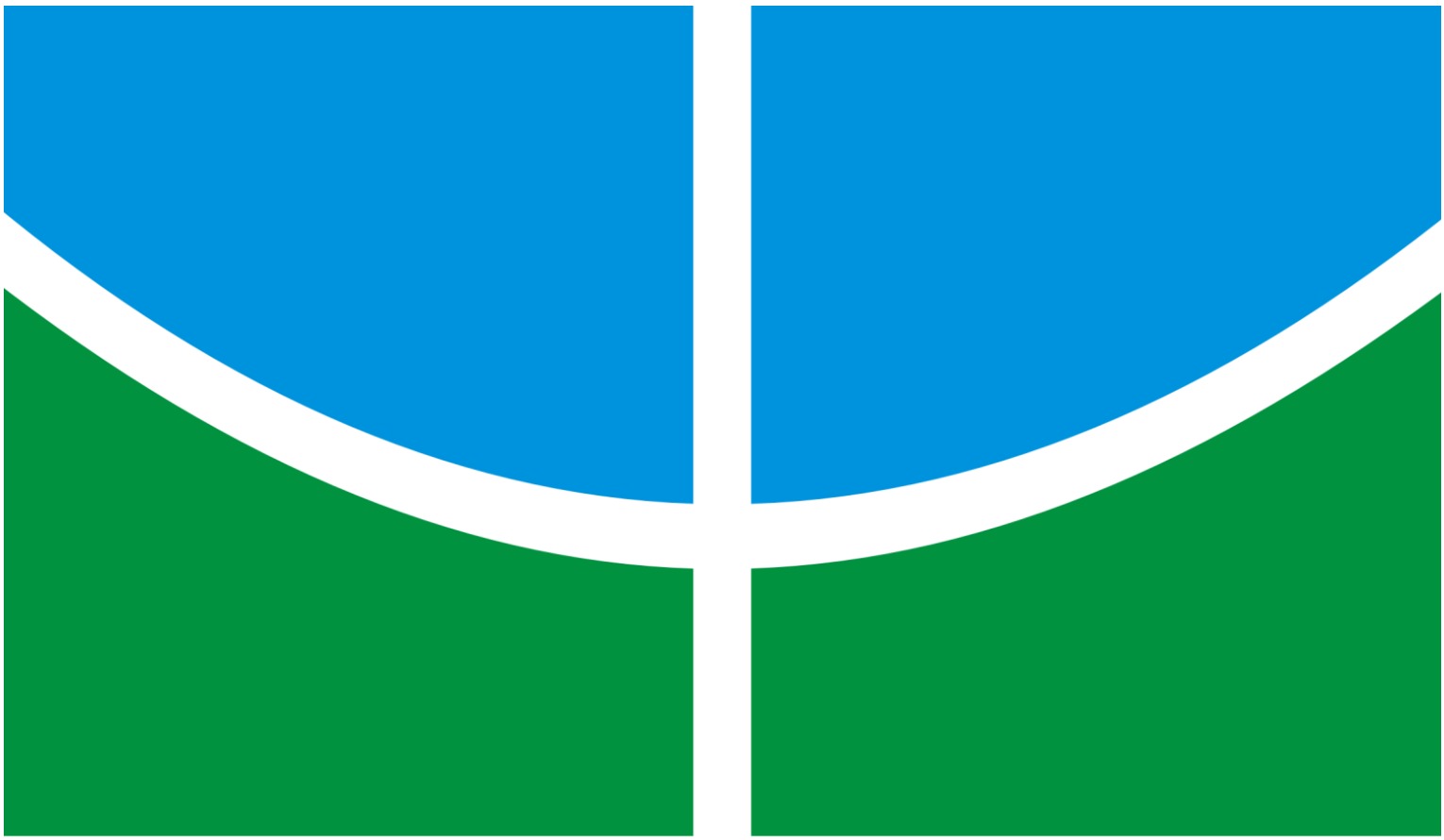

PROJETO DE GRADUAÇÃO

SISTEMA DE SEGURANÇA INTEGRADO À REDE MÓVEL CELULAR

Rodrigo Rozário Bezerra

Brasília, 28 de fevereiro de 2014

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia
ENE – Departamento de Engenharia Elétrica

PROJETO DE GRADUAÇÃO

Sistema de Segurança Integrado à Rede Móvel Celular

Rodrigo Rozário Bezerra

RELATÓRIO SUBMETIDO AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO REQUISITO PARCIAL PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE REDES DE COMUNICAÇÕES

Aprovada por

Prof. D. Sc. Ricardo Zelenovsky, PUC-RJ, UnB/ENE
Orientador

Prof. D. Sc. Ugo Silva Dias, UnB/ ENE
Examinador interno

Prof. D. Sc. Alexandre Ricardo Soares Romariz, UnB/ ENE
Examinador interno

Brasília, 28 de fevereiro de 2014

FICHA CATALOGRÁFICA

BEZERRA, RODRIGO ROZÁRIO
Sistema de Segurança Integrado à Rede Móvel Celular [Distrito Federal] 2014
X, 69p., 210 x 297 mm (ENE/FT/UnB, Engenharia de Redes de Comunicação).
Monografia de Graduação – Universidade de Brasília, Faculdade de Tecnologia
Departamento de Engenharia Elétrica

1. Hardware
2. Arranjos de Microfones
3. Amplificadores
4. Filtros

I. ENE/FT/UNB
II. Título (Série)

REFERÊNCIA BIBLIOGRÁFICA

BEZERRA, RODRIGO R. (2014). Sistema de Segurança Integrado à Rede Móvel Celular, Relatório de Graduação em Engenharia de Redes de Comunicações, publicação XXXXXXXXX, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 69p.

CESSÃO DE DIREITOS

AUTOR: Rodrigo Rozário Bezerra
TÍTULO: Sistema de Segurança Integrado à Rede Móvel Celular
GRAU: Engenheiro de Redes de Comunicação
ANO: 2014

Rodrigo Rozário Bezerra

UnB – Universidade de Brasília
Campus Universitário Darcy Ribeiro
FT – Faculdade de Tecnologia
ENE – Departamento de Engenharia Elétrica
Brasília – DF – 70919-970
Brasil

Dedicatória

Este trabalho é para todos que fizeram parte a da minha vida e que, com toda certeza, contribuíram para a minha formação pessoal e professional. Aos amigos que fiz durante esses longos anos de dedicação. Também para os que ficam, como incentivo a nunca desistir. Aos professores que em muito contribuíram para a minha formação.

Rodrigo Rozário Bezerra

Agradecimentos

Gostaria de agradecer a toda a minha família que me deram apoio durante toda a minha vida. Em especial aos meus pais, que mesmo nos momentos difíceis tinham palavras de incentivo. Com grande admiração, agradeço a todos os professores que participaram da minha formação.

Rodrigo Rozário Bezerra

RESUMO

Este trabalho apresenta uma proposta de integração para sistemas de vigilância eletrônica com a rede celular GSM. Durante o trabalho, mostraremos que os sistemas atuais são altamente padronizados e com funções amplamente difundidas em soluções já lançadas no mercado e que atendem bem alguns tipos de aplicações. Este trabalho propõe novas formas de integração, acesso e agregar a possibilidade de desenvolvimento de novas funcionalidades em um ambiente que utiliza totalmente *software* livre.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

Esta seção apresenta considerações preliminares relacionadas ao desenvolvimento do trabalho e sua motivação.

1.1 CONSIDERAÇÕES INICIAIS

Uma pesquisa realizada pelo Instituto de Pesquisa Econômica e Aplicada (Ipea) mostrou que nos últimos dez anos a segurança privada cresceu 74% no Brasil. Quase R$ 40 bilhões são gastos com seguro e contratação de trabalhadores de segurança [1].

O aumento no setor pode ser atribuído ao crescimento dos níveis de violência que tem acontecido nos últimos anos. Em pesquisa realizada pela CNI (Confederação Nacional da Indústria), em parceria com o IBOPE, aponta que 30% (para pessoas que tem renda familiar de até um salário mínimo) e 27% (para pessoas que tem renda familiar acima de 10 salários mínimos) dos entrevistados escolheram o item Combater a Criminalidade e a Violência para prioridade do governo federal em 2014 [2].

Fica claro que a população tem investido mais em segurança privada, tanto nas residências quando nos estabelecimentos comerciais. Nesse sentido, câmeras de vigilância têm um papel importante como fator inibidor de furtos, roubos e outros tipos de crimes.

Há alguns meses, vimos também a execução (ainda não concluído) de projeto para utilização de câmeras de vídeo por parte da Polícia Militar do Distrito Federal para monitorar atitudes suspeitas em locais públicos. Da central de monitoramento, os agentes poderiam acionar uma viatura para uma ocorrência.

Inicialmente, os sistemas de segurança e vigilância eletrônica faziam uso de cabos coaxiais e DVRs (*Digital Video Recorder*) que ainda são utilizados em muitos locais. Diversas soluções para vigilância eletrônica têm surgido nos últimos anos devido a grande necessidade de monitorar ambientes remotamente. A disseminação e capilarização da Internet dentro de instituições, residências ou até mesmo na rua torna possível sua utilização para prover acesso a diversos dispositivos de vigilância instalados dentro de redes locais.

A câmera de vídeo IP, ou simplesmente Câmera IP, é uma câmera de vídeo que pode ser acessada e controlada por qualquer dispositivo conectado à rede IP, seja em uma LAN (*Local Area Network*), Intranet ou Internet. Essas são basicamente computadores com um servidor *web* interno conectado à uma câmera de vídeo. Os modelos atuais são compatíveis com tecnologias Ethernet e Wi-Fi e em alguns modelos podem até ser acessadas remotamente através da Internet.

No Brasil, redes GSM (*Global System for Mobile Communications*) estão instaladas na maioria das cidades que possuem serviço celular móvel. No mundo, é a tecnologia mais popular para comunicações móveis. No Brasil, vemos na Figura 1 e na Tabela 1 que a maior parcela de dispositivos celulares são da tecnologia GSM.

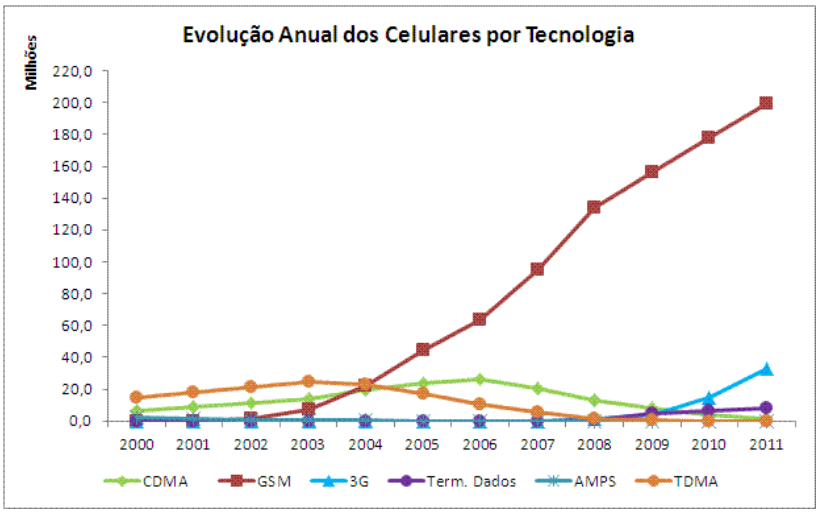

Figura 1 Evolução Anual de Celulares por Tecnologia

Fonte: teleco.com.br/tecnocel.asp

Tabela 1 Distribuição do Número de Celulares Por Tecnologia
Fonte: teleco.com.br/ncel.asp

Tecnologia	Dezembro 2012	Dezembro de 2013			
		Nº Celulares		Cresc. mês	Cresc. ano
GSM*	195.731.115	159.674.015	58,90%	(3,8%)	(18,4%)
3G (WCDMA)*	52.467.528	94.763.509	34,96%	7,1%	80,6%
LTE	-	1.309.771	0,48%	41,8%	-
CDMA*	125.060	21.637	0,01%	(9,0%)	(82,7%)
Total Terminais de Dados	13.484.200	15.330.867	5,66%	0,9%	13,7%
- Term. Dados Banda larga	6.717.538	7.034.289	2,59%	0,1%	4,7%
- Term. Dados M2M	6.766.662	8.296.578	3,06%	1,5%	22,6%
Total	**261.807.903**	**271.099.799**	**100,00%**	**0,2%**	**3,5%**

Ainda sobre a penetração da rede celular, o *ranking* da Tabela 2 mostra as dez cidades com maior densidade de celulares, calculada com base em um grupo de 100 habitantes. Podemos ver que o distrito federal em dezembro de 2013 tinha 222,95 celulares para um grupo de 100 habitantes, que mostra que cada habitante, em média, possui mais de 2 celulares.

Tabela 2 Densidade de Celulares Por Estado Por 100 habitantes
Fonte: teleco.com.br/nceluf.asp

--	(Milhares)	Dez/12	Nov/13	Dez/13
1	Distrito Federal	220,52	221,42	222,95
2	São Paulo	150,12	154,15	153,70
3	Mato Grosso do sul	149,51	151,98	152,56
4	Rondônia	150,11	151,94	152,53
5	Goiás	144,37	148,31	149,00
6	Rio de Janeiro	143,17	148,39	148,63
7	Rio Grande do sul	140,12	145,05	145,70
8	Mato Grosso	140,70	140,95	141,32
9	Tocantins	134,85	139,80	141,32
10	Rio Grande do Norte	132,75	136,98	137,23

A densidade média para todo o Brasil é de 136,24 celulares para um grupo de 100 habitantes, com um total de 271.100 celulares (dados de dezembro de 2013).

O esquema GSM diferencia-se dos seus antecessores pois os canais de voz são tratados de forma digital, caracterizando-o como um sistema celular de segunda geração (2G). Este sistema utiliza dois conjuntos de frequências na faixa dos 900 MHz: a primeira faixa compreende de 890 a 915 MHz, que é utilizados para transmissões entre o terminal e a estação rádio base (ERB), e a segunda faixa compreende de 935 a 960 MHz, para transmissões da rede.

O GSM utiliza simultaneamente dois métodos de acesso ao meio, combinando o TDMA (*Time Division Multiple Access*) e o FDMA (*Frequency Division Multiple Access*). O FDMA divide os 25 MHz disponíveis em 124 canais de 200 kHz e é utilizado um ou mais canais em cada estação rádio base. Cada um desses canais é compartilhado novamente utilizando o TDMA, em oito espaços de tempo (*timeslots*). Cada terminal móvel que requisita uma chamada à ERB utiliza dois *timeslots* para a comunicação, sendo um para a comunicação no sentido móvel para ERB e outro para no sentido ERB para móvel.

