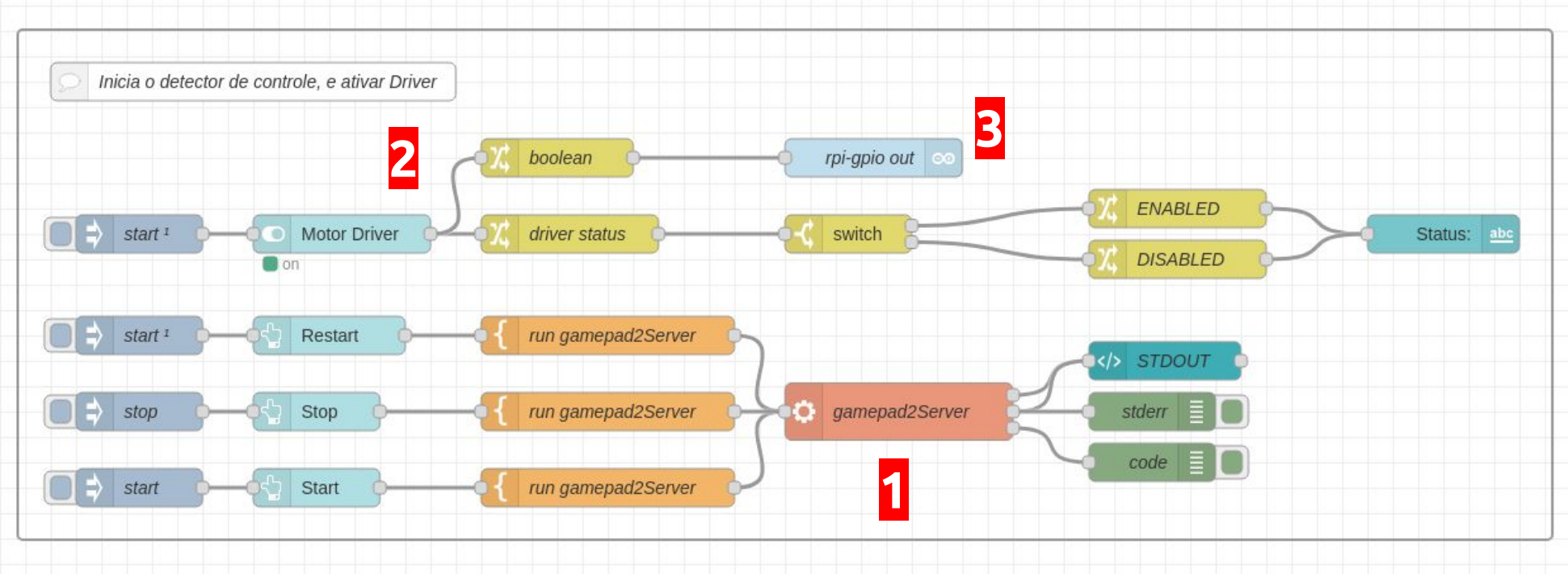
Controle

UDP-JSON

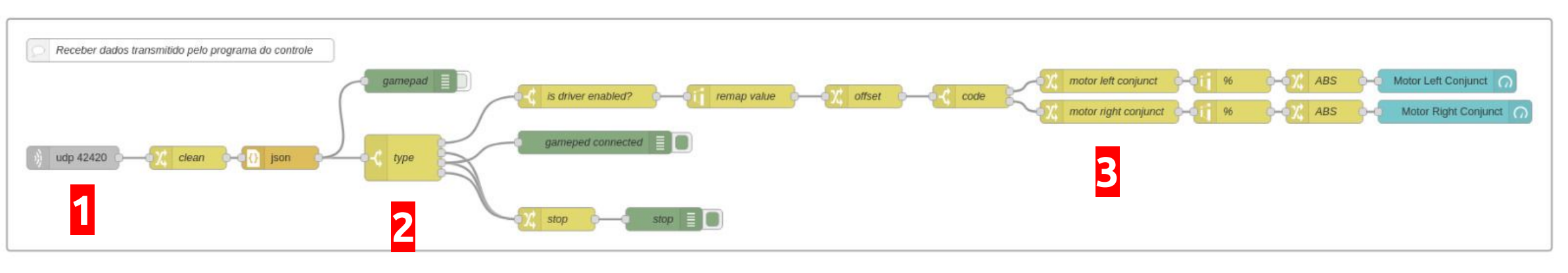
UART

RPI-PICO

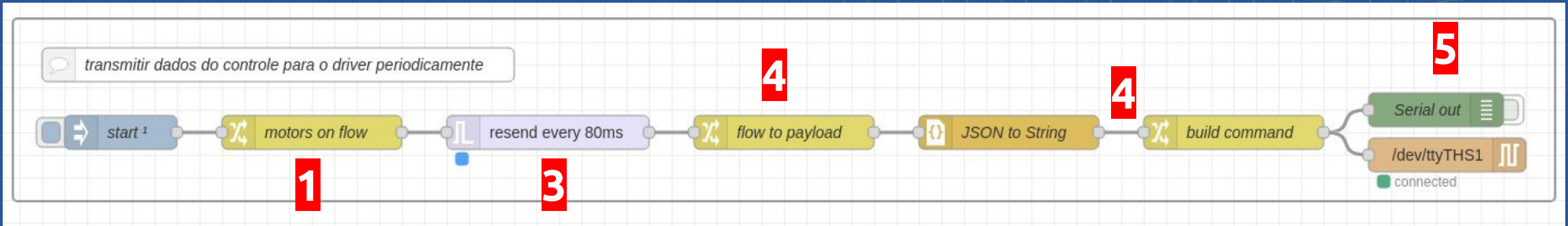
Node-RED - Controle e Driver



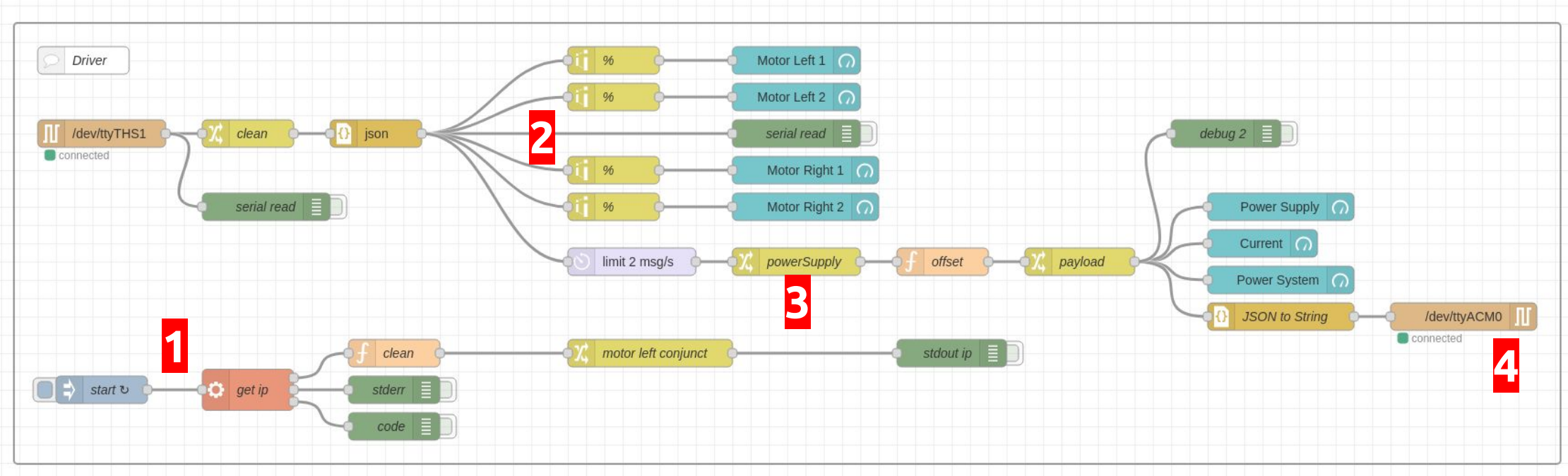
