UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTOS ACADÊMICOS DE ELETRÔNICA E MECÂNICA
CURSO SUPERIOR DE TECNOLOGIA EM MECATRÔNICA INDUSTRIAL

LUIZ AUGUSTO SANTOS DE OLIVEIRA
PEDRO DOS SANTOS HORST

SISTEMA DE MONITORAMENTO DE EMBARCAÇÃO UTILIZANDO COMUNICAÇÃO BLUETOOTH EM SMARTPHONE

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2017

LUIZ AUGUSTO SANTOS DE OLIVEIRA
PEDRO DOS SANTOS HORST

SISTEMA DE MONITORAMENTO DE EMBARCAÇÃO UTILIZANDO COMUNICAÇÃO BLUETOOTH EM SMARTPHONE

Proposta de Trabalho de Conclusão de Curso de Graduação, apresentado ao Curso Superior de Tecnologia em Mecatrônica Industrial, dos Departamentos Acadêmicos de Eletrônica e Mecânica, da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.
Orientador: Prof. Msc. Francisco Muller.

CURITIBA
OUTUBRO/2017

TERMO DE APROVAÇÃO

LUIZ AUGUSTO SANTOS DE OLIVEIRA
PEDRO DOS SANTOS HORST

SISTEMA DE MONITORAMENTO DE EMBARCAÇÃO UTILIZANDO COMUNICAÇÃO BLUETOOTH E SMARTPHONE

Este trabalho de conclusão de curso foi apresentado no dia 17 de outubro de 2017, como requisito parcial para obtenção do título de Tecnólogo em Mecatrônica Industrial, outorgado pela Universidade Tecnológica Federal do Paraná. Os alunos foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Prof. Dr. Milton Luiz Polli
Coordenador de Curso
Departamento Acadêmico de Mecânica

_____
Prof. Msc. Sérgio Moribe
Responsável pela Atividade de Trabalho de Conclusão de Curso
Departamento Acadêmico de Eletrônica

BANCA EXAMINADORA

_____
Prof. Esp. Sergio Luiz Bazan de Paula
UTFPR

_____
Prof. Msc. Sérgio Moribe
UTFPR

_____
Prof. Msc. Alexandre Jorge Miziara
UTFPR

“A Folha de Aprovação assinada encontra-se na Coordenação do Curso

Dedicamos este trabalho aos nossos pais, pois estes foram os maiores incentivadores das nossas realizações.

AGRADECIMENTOS

Agradecemos a nosso pais que sempre nos incentivaram e se esforçaram muito para nos ajudar a alcançarmos nossos objetivos.

Agradecemos a nossos professores por nos passarem o conhecimento necessário durante o período acadêmico nos capacitando para o mercado de trabalho e para a vida.

Agradecemos especialmente nossos orientadores Eduardo Bertonha e Francisco Muller por realizarem uma gestão eficiente do nosso trabalho e também por serem ótimas pessoas, o que tornou possível alcançarmos esse objetivo de forma bastante tranquila.

Agradecemos aos amigos durante período acadêmico, pois juntos, realizamos muitas atividades que nos proporcionaram crescimento pessoal e profissional, e muitos se tornaram amigos que levaremos para a vida toda.

RESUMO

OLIVEIRA, Luiz Augusto Santos de; HORST, Pedro dos Santos. **Sistema de Monitoramento de Embarcação utilizando Bluetooth e Smartphone**. 2017. 64f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Mecatrônica Industrial), Departamentos Acadêmicos de Eletrônica e Mecânica, Universidade Tecnológica Federal do Paraná. Curitiba 2017.

A automação se torna cada vez mais comum no dia a dia das pessoas, e sua aplicação se estende nos mais diversos campos de atuação. Na indústria naval a utilização de equipamentos como PLC's, IHM's e Pc's industriais, se tornaram indispensáveis em grandes embarcações no aprimoramento de funções como: sistemas de navegação, gestão dos motores, controle e monitoração da carga, gerenciamento de energia, gerenciamento de potência e posicionamento dinâmico. (VIDAL, 2009). Este trabalho tem a proposta de apresentar um protótipo de um sistema de monitoramento microcontrolado para embarcações de médio porte, onde através de um *Smartphone* será possível, emular o monitoramento de sinais e a realização de comandos existentes em um painel de controle de uma embarcação.

Palavras chave: *Automação. Aplicativo. Microcontrolador. Bluetooth*. *Smartphone*. *Software*.

ABSTRACT

OLIVEIRA, Luiz Augusto Santos de; HORST, Pedro dos Santos. Sistema de Monitoramento de Embarcação utilizando Bluetooth e Smartphone. 2017. 64f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Mecatrônica Industrial), Departamentos Acadêmicos de Eletrônica e Mecânica, Universidade Tecnológica Federal do Paraná. Curitiba 2017.

Automation becomes more and more common in people's daily lives, and its application extends to a wide range of fields. In the naval industry, the use of equipment such as PLC's, IHM's and industrial Pc's have become indispensable in large vessels in the improvement of functions such as navigation systems, engine management, cargo control and monitoring, power management, power management and positioning dynamic. (VIDAL, 2009). This work has the proposal to present a prototype of a microcontrolled monitoring system for medium sized vessels, where through a smartphone it will be possible to emulate the monitoring of signals and the realization of existing commands in a control panel of a vessel.

Keywords: *Automation. App. Microcontroller. Bluetooth. Smartphone. Software. System.*

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

DSSS	Espectro de Propagação de Sequência Direta *(Direct Sequence Spread Spectrum)*
EEPROM	Memória Programável Eletricamente Apagável Somente de Leitura *(Eletrically Erasable Programmable Read-Only Memory)*
FHSS	Espectro de Propagação de Frequência *(Frequency Hopping Spread Spectrum)*
GPS	Sistema de Posicionamento Global (*Global Position System)*
GSM	Sistema Global para Comunicações Móveis *(Groupe Special Mobile)*
HID	Perfil de Dispositivos de Interface Humana (*Human Interface Device Profile)*
HTTP	Protocolo de Transferência de Hipertexto (*HyperText Transfer Protocol)*
IDC	Centro de Dados da Internet *(Internet Data Center)*
IP	Protocolo de Internet (*Internet Protocol)*
ISM	Industrial, Científico e Médico (*Industrial, Scientific, Medical*),
ISSO	Organização Internacional de Padronização (*International Standards Organization)*
ISP	Fornecedor de Acesso à Internet (*Internet Service Provider)*
LED	Diodo Emissor de Luz *(Light Emitting Diode)*
MISO	Entrada Mestre e Saída *Escravo (Master In Slave Out)*
MOSI	Saída Mestre e Entrada Escravo *(Master Out Slave In)*
PDA	Assistente Pessoal Digital (*Personal Digital Assistants)*
RAM	Memória de Acesso Aleatório *(Random Access Memory)*
PWM	Modulação por Largura de Pulso (*Pulse Width Modulation)*
RTC	Relógio de Tempo Real *(Real Time Clock)*
SCK	Clock de Sincronização *(Serial Clock)*
SCL	Sinal de Clock *(Clock Signal)*
SD	Digital Seguro (*Secure Digital)*
SDA	Sinal de Dados (Data Signal)
SPI	Interface Serial Periférica *(Serial Peripheral Interface)*
SS	Seletor de Escravo *(Select Slave)*
USB	Conexão Serial Universal *(Universal Serial Bus)*
WWW	Rede de Alcance Mundial (*World Wide Web)*
UART	Transmissor/Receptor Universal Assíncrono *(Universal Asynchronous Receiver Trasmitter)*
USART	Transmissor/Receptor Universal Síncrono e Assíncrono *(Universal Synchronous Asynchronous Receiver Trasmitter)*

SUMÁRIO

1 INTRODUÇÃO

Desde os primórdios de sua origem, o ser humano sempre utilizou ferramentas e utensílios que o auxiliaram na realização de diversas atividades relacionadas às suas necessidades de sobrevivência.

A partir da invenção da máquina a vapor de James Watt em 1769, houve um acentuado progresso em termos de automação de processos produtivos. Para controlar a velocidade das máquinas a vapor Watt desenvolveu um mecanismo chamado de regulador de esferas, este foi o primeiro controlador automático com retroação criado. (DORF; BISHOP, 2006).

"A automação é uma associação de equipamentos eletrônicos e/ou mecânicos que controlam seu próprio funcionamento, quase sempre sem a intervenção humana. " (VIDAL, 2009, p13).

O trabalho de conclusão de curso de Tecnologia em Mecatrônica Industrial tem como proposta desenvolver um sistema de automação para uma embarcação de médio porte, e monitorada por um sistema operacional de dispositivos móveis.

1.1 TEMA

O objetivo deste projeto é apresentar um protótipo para simular o recebimento de alarmes e execução de comandos via Bluetooth, entre um *Smartphone* e uma placa eletrônica contendo um microcontrolador 8-BIT. Os sinais simulados serão os de um painel de controle de uma embarcação de médio porte utilizada para transporte de passageiros.

1.2 PROBLEMA

O consumo de *Smartphones* vem crescendo ultimamente e a utilização de aplicativos está cada vez mais comum no dia a dia das pessoas, de forma que a praticidade que várias dessas aplicações oferecem, aliadas ao baixo custo, tornam esse mercado muito atrativo.

Dados da consulta IDC indicam que no segundo trimestre de 2013 foram vendidos 8,3 milhões de *Smartphone*s no Brasil, um aumento de 110% da comercialização desses equipamentos em relação ao mesmo período de 2012 (EXAME, 2014).

Nesse contexto, a pergunta é a seguinte: É possível desenvolver um aplicativo para *Smartphone* que facilite o dia a dia dos tripulantes de uma embarcação para o monitoramento de dados desta?

A premissa do projeto consiste em apresentar um protótipo para simular a supervisão e controle feita na sala de comando da embarcação, que normalmente é feita através do monitoramento de instrumentos e execução de comandos no painel de controle desta, para que também se possa efetuar essa supervisão e controle de forma remota usando-se um *Smartphone* que se comunicará com uma placa eletrônica microcontrolada que se integrará aos circuitos existente na embarcação responsáveis pelo controle e supervisão da mesma.

1.3 JUSTIFICATIVA

A automação em embarcações de grande porte é fundamental para a navegação e controle de dispositivos destas, porém com a popularidade que os microcontroladores ganharam, cada vez mais são desenvolvidos sistemas dedicados para embarcações de médio e pequeno porte.

O projeto consiste em disponibilizar remotamente o monitoramento de variáveis e comando de dispositivos, que em modo local são acionados por chaves e botões no painel de controle da cabine de comando, para que se possa também, através de um celular em modo remoto, dentro de um raio de 20 metros da sala de comando, efetuar tais procedimentos.

As variáveis a serem monitoradas serão: temperatura do motor, pressão do óleo e rotação do motor em rotações por minuto (rpm).

Os acionamentos a serem feitos serão: as luzes internas da embarcação, luzes de navegação, bomba de porão, partida e desligamento do motor. O sistema utilizará um microcontrolador na placa de controle e supervisão e um *Smartphone*, ambos se comunicando via Bluetooth.

1.4 OBJETIVOS

1.4.1 OBJETIVO GERAL

Desenvolver um sistema de automação que possibilite receber alertas e executar comandos provenientes do painel de controle de uma embarcação de transporte de passageiros, em modo remoto utilizando-se um *Smartphone* e uma placa de controle dedicada microcontrolada ambas com comunicação de dados via *Bluetooth*.

1.4.2 OBJETIVOS ESPECÍFICOS

- Estudar sistemas de automação naval existentes;
- Identificar os dispositivos de controle de uma embarcação;
- Estudar o funcionamento do *Bluetooth* em um *Smartphone*;
- Desenvolver um circuito de comunicação com microcontrolador via *Bluetooth*;
- Desenvolver os circuitos necessários para os sensores que serão utilizados;
- Desenvolver a programação para o microcontrolador;
- Desenvolver o software do aplicativo do *Smartphone*;
- Integrar os dispositivos desenvolvidos;

1.5 PROCEDIMENTOS METODOLÓGICOS

O desenvolvimento deste trabalho envolve as seguintes etapas: pesquisa sobre os sistemas de automação naval existentes; definição dos comandos e informações que serão monitoradas; comunicação Bluetooth entre um *Smartphone* e uma placa eletrônica microcontrolada; desenvolvimento dos circuitos eletrônicos; desenvolvimento do aplicativo para ser instalado no *Smartphone*; desenvolvimento do *firmware* do microcontrolador e integração de todos os sistemas.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 AUTOMAÇÃO NAVAL

“A automação aplicada na indústria naval está presente em dois níveis: na construção e na operação do navio. Tem como objetivos principais: minimizar o esforço humano, aumentar a qualidade, diminuir custos e aumentar a segurança e a comodidade. ” (VIDAL, 2009, p19).

A automação naval teve seu início no Japão por volta da década de 60, e os fez possuir a maior e mais moderna indústria naval do mundo, mas a economia japonesa passava por um período de inflação elevada e os profissionais da indústria naval precisavam ser altamente qualificados, o que implicava em altos salários, tornando o custo de produção elevado, o que forçou a substituição da mão-de-obra por técnicas de construção automatizadas. Os estaleiros japoneses passaram a construir embarcações em módulos através do método de construção em blocos, permitindo uma boa redução no tempo de construção de um navio. Posteriormente, com a evolução da instrumentação e controle industrial, foram acrescentados esses dispositivos e técnicas na construção e operação das embarcações. (VIDAL, 2009).

A automação da operação em embarcações abrange casos de sistemas de navegação, gestão dos motores, controle e monitoração da carga, gerenciamento de energia, gerenciamento de potência e posicionamento dinâmico. (VIDAL, 2009).

2.2 MICROCONTROLADOR

Os primeiros microcontroladores surgiram em meados da década de 80, e ao contrário dos microprocessadores são dispositivos simples e pequenos (FÁBIO, 2007). Os microcontroladores incorporam em um único encapsulamento uma unidade central de processamento (CPU) e periféricos, tais como temporizadores, contadores, interfaces de comunicação serial, conversores analógico-digitais, moduladores por largura de pulso (PWM) e memória de programa do tipo Flash que armazena as instruções do firmware e que a CPU interpreta e executa.

A questão custo, tamanho reduzido e o baixo consumo de energia fizeram com que eles fossem utilizados na maioria dos sistemas eletrônicos microcontrolados.

Sendo programáveis eles podem ser utilizados nas mais diversas aplicações de sistemas embarcados como em celulares, relógios, máquinas, etc. Em um estudo realizado pela Semico, mais de quatro bilhões de microcontroladores de oito bits foram vendidos em 2006, antecipando que o mercado de microcontroladores iria crescer em 36.5% em 2010 e 12% em 2011. (SEMICO, 2011).

