UNIVERSIDADE FEDERAL DO PARANA

Setor de Tecnologia

Departamento de Engenharia Elétrica

Thiago Kenji Batisti Sato

Sistema Automotivo de Notificação de Acidente

Curitiba

2010

UNIVERSIDADE FEDERAL DO PARANA
Setor de Tecnologia
Departamento de Engenharia Elétrica
Thiago Kenji Batisti Sato

Sistema Automotivo de Notificação de Acidente

Monografia apresentada à disciplina TE105 - Projeto de Graduação, como requisito parcial à conclusão do Curso de Engenharia Elétrica da Universidade Federal do Paraná.

Orientador: Professor Alessandro Zimmer

Curitiba

2010

UNIVERSIDADE FEDERAL DO PARANA
Setor de Tecnologia
Departamento de Engenharia Elétrica
Thiago Kenji Batisti Sato

TERMO DE APROVAÇÃO

Sistema Automotivo de Notificação de Acidente

Monografia aprovada como requisito parcial à conclusão do curso de Engenharia Elétrica da Universidade Federal do Paraná pela seguinte banca examinadora:

Prof. Dr. Alessandro Zimmer - Orientador

Prof. Dr. Eduardo Parente Ribeiro

Prof. Dra. Giselle Ferrari

Curitiba

2010

AGRADECIMENTO

Agradeço a Deus.

Aos meus pais, Nelma Batisti Sato e Jorge Sato, pelo seu amor. Por todo esforço dedicado a minha educação, Além do apoio e incentivo durante toda minha vida.

A Universidade Federal do Paraná e aos professores pelo aprendizado e experiência vivida nesta grande instituição.

Ao meu orientador, professor Alessandro Zimmer, pelo auxilio e orientação neste projeto.

RESUMO

Este documento descreve um sistema de notificação automática de acidente de trânsito. O objetivo deste sistema é minimizar o número de fatalidades diminuindo a espera pelo atendimento médico de emergência. Isto é otimizado através da pronta notificação e fornecendo informações úteis. A mensagem de notificação pode conter informações como: hora, local, velocidade de colisão, identificação do veículo, dados do motorista (nome, tipo sanguíneo e pessoa a contatar). O acidente é identificado pelo recebimento de uma mensagem da unidade de controle do *airbag*, através do barramento CAN, notificando o disparo deste.

O sistema é constituído por um microcontrolador, um módulo GPS, uma interface de protocolo CAN e um módulo de comunicação com o exterior. Para o desenvolvimento do sistema optou-se pela solução Arduino, uma plataforma *open-software* e *open-hardware*.

Palavras-chave: Acidente de trânsito, mortalidade, colisão, CAN, Arduino, *Airbag*.

ABSTRACT

This document describes a car crash automatic notification system. The objective of the system is to minimize the number of fatalities by shorting the waiting for the urgent medical assistance. This is optimized by a prompt notification and providing relevant information. The notification message may contain information such as time, location, crash speed, vehicle and driver's information (name, blood type, and person to contact). The accident is identified through a CAN (Controller Area Network) message sent from the ACU (Airbag Control Unit) notifying the airbag deployment.

signalizing the deployment.

The system is constituted by a microcontroller, a GPS module, a CAN interface and a communication interface with the outside. For system development the Arduino solution was chosen, an open-software and open-hardware platform.

Keywords: Car accidents, collision, car crash, road mortality rate, Airbag, CAN, Arduino.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 Introdução

A importância do automóvel para a sociedade moderna é indiscutível. Difícil imaginá-la sem a indústria automotiva. O progresso decorrente à sua invenção é visível, contudo existem problemas conseqüentes como a poluição, congestionamentos e acidentes. Para gerenciar o trânsito, governos instituíram políticas e normas (por exemplo, o Código Nacional de Trânsito Brasileiro). Essas políticas incluem regras de trânsito, formação dos condutores, fiscalização, campanha de conscientização e serviços de emergência.

Em resposta ao grande número de mortes nos acidentes e à demanda da sociedade, a indústria automotiva desenvolveu sistemas de segurança veicular. Inicialmente os dispositivos passivos tais como o cinto de segurança e o *airbag* nos anos 70. Seguidos por sistemas de segurança ativos como o ABS e direção assistida. Os sistemas de segurança de integração intraveicular, com comunicação entre veículos e com o ambiente são o próximo passo, pois até então cada sistema é isolado e independente. Por exemplo, poderemos ter veículos se comunicando com uma central que gerencie o tráfego, indicando o melhor caminho e velocidade de translado otimizando tempo de percurso e evitando engarrafamentos. Outro exemplo, um semáforo ou um outro veículo poderia alertar sobre situações de risco, como acidentes, objetos ou pedestres na pista. Entretanto, por mais que se tente evitar, acidentes sempre ocorrerão.

Figura 1 – Taxa de mortalidade no trânsito do Brasil [3]

Segundo a Organização Mundial de Saúde (OMS), o acidente de trânsito é a principal causa de morte entre a população jovem, de 15 a 29 anos, no mundo[2]. Ela está entre as três principais causas na população entre 5 e 44 anos. O problema é tão grave que a ONU (Organização das Nações Unidas) declarou esta década de 2010 como a “Década de Ações para Segurança Viária no mundo”[1]. Na tentativa de sensibilizar os governos e a população mundial para o problema e para desenvolver ações de combate a estas fatalidades.

Em caso de acidente, um ágil atendimento médico é crucial. Para tanto, a pronta notificação do acidente com informações precisas é importantíssima. Com essa visão a Comissão Européia trabalha atualmente no projeto de um sistema de notificação automática de acidentes[5]. O projeto *eCall* será um sistema único para toda a União Européia. O principio de funcionamento do sistema está ilustrado na figura 02, sua implantação é prevista para 2014.

Este projeto é inspirado neste conceito. O objetivo é estudar e desenvolver um protótipo que reconheça uma colisão e envie uma mensagem com as coordenadas do acidente. A idéia é que uma central de emergência seja alertada para um pronto atendimento, assim aumentando as chances de sobrevivência dos acidentados.

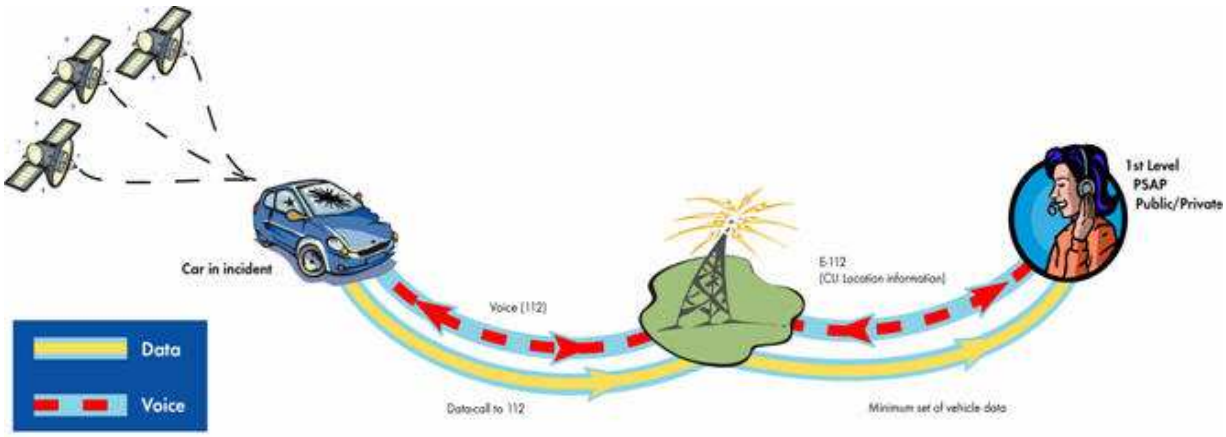

Figura 2 – Projeto eCall [5]

2 Visão Geral

O sistema proposto é composto 4 blocos, conforme exposto na figura 3:

- Um microcontrolador;
- Um módulo GPS;
- Uma interface CAN;
- Um celular, para a comunicação externa.

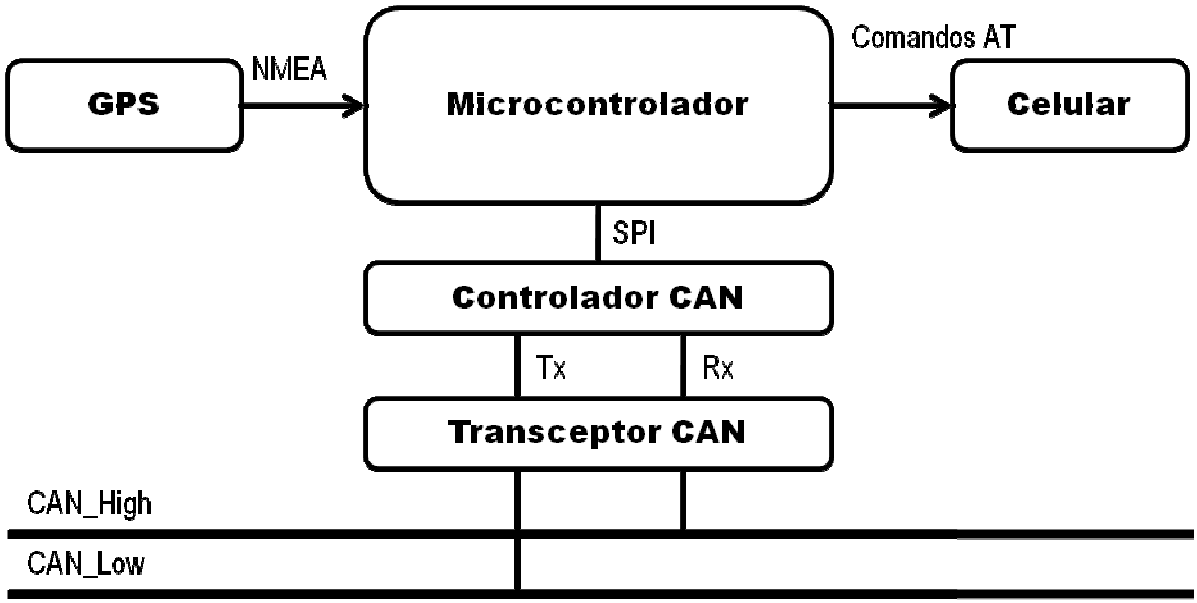

Figura 3 – Diagrama de blocos

A detecção de uma colisão pode ser feita através de sensores de impacto e de desaceleração. Porém, pensando em custo, num impacto mínimo sobre a arquitetura e nas funções disponíveis, o sistema de airbags é aproveitado para realizar a detecção. É fato que hoje nem todos os carros possuem o sistema, mas a partir de 2014 todo carro produzido no Brasil terá airbags frontais de série segundo lei sancionada pelo presidente em 19/03/2009. As vantagens são os ganhos no custo e espaço na reutilização de sensores e em eficiência, pois o calculador do sistema de airbags possui um algoritmo especializado na detecção de colisão para evitar disparos desnecessários. Nosso sistema receberá através do barramento CAN a mensagem do disparo do airbag fornecida pela ACU.

