KLEBER NOGUEIRA HODEL

LIMITES DO PROTOCOLO CAN (CONTROLLER AREA NETWORK) PARA APLICAÇÕES QUE EXIGEM ALTO GRAU DE CONFIABILIDADE TEMPORAL

São Paulo

2009

KLEBER NOGUEIRA HODEL

LIMITES DO PROTOCOLO CAN (CONTROLLER AREA NETWORK) PARA APLICAÇÕES QUE EXIGEM ALTO GRAU DE CONFIABILIDADE TEMPORAL

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do Título de Mestre em Engenharia.

São Paulo

2009

KLEBER NOGUEIRA HODEL

LIMITES DO PROTOCOLO CAN (CONTROLLER AREA NETWORK) PARA APLICAÇÕES QUE EXIGEM ALTO GRAU DE CONFIABILIDADE TEMPORAL

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do Título de Mestre em Engenharia.

Área de concentração:
Engenharia Elétrica

Orientador:
Prof. Dr. Sebastião Gomes dos Santos Filho

São Paulo
2009

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, de janeiro de 2009.

Assinatura do autor _____

Assinatura do orientador _____

ed. rev.

FICHA CATALOGRÁFICA

Hodel, Kleber Nogueira
Limites do protocolo CAN (controller area network) para aplicações que exigem alto grau de confiabilidade temporal / K.N. Hodel. -- ed.rev. --São Paulo, 2009.
p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1.Eletrônica embarcada 2.Engenharia automotiva 3.Rede de comunicação 4.Análise de séries temporais I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II.t.

DEDICATÓRIA

Dedico este trabalho à minha Família

AGRADECIMENTOS

A Deus acima de tudo, razão das minhas ações.

Ao meu orientador Prof. Dr. Sebastião Gomes dos Santos Filho pelo incentivo para realização deste trabalho.

Ao Prof. Dr. Armando A. M. Laganá pela ajuda neste trabalho.

A minha família por estruturar um alicerce que me deixa extremamente seguro ao caminhar.

À Mercedes Benz do Brasil Ltda. que possibilitou a realização deste trabalho, através da colaboração de diversos colegas do Centro de Desenvolvimento Tecnológico.

A todos que de alguma forma contribuíram na realização deste trabalho.

RESUMO

O CAN possui diversas características intrínsecas que o fazem mais apropriado para aplicações automotivas. Neste trabalho serão destacadas algumas dessas características aplicadas a sistemas que exigem um nível muito alto de confiabilidade, através de dois estudos de casos, um para analisar as características gerais do funcionamento da rede CAN, numa arquitetura eletrônica distribuída de um veículo em diversas configurações, cuja taxa de ocupação do barramento é alterado através da modificação do tempo de transmissão de algumas mensagens, e outro estudo para verificar o comportamento de um sistema de controle em malha fechada sobre a rede, focando no impacto do atraso temporal da rede.

Palavras-chaves: Eletrônica embarcada. Engenharia automotiva. Rede de comunicação. Análise de séries temporais.

ABSTRACT

The CAN has several inherent characteristics that are more appropriate for automotive applications. In this work will be highlighted some of those characteristics applied to systems that require a very high level of reliability. It is presented two case studies, one to analyze the general characteristics of operation of the CAN network in a electronic distributed architecture of a vehicle for various configurations, whose busload is amended by modifying the time of transmission of some messages, and another study to verify the behavior of a control system in closed loop on the network, focusing on the impact of the delay time in the network.

Keywords: Embedded Electronics. Automotive Engineering. Communication Network.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABS	*Antlock Braking System*
ACK	*Acknowledge*
A/D	Analógico para Digital
AMP	*Arbitration on Message Priority*
CAL	*CAN application layer*
CAN	*Controller Area Network*
CD	*Collision Detection*
CiA	*CAN in automation*
CI	Circuito integrado
CPU	Unidade de Processamento Central
CSMA	*Carrier Sense Multiple Access*
CRC	*Checksum*
DLC	*Data length code*
DM	*Deadline-Monotonic*
ECU	*Electronic Control Unit*
EMI	*Electromagnetic Interference*
EUA	Estados Unidos da América
ICC	*Internacional CAN conference*
IDE	*Mensagem Identifier*
ISO	*International Organization for Standardization*
LAV	*Landtechnik Vereinigung* (Fabricantes de máquinas agrícolas)
NRZ	*Non Return to Zero*
ODVA	*Open DevineNet Vendor Association*
OSEK-COM	*Communication for automotive control unit application software*
OSEK-NM	*Functionality of inter-networking by standardized interfaces*
OSI	*Open System Interconnection*
RM	*Rate-Monotonic*

RTR	Requisição de transmissão remota
SAE	*Society of Automotive Engineering*
STZP	*Steinbeis Transfer Center for Process Automation*
SDS	*Smart Distributed System*
SRR	Substituto da requisição remota
SOF	*Start Of frame*
TDM	*Time Division Multiplexer*
TTCAN	*Time triggered* CAN
UK	*United Kingdom*

SUMÁRIO

1 – INTRODUÇÃO

1.1 – Motivação

Nas quatro ultimas décadas tivemos um crescimento exponencial de sistemas eletrônicos sofisticados em veículos, e analistas estimam que atualmente 90% das inovações na indústria automobilística estão ligadas aos sistemas eletrônicos [1]. A figura 1.1 [2] mostra a evolução da eletrônica embarcada na indústria automobilística.

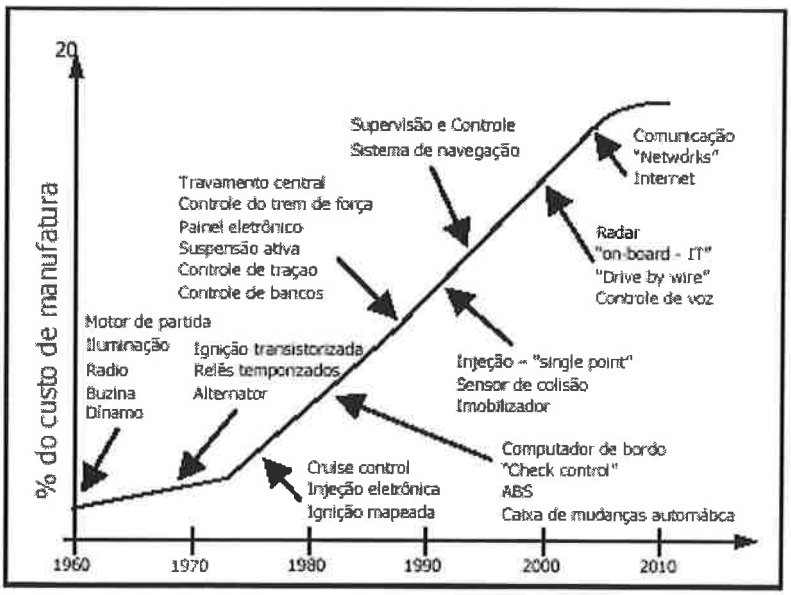

Figura 1.1 - Evolução das funcionalidades da eletrônica embarcada em sistemas automotivos [2].

Um dos fatores que influenciam essa evolução são os sensores, atuadores e microcontroladores, que melhoraram sua capacidade, desempenho, custos e novas características, facilitando a possibilidade de integração ou substituição dos sistemas puramente mecânicos e hidráulicos [3] [4]. A indústria automobilística é um dos ícones dos desafios tecnológicos dos sistemas de *hardware* e *software* embarcados. Hoje, é possível

encontrar em um veículo com uma centena de controladores programáveis que são instalados por meio de arquiteturas totalmente distribuídas [1].

Na medida em que se aumenta o número de sistemas eletrônicos em um veículo, passa a haver a necessidade de esses aplicativos interagirem entre si. Com isso, muitas discussões e modificações têm ocorrido com relação à arquitetura eletrônica ideal do veículo.

Os sistemas em geral, tais como automotivos, aeronáuticos, ferroviários, industrial, entre outros, geralmente são gerenciados a partir de um sistema de controle, o qual é composto de unidades do tipo sensores, atuadores e central de processamento. Inicialmente, o processamento de controle era concebido na forma de uma arquitetura centralizada, de acordo com as limitações tecnológicas que existiam. Com a evolução dos microprocessadores e das redes de comunicação, a filosofia de processamento distribuído foi se destacando, principalmente pelas seguintes características:

- Redução de volume e quantidade de cabos;
- Redução de conectores;
- Redução de peso;
- Aumento de processamento;
- Facilidade de integração de novos sistemas;
- Facilidade de atualização, configuração e manutenção;
- Diagnósticos;

A base para essa integração de sistemas está diretamente ligada a sistemas de comunicação de dados. Dentre os vários sistemas de comunicação de dados apresentados à indústria automobilística, o CAN foi o que obteve maior sucesso, pois possui algumas características muito interessantes conforme citadas abaixo:

- Flexibilidade (capacidade de interligar diversos "nós", garantindo futuras expansões do sistema);

- Conceito multi-mestre (elimina a existência de um módulo principal responsável pelo gerenciamento da rede);
- Robustez suficiente para garantir seu funcionamento em ambientes móveis e nocivos;
- Tolerância à falha;

No Brasil, a aplicação das redes de comunicação entre sistemas eletrônicos apresenta uma resistência imensa em função dos custos que os mesmos agregam ao veículo, mas com introdução de novos e variados sistemas eletrônicos, é praticamente impossível não evoluir no sentido de se utilizar um protocolo de comunicação.

Mais recentemente, a eletrônica embarcada ganhou notável importância no controle de sistemas mecânicos, hidráulicos e pneumáticos dos veículos. Como exemplo, podemos citar os sistemas *x-by-wire*, *brake-by-wire* e *steering-by-wire*, que interagem com sistemas que eram puramente mecânicos, melhorando o desempenho dos sistemas automotivos. Essas aplicações representam novos desafios para o desenvolvimento de sistemas embarcados, pois exige um nível de confiabilidade bem superior comparado com os aplicativos eletrônicos convencionais.

1.2 Estado da arte do desempenho do CAN

Tipicamente sistemas de tempo real são especificados para que a atuação sobre o sistema controlado seja feita conforme determinado, tanto do ponto de vista lógico como também do ponto de vista temporal. Em sistemas distribuídos em redes, como a rede CAN, cada mensagem sob rede possui um nível de prioridade para acessar o barramento, e um período de tempo pré-determinado (*deadline*) para que sua mensagem seja transmitida.

A demanda pelas diversas aplicações tem demonstrado que a rede CAN oferece boas propriedades para transmissão de mensagens com requisitos temporais (Lian, Moyne e Tilbury, 2001) [5]. Além disso, é importante a proposta de estudos para rede sob situações como a presença de *jitter* no conjunto de mensagens e o funcionamento com alta carga de

mensagens. Assim, em aplicações de sistemas distribuídos tem sido comum a busca pelo desenvolvimento de modelos e experimentos para análise de desempenho, os quais permitem o dimensionamento, otimizando o funcionamento desses sistemas.

Os trabalhos Tindell e Burns [6] e Tindell, Burns e Wellings [7] tornaram-se referência na análise de redes baseadas no protocolo CAN. A partir desses trabalhos, diversas abordagens dessas análises de rede CAN podem ser encontradas, como análise de atraso (*delay*) de comunicação abordado por Upender e Dean [8], e análise de desempenho de um sistema de controle sob condições de erro [9] para aplicações automotivas. Jeon et al. [10] também apresenta um estudo de viabilidade de aplicações de uma arquitetura distribuída em um AGV (*Automated Guided Vehicles* – Veículo Auto Guiado), Santos Stemmer e Vasquez (2002) [11] descrevem uma análise de escalonabilidade de mensagens de uma rede CAN para um helicóptero autônomo. Trabalhos recentes como em Brill et al. [12] e Davis et al. [13] reforçam a importância dessas equações de análise e apresentam novas discussões e adaptações.

1.3 Objetivos

Nos dois estudos de casos que serão abordados neste trabalho, analisar-se-á o desempenho do CAN usando as equações de Tindell [6] [7], focando a confiabilidade do sistema, sendo que no primeiro estudo de caso serão avaliadas as características gerais do funcionamento da rede CAN em uma arquitetura eletrônica distribuída em diversas configurações, onde o *busload* é alterado por meio de uma modificação do tempo de transmissão de mensagens, e no segundo estudo de caso, será analisado, o comportamento de um sistema de controle em malha fechada sobre a rede, evidenciando o impacto do atraso temporal da rede no sistema realimentado.

O desafio deste trabalho é analisar o desempenho da rede de comunicação CAN sob condições diversas, como segue:

- Analisar o comportamento de sistemas eletrônicos distribuídos numa rede no CAN;

- Analisar a rede CAN em diversas configurações de *busload*;
- Analisar o funcionamento de um sistema de malha fechada na rede;
- Apresentar os pontos frágeis do protocolo de comunicação CAN, para um sistema que necessita maior grau de confiabilidade temporal;
- Propor soluções para viabilizar um sistema de controle via rede CAN.

1.4 Conteúdo e Organização

No capítulo dois sobre arquiteturas, será apresentado a evolução das arquiteturas eletrônicas, com ênfase no conceito de arquitetura distribuída associado a protocolos de comunicação.

O capitulo três apresenta o histórico do sistema CAN, o estado da arte de estudos de desempenho e as características do CAN que o fazem ser o protocolo mais utilizado no meio automobilístico, mostrando também as equações temporais descritas por Tindell[6][7].

O capítulo quatro sobre Sistemas de Tempo Real apresenta alguns conceitos gerais temporais, que ajudam a entender as tarefas dentro de um sistema com arquitetura distribuída.

No capítulo cinco sobre sistemas *x-by-wire* é apresentada a evolução dos sistemas de segurança no veículo, e seus requisitos básicos de funcionalidade.

No capítulo seis são apresentados dois estudos de casos; o primeiro sobre o funcionamento do protocolo CAN em diversas configurações em uma aplicação real de um veículo, o segundo, sobre o comportamento de um sistema de controle em malha fechada sobre a rede CAN, utilizando as mesmas configurações do primeiro estudo de caso.

No capítulo sete são apresentados às conclusões e perspectivas.

2. ARQUITETURAS ELETRO/ELETRÔNICAS

O conjunto de diversos sistemas eletrônicos implementados e conectados em uma dada aplicação embarcada é chamado de Arquitetura Eletroeletrônica ou simplesmente Arquitetura Elétrica [14] [15]. Essa nomenclatura é bastante utilizada pelo setor automotivo.

Considerando os diversos tipos de arquiteturas existentes, destacam-se as que utilizam o conceito de Sistemas Individuais, Centralizados e Distribuídos.

2.1 Arquitetura de sistemas individuais

Pode-se dizer que a arquitetura de sistemas individuais é uma das mais antigas e data dos primórdios da introdução de sistemas elétricos nos veículos, chamada de arquitetura analógica. Nesse tipo de arquitetura não existe uma interação entre os aplicativos, os sistemas trabalham individualmente e não existe dependência entre eles conforme mostra a figura 2.1.

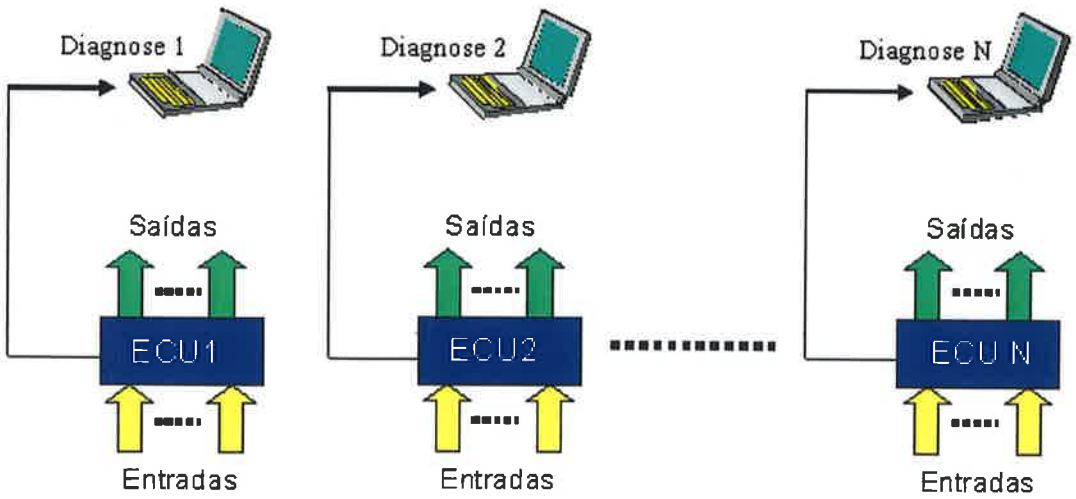

Figura 2.1 – Arquitetura Eletrônica de sistemas individuais.

