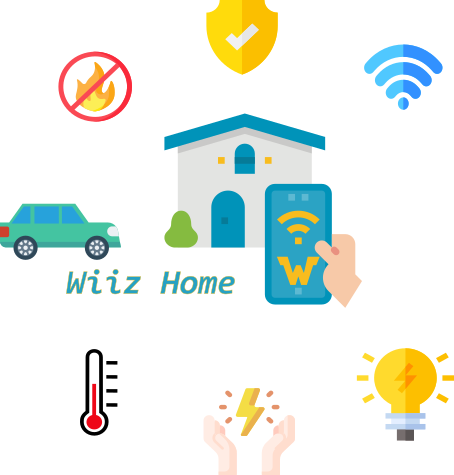
Wiiz Home

*Sua casa sempre inteligente e segura.  
Stay safe and smart*

Versão 1.0

**

*Desenvolvido por: Danilo Faria Dutra*

**ÍNDICE**

**PREFÁCIO ........................................................................................................ 4**

1. **INTRODUÇÃO AO DOCUMENTO**
   1. TEMA .................................................................................................... 5
   2. OBJETIVO DO PROJETO .................................................................... 5
   3. DELIMITAÇÃO DO PROBLEMA .......................................................... 5
   4. JUSTIFICATIVA DA ESCOLHA DO TEMA .......................................... 5
   5. MÉTODO DE TRABALHO .................................................................... 5
   6. ORGANIZAÇÃO DO TRABALHO ......................................................... 5
2. **DESCRIÇÃO GERAL DO SISTEMA**
   1. DESCRIÇÃO DO PROBLEMA ............................................................. 5
   2. PRINCIPAIS ENVOLVIDOS E SUAS CARACTERÍSTICAS ................ 6
   3. DESENVOLVEDORES DO PROTÓTIPO ............................................ 6
   4. REGRAS DE NEGÓCIO ....................................................................... 6
3. **ANÁLISE E DESIGN**
   1. ARQUITETURA DO SISTEMA ............................................................. 7
   2. DADOS DA COMPILAÇÃO NA IDE ..................................................... 8
   3. DIAGRAMA DE BLOCOS ..................................................................... 9
   4. FLUXOGRAMA ................................................................................... 10
      1. FLUXOGRAMA DO SENSOR LDR ............................................ 10
      2. FLUXOGRAMA SENSOR ULTRASSÔNICO ............................. 11
      3. FLUXOGRAMA DO SENSOR DE GÁS ..................................... 12
   5. CÓDIGO FONTE ................................................................................ 13
      1. PRÉ-ALOCAÇÕES: CONSTANTES E BIBLIOTECAS NA MEMÓRIA DO PROCESSADOR .................................................. 13
      2. DECLARAÇÃO DO OBJETO LCD ............................................. 15
      3. DECLARAÇÃO DAS VARIÁVEIS ............................................... 16
      4. CONFIGURAÇÃO DAS CONSTANTES E INICIALIZAÇÃO DO OBJETO LCD ................................................................................ 17
      5. CONSTRUÇÕES DAS FUNÇÕES ............................................. 18
      6. INICIALIZAÇÃO DO LOOP ........................................................ 20
   6. AMBIENTE DE DESENVOLVIMENTO ............................................... 22
   7. SISTEMAS E COMPONENTES EXTERNOS UTILIZADOS .............. 23
4. **TESTES** ...................................................................................................... 24
5. **MANUAL DO USUÁRIO** ............................................................................ 24
6. **CONCLUSÕES E CONSIDERAÇÕES FINAIS** ......................................... 25

**BIBLIOGRAFIA** .......................................................................................... 26

**GLOSSÁRIO** .............................................................................................. 27

**Prefácio**

Em uma era digital, onde todos e tudo estão quase sempre acessíveis, conectar dispositivos e torna-los “inteligentes” e mais dinâmicos é quase uma obrigação, ou tornar-se obsoleto ao longo dos tempos.

Pensando nisso, o Wiiz Home foi estudado e desenvolvido, para tornar tarefas corriqueiras e diárias, em algo totalmente prático, dinâmico e automatizado.

Sua versão 1.0 do sistema, traz automações em tarefas mais comuns utilizadas no dia a dia de um lar brasileiro; acender e desligar um interruptor, cozinhar, estacionar um automóvel e etc.

Sugestões e Comentários podem ser enviados para: [danilo.dutra1444@gmail.com](mailto:danilo.dutra1444@gmail.com)

1. **INTRODUÇÃO AO DOCUMENTO**
   1. **TEMA**

A temática a ser abordada neste projeto é baseada na Internet das Coisas, ou IOT (Internet of Things).

* 1. **OBJETIVO DO PROJETO**

O objetivo do projeto é simplificar pequenas e repetidas tarefas diárias, automatizando e dinamizando-as.

* 1. **DELIMITAÇÃO DO PROBLEMA**

A ideia do projeto é tornar as tarefas e coisas do dia a dia de uma pessoa em comum, em tarefas automatizadas, delimitando-se ao uso domiciliar.

* 1. **MÉTODO DE TRABALHO**

O código foi testado e construído na plataforma TinkerCad, por conta da limitação do projeto não ser físico, porém se aplica em perfeita forma em embarcados e arduinos.

A estrutura do algoritmo foi baseada em orientação ao objeto, utilizando-se de funções para a realização do loop automático do Arduino.

* 1. **ORGANIZAÇÃO DO TRABALHO**

O documento está dividido em partes detalhadas: Introdução; Descrição; Análise; Implementação; Testes; Implantação; Manual do Usuário e Conclusão.

