UNIVERSIDADE FEDERAL DE VIÇOSA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

ROBERTO FREIRE FERRAZ PASSOS

TELEMETRIA E SUPERVISÓRIO DE SISTEMA EM JAVA DO VEÍCULO UFVBAJA

VIÇOSA

2016

ROBERTO FREIRE FERRAZ PASSOS

TELEMETRIA E SUPERVISÓRIO DE SISTEMA EM JAVA DO VEÍCULO UFVBAJA

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.
Orientadora: Dra. Kétia Soares Moreira.

VIÇOSA
2016

Nesta página será inserida a ficha catalográfica correspondente à sua Monografia. Ela será elaborada pelo pessoal da Biblioteca Central da UFV.

A priori deixa a página em branco.

ROBERTO FREIRE FERRAZ PASSOS

TELEMETRIA E SUPERVISÓRIO DE SISTEMA EM JAVA DO VEÍCULO UFVBAJA

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovada em 7 de Novembro de 2016.

COMISSÃO EXAMINADORA

Prof. Dra Kétia Soares Moreira - Orientadora

Universidade Federal de Viçosa

Prof. Dr. Denilson Eduardo Rodrigues - Membro

Universidade Federal de Viçosa

Prof. Erick Matheus da Silveira Brito - Membro

Universidade Federal de Viçosa

"*Our knowledge has made us cynical; our cleverness, hard and unkind. We think too much and feel too little. More than machinery, we need humanity. More than cleverness, we need kindness and gentleness. Without these qualities, life will be violent and all will be lost*"

(Charles Chaplin , O Grande Ditador – 1940).

A minha família, amigos, professores, bajeiros e ex-bajeiros.

Agradecimentos

Dedico este trabalho a todos que fizeram parte desta história. Dedico primeiramente a meu pai, Roberto Santos Passos, que sempre comenta, muito feliz, sobre o filho que saiu de casa, estudou engenharia numa Universidade Federal e ganhou bolsa de estudos nos Estados Unidos. A minha Tia, Ana Virginia Santos Passos, que sempre me deu suporte. A minha mãe, Eliana Freire Ferraz que sempre me incentivou a estudar. A minha madrasta, Valdênia Bittencourt Silva que sempre foi como uma tia para mim. A minha Tia, Raimunda Rodrigues da Cruz Ferraz, que me incentivou a tentar o intercâmbio. A minha avó, Rosália Freire Ferraz, que sempre me ajudou. Ao apoio dos meus irmãos, primos e parentes. Aos amigos da Bahia, as amizades que fiz em Viçosa e no intercâmbio, aos colegas, professores e todas as pessoas que foram importantes nesta história.

Agradeço também a FAPEMIG pelo apoio financeiro concedido, sem o qual esta pesquisa não teria sido possível.

Resumo

Esta monografia é o resultado do desenvolvimento e estudo de um sistema de transmissão dos dados do veículo baja por telemetria e seu monitoramento via computador através de um sistema de supervisório. Anteriormente, Jullierme Emiliano Alves Dias também havia desenvolvido um sistema de transmissão e monitoramento de dados na monografia intitulada Eletrônica, Instrumentação e Telemetria Do Veículo Ufvbaja. Houve, então, a substituição do antigo supervisório, em Matlab para Java. Um novo software gratuito, de fácil leitura dos dados e exigirá menos processamento do computador. Além disso, houve redução do conjunto de dados transmitidos e a modificação dos cabos seriais, DB9 e DB25, para USB através do *XBee Adapter,* simplificando o sistema eletrônico de telemetria. Os resultados foram abordados de forma qualitativa, com a análise e verificação da troca de informações entre o carro e o PC. No controle e análise de dados automobilístico, verificou-se principalmente a importância da telemetria e dos programas em Java.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO COMPREENSIVA

No início do ano de 2016, chefes de governo e ministros, bem como representantes de empresas, bancos e diferentes grupos sociais se reuniram para o encontro anual organizado pelo Fórum Económico Mundial para discutir várias questões políticas e econômicas. Entre elas, a Quarta Revolução Industrial, também chamada de Indústria 4.0 e nos Estados Unidos, Internet das Coisas [1]. A Fig. (1), curiosamente, resume e ilustra o progresso do setor industrial ao longo das 4 revoluções: vapor, eletricidade, automação e Internet das Coisas.

Figura 1. *4 revoluções industriais [2].*

A necessidade da quarta revolução industrial, fica elucidada considerando uma grande fazenda produtora. Neste local, para optimização do processo, deseja-se que todos os instrumentos estejam interconectados em rede, ou em nuvem, ou em outros sistemas de conectividade. Esta interconexão possibilita que informações, como a necessidade de manutenção do maquinário e a atualização constante do valor do produto internacionalmente, para a estimada demanda na fábrica, sejam conhecidas em tempo real. Além disto, esta interconexão permite: interagir com as informações presentes em tempo real em qualquer lugar; monitorar todas as variáveis a fim de facilitar a tomada de decisões; pré-programar rotinas das atividades da fábrica; maximizar a produção por meio de um sistema inteligente; conectar a produção da fábrica e seus fornecedores; a atualizar dados do clima para regulamentar a produção, entre outros. Assim é o *Cyberphysical Systems* (Sistema Ciber-Físico), sistema composto por elementos computacionais colaborativos com o intuito de controlar entidades físicas.

Neste contexto, a 4ª revolução industrial visa trazer soluções principalmente com o novo conceito chamado Internet das Coisas.

Hyggo Almeida [3] define Internet das Coisas da seguinte forma:

> [...] Do inglês Internet of Things (IoT), a Internet das Coisas refere-se à integração de objetos físicos e virtuais em redes conectadas à Internet, permitindo que “coisas” coletem, troquem e armazenem uma enorme quantidade de dados numa nuvem, em que uma vez processados e analisados esses dados, gerem informações e serviços em escala inimaginável. Apontada como uma revolução tecnológica iminente e com mercado mundial estimado em 1,7 trilhão de dólares em 2020 (idc.com, 2015), a IoT

> gera impacto em todas as áreas, incluindo indústria, eletrônica de consumo, saúde, e, de maneira transversal, na forma como a sociedade consome informação (Hyggo Almeida, 2016, p. 6).

Na área automobilística não é diferente. Carro e computadores trocam informações a partir da telemetria, implementada usualmente por transmissão de rádio frequência ao invés do uso da internet. Na Fórmula 1, a telemetria é muito usada para armazenamento e análise dos seguintes dados; temperatura dos freios, unidades de controle da água, óleo, combustível, motor, transmissão e pressão das partes mecânicas de todos os sistemas hidráulicos (água, combustível, freios, transmissão), RPM, injeção e pressão, combustível e etc [4].

A Magneti Marelli, subsidiária do grupo Fiat, que produz sistemas para uso em automóveis, por exemplo, produziu o software WinTAX4, dedicado à análise de dados de telemetria. Este *software* está no topo do ranking das aplicações de PC para aquisição e análise de dados em competições Motorsport. Resultado de mais de 20 anos de evolução contínua e estreita parceria com as equipas de topo na F1, DTM, WRC, Le Mans Series, GT, MotoGP e SuperBike. Interfaces de APIs (*Application Programming Interface*, em português Interface de Programação de Aplicativos) permitem o WinTAX4 de ser usado em conjunto com aplicativos externos como o Excel [5]. APIs são conjunto de rotinas e padrões de programação para acesso a um aplicativo de software ou plataforma baseado na Web.

Neste trabalho, o novo sistema de telemetria, com redução dos dados transmitidos, e o supervisório em Java, substituindo a antiga plataforma em Matlab, será aplicado ao veículo projetado e construído pela equipe UFVBaja. Esta equipe é composta por estudantes das engenharias e vinculados ao DEP, DEL e DEA (Departamento de Engenharia Mecanica e Produção, Elétrica e Agrícola, respectivamente) da UFV. Baja SAE é uma categoria automobilística de nível internacional para veículos de competição fora de estrada criada pela SAE (*Society of Automotive Engineers*, traduzido como Sociedade de Engenheiros da Mobilidade) [6] em 1976 nos Estados Unidos. A Baja SAE tem como objetivo estimular jovens estudantes de engenharia a projetarem e construírem estes veículos. Ela é regida por uma série de regulamentos a serem seguidos para garantir segurança e competitividade. O veículo deve estar preparado para vencer terrenos acidentados e condições climáticas severas sem apresentar danos em sua estrutura. A Fig. (2) mostra uma fotografia do veículo UFVbaja na 22ª Competição Nacional Baja SAE Brasil – Petrobras.

Figura 2. Carro durante a competição nacional de 2016 (foto tirada pela equipe).

William J. descreve em seu livro [7], Introdução ao MATLAB para Engenheiros, a importância do Matlab:

> [...] Formerly used mainly by specialists in signal processing and numerical analysis, MATLAB® in recent years has achieved widespread and enthusiastic acceptance throughout the engineering community. [...] MATLAB is programmable and has the same logical, relational, conditional, and loop structures as other programming languages, such as Fortran, C, BASIC, and Pascal. [...] The popularity of MATLAB is partly due to its long history, and thus it is well developed and well tested. People trust its answers. Its popularity is also due to its user interface, which provides an easy-to-use interactive environment that includes extensive numerical computation and visualization capabilities. Its compactness is a big advantage. For example, you can solve a set of many linear algebraic equations with just three lines of code, a feat that is impossible with traditional programming languages. MATLAB is also extensible; currently more than 20 “toolboxes” in various application areas can be used with MATLAB to add new commands and capabilities (William J. Palm III, 2011, p. 10).

Assim, indiscutivelmente, o Matlab é uma ferramenta computacional extremamente poderosa na resolução e simulação de problemas matemáticos, físicos, nas áreas de engenharias entre outros principalmente por trabalhar com matrizes. Entretanto, é pouco aplicado na área de automação industrial. Por outro lado, a linguagem Java está presente nas maiores empresas que prestam este serviço.

A antiga plataforma teve como pontos negativos: o software dispendioso e exigência de alta capacidade de processamento da máquina para interface gráfica. Além disso, possuia uma interface gráfica carregada de informações e apresentou muitos erros durante o recebimento de dados.

Este trabalho de conclusão de curso servirá como um suporte para os futuros integrantes do Baja que venham a se interessar pela área de Telemetria e Supervisório de Sistema. Além disso, esta monografia dará prosseguimento à monografia de Jullierme Emiliano Alves Dias, chamada Eletrônica, Instrumentação e Telemetria Do Veículo Ufvbaja, escrita em 2010 [8]. Vale salientar, que foi a única monografia feita até então no Departamento de Engenharia Elétrica da UFV referente a este projeto. Assim, mais informações sobre eletrônica e instrumentação podem ser encontradas nessa monografia [8]. Este trabalho, diferentemente da antiga monografia, usa um novo sistema eletrônico de recepção dos dados e o novo sistema de supervisório que serão explicados na seção Materiais e Métodos.

