Curso de Ciência da Computação

Universidade Estadual do Mato Grosso do Sul

Projeto de Sistema Embarcado para Reconhecimento de Palavras Ditas

Felipe da Rocha Moralles Guterres

MSc. André Chastel Lima

(Orientador)

Dourados– MS

2017

Projeto de Sistema Embarcado para Reconhecimento de Palavras Ditas

Felipe da Rocha Moralles Guterres

Este exemplar corresponde à redação final da monografia da disciplina Projeto Final de Curso devidamente corrigida e defendida por mim Felipe da Rocha Moralles Guterres e aprovada pela Banca Examinadora, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Dourados, 13 de Novembro de 2017

Prof. MSc. André Chastel Lima

(Orientador)

Curso de Ciência da Computação

Universidade Estadual do Mato Grosso do Sul

Projeto de Sistema Embarcado para Reconhecimento de Palavras Ditas

Felipe da Rocha Moralles Guterres

Novembro de 2017

Banca Examinadora:

Prof. MSc. André Chastel Lima (Orientador)

Sistemas de Computação - UEMS

Prof. Dr. Osvaldo Vargas Jaques

Inteligência Artificial - UEMS

Prof. Dr. Rubens Barbosa Filho

Redes e Comunicação de Computadores - UEMS

Dedico este trabalho a minha família, porque sem eles não teria chegado até aqui.

AGRADECIMENTOS

Agradeço, primeiramente, á minha família por fornecer todo apoio necessário e me incentivando durante todos esses anos que estive na universidade.

De agradecer ao professor André Chastel Lima, por toda dedicação e empenho durante a realização deste trabalho.

Agradeço a todo corpo docente do curso de Ciência da Computação por todo conhecimento adquirido. A secretária acadêmica pela atenção dedicada a minha vida acadêmica.

Agradeço aos meus colegas e amigos, pelos momentos de alegrias, e por sempre me ajudarem nos momentos mais difíceis.

RESUMO

Desde o surgimento dos sistemas embarcados diversas áreas de estudos e aplicação puderam utilizar de seus benefícios. Uma forma lúdica para criar aplicações para sistemas embarcados é utilizar componentes que forneçam uma estrutura inicial de hardware e software que seja de fácil utilização. Os microcontroladores da Raspberry Pi conseguem fornecer essas características, sendo que sua placas suportam até versões compactas de sistemas operacionais. Com esse tipo de *hardware* é possível criar diferentes tipos de projetos para sistemas embarcados. Uma forma de interação bastante usual para utilização de sistemas embarcados, é por meio de fala, então o sistema deve reconhecer o que foi dito pelo locutor, realizando a interação com o sistema. Para o reconhecimento de palavras, precisa ser estudados formas que permitem a interação do locutor com o sistema, algumas técnicas e bibliotecas devem ser avaliadas e testadas para diferentes tipos de palavras, tons vocais e o equipamento que será utilizado para a aplicação.

Palavras Chaves: Sistema Embarcado; Raspberry Pi; Reconhecimento de fala; Assistente de Voz; MFCC .

Abstract

Since the emergence of embedded systems various areas of study and application to use its benefits. One playful way to build applications for embedded systems is to use components that provide an easy-to-use hardware and software initial structure. Raspberry Pi's microcontrollers are able to deliver these features, and their boards support even compact operating system versions. With this type of hardware it is possible to create different types of projects for embedded systems. A very usual form of interaction for the application of embedded systems, is through speech, then the system of recognition of work, through an interaction with the system. For word recognition, forms need to be studied that allow the speaker to interact with the system, some techniques and libraries must be evaluated and tested for different types of words, tons of vocals and equipment that are used for an application.

Keywords: Embedded System; Raspberry Pi; Speech Recognition; Voice Assistant; MFCC.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CPU	Central Processing Unit
CSI	Camera Serial Interface
DSI	Display Serial Interface
GPU	Graphics Processing Unit
GPS	Global Positioning System
HDMI	High-Definition Multimedia Interface
PCB	Printed Circuit Board
SD	Secure Digital Card
SE	Sistema Embarcado
SO	Sistema Operacional
SOC	System On a Chip
USB	Universal Serial Bus
MFCC	Mel-Frequency Cepstral Coefficients

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

1 INTRODUÇÃO

Com o surgimento das aplicações para os sistemas embarcados, a sociedade desde então está se adaptando a utilizar seus recursos, estes que estão em constante mudança e aperfeiçoamento. Os primeiros sistemas embarcados, eram sistemas elétricos mais simples construído a nível de portas lógicas, a forma de interação desses sistemas, quando possuíam eram realizados por linha de comando, um usuário comum dificilmente entenderia as respostas dessas aplicações. As arquiteturas e organizações para os sistemas embarcados evoluíram, assim como as linguagens de programação. Os sistemas embarcados evoluíram possuindo interfaces diversas para iteração com usuário.

Os sistemas embarcados estão presentes desde sistemas de aplicações simples, como os temporizadores de semáforos, até sistema mais robusto como controle de navegação aérea. Com o crescente número de especialistas e entusiastas de sistemas embarcados, se fazia necessário a criação de um componente que fosse lúdico e pudesse fornecer a teoria e prática para o desenvolvimento de um sistema embarcado. Assim em 2006, o Engenheiro Eben Upton e seu grupo de pesquisa criaram as primeiras placas do Raspberry Pi, esta tinha por finalidade o aprendizado para crianças, jovens e entusiastas, estas placas eram concebidas como microcomputadores, por possuir a maioria das interfaces de um computador pessoal, e está devia ser de fácil aquisição aos usuários. (HEIN, 2013)

O Raspberry Pi proporciona ao usuário diversos formas de interação, entre elas o reconhecimento de palavras, estabelecendo uma forma de comunicação do homem e sistema. Alguns projetos permitem ao usuário criar sistemas utilizando de comando de voz, podendo fazer aplicações para automação residencial e empresarial, assistente pessoal, sistemas para segurança e demais aplicações possíveis para reconhecimento de palavras.

Para o Raspberry Pi reconhecer as palavras ditas deve ser utilizados métodos e equipamentos que permitem a captura de áudio, assim processando cada palavra dita para uma operação estabelecida no sistema.

1.1 Objetivo

O objetivo desse projeto é desenvolver um protótipo para sistema embarcado que faça o reconhecimento de palavras ditas. Por fim, verificando a taxa de acertos para palavras criadas na aplicação

1.1.1 Objetivos Específicos

a) Estudar Sistemas Embarcados: o que é, processo de criação, documentação.

b) Estudar o aplicativo de Comparação por MFCC desenvolvido no Projeto Final de Curso (Estudo e Análise de Métodos para Reconhecimento de Palavras Ditas) dos autores Raiza Artemam de Oliveira e Willian de Sousa Santos a fim de migrá-las para um SE.

c) Testar a melhor solução para o reconhecimento de palavras para um sistema embarcado.

1.2 Problemática

Com a crescente miniaturização dos projetos de sistemas convencionais, faz-se necessário uma tecnologia que possibilite a redução arquitetural e organizacional, os sistemas embarcados propiciaram a evolução desses sistemas.

Um dos campos a ser aprofundados sobre sistema embarcados, é a possibilidade do emprego de comandos de voz realizar interações com a aplicação. Assim, o sistema embarcado deve possuir uma estrutura que permita o reconhecimentos de palavras ou frases, que viabilizam as operações da aplicação.

1.3 Justificativa

Devido existem inúmeros tipos de arquiteturas para sistemas embarcados, foi utilizado Raspberry Pi por ser uma arquitetura de fácil aprendizado e utilização, e baseado nele foi estudado a viabilização de um aplicação de reconhecimento de palavras. Para isso foi empregado plataformas e técnicas que possibilitam o reconhecimento de palavras para um sistema embarcado. Com esse estudo inicial, fornece uma base para utilização das técnicas estudadas e avaliadas para diversos segmentos de atuação de sistemas embarcados que utilizam iterações por comando de voz.

1.4 Metodologia

A metodologia adotada para a realização deste projeto consistiu em pesquisa em livros, sites, artigos, fóruns sobre sistemas embarcados e técnicas de implementação, pesquisa de bibliotecas de reconhecimento de fala. Após a fase de pesquisa foi realizado o teste para a melhor aplicação para o projeto, foram instalados e configurados a plataforma

do Jasper, a biblioteca da Google Assistant SDK embedded e o Estudo e análise de métodos para Reconhecimento de palavras ditas. Após estes testes preliminares foram feitos testes para o reconhecimento de palavras ditas por comparação do MFCC.

1.5 Roteiro

No capítulo 2 é explicado o conceito de sistemas embarcados, mencionando suas características e importância. No capítulo 3 é abordado sobre a arquitetura e utilização do Raspberry Pi. No capítulo 4 são apresentando as bibliotecas que foram pesquisadas e testadas neste trabalho. No capítulo 5 é demonstrado o que foi necessário para instalar e testar o Raspberry e as bibliotecas abordadas no capítulo anterior. No capítulo 6 é realizado os testes e resultados obtidos neste trabalho. No capítulo 7 é realizada as considerações finais do trabalho e sugestões para futuros trabalhos.

2 SISTEMAS EMBARCADOS

Um sistema é categorizado como embarcado quando é encapsulado para apenas uma finalidade, isso quer dizer, que o sistema embarcado (SE) é construído e desenvolvido para um propósito específico. Sistemas desse tipo interagem continuamente no meio no qual está aplicado por meio dos sensores e atuadores. O SE podem interagir ou não com usuários humanos, dependendo de seu objetivo. (CHASE, 2007)

2.1 História

O surgimento dos sistemas embarcados começou no final da década de 60, com as novas necessidades do mercado e associada com os avanços tecnológicos da microeletrônica, grandes empresas do ramo perceberam um mercado promissor nessa tecnologia, por sua vez, otimizando os componentes eletrônicos para esse segmento.

A evolução dos microcontroladores, que são um conjunto de diferentes tipos *chips* que integrados possuem quase todos os aspectos e componentes de um computador pessoal, como: processador, memória RAM, periféricos de entrada e saída, conversor Analógico\Digital, etc. Mas diferente de um computador pessoal (*desktop* e *notebook*) que pode atender diferentes tipos de necessidades do usuário. Os microcontroladores por possuir recursos limitados até podem possuir execuções de diferentes operações, mas seu propósito é para operar geralmente apenas uma tarefa exclusiva.

Na mesma época, surgiram os microprocessadores que foram muito importantes para os SEs, que é basicamente um circuito integrado que realiza as funções de cálculo e tomada de decisões, este executa as funções que os outros componentes externos o solicitam, como o microprocessador não trabalha individualmente e nem possui memória para que possa ser programado, trabalha como um processador de informações que os outros componentes (memórias, periféricos e conversores) os envia para execução.

O primeiro sistema embarcado digital que se tem conhecimento foi AGC (Apollo Guidance Computer). Era um sistema responsável pelo controle das espaçonaves do projeto Apollo. O AGC não possuía processador, era construído com portas NOR (operação lógica que opera com dois valores lógicos), este sistema foi responsável por levar o homem ao espaço nas décadas de 60 e 70. Sua utilização foi de extrema importância para o projeto Apollo, para que os cálculos exatos de navegação fossem realizados no espaço, independentemente dos cálculos de rotas que eram feitos na Terra.

Maioria dos primeiros programas eram feitos em linguagem de máquina (*Assembler*),

já nas décadas seguintes começavam a surgir bibliotecas de códigos direcionados para os SEs para determinados processadores. Atualmente os sistemas podem ser desenvolvidos em linguagens de alto nível ou até mesmo possuir versões especiais de sistemas operacionais. (EMBARCADOS, 2013)

2.2 Importância

Atualmente, os sistemas embarcados são a categoria de sistemas processados de maior utilização no mundo, superando em número os PCs, *notebooks*, servidores e semelhantes. Está presente em uma categoria de sistemas de computação dita oculta, no qual esses sistemas quase não são notados pelos humanos. Os vários sistemas deste tipo podem fazem parte de diversos segmentos de extrema importância na vida humana, desde simples sistema até os mais sofisticados. Um exemplo do nosso cotidiano é o elevador que possui uma interface com inúmeros botões, onde cada um representa um andar ou alguma ação do elevador, o sistema do elevador então faz o controle das paradas dos andares ou interrupção emergencial, sendo que os passageiros não possuem a informação de como o sistema comanda cada ação. (MORIMOTO, 2007)

Os SEs também podem ser utilizados englobando um sistema maior, por exemplo, o transponder e o sistema ADS-B que fazem parte do Sistema Aéreo de Alerta e Controle. O transponder é um dispositivo de comunicação de rádio, cujo objetivo é receber as informações do radar e transmitir para o controle de tráfego aéreo, nessas informações constam a identificação da aeronave, velocidade, altitude e a posição. O ADS-B é equipamento semelhante ao funcionamento do transponder, só que ele utiliza o sistema de satélite por GPS (Global Positioning System). (MARINHO, 2015)

2.3 Característica

O sistema embarcado geralmente é formado por um microcontrolador e um software (firmware) dedicados especificamente para o sistema no qual será projetado, alguns fatores são importantes para definir um SE.

- **Dimensões físicas**: desde do esboço inicial do projeto tem que haver atenção ao tamanho e peso do sistema desenvolvido. Com a crescente miniaturização dos equipamentos eletroeletrônicos. O tamanho e o peso são fatores importantes para a locomoção ou para o espaço físico onde será implantado o sistema. As dimensões reduzidas destes componentes impõem, ao sistema que a concepção da parte eletrônica sofra alguma restrição.

- **Consumo de energia**: quanto maior a autonomia do sistema, menor será sua necessidade de recarga e menor será seu consumo de energia, o que gera economia nos geradores elétricos móveis (ou células eletroquímicas) como as baterias e pilhas. E no caso de a alimentação for contínua a redução de energia elétrica.

  Os componentes dos SEs podem ser equipamentos eletrônicos móveis como um aparelho de GPS, neste caso são utilizadas como fonte de energia as células eletroquímicas, que possuem uma autonomia limitada para funcionamento. Essa é uma das principais exigências dos SEs, possuir um consumo mínimo de forma que a autonomia seja a máxima alcançável, para obter uma autonomia elevada é necessário que a frequência de relógio (*Clock*) opere em baixa frequência, fazendo com que o processador diminua a energia que é dissipada por seus circuitos. Neste caso deve haver um estudo de quais e o tamanho das *deadlines*[1] que o processador deverá executar, pois quantos menores forem as deadlines, menor será a necessidade de processamento pelo processador, fazendo com que ele possua uma menor frequência de relógio. Outro fator importante, é que o SE empregue o mínimo necessário de componentes elétricos a serem alimentados e que utilizem um sistema de gerenciamento de energia.

  Além dos sistemas que são alimentados por baterias, existem os sistema que são fixados ao ambiente onde operam, como controladores de produção, que são alimentados de forma contínua, neste caso é importante a redução de consumo de energia para caracterizar o sistema de forma econômica e ecológica.(GERVINI et al., 2003)

  As restrições de consumo podem interferir na lógica do desenvolvimento dos softwares destes sistemas. O consumo de energia para a execução de um programa varia com o número de instruções utilizadas, pois existem variações de consumo para cada tipo de instrução, geralmente as operações que envolvem maior consumo de energia estão relacionadas com as instruções de memória.

  O planejamento de software considerando uma organização bem reduzida e com instruções adequado de dados de memória permitem otimizações de consumo no sistema embarcado. A maior parte dos sistemas embarcados pode ser dividida em três categorias, baseado no seus consumos de energias: consumo móvel (por bateria), consumo fixo e sistemas de alta densidade (alta performance). (JUNIOR; WANNER; FRÖHLICH, 2006)

- **Resistência e Durabilidade**: muitos sistemas são projetados para operar em condições adversas (vibrações, calor, poeira, variações de tensão de alimentação, interferências eletromagnéticas, raios, umidade, corrosão, etc.). É necessário que o sistema conviva com essas condições sem sofrer interferências externas no ambiente no qual será empregado, assim deve haver um estudo sobre esse meio em que ficará

[1] *deadline* em termo computacional, representa o tempo máximo em que uma ou conjunto de operações devem ser executados

hospedado, e as possíveis soluções para aumentar a vida útil do sistema, um exemplo seria o revestimento do circuito por uma camada protetora. (III; STUBBERUD; WILLIAMS, 1981)

- **Arquitetura e Desenvolvimento**: a escolha do ambiente de desenvolvimento é outro fator determinante para o projeto podendo ser utilizado diferentes tipos de linguagens, compiladores, depuradores, ferramentas de *trace*, canais de *debug*. Dependendo dos dispositivos que serão utilizados, se forem mais simples e exclusivos para um tarefa, provavelmente terão que utilizar uma linguagem de baixo nível, neste caso o conhecimento sobre o dispositivo e a linguagem utilizada deve ser de grande perícia, pois o dispositivo trabalha quase em nível elétrico, onde um descuido pode ocasionar em perda dos dispositivos ou até mesmo do sistema inteiro.