1.2 MOTIVAÇÃO

Diante da necessidade inegável de dispositivos mais dinâmicos no tratamento de eventos relacionados à segurança de ambientes, propomos novas funcionalidades que vem de encontro aos atos criminosos e que possam ser adicionadas de forma relativamente fácil a um sistema de câmeras IP já instalado.

A abundância de disponibilidade de redes celulares e a popularidade de dispositivos móveis com cada vez mais capacidade de processamento permitem lançar mão dessa infraestrutura para integrar

sistemas de segurança, dando-lhes novas funcionalidades e aumentando as possibilidades de utilização de um sistema praticamente padronizado no mercado.

Diante do exposto, os tópicos abaixo caracterizam os sistemas atuais:

- Alto custo dos dispositivos;
- Produtos altamente padronizados, sem possibilidade de personalização.
- Difícil acesso a novos itens.

O esquema proposto vem contribuir de forma a deixar o sistema mais personalizável e com maior dinamicidade, onde os módulos e funcionalidades possam ser inseridos e removidos para atender necessidades diferentes, tendo como base as características abaixo:

- Redução do custo de produção;
- Utilização de softwares livres;
- Possibilidade de personalização;
- Possibilidade de inclusão de funcionalidades para atender diferentes cenários;
- Possibilidade de desenvolvimento colaborativo e multiplataforma;

1.3 MODELAGEM DO PRODUTO

O sistema proposto é composto por dois processadores, conforme Figura 2. O processador principal é responsável pelo controle das atividade que exigem maior processamento, que são as duas câmeras instaladas diretamente nele. Esse processador é também responsável pelo acesso à rede local e à Internet, provendo conectividade ao sistema.

O segundo processador é responsável por controlar o módulo GPRS/GSM, e executar as rotinas necessárias para que a conexão seja estabelecida e executar as modificações necessárias que o sistema exija.

Por fim, o bloco Detector DTMF é o dispositivo que interliga o processador principal ao secundário. O tom gerado em uma ligação recebida pelo módulo GPRS/GSM é disponibilizado ao Detector DTMF, que identifica qual é o tom e entrega o código correspondente ao processador principal, que identifica e executa a ação configurada.

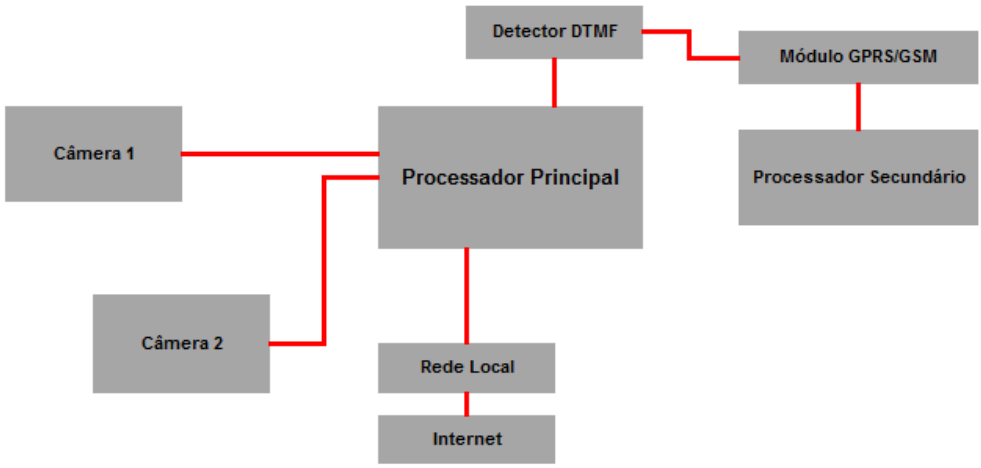

Figura 2 Diagrama de Blocos do Sistema

1.4 ESTRUTURA E COMPOSIÇÃO DO TRABALHO

O trabalho apresentado está dividido da seguinte forma:

Capítulo 1: Introdução

Apresenta o panorama da segurança no Brasil, uma breve descrição da proposta e motivação para desenvolvimento do produto, a estrutura do produto e a forma como o trabalho foi organizado.

Capítulo 2: Embasamento Teórico para o Presente Trabalho

Nesta seção, apresentaremos as tecnologias disponíveis e empregadas para o desenvolvimento do dispositivo final, mostrando a motivação para a sua escolha e descrevendo as funcionalidades aplicadas ao conjunto final. Apresentaremos também os dispositivos não empregados no projeto, seja por incompatibilidade ou por conveniência no desenvolvimento.

Seção 3: Hardware e Software

Nesta seção será apresentada a integração proposta na Figura 2 bem como os softwares desenvolvidos e as particularidades de cada dispositivo. Serão apresentados também resultados de alguns testes realizados, tanto de *hardware* quando de desempenho para demonstrar que o sistema é viável e que tem capacidade de executar as funções propostas no capítulo 4. Por fim, apresentaremos algumas rotinas específicas de cada *software*, tanto os desenvolvidos quanto os instalados, focando nos seus aspectos funcionais dentro do sistema e condicionado ao objetivo atingido com cada um.

Capítulo 4: Serviços oferecidos

Nesta seção serão apresentados os serviços oferecidos pelo sistema e suas opções de personalização. Nesse sentido, serão descritas as funcionalidades disponíveis e suas formas de utilização, desde o aspecto específico até as opções gerais disponíveis. Apresentaremos também algumas propostas de

serviços que ainda podem ser desenvolvidos diante do *hardware* proposto. Por fim, sugerimos também inclusão de novas funcionalidades que dependam do desenvolvimento de novos módulos.

Capítulo 5: Conclusões

Apresentamos nesta seção algumas ideias não implementadas que podem ser agregadas ao sistema, deixando-o cada vez mais completo de forma a atender diferentes necessidades. Apresentamos também as conclusões sobre o produto e o trabalho desenvolvido.

2 EMBASAMENTO TEÓRICO PARA O PRESENTE TRABALHO

Esta seção objetiva conceituar os dispositivos utilizados e mostrar o embasamento para a escolha de cada um deles.

Para o desenvolvimento de produtos de mercado, diversas tecnologias disponíveis têm se integrado. Nessa visão, integrar vários dispositivos para compor um produto no qual os requisitos levantados são atendidos é o objetivo principal da fase de projeto. Como a proposta é desenvolver um dispositivo modular e com alta capacidade de personalização, projetar módulos que atendam a determinadas funções permite que o sistema seja eficiente e ao mesmo tempo personalizável.

Nesta seção, apresentamos as tecnologias disponíveis e utilizadas para o desenvolvimento do dispositivo final, mostrando a motivação para a sua escolha e descrevendo as funcionalidades aplicadas ao conjunto final. Como exposto, foi projetado um sistema modular, que executa funções específicas mas que pode desempenhar mais de uma tarefa.

Apresentamos também os dispositivos não empregados no projeto, seja por incompatibilidade ou por conveniência no desenvolvimento.

2.1 RASPBERRY PI

O Rasberry Pi é um computador com processador ARM (*Advanced* RISC *Machine*) do tamanho de um cartão de crédito. Ele conta com um processador gráfico – GPU – e 512 MB de memória RAM interna.

O chip processador é baseado no Broadcom BCM2835, que inclui um processador ARM de 700MHz, uma GPU e 512MB de memória RAM. Atualmente, é comercializado em dois modelos: o modelo A e o Modelo B. O último possui controlador Ethernet e duas portas USB enquanto que o primeiro não possui controlador Ethernet e apenas uma porta USB. Um passo a passo para ligar o dispositivo é mostrado na Figura 3, onde podemos ver os principais recursos disponíveis.

O Raspberry Pi é compatível com sistemas operacionais Linux e diversas distribuições apresentam suporte. A mais comum é a baseada no Debian, denominada Raspbian, que é considerada a distribuição oficial deste dispositivo. A Tabela 3 mostra a lista de distribuições suportadas oficialmente, conforme site do fabricante [3].

Tabela 3 Distribuições Linux Disponíveis para Raspberry Pi

Distribuição	Descrição
Raspbian	Distribuição baseada no Debian otimizado para a utilização no Raspberry Pi.
Pidora	Distribuição baseada no Fedora Linux otimizado para utilização no Raspberry Pi.
RaspBMC	Distribuição baseada no Debian que inclui uma Central de mídia XBMC.
OpenELEC	Distribuição Linux com Central de mídia XBMC, rápida e com interface amigável.
RISC OS	É um sistema rápido e compacto.
Arch	Distribuição especifica para dispositivos com processador ARM.

Diversas linguagens são suportadas para programação, desde que possa ser compilada para a arquitetura ARM. A linguagem mais popular é o Python, mas pode ser utilizado com linguagens mais tradicionais como o C, C++ ou JAVA.

No desenvolvimento do projeto, optou-se por utilizar o C para desenvolver o programa principal que controla todas as ações tomadas pelo Raspberry Pi. A opção por esta linguagem foi motivada pela facilidade de utilização da biblioteca GPIO utilizada para controlar os pinos de IO e que são necessários algumas funções definidas nos outros capítulos.

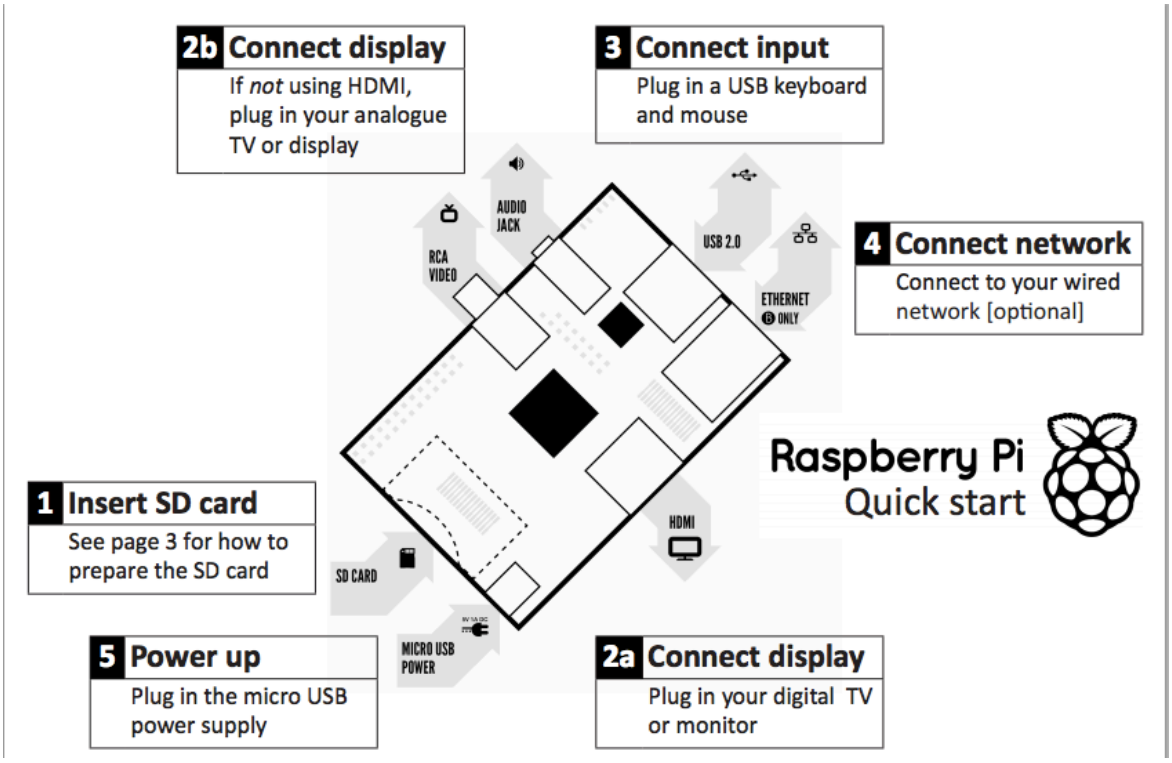

Figura 3 Apresenteção dos Principais Recursos Presentes no Raspberri Pi
Fonte: Site do fabricante [3]

Apesar de pequeno, algumas funcionalidades presentes permitem desenvolver diversas aplicações. A Figura 3, como já foi afirmado, mostra as principais itens inclusos na plataforma, sendo eles:

1. SD Card – Slot para cartão de memória do tipo SD. Essa memória é onde fica instalado o sistema operacional do dispositivo e de onde são lidos os arquivos de inicialização.
2. Saídas de vídeo – Conexões para TVs ou monitores analógicos e digitais. Temos conexão HDMI para dispositivos de vídeo digitais e conexão RCA para dispositivos de vídeo analógicos.
3. Entradas USB 2.0 – Conexões USB que podem ser usadas para uma infinidade de dispositivos que utilizam essa conexão e que estão presentes no mercado, como câmeras de vídeo, teclados, mouses e etc.
4. Conexão com a rede Local – Um conector RJ45 para conexão à rede local com fio. Isso possibilita conectar o Raspberry Pi à Internet.
5. Alimentação – O Raspberry Pi é alimentado através do conector micro USB com tensão de 5V.

Pinos de IO são importantes ferramentas para comunicação. O Raspberry Pi possui dois conjuntos de pinos, denominados P1 e P5 que podem ser utilizados como pinos de IO, portas seriais e outras funções. A Figura 4 e a Figura 5 mostram os dois conjuntos de pinos e a função de cada um dos pinos.