1. **DESCRIÇÃO GERAL DO SISTEMA**
   1. **DESCRIÇÃO DO PROBLEMA**

É fato que tempo é dinheiro, e que tempo perdido não se recupera mais. Pensando nisso, foi desenvolvido este projeto, utilizando-se de eletrônica embarcada, analógica e digital para resolver os seguintes problemas:

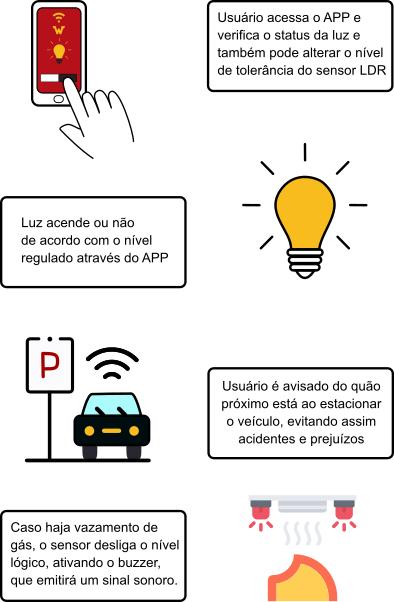
* Prevenção de incêndios
* Economia de energia
* Economia no uso do tempo pessoal
* Digitalização de informações analógicas

Portanto, é altamente recomendado o uso do protótipo, pelo baixo custo que oferece e pelo retorno financeiro e de tempo que o mesmo trás, beneficiando famílias e famílias.

* 1. **PRINCIPAIS ENVOLVIDOS E SUAS CARACTERÍSTICAS**

**USUÁRIOS**

O Wiiz Home é indicado para o uso residencial.



* 1. **DESENVOLVEDORES DO PROTÓTIPO**

O protótipo foi desenvolvido por Danilo Faria Dutra, afins de estudos e projeto semestral para o curso bacharelado em Ciências da Computação, cursado na Faculdade Bilac.

* 1. **REGRAS DE NEGÓCIO  
       
     Próximas possíveis atualizações:**Existe uma gama de possibilidade para expansão de áreas neste:
* Tranca inteligente
* Identificação pessoal de cada morador
* Câmeras automáticas com ativação por movimento
* Identificação de visitantes
* Sensor de “invasão” de pets (avisa quando o pet entra nos domínios da casa)
* Alarmes de afazeres domésticos

São inúmeras as possibilidades que podem ser incluídas no projeto, tudo depende da necessidade de casa residência.

1. **ANÁLISE E DESIGN**
   1. **ARQUITETURA DO SISTEMA**

“O ATmega328 é um [microcontrolador](https://pt.wikipedia.org/wiki/Microcontrolador)  tipo chip único criado pela [Atmel](https://pt.wikipedia.org/wiki/Atmel) na família [megaAVR](https://pt.wikipedia.org/wiki/Atmel_AVR).” (Fonte: Wikipédia)

”O Atmel [de 8 bits](https://pt.wikipedia.org/wiki/8_bits) [AVR](https://pt.wikipedia.org/wiki/Atmel_AVR) é baseado em [RISC](https://pt.wikipedia.org/wiki/RISC), o microcontrolador combina 32 kB ISP [flash](https://pt.wikipedia.org/wiki/Mem%C3%B3ria_flash) de memória de leitura-enquanto-grava, 1 kB [de EEPROM](https://pt.wikipedia.org/wiki/EEPROM), 2 kB [de SRAM](https://pt.wikipedia.org/wiki/SRAM), 23 linhas de I/O de propósito geral, 32 [registradores](https://pt.wikipedia.org/wiki/Registrador_(inform%C3%A1tica)) para diversos propósitos, 3 timer/[counters](https://pt.wikipedia.org/wiki/Contador_(eletr%C3%B4nica)) flexíveis com comparador de modos, [interrupções](https://pt.wikipedia.org/wiki/Interrup%C3%A7%C3%A3o_de_hardware) internas e externas, serial programável [USART](https://pt.wikipedia.org/wiki/USART), 1 interface I²C [SPI](https://pt.wikipedia.org/wiki/Serial_Peripheral_Interface) porta serial, [conversor A/D](https://pt.wikipedia.org/wiki/Conversor_anal%C3%B3gico-digital) de 6-canais e 10-bits (8-canais nos encapsulamentos [TQFP](https://pt.wikipedia.org/wiki/Quad_Flat_Package) e [QFN](https://pt.wikipedia.org/wiki/Quad_Flat_No_leads)/[MLF](https://pt.wikipedia.org/wiki/Quad_Flat_No_leads)), [temporizador do watchdog](https://pt.wikipedia.org/wiki/Watchdog_timer) interno programável com [oscilador](https://pt.wikipedia.org/wiki/Oscilador_eletr%C3%B3nico), e 5 modos de economia de energia selecionáveis por software. O dispositivo opera entre 1,8 e 5,5 volts. O dispositivo alcança a taxa de transferência aproximando-se de 1 [MIPS](https://pt.wikipedia.org/wiki/MIPS_(medida)) por MHz.” (Fonte: Wikipédia)

Dados técnicos sobre o Atmega328 da Microchip:

|  |  |
| --- | --- |
| **Parâmetro** | **Valor** |
| Tipo de CPU | 8-bit AVR |
| Desempenho | 20 MIPS às 20 MHz |
| Memória Flash | 32 kB |
| SRAM | 2 kB |
| EEPROM | 1 kB |
| Contagem de Pinos | 28 pinos PDIP, MLF, 32 pinos TQFP, MLF |
| Frequência máxima | 20 MHz |
| Número de canais de toque | 16 |
| Hardware de Aquisição QTouch | Nenhum |
| Máximo de pinos de I/O | 23 |
| Interrupções externas | 2 |
| Interface USB | Nenhum |

(Fonte: Wikipédia)

Caso o projeto seja implementado em um Arduino Mega no modelo 2560 R3, que utiliza o microcontrolador da Microchip Atmega2560, a expansão de portas digitais irá para 53, possibilitando a expansão de sensores e módulos, aumentando assim a capacidade de automação do projeto.  
  
Outra possibilidade também seria a utilização de CIs 8 Bit Shift Register 74HC595, da fabricante ***Texas Instruments*** ou similares de outras fabricantes, ou também o uso de módulos de expansão de I/O (Input, Output) de portas Analógicas/Digitais no Arduino Uno R3.

* 1. **DADOS DA COMPILAÇÃO NA IDE:**

O sketch usa 7610 bytes (23%) de espaço de

armazenamento para programas. O máximo são

32256 bytes.