1.1 OBJETIVO GERAL

O objetivo geral será desenvolver um sistema eficiente de telemetria de transmissão de dados e um novo Sistema de Supervisório em Java, a fim de substituir a antiga plataforma em Matlab do projeto Baja. A nova plataforma deverá possuir interface limpa e clara para monitoramento e controle das variáveis de forma eficiente e inteligente.

1.2 OBJETIVOS ESPECÍFICOS

Para os objetivos específicos, a nova plataforma deverá:

- Verificar o conjunto de dados recebidos (status recebido, vazio ou incompleto).
- Amostrar as 4 variáveis dos 4 sensores/transdutores disponíveis: velocidade da roda dianteira, tensão da bateria, temperatura da caixa da *CVT* (Transmissão continuamente variável) e nível da gasolina no tanque.
- Reduzir o conjunto de dados a ser transmitidos de 64 caracteres para 12.
- Interface gráfica simplificada.
- Criar um arquivo executável para fácil uso do aplicativo sem necessidade de conhecimentos prévios de programação.
- Desenvolver a plataforma em Java.
- Enviar informações de volta para o carro como sinal de aviso para o carro voltar ao boxe.
- Avisar o estado crítico da temperatura por meio de alarme sonoro.
- Criar uma árvore de decisão simplificada para o desligamento automático dos sensores à medida que a carga da bateria diminua.
- Criar um relatório em bloco de notas/planilha das variáveis.

2 REVISÃO BIBLIOGRÁFICA

2.1. COMUNICAÇÃO SERIAL UART NO PADRÃO RS-232

A comunicação serial (comunicação dos bits dos dados enviados em série em apenas um barramento) tem como vantagem em comparação a paralela (comunicação dos bits dos dados enviados em conjunto em vários barramentos), o fato de ser economicamente mais viável e de poder ser aplicada para longas distâncias. Existe um tipo de máquina quando se trata de comunicação serial chamado USART (*Universal Synchronous/Asynchronous Receive/Transmit* em inglês) que pode ser operado no modo de transmissão UART [9].

A transmissão síncrona ocorre quando o transmissor e o receptor estão sincronizados. Assim, transmite-se dados em uma taxa mais regular, ou em alta velocidade. Entretanto, este tipo de transmissão exige uma dispositivo mais sofisticado para a sincronização constante entre o receptor e transmissor [9].

A Transmissão Assíncrona, entretanto, ocorre quando os dados transmitidos são gerados em intervalos aleatórios. De acordo com Lucas Ferreira [10]:

> [...] Com esta forma de transmissão o receptor tem de ser capaz de re-sincronizar no início de cada novo caractere recebido. Para permitir que o equipamento de recepção determine o início e o fim de cada caractere, o código de cada caractere é precedido por um bit que indique seu início e é sucedido por um ou mais bits que indiquem seu término. Eles são chamados de bits de START e STOP. Provavelmente a forma mais usada de código para transmissão assíncrona é o código ASCII. Este código tem 7 bits de dados (Lucas Ferreira, 2007).

Na comunicação serial, o padrão RS-232 é uma dos mais difundidos no mundo da automação e controle. RS-232 é um padrão de protocolo para troca serial de dados binários no qual os caracteres são enviados um a um como um conjunto de bits. A velocidade transmissão de dados seriais costuma ser expressa como bps (*bits-per-second*) ou baud (*baudot rate*) [11]. Este método permite que seja utilizado um único fio para a transmissão (*TX*) e recepção (*RX*) [9].

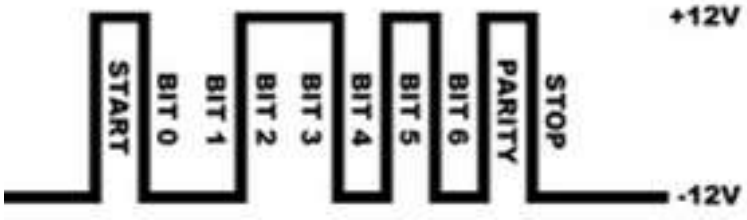

Figura 3. Envio de um byte [20].

Na Fig. (3), observa-se o exemplo de um caractere sendo enviado por transmissão serial assíncrona no padrão RS-232 com um bit de início, seguido por sete bits de dados, possivelmente um bit de paridade e um bit de paragem sendo então necessários pelo menos 10 bits para enviar um único caractere [12]. Vale notar que o estado alto e baixo é dado neste caso respectivamente pela voltagem de 12 (V) e de -12 (V).

2.2 TELEMETRIA

A Telemetria é normalmente usada para controle, medição ou rastreio de dispositivos à distância ou em movimento constante – como automóveis [13].

Maurício Marinho define telemetria, em sua dissertação de mestrado [14], além das aplicações:

> [...] Telemetria é a transferência (via rede fixa ou sem fio) e utilização de dados provenientes de múltiplas máquinas remotas, com sensores de captação de dados e/ou medidas, distribuídas em uma área geográfica de forma pré-determinada, para a sua monitoração, medição e controle [...] Um sistema embutido (também é utilizado o termo "sistema embarcado") consiste, normalmente, de um conjunto de componentes de hardware e de software que interagem para executar um conjunto de operações prescritas. Aplicações embutidas incluem sistemas aeroespaciais, sistemas de controle de automóveis, sistemas multimídia, telecomunicações (incluindo telefones móveis), controladores industriais, dentre outros (Maurício Marinho, 2005).

2.2.1 ZIGBEE: XBEE PRO SERIES 2

As Redes ZigBee, que utilizam transmissão por rádio frequência, tem como principais vantagens oferecer uma excelente imunidade contra interferências, a capacidade de hospedar milhares de dispositivos numa rede (mais que 65.000) e durabilidade de carga de meses ou anos devido ao baixo consumo [15].

O sistema ZigBee permite várias topologias de rede de transmissão de dados na qual cada dispositivo desempenha uma função:

- Coordenador (*ZigBee Coordinator*): Nó inicial que controla toda a rede, inicia e mantém os dispositivos na rede.
- Roteador (*ZigBee Router*): dispositivos, opcionais, intermediários na rede que permitem encontrar o menor caminho para se chegar ao destino.
- Dispositivo final (*ZigBee Endpoint*): são os nós finais das topologias.

A Fig. (4) ilustra a topologia Estrela, mais simples, na qual os dados são enviados diretamente do *Coordinator* para o *Endpoint.*

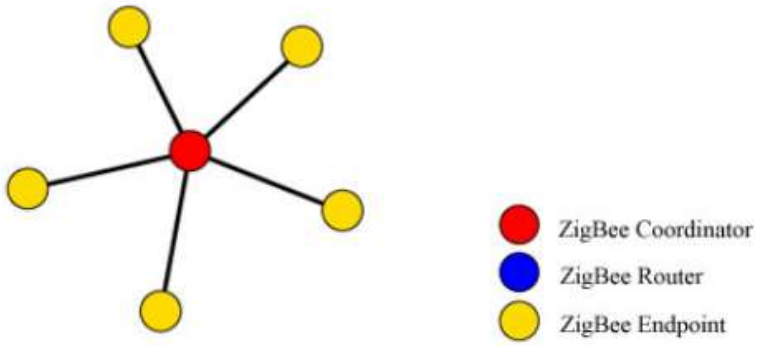

Figura 4. Topologia Estrela (Star) [*16*].

2.3 ANTENAS OMNIDIRECIONAIS

O ganho de uma antena é calculado pelo que é adicionado ao sistema, normalmente mensurado em dBi (decibel). Portanto, para um ganho de 2 (dBi), soma-se 2 (dBi) ao sistema [17].

O calculo da Razão de Onda Estacionária (ROE) é dado pelo módulo da divisão da amplitude de uma onda estacionária em um ponto de máximo pelo seu ponto de mínimo. O valor ROE de 1.5:1, por exemplo, demonstra que a máxima amplitude da onda estacionária é 1.5 vezes maior que o valor do mínimo da onda estacionária [17], [18] e [19].

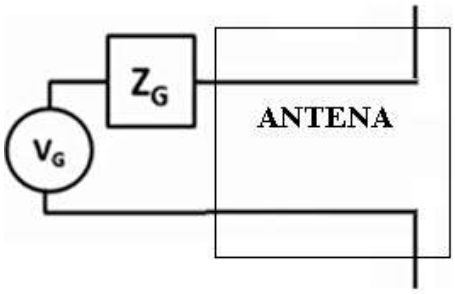

Figura 5. Representação modificada de antena [18].

A Fig. (5) foi criada para ilustrar uma antena com embasamento nos conhecimentos sobre linhas de transmissão, Elementos de Eletromagnetismo [18]. Neste contexto, Vg representa a voltagem da onda da fonte (transmissor) e Zg, a impedância interna do cabo do transmissor para a antena. Se houver o casamento de impedâncias entre Zg e antena, haverá a máxima energia transferida do cabo para a antena, mínimo de perdas. Como consequência, o transmissor irá operar a um valor baixo de ROE [17] e [18].

Caso não haja o casamento de impedâncias, uma parte da energia elétrica não será transferida para antena. A energia não transferida então é refletida para o transmissor [18]. A energia refletida de volta poderá apresentar três possíveis problemas em transmissão de radiofrequência (RF): aumento das perdas de potência nas ondas transmitidas, presença de ruídos no sinal além de danos aos circuitos do transmissor [19].

Um dispositivo com antena omnidirecional irradia o sinal 360° horizontalmente, permitindo que um outro dispositivo troque informações com aquele a partir de qualquer ponto na área em torno do ponto de acesso, contanto que esteja na mesma altitude. Isto porque, antena omnidirecional normalmente irradiam pouco sinal na vertical. Sendo assim, na tabela do produto desta antena omnidirecional, será descrito como polarização do tipo vertical. Portanto, esta antena apresentará polarização horizontal de 360° e algum valor de ângulo de polarização vertical [20] como ilustrado na Fig. (6).