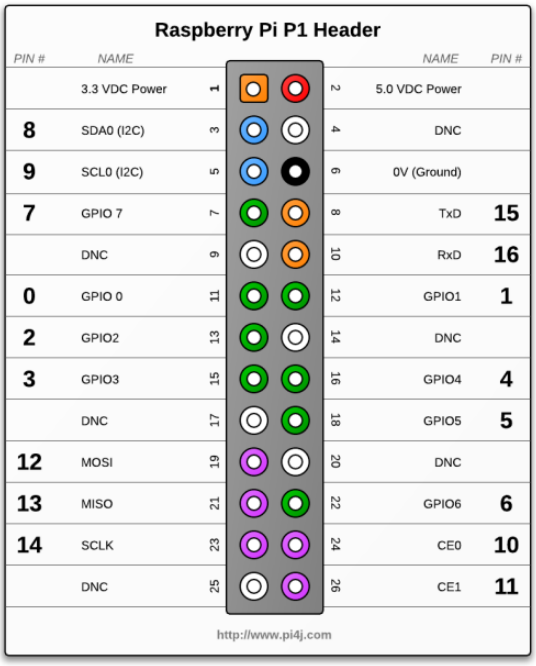

Figura 4 Descrição do Conjunto de Pinos P1
Fonte: Pi4j.com [4]

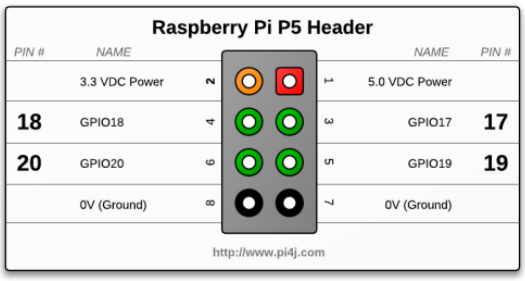

Figura 5 Descrição do Conjunto de Pinos P5
Fonte: Pi4j.com [4]

O Raspberry Pi é responsável pelas funções listadas abaixo:

- Processamento e armazenamento das imagens capturadas pelas câmeras instaladas;
- Conexão com a rede local e Internet através do dispositivo USB Wi-Fi.
- Processar imagens com objetivo de detecção de movimento.
- Identificar comandos recebidos através do detector DTMF.
- Alterar propriedades do sistema de acordo com o comando recebido através do detector DTMF.
- Fornecer ao usuário plataforma de personalização das funcionalidades disponíveis.
- Manter o serviço de visualização das câmeras ao vivo.

2.2 ARDUINO

Arduino é uma plataforma de código aberto para implementar sistemas eletrônicos de forma flexível, com hardware e software fáceis de se usar. Ele é utilizado por artistas, *designers*, entusiastas e qualquer interessado em criar objetos interativos ou ambientes.

O Arduino, mostrado na figura 6, é um microcontrolador baseado no Atmega328. Na versão UNO (utilizada no projeto) possui 15 pinos digitais que podem funcionar como entrada ou saída (6 deles podem ser usados como saídas PWM), 6 entradas analógicas, um cristal de 16MHz e uma conexão USB.

O Atmega328 possui 32 KB de memória, com 0,5 KB usados para o *bootloader*. Ele também possui 2 KB de SRAM e 1 KB de EEPROM, que pode ser utilizada com a biblioteca "*EEPROM library*".

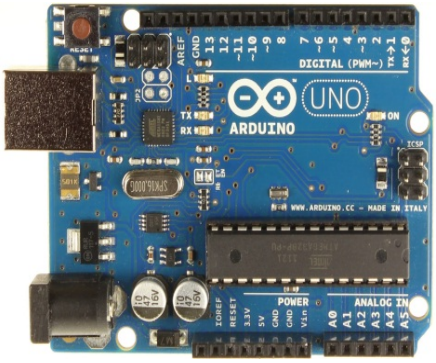

Figura 6 Logomarca da Comunidade Arduino
Fonte: arduino.cc

O Arduino é responsável pelas funções listadas abaixo:

- Controlar o módulo GPRS/GSM;
- Apresentar o estado da conexão com a rede GSM no display LCD

A escolha pelo Arduino para controlar o módulo GPRS/GSM é pelo fato de existir biblioteca que pode ser facilmente utilizada. Para o Rasberry essa utilização seria dificultada.

2.3 MÓDULO GPRS/GSM

O módulo GSM/GPRS é um dispositivo que funciona de forma semelhante a um telefone celular. Ele foi empregado para ao Arduino estabelecer conexão com a rede celular para transferência de voz (GSM) e dados (GPRS). O módulo empregado é fabricado com base no chip SIM900 da SIMCom, mostrado na figura 7. O SIM900 é um módulo wireless QUAD-BAND, que opera nas frequências de 850 MHz, 900 MHz, 1800 MHz e 1900 MHz.

Figura 7 Chip SIM900 - Módulo GSM/GPRS
Fonte: Datasheet do fabricante [5]

A comunicação do módulo GSM/GPRS com o Arduino é feita via interface Serial. Por esta interface são enviados os comandos AT, através dos quais podemos configurar o módulo para realizar e encerrar chamadas, enviar mensagens de texto além de outras funções exemplificadas na Tabela 4.

Tabela 4 Lista de Comandos AT
Fonte: itu.int [6]

Comando	Descrição
ATA	Atende uma chamada que esteja em andamento.
ATD><N>	Inicia chamada para o número indicado em <N>
ATDL	Inicia chamada para o último número de telefone usado.
ATS7=<n>	Determina o tempo máximo (em segundos) de aguardo para que seja completada uma chamada
AT+CMGF	Determina o formato de SMS
ATD<n>	Realiza uma chamada de voz para o número <n>

Os comandos AT foram criados com base na norma da ITU-T (*International Telecommunication Union, Telecommunication sector*) V.25ter. Esses comandos podem ser divididos em quatro tipos, e para cada um deles é esperado um tipo de resposta. A Tabela 5 apresenta os tipos de comandos AT.

Tabela 5 Tipos de Comando AT
Fonte: Datasheet do Fabricante [5]

Tipo de comando	Sintaxe	Descrição da resposta
Comando de teste	AT+<x>=?	O equipamento móvel retorna a lista de parâmetros e os intervalos de valores configuráveis do comando de escrita correspondente
Comando de Leitura	AT+<x>?	Este comando retorna o valor corrente do parâmetro indicado.
Comando de Escrita	AT+<x>=<...>	Este comando configura o parâmetro passado pelo usuário.
Comando de Execução	AT+<x>	O comando de execução lê parâmetros não variáveis afetados pelo processo interno no mecanismo GSM.

Para todos os casos, <x> representa o comando a ser executado e <...> os argumentos para o comando. Por exemplo, o comando para atender uma ligação é "ATA" e é um Comando de Execução, pois o comando executado é indicado pelo último "A" (executa o comando Atender Ligação em Curso – *Answer an Incoming Call*).

O chip SIM900 possui duas portas UART integradas, uma é chamada de *Serial Port* e a outra de *Debug Port*. A primeira é usada para receber os comandos AT do processador principal (no caso, o Arduino), enquanto a segunda é utilizada para atualização de Firmware e buscas de erros. O esquema de ligação é mostrado na Figura 8.

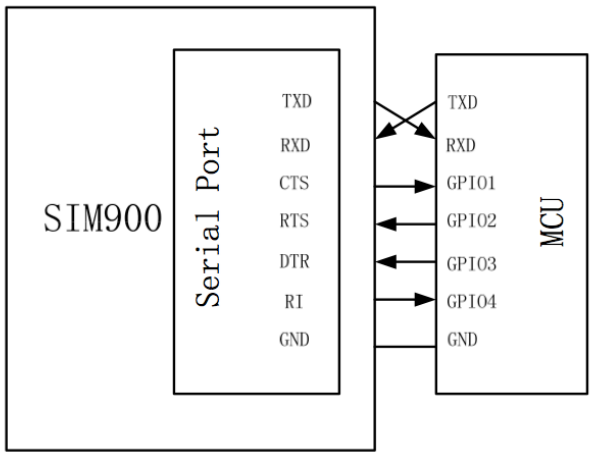

Figura 8 Comunicação Serial do Módulo GSM/GPRS
Fonte: Datasheet do fabricante [5]

Uma particularidade do módulo GSM/GPRS percebida durante a realização de testes é que é necessário uma alimentação externa de 5V para que o módulo funcione como o esperado. Nos testes realizados somente com a alimentação USB do Arduino o dispositivo ao receber o comando para ligar o SIM900 (que é o chip responsável pela conexão à rede celular) não consegue estabelecer a conexão, desligando o *chip* ao realizar o procedimento.

Este módulo GPRS/GSM possui um LED indicador do estado da conexão com a rede celular, baseado no esquemático mostrado na Figura 9. A Tabela 6 mostra os estados e seus respectivos indicativos neste LED.

Tabela 6 Descritivo do LED indicador de Rede
Fonte: Datasheet do fabricante [5]

Estado do LED	Função do SIM900	Designação do Estado
Desligado	SIM900 está desligado	1
64ms ligado e 800ms desligado	SIM900 não encontrou uma rede	2
64ms ligado e 3000ms desligado	SIM900 encontrou rede	3
64ms ligado e 300ms desligado	Comunicação GPRS ativa	4

Ao ser enegizado o módulo inicia no estado 1, com o SIM900 desligado. Ao receber o comando via porta serial ou acionando manualmente o botão no módulo o SIM900 vai para o estado 2, onde inicia a procura por redes para se conectar. O módulo permanecerá nesse estado caso não encontre uma rede para conexão ou não esteja com um SIMCARD instalado. Ao encontrar uma rede disponível, o SIM900 vai então para o estado 3 e fica esperando comandos para serem executados.

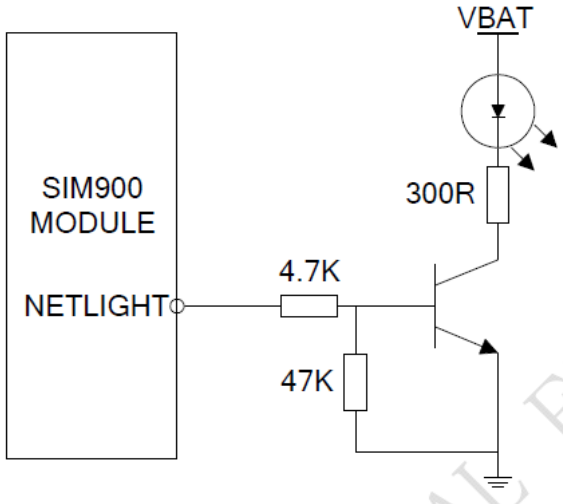

Figura 9 Esquemático do LED indicador de Rede
Fonte: Datasheet do fabricante [5]

2.4 DETECTOR DUAL TONE MUITI FREQUENCY (DTMF)

O HT9170B é um receptor DTMF constituído por um decodificador digital e filtro divisor de frequências. Esse dispositivo é capaz de detectar e decodificar os 16 pares de tons DTMF em um código digital de 4 bits [7]. A Figura 10 mostra a matriz de caracteres que podem ser identificados.

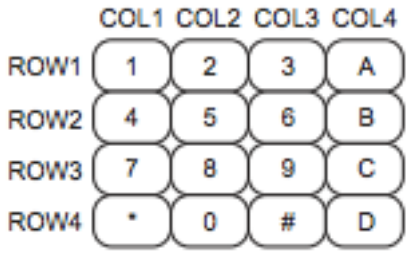

Figura 10 Relação de Teclas Disponíveis
Fonte: Datasheet do fabricante [7]

Cada tecla é sinalizada por duas frequências distintas. Estas frequências são divididas em dois grupos: o grupo de frequências baixas e o de frequências altas. No primeiro grupo temos frequências entre 697 Hz e 941 Hz e no segundo entre 1209 Hz e 1633 Hz. A divisão e combinação das frequências para formação dos dígitos está mostrada na Figura 11.

DTMF data output table

Low Group (Hz)	High Group (Hz)	Digit	OE	D3	D2	D1	D0
697	1209	1	H	L	L	L	H
697	1336	2	H	L	L	H	L
697	1477	3	H	L	L	H	H
770	1209	4	H	L	H	L	L
770	1336	5	H	L	H	L	H
770	1477	6	H	L	H	H	L
852	1209	7	H	L	H	H	H
852	1336	8	H	H	L	L	L
852	1477	9	H	H	L	L	H
941	1336	0	H	H	L	H	L
941	1209	*	H	H	L	H	H
941	1477	#	H	H	H	L	L
697	1633	A	H	H	H	L	H
770	1633	B	H	H	H	H	L
852	1633	C	H	H	H	H	H
941	1633	D	H	L	L	L	L
—	—	ANY	L	Z	Z	Z	Z

Note: "Z" High impedance; "ANY" Any digit

Figura 11 Relação das Teclas e as Frequências Correspondentes
Fonte: Datasheet do fabricante [7]

A montagem sugerida pelo fabricando no manual do produto está mostrada na Figura 12. Dois problemas surgiram quando realizamos testes com essa montagem sugerida. O primeiro deles foi a não detecção dos tons emitidos pelo módulo GSM/GPRS.

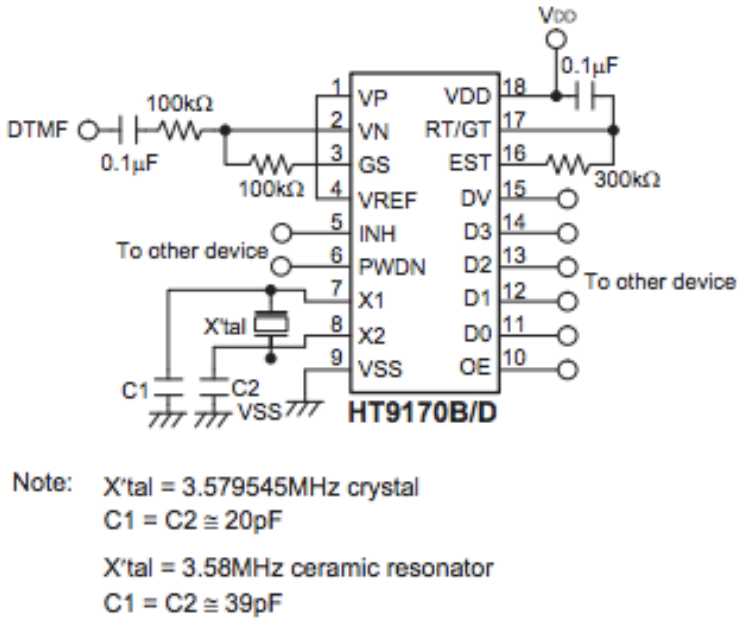

Figura 12 Montagem sugerida para o Detector DTMF
Fonte: Datasheet do fabricante [7]

O problema da não detecção dos tons foi resolvido removendo o capacitor de 0.1uF e o resistor de 100K inseridos em série na entrada de áudio do detector DTMF. A suspeita inicial (e que foi confirmado) era de que o nível de sinal de áudio na saída do módulo GPRS estava muito baixo, sendo atenuado ainda mais pelo conjunto de resistores das entradas VN e GS. Constatou-se que a saída do módulo GPRS não tinha nível DC, o que já foi motivo para eliminar o capacitor de 0,1uF. Assim, removendo o resistor e o capacitor o sinal que antes era atenuado por estes componentes foi entregue com maior nível, sendo possível a detecção dos dois tons. Após os procedimentos citados, detectou-se o segundo problema, pois o detector DTMF passou a reconhecer apenas algumas teclas (antes nenhuma tecla era reconhecida).

A partir de então, tínhamos algumas teclas funcionando bem e outras não. Num estudo dos pinos RT/GT e EST do HT9170B, percebemos que são pinos para medir a duração efetiva do sinal na entrada do detector e para proteção contra perda de sinal válido. Esse circuito RC é que indica o limiar de detecção do circuito. O pino EST normalmente está em nível baixo e mantém o pino RT/GT também em nível baixo através do descarregamento do circuito RC. Quando um tom válido é detectado na entrada, EST vai para nível alto para carregar RT/GT através do circuito RC. Quando a tensão de RT/GT passa de 0 para Vtrt (tensão limiar de detecção), o sinal de entrada é detectado e o código correto é criado pelo detector nos pinos de saída.

O que estava ocorrendo é que o nível de tensão da entrada nem sempre era suficiente para alcançar a tensão Vtrt no pino RT/GT, que resultava na não detecção do tom. A solução então foi diminuir o resistor, aumentando a sensibilidade de detecção. Na montagem proposta, foi substituído o resistor de 300KΩ por um de 33KΩ.

2.5 O DISPLAY LCD

O *display* LCD utilizado é o mostrado na Figura 13. Possui duas linhas com 16 caracteres cada linha. O funcionamento desse dispositivo é bastante simples. Cada posição do *display* tem uma espaço correspondente na memória dos dispositivo. Quando precisamos armazenar dados em uma determinada posição do *display*, selecionamos a posição de memória correspondente, disponibilizamos os dados nos pinos DB0~DB7 e habilitamos o pino *Enable* colocando-o em nível baixo. Nesse instante é gravado na memória os dados disponibilizados nos pinos DB0~DB7. A cada caractere escrito, um ponteiro é incrementado para garantir que sejam armazenados no display em sequência.

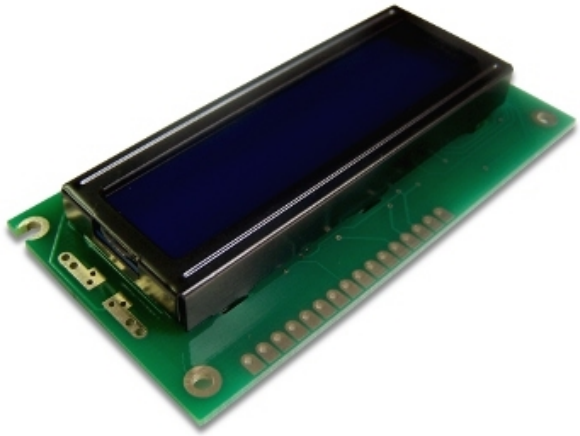

Figura 13 Display LCD Utilizado

2.6 CÂMERA USB

A câmera USB selecionada foi a Logitech HD Webcam C270, apresentada na Figura 14, que tem capacidade de filmar em HD com qualidade de 720p e capturar fotos com 3 Megapixels. A escolha por esta câmera foi devido à já conhecida qualidade e compatibilidade com o Raspabian (Sistema Linux baseado no Debian instalado no Raspberry Pi).

Figura 14 Câmera Logitech C270 Utilizada
Fonte: logitech.com [8]

2.7 RASPCAM

Outra opção de câmera utilizada no desenvolvimento do projeto foi o módulo de Câmera para Raspberry Pi ou RaspCam, apresentada na Figura 15. Este módulo é capaz de capturar imagens com 5 megapixels, em 720p em 60 quadros por segundo. Uma vantagem da utilização desta câmera é que a captura e o processamento para armazenamento é feito pela GPU dedicada presente no Raspberry Pi, deixando livre o CPU para outras aplicações.

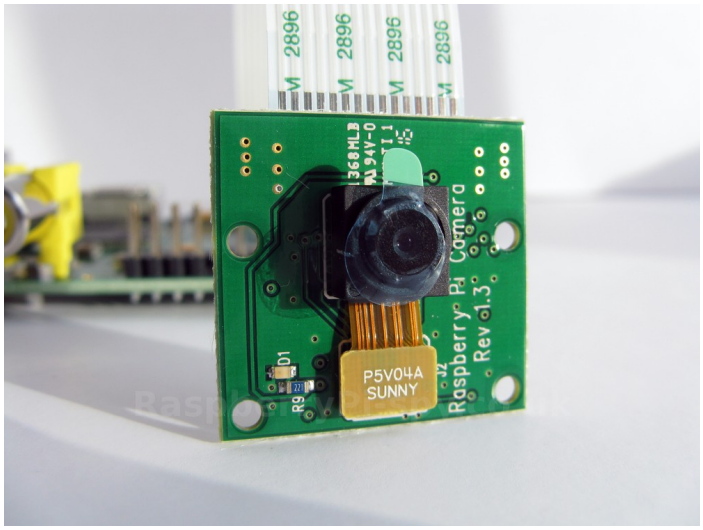

Figura 15 RaspCAM
Fonte: raspberrypi-spy.co.uk [9]

2.8 CAMERA SERIAL C328

Em testes preliminares, pretendia-se utilizar a câmera serial C328, que apresentou problemas e foi substituída pelas câmeras apresentadas anteriormente (Câmera USB e RASPCAM). As duas unidades que dispúnhamos vindas de outro projeto não respondiam pela porta serial, o que leva a concluir que estavam, ambas, defeituosas. A Figura 16 mostra o modelo da câmera.

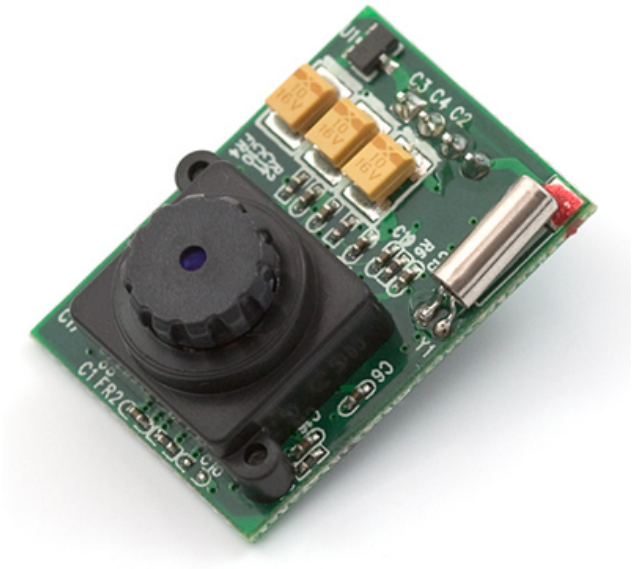

Figura 16 Câmera C328
Fonte: mbed.org [10]

2.9 WIPI – MÓDULO WIFI

Utilizamos um módulo Wifi conectado à porta USB do Raspberry Pi para dar maior flexibilidade na instalação do dispositivo, dispensando assim o uso de cabos. O módulo utilizado o WiPi, que é um dispositivo USB desenvolvido para Raspberry Pi.

O WiPi (Figura 17) é um dispositivo de alta performance, que suporta o padrão 802.11n e pode transmitir até 150Mbps. Além disso, tem suporte a conexão nos padrões 802.11b e 802.11g.

Figura 17 Módulo Wifi Empregado
Fonte: Detroit-electronics.com [11]

Dispositivos Wifi USB são comuns no mercado e diversas marcas estão disponíveis. A razão da opção pelo WiPi foi a certeza da compatibilidade, uma vez que é desenvolvido e testado para funcionar com o Raspberry Pi.

3 HARDWARE E SOFTWARE

Serão apresentadas o Hardware proposto, o Software desenvolvido e os testes realizados para verificação do correto funcionamento do sistema.

Nesta sessão serão apresentados a integração proposta no capítulo 1, bem como os *softwares* desenvolvidos e as particularidades de cada dispositivo. Serão apresentados também resultados de alguns testes realizados, tanto de *hardware* quando de desempenho para demonstrar que o sistema é funcional e que tem capacidade de executar as seguintes funções:

- Realizar captura e armazenamento de imagens e vídeos;
- Detectar movimento para início e fim do armazenamento das imagens, economizando assim espaço de armazenamento interno.
- Permitir configuração do modo de armazenamento de vídeo e envio de comandos via celular.
- Avisar, via rede GPRS/GSM, de movimentos detectados nas câmeras.
- Ativar e desativar o sistema de aviso de movimentos pelo celular.

Essas funções estão detalhadas no capítulo 4. Por fim, apresentaremos algumas rotinas específicas de cada *software*, tanto as desenvolvidas quanto as instalados, focando nos seus aspectos funcionais dentro do sistema e objetivo atingido com cada uma.

3.1 O MOTION

Motion é um programa que monitora o sinal de vídeo das câmeras e é capaz de identificar se uma parte significativa da imagem foi alterada, detectando assim se houve movimento. Foi desenvolvido em C e feito para sistema operacional Linux.

Quando falamos de detecção de movimento, temos duas técnicas para utilizar: detecção comparando com um quadro fixo ou então comparando com o último quadro. A primeira técnica consiste em definir um fundo fixo e comparar cada quadro recebido com primeiro quadro armazenado para decidir se há movimento ou não. A segunda técnica atua de forma diferente, comparando sempre o quadro atual com o quadro anterior, armazenado para comparação. Após a fase de comparação e decisão o quadro anterior é descartado e o atual é armazenado temporariamente para a próxima comparação.

Em determinadas aplicações é possível que o ambiente inicial mude de forma definitiva, fato que levaria sistemas que utilizem a segunda técnica a armazenar imagens da nova forma do ambiente em todos os quadros após a mudança. Outras aplicações necessitam dessa característica para monitorar quando o ambiente monitorado volta ao estado inical.

A segunda técnica é, à primeira vista, mais adequada para a maior parte das aplicações. Apesar disso, é necessário considerar que ela utiliza mais CPU que a primeira devido ao processo razoavelmente mais complicado.

Para verificar se houve ou não movimento ambas as técnicas procedem da mesma forma na comparação das imagens: é feita uma subtração pixel a pixel das duas imagens. Se a modificação foi considerável (acima de um limiar determinado), é decidido que houve movimento e a imagem é armazenada.

A especificação desse limiar é difícil. A escolha de um limiar muito baixo pode ocasionar no armazenamento de imagens que sofreram pouca variação, por exemplo uma variação de luminosidade do ambiente (problema tratado pelo parâmetro *lightswitch* presente no arquivo de configuração do Motion). Por outro lado, a escolha de um limiar muito alto pode incorrer na não detecção de um movimento real ocorrido. O *lightswitch* é um parâmetro de limiar de detecção de movimento para mudanças na intensidade de luminosidade da imagem. O programa ignora as mudanças de luminosidade de acordo com a porcentagem da área da imagem que sofreu alteração. O parâmetro pode receber um número de 0 a 100, onde 0 (padrão do software) indica que essa função está desabilitada. Quando uma mudança na luminosidade é detectada, a detecção de movimento é desabilitada por 5 quadros de imagens. Para o sistema desenvolvido, os testes realizados não apresentaram mudanças de luminosidade significativa para justificar a alteração do parâmetro *ligtswitch*.

Outro ponto que pode gerar problemas na detecção é a taxa com que os quadros são capturados pela câmera. Se adquirirmos quadros em uma taxa muito alta, corremos o risco elevar muito o processamento do sistema e, em alguns casos torná-lo indisponível. Por outro lado, uma taxa muito baixa pode afetar na detecção de movimentos rápidos .Temos, novamente, que encontrar um ponto de equilíbrio para o tipo de sistema que estamos desenvolvendo. Caso o sistema tenha que detectar ações muito rápidas, é necessário uma taxa de frames maior.

A RaspCAM (descrita no tópico 2.7 do Capítulo anterior) não funciona com o Software Motion citado. Para isso, membros do fórum do site RasberryPi.org escreveram um módulo para o Motion que lê frames diretamente da RaspCAM através da API MMAL (*Multi-Media Abstraction Layer*). A API MMAL é um *framework* que é usado para prover uma interface simples de relativo baixo nível aos componentes multimídia que rodam no *VideoCore* . Ela também oferece uma interface de componentes, para que novos dispositivos possam ser facilmente criados e integrados no *framework*.

VideoCore é uma Arquitetura de processador de baixa consumo de energia para processamento de imagens e vídeos. É uma arquitetura de DSP (*Digital Signal Processing*) que torna flexível e eficiente decodificar e codificar codecs multimídia via *software*, mantendo baixo consumo de energia. Essa arquitetura foi usada em diversos aparelhos dos principais fabricantes de celulares do mercado mundial. Na Figura 18 apresentamos como esta arquitetura está presente no Raspberry Pi [12].

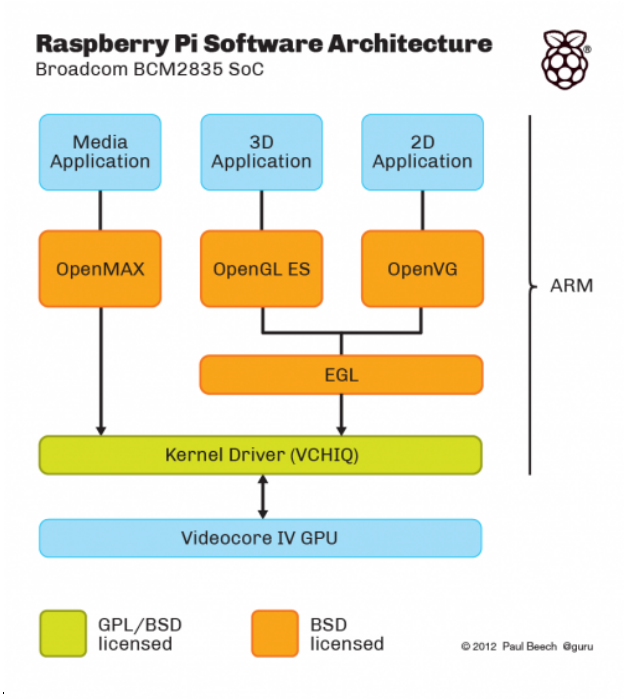

Figura 18 Estrutura da Arquitetura de Software do Raspberry Pi
Fonte: elinux.org [13]

A API MMAL utilizada o VideoCore presente no Raspberry Pi para capturar os quadros recebidos da câmera. Controlado pelo processador ARM o software Motion MMAL faz a interface com o VideoCore IV GPU.