  Uma arquitetura mais robusta, geralmente possui um custo mais elevado, que permita o desenvolvimento em linguagem de alto nível, ou até mesmo um sistema operacional completo como Raspberry Pi.(BANAKAR et al., 2002)

- **Tempo de execução**: é o tempo-real responsável pela execução das tarefas, cada tarefa possui um tempo crítico de execução, também chamado deadline, o sistema obrigatoriamente deve executar antes do *deadline*. Conforme o tipo tarefa o *deadline* pode ter tempos diferentes para cada aplicação, a execução de um programa em um tempo maior que o deadline, torna a aplicação de pouca utilidade e confiança. Existem dois tipos de execução para os sistemas embarcados;

  - *Soft Real-Time*: neste tipo de sistema o tempo de execução possui um ponto crítico, mas atrasos são toleráveis, que apenas degrada a qualidade da resposta do sistema, exemplos são, sistemas transmissão de áudio/vídeo, sistemas de medição(satélites, microscópios,etc.), sistemas de automação residencial e etc. A maioria das tarefas são executadas antes de atingir o tempo crítico. Existe ainda, a *Firm real-time* que também é semelhante a *soft real-time*, apenas possui restrições temporais firmes, mas permitindo algum atraso na execução;
  - *Hard Real-Time*: o tempo de execução é absolutamente crítico e não deve tolerar atrasos, um sistema que não cumprir o tempo menor que o *deadline* é considerado ineficaz para o seu tipo de tarefa, exemplos são, sistemas de médicos, sistemas de frenagem automotivo, sistemas de navegação, e etc. Nesta categoria o sistema deve garantir um tempo de execução determinístico, isso quer dizer, todas as operações devem ser executadas antes do *deadline*. (LI; YAO, 2003)

- **Heterogeneidade**: cada projeto de sistema embarcado pode envolver diversos componentes físicos, podendo customizar um componente já existente para uma alguma característica específica do projeto. As necessidades em termos de desempenho, de

interfaces e de armazenamento, entre outras, requer um planejamento característico para cada sistema. Com relação ao hardware, existem inúmeros tipos de microcontroladores com processamento de 4, 8, 16, 32 e 64 bits, além de DSPs (*Digital Signal Processors* - Processadores Lógicos Programáveis) e FPGAs (*Field Programmable Gate Array* - Dispositivos Lógicos Programáveis), a escolha de um destes componentes determina a capacidade de processamento disponível e, por consequência, deve considerar o custo, o tamanho, a variação de frequência de *clock* das diferentes arquiteturas.

Comparando a relação de *hardware* com a diversidade de usos de sistemas embarcados, o *software* embarcado também apresenta grande heterogeneidade tanto em termos funcionais como eventos regulares e irregulares, variações nas tolerâncias temporais, atividades sequenciais e concorrentes, reatividade, diversidade de controladores de dispositivos (*device drivers*) para interfaces e periféricos, diversidade de protocolos de comunicação. (STADZISZ; RENAUX, 2014)

A heterogeneidade em sistemas embarcados é uma consequência dos inúmeros propósitos, das variações de tecnologias e das restrições envolvidas no projeto. Este fator de padronização é um grande complicador para o desenvolvimento dos SEs, diferente do que ocorre em sistemas baseados em computadores pessoais.

- **Custos**: o desenvolvimento de sistemas embarcados sofre grandes imposições por redução de custos, pois praticamente todos os componentes que integram o sistema embarcado são produtos eletrônicos de consumo. O objetivo do mercado destes componentes é tornar o produto mais enxuto possível em termos de componentes de *hardware* e *software* para se obter menores custos na produção e na utilização dos mesmos.

  Estas reduções de custos desencadeia uma série de restrições sobre o planejamento do desenvolvimento dos sistemas embarcados, a fim de reduzir os custos do sistema em geral.

- **Recursos**: como resultado de todas restrições possíveis no planejamento do sistema embarcado como as restrições de custo, dimensão, consumo e o desenvolvimento do software, o planejamento do SE deve se limitar aos mínimos recursos possíveis para atendimento dos requisitos funcionais solicitados. Essas restrições podem ocasionar em limitações de memória volátil e de armazenamento, podem delimitar a capacidade de processamento, restringe a utilização de interfaces, assim como limita os recursos que o software poderá oferecer ao sistema, essas condições que definirão como será o planejamento e os objetivos para o sistema que estiver sendo desenvolvido.

- **Usabilidade**: os sistemas embarcados possuem grande apelo comercial por terem um custo reduzido,sendo utilizado em diversos segmentos de atuação. A maioria dos

processadores fabricados hoje em dia são utilizados em SEs. As aplicações dos sistemas embarcados estão gradativamente presentes em nosso cotidiano, em eletrodomésticos, no setor automotivo, em sistema controle de produção, equipamentos médicos, automação residencial e muitas outros segmentos. Os avanços tecnológicos e o aumento da capacidade de desempenho dos circuitos integrados, deram aos sistemas embarcados um grande poder de processamento, aperfeiçoando os SEs já existentes e desenvolvendo novas aplicações.

2.4 Tipos de Controles de Malha

Um sistema embarcado composto de vários componentes e periféricos é dito um sistema em malha, o controle dessa malha pode ser de duas formas, em malha aberta e a malha fechada.

No sistema em malha fechada, ou sistema retroalimentado, necessita-se das informações coletadas pelos sensores ou transdutores, enviando os sinais capturados para um controlador que comparará o sinal de saída do processo com um sinal de referência. Desta Comparação resultará em um erro que é utilizado para determinar o sinal de controle que será aplicado ao processo, o controlador envia o sinal aos atuadores que farão a correção do ambiente. Assim o erro deve ser recalculado para que seja reduzido a zero.

Em um sistema de malha aberta, existe um sinal de controle pré-definido, podendo ser uma condição fixa, como uma variável de tempo. Neste tipo de sistema de controle, não é utilizado um sensor que faria o monitoramento do meio. Quando o valor da condição é atingindo, o controlador envia a informação aos atuadores para fazer a ação de correção do meio.

Um exemplo de sistema embarcado para esses dois tipos de sistema de controle, é uma estufa que precisa manter aquecida as folhas de papel para impressão, evitando a umidade de forma que comprometa a qualidade das cópias. Nesse SE deve possuir como componentes um microcontrolador como unidade de controle, o atuador pode ser uma lâmpada incandescente e uma relê que fará o acionamento da lâmpada. Com o sistema de malha aberta, o microcontrolador deve estabelecer um tempo constante em que a lâmpada fique acesa e outro tempo que fique apagada para manter em uma temperatura adequada ao meio, como não há nenhum evento externo que interfira no acionamento da lâmpada, somente irá existir a contagem de tempo realizada pela unidade de processamento, que decidirá o estado enviado para relê para lâmpada (ligada ou desligada). (SILVA, 2000)

No sistema de malha fechada será necessário a inclusão de sensor de umidade, que coletará as informações sobre a umidade do ar no meio. Neste tipo de sistema, o sensor de umidade enviará os sinais de saída para a unidade de processamento (CPU), conforme

o sinal de referência utilizado pela *CPU*, tomará a decisão do estado enviado à relê, e subsequente a lâmpada.

Sempre que for desenvolvido um sistema embarcado deve-se ter conhecimento sobre o tipo de controle. Quando o sistema ou meio apresentam muitas variações de informações e de estado é aconselhável utilizar o sistema de malha fechada, pois existirá um sensor que fará a medição do processo, mas se o sistema apresenta características definitivas sobre os sinais de entrada e saída e não apresenta interferências externa pode-se utilizar o sistema de malha aberta. O sistema de malha aberta é mais fácil de ser projetado, visto que há poucas interferências de fatores externos. Os sistema de malha fechada são geralmente mais caros, pois utilizam mais periféricos e são sistemas mais robustos. Portanto no sistema de malha aberta a ação do controle é independente, assim os sinais de saída não causam interferência na ação do controle. Já no sistema de malha fechada, o controle depende dos sinais de saída, causando a interferência na ação do controle. Uma combinação de sistemas de controle de malha aberta e fechada é possível, se o projeto for bem modelado pode-se ter um ganho econômico e aumento de desempenho se fosse utilizado apenas um dos sistemas de controle.

2.5 Organização

Em Sistemas embarcado, deve-se considerar toda a arquitetura, englobando a arquitetura da plataforma computacional, as especificações do hardware, do microcontrolador (processador), do sistema operacional e o software de aplicação. Geralmente define-se que o SE é baseado em três camadas de abstração: *hardware*, núcleo operacional e a aplicação do projeto. Assim o Sistema Embarcado pode estabelecer um conjunto de tarefas pré-definidas da combinação do *software* e *hardware*, de forma que todos os seus recursos sejam otimizados para atender o propósito do projeto.

2.5.1 Hardware

Em SE geralmente possui a seguinte subdivisão de hardware: unidade central de processamento, memória e periféricos.

A unidade central de processamento (do inglês *Central Processing Unit* - CPU) é o responsável pela administração no processamento das instruções, fazendo realização dos cálculos de operações lógicas e aritméticas, controlando a tomada de decisões, eventos e interrupções de todos os dispositivos e tarefas que estão presentes no sistema.

A CPU é fisicamente um processador, sendo composta basicamente por três componentes:

- **Unidade Lógica e Aritmética (ULA)**, realiza um conjunto de operações necessárias à execução das instruções;
- **Unidade de Controle (UC)**, é a máquina de estados do processador, tendo a função de habilitar a memória central, memória externna e a as operações da ULA;
- **Memória Central**, é um conjunto de registradores, que possuem pouco espaço de armazenamento, mas possuem alta velocidade na transferência de informações, usada para armazenar resultados temporários.

A memória no sistema embarcado armazena dados e instruções que são vinculadas às operações da CPU. Na Arquitetura Harvard, que é mais utilizada em SEs, as instruções e os dados são fisicamente separados em memórias diferentes.

Os periféricos são interfaces que recebem e enviam informações à CPU. Em sistemas embarcados duas categorias de periféricos se destacam, os sensores e os atuadores. (CONTI, 2014)

O sensor faz a detecção de alguma medida física, e gera uma interrupção para unidade de processamento. O sensor deve estar sempre calibrado e funcionando corretamente, para fornecer informações confiáveis, sem alterar os dados do ambiente monitorado. Exemplos de sensores são, de distância (sonar, infravermelho), de temperatura (termistores), movimento (acelerômetros). Outro tipo de sensor é o transdutor, que além de poder fazer a leitura do ambiente, ainda pode converter sinal recebido em um outro tipo de sinal. Um exemplo de transdutor é o microfone que converte o som recebido em sinais elétricos.

Os atuadores operam comandos que controlam o ambiente com ações manuais, elétricos ou mecânicos, estes periféricos recebem uma informação da CPU e interferem no processo em controle, exemplo são: motores, *coolers*, luzes, aquecedores, etc.

Dependendo da necessidade do projeto poderá utilizar um único circuito integrado, também conhecido como SOC (*System-on-Chip*), é uma miniatura de sistema computacional em um microcontrolador, consiste de uma CPU, memória e portas de entrada e saída, é um circuito único integrado mais complexo e apresenta mais recursos. Por outro lado, existe um circuito dedicado em um módulo SOM (*System-on-Module*), que consiste em único módulo de *hardware*. Um SOM, combinado a um circuito que implementa interfaces específicas forma então o *hardware* do sistema embarcado. (GONÇALVES, 2008)

2.5.2 Microcontrolador

Os microcontroladores (processadores) podem ser classificados referentes pelo tamanho de processamento dos dados, que equivale ao tamanho dos registradores. Existem microcontroladores de pouco processamento, como os de 4 bits até os maiores de 64 bits,

os microcontroladores de 4 bits foram mais utilizados nos primeiros sistemas embarcados, e os de 64 bits são mais utilizados em *workstations*, servidores e computadores de grande porte, com variações de tamanho de memória, de poucos *kilobytes* a muitos megabytes.

Pode-se utilizar dois tipos de abordagens de arquitetura de processadores, a arquitetura RISC, ou Computador com um conjunto reduzido de instruções (do inglês, *Reduced Instructuion Set Computer*) que visa unidades de controle mais simples, rápidas e baratas. A pouca variedade nos tipos de instruções traz um importante benefício no ganho de desempenho com a *Pipeline*[2]. Outra característica dos processadores RISCs sobre o fato de terem um número reduzido de circuitos internos, que permite trabalhar em *clocks* mais altos.

A outra abordagem de arquitetura, são os processadores do tipo CISC, ou Computador com um conjunto complexo de instruções (do inglês, *Complex Instructuion Set Computer*) sendo capaz de executar centenas de instruções complexas diferentes, tornando o processador poderoso e versátil. Este tipo de arquitetura executa tarefas complexas como a Execução Fora de ordem e a execução superescalar. Na execução fora de ordem, ela analisa a sequência de instruções e busca aquelas que não possuem dependências, otimizando a *pipeline*. Com a execução superescalar, o processador possui um número maior *pipelines* que executam em paralelo em diversas unidades de execução, como unidades de pontos flutuantes e unidades de inteiros.

Atualmente os processadores que utilizam uma combinação das abordagens das arquiteturas RISC e CISC, criam componentes do tipo híbrido aperfeiçoando o poder e a velocidade de processamento. (EMBARCADOS, 2015)

Uma outra abordagem onde se necessita customizar e criar um microcontrolador, é classificado como um circuitos integrado aplicação específica ASICs (*Aplication Specific Integrated Circuits*), são circuitos que precisam de um processo de fabricação especial são constituídos por portas lógicas (*And*, *Or*, *Not* e *Flip-flops*) e necessitam de vários componentes externos para realização de uma função específica. Com a evolução dos PLDs (*Programmable Logic Devices*), simplificou e aperfeiçoou os circuitos do tipo ASICs, os PDLs são circuitos integrados que podem ser configurados pelo próprio desenvolvedor, não apresentando uma função lógica definida, até que seja configurada. Os PLDs foram os dispositivos eletrônicos que possibilitaram a implementação de lógica estruturada, podem ser classificados em:

[2] *pipeline* é uma técnica que consiste em dividir e alocar na memória central as próximas instruções a serem executadas, para execução múltiplas de instruções simultaneamente no mesmo processador

2.5.3 Sistema Operacional

O RTAI (*Real-Time Application Interface*) foi um projeto desenvolvido pelo departamento de Engenharia Aeroespacial do Politécnico de Milão, seu objetivo inicial era tornar disponível o desenvolvimento de atividades relacionadas a controle ativo de atuadores aerodinâmicos. O RTAI foi desenvolvido como variação do Linux para sistemas embarcados com restrições temporais, para uso em sistemas de tempo real, tendo baixa latência nas interrupções das tarefas.(DOUGLASS, 1997)

Uma característica importante são os serviços de escalonamento de tarefas a serem executados em tempo real, permitindo interações com as tarefas escalonadas do padrão Linux, evitando o atraso de qualquer serviço de tempo real no sistema. As tarefas desenvolvidas para executar sobre o RTAI podem ser executadas tanto no espaço de usuário como no *kernel*, existe uma interface única para os dois casos, tornando o controle de tarefas mais transparente.

O Windows CE é um sistema operacional desenvolvido pela Microsoft para os sistemas embarcados. As primeiras versões desse SO foram para dispositivos *hand-helds* (*palmtops*), já as versões posteriores passaram a utilizar uma estrutura de micro-kernel específica para customização para SEs, ampliando sua área de atuação, tornando um sistema em tempo-real e modular baseado em componentes. (ANDRADE, 2013)

O UClinux é uma do distro do Linux para sistemas embarcados, foi baseado no *kernel* 2.0 do Linux. É destinado para microcontroladores sem unidade de gerenciamento de memória (MMUs). Atualmente é uma distribuição completa do Linux possuindo uma coleção de aplicativos, bibliotecas e cadeias de ferramentas para os usuários. (ALMEIDA, 2008)

2.5.4 Desenvolvimento de Software

Para o desenvolvimento do sistema embarcado, deve-se possuir o conhecimento prévio das características e limitações do hardware a ser utilizado e dos softwares empregados (sistema operacional e as linguagens de desenvolvimento). Os princípios do desenvolvimento do software embarcado, baseia-se nos mesmos princípios de qualquer outro projeto de sistema informatizado, com acréscimos das particularidades dos sistemas embarcados.

O modelo de cascata, pode ser utilizado para descrever os processos sequenciais do SE, o resultado de um fase é utilizada como entrada na próxima fase do processo; neste modelo cada fase possui uma dependência da fase anterior. O modelo de cascata é de fácil aplicação, implementação, gerência e compreensão. Existem outros modelos de melhor eficiência e qualidade do produto. Entre os modelos utilizados estão: o modelo de prototipação, o modelo em espiral (modelo por refinamentos sucessivos, proposto por

Pressman), o modelo em V, processo unificado proposto por Jacobson, entre outros.