O microcontrolador escolhido para o desenvolvimento do protótipo foi o ATmega2560 fabricado pela empresa Atmel. Este é o microcontrolador presente na placa de desenvolvimento conhecida como Arduino Mega utilizada no projeto. Esta placa de desenvolvimento do ATmega2560 conta com uma memória de programa tipo Flash de 256 *kbytes*, SRAM de 8 *kbytes* e ainda, EEPROM de 4 *kbytes*, além de um *clock* da placa Arduino operando em 16 MHz. O ATmega2560 possui multiplicador de *clock* por hardware e diversos periféricos, tais como 4 canais de comunicação serial assíncrona (USART), 16 entradas de conversão analógico digitais e 15 saídas PWM.

2.3 BLUETOOTH

Criado para interligar periféricos próximos sem a utilização de cabos, a interface Bluetooth é uma rede de curta distância de baixo consumo elétrico desenvolvida em 1999 com o propósito de ser usada em dispositivos pequenos que não comportassem uma interface *wireless* (MORIMOTO, 2008).

Contudo, os estudos que levaram a criação do Bluetooth foram iniciados em 1994 quando a *Ericsson Mobile Communications* começou a estudar uma forma alternativa para os cabos que conectavam os celulares aos seus acessórios. Ultimamente os esforços levaram à criação do Bluetooth *Special Interest Group* (SIG) em fevereiro de 1998, por um grupo de empresas que trabalharam juntas para promover e definir as especificações do Bluetooth:

- *Ericsson Mobile Communications AB.*
- *Intel Corporation.*
- *IBM Corporation.*
- *Toshiba Corporation.*
- *Nokia Mobile Phones.*

Em maio de 1998, essas empresas anunciaram publicamente a criação da SIG e convidaram outras empresas para participar, em troca elas ajudariam e apoiariam a criação das especificações do Bluetooth.

Por ter uma velocidade de alcance baixos, o Bluetooth acaba por se tornar pouco utilizado em redes, sendo suficiente para comunicação entre dois dispositivos na transferência de pequenos arquivos ou pacotes de dados.

O Bluetooth é uma tecnologia criada para funcionar no mundo todo, razão pela qual se fez necessária a adoção de uma frequência de rádio aberta e aceita em praticamente qualquer lugar do planeta. A faixa *ISM* (*Industrial, Scientific, Medical*), que opera à frequência de 2,45 GHz, é a que me mais se aproxima desta necessidade, sendo utilizada em vários países, com subbandas de 1MHz entre 2,4 GHz a 2,5 GHz. De acordo com RUFINO (2007), por operar na mesma faixa de frequência do *WiFi*, O Bluetooth também está sujeito aos mesmos problemas de interferência desta faixa. A tecnologia de propagação, contudo, apesar de usar a mesma faixa, usa modulação diferente, sendo a FHSS para o Bluetooth e a DSSS para o WiFi.

Pelo fato de transmitir informações a elementos fisicamente próximos um do outro, pelo seu baixo consumo de energia e ainda devido ao fato de ser uma tecnologia vastamente disponível em *Smartphones*, optou-se pelo uso do Bluetooth como solução para este projeto.

2.4 SENSORES

A medição de grandezas físicas é feita por elementos primários de controle. Estes elementos, chamados de sensores ou transdutores, têm a função de medir uma grandeza e convertê-la em um sinal para ser utilizada como controle. Sensor é a denominação usada quando o sinal for diretamente proporcional à grandeza medida. Transdutores é a denominação usada quando este produz um sinal inversamente proporcional à grandeza medida (VIDAL, 2009, p15).

2.5 SMARTPHONE

Smartphone pode ser definido basicamente como uma combinação de aparelho celular junto com assistente pessoal, como os antigos *Palms* e os PDAs que,

através de conexões 3G ou Wi-Fi, permitem uma enorme variedade de recursos (MORIMOTO, 2009).

A possibilidade de instalar aplicativos adicionais, permitindo que um único dispositivo execute diversas outras funções, torna-o uma ferramenta de recursos poderosos. Um simples equipamento que pode ser carregado no bolso, com acesso contínuo à internet e que possui uma grande quantidade de aplicativos extras, faz com que o *Smartphone* seja cada vez mais indispensável nos dias de hoje (MORIMOTO, 2009).

Aparelhos que combinavam telefonia e processamento computacional foram conceituados por Theodore G. Paraskevakos em 1971, e disponibilizados para comercialização em 1993. De acordo com uma pesquisa da Ericsson, O volume de *Smartphones* ativos no mundo deve somar 1,1 bilhão em 2012 e triplicar em 2018. (GLOBO, 2012).

Devido à necessidade cada vez maior de transmitir dados e informação de forma quase instantânea, fácil e em uma plataforma móvel, os *Smartphones* acabaram se tornando uma necessidade para pessoas que precisam acessar dados de forma frequente, desde escrever e enviar e-mails até acessar câmeras de segurança de sua casa.

2.6 SISTEMA OPERACIONAL ANDROID

O mercado de celulares tem crescido cada vez mais nos dias atuais, e junto com esse crescimento vem a procura de dispositivos com mais funcionalidades para facilitar a vida das pessoas. Músicas, câmeras de vídeo, comunicação Bluetooth, jogos, internet, GPS e até mesmo TV são as principais características que chamam a atenção do usuário ao escolher um aparelho celular do tipo Smartphone.

Para acompanhar a evolução da tecnologia e satisfazer os usuários, a empresa Google, criou uma nova plataforma de desenvolvimento de aplicativos móveis, baseada em um sistema operacional Linux, que frisasse a modernidade e flexibilidade no desenvolvimento de aplicativos coorporativos, o Android (LECHETA, 2009).

Como os dispositivos baseados em Android são geralmente movidos a bateria, o Android foi projetado para gerenciar o uso da memória RAM de modo a manter o consumo de energia no mínimo, em comparação com os sistemas operacionais do

tipo desktop que são ligados à rede ilimitada de energia elétrica. Quando um aplicativo Android já não está em uso, o sistema pode suspendê-lo automaticamente da memória, mesmo estando o aplicativo ainda "aberto". Aplicativos suspensos não consomem recursos, como energia da bateria ou poder de processamento, aguardando em segundo plano até que sejam novamente requisitados. Isso traz um benefício duplo, aumentando a capacidade de resposta geral dos dispositivos baseados em Android, já que os aplicativos não precisam ser fechados e reabertos a partir do zero a cada vez que são requisitados, devido aos aplicativos de fundo não consumirem energia desnecessariamente.

Pelo Android ser uma plataforma livre e de código aberto, ou seja, que permite que cada fabricante possa realizar alterações no código-fonte para customizar seus produtos, além de ser gratuito, o aperfeiçoamento da ferramenta é muito facilitado, pois desenvolvedores do mundo todo podem contribuir adicionando funções ou até mesmo corrigindo falhas. Essa foi a principal razão pela escolha desse sistema operacional para desenvolver a interface de monitoramento.

2.7 APP INVENTOR

O MIT App Inventor é um ambiente de desenvolvimento que veio com uma proposta de democratizar a criação de aplicativos, pois a sua linguagem em blocos de construção visual do tipo arraste e solte, ou “*drag and drop*”, substitui os códigos baseados em texto, o que os torna muito mais simples e intuitivo para o desenvolvimento.

Através da página http://appinventor.mit.edu/ é possível acessar tutoriais, fóruns, assim como também desenvolver os aplicativos via web. É necessário antes ter-se cadastro no Google.

2.7.1 PÁGINA INICIAL

A figura 1 ilustra a página principal do ambiente de desenvolvimento Android App Inventor criado pelo MIT. Clicando-se no botão “*Create apps !*”, faz-se o direcionamento para a página de desenvolvimento de aplicativos.

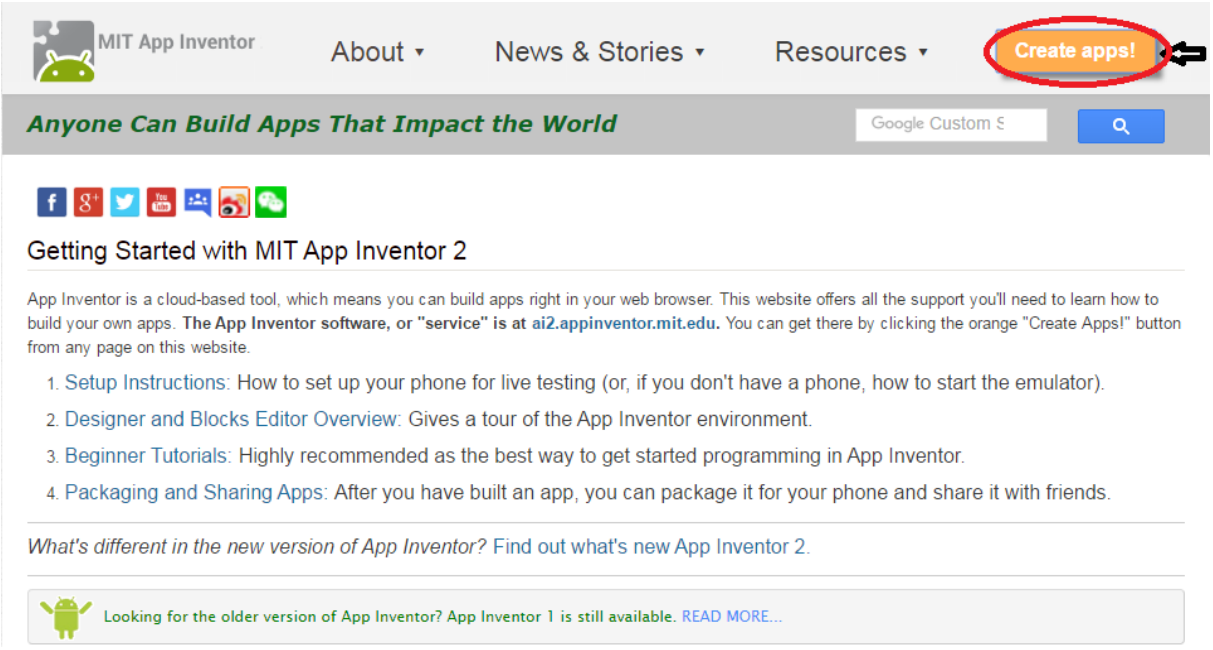


Figura 1–Página Inicial
Fonte: http://appinventor.mit.edu/explore/

2.7.2 DESENVOLVIMENTO EM WEB

A próxima página disponibilizada no MIT, uma vez tendo-se optado por criar um novo *App*, está ilustrada na figura 2. Alguns botões tornam-se disponíveis como os mostrados na coluna à esquerda da figura 2.

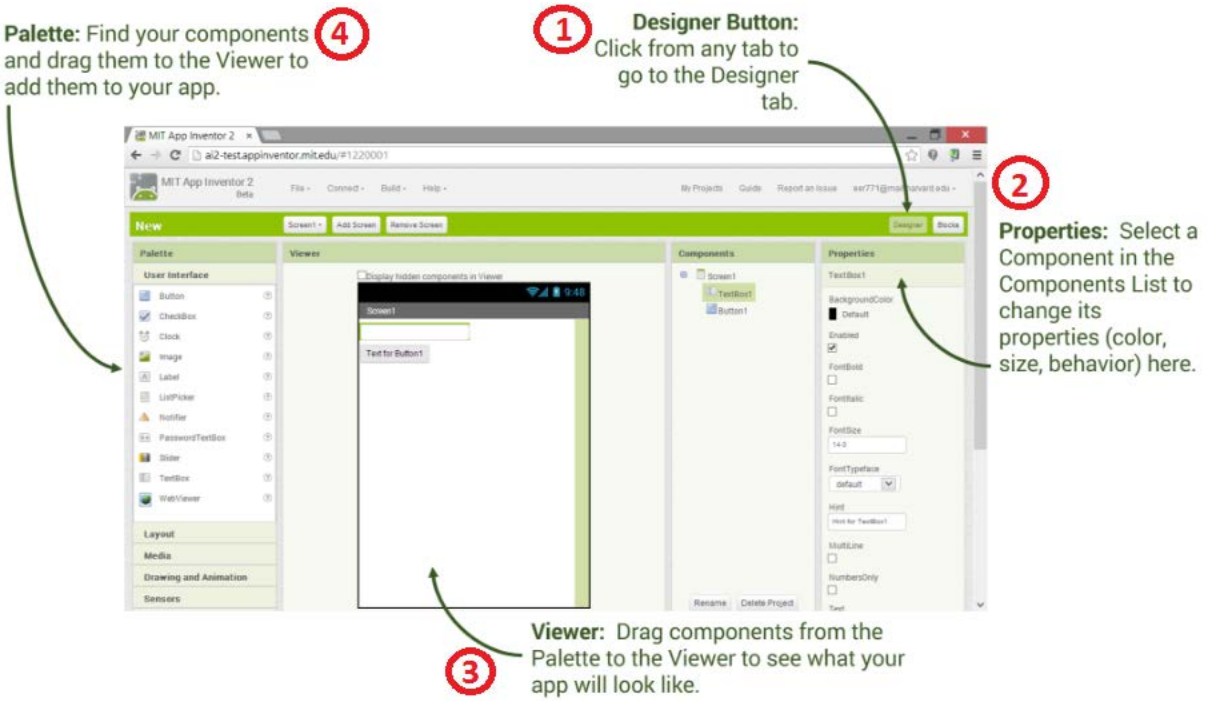


Figura 2– Web Desenvolvimento
Fonte: http://appinventor.mit.edu/ ?locale=en#6064039009189888/

Dentre as opções disponíveis, destacam-se:

1. *Designer Button*: Clicando no botão “*Designer*” é direcionado para a janela de programação.
2. *Properties*: Na barra “*Properties*” é possível selecionar e customizar propriedades de componentes como cor, tamanho, largura e etc.
3. *Viewer*: Os componentes são arrastados da janela “*Pallet*” para o “*Viewer*” na qual é feito o layout das telas do *App*.
4. *Pallet*: É onde estão localizados os componentes para diversas aplicativos no desenvolvimento do *App*.

2.7.3 TUTORIAIS

Uma próxima página, como a apresentada na figura 3, disponibiliza tutoriais que auxiliam no desenvolvimento dos aplicativos. Os padrões dos aplicativos vão do nível iniciante ao nível avançado. Em uma forma didática são apresentados vídeos de forma sequencial para auxiliar no desenvolvimento dos aplicativos.

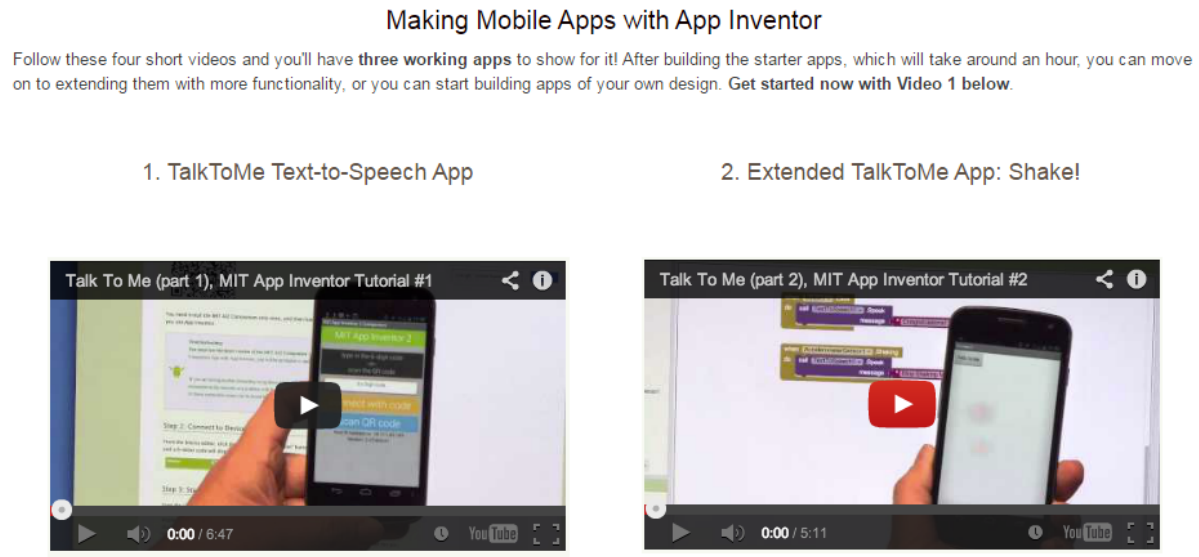


Figura 3–Tutoriais
Fonte: http://appinventor.mit.edu/ explore/ai2/tutorials.html

3 DESENVOLVIMENTO DO TEMA

Para desenvolver o sistema foi necessário então estudar o funcionamento da embarcação, assim como compreender a funcionalidade apresentada pelo painel de controle da embarcação. Neste painel são realizados os comandos para os circuitos de acionamentos da embarcação. Através dele são, também, monitoradas as grandezas relacionadas ao funcionamento destas. Um objetivo deste projeto é que todas as tarefas pudessem ser monitoradas a partir de um único ponto.

Nas embarcações a tripulação é obrigatoriamente formada por no mínimo duas pessoas. Uma delas é o Mestre e a outra é o marinheiro. É comum a pratica de revezamento no comando da embarcação entre Mestre e Marinheiro, estando o responsável obrigado a permanecer sempre na cabine.

O sistema foi pensando em minimizar falhas causadas por desatenção no monitoramento da embarcação em modo local, assim como também gerar uma possibilidade de monitoramento fora da cabine afim de aumentar a rapidez de diagnóstico de um eventual problema na embarcação. Foi desenvolvido um protótipo de um sistema de monitoramento remoto, o qual emula sinais do painel de controle para um *Smartphone*.

Este trabalho consiste em utilizar o circuito interno já existente na embarcação junto a uma placa dotada de um microcontrolador Arduino, comunicando com um *Smartphone* por um aplicativo Android através de uma comunicação Bluetooth.

Determinou-se que apenas o acionamento de lâmpadas poderia ser feito remotamente. O restante das informações seria apenas para fins de monitoramento e armazenamento das informações.

Pode-se monitorar velocidade, temperatura do motor, pressão de óleo, nível de combustível. Pode-se também verificar o estado de acionamento da iluminação interna da embarcação, e a de navegação. Pode-se registrar todos esses dados internamente em um cartão SD que vai estar integrado a placa, com a possibilidade de se fazer o *download* desses registros pelo próprio aplicativo Android.

3.1 HARDWARE

3.1.1 PLACA ARDUINO MEGA 2560

Durante o desenvolvimento do protótipo houve a necessidade de fazer a conexão do circuito com periféricos através de saídas e entradas digitais assim como também saídas analógicas a fim de simular os sensores e atuadores presentes em uma embarcação. Decidiu-se então por utilizar uma placa de desenvolvimento a fim de facilitar a prototipação do projeto, sendo que a placa escolhida foi a de um Arduino Mega 2560.

Como ilustrado na figura 4, a placa de desenvolvimento Arduino Mega 2560 consiste de uma plataforma eletrônica de código aberto que possui 54 entradas/ saídas digitais. Tem-se 15 pinos que podem ser usados como saídas PWM, 16 como entradas analógicas, 4 como UARTs, um *clock* de 16 MHz, uma conexão USB, uma entrada de alimentação externa, um botão de reset e alguns pinos que fornecem voltagens de operação de 5V e de 3.3V para os diferentes módulos que forem necessários no sistema.

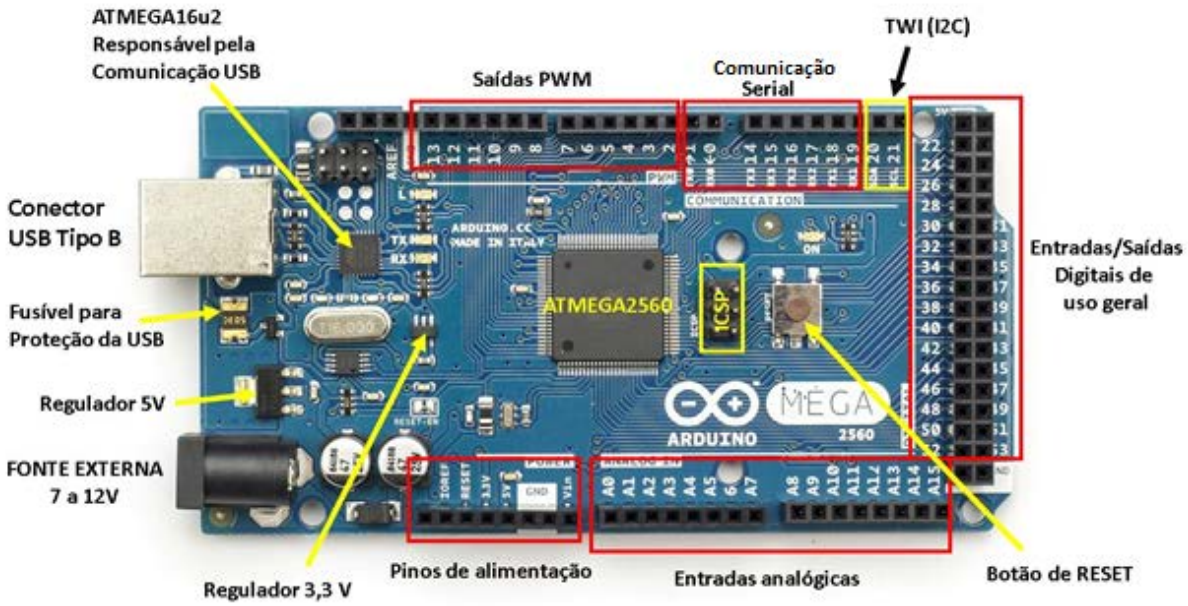


Figura 4– Arduino MEGA 2560
Fonte: www.embarcados.com.br

A alimentação da placa pode ser feita pela USB, ou por uma alimentação externa. Neste caso, faz-se uso de um conector do tipo *Jack* com positivo no centro.

O valor da tensão de alimentação deve estar entre os limites de 6V à 20V. Quando a alimentação é feita via USB, não é necessário estabilizar a tensão pelo regulador, já que o próprio circuito da placa apresenta componentes que protegem a porta USB do computador em caso de alguma anormalidade.

Um microcontrolador ATMEL ATMEGA16U2 realiza o interfaceamento entre essa placa e a interface USB do computador, possibilitando o *upload* do código binário gerado pelo compilador para o Arduíno.

3.1.2 MÓDULO BLUETOOTH

Para realizar a comunicação Bluetooth do protótipo com o sistema Android, utilizou-se o módulo de comunicação *wireless* HC-06 como ilustrado na figura 5. O alcance do módulo segue o padrão de comunicação Bluetooth, que é de aproximadamente 10 metros. Esse módulo funciona apenas em modo *slave*, ou seja, ele permite apenas que outros dispositivos se conectem a ele.

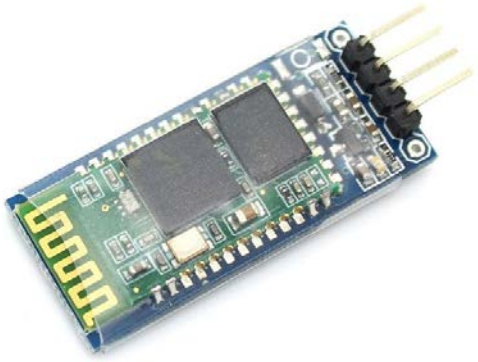


Figura 5– Módulo Bluetooth HC-06
Fonte: www.intorobotics.com

A partir do momento que o módulo estabelece a comunicação com o *Smartphone,* uma ponte bidirecional de dados se estabelece, podendo as informações serem enviadas do HC-06 para o *Smartphone* ou vice-versa. As informações podem tratar de quaisquer dados controlados pela placa, já mencionados anteriormente. Pode-se também se referir ao envio das informações contidas no cartão SD, integrado ao circuito, para o *Smartphone*.

O módulo HC-06 possui quatro pinos: GND e VCC para a alimentação de 3.3V, um pino para transmissão e outro para a recepção de dados provenientes dos pinos

de recepção e transmissão do Arduino, respectivamente. A figura 6 apresenta o esquema de conexões do módulo HC-06 à placa do Arduíno.

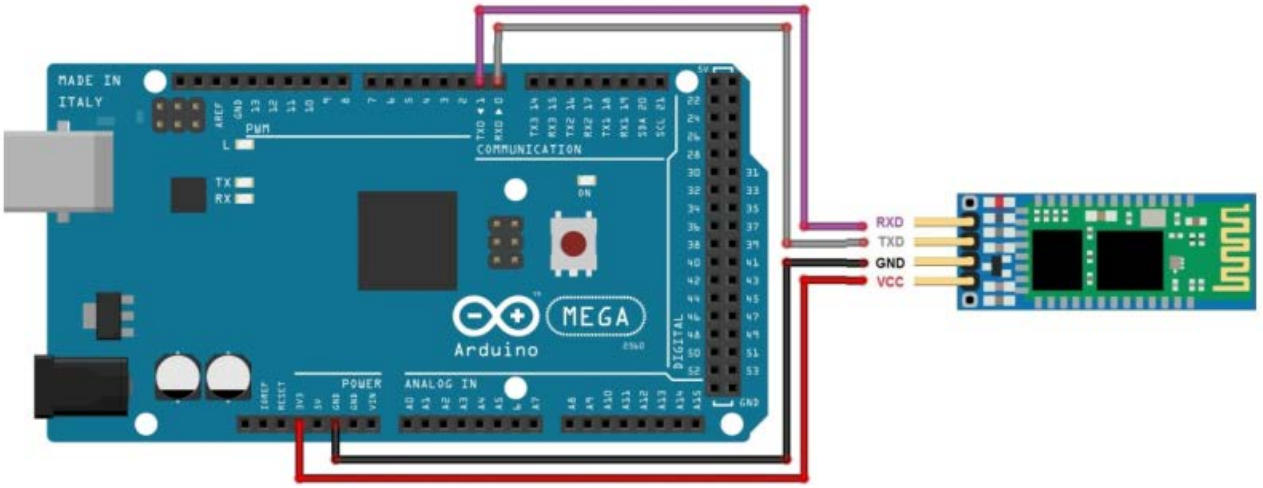


Figura 6– Conexão do módulo HC-06 ao Arduino MEGA 2560
Fonte: www.safaribooksonline.com

3.1.3 MÓDULO CARTÃO MICRO-SD

A fim de possibilitar uma análise completa dos dados fornecidos pelo sistema, houve a necessidade de se armazenar todos os eventos ocorridos durante o funcionamento da embarcação em um banco de dados. Incorporou-se ao circuito um módulo adaptador para cartão de memória Micro-SD fabricado pela empresa CATALEX, como ilustrado na figura 7. Tal recurso possibilitou agregar-se uma função similar ao de um diário de bordo ao protótipo.

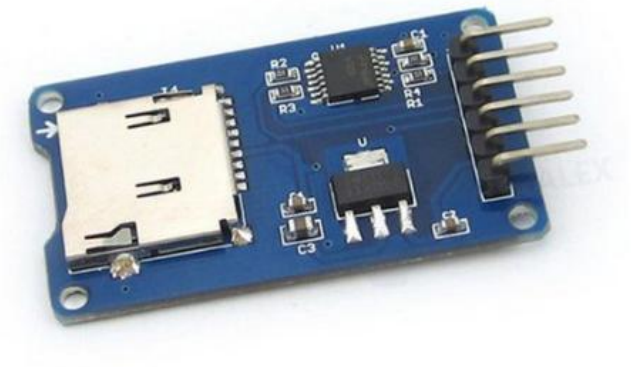


Figura 7– Módulo Adaptador Micro-SD
Fonte: www.blogsmayan.blogspot.com.br

Esse módulo possui 6 pinos. Um pino para VCC e outro de GND, responsáveis pela alimentação de 5V. Os demais quatro referem-se à comunicação com a interface periférica serial, SPI. Na figura 8 é possível ver as conexões desse módulo com o protótipo, pelos pinos 50 a 53. Pode-se também visualizar as conexões de alimentação do módulo.

Os pinos 50 a 53 tem as seguintes funções:

- Pino 50 - *MISO* (*Master In Slave Out*)
- Pino 51 - *MOSI* (*Master Out Slave In*)
- Pino 52 - *SCK* (*Serial Clock*)
- Pino 53 - *SS* (*Slave Select*)

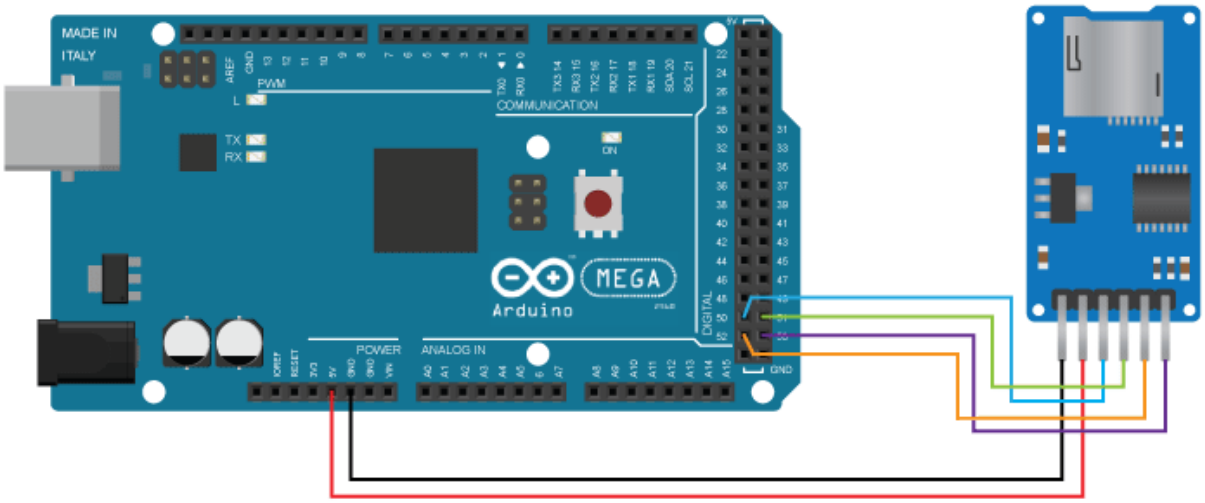


Figura 8– Conexão do módulo Adaptador Micro-SD ao Arduino MEGA 2560
Fonte: www.howtomechatronics.com

A partir do momento que o protótipo se integra à embarcação, inicia-se o registro de todos os dados recebidos juntamente ao cartão Micro-SD ao módulo.