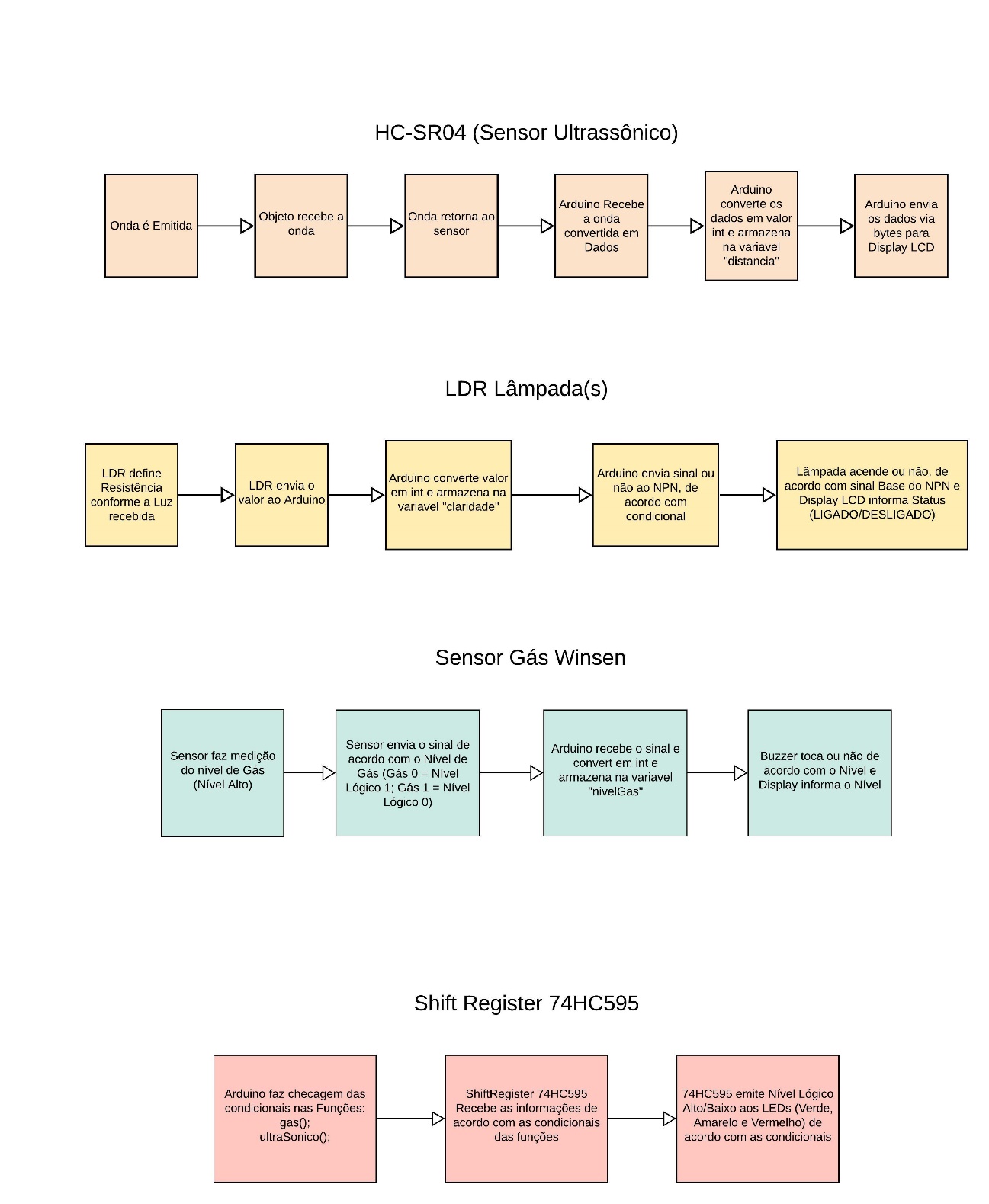
Variáveis globais usam 385 bytes (18%) de memória dinâmica, deixando 1663 bytes para variáveis locais. O máximo são 2048 bytes.

* 1. **DIAGRAMA DE BLOCOS**

São quatro processos identificados pelo Sketch do Wiiz Home, são eles:

* Sensor de distância Ultrassônico
* Sensor Nível de Gás Winsen
* Sensor de Luminosidade medido pelo LDR
* Registrador de 8 bit 74HC595

**Diagrama de Blocos - Wiiz Home**

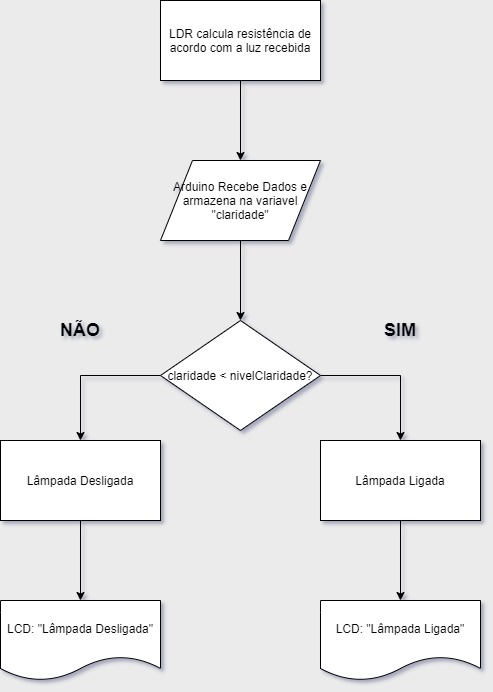


Dois dos processos do diagrama, Sensor de Gás e Sensor Ultrassônico, mandam informações ao Registrador 8-bit 74HC595, indicando visualmente, através dos LEDs. Verde, Amarelo e Vermelho, o nível de cada medição, sendo basicamente intuitivo; **Verde**: Bom; **Amarelo**: Alerta e **Vermelho**: Ruim.