Além dos modelos tradicionais já citados, existe um modelo desenvolvimento específico para sistemas concorrentes e em tempo real, comumente utilizado em projetos de sistemas embarcados, o COMET (*Concurrent Object Modeling and Architectural Method*) proposto por Hassan Gomaa, o COMET é baseado nos casos de uso da UML (*Unified Modeling Language*), o COMET é um processo iterativo, e os requisitos funcionais são definidos em termos de atores e casos de uso. Este modelo é compatível com o Modelo de Processo Unificado. (GOMAA, 2004)

3 RASPBERRY PI

O Raspberry Pi é um microcontrolador que possui o tamanho pouco maior de um cartão de crédito, foi desenvolvido pelo engenheiro britânico Eben Upton e sua equipe, então foi criada a Fundação Raspberry PI. O intuito desse projeto é fornecer um meio de tecnologia de baixo custo, sendo prático e acessível para que pessoas de diferentes idades pudessem aprender e desenvolver programas de modo mais fácil e intuitivo.(PORTUGAL, 2017a)

O objetivo inicial do projeto era desenvolver e comercializar um computador de placa única (SoC), de tamanho reduzido e de fácil aquisição pelos interessados no assunto. Apesar do tamanho enxuto e das características pouco convencionais, o Raspberry Pi é um computador como outro qualquer. Dependendo de sua finalidade, pode até ser utilizado como um computador de uso pessoal, tendo os mesmos recursos de computador convencional, como navegadores de internet, reprodução de conteúdo multimídia, ferramentas de edição de texto (Pacote LibreOffice).(RASPBERRY, 2013)

Os primeiros conceitos do Raspberry Pi surgiram em 2006 por Upton e sua equipe, mas só em agosto de 2011 que os primeiros componentes começaram a ser distribuídos para o público em geral.

3.1 Características

Dependendo da aplicação que for utilizada no Raspberry Pi, será preciso tomar algumas decisões sobre os dispositivos periféricos e os softwares que serão empregados. Para utilizar o Raspberry primeiro deve possuir algum dos seus modelos, que variam entre $25 e $35 (valores padronizados pela fundação Raspberry Pi para seus distribuidores, este valor em Reais contando taxa cambial, imposto de importação, ICMS (varia por Estado), pode chegar em torno de R$133,00 a R$190,00), isso para os tipos de uso pessoal, o modelo para uso industrial custo por volta de $90 (com o módulo CM1).(GARETT, 2016)

Os modelos ainda suportados oficialmente pela fundação oferecem diferentes características para uso. A nomenclatura utilizada pela Fundação Raspberry Pi para definir os modelos de suas placas pode ser um pouco confuso em um primeiro momento. As versões com complemento Model B representam os modelos de maior desempenho da geração, os que apresentam o sinal de mais (“+”) representam os modelos revisados da versão original. As primeiras versões de entrada de uma geração, que geralmente são mais simples recebem o complemento de Model A. Há também o modelo Pi “Zero”, que é a versão condensada de menor custo, e de poucas interfaces de entrada.

3.2 Arquitetura

A arquitetura dos modelos do Raspberry Pi tendem a ser similar a uma placa desenvolvimento como Arduino, ou como um substituto para *desktop*(minicomputador), mas é mais similar aos componentes internos de um *smartphone*, porém de forma exposta com muitos conectores de diversas interfaces.

O formato da arquitetura das placas do raspberry Pi são do tipo SoC (System-on-a-Chip), integrando todos os componentes básicos de um computador e outros conectores eletrônicos. O Raspberry Pi é um microcontrolador que possui processador, unidade de processamento gráfico (GPU), memória RAM, conectores para periféricos de (HDMI, USB, microUSB, microSD, conector de áudio, Ethernet) e módulo Wi-Fi (dependendo do modelo). (RICHARDSON; WALLACE, 2013)

A arquitetura apresentada a seguir é do modelo Raspberry Pi 3 Model B, por ser o modelo mais comum no mercado para aquisição.

3.3 Raspberry Pi 3 Model B

O Model B da terceira geração se destaca pelo aumento de performance e pelo hardware utilizado, até então o último modelo lançado pela Raspberry Pi Foundation em fevereiro de 2016. Ver Figura 1

Figura 1 – Raspberry Pi 3 Model B

Fonte: (RASPBERRY FOUNDATION, 2017)

Seu grande diferencial deve-se ao processador de ARM Cortex-A53 de 1.2 GHz *Quad-Core* (quatro núcleos para processamento), permitindo trabalhar em 64 bits com 1GB de memória RAM, o acelerador gráfico(GPU) é o VideoCore IV Dual-Core, essa composição faz parte da última versão do *chip* Broadcom BCM2837, comparado com o Pi 2 Model B que também utiliza o mesmo *chip*, mas com velocidade de *clock* menor, dando ao Pi 3 Model B um melhor desempenho para execução e também pelo fato dos núcleos serem eficientes e o conjunto de instruções conseguir realizar mais instruções com processamento

de 64 bits. Devido a estas modificações e a outros melhoramentos na organização da placa, pode-se ter um ganho de desempenho de até 50%, se comparado ao modelo antecessor

Além do processador, outra importante mudança foi a inclusão de duas conexões sem fio, uma Wireless Lan 802.11 bgn e a outra é o Bluetooth 4.1, o que torna a utilização da placa muito cômoda, conectando a placa com o Wi-Fi ou por conexão *bluetooth*.

Em relação às outras interfaces de expansão, o Model B da terceira geração oferece os mesmos conectores e mesma configuração do modelo da segunda geração: GPIO de 40 pinos, quatro portas USB, interface Ethernet 10/100, saída de som, câmera interface (CSI), display interface (DSI), *slot* para o cartão microSD, além do *slot* microUSB para alimentação, necessitando de uma fonte de 5,1V com 2,5A. As dimensões também são semelhantes do modelo da segunda geração, tendo 85mm x 56mm x 17mm. (PORTUGAL, 2017b)

A performance desta placa é ideal para quem deseja desenvolver projetos onde precisam de um grande processamento e que necessite, de comunicação sem fio através do Bluetooth ou do Wireless.

Este foi o modelo utilizado neste trabalho, os demais Modelos podem ser vistos no APÊNDICE A, deste trabalho.

3.4 Processador

O principal componente do Raspberry Pi 3, o *chip* Broadcom BCM2837 com o processador Cortex-A53, possui multiprocessamento simétrico de quatro núcleos em forma de *cluster*, cada uma, tendo uma memória cache L1 (Level 1 de 32 kB) e uma memória cache L2 (Level2 32 kB) compartilhada, a taxa de execução em seus núcleos é de 1.2 GHz, e utilizando um *chip* de memória Elpida B8132B4PB-8D-F de 1GB em formato LPDDR (é uma SDRAM de dupla transição utilizada para dispositvos móveis).

O Cortex-A53 está baseada na arquitetura(de código) Armv8-A (família A), a Armv8 possui suporte de instruções 64 bits (AArch 64) e completa compatibilidade com arquitetura antecedente a Armv7 que utiliza instruções em 32 bits (AArch 32), incluindo recursos como o NEON, suporte para virtualização por e recursos de segurança (TrustZone).

A tecnologia NEON é uma arquitetura de extensão SIMD (*Single Instruction Multiple Data*), que otimiza os algoritmos e as funções de processamento de sinais de áudio, vídeo, reconhecimento de voz e facial.

A TrustZone é uma tecnologia que fornece o isolamento de hardware para todo o sistema, criando um ambiente seguro e isolado que possa fornecer confidencialidade e integridade ao sistema.

O Cortex-A53 possui uma performance balanceada de desempenho e eficiência, construído de forma por utilizar baixa potência, mas que possa fornecer ao dispositivo um alto desempenho com restrição de energia nos ambientes.

O chip BCM 2837 ainda possui um processador gráfico (GPU) de baixo consumo, o VideoCore IV de 400MHz baseada na arquitetura DSP (Digital Signal Processing), dando a GPU eficiência e flexibilidade para codificar e decodificar uma série de *codecs*[3] para multimídia, mantendo o baixo consumo de energia, tudo isso permitindo ao Raspberry Pi uma resolução Full HD 1080p. A arquitetura DSP utiliza de processamento digital para calcular grande variedades de operações em uma sequência de amostras, esse processamento pode envolver operações lineares e não lineares. (ARM, 2017)

3.5 Conectividade

A maioria dos modelos possuem uma porta Ethernet de 100MB/s de padrão RJ45, com exceção do Pi 1 Model A e do Pi Zero (a versão Pi Zero W possui conexão Wi-Fi 802.11n e Bluetooth 4.0), mas podem ser usados adaptadores de rede do tipo Ethernet USB (ou um adaptador Ethernet microUSB, para o Pi Zero), ainda para esses dois modelos pode ser usado um dongle, que é receptor USB que permite conectar-se a uma rede Wi-Fi (para o Pi Zero terá que utilizar um adaptador USB-microUSB também chamado de USB OTG).

O modelo Pi 3 Model B além de possuir a porta Ethernet RJ45 está embutido nessa versão o chip da Broadcom que serve para realizar conexão *wireless* com a placa, o módulo BCM43438 fornece Wi-Fi 802.11n (possui velocidade de conexão entre 20 e 40 Mb/s), Bluetooth 4.1 (velocidade de conexão em torno de 25 Mb/s) e o BLE. Neste módulo de rede há ainda um receptor FM, mas se encontra desabilitado por padrão (Pode estar desligado do hardware ou não possui um firmware para seu funcionamento).

O BLE (*Bluetooth Low Energy*) ou *Bluetooth Smart* é uma tecnologia de rede *wireless*, que possui como principal característica um baixo consumo de energia e preço reduzido comparado ao Bluetooth tradicional, essa tecnologia foi desenvolvida para aplicações que necessitam enviar informações de tamanho pequeno e de distância de menor alcance para comunicação. Esta tecnologia é bastante usada para IoT (Internet das Coisas), devido suas características. (CYPRESS, 2017)

Uma forma bastante usual de conectar a Raspberry em uma rede é utilizado o cabo RJ45 na porta Ethernet da placa com auxílio de um notebook ou *desktop* (máquina *host*), este devendo possuir uma placa de rede wireless ou uma segunda porta Ethernet. Para esta técnica funcionar deve utilizar a placa Wireless do computador ou uma porta

[3] técnica de codificação de formato de mídia

Ethernet para conectar o computador a rede. Em seguida deve conectar o cabo (padrão *crossover*) do computador na porta Ethernet do Raspberry.

Em um computador com sistema operacional Windows, deve acessar a rede compartilhamento e alterar as configurações do adaptador de rede, selecionando a conexão de rede sem fio e a conexão local (entre o Raspberry e o computador), criando uma conexão em ponte. Para que não exista problema com IP, é recomendado deixar com a configuração automática para escolha de um IP na rede.

Com essa configuração o Raspberry já consegue conectar-se a rede, mas pode ocorrer do Raspberry não conseguir obter um IP. Para isso deve configurar a interface de rede na Raspberry, com o comando $sudo leafpad /etc/network/interfaces configure um IP *static*: adress seguindo como base o padrão do roteador da rede, e o terceiro endereço deve ser o mesmo IP do computador *host*, a máscara de rede o endereço padrão e o gateway também deve ser o mesmo do *host*, por fim salve e feche este arquivo. Deve também configurar o DNS utilizando o comando $sudo leafpad /etc/resolv.conf o DNS deve ser igual ao gateway do *host*, salve e feche o arquivo. A conexão estará estabelecida, pode ser usado o comando ping da Raspberry para o roteador, para testar se a conexão de rede foi realmente estabelecida.

Em uma máquina *host* com *kernel* Linux também é possível criar conexão de ponte. Para isso deve ir em Configurações do Sistema, após na configuração de Rede e clique em conexão cabeada (ou com fio) na aba de Configurações IPv4 ou IPv6 selecione Compartilhamento com outros computadores e salve as configurações. Caso de não estabelecer a conexão faça a mesma configuração já mencionada no Raspberry.

3.6 Armazenamento

Nenhum dos modelos do Raspberry possuem um componente que funcione como um disco rígido (HD), tanto o sistema operacional como o armazenamento de arquivos deve ser gravados em um cartão de memória microSD, o microSD fornece o armazenamento inicial para o Sistema Operacional e para os arquivos. O armazenamento pode ser estendido através dos conectores às portas USB.

Na formatação do microSD deve utilizar uma instalação particionada do sistema operacional e do sistema de arquivo, recomenda-se para utilizar o Raspberry um microSD com espaço de armazenamento de no mínimo de 4GB, para aplicações que utilizem um sistema operacional completo e que necessite de espaço para armazenar os demais dados, também recomenda-se utilizar um cartão de classe 10, por causa da taxa de leitura e gravação, podendo utilizar cartões de classes inferiores, mas a velocidade de execução da aplicação no Raspberry poderá ficar comprometida.

Na PCB Raspberry Pi possui uma taxa de transferência de 25 MB/s e a velocidade de leitura e gravação de 22 MB/s.

Os modelos Pi 1 Model A e Pi 1 Model B utilizam o cartão SD completo, os demais utilizam o formato microSD.

Quando o Raspberry Pi está conectado a uma fonte de alimentação, é executado um programa de inicialização chamado de *bootlader*, que lê o código especial que está gravado no microSD, utilizado para iniciar o sistema operacional no Raspberry. Se não houver nenhum cartão SD inserido, o Raspberry não inicializará. Para colocar o cartão SD deverá desligar a alimentação primeiro, para depois inserir o cartão. Não deve ser inserido o cartão, após a placa estiver ligado, e nem deve retirar o microSD sem o desligamento correto da Raspberry, pois pode corromper os dados do cartão.

Pelo fato do cartão SD ser facilmente removível, pode-se ter vários cartões, cada um com algum Sistema Operacional ou aplicação diferente, bastando desligar a placa, trocar os cartões e reconectar a energia.

3.7 Alimentação

Por padrão nenhum dos modelos da Raspberry Pi possuem um interruptor de alimentação embutido na placa, somente há uma porta microUSB que possui a função de ser o conector para fonte de alimentação, quando o conector é encaixado, a placa liga a aplicação. O conector microUSB é utilizado por ser barato e fontes com este tipo de conector são fáceis de serem encontradas.

O dispositivo de carregamento para o Raspberry Pi deve seguir os parâmetros de voltagem e amperagem, precisando atender as especificações de acordo com os modelos da Raspberry, a seguir uma tabela que informa as necessidade para cada modelo, lembrando que os números de corrente, medidos em amperes são os mínimos requeridos.

Tabela 1 – Especificação de Energia para os Modelos

Modelo	Voltagem/Corrente
Raspberry Pi 1 Model A+	5 volts/0,7 amperes
Raspberry Pi 1 Model B+	5 volts/1,8 amperes
Raspberry Pi 2 Model B	5 volts/1,8 amperes
Raspberry Pi 3 Model B	5,1 volts/2,5 amperes
Raspberry Pi zero	5 volts/0,6 amperes
Compute Module IO Board V3	5 volts/2,5 amperes

Fonte: (Elaborada pelo autor)

Ao utilizar o carregador deve-se cuidar com suas especificações, referentes as informações de saída, que são a voltagem e a corrente que serão convertidos pelo componente

a partir da fonte (fixa ou continua) que será passado a placa. Os riscos de utilizar um equipamento que não atende as necessidades da placa, podem acarretar e não funcionamento correto da placa ou em danos de hardware.

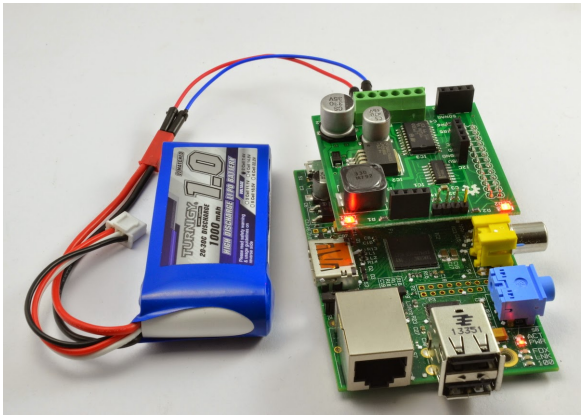
Se a voltagem utilizada for menor do que o necessário a placa poderá ficar em estado de *undervoltage*. Em caso de *undervoltage* a placa não ligará e o led vermelho (embutido na placa) irá piscar de forma ininterrupta, e o led verde que representa a inicialização do sistema operacional não acenderá. Quando o SO inicializa e a energia começa a oscilar irá aparecer no canto superior um desenho de raio, informando que a placa não está recebendo a energia necessária (em caso de superaquecimento da placa aparece um ícone de temperatura).

Com relação a corrente elétrica, se o carregador possuir menos amperes do que necessário, poderá até ligar a placa, mas possivelmente os componentes da placa deixem de funcionar como as portas USBs, ou até mesmo o comprometendo a leitura do cartão de memória. Além de provocar instabilidade no funcionamento de periféricos conectados ao Raspberry, pode ocorrer de corromper os arquivos no cartão de memória durante a gravação.

A porta microUSB por ser muito difundida por possuir a mesma conexão para os carregadores de dispositivos móveis, cria uma falsa percepção que o mesmo pode ser usado para alimentação da placas da Raspberry, os atuais carregadores para celular possuem voltagem de 5V e corrente elétrica de 2 amperes para saída. Para o modelo Pi 3 Model B este tipo de carregador fará que a placa funcione, mas pode fazer com que algumas das portas USB não atuem corretamente.