Esses registros podem ser exportados para o aplicativo Android junto ao *Smartphone* via conexão Bluetooth. Existe também a possibilidade de se realizar o *download* desses dados em forma de planilha, facilitando a utilização deles para diversas funções, como na identificação de falhas, no registro de *start/stop* de máquinas, e na determinação da manutenção preventiva e corretiva.

3.1.4 MÓDULO RTC

O módulo de *real-time clock* (RTC) foi integrado ao circuito devido à necessidade de se registrar os eventos acompanhados da informação do exato

instante em que estes ocorrem, tornando deste modo os dados úteis à análise posterior do sistema.

O módulo utilizado foi o DS3231 apresentado na figura 9. Esse módulo trabalha com uma tensão de operação de 3.3V, apresentando a capacidade de contar o tempo em ano, mês, dia, hora, minuto e segundo. Ele é capaz de manter as informações internas por mais de um ano, independente de interferências externas.

Figura 9– Módulo DS3231 RTC
Fonte: www.potentiallabs.com

O módulo DS3231 utiliza o protocolo I2C de comunicação através dos pinos 21 (SCL) e 20 (SDA), sendo possível armazenar todo o volume de informação através desses 2 pinos. Na figura 10 apresentam-se as conexões entre o módulo DS3231 RTC e a placa de desenvolvimento do protótipo.

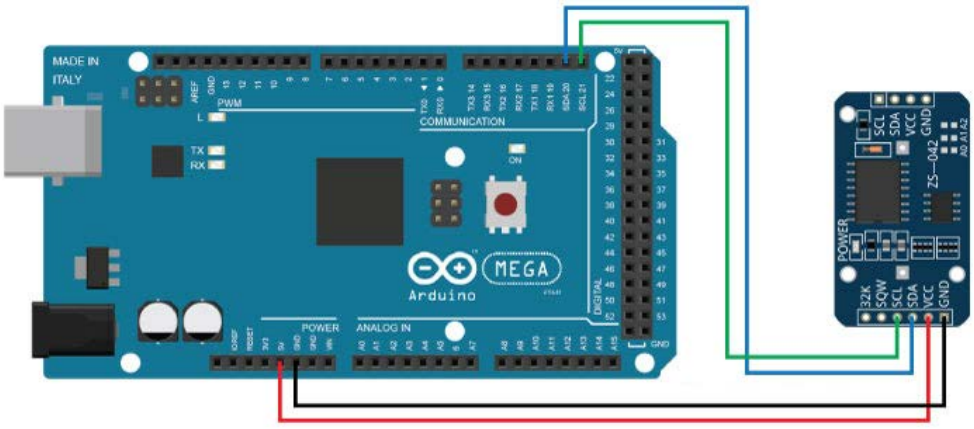


Figura 10– Conexão do DS3231 RTC ao Arduino MEGA 2560
Fonte: www.howtomechatronics.com

3.1.5 PLACA DE CONTROLE

Durante o desenvolvimento do projeto decidiu-se por utilizar uma placa padrão para desenvolvimento de protótipos, também conhecida como *protoboard*. O fato dessa plataforma não necessitar de soldas nas conexões entre os componentes, além de ser reutilizável, tornou-a como a escolhida no desenvolvimento de protótipos temporários além de se tornar popular entre os estudantes de tecnologia.

Como é possível ver na figura 11, pode-se montar o circuito de uma maneira bastante flexível, permitindo conectar os componentes sem utilização de solda, apenas conectando-os aos terminais da placa.

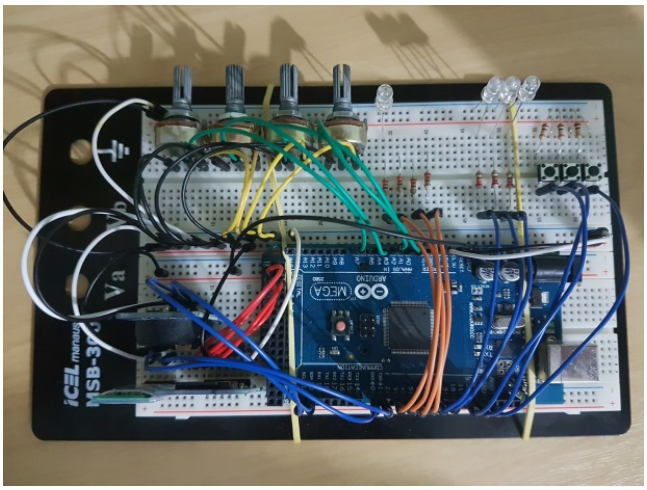


Figura 11– Placa de controle
Fonte: Autoria própria.

Os módulos adicionais necessários ao funcionamento do sistema também são de fácil conexão com o *protoboard,* como demonstrado na figura 12. A facilidade de conexão foi interessante durante o desenvolvimento do protótipo devido a necessidade de se testar diferentes módulos no sistema para se determinar quais melhores serviriam ao projeto. É, porém, importante ressaltar que sempre que foi compilado uma nova versão de programa para o microcontrolador, necessita-se desconectar o módulo Bluetooth HC-06 dos pinos TX e RX, transmissão e recepção, pelo fato desses serem os mesmos utilizados durante o *upload* de uma nova programação ao microcontrolador.

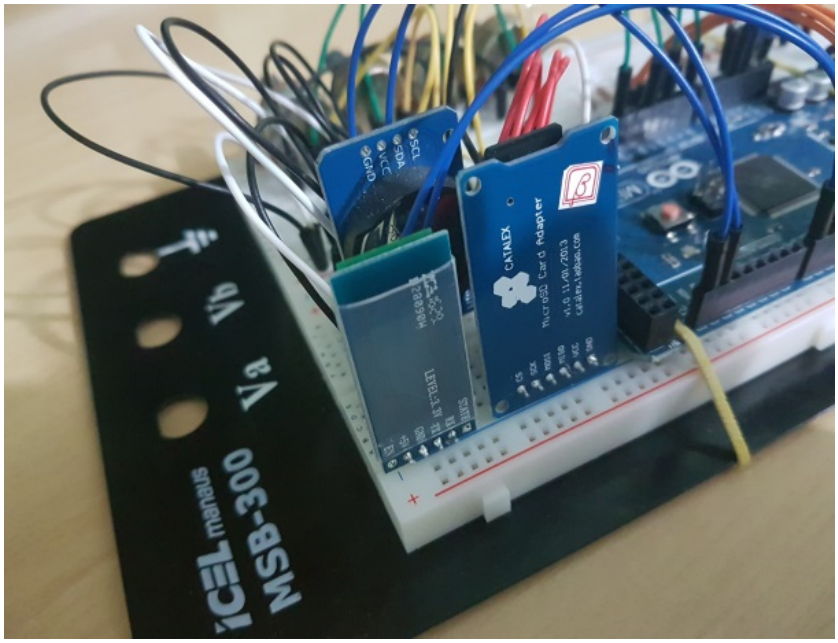


Figura 12– Placa de controle (módulos Bluetooth, Micro-SD & RTC)
Fonte: Autoria própria

A placa de controle do protótipo contém alguns botões, como ilustrado na figura 13, que foram conectados a resistores de 10k ohms, com a finalidade de se simular acionamentos manuais das lâmpadas da embarcação feitos a partir da cabine. Para demonstrar o estado das lâmpadas, adicionaram-se *LEDs* conectados a resistores de 220 ohms.

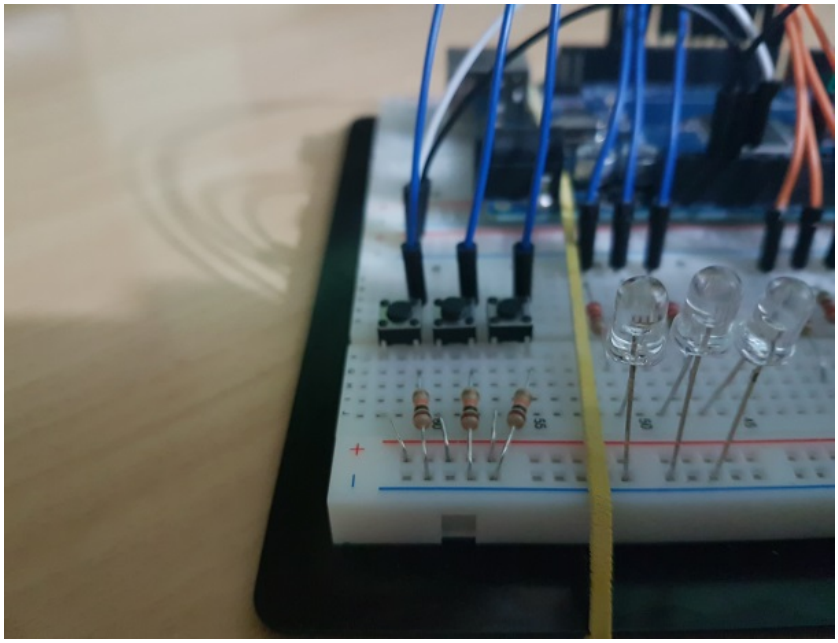


Figura 13 Placa de controle (acionamento das lâmpadas)
Fonte: Autoria própria

Na placa de controle utilizaram-se potenciômetros com a finalidade de simular as variações das grandezas velocidade, temperatura de motor, nível de combustível e pressão de óleo. Essas conexões estão apresentadas na figura 14. Pode-se também observar os *LEDs* simulando os alarmes que monitoram essas grandezas.

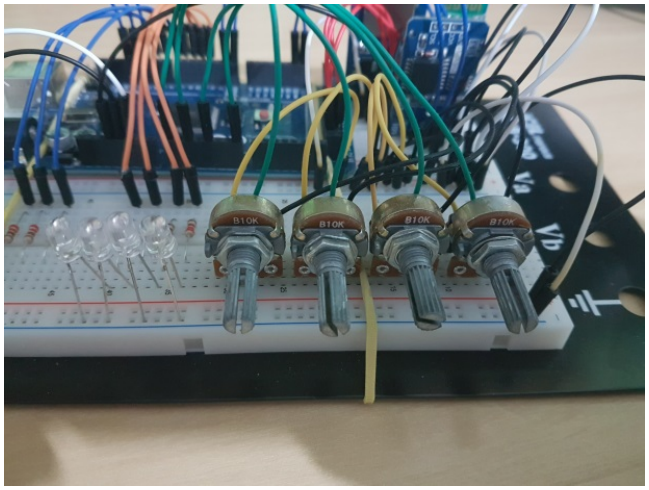


Figura 14– Placa de controle (entradas analógicas & alarmes)
Fonte: Autoria própria

3.2 SOFTWARE

3.2.1 PROGRAMAÇÃO ARDUINO

Devido à utilização da placa de desenvolvimento Arduino Mega 2560, realizou-se a programação do microcontrolador fazendo uso do próprio ambiente de desenvolvimento Arduino, como ilustra a figura 15. Essa plataforma contém todos os recursos de edição de *software*, e é compatível com todos os sistemas operacionais Windows além de também possibilitar o desenvolvimento da programação via *web*.

Figura 15– Plataforma de programação Arduino
Fonte: www.arduino.cc/en/Main/Software

A linguagem de programação utilizada nessa plataforma é basicamente C++, com pequenas variações. A plataforma é intuitiva ao usuário e contém vários exemplos de linhas de código disponíveis.

Nessa plataforma, existe uma função que mostra as informações da interface serial sendo transmitidas em tempo real, facilitando deste modo os testes feitos durante o desenvolvimento do projeto, conforme figura 16.

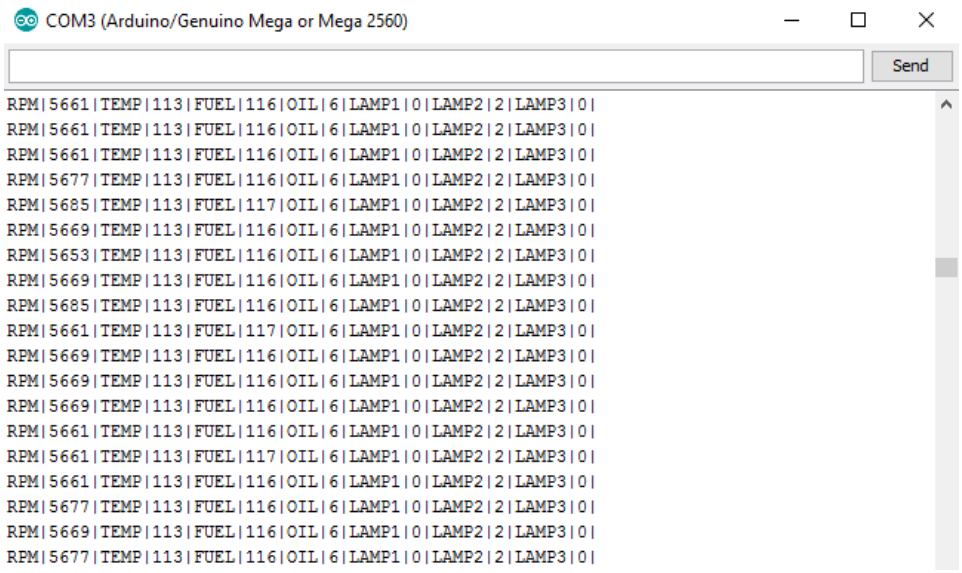


Figura 16– Serial durante comunicação com *App*
Fonte: Autoria própria

A programação foi concebida em etapas de modo a facilitar o desenvolvimento do programa Arduino, como também avançar em conjunto com o desenvolvimento do *hardware*, e também da programação do aplicativo *Smartphone*.