Node-RED - Controle e Driver



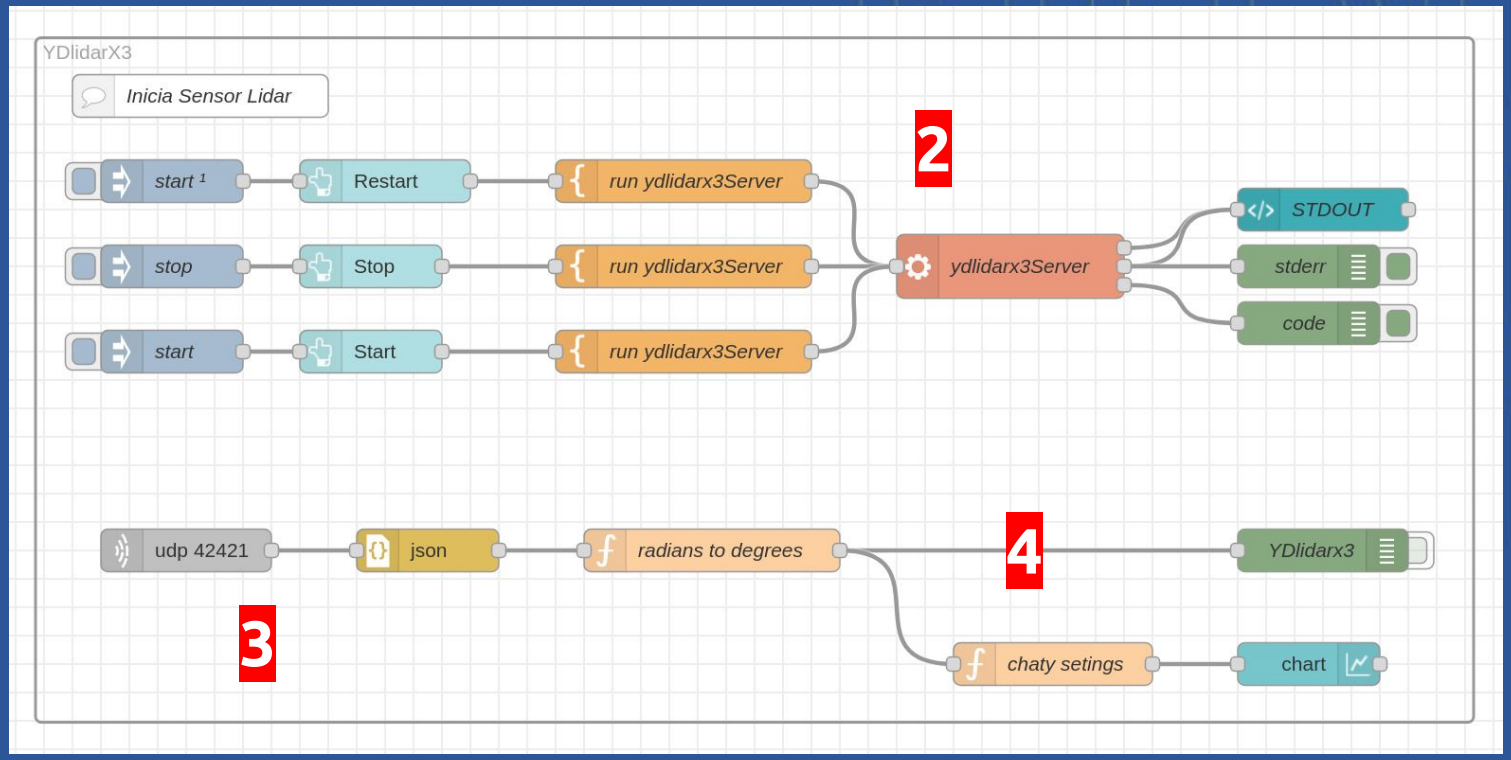
Node-RED - Controle e Driver



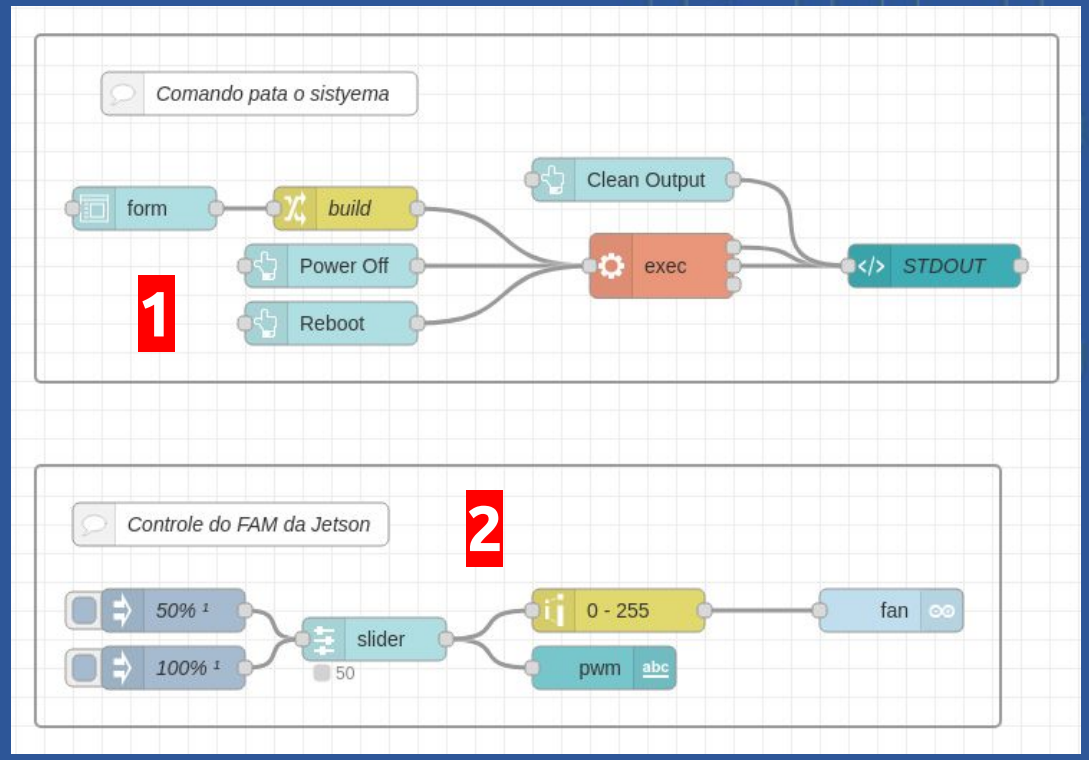
Node-RED - Controle e Driver



Node-RED - Lidar



Sistema

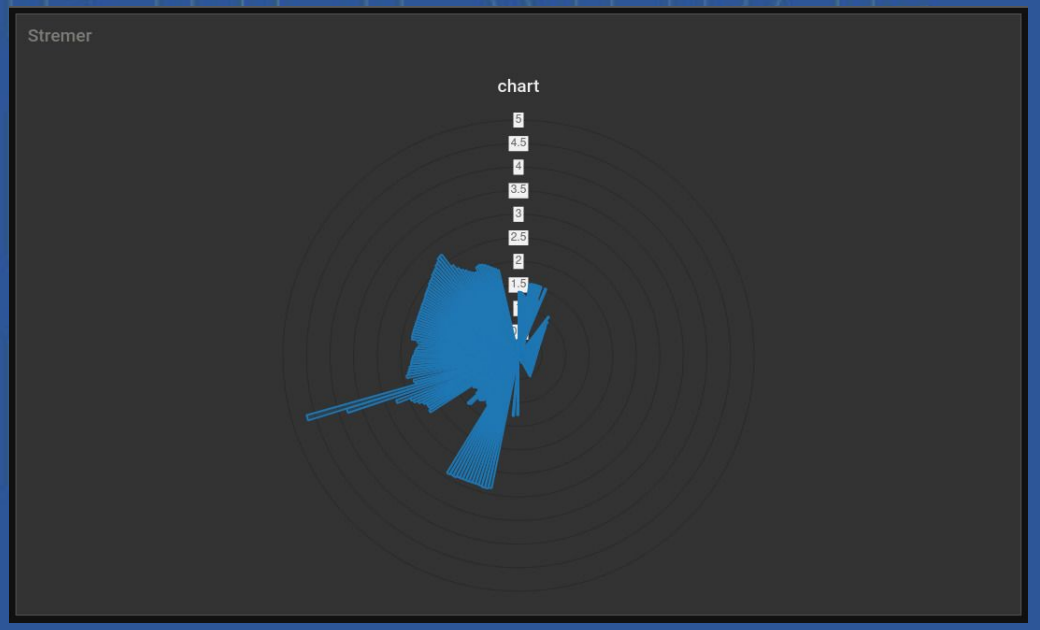