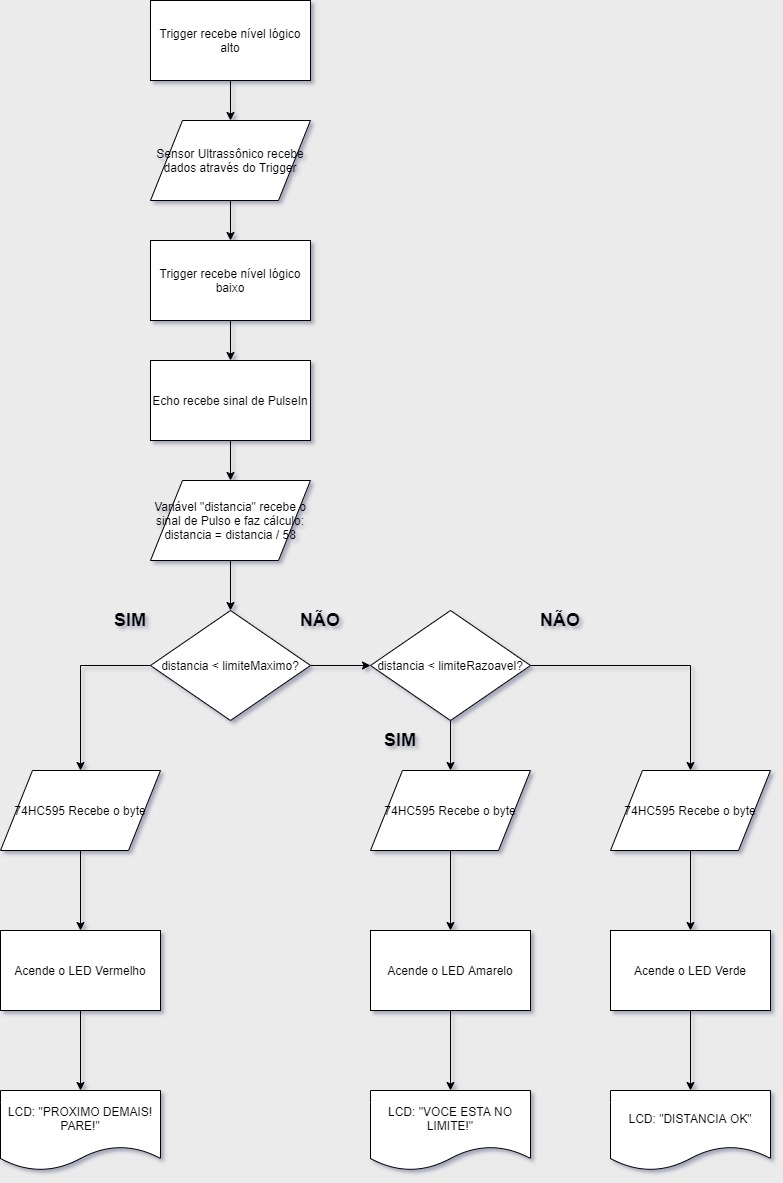
* 1. **FLUXOGRAMA**

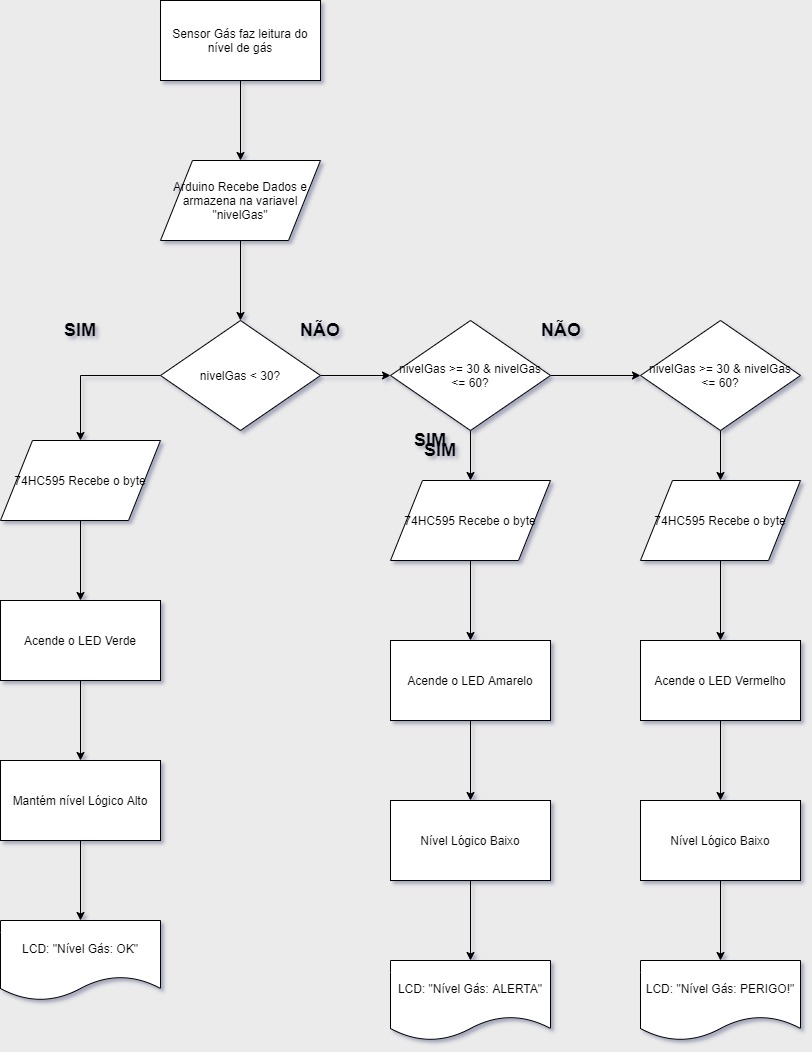
Assim como o Diagrama de Blocos, o Fluxograma segue o padrão dos 3 sensores: LDR, Gás e Ultrassônico.

**Fluxograma do Sensor LDR:**



**Fluxograma do Sensor Ultrassônico:**

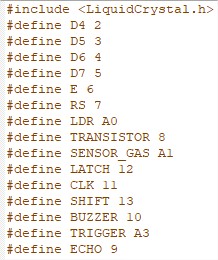


**Fluxograma do Sensor de Gás:**  
  


* 1. **CÓDIGO FONTE**

O código é construído da seguinte forma, seguindo a ordem:

1. PRÉ-ALOCAÇÕES: Constantes e Bibliotecas na MEMÓRIA DO PROCESSADOR (#INCLUDE e #DEFINE).
2. Declaração do OBJETO LCD, utilizado junto a biblioteca
3. Declaração de variáveis
4. Configuração das constantes e inicialização do objeto criado (Setup)
5. Construção das funções
6. Inicialização do Loop
   * 1. **PRÉ-ALOCAÇÕES**

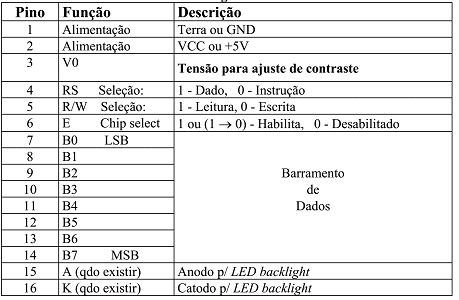
****

Utilizando-se de pré-alocações, foi dado a preferência de alocar-se as constantes desta forma.