A fonte de alimentação do Raspberry é por meio de uma carregador fixo na tomada, mas existem outras formas de alimentação para aplicações móveis, pode-se utilizar bateria portátil de celular como os *Power Bank*, ou podem ser utilizada um módulo de alimentação móvel específico para o Raspberry, como mostra a figura2 a seguir.

Figura 2 – Módulo de alimentação para o Raspberry



Fonte: (Dr. Monks's, 2017)

3.8 GPIO

O GPIO (General Purpose Input/Output) são portas programáveis de entrada e saída para periféricos elétricos e digitais para as placas do Raspberry. Dos 40 pinos presentes nas placas, 26 pinos são GPIO, os demais são para energia, aterramento e mais dois ID EEPROM que devem ser utilizados com conhecimento do protocolo I^2C. Estes pinos de GPIO podem ser programados com a linguagem Python utilizando a biblioteca RPi.GPIO, que fornece os principais comandos para utilização dos pinos na placa.

Para quem está iniciando os estudos com programação, ou que não possui conhecimento da linguagem Python, é possível manipular os pinos com o Scrath que é o *software* que realiza programação por blocos lógicos, os comandos são blocos bem intuitivos, tornando a programação atraente em um primeiro contato. O Scrath possui licença de código aberto, mas com componentes proprietários, ele é um projeto do Lifelong Kindergarten Group do MIT Media Lab, sendo disponibilizado gratuitamente, o Scrath é instalado no Raspberry como software padrão para o Sistema Operacional Raspbian. Na figura 3 a seguir o mesmo código para piscar um led.

Figura 3 – Manipulação de GPIO, usando Scrath e a liguangem Python

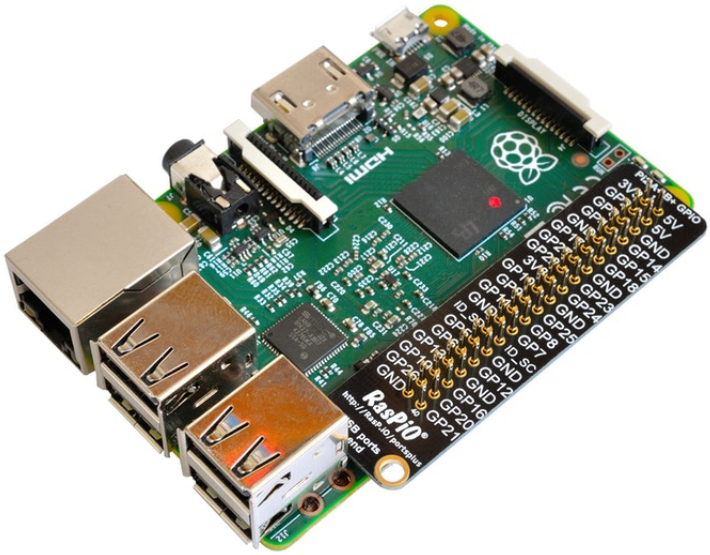
Fonte: (RASPBERRY FUNDATION, 2017)

Fazer projetos com os pinos da placa do Raspberry devem possuir certos cuidados, pois as placas da Raspberry não possuem um sistema de proteção, como no Arduíno, que possui um regularizador de tensão. Nos modelos do Raspberry utilizam como nível alto lógico 3,3V, diferente do Arduíno que utiliza o sistema de lógica digital padrão (TTL), onde nível alto lógico é de 5V. Se utilizar uma tensão maior do que o nível alto lógico poderá

danificar o pino ou até mesmo queimar a placa. Outro fator importante ao manusear a placa, para que não aconteça um curto circuito entre os pinos, devido algum pino estar torto e estando próximo a outro.

Nas placas da Raspberry não possuem as identificações dos pinos, para que não aconteça nenhum incidente é recomendado saber as posições de cada pino, existe um rótulo de identificação que é colocado sobre os pinos, sem obstruir o acesso, um exemplo é o Raspio, como mostra a figura 4.

Figura 4 – Identificações dos Pinos GPIO

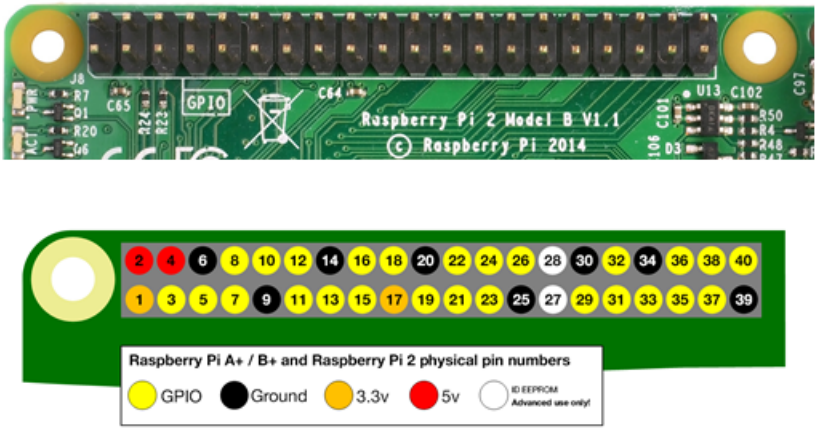


Fonte: (RASPIO, 2017)

Com esse pinos é possível acionar LEDs, Motores, Relés, fazer leitura de sensores e botões, entre outros tipos de dispositivos que podem integrar um projeto embarcado.

Os pinos são numerados fisicamente, iniciando do sentido oposto das portas USBs, da trilha interna de número um, o seguinte é o vizinho de pino dois que fica na trilha externa, a numeração física é crescente fazendo um percurso em ziguezague até próximo das portas USB (ver a figura 5 com a numeração). A numeração física é importante porque os comandos são referentes a sua posição na placa. Há também a numeração GPIO, que o modo de como o computador entende cada pino, esta numeração é difícil de memorizar, pois não segue uma ordem exata, ela pode ser visualizada nas identificações da figura 4. Estes dois tipos de numeração é bom ter conhecimento, pois dependendo da linguagem pode variar, com o Scrath GPIO utiliza a numeração física, já no Python pode escolher o tipo de numeração.

Figura 5 – Numeração dos Pinos GPIO



Fonte: (RASPBERRY FUNDATION, 2017)

Sobre os pinos os 2 e 4 como representados na figura 5, são usados para saída da alimentação com tensão de 5V, esses dois pinos devem possuir cuidado redobrado pois não podem entrar em contato com as portas ao redor. Os pinos 1 e 17 também são usados como uma saída de alimentação, mas com tensão de 3.3V, com este pino é possível comunicar com outras portas, mas por segurança é recomendável utilizar um resistor para limitar a corrente entre os componentes.

Os pinos de cor preta são as portas Terra (*Ground*), usado para referência zero potencial e protege contra possíveis descargas elétricas na placa. Por praticidade é bom utilizar aqueles que deixarão a ligação mais organizada, ou que fique o mais próximo o possível dos pinos que farão parte da ligação.

Existem dois pinos na placa que podem ser programados utilizando o protocolo I^2C (*Inter-Integrated Circuit*), é um sistema que foi desenvolvido pela Phillips, para fazer a conexão entre periféricos de baixa velocidade. Este protocolo de circuito inter-integrado utiliza um barramento entre dois fios bidirecionais de dreno aberto (padrão de entrada e saída para projetos digitais), sendo um com dados (*Serial Data* - SDA) e outro com *clock* (*Serial Clock* -SCL), os pinos 3 e 5 respectivamente, para comunicação serial entre os circuitos integrados projetados na mesma placa. Os pinos 27(SDA) e 28(SCL) também podem utilizar o protocolo I^2C, mas possuem um adicional de realizar comunicação com um EEPROM (*Electrially-Erasable Programmable Read-Only Memory*). Periféricos com este tipo podem ser apagados e programados várias vezes, através de uma tensão elétrica interna ou externa. Estes pinos são de grande importância, pois podem conectar diferentes tipos de periféricos externos, eles incluem um resistor de *pull-up* de 1.8kohms para 3.3V.

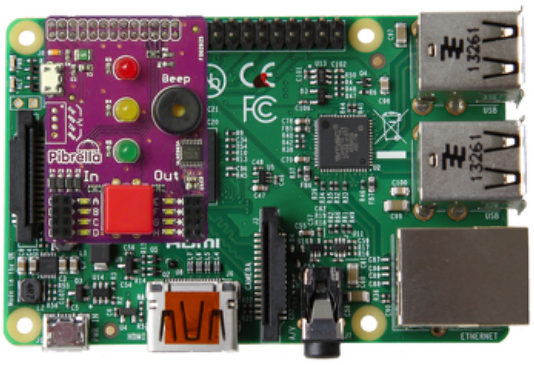
Os pinos 8 e 10 são pinos seriais, que utilizam o protocolo RS-232 (protocolo para troca serial de dados binários entre um terminal e o comunicador, muito utilizado nas portas seriais dos *desktops*) para envio e recebimento de sinal digital.

Os pinos amarelos são os pinos GPIO padrão, servem para fazer envio e recebimento de dados digitais.

Os pinos 19, 21, 23, 24 e 26 também são pinos de entrada e saída de dados digitais. Estes pinos têm a capacidade de realizar a comunicação serial *Full Duplex* síncrono, que permite o processador comunicar com periférico externo de forma bidirecional. (ARAÚJO, 2014)

Para utilizar os pinos presentes na placa sem se preocupar com algum incidente que possa danificar a placa, existe uma placa auxiliar chamada Pibrella (adquirido separadamente), com esta placa podem ser feitos projetos de forma mais segura, ela possui embutida leds, botão e um *buzzer* (alto-falante que emite *beeps*). Nesta placa podem ser utilizadas dispositivos externos como motores, leds, relés e demais componentes. O Pribella fornece um área mais amigável para criação de projetos, ela é encaixada nos primeiros 26 pinos da placa do Raspberry (como na figura 6).

Figura 6 – Pribella integrada com o Raspberry



Fonte: (PRIBELLA, 2017)

3.9 Interfaces para Conexão de Periféricos

3.9.1 USB

O Raspberry Pi 3 Model B utiliza 4 portas USB 2.0 suportando até 1.2A de forma contínua. Nos primeiros modelos das placas do Raspberry Pi as portas USBs foram limitadas devido à quantidade de energia que elas poderiam fornecer. Para os modelos que possuem uma ou duas portas USB pode ser utilizado um *hub* (adaptador) para aumentar a quantidade de periféricos que poderão ser conectados, deve utilizar um *hub* (que atende as

especificações da placa). *Hubs* de tipo USB 3.0 não funcionam corretamente, a velocidade de transmissão será reduzida. Os modelos da Raspberry tem compatibilidade com conexões *plug and play*, permitindo conectar qualquer periférico como, teclado e mouse USB, que possuem as versões sem fio (via *bluetooth*).

O modelo Pi Zero que não possui portas USB, pode ser utilizado o acesso remoto para fazer utilização da placa sem precisa de periféricos de entrada como mouse e teclado, isso equivale aos outros modelos.

3.9.2 HDMI

O conector HDMI oferece saída de áudio e vídeo digital, podendo conectar a uma grande variedade de monitores como televisores e monitores de computador, dependendo do monitor será necessário um adaptador de VGA ou DVI para o HDMI. Em casos onde precisar do adaptador o som poderá não funcionar e a resolução de tela também pode ficar diferente da estabelecida (há casos que deverá editar as configurações de resoluções).

Um outro formato de entrada de vídeo composto, que utiliza o conector do tipo RCA, que fornece sinais de vídeo composto nos formatos NTSC ou PAL. Esses dois formatos possuem resolução muito inferior se comparada com a resolução que o HDMI transmite. O RCA dificilmente será encontrado nos modelos atuais da Raspberry.

3.9.3 JACK

Existe uma entrada de áudio analógico que permite conduzir cargas de alta impedância (como alto falantes amplificados) utilizando o *plug* Jack de 3,5mm, que inclui o sinal de vídeo composto, o que permite a remoção do soquete de vídeo composto (RCA). O soquete do padrão Jack possui 4 polos que permite a transferência de sinais de áudio e vídeo, este padrão é encontrado em dispositivos como os conectores de áudio para *smartphones* (fones de plug P2).

3.9.4 DSI

O Conector de Interface Serial do *Display* (DSI) pode empregar no Raspberry uma tela específica de LCD ou de OLED, podendo criar sistemas com tela dedicada. O DSI recebe um cabo em fita plana (*flat cable*) de 15 pinos, ele foi projetado para funcionar com modelos que possuem furos de montagem. Este conector fornece uma interface de exibição de alta resolução dedicada e rápida transmissão de envio de informações de vídeo da GPU para o *display*.

3.9.5 CSI

O conector de Interface Serial de Câmera (CSI), permite que um módulo de câmera seja conectado diretamente à placa do Raspberry. A versão utilizada no Raspberry é o MIPI CSI-2 versão 1.01 que suporta até quatro faixas de dados, cada uma com largura de 1Gbps de banda máxima, essa interface utiliza um número menor de conexões elétricas, reduzindo a eletricidade na PCB (*Printed Circuit Board*).

Os conectores CSI e DSI utilizam o formato MIPI (*Mobile Industry Processor Interface*), que é uma organização global que padroniza algumas especificações para aparelhos móveis. A maioria das grandes empresas do setor móvel seguem essas padronizações.

3.9.6 Led Status

A placa da Raspberry possui quatro leds, dois deles estão embutidos na placa que indicam a fonte na PCB e atividade do microSD (ver figura 7) os outros dois estão embutidos na placa de rede (LAN) para informar a conexão.

O PWR é um led vermelho que fica embutido na placa, ao lado do conector microUSB, este led indica a energia presente na placa, se a energia presente é de 5V o led fica acesso constantemente, caso a energia fique abaixo de 4,63V o led irá piscar até atingir alimentação mínima recomendada.

O ACT é um led verde que também é embutido na PCB, indica a atividade do cartão microSD na placa, se o led ficar piscando está ocorrendo alguma atividade (leitura ou gravação) no microSD, caso o led não ligar quando a Raspberry for conectada cartão, provalvemente o cartão microSD não está encaixado corretamente no *slot*.

Figura 7 – Led de atividade da Raspberry

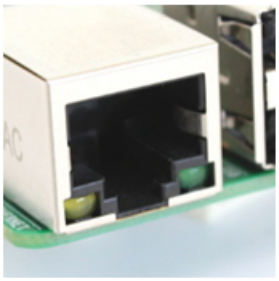
Fonte: (RASPERRY PORTUGAL, 2017)

Na placa de rede Ethenet (ver figura 8) possui um led verde que é utilizada por duas funções. Uma das funções é indicar que o cabo conectado é *Full Duplex*, essa função possui rótulo de FDX, se a conexão é estabelecida o led acende. A outra função é o LNK

que indica a atividade na porta fazendo que o led fique piscando continuamente, caso o led não ligue a conexão não está estabelecida.

O outro led é denominado de 10M/100 de cor amarela, este led acende quando o cabo está conectado e a conexão está funcionando em 100mbps.(HOWSE, 2015)

Figura 8 – Led de atidade da placa de Ethernet



Fonte: (RASPERRY PORTUGAL, 2017)

3.10 Sistema Operacional

O Raspberry foi projetado para utilizar sistemas operacionais baseados com *kernel* Linux, assim algumas dos SOs tradicionais para *desktop* como Ubuntu, Debian, Fedora e o Arch possuem versões de sistema operacional específicas para o Raspberry Pi.

Como o Raspberry Pi é baseado em um *chipset* de dispositivo móvel, ele possui requisitos de software diferentes de um computador de mesa. Além disso os processadores da Broadcom, possui características proprietárias, tendo que utilizar drivers do tipo *Binary Blob*[4].

Os sistemas operacionais devem operam com as restrições de memória RAM e espaço de armazenamento. As principais distribuições do Linux são: Raspbian, Occidentalis, Arch Linux, Xbian e QtonPi.

3.10.1 Raspbian

O Raspbian é um sistema operacional gratuito baseado no Debian, mas que foi otimizado para o *hardware* do Raspberry Pi.

Este Sistema Operacional é indicado pela fundação Raspberry como oficialmente recomendada. Ao ser instalado o Raspbian possui diversos softwares e linguagens nativas em seu sistema, *softwares* que variam desde uso educacional, programação e uso geral, como Sonic Pi, Mathematica, nodeJS, Python, Java, LibreOffice e muitos outros. Apesar

[4] termo utilizado para projetos *open-source*, sendo o código-fonte do projeto não disponibilizado para visualização ou modificação

do Raspbian ser recomendado pela Fundação da Raspberry, não possui vínculo com a Fundação Raspberry. O Raspbian é o SO recomendado por quem está começando a utilizar o Raspberry Pi, por este motivo ele terá mais ênfase neste trabalho. (RASPBIAN, 2017)

Figura 9 – Sistema Operacional Raspbian

Fonte: (RASPBIAN, 2017)

3.10.2 Arch Linux ARM

A Arch Linux possui uma versão de sistema operacional especificamente para computadores baseados na arquitetura ARM, suportando todos os modelos da Raspberry Pi. Este SO tem como lema a simplicidade e a personalização, dando o usuário o controle e a responsabilidade completo do sistema operacional. Por suas características torna do Arch Linux um sistema operacional extremamente poderoso, podendo configurar os sistemas para tarefas específicas, assim podendo ser todo personalizado. (ARCH, 2017)

Figura 10 – Sistema Operacional Arch Linux

Fonte: (ARCHLINUX ORG, 2017)

3.10.3 OSMC

O OSMC (*Open Source Media Center*) é um sistema operacional que torna o Raspberry Pi em uma central multimídia, é um SO *open-source* baseado no Linux Debian. O OSMC permite reproduzir mídia da rede local e de repositórios online (estando conectado à Internet), ele é uma variante do Projeto Kodi.

