ANÁLISE DOS TEMPOS DE RESPOSTA DE UM SISTEMA DE AQUISIÇÃO DE DADOS DISTRIBUÍDO BASEADO NA REDE CAN

Jadsonlee da Silva Sá*, Antonio Marcus Nogueira Lima*, José Sérgio da Rocha Neto*

* *Universidade Federal de Campina Grande*
Departamento de Engenharia Elétrica
Laboratório de Instrumentação Eletrônica e Controle
Av. Aprígio Veloso, 882 - Bodocongó - 58109-970
Campina Grande – PB – Brasil

Emails: jadsonlee, amnlima, zesergio@dee.ufcg.edu.br

Abstract— This paper presentes the response time analysis messages of a distributed data acquisition system based on CAN network. The data acquisition system was projected to automatically acquire the physical properties of a thermoelectrical generation system. The analysis compute the worst-case response times of CAN messages to verify if the timing requirement of the system will be satisfied.

Keywords— Distributed Data Acquisition Systems, CAN Network, Real-Time Scheduling, Worst-Case Response Time Analysis.

Resumo— Neste trabalho apresenta-se a análise dos tempos de resposta das mensagens de um sistema de aquisição de dados distribuído baseado na rede CAN. O sistema de aquisição de dados foi projetado para adquirir automaticamente as grandezas físicas de um sistema de geração termoelétrica. A análise determina os tempos de resposta no pior caso das mensagens CAN para verificar se os requisitos de tempo do sistema serão satisfeitos.

Keywords— Sistemas de Aquisição de Dados Distribuído, Rede CAN, Escalonamento em Tempo Real, Análise dos Tempos de Resposta no Pior Caso.

1 Introdução

Redes de comunicações tornaram-se populares em aplicações que envolvem a aquisição automática das grandezas físicas provenientes de processos. Sistemas com esta finalidade são normalmente chamados de Sistemas de Aquisição de Dados Distribuído (SADDs)(Sá, 2007).

A utilização de arquiteturas em rede ao invés das tradicionais arquiteturas centralizadas deve-se a contínua diminuição no preço dos componentes de *hardware* e principalmente, ao aumento da complexidade das aplicações, as quais dispõem de um maior número de pontos de medição dispersos geograficamente. Além disto, existem vantagens tecnológicas quando comparadas aos sistemas com arquiteturas centralizadas, tais como: redução da quantidade de cabos; maior capacidade de processamento; facilidade na instalação, diagnóstico e manutenção; flexibilidade; modularidade e confiabilidade (Broster, 2003).

Atualmente, existem no mercado diversas tecnologias de rede. Dentro do contexto de redes com fio projