FEEC - Faculdade de Engenharia Elétrica e de Computação

EA075 – INTRODUÇÃO AO PROJETO DE SISTEMAS EMBARCADOS

Projeto: Veículo Auto Guiado (AGV)

Grupo 3

Integrantes	**RA's**
André de Almeida Pinto	*164022*
Daniel Herbert Barbosa	*166221*
Davi Pincinato	*157810*
Diogo Melo de Abreu	*166610*
Matheus Freire Rodrigues	*174533*

1) Descrição funcional do seu AGV

Com o advento da internet, o uso de documentos digitais e e-mails se massificou de maneira praticamente instantânea, e as empresas têm cada vez menos utilizado documentos em papel em seu dia-a-dia. Sendo assim, a profissão de Office Boy em que uma das principais atribuições era prover a circulação desses documentos têm se tornado cada vez mais inexistente. Por outro lado, para determinados documentos e correspondências, se faz necessária a utilização de vias físicas, em papel. Nesse contexto, uma aplicação de robótica de um mecanismo que facilite essa troca pode facilitar o dia-a-dia do escritório de determinadas empresas.

Sendo assim, propomos o desenvolvimento de um Veículo Auto Guiado (AGV - *Automated Guided Vehicle*) para promover a troca de documentos impressos entre as estações de trabalho. Este consiste em um carrinho que irá passar por todas as estações de trabalho e realizar a coleta e entrega dos papéis para determinado colaborador ou área.

Na estrutura física do carro, teremos compartimentos individuais, permitindo assim que possa-se identificar a quem se destina determinada documentação. A rota a ser percorrida é indicada através de linhas no piso, sendo assim essa leitura será realizada através de sensores de infravermelho, comuns nesse tipo de aplicação. Uma representação do uso deste projeto pode ser vista na Figura 1. Como medida de segurança, o carro será provido de identificação de obstáculos, implementada com sensores ultrassom.

Na figura representada abaixo temos uma representação de uma aplicação para o sistema, onde temos seis estações de trabalho em uma configuração que permite o uso de três pontos de parada, ou seja, em cada parada dois funcionários podem recolher ou adicionar os documentos desejados, ao chegar no final, na terceira estação o carrinho inverte o sentido de movimentação e realiza o percurso inverso e assim sucessivamente.

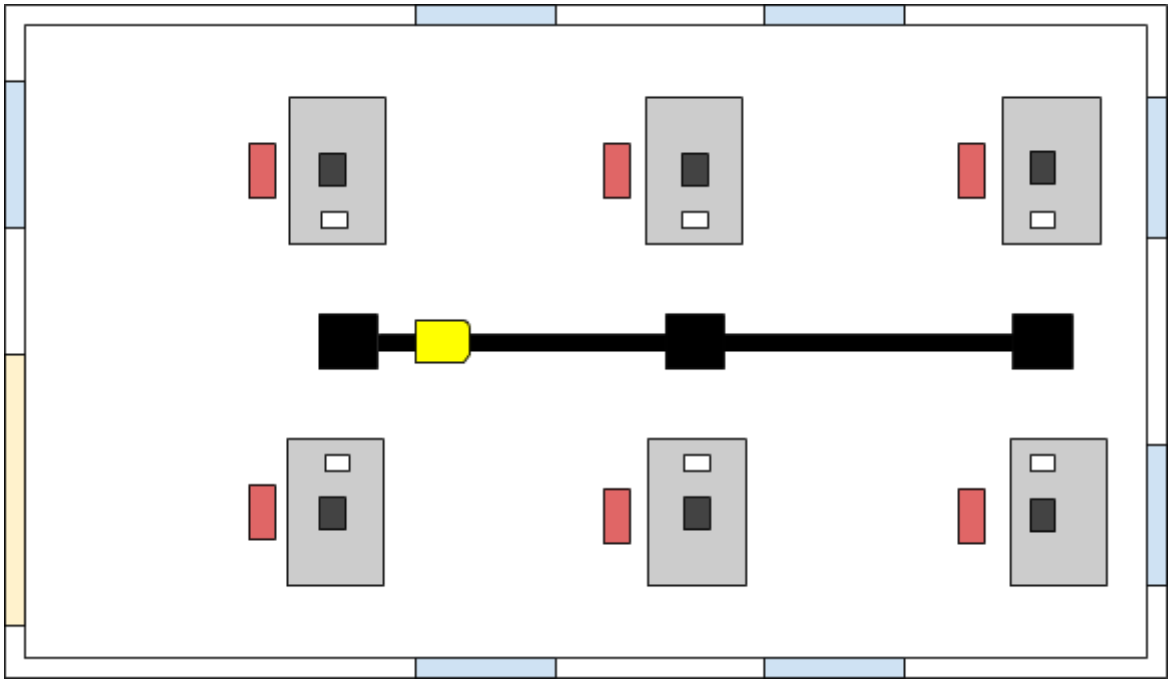


Figura 1: Representação de uma aplicação real para o uso do AGV projetado.

Como meio de indicação visual, teremos dois LEDs com a função de indicar o sentido de movimento, permitindo assim que o colaborador saiba de antemão se a próxima estação será a da frente ou a de trás. Adicionalmente, com o intuito de saber se foi adicionado um novo documento ou não, iremos implementar uma célula de carga que identifica variações de peso na bandeja de documentos.

Como muitas das vezes o sentido de movimento do carro pode não ser o desejado, adicionamos um botão que dá ao usuário a opção de inverter o sentido de movimento original, fornecendo assim mais flexibilidade ao sistema. E por fim, sempre que desejar-se mais tempo para adicionar documentos, o colaborador terá a opção de adicionar mais tempo à espera do carrinho. Isso será feito através de um botão, que cada vez que for acionado adiciona trinta segundos ao tempo de espera.

Uma vez definidas as funções a serem implementadas, podemos entender melhor a dinâmica de funcionamento do sistema, como segue: o carro está no estado parado em determinada estação, ele permanecerá por padrão por trinta segundos nessa mesa. Em seguida temos algumas opções: Se for solicitado mais tempo, a contagem dos trinta segundos é reiniciada; se for adicionada carga veículo aguarda dez segundos para seguir para a próxima estação; se for apertado o botão de inverter o movimento, ele irá para o estado trás e o sentido de movimentação

será invertido em relação ao original. Caso contrário, depois que se encerrar o tempo ele seguirá para o estado frente. Se durante o deslocamento entre as estações algum obstáculo for identificado, o carro interrompe o movimento e só irá prosseguir com a remoção do impedimento.

2) Componentes utilizados

Arduino UNO R3:

Para a implementação deste projeto, será utilizado um Arduino como plataforma de prototipagem eletrônica de hardware livre. Tal escolha deve-se ao fato deste dispositivo ser de baixo custo, funcional, programável em linguagem C/C++ e de possuir uma vasta documentação online disponível bem como uma gama de componentes, como sensores e módulos, compatíveis com essa plataforma. Além disso, consideramos esta como a melhor escolha devido à quantidade de portas necessárias para o projeto, que não seria atendida com um Arduino Nano, por exemplo, e também seria super dimensionado com um Arduino Mega.

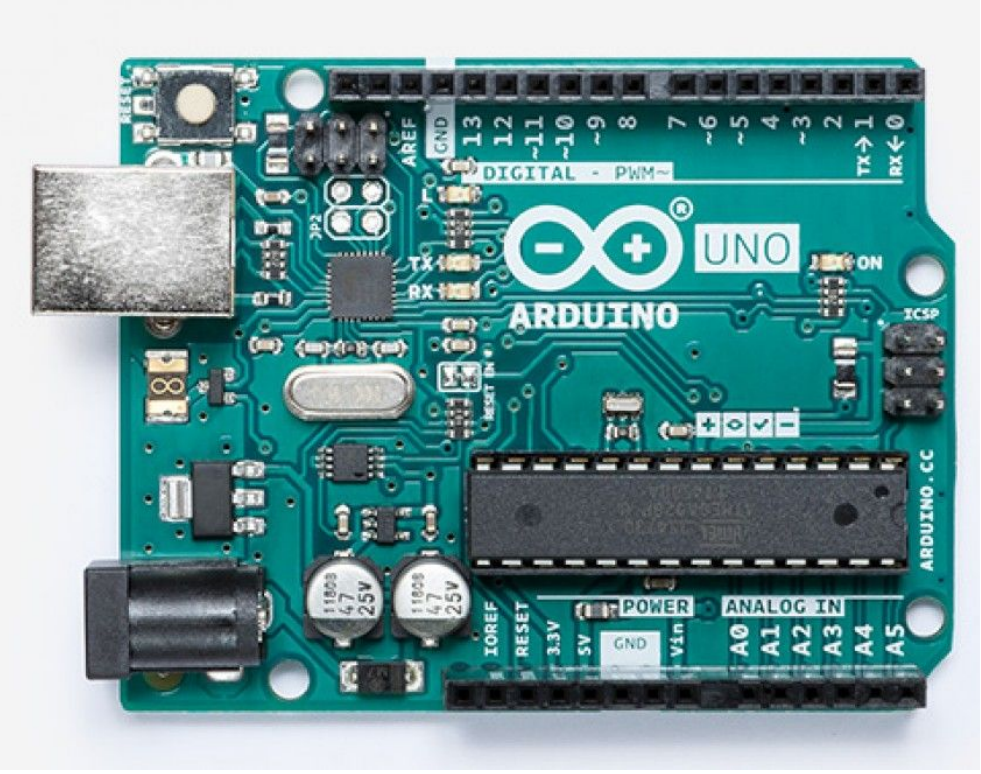


Figura 2: Plataforma de prototipagem utilizada no projeto - Arduino UNO R3

O Arduino UNO R3 é uma placa baseada no microprocessador ATmega328P - microcontrolador de oito bits. Ele Possui 14 portas digitais de entrada/saída (das quais 13 serão utilizadas neste projeto), 6 entradas analógicas, um cristal de quartzo de 16MHz, fácil conexão com computadores por meio de comunicação USB para se comunicar com a sua IDE (Ambiente de Desenvolvimento Integrado) que também é open source e utiliza uma linguagem baseada em C/C++ para fazer a programação do Arduino.

As principais características do Arduino UNO R3 são:

Microcontrolador: ATmega328 (Datasheet ATmega328);
Conversor USB/Serial: ATmega16U2;
Velocidade do Clock: 16 MHz;
Memória ROM: 1 Kb (ATmega328);
Memória SRAM: 2 Kb (ATmega328);
Memória Flash: 32 Kb (0,5 Kb usado pelo Bootloader);
Tensão de Alimentação: 7 à 12 Vdc (Conector Jack e pino Vin);
Tensão de Operação: 5 Vdc;
Tensão de Nível Lógico: 5,0 Vdc (Tolera 3,3 Vdc);
Interfaces: UART(1 canal), SPI (1 canal), I2C (1 canal);
Tipos GPIO: Pinos digitais I/O (14), pinos analógicos 10-Bits (6 canais), pinos PWM (6 canais);
Temperatura de trabalho: -40° à +85° C;

HC-SR04: Módulo Sensor de Distância Ultrassônico

Figura 3: Sensor Ultrassônico de distância - HC-SR04.