A primeira etapa consistiu em tratar os dados recebidos dos periféricos, como os *push-buttons,* além de se transmitir as informações para as saídas com *LEDs*. Durante essa etapa desenvolveu-se o programa em linhas de código com o objetivo de traduzir os valores analógicos recebidos pelos potenciômetros para valores que pudessem ser lidos dentro de uma escala utilizável em uma embarcação. No quadro I pode-se visualizar a escala utilizada, assim como pode-se também visualizar quais são os valores escolhidos para acionar seus respectivos alarmes:

Sensor	Min	Max	Alarme estágio 1	Alarme estágio 2
Velocidade Angular (rpm)	0	8000	>6000	>7000
Temperatura Motor (°C)	0	150	>110	>130
Nível de Combustivel (l)	0	200	<50	<25
Pressão Óleo (bar)	0	9	>7	>8

Quadro 1– Acionamento alarmes
Fonte: Autoria própria

Como apresentado na figura 17, observa-se as linhas de código que traduzem os valores recebidos pelos potenciômetros para valores que podem ser trabalhados dentro da escala.

```
//Reading potentiometers & push-buttons
vaPin1 = analogRead(aPin1);
vaPin2 = analogRead(aPin2);
vaPin3 = analogRead(aPin3);
vaPin4 = analogRead(aPin4);
pbState1 = digitalRead(pbPin1);
pbState2 = digitalRead(pbPin2);
pbState3 = digitalRead(pbPin3);

//Scaling information from inputs
rpm = map(vaPin1, 0, 1023, 0, 8000); //0 to 8000 rpm
temp = map(vaPin2, 0, 1023, 0, 150); //0 to 150 celcius
fuel = map(vaPin3, 0, 1023, 0, 200); //0 to 200 liters
oil = map(vaPin4, 0, 1023, 0, 9);    //0 to 9 bar
```

Figura 17– Programação Arduino (utilização de strings)
Fonte: Autoria própria

Na segunda etapa tratou-se de pesquisar a melhor forma de se tratar as informações enviadas pelo microcontrolador via interface serial. Nesse momento ainda não houve nenhuma integração com os outros módulos.

Uma das formas encontradas durante a pesquisa foi a utilização do comando "*string()".* Como destacado na figura 18 esse comando possibilita organizar, em apenas uma linha, informações diferentes para a serial, como por exemplo, o tempo da amostra, a velocidade, o estado de alarme, etc.

```
//Write Log File Header
File logFile = SD.open("LOG.csv", FILE_WRITE);
if(logFile)
{
  //logFile.println(", , , , , , , , , ,"); //Blank lines, incase there was previous data
  String header = "ID, DATE, TIME, RPM, TEMP, FUEL, OIL, A RPM, A TEMP, A FUEL, A OIL";
  logFile.println(header);
  logFile.close();
  //Serial.println(header);
}
else
{
  Serial.println("Couldn't open log file");
}
```

Figura 18– Programação Arduino (utilização de strings)
Fonte: Autoria própria

Com a segunda etapa concluída, iniciou-se então a integração dos diferentes módulos à placa de controle. Em seguida fez-se as alterações do código de programação Arduino. Inicialmente realizou-se a integração dos módulos de cartão Micro-SD e do módulo RTC DS3231.

Um dos desafios nessa etapa foi a de projetar as informações a serem coletadas pelo microcontrolador pela interface serial, a cada 1000ms. Importante observar que esses valores serão armazenados no cartão Micro-SD no exato momento em que ocorrerem. Portanto, o código não pode conter nenhum comando de atraso, como "*delay*()", devido ao fato de tal comando atrasar o momento em que as amostras são transmitidas à serial.

A solução encontrada envolveu a utilização de uma variável interna do próprio microcontrolador, chamada de "millis()", que nada mais é do que o número de milissegundos decorridos desde o início do programa. Como destacado na figura 19, a solução consiste em se ter uma variável contendo o tempo atual obtida por "*currentMillis*", que é constantemente alimentada por "millis". Uma variável chamada de "*stampLog*" carrega o valor inicial de 0ms e uma terceira variável com o nome de "*intervalLog*" contém o valor de 1000ms. Portanto, a cada segundo decorrido, a variável "*stampLog*" obtém o valor da variável "*millis*" e o código vai enviar os novos valores à serial apenas após 1 segundo decorridos do último envio de informações.

```
unsigned long currentMillis = millis();

//Printing Log information every 1000 milliseconds while Android is not downloading the information
if((currentMillis - stampLog > intervalLog) && (download == 0))
{
  stampLog = currentMillis;
```

Figura 19– Programação Arduino (utilização de millis)
Fonte: Autoria própria

Na integração do módulo Micro-SD necessitou-se considerar que o arquivo a ser criado para registrar as informações dos sensores deve armazenar tais informações em uma forma de fácil visualização e utilização.

A extensão criada, portanto, foi de "*LOG.csv*", sendo essa extensão tendo o significado de *comma separated values,* formato interessante por já armazenar os dados separados em colunas, assim ideal para geração de gráficos de informação. A figura 20 apresenta as linhas de código no momento em que tal arquivo é criado dentro do cartão Micro-SD.

```
//SD card setup
Serial.println("Initializing Card");

//CS Pin is an output
pinMode(CS_pin, OUTPUT);

//Check if card ready
if(!SD.begin(CS_pin))
{
  Serial.println("Card Failed");
  return;
}
Serial.println("Card Ready");

//Write Log File Header
File logFile = SD.open("LOG.csv", FILE_WRITE);
if(logFile)
{
  //logFile.println(", , , , , , , , , ,"); //Blank lines, incase there was previous data
  String header = "ID, DATE, TIME, RPM, TEMP, FUEL, OIL, A RPM, A TEMP, A FUEL, A OIL";
  logFile.println(header);
  logFile.close();
  //Serial.println(header);
}
else
{
  Serial.println("Couldn't open log file");
}
```

Figura 20– Programação Arduino (LOG.csv)
Fonte: Autoria própria

Por último adicionou-se ao código as linhas responsáveis pela interação com o módulo de Bluetooth que se comunica com o aplicativo Android. Primeiramente necessitou-se estabelecer um protocolo de comunicação, ou seja, uma convenção que possibilitasse uma conexão com transferência de dados entre dois sistemas computacionais. No quadro II abaixo os comandos numéricos são enviados pelo aplicativo Android e armazenados no microcontrolador na variável chamada de "*state*".

Botão *App* Android	Comando	Função para microcontrolador executar
Luz Cabine	1	Alterar estado da lâmpada 1
Luz Proa	2	Alterar estado da lâmpada 2
Luz Convés	3	Alterar estado da lâmpada 3
Connect	4	Enviar dados das variáveis a porta serial
Download	5	Enviar dados de LOG.csv ao cartão Micro-SD
Disconnect	6	Parar de enviar dados a serial

Quadro 2 - Protocolo de comunicação Bluetooth
Fonte: Autoria própria

O microcontrolador inicia o envio das informações à serial a partir do momento em que o aplicativo Android conecta-se ao módulo Bluetooth integrado à placa quando o comando "4" é recebido, como ilustrado na figura 21.

```
//When Android is connected send sensor information to serial
if(state == '4')
{
  printSerial = 1;
  state = 0;
}
```

Figura 21– Programação Arduino (*print serial*)
Fonte: Autoria própria

Observa-se na figura 22 a parte do código de programação que corresponde a atuação do usuário do aplicativo no acionamento das lâmpadas:

```
//Android information to change lamps On/Off
//Push-Button 1 - Lamp1
if((state == '1') && (flag1 == 0))
{
  digitalWrite(ledPin5,HIGH);
  flag1 = 1;
  state = 0;
}
else if((state == '1') && (flag1 == 2))
{
  digitalWrite(ledPin5,LOW);
  flag1 = 0;
  state = 0;
}
//Push-Button 2 - Lamp2
if((state == '2') && (flag2 == 0))
{
  digitalWrite(ledPin6,HIGH);
  flag2 = 1;
  state = 0;
}
else if((state == '2') && (flag2 == 2))
{
  digitalWrite(ledPin6,LOW);
  flag2 = 0;
  state = 0;
}
//Push-Button 3 - Lamp3
if((state == '3') && (flag3 == 0))
{
  digitalWrite(ledPin7,HIGH);
  flag3 = 1;
  state = 0;
}
else if((state == '3') && (flag3 == 2))
{
  digitalWrite(ledPin7,LOW);
  flag3 = 0;
  state = 0;
}
```

Figura 22– Programação Arduino (*acionamento lâmpadas pelo App*)
Fonte: Autoria própria

Quando o usuário do aplicativo requisita as informações contidas no cartão Micro-SD, o envio das informações dos sensores à serial é parado e iniciam-se as linhas de códigos apresentadas na figura 23, responsáveis pela leitura dos dados do cartão Micro-SD. Deste modo enviam-se os dados para a serial, oposto das informações contidas nos periféricos. Ao se finalizar o *download* do arquivo pelo aplicativo, as informações dos sensores voltam a ser transmitidas à serial.

```
//Sending "LOG.csv" information to Android via Bluetooth
if(state == '5')
{
  printSerial = 0;
  download = 1;
  delay(1000);

  Serial.println("");
  Serial.println("Discard information above");
  Serial.println("Dumping information from LOG.csv now:");
  Serial.println("");

  File logFile = SD.open("LOG.csv");
    if(logFile)
    {
      while (logFile.available())
    {
      Serial.write(logFile.read());
    }
      logFile.close();
      SD.remove("LOG.csv");

      Serial.println("");
      Serial.println("LOG.csv reseted inside SD card");
      Serial.println("");

      needHeader = 1;
    }

    //Re-create LOG.csv in SD card and add header
    if(needHeader = 1)
    {
      File logFile = SD.open("LOG.csv", FILE_WRITE);
      if(logFile)
      {
        String header = "ID, DATE, TIME, RPM, TEMP, FUEL, OIL, A RPM, A TEMP, A FUEL, A OIL";
        logFile.println(header);
        logFile.close();
        needHeader = 0;
      }
    }
  delay(1000);
  state = 0;
  download = 0;
  printSerial = 1;
}
```

Figura 23– Programação Arduino (download de dados LOG.csv)
Fonte: Autoria própria

3.2.2 APLICATIVO ANDROID

O aplicativo Android foi desenvolvido utilizando a plataforma MIT *App Inventor 2*, uma plataforma gratuita e inovadora para iniciantes de programação e criação de aplicativos que transforma a linguagem complexa de codificação baseada em texto em blocos de construção visuais de fácil entendimento. Com mais de 400 mil usuários ativos a cada mês em 195 países diferentes que já desenvolveram por volta de 22 milhões de aplicativos utilizando essa plataforma, *MIT App Inventor* ajuda a revolucionar a maneira como os aplicativos são desenvolvidos.

A programação e a interface do aplicativo é feita pelo próprio *browser* sendo possível acompanhar o progresso do mesmo, passo-a-passo, pelo próprio *Smartphone* fazendo o *download* do aplicativo MIT *App Inventor* 2 no *Smartphone*, ou por um emulador visual no próprio *brownser*.

No primeiro momento foi desenvolvido apenas o *layout* do aplicativo levando em consideração quais funções seriam necessárias disponibilizar ao usuário como ilustrado na figura 24.

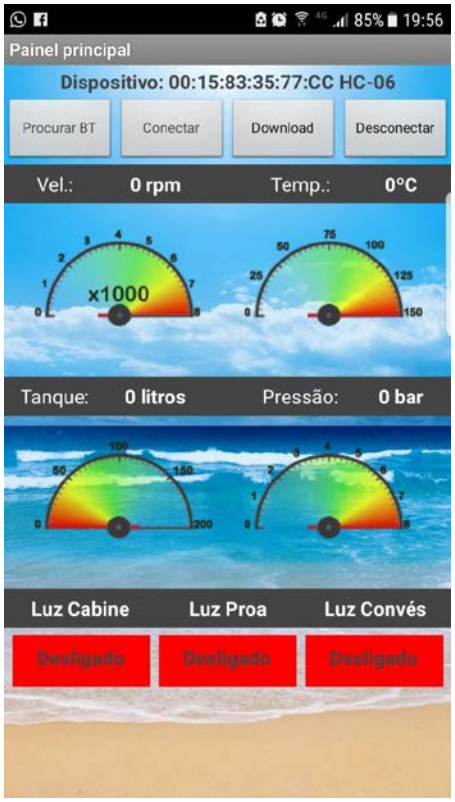


Figura 24– Layout do aplicativo *Android*
Fonte: Autoria própria

Durante o desenvolvimento do *layout* decidiu-se também de qual forma os dados recebidos da comunicação Bluetooth seriam mostrados ao usuário. Utilizaram-se duas formas de mostrar as informações da embarcação. Uma forma é a digital onde a cor do valor varia de acordo com o estágio do alarme, e outra de forma analógica, simulando o painel de uma embarcação.

Após determinar qual seria o *layout* do aplicativo, iniciou-se então a programação de blocos levando em consideração o funcionamento interno do aplicativo como também o tratamento das informações que vão ser trocadas através da comunicação Bluetooth.

A programação em blocos do projeto é basicamente dividida em uma parte referente a iniciação do aplicativo, e outra parte pela função de cada botão. Os blocos de programação referentes à iniciação do aplicativo podem ser vistos na figura 25. A função tem por objetivo apenas mostrar ao usuário o nome dos *Smartphones* que já se conectaram com o aplicativo anteriormente, buscando-os em um banco de dados interno gerado pelo aplicativo chamado de "*TinyDB1*".

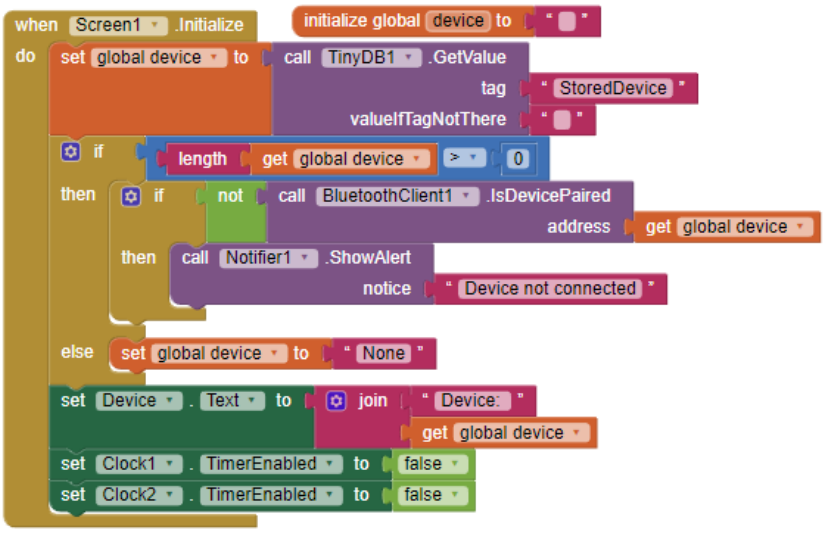


Figura 25– Desenvolvimento blocos inicialização
Fonte: Autoria própria

Quando o usuário pressiona o botão "*BT Search*", uma nova função é disparada, responsável pela busca por dispositivos Bluetooth nas proximidades. Cada dispositivo descoberto é alertado sobre isto e questionado, quando ainda não pareado, se deseja se conectar com o aplicativo no *Smartphone*.

Estando o protótipo dentro do alcance do *Smartphone*, e estando o *hardware* Bluetooth do *Smartphone* ligado, *a* opção de se conectar com o módulo HC-06 aparecerá na tela. Este processo está ilustrado na figura 26.

Figura 26– Desenvolvimento blocos BT search
Fonte: Autoria própria

A função "*Connect*" inicia-se logo após o usuário ter escolhido o módulo HC-06, pressionando o botão para realizar a conexão. Deste modo inicia-se a rotina chamada "*Clock1*". O estabelecimento da conexão é confirmado ao usuário através de mensagem na tela do smarphone e por vibração do mesmo. Este processo é apresentado na figura 27.