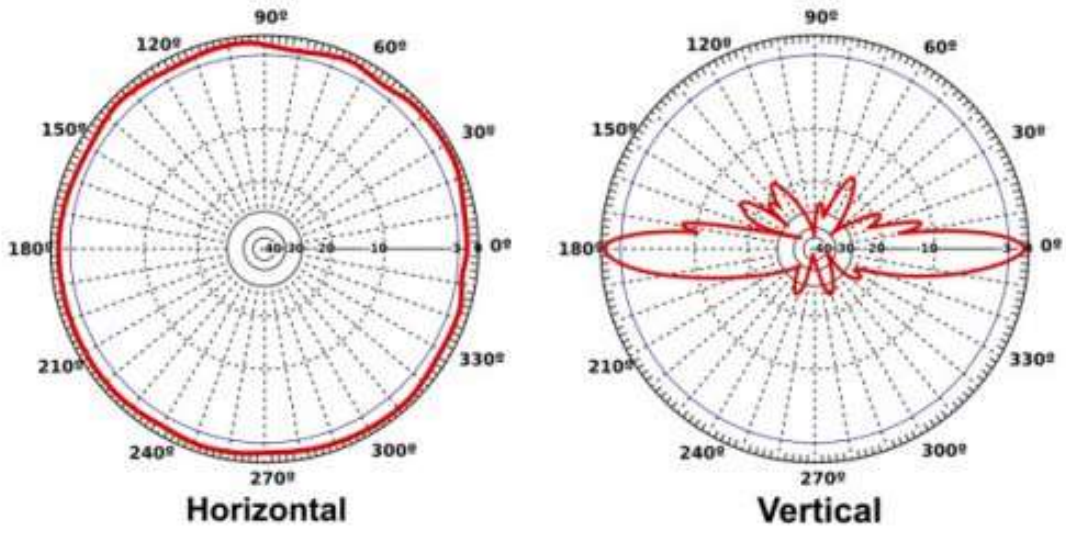

Figura 6. Ilustração da polarização horizontal e vertical [20].

As antenas devem estar todas alinhadas, na vertical, evitando assim perda de transmissão dos dados devido à uma grande diferença de altura entre elas. Neste caso, o ângulo entre as antenas, numa visão de perfil, será maior que o ângulo de polarização vertical das antenas. Além disso, o sinal deve trafegar até as antenas receptoras sem muitos desvios. Isso porquê obstáculos diminuem consideravelmente a transmissão [20].

Pela redução da irradiação do sinal verticalmente, uma antena ominidirecional oferece um ganho ou aumento na potência de transmissão. Isso gera melhor qualidade da recepção do sinal comparado a uma antena que irradiasse o sinal igualmente em todas as direções, verticalmente e horizontalmente. Assim, o uso de uma antena de maior ganho em dB, ou seja, por uma antena que concentre ainda mais o sinal, permite que ele atinja distâncias ainda maiores [21]. Este ganho médio varia entre 2,15 (dBi) à 14,15 (dBi) [22] e [23]. Embora antenas omnidirecionais possuam valor comercial de 25 (dBi), o ganho real será de 14,15 (dBi) .

2.4 ARDUINO

Arduino é uma plataforma recente, desenvolvida em 2005. As principais características que fizeram o arduino ter sucesso no mercado são que ele é relativamente barato, de fácil implementação, linguagem de programação própria, simplificada e baseada nas linguagens C/C++, disponibilizado em código aberto (*OPEN SOURCE* em inglês), possui inúmeros drivers e sensores conectáveis com bibliotecas prontas entre outros dispositivos feitos especialmente para arduino e assim criando infinitas aplicações, além de ser conectável a internet [24] e [25].

O código ser *Open source* traz benefícios. A licença do programa Arduino é totalmente gratuita, com livre distribuição pelo desenvolvedor e software tem permissão para qualquer modificação [26]. Esta última característica é a mais importante no sentido de abrir espaço para novas aplicações.

Existem vários modelos diferentes de Arduino dependendo da aplicação. A Fig. (7) ilustra o modelo arduino Uno. Ele possui as características de um microcontrolador PIC como pinos analógicos e digitais, PWM, a conexão USB (*USB PLUG* em inglês) para alimentação e gravamento do programa via computador, pinos RX e TX para leitura e transmissão serial respectivamente entre outros. O arduino Uno utiliza o microcrontrolador Atmega328 com mesmo tipo de processador dos microcontroladores PIC, *RISC* (*Reduced Instruction Set CPU* em ingles ou CPU com Set de Instruções Reduzido em português).

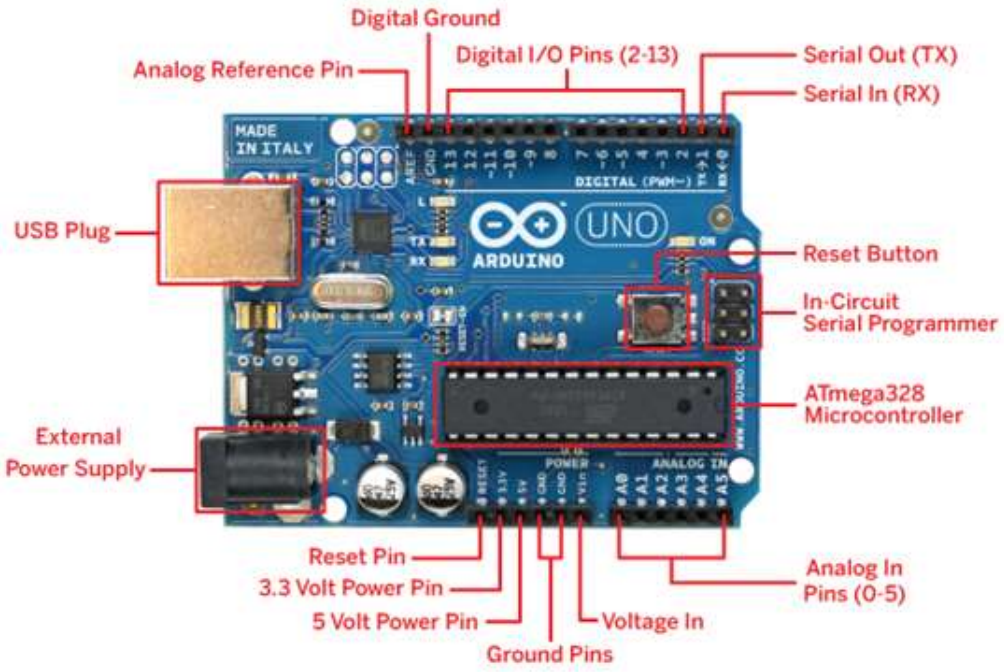

Figura 7. Arduino Uno [27].

2.5 SISTEMA SUPERVISÓRIO

Basicamente, os sistemas clássicos SCADA (*Supervisory Control and Data Acquisition* em inglês ou em português, Sistemas de Supervisão e Aquisição de Dados) monitoram e controlam dispositivos - sensores ou atuadores - através de programas de interface gráficas [28].

Ana Paula e Marcelo Salvador explicam a composição clássica dos sistemas supervisórios em seu artigo [29]:

> [...] Para permitir isso, os sistemas SCADA identificam os tags, que são todas as variáveis numéricas ou alfanuméricas envolvidas na aplicação, podendo executar funções computacionais (operações matemáticas, lógicas, com vetores ou strings, etc) ou representar pontos de entrada/saída de dados do processo que está sendo controlado. Neste caso, correspondem às variáveis do processo real (ex: temperatura, nível, vazão etc), se comportando como a ligação entre o controlador e o sistema. É com base nos valores das tags que os dados coletados são apresentados ao usuário [...] Os sistemas SCADA podem também verificar condições de alarmes, identificadas quando o valor da tag ultrapassa uma faixa ou condição pré-estabelecida, sendo possível programar a gravação de registros em Bancos de Dados, ativação de som, mensagem, mudança de cores, envio de mensagens por pager, e-mail, celular, etc. (Ana P. e Marcelo S., 2005).

Entretanto, os novos conceitos de sistemas Cyber-Físicos tendem a tornar os sistemas supervisórios cada vez mais eficientes, autônomos e customizáveis. O sistema supervisórios tem novas funções como obter conclusões dos processos, maior automatização reduzindo principalmente o fator erro-humano, previsões de falhas da máquina, maximinização da produção, maior segurança para os operadores entre outros fatores [30] e [31].

Usando como referência a empresa Citisystems, que fez projetos em grandes empresas como Schneider, Siemens e GE, os novos supervisórios de sistemas normalmente possuem os recursos descritos na Tabela (1) sobre os recursos de um supervisório de Sistemas [31].

Tabela 1. Recursos de supervisório de Sistema [2].

Ferramentas avançadas de Análise.
Integração com vários CLPs de mercado através de ferramentas de OPC.
Monitoramento remoto
Integração de sistemas (C#, Java)
Aquisição e tratamento de dados
apontamentos para obtenção de melhor rendimento, etc
Análise de confiabilidade
Threads independentes de controle
Processamento paralelo para acionamentos, monitoração e cálculo de trajetórias
Obtenção da melhor performance do equipamento industrial e aproveitamento do produto com implementação de algoritmos de análise combinatorial.
Envio e recebimento de dados via rede (telemetria).

Leitura e escrita em arquivos texto ou planilhas (xls, txt, xml). Rastreamento de falhas com visualização gráfica.
Algoritmos de análise combinatorial.
Facilidade de manutenção.
Controle de acesso de usuários.
Indicadores gráficos de produção.
Registro histórico de OEE (*Overall equipment effectiveness*).
Viabilidade econômica.
Cálculo de vida útil.
Integração com ERPs.
Custo de aquisição.
Exportação dos registros para editores como o Excel, Word, etc.
Integração com banco de dados.
Registro histórico de variáveis de produção.
Apontamento automático de Produção.
Gestão de produção por período.

2.5.1 AUTOMAÇÃO INDUSTRIAL E JAVA

Como citado na seção 1, A linguagem Java está presente fortemente na área de automação industrial. Dr. James Lathrop, Diretor de *Professional Services* discutiu na revista *Control Engineering* [32] as vantagens de Java para a automação industrial:

> [...] the time is ripe for industrial automation to reap the benefits of the Java programming language [...] What makes Java fit the industrial automation world are the concepts of JVM and class file. A class file is a sequence of machine-independent instructions interpreted by the JVM running on a host computer. Java virtual machines and class files have been used on the Internet for several years to download and execute programs inside web browsers [...] Because class files are machine independent, the classloader and JVM combine to load and execute Java files in exactly the same manner, with the same results, on any hardware platform. (James Lathrop, 2001).