Para a identificação de possíveis obstáculos (como pessoas ou objetos) que possam obstruir o caminho percorrido pelo o AGV, será utilizado o módulo de Sensor Ultrassônico de distância - HC-SR04. Este pode efetuar a medição de distâncias entre 2cm a 4m com uma precisão de 3mm. Além disso, ele poussui um baixo custo de implementação (cerca de R$10,00 o módulo). Possui um circuito pronto com emissor e receptor acoplados e 4 pinos (VCC, Trigger, ECHO, GND) para medição. Neste projeto, serão utilizados dois módulos HC-SR04, que irão monitorar o caminho a ser percorrido, na ida e na volta (conectados ao Arduino através das portas 2, 3, 4 e 5).

- Funcionamento:

Para a etapa de configuração e implementação, será necessário alimentar o módulo (VCC) e colocar o pino Trigger em nível alto por mais de 10us. Assim, o sensor emitirá uma onda sonora que, ao encontrar um obstáculo, rebaterá de volta em direção ao módulo. Durante o tempo de emissão e recebimento do sinal, o pino ECHO ficará em nível alto. Logo, o cálculo da distância pode ser feito de acordo com o tempo em que o pino ECHO permaneceu em nível alto após o pino Trigger ter sido colocado em nível alto.

$$Distância\ medida = \frac{[Tempo\ ECHO\ em\ nível\ alto * Velocidade\ do\ som]}{2}\ \text{[m/s]}$$

Onde a velocidade do som pode ser considerada idealmente igual a 340 m/s. A divisão por 2 na equação se dá pelo fato da onda propagada percorrer duas vezes a distância medida, uma no sentido do sensor ao objeto e outra no sentido do objeto ao sensor.

- Pinagem:

Vcc: Deve ser conectado ao pino 5V
Trig: Deve ser conectado a um pino digital configurado como saída.
Echo: Deve ser conectado a um pino digital configurado como entrada.
Gnd: Deve ser conectado a um pino GND.

- Especificações do produto:

Tensão de Alimentação: 5V DC
Corrente consumida: 15mA
Frequência de operação: 40kHz
Distância máxima (medida): 4m
Distância mínima (medida): 2cm
Ângulo de medição: 15º
Sinal de entrada [Trigger]: Pulso TTL (5V) de 10us
Sinal de saída [Echo]: Pulso TTL (5V) proporcional à distância detectada

Sensores Seguidores de Linha:

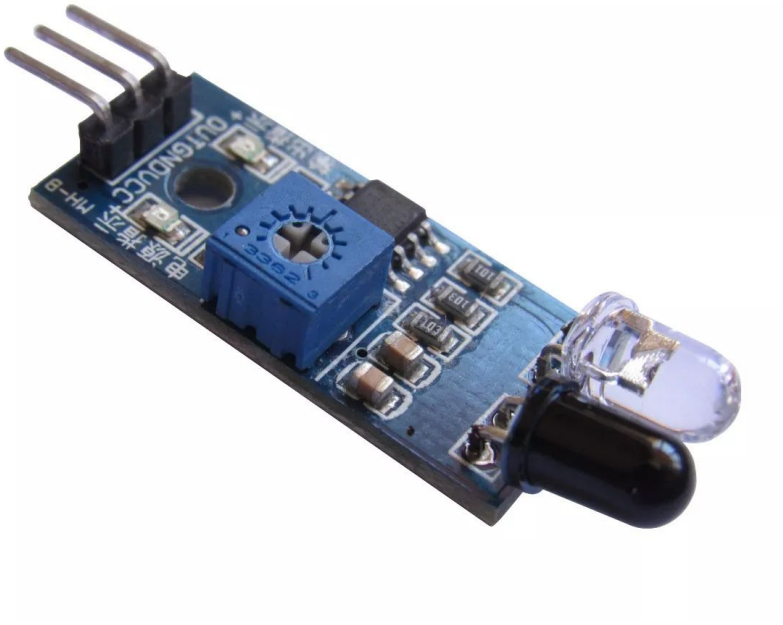


Figura 4: Sensor infravermelho seguidor de linha.

Para o monitoramento do movimento do AGV, garantindo que ele se mova sempre sobre um caminho pré determinado, serão utilizados três sensores infravermelhos, dos quais dois serão responsáveis pelo monitoramento do caminho percorrido pelo veículo e identificar as estações de trabalho onde o AGV irá parar temporariamente. O terceiro será responsável por identificar o final do trajeto, mostrando que o AGV deve-se mover na direção oposta.

Este sensor tem a capacidade de identificar o caminho a ser seguido a partir de uma linha que pode ser na cor branca (neste caso o chão deverá ser preto) ou na cor preta (neste caso o chão deverá ser branco). A base de funcionamento do Sensor Seguidor de Linha TCRT5000 é um sensor reflexivo TCRT500, que é um sensor infravermelho (emissor e receptor) que funciona a partir da reflexão de sinais. Esse sensor é digital, sendo que o valor de tensão obtido do sensor reflexivo TCRT500 é comparado com uma tensão de referência a qual é ajustável por meio de um potenciômetro. A conexão dos sensores ao Arduino serão através das portas 8, 9 e 10.

- Especificações do produto:

Controlador: LM393

Tensão de operação: 3,3 – 5Vdc (no caso deste projeto, 5Vdc)

Saída Digital e Analógica

LED indicador de sensor ativado

LED indicador de tensão no sensor

Sensibilidade ajustável através de trimpot

Botões de Comando

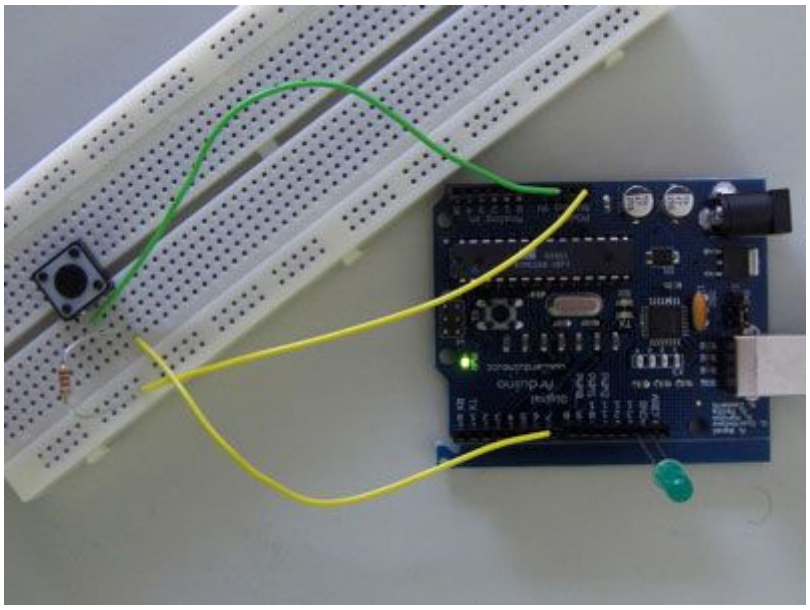


Figura 5: Botão de comando, conectado ao Arduino UNO.

A Chave Táctil / Push Button como também é conhecido, é um dos componentes eletrônicos mais utilizados para prototipagem de projetos. Esta chave é um tipo de interruptor pulsador (conduz somente quando está pressionado).

Serão utilizados neste projetos dois *push buttons.* Um deles, quando acionado, será responsável por fornecer ao usuário a opção de inverter o sentido de movimento original, fornecendo assim mais flexibilidade ao sistema. O outro, será responsável por adicionar mais tempo de espera do AGV em uma determinada posição. Cada vez que acionado, botão “adicionará” trinta segundos ao tempo de espera.

- Especificações do produto:

Tipo de chave: táctil

Tensão máxima: 12VDC (no projeto, será utilizada a porta VCC - 5Vdc do Arduino UNO R3)

Corrente máxima: 50mA

- Conexão:

Serão conectados três fios à placa Arduino. O primeiro vai de uma perna do botão através de um resistor de pull-up (2,2 kΩ) até a alimentação de 5 volts. O segundo vai da perna correspondente do botão para o terra. O terceiro se conecta a um pino de E / S digital (sendo o botão 1 conectado à porta D6 do Arduino UNO R3 e o outro à porta D7), que lê o estado do botão.

L298 - Acionamento dos motores:

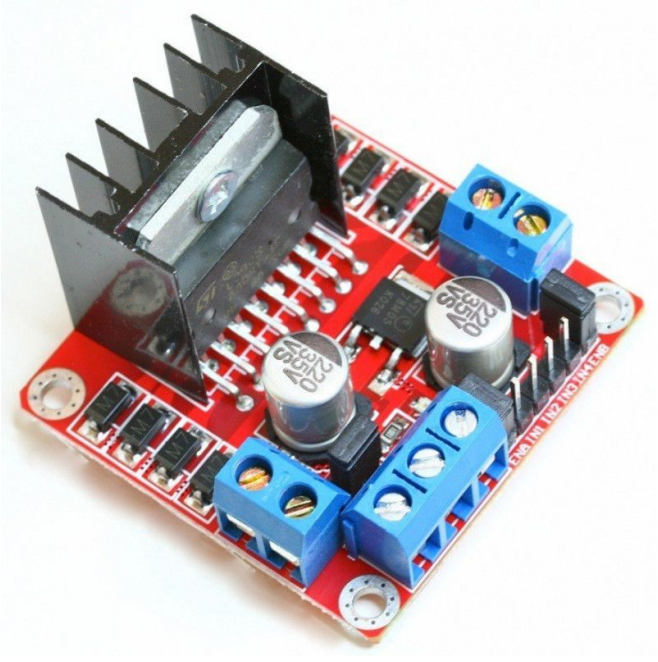


Figura 6: Ponte H L298N.

Para o acionamento e o controle da velocidade dos dois motores deste projeto, será utilizado um módulo L298. Neste módulo, cada ponte H possui um pino que ativa ou não a ponte H. Caso tenha um sinal de 5V inserido nele, a ponte estará ligada, caso seja 0V a ponte estará desligada. Como temos 2 pontes H, temos o Enable A (Ativa A) e o Enable B (Ativa B), um para cada motor. Como os dois motores estarão no mesmo eixo do AGV, para a conexão deste módulo ao Arduino, será utilizado o pino D11 da placa - com o PWM deste pino, será feito o controle de velocidade ao veículo. Nele estarão ligados tanto o Enable A quanto o Enable B. Para as entradas do módulo, pelo mesmo motivo, será utilizada somente a porta D12 do Arduino. Nas saídas do módulo estarão conectados os motores DC.

- Especificações do produto:

CI controlador: L298N
2 Canais independentes
Tensão de Operação: 5 - 35V
Corrente de Operação máxima: 2A por canal
Potência Máxima: 25W
Tensão lógica: 5V
Corrente lógica 0-36mA

LEDs de sinalização

Figura 7: LEDs de sinalização.