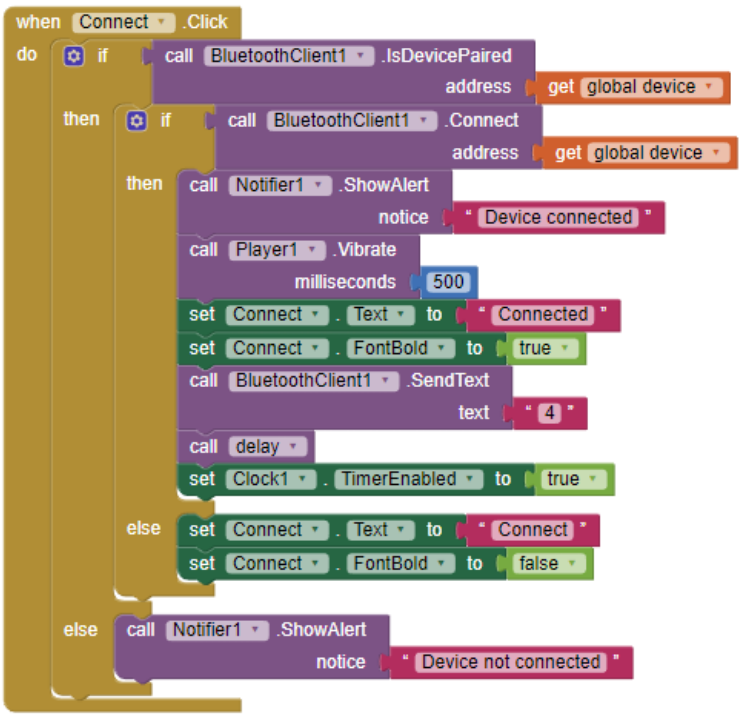


Figura 27– Desenvolvimento blocos 'Connect'
Fonte: Autoria própria

Com a conexão Bluetooth estabelecida, o aplicativo imediatamente começa a receber as informações provenientes da serial da placa de controle. Conforme descrito anteriormente, as informações recebidas estão em forma de texto, chamadas de *strings*.

As *strings* contém as informações dos periféricos em um determinado momento de tempo, condensadas em uma linha, como observou-se na figura 16. Portanto desenvolveu-se o aplicativo com a finalidade de separar as informações dessas *strings* em porções menores. Por isto, fez-se uso do caractere "|" no envio dos dados à serial como um caractere de separação de informações. Assim, pode-se distinguir cada trecho de informação, como ilustrado na figura 28. Após a separação das informações, elas são armazenadas em um outro banco de dados.

Na figura pode-se também observar a lógica na programação de blocos a respeito das informações passadas ao usuário em relação ao nível dos alarmes e seus respectivos valores.

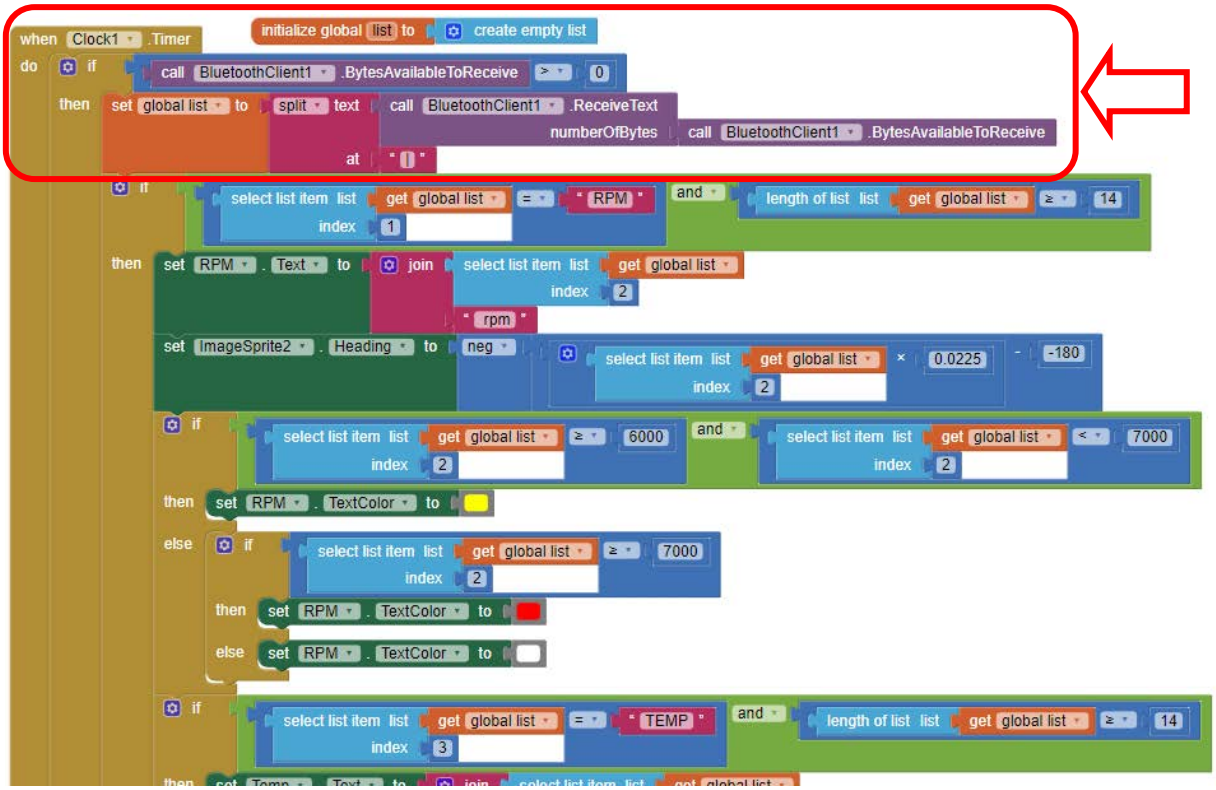


Figura 28– Desenvolvimento blocos 'BT Search'
Fonte: Autoria própria

A função de "*Download*" do registro no cartão Micro-SD se tornou a mais complexa a ser desenvolvida dentro do aplicativo. Como demonstrado na figura 29, a

função primeiramente apaga o arquivo "*LOG.csv*" no local previamente estabelecido na memória do *Smartphone,* caso ele já exista. Em seguida o sinal de *download* é enviado ao microcontrolador para que se inicie a leitura dos dados contidos no arquivo do cartão Micro-SD da placa de controle e os envie à serial. Deste modo o aplicativo inicia uma segunda rotina chamada de "*Clock2*" com o objetivo de registrar os dados recebidos pela comunicação Bluetooth, assim como também cria um novo arquivo chamado de "*Log.csv"* no *Smartphone* que registra todas as informações provenientes da serial a esse novo arquivo.

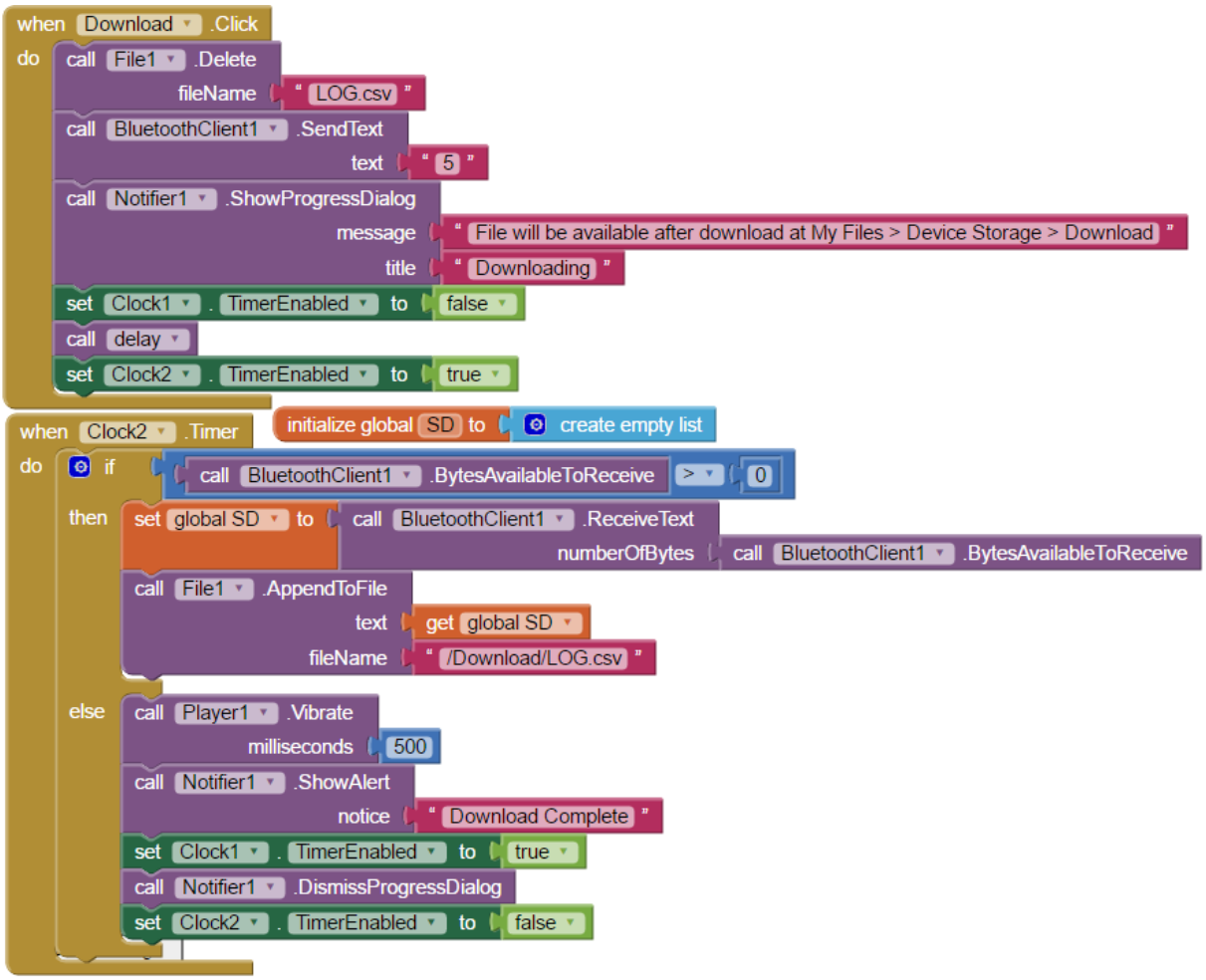


Figura 29– Desenvolvimento blocos 'Download'
Fonte: Autoria própria

O acionamento ou o desligamento das lâmpadas através do aplicativo é realizado pelo envio do respectivo sinal ao microcontrolador. De outro lado, a função "*Disconnect*" tem como objetivo fazer o *reset* de todas as variáveis do aplicativo, basicamente deixando-o em um estado de desconexão do *hardware* Bluetooth. Ambas estas funções estão apresentadas na figura 30 abaixo:

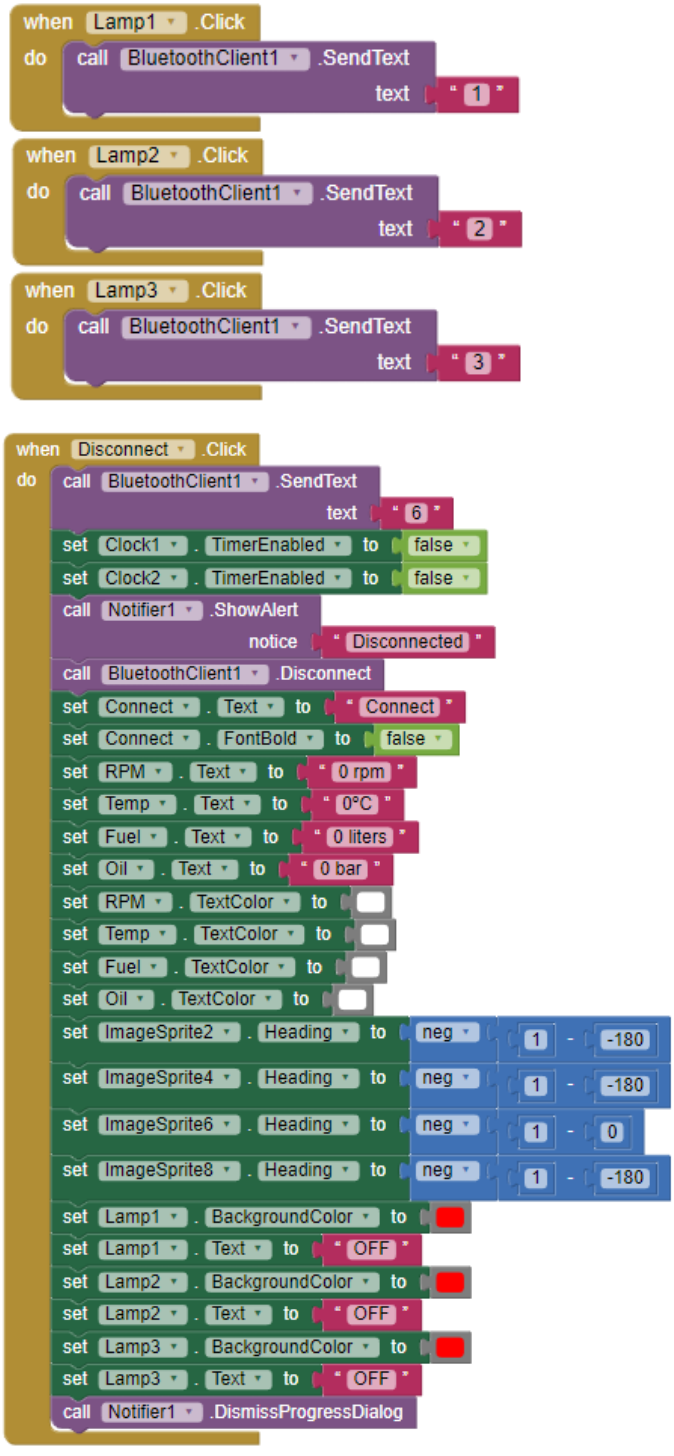


Figura 30– Desenvolvimento blocos 'Lamps & Disconnect'
Fonte: Autoria própria

A instalação do aplicativo em um *Smartphone* pode ser feita de duas formas. Uma delas é através de um arquivo com extensão "*apk*" que deve ser salvo em um computador para então passá-lo ao *Smartphone*. A segunda maneira, escolhida no

desenvolvimento deste trabalho, é de gerar um "*quick response code"*, mais conhecido por código QR, ou *QRcode*.

O primeiro passo é gerar um *QRcode*, clicando-se em "*Built"*. Para isto, basta selecionar a primeira opção que está indicada em vermelho na figura 31.