Assim, Dr. James ressaltou a importância da linguagem Java na Automação Industrial e ainda aponta que os conceitos de *JVM* (*Java Virtual Machine* em ingles ou em português, máquina virtual em Java), de classe e a independência de máquina (em inglês *Hardware independence*) como fatores vitais da aceitação de Java nessa área da indústria. Na maioria das linguagens, a programação é compilada em código binário (*bytecode* em inglês) para o específico Sistema Operacional (*OS, Operational System*), cada uma com suas próprias linguagens de programação e bibliotecas. Entretanto, em *Hardware independence*, como no caso do Java, se deve ao fato que a programação (arquivo *.java*), que será desenvolvida para qualquer máquina, será compilada em *bytecode* (arquivo *.class*) e então a *JVM* executará o programa em Java em cada *OS* específico. Portanto, apenas a máquina virtual será dependente da máquina [32].

Dr. James particularizou essas vantagens começando pela *Hardware independence* e sua aplicabilidade junto aos CLP, ou PLC (controlador logico programavel):

> [...] One of Java's primary advantages is its ability to run programs on different hardware platforms without recompiling source code. For industrial automation, this has several benefits. First, as hardware is updated and newer controllers replace older controllers, no change in the binary software is required. For example, if a PLC that fully utilizes a 40-MHz Intel processor is upgraded to a 100-MHz Power PC processor, then the same binary class files can be used for both controllers, assuming that hardware ports and external interfaces are programmed for portability. Second, hardware independence allows developers to "simulate" the run-time environment on a desktop computer, greatly reducing development time. With the simple addition of low-level hardware simulation, the same application class files that run on the target PLC can also execute on a desktop computer for development and debugging (James Lathrop, 2001).

Além disso, Dr. James também pontuou as vantagens no processo de *debugging* (processo de encontrar e reduzir erros num aplicativo de software ou hardware):

> [...] Familiar debugging tools and development environments not tied to specific hardware platforms are easier for developers to learn and use, and testing does not require the constant downloading and setup of the target PLC. Another benefit is that debugging is only required on a single version of source code—once for all platforms. This reduces project overhead and maintenance by greatly simplifying project build configurations, reducing coding errors, and simplifying bug tracking and change requests (James Lathrop, 2001).

Dr. James também citou as bibliotecas Java que facilitam o desenvolvimento de aplicações e por fim pontuou mais uma vez que Java é vantajoso na Automação Industrial :

> [...] Java also provides software safety, developer productivity, and access to a large set of libraries that provide rapid design and coding of applications. With embedded Java libraries designed to provide pure Java interrupts and input/output functions, developers can write platform-independent device drivers, providing further cross-platform compatibility. Rapid improvements in Java technology and advancements in hardware have made Java a viable solution for industrial automation. The question is not whether Java will be adopted for use in industrial automation, but rather who will be the first to take advantage of what Java offers (James Lathrop, 2001).

2.6 PROGRAMAÇÃO EM JAVA

Para o desenvolvimento de qualquer aplicação em java é necessário o JDK (*Java Development Kit* em inglês), formado pela JRE somado a ferramentas para o desenvolvimento de aplicações em Java. O JRE (*Java Runtime Environment* em inglês ou ambiente de execução Java, em português) é formado pela JVM (*Java Virtual Machine*), bibliotecas entre outros.

Além disso, é necessário o *IDE* (*Integrated Development Environment* em inglês ou no português, Ambiente de Desenvolvimento Integrado). Ele contém várias ferramentas, como o compilador, *debugger*, *editor de texto* entre outros.

Há dois conceitos importantes de Java. Proteção dos dados no qual o dado só pode ser exibido caso haja um método para este fim. A proteção de informações esta entre as principais preocupações dentro da indústria.

Segundo, a programação é orientada a objeto. Portanto, segue uma lógica de programação diferente da usual como C,C++ ou ASSEMBLY. Uma classe é uma série de variáveis/atributos padrões. Ao se inicializar um objeto, ele deve pertencer a uma classe. Assim, objetos de uma mesma classe possuem os mesmos tipos de variáveis e métodos. Um programa pode ser composto por uma ou mais classes e objetos.

O código "miolo", em main, ilustrado pela Fig. (8), é a parte principal do programa e será executado quando chamarmos a máquina virtual. Assim, qualquer código em linhas anteriores ao método main, sendo uma variável ou método, deve ser invocado dentro do código "miolo" [33].

```
class MeuPrograma {
    public static void main(String[] args) {

        // miolo do programa começa aqui!
        System.out.println("Minha primeira aplicação Java!!");
        // fim do miolo do programa

    }
}
```

Figura *8*. Amostra do código main [*33*].

A linguagem Java também oferece um extenso conjunto de classes e interfaces para o desenvolvimento de interface gráfica do usuário ou *Graphical User Interface* (*GUI*). Além disso, o NetBeans IDE 8.1 possui o *GUI Builder* que é um construtor gráfico de alto nível ao invés da criação através de linhas de código. Na verdade, o *GUI Builder* converte a plataforma gráfica em código em Java e assim os textos, botões ou qualquer outro componente gráfico será um variável que pode ser editada na preferência do programador [33].

3 MATERIAIS E MÉTODOS

3.1 MATERIAIS

Visando cumprir os objetivos gerais e específicos, descritos nas seções 1.1 e 1.2, este trabalho utilizará o dispositivo Xbee para telemetria através do Java API (bibliotecas). O Xbee e o Java API foram desenvolvidos pela empresa *DIGI* para desenvolver um sistema de telemetria eficiente de transmissão de dados. O *Sofware* Netbeans *IDE* 8.1 será usado para o desenvolvimento e criação da aplicação em Java. Este *IDE* contém ferramentas que facilitarão a criação de interfaces gráficas.

Haverá 2 espaços de trabalho. O primeiro espaço será a oficina mecânica dentro da UFV cedida pelo DEA; para montagem, reparos e testes da parte elétrica e mecânica do veículo UFVBaja. O segundo ambiente será uma sala do DEL, apropriada para o desenvolvimento de projetos na Telemetria e Supervisório de Sistema.

O custo de todos os equipamentos eletrônicos e de instrumentação serão relativamente baixos, comparado ao custo total do carro. O computador (PC) do projeto foi usado além de uma série de componentes elétrônicos e de medicão. As placas de circuito impressas da central de processamento de dados e da telemetria, presentes no carro, serão ilustradas na Fig. (11), da seção 4. A tabela (2) lista os principais materiais utilizados neste trabalho.

Outros materiais secundários serão usados como: silver tape, conduíte, fios elétricos, conexões, cabo USB para o Xbee pro S2, ferro de solda, multímetros, osciloscópio portátil entre outros.

Tabela 2. Lista dos principais materiais usados neste trabalho.

Sensor de temperatura LM35
Transdutor indutivo de velocidade
Transdutor de nível de Combustível
Arduinos Uno e Mega
PIC modelo 18F4550
Computador do Baja
2 Xbees pro S2
Placas de circuito impressas

Os sensores do veículo continuam mostrando desempenho satisfatório e por isso não foram modificados da monografia anterior [8].

O antigo sistema de recepção dos dados está presente na figura (36) da monografia anterior. Este sistema era composto por uma placa de circuito impresso chamado sistema de

boxe, usando lcds, leds, microcontroladores, grandes conectores DB9 (9 pinos) e DB25 (25 pinos) para comunicação entre outros componentes e finalmente conexão USB para o computador. Todo este circuito será atualizado para um adapatador *XBee Adapter*, mostrado na Fig. (9).

Essa alteração simplificará o sistema eletrônico de recepção dos dados. Importante esclarecer que a comunicação serial assíncrona RS 232 será utilizada em todas as transmissões de dados; da placa embarcada do PIC para a placa do Xbee e para o *display* do carro e do Xbee presente no carro para o Xbee conectado ao adaptador usb que trasmitirá os dados ao computador.

Figura 9. XBee Adapter USB

A seguir, será explicado as características técnicas da antena e do Xbee utilizados. A Tabela (3) apresenta os valores da antena que será utilizada na transmissão e recepção dos dados da telemetria. Cada Xbee possuirá uma antena de 25 (dBi). Esta antena foi escolhida devido a sua frequência que trabalha na mesma frequência do dispositivo Xbee. A antena também possui um valor comercial de ganho de 25 (dBi). Entretanto, como visto na seção 2.3, o ganho médio máximo de uma antena é tecnicamente de 14,15 (dBi). Na verdade, ganhos acima deste valor não são ganhos da antena propriamente dito mas sim técnicas de aumento da onda refletida.

Além disso, a polarização horizontal de 360° e vertical de 25° permitirá um abrangente alcance do sinal e o valor ROE 1.5:1 será próximo do ideal como explicado na seção 2.3.

Tabela 3. Valores comerciais de Antena Omnidirecional 25 (dBi) [34].

Frequência	2.4 – 2.5 (Ghz)
Ganho	25.2 (dBi)
Tipo de polarização	Vertical
Polarização vertical	25°
Polarização horizontal	360°
Impedância	50 (Ohm)
ROE (Razão de ondas estacionárias)	1.5:1

A Fig. (10) contém os valores do Xbee pro Series 2. Além dos motivos explicados na secção 2.2.1, o Xbee será usado também pelo baixo consumo devido a baixa potência de operação, alcance em ambiente aberto de 1.6 (Km), encriptação de dados para proteção dos dados e alta taxa de transmissão de dados. Além disso, a topologia estrela será a escolhida por haver apenas dois dispositivos.

Platform	XBee-PRO® ZNet 2.5 (Series 2)
Performance	
RF Data Rate	250 kbps
Indoor/Urban Range	300 ft (100 m)
Outdoor/RF Line-of-Sight Range	1 mi (1.6 km)
Transmit Power	50 mW (+17 dBm) / Int'l 10 mW (+10 dBm)
Receiver Sensitivity (1% PER)	-102 dBm
Features	
Serial Data Interface	3.3V CMOS UART
Configuration Method	API or AT Commands, local or over-the-air
Frequency Band	2.4 GHz
Serial Data Rate	1200 bps - 1 Mbps
Networking & Security	
Encryption	128-bit AES
Power Requirements	
Supply Voltage	3.0 - 3.4VDC
Transmit Current	295 mA @ 3.3VDC
Receive Current	45 mA @ 3.3VDC

Figura 10. Tabela modificada do datasheet do Xbee Pro S2 [35].

3.2 MÉTODOS

A seguir, o passo a passo das atividades necessárias para a conclusão deste trabalho. Importante ressaltar que cada nova atividade só será realizada caso a anterior for concluída de forma satisfatória:

Primeira atividade será o envio de um caractere (uma letra do tipo *char* que é um pacote de dados na transmissão serial) do arduino para a plataforma em Matlab.