**#include <LiquidCrystal.h>:** chama a biblioteca que faz o gerenciamento do objeto LCD, para o bom funcionamento do módulo LCD de 16 colunas e 2 linhas.

**#define D4, #define D5, #define D6, #define D7, #define E, #define RS:** Define as entradas para envio de dados para o módulo LCD, em constantes, utilizando-se do pré-alocador ***#define.***

***Interface de conexão externa:***

**

*Fonte (Embarcados*)

**#define LDR:** Define como entrada, a porta analógica A0, para leitura de dados, a resistência do LDR.

**#define TRANSISTOR:** Define como saída, a porta digital 8, para envio de corrente ao transistor.

**#define SENSOR\_GAS:** Define como entrada, para leitura de dados, a porta analógica A1 (faz a medição do nível de Gás).

**#define LATCH, #define CLK, #define SHIFT:** Define as portas digitais como saída para fazer a troca de informações (FLIP-FLOP), ao Shift-Register (Registrador 8-bit).

**#define BUZZER:** Define como saída, a porta digital 10, utilizada em PWM (Pulse Width Modulation), para enviar frequências ao Buzzer Ativo.  
  
**#define TRIGGER, #define ECHO:** Define como entrada, a porta digital 9 e a porta analógica A3, para recebimento de bytes, que definem o tempo de retorno dos sinais emitidos pelo sensor, que convertidos matematicamente, retornam em centímetros a distancia dos objetos que por ele passam ou se aproximam.

* + 1. **DECLARAÇÃO DO OBJETO LCD**

****

A biblioteca ***“LiquidCrystal”*** facilita e dinamiza o uso do objeto LCD, pois com alguns métodos como; print(), begin(), setCursor(), clear(), setBackLight(), dentre outros, conseguimos manipular facilmente o Módulo LCD.

**LCD:** É criado e endereçado o objeto ***“LCD”***, para manipular os dados enviados ao Módulo.

**Alguns Métodos utilizados:**

* **print():** Envia os dados a serem impressos no display.
* **begin():** inicializa a biblioteca LiquidCrystal.
* **setCursor():** Define em qual linha e coluna os dados serão impressos no display.
* **clear():** limpa os dados impressos no display.
* **setBackLight():** define se a luz de fundo será ligada ou desligada.  
    
    
    
    
    
    
    
    
  + 1. **Declaração de variáveis**

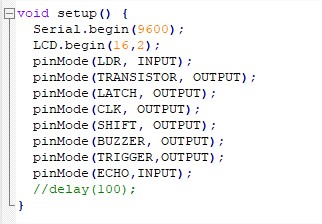
****

**Variáveis do tipo int:**

* **claridade:** recebe o valor medido pelo sensor LDR, para que seja comparada o nível de claridade com o nível do sensor LDR, acionando ou não o transistor, que faz o ligamento/desligamento da(s) lâmpadas.
* **nivelClaridade:** valor definido para comparar-se ao valor recebido pela variável ***“claridade”***.
* **nivelGas:** recebe o valor medido pelo sensor de Gás.
* **limiteMaximo:** valor definido como “máximo” para medir a distância do automóvel.
* **limiteRazoavel:** valor definido como “razoável” para medir a distância do automóvel.

**Variável do tipo float:**

* **distancia:** recebe o valor em centímetros, através de conversão matemática, para impressão no display LCD.  
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
  + 1. **Configuração das constantes e inicialização do objeto criado (Setup)**

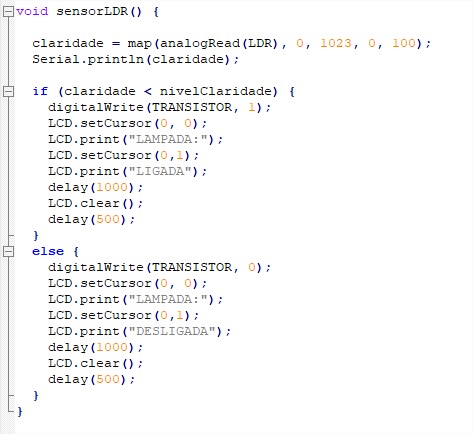
****

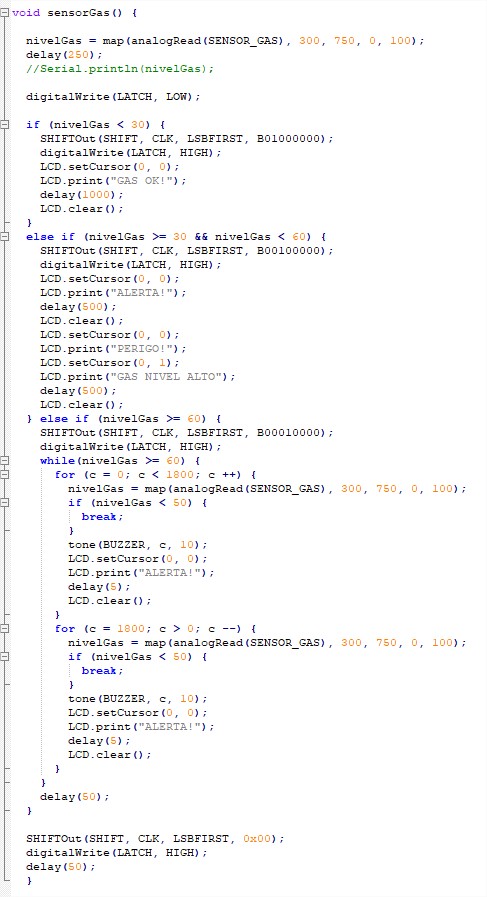
Nesta parte do código, é feita a configuração I/O das constantes e definido alguns parâmetros; objeto LCD e frequência de Leitura da porta Serial (opcional, para debug do código).