O barramento CAN é um barramento de comunicação criado pela BOSCH. Ele está cada vez mais presente nos automóveis graças às vantagens que apresenta frente a problemas dos sistemas elétricos e eletrônicos do veículo. É importante dizer que cada montadora mantém em sigilo o seu dicionário de dados, o conjunto de mensagens trocadas dentro do veículo. O que impossibilita decifrar uma mensagem sem seu identificador. Dito isto, o disparo do airbag será simulado por uma mensagem CAN.

Detectada a colisão, é necessário reunir as informações necessárias para o envio da mensagem ao serviço de emergência. Através de um receptor GPS amostramos continuamente as coordenadas do veículo. Desta forma, obtêm-se as coordenadas do local da colisão. Além da localização e a identificação do veículo (número da placa ou o RENAVAM), pode-se transmitir outras informações úteis ao atendimento. Por exemplo, a velocidade do veículo no momento da colisão ou o tipo sanguíneo do condutor.

Os GPS estão cada vez mais populares e acessíveis. Os veículos de mais alta gama têm os navegadores GPS de fábrica. Está em fase de implantação aqui no Brasil o sistema de rastreamento obrigatório em todo veículo produzido no país. O interessante é o módulo GPS e GPRS, o que facilita a implantação de nosso sistema, caso haja uma interface de comunicação entre os dois, ou a integração de nossa funcionalidade a este sistema.

Para a comunicação com o exterior, optou-se por utilizar um celular como solução. Para o funcionamento de tal sistema é necessário que a central receptora esteja preparada para receber e interpretar a mensagem. No entanto, neste trabalho omitiremos a central, nos restringindo a enviar uma mensagem SMS com as informações. Ao invés de uma mensagem SMS poderíamos fazer uma chamada ou mesmo ambos, o que permitiria confirmar se não é um engano ou se os ocupantes do veículo estão conscientes.

3 FUNDAMENTACÃO TEÓRICA

3.1 Sistema de *Airbags*

Airbag é um sistema automotivo de segurança passiva que funciona a partir de um conceito simples: uma bolsa de ar para absorver o impacto de uma colisão. Quando o carro sofre um grande impacto, uma bolsa de ar é rapidamente inflada para amortecer o choque, evitando que motorista e passageiro sofram danos físicos principalmente no rosto, peito e coluna.

O coração do sistema é a unidade de controle Airbag (ACU - figura 4), um tipo específico de calculador que monitora um grande número de sensores no interior do veículo. Incluindo acelerômetros, sensores de impacto, sensores laterais, pressão, sensores de velocidade de rodas, giroscópios, sensores de pressão de frenagem e sensores de ocupação. Quando o requisito limiar é atingido a ACU aciona a ignição de um propelente gerador de gás para inflar rapidamente a bolsa de nylon e com o ocupante do veículo pressionando a bolsa, o gás escapa de uma forma controlada através de pequenos orifícios de ventilação.

Figura 4 – Diagrama de blocos do sistema de airbag

3.1.1 Unidade de Controle Airbag - ACU

Os sinais dos diversos sensores alimentam a unidade de controle de airbag, a qual determina o ângulo de incidência, a severidade e a força do impacto, juntamente com outras variáveis. Do resultado desses cálculos, a ACU aciona os airbags necessários e com velocidade de acordo com a intensidade da colisão. A unidade trabalha com algoritmos de disparo de airbag cada vez mais complexos na tentativa de reduzir os disparos desnecessários e para adaptar a velocidade de implantação às condições da colisão.

Normalmente, a decisão de acionar um airbag em uma colisão frontal é feita dentro de 15 a 30 milissegundos após o início da colisão, e ambos os airbags frontais são totalmente inflados dentro de cerca de 60-80 milissegundos após o primeiro momento de contato do veículo. Se um airbag é acionado tarde ou lento demais, o risco de ferir os ocupantes no contacto com o airbag aumenta. Por estas razões se busca um algoritmo ótimo. Algoritmos experimentais levam em conta fatores como o peso do ocupante, a localização do assento, uso do cinto de segurança, e até tentativa de determinar se um assento para bebê está presente.

A ACU também é uma unidade de diagnóstico que monitora a prontidão do sistema de airbag. Se for identificado algum problema, uma luz de alerta no painel avisa o motorista que deve examinar o airbag. Ela também sinaliza às outras ECU o disparo do airbag. A maioria das unidades de diagnóstico contém um dispositivo que armazena uma quantidade suficiente de energia elétrica para ativar o airbag, no caso da bateria do veículo ser destruída no início da colisão. A figura 5 mostra um exemplo de ACU.

Figura 5 – ACU, unidade de controle de airbag

3.2 *Controller Area Network* – CAN

CAN (*Controller Area Network*) é um protocolo de comunicação serial desenvolvido inicialmente pela BOSCH em 1986 para aplicações automotivas. Hoje ela está na sua versão 2.0 [6] e é um padrão mundial ISO11898 (*International Standardization Organization*) desde 1994. O protocolo CAN dispensa *host*, pois o acesso ao meio de transmissão tem base na técnica de CSMA/CR (*Carrier Sense Multiple Access/Collision Resolution*). Isso significa que sempre que ocorrer uma colisão entre mensagens, o acesso é assegurado através de arbitragem bit a bit à mensagem de maior prioridade. Prosseguindo a retransmissão da mensagem sem sua perda.

As características básicas do protocolo CAN são as seguintes: 8 *bytes* de dados, velocidade de até 1Mbps, priorização de mensagens, recepção *multicast* com sincronização, arbitragem *bitwise*, detecção de erro, sinalização e retransmissão automática. Além da imunidade ao ruído, redução no cabeamento e flexibilidade na configuração da rede. Todas estas características propiciam simplicidade, alta confiabilidade e segurança, Além de baixo custo. A afirmação e o crescimento do protocolo CAN estão fortemente baseados na organização dos fabricantes em torno de associações, como a CiA (*CAN in Automation*), assim como na existência de uma série de software para o desenvolvimento, simulação, configuração e monitoramento de aplicações, e na disponibilidade de hardware na forma de placas de controle e circuitos integrados.

FABRICANTE	***CONTROLADOR***
Infineon	***81C90***
Intel	*82527*
Microchip	***MCP2515***
Philips	*SJA1000*

Tabela 1 – Controladores CAN

O protocolo é usualmente implementado em um controlador na forma de um circuito integrado, mas também se encontram no mercado microcontroladores com CAN integrados. A tabela 1 contém exemplos de controladores CAN disponíveis hoje no mercado. Os controladores CAN e os microcontroladores com a interface CAN são fabricados por um grande número de indústrias, como Intel, Motorola, Philips, Siemens e Texas Instruments. A utilização do protocolo CAN na indústria automobilística

resultou numa produção em grande escala de controladores CAN, o que ajudou na popularização e acesso à tecnologia.

3.2.1 Protocolo CAN e Modelo OSI

De acordo com o modelo OSI/ISO, o padrão CAN 2.0 especifica as duas primeiras camadas: a camada física e a camada de enlace. Conforme mostra a figura 6.

A camada física define o nível de sinal de transmissão e a representação dos bits recebidos através do barramento, Além de ser responsável pelo ajuste de tempo de bit (*bit timing*) e sincronização entre os nós que participarão do processo de arbitragem (que define qual nó terá acesso ao meio físico para transmitir sua mensagem).

A camada de enlace de dados divide-se em controle de link lógico (LLC – *Logic Link Control*) que é responsável pelo controle de aceitação de mensagens que são recebidas e notificação de sobrecarga do nó conectado a rede CAN, e controle de acesso ao meio (MAC – *Medium Access Control*) que mantém e oferece condições básicas para operação de um sistema, além de detecção e sinalização de erros, reconhecimento de mensagens recebidas e desencapsulamento.

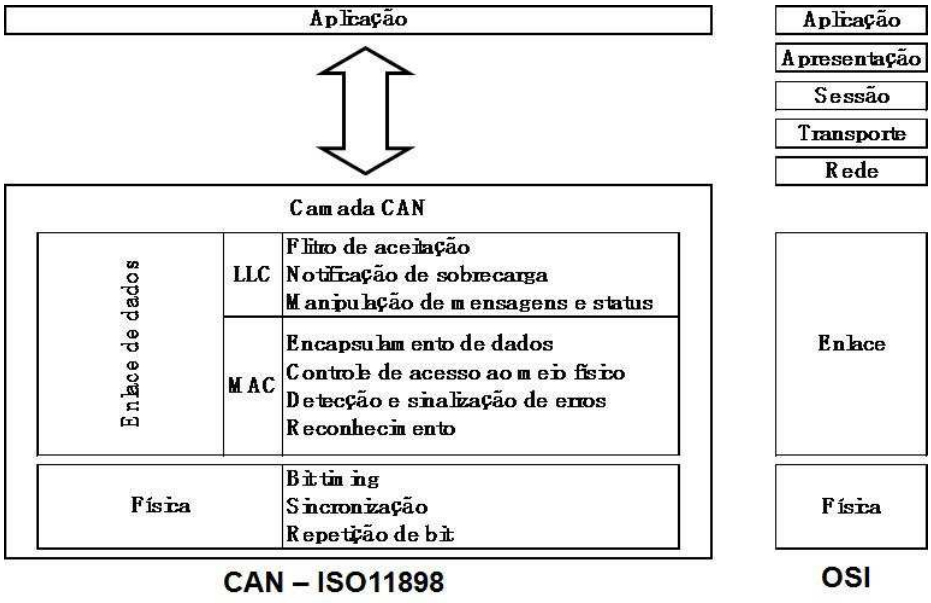

Figura 6 – CAN vs OSI [7]

3.2.2 Mensagens CAN

O Protocolo CAN 2.0 tem duas versões, 2.0A e 2.0B. Ambas são compatíveis com a versão CAN1.0, mas a diferença entre as versões é o tamanho do identificador

das mensagens. As versões 1.0 e 2.0A com 11bits (figura 7) e a versão 2.0B com 29bits (figura 8). Um universo de 2.048 e 536.870.912 mensagens respectivamente.

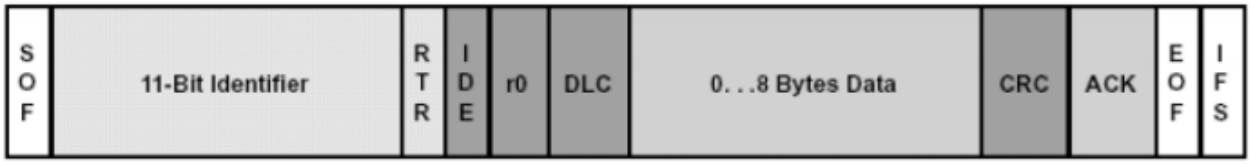

Figura 7 – CAN 2.0A, frame padrão [7]

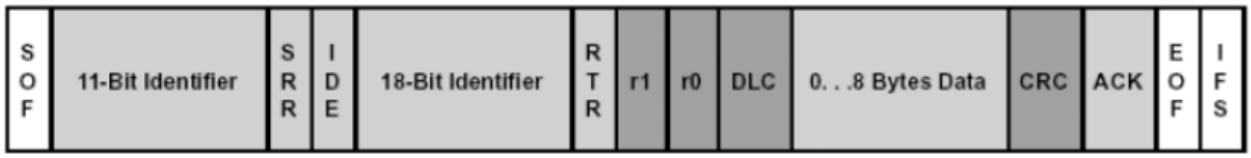

Figura 8 – CAN 2.0B, frame estendido [7]

Os bits que compões as mensagens CAN, representadas nas figuras 7 e 8, são descritos a seguir:

SOF – *Start of Frame*: inicio da mensagem que sincroniza os nós.

Identificador: define a identificação e a prioridade da mensagem. O menor valor binário possui a maior prioridade de transmissão.

RTR – *Remote Transmission Request*: indica a requisição de transmissão remota.

SRR (exclusivo CAN 2.0B) – *Substitute Remote Request*: substitui o RTR do frame padrão.

IDE – *Identifier Extension*: indica se é um frame padrão ou estendido.

r0 e r1 (exclusivo CAN 2.0B): reservado.

DLC – *Data Length Code*: número de bytes de dados a transmitir.

Dados: os dados da mensagem.

CRC – *Cyclic Redundancy Check*: detecção e correção de erro. Detecta até 6 erros de bit simples.

ACK – *Acknowledge*: o nó que recebe uma mensagem correta subscreve esse bit recessivo na mensagem original com um bit dominante, sinalizando ao transmissor o recebimento.

EOF – *End of Frame*: o fim do frame desabilita o *bit stuffing* e indica erro caso ocorra.

IFS – *Inter Frame Space*: tempo para o controlador CAN armazenar a mensagem recebida.

3.2.2.1 Frame de dados

Frame com a função de transmissão da carga útil dentro da rede CAN. O campo DLC varia de 0 a 8, conforme o dada transmitido.

3.2.2.2 Frame remoto

Caracterizado pelo bit RTR, este *frame* solicita uma dada mensagem à rede. Não contem bytes de dados.

3.2.2.3 Frame de erro

Ele tem a função de notificar um erro no recebimento de um pacote, qualquer nó pode gerá-lo. O frame possui dois campos, flag de erro e delimitador que informa o tipo de erro e ativa a comunicação após o erro, respectivamente conforme figura 9.

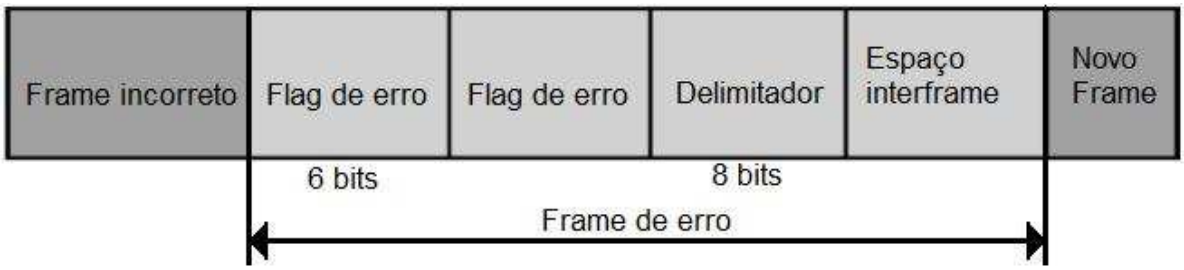

Figura 9 – Frame de erro [7]

3.2.2.4 Frame de sobrecarga

Sua função e formato são similares ao frame de erro. Ele sinaliza a sobrecarga de um nó, impossibilitando-o de receber nenhuma mensagem. O transmissor precisa aguardar o próximo processo de arbitragem.

3.2.3 Arbitragem

Uma das características mais importantes do CAN é o esquema de arbitração binária (*bitwise arbitration*) que fornece uma boa maneira de resolver colisão de mensagens. Sempre que dois nós começam a transmitir mensagens ao mesmo tempo, o

mecanismo de arbitragem garante que a mensagem de mais alta prioridade será enviada. Isso é conseguido pela definição de dois níveis de barramento chamados recessivo e dominante.

Um nível dominante sempre sobrescreve um nível recessivo. Assim, enquanto um nó está enviado uma mensagem, ele compara o nível do bit transmitido com o nível monitorado no barramento. Se um nó tenta enviar um nível recessivo e detecta um dominante, ele perde a arbitragem e interrompe o processo de transmissão.

3.3 Mensagem SMS

O SMS (*Short Message Service*), é uma forma de comunicação via mensagem de texto entre telefones celulares. A primeira mensagem SMS foi enviada pela rede GSM Vodafone no Reino Unido em 3 de dezembro de 1992, de um PC para um telefone. O texto da mensagem era "*Merry Christmas*". Os padrões GSM e SMS foram originalmente desenvolvidos pela ETSI (*European Telecommunications Standards Institute*). Atualmente, o 3GPP (*Third Generation Partnership Project*) é responsável pelo desenvolvimento e manutenção dos padrões.

Hoje o SMS é um sucesso em todo o mundo. Além de texto, as mensagens SMS também podem carregar dados binários. Uma grande vantagem do SMS é que ele é compatível com 100% dos celulares GSM. E quase todos os planos das operadoras incluem serviço de mensagens SMS barato. O número de mensagens SMS trocadas todos os dias é enorme (4,1 trilhões de mensagens enviadas em 2008.). O que faz dele atualmente uma das mais importantes fontes de receita das operadoras móveis. (Uma indústria global de mais de 81 bilhões de dólares em 2006).

3.3.1 Comandos AT

Os comandos AT são instruções usadas no controle de modem. Eles têm este nome porque cada comando se inicia com os caracteres AT, uma abreviação de "*attention*". Muitos dos comandos que são usados para controle de modem dial-up também são suportados por modem GSM/GPRS e telefones celulares. Alguns comandos Hayes estão na tabela 2, este são conhecidos como comandos AT básicos.

COMANDO	***FUNCÃO***
ATD	*Discar*
ATA	*Atender à chamada*
ATH	*Desligar*
ATO	*Voltar ao estado de dados on-line*

Tabela 2 – Exemplo de comandos AT básicos

Além do conjunto de comandos AT básicos, modems GSM/GPRS e telefones celulares suportam um conjunto de comandos AT específicos para tecnologia GSM, que incluem comandos relacionados ao SMS, alguns deste estão na tabela 3.

COMANDO	***FUNCÃO***
AT+GMM	***Informação do modelo do aparelho***
AT+MODE?	***Mostra atual modo de operação do***

	aparelho
AT+CLAC	***Lista todos os comandos AT suportados***
AT+CMGF	***SMS para modo texto***
AT+CMGW	***Escreve um SMS na área de armazenamento de mensagens***
AT+CMSS	***Manda o SMS a partir da área de armazenamento de mensagens***

Tabela 3 – Exemplo de comandos AT estendidos

O ponto de partida é o prefixo "AT" que informa ao modem sobre o início de uma linha de comando. O prefixo AT não faz parte do nome do comando AT. Por exemplo, D e +CMGS são os reais nomes dos comandos ATD e AT+CMGS.

Fabricantes de telefones celulares geralmente não implementam todos os comandos AT. O comportamento aos comandos AT pode ser ligeiramente diferente daquele definido em norma. Em geral, os modems GSM/GPRS projetado para aplicações sem fio têm um melhor suporte de comandos AT do que celulares.

3.4 *Global Positioning System* - GPS

O Sistema de Posicionamento Global (GPS) é um sistema de navegação global por satélite que fornece informações confiáveis de localização e hora a todo momento e em qualquer lugar do globo terrestre, contanto que 4 ou mais satélites estejam visíveis ao receptor. Em outras palavra, é necessária a recepção de sinais de, no mínimo, 4 satélites diferentes. O GPS foi idealizado e realizado originalmente com uma constelação de 24 satélites pelo Departamento de Defesa dos Estados Unidos (DoD) para superar as limitações dos sistemas de navegação anterior. O sistema é mantido pelo governo dos Estados Unidos sendo livre o acesso a qualquer pessoa com um receptor GPS.

O GPS tornou-se uma ferramenta muito útil e encontrou um uso amplo nos transportes, agricultura, navegação, comércio, pesquisas cientificas, mapeamentos, rastreamento e vigilância. A precisão da hora do GPS facilita atividades do dia-a-dia como operações bancárias, operações de celulares, e até mesmo controle de redes de energia. Agricultores, cartógrafos, geólogos e incontáveis outros executam o seu trabalho de forma mais eficiente, segura, econômica e precisa.

3.4.1.1 Trilateração

O receptor utiliza as mensagens que recebe para determinar o tempo de trânsito de cada mensagem e calcula as distâncias de cada satélite. Estas distâncias junto com a localização dos satélites são usadas para fazer a trilateração através de um algoritmo para calcular a posição do receptor. Informações derivadas, como direção e velocidade, são calculadas a partir de mudanças de posição.

Três satélites podem parecer suficientes para encontrar a posição, já que o espaço tem três dimensões e uma posição perto da superfície da Terra pode ser assumida. No entanto, mesmo um erro de relógio muito pequena multiplicada pela velocidade da luz, a velocidade na qual os sinais de satélite se propagam, resulta em um grande erro de posicionamento. Portanto, os receptores usam quatro ou mais satélites para calcular a localização do receptor e hora. Algumas aplicações especializadas fazem uso da hora, que incluem sincronização de clock, temporização de sinais de trânsito e sincronização de estações de comunicação.