Segunda atividade será o envio dos dados simulados do microcontrolador PIC modelo 18F4550 para a plataforma em Matlab. Como já é sabido que a medição e a amostragem dos 4 sensores funciona corretamente, este trabalho dispensa esse teste e portanto os dados serão simulados.

A terceira atividade será a construção da nova plataforma em Java de acordo com os objetivos.

Por fim, na quarta atividade, o arduino será utilizado novamente por motivos de praticidade e rapidez nos testes da telemetria. Além disso, o arduino possui um software próprio no PC (que estará conectado ao arduino através da porta USB) com monitor serial que exibe os dados enviados. Assim, haverá o teste da transmissão do conjunto de dados simulados entre o arduino e a nova plataforma. Os dados simulados facilitarão os teste da telemetria e da plataforma em Java sem a necessidade da oficina do Baja nesta etapa. Assim, será enviado o conjunto dos dados simulados no arduino.

A abordagem da análise dos resultados será qualitativa. Portanto, sem cálculos estatísticos. O objetivo principal da análise será se houve a troca de informações entre o carro e o PC e se o *delay* nesta transmissão será desprezível, não comprometendo a análise dos dados. Além disso, a verificação se o pacote foi enviado de forma completa, vazia ou incompleta. Na conclusão, será verificada se a lista dos objetivos gerais e específicos será atendida.

Na última seção, anexos, serão apresentados algumas fotos, trechos de códigos e algumas explicações para a replicação deste trabalho.

4 RESULTADOS E DISCUSSÕES

O principal resultado e teste do sistema desenvolvido aconteceu na 22ª Competição Nacional Baja, realizada em São José dos Campos em início deste ano. O novo sistema de telemetria foi implementado com sucesso ainda na antiga plataforma Matlab. Após a competição, uma nova interface foi desenvolvida. Assim, o trabalho se dividiu em duas partes, antes da competição, o desenvolvimento do sistema de telemetria e após a competição, o desenvolvimento da nova interface gráfica.

4.1 CENTRAL DE PROCESSAMENTO DE DADOS E DA TELEMETRIA

A central de processamento de dados e da telemetria são mostradas na Fig. (11). A alimentação é de 12 (V), simulando a bateria do carro, como ilustrado na figura.

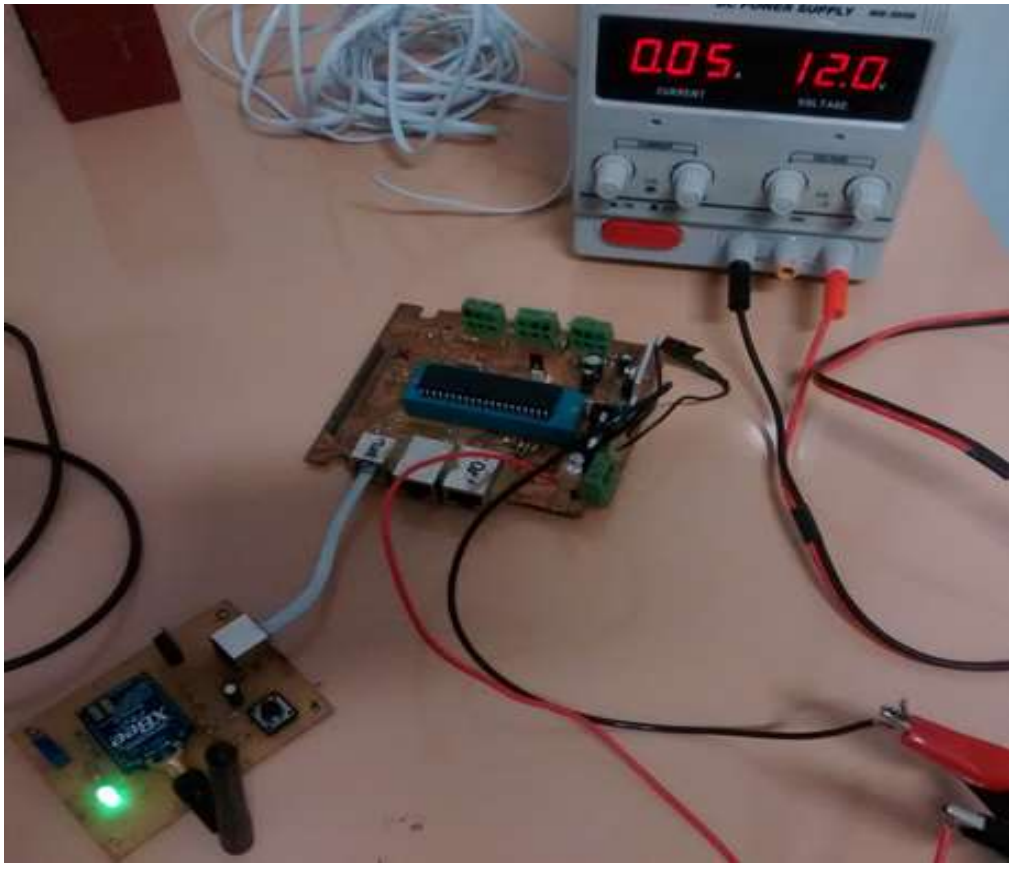

Figura 11. Placas de circuito impressas da Central de processamento de dados e da telemetria.

O consumo de corrente teve o mínimo de 50 (mA) quando o sistema embarcado estava ligado porém com nenhum sensor em funcionamento e logo nenhum dado enviado pela

telemetria. Quando todos os 4 sensores estavam ligados e os dados foram enviados, a corrente máxima era cerca de 200 (mA). A potência consumida foi relativamente baixa, variando entre 0.6 (W) à 2.4 (W). Notou-se que a antena, durante a competição, funcionou corretamente mesmo em distâncias superiores a meio quilômetro com poucas perdas de dados, mesmo com os obstáculos naturais da pista.

Todo impulso elétrico que corre através de um metal enfrenta uma resistência ao tráfego desse sinal, chamada de resistência ôhmica, que é dada por ohms/km. Por isso, todo sinal tende a ter perdas em grandes distancias. Percebeu-se isso no momento que usou-se um cabo de aproximadamente 1 metro de comprimento na transmissão serial entre a placa do PIC para a placa do Xbee. Como solução ao problema, o cabo foi encurtado como mostra a Fig. (11) e Fig. (12). Nela, podemos observar um cabo branco de menos de 10 (cm) comunicando entre a placa do xbee e a placa central.

Assim, para um conjunto de dados enviados Matlab, “v54$r0900$c3.0$b3$t010.0$h09$m09$i09$s10$d01000$a01000$f3$o3$p3$”, este, “þøa01000$f3$àøo3p3$àøàøàøàøàøv54àø$r0900$c3”, foram os dados recebidos devido a perda de sinal explicada anteriormente, ainda na etapa de desenvolvimento do sistema de telemetria antes da competição. Vale ressaltar, como explicado na seção 2.1, que a comunicação serial RS232 envia pacotes de 1 byte de caractere por vez (1 *byte* de 8 bits) e portanto para o envio de toda o conjunto de dados mostrado anteriormente foi necessário 64 pacotes (cada pacote, um caractere).

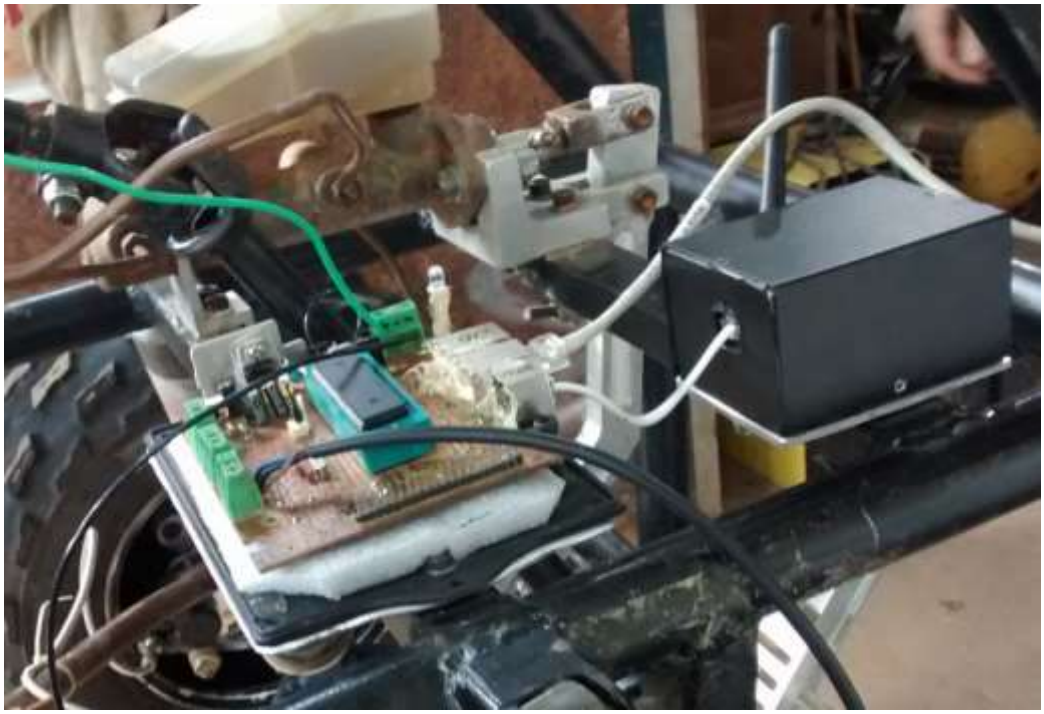

Figura 12. Placas no carro.

Conforme as normas da Baja SAE, como explicado na seção 1, o veículo deve estar preparado para terrenos adversos capaz de suportar impactos, lama, água dentre outros. A Fig. (12) mostra o circuito embarcado no carro do Baja que apresenta a estrutura de proteção. A

placa do PIC foi aberta para mostrar o sistema de espuma para amortecimentos de impactos. Além disso, foi colocada uma espuma sobre a parte plástica do PIC e comprimida contra a parede superior da caixa de proteção quando fechada. Isto foi feito para evitar qualquer contato entre a espuma e partes metálicas evitando qualquer chance de incêndio na placa.

Após o fechamento das caixas, foi usado conduíte automotivo para isolar os cabos de transmissão serial. É importante ressaltar que o mesmo cabo foi usado em toda a fiação elétrica no carro. O conduíte automotivo protege os cabos internos contra influência externas como agentes químicos, choques mecânicos, elétricos, entre outros. Além disso, todas as aberturas das caixas foram vedadas com silver tape antes das corridas evitando a entrada de lama e água.