A seguir vamos enumerar as vantagens e desvantagens.

Vantagens:

- Independência para desenvolvimento de sistemas;
- Simplicidade do *hardware*, constituído basicamente pelas entradas e saídas de seus sensores, atuadores e controlador.

Desvantagens:

- Cada sistema tem o seu sistema particular de diagnóstico com diferentes tipos de ferramentas, o que dificulta a manutenção do sistema;
- Duplicidade de sensores, pois não são compartilhadas as informações entre os sistemas;
- Grande quantidade de cabeamento requerido para conectar os sensores e atuadores para cada sistema, dificultando a instalação e a manutenção dos sistemas.

2.2 Arquitetura Centralizada:

Uma única ECU (*Electronic Control Unit*) é responsável por receber todos os sinais de entrada, por exemplo, sinais de sensores e chaves de comando. A ECU irá processar e comandar as respectivas saídas de controle do sistema, como válvulas e relés. O diagrama esquemático que representa esse conceito de arquitetura está representado pela figura 2.2.

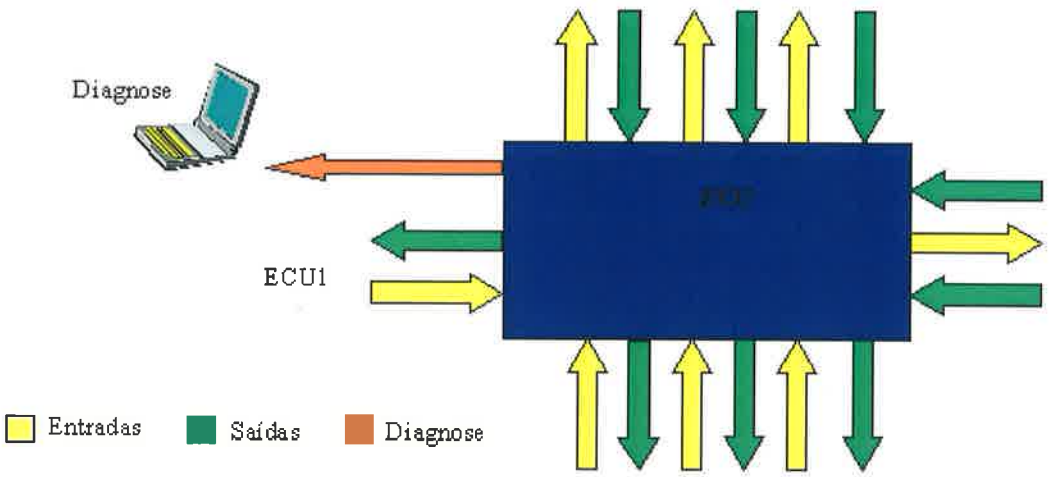

Figura 2.2 – Arquitetura Eletrônica centralizada.

A seguir são enumeradas as vantagens e desvantagens deste tipo de arquitetura.

Vantagens:

- Um único sistema de diagnóstico para toda a arquitetura [16];
- Simplicidade do *hardware* utilizado, sendo constituído basicamente pelos sensores e atuadores, uma ECU para gerenciamento do sistema e do cabeamento que os conecta [17];
- Todos os dados estão disponíveis à ECU durante toda a operação do sistema e não é crítica a lógica de varredura e coleta dos dados de cada sensor [18].

Desvantagens [18]:

- Grande quantidade de cabeamento requerido para conectar os sensores à ECU o que dificulta a instalação e a manutenção da rede;
- Limitação da possibilidade de expansão do sistema.

2.3 Arquitetura Distribuída:

A arquitetura distribuída possibilita a utilização de várias ECUs, dividindo entre elas as diversas responsabilidades relacionadas ao sistema. O diagrama esquemático que representa esse conceito de arquitetura está representado pela Figura 2..3.

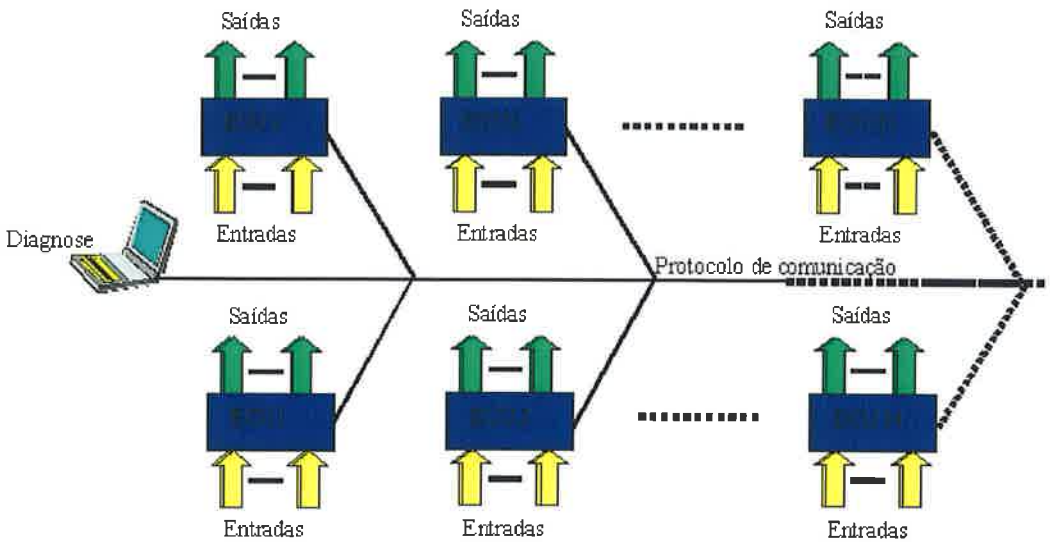

Figura 2.3 – Arquitetura Eletrônica distribuída.

A seguir também enumeramos as vantagens e desvantagens deste tipo de arquitetura.

Vantagens:

- Quantidade reduzida de cabeamento da rede sendo que as ECUs são instaladas bem próximas aos sensores, atuadores e pelo número limitado de fios conectados nas ECUs [18];
- Menor tempo de instalação do sistema, devido à menor quantidade de cabeamento necessário [19];
- Um único sistema de diagnóstico para toda a arquitetura [16];

Desvantagens:

- Necessita de um *software* de controle da rede, cuja dificuldade de desenvolvimento depende diretamente da escolha do protocolo de comunicação entre as ECUs [20];
- Dificuldade na determinação da taxa de transmissão ideal para uma dada aplicação e na escolha dos componentes eletrônicos a serem utilizados no projeto das ECUs [21];

2.4 Conceitos para definir a Arquitetura

A decisão de escolha da arquitetura elétrica mais apropriada para uma dada aplicação automotiva depende da ponderação entre diversos fatores. Dentre eles pode-se destacar:

- A complexidade do sistema a ser controlado (quantidade de variáveis de entrada e saída e o tamanho físico do sistema);
- A disponibilidade dos dispositivos eletrônicos necessários no mercado (tempo de vida);
- O tempo necessário para a implantação da arquitetura (projeto, construção de protótipos, validação e instalação final);
- Custo desejado do sistema final;
- Características do usuário do aplicativo;

A importância relativa de cada um dos fatores mencionados é o que determinará o conceito de arquitetura mais apropriado para o sistema a ser controlado.

Uma das maiores dificuldades da engenharia de produtos de uma empresa montadora de veículos é determinar a arquitetura elétrica de um novo modelo, garantindo o mínimo de funções desejadas pelos futuros clientes, dentro dos limites de custo de projeto e produto final determinados pela empresa.

As principais montadoras automotivas desenvolvem um novo veículo em torno de três anos antes de seu lançamento, o que complica ainda mais a tomada de decisão sobre qual seria a melhor solução de engenharia para um determinado projeto, pois nem sempre é possível ter

todas as informações necessárias. Essa atividade precisa das visões de engenharia avançada (tecnologia), *marketing* estratégico (mercado) e político-econônica (orçamento) da região a que se destina o novo produto, de maneira que esses elementos também afetam indiretamente o tipo de arquitetura. Existem vários exemplos de empresas como a GM do Brasil que iniciou a produção de um dos seus veículos em 2003 com arquitetura distribuída e após um ano de produção precisou modificar o projeto para arquitetura analógica para reduzir os custos.

3. CAN (CONTROLLER AREA NETWORK)

Com a evolução das arquiteturas elétricas, a filosofia de processamento distribuído foi destacando-se em função da capacidade de processamento. A principal base para arquitetura distribuída foi a utilização dos sistemas de comunicação motivados pelos seguintes benefícios:

- Redução de volume [4];
- Facilidade de diagnóstico;
- Redução de peso;
- Aumento de processamento.

O protocolo de comunicação CAN tornou-se o principal sistema utilizado pela indústria automotiva, visto que apresenta todos os benefícios mencionados e possui características intrínsecas que atendem as necessidades da indústria automobilística.

3.1 Principais características de um protocolo de comunicação para arquitetura elétrica distribuída

Dentre as principais características de um protocolo para arquiteturas distribuídas, pode-se destacar [22]:

- Ser capaz de transmitir altas taxas de informação, uma vez que os sistemas operam com informações em tempo-real;
- Boa flexibilidade das linhas de comunicação para facilitar a instalação do chicote no veículo;
- Capacidade de interligarem diversos nós, garantindo futuras expansões do sistema;
- Trabalhar dentro do conceito multi-mestre, eliminando a existência de um módulo principal responsável pelo gerenciamento da rede;
- Robustez suficiente para garantir seu funcionamento em ambientes móveis e nocivos;
- Capacidade para detectar e tratar eventuais falhas geradas por problemas em *hardware* e *software* ou interferências externas, como os gerados por efeitos eletromagnéticos.

3.2 Histórico do CAN

Em 1980, surgiu por meio dos engenheiros da empresa Bosch da Alemanha, a idéia de desenvolver um protocolo de comunicação para veículos de passeio, porque não existia nenhum protocolo que atendesse completamente aos requisitos da engenharia automotiva. Portanto, em 1983, o Uwe Kiencke (Bosch) iniciou o desenvolvimento de um novo protocolo de comunicação serial [23].

A principal idéia desse novo protocolo de comunicação da Bosch era adicionar novas funções, entretanto, a redução de cabos não foi o ponto principal deste novo desenvolvimento. Os engenheiros da Mercedes-Benz na Alemanha envolveram-se desde o início na fase de especificações desse sistema. A Intel, como o principal fabricante de semicondutores, contratou o professor Dr.Wolfhard Lawrenz da Universidade de Ciências Aplicadas em Braunschweig Wolfnbüttel na Alemanha, que chamou o protocolo de "Controller Area Network". O professor Dr.Horst Wettstein da Universidade de Karlsrushe também forneceu assistência acadêmica a esse desenvolvimento [23].

O CAN foi apresentado pela primeira vez para o mundo automobilístico em fevereiro de 1986, no congresso de engenharia automotiva (SAE) em Detroit (EUA). Esse novo sistema de comunicação foi exibido como "*Automotive Serial Controller Area Network*" [24]. Apresentado por Uwe Kiencke, Siegfried Dais e Martin Litschel, que mostraram um protocolo tipo multi-mestre baseado no conceito de arbitrário não destrutivo, no qual o barramento é gerenciado por prioridade sob as mensagens.

Os autores citados acima e outros colaboradores da Bosch, Wolfgang Borst, Wolfgang Botzenhard, Otto Karl, Helmut Schelling e Jan Unruh implementaram muitos mecanismos de detecção de falhas, contagem de erros de transmissão e recepção e desconexão automática dos nós que possuem excesso de falhas, mantendo assim o barramento em funcionamento para os outros nós. Os identificadores não foram nomeados como quase todos os outros sistemas, pelos nós de quem transmitem ou de quem recebem as mensagens, mas sim pelo conteúdo da mensagem que é diferenciado de acordo com a prioridade da informação do sistema [23].

Na tabela 3.1 são apresentados os principais pontos históricos da evolução do protocolo CAN.

Tabela 3.1 – Resumo dos principais pontos da história do CAN

Data	Assunto
1983	Inicio do desenvolvimento de uma rede de comunicação de dados para aplicação automotiva.
1986	Introdução oficial do protocolo CAN, apresentado na SAE.
1987	Primeiros chips de controladores CAN da Intel e Philips semicondutores.
1991	Introdução do protocolo CAN Kingdom.
1992	Estabelecido o grupo de internacional de fabricantes e usuários de sistemas de automação.
1992	CiA publica o primeiro protocolo de aplicações CAN.
1992	Primeira veículo que utilizou rede CAN (Mercedes-Benz).
1993	Publicação da norma ISO 11898.
1994	Primeira conferência internacional sobre CAN organizada pela CiA.
1994	Introdução do protocolo Devicenet pela Allen-Bradley
1995	Publicação do protocolo CANopen.
2000	Desenvolvimento do protocolo de comunicação TTCAN pela CiA.

3.3 Primeiro Circuito integrado

Muitas publicações e apresentações foram feitas em meados de 1987 descrevendo o conceito do protocolo de comunicação CAN, depois que a Intel implementou o primeiro controlador CAN, o 82526, que se tornou uma realidade comercial quatro anos depois. Posteriormente, a Philips Semicondutores desenvolveu também um circuito integrado de controlador CAN, o 82C200. Esses dois primeiros CIs possuíam alguns conceitos diferentes concernente aos filtros de recepção de mensagens [23].

3.4 Normalização

O CAN foi originalmente desenvolvido para carros de passeio, mas suas primeiras aplicações foram em segmentos diferentes. Especialmente no norte da Europa, o CAN tornou-se muito popular em seus primeiros dias. Na Finlândia, o CAN foi utilizado pela empresa Kone de elevadores, na Suécia, os engenheiros da empresa Kvaser sugeriram o CAN para algumas empresas de máquinas têxteis (Lindauer Donier e Sulzewr) e, também, para alguns fornecedores, como o protocolo de comunicação dentro da máquina. Com isso, foi instituído um grupo de usuários de CAN de indústrias têxtil, liderado por Lars-Berno Fredriksson, em 1989 este grupo desenvolveu alguns princípios de comunicação que ajudaram a formar o "CAN Kingdom" no começo dos anos de 1989. O CAN Kingdom não segue o modelo de referência OSI e pode ser considerado o antecessor do *higher layer* da base do CAN [23] .

Na Holanda, a Philips, que fornecia equipamentos médicos, juntou-se também ao grupo de usuário do CAN industrial, empregando habitualmente o CAN como protocolo de comunicação interna das máquinas de raios-X. As especificações da Philips foram desenvolvidas por Tom Suters, que desenvolveu o primeiro *layer* de aplicação da rede CAN. O Professor Dr.Konrad Etschberger da Universidade de Ciências Aplicadas na Weingarten, Alemanha, também desenvolveu no "*Steinbeis Transfer Center for Process Automation* (STZP)" praticamente o mesmo protocolo [23].

Nesta época começou a surgir a normalização de alto nível do protocolo CAN, a maioria dos pioneiros utilizavam as funções de comunicação, gerenciamento de rede e aplicação como uma parte do *software*. Os esforços para melhorar a camada de alto nível, mesmo nos dias de hoje não são muito explorados. Em 1990 um grupo de usuários reuniu-se para normalizar soluções diferentes. E, em 1992, Holger Zeltwanger (nesta data era editor chefe da revista VMEbus) trouxe juntamente com o grupo de usuários técnicas para melhorar o CAN.