3.2 PROGRAMA DESENVOLVIDO EM C PARA O RASPBERRY PI

Foi desenvolvido um programa em linguagem C, mostrado no Apêndice 1, que atende à interrupção a cada transição do nível baixo para o nível alto do pino DV do detector DTMF. O pino DV em nível alto indica que o detector recebeu um tom válido que foi decodificado e disponibilizado na saída. Cada interrupção lê o código referente ao tom recebido disponibilizado nos pinos D0, D1, D2 e D3. A Tabela 7 mostra os dígitos correspondentes à tecla digitada.

Tabela 7 Correspondência Entre a Tecla Digitada e a Saída do Detector DTMF

Código D0 D1 D2 D3	Tecla digitada
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	0
1011	*
1100	#

A Tabela 8 mostra a correspondência dos pinos do detector DTMF com os pinos do Raspberry PI.

Tabela 8 Correspondência Entre os Pinos do DTMF e os pinos do Raspberry Pi

Pinos DTMF	Pinos Raspberry Pi	Nº do Pino no P1
D0	GPIO0	Pino 11
D1	GPIO1	Pino 12
D2	GPIO2	Pino 13
D3	GPIO3	Pino 15
DV	GPIO5	Pino 16

De forma resumida o programa está apresentado na Figura 19 e na Figura 20: Programa Principal e Processo Interrupção. Trata-se de dois processos distintos porém interligados e dependentes. O primeiro diagrama mostra que o programa inicia definindo constantes e variáveis que serão utilizadas durante a execução. Imediatamente após, as variáveis são declaradas como globais pois podem ser

alteradas tanto na rotina principal quanto na rotina da interrupção. Nesse ponto são definidas os valores de cada um dos pinos que serão lidos no detector DTMF, conforme mostrado na Tabela 8. As variáveis definidas nesse ponto são as de uso global, ou seja, que serão utilizadas em qualquer ponto do programa, como por exemplo o contador geral de dígitos recebidos (variável *digito recebido* do programa **gpioread.c**).

Logo após é declarada a rotina da interrupção com as instruções definidas no diagrama Processo Interrupção, são iniciadas as configurações da biblioteca wiringPi e do modo de cada um dos Pinos de IO. Então o programa configura a interrupção para ser ativada pelo pino DV e entra em um laço para esperar que interrupções ocorram.

Sempre que o pino DV for para nível alto significa que um digito válido foi recebido e que se deve iniciar uma interrupção para executar o comando recebido. Assim, o diagrama Processo Interrupção inicia armazenando os dígitos recebidos através dos pinos de IO do Raspberry Pi conectados ao detector DTMF. A seguir o programa compara os quatro dígitos recebidos com a tecla correspondente digitada, conforme Tabela 7, e executa a função determinada para cada uma delas. Caso detecte uma tecla, o programa executa a função determinada e retorna da interrupção. Caso não encontre nenhuma ação a ser executada a interrupção também é finalizada.

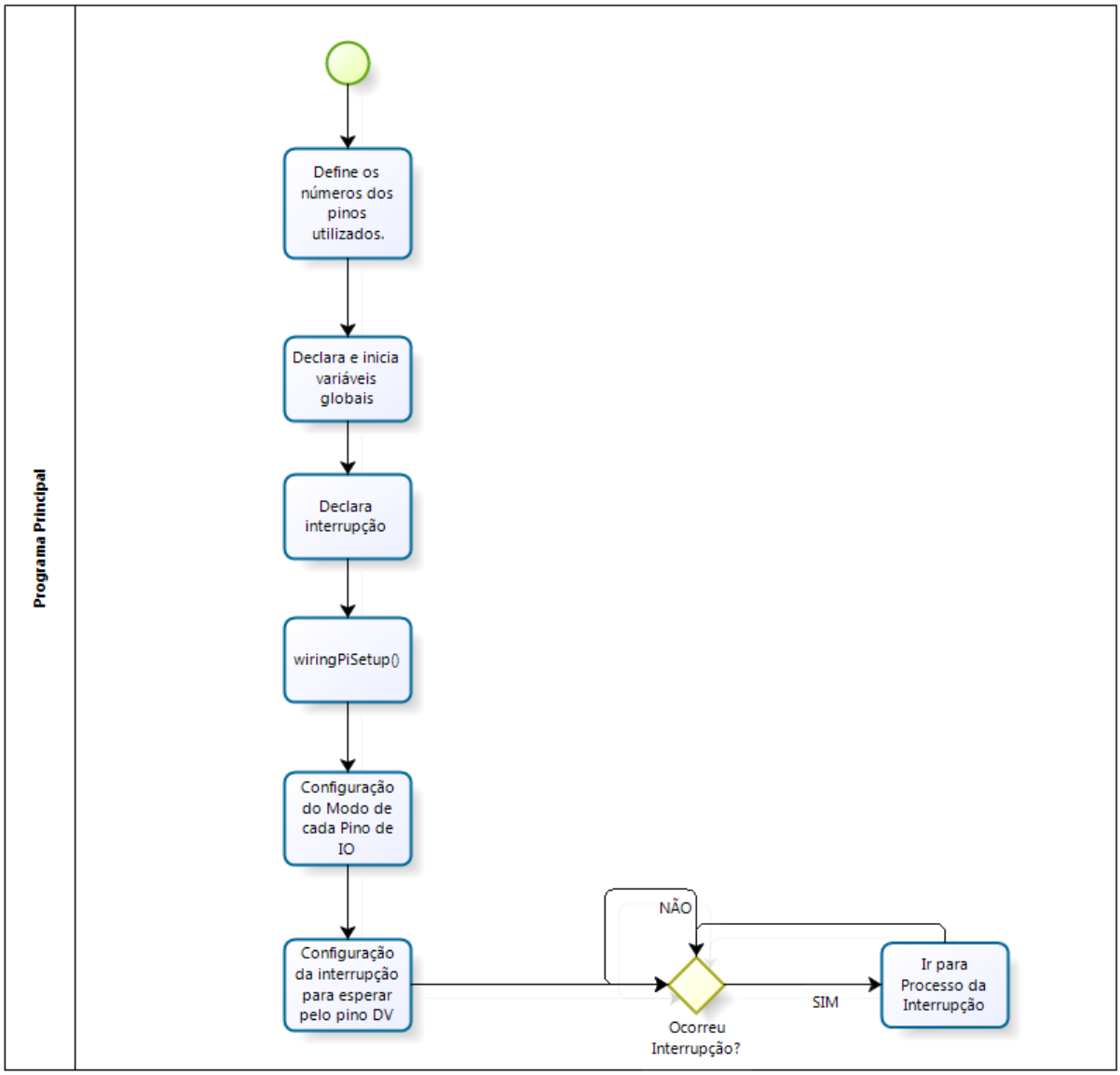

Figura 19 Programa Principal

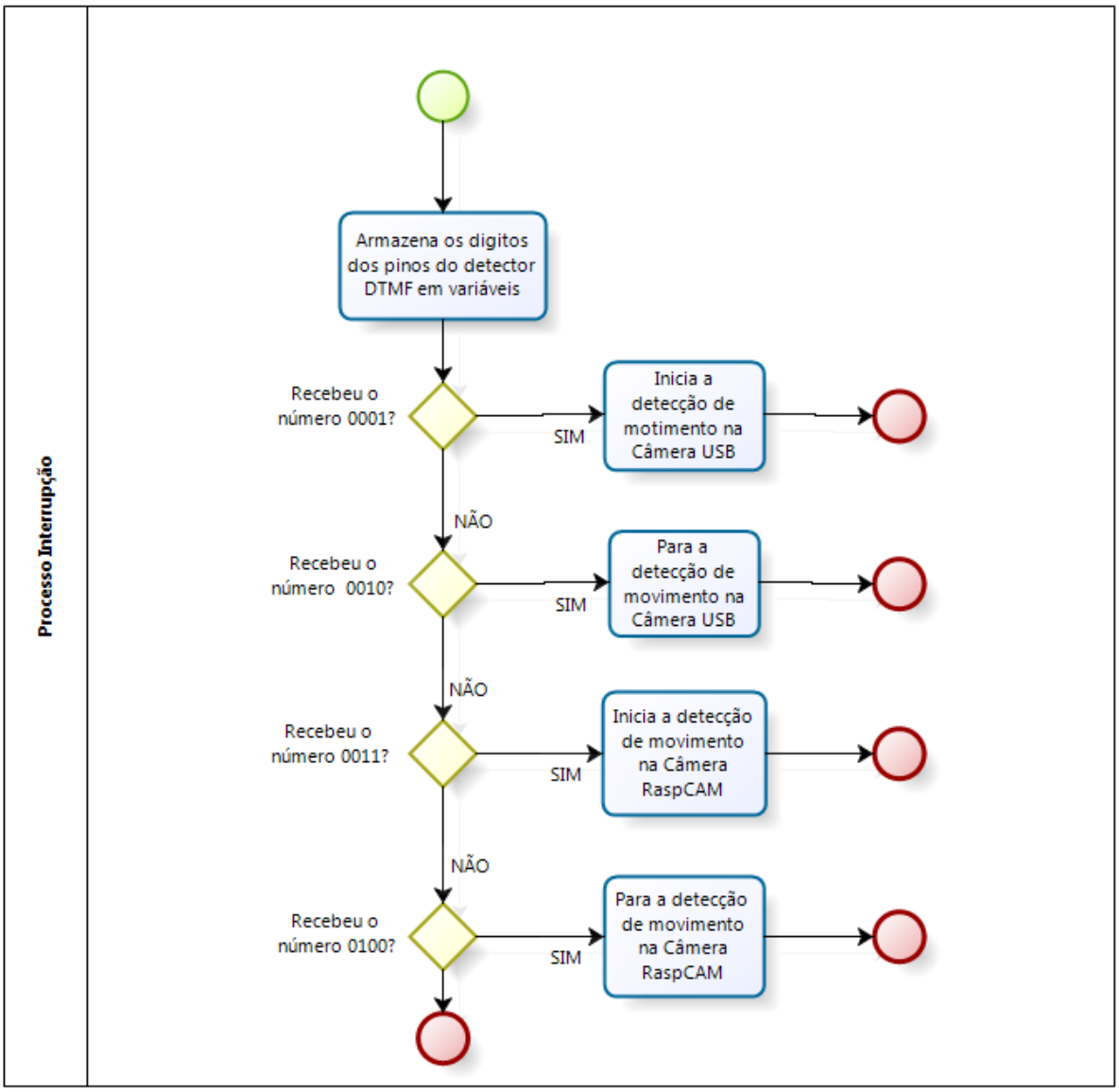

Figura 20 Interrupção do Programa Princial

3.3 SERVIÇO CRON – LINUX

O CRON é um serviço do Linux que é carregado durante o processo de *boot* do sistema e permite programar a execução de comandos e processos de forma agendada. Podem ser agendados comandos executados em *shell script*, tarefas ou serviços do sistema.

O sistema precisa de estar funcional (com todos os serviços ativos) seja na primeira inicialização ou após um possível desligamento. Para tornar isso possível foi agendado a execução do programa **gpioread.c** durante o processo de boot do sistema, para que a detecção esteja disponível no momento em que o sistema é ligado.

3.4 BIBLIOTECA WIRINGPI EM C

WiringPi é uma biblioteca de acesso aos pinos de IO do BMC2835 (circuito integrado que que integra o Raspberry Pi) escrita em C. O Raspberri Pi tem 26 pinos de que podem ser utilizados com entrada e saída de dados (GPIO – General Purpose Input/Output).

Essa biblioteca inclui um utilitário de linha de comando que pode ser usado monitorar, para programar e configurar pinos GPIO. A biblioteca também permite escrever *scritps shell* ou linguagem C para executar as mesmas funções. Devido a praticidade, os programas foram desenvolvidos em C para controle dos pinos GPIO do Raspberry Pi, onde também são tratados as informações recebidas nesses pinos.

A seguir, relacionamos algumas informações acerca das principais funções presentes nesta biblioteca e as principais características:

- pinMode : Esta função permite configurar o modo de um determinado pino, ou seja, se o ele vai trabalhar como pino de entrada, saída, saída de PWM (Pulse-Width Modulation) ou um relógio. Essa função recebe dois argumentos: o número do pino e o modo de operação.
- digitalWrite: Esta função escreve os valores Alto ou Baixo (1 ou 0) para o pino indicado. Recebe também dois argumentos: o número do pino e o valor a ser escrito.
- digitalRead: Esta função retorna o lavor lido no pino indicado.

3.5 BIBLIOTECA LIQUIDCRYSTAL PARA ARDUINO

Esta biblioteca permite um Arduino controlar um *display* de cristal líquido (LCD) baseado no *chipset* Hitach HD44780, que é o mais comum nos mostradores LCD alfanuméricos. A biblioteca pode trabalhar com barramentos de 4 ou 8 bits, auxiliados pelas linhas de controle (RS, *enable* e RW). O funcionamento do *display* LCD está mostrado o item 2.5 deste documento.

O display LCD nesse projeto foi destinado a exibição de mensagens de informação a respeito do estado do módulo GPRS. A informação “PRONTO” na primeira linha do *display* indica que o Arduino foi iniciado corretamente e que o programa para ativação da rede celular está em andamento. Na segunda linha aparece “Atend. Auto” indicando que o atendimento automático foi ativado com sucesso.

O display LCD utilizado possui 16 pinos, conforme Tabela 9. Para que a biblioteca pudesse ser utilizada, o display LCD foi montado conforme mostrado na Figura 21.