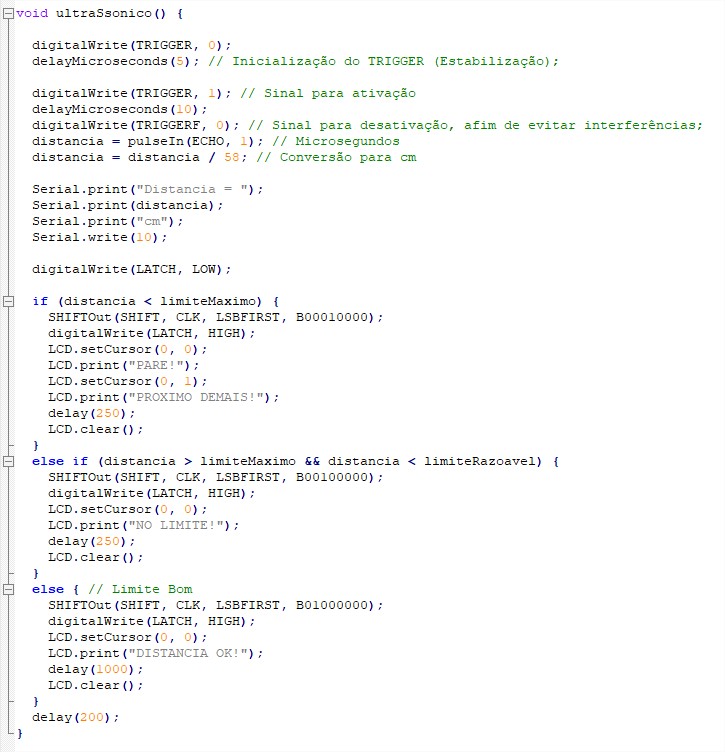
* **Serial.begin (9600):** Define o **Baud Rate** (9600), velocidade de comunicação de dados entre a Porta Serial e o Arduino.
* **LCD.begin (16, 2):** Define o objeto LCD, como tendo **16 Colunas e 2 Linhas** (Parâmetros externos já determinados fisicamente pelo Módulo).
* **pinMode (LDR, INPUT):** Define a porta de valor atribuída a constante **LDR**, como **ENTRADA**.
* **pinMode (TRANSISTOR, OUTPU):** Define a porta de valor atribuída a constante **TRANSISTOR**, como **SAÍDA**.
* **pinMode (LATCH, OUTPUT):** Define a porta de valor atribuída a constante **LATCH**, como **SAÍDA**.
* **pinMode (CLK, OUTPUT):** Define a porta de valor atribuída a constante **CLK**, como **SAÍDA**.
* **pinMode (SHIFT, OUTPUT):** Define a porta de valor atribuída a constante **SHIFT**, como **SAÍDA**.
* **pinMode (BUZZER, OUTPUT):** Define a porta de valor atribuída a constante **BUZZER**, como **SAÍDA**.
* **pinMode (TRIGGER, OUTPUT):** Define a porta de valor atribuída a constante **TRIGGER**, como **SAÍDA**.
* **pinMode (ECHO, INPUT):** Define a porta de valor atribuída a constante **ECHO**, como **ENTRADA**.
* **delay (100): Opcional**, caso queira garantir que todos os dados sejam gravados e para que garanta que haja tempo de o Setup ser concluído com sucesso.
  + 1. **Construções das funções**

Nesta parte do código, é construída as funções, de modo que se torne mais fácil a manutenção do código, caso haja necessidade de alterações, fique mais fácil a modificação esporádica.

Basicamente, as funções seguem o funcionamento apresentadas pelo Diagrama de Blocos e o Fluxograma.

**void sensorLDR():** 

**void sensorGas():**

**void ultraSsonico():** 

* + 1. **Inicialização do Loop**

O void loop() é a parte que utilizaremos a repetição das funções, de forma que as funções se repitam, sempre realizando novas checagens nos sensores.

* 1. **Ambiente de Desenvolvimento**

**Plataforma:**

Todo o projeto foi idealizado inicialmente para ser aplicado fisicamente, porém foi construído e testado na plataforma da Autodesk, TinkerCad.

O TinkerCad é uma plataforma online de simulação de modelagem 3D, que dentro dela, existe a possibilidade de simulação de Arduino.

**Linguagem:**

A linguagem utilizada para a programação do equipamento é C ++.

O paradigma utilizado é o estrutural, para que o Arduino possa ler constantemente e de forma linear o código.

* 1. **Sistemas e Componentes Externos utilizados:**

**Sistemas:**

O Sistema utilizado para simulação e prototipação do projeto em forma virtual, foi o TinkerCad.

Passando para a forma física e em aplicação prática, utilizaremos a IDE nativa da fabricante Arduino, hoje estando na **versão 1.8.12**, para podermos compilar e gravar o código junto ao Microcontrolador.

**Componentes Externos:**

Fisicamente, utilizaremos nesta primeira versão, os seguintes itens:

|  |  |
| --- | --- |
| **QTD** | **Item** |
| 1 | Arduino Uno R3 |
| 1 | 74HC595 |
| 1 | Resistor 4.7k ohms |
| 6 | Resistor 220r ohms |
| 1 | LED Amarelo |
| 1 | LED Verde |
| 1 | LED Vermelho |
| 1 | LCD 16x2 |
| 1 | Potenciômetro |
| 1 | Lâmpada |
| 1 | Fotorresistor (LDR) |
| 1 | NPN ou (Relé) |
| 1 | Fonte de Alimentação |
| 1 | Resistor 1k ohms |
| 1 | Piezo |
| 1 | Sensor de gás |
| 1 | HC-SR04 |

1. **TESTES**

Testes foram feitos em todos os sensores, para assim adequar ao número exato ou mais próximo possível de desejado, sempre podendo ser reajustado para se adequar a futuros usuários.

Por exemplo, no Sensor de Gás foi ajustado através do método **“map”,** o valor inicial, e o valor final, reajustado para medir de 0 a 100, totalizando um nível de 0% a 100%, sendo 0 totalmente sem fumaça, e 100 repleto de fumaça.