Para indicação visual, serão utilizados dois LEDs com a função de indicar o sentido de movimento, permitindo assim que o colaborador saiba de antemão se a próxima estação será a da frente ou a de trás. A conexão do LED ao Arduino deverá ser feita considerando o sentido da corrente que percorrerá o LED, entrando pelo terminal Anodo (redondo) e saindo pelo terminal Catodo (Achatado), conforme exemplo abaixo:

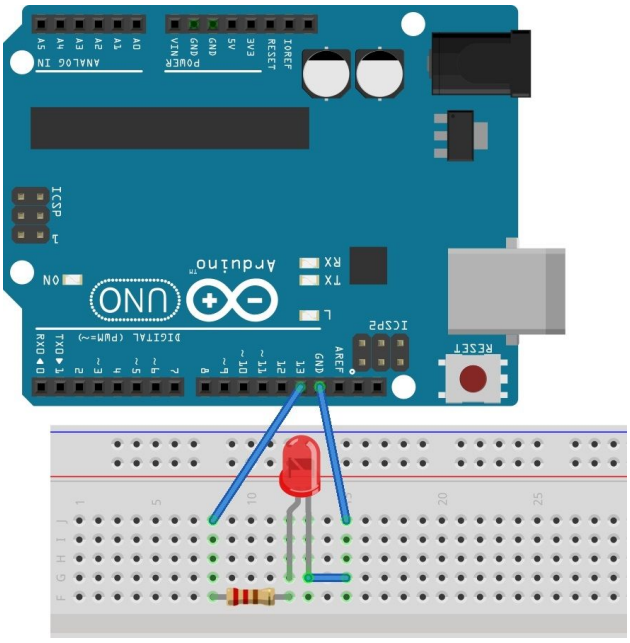


Figura 8: Conexão de um LED ao Arduino.

- Especificações do produto:

Tamanho: 5 mm

Tensão: 2.0 ~ 2.21V

Corrente: 25mA

Intensidade luminosa: 1000 MCD

Vida útil : 50,000 Horas

Ângulo de abertura: 120º graus.

Devido à tensão de operação dos LEDs, será necessária a utilização de um resistor em série à cada LED. Pela primeira Lei de Kirchoff, verifica-se que pode ser utilizado um resistor de 220Ω para cada LED.

- Conexão:

A conexão do LED 01 será feira entre o pino 0 e GND do Arduino, e do LED 02 entre o pino 1 e o GND.

Motores

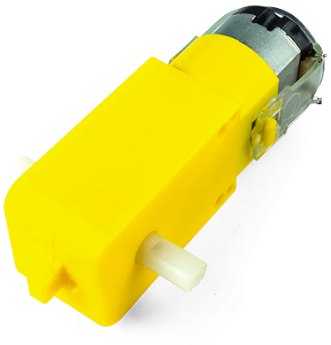


Figura 9: Motor DC 3-6V com Caixa de Redução e Eixo Duplo.

Serão utilizados dois motores DC, que deverão ser conectados às rodas do AGV. Esses, por sua vez, serão conectados às saídas do módulo L298, já descrito anteriormente. Cada motor conta com uma caixa de redução reta. O conjunto possui um eixo de saída de 9mm.

- Especificações do produto:

Tensão de alimentação recomendada: 4,5V
Corrente em aberto: 190mA
Corrente com máxima carga: 250mA
Corrente de Stall: 1.2A @ 6V (0.6A @ 3V)
Relação da caixa de redução: 48:1
Velocidade: 140RPM @ 4,5V sem carga
Torque: 800 gf.cm

Alimentação:

A alimentação do veículo será feita utilizando um conjunto de duas baterias 9V (capacidade de 250mAh cada), conectadas em paralelo, suprindo o circuito com 500mAh.

Figura 10: Bateria utilizada no projeto AGV.

Sabendo que o Arduino deverá ser alimentado com 5Vdc, será utilizado o LM 7805, responsável pela regulação de tensão que alimentará a placa.

As especificações do LM7805 são:
Corrente de saída max: 1.5A
Tolerância máxima em Vout: 5%
Proteção interna contra sobrecarga
Tensão de saída: 5.0V, 6V, 8V, 9V, 10V, 12V, 15V, 18V, 24V (será utilizado 5V para este projeto)

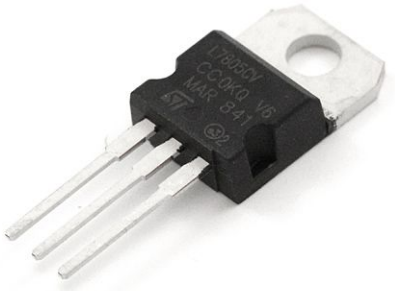


Figura 11: Módulo Regulador de Tensão LM7805.

Célula de Carga:

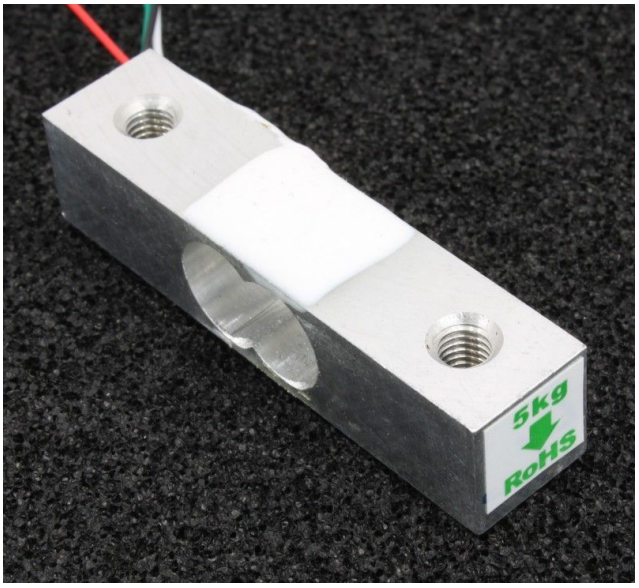


Figura 12: Módulo Célula de Carga CZL635.

A célula de carga é responsável por medir uma deformação que ocorrerá devido à variação do peso na bandeja do AGV (devido à retirada de documento). O sinal proveniente dessa medição será transformado em uma saída de tensão. No caso, a deformação do corpo será proporcional ao peso aplicado a ele, ou seja: quanto menor a alteração, menor o peso.

A conexão deste módulo ao Arduino será feita utilizando as portas I2C da placa, pinos 28 e 27.

- Especificações do produto:

Mechanical

Housing Material Aluminum Alloy
Load Cell Type Strain Gauge
Capacity 5kg
Dimensions 55.25x12.7x12.7mm
Mounting Holes M5 (Screw Size)
Cable Length 550mm
Cable Size 30 AWG (0.2mm)
Cable - no. of leads 4

Electrical

Precision 0.05%
Rated Output 1.0±0.15 mv/V
Non-Linearity 0.05% FS
Hysteresis 0.05% FS
Non-Repeatability 0.05% FS
Creep (per 30 minutes) 0.1% FS
Temperature Effect on Zero (per 10°C) 0.05% FS
Temperature Effect on Span (per 10°C) 0.05% FS
Zero Balance ±1.5% FS
Input Impedance 1130±10 Ohm
Output Impedance 1000±10 Ohm

Insulation Resistance (Under 50VDC) ≥5000 MOhm

Excitation Voltage 5 VDC

Compensated Temperature Range -10 to ~+40°C

Operating Temperature Range -20 to ~+55°C

Safe Overload 120% Capacity

Ultimate Overload 150% Capacity

3) Especificação detalhada do projeto

3.a) Descritivo de hardware

Basicamente, este projeto de hardware do AGV é constituído por 8 blocos funcionais de circuitos que foram instituídos para cumprimento das funções estabelecidas no descritivo inicial. São eles: (1) Circuito da célula de carga; (2) Sensor seguidor de linha; (3) Regulação de tensão; (4) LED's e botões; (5) Driver do motor + motor; (6) ultrassom. O esquemático completo se encontra na seção de Apêndice I.

3.a.1) Circuito da célula de carga- Medição de massa do compartimento do carrinho

Para realizar a medição da variação de massa do compartimento do carrinho, foi utilizada uma célula de carga CZL635-1 [1] que permite uma aferição linear de até 5Kg. Para tanto, esta célula se utiliza de uma estrutura especificada para deformação unidimensional, a qual comprime ou tenciona pequenos "resistores" internos, ocasionando na alteração de suas resistências.

Elas também são construídas de forma a minimizar a variação de sua resistência imposta pela mudança de temperatura ambiente, no entanto, conforme referência [1], em alguns casos são utilizados arranjos de 4 células para que alguns sofram tensionamento e outros compressão, ou até mesmo que se disponham como padrões sem utilização direta, a fim de possibilitar o dimensionamento do efeito da mudança de temperatura e rejeitá-la ao máximo.

Efetuando a leitura do datasheet da célula e de outras referência de exemplo de utilização, em primeira instância, foi confuso verificar se a célula necessita ou não ser aplicada em um dos ramos de uma ponte de Wheatstone, pois a variação de sua impedância de entrada é muito pequena em relação à variação de força aplicada, conforme veremos a seguir.

No entanto, através da leitura de um tutorial [2] da interface SEN0160, a qual utiliza um HX711, concluiu-se que a ponte já está integrada na célula de carga, podendo estabelecer uma conexão direta com a placa de conversão analógica-digital (serial), conforme se verifica na figura 13 e 14.

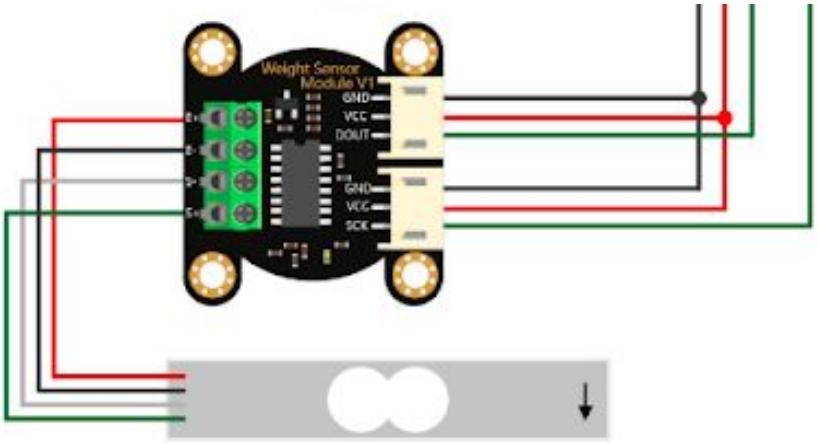


Figura 13: Interligação da célula CZL635 com a interface SEN060.

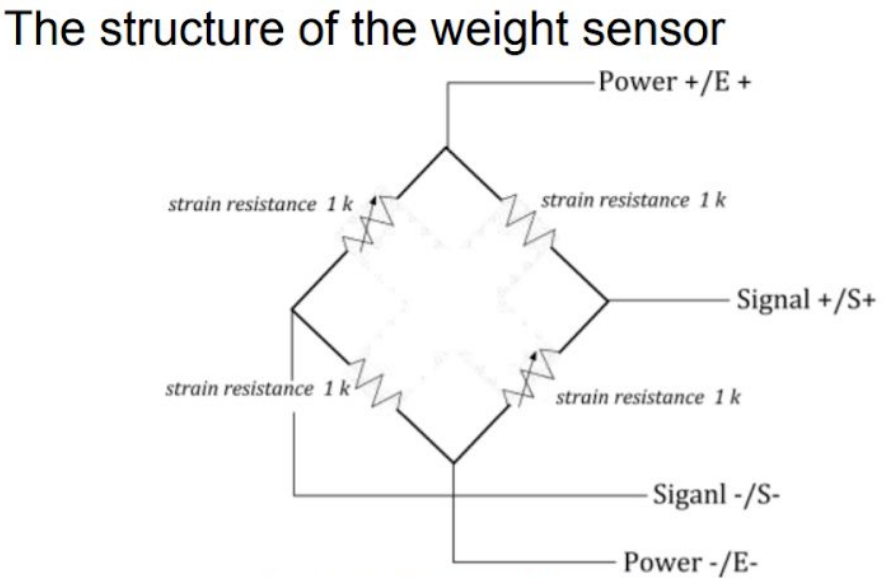


Figura 14: Diagrama interno da célula de carga.