4.2. PLATAFORMA DO SUPERVISÓRIO DE SISTEMA

4.2.1. ANTIGA PLAFORMA EM MATLAB

A Fig. (13) exibe a interface gráfica em Matlab durante teste de velocidade.

Figura 13. Supervisório em Matlab.

Vale notar que esta plataforma possui um recurso extremamente interessante que é a apresentação gráfica dos valores dos dados de forma temporal. Isto ajuda o usuário a obter um entedimento melhor dos dados do carro.

A antiga plataforma possuía um conjunto de dados transmitidos de 64 caracteres, prevendo-se o acréscimo futuro de sensores ao carro, 15 no total.

4.2.2. PLAFORMA EM JAVA

A nova plataforma em Java contemplou as características exibidas na Tabela (4) quando comparada com a Tabela (1) na seção 2.5.

Tabela 4. Características do Sistema supervisório.

Monitoramento remoto
Integração de sistemas (C#, Java)
Aquisição e tratamento de dados
Análise de confiabilidade
Envio e recebimento de dados via rede (telemetria).
Leitura e escrita em arquivos texto ou planilhas (xls, txt, xml). Rastreamento de falhas com visualização gráfica.
Facilidade de manutenção.
Exportação dos registros para editores como o Excel, Word, etc.

Assim, este supervisório de sistema criado possui algumas características de supervisórios desenvolvidos pela empresa Citisystems, referência no ramo da Automação Industrial.

Este trabalho adiciona um alarme de aviso caso o dado da temperatura supere 90°C, dentro da *CVT*, valor que pode vir a danificá-la.

Um exemplo do conjunto de dados para a nova plataforma está mostrado abaixo. O "$" significa fim do conjunto de dados. A letra "v" de velocidade, o primeiro valor de 2 dígitos "32" corresponde a velocidade do carro na unidade de Km/h. "t" de temperatura, o segundo valor é relativo a temperatura da *CVT*, "56". "b" de tensão da bateria seguido do valor "12" em volts e por fim "g" de nível do tanque de gasolina com seu valor "2".

v32t56b12g2$

Em seguinte, o tratamento dos dados. Dessa forma, a aplicação Java irá "quebrar" o conjunto de dados em pedaços menores o que na programação é chamado manipulação de strings. Finalmente, cada dado é alocado para cada variável String; velocidade, temperatura, gasolina e bateria e então esses valores serão exibidos na interface gráfica desenvolvida na classe NewJFrame. Na verdade, esta aplicação foi desenvolvida a partir de 2 únicas clásse; a NewJFrame que foi totalmente dedicada para criar a interface gráfica do Sistema supervisório

e a classe MainApp que possui o método “main” que é o centro do programa como explicado na seção 2.6.

Além disso, a classe MainApp contém todas as outras variáveis, códigos responsáveis pelo envio e recebimento dos conjuntos de dados, pela escrita dos dados em planilhas entre outras partes de código.

Foi acrescentado ao projeto, a árvore de decisão. Desta forma, foi desenvolvido uma árvore de decisão simples em nosso Sistema supervisório a fim de corrigir um problema bastante comum nas competições: o descarregamento rápido da bateria. A árvore de decisão segue o seguinte raciocínio:

- Se bateria < 11 (V) desliga sensor de combustível.
- Se bateria < 10 (V) desligar sensor de velocidade.
- Se bateria < 9 (V), desligar sensor de bateria e temperatura.

Segue abaixo a Fig. (14) da nova plataforma em Java.

Figura 14. Supervisório em Java.

É importante ressaltar que o relatório em bloco de notas/planilha serve para a análise posterior das variáveis, para a criação de gráficos no tempo e para o cruzamento e verificação

da correlação entre elas. Por isso, este relatório é automaticamente feito de forma que cada dado esteja em uma coluna.

Esta aplicação foi finalizada como arquivo executável e assim sem a necessidade de um *IDE* para roda-la. As 4 tags, 4 variáveis numéricas, possui texto em fonte grande para a melhor visualização. A entrada COM é referente as portas de comunicação USB. Para cada variável, existe um botão ON/OFF para desligamento de cada sensor no carro além do botão Boxe para aviso de alerta para o piloto retornar aos boxes. Vale resaltar que estes 5 botões foram já programados para futura implementação eletrônicas no carro. Na parte inferior, há a verificação do conjunto de dados.

5 CONCLUSÕES E PERSPECTIVAS

Com como conclusão do resultados da amostragem por telemetria dos dados simulados, nota-se os principais pontos:

1. A verificação de conjunto de dados se mostrou importantíssimo para um melhor entendimento de possíveis problemas da transmissão como por exemplo a desconexão de pinos do dispositivo Xbee.
2. Foi possível amostrar os 4 valores dos sensores fielmente. É interessante explicitar que a transmissão na plataforma Java se mostrou com menos erros que o antigo Supervisório de Sistema.
3. A redução do conjunto de dados de 64 caracteres para 12 caracteres aumentou a eficiência da transmissão de dados pela diminuição do tempo de resposta entre cada recebimento dos dados.
4. A nova interface, mostrada na figura (14), possui interface gráfica simplificada
5. Após o desenvolvimento do novo Supervisório de Sistema, usou-se a função *build* para construir um arquivo executável e assim qualquer integrante do Baja, com ou sem conhecimento de programação, possa manipula-lo.
6. A plataforma foi feita em Java, linguagem usual para sistema supervisório.
7. Foi possível enviar o dado por telemetria para voltar ao boxe. Entretanto, este sinal de aviso deve ser feito na eletrônica no veículo.
8. Foi implementado o alarme sonoro para o estado crítico da temperatura.
9. A árvore de decisão foi implementada e também será implementada na eletrônica do carro para desligamento e acionamento dos sensores posteriormente.
10. O relatório em Excel para análise dos dados de forma organizada.

É importante ressaltar que este trabalho abre portas para inúmeras outras aplicabilidades. Portanto, este trabalho pode ser aplicado em qualquer outro contexto, acadêmico ou não, em que se queria medir dados, envia-los por telemetria para monitoramento e controle.

Esta monografia visou descrever os materiais e detalhes do desenvolvimento desta plataforma e assim, ser uma sugestão para que os futuros integrantes do Baja deem continuidade ao projeto. Acredito que este trabalho acrescentou imensamente ao meu currículo em conhecimentos que me darão base para trabalhar futuramente na área de Automação industrial.

Para trabalhos futuros, a criação de gráficos das variáveis no tempo podem ser feitas como no programa em Matlab. O cruzamento dos dados das variáveis velocidade e temperatura por que já se é sabido que elas estão relacionadas. O calculo integrativo da variável velocidade no tempo para encontrar a distância percorrida. O cruzamento dos dados das variáveis distancia e gasolina para calcular o rendimento do combustível.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] "Quarta revolução industrial: Como o Brasil pode se preparar para a economia do futuro" BBC, 18 agosto 2016 [Online]. Available http://www.bbc.com/portuguese/noticias/2016/01/160122_quarta_revolucao_industrial_mw_ab [Acesso em 18 agosto 2016].

[2] "Etch Abatement Needed at 200mm Fabs to Meet WSC Goals for 2020" Electroiq , 4 Setembro 2016 [Online]. Available http://electroiq.com/petes-posts/category/uncategorized/ [Acesso em 4 Setembro 2016].

[3] ALMEIDA, Hyggo. Revista da Sociedade Brasileira de Computação. Computação Brasil. Porto Alegre, RS. p. 6-8, 29 abril 2015.

[4] "Telemetria" Magneti Marelli, 24 julho 2016 [Online]. Available: https://www.magnetimarelli.com/pt/business_areas/motorsport/excel%C3%AAncias-tecnol%C3%B3gicas/telemetria [Acesso em 24 julho 2016].

[5] "Wintax4" Magneti Marelli , 4 Setembro 2016 [Online]. Available https://www.magnetimarelli.com/pt/business_areas/motorsport/software/wintax4 [Acesso em 4 Setembro 2016].

[6] "Sae brasil" Manual de Identidade Visual, 3 Setembro 2016 [Online]. Available: http://www.saebrasil.org.br/manual/saebrasil_identidade.pdf [Acesso em 3 Setembro 2016].

[7] J. Palm III, William. Introduction to MATLAB® for Engineers University of Rhode Island, p. 10, 2011.

[8] J. E. A. Dias, Eletrônica, instrumentação e telemetria do veículo ufvbaja - Trabalho de conclusão de curso UFV, 2010.

[9] Cursos de linguagem C. Apostila do curso oferecido pelo da UFPR. 24 julho 2016 [Online]. Available: http://www.eletrica.ufpr.br/~rogerio/MSP430/00%20-%20CD%20DO%20ALUNO%20-%20FRAM/04%20-%20APOSTILAS/APOSTILA%20MSP430%20-%20C%20-%20PARTE%20IV.pdf [Acesso em 24 julho 2016].

[10] Cursos de telecomunicação. Apostila do curso oferecido pelo da Coltec-UFMG, PRÁTICA 3, 2007., 24 julho 2016 [Online]. Available: http://www.cpdee.ufmg.br/~lfamaral/tele/pratica03/Prat03A15A1.pdf [Acesso em 24 julho 2016].

[11] Soares, M. J. Microcontrolador: Conectado um Microcontrolador a um PC. Eletrônica Total, número 102 de Setembro/Outubro de 2004.

[12] "Portas Seriais e Comunicação Serial" Automação Livre para Laboratórios de Águas, 24 julho 2016 [Online]. Available: http://www.c2o.pro.br/automacao/x834.html [Acesso em 24 julho 2016].

[13] "Telemetria conceitos e aplicações" Centro de ciências do mar, 14 Setembro 2016 [Online]. Available http://cfrg.ccmar.ualg.pt//documents/Telemetry%20concepts%20and%20applications.pdf [Acesso em 14 Setembro 2016].

[14] "Comunicação de dados para um sistema de telemetria de baixo custo" Dissertação de mestrado, 21 julho 2016. [Online]. Available: http://docs.computacao.ufcg.edu.br/posgraduacao/dissertacoes/2005/Dissertacao_MauricioMarinhoFormiga.pdf [Acesso em 21 julho 2016].

[15] " Zigbee – estudo da tecnologia e aplicação no sistema elétrico de potência" Trabalho de conclusão de curso, 21 julho 2016. [Online]. Available:

http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/886/1/CT_COTEL_2012_2_01.pdf [Acesso em 21 julho 2016].