Em 1992, foi oficialmente fundado o grupo do usuário do CAN em automação (CiA) e depois de quatro semanas foi publicado o primeiro trabalho a respeito da camada física, a CiA recomendava a utilização *transceivers* CAN, da mesma forma que a ISO 11898 [25].

Anteriormente, as aplicações utilizavam o *transceiver* RS485 na qual nem sempre era compatível, mas logo foi completamente banido.

Uma das primeiras tarefas do CiA foi a criação da especificação da camada de aplicação do protocolo CAN, usando o material existente da empresa de equipamentos medicinais da Philips e STZP, com ajuda de outros membros da CiA fizeram essa especificação, sendo chamada de "*Green Book*". [23].

Em 1980, iniciou-se o desenvolvimento de um CAN para aplicações agrícolas em LAV na associação de veículos agrícolas, mas antes deste trabalho ser completado, o comitê internacional decidiu em favor da solução, J1939 (ISO 11783), baseada no CAN que foi desenvolvida pelo comitê da SAE no grupo de caminhões e ônibus com característica não modular e fácil de usar, porém não muito flexível.

Na indústria automobilística atentam-se dois tipos de protocolos CAN: o CAN 2.0A que se tornou o padrão ISO 11898-1 [25] em 1993 e o protocolo CAN 2.0B. A maior diferença entre as duas versões é a quantidade de bits no identificador, onze *bits* para o 2.0A e vinte e nove *bits* para o 2.0B.

Os padrões ISO para CAN ou com origem em redes deste tipo são os seguintes: ISO 11898-2 (alta velocidade), ISO 11898-3 (tolerante a falhas / baixa velocidade), ISO 11992-1 [26] e SAE J2411 (meio de transmissão utiliza uma única linha para transmissão e recepção).

3.5 Primeiras aplicações

Obviamente, os fabricantes que implementaram os CIs controladores de CAN tiveram seu foco na indústria automotiva. Desde 1990 a Infineon e Motorola enviaram uma grande quantidade de chips para a indústria de carro de passeio da Europa e seus fornecedores.

Desde 1992, a Mercedes-Benz utiliza o CAN nos veículos de passeio de luxo. Num primeiro momento, o desenvolvimento focou-se para comunicação do gerenciamento do motor, e

depois para controle dos componentes gerais formando duas redes de CAN fisicamente separadas. Muitos outros fabricantes utilizam o mesmo conceito, como é o caso da BMW, Fiat, Renault, Saab, Volkswagem e Volvo.

Em 1990, os engenheiros da companhia de engenharia Cincinnati Milacron fizeram uma junção com a Allen-Bradley e Honeywell Microswitch para desenvolverem em conjunto um projeto de rede de comunicação utilizando a base CAN, porém depois de um breve período, muitos membros deste grupo trocaram de emprego. Consequentemente, essa junção faliu, mas a Allen-Bradley e Honeywell continuaram trabalhando separadamente, por conseguinte surgiram tipos de sistemas diferentes "DeviceNet" e "Smart Distributed System" (SDS), que possuem camada de comunicação nível baixo igual. Em 1994, a Allen-Bradley lançou a especificações DeviceNet como "Open DevineNet Vendor Association" (ODVA), que popularizou com grande força o "DeviceNet", já a Honeywell não seguiu o mesmo caminho e o SDS se tornou uma solução interna da empresa. O DeviceNet foi desenvolvido especialmente para fábrica de automação, um adversário direto dos protocolos como Profibus-DP e Interbus. Com a característica de sistema *plug-and-play*, o DeviceNet se tornou o sistema de comunicação de dados líder para este mercado especifico de aplicações nos EUA [23].

Na Europa muitas companhias tentaram utilizar o CAL. Embora tivesse uma abordagem academicamente correta, podia ser utilizado para aplicações industriais, todos os usuários precisavam desenhar um novo perfil, porque o CAL era exatamente a camada de aplicação. O CAL é visto como uma fase acadêmica necessária para soluções de CAN com aplicações independente, todavia nunca teve uma grande aceitação no mercado [23].

Desde 1993, dentro do escopo do projeto Esprit ASPIC, a Bosch, líder do consórcio europeu, desenvolveu um protótipo que se tornou o CANopen, e era baseado no CAL para redes internas de produção de células. Do lado acadêmico o professor Dr.Gerhard Gruhler da Universidade de Ciências Aplicadas de Reutlingen, na Alemanha, e o professor Dr.Mohammed Farsi da Universidade de Newcastle no UK, participaram deste projeto e foi um grande sucesso. Depois do fim do projeto, o CANopen foi enviado para CiA para ajustes

finais e em 1995 a revisada versão do CANopen foi liberada e dentro de cinco anos tornou-se a mais importante normalização de redes para eletrônica embarcada na Europa [23].

3.6 - Características Gerais

O CAN é um protocolo de comunicação serial síncrono. O sincronismo entre os módulos conectados a rede é feito em relação ao início de cada mensagem lançada ao barramento (evento que ocorre em intervalos de tempo conhecidos e regulares). O CAN é um barramento serial para interligar dispositivos em rede, como já citado, foi criado inicialmente para uso em sistemas de automóveis, mas logo alcançou o uso estendido para aplicações industriais. O CAN possui facilidades que são muito desejadas, como tolerância a EMI, prioridade de mensagens, recuperação de falhas, entre outras, conforme [27].

Uma rede CAN interliga até dois mil e trinta e dois dispositivos, sendo que o limite prático é de aproximadamente cento e dez. Cada um destes dispositivos é tratado como um nó da rede. No nível físico, o enlace serial mais usado é composto de dois fios, o sinal tem característica diferencial, é capaz de operar até 1 Mbps e tem restrições de velocidade em virtude da distância entre os nós. Para uma rede com extensão de 1 km, a velocidade por ser reduzida até 50Kbps. Cada nó ligado a este enlace serial é capaz de perceber simultaneamente a outros nós e os dados transmitidos na rede. A escrita, porém é uma operação permitida somente para um dispositivo por vez.

3.6.1 – Arquitetura

A rede CAN é multi-master, pode ter mais de um nó controlador, o que facilita a criação de um sistema redundante. Para isto, usa como protocolo de acesso o CSMA/CD+AMP (Carrier Sense Multiple Access/Collision Detection + Arbitration on Message Priority). Dessa forma o CAN trabalha semelhantemente o *ethernet* comum, mas ao invés de corrigir colisões de transmissão, faz com que os dois nós em conflito parem de transmitir, a rede CAN usa um

arbítrio de comparação binária (figura 2.1) para definir a prioridade das mensagens e decide qual será enviada, contudo quanto menor o valor associado maior a prioridade. Os bits que trafegam na rede recebem uma denominação de dominante e recessivo, um bit dominante representa o valor lógico 0 e o recessivo, o valor lógico 1 [16](figura 2.2).

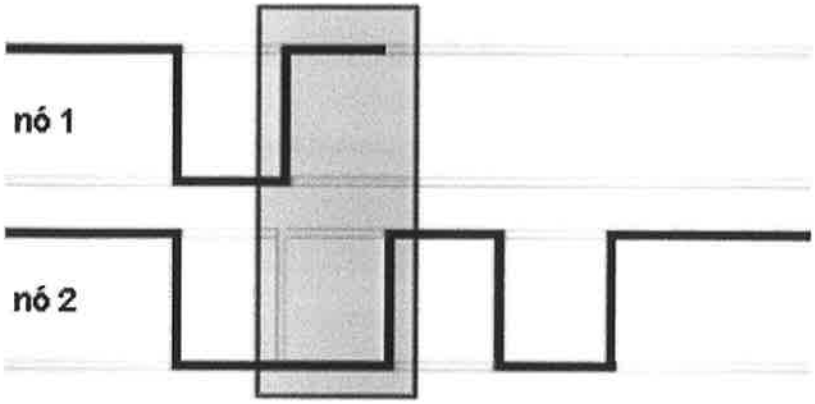

Figura 3.1 – Exemplo do arbítrio sendo aplicado durante uma transmissão entre dois nós

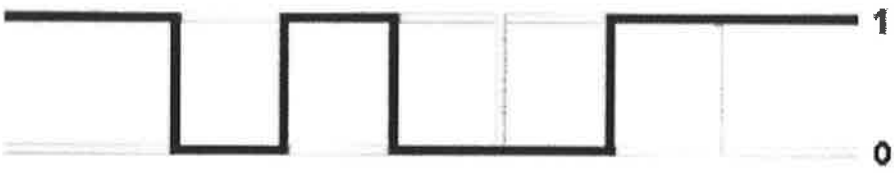

Figura 3.2 – Estada lógico do barramento em uma transmissão

As mensagens transmitidas no barramento não contêm endereços de transmissor ou receptor, mas possuem um identificador único de acordo com o conteúdo e assim que o receptor faz a seleção das mensagens que o interessa.

A especificação de CAN para as camadas de rede, de acordo com a referência ISO/OSI se classifica da seguinte forma [28]:

Camada de Enlace:

- LLC: responsável pela filtragem de mensagens, notificação de sobrecarga e controle de recuperação;

- MAC: encapsula/ desencapsula dados, realiza codificação dos quadros (*Bit Stuffing*: caso cinco bits consecutivos apresentam o mesmo nível lógico, insere-se um bit com valor inverso), controle de acesso ao meio, detecção e sinalização de erros.

Estas duas subcamadas são responsáveis ainda pelo confinamento de falhas, ou seja, um nó que estiver com muitos erros de transmissão ou recepção será automaticamente desligado da rede. O controlador CAN é responsável por lidar automaticamente com estes serviços de forma transparente ao *software*.

Camada Física:

- Realiza codificação e decodificação dos bits utilizando NRZ (*Non Return to Zero*) para que o valor médio de ocorrência de bits recessivos e dominantes seja equilibrado, temporização e sincronização do sinal. As características desta camada não são definidas pela especificação da BOSCH, porém a norma ISO define as características padrões para um *transceiver* [25] [29] e é também responsável pelo confinamento de falhas (juntamente com a camada de enlace) e tratamento de falhas provenientes do barramento.

Camada de Aplicação:

- É definida em nível de usuário e não consta na especificação. Hoje, existem algumas camadas especificadas: NMEA2000, CANopen [30], CANaerospace, DeviceNet , CAN Kingdom, entre outras.

3.6.1.1 Tipos de Quadros

O CAN utiliza variados tipos de quadros para envio de dados, requisição de dados, propagação de erros e mensagens de notificação de sobrecarga.

3.6.1.2 Quadro de Dados

O esquema de quadro de dados (como pode ser visto nas figuras 3.3 e 3.4) é apresentado em dois formatos, um para o CAN 2.0A e outro o CAN 2.0B. Podemos definir os campos da seguinte maneira [28]:

- O Início da transmissão com o *Start of Frame* (SOF) consiste de um *bit* dominante, neste *bit* é realizada a primeira sincronização entre os nós.

Nos próximos campos do quadro temos os seguintes *bits*:

- RTR (*remote transmission request*, requisição de transmissão remota): requisição de transmissão remota indica se o quadro é de dados ou de requisição remota, dominante para dados;

- IDE (*identifier extension*, identificador de extensão): indica se o quadro é padrão ou estendido, com valor recessivo para estendido;

- SRR (*substitute remote request,* substituto da requisição remota): para manter a compatibilidade entre o quadro estendido e o padrão, no caso do formato estendido é um bit recessivo;

- R0 e R1: *bits* reservados;

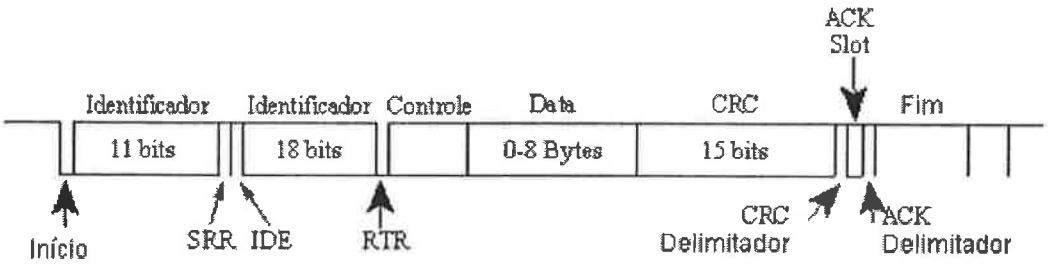

Figura 3.3 – Quadro 2.0B

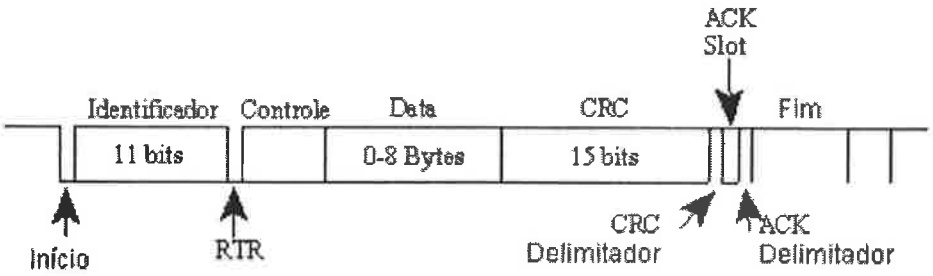

Figura 3.4 – Quadro 2.0A

O campo de identificação também é chamado de campo árbitro [28], sendo que este campo é utilizado para decidir quem continuará no barramento, agrupa os *bits* de identificação (onze bits) e o RTR no formato de quadro padrão. Os *bits* SRR, IDE e RTR fazem parte deste campo no formato estendido, sendo que a identificação possui onze bits na primeira parte e dezoito *bits* na segunda, totalizando vinte e nove *bits* para identificação.

O campo de controle agrupa os *bits* IDE, r0 e mais quatro *bits* do DLC (*data length code*), este último informa tamanho do campo de dados. Utilizando o quadro estendido, o campo de controle passa a agrupar os *bits* R1, R0 e DLC.

O campo de dados é um segmento de 0 a 8 *bytes* de dados.

O campo de CRC, composto pela seqüência numérica gerada, é utilizado para verificação de erros no quadro, e um *bit* delimitador, sempre recessivo.

Seguindo o campo de CRC tem-se o campo ACK, que possui dois *bits*. Verificando os *bits* enviados pelo nó transmissor, nota-se o envio de 2 *bits* recessivos, assim este campo delimita um tempo para que o receptor responda enviando um *bit* dominante caso receba o quadro com sucesso. Como no campo de CRC, o campo ACK é finalizado com um *bit* recessivo.

Para o fim do quadro tem-se o campo Final de quadro composto por sete *bits* recessivos. Esse campo terá a funcionalidade mais detalhada na parte de detecção de erros.

3.6.1.3 Quadro de Requisição Remota

Quando um nó necessita de algum dado, este nó envia um quadro remoto, uma requisição de dados. Este quadro remoto é idêntico ao quadro de dados, porém não contém o campo de dados, e como já citado no item anterior o bit RTR é recessivo.

3.6.1.4 Quadro de Erro

Possui um campo *flag* e um campo delimitador, este último consiste de oito *bits* recessivos, já o campo de *flag* é composto por seis bits. Os *bits* podem ser todos recessivos para um *flag* de erro ativo ou todos recessivos para um *flag* de erro passivo. A utilização do *flag* ativo ou passivo depende do estado de falha do nó.

3.6.1.5 Quadro de Sobrecarga

Um quadro de sobrecarga pode ser enviado por um nó para sinalizar que o receptor atual necessita de um tempo até o recebimento do próximo quadro, geralmente por falta de processamento ou memória.

O quadro é composto por seis *bits* dominantes para o campo de *flag*, e mais oito bits recessivos no campo delimitador do quadro.

Este quadro deve vem após um fim de quadro (para quadro de dados/remoto), um quadro de erro ou outro quadro de sobrecarga, porém deve-se respeitar a regra de no máximo dois quadros de sobrecarga consecutivos.

3.6.1.6 Espaço entre Quadros

O espaçamento entre quadros é sempre utilizado para separar quadro de dados ou requisição dos demais tipos de quadros. Este espaço é utilizado também para controle de transmissão de nós passivos.