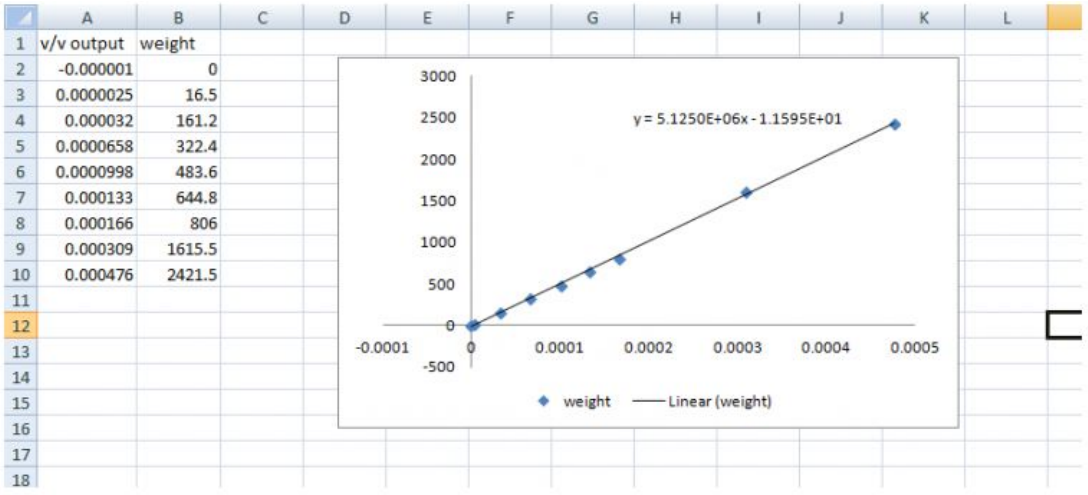
Dessa forma, é essencial que se tenha uma alimentação para os terminais E+ e E- desta ponte de Wheatstone (CZL635) e os sinais analógicos correspondentes à carga aplicada serão retirados dos terminais S+ e S-. No manual da célula, observa-se então que a tensão de excitação (entre os terminais E+ e E-) deve ser de aproximadamente 5VDC.

Electrical	
Precision	0.05%
Rated Output	1.0±0.15 mv/V
Non-Linearity	0.05% FS
Hysteresis	0.05% FS
Non-Repeatability	0.05% FS
Creep (per 30 minutes)	0.1% FS
Temperature Effect on Zero (per 10°C)	0.05% FS
Temperature Effect on Span (per 10°C)	0.05% FS
Zero Balance	±1.5% FS
Input Impedance	1130±10 Ohm
Output Impedance	1000±10 Ohm
Insulation Resistance (Under 50VDC)	≥5000 MOhm
Excitation Voltage	5 VDC
Compensated Temperature Range	-10 to ~+40°C
Operating Temperature Range	-20 to ~+55°C
Safe Overload	120% Capacity
Ultimate Overload	150% Capacity

Figura 15: Especificações elétricas da célula CZL635.

Para termos conhecimento da curva que relaciona a variação da tensão de saída dos sinais S+ e S- com a aplicação de força, foi obtido um exemplo de calibração indicado na figura 16, na qual pode-se verificar que há uma relação linear entre a massa em (g) e a tensão de saída analógica regida pela função abaixo:

$$M(g) = 5,125.10^{6}.Vout(V) - 1,1595$$



	A	B
1	v/v output	weight
2	-0.000001	0
3	0.0000025	16.5
4	0.000032	161.2
5	0.0000658	322.4
6	0.0000998	483.6
7	0.000133	644.8
8	0.000166	806
9	0.000309	1615.5
10	0.000476	2421.5

I have generated this trendline using Excel's built in linear regression function and you can see that the line fits the data quite closely. So we now have our function f(x):

$$f(x) = 5.125E6x - 1.1595$$

Figura 16: Curva de calibração da célula de carga de 5Kg CZL635.

Aplicando 5Kg na função anterior, obtemos o valor de 0,97mV, o qual está condizente com a tensão de saída de fundo de escala especificada na figura 15. Através de um estudo do manual do HX711, o qual integra o módulo SEN0160, verifica-se que é necessária a construção de um circuito equivalente ao da Figura 17.

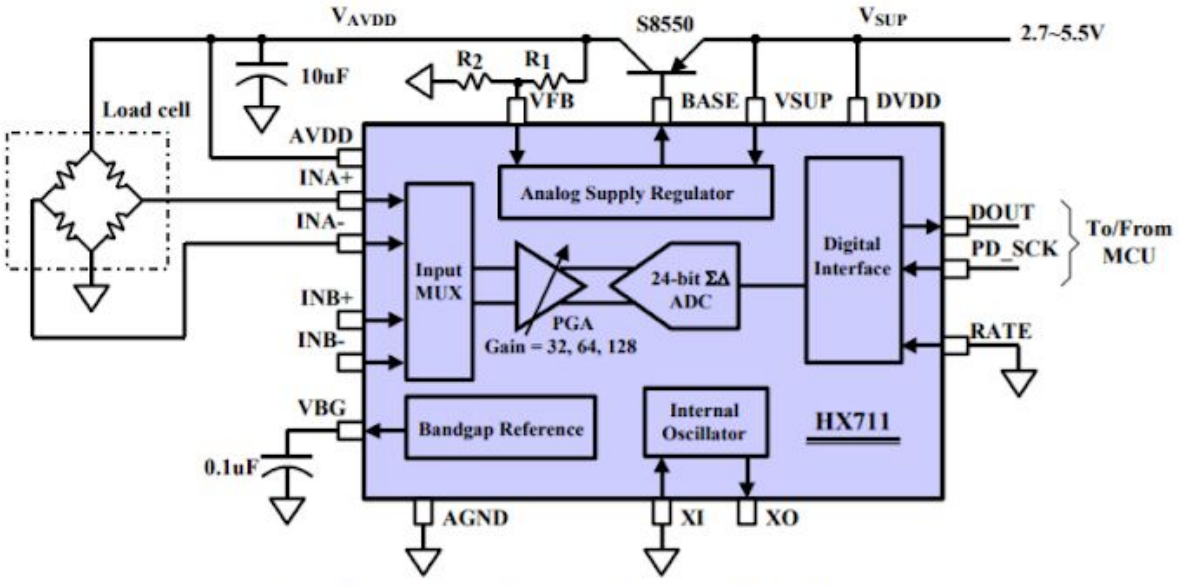


Figura 17: Circuito típico do HX711.

No entanto, utilizando o módulo de conversão SEN0160, foi economizado o tempo gasto para cálculos dos resistores R1 e R2, do transistor S8550, e do filtro de alimentação da célula de carga através da saída AVDD. Tal módulo, então, já se apresenta integrado a CZL635 na figura 18 abaixo.

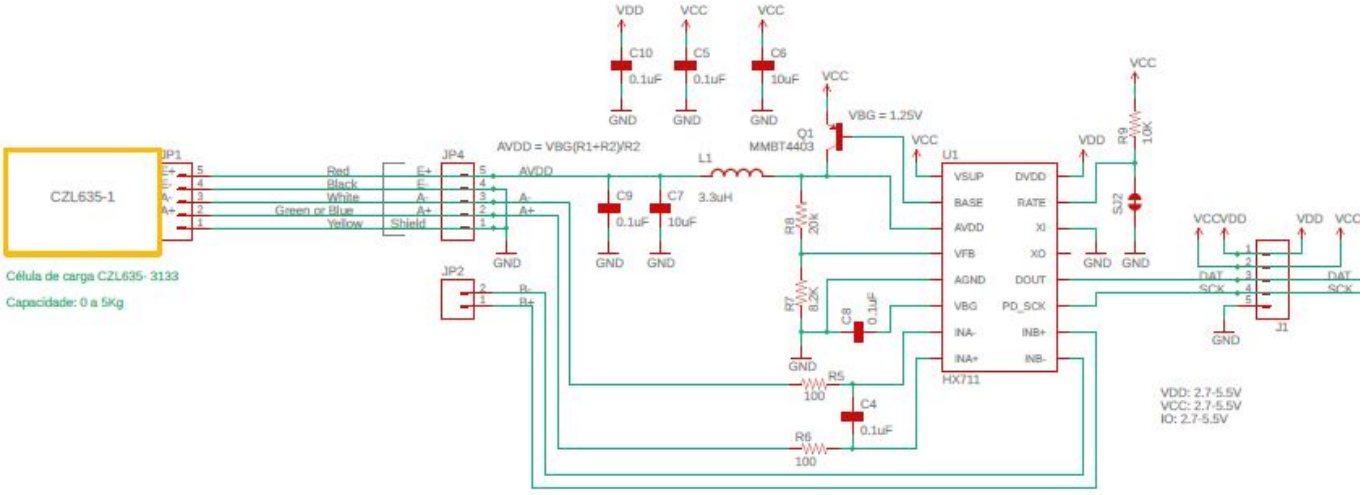


Figura 18: Integração da célula de carga CZL635 ao módulo SEN0160.

Segundo o manual do HX711, é solicitada que a alimentação AVDD esteja dentro da faixa de 2,6V a 5,5V. Assim, o transistor Q1 é polarizado pela saída “Base”, a qual é tem seu potencial ajustado por um regulador interno. Logo, este circuito efetua a alimentação de AVDD e em seguida, foi construído um divisor de tensão para alimentação do circuito digital interno por meio de VFB com uma tensão em 1,5V, ressaltando também a necessidade de instalação de um capacitor de desacoplamento (C8) para o terra do circuito digital.

No intuito de eliminar possíveis ruídos de alta frequência da alimentação da célula, o circuito do módulo possui um filtro passa baixas de segunda ordem com frequência de corte de aproximadamente 20 kHz (valor calculado a partir dos valores de L1, C7 e C9).

Já em relação a entrada diferencial INA+ e INA-, teremos a aplicação de no máximo 1mV nos terminais dos resistores de R5 e R6 de 100 ohms, enquanto que o capacitor C4 desempenha a função de inserção de um ruído de mesmo nível em ambas as entradas, a fim de eliminá-lo no processo de comparação com uma rejeição de modo diferencial.

Enquanto ao processo de conversão analógico-digital, não há necessidade de fornecimento de um sinal de relógio para o conversor, uma vez que há um oscilador interno, conforme [12], mas deverá ser cedido um sinal de clock para o processo de comunicação serial como veremos a seguir.