Como seu propósito é criar uma central multimídia, o usuário não terá problema na utilização caso não entenda de Sistemas Operacionais baseados em *Kernel* Linux, ele é facilmente manuseado pelo controle remoto da televisão ou do monitor, sem necessidade de mouse, teclado ou qualquer interface de acesso remoto, como uma aplicação de *smartphone*.

Existem sistemas semelhantes ao OSMC que também são softwares livres e de código aberto (GPL), tendo variações do Projeto Kodi, os mais conhecidos são: Xbian, OpenELEC e LibreELEC. (OMSC, 2017)

Figura 11 – Sistema Operacional OSMC

Fonte: (KODI TV, 2017)

3.10.4 Snappy Ubuntu Core

O Snappy Ubuntu é uma versão enxuta do Tradicional Ubuntu para *desktop*, está versão para Raspberry possui a qualidade de ser um sistema operacional leve e projetado especificamente para aplicações de sistemas embarcados e IoT (*Internet of Things*, internet das coisas). O Snappy utiliza pacotes de aplicativos seguros e remotamente estáveis denominados de snaps.

Esse SO possui uma separação arquitetural dos *kernels* dos dispositivos, o sistema operacional e dos aplicativos, deste modo, engenheiros e programadores podem trabalham em paralelo. (CANONICAL, 2017)

Figura 12 – Sistema Operacional Snappy Ubuntu Core

Fonte: (CANONICAL, 2017)

3.10.5 Microsoft 10 IoT CORE

Além dos sistemas operacionais baseados em *kernels* Linux, existe uma aplicação da Microsoft específico para Internet das coisas (IoT), que possibilita inúmeros tipos de prototipagem para sistemas embarcados. O Windows 10 IoT Core é voltado para dispositivos pequenos que podem ou não ter telas.

Como o Windows 10 IoT Core pertence à Microsoft, sua aplicação é gratuita, necessitando de um computador com a versão mais recente do Windows 10. Os comandos para a placa são enviados através da versão mais recente Visual Studio (possui uma versão gratuita).

Aplicação IoT Core não fornece uma interface gráfica para o Raspberry, ela apenas pode ser acessada remotamente pelo computador, tornando a usabilidade semelhante aos projetos do Arduíno. (MICROSOFT, 2017)

Para o desenvolvimento de aplicações o Visual Studio fornece algumas linguagens para programação como C++, C#, JavaScript, Visual Basic e fornecendo suporte para python e Node.JS

Figura 13 – Sistema Operacional Windows IoT

Fonte: (WINDOWS TEAM, 2017)

4 BIBLIOTECAS DE RECONHECIMENTO DE VOZ

Neste capítulo serão mostradas as bibliotecas que foram estudas para o projeto.

4.1 Google Assistant Embedded SDK

4.1.1 Introducação

O Google Assistant é um sistema de inteligência artificial,é executado como um assistente virtual controlado por comandos de voz. Este recurso é um aperfeiçoamento do Google Now presente nos dispositivos, principalmente, do sistema operacional Android, mas a nova versão permite uma interação (conversa) bidirecional, utilizando o processamento de linguagem natural do Google. Este recurso foi lançado no evento Google I/O de 2016, que aborda novas tecnologias e ferramentas para comunidade de desenvolvimento. No Brasil, o Google Assistant começou a oferecer suporte em Agosto de 2017.

Esta ferramenta possui arquitetura para diversas plataformas podendo ser usada em computadores pessoais, smartphones e microcontroladores de diversas arquiteturas, tendo amplo suporte para o Raspberry Pi. Atualmente, a versão estável é a versão 0.0.3 de julho de 2017. Este assistente possui funções de pesquisas na internet ou pode ser utilizado um repositório individual para a aplicação. Além das pesquisas, os comandos podem realizar outras operações, como abrir arquivos de multimídia e documentos.

A aplicação do assistente da Google no Raspberry é ativado através de um *hotword* , que é uma palavra específica para ativar a interface de voz da aplicação, a partir dela o assistente começa a fazer captação das ações que serão recebidas, por padrão para ativar o sistema deve ser utilizado o comando "*hey google*" ou "*ok google*", em seguida já começará a interação do assistente com locutor.

Para criar um projeto utilizando o Google Assistant deve-se acessar o repositório de aplicações do Google, denominado "Actions on Google". Nesse repositório estão todos o recursos e APIs que o Google fornece para desenvolvimento, podendo fazer a integração de vários recursos disponíveis. Maioria das ferramenta disponíveis possuem uma cota de utilização que é disponibilizada gratuitamente, ao passar dessa cota, será notificado que a cota foi ultrapassada e o valor que será cobrado ao desenvolvedor para utilização.

4.1.2 Assistente Pessoal

O Google Assistant proporciona aos desenvolvedores de sistemas, a possibilidade de utilizar os recursos da API em projetos externos. A Google utiliza os recursos dessa API no Google Home que também é um assistente controlado por voz, mas que é utilizado como assistente residencial, onde consegue distinguir as pessoas e os comandos que são dados.

A Google e a Fundação Raspberry Pi possuem um projeto que é um kit de hardware do tipo Voice HAT (*Hardware Accessory on Top*), este projeto é chamado de AIY Projects (*Artificial Intelligence Yorself*), é conjunto de componentes de alto falante, microfone, botão, placa de expansão HAT, cabos, armação de papelão e outros componentes que montados juntamente com a placa do Raspberry Pi 3 criam um assistente pessoal controlado por voz, semelhante ao Google Home. Com montagem desses componentes o projeto final é uma caixa de papelão, que basta acionar o botão ou bater palmas para acionar o Google Assistant.

4.1.3 Segurança

Uma questão muito importante é sobre a segurança do Assistant, como a aplicação armazena alguns comandos e informações que podem ser compartilhados com outros serviços podendo gerar algumas falhas de segurança. Nas configurações do assistente podem ser configurados que tipo de informações podem ou não ser compartilhadas. Se a aplicação estiver vinculado com a conta do Google, todas as conversas ficarão salvas em um histórico de conversas, assim se houver uma informação sigilosa poderá ser excluída.

Segundo as diretrizes do Google algumas das informações podem ser armazenadas para tornar os seus serviços mais ágeis para utilização, com essas informações a aplicação pode aprender mais sobre o utilizador melhorando o conhecimento. As informações passadas ao assistente possuem a mesma política do seu navegador, na qual utiliza os dados do usuário para focar melhor nos anúncios ao usuário mostrando anúncios que sejam de possível interesse ao usuário.

No caso de aplicações de terceiros, onde pode ser necessário a troca informações, existe um controle das informações que podem ser compartilhadas, mas a informação pessoal não é vendida ou enviada a terceiras sem o consentimento do utilizador, outros tipos de restrições devem ser lida nos termos de política de privacidade das aplicações do Google.

4.1.4 Desenvolvimento

O Kit de desenvolvimento (SDK) do Google Assistant permite configurar o *hotword* de detecção, o controle de voz, o processamento da linguagem natural e o processamento nos dispositivos. Este SDK suporta duas formas de integração para a aplicação.

A biblioteca do Python para o Google Assistant permite criar uma aplicação, está biblioteca é suportada para as arquiteturas linux-armv7l e linux-x86_64 (arquitetura do Raspberry Pi 3).

A linguagem python é baseada em eventos de alto nível, mas de possível extensão. Alguns recursos estão disponíveis para ser utilizados como:

- Ativação no formato *Hands-Free* (ativação que não necessita tocar em algum botão para ativar) pelo *hotword* padrão "*Hey google*" ou "*Ok Google*", como utilizado no Google Home;
- Captura e reprodução de áudio;
- Gerenciamento das ações na conversação;
- Gerenciamento de temporizador e alarme.

A outra forma de implementação é utilizado a API gRPC (conjunto de linguagens e ferramentas para desenvolvimento), esta opção possui maior flexibilidade e diversas plataformas para suporte. Esta API trabalha em baixo nível manipulando diretamente os bytes de áudio em uma conversação do assistente com o locutor. Na API gRPC podem ser utilizados diversos tipos de linguagem como o Node.js, Go, C++, Java e bem como outras linguagens que se encontram disponíveis no repositório do gRPC. O gRPC também utiliza o python como o código de referência para a captura de áudio, reprodução, e o gerenciamento da transição de estados da conversação.

Utilizando as formas de integração do SDK permite criar os protótipos nos dispositivos que estão sendo usados, podendo adicionar funcionalidades extras com o repositório de aplicações do Google (Actions on Google). Atualmente o SDK do assistente está em modo de teste para desenvolvimento. A utilização é gratuita para os testes de desenvolvimento e tem um limite diário de solicitações que pode ser utilizado. Para colocar a aplicação em produção é necessário entrar em contato com setor de desenvolvimento comercial da Google fazendo um relatório da aplicação a ser disponibilizada.

4.1.5 Instalação

Para instalar o Google Assistant no Raspberry Pi 3 é preciso estar com sistema operacional Raspbian ou algum SO derivado do Ubuntu ou Debian, para suportar a instalação do assistente.

O Primeiro passo é criar o projeto no Google Cloud Plataform, onde são criados os projetos utilizando os recursos e ferramentas do google. O projeto então é nomeado, e será selecionado aplicação e a arquitetura que serão utilizadas no projeto, em seguida será necessário as credenciais para autenticar o projeto, ainda nessa parte o Google vinculará as informações da conta com o projeto do Assistant. Com essas configurações o projeto estará criado. Mas ainda deve ser instalado na Raspberry Pi 3.

Na placa deve ser configurada o microfone e a saída de som caso a placa possua mais de um tipo de um dispositivo de som (entrada e saída). Em seguida pode ser instalado o SDK com python no Raspberry, no site de desenvolvimento do Google estão os comandos necessário para instalar o assistente, após a instalação a aplicação irá requerer autenticação que foi feita na criação do projeto.

Com Assistant instalado no Raspberry é possível fazer testes de comandos com a versão de demonstração (demo). Com essa versão demo podem ser feitas consultas na internet, definir um alarme, fazer perguntas referentes às informações vinculadas com a conta utilizada na aplicação e algumas outras conversações simples do locutor com o assistente. Até o momento essa SDK para o Raspberry está em versão de desenvolvimento, o assistente só possui a língua inglesa para conversação, assim os comandos e ações deve ser feitas com a língua Inglesa. Possivelmente terão mais atualizações com o decorrer do desenvolvimento deste recurso fornecido pela Google. (TECHTUDO, 2017)

4.2 Jasper

O Jasper é uma plataforma para controle de voz em código aberto que funciona em diversas arquiteturas, sendo principalmente para o Raspberry Pi. Com essa plataforma é possível criar uma aplicação que seja controlada por voz, podendo fazer ações como reproduzir músicas e acessar o calendário ou até mesmo poder automatizar uma casa. O Jasper consiste de pacotes para conversão de voz para texto (Speech To Text - STT) e a conversão de texto para fala (Text-To-Speech - TTS).

4.2.1 Desenvolvimento

O Jasper fornece uma API simples, para o desenvolvedor criar os módulos controlados por voz. Existem dois tipos de módulos, o padrão e o de notificação.

O módulo padrão é a interação que o usuário faz com a aplicação do Jasper, assim fazendo o Jasper responder com uma ação. Para a criação de um módulo no Jasper, deve definir algumas parâmetros na implementação como, as palavras chaves que precisam ser reconhecidas na fala do usuário, quais serão as palavras entradas que terão validade no módulo e as ações que a aplicação terá sobre cada fala do usuário.

Segundo a documentação do Jasper, para uma melhor precisão da aplicação, é melhor utilizar um dicionário de palavras ou de texto pequeno, e que cada módulo possua as palavras-chaves exatas da fala. Após a fase de reconhecimento de fala do usuário, começará a fase de validação das palavras-chaves que estão na fala, e a baseada nelas que a aplicação, executará suas ações de resposta. Pode acontecer de diferentes módulos utilizarem as mesmas entradas, para evitar que haja falha de resposta para os módulos que utilizem as mesmas entradas, pode-se utilizar os níveis de prioridade.

Para cada módulo pode ser definido um nível de prioridade da mais baixa para a mais alta, os módulos com prioridade mais alta geralmente são aplicações mais específicas ou restritas de resposta. O tipo de interação com usuário pode mudar dependendo da necessidade do módulo, além de poder responder a ação com um áudio, pode também enviar um link de uma busca ao seu e-mail, por exemplo. Todos os módulos possuem acesso às informações do usuário, permitindo a aplicação saber sobre datas importantes (agenda), o fuso na qual o usuário se encontra e de outras informações como o e-mail e outros tipos de contas como de redes sociais ou de serviços de outras aplicações. Com esses parâmetros o Jasper irá detectar qual será o módulo usado, para uma determinada entrada e se essas entradas são válidas para algum módulo estabelecido em sua base de informações.

O Módulo de notificação possui interação em estilo passivo, a plataforma do Jasper pode fazer monitoramento nas redes sociais e no e-mail vinculado do usuário, caso possua alguma notificação nessas mídias o Jasper poderá emitir um alerta. Existem ainda uma série de módulos já construídos e disponibilizados na página do Jasper para várias finalidades que podem ser baixadas e testadas.

4.2.2 Segurança

Segundo o site do Jasper as informações do usuário só são usadas para aplicação dos módulos, por exemplo, tem-se o módulo meteorológico que pode informar o clima na área onde estiver o usuário. As informações do usuário ficam armazenadas diretamente no

Raspberry Pi, e elas só podem ser usadas em outros lugares caso o módulo que estiver usado precise utilizar em serviços externos da aplicação do Jasper.

4.2.3 Licença

O Jasper é disponibilizado sobre a licença MIT License, esta licença de software livre permite ao desenvolvedor fazer com o código fonte da plataforma Jasper todas as modificações necessários, sendo que o mesmo deve ser responsabilizado de suas modificações, direitos de uso, cópia, publicação, distribuição, comercialização.

Na licença do Jasper existe um termo informando que ele é fornecido no estado em que se encontra, sem a garantia de fornecimento de suporte ou de atualizações em seu código-fonte. Por fim, os autores não se responsabilizam por qualquer tipo de ação da plataforma, como reclamações, danos, ações judiciais e demais dolos que surgirem a partir de aplicações que utilizem o Jasper.

4.2.4 Instalação

Para instalar o Jasper no Raspbian, deve-se primeiramente atualizar o sistema e instalar a bibliotecas necessárias como libasound2, libportaudio e outras que utilizam o áudio do sistema baseadas em Python, em seguida deve ser configurado a interface de áudio de entrada e saída. Na página oficial do Jasper estão os comandos necessários para realizar a instalação do código fonte do projeto.

Com o núcleo principal do Jasper instalado, deve ser realizado a instalação das dependências, serão os tipos de mecanismos de Speech-To-Text (STT) e Text-To-Speech(TTS) que o Jasper utilizará em seus comandos, assim os mecanismos devem ser devidamente configurados para serem utilizados nos módulos. Por fim deve ser editado o perfil do usuário, com as informações mais relevantes, a partir delas que os módulos do Jasper irão tomar como base, para personificação de suas ações. (JASPER, 2017)

4.3 Comparação por MFCC

4.3.1 Introdução

Baseado no Estudo e Análise de Métodos para Reconhecimento de Palavras, trabalho realizado pelos autores Raiza Arteman de Oliveira Willian de Souza Santos, projeto apresentado em 2016. Uma das aplicações criadas no projeto para o sistema

operacional Ubuntu 15.04 voltado para computadores do tipo pessoal (*Desktop* e *Notebook*), foi realizada a portabilização da aplicação para o Raspberry Pi.

A aplicação utilizada no projeto foi a de comparação por MFCC (coeficientes mel-ceptrais).

O MFCC é uma técnica para extração de atributos *Mel-Frequency Cepstral Coefficients*, que permite criar uma análise de características espectrais de tempo curto, baseando-se em escala de frequência (MEL) que converte os espectros de vozes, essa escala tende a imitar as características do ouvido humano.

Na implementação é realizado o reconhecimento de palavras isoladas em fluxo contínuo independente do locutor para vocabulário de tamanho pequeno (simples palavras).