O espaçamento de quadro é dividido em duas áreas para os nós ativos. A área de interdição, no qual um nó pode enviar um quadro de sobrecarga, é formada por três bits recessivos. A segunda área é aquela em que o barramento ficará livre até que ocorra alguma transmissão. Para um nó passivo existe uma área intermediária em que ele não transmitirá quadros, mas este tempo é requerido quando ele transmitiu um quadro. O tamanho desta área é de oito bits recessivos que são transmitidos pelo nó passivo responsável pela transmissão do quadro anterior. Os outros nós não são impedidos de realizarem transmissões neste tempo intermediário, caso isto aconteça o nó passivo pode tornar-se receptor da mensagem antes de completar o envio dos *bits* no barramento.

3.6.2 Filtragem e Validação de Mensagens

Para um nó receber uma mensagem transmitida no barramento necessitará que o seu filtro esteja preparado para receber esta mensagem. O filtro é feito pelo identificador.

A comparação entre o identificador esperado e o identificador recebido é realizada por mascarás. Para o receptor, quando uma mensagem válida é encontrada, ou seja, não ocorreu erro até o *bit* final de quadro ser recebido, passou pelo filtro, o nó guardará os dados e sinalizará para que a mensagem seja recebida com sucesso.

3.6.3 Configuração de Sincronização e Velocidade

Esta é uma parte crítica da rede em que há uma má configuração resultará em uma degradação do desempenho da rede ou até uma falha da mesma. Para atingir a velocidade de transmissão requerida, existe uma grande combinação de configurações que devem ter seus valores respeitados e configurados de acordo com o meio de transmissão.

O CAN possui uma unidade temporal chamada "*time quantum*" ou tq, essa unidade é definida pelo selecionador de taxa de transmissão do subsistema CAN. Um bit transmitido no CAN possui a seguinte divisão, conforme figura abaixo [28][31]:

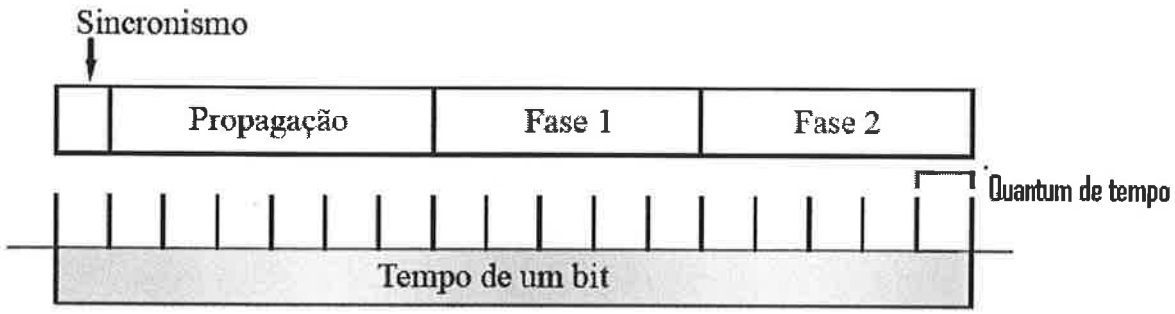

Figura 3.5 – Segmentos do *bit* transmitido

- Segmento de sincronismo: possui tamanho fixo de um tq, é utilizado para o sincronismo entre a entrada do barramento e o relógio do sistema.

- Segmento de propagação: compensa os atrasos causados pelo meio físico, incluindo tempo de transmissão, propagação do sinal no barramento e tempo para recepção do dado. Possui tamanho de um até oito tq.

- Segmento de fase 1: indicará onde será feita a aquisição do valor do sinal, ao qual está sendo lido no momento. Também pode ter tamanho de um até oito tq, este tamanho pode ser variável durante o funcionamento para sincronização dos relógios.

- Segmento de fase 2: transmissões ocorrerão após este segmento. Pode ser configurado com um até oito tq.

Tem-se ainda um fator de correção para a sincronização, que pode terá de um até quatro tq, e é limitado também por não poder ser maior que qualquer um dos segmentos de fase. Esta variável indica o quanto o ponto de aquisição pode ser movido para efetuar uma resincronização.

3.6.4 Tratamento de Erros

O CAN possui um sistema muito confiável de tratamento de erros, todos os erros globais ao sistema são detectáveis, como os erros locais ao transmissor. Uma mensagem pode conter até cinco erros distribuídos aleatoriamente, rajadas de erros com comprimento máximo de quinze bits ou de tamanho ímpar são detectados. Vejam abaixo os cinco tipos de erros detectáveis [28]:

- erro de *Bit*: todo emissor monitora os dados do barramento durante a transmissão, caso o bit monitorado no barramento, tenha valor diferente do que estava sendo enviado, é sinalizado este tipo de erro. Este erro não é levado em consideração, caso o emissor transmita dados do campo árbitro, neste caso, o emissor, cuja detecção apontou o erro, perde o controle do barramento. Existe ainda a possibilidade de o emissor enviar um quadro de erro com *flag* passivo, assim, caso o *bit* monitorado seja diferente, ele será ignorado;

- erro de codificação: este erro acontece quando o *bit* monitorado tem o mesmo valor seis vezes, na sexta ocorrência o erro é sinalizado;

- erro de CRC: caso o valor do campo CRC transmitido não for igual ao do CRC calculado no receptor, este erro sinalizará;

- erro de formação: ocorre quando um campo de formato pré-definido possui um ou mais *bits* ilegais. Caso o valor do DLC seja maior que oito, não existe uma sinalização de erro definida, geralmente o erro é ignorado e o valor é dado como o maior possível;

- erro no campo ACK: sinalizará sinalizado este erro caso o transmissor não detecte um bit dominante durante a transmissão do campo ACK;

A sinalização de erro é aplicada igualmente a todos os tipos, exceto para o erro de CRC. Enquanto a sinalização padrão é transmitida no próximo bit após a detecção do erro. O erro de CRC é transmitido somente depois do recebimento do delimitador no campo ACK.
Para controle do erro no barramento, o confinamento de falhas em cada nó, possui um contador de erros de transmissão e recepção. Com base nestes dois contadores o nó fica definido como:

- ativo: nó em funcionamento normal;

- passivo: nó funcional, porém com certas restrições na transmissão de pacotes. O nó entra neste estado, após o contador de erros de transmissão ter o valor maior ou igual a 128. Um nó entrando neste estado somente voltará a ser um nó ativo, caso os contadores de erros de transmissão e também de recepção estejam com valores inferiores a 128;

- inativo: este nó não pode exercer modificações no estado do barramento, caso o contador de erros de transmissão ultrapasse ou iguale o valor 256, ele deve entrar neste estado. Deste estado, um nó, tornar-se-á ativo após ocorrerem 128 pacotes contendo onze *bits* recessivos, ao voltar ao estado ativo, o nó, terá os contadores de erros zerados.

3.7 - Análise matemática do comportamento temporal do CAN

O número de estudo e trabalhos ampliaram-se intensamente no decorrer dos anos, pois a cada dia que passa os sistemas tornam-se mais complexos e interagem com itens que exigem muita segurança, como por exemplo, os sistemas "*x-by-wire*".

A previsibilidade no uso do CAN é um pouco complexo, pois as mensagens que possuem alta prioridade poderão acessar o barramento em primeiro lugar desde que o mesmo se encontre em repouso, caso contrário deve esperar a transmissão ser finalizada para poder ter acesso. No caso das mensagens de baixa prioridade o tempo de acesso ao barramento está sujeito à disponibilidade do mesmo, que é muito variável dependendo muito das características de prioridade das mensagens da aplicação.

Dentro de toda essa dificuldade de previsibilidade do tempo de transmissão e resposta das mensagens, podemos analisar através de cálculos matemáticos quais serão os piores tempo possíveis de transmissão de cada mensagem.

Abaixo se encontra um resumo do que foi publicado por Tindell em 1995 [6][7], que descreve as equações de análise do modelo matemático do barramento CAN, conforme será visto a seguir.

O tempo de transmissão (R_m) de uma mensagem (m) em uma rede com protocolo será definido pela equação 3.1 como segue:

$$R_m = J_m + W_m + C_m \quad (3.1)$$

C_m representa o tempo gasto para transmitir uma mensagem (m) fisicamente sobre o barramento. W_m representa o atraso na fila no pior caso. J_m representa o *Jitter* da mensagem e é obtido empiricamente como sendo $J_m = 0{,}1$ mili-segundo (ms).

O quadro de dados de uma mensagem CAN contém uma quantidade (T) de *bits* de *overhead* por mensagem e largura de *stuff* de cinco bits, sendo apenas que um número (O) desta quantidade (T) de *bits* de *overhead* que está sujeito ao *stuffing*. Assim, para cada um dos termos da equação 1 apresentada, são definidas as equações 3.2, 3.3, 3.4 e 3.5 a seguir:

$$C_m = \left(\left[\frac{T_m + 8.S_m}{5} \right] + O + 8.S_m \right) \tau_{bit} \quad (3.2)$$

sendo:

T_m = quantidade de *bits* de *overhead* sujeita ao *bit* "stuffing"

O = número máximo de bits de "overhead" por mensagem

S_m = tamanho limitado da mensagem (m) em *bytes* (tamanho máximo de 8 *bytes*)

τ_{bit} = tempo necessário para transmitir um *bit* sobre o barramento

Na tabela abaixo são apresentados os valores de algumas das variáveis da equação 3.2.

Tabela 3.1 – Valores das Variáveis das equações para os tipos de mensagens CAN

Variável nas Equações	CAN 2A Standard	CAN 2B Extended
T	34	54
O	47	67

$$W_m = B_m + \sum_{\forall j \in hp(m)} \left[\frac{W_m + J_j + \tau_{bit}}{T_j} \right] . C_j \quad (3.3)$$

Onde:

hp (m) = conjunto de mensagens no sistema com prioridade maior que a mensagem (m);

B_m = tempo de bloqueio no pior caso da mensagem (m);

T_j = período de uma mensagem (j);
J_j = "*Jitter*" de uma mensagem (j);

$$B_m = \max_{\forall k \in lp(m)} (C_K) \quad (3.4)$$

lp (m) = conjunto de mensagens com prioridade menor que a mensagem (m). Se a mensagem (m) for à de menor prioridade, então Bm é zero. O s valores de C_j, C_k são obtidos da equação 3.2.

Para que a equação 3.3 possa ser utilizada, é necessária uma relação de recorrências, com valor inicial $W_M^0 = 0$ que resulta na equação 3.5.

$$W_m^{n+1} = B_m + \sum_{\forall j \in hp(m)} \left\lceil \frac{W_m^n + J_j + \tau_{bit}}{} \right\rceil C_j \quad (3.5)$$

O conjunto de equações apresentadas é um modelo matemático que se utilizará para análise de redes CAN sob configurações e parâmetros determinados para aplicações na área automotiva.

4 – SISTEMAS DE TEMPO REAL

O avanço tecnológico das últimas décadas colocou de forma definitiva os sistemas eletrônicos no cotidiano da vida moderna, seja para um sistema veicular, bancário, de transporte, de saúde, de comunicação, de manufatura, entre tantos outros. Se de um lado tal inserção traz inúmeros benefícios às nossas vidas, de outro coloca novos desafios tecnológicos para a construção de sistemas computacionais mais confiáveis e seguros, sem os quais prejuízos e desastres tornar-se-ão inevitáveis.

O sistema de tempo real interage diretamente com o ambiente, não somente por meio de uma interface homem-máquina, mas, sobretudo através de dispositivos que captam informações e que interferem no ambiente, chamados de sensores e atuadores. Exemplos desses sistemas vão desde um simples controle de temperatura da água do motor até controle automático de vôo. Nesse sentido, o termo previsibilidade que se refere à capacidade do sistema em executar ações desejadas dentro de intervalos de tempos determinados, normalmente ditados pelo próprio ambiente no qual o sistema computacional está imerso [32]. Por exemplo, o controle de pouso de um avião tem que acionar o trem de pouso tão logo seja detectado a proximidade do solo através de um sensor de altitude.

A possibilidade dos vários componentes de um sistema de tempo-real estarem distribuídos em diferentes sistemas eletrônicos e se comunicarem por meio de protocolos de comunicação, para a realização de tarefas, coloca uma dificuldade adicional na garantia da previsibilidade, uma vez que o mau funcionamento de um simples componente ou do meio de comunicação causará violação de prazos ou mesmo a interrupção total do serviço.

Dentro desse contexto, define-se um sistema como previsível se o seu comportamento funcional e temporal será de algum modo antecipado ou previsto. O comportamento funcional refere-se ao resultado do processamento do sistema de controle, como no acionamento do trem de pouso do exemplo citado. Já o comportamento temporal refere-se ao instante em que determinada ação é produzida (observe que o trem de pouso tem que ser acionado antes da aterrissagem). Para que o sistema funcione de forma previsível é necessário em primeiro lugar que haja um gerenciamento adequado dos recursos disponíveis, a fim de que ações de tempo-real não sofram atrasos imprevistos, por exemplo, devido à indisponibilidade de memória na CPU, banda passante na rede de comunicação ou mesmo a execução simultânea de diferentes tarefas, acionadas a partir de estímulos diversos do ambiente.

Havendo a garantia de que os recursos necessários para a execução do sistema estão disponíveis, o único impedimento que sucederá é a ocorrência de defeitos nos componentes sejam eles no software ou no hardware. De fato, uma das poucas certezas que existe na área eletrônica micro controlada é sobre a impossibilidade de se construir sistemas totalmente

imunes a defeitos, ou seja, totalmente previsíveis. O que pode ser feito, entretanto, é minimizar a ocorrência desses defeitos através de mecanismos e técnicas diversas, com a seguir:

- Técnicas para diminuir a possibilidade de se introduzir falhas já na etapa de especificação do sistema, por exemplo, especificação e verificação formal do *software* e *hardware* dos componentes;

- Técnicas que introduzem robustez no sistema de tal forma que falhas, mesmo que ocorram em tempo de execução, podem ser toleradas sem comprometer a previsibilidade, por exemplo, com a replicação de componentes no sistema. Vale ressaltar que o esforço e recursos depreendidos para se verificar e colocar redundância no sistema estão diretamente relacionado ao custo advindo da violação de sua previsibilidade. Os sistemas de segurança críticos, como controle de plantas nucleares e de vôo de aeronaves tripuladas, requerem uma previsibilidade próxima de cem por cento. Por outro lado, ações de sistemas convencionais de um veículo como por exemplo, rádios, sistema de navegação e ar-condicionado, podem ser realizadas num intervalo de tempo muito menos rígido. Sendo assim, não há a necessidade de se garantir à previsibilidade em tais sistemas, apesar de ser imprescindível assegurar a realização destas tarefas.

Garantir a previsibilidade de um sistema de tempo-real não é tarefa simples. Pode-se citar alguns fatos que envolvem essa tarefa:

- especificar as funções do sistema de tal forma que tanto seu comportamento funcional quanto o temporal estejam precisamente definidos;

- usar ferramentas adequadas e linguagens de programação para facilitar a implementação dessas especificações;

- prover o gerenciamento dos recursos computacionais disponíveis a fim de que o sistema cumpra sua especificação;

- quando necessário, prover redundância dos componentes do sistema para aumentar a confiança no seu funcionamento.

4.1 Conceitos Básicos de Sistemas de Tempo-Real

Aplicações de tempo-real precisam expressar seus requisitos em função do tempo, o que inclui momentos para disparar determinados eventos, determinação dos momentos em que eventos ocorreram, medição da duração de operações, ordenação temporalmente mensagens recebidas, etc. Portanto, é imprescindível a existência de uma referência de tempo comum para todos os elementos envolvidos na consecução das tarefas, e isso exige a implementação de um sistema de relógios sincronizados entre si (sincronização interna) e em relação a uma referência externa (sincronização externa). A sincronização interna é suficiente quando os requisitos das aplicações envolvem apenas medidas de duração de operações. Contudo, se as medidas precisam ser relacionadas às horas do dia, então a sincronização externa precisa ser realizada.

A seguir, introduziremos outros aspectos de projetos relacionados à garantia de previsibilidade de sistemas de tempo-real.