Os níveis de estabilidade e de limites foram ajustados também, sendo que abaixo de 30% de fumaça, estará em um nível aceitável, entre 30% e abaixo de 60%, estará em um nível de alerta, e acima de 60% estará em nível de perigo, acionando o Buzzer sonoro.

No Fotorresistor foi ajustado também com a função map, para que o LDR entenda como 0% totalmente claro e 100% totalmente escuro.

Sendo assim, a Lâmpada somente acenderá se o LDR atingir um nível de leitura ajustado como um nível de claridade baixo, podendo também ser ajustado facilmente.

O sensor Ultrassônico também ajustado e aplicado uma fórmula matemática para conversão dos dados em centímetros:

O sensor faz a leitura de pulso e converte e guarda em uma variável do tipo inteiro:

*distancia = pulseIn(echo, 1);*

*distancia /= 58;*

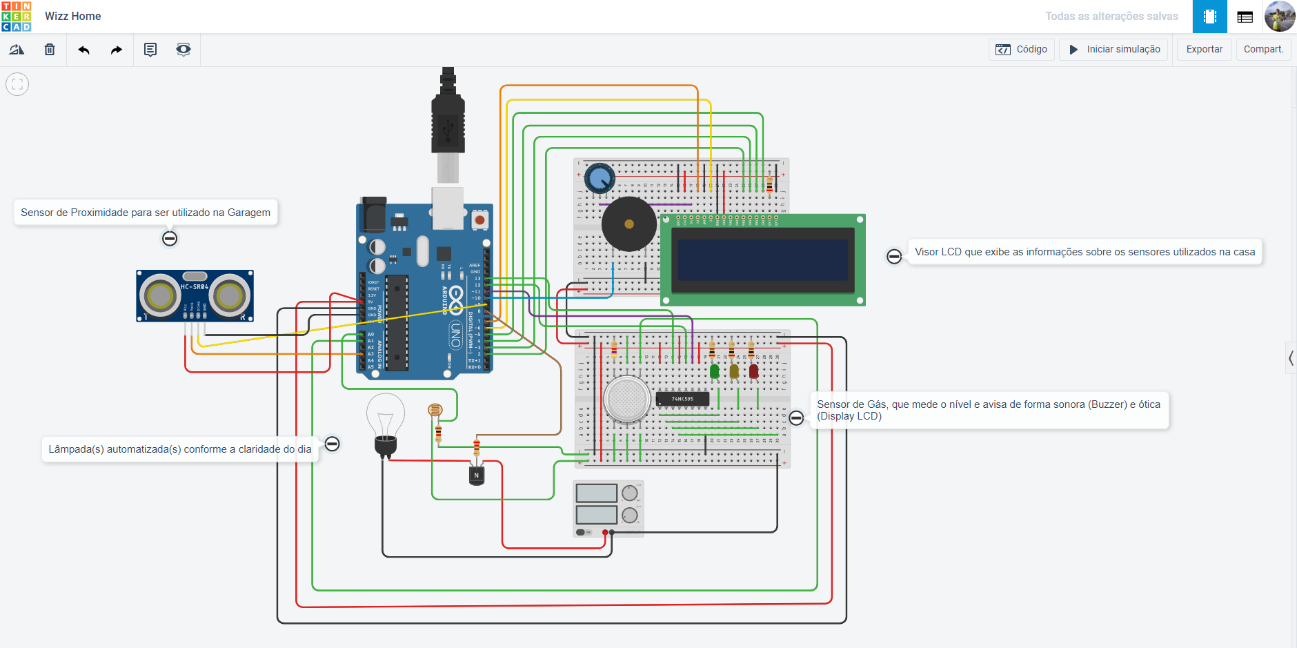
1. **MANUAL DO USUÁRIO**

**Restrições:**

Para o bom funcionamento do equipamento e de seus componentes, é recomendado alguns cuidados, como:

* Tensão recomendada ao Arduino: 5v a 12v.
* Tensão máxima recomendada ao NPN: 227v (substituição por Relé em projeto físico).
* Corrente máxima recomendada ao NPN: 1A (substituição por Relé em projeto físico).
* Tensão máxima permitida ao backlight do LCD e ao Display LCD: 5v.
* Corrente máxima permitida ao backlight do LCD: 20mAh.
* Tensão máxima permitida ao Sensor de Gás Winsen: 5v.
* Tensão máxima permitida ao 74HC595: -0.5v a 7v.
* Corrente máxima de saída por pino 74HC595: +- 35mA.
* Tensão e corrente máxima permitida ao HC-SR04 (Módulo ultrassônico): 5v e 2mA.
* Tensão de consumo média e corrente máxima de entrada nos LEDs: 1.7v a 2v e 20mA.
* Montar de acordo com a esquematização e prototipação proposta. Qualquer alteração é recomendada o teste de bancada antes de aplicar fisicamente.

Abaixo segue a prototipação realizada na plataforma TinkerCad:



1. **CONSIDERAÇÕES FINAIS**

Para finalizar, o Wiiz Home foi desenvolvido pensando em todo um esquema IOT a ser implantado a um lar comum, em que praticamente qualquer residência que construída anteriormente a chegada dessa nova possibilidade, possa ser contemplada com o dinamismo, a praticidade e facilidade em que a Internet

das Coisas trás.

Considerando a limitação de portas analógicas e digitais, e de bibliotecas personalizadas da plataforma TinkerCad, há inúmeras possibilidades reais para a expansão e alcance de realizações do Wiiz Home, como já citadas; identificação do morador, via código especifico para cada pessoa; acionamento e notificação de câmeras de segurança, via mensagens pelo smartphone, tornando câmeras comuns em câmeras inteligentes e integradas a Rede IOT; Portões automatizados para a entrada de automóveis, via frequência Rádio, acionadas por controles remotos ou acionamento a x metros de distância da casa.

Enfim, há inúmeras possibilidades para a expansão e atualização do projeto.

**BIBLIOGRAFIA**

ATMEGA328, Microchip. Informações Técnicas sobre. Disponível em <<https://www.microchip.com/wwwproducts/en/ATmega328>>. Acesso em: 16 de abril 2020.

ATMEGA328, Wikipedia. Informações Técnicas sobre. Disponível em <<https://pt.wikipedia.org/wiki/ATmega328>>. Acesso em: 16 de abril 2020.