3.4.2 Estrutura do sistema

O sistema GPS é composto por três segmentos principais. São eles: o segmento espacial, o segmento de controle e o segmento de usuários. A Força Aérea Norte-Americana desenvolve, mantém e opera os segmentos espacial e de controle.

O segmento espacial é composto pelos satélites GPS em órbita e o segmento de controle por todas as estações e antenas de controle e monitoramento. O segmento do usuário é composto de centenas de milhares de usuários militares do serviço seguro de posicionamento GPS de precisão, e dezenas de milhões de usuários civis, comerciais e científicos do Serviço Padrão de Posicionamento.

3.4.2.1 Mensagem do satélite GPS

Cada satélite GPS continuamente transmite uma mensagem de navegação, a uma taxa de 50 bits por segundo. A tabela 4 mostra a estrutura de *frame*. Cada mensagem completa é composta de *frames* de 30 segundos, agrupamentos distintos de 1.500 *bits* de informação. Cada quadro é subdividido em 5 *sub-frames*, conforme quadro abaixo, de comprimento de 6 segundos e com 300 *bits* cada. Cada frame se inicia precisamente no minuto ou meio minuto indicado no relógio atômico de cada satélite.

SUB-FRAME	***DESCRICÃO***
1	***Clock do satélite / Relação temporal***
2 – 3	***Efemérides***
4 – 5	***Componente Almanaque***

Tabela 4 – Frame de mensagem GPS

A primeira parte da mensagem codifica o número da semana e a hora na semana, bem como os dados sobre o status do satélite. As efemérides prevêem a órbita precisa para o satélite. O almanaque contém a órbita grosseira e informações de status para todos os satélites da rede, bem como dados relacionados à correção de erros.

3.4.3 Protocolo NMEA

O protocolo NMEA é uma combinação da especificação elétrica e de dados para comunicação entre dispositivos eletrônicos navais, inclusive o GPS. Definida e controlada pela Associação Nacional de Eletrônicos Navais Norte-Americana (NMEA), ele é um protocolo de comunicação serial e todas as mensagens são compostas de caracteres ASCII. A norma define o conteúdo de cada tipo de mensagem na camada de

aplicação, à exemplo da mensagem na figura 10, para que todos os ouvintes possam analisá-las e utilizá-las em suas aplicações.

Apesar de ser um protocolo proprietário, muitas referências têm sido compiladas a partir de registros públicos. O que permite trabalhar com ele sem violar as leis de propriedade intelectual.

Figura 10 – Exemplo de trama NMEA0183

As mensagens NMEA se estruturam da seguinte forma:

- O caractere inicial de toda mensagem é o cifrão;
- Os próximos cinco caracteres identificam o locutor (dois caracteres) e do tipo de mensagem (três caracteres);
- Todos os campos de dados que seguem são separadas por vírgula;
- O caractere que precede o último caráter de dados é um asterisco;
- O asterisco é imediatamente seguido por um *Checksum* de dois algarismos que representam um número hexadecimal;
- <CR> <LF> termina a mensagem.

4 ESPECIFICAÇÃO DO *HARDWARE*

4.1 Microcontrolador

Um microcontrolador (por vezes abreviado µC, uC ou MCU) é um pequeno computador em um único circuito integrado contendo um núcleo processador, memória e periféricos de entrada/saída programáveis. Memória de programa, em forma de *flash* ou OTP ROM (O*ne-Time Programmable Read-Only Memory*), também é muitas vezes incluída no chip, bem como tipicamente uma quantidade pequena de memória RAM (*Random Access Memory*). Microcontroladores são projetados para aplicações embarcadas, em contraste com os microprocessadores usados em computadores pessoais ou outros aplicativos de uso geral.

No desenvolvimento do projeto foi utilizado o Arduino Duemilanove que possui um microcontrolador ATMEGA328.

4.1.1 Arduino

Arduino é uma plataforma de prototipagem eletrônica open-source baseada em hardware e software flexível e fácil de usar. A linguagem de programação é baseada em C/C++. Ele é destinado ao público em geral, não somente a projetistas, estudantes de engenharia ou engenheiros. A figura 11 mostra um Arduino Duemilanove.

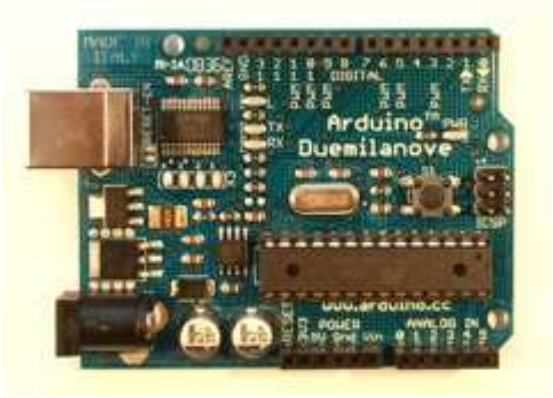

Figura 11 - Arduino Duemilanove [8]

O Arduino é um microcontrolador de placa única e um software para programá-lo. O hardware consiste de um projeto open-hardware simples e para o controlador com processador AVR Atmel e suporte *I/O on-board*. O software consiste de uma linguagem

de programação padrão e do boot loader (gerenciador de inicialização) que é executado na placa.

4.1.2 ATmega328

Microcontrolador AVR, com tecnologia picoPower, fabricado pela ATMEL. O ATmega328 é um microcontrolador 8 bits de arquitetura RISC (*Reduced Instruction Set Computing*) avançada. Executando a maioria das 131 instruções em um único ciclo ele se aproxima de 1MIPS (Milhões de Instruções Por Segundo) por MHz, permitindo uma otimização do consumo contra a velocidade de processamento. A tabela 5 a seguir resume algumas características do microcontrolador.

32 registros de uso geral
2 Timer/Contador de 8 bits com Prescaler
1 Timer/Contador de 16 bits com Prescaler
Memória
FLASH 32K Bytes
EEPROM 1K Bytes
SRAM2K Bytes
Periféricos
23 canais I/O programáveis
6 canais PWM
6 canais ADC de 10 bits
1 USART programável
1 Interface serial SPI Mestre/Escravo
1 Watchdog programável
Tensão de operação: 1.8 - 5.5V
Consumo em 1 MHz, 1.8V(25℃:
Modo Ativo: 0.2 mA
Modo Power-down: 0.1 µA
Modo Power-save: 0.75 µA

Tabela 5 – caracteristhcas ATmega328 [9]

4.1.3 Interface SPI

O SPI (*Serial Peripheral Interface Bus*) é um protocolo de comunicação serial síncrona padrão, que opera em modo *full duplex*. Dispositivos se comunicam em modo mestre/escravo, com o mestre iniciando o *frame* de dados. Esta interface permite a comunicação do microcontrodador com diversos componentes periféricos, formando uma rede, Por vezes SPI é chamado de barramento serial *four-wire*, por causa dos seus 4 sinais lógicos: Serial Clock, MOSI (*Master Output, Slave Input*), MISO (*Master Input, Slave Output*) e SS (*Slave Select*). A figura 12 mostra a conexão mestre-escravo SPI.

O Atmega328 possui 3 registros para a interface SPI. São eles o registro de controle SPCR (SPI *Control Register*), o registro de estado SPSR (SPI *Status Register*) e o registro de dados SPDR (SPI *Data Register*). No SPCR habilitamos ou não a interface SPI e sua interrupção, definimos se o microcontrolador será o mestre ou um escravo, a freqüência, fase e polaridade do clock (SCK). Um registro de inicialização. Já o SPSR nos indica o estado da comunicação e o SPDR os dados da comunicação. Escrever no SPDR inicia a transmissão dos dados e a leitura deste registro causa a leitura do buffer de recebimento do Shift Register.

Para iniciar a comunicação, primeiro o mestre configura o clock (SCK) de acordo com a freqüência suportada pelo escravo. O mestre então habilita o dispositivo escravo para a transmissão colocando SS em nível baixo.

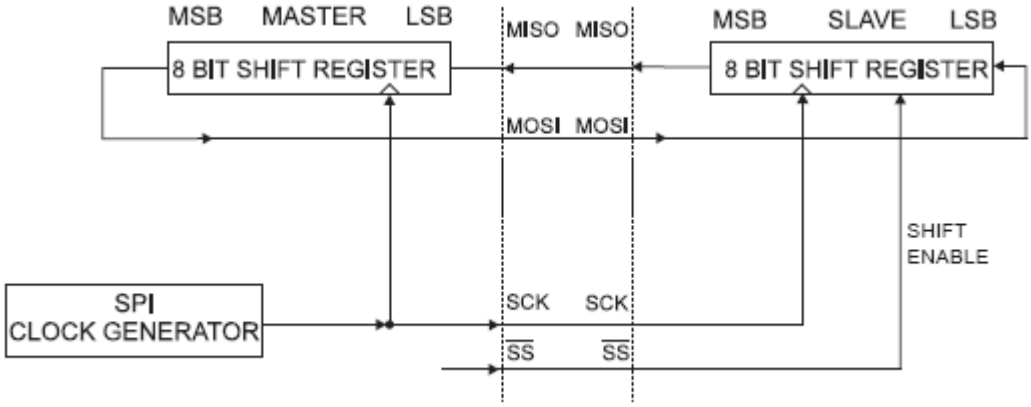

Figura 12 – Interconexão SPI, mestre-escravo [9]

Durante cada ciclo de clock SPI ocorre uma transmissão de bit *full-duplex*. Transmissões envolvem dois *Shift Register*, um nó mestre e um nó escravo e eles estão conectados em anel. Geralmente a transmissão começa pelo bit mais significativo

(MSB). Ao final mestre e escravo trocaram os valores de seus registros. Se houver mais dados a serem trocados, o *Shift Register* é carregado novamente. Quando não há mais dados a serem trocados o mestre para de enviar o sinal de *clock*. Encerrando a transmissão voltando SS ao nível alto.

4.2 Módulo GPS ME-1000RW

O módulo GPS ME-1000RW, fabricado pela ME COMPONENTES Tech, é extremamente versátil. Alimentação é de 3,8V a 6,0V DC, uma antena embutida e 2 saídas seriais padrão NMEA-0183 V3.01, uma em nível LVTTL e a outra RS232. A saída RS-232 permite a conexão direta à porta serial de qualquer computador. Já a saída LVTTL é usada para ligá-lo a um microcontrolador. Graças a estas características este módulo é muito prático.

Em sua saída serial o módulo fornece 5 tipos de mensagem: GPGGA, GPGSA, GPRMC, GPVTG e GPGSV. As mensagens GPGGA e GPRMC contêm as coordenadas de localização, principal informação para a nossa aplicação. Na prática a sentença GPRMC (*Recommended Minimum Specific GPS/Transit Data*) foi escolhida porque possui, além da latitude e longitude, informações adicionais como hora, data e velocidade.