4.2 Tarefas de Tempo-real e seus Atributos

Um sistema eletrônico de tempo-real típico controla ou monitora elementos do mundo real, também denominado de ambiente, representados pelos sensores, atuadores e operadores do sistema. O sistema microcontrolado de tempo-real reagirá a eventos oriundos das entidades que pertencem ao ambiente, como mudança da temperatura da água do motor, comando do motorista (através da interface homem-máquina) ou, até mesmo, a própria passagem do tempo [32].

Para sistemas que possuem diversas tarefas podemos descrever essas tarefas através de atributos, conforme segue abaixo [32] [33]:

- *Deadline* – é a especificação temporal de um sistema de tempo-real, na maioria das vezes são expressas em função do intervalo máximo de tempo que uma tarefa será executada. Esse intervalo de tempo é relativo ao início da execução da tarefa. Para cada tarefa t_i, seu *deadline* relativo será representado por D_i. Existem algumas tarefas que não podem violar seus *deadlines*, pois gerariam resultados incorretos.

Tarefas de *deadline* rígido estão associadas a serviços críticos, que envolvem vidas humanas ou grandes quantidades de recursos financeiros denominados tarefas críticas. Veja seus limites temporais representados na figura 4.2.

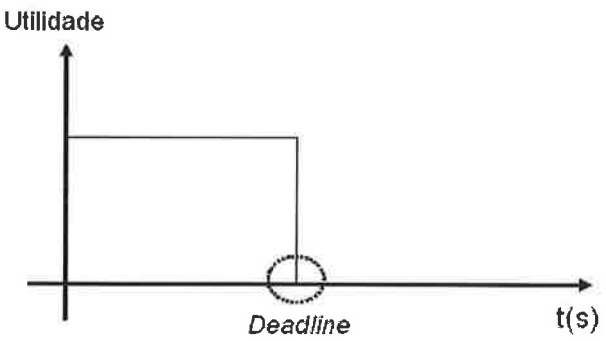

Figura 4.2 – Restrição temporal firme

Os sistemas que utilizam esse tipo de restrição da figura 4.2 são chamados de sistemas de tempo-real críticos (*safety critical systems*), como por exemplo, controle de vôo e freio eletrônico de um veículo (Brake-by-wire), que possuem alto nível de previsibilidade.

Quando não há tarefas com *deadline* rígido no sistema, denomina-se como sistema de tempo real brando (*soft real-time systems*). Descumprir *deadlines* nesse caso é aceitável em termos de custos para a aplicação ou para o usuário, embora não seja desejável, como exemplo, os

sistemas de entretenimento de um veículo (rádio, vídeo *game,* etc.). Veja seus limites temporais representados na figura 4.3.

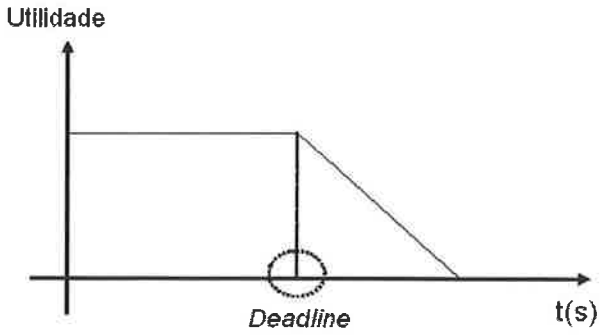

Figura 4.3 – Restrição temporal suave

- Período - este atributo refere-se ao tempo de ativação das tarefas no sistema, representado por T_i para cada tarefa t_i. Se há um período de ativação da tarefa conhecido, a tarefa é chamada de periódica. Algumas tarefas são chamadas de esporádicas, geralmente para essas tarefas o período significa o mínimo tempo entre duas ativações consecutivas. Quando nada se sabe sobre as ativações de uma tarefa, chama-se de aperiódica.

- *Jitter* - esse atributo representa a variação máxima de duas ativações consecutivas que uma tarefa periódica ou esporádica apresenta. Por exemplo, se uma tarefa periódica T_i possui *jitter* $J_i = 0$, então ela é ativada no sistema a cada T_i unidades de tempo. Considerar esse atributo é importante, pois variações no intervalo entre ativações de uma tarefa podem afetar o comportamento temporal do sistema.

- Tempo máximo de execução - esse atributo representa o tempo de execução da tarefa no pior caso, considerando uma dada arquitetura. Essa informação é fundamental para prever o comportamento temporal do sistema, por exemplo, para saber se todas as tarefas terminam antes de seus deadlines. Esse tempo é derivado por técnicas especiais [34] que levam em

consideração detalhes do sistema do micro controlador *hardware* e *software* cujas tarefas serão executadas. Representa-se por C_i o tempo máximo de execução da tarefa t_i.

Dado que os atributos acima são conhecidos, é necessário saber como as tarefas serão ativadas no sistema. Há duas abordagens conhecidas:

- orientada a eventos (*event-triggered*), consiste em ativar as tarefas tão logo os eventos associados sejam percebidos pelo sistema [35].

- tempo pré-determinado (*time-triggered*), suas ativações ocorrerão em momentos específicos no tempo.

Para comparar ambas as abordagens, suponha que uma tarefa está associada ao evento temperatura do motor a 120 graus. Na primeira abordagem, tal tarefa será ativada tão logo o sistema computacional perceba o evento através da sua interface. Na segunda, contudo o sistema somente acionará a tarefa em um momento pré-definido, independente da ocorrência do evento.

4.2.2 Escalonamento

Quase todos os subsistemas em uma aplicação possuem uma interação ao longo do tempo. Isso implica alguns requerimentos, pois muitas funções precisam ser adicionadas e/ou modificadas ao longo do tempo. Uma arquitetura com escalonamento será capaz de acompanhar estas alterações.

As aplicações automotivas com protocolo CAN utilizam prioridade de escalonamento fixa que se iniciou com a publicação do algoritmo *Rate-Monotonic* (RM) [36], uma simples heurística de atribuição de prioridades, que se dá em ordem inversa do período das tarefas. Quanto maior o período da tarefa menor é sua prioridade. Atribuir prioridades fixas às tarefas, tomando como parâmetro seus *deadlines,* é uma outra política clássica de escalonamento

chamada de *Deadline-Monotonic* (DM) [37]. De acordo com DM, prioridades são atribuídas em ordem inversa de *deadlines*.

5 – SISTEMAS *X-BY-WIRE*

Com a introdução da arquitetura eletrônica distribuída e um protocolo de comunicação, a interação entre sistemas eletrônicos e suas variáveis aumentou, possibilitando a introdução de sistemas bastante sofisticados, como é o caso dos sistemas *x-by-wire*, no qual as tarefas de controle de um automóvel, aeronaves e locomotivas que eram realizadas por sistemas mecânicos e ou hidráulicos estão sendo substituídas, gradativamente, por sistemas eletrônicos inteligentes, beneficiando-se os custos, manutenção, volume e peso, entre outros fatores [38]. Estes sistemas inteligentes são, por exemplo: controle de direção veicular, controle de freios, controle de *flaps* de uma aeronave e controle de aceleração.

Tais sistemas são denominados *control-by-wire*, ou seja, controle por fios conhecido por *x-by-wire*. O termo x em *x-by-wire* representa o nome do aplicativo mecânico que passa a ser controlado eletronicamente, como no caso de um sistema de direção sem acionamentos mecânicos (do motorista até às rodas) que se denomina de *steer-by-wire* [39], para o sistema de freios, *brake-by-wire* [40][41], para um sistema de controle de *flaps* das asas de um avião é chamado de *fly-by-wire*, e assim sucessivamente.

A principal diferença dos sistemas *x-by-wire* para os sistemas convencionais, está relacionada à confiabilidade, pois são aplicativos de sistemas de segurança críticos, que necessitam certo grau de tolerância à falhas e com latência garantida [42].

Existe uma tendência na indústria automobilística do aumento da necessidade de confiabilidade dos sistemas eletrônicos que são diretamente responsáveis pela segurança ativa e passiva do veículo. Estas aplicações aumentarão a segurança dos veículos, pois liberam o

motorista de tarefas rotineiras e ajudam o motorista a encontrar soluções em situações críticas. Na figura 4.2 apresentamos um diagrama básico um sistema *brake-by-wire*.

5.1 - Brake-by-wire

O aplicativo *Brake-by-wire* (BBW) surgiu como um novo e promissor sistema de controle de frenagem, e consiste em sensores e atuadores eletromecânicos interligados através de rede de comunicação, ao invés de componentes hidráulicos ou eletro-hidráulicos convencionais de freios. Isso oferece melhor segurança e conforto com uma redução de custo quando utilizado para substituição por completo dos sistemas mecânicos.

A atuação dos sistemas de freio é dada por quatro atuadores que estão em cada uma das rodas, oferecendo uma frenagem bastante eficiente e um melhor controle de estabilidade, alem das vantagens descritas abaixo:

- eliminação das complexas e pesadas peças mecânicas e hidráulicas;

- maior eficiência e estabilidade do freio, devido ao rápido e preciso controle de torque de frenagem gerado pelos motores elétricos;

- reforço da capacidade de diagnóstico do sistema de travagem;

- fácil adaptação dos sistemas de assistência, por exemplo, *antiblock system* (ABS) e programa eletrônico de estabilidade (ESP), sem adicionar componentes mecânicos ou componentes hidráulicos;

- a redução de custos em fases de concepção, construção, montagem e manutenção;

- economizando espaço e menor peso;

- eliminação das preocupações ambientais associados a tradicionais sistemas de travagem hidráulicos.

No exemplo da figura 5.1, temos um sistema eletrônico de freio com um micro controlador e um atuador para cada roda e no pedal de freio. Esse sistema aumenta excessivamente a

precisão e qualidade da frenagem. Neste exemplo, o sistema eletrônico possui redundância de comunicação, para evitar problemas que possam ocorrer na rede, porém atualmente é reduzido o número de empresas que utilizam esse tipo de sistema. A maioria dos aplicativos como *brake-by-wire* e *steering-by-wire*, não possuem redundância na rede de comunicação, pó isso utilizam *backup* mecânicos que suprem problemas que possam ocorrer.

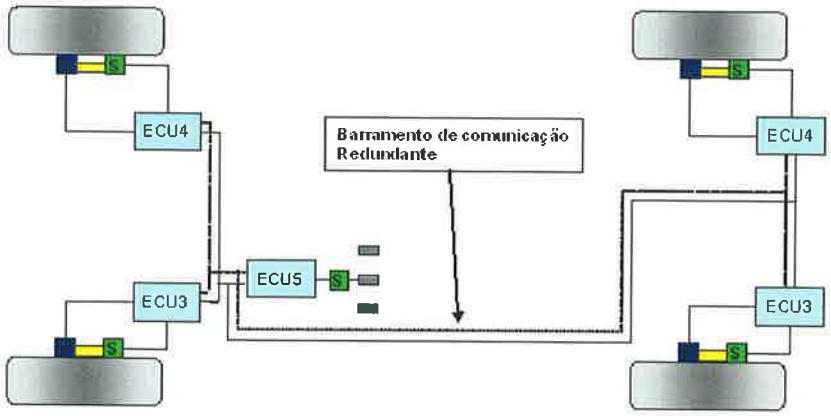

Figura 5.1 – Sistema *brake-by-wire*

5.2 Requisito dos sistemas que exigem segurança critica (*x-by-wyre*)

Atualmente os sistemas eletrônicos hoje são utilizados na maioria das vezes, para sistemas de conforto ou funções que não precisam de um rigoroso requisito de confiabilidade. No entanto, existe uma demanda para aumentar as funções nos veículos, por meio da introdução de sistemas que ajudam o condutor a lidar com situações críticas. Esse tipo de aplicação exige sistemas de comunicação confiáveis com previsibilidade. Abaixo, estão descritos alguns requisitos básicos para aplicações que possui funções de segurança críticas [41]:

- A comunicação através da rede deve ser determinística, por exemplo, as mensagens são entregues com pequenas e delimitadas latências. Essas latências precisam ser conhecidas para que o sistema efetue as devidas correções;

- Todas as partes do sistema (diferentes fornecedores quando existentes) devem ser testadas individualmente. Esse recurso reduz os esforços na fase de integração do sistema;

- Quando um simples componente falha devido a erros, envelhecimento, ou outras causas, existe a necessidade de introduzir tolerância à falhas, ou seja, cada componente de hardware é replicado em pelo menos uma vez. O sistema de comunicação deve suportar distribuição de replicas para evitar falha de modo comum, por exemplo, uma única falha não pode afetar todos os nós de réplica;

- O barramento de comunicação deve ser vigiado por um dispositivo independente, evitando *"blabbling idiota"*, ou seja, um nó permanentemente monopolizando a rede. Este requerimento implica a utilização de *"time-triggered"* protocolo;

- O malha de controle deve ser sincronizada, portanto requer uma base de tempo comum para todos os participantes. A comunicação precisar ser capaz de distribuir um *"clock"* sincronizado com uma precisão de micro segundos;

- Todos participantes da rede precisam conhecer as propriedades ou falhas de todos os nós. Este tipo de função é denominado *"membership service"*;

- O sistema de comunicação deve possuir *"gateways"* para diferenciar sistemas relacionados e não-relacionados com aplicações, tanto em alta como em baixa velocidade.

Todos os itens citados anteriormente devem ser analisados no desenvolvimento de um novo projeto.

6. ESTUDO DE CASOS

Observando toda essa evolução dos sistemas eletrônicos que exigem maior confiabilidade e previsibilidade, será analisada por meio de dois estudos de caso a situação real de uma rede de comunicação CAN de um veículo. A primeira parte é para avaliar as características gerais do funcionamento da rede CAN, em uma arquitetura eletrônica distribuída em diversas configurações e em diferentes valores de *busload* para cada uma das medições, sendo provocado pela modificação do tempo de transmissão de algumas mensagens. O segundo estudo de caso é para verificar o comportamento de um sistema de controle em malha fechada sobre a rede, e visa o impacto do atraso temporal da rede no sistema realimentado.

6.1 Primeiro estudo de caso

Como já foi visto nos capítulos anteriores, o protocolo de comunicação CAN é o sistema mais utilizado em aplicações automotivas, devido suas características de robustez e flexibilidade. Neste estudo de caso serão analisadas as características temporais deste protocolo influenciadas pelo Busload em diferentes configurações mostrando de forma bastante prática a flexibilidade do CAN.

6.1.1 Definições do experimento

Para analisar as características reais de um barramento CAN, utiliza-se uma rede com taxa de transmissão de 500Kbps e com 11 bits (CAN 2.0 A) para os identificadores. Dentro desta rede disponibiliza-se 22 mensagens, transmitida periodicamente, de acordo com sua prioridade e o tempo pré-definido (*time-triggered - capitulo* 4, item 4.2) que está na primeira coluna de cada teste da Tabela 6.1.

Na segunda coluna da tabela 6.1 está a quantidade de dados de cada mensagem. Essa informação foi utilizada para calcular o tamanho do pacote da mensagem, e também para saber qual é o tempo que a mensagem utiliza o barramento.

Com as equações de Tindell [7], conforme item 3.7, calcula-se o tempo de transmissão das mensagens e a porcentagem de utilização da rede (*Busload*) que estão na primeira coluna de cada teste.

Os valores reais foram medidos fisicamente no barramento por meio de um analisador CAN da empresa Vector e estão na segunda coluna de cada teste. A ferramenta CANalyser calcula o *busload* seguinte forma (informações do fabricante do equipamento do CANalyser, empresa Vector CANtech):

- mede-se três segundos do estado do barramento;
- separa-se o tempo do barramento em estado ativo, quando está transmitindo, e estado passivo, quando está em repouso.
- a partir do *bit* "*start of bit*" conta-se com estado ativo e após os 7 bits do "*end of bit*" como estado passivo.
- a fórmula do *busload* é:

$$Busload = \frac{Periodomedido - periodoemrepouso}{Periodoativo}$$

A última coluna de cada teste representa a diferença do tempo de transmissão desejado para o medido. Quando o valor é positivo, significa que a mensagem antecipou-se e caso seja negativo a mensagem atrasou-se.