4.3.2 Reconhecimento de Padrões

Nesta aplicação é utilizado o princípio de reconhecimento de padrões. Para utilizar este princípio existem duas fases diferentes que devem ser aplicadas, a fase de treinamento e a de reconhecimento. Na fase de treinamento são criados os padrões, e ficam armazenados no sistema, servindo como referência. Na fase de reconhecimento, os padrões são calculados e comparados com os padrões armazenados no sistema, deste cálculo obtêm-se o resultado de similaridade entre os padrões. O padrão armazenado que for mais similar com o padrão da fase reconhecimento será o resultado, caso o padrão não seja compatível com nenhum dos padrões armazenados, não será conhecido.

A comparação de padrões para reconhecimento de fala caracteriza-se por sua simplicidade de uso, e por ser um método de fácil aplicação e entendimento, possuindo uma fundamentação matemática aplicada. É um método robusto e aplicável para diferentes tipos de vocabulários, sendo amplamente utilizado.

4.3.3 Filtragem de Ruídos

Para o som ser gravado deve passar por alguns filtros digitais que processará o áudio até ser armazenado. No pré-processamento do áudio existem interferências externas do sinal que devem ser tratadas, para ressaltar as informações úteis do áudio. Também deve ser realizado a normalização da amplitude do sinal, para que as amostras de diferentes altura sejam processados igualmente. Os intervalos de silêncio do áudio devem ser retirados para que só a informação útil seja gravada.

Após a realização do pré-processamento é realizado a extração das características do MFCC, dessa extração teremos os padrões que serão armazenados na aplicação, e posteriormente a comparação com som para reconhecimento. Neste caso é usado um desvio

padrão para estabelecer o reconhecimento entre os padrões.

O método empregado para fazer a comparação é o determinístico, este compara a correlação entre os dois vetores de características do MFCC, quanto menor o resultado de correlação mais semelhante serão os padrões.

4.3.4 Implementação

Na aplicação utilizada que é baseada na linguagem, C são usadas as bibliotecas asoundlib.h utilizada pela captura do áudio e a dirent.h para manipulação das palavras e os arquivos armazenados.

Para aplicação do MFCC foi utilizado a biblioteca ALSA (*advanced linux sound architeture*), os *drivers* de áudio do *kernel* Linux utilizam esta biblioteca, ela fornece uma API de alta eficiência e baixo custo computacional para aplicações de áudio.

O áudio é utilizado no formato WAVE (.wav), neste formato o som é gravado em sequência numérica, isso quer dizer, quando o som é gravado, seus dados são convertidos e armazenados bit a bit. O formato .wav pode ter uma variação na taxa de amostragem (número de amostras por segundo), a taxa de amostragem utilizada é o CD que possui uma taxa de amostras de 44100 (44100Hz) por segundo. A quantidade de bits também interfere na amplitude do sinal. Por exemplo uma gravação de 8 bits terá até 256 níveis de amplitude, já uma gravação de 16 bits possui disponível até 65536 níveis. Para uma aplicação de palavras curtas uma gravação de 16 bits é satisfatória. (MUDA; BEGAM; ELAMVAZUTHI, 2010)

5 DESENVOLVIMENTO

5.1 Iniciando o Raspberry

5.1.1 Preparando o microSD

Inicialmente foi necessário a utilização de um computador para preparar o cartão microSD e realizar a instalação do sistema operacional ou de uma plataforma de desenvolvimento no Raspberry Pi.

O primeiro passo é preparar o cartão microSD, o cartão deve ser formatado usando os sistemas de arquivos no formatos FAT ou FAT32, estes são os formatos que o Raspberry suporta para leitura dos arquivos. Para isso pode-se utilizar algumas ferramentas para fazer a formatação e particionamento do cartão microSD. Nos computadores com distro do Linux pode ser utilizado o *gparted* para formatação do microSD, Já nos computadores com sistema operacional Windows pode ser utilizado um software externo, como SD Formatter (pertencente a SD Association) que é ferramenta gratuita para formatação de SD, sendo disponível para computadores com sistema MAC.

No Raspberry pode-se instalar um sistema operacional de duas formas, uma é utilizado a imagem do sistema operacional desejado para aplicação. E a outra forma é utilizado instalador NOOBS que é disponível no próprio site do Raspberry Pi. Este instalador necessita apenas de um espaço mínimo de 4GB de espaço no cartão microSD e não necessita de internet para instalar o sistema necessitado. Se estiver conectado a internet o NOOBS apresenta um catálogo maior de sistemas operacionais que podem ser instalados no cartão. Entre os sistemas presentes no NOOBS estão: Raspbian, Arch Linux, Risc OS, Windows 10 IoT Core e diversos sistemas de multimídia como OSMC e OpenELEC.

O segunda etapa é inserir a imagem ou os arquivos do sistema operacional no cartão microSD. Para utilizar o NOOBS, deve fazer o download no site do raspberry (https://www.raspberrypi.org/downloads/noobs/), após deve-se extrair os arquivos com um descompactador e copiar os mesmos no microSD já formatado. Se for utilizar a imagem de um sistema operacional, este deve ser gravado com auxílio de algum software ou comando por terminal. Um software de fácil utilização é o Win32 Disk Manager, que permite fazer a gravação da imagem no cartão microSD. Esta ferramenta ainda pode ser utilizada para fazer cópias do microSD por motivos de segurança criando backups do sistema e dos arquivos presentes no Raspberry Pi.

5.1.2 Configurando o Raspbian

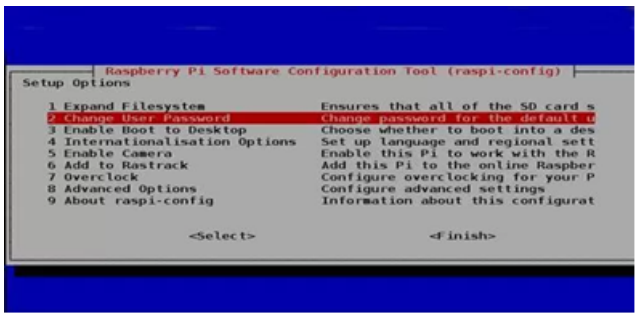
Com microSD formatado e o sistema operacional com a imagem criada, a primeira inicialização do Raspberry Pi será mais demorada, pois irá fazer a instalação do sistema operacional do cartão microSD, e realizando alguns particionamentos para um espaço do sistema operacional, uma para sistema de arquivo, uma para área de *swap* e outras partições que podem ser reservadas para o sistema operacional.

As configurações iniciais do Raspbian sobre língua, fuso e o tipo de teclado são baseados aos utilizados no Reino Unido (país de origem). Existem outras opções que também são de extrema importância para quem necessite fazer personificações na Raspberry. Por padrão o usuário do Raspbian é pi e a senha é Raspberry.

As configurações na Raspbian podem ser feitas de duas maneiras uma utilizando a interface gráfica com um botão de configuração do sistema operacional, onde esta é a maneira mais amigável de fazer as mudanças, por causa da interface gráfica.

A outra maneira é utilizando o terminal do Raspbian, dando o comando **$ sudo raspi-config**. Após o comando irá aparecer uma janela como na Figura 14, nesta janela para mover entre as opções deve-se utilizar as setas do teclado, a tecla Enter para selecionar a opção e a tecla *Tab* para alterar os campos ou mover o cursor para os botões na parte inferior da janela.

Figura 14 – Configuração do Raspberry



Fonte: (RASPBIAN, 2017)

Seguindo a ordem das opções da janela de configuração:

- **Expand Filesystem**: Permita expandir o sistema operacional para que utilize todo o espaço disponível do cartão. Para quem foi utilizar o NOOBS, o sistema de arquivos será expandido automaticamente. Esta opção é utilizada quando um cartão SD é copiado para um maior, não alocado o espaço total ao cartão destinatário. Esta opção precisará reiniciar a Raspberry;

- **Change User Password**: permite alterar a senha do usuário pi no Raspbian, o sistema irá solicitar a senha padrão raspberry;
- **Enable Boot to Desktop**: pode mudar a inicialização do Raspbian, podendo ser utilizado a inicialização para linha de comando, interface gráfica ou diretamente com Scratch;
- **Internationalisation Options**: permite mudar a língua do sistema operacional, o fuso horário o padrão de teclado. Estes estarão em sub-menus;
  - *Change Locale*: permite selecionar o idioma, o conjunto de caracteres associado, etc. No caso do Brasil será a opção pt_BR.UTF-8 UTF-8;
  - *Change Timezone*: permite selecionar o fuso local, por Continente e região e, como *America* e *West*. O Raspbian pode detectar a data e hora a partir da internet automaticamente quando se liga.
  - *Change Keyboard Layout*: permite alterar a configuração do teclado. É recomendável selecionar o padrão genérico com 105 teclas.
- **Enable Camera**: para utilizar o módulo de câmera padrão do Raspberry (entrada flat), deve-se habilitada. Ao ativar a câmera estará ligada diretamente com o processamento da CPU, fornecendo pelo menos 128MB de memória RAM dedicada da GPU.
- **Add to Rastrack**: adiciona a posição do GPS do Raspberry Pi a um mapa mundial, funciona apenas para demonstrar onde está sendo usado Raspberry.
- **Overclock**: permite alterar a velocidade do processador, o *overclock* pode ser selecionado por entre várias configurações. Mas deve ter em mente que a modificação reduzirá a vida útil, e gerando calor no processador, sendo recomendável dissipadores e algum sistema de resfriamento.
- **Advanced Options**: esta opção apresenta sub-menus com configurações importante;
  - *Overscan*: serve para apagar bordas pretas que aparecem em alguns monitores mais antigos;
  - *Hostname*: serve para definir o nome visível do Raspberry em rede;
  - *Enable SSH*: utilizado para acessar o Raspberry remotamente a partir de um cliente SSH. O SSH significa Secure *Shell*, que é uma forma segura de acessar o Raspberry através de uma rede. Esta opção é interessante, pois pode utilizar o Raspberry sem a necessidade de um monitor, teclado e mouse, para controlar o Raspberry diretamente, assim podendo controlar remotamente de um outro computador;

  - *Device Tree*: permite habilitar ou desabilitar dependências em formato de árvore, permite gerenciar e organizar as alocações de recursos e carregamento de arquivos no Raspberry;
  - *SPI*: serve para ativar o uso de circuitos integrados no Raspberry Pi;
  - *I²C*: habilita as interfaces do tipo I²C que estão presentes nos pinos de GPIO, assim podendo fazer o carregamento automático do módulo;
  - *Serial*: habilita mensagens por shell sobre a conexão serial;
  - *Audio*: habilita a saída de áudio através do HDMI ou do JACK 3,5mm (saída de som analógico).
  - *Update*: serve para atualizar o sistema, para atualizações de bibliotecas e de programas instalados. Esta opção faz a mesma coisa que o comando **$ sudo apt - get update**.

- **About Raspi-Config**: mostra uma mensagem sobre as modificações do comando raspi-config.

As modificações presentes no comando raspi-config são as mesma para opção de interface gráfica. Após feitas as modificações deverá clicar no botão <Finish>, se houver alguma modificação no sistema irá pedir para reiniciar se a opção de expansão tiver sido alterado, poderá levar um pouco de tempo a mais para reinicialização do Raspberry.

5.2 Utilizando as Bibliotecas Para Reconhecimento

Neste capítulo é mencionado os recursos que foram utilizados para desenvolver a interação com Raspberry, com a utilização do sistema operacional GNU/Linux Raspbian Jessie.

As bibliotecas citadas no capítulo anterior foram testadas, para avaliar qual teria o melhor desempenho para a uma aplicação o utilizando a Raspberry, e assim sendo escolhida para realizar os testes de reconhecimento das palavras selecionadas.

5.2.1 Periféricos utilizados

Para utilização da Raspberry foi utilizado um teclado e mouse, além de um *headset* (todos de conexão USB). O *headset* possui um microfone embutido, com ele que algumas palavras e comandos puderam ser testados. Foi utilizado o *headset*, pois a captação do áudio gravado pelo seu microfone foi satisfatória para os testes dessa aplicação, e por ser de fácil aquisição e manuseio. A saída de som do *headset* também teve utilização

para interação das bibliotecas e para verificar o armazenamento das amostras salvas na aplicação. Ainda para saída de som, foi testada com caixas de som (as mesmas que são utilizadas em computadores pessoais, tendo a conexão USB para alimentação e o *plug* P2 para a transmissão do áudio), utilizada para a resposta das bibliotecas do Google Assistant e do Jasper.

5.2.2 Usando Jasper

O Jasper por ser uma plataforma *open source*, possibilita ter seu código-fonte baixado e modificado, possui muitas aplicações que utilizam a plataforma como base.

Após a instalação e a configuração inicial do Jasper na Raspberry, o mesmo já pode ser avaliado por uma aplicação de demonstração que está empregado em seu código. A partir desse código de demonstração que foram feitos alguns comandos específicos que consta em sua documentação.

Quando a demonstração é executada, a aplicação fica continuamente escutando o ambiente, até o acionamento da palavra de ativação (*hotword*) que é "Jasper". Após dizer o *hotword*, a aplicação envia um *beep* para que próxima palavra ou frase seja um comando já gravado na aplicação, caso a aplicação não reconhecer o comando que foi informado, ele retornará uma mensagem solicitando que o comando seja repetido ou apenas irá parar de escutar, até ser acionado novamente. Caso o comando for reconhecido pela aplicação ele irá emitir um outro *beep* e logo em seguida a resposta por áudio, os comandos que a essa aplicação inicialmente reconhece está em sua documentação, um exemplo é perguntar que horas são ("*What's the time?*").

A plataforma do Jasper por padrão inicial só aceita comandos de voz e de texto na língua inglesa. Como a plataforma do Jasper é aberta para modificação pode realizar a substituição em sua biblioteca para aceitar outras línguas, e até mesmo poder mudar a palavra de acionamento, podendo utilizar ou não.

Na avaliação dessa biblioteca houve situações em que a aplicação não reconheceu corretamente os comandos que o locutor falou, e algumas vezes o processamento da plataforma ficava lenta, mas retornava a mensagem de resposta. Ocorreram ainda situações em que o Jasper não reconheceu a *Hotword*.

Essas anomalias podem ter ocorrido devido o projeto do Jasper ter iniciado no começo de 2015, e após isso ter tido apenas algumas atualizações pontuais, mas suas bibliotecas não foram mais atualizadas e nesse período tanto a placa da Raspberry quanto o SO do Raspbian tiveram atualizações e novas versões.

5.2.3 Usando Google Assistant Embedded SDK

A biblioteca Google Assistant foi utilizada por ser uma aplicação que possui uma versão específica para o Raspberry Pi 3, e pelo fato de ser uma aplicação da Google Inc. e há a expectativa de fornecer suporte por um tempo maior. Essa biblioteca está disponível desde o começo de 2017, ela basicamente é uma evolução de uma aplicação da própria Google, que era para uma plataforma (Android) e acabou se tornando multiplataforma.

Semelhante a plataforma do Jasper, a Google Assistant após ser instalada e configurada as informações iniciais, disponibiliza uma aplicação de demonstração, baseando-se nela que foi possível avaliar seu desempenho no Raspberry Pi.

A versão de demonstração do Google Assistant ao ser executada, fica esperando até o locutor acionar a *hotword* padrão ("*hey Google*" ou "*Ok Google*"), após o assistente notificará que está esperando um comando válido para aplicação, existe uma lista de comandos que são aceitos para essa demonstração. No caso da Google Assistant é possível fazer simples pesquisas na internet, caso a aplicação de demonstração reconheça o comando, em instantes ele retorna uma mensagem de resposta do comando.

Na avaliação da Google Assistant o reconhecimento de palavras e o processamento de resposta foi melhor que a aplicação de demonstração do Jasper. Para a Google Assistant as mensagens e o comandos só possuem na língua inglesa até momento, por se tratar de uma biblioteca que ainda está adquirindo recursos e atualizações, possivelmente terá uma versão para outras línguas (para as aplicações em Android a tradução somente aconteceu na metade agosto de 2017). Um outro fator importante a ser evidenciado será a compatibilidade de uso com outros recursos e ferramentas disponíveis pela Google Inc.

5.2.4 Comparação por MFCC

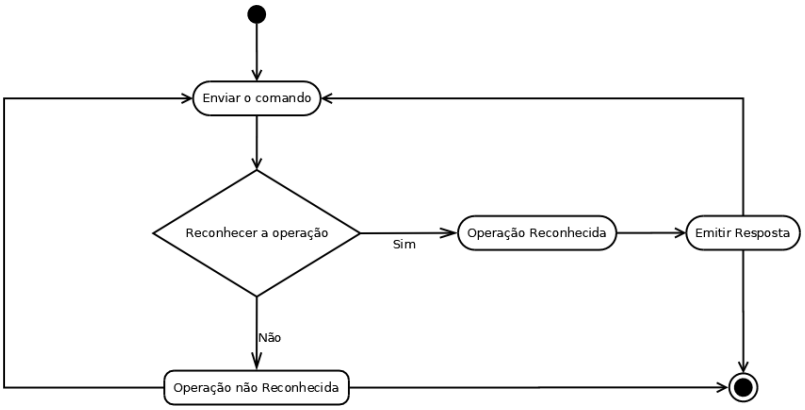
A aplicação de comparação por MFCC, que no projeto foi implementado em um computador convencional no Sistema Operacional GNU/Linux Ubuntu 15.04, utilizando a linguagem C. Necessitando fazer a portabilidade para o Raspbian, como a arquitetura dos dois SOs são semelhantes não houve grandes problemas de dependências de bibliotecas ou de recursos.