Comando pata o sistyema
form
build
Power Off
Reboot
1
Clean Output
exec
STDOUT
Controle do FAM da Jetson
2
50% ¹
100% ¹
slider
50
0 - 255
pwm
abc
fan

Supervisão

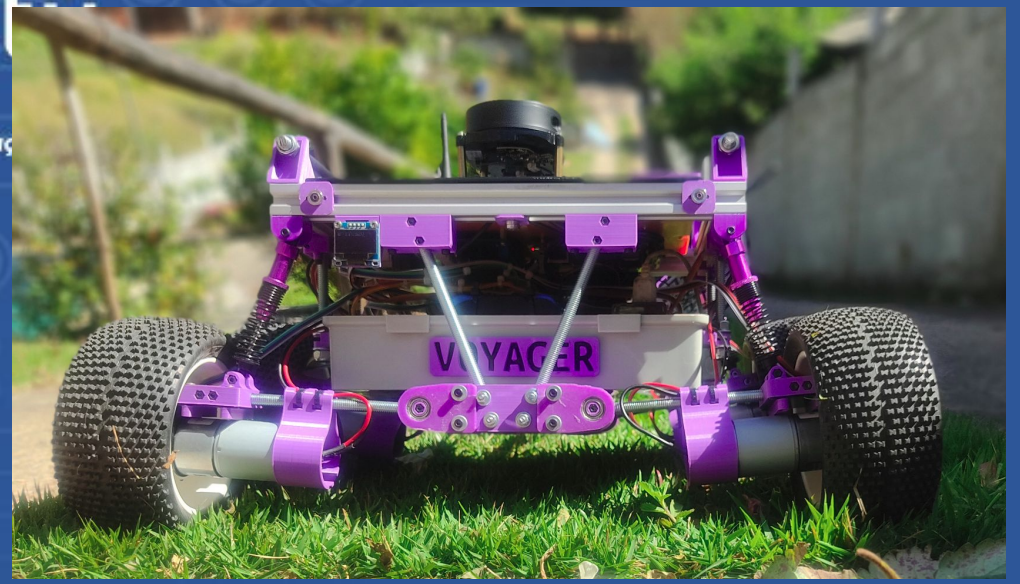

Terminal
Command
EXEC
CLEAN
STDOUT:
CLEAN OUTPUT
POWER OFF
REBOOT
Gamepad
START
RESTART
STOP
STDOUT:
Lidar
START
RESTART
STOP
STDOUT:
FAN
pwm
50

Supervisão


Driver
Motor Driver
Status:
ENABLED
Motor Left Conjunct
Motor Right Conjunct
Motor Left Power
Motor Left 1
Motor Left 2
Motor Right Power
Motor Right 1
Motor Right 2
Power
Power System
4.94V
Power Supply
11.91V
Current
0.5A


Stremer
chart

Resultados!



Grato pela atenção **XD**

Alguma dúvida ?

pibscontato@gmail.com @pedro_ibs

SCAN ME