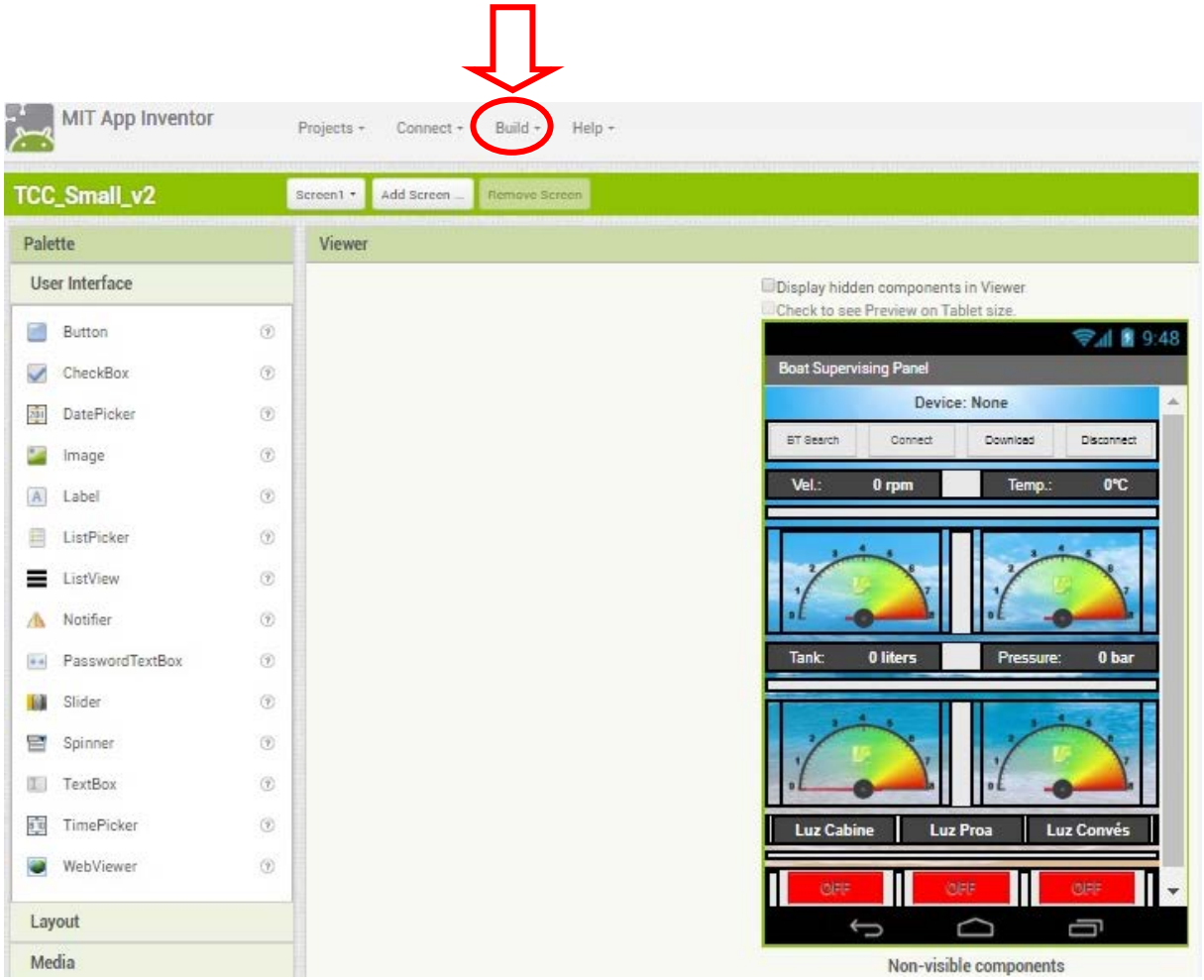


Figura 31– Built QR code
Fonte: Autoria Própria

Após gerar o código QR, como é possível ver na figura 32, necessita-se abrir o Aplicativo do MIT no *Smartphone* de modo a realizar-se a leitura do código e posteriormente a instalação do aplicativo projeto.

Com o aplicativo instalado inicia-se a conexão Bluetooth com a placa de controle, assim como as demais operações.

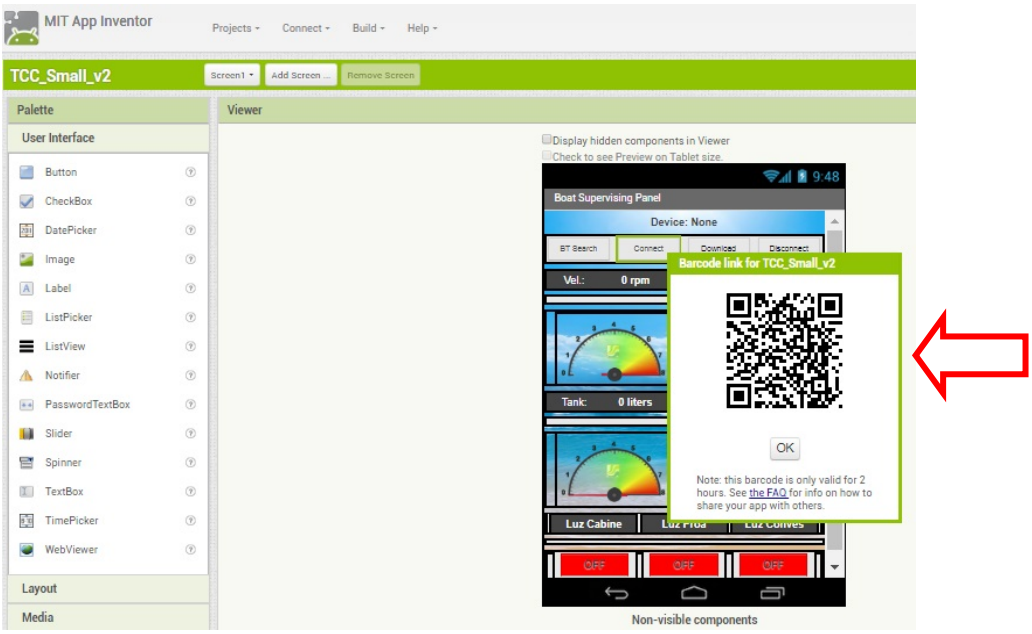


Figura 32– QR code
Fonte: Autoria Própria

Será necessário antes de tudo instalar o aplicativo "*MIT AI2 Companion*'' indicado em vermelho na figura 33. Através dele é que se faz a leitura do código QR e a instalação do aplicativo projeto.

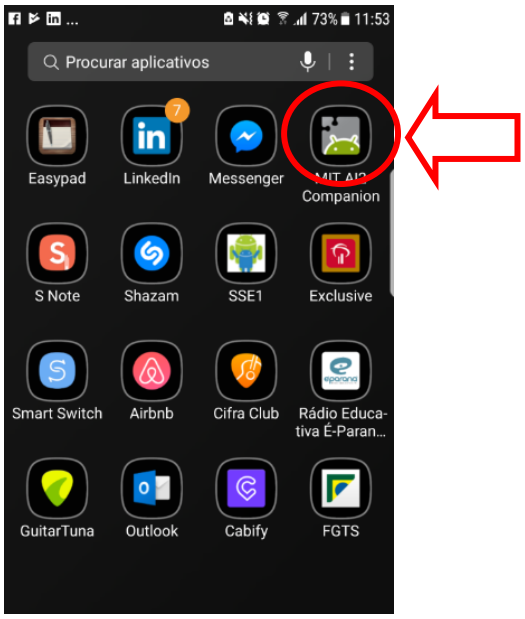


Figura 33– MIT AI Companion
Fonte: Autoria Própria

Com o aplicativo "*MIT App Inventor 2*" instalado, ao clicar na opção 1 indicada em vermelho na figura 34, é necessário escanear o *QRcode* para a instalação do aplicativo projeto.

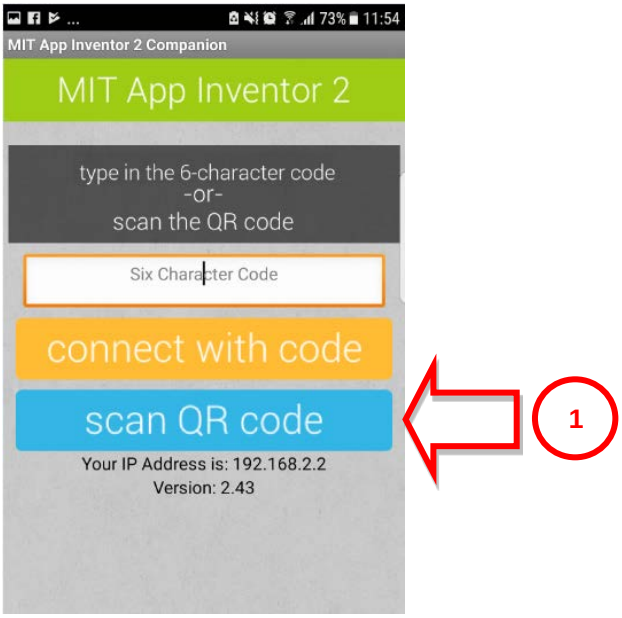


Figura 34– Scan QRcode
Fonte: Autoria Própria

Feito o *scan* do código, é necessário clicar na opção 2 indicada em vermelho na figura 35 para iniciar a instalação do aplicativo.

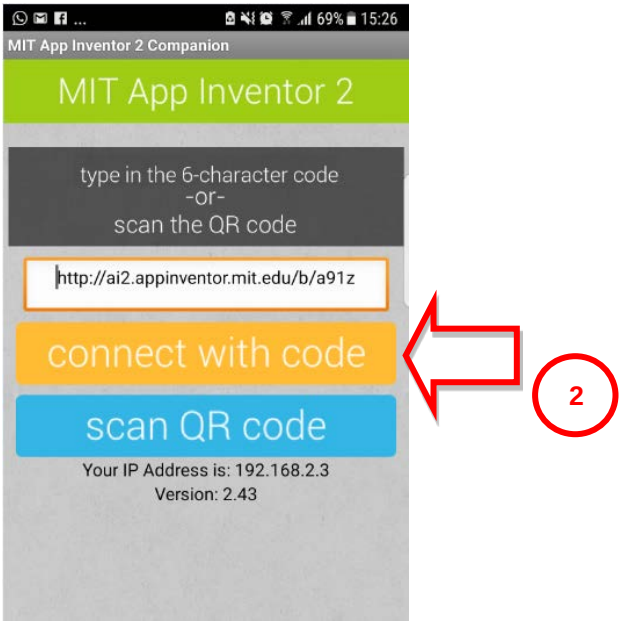


Figura 35– Connect with code
Fonte: Autoria Própria

A figura 36 apresenta a tela que indica o final da instalação do aplicativo. Após a finalização da etapa de instalação o aplicativo abre a tela principal na qual é feita a busca dos dispositivos Bluetooth disponíveis através do comando “BT *Search*”, encontrado o Mac *address* da placa, exemplo: “00:16:83:35:77:CC HC-06”, basta clicar no botão “Connect”, para parear o aparelho celular com a placa de controle.

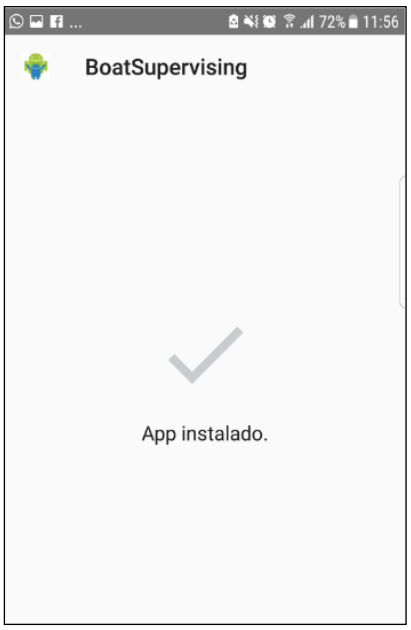


Figura 36- *App* instalado
Fonte: Autoria Própria

3.2.3 FLUXOGRAMA DE FUNCIONAMENTO

A figura 37 apresenta o fluxograma geral do aplicativo desenvolvido.

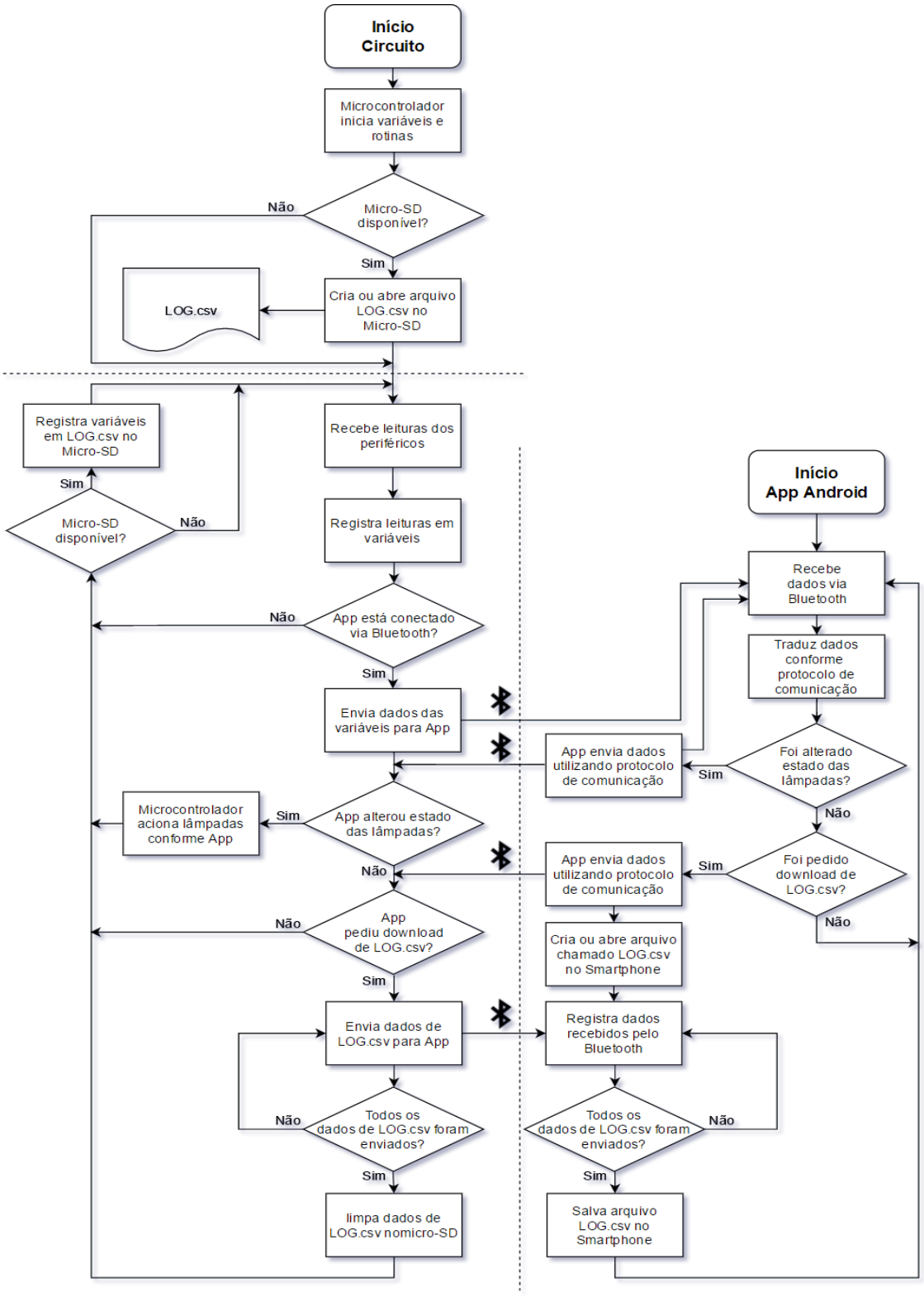


Figura 37– Fluxograma de funcionamento do sistema
Fonte: Autoria própria

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Este trabalho mostrou que é possível utilizar os recursos d*o hardware* Bluetooth disponível em *Smartphones.*

A uma taxa de transmissão serial de 9600MB/s, verificou-se em testes realizados, que o protótipo permite realizar de forma satisfatória o recebimento e transmissão de sinais de dispositivos Bluetooth em uma interação entre o *Smartphone* e placa do Arduino. A comunicação Bluetooth apresentou, contudo, uma limitação com relação ao número de usuários que podem acessar o sistema, pois este protocolo permite o acesso de apenas um usuário por vez. Este fato reforça a questão do monitoramento em modo remoto, pois quando o tripulante está ausente da cabine de comando, torna-se mais importante o monitoramento através do *Smartphone.*

Foi constatado o armazenamento de todas as variáveis analógicas do processo, pelo uso do cartão de memória Micro-SD no circuito. Foi constatado que o aplicativo com sua função de importar os dados do cartão de memória para o Smartphone possibilita o arquivamento de planilhas no formato Excel para posterior analise desses dados. A utilização do recurso RTC (*Real Time Clock*) no circuito se mostrou muito importante para refinar a base de dados do sistema, registrando todos os eventos no formato data e hora.