Item	**Especificação**
Canais	65 Canais
Sensibilidade	- 161 dBm
Freqüência	L1 - 1.575,42MHz
Tempo de partida	Partida a Frio 29 seg. Partida Intermediária 28 seg. Partida Quente 1 seg.
Taxa de Atualização	1 Hz
Interface Serial	RS232 e LVTTL
Precisão	Posição 5 metros Velocidade 0,1 m/seg Tempo 300 ns
Protocolo	NMEA-0183 V3.01
Limites operacionais	Altitude < 18.000 m Velocidade < 515 m/s
Dinâmica	4G (39.2m/sec2)
Faixa de Tensão	3.6V a 6V
Corrente no modo de rastreamento	~ 23mA
Corrente no modo de aquisição	~ 70 mA

Tabela 6 – Especificações GPS ME-1000RW [10]

4.2.1 Teste do módulo ME-1000RW

O módulo vem com um led indicador de aquisição. O led fica sempre ligado quando a posição não está fixada, e pisca numa freqüência de 2 Hz quando fixada.

Para averiguar o funcionamento do módulo, conectou-se a saída RS232 ao computador e através do programa Hyperterminal foram recuperadas diversas mensagens NMEA. A figura 13 mostra uma dessas mensagens GPRMC.

$GPRMC,	10.513.993,00	A,	25.291.702,00	S,	49.200.997,00	W,	000.0,	088.3,	201010	,	,	A,	*65	<CR><LF>
	1	2	3	4	5	6	7	8	9	10	11	12	13	

Figura 13 – Mensagem GPRMC

Campo	Nome	Exemplo	Descrição
1	Tempo UTC	10.513.993	Horário UTC* Variação de (000000.000 ~ 235959.999)
2	Status	A	'V' = GPS aquecendo; 'A' = Dados válidos
3	Latitude	25.291.702	Latitude no formato ddmm.mmmm
4	Indicador	S	N Hemisfério, 'N' = Norte, 'S' = Sul
5	Longitude	49.200.997	Longitude no formato dddmm.mmmm
6	Indicador	W	E Hemisfério, 'E' = Leste, 'W' = Oeste
7	Velocidade	000.0	Velocidade em nós (000.0 ~ 999.9)
8	Curso	088.3	Curso em graus (000.0 ~ 359.9)
9	Data UTC	201010	Data UTC de uma posição fixa no formato, ddmmyy
10	Variação magnética		Em graus (000.0 ~ 180.0)
11	Variação magnética		Em Direção 'E' = Leste; 'W' = Oeste
12	Indicador de Modo	A	'N' = Dados não válidos 'A' = Modo autônomo 'D' = Modo Diferencial 'E' = Modo Estimado (dead reckoning DR) 'M' = Modo de entrada manual 'S' = Modo de Simulação
13	Checksum	*65	É o OU EXCLUSIVO de todos os caracteres entre "$" e o "*"

Tabela 7 – Campos da mensagem GPRMC

4.3 *Display* LCD

Um *display* LCD foi adicionado ao sistema para imprimir as informações recuperadas do módulo GPS. Como uma forma de validação intermediária e também para auxiliar o desenvolvimento do sistema e na depuração do código. O *display* utilizado é o INTECH ITM1602B, um display de 2x16 caracteres. Seu diagrama de blocos é apresentado na figura 14 e sua pinagem na tabela 8.

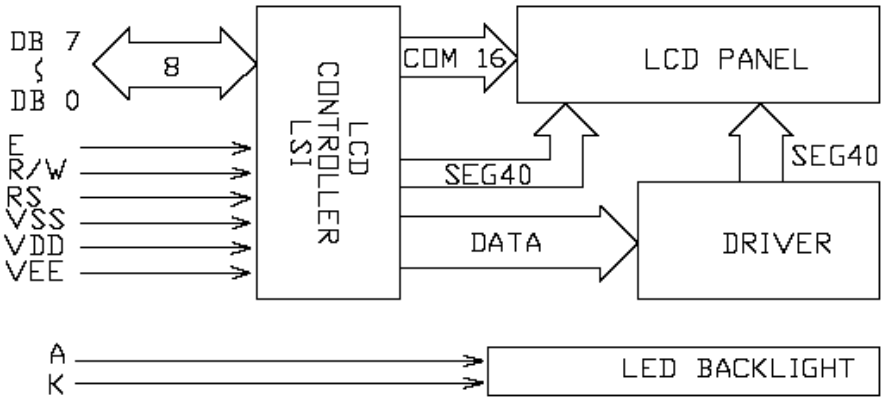

Figura 14 – Diagrama de blocos do módulo LCD INT1602B [11]

PINO	SIMBOLO	DESCRIÇÃO
1	A	LED BACKLIGHT +
2	K	LED BACKLIGHT -
3	VSS	GND
4	VDD	VCC (7V DC máx)
5	VEE	Tensão para o painel LCD
6	RS	Seletor de Registro
7	R/W	Seletor Read/Write
8	E	Read/Write Enable
9	DB0	Data Bus
10	DB1	
11	DB2	
12	DB3	
13	DB4	
14	DB5	
15	DB6	
16	DB7	

Tabela 8 – Pinagem do módulo LCD ITM1602B [11]

4.4 MCP2515 – Controlador CAN com Interface SPI

Para implementar a interface CAN, uma vez que o Atmega328 não tem o controlador CAN integrado, foram escolhidos os circuitos integrados da Microchip para fazer a interface com o barramento CAN, o controlador CAN MCP2515[15] e o transceptor MCP2551[16].

O Controlador CAN MCP2515 da Microchip Technology implementa a especificação CAN versão 2.0B. A figura 15 apresenta o diagrama de blocos do controlador. Ele é capaz de transmitir e receber os frames padrão, estendido e remoto. Até 8 bytes no campo de dados, numa taxa de até 1Mbps. Possui duas máscaras e seis filtros de aceitação que são utilizados para filtrar mensagens indesejadas, poupando assim o processamento do microcontrolador. A interface com os microcontroladores (MCUs) é feita através de uma *Serial Peripheral Interface* (SPI), padrão industrial. A interface SPI é de alta velocidade, 10MHz. O Controlador MCP2515 tem pinos de interrupção que permitem uma maior flexibilidade ao sistema. Há um pino de interrupção de propósito geral e pinos de interrupção específicos cada um dos 2 buffers de recepção, que podem ser usados para indicar o recebimento de uma mensagem valida.

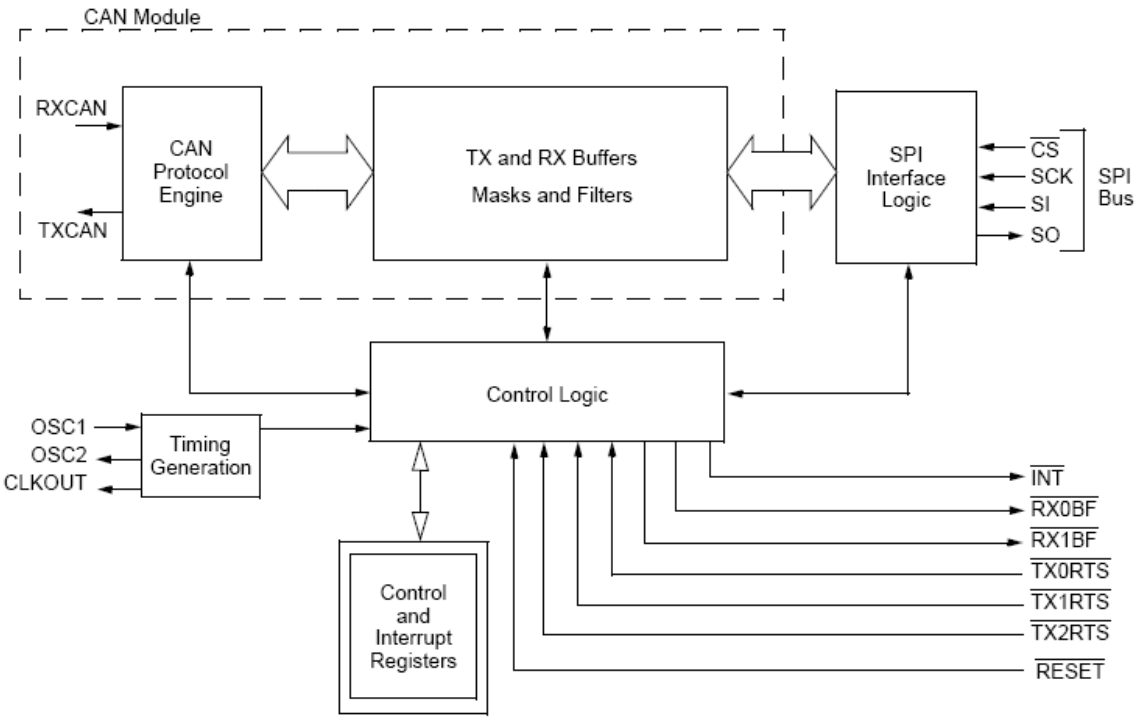

Figura 15 – Diagrama de blocos do controlador MCP2515 [15]

O MCP2515 é composto por um módulo CAN (com mecanismo de protocolo CAN, máscaras, filtros e buffers de transmissão e recepção); por registros de controle lógico para configurar o dispositivo e sua operação; e o bloco do protocolo SPI.

Existem 5 modos de operação: modo de configuração, modo normal, modo *Sleep*, modo *Listen-only* e modo *Loopback*. As transições entre modos está ilustrada pela figura 16. O modo de configuração é definido pelos bits CANCTRL. REQOP, e deve ser confirmado nos bits CANSTAT.OPMODE.