Basicamente, o fundo de escala deste dispositivo é proporcional à tensão aplicada em AVDD, sendo que caso seja aplicada uma tensão em torno de 5V, teremos um fundo de escala da entrada diferencial de 40mV. Dessa forma, considerando um ganho configurado em 32 vezes, iremos multiplicar nosso 1mV de fundo de escala inicial para um novo limite de 32mV. Além disso, como o HX711 converte estes 40mV em um fundo de escala das entradas INA+ e INA- em um dado de 24 bits correspondente a “FFFFFF” (hex), ao aplicarmos a tensão de 32mV, teremos o valor máximo de “CCCCCC” (hex) convertido em complemento de 2.

Por outro lado, necessitamos configurar corretamente a comunicação serial para obtenção do ganho especificado de 32 e também para que a saída de dados “DOUT” emita o dado convertido. Podemos verificar que o protocolo de comunicação à primeira vista se assemelha com a comunicação I2C. No entanto, lendo o manual do componente e um código exemplo [14] verificamos que é um protocolo desenvolvido de forma customizada por um fabricante.

Logo, observando as figuras 19 e 20, para início da comunicação, o HX711 deve sinalizar o status ok do canal, realizando um borda de descida do pino “DOUT” com a entrada de clock em nível baixo. A transmissão de cada um dos bit dos 24 convertidos será dada, então, por cada borda de subida realizada pelo controlador Arduino no nosso caso. Na vigésima quinta borda, o sinal “DOUT” é transitado para nível alto, enquanto são dados os pulsos finais de clock. Vale ressaltar, neste ponto, que a quantidade de pulsos de CLK seleciona nosso ganho. Portanto, deverá ser configurados no código do Arduino, uma sequência serial de 26 pulsos para o ganho de 32.

Serial Interface

Pin PD_SCK and DOUT are used for data retrieval, input selection, gain selection and power down controls.

When output data is not ready for retrieval, digital output pin DOUT is high. Serial clock input PD_SCK should be low. When DOUT goes to low, it indicates data is ready for retrieval. By applying 25~27 positive clock pulses at the PD_SCK pin, data is shifted out from the DOUT output pin. Each PD_SCK pulse shifts out one bit, starting with the MSB bit first, until all 24 bits are shifted out. The 25th pulse at PD_SCK input will pull DOUT pin back to high (Fig.2).

Input and gain selection is controlled by the number of the input PD_SCK pulses (Table 3). PD_SCK clock pulses should not be less than 25 or more than 27 within one conversion period, to avoid causing serial communication error.

PD_SCK Pulses	Input channel	Gain
25	A	128
26	B	32
27	A	64

Table 3 Input Channel and Gain Selection

Figura 19: Tabela de relação de ganho com a quantidade de pulso no terminal de clock da comunicação serial.

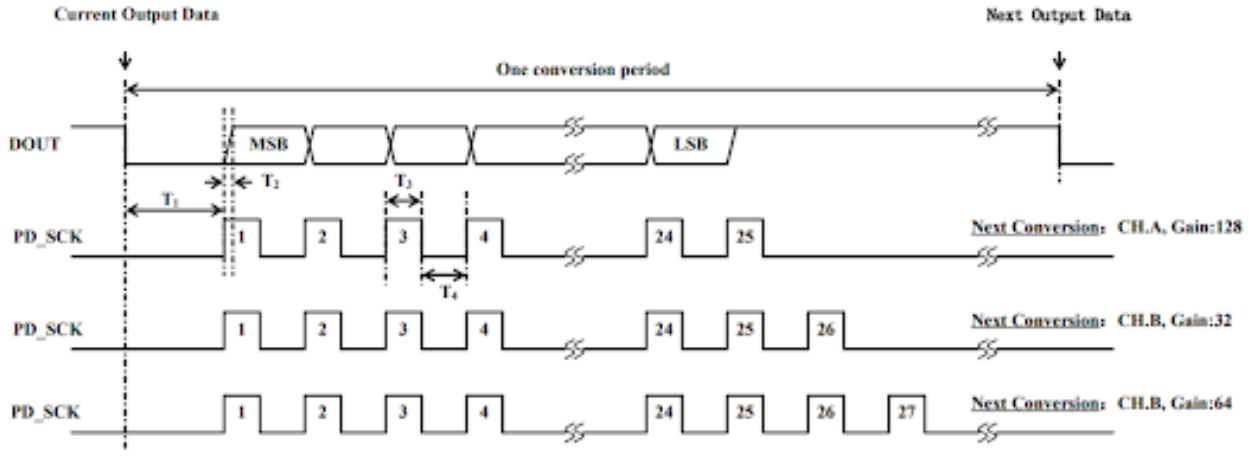


Figura 20: Diagrama de tempo da comunicação serial do HX711.

Assim, interligamos os pinos DAT e SCK do conversor com os pinos SDA e SCK, respectivamente, do Arduino, conforme ilustra a Figura 21.

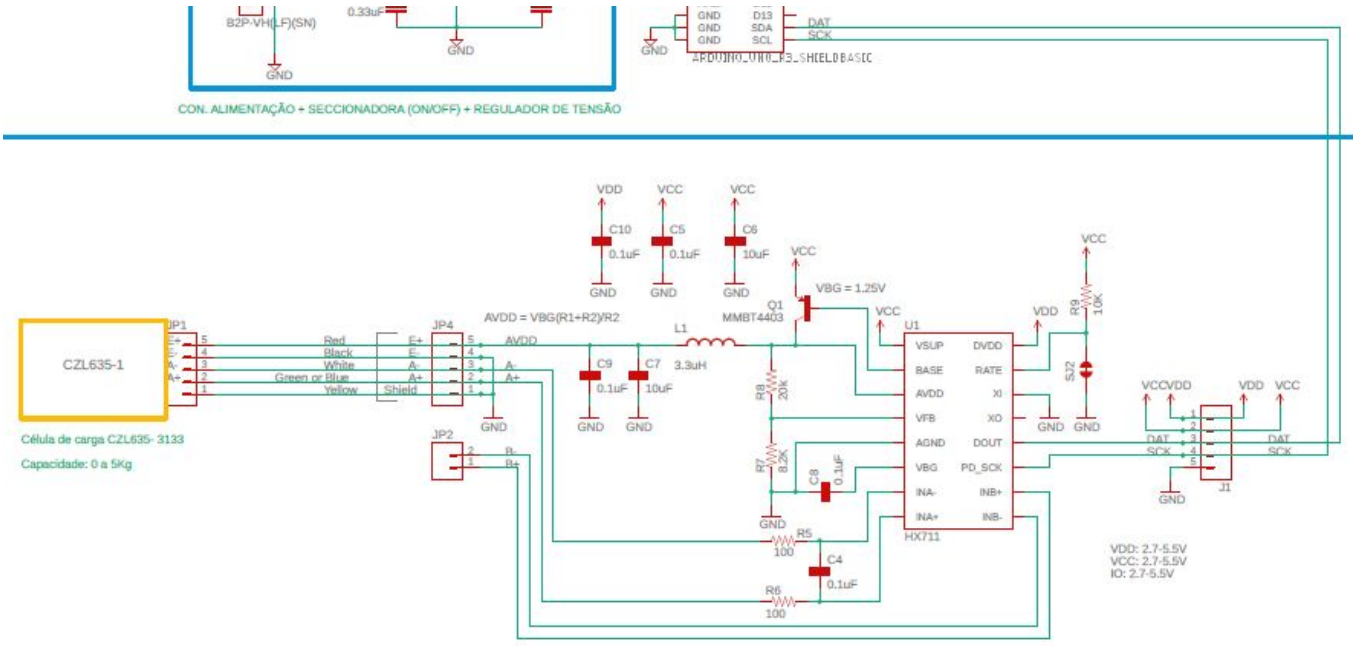


Figura 21: Circuito da célula de carga e integração com o Arduino.

Para a futura configuração da comunicação, podemos nos basear no segundo o trecho de código [14] abaixo, substituindo a entrada A1 por SDA e A0 por SCL. Enquanto à configuração de ganho, deveríamos apenas realizar uma modificação enquanto à quantidade de chamadas da rotina CLK(), que neste caso está setada com 25 chamadas, mas deverá ser de 26.

```
        /* This program takes 10 samples from LC + HX711B at
   1-sec interval and then computes the average.
*/

unsigned long x = 0, y=0;
unsigned long dataArray[10];
int j = 0;
void setup()
{
  Serial.begin(9600);
  pinMode(A1, INPUT); //data line  //Yellow cable
  pinMode(A0, OUTPUT);  //SCK line  //Orange cable
}
```

```
void loop()
{

  for (int j = 0; j < 10; j++)
  {
    digitalWrite(A0, LOW);//SCK is made LL
    while (digitalRead(A1) != LOW) //wait until Data Line goes LOW
      ;
    {
      for (int i = 0; i < 24; i++)  //read 24-bit data from HX711
      {
        clk();      //generate CLK pulse to get MSB-it at A1-pin
        bitWrite(x, 0, digitalRead(A1));
        x = x << 1;
      }
      clk();  //25th pulse
      Serial.println(x, HEX);
      y = x;
      x = 0;
      delay(1000);
    }
    dataArray[j] = y;
  }

  Serial.println("===averaging process=========");
  unsigned long sum = 0;

  for (j = 0; j < 10; j++)
  {
    sum += dataArray[j];
  }
  Serial.print("Average Count = ");
  sum = sum / 10;
  Serial.println(sum, HEX);
 // float W = (float)0.90*(sum-901002)/946560 + 0.75;//0.005331 * sum -
1146.176;
  //W = (float)W / 1000.00;
```

```
  Serial.println(W, 2);
}

void clk()
{
  digitalWrite(A0, HIGH);
  digitalWrite(A0, LOW);
}
```

3.a.2) Sensor seguidor de linha

Fundamentalmente, o sensor seguidor linha no qual está integrado o conjunto emissor-receptor TCRT5000, irá levar a saída digital para nível alto quando a luz infravermelha (950nm) do emissor incidir no fototransistor (figura 22) em uma intensidade suficiente para que o sinal da entrada não-inversora do LM393 seja suficientemente superior à entrada de referência (porta inversora). Ou seja, teremos reflexão (nível alto na saída do módulo), quando o emissor estiver incidindo luz em uma região não opaca, clara e reflexiva.