A diferença entre o primeiro e o segundo teste está na modificação do tempo de transmissão de duas mensagens, uma do identificador 596 que transmitia sua mensagem em períodos de 10ms e passou a transmitir em 1ms, e a outra do identificador 1064 que transmitia em 100ms e passou a transmitir em 50ms, visualizando um *busload* intermediário na rede na faixa dos 60%. Com isso mostraremos a flexibilidade do CAN quando existe alteração no tempo de transmissão das mensagens.

A diferença entre o segundo e o terceiro teste está na modificação do tempo de transmissão de duas mensagens, uma do identificador 772 que transmitia sua mensagem em períodos de 20ms e passou a transmitir em 2ms, e a outra do identificador 884 que transmitia em 20ms e passou a transmitir em 50ms, isso para visualizar um *busload* superior a 75% na rede, esse teste visa mostras algumas das conseqüências da flexibilidade do CAN no comportamento temporal da rede.

Tabela 6.1 – Dados e resultados das configurações de cada experimento

		1º Teste			2º Teste			3º Teste		
Idetificadores	N° de Bytes de Dados	Período teórico das mensagens (ms)	Medição do período médio real das mensagens na rede (ms)	Diferença do tempo teórico para a medição real (ms)	Período teórico das mensagens (ms)	Medição do período médio real das mensagens na rede (ms)	Diferença do tempo teórico para a medição real (ms)	Período teórico das mensagens (ms)	Medição do período médio real das mensagens na rede (ms)	Diferença do tempo teórico para a medição real (ms)
592	8	1	1,0000	0,0000	1	1,6080	-0,6080	1	0,8010	0,1990
596	7	10	9,6390	0,3610	1	1,6020	-0,6020	1	0,8010	0,1990
772	8	20	19,1820	0,8180	20	21,0750	-1,0750	2	2,2530	-0,2530
848	6	20	20,5320	-0,5320	20	19,1930	0,8070	3	3,3240	-0,3240
900	8	20	18,5770	1,4230	20	21,0750	-1,0750	20	17,8430	2,1570
901	7	20	19,4500	0,5500	20	19,7340	0,2660	20	18,9160	1,0840
1064	7	100	49,7540	50,2460	50	47,3170	2,6830	50	53,1490	-3,1490
1104	6	50	47,3500	2,6500	50	48,6580	1,3420	50	51,5500	-1,5500
1144	8	50	49,4840	0,5160	50	49,4630	0,5370	50	50,7390	-0,7390
1360	8	100	101,0520	-1,0520	100	100,5320	-0,5320	100	98,1290	1,8710
1362	8	100	99,4400	0,5600	100	100,5380	-0,5380	100	100,8060	-0,8060
1364	8	100	98,3830	1,6170	100	97,5930	2,4070	100	101,0840	-1,0840
1368	7	100	100,7760	-0,7760	100	101,0640	-1,0640	100	101,0760	-1,0760
1400	6	100	101,0500	-1,0500	100	98,6560	1,3440	100	98,3980	1,6020
948	8	20	19,3010	0,6990	20	19,9988	0,0012	20	19,3020	0,6980
949	8	20	20,1070	-0,1070	20	19,9957	0,0043	20	19,1980	0,8020
1440	8	100	99,9720	0,0280	100	99,9850	0,0150	100	100,0230	-0,0230
1441	6	100	99,9190	0,0810	100	100,4720	-0,4720	100	99,5570	0,4430
1448	2	100	100,0350	-0,0350	100	99,9790	0,0210	100	100,1460	-0,1460
1696	4	100	99,9952	0,0048	100	99,9947	0,0053	100	999,6300	-899,6300
1716	8	100	99,9914	0,0086	100	99,9919	0,0081	100	999,9160	-899,9160
1717	8	100	99,9920	0,0080	100	99,9939	0,0061	100	1000,1270	-900,1270
	Bus Load	37,75	37,69%		59,77	60,05%		77,52	78,40%	

6.1.2 Análise dos resultados

No primeiro teste, a rede está com porcentagem de utilização baixa em torno de 37,69 %, conseqüentemente reduz excessivamente o efeito causado pela fila para transmitir. Com isso evita-se bastante o atraso das mensagens, podendo ser observado na coluna de diferenças entre valor de transmissão desejado para o medido de cada mensagem. Mesmo com um

"busload" baixo pode-se observar que houve um atraso em algumas mensagens, mostrando que o CAN não possui previsibilidade garantida.

Na segunda configuração deste experimento varia-se o tempo de transmissão de duas mensagens, aumentando a porcentagem de utilização da rede, causando uma variação maior no sistema, visto que o acesso ao barramento tornou-se mais difícil. No primeiro teste tivemos apenas seis mensagens atrasadas, já no segundo oito. Nesta configuração pode-se observar que o CAN possui uma flexibilidade explícita, visto que a alteração ocorreu individualmente em dois nós e automaticamente a rede toda se adaptou a essa mudança. O ponto frágil que se observa neste teste é a escalonabilidade não garantida que possibilitou a reconfiguração da rede, e alterou o tempo de transmissão de vários outros nós de menor prioridade.

Os resultados do terceiro teste mostram uma variação bastante expressiva no tempo de transmissão das mensagens, pois o *"busload"* atingiu um valor elevado, aumentando o tempo de fila, principalmente para as mensagens de baixa prioridade. As três últimas mensagens de menor prioridade obtiveram um atraso em média dez vezes maior que o tempo de transmissão desejado. Atraso nessas proporções pode trazer grandes conseqüências ao sistema, dependendo do tipo de aplicação.

Como já visto nos parágrafos acima, a diferença do primeiro teste para o segundo e do segundo teste para o terceiro, é apenas a alteração do tempo e de transmissão de duas mensagens, porém a variação nos resultados foi muito expressiva, considerando que os atrasos de algumas mensagens geram muitos transtornos para as aplicações. O CAN se mostrou bastante flexível, porém isso trouxe alteração nas características temporais da aplicação.

Na figura 6.1 observa-se que o atraso das mensagens está ligado diretamente a aumento da porcentagem de utilização da rede.

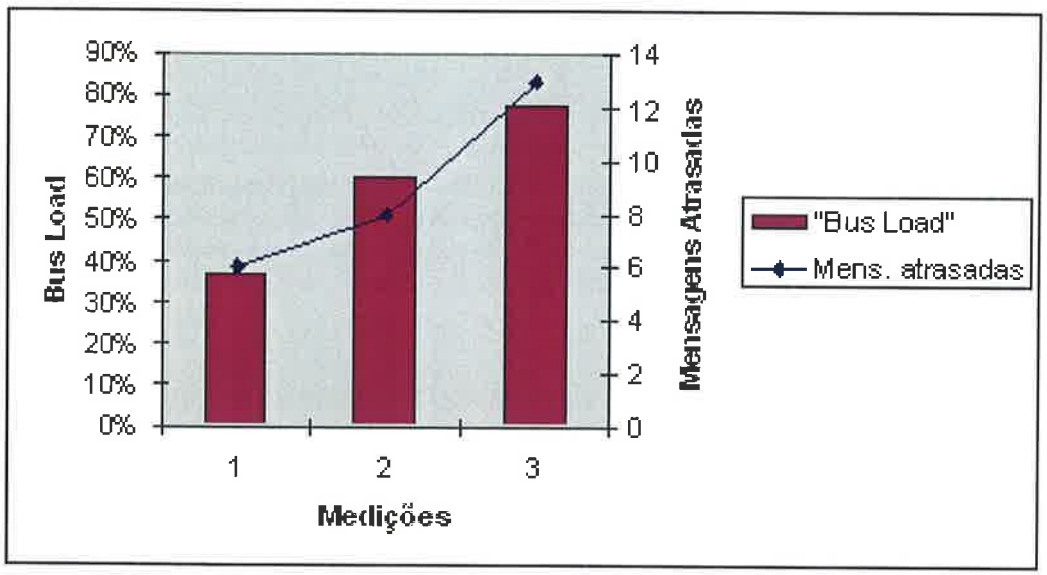

Figura 6.1 – Indicação da quantidade de mensagens atrasadas em função da porcentagem de utilização da rede cada teste.

Um dos principais motivos do atraso na transmissão do teste é gerado pelas características do protocolo CAN de detecção de colisão sem destruição (*Collision Detection with Non-Destructive Arbitration*). Cada sistema somente transmite sua mensagem quando o barramento está disponível e sua mensagem possui maior prioridade naquele instante. A mensagem não pode frequentemente ser transmitida no momento exato em que foi programada, pelos seguintes motivos:

- obedecer a uma ordem de prioridade;
- esperar uma mensagem terminar a transmissão;
- mensagem de erro na rede.

De acordo com o teste realizado, quando o *busload* estava com 37,69% observou-se um atraso em seis mensagens, quando estava com 60,05% o atraso ocorreu em oito mensagens e quando estava com 78,4, ocorreu atraso em treze mensagens, conforme figura 6.1. Na maioria dos testes, os atrasos relevantes ocorreram nas mensagens de baixa prioridade, mas dependendo do *busload* as mensagens de prioridade intermediária também são afetadas. Na

tabela 6.1, os identificadores 1669, 1716 e 1717 chegaram a atingir com o "*busload*" 78,4% um atraso com valor 10 vezes o tempo teórico do período.

O CAN trabalha com a priorização das mensagens pelo número do identificador, quando uma mensagem de maior prioridade precisa ser transmitida e o barramento já está ocupado com uma mensagem de menor prioridade, é necessário esperar o fim da transmissão desta mensagem para que a de maior prioridade inicie sua transmissão. Isso justifica o porquê mensagens de alta prioridade também apresentaram atrasos.

Outra funcionalidade do CAN que gera atrasos é a metodologia utilizada quando uma mensagem transmitida possui erros, sua retransmissão é realizada na seqüência. Caso essa mensagem possua a maior prioridade da rede, o barramento neste instante fica monopolizado, impossibilitando a transmissão de outras mensagens.

Na prática, muitas empresas automobilísticas utilizam protocolo padrão com dicionários de palavras universais que são projetados, considerando uma margem de segurança bastante razoável para o *busload*, porém quando o veículo é vendido não existe um controle exato da quantidade de equipamentos que serão montados em pós-vendas, as montadoras recomendam somente a instalação de equipamentos homologados. Esse tipo de instalação de opcionais pode gerar problemas gravíssimos, quando não segue os padrões homologados, ocasionando falhas graves no sistema e até risco de vida.

As sugestões para sistemas que exigem um alto grau de confiabilidade e estão conectados a uma rede CAN distribuída como a deste estudo de caso, são:

- Avaliar quais são limites de atraso permitido para cada uma das mensagens disponíveis na rede;

- Verificar se a rede receberá novos sistemas, ou seja, mais identificadores, pois com a introdução de outras mensagens o sistema pode ter efeitos parecidos com o observado neste estudo de caso, porque o *busload* da rede é alterado;

- Criar em um mesmo veículo várias sub-redes de alto, médio e baixo grau de confiabilidade, com isso isolaria os sistemas que exigem maior confiabilidade, gerando assim um controle mais otimizado da rede e eliminando possível de interferência de sistemas de baixa confiabilidade, o *busload* diminuiria bastante, o que aumentaria a previsibilidade da rede, veja exemplo na figura 6.2 a seguir;

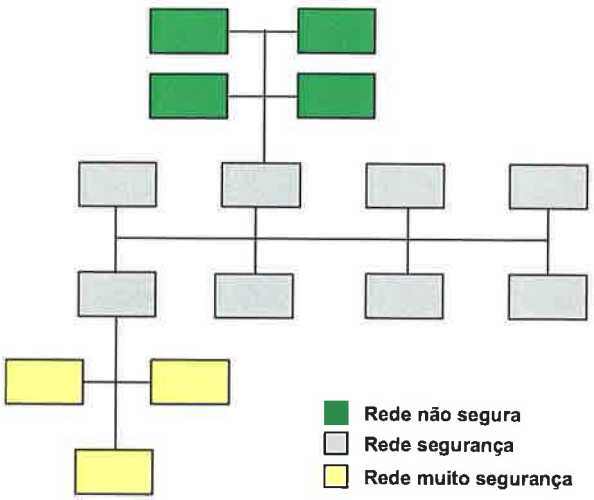

Figura 6.2 – Rede subdivida por nível de segurança.

- Outra solução é aumentar a taxa de transmissão do barramento, diminuindo o *busload* da rede, deixando o barramento com folga para minimizar os impactos quando à necessidade de introdução de novas mensagens;

- Utilizar outros tipos de protocolo como TTP/C [43], Byteflight [44] e FlexRay [45] que utilizam de guardas para as mensagens que exigem uma previsibilidade mais apurada.

6.2 Segundo estudo de caso

Como a maioria das redes CAN com arquitetura distribuída nos veículos, é composta de sistemas de controle conectado a rede, como no caso do sistema *brake by wire* que utiliza um

sensor de pressão no pedal de freio e um atuador em cada roda, aplica-se nesse segundo estudo de caso como atuador um motor DC [46] [47] e um sensor no barramento CAN (figura 6.3) conectados ao controlador através da rede. O barramento CAN é o mesmo utilizado no primeiro estudo de caso, considerando a mesma configuração de cada um dos testes.

Serão utilizadas duas mensagens de menor prioridade (tabela 6.1), a mensagem 1716 (ms1) para o sensor e a mensagem 1717 (ms2) para o controlador.

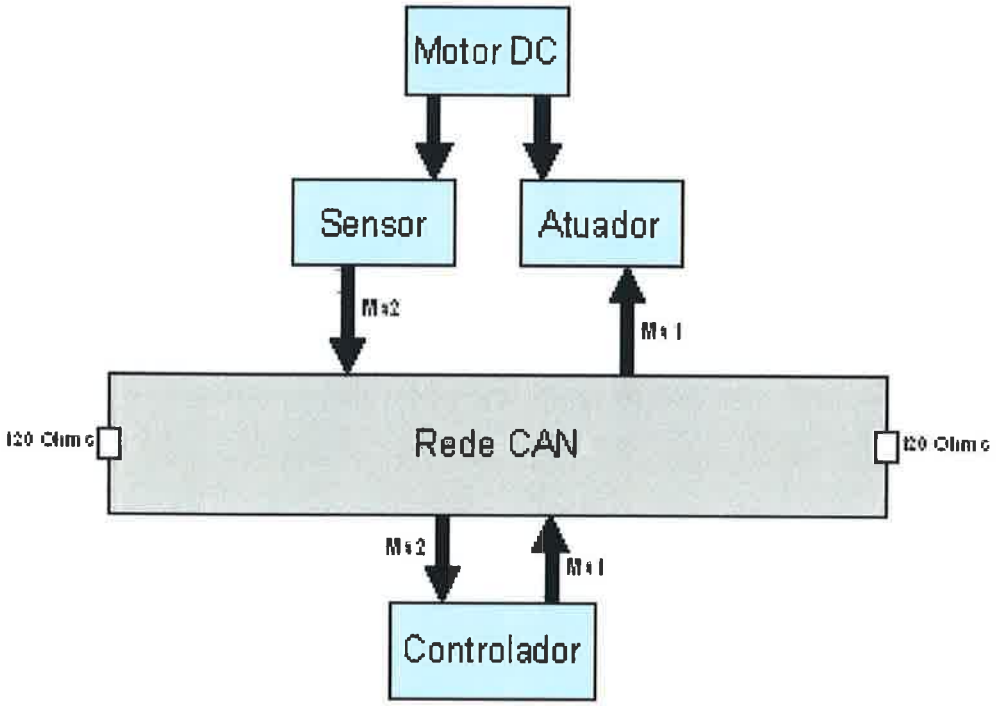

Figura 6.3 – Diagrama em bloco do sistema de controle em malha fechada no CAN.