]

Tabela 9 Descrição dos Pinos do Display LCD
Fonte: labdegaragem.com [14]

Pino	Símbolo	Função
1	VSS	GND – Terra
2	VDD	5V
3	V0	Ajuste de contraste
4	RS	Habilita/Desabilita o seletor de registrador
5	R/W	Leitura/Escrita
6	E	Habilita escrita no LCD
7	D0	Dado
8	D1	Dado
9	D2	Dado
10	D3	Dado
11	D4	Dado
12	D5	Dado
13	D6	Dado
14	D7	Dado
15	A	5V da iluminação
16	K	GND da iluminação

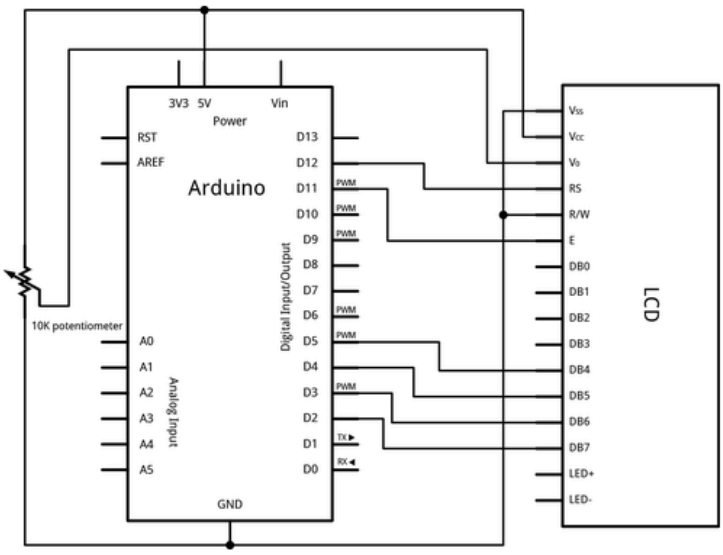

Figura 21 Esquemático de Montagem do Display LCD
Fonte: labdegaragem.com [14]

A biblioteca LiquidCrystal disponível disponibiliza três importantes funções que foram empregadas no projeto:

- begin();
- print();
- setCursor();

A função begin() inicializada a interface com o *display* LCD e especifica as dimensões (tamanho e largura) do modelo utilizado. Essa função precisa ser executada antes de qualquer outra função da biblioteca LCD. No nosso caso, o *display* é de 16 colunas e 2 linhas, configurado pelo comando da linha 118 no Anexo B.

A função print() é a função executada quando se precisa enviar dados ao display. Podemos passar como argumento variáveis do tipo *char*, *byte*, *int*, *long* ou *string*.

A função setCursor() posiciona o cursor, isto é, configura a localização a partir da qual o próximo texto escrito será mostrado. Essa função recebe dois argumentos: coluna e linha, respectivamente.

3.6 BIBLIOTECA SERIAL PARA ARDUINO

A comunicação entre o Arduino e o módulo GPRS é feita através da porta serial e para isso usamos a biblioteca Serial disponível. A versão UNO do Arduino tem disponível apenas uma porta serial, que pode ser utilizada para comunicação com um computador, via interface USB, ou com módulos instalados.

As funções da biblioteca Serial utilizadas no projeto foram as seguintes:

- begin();
- print();

A função begin() permite configurar a taxa na qual os dados serão transmitidos pela interface. Essa taxa tem que ser a mesma configurada no dispositivo instalado na outra extremidade, para que não haja erros na comunicação. As taxas suportadas para comunicação com um computador são: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 ou 115200. O módulo GPRS empregado no projeto não funciona bem com taxas acima de 9600, por esse motivo a taxa escolhida foi exatamente essa.

A função print() permite enviar dados pela porta serial como um texto ASCII. Pode receber argumentos do tipo *int, float, char*, ou *string*. A função println() é uma variação da função print(), adicionando sempre uma quebra de linha no fim de cada argumento recebido.

3.7 RECEBENDO CHAMADA COM ATENDIMENTO AUTOMÁTICO

Com o módulo GPRS/GSM instalado no Arduino e com a comunicação serial funcional, passé possível iniciar a configuração da funcionalidade de atendimento automático que permite que ligações recebidas sejam atendidas sem a necessidade de alguma intervenção humana no sistema.

Para configurar, enviamos o comando "ATS0=1" via interface serial através da função println(). De acordo com o protocolo AT, o atendimento automático pode ativado (ATS0=1) ou desativado (ATS0=0).

3.8 HARDWARE FINAL DO DETECTOR DTMF

Como citado no Capítulo 2, a montagem do detector DTMF sugerida pelo fabricante não apresentou bom funcionamento junto com o resto do hardware do sistema, mas foi contornado como citado no item 2.4.

A Figura 22 mostra conector do cabo utilizado para conexão entre o detector DTMF e o módulo GPRS/GSM (Figura 23), onde foi utilizado somente o contato mais externo (*left*).

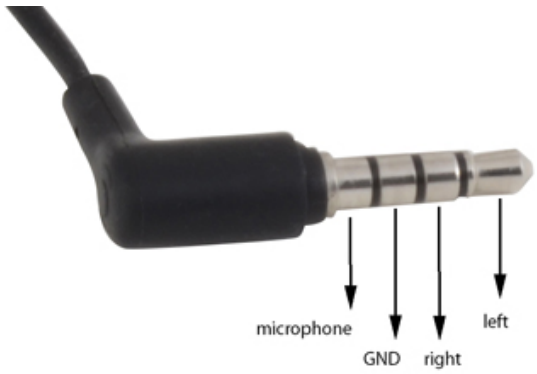

Figura 22 Conetor de Audio e Seus Respectivos Pinos
Fonte: cooking-hacks.com [15]

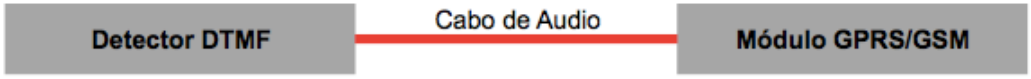

Figura 23 Utilização do Cabo de Áudio

3.9 VERIFICAÇÃO DE FUNCIONAMENTO DO SISTEMA

Foram identificados alguns pontos que poderiam gerar falhas no sistema e para validar seu bom funcionamento, realizamos alguns testes. Os possíveis pontos de falha são:

- Devido à CPU limitada, pode haver alto processamento quando as duas câmeras estiverem configuradas para capturar imagens.
- Baixo nível de sinal de celular pode causar indisponibilidade dos comandos remotos de forma permanente dependendo de como o módulo GPRS/GSM reagir a essa situação.
- Pouca capacidade de armazenamento pode tornar todo o sistema indisponível.

Para verificar o primeiro possível ponto de falha, ativamos a gravação mediante movimento para as duas câmeras e monitoramos o uso da CPU através do comando “top” no Linux. Analisando os processos do Motion e do Motion MMAL, percebemos que o gasto de CPU para a execução dessas funções é relativamente pequeno quando não há movimento detectado (2,9% e 21,8% respectivamente, conforme Figura 24).

Pode-se atribuir o maior uso do processador para o Motion MMAL pela maior qualidade de imagem da câmera que ele controla, a RaspCAM, que tem 5 megapixels contra 3 megapixels da câmera USB. Outra fato que pode contribuir é que o Motion MMAL é uma versão adaptada do software Motion para utilização junto com a RaspCAM, e que ainda está em desenvolvimento.

A diferença aumenta mais ainda quando comparamos o uso da CPU quando ambas as câmeras estão detectando movimento e têm que armazenar os quadros recebidos. O processamento para o Motion MMAL atinge o patamar de 60,8% enquanto o Motion fica em 18,8%, conforme Figura 25. Somente as câmeras estão gastando 79,6% do processamento. Apesar do alto consumo, nos testes realizados o sistema não ficou indisponível em momento algum.

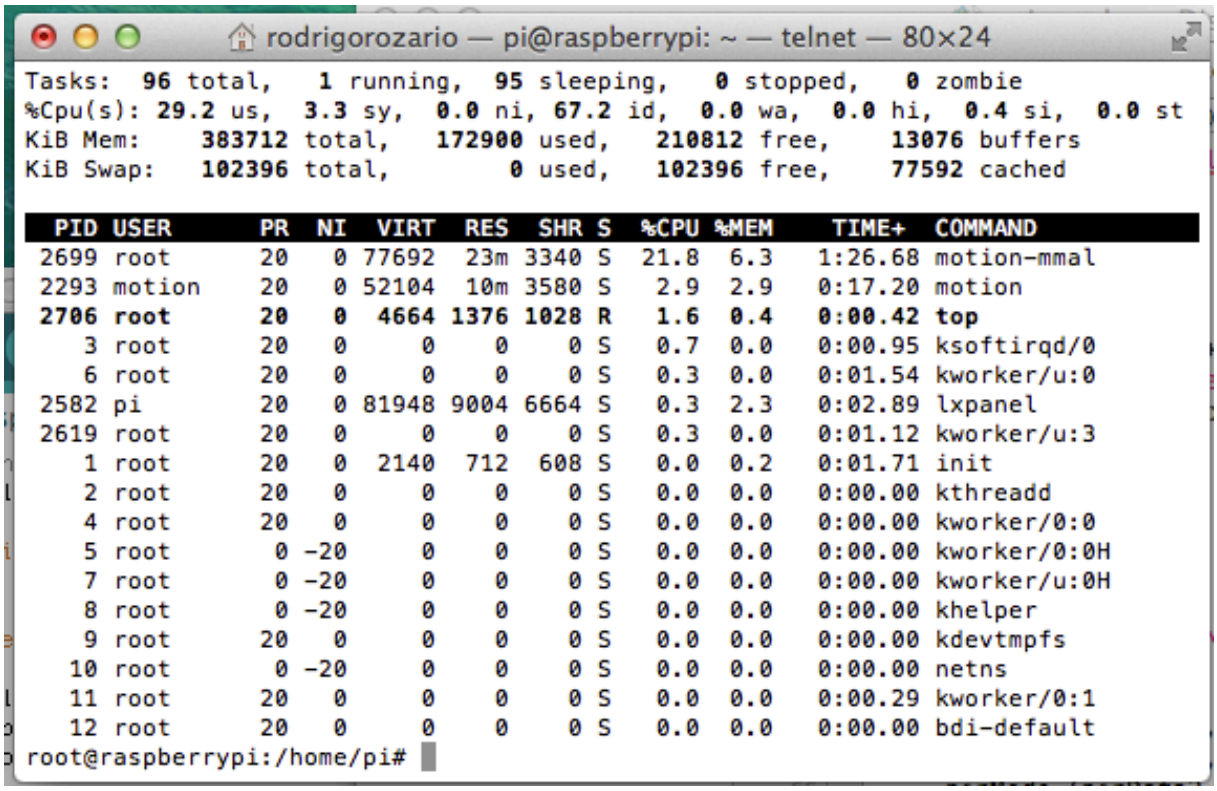

Figura 24 Teste de Processamento Sem Movimento Diante das Câmeras

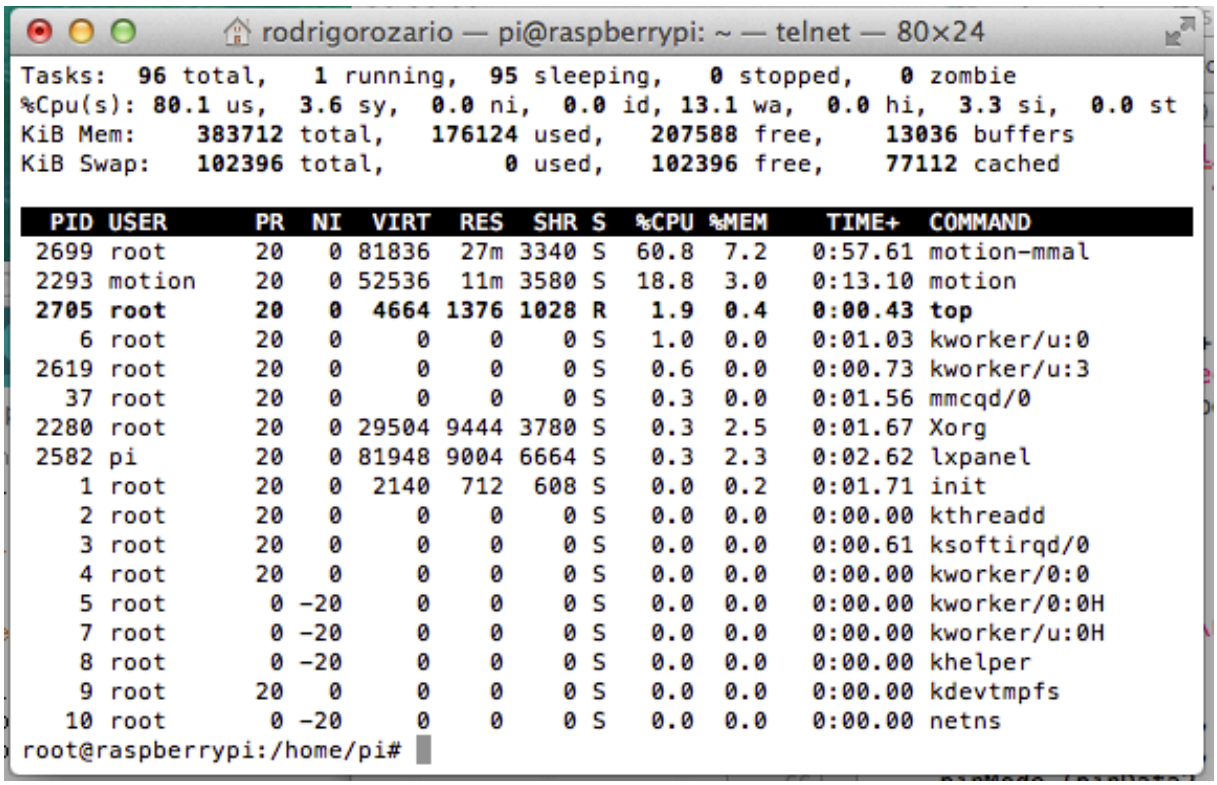

Figura 25 Teste de Processamento Com Movimento Diante das Câmeras

Para verificação do segundo ponto, a estratégia utilizada foi reduzir o sinal recebido removendo a antena instalada no módulo. O que poderia causar problema é o módulo entrar em um estado que necessitasse de ação sobre o módulo, como desligar e ligar, para que pudesse novamente conectar à rede celular. Felizmente, ao instalar a antena o módulo automaticamente reconecta sem necessidade de alguma ação.

Em relação a utilização da memória interna instalada, podemos verificar o tamanho da imagem gerada e a disponibilidade de memória interna. Em um espaço amostral de 54 imagens coletadas pelo sensor de movimento na RaspCAM (que é a câmera de maior resolução, e portanto, maior tamanho da imagem) a média de tamanho é 46,57kB por imagem. Na Figura 25 podemos ver que temos 3.7GB de espaço livre. Assim, poderemos armazenar aproximadamente 83.305 imagens no espaço restante.

```
root@raspberrypi:/home/pi# df -h /
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       5.9G  2.0G  3.7G  36% /
root@raspberrypi:/home/pi#
```

Figura 26 Espaço Livre no Protótipo de Testes

Por padrão, o Motion MMAL captura dois quadros por segundo. Assim, teríamos disponível 11,57 horas (11 horas e 34 minutos aproximadamente) de gravação de movimento. Para casos em que for

necessário maior capacidade de armazenamento, é possível incluir um HD externo através da porta USB e ampliar assim a capacidade de armazenamento.

4 SERVIÇOS OFERECIDOS

Esta seção propõe-se a apresentar os serviços oferecidos aos usuários do sistema e as opções de personalização oferecidas.