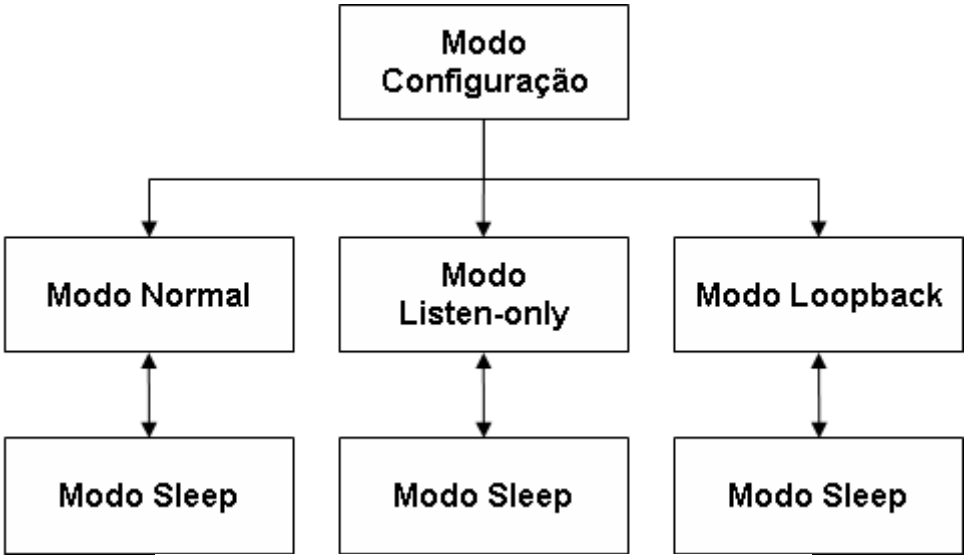

Figura 16 – Modos de funcionamento do MCP2515 [15]

No estado final do sistema de notificação ele operaria no modo *Listen-only* aguardando pela mensagem do ACU. Ou no modo normal caso haja a necessidade de enviar alguma mensagem ao barramento, talvez um frame remoto solicitando algum dado. Na fase de desenvolvimento e teste do sistema, o modo *Loopbak* é empregado. O modo *Loopback* permite a transmissão interna de mensagens do *buffer* de transmissão para o *buffer* de recepção.

5 ESPECIFICAÇÃO DO *SOFTWARE*

Uma vez definida a configuração em hardware, passamos ao desenvolvimento do software. Foi utilizada a linguagem de programação do Arduino que é baseada em C/C++. O firmware é dividido, conforme figura 17, em 3 partes: inicialização, aquisição das coordenadas e envio de mensagem SMS.

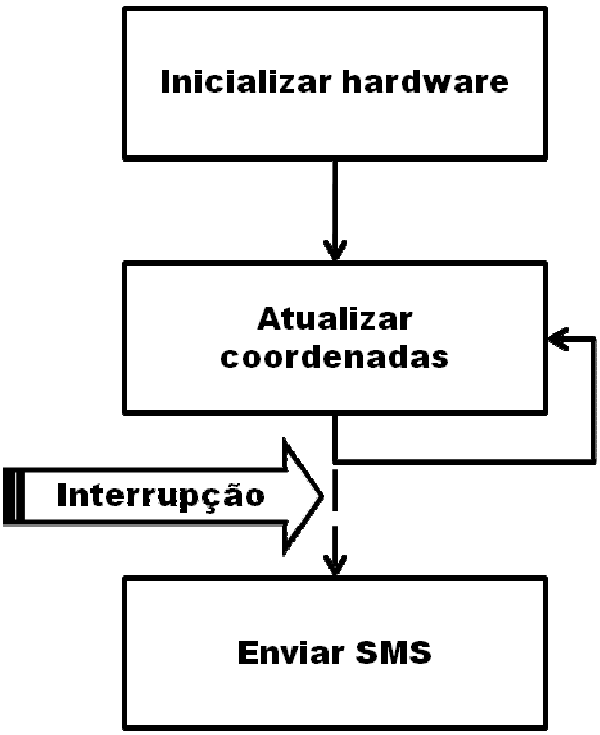

Figura 17 – Rotina principal do firmware

O desenvolvimento do firmware foi dividiu em 3 etapas. Correspondentes as 3 interfaces hardware do microcontrolador. A comunicação com o módulo GPS, com o controlador CAN e o envio de mensagem SMS.

5.1 Comunicação com o módulo GPS

A interface de comunicação com o módulo GPS é a interface serial com protocolo NMEA a uma taxa de transmissão de 9600bps. A figura 18 mostra o fluxograma de aquisição de coordenadas. A mensagem NMEA é recebida bit a bit até estar completa. Em seguida, identifica-se o tipo de mensagem pelos seus 5 primeiros caracteres. Se a mensagem é uma mensagem GPRMC as coordenadas são armazenadas. Senão ela é descartada. Esta rotina ser repete continuamente.

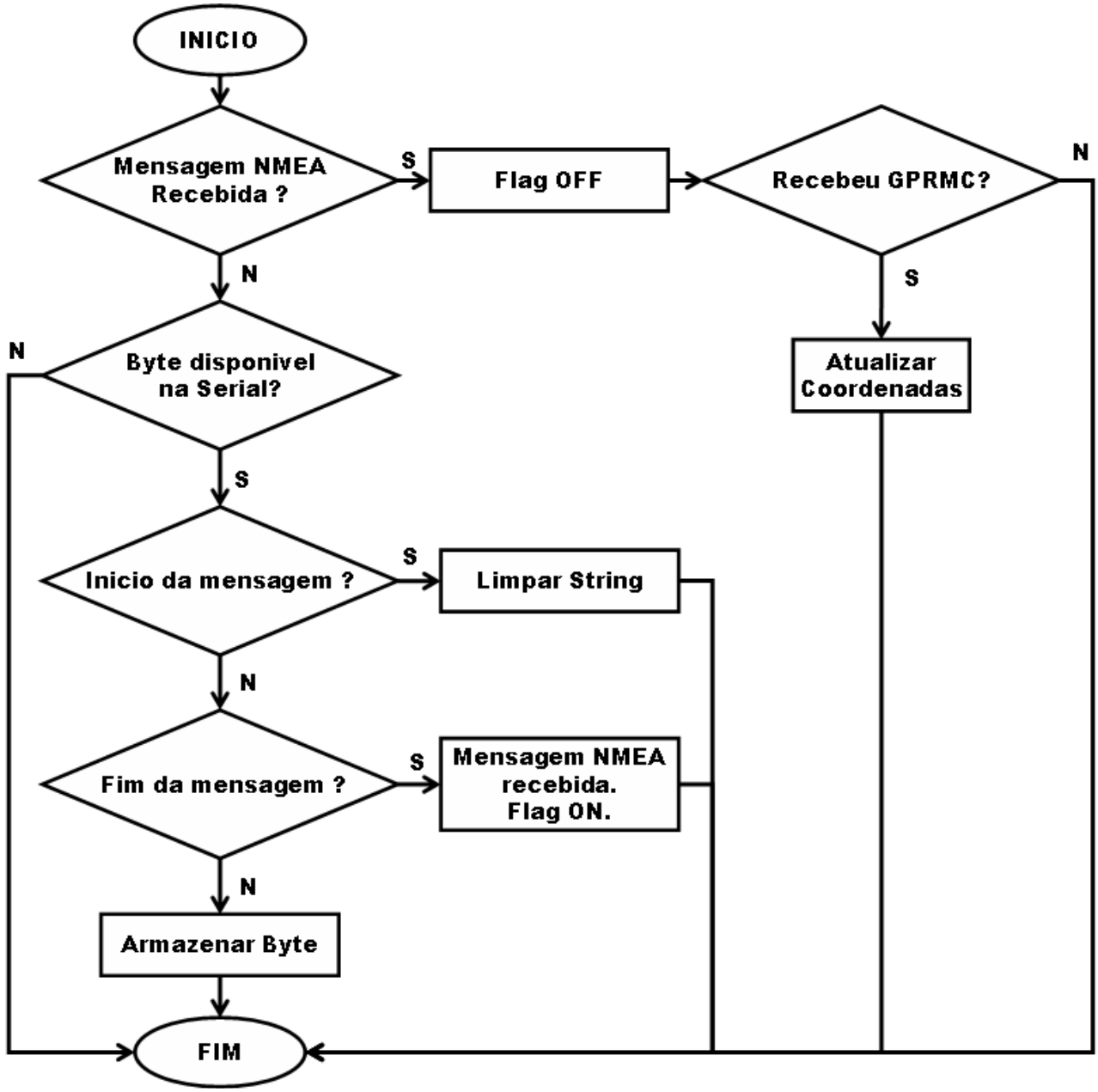

Figura 18 – Fluxograma de aquisição de coordenadas

5.2 Interface com o controlador CAN

5.2.1 SPI

A comunicação com o controlador CAN é feita através da interface de comunicação serial SPI. O Arduino possui uma biblioteca dedicada a SPI. Essa biblioteca possui a função *SPI.begin* que habilita e configura a interface SPI do microcontrolador. A função inicializa o registro SPCR do ATMega328 que é o registro de controle do SPI. Os bits deste registro estão ilustrados na figura 19.

Bit	7	6	5	4	3	2	1	0
SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
Valor	0	1	0	1	0	1	1	1

Figura 19 – Registro SPCR [9]

A biblioteca SPI do Arduino possui também a função *SPI.transfer* para realizar a transferência de dados através da interface, conforme figura 20. A função carrega o dado a transferir no registro de dados SPDR. O restante do processo é feito pelo automaticamente pelo microcontrolador, restando somente se assegurar que a transferência foi completada. O fim da transferência é indicado por um flag no registro de estados SPSR, em seguida a função lê o registro SPDR e retorna este valor.

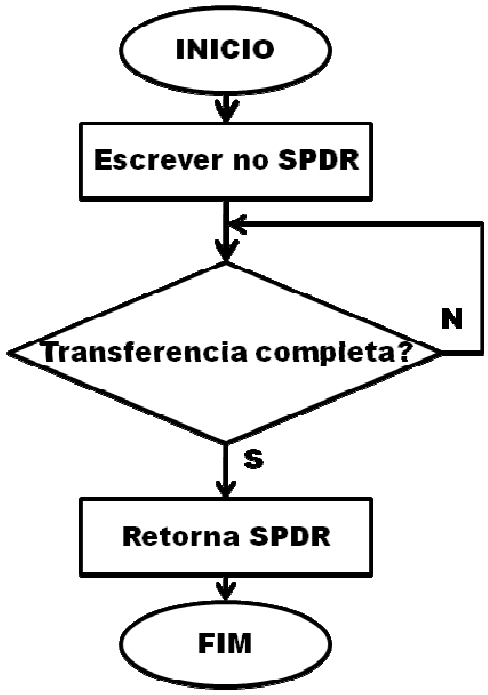

Figura 20 – Fluxograma da função SPI.transfer

5.2.2 Inicialização do controlador CAN

O controlador MCP2515 possui um conjunto de 9 instruções. Algumas especializadas no envio e recepção de mensagens. As figuras 22 e 23 abaixo ilustram as instruções básicas *WRITE* e *READ*, utilizadas para manipular os registros do controlador. Os passos da inicialização do controlador CAN estão resumidos na figura 21.