ARDUINO MEGA 2560 R3, Embarcados. Introdução à Arduino MEGA 2560R3. Disponível em <<https://www.embarcados.com.br/arduino-mega-2560>>. Acesso em: 16 de abril 2020.

ARDUINO MEGA 2560 R3, Texas Instruments. Datasheet do ATMEGA 2560. Disponível em <<http://www.ti.com/lit/ds/symlink/sn74hc595.pdf>>. Acesso em: 17 de abril 2020.

SHIFT REGISTER 74HC595, Texas Instruments. Datasheet do Registrador 8-bit 74HC595. Disponível em <<https://www.ti.com/lit/ds/symlink/cd74hc595.pdf>>. Acesso em: 18 de abril 2020.

DO BIT AO BYTE, Como utilizar o Shift Register 74HC595. Disponível em <<https://www.dobitaobyte.com.br/como-usar-o-shift-register-74hc595>>. Acesso em: 19 de abril 2020.

BR ARDUINO, Aumentando as portas digitais do Arduino. Disponível em <<https://br-arduino.org/2015/01/aumentando-as-portas-digitais-do-arduino-com-o-ci-74hc595-shift-register.html>>. Acesso em: 19 de abril 2020.

EMBARCADOS, Entenda o funcionamento do Módulo LCD. Disponível em <<https://www.embarcados.com.br/modulo-de-display-lcd>>. Acesso em: 20 de abril 2020.

ARDUINO.CC, Referencia da Biblioteca LiquidCrystal. Disponível em <<https://www.arduino.cc/en/Reference/LiquidCrystalConstructor>>. Acesso em: 20 de abril 2020.

BAU DA ELETRONICA, Conhecendo a Biblioteca LiquidCrystal. Disponível em <<http://blog.baudaeletronica.com.br/conhecendo-biblioteca-liquidcrystal>>. Acesso em: 21 de abril de 2020.

PAULO TRETIN, Baud Rate: Diferença entre Bit Rate e Baud Rate. Disponível em <<https://www.paulotrentin.com.br/eletronica/diferenca-entre-bit-rate-e-baud-rate>>. Acesso em: 22 de abril de 2020.

AUTODESK, Plataforma TinkerCad. Disponível em <<https://www.tinkercad.com>>. Acesso em: 14 de abril 2020.

ARDUINO.CC, IDE de Arduino. Disponível em <<https://www.arduino.cc/en/Main/Software>>. Acesso em: 30 de abril 2020.

**GLOSSÁRIO**

* **IOT:** Internet das Coisas, revolução tecnológica que integra itens utilizados no dia a dia à rede mundial de computadores.
* **TinkerCad:** Plataforma de simulação online da Empresa AutoDesk
* **Arduino:** Plataforma de prototipagem eletrônica de hardware livre, projetada com microcontroladores Atmel.
* **Microcontrolador:** Pequeno computador (**SoC**) em um único circuito integrado, no qual contém um núcleo de processador, memória, periféricos programáveis de entrada e saída.
* **ATMEGA328:** Microcontrolador fabricado pela empresa ATmel.
* **ATMEGA 2560:** Microcontrolador fabricado pela empresa ATmel**.**
* **ISP:** ISP (**In-system programming**) ou ICSP (**In-Curcuit Serial Programming**), É um protocolo de comunicação. Este protocolo é usado na maioria dos gravadores existentes para a maioria dos microcontroladores. Neste protocolo ustiliza-se apenas 05 sinais:
* **VDD**(+ Alimentação)
* **GND** (-Alimentação)
* **VPP** (Tensão de gravação)
* **CLOCK** (Sinal sincronismo)
* **DATA** (é o dado propriamente dito)
* **SPI: Serial Peripheral Interface** é um protocolo de informação síncrono utilizado por microcontroladores para a comunicação rápida entre um ou mais dispositivos a curta distância.
* **EEPROM: Electrically-Erasable Programmable Read-Only Memory**, é um tipo de memória não-volátil usada em computadores e outros dispositivos eletrônicos para armazenar pequenas quantidades de dados que precisam ser salvos quando a energia é removida, por exemplo, dados de configuração do dispositivo.
* **USART:** USART se refere a **Universal Synchronous Asynchronous Receiver Transmitter**, significando Transmissor/Receptor Universal Síncrono e Assíncrono. É um formato padrão para comunicação de dados de forma serial.
* **SRAM:** A memória estática de acesso aleatório é um tipo de memória de acesso aleatório semicondutor que usa circuitos de trava biestáveis ​​para armazenar cada bit. A SRAM exibe remanência de dados, mas ainda é volátil no sentido convencional de que os dados acabem sendo perdidos quando a memória não é alimentada.
* **I/O:** Se refere a INPUT/OUTPUT (Entrada/Saída).
* **A/D:** Se refere a Analógico/Digital.
* **Shift Register:** Shift Register ou **Registrador de Deslocamento, e**m eletrônica digital um registrador de deslocamento é um conjunto de registradores configurados em um arranjo linear de tal forma que a informação é deslocada pelo circuito conforme ele é ativado.
* **CI:** Se refere a Circuito Integrado.
* **Sketch:** Esboço (tradução literal), é onde se realiza o projeto do Arduino e consiste tipicamente em duas partes (rotinas): a rotina de setup, que inicializa o sketch, e a rotina de loop, que normalmente contém o código principal do programa (é como a função main() da linguagem C).
* **LDR:** Se refere a Light Diod Resistor ou Fotorresistor.
* **Int:** Int(integer) ou inteiro. Variável numérica do tipo inteiro.
* **Float:** Variável númerica com ponto flutuante.

* **INPUT:** Entrada. Utilizado para entrada de sinais ou dados.

* **OUTPUT:** Saída. Utilizado para saída de sinais ou dados.
* **delay:** efeito temporizador, criando espaços de tempos em segundos, milissegundos, microssegundos etc.
* **Paradigma de linguagem:** Paradigma de programação é um meio de se classificar as linguagens de programação baseado em suas funcionalidades. As linguagens podem ser classificadas em vários paradigmas. Um paradigma de programação fornece e determina a visão que o programador possui sobre a estruturação e execução do programa.
* **backlight:** Luz interna utilizada pelo Display de LCD.