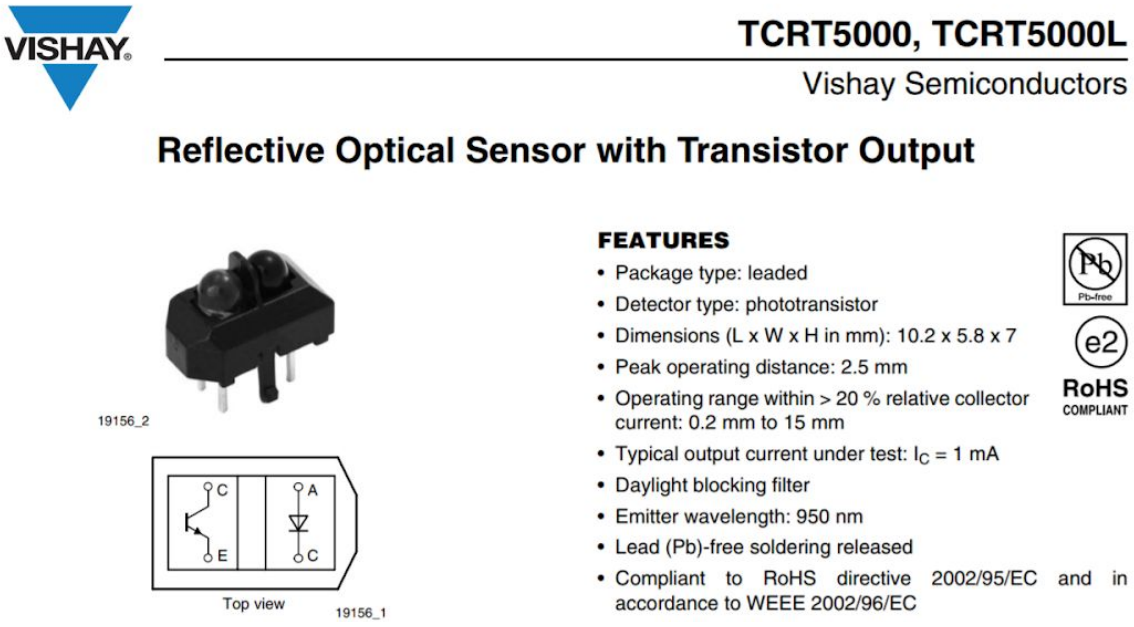


Figura 22: Detalhe para os componentes do sensor óptico reflexivo.

Assim, foram alimentados com a tensão de saída do regulador os três sensores reflexivos, conforme descritos anteriormente e, reservadas as entradas digitais D8, D9 e D10 para esta aplicação de acordo com a figura 23. Devemos citar apenas uma ressalva que seria em relação aos resistores de pull-down que podem ser necessários, caso a saída digital não esteja dentro dos limites máximo de nível baixo. No entanto,

em primeiro momento eles não foram implementados, pois já devem ter sido implantados no próprio módulo.

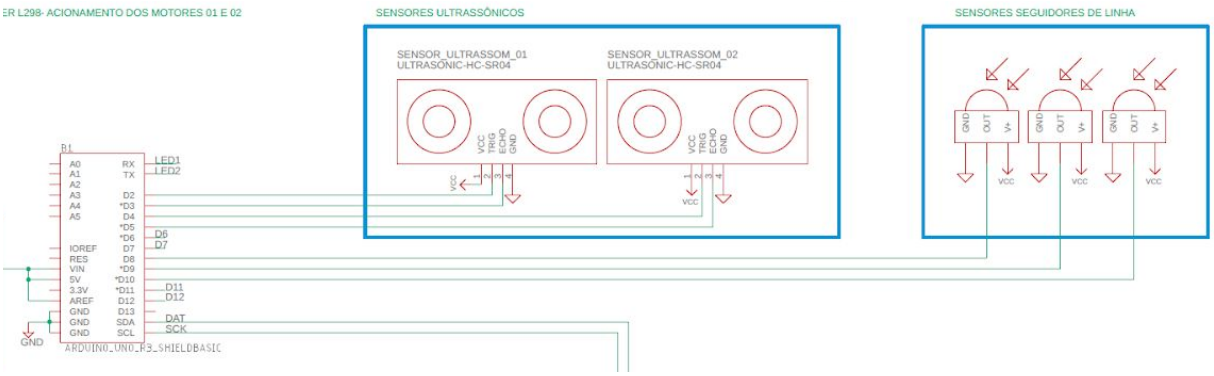


Figura 23: Esquemático da implementação dos sensores seguidores de linha.

3.a.3) Circuito de regulação e seccionamento (ON/OFF)

Como foi determinada uma alimentação de 5V para o controlador e seus periféricos, foi utilizado o 7805 para realizar o papel de regulação da tensão, junto a dois capacitores (C2 e C3) atuando como filtros passa-baixas, os quais tiveram seus valores baseados no exemplo de aplicação do próprio datasheet mostrado a seguir.

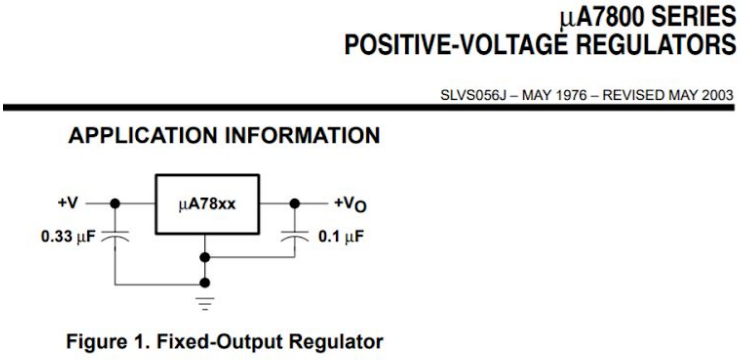


Figura 24: Circuito típico do regulador 7805.

Além disso, foi instalada a chave SW1 na entrada do regulador, a fim de viabilizar a energização e desenergização do circuito, caso necessário.

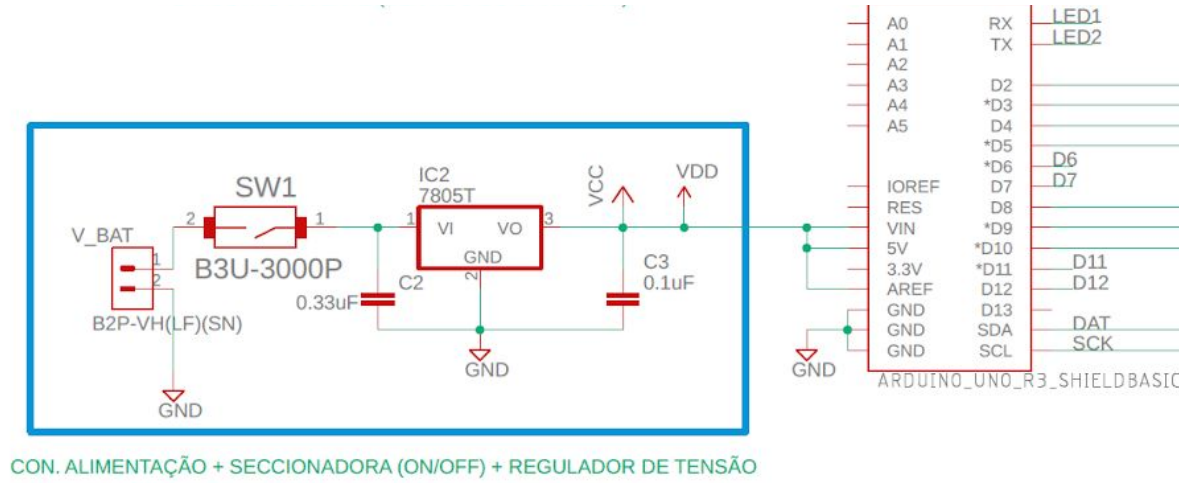


Figura 25: Circuito de regulação e seccionamento.

3.a.4) Circuito de sinalização de direção (LED's) e botões de comando.

No intuito de possibilitar a direção atual do carrinho, foram interligados dois LED's em modo de cátodo comum utilizando os pinos que estão nomeados como RX e TX no esquemático, mas que futuramente teriam suas funções reservadas como GPIO's. Logo, foi estabelecida determinada uma corrente de operação em torno de 15 mA e uma queda de 1,8V nos LED's, resultando em resistores de polarização de 220 ohms.

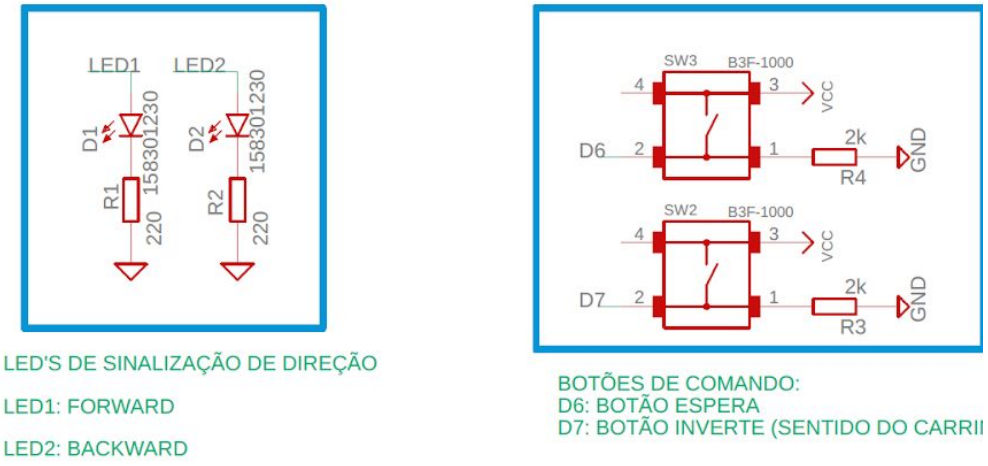
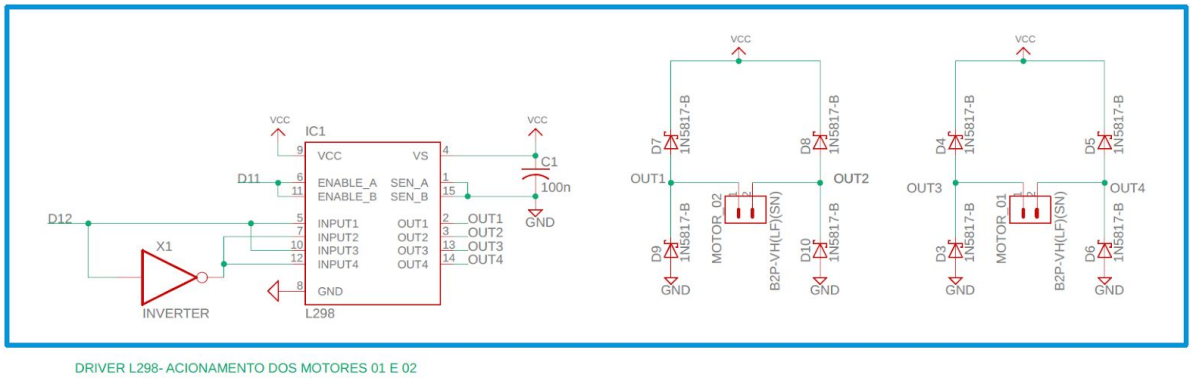


Figura 26: Circuitos de sinalização e comandos através de push-buttons

Por outro lado, para realizar os comandos de reversão e de espera, foram inseridos dois botões de acionamento em nível alto com resistores de pull-down, a fim de evitar a flutuação das tensões nas portas D6 e D7.

3.a.5) Driver do motor

O bloco de esquemático para o driver dos motores é mostrado abaixo.



DRIVER L298- ACIONAMENTO DOS MOTORES 01 E 02

Figura 27. Driver para acionamento dos motores.

O driver utilizado, L298, apresenta uma ponte H dupla na qual as entradas 1 e 2 controlam as saídas 1 e 2, enquanto as entradas 3 e 4 controlam as saídas 3 e 4. Quando as entradas são acionadas como 10 os motores são acionados com uma polarização inversa àquela quando as entradas são 01. Como os motores sempre funcionam em conjunto, usou o mesmo pino do Arduino e um inversor para controlar essas entradas.