Diante da evidente integração com o usuário do sistema, será necessário apresentar as opções de configuração e personalização do sistema proposto para que este possa ser bem operado, apesar da simplicidade. Devido a simplicidade e de estar projetado de forma modular, iremos propor também alguns módulos a serem implementados futuramente e a maneira como serão integrados ao sistema atual.

Nesta seção serão apresentados os serviços oferecidos pelo sistema e suas opções de personalização. Nesse sentido, serão descritas as funcionalidades disponíveis e suas formas de utilização, desde o aspecto específico até as opções gerais disponíveis. Apresentaremos também algumas propostas de serviços que ainda podem ser desenvolvidos diante do hardware proposto. Por fim, propomos também inclusão de novas funcionalidades que dependam do desenvolvimento de novos módulos.

A seguir propomos as diretrizes gerais para desenvolvimento do sistema e as opções de personalização desenvolvidas. Como objetivo final, o sistema deve prover ao usuário as seguintes funcionalidades:

- Realizar captura e armazenamento de imagens e vídeos;
- Detectar movimento para início e fim do armazenamento das imagens, economizando assim espaço de armazenamento interno.
- Permitir configuração do modo de armazenamento de vídeo e envio de comandos via celular.
- Avisar, via rede GPRS/GSM, de movimentos detectados nas câmeras.
- Ativar e desativar o sistema de aviso de movimentos pelo celular.

4.1 COMANDOS ENVIADOS VIA CELULAR

O usuário do sistema poderá alterar o estado do sistema através de uma ligação ou envio de mensagem para o SIMCARD conectado ao módulo GSM/GPRS. Os comandos já desenvolvidos são apresentados na Tabela 10 onde estão descritas as ações executadas pelo sistema após o recebimento do comando. Por exemplo, é possível ativar e desativar o detector de movimento em cada das câmeras instaladas.

O sistema foi configurado de forma que ao receber chamadas a partir de um telefone, seja ele móvel ou fixo, atenda automaticamente e espere receber o tom do número correspondente à ação que quer executar. A Tabela 10 especifica qual ação é executada quando o número correspondente é digitado:

Tabela 10 Comandos Remotos Via Ligação

Número digitado	Ação
1	Ativa gravação da câmera USB.
2	Desativa gravação da câmera USB.
3	Ativa gravação da câmera RaspCAM
4	Desativa gravaçãoo da câmera RaspCAM

As teclas não citadas (5, 6, 7, 8, 9, 0, # e *) não possuem função definida para esta versão do sistema. Em versões futuras, em que forem adicionados novos serviços, essas teclas poderão ser utilizadas.

As mesmas ações citadas na tabela anterior poderiam ser realizadas com o envio de mensagem. A diferença está no comando enviado no corpo da mensagem. A configuração dar-se-á conforme a Tabela 11.

Tabela 11 Comandos Remotos Via Mensagem de Texto

Número digitado	Ação
Conf: desativa movimento	Desativação do alarme de movimento.
Conf: ativa movimento	Ativação do alarme de movimento.
Conf: ativa grav 1	Ativa gravação da câmera 1.
Conf: desativa grav 1	Desativa gravação da câmera 1.
Conf: ativa grav 2	Ativa gravação da câmera 2.
Conf: desativa grav 2	Desativa gravação da câmera 1.

É de grande importância o cadastramento do número do celular que poderá realizar as configurações citadas acima. Isso evita que outro usuário que conheça o número do SIMCARD instalado no sistema possa, de forma intencional ou não, alterar configurações. O cadastramento dos números autorizados a alterar as configurações será feito através da interface web, que poderá ser acessada via rede local, onde será cadastrado o número do celular, nome do usuário e uma senha numérica.

A utilização da senha numérica e do número do celular é um incremento na segurança em casos de roubo ou extravio de aparelho. Sua utilização para autenticação do usuário para envio de comandos por mensagem torna o sistema inseguro, haja visto que as mensagens enviadas ficam no histórico do

celular e que podem ser visualizadas por qualquer um que tenha acesso ao aparelho. Por esse motivo, essa funcionalidade não foi implementada.

Diversos aplicativos desenvolvidos para os celulares mais modernos (os Smart Phones) são capazes de enviar mensagens. Para implementação da ideia anterior, seria necessário o desenvolvimento de aplicativo próprio para autenticação do usuário de forma local, no próprio aparelho celular. Assim, a autenticação através da senha direto no aplicativo do celular evitaria a falha de segurança anteriormente citada.

4.2 ESTADOS DO MÓDULO GSM/GPRS

O módulo GSM/GPRS foi projetado de forma independente do resto do sistema. Esse módulo é capaz de executar as funções designadas a ele, processar e entregar ao Raspberry Pi o código do comando a ser executado.

Na Tabela 12 estão descritos os estados que o módulo GPRS pode assumir e a descrição do que é executado em cada um deles.

Tabela 12 Estados do Módulo GPRS/GSM

Estado	Descrição
Inicial	Este é o estado que o módulo estará quando é ligado
Conectando à rede	Etapa em que o módulo executa os procedimento necessários para conectar-se à rede celular
Loop Esperar ligação	Laço onde o módulo espera receber ligação.
Ligação ativa	Etapa em que o módulo está com uma ligação ativa.
Comando recebido	Com uma ligação ativa, o módulo entra neste estado quando recebe, através do detector DTMF algum comando.
Término da ligação	Estado momentâneo que marca o fim de uma ligação.

Na Figura 26 apresentamos a máquina de estados sob a qual o sistema foi estruturado e programado.

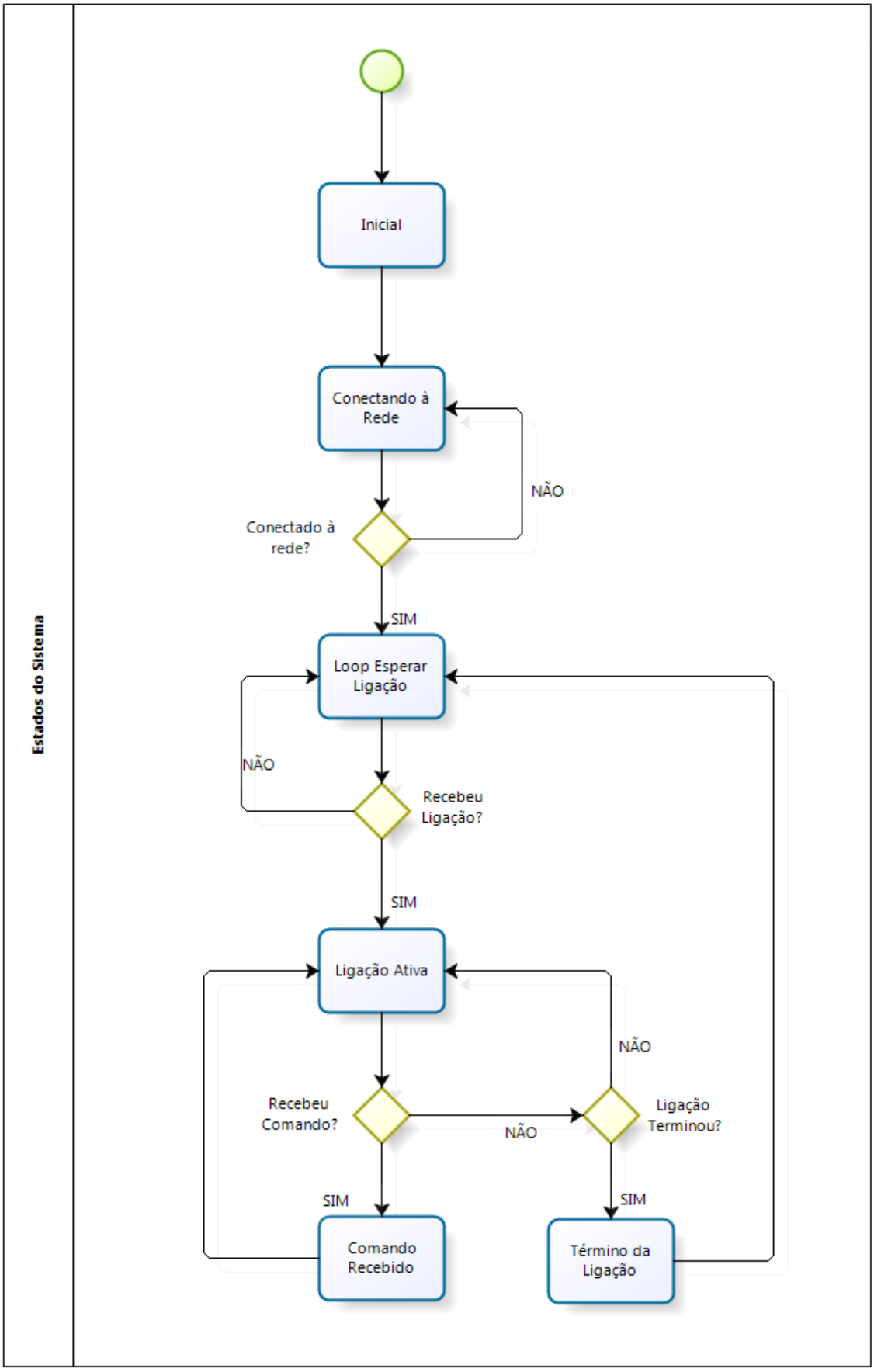

Figura 27 Diagrama de Estados do Sistema

Assim que o sistema é ligado, ele tenta se conectar à rede celular através do módulo GPRS/GSM. Permanece no estado “Conectando à Rede” até que a conexão esteja estabelecida. Quando conectado, entra então no estado “Loop Esperar Ligação” em que permanece até que receba uma ligação. Quando recebe uma ligação, que é atendida automaticamente o sistema entra no estado “Ligação Ativa” onde espera receber um comando a ser executado ou que a ligação termine. Ao receber um comando válido, o sistema entra no estado “Comando Recebido” e assim que termina identificar e executar o comando retorna ao estado “Ligação Ativa”. Quando a ligação é desligada, o sistema entra no estado “Término da Ligação” e logo vai para o estado “Loop Esperar Ligação”.

4.3 MENSAGENS NO DISPLAY LCD

O *Display* é utilizado para indicar o estado operacional do módulo GPRS e auxiliar em possíveis problemas de conexão à rede GSM/GPRS. A Tabela 13 indicam os estados e seus respectivos significados. Alguns desses estados têm mensagens apresentadas no display LCD, outros não.

Tabela 13 Estados Específicos do Módulo GPRS/GSM

Mensagem	Descrição
Iniciando...	É o estado em que o sistema se encontra logo que o módulo é ligado. Indica que o sistema está iniciando os componentes necessários.
Conectando...	Estado em que o SIM900 tenta se conectar à rede celular.
Conectado!	Indica que há conexão com a rede celular ativa e pronta para ser utilizada.
Desconectado!	Indica que o módulo iniciou mas não está conectado à rede celular.
Sem SIMCARD	Indica que não há SIMCARD instalado no módulo.

O diagrama apresentado na Figura 27 é um detalhamento do estado “Conectando à Rede” apresentado no diagrama da Figura 26. Esse processo começa verificando se o SIMCARD esta instalado no módulo GPRS/GSM. Caso não esteja, o sistema entra no estado “Sem SIMCARD”, onde é disponibilizado no display LCD essa informação. Se o SIMCARD está devidamente instalado módulo GPRS/GSM entra no estado “Conectando...” em que inicia a tentativa de autenticação com a rede celular. Caso encontre rede disponível e consiga se conectar, o sistema entra no estado “Conectado”, saindo somente quando perder a conexão obtida anteriormente. Caso perca a conexão, o sistema entra no estado “Desconectado.”, que prontamente é levado novamente ao estado “Conectando” e processo de solicitação de conexão é novamente iniciado.

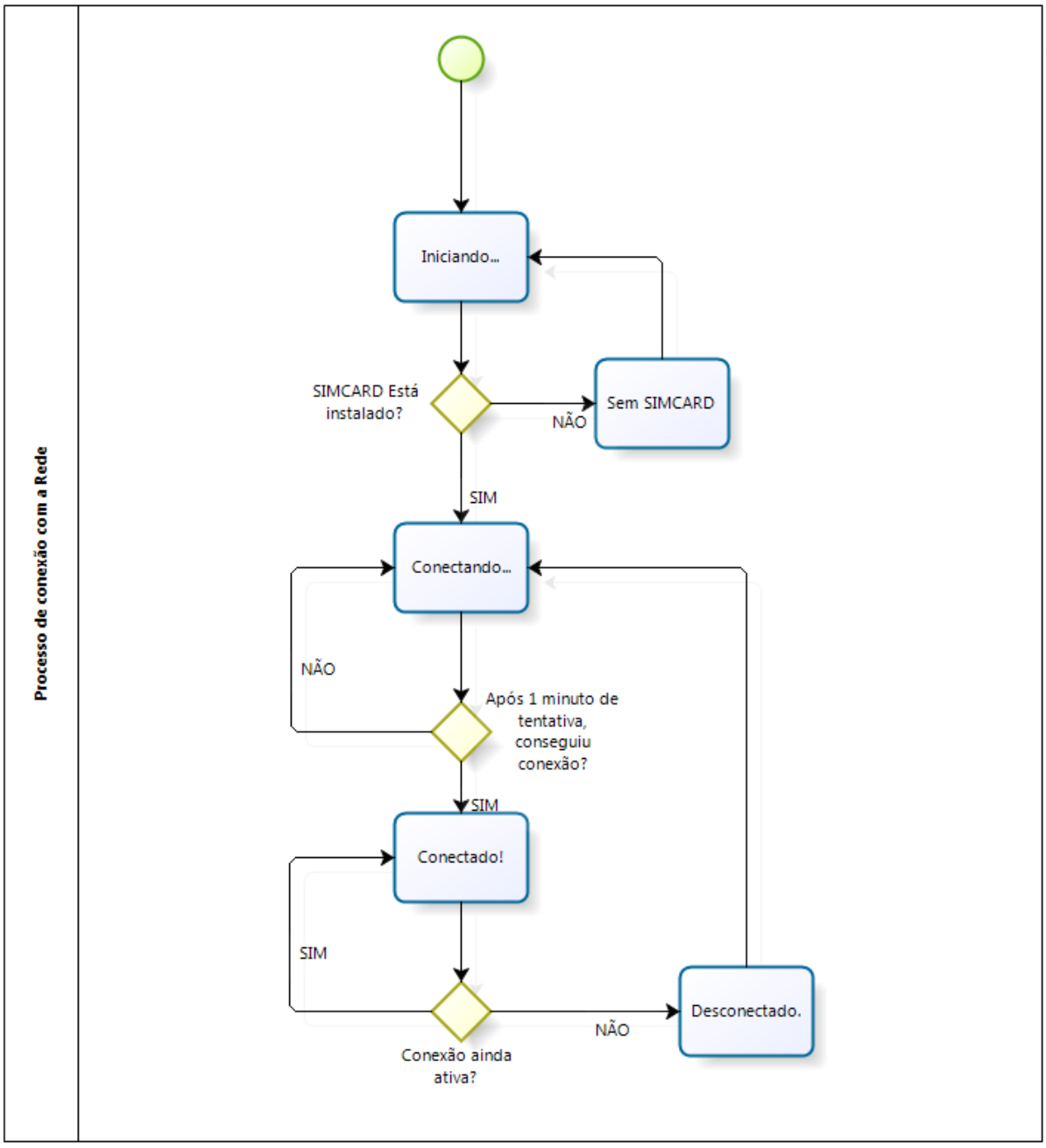

Figura 28 Diagrama de Estados do Módulo GPRS/GSM

4.4 CENÁRIO DE APLICAÇÃO

O sistema desenvolvido pode ser aplicado em diversos ambientes. As aplicações mais comuns são as residenciais, capturando movimento de pessoas dentro de casa. Nesse tipo de ambiente, a principal necessidade é o monitoramento do ambiente nos momentos em que a residência esta sozinha, momentos em que os donos estão fora para trabalho ou alguma viagem.