6.2.1 Definições do experimento

Para esclarecer às interações do motor DC, sensor e atuador.; serão definidas as equações que descrevem o comportamento do motor DC de acordo com a figura 5.4

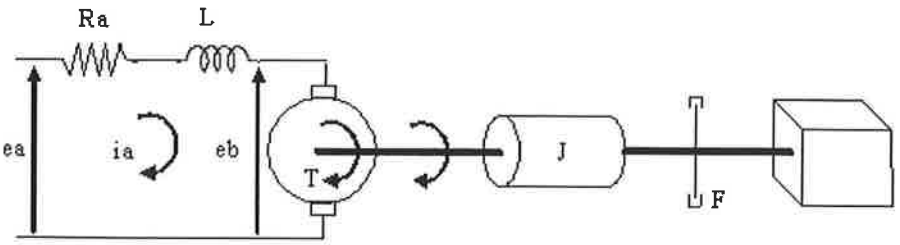

Figura 6.4 – Motor DC.

As equações que descrevem o modelo dinâmico do Motor DC são:

$$e_a = e_{b+} L_a \frac{di_a}{dt} + R_a i_a \quad (6.1)$$

$$J\frac{d^2\theta}{dt^2} + F\frac{d\theta}{dt} = q \quad (6.2)$$

Considerando e_b e q são proporcionais a $\dot{\theta}$ e i_a respectivamente, isto é:

$$e_b = k_1 \frac{d\theta}{dt} \quad (6.3)$$

$$q = k_2 i_a \quad (6.4)$$

Tomando a transformada de *Laplace* das equações 6.1 e 6.4 obtem-se por simples substituição:

$$\frac{\theta(s)}{E_a(s)} = \frac{k}{s[L_a J s^2 + (L_a F + R_a J)s + R_a F + k_2 k_1]} \quad (6.5)$$

Desprezando a indutância de armadura L_a, a função de transferência simplificada, para a posição do eixo como saída, resulta:

$$\frac{\theta(s)}{E_a(s)} = \frac{K}{s(\tau s + 1)} = G_p(s) \quad (6.6)$$

onde:

$$k = \frac{k_2}{(R_a F + k_2 k_{1)}} \quad (6.7)$$

$$\tau = \frac{R_a J}{(R_a F + k_2 k_1)} \quad (6.8)$$

Podemos representar o motor DC em diagrama de bloco, conforme figura 6.5:

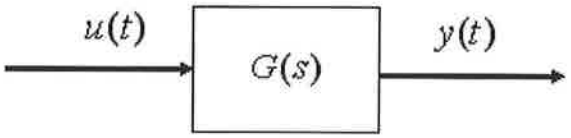

Figura 6.5 – Diagrama em bloco do motor.

Para estudo do sistema de controle na rede CAN, iremos utilizar mesmo motor utilizado na referência [43].

$$G_p(s) = \frac{2029{,}826}{(s + 26{,}29)(s + 2{,}296)} \quad [48]\ (6.9)$$

Para controle do motor utilizar-se-á um controlador PI, definido pela seguinte equação:

$$G_c(s) = \frac{\beta K_P [s + (K_1 / K_P)]}{s} \quad (6.10)$$

$K_P = 0{,}1701$ e $K_I = 0{,}378$

K_P é o ganho proporcional

K_I é o ganho integral

$G_p(s)$ é a planta do motor DC [48]

β é o parâmetro de ajuste do K_P e K_I ,neste caso iremos utilizar-se-á $\beta = 1$.

Para verificar o comportamento do sistema de controle sob a rede CAN simulou-se a função de transferência do controlador e motor. Conforme figura 6.6, que empregou os conceitos utilizando os conceitos utilizados no trabalho [49].

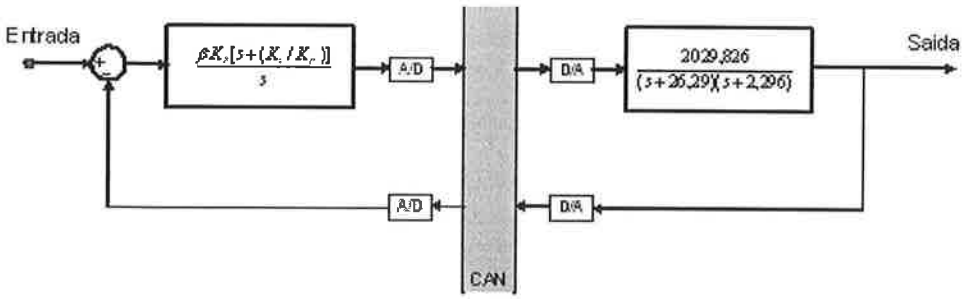

Figura 6.6 – Diagrama em bloco do sistema de controle no CAN.

Como já foi citado no item 4.2 existem duas formas de transmitir as mensagens, periodicamente ou por evento. Neste exemplo iremos utilizar mensagens periódicas. O comando de acionamento do motor está conectado ao controlador e é transmitido periodicamente através da Ms1, e o atuador possui um filtro para receber essa mensagem e acionar o motor.

O controlador do motor terá como referência para controlar o motor a mensagem Ms2 do sensor que será transmitida periodicamente para a rede. O controlador possui um filtro para receber e gerenciar por meio desse dado o controle do motor.

Com a introdução do sistema de controle na rede CAN, algumas variáveis temporais são adicionadas. Na figura 6.7 é apresentada essas variáveis como tarefas.

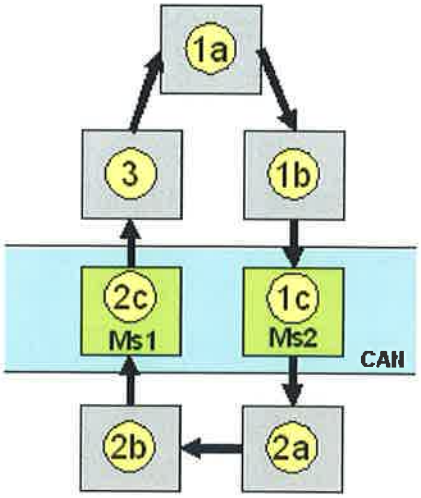

Figura 6.7 – Diagrama temporal do sistema de controle.

Tarefa 1a (T1a) - Medir a saída do motor DC e gerar a mensagem CAN com a informação.

Tarefa 1b (T1b) – Transmitir a mensagem do sensor (Tempo na fila para transmissão).

Tarefa 1c (T1c) – Transmissão da mensagem fisicamente no barramento.

Tarefa 2a (T2a) – Receber a mensagem do sensor e processá-la e gerar uma mensagem de correção para o atuador.

Tarefa 2b (T2b) – Transmissão da mensagem do atuador (Tempo na fila para transmissão).

Tarefa 2c (T2c) - Transmissão da mensagem fisicamente no barramento.

Tarefa 3 (T3) - Atuador recebe a mensagem do controlador e atua.

Cada uma das tarefas é considerada um atraso no *loop* de controle. Para esse estudo de caso vamos simular-se-á o atraso das T1a, T2a e T3 como zero, pois o tempo gasto para essas tarefas não estão ligados ao desempenho da rede CAN.

O tempo total do *loop* de controle é descrito pela equação:

$$\sum \tau_{total} = \tau_{T1a} + \tau_{T1b} + \tau_{T1c} + \tau_{T2a} + \tau_{T2b} + \tau_{T2c} + \tau_{T3} \quad (6.11)$$

como:

$\tau_{T1a} = 0$, $\tau_{T2a} = 0$ e $\tau_{T3} = 0$

$$\sum \tau_{total} = \tau_{T1b} + \tau_{T1c} + \tau_{T2b} + \tau_{T2c} \quad (6.12)$$

$\tau =$ tempo

Individualizando os atrasos para cada uma das mensagens pode-se dizer que:

atraso ms1 = $\tau_{T1b} + \tau_{T1c}$

atraso ms2 = $\tau_{T2b} + \tau_{T2c}$

Os tempos de transmissão fisicamente das mensagens no barramento τ_{T1c} e τ_{T2c}, são calculados utilizando as seguintes equações:

$$C_m = \left(\left[\frac{T_m + 8.S_m}{5} \right] + O + 8.S_m \right) \tau_{bit} \quad (6.13)$$

T_m = quantidade de *bits* de overhead sujeita ao *bit* "*stuffing*".

O = número máximo de *bits* de "*overhead*" por mensagem.

S_m = tamanho limitado da mensagem (m) em *bytes* (tamanho máximo de 8 *bytes*).

τ_{bit} = tempo necessário para transmitir um bit sobre o barramento.

C_m = tempo gasto para transmitir uma mensagem fisicamente sobre o barramento.

As variáveis T_m e O estão diretamente ligadas à classificação do CAN que estamos utilizando, conforme tabela 3.2.

Para este estudo de caso utilizaremos o CAN 2A de 500Kbps e o tamanho das mensagens de 8 bytes, portanto teremos:

$\tau_{bit} = 2\,\mu s$

T_m =34bits

O =47 bits

S_m = 8 bytes

portanto $C_m = 0{,}5\,ms$

Como as duas mensagens ms1 e ms2 utilizam o mesmo protocolo e possuem o mesmo tamanho de dados, podemos dizer que:

$\tau_{T1b} = \tau_{T2b} = 0{,}5ms$

Os tempos de atraso τ_{T1c} e τ_{2c} serão gerados pelo atraso na fila podem ser tirados da tabela 6.2 do primeiro estudo de caso, que é a diferença entre o valor teórico de transmissão da mensagem, para o valor real, abaixo estão aos dados retirados do primeiro estudo de caso:

Tabela 6.2 – Tempo na fila de transmissão das mensagens do sistema de controle

	1ª Configuração	2ª Configuração	3ª Configuração
Porcentagem de utilização da rede (%)	37,69%	60,05%	78,40%
Tempo de fila de Ms1 (ms)	-0,008	-0,008	899,916
Tempo de fila de Ms2 (ms)	-0,008	-0,006	900,127

Quando o valor é negativo, significa que a mensagem foi transmitida antecipadamente e para o valor positivo chegou atrasada.

As ms1 e ms2 serão transmitidas a cada 100ms (*time triggered*), por isso além do tempo gasto na fila para transmissão, o valor de 100ms será considerado um atraso na malha de controle.

Para a primeira configuração:

atraso total de ms1 = $\tau_{T1b} + \tau_{T1c}$ +100ms = 0,0005 -0,008 +100ms = 92,5 ms;

atraso total de ms2 = $\tau_{T2b} + \tau_{T2c}$ +100ms = 0,0005-0,008+100ms = 92,5 ms;

Para a segunda configuração:

atraso total de ms1 = $\tau_{T1b} + \tau_{T1c}$ +100ms = 0,5ms - 8ms + 100ms = 92,5 ms;

atraso total de ms2 = $\tau_{T2b} + \tau_{T2c}$ +100ms = 0,5ms - 6ms + 100ms = 94,5 ms;

Para a terceira configuração:

atraso ms1 = $\tau_{T1b} + \tau_{T1c}$ +100ms = 0,5ms + 899ms + 100ms = 999,5 ms;

atraso ms2 = $\tau_{T2b} + \tau_{T2c}$ +100ms = 0,5ms + 900ms+ 100ms = 1000,5 ms;

Quando os valores τ_{T1c} e τ_{T2c} da tabela IV são positivos significa que a mensagem chegou adiantada, portanto nesse caso não é considerado um atraso e por esse motivo o valor é somado com sinal negativo.

A figura abaixo está com a simulação feita no MATLAB/Simulink utilizando os tempos de atraso de fila de uma rede CAN real [49], empregada no primeiro estudo de caso.

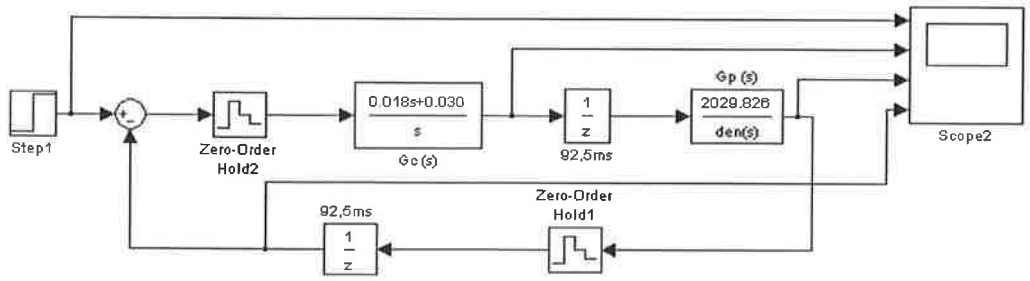

Figura 6.8 – Diagrama de simulação do sistema de controle no MATLAB/Simulink

Para estimular o sistema foi utilizada uma função degrau na entrada e os atrasos foram colocados nos blocos 1/z, como um tempo discreto de acordo com a equação 6.14 [49].

$$\frac{1}{z} = \frac{1}{e^{-Ts}} \cong \tau_{total} \quad (6.14)$$

Quando se introduz um sistema de controle de malha fechada na rede, adiciona-se um atraso que tem dimensões variadas, de acordo com a configuração do sistema. Para facilitar nossa análise será dividido esse atraso em dois grupos:

a) Atrasos Fixos – tempo gasto para o sistema de controle ler, processar e codificar as mensagens e disponibilizá-las para transmissão na rede, todavia não são influenciados pela configuração da rede CAN;

b) Atrasos variáveis – Tempo gasto na fila e para transmitir a mensagem fisicamente no barramento.

Vários são os motivos que fazem os atrasos variáveis modificar seus valores, pelo teste pode-se mencionar:

- **Taxa de transmissão** – utilizou-se neste teste uma taxa de transmissão de 500kbps, como o CAN pode ser utilizado até uma taxa de transmissão de 1Mbps, caso utilizássemos esse valor poderíamos diminuir o busload do nosso sistema.

- **Porcentagem de utilização da rede** – foram testados três tipos diferentes de porcentagem de utilização da rede, entre a primeira e segunda configuração da rede ocorreram atrasos nas mensagens, porém não foram tão agressivos, visto que estavam dentro de uma margem fácil de controlar. No caso da terceira configuração a mudança foi muito drástica, pois as mensagens ms1 e ms2 que foram utilizadas para nosso sistema de controle atrasaram em torno de 10 vezes o tempo programado;

- **Prioridade da mensagem** – observando a tabela completa com todas as mensagens, nota-se que as mensagens com maior prioridade não possuem grandes alterações no tempo de

transmissão em nenhuma configuração, mas no caso do sistema de controle que utilizava as mensagens de menor prioridade na rede foram as mais afetadas quando a rede estava com uma taxa de utilização muito alta;

- **Tamanho da mensagem** – o tamanho dos dados dentro da mensagem no protocolo CAN tem de zero até oito *bytes*, isso modifica o tempo de transmissão da mensagem fisicamente, porém pelos cálculos que fizemos no capitulo de definições deste estudo de caso, a diferença foi tão pequena que não altera muito os resultados;

- **Tipo de *frame* do CAN** – existem dois tipos de *frames* utilizado para transmissão de mensagens, o CAN 2A que possui 11 *bits* de identificadores e o CAN 2B com 29 *bits*, essas diferenças em torno de 18 *bits* modificam o valor do tempo de transmissão da mensagem fisicamente no barramento, porém conforme o item anterior esse valor é tão insignificante que não modifica o resultado dos testes.

Na figura 6.9 estão os gráficos referentes a cada um dos testes. O primeiro teste está com o sistema funcionando com a porcentagem da taxa de utilização de 37,69%, e o segundo com 60,05% e o terceiro 78,5%.