Foi necessário fazer algumas mudanças do código para otimizar os resultados comparados e o acréscimo de um temporizador para estabelecer o tempo de processamento para cada comparação. Dentre das mudanças de código, a que representou mudança de significativa para aplicação foi alteração do valor de desvio padrão, este desvio é utilizado para compara dois padrões.

Diferente das outras bibliotecas, com ela pode-se criar um dicionário local com

palavras pré-estabelecidas, e fazer a comparação das palavras. Na figura 15 mostra as atividades neste método.

Figura 15 – Diagrama de Atividade da Comparação do MFCC



Fonte: (Elaborado pelo Autor)

5.3 Comparação Google Assistant SDK e Jasper

Para testar a plataforma que apresente os melhores resultados, foi necessário um teste prévio com 3 locutores. OS locutores deveriam acionar a aplicação pelo *hotword*, e após solicitar a operação. Com estes testes pode-se chegar nos resultados da Tabela 2.

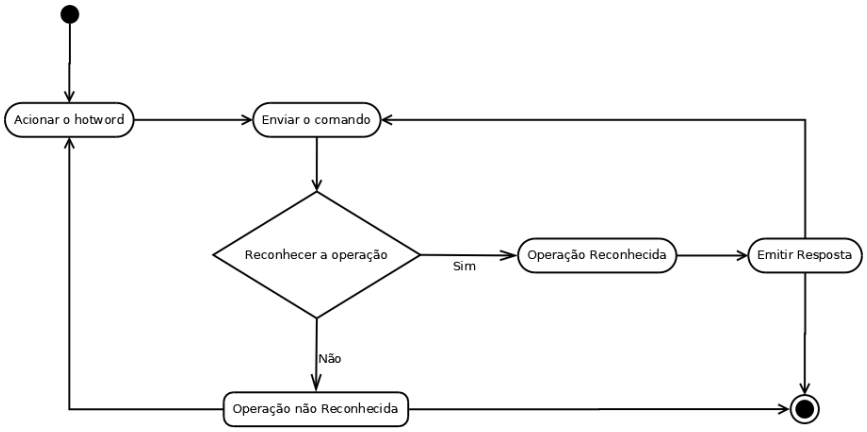
Tabela 2 – Comparação das duas plataforma

Plataforma	Taxa de acerto do *Hotword*	Taxa de acerto do comando
Jasper	25%	15%
Google Assistant SDK	95%	80%

Tanto a Google Assistant quanto o Jasper possui como a principal finalidade de ser um assistente virtual pessoal, mas que podem ser modificados dependendo da necessidade, O Google Assistant possui um pouco mais de restrição, pois seu código-fonte não é totalmente liberado para modificação.

Na figura 16 mostra as atividades do funcionamento da duas plataformas já citadas

Figura 16 – Diagrama de Atividade para o Google Assistant e do Jasper



Fonte: (Elaborado pelo Autor)

6 TESTES E RESULTADOS

Para realizar a avaliação das bibliotecas do Jasper e do Google Assistant foi estabelecido alguns comandos que essas bibliotecas deveriam reconhecer, com auxílio de participantes que os testaram. Os comandos utilizados foram para o teste são as operações que estão pré-definidos e suas versões de demonstração. E analisando o reconhecimento das palavras e número de acertos

Na comparação por MFCC foi testada o número de acertos para o reconhecimento de palavras e o tempo médio tempo de processamento (em segundos), foi criado um dicionário na qual diferentes pessoas gravaram na aplicação.

As palavras *socorro*, *ajuda*, *acidente*, *fogo* e *incêndio* foram escolhidas para testar a aplicação como se fosse um sistema de segurança, por isso foram selecionadas estas palavras, por serem mais utilizadas em momentos de situação de emergência ou perigo. As palavras inseridas apresentam uma sonoridade de fácil compreensão para que estiver falando.

Inicialmente foi inserido no banco da aplicação, as cinco palavras pré selecionadas para a comparação, foram realizados a gravação dessas palavras por cinco locutores (três do sexo masculino e 2 do sexo feminino. Para cada palavra foi inserida nove diferentes tipos de pronúncias (normal, rápido, lento, silábico, silábico lento, silábico rápido, sussurro, emergência e grito), assim cada pessoa inseriu 45 amostras para comparação (5 palavras vezes as 9 pronúncias), ao total foram inseridos 225 amostras para aplicação.

Com o banco concluído, foi testado o reconhecimento das palavras com vozes de pessoas que não gravaram no banco de palavras, para testar o reconhecimento das palavras selecionadas. O teste será realizado em um ambiente fechado e sem barulhos externos, como um laboratório. Com os testes foram calculados a estimativa de acerto e erro com as palavras reconhecidas.

Os teste foram realizados com 6 pessoas de ambos sexo (três do sexo masculino e 3 três do sexo feminino).

Para avaliar o desempenho de reconhecimento e número de amostras foram utilizados cinco diferentes bancos com amostras. O primeiro banco contém 45 amostras (cinco palavras por 9 pronúncias) de uma voz gravada do sexo masculino, o segundo banco contém 90 amostras com duas vozes uma com sexo masculino e outra do feminino, o terceiro com 135 amostras com três vozes, o quarto banco contém 190 amostras com quatro vozes de ambos os sexos e por último a quinta com as 225 amostras de com vozes três masculinas e duas femininas.

Para o reconhecimento das palavras os locutores de teste tiveram que repetir duas vezes a mesma palavra (das cinco inseridas no banco) e falar outras cinco palavras que não estavam cadastradas no banco, essa ordem de palavras teve que ser repetida nos cinco diferentes bancos (cada banco havia acréscimo de uma voz). Assim a aplicação de teste teve que avaliar as quinze amostras de reconhecimento, duas repetições das cinco palavras cadastradas e mais cinco de palavras não cadastradas, isso para cada um dos cinco bancos. Para cada execução da aplicação o locutor de teste teve que repetir o conjunto de 15 amostras para cada um dos cinco bancos, totalizando 75 amostras (15 conjunto para 5 bancos). Para cada comparação os locutores poderiam usar diferentes tipos de entonação de voz. Objetivo principal é fazer que a palavra dita na fase de reconhecimento seja semelhante a uma que está no banco não importando a voz do banco, ou o modo da entonação da palavra reconhecida.

Com essas informações será registrado se a amostra foi reconhecida e qual foi a palavra cadastrada no banco que casou com a palavra dita, e o tempo que aplicação levou para comparar as amostras. Tendo em vista que foram 450 comparações realizadas (75 amostra por 6 locutores). Os resultados podem ser vistos nas tabelas seguintes, para melhor representação foi criada uma tabela para cada banco, nela possui o número de acertos e o tempo médio para comparação. A TMP presente nas tabelas, significa o tempo médio das palavras não identificadas.

Tabela 3 – Resultados Obtidos com 1 Voz

Palavras	Número de acertos	Tempo Médio (seg)
Socorro	4	1,59
Ajuda	9	1,6
Acidente	3	2
Fogo	2	1,13
Incêndio	3	1,93
Falso Positivo	0	
TMP Não identificadas		1,64

Fonte: (Elaborada pelo autor)

Tabela 4 – Resultados Obtidos com 2 Vozes

Palavras	Número de acertos	Tempo Médio(seg)
Socorro	3	3,37
Ajuda	5	3,19
Acidente	1	3,98
Fogo	0	
Incêndio	2	3,6
Falso Positivo	0	
TMP Não identificadas		3,32

Fonte: (Elaborada pelo autor)

Tabela 5 – Resultados Obtidos com 3 Vozes

Palavras	Número de acertos	Tempo Médio (seg)
Socorro	1	4,76
Ajuda	3	4,85
Acidente	1	6,48
Fogo	0	
Incêndio	3	5,92
Falso Positivo	1	4,49
TMP Não identificadas		5,17

Fonte: (Elaborada pelo autor)

Tabela 6 – Resultados Obtidos com 4 Vozes

Palavras	Número de acertos	Tempo Médio (seg)
Socorro	3	6,19
Ajuda	8	7,08
Acidente	4	7,99
Fogo	5	6,37
Incêndio	5	7,28
Falso Positivo	1	8,22
TMP Não identificadas		6,67

Fonte: (Elaborada pelo autor)

Tabela 7 – Resultados Obtidos com 5 Vozes

Palavras	Número de acertos	Tempo Médio (seg)
Socorro	4	8,39
Ajuda	5	8,38
Acidente	10	9,4
Fogo	1	8,91
Incêndio	2	10,23
Falso Positivo	2	9,32

Fonte: (Elaborada pelo autor)

Dessas cinco tabelas, obteve-se uma a tabela abaixo com os dados gerais do teste, nele consta a porcentagem de acerto e o tempo Médio para cada banco testado.

Tabela 8 – Resultado Geral dos Bancos de Vozes

Número de vozes	Porcentagem de acertos	Tempo Médio (seg)
1	42%	1,65
2	22%	3,49
3	18%	5,28
4	52%	7,11
5	48%	8,98

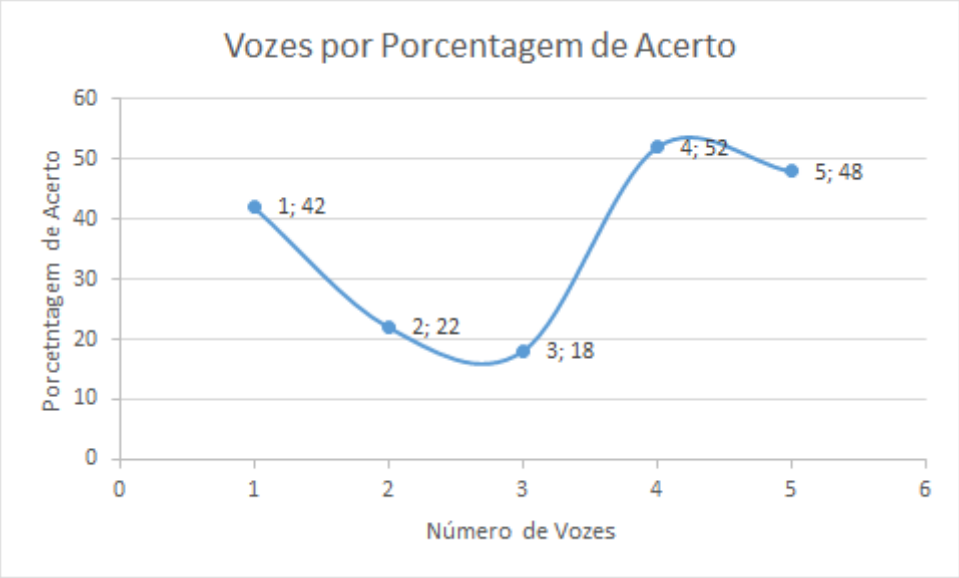
Fonte: (Elaborada pelo autor)

Para melhor representação foi feito dois gráficos, o primeiro na figura 17 com relação ao número de vozes por tempo médio de reconhecimento. E o segundo (figura 18) o número de vozes por porcentagem de acertos.

Figura 17 – Gráfico de Número de Vozes por Tempo Médio

Fonte: (Elaborado pelo Autor)

Figura 18 – Gráfico de Número de Vozes por Número de Acertos



Fonte: (Elaborado pelo Autor)

Com essas informações foi possível analisar o resultado do teste de aplicação para o reconhecimento de palavras, apesar do número de acertos ser abaixo do esperado a aplicação tende a ser útil para o reconhecimento de palavras.

As palavras de fonemas diferentes tiveram melhor reconhecimento pela aplicação. Nos casos de falso positivo, ocorreu quando o locutor falou a palavra fogo e a aplicação reconheceu como fosse socorro. O aumento crescente do tempo de execução se dá ao fato ao grande número de amostras, onde aplicação tende a escolher a amostra que mais próxima do padrão da amostra que será reconhecida. Analisando os padrões que foram reconhecido, possivelmente a aplicação poderia apresentar melhores resultados caso fosse gravado um com número maior de vozes, mas com um número menor entonações.

7 CONCLUSÃO

O trabalho apresentado teve como objetivo estudar e desenvolver um protótipo para reconhecimento de palavras para um sistema embarcado, no caso foi utilizado o Raspberry Pi 3 como arquitetura para desenvolvimento. As bibliotecas utilizadas para fazer o reconhecimento de palavras independente de voz, foram utilizados a plataforma do Jasper, a Google Assistant Embedded SDK e a comparação por MFCC.

Analisando e testando essas bibliotecas foi possível avaliar qual teve o melhor desempenho e aplicabilidade para este projeto, com os resultados dos comandos que foram reconhecidos pela aplicações do Jasper e do Google Assistant. Na comparação por MFCC foi possível criar bancos com diferentes palavras e diferentes tipos de vozes e entonações.

Na plataforma do Jasper os comandos eram reconhecidos, mas aplicação leva bastante tempo para reconhecimento e muito das vezes não reconhecia o comando solicitado. O Google Assistant apresentou melhores resultados de processamento e reconhecimento de comandos, possuindo uma aplicação mais abrangente com relação a recursos e comandos.

Na comparação por MFCC apesar da quantidade baixa de números de acertos, a aplicação foi satisfatória, alguns fatores podem ser modificados, uma delas são os tipos de coeficientes utilizados para criação de cada padrão de palavra, pode ser alterado o desvio padrão de casamento das palavras, permitindo que seja diminuindo ou aumentando essa tolerância, o valor utilizado foi de um ponto médio. Teria que ser avaliado até que momento esse desvio pode ser modificado, que permita um maior número de acertos, sem ocasionar em resultados falsos positivos.

Sobre o Jasper apesar de ser uma plataforma onde os resultados não foram os esperados, e devido a uma falta de continuação e atualização deste projeto, sugiro que ainda deva ser estudado, pois se trata de uma aplicação *open-source*, e pode haver algo em sua arquitetura que possa ser reutilizado e outros projetos. Já o Google Assistant permite integrar diversas recursos e ferramentas de sua plataforma, podendo ser criadas diversas aplicações para as mais diversas áreas de atuação, por se tratar de uma empresa, a Google fornece planos de livre desenvolvimento, poderá ser cobrado, caso aplicação seja colocada em produção.

Para futuros trabalhos, posso sugerir baseado com o estudo desse trabalho, a implementação de aplicações para sistemas embarcados para arquiteturas mais simples, onde pode ser empregado um sistema de reconhecimento de fala em uma arquitetura de hardware mais simples, podendo até ser totalmente encapsulado, para essa finalidade. Também pode ser estudado o desempenho e a funcionalidade das bibliotecas citadas para aplicações em diversas áreas, para emprego de sistemas embarcados, como assistentes

residenciais, sistema de vigilância por captura de som e demais áreas afins.

REFERÊNCIAS

ALMEIDA, M. B. *Sistemas embarcados com Linux*. [S.l.]: Linux a Bordo, 2008.

ANDRADE, G. E. Arquitetura de sistemas embarcados. 2013.

ARAÚJO, T. *Raspberry Pi B+: Introdução a Porta GPIO*. [S.l.]: Fazedores, 2014. Disponível em<http://blog.fazedores.com/raspberry-pi-b-introducao-porta-gpio/>. Acesso em 28/06/2017.

ARCH, L. *Arch Linux - Uma distribuição simples e leve*. 2017. Disponível em<https://www.archlinux.org/>. Acesso em 28/07/2017.

ARM, D. *Cortex-A53*. 2017. Disponível em<https://developer.arm.com/products/processors/cortex-a/cortex-a53>. Acesso em 20/06/2017.

BANAKAR, R. et al. Scratchpad memory: design alternative for cache on-chip memory in embedded systems. In: ACM. *Proceedings of the tenth international symposium on Hardware/software codesign*. [S.l.], 2002. p. 73–78.

CANONICAL, L. *Snappy Ubuntu Core*. [S.l.]: Yoctov Project, 2017. Disponível em<https://docs.snapcraft.io/core/>. Acesso em 29/07/2017.

CHASE, O. *Sistema Embarcados*. [S.l.: s.n.], 2007. Www.sbajovem.org.

CONTI, F. *Microinformática*. [S.l.]: Universidade Federal do Pará (UFPA), 2014. Disponível em<http://www.ufpa.br/dicas/mic/mic-proc.htm>. Acesso em 15/06/2017.

CYPRESS, E. T. Single-chip ieee 802.11ac b/g/n mac/baseband/radio with integrated bluetooth 4.1 and fm receiver. *IEEE*, 2017.

DOUGLASS, B. P. *Real-time UML: developing efficient objects for embedded systems*. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 1997.

EMBARCADOS, E. *Sistema Embarcado - O que é? Qual a sua importância?* 2013. Disponível em<https://www.embarcados.com.br/sistema-embarcado/>. Acesso em 26/04/2017.

EMBARCADOS, S. O. *Processadores: Arquitetura RISC e CISC*. 2015. Disponível em<http://www.sistemasembarcados.org/2015/11/15/processadores-arquitetura-risc-e-cisc/>. Acesso em 20/06/2017.