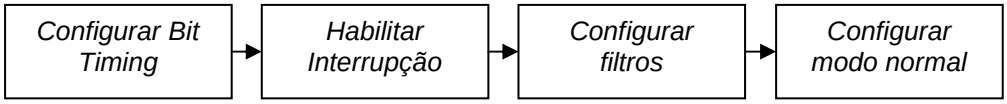

Figura 21 – Inicialização do MCP2515

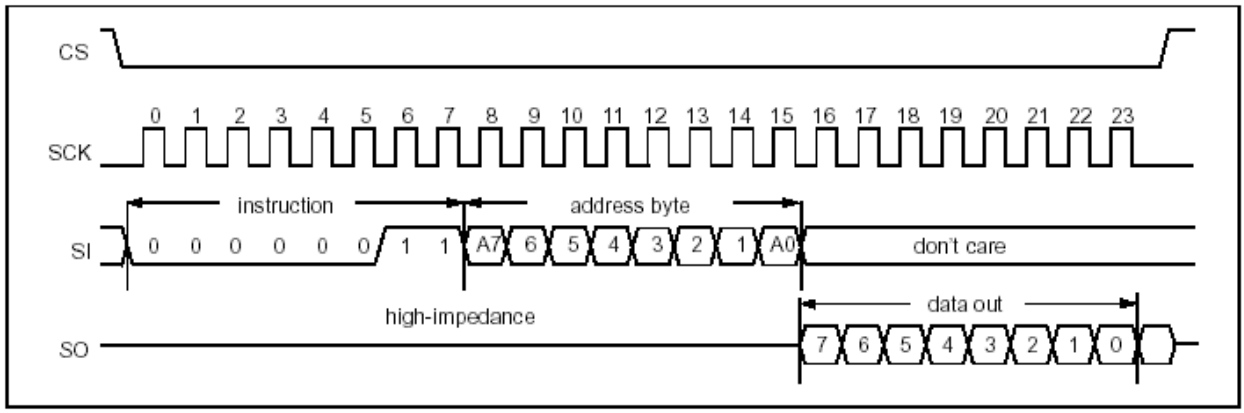

Figura 22 – Instrução READ

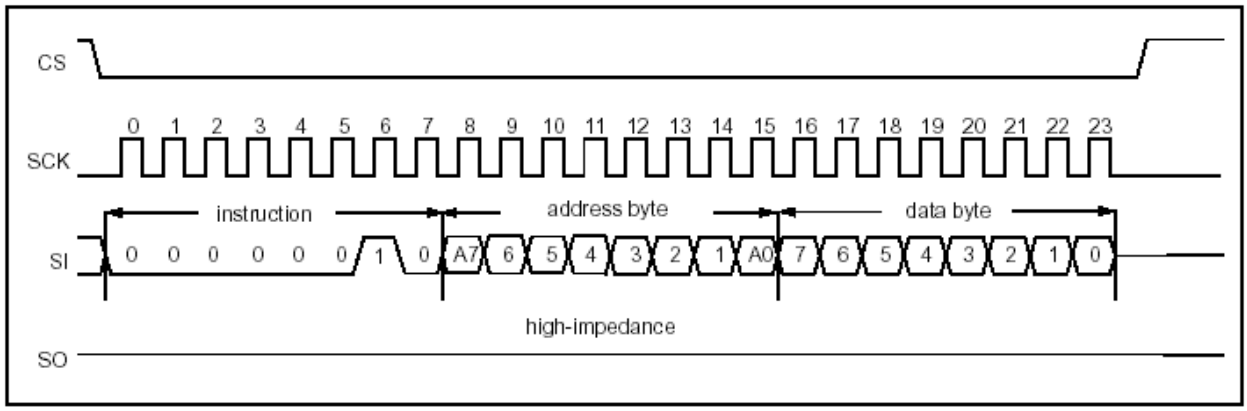

Figura 23 – Instrução WRITE

6 VALIDAÇÃO

A validação do sistema será feita com o envio de um SMS após o recebimento de uma mensagem CAN. A mensagem SMS conterá as coordenadas adquiridas e um número representando a identificação do veiculo. A batida será simulada por um *push button*, que causa o envio de uma mensagem pelo controlador CAN.

Numa configuração com somente 1 microcontrolador, conforme ilustrado na figura 24, o controlador CAN estará em modo *Loop Back* no qual a mensagem enviada (*buffer* de transmissão) é transferida diretamente ao *buffer* de recepção sem a passagem da mensagem pelo barramento.

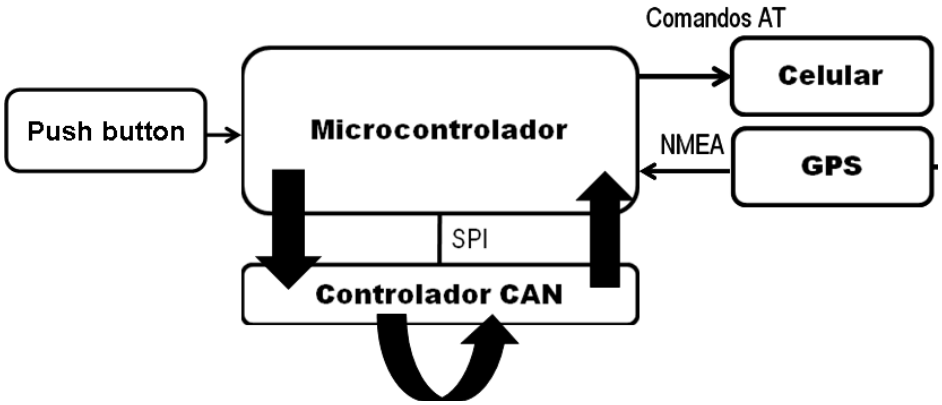

Figura 24 – Validação com 1 microcontrolador

Num passo seguinte uma validação com 2 microcontroladores, conforme figura 25, é desejada. Nesta configuração um microcontrolador simula o ACU e a mensagem CAN trafega pelo barramento.

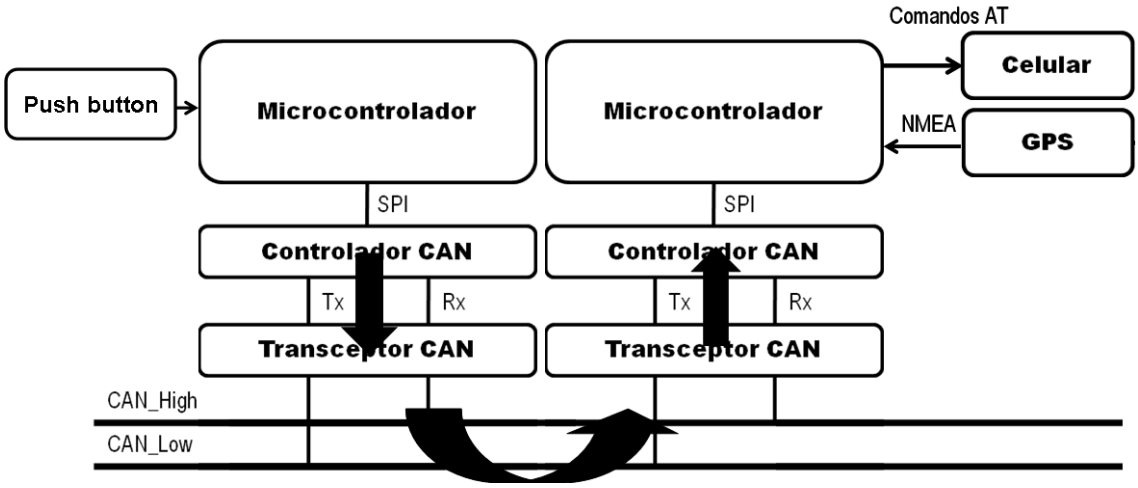

Figura 25 – Validação com 2 microcontroladores

7 RESULTADOS

O sistema proposto consiste de 3 grandes módulos: módulo GPS, interface CAN e o módulo de comunicação com o exterior. O desenvolvimento também se organizou e dividiu-se nessas 3 partes: GPS, CAN e SMS. A figura 26 exibe o diagrama de blocos.

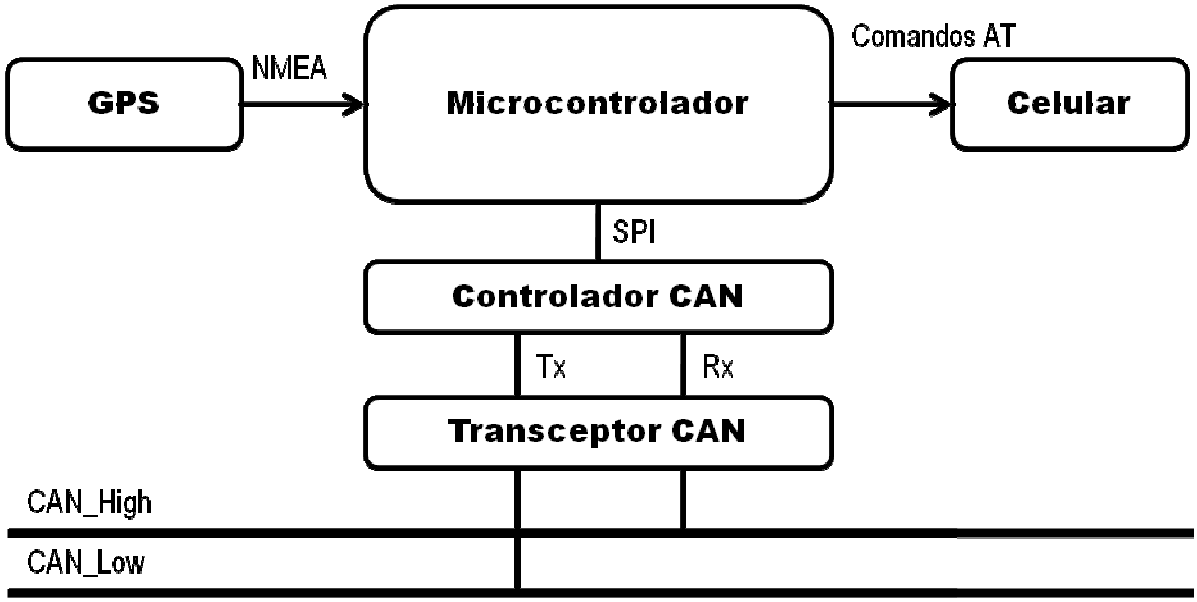

Figura 26 – Diagrama de blocos

A aquisição de coordenadas foi realizada com sucesso. Os sinais GPS transmitidos sob protocolo NMEA foram decodificados. A recuperação de dados, por exemplo as coordenadas, só é possível conhecendo o tipo da mensagem NMEA e seus campos. A aquisição e atualização constante das coordenadas foram então feitas com êxito. A validação é feita pela impressão das coordenadas em um display LCD. Como um extra, foi programada uma função no algoritmo que recupera da informação de data e hora, e as imprime no display LCD.