Nos baseamos em alguns fatores lógicos para realizar o cálculo de capacidade de armazenamento dos dados que são transmitidos para o Cartão Micro-SD. O método utilizado para realizar o cálculo está descrito abaixo:

1. Desenvolveu-se um formato tipo “string” para transmitir a cada um segundo informações em um arquivo Log.csv.
2. Utilizou-se um cartão de memória Micro-SD com capacidade de armazenamento de 8GB.
3. O arquivo Log recuperado tem o tamanho de aproximadamente 50KB conforme Figura 38.

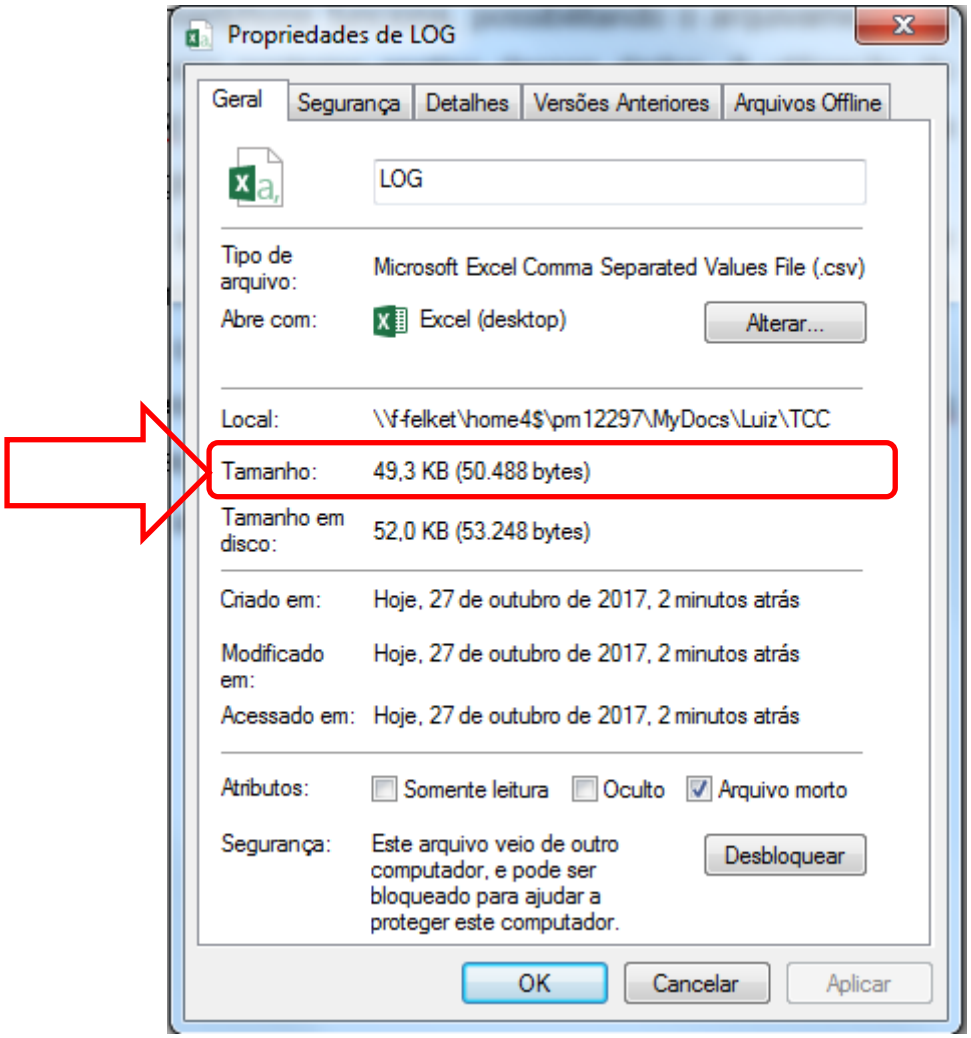
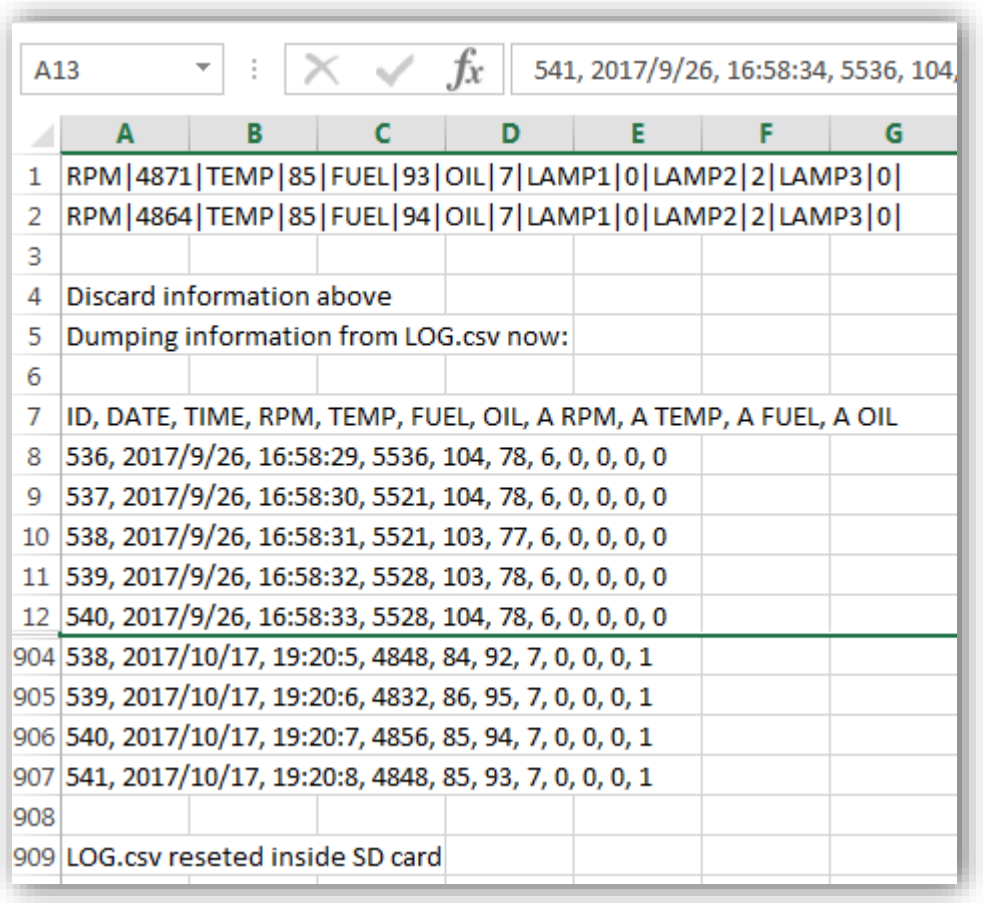


Figura 38– Tamanho LOG.cvs
Fonte: Autoria própria

4. Contabilizou-se o número de linhas na planilha com 900 linhas de armazenamento de dados, conforme mostrado na Figura 39.



A13 | 541, 2017/9/26, 16:58:34, 5536, 104,

	A	B	C	D	E	F	G
1	RPM|4871|TEMP|85|FUEL|93|OIL|7|LAMP1|0|LAMP2|2|LAMP3|0|						
2	RPM|4864|TEMP|85|FUEL|94|OIL|7|LAMP1|0|LAMP2|2|LAMP3|0|						
3							
4	Discard information above						
5	Dumping information from LOG.csv now:						
6							
7	ID, DATE, TIME, RPM, TEMP, FUEL, OIL, A RPM, A TEMP, A FUEL, A OIL						
8	536, 2017/9/26, 16:58:29, 5536, 104, 78, 6, 0, 0, 0, 0						
9	537, 2017/9/26, 16:58:30, 5521, 104, 78, 6, 0, 0, 0, 0						
10	538, 2017/9/26, 16:58:31, 5521, 103, 77, 6, 0, 0, 0, 0						
11	539, 2017/9/26, 16:58:32, 5528, 103, 78, 6, 0, 0, 0, 0						
12	540, 2017/9/26, 16:58:33, 5528, 104, 78, 6, 0, 0, 0, 0						
904	538, 2017/10/17, 19:20:5, 4848, 84, 92, 7, 0, 0, 0, 1						
905	539, 2017/10/17, 19:20:6, 4832, 86, 95, 7, 0, 0, 0, 1						
906	540, 2017/10/17, 19:20:7, 4856, 85, 94, 7, 0, 0, 0, 1						
907	541, 2017/10/17, 19:20:8, 4848, 85, 93, 7, 0, 0, 0, 1						
908							
909	LOG.csv reseted inside SD card						

Figura 39 - Planilha LOG.cvs
Fonte: Autoria própria

5. Realizou-se o cálculo de capacidade de armazenamento considerando que um arquivo com o tamanho de 50KB, leva 900 segundos para ser preenchido. Desta forma para descobrir o tempo para esgotar a capacidade do cartão Micro-SD, multiplicou-se o valor de 8GB por 150 minutos e então esse resultado é dividido por 50KBytes, o que resulta aproximadamente 40 horas de capacidade de armazenamento. Considerando um circuito que acione a placa de controle somente enquanto a embarcação estiver em funcionamento, foi considerada uma taxa de utilização diária de 6 horas. Considerando essas informações a embarcação poderia trabalhar aproximadamente uma semana armazenando os dados de seu funcionamento.

5 CONSIDERAÇÕES FINAIS

Concluímos que o protótipo desenvolvido atende as expectativas segundo testes realizados e servirá como base para desenvolvimento de futuros produtos, levantamos alguns pontos necessários a serem desenvolvidos para passar da fase de protótipo para a fase de produto:

- Desenvolver uma caixa com grau de proteção IP67 para proteger a placa Arduíno e seus periféricos;
- Avaliar necessidade de implementação de um módulo de comunicação ethernet, visando possibilitar o monitoramento por vários usuários simultaneamente;
- Desenvolver um projeto que possibilite conectar a placa Arduino ao circuito existente do painel de comandos da embarcação de forma simples, através de chicotes e conectores de engate rápido;
- Implementar envio de mensagens ao *Smartphone* acompanhadas da função alarme sonoro e vibratório conforme a criticidade do alarme;
- Armazenar no banco de dados mensagens de texto referentes aos alarmes e falhas gerados durante o uso da embarcação;
- Adicionar ao aplicativo a função de diário de bordo, possibilitando ao tripulante preencher um *checklist* eletrônico sobre condições de máquina, número de passageiros a bordo, rota a ser realizada, registro de saída de origem e chegada ao destino;
- Criar senha para usuários autorizados utilizarem o sistema de forma segura, sem que outras pessoas tentem parear seus *Smartphone*s interferindo na conexão Bluetooth.

.

REFERÊNCIAS

APPINVENTOR. Disponível em: <https://www.appinventor.mit.edu/explore/resources.html>. Acesso em: 09 dez. 2016.

ARDUÍNO. Disponível em: <https://www.arduino.cc/en/Tutorial/Home> . Acesso em: 10 dez. 2016.

ARDUÍNO. Disponível em: <https://www.arduino.cc/en/Mainl/Software> Acesso em: 23 mai. 2017.

DORF, Richard C; BISHOP, Robert H. **Sistemas de Controle Modernos**. 8. Ed. Rio de Janeiro: LTC Editora., 2006.

Exame. Disponível em: <http://exame.abril.com.br/tecnologia/noticias/8-3-milhoes-de-smartphones-sao-vendidos-no-segundo-trimestre>. Acesso em 12 jan. 2014.

Globo. Disponível em: <http://g1.globo.com/tecnologia/noticia/2012/11/numero-de-smartphones-no-mundo-vai-triplicar-ate-2018-aponta-estudo.html>. Acesso em 13 jan. 2017.

PEREIRA, Fábio. **Microcontroladores PIC – Técnicas Avançadas**. 6 ed. São Paulo, 2007.

LECHETA, Ricardo R. **Google Android: Aprenda a criar aplicações para dispositivos móveis com o Android SDK**. 3. ed. São Paulo: Novatec Editora LTDA., 2009.

MORIMOTO, Carlos Eduardo. **Redes, guia prático.** Porto Alegre: Sul Editores, 2008.

MORIMOTO, Carlos E. **Smartphones: Guia Prático**. 1. ed. Porto Alegre: Editora Meridional, 2009.

RUFINO, Nelson Murilo de Oliveira. **Segurança em Redes sem Fio.** 2. ed. São Paulo: Novatec Editora LTDA., 2007.

SEMICO. Disponível em: http://semico.com/content/microcontroller-market-analysis-forecast Acesso em: 11 fev. 2015.

SEMICO. Disponível em: http://semico.com/content/momentum-carries-mcus-2011 Acesso em: 11 fev. 2015.

VIDAL, Rogério Carlos dos Santos. **Eletricidade e Automação 1**. Ed. Belém: Diretoria de Portos e Costas, 2009.