GARETT, F. *Raspberry Pi: conheça os modelos e saiba qual o mais indicado para você*. 2016. Disponível em<http://www.techtudo.com.br/listas/noticia/2016/03/raspberry-pi-conheca-os-modelos-e-saiba-qual-o-mais-indicado-para-voce.html>. Acesso em 29/06/2017.

GERVINI, A. I. et al. Avaliação de desempenho, área e potência de mecanismos de comunicação em sistemas embarcados. *SEMISH'03–XXX Seminário Integrado de Software e Hardware*, 2003.

GOMAA, H. *Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures*. 1. ed. [S.l.: s.n.], 2004. ISBN 978-0-201-77595-2.

GONÇALVES, B. *Unidade Central de Processamento (CPU)*. [S.l.]: Departamento de Informática-Universidade Federal do Espírito Santo (UFES), 2008. Disponível em<https://goo.gl/99sDVb>. Acesso em 15/06/2017.

HEIN, W. Raspberry pi aplicado a projetos do mundo real. *Linux Magazine*, 2013.

HOWSE, J. *OpenCV for Secret Agents*. [S.l.]: Packt Publishing Ltd, 2015.

III, J. J. D.; STUBBERUD, A. R.; WILLIAMS, I. J. *Sistemas de controle*. [S.l.]: Bookman Editora, 1981.

JASPER. *Control Anything With Your Voice*. 2017. Disponível em<http://jasperproject.github.io/documentation/>. Acesso em 02/08/2017.

JUNIOR, A. S. H.; WANNER, L. F.; FRÖHLICH, A. A. Gerenciamento do consumo de energia dirigido pela aplicação em sistemas profundamente embarcados. *Laboratório de Integração Software/Hardware-Campus Universitário - UFSC*, 2006.

LI, Q.; YAO, C. *Real-time concepts for embedded systems*. [S.l.]: CRC Press, 2003.

MARINHO, D. *Entenda os benefícios do ADS-B, sistema de vigilância aérea a ser utilizado em breve no País*. [S.l.]: Departamento de Controle dp Espaço Aéreo, 2015. Disponível em<https://www.decea.gov.br/blog/?p=418>. Acesso em 28/04/2017.

MICROSOFT, D. *Windows 10 IoT Core - The operating system built for your Internet of Things*. 2017. Disponível em<https://developer.microsoft.com/pt-br/windows/iot>. Acesso em 29/07/2017.

MORIMOTO, C. E. *Entendendo os sistemas embarcados*. 2007. Disponível em<http://www.hardware.com.br/artigos/entendendo-sistemas-embarcados/>. Acesso em 15/04/2017.

MUDA, L.; BEGAM, M.; ELAMVAZUTHI, I. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *arXiv preprint arXiv:1003.4083*, 2010.

OMSC, T. *OSMC*. [S.l.]: Team Kodi, 2017. Disponível em<https://osmc.tv/about/>. Acesso em 29/07/2017.

PI, R. *Raspberry Pi Model B*. 2015.

PORTUGAL, C. R. P. *Raspberry Pi*. 2017. Disponível em<http://www.raspberrypiportugal.pt/raspberry-pi/>. Acesso em 28/05/2017.

PORTUGAL, C. R. P. *Raspberry Pi 3*. 2017. Disponível em<http://www.raspberrypiportugal.pt/raspberry-pi-3/>. Acesso em 28/05/2017.

RASPBERRY, P. Raspberry pi. *Raspberry Pi*, v. 1, p. 1, 2013.

RASPBIAN, O. *Sistema Raspbian*. [S.l.]: Debian Project, 2017. Disponível em<https://www.raspbian.org/>. Acesso em 28/07/2017.

RICHARDSON, M.; WALLACE, S. *Primeiros Passos com o Raspberry Pi*. Primeira edição. [S.l.]: Editora Novatec, 2013. ISBN 978-85-7522-345-1.

SILVA, J. M. G. *Ajuste de Controladores PID*. [S.l.]: Departamento de Engenharia Elétrica-Universidade Federal do Rio Grande do Sul (UFRGS), 2000. Disponível em<http://www.ece.ufrgs.br/ jmgomes/pid/Apostila/apostila/apostila.html/>. Acesso em 07/06/2017.

STADZISZ, P. C.; RENAUX, D. P. B. *sofware Embarcado*. [S.l.]: Universidade Tecnológica Federal do Paraná (UTFPR), 2014.

TECHTUDO. *Google Assistente*. 2017. Disponível em<http://www.techtudo.com.br/tudo-sobre/google-assistant.html>. Acesso em 20/08/2017.

A OUTROS MODELOS

A.0.1 Raspberry Pi 1 Model A+

Esse modelo foi lançado em novembro de 2014, este microcontrolador possui algumas limitações de conexão com dispositivos periféricos. Esse modelo é uma versão atualizada dos modelos originais, tendo a limitação de processamento de memória de 512MB RAM, operando na frequência de 400MHz, dividindo-se entre a CPU e GPU (Processador Gráfico), possui o processador Broadcom BCM2835 Processor, de 700MHz single core ARM1176JZF-S CPU, este processador é do grupo RISC de 32 bits, está integrado a GPU Dual Core VideoCore IV.

Em relação às interfaces de expansão ou portas de conexão de entrada, neste modelo possuem a porta GPIO com 40 pinos, diferente dos primeiros modelos que possuíam apenas 26 pinos, possui uma conexão USB 2.0. Ainda possui uma porta CSI para conexão de câmera específica para a Raspberry e uma porta DSI para conexão de uma tela *touch screen*. Há também as portas para saída de vídeo HDMI e para o som existe um conector Jack (entrada de fones do tipo P2), além do *slot* para o cartão de memória microSD (no modelo A utilizava um slot SD). A fonte de energia necessária para o funcionamento deve ser de 5V com 600mA (diferente do primeiro modelo que precisava de 750mA), utiliza até 45% menos de energia do que o modelo B+.

Por ser o Model A+ da primeira geração, é um modelo mais antiga, simples e de menor poder computacional. As suas dimensões são 65x56x12mm. Para utilizar o acesso à internet via Wi-Fi deve ser utilizado um *dongle* USB (que tem alcance de captar as redes Wi-Fi).

Figura 19 – Raspberry Pi 1 Model A+

Fonte: (RASPBERRY FOUNDATION, 2017)

A.0.2 Raspberry Pi Zero

Este é o modelo mais simples e barato lançado pela Fundação Raspberry Pi, porém possui hardware mais poderoso se comparado com Model A+, mesmo tendo o preço mais baixo e quase metade do tamanho.

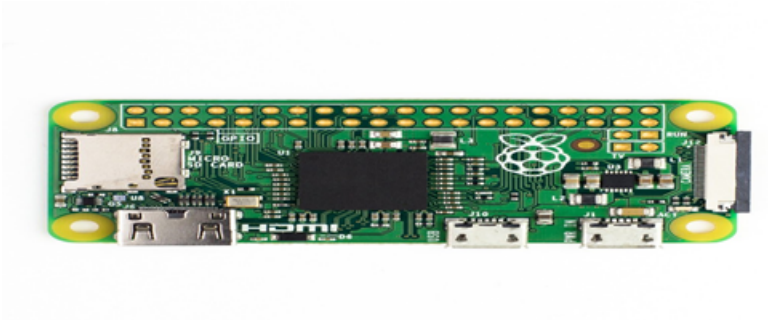
O processador é o mesmo utilizado no Pi 1(Broadcom BCM2835), tendo o *clock* de até 1GHz, sendo até 40% mais rápido, com os mesmos 512MB de memória RAM. Possui um microHDMI (para utilizar nos monitores deve possuir o cabo, ou o adaptador para HDMI)com capacidade de saída de vídeos em 1080p e 60 quadros por segundo, também utiliza 40 pinos de GPIO, mas que são *unpopulated* (inabitadas) tendo que ser fixadas ou soldadas para serem utilizadas, pode utilizar pinos que são pré-soldados.

As restrições dessa versão são causadas pelo seu próprio tamanho, possuindo 65mm x 30mm x 5mm. Com pouco espaço, poucos recursos puderam ser empregados na placa, na placa possui apenas uma entrada microUSB, uma entrada para o cartão microSD e a saída de som (no formato P2). Assim recursos como portas USBs, porta ethernet, conector DSI (para tela) e conector CSI (para câmera) tiveram que ser abolidos, com isso ele consome uma quantidade menor de energia sendo necessário uma fonte que utiliza 5V com 140mA.

No modelo Pi zero existe uma variante que possui conexão Wi-Fi e Bluetooth on-board (integrado na placa), mantendo as mesma características físicas da versão original, essa versão é chamada de Raspberry Pi Zero W ou Pi Zero Wireless. Lembrando que existem conectores e adaptadores para este microcontrolador que podem ser utilizados, assim tendo acesso a mais recursos e periféricos.

Apesar de ser mais barata que o modelo A+ e de maior poder computacional, pode ser de difícil utilização por apresentar poucas portas de expansão, mas pode compensar pelo seu tamanho enxuto para alguns projetos.

Figura 20 – Raspberry Pi Zero



Fonte: (RASPBERRY FOUNDATION, 2017)

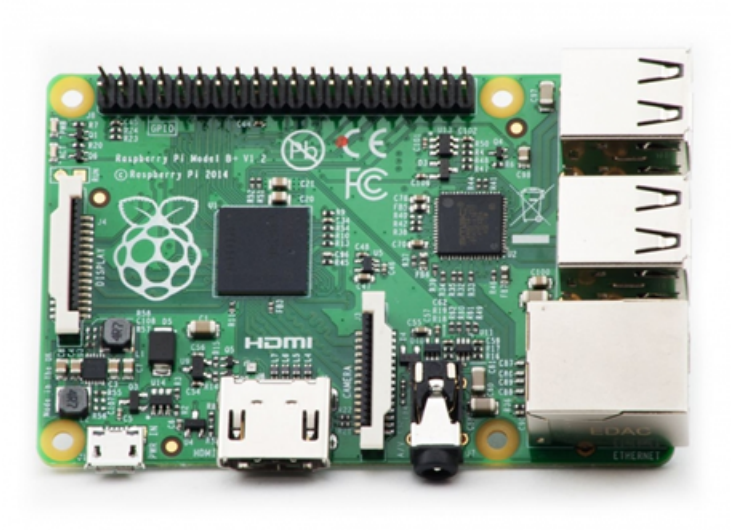
A.0.3 Raspberry Pi 1 Model B+

Lançado em julho de 2014, este modelo é uma atualização do Pi 1 Model B. O Pi 1 Model B+ é muito semelhante ao modelo Pi 1 A+ utiliza o mesmo chip Broadcom BCM2835 de procesador ARM11 à 700Mhz com 512 de memória RAM e a GPU VideoCore IV com capacidade de reprodução de vídeo em 1080p HD no conetor HDMI. Este modelo utiliza 40 pinos de GPIO, as primeiras 26 são compatíveis com o modelo B, também possui o *slot* MicroSD para carregar o sistema operacional e para armazenamento de dados, a porta Ethernet 10/1000 para conexão em rede, os conectores CSI e DSI para conexão da câmera e do *touch screen*, o conector P2 para saída de áudio.

A diferença fica pelas 4 portas USB 2.0 e pelo consumo de 600mA pelo conector microUSB, sendo mais eficiente que o modelo B. Em relação ao tamanho, possui tamanho um pouco maior devido as 4 portas USB, possuindo 85mm x 56mm x 17mm.

Este modelo é bastante interessante por possuir as 4 entradas para USB, podendo utilizar mais periféricos diretamente na placa. Esse modelo dificilmente é encontrado no mercado, pois foi substituído pelas gerações sucessoras.

Figura 21 – Raspberry Pi 1 Model B+



Fonte: (RASPBERRY FOUNDATION, 2017)

A.0.4 Raspberry Pi 2 Model B

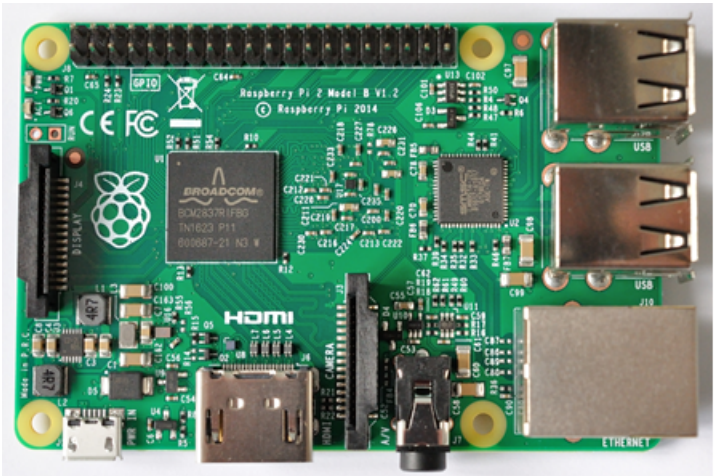
Esse é o modelo da segunda geração dos computadores desenvolvidos pela Raspberry Pi, essa versão substitui o modelo Pi 1 Model B+ em fevereiro de 2015, a última versão desse modelo é denominado Pi 2 Model B V1.2, ele se destaca pelo aumento de desempenho em relação às versões anteriores, isso devido pelo chip Broadcom BCM 2837 que contém o processador (CPU) quad-core de 64bits ARM Cortex-A53 de 900 MHz e o acelerador

gráfico (GPU) VideoCore IV de 400 MHz. As versões anteriores do Pi 2 model B utilizavam o chip da Broadcom BCM2836 que era semelhante a arquitetura do BCM2835, utilizando o processador quad-core Cortex-A7 também de 900 MHz. O desempenho do Pi 2 model B teve um ganho de velocidade de até 6 vezes com substituição do chip para o BCM2837, ele é acompanhado da memória de 1GB RAM operando a 400 MHz.

Esta placa oferece o mesmo conjunto de interfaces de extensão do modelo antecessor, o Pi 1 Model B+, tendo os: 40 pinos estendidos da GPIO, quatro portas USB 2.0, interface Ethernet, HDMI, saída de som (P2), CSI (conexão para câmera em formato *flat*), DVI (conexão para tela *touch* em formato *flat*), além do *slot* para o cartão de memória microSD e do microUSB para alimentação, sendo necessário uma fonte de 5V com 1.8A.

Esse modelo até terceira geração dos microcontroladores da Raspberry possuíam o hardware mais potente pela utilização do processador e arquitetura recente para época, fazendo dele até um mini desktop compatível com alguns sistemas operacionais como Raspbian, RaspBMC, Android, Arch Linux, RISC OS, OpenElec, Pidora e o Windows 10 (versão utilizada para internet da coisas). Essas características fazem do Pi 2 Model B, como um alternativa para rodar aplicações mais complexas que necessitam do suporte de um sistema operacional, dando ao usuário uma interface mais acessível e transparente.(PI, 2015)

Figura 22 – Raspberry Pi 2 Model B



Fonte: (RASPBERRY FOUNDATION, 2017)

A.0.5 Compute Module IO Board V3

Este modelo na verdade é um kit de desenvolvimento, para que deseja fazer do Raspberry Pi um produto mais flexível e de grande poder computacional, é destinado para aplicações industriais. O IO Board V3 funciona apenas como placa de entrada e saída, pode ser utilizado com os módulos de processamento CM3, CM3L e tem suporte para

o CM1. Esta placa possui 120 pinos de GPIO, porta HDMI, duas portas para câmeras (CSI), duas portas para *display* (DSI) e um conector USB do tipo A.

O processamento dessa placa, depende do tipo de memória que for utilizado, o Compute Module 1 (CM1) é equivalente ao Raspberry Pi 1 com o *chip* BCM2835 de 512 RAM com 4GB eMMC (memória Flash, que equivale o SD no Pi 1), todos esses componentes são integrados em pequeno módulo de 67.6 x 30mm com o conector DDR2 SODIMM (mesmo utilizado para memórias de notebook). Essa memória é conectada diretamente no slot da do Board V3.

O Compute Module 3 (CM3) é equivalente ao Raspberry Pi da terceira geração possui o chip BCM2837 de 1GB RAM com 4GB eMMC, tendo a velocidade de 1.2GHz, também como no CM1 está tudo empregado em módulo de 67,6 x 31 mm de conector DDR2. Existe versão variante do CM3 denominada de CM3L (versão *Lite*) que não possui os 4GB de eMMC, mas possui uma interface com pinos onde o usuário pode conectar um eMMC ou um cartão SD.

Por comportar mais dispositivos o IO Board V3 precisa de uma alimentação 5V à 2,5A via microUSB.

O modelo IO Board V3 é uma placa de fácil personalização de componentes periféricos e de processamento, para quem precisa de alto desempenho e uma grande quantidade de dispositivos de entrada e saída. Mas por ser um placa mais robusta e por possui módulos de memórias, que são adquiridos a parte o seu preço de aquisição é muito superior do que os seus semelhantes para projetos pessoais.

Figura 23 – Compute Module IO Board V3 com a placa CM3

Fonte: (RASPBERRY FOUNDATION, 2017)