[16] "Por que Zigbee" fe, 24 julho 2016 [Online]. Available: https://web.fe.up.pt/~ee02055/Relatorio_ZigBee_Hardware.pdf [Acesso em 24 julho 2016].

[17] William H. Hayt, Jr e John A. Buck. "ELETROMAGNETISMO", 6ª Edição, editora LTC, 2003.

[18] Sadiku,Matthew N. O, "Elementos de Eletromagnetismo", 5ª Edição, editora Bookman 2012.

[19] Hutchinson, Chuck, "The ARRL Handbook for Radio Amateurs 2001". ARRL - The national association for Amateur Radio, Edição 2000.

[20] "Redes wireless, parte 4: Antenas e conectores" hardware, 23 julho 2016 [Online]. Available: http://www.hardware.com.br/tutoriais/alcance-antenas-conectores-potencia/ [Acesso em 23 julho 2016].

[21] "Importância da antena" Ibytes, 20 Outubro 2016 [Online]. Available http://www.ibytes.com.br/as-antenas-sao-tao-importantes-quanto-equipamentos-de-ultima-geracao/ [Acesso em 20 Outubro 2016].

[22] "Curso Telecom e Telematica" unisanta, 14 Setembro 2016 [Online]. Available http://professores.unisanta.br/santana/downloads%5CTelecom%5CSistemas_Telecom%5CRadio%5CCurso%20de%20Antenas.pdf [Acesso em 14 Setembro 2016].

[23] "Diferença entre Antenas Omnidirecional, qual devo usar?" Empresa EMSTECH, 20 Outubro 2016 [Online]. Available https://www.youtube.com/watch?v=sVMLSDMKqgA [Acesso em 20 Outubro 2016].

[24] "What is Arduino?" Arduino, 20 julho 2016. [Online]. Available: https://www.arduino.cc/en/Guide/Introduction. [Acesso em 20 julho 2016].

[25] "Algumas aplicações do arduíno para prototipagem eletrônica" Opservices, 23 julho 2016 [Online]. Available: http://www.opservices.com.br/o-que-e-o-arduino/ [Acesso em 23 julho 2016].

[26] "O que é open source?" Canaltech, 23 julho 2016 [Online]. Available: http://canaltech.com.br/o-que-e/o-que-e/O-que-e-open-source/ [Acesso em 23 julho 2016].

[27] "Arduino – Pinagem do Arduino UNO R3" Bod Garage, 20 julho 2016. [Online]. Available: http://bodgarage.repofy.com/?p=959. [Acesso em 20 julho 2016].

[28] "O papel do supervisório no atual contexto tecnológico" Artigo Aquarius Software, 21 julho 2016. [Online]. Available: http://www.aquarius.com.br/Boletim/InTech132_artigo.pdf. [Acesso em 21 julho 2016].

[29] A. P. G. Da Silva, M. Salvador, "O que são sistemas supervisórios?" 20 julho 2016. [Online]. Available: http://www.wectrus.com.br/artigos/sist_superv.pdf. [Acesso em 20 julho 2016].

[30] "Sete benefícios conquistados através da Automação Industrial" Citisystems, 20 julho 2016. [Online]. Available: http://www.citisystems.com.br/sete-beneficios-automacao-industrial/. [Acesso em 20 julho 2016].

[31] "Sistemas Supervisórios" Citisystems, 20 julho 2016. [Online]. Available: http://www.citisystems.com.br/sistemas-supervisorios-automacao-industria/. [Acesso em 20 julho 2016].

[32] "Java brings opportunities to industrial automation" Dr. James Lathrop, 5 janeiro 2001, 19 agosto 2016 [Online]. Available http://www.controleng.com/single-article/java-brings-opportunities-to-industrial-automation/3a8961fbd1fc32a1ca9f7b52f1427c9f.html [Acesso em 19 agosto 2016].

[33] "Java e orientação a objetos" Curso FJ-11, 4 agosto 2016 [Online]. Available: https://www.caelum.com.br/apostila-java-orientacao-objetos/ [Acesso em 4 agosto 2016].

[34] “Antena Omnidirecional 25dbi” mercado livre, 23 julho 2016 [Online]. Available: http://produto.mercadolivre.com.br/MLB-690706942-antena-omnidirecional-25dbi-internet-wireless-p-roteador-_JM [Acesso em 23 julho 2016].

[35] “ XBee Pro Series2 RF module” Seeedstudio, 21 julho 2016. [Online]. Available: http://www.seeedstudio.com/wiki/XBee_Pro_Series2_RF_module. [Acesso em 21 julho 2016].

[36] “XBee Java Library” Digi, 19 agosto 2016 [Online]. Available http://docs.digi.com/display/XBJLIB/XBee+Java+Library [Acesso em 19 agosto 2016].

ANEXOS

A Fig. (15) ilustra um recorte do desenvolvimento do programa feita com dados simulados constantes para teste da primeira versão da plataforma em Java ainda dentro da *IDE*.

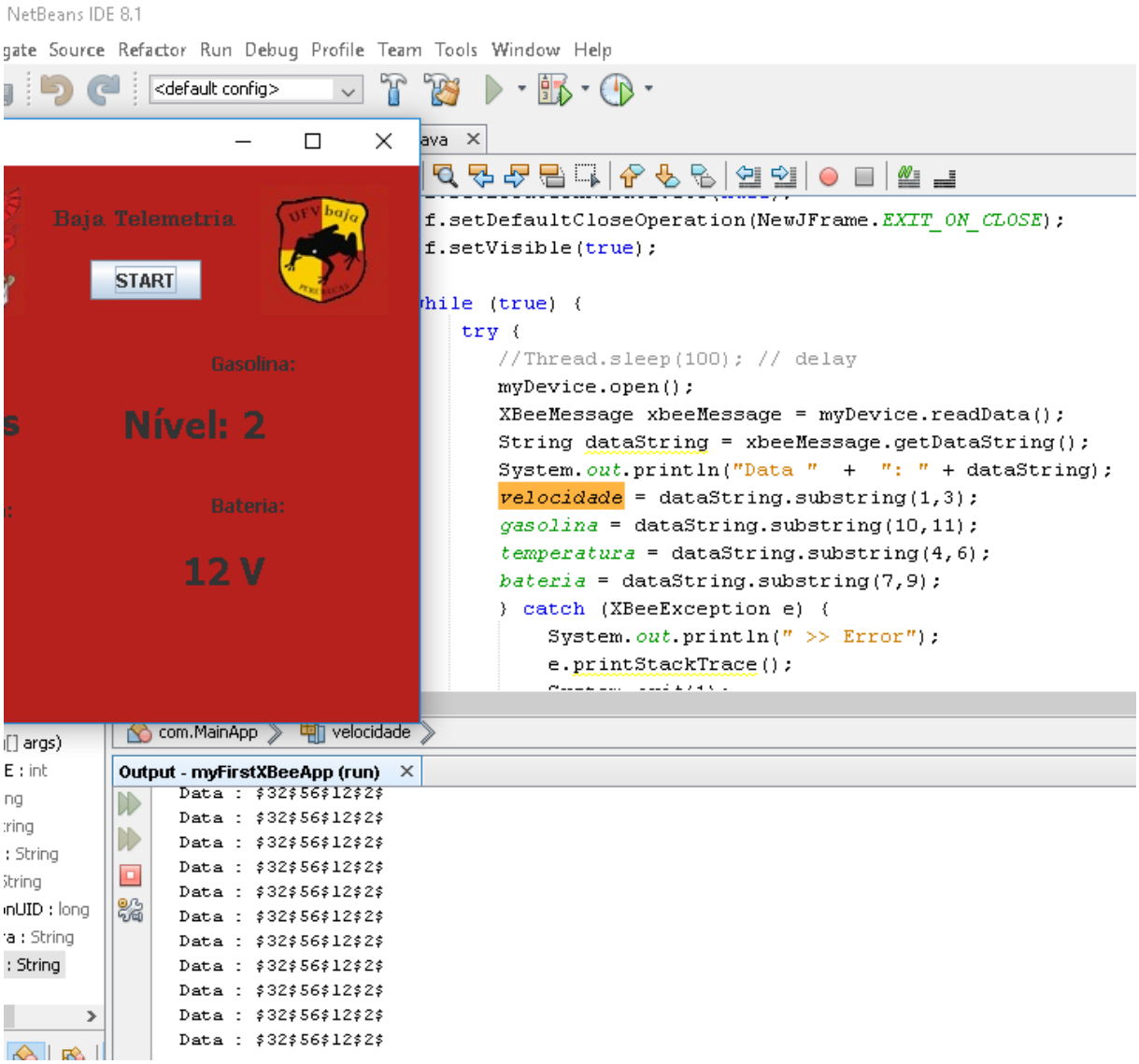

Figura 15. Recorte da programação em Netbeans IDE 8.1.

A Fig. (16) mostra a montagem no protoboard do circuito com PIC e Xbee. Um circuito de Led piscando foi adicionado para verificar o funcionamento da telemetria.