Cenários empresariais também podem ser monitorados com o sistema proposto, desde os convencionais (escritórios, restaurantes, etc.) até os críticos (datacenters, salas cofre, agencias bancárias).

O software Motion permite subdividir um frame em diversos setores e detectar movimento separadamente em cada um deles e a configuração do nível de limiar de movimento. Essa opção permite que o sistema seja implantado, por exemplo, em um estacionamento privado para monitorar os carros. Seria necessário que o dono do carro, no momento em que fosse retirar o carro da vaga, desativasse o alarme de movimento para o setor da câmera que está monitorando o local.

5 CONCLUSÕES

Serão apresentados as considerações finais sobre o desenvolvimento do sistema proposto.

Diante do evidente aumento da violência nas grandes cidades brasileiras, são necessárias ações para tentar minimizar ou suprimir os danos causados à população. Não obstante, politicas públicas para combate a esse problema são de fundamental importância para que o caos não seja instaurado.

O sistema proposto no presente trabalho inicia o desenvolvimento de um produto modular, integrável e dinâmico capaz de se adaptar a diversos ambientes e aplicações. Diferentemente dos dispositivos disponíveis hoje no mercado, que são altamente padronizados, apresentamos um protótipo para um ambiente totalmente modular. O sistema foi desenvolvido em código aberto assim é possível o desenvolvimento de novos módulos e dispositivos que agreguem funcionalidades antes não disponibilizadas.

Inicialmente utilizamos o Raspberry Pi como processador principal para controlar as rotinas necessárias para aquisição de imagens, com dois modelos de câmeras no intuito de dar flexibilidade na escolha dos dispositivos. Assim, surgiu a necessidade de executar comandos de forma remota, que foi suprida incluindo um módulo GPRS/GSM. Por conveniência e facilidade de implementação de funções disponíveis para Arduino, optamos por controlar o módulo GPRS/GSM com um Arduino e não com o Raspberry Pi.

O detector DTMF completa o módulo GPRS/GSM na tarefa de tornar disponível a execução de comandos remotos, na medida em que permite identificar teclas digitadas em uma ligação ativa. O detector DTMF, em particular, apresentou problemas de funcionamento e teve que ser modificado para que pudesse integrar o sistema (problema relatado no item 2.4).

Outro problema encontrado é a instabilidade do programa Motion MMAL, que diversas vezes é interrompido e para a detecção de movimento na RaspCAM. Algumas vezes, quando desativamos o programa e tentamos reativa-lo ele não retorna, sendo necessário o reinício do Raspberry Pi para que volte a funcionar.

Apesar dos problemas encontrados, contornados ou não, o sistema final proposto é funcional no sentido de que atende de forma satisfatória os objetivos propostos no item 1.2 além de executar funções presentes nos dispositivos comercializados atualmente. Adicionalmente, conseguimos desenvolver o dispositivo sem a utilização de *softwares* proprietários e totalmente modular, característica que torna possível o desenvolvimento colaborativo de novos dispositivos e de novas funções.

Pensando na possibilidade de desenvolvimento colaborativo de novos dispositivos, algumas propostas inicias são apresentadas a seguir que tornarão o sistema ainda mais funcional.

Uma proposta é uma funcionalidade geral, que abrange todos os módulos instalados: uma página web para alterar configurações do sistema. Nessa página seriam disponibilizadas as seguintes funcionalidades:

- Acesso às câmeras ao vivo;
- Alterar características das imagens e câmeras;
- Características gerais do sistema;
- Configurações do Sistema;
- Verificar estado do módulo GPRS/GSM;
- Apresentar logs de movimentos e acesso às imagens armazenadas.

Uma segunda proposta seria desenvolver um módulo 3G ou 4G para integração com redes mais atuais e disponibilizar o acesso à Internet através dessas redes ao sistema, que poderia enviar atualizações de eventos ocorridos a um servidor disponível na rede mundial, e que poderia ser prontamente acessado de qualquer lugar.

A depender da necessidade da aplicação, é possível a inclusão de sensores (presença, luminosidade, temperatura, pressão, nível de líquidos, etc) para monitoramento de ambientes. Esses dispositivos são facilmente integráveis e podem ser empregados em diversas aplicações. Um projeto que mostra a utilização do Raspberry Pi com sensores de umidade é mostrado na referência [17], onde o autor desenvolve um dispositivo automático para monitorar umidade em plantações.

Assim, o sistema projetado atende satisfatoriamente às funções a que foi designado além de manter a característica modular que é essencial para a diferenciação do sistema proposto se comparado com os sistemas atualmente presentes no mercado.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] **Matéria do Site Jornal Hoje**. Disponível em: http://g1.globo.com/jornal-hoje/noticia/2012/07/seguranca-privada-no-brasil-cresce-74-nos-ultimos-dez-anos.html. Acessado em: 20/02/2014.

[2] **Matéria do Portal ODM**. Disponível em: http://www.portalodm.com.br/brasileiro-elege-saude-seguranca-e-educacao-como-prioridades--n--1161.html. Acessado em: 20/02/2014

[3] **Site do Rasberry Pi**. Disponível em: http://www.raspberrypi.org/downloads. Acessado em: 20/02/2014.

[4] **Projeto Pi4J**. Disponível em: http://pi4j.com/ Acessado em: 20/02/2014

[5] **Datasheet do SIM900.** Disponível em: ftp://imall.iteadstudio.com/IM120417009_IComSat/DOC_SIM900_Hardware%20Design_V2.00.pdf. Acessado em: 20/02/2014.

[6] **Normas ITU para commandos AT**. Disponível em: http://www.itu.int/rec/T-REC-V.25ter-199508-S/en). Acessado em: 20/02/2014

[7] Datasheet do DTMF

[8] **Site Logitech**. Disponível em: http://www.logitech.com/pt-br/home Acessado em: 20/02/2014

[9] **Site do modulo camera para Raspberry Pi.** Disponível em: http://www.raspberrypi-spy.co.uk/2013/05/the-official-raspberry-pi-camera-module/. Acessado em: 20/02/2014

[10] **Biblioteca para Camera Serial C328**. Disponível em: http://mbed.org/users/shintamainjp/notebook/camera_c328_en/. Acessado em: 20/02/2014.

[11] **Módulo Wifi para Raspberry Pi**. Disponível em: http://detroit-electronics.com/WiPi-Wifi-For-Raspberry-Pi-Comfast Acessado em: 20/02/2014.

[12] **Documentação da API MMAL**. Disponível em: http://www.jvcref.com/files/PI/documentation/html/. Acessado em: 20/02/2014.

[13] **Documentação do Raspberry Pi**. Disponível em: http://elinux.org/Raspberry_Pi_VideoCore_APIs . Acessado em: 20/02/2014.

[14] **Tutoral de LCD no site Labdegaragem.com**. Disponível em: http://labdegaragem.com/profiles/blogs/tutorial-lcd-com-arduino. Acessado em: 20/02/2014.

[15] **Nazzetta, Diogo Cian**. 2013. INTERSAFE – Segurança Veicular Interativa.

[16] **Tutorial de montagem e programação de modulo 3G**. Disponível em: http://www.cooking-hacks.com/documentation/tutorials/raspberry-pi-3g-gprs-gsm-gps. Acessado em: 20/02/2014.

[17] **Projeto utilizando Raspberry Pi para monitorar umidade**. Disponível em: http://www.esologic.com/?tag=raspberry-pi . Acessado em: 20/02/2014.

[18] **Huang, Shih-Chia e Cheng, Fan-Chieh**. 2012. Motion detection with pyramid structure of background model for intelligent surveillance systems. *Engineering Applications of Artificial Intelligence*. 2012.

[19] **Zaki, Wan, Hussein, Aini e Hedayati, Mohamed**. 2013. Moving object detection using keypoints reference model. *EURASIP Journal on Image and Video Processing*. 2013.

APÊNDICES

Apêndice A. GPIOREAD.C – PROGRAMA PRINCIPAL

```
#include <wiringPi.h>
#include <stdio.h>
#include <time.h>
#include <wiringSerial.h>
#include <errno.h>
#include <string.h>

#define pinDV 4
#define pinData0 0
#define pinData1 1
#define pinData2 2
#define pinData3 3
#define pinOE 5


volatile int eventCounter = 0;
int digitosrecebidos = 0;
int dado0 = 0;
int dado1 = 0;
int dado2 = 0;
int dado3 = 0;

// myInterrupt:  called every time an event occurs
void myInterrupt(void) {
      printf("\nDigito válido recebido\n");
      dado0 = digitalRead(pinData0);
      dado1 = digitalRead(pinData1);
      dado2 = digitalRead(pinData2);
      dado3 = digitalRead(pinData3);

    printf("dados: %d %d %d %d\n", dado0, dado1, dado2, dado3);

    if ((dado0 == 1)&&(dado1 == 0)&&(dado2==0)&&(dado3==0)){
      system("service motion start");
      printf("Tecla 1 pressionada!");
    }
    if ((dado0 == 0)&&(dado1 == 1)&&(dado2==0)&&(dado3==0)){
      system("service motion stop");
      printf("Tecla 2 pressionada!");
    }
    if ((dado0 == 1)&&(dado1 == 1)&&(dado2==0)&&(dado3==0)){
      system("./mmal/startmotionmmal");
      printf("Tecla 3 pressionada!");
    }
    if ((dado0 == 0)&&(dado1 == 0)&&(dado2==1)&&(dado3==0)){
      system("./mmal/stopmotionmmal");
      printf("Tecla 4 pressionada!");
    }



    digitosrecebidos++;
    printf("Digitos recebidos: %d\n", digitosrecebidos);
    delay(2000); //Espera 2 Segundo
}

```

```
int main(void){
      //int pin;
      wiringPiSetup();

      printf("\nPronto.\n");


      pinMode (pinData0, INPUT);
      pinMode (pinData1, INPUT);
      pinMode (pinData2, INPUT);
      pinMode (pinData3, INPUT);
      pinMode (pinOE, OUTPUT);
      pinMode (pinDV, INPUT);

      // set Pin 17/0 generate an interrupt on high-to-low transitions
      // and attach myInterrupt() to the interrupt
      if ( wiringPiISR (pinDV, INT_EDGE_RISING, &myInterrupt) < 0 ) {
      fprintf (stderr, "Unable to setup ISR: %s\n", strerror (errno));
      return 1;
      }

      //wiringPiISR(pinDV, INT_EDGE_RISING, myInterrupt());

      while(1){

      }

}
```

Apêndice B. Espera chamada – Programa para arduino

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12,11,5,4,3,2);

void switchModule();
boolean check_response(char *str);

int led = 9;
int button = 8;
int onModulePin = 2;        // the pin to switch on the module (without press on button)

int timesToSend = 1;        // Numbers of calls to make

void switchModule(){
    digitalWrite(onModulePin,HIGH);
    delay(2000);
    digitalWrite(onModulePin,LOW);
}

void setup(){

    Serial.begin(9600);               // UART baud rate
    //delay(2000);
    //pinMode(button, INPUT);
    //pinMode(led, OUTPUT);
    //pinMode(onModulePin, OUTPUT);
    //switchModule();                  // switches the module ON
    //Configuracoes LCD
    lcd.begin(16, 2); //configura o LCP para 2 linhas e 16 colunas
    lcd.print("pronto!"); //escreve Pronto

    Serial.println("ATS0=1"); //configura modulo para atender automaticamente (=0 para desa-
tivar)
    lcd.setCursor(0,1);//Posiciona o cursor na posicao 0 da segunda linha
    lcd.print("atend. auto"); //Escreve atend. auto


    for (int i=0;i < 5;i++){
        //delay(5000);
    }
}

void loop(){

    Serial.println("iniciando loop");
    //while(!check_response("RING"));   //waits for a call

    //Serial.println("ATA");          //answers an incoming call



    //delay(10000);

    //while(digitalRead(button)==1);

    //Serial.println("AT+CHUP");           // disconnects the existing call
    char res = 'a';
    char inData[20];
```

```
    char inChar;
    byte index = 0;

    while(true){
    //Serial.println("aqui");

      Serial.flush();
      Serial.println("AT+CREDG?");
      Serial.println(Serial.available());
      while (Serial.available() > 0) {
         if (index< 19){
           inChar = Serial.read();
           inData[index] = inChar;
           index++;
           inData[index] = '\0';
         }
      }
      res = inData[2];
      Serial.println(inData[3]);
      //lcd.setCursor(1,1);
      //lcd.print(inData);
      delay(5000);
    }

}
```

Apêndice C. STARTMOTIONMMAL - SCRIPT PARA INICIAR A CAPTURA DE IMAGENS NO MOTION MMAL

```
#!/bin/sh
nohup /home/pi/mmal/motion-mmal -n -c motion-mmalcam.conf 1>/dev/null 2>&1 </dev/null &
```

Anexo D. STOPMOTIONMMAL - SCRIPT PARA PARAR A CAPTURA DE IMAGENS NO MOTION MMAL

```
#!/bin/sh
ps -ef | grep motion-mmal | awk '{print $2}' | xargs kill
```