Para o envio da mensagem SMS foi escolhido é um aparelho celular Motorola V3. A transferência de dados deste aparelho é feito através de uma interface USB. Primeiramente foi verificado que o aparelho reconhece comandos AT e quais comandos são implementados no aparelho. Para tal, o aparelho foi conectado ao computador e comandos AT foram enviados pelo programa Hyperterminal. Foi necessária a instalação de um driver USB obtido no site da Motorola. A resposta OK ao comando AT, comprovou que a comunicação foi estabelecida e que o aparelho aceita comandos AT. O comando AT+CLAC forneceu a lista de comandos reconhecidos pelo aparelho. Uma

ligação foi realizada com sucesso com o comando ATD e uma mensagem SMS foi enviada com as seguintes linhas de comando presentes na figura 27. O conversor UART-USB FT232RL foi adquirido para realizar a comunicação entre o microcontrolador e o aparelho celular.

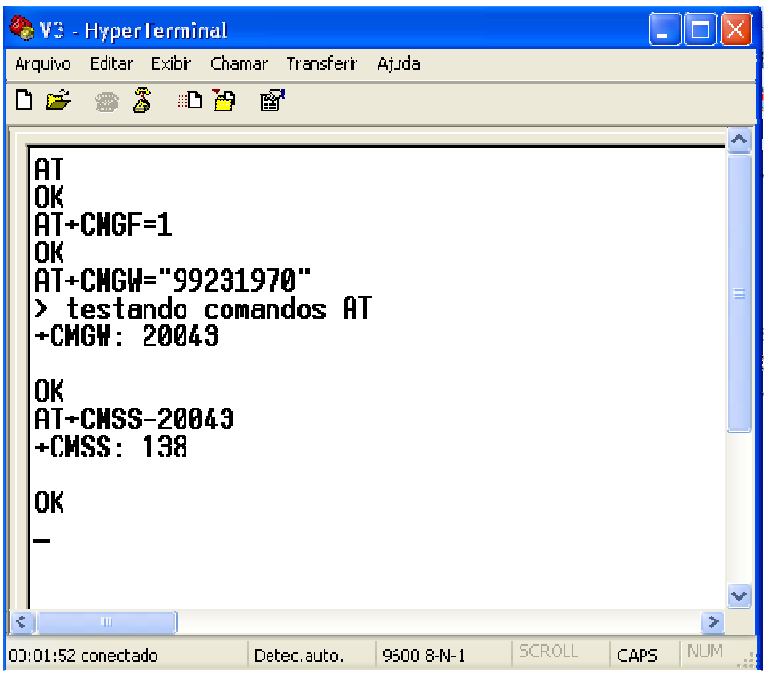

Figura 27 – Envio de SMS com comandos AT via Hyperterminal

O conversor FT232RL foi testado e funcionou perfeitamente na comunicação entre a interface serial do microcontrolador e a interface USB do computador. Porém a reconfiguração do modo de alimentação é necessária, pois circuito conversor é alimentado via USB e aparelho celular não fornece a esta alimentação. Para que o circuito esteja no modo auto-alimentado, alimentado pelo microcontrolador. A reprogramação da EEPROM interna do circuito integrado é feita através da interface USB com o software FT_PROG. O software está disponível no site do fabricante. A reconfiguração foi realizada com sucesso, porém não foi possível estabelecer comunicação com o aparelho celular, seria necessária a implementação do driver. A dificuldade põe em questão se não haveria uma solução mais simples que a escolhida.

Para implementar a interface CAN é necessário realizar trocas de dados através da interface SPI. A comunicação pela SPI foi testada enviando uma instrução de escrita e outra de leitura de um registro do controlador CAN, observando a cada passo o registro de dados SPDR do microcontrolador.

Num primeiro momento leituras do mesmo registro retornavam valores diferentes, aparentemente aleatórios.

O primeiro ponto verificado foi a questão do sincronismo, pois o controlador CAN tem seu *clock* gerado por um cristal independente do microcontrolador. Foi os registros de controle da interface SPI do microcontrolador, o valor do cristal e capacitores no controlador CAN. O valor utilizado é o valor mencionado no *datasheet*.

Após trocar o controlador, *protoboard*, cristal e capacitor, não se lia mais valores aleatórios. Porém não se chegou ao comportamento desejado. O resultado é inconclusível. Ao final do ciclo de transferência SPI o dado lido no registro SPDR era exatamente o mesmo carregado no início, o que se esperaria somente ao final do próximo ciclo. Há 2 possíveis causas, ou o *byte* de dado não está sendo transferido ou o *byte*. Porém a leitura anterior de valores aleatórios no SPDR leva a crer que a transferência está ocorrendo.

Para testar o controlador CAN pretendia-se inicializá-lo no modo *LoopBack*, enviando uma mensagem no *buffer* de transmissão e lendo em seguida a mesma mensagem no *buffer* de recepção. Em seguida uma comunicação entre 2 controladores CAN através do barramento.

8 CONCLUSÃO

Um sistema de notificação automático tem o potencial benéfico para a sociedade. A existência de um projeto semelhante da comissão européia reforça este embasamento. Embora os benefícios não possam ser quantificados neste momento, espera-se que tal sistema facilite o atendimento médico de emergência a fim de reduzir o número de fatalidades e a gravidade dos ferimentos causados por acidentes de trânsito. Há perspectivas para que este sistema seja integrado com um sistema de rastreamento, e um sistema de diagnóstico on-line.

Um ponto a destacar na realização deste projeto é o melhor conhecimento adquirido na área de segurança veicular. Houve um real aprendizado em programação de microcontroladores, módulos CAN e GPS, sobretudo dos protocolos de comunicação NMEA, SPI e CAN. Desafios e perspectivas da segurança veicular no contexto nacional e mundial foram analisados sob diferentes ângulos, não somente tecnicamente. Uma das dificuldades foi a falta de experiência em gestão de projetos. Sobre tudo o que se refere à gestão do tempo e tomada de decisão, conhecimento da área de aplicação e ferramentas. Embora este não tenha sido o ponto forte deste trabalho, foi possível identificar oportunidades de melhoria para futuros projetos.

O objetivo de desenvolver um protótipo infelizmente não foi alcançado. Todavia, existem realizações e pontos a serem destacados. Entre eles a aquisição constante de coordenadas, armazenamento e exibição no display LCD realizada com sucesso. Ainda é necessário depurar a inicialização do controlador MCP2515. Em teste não se pode confirmar que as instruções de escrita e leitura de seus registros foram corretamente executadas pelo controlador MCP2515. O envio de mensagem SMS com comandos AT foi realizado com sucesso através do computador. Porém, dificuldades de realizar o mesmo via microcontrolador levanta a questão de se rever a solução escolhida para envio de mensagem ao serviço de emergência.

Trabalhar com hardware e software foi desafiador, principalmente o que diz respeito a escolha do hardware. A escolha adequada do microcontrolador (com ou sem a interface CAN) e das ferramentas de desenvolvimentos (programador, ambiente de desenvolvimento software, depurador) foi prejudicada pela falta de conhecimento de preços e indisponibilidade de dispositivos no mercado local. A escolha da plataforma

Arduino foi assertiva em função do preço, disponibilidade, facilidade de uso e flexibilidade de aplicação.

A oportunidade única que foi este projeto me permitiu aplicar de forma prática os conhecimentos adquiridos na universidade, em algo útil para a sociedade. Analisando o contexto deste trabalho, considerando os conhecimentos adquiridos, minhas habilidades e dificuldades pessoais, tive uma melhor percepção de oportunidades profissionais e perspectiva de especialização.

9 REFERENCIAS

1. Organização Mundial de Saude. Disponível em: <http://www.who.int/violence_injury_prevention/road_safety_status/2009/en/index.html>

2. ORGANIZAÇÃO MUNDIAL DE SAUDE – OMS. **Global status report on road safety**. Disponível em: <http://whqlibdoc.who.int/publications/2009/9789241563840_eng.pdf>

3. ORGANIZAÇÃO MUNDIAL DE SAUDE – OMS. **Perfis de países**. Disponível em: <http://www.who.int/violence_injury_prevention/road_safety_status/country_profiles/en/index.html>

4. MINISTÉRIO DA SAUDE. **Sistema de Informação de Mortalidade**. Disponível em: <http://portal.saude.gov.br/portal/saude/visualizar_texto.cfm?idtxt=21377>

5. Comissão Européia. **Projeto eCall**. Disponível em: <http://ec.europa.eu/information_society/activities/esafety/ecall/index_en.htm>

6. Can in Automation - CiA. **Especificação CAN 2.0**. Disponível em: <http://www.can-cia.org>

7. LUGLI, Alexandre Baratella; SANTOS, Max Mauro Dias. **Sistemas Fieldbus para Automação Industrial: DeviceNet, CANopen, SDS e Ethernet**. São Paulo: Érica, 2009.

8. Arduino**. Software, fórum e referência**. Disponível em: <www.arduino.cc>

9. Atmel. **Datasheet do microcontrolador ATmega328**. Disponível em: <http://www.atmel.com/dyn/resources/prod_documents/8271S.pdf>

10. ME Componentes. **Datasheet do módulo GPS ME-1000RW**. Disponível em: <http://www.mecomp.com.br/rumo/ME-1000RW.pdf>

11. Intech. **Datasheet do módulo LCD ITM1602B**. Disponível em: <http://www.intech-lcd.com/image/Character_LCM/1602b-c.pdf>

12. Motorola. **Driver USB Motorola v.4.7.1**. Disponível em: <http://www.motorola.com/consumers/v/index.jsp?vgnextoid=1f14f616d8042210VgnVCM1000008806b00aRCRD>

13. FTDI Chip. **Datasheet FT232RL**. Disponível em: <http://www.sparkfun.com>

14. FTDI Chip. **Software FT_PROG**. Disponível em: <http://www.ftdichip.com/Products/ICs/FT232R.htm>

15. Microchip. **Datasheet controlador CAN MCP2515**. Disponível em: <http://ww1.microchip.com/downloads/en/DeviceDoc/21801F.pdf>

16. Microchip. **Datasheet do transceptor CAN MCP2551**. Disponível em: <http://ww1.microchip.com/downloads/en/DeviceDoc/21667f.pdf>