Além disso, para que o drive funcione existem dois pinos de habilitação, um para cada ponte H, os quais devem ser controlados pelo Arduino. Para que possibilite um controle da velocidade do motor esses pinos foram conectados a uma saída de PWM do Arduino, a qual possibilita que seja aplicado um sinal modulado por largura de pulso que habilita/desabilita o módulo de acordo com o ciclo de trabalho regulando, assim, a velocidade de rotação do motor.

É importante destacar nesse bloco do circuito que para motor acionado foi montado uma ponte H de diodos. A função desses diodos é fornecer um caminho de baixa impedância para as variações de corrente quando o driver é desabilitado uma vez que por ser uma carga indutiva o motor não permite variações abruptas na corrente. Sem essa ponte de diodo o driver de acionamento poderia ser danificado pela corrente de retorno. Para a escolha do diodo seguiu-se a orientação do datasheet do driver e escolheu um diodo com tempo de restauração reversa menor que 200ns.

3.a.6) Sensores de ultrassom

A interface dos sensores de ultrassom com o Arduino é muito simples. são utilizados apenas duas gpio's, uma configurada como saída, Triger, e outra como entrada, Echo. O funcionamento desses sensores já foi explicado na descrição dos componentes abaixo segue o bloco de esquemático para interligação desses componentes.

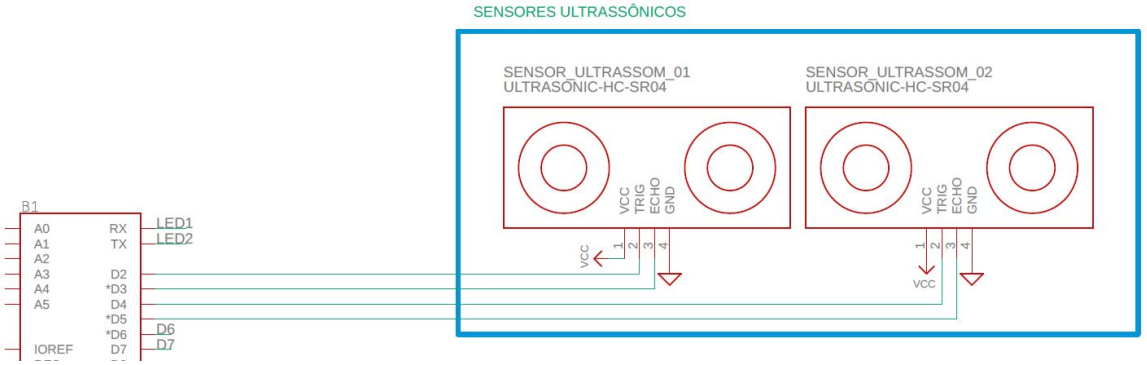


Figura 28. Esquemático sensores ultrassom.

3.b) Software:

Na figura abaixo, observa-se uma representação da lógica de funcionamento através de uma FSMD.

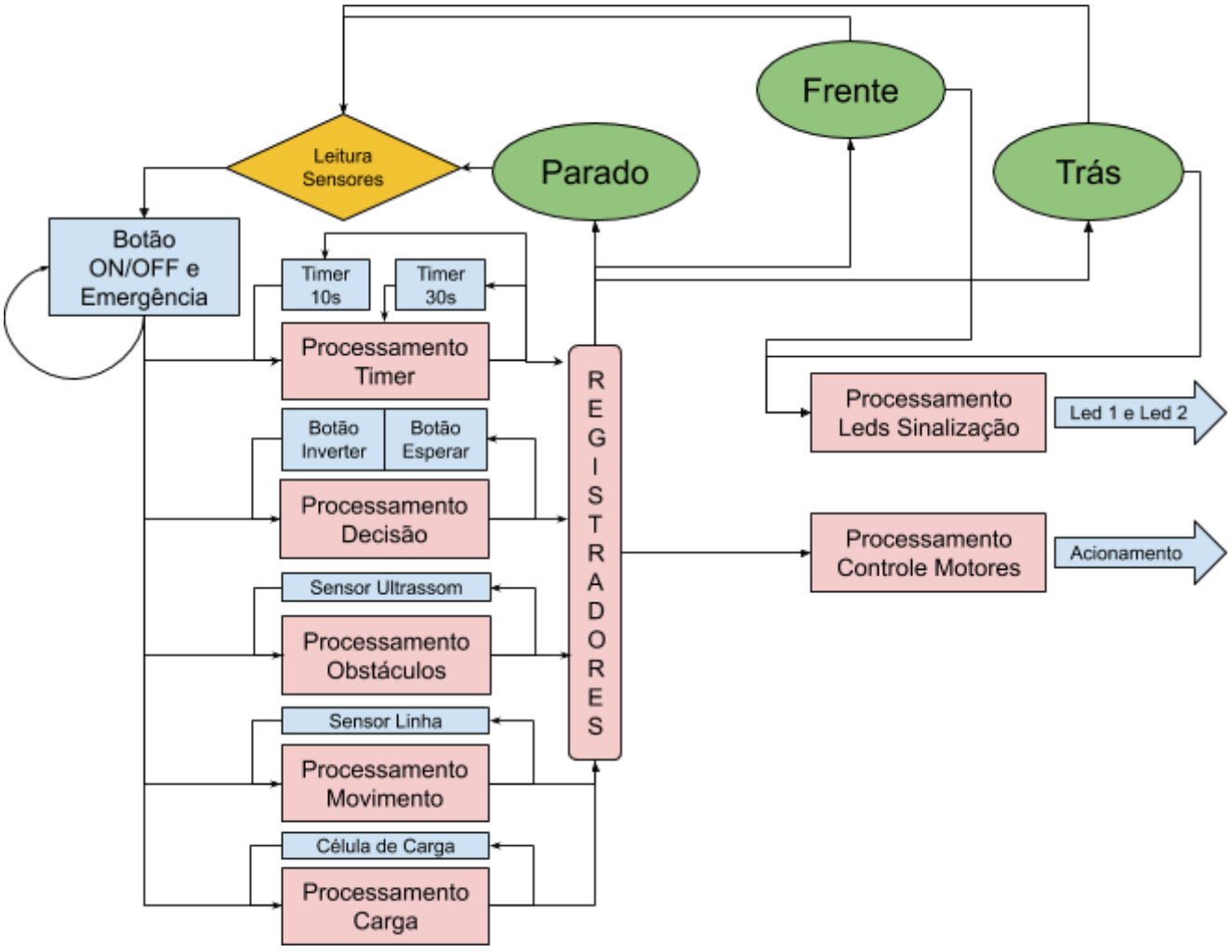


Figura 29. Representação da lógica de funcionamento do AGV através de uma FSMD.

Basicamente podemos modelar o AGV com três estados: parado, frente, trás. No estado parado devem ser monitorados o tempo no qual o veículo deve permanecer em stand-by e a carga presente no mesmo, a fim de determinar quando o mesmo deve movimentar-se transitando para os estados frente e trás. Além disso, enquanto o veículo está parado pode-se alterar a direção do movimento e solicitar mais tempo pelo acionamento dos botões. Nos dois últimos estados devem ser

monitorados o sensor de ultrassom para determinar quando há um obstáculo e os sensores seguidores de linha. Também deve ser controlado o driver do motor e os leds de identificação.

Para a implementação do projeto, faz-se necessário o uso de software para controle, configuração dos pinos e tomadas de decisão do AGV de acordo com os requisitos pré-estabelecidos. Deve-se configurar como pinos de entrada os botões, o echo e sensores seguidores de linha. Quanto aos LEDs, pinos do driver do motor e trigger dos sensores de ultrassom devem ser configurados como saída. Como já mencionado o AGV fica parado durante 30 segundos numa estação a espera de que haja uma variação de carga, que caso se concretize o carro aguarda mais 10 segundos e retoma o movimento. Para tanto aciona-se o contador TIMER1_COMPA para gerar interrupção uma vez a cada segundo, para toda vez que interromper o fluxo do programa seja desviado para uma rotina de serviço que decrementa de um a variável auxiliar (remaining_time), até que esta se anule. O valor da variável inicial é 30 e quando detecta-se um delta de carga, ele é atualizado para 10. Como existe ainda um botão para aumentar 30 segundos ao tempo de espera do carro, é configurada uma interrupção para incrementar 30 na variável auxiliar.

Além disso, uma função do AGV é a detecção de um obstáculo a uma determinada distância. Caso esta de 2cm, o carrinho fica para o movimento. Para esta funcionalidade, utiliza-se o contador TIMER2_COMPA do arduino configurando-o para gerar uma chamada interrupção a cada 50us assim consegue-se a medição de uma distância de até 2cm (conforme mencionado no módulo do Sensor HC-SR04) .

Sobre os estados de movimento do AGV, necessita-se do acionamento do pino D11 configurado como PWM, ao qual será feito o controle da velocidade dos dois motores. Desta forma basta configurá-lo para estado parado (duty cicle = 0%) ou para duty cicle não nulo para entrar em movimento.