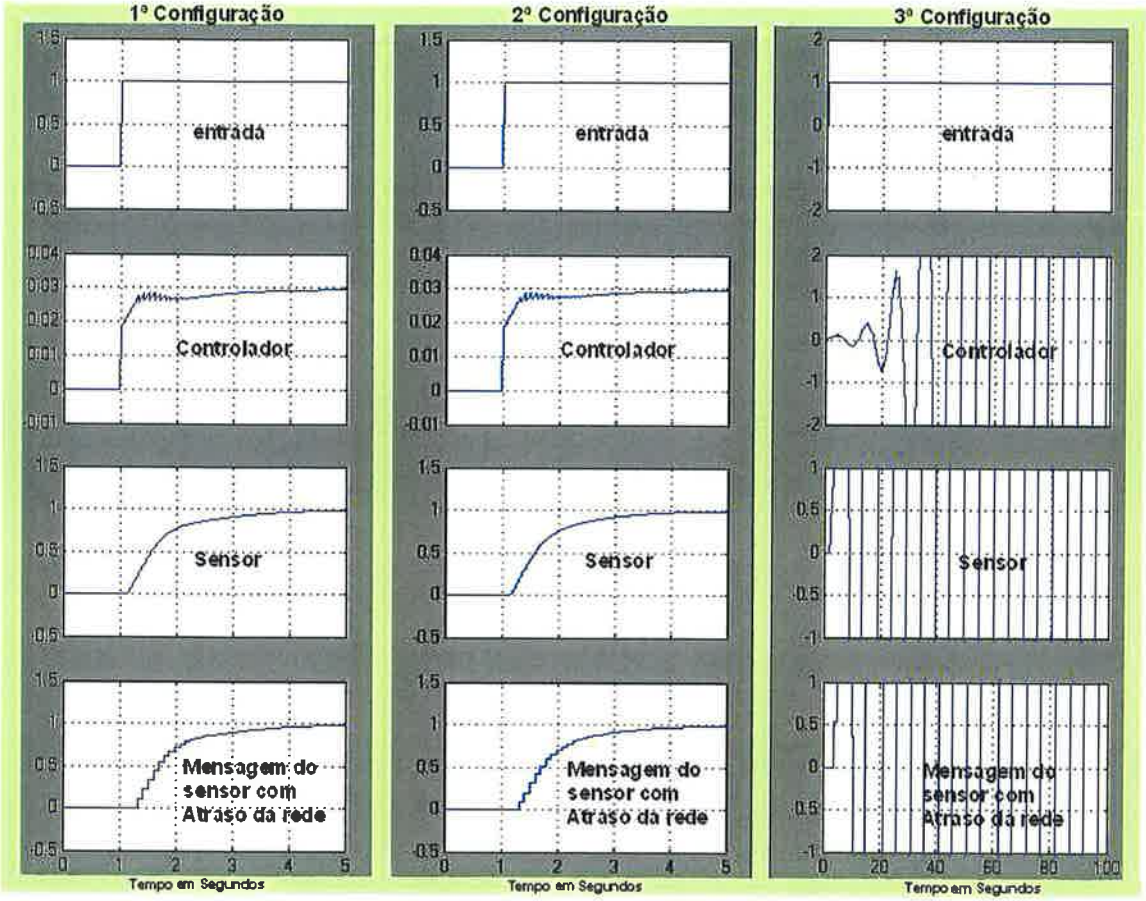

Figura 6.9 – Gráficos com os resultados da simulação

Observa-se que a estabilidade do sistema de controle está diretamente ligada ao tempo de atraso das mensagens. Quando as mensagens estão sem atrasos, o sistema de controle funciona perfeitamente, porém no último teste no qual o sistema de controle funcionou em uma rede com uma taxa de utilização muito alta, obteu-se um atraso maior nas mensagens de menor prioridade e como o sistema utilizou essas mensagens o controle ficou totalmente instável.

Atualmente, a maioria dos sistemas eletrônicos na indústria automobilística possui aplicativos que possibilitam atraso em na sua malha de controle, pois não estão ligados a sistemas de risco do veículo, mas com a introdução dos sistemas *x-by-wire* (sem *backup* mecânico) que substituem e atuam em itens do veículo como freio e direção, esse tipo de comportamento não

é permitido, por exemplo, se o sensor ou atuador de um sistema *steering-by-wire* estiver com um atraso com na configuração três deste estudo caso, e o veículo está na velocidade de 150 km/h, isso resultaria em uma grande instabilidade no seu funcionamento.

7 - CONCLUSÃO

O objetivo planejado para este trabalho foi de avaliar o comportamento temporal da rede CAN, em face da evolução dos sistemas eletrônicos que exigem um alto grau de confiabilidade dentro de um veículo.

Na revisão bibliográfica apresentam-se as características e as necessidades utilizadas para desenvolvimento do protocolo CAN. O protocolo CAN foi concebido para aperfeiçoar as aplicações de sistemas eletrônicos dentro de um veículo, trazendo alguns benefícios, pode-se citar:

- Redução de volume;
- Aumento de processamento;
- Compartilhamento de sensores e atuadores;
- Facilidade adicionamento de novos aplicativos no veículo (flexibilidade).

Atualmente esses benefícios não são mais as premissas de um protocolo de comunicação automotivo, o mais importante é conseguir prever condições estáveis para os aplicativos, garantindo um alto grau de previsibilidade.

Quando é analisado o comportamento temporal do CAN, verifica-se a flexibilidade da rede CAN, isso se torna um problema, pois quando um sistema é adicionado ou modificado dentro da rede, toda a rede pode ser afetada, possibilitando a modificação da referência temporal dos aplicativos conectados à rede e mostrando a falta de previsibilidade.

Dentro de todos os estudos de caso avaliam-se também varias modificações que podem ser realizadas nos projetos de rede CAN, para aumentar o desempenho do comportamento temporal e a estabilidade do sistema. Quando o CAN não atinge a confiabilidade que o aplicativo requer, outros tipos de protocolo podem ser estudados, como por exemplo, TTP/C [43], Byteflight [44] e FlexRay [45].

No Brasil há um risco maior para se obter problemas com a rede CAN, pois diferentemente dos países desenvolvidos, muitos aplicativos automotivos são instalados ou modificados nos veículos após a fabricação do mesmo, tais como: sistema de ar-condicionado, alarmes e re-potenciamento do motor, e esses sistemas na maioria dos casos não foram desenvolvidos e homologados juntamente com os fabricantes de veículos. Antigamente, esses aplicativos eram sistemas analógicos, ligados separadamente, o que diminuía a possibilidade se de interferir em outros sistemas instalados no veículo. Atualmente, porém, não existem saídas ou entradas analógicas para instalação dos sistemas e quase todos os sinais e informação estão disponíveis apenas na rede de comunicação.

REFERÊNCIAS

[1] Broy, M.; **Automotive Software and Systems Engineering.** Formal Methods and Models for Co-Design. In: MEMOCODE 05 - Third ACM and IEEE International conference, p.143-14, 2005.

[2] SANTOS, G.; GALVÃO, B.S.M.C.; ONUSIC, H.; SARTORE, C.A.F. **Comercial Vehicles - EMC Evaluations in South América.** In: 4th European Symposium on Electromagnetic Compatibility.Brugge, Belgium, p. 61-65, v. 2, 2000.

[3] BRENNAN, S.; BUCKLAND, J.; CHRISTEN, U.; **Especial Issue on Control Applications in Automotive Engineering** – IEEE Transactions on Control Systems Technology, v.15, n. 3, p. 403-405, 2007.

[4] MAHMUD, S. M.; ALLES, S.; **In vehicle Network Architectture for the Next Generation Vehicles.** In: SAE Word Congress in Michigan, paper number 2005-01-1531, 2005.

[5] Lian, F. L.; MOYNE, J. R.; TIBURY, D. M.; **Performance evaluation of control network: Ethernet Controlnet and Device Net.** In: IEEE Control System Magazine, p.66-83, 2001.

[6] TINDELL, K.; BURNS, A.; **Guaranteed Message Latencias for Distributed Safety Critical HARD Real-Time Networks**, Technical Report YCS 299, Department of Computer Science, University of York, England, 1994.

[7] TINDELL, K.; BURN, A.; WELLINGS, A.; **Calculating Controller Area Network Message Response Time**; In Control Engineering Practice, v. 3, n. 8, p. 1163-1169, 1995.

[8] UPENDER, B. P.; DEAN, A. G.; **Variability of CAN Network Performance.** In: IEEE International CAN Conference, Paris, France, p. 1151-1159, 1996.

[9] PUNNEKKAT, S.; HANSSON, H. A.; NORSTROM, C.; **Response Time Analysis under Errors for CAN.** In: IEEE Symposium of Real Time Technology and Applications, Washington, USA, p.258-266, 2000.

[10] JEON, J. G.; KIM, D. W.; KIM, H. S.; CHO, Y. J.; LEE, B. H.; **An Analysis of Network Based Control System Using CAN Protocol.** In: IEEE International Conference on Robotics e Automation, Seoul, Korea, v. 4, p. 3577-3581, 2001.

[11] SANTOS, M. M.; stemmer, m. r.; VASQUEZ, F.; **Schedulability Analysis of Messages in a CAN Network Applied to an Unmanned Airship.** In: Annual Conference of the IEEE Industrial Electronics Society, Serville, Spain, nov. 2002.

[12] BRILL, R. J.; LUKKIEN, J. J.; DAVIS, R. L.; BURNS, A.; **Message Response Time Analysis for Ideal Controller Area Network Refused.** In: International Workshop on Real-Time Network, Dresden, Germany, jul. 2006.

[13] DAVIS, R. I.; BURNS, A.; BRILL, R. J.; LUKKIEN, J. J.; **Controller Area Network Schedulability Analysis: Refuted, Revisited and Revised.** Technical Report from University of York, jan. 2007.

[14] POWERS, C.; KIRSON, A.; ACTON, D.; **Today's Eletronics in Today's Vehicles**. In: JURGEN, R. K.; Multiplexing and Networking: Automotive Electronics Series. SAE PT-78, ISBN 0-7680-0472-1, p. 245-250, 1999.

[15] MIESTERFELD, F.; **The Next Generation Vehicle Architecture**. In: JURGEN, R. K.; Multiplexing and Networking: Automotive Electronics Series. SAE PT-78, ISBN 0-7680-0472-1, p.645-652, 1999.

[16] HODEL, K.; SPECHT, S.; ONISIC, H.; **The On-board Electronics Innovations and Future Trends based on Customers Experiences.** Artigo SAE 2002-01-3376, 2002. In: SAE Brasil 2002, São Paulo, 2002.

[17] GUIMARÃES, A. A.; **O Protocolo CAN Bus nas Aplicações Off-Road: Uma Análise Comparativa entre os Padrões Existentes**. Artigo SAE 2001-01-3853, 2001. In: Congresso SAE BRASIL 2001, São Paulo, 2001.

[18] STRAUSS, C.; CUGNASCA, C. E.; SARAIVA, A. M. **Protocolos de Comunicação para Equipamentos Agrícolas**. In: CONAI, São Paulo, 1998. Anais, 1998.

[19] FREDRIKSSON, L.B.; **Controller Area Networks and the protocol CAN for machine control systems** - In: Mechatronics, v.4, n.2. p.159-192, 1994.

[20] ZUBERI, K.M.; SHIN, K.G.; **Real-time decentralized control with CAN**. In: IEEE Conference on emerging technologies and factory automation. IEEE 1996, p.93-99, 1996.

[21] HOFSTEE, J.W.; GOENSE, D.; **Simulation of a Controller Area Network - Implement Data Bus according to ISO 11783.** In: Journal of Argricultural Engineering Reserarch, v.73, p. 383-394, 1999.

[22] BRAGAZZA, B.D.; (Org.) Treinamento BOSCH Modulo 6 – **Controller Area Network (CAN)**. EAS31, Versão 3, 2000.

[23] CAN in Automation "CiA". **CAN History**. Disponível em **http://www.can-cia.de**. Acesso em: 06 jun. 2008.

[24] KIENCKE, U.; DAIS, S.; LITACHEL, M.; **Automotive Serial Controller Area Network.** SAE paper 860391, 1986, In: SAE International 1986, Detroit, Michigan, 1986.

[25] ISO 11898 - **Road Vehicle** - Interchange of Digital Information - Controller Area Network (CAN) for High-Speed Communication, 2003.

[26] ISO 11992 - **Road Vehicle** - Interchange of digital information on electrical connections between towing and towed vehicles, 2003.

[27] LIVANI, A.M.; KAISER, J.J.W.; **Scheduling hard and soft realtime communication in a controller area network.** In: Control Engineering Practice 7. jul. 1999.

[28] BOSCH. **CAN Specification Version 2.0. Robert Bosch GmbH** - Stuttgart, 1991.

[29] RUFINO, J.; **An Overview of the Controller Area Network**. Braga, jan. 1997.

[30] BOTERENBROOD, H.; **CANopen specification, high-level for CAN-bus**. Nikhef, Amsterdam, mar. 2000.

[31] HARTWICH, F.; BASSEMIR, A.; **The Configuration of the CAN Bit Timing.** In: 6th International CAN Conference. Turin, nov. 1999.

[32] KOPETZ, H.; **Real-Time Systems Design for Distributed Embedded Applications.** Kluwer Academic Publishers, 1997.

[33] BURNS, A.; WELLINGS, A.J.; **Real-time systems and programming languages.** Ada 95, real-time Java, and real-time POSIX. Addison-Wesley, Reading, MA, USA, third edition, 2001.

[34] PUSCHNER, P.; BURNS, A.; **A Review of Worst-Case Execution-Time Analysis.** Real Time Systems, p.115–128, 2000.

[35] BURNS, A.; **Real Time Systems**. In: Encyclopedia of Physical Science and Technology, v. 14, p. 45–54. Academic Press, 2002.

[36] LIU, C. L.; LAYLAND, J. W.; **Scheduling Algorithms for Multiprogram in a Hard Real-Time Environment.** In: Journal of ACM, p 40–61, 1973.

[37] AUDSLEY, N. C.; BURN, A.; RICHARDSON, TINDELL, K.; WELLINGS, A. J.; **Applying New Scheduling Theory to Static Priority Pre-Emptive Scheduling.** In: Software Engineering Journal, p. 284–292, 1993.

[38] QUIGLEY, C. P.; MCMURRN, R.; JONES, R. P.; FAITHFULL,P. T.; **An Invetigation into Cost Modeling for Design of Distributed Automotive Electrical Architectures -** Automotive Electronics. In: 3rd Institution of Engineering and Technology Conference, p. 1-9, jun. 2007.

[39] YIH, P.; GERDES, C.; **Modification of Vehicle Handling Characteristics via Steer-by Wire** – In: IEEE Transactions on control systems technology, v. 13, n. 6, p. 965-976, nov. 2005.

[40] XIANG, W.; RICHARDSON, P. C.; ZHAO, C.; MOHAMMAD, S.; **Automobile Brake-by-Wire Control System Design and Analysis** – In: IEEE Transactions on vehicular technology, v. 57, n.1, jan. 2008.

[41] ISERMAN, R.; SCHWARZ, R.; STÖLZL, S.; **Fault-tolerant Drive-by-Wire Systems** – In: IEEE Control Systems Magazine, v. 22, n. 5, p. 64– 81, oct. 2002.

[42] Mei, T. X.; Shafik, M.; Lewis, R.; Walilay, H.; Whitley, M.; Baker, D.; **Fault Tolerant Actuation for Steer-by-Wire Applications -** Automotive Electronics. In: 3rd Institution of Engineering and Technology Conference, jun. 2007.

[43] TTTech - **TTP/C High-Level Specification Document** - Protocol Version 1.1. Document number D-032-S-10-028, nov. 2003.

[44] CENA, G.; VALENZANO, A.; **Performance Analysis of Byteflight Networks - Factory Communication Systems** – In: IEEE International Workshop – p. 157-166, sep. 2004.

[45] FlexRay Consortium - **FlexRay Communications System Protocol Specification v2.1 - Revision A** – mar. 2006

[46] CERVIN, A.; HENRIKSSON, D.; LINCOLN, B.; JOHAN, E.; ARZÉN, K. E.; **How Does Control Timing Affect Performance?** – In: IEEE Control Systems Magazine - 0272-1708/03, jun. 2003.

[47] SANTOS, M. .M. D.; VASQUES, F.; STEMMER, M. R.; **Avaliação das Propriedades Temporais de duas Redes de Controle: CAN e PROFIBUS** – In: Acta Scientiarum. Technology, Maringá, v. 25, n. 2, p. 193-201, 2003.

[48] TIPSUWAN, Y.; CHOW, M.; VANIJJIRATTKHN, R.; **An Implementation of a Networked PI Controller over IP Network.** In: Industrial Electronics Society, IECON - The 29th Annual Conference of the IEEE - v. 3, p. 2805- 2810, nov. 2003.

[49] CHOW, M.; TIPSUWAN, Y.; **Gain Adaptation of Networked DC Motor Controllers** – In: IEEE Transactions on Industrial Electronics, v. 50, n. 5, October 2003.