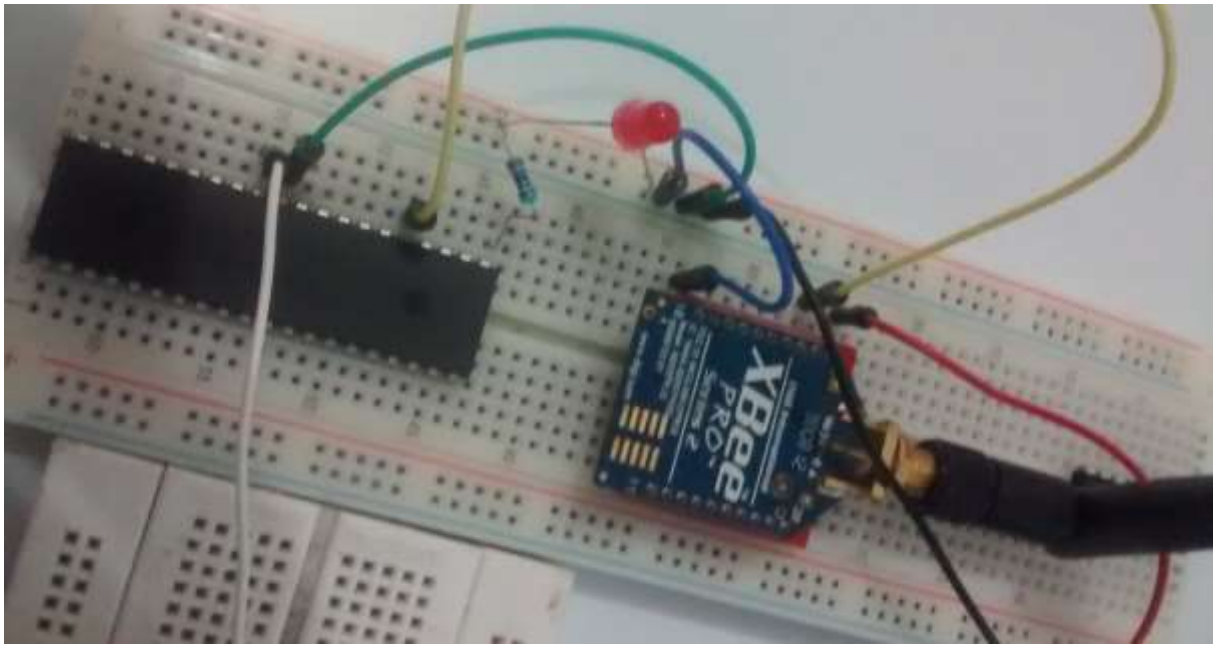

Figura 16. Xbee e pic

Segue abaixo os trechos do código referentes ao envio de dados do PIC ilustrado na Fig. (16). Segue a explicação do código usando o comando de comentário de código "//".

```
// Seta algumas configurações iniciais do PIC
#use delay(clock=2000000)
#use rs232(uart1, baud=9600,parity=N,xmit=PIN_D2,rcv=PIN_D3,bits=8, stream=transmissor)
[…] // […] foi usado para omitir trechos não essenciais ao entendimento da telemetria
// Variáveis
// O conjunto de dados do Matlab possui o formatoabaixo de 64 caracteres
// 'v00$r0000$c0.0$b0$t000.0$h00$m00$i00$s00$d00000$a00000$f0$o0$p1$'
char VEL[4]= {'5', '4', '$'};
char R[6]= {'0','9','0','0','$'};
char COMB[5]={'3','.','0','$'};
char B[3]= {'3','$'};
char TEMP[7]={'0','1','0','.','0', '$'};
char H[4]= {'0','9', '$'};
char M[4]= {'0','9', '$'};
char IN[4]= {'0','9', '$'};
char S[4]= {'1','0', '$'};
char D[7]= {'0','1','0','0','0', '$'};
char A[7]= {'0','1','0','0','0', '$'};
char F[3]= {'3','$'};
char O[3]= {'3', '$'};
char PULSOS [3]={'3','$'};
[...]
// Comunicação serial através da telemetria
void envia(){
    fputc('v',transmissor);
    for (i=0; i<3; i++)  {fputc(VEL[i],transmissor); delay_ms(10);};
    fputc('r',transmissor);
    for (i=0; i<5; i++)  {fputc(R[i],transmissor); delay_ms(10);};
    fputc('c',transmissor);
    for (i=0; i<4; i++)  {fputc(COMB[i],transmissor); delay_ms(10);};
    fputc('b',transmissor);
```

```
    for (i=0; i<2; i++)  {fputc(B[i],transmissor); delay_ms(10);};
    fputc('t',transmissor);
    for (i=0; i<6; i++)  {fputc(TEMP[i],transmissor); delay_ms(10);};
    fputc('h',transmissor);
    for (i=0; i<3; i++)  {fputc(H[i],transmissor); delay_ms(10);};
    fputc('m',transmissor);
    for (i=0; i<3; i++)  {fputc(M[i],transmissor); delay_ms(10);};
    fputc('i',transmissor);
    for (i=0; i<3; i++)  {fputc(IN[i],transmissor); delay_ms(10);};
    fputc('s',transmissor);
    for (i=0; i<3; i++)  {fputc(S[i],transmissor); delay_ms(10);};
    fputc('d',transmissor);
    for (i=0; i<6; i++)  {fputc(D[i],transmissor); delay_ms(10);};
    fputc('a',transmissor);
    for (i=0; i<6; i++)  {fputc(A[i],transmissor); delay_ms(10);};
    fputc('f',transmissor);
    for (i=0; i<2; i++)  {fputc(F[i],transmissor); delay_ms(10);};
    fputc('o',transmissor);
    for (i=0; i<2; i++)  {fputc(O[i],transmissor); delay_ms(10);};
    fputc('p',transmissor);
    for (i=0; i<2; i++)  {fputc(PULSOS[i],transmissor); delay_ms(10);};

    // teste de funcionamento da telemetria pelo led vermelho
    output_high(PIN_B7);
    delay_ms(200);
    output_low(PIN_B7);
    delay_ms(300);
}
[...]
 while (true) {
    envia();  // delay médio de 22ms
    delay_ms(20);
    }
}
```

Segue abaixo os trechos do código referentes ao envio e recebimento de dados do arduino. Segue a explicação do código usando o comando de comentário de código "//".

```
#include <SoftwareSerial.h> // biblioteca referente a comunicação serial

char c1[15] = "$32$56$12$2$"; // conjunto de dados simulados 1 para envio
char c2[15] = "$00$00$12$2$"; // conjunto de dados simulados 2 para envio
String c3 = "            "; // conjunto de dados para leitura

void setup()  {     // Setup configura o arduino
  Serial.begin(9600); // define a baud rate em 9600 da comunicação serial
  pinMode(8, OUTPUT); // led vermelho 1
  pinMode(9, OUTPUT); // led vermelho 2
  pinMode(10, OUTPUT); // led vermelho 3
  pinMode(11, OUTPUT); // led vermelho 4
  pinMode(12, OUTPUT); // led verde
}

void loop()  { // Programa para repetição (Loop)
 Serial.println(c1); // envia dados 1
 delay(400); // espera 0.4 segundos
```

```
 Serial.println(c2); // envia dados 2
 delay(400); // espera 0.4 segundos

 [...]

 c3 = Serial.readString();
 if (Serial.readString()== "$1$0$0$0$0$"){
  digitalWrite(8, HIGH); // acende led 1
  delay(1000); // espera 1 segundo com led 1 acesso
 }

 [...]

 digitalWrite(8, LOW); // desliga led 1
 delay(200); // espera 0.2 segundos
}
```

A Fig. (17) mostra a montagem no protoboard do circuito com arduino e Xbee. Cada um dos 4 leds vermelhos mostrados representaram o desligamento de um dos 4 sensores do carro e o led verde representa a luz de alerta para voltar aos boxes. A luz verde do arduino representa que ele está ligado e as Luzes amarelos RX e TX significam respectivamente que o arduino está lendo e transmitindo dados.

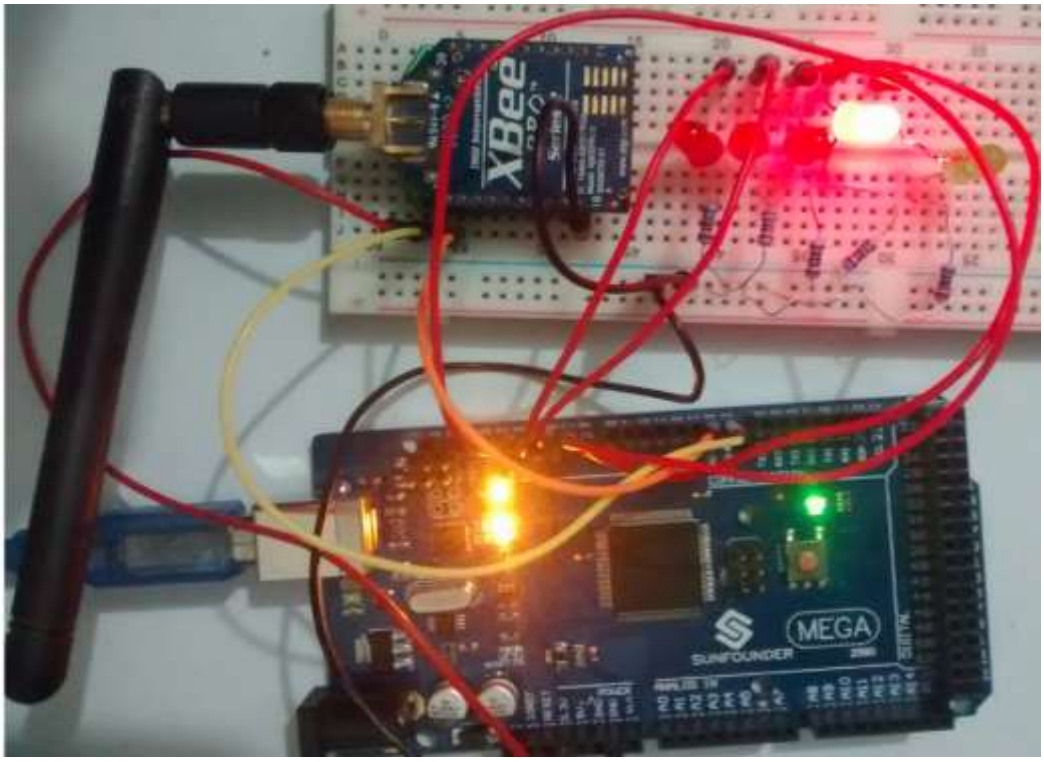

Figura 17. Arduino e xbee

Como explicitado na seção 2.5.3, há uma série de bibliotecas prontas em Java para diversos instrumentos da indústria. A Empresa *Digi*, que produz o dispositvo Xbee fornece, em seu site [36], o download das bibliotecas apropriadas para a implementação do codigo em Java para o recebimento e envio dos dados deste trabalho.

Segue abaixo uma amostra de trechos do código referentes a biblioteca fornecida pela Empresa Digi para apenas recebimento de dados. Segue a eplicação do código usando o comando de comentário de código "//".

```
// Biblioteca para receber dados do xbee
import com.digi.xbee.api.XBeeDevice;
import com.digi.xbee.api.exceptions.XBeeException;
import com.digi.xbee.api.models.XBeeMessage;
import com.digi.xbee.api.listeners.IDataReceiveListener;

[...]

// cria variável privada stática texto com valor referente a porta serial USB COM5
private static String PORT="COM5";
// cria variável privada stática inteiro com valor referente a taxa de transmissão serial 9600 (bits por segundo)
private static final int BAUD_RATE = 9600;
// cria variável stática texto dataString com valor inicial "vazio"
static String dataString = "vazio";

[...]

// Cria objeto myDevice da classe XBeeDevice e o inicializa usando comando new
XBeeDevice myDevice = new XBeeDevice(PORT, BAUD_RATE);
//  Leitura do conjunto de dados serial
XBeeMessage xbeeMessage = myDevice.readData();
//  Extrai o texto dos dados e armazena na variável dataString
String dataString = xbeeMessage.getDataString();
```