Por último, quanto à direção do veículo, é possível mudá-la através do pino D12. As funcionalidades implementadas por software podem ser vistas no pseudo - código a seguir:

```
/*****************************************************************************
* Pseudo Código AGV
*****************************************************************************/
// Variaveis
bool IR_R = 0, IR_L = 0, IR_C = 0; // estado dos sensores infravermelho (direita, esquerda e central)
bool dir = 0; // indica direção do movimento
bool moving = 0; // indica se veiculo esta em movimento
bool station = 0; // indica que AGV esta em uma estação e deve andar até sair dela
bool echo_request = 0; // indice se sinal do ultrassom foi solicitado
int carga = 0; // armazena valor de carga no veiculo
int remaining_time = 0; // tempo restante em segundos para veiculo voltar a movimentar
int dist = 0; // distancia até obstaculo em cm
int echo_count = 0; // contador para determinar tempo do pulso de echo

/*****************************************************************
* Rotinas principal
*****************************************************************/
int main() {
   setup();
   while(true) {
      loop();
   }
}

/*****************************************************************
* Rotinas normais
*****************************************************************/
void setup() {
   configGPIOs();
   configInterrupt();
   configTimers();
}

void loop() {
   if (moving) {
      // Pulso no triger de ultrasom de acordo com direção
      set(Triger + dir);
      delay(10);
      clear(Triger + dir);
      echo_request = 1;
      while(echo_request){};
      if (dist < 10) {    // para quando distancia menor que 10 cm
         stopAGV();
         return;
```

```
        }
        // Check estado sensores seguidores de linha para determinar movimento
        if (IR_C == 0) {
            dir = !dir;
            runAGV();
            return;
        }
        if (IR_R == 1 && IR_L == 1 && station == 0) {
            stopAGV();
            station = 1;
            return;
        }
        if (IR_R == 0 && IR_L == 0) {
            station = 0;
            return;
        }

    } else {
        // Lê celula de carga e verifica se houve variação na carga
        leCarga();
        if (deltaCarga) {
            remaining_time = 10;
        }

        if (remaining_time <= 0) {  // Esgotou tempo parado
            runAGV();
            return;
        }

    }

}

void stopAGV() {
    // Para AGV desabilitando motor. Escrevendo PWM nulo, configura variáveis e habilita
timer.
    stopMotor();
    moving = 0;
    remaining_time = 30;
    enableTimer1();
}

void runAGV() {
    // Aciona pino de direção do driver de acordo com variavel dir, configura variaveis e
desabilita timer.
    if (dir) {
```

```
        set(DIR);
    } else {
        clear(DIR);
    }
    runMotor(); //escreve PWM com velocidade desejada
    moving = 1;
    set (LED + dir); // aciona led correto de acordo com direção
    disableTimer1();
}

configGPIOs() {
    /* Os leds, os pinos do driver do motor e trigger dos sensores de ultrassom
    devem ser configurados como saída. Já os botões, echo e sensores seguidores de linha
    devem ser configurados como entrada. */
}

configInterrupt() {
    /* Serão utilizadas as seguintes rotinas de interrupção:
    - TIMER1_COMPA: Timer1 do Atmel configurado para interrupção a cada segundo
    para controlar tempo parado;
    - TIMER2_COMPA: Timer2 configurado para interrupção a cada 50us para determinar
    tempo do pulso de echo do ultrassom. Com esse valor temos precisão de 2cm.
    - PCINT0: interrupção da GPIO relacionada aos sensores infravermelho (borda subida e
descida);
    - PCINT2: interrupção da GPIO relacionada aos botões (borda subida) e echo do
ultrassom
    (borda subida e descida);
    */
}

configTimers() {
    // Configura Timers 1 e 2 como descrito anteriormente.
}

/*********************************************************************
* Rotinas de Interrupção
*********************************************************************/
void TIMER1_COMPA() {
    // Decrementa remaining timer
    remaining_time = remaining_time - 1;
}

void TIMER2_COMPA() {
    echo_count = echo_count + 1;
}
```

```
void PCINT0() {
   // Confira flags para os sensores de infravermelho.
   findChange();
   configFlags();
}

void PCINT2() {
   findChange(); // determina qual pino gerou interrupção e qual transição (alto p/ baixo ou
baixo p/ alto)

   if(button_wait) {  // botão de espera foi pressionado
      remaining_time = remaining_time + 30;
   }

   if (button_dir) {  // botão para mudar direção foi pressionado
      dir = !dir;
   }

   if (echo1LH || echo2LH) {  // echo1 ou echo2 passa de baixo para alto
      echo_count = 0;
      enableTimer2();
   }

   if (echo1HL || echo2HL) {  //echo1 ou echo2 passa de alto para baixo
      disableTimer2();
      dist = calcDist();     // calcula distancia em cm
      echo_request = 0;
   }
}

/*****************************************************************/
```

4) Planejamento de testes a serem realizados ao longo do desenvolvimento.

Para validação das funções a serem implementadas e do hardware escolhido, durante toda a implementação do projeto propomos a realização de testes. Para tal, os testes serão divididos em testes unitários e testes de integração, conforme descrito abaixo:

Testes unitários:

- Teste da fonte de alimentação;
- Teste da célula de carga com o Arduino;
- Teste do sensor de ultrassom com o Arduino;
- Teste do sensor de linha com o Arduino;
- Teste dos botões;
- Teste dos leds de identificação;
- Teste dos motores de forma individual;

Para cada caso de teste, propomos a implementação de um código de teste onde a intenção é exercitar e testar a aquisição e tratamento dos dados dos periféricos assim como o acionamento dos motores.

Uma vez que tenhamos a garantia de que os módulos estão funcionando perfeitamente de forma individual, partimos para a integração com o microcontrolador e com os testes de integração. Sendo que o passo a passo exige que ao introduzir cada um dos módulos seja verificado que o sistema permanece funcionando. Assim, se por algum motivo o sistema tenha algum tipo de comportamento não esperado, saibamos a qual parte do sistema pode ser atribuída tal anomalia.

Como meio de garantir a estabilidade do sistema em diferentes condições de funcionamento, propomos que sejam testados todos os casos de funcionamento por 3 vezes em cada situação.

Depois de validado o protótipo e que seja definido a liberação para fabricação final, se faz necessário a realização de testes de certificação junto à Anatel e Inmetro, sendo testes de comunicação, EMC, segurança elétrica, etc.

5) Referências

[1] Datasheet da célula de carga:

https://www.robotshop.com/media/files/pdf/datasheet-3133.pdf

[2] Tutorial de utilização da célula com a interface SEN0160:

https://www.sigmaelectronica.net/wp-content/uploads/2018/08/sensor-de-peso-SEN0

[3] Arduino UNO R3:

https://store.arduino.cc/usa/arduino-uno-rev3

[4] Sensor Ultrassônico de distância - HC-SR04:

https://i0.wp.com/portal.vidadesilicio.com.br/wp-content/uploads/2017/05/foto_4.jpg?w=450&ssl=1

[5] Sensor seguidor de linha:

https://www.eletrogate.com/sensor-de-obstaculo-reflexivo-infravermelho?utm_source=Site&utm_medium=GoogleMerchant&utm_campaign=GoogleMerchant&gclid=Cj0KCQiAz53vBRCpARIsAPPsz8W0T5tZsuJ_yIZOKXVxP9tICqlG82TxJ073M9w3kaYn3JslwghFWLEaAhaFEALw_wcB

[6] Push-button:

https://www.arduino.cc/en/tutorial/pushbutton

[7] Push-button:

https://portal.vidadesilicio.com.br/ponte-h-l298n-controle-velocidade-motor/

[8] LED:

https://www.baudaeletronica.com.br/led-difuso-5mm-vermelho.html?gclid=Cj0KCQiAz53vBRCpARIsAPPsz8WrfAgR9scShaP5rcF66JIaIeBzbCS9r-6c5pE2E7bx7Co1bmSkG5QaAq9zEALw_wcB

[9] Motores:

https://www.robocore.net/loja/motores/motor-dc-3-6v-com-caixa-de-reducao-e-eixo-duplo

[10] Bateria:

http://www.elgin.com.br/institucional/produto.php?prod=MTIx

[11] Regulador de tensão:

https://pdf1.alldatasheet.com/datasheet-pdf/view/85503/ETC/LM7805.html160.pdf

[12] Datasheet do HX711:

http://image.dfrobot.com/image/data/SEN0160/hx711_english.pdf

[13] Exemplo de curva de calibração da célula de carga:

https://www.phidgets.com/docs/Calibrating_Analog_Sensors

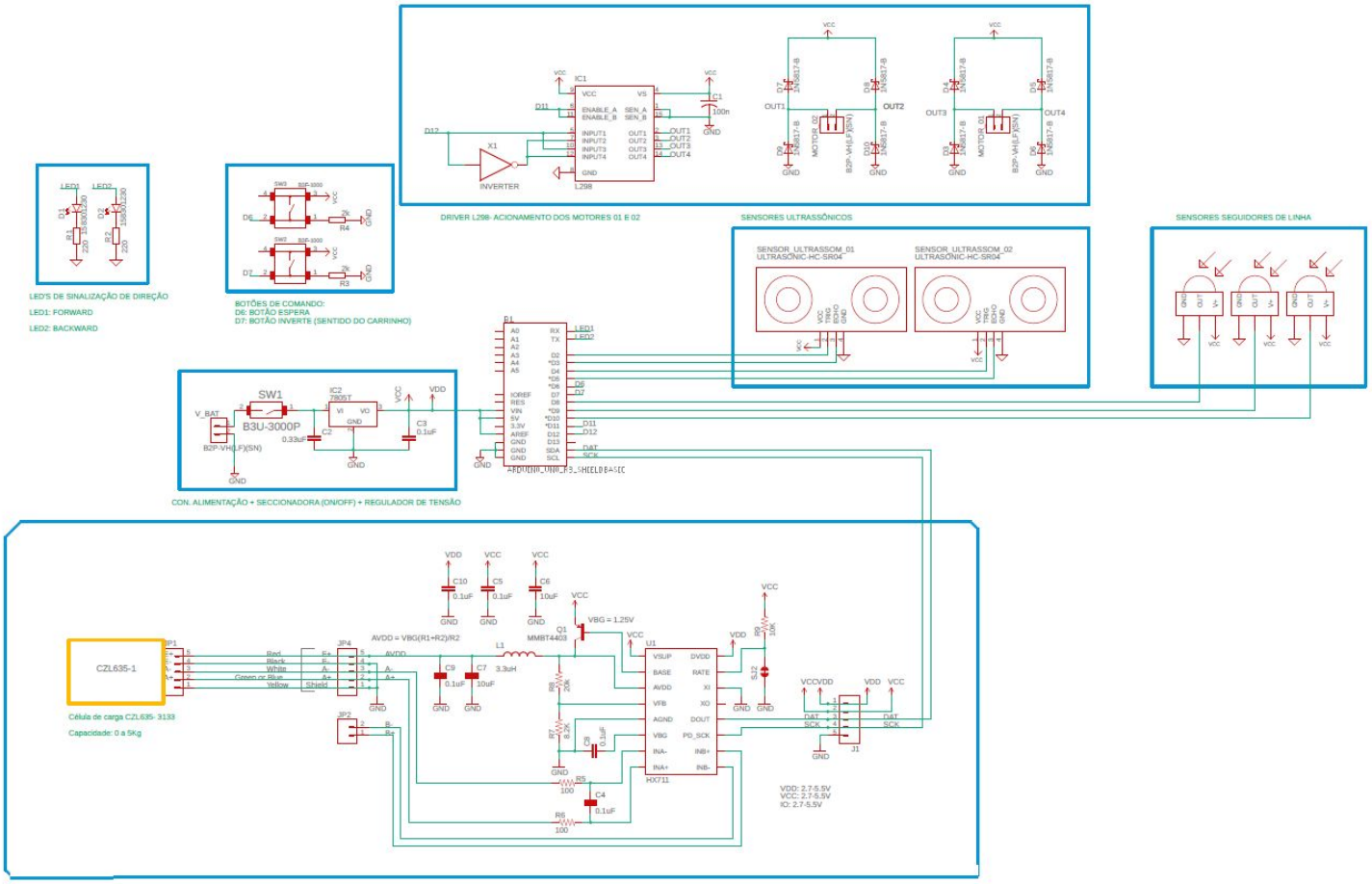
[14] Código exemplo de comunicação do HX711:

https://forum.arduino.cc/index.php?topic=418170.0

[15] Módulo do sensor seguidor de linha:

https://www.masterwalkershop.com.br/sensor-seguidor-de-linha-tcrt5000

APÊNDICE I


LED'S DE SINALIZAÇÃO DE DIREÇÃO
LED1: FORWARD
LED2: BACKWARD
BOTÕES DE COMANDO:
D6: BOTÃO ESPERA
D7: BOTÃO INVERTE (SENTIDO DO CARRINHO)
DRIVER L298- ACIONAMENTO DOS MOTORES 01 E 02
SENSORES ULTRASSÔNICOS
SENSOR_ULTRASSOM_01
ULTRASONIC-HC-SR04
SENSOR_ULTRASSOM_02
ULTRASONIC-HC-SR04
SENSORES SEGUIDORES DE LINHA
CON. ALIMENTAÇÃO + SECCIONADORA (ON/OFF) + REGULADOR DE TENSÃO
CZL635-1
Célula de carga CZL635- 3133
Capacidade: 0 a 5Kg
VDD: 2.7-5.5V
VCC: 2.7-5.5V
IO: 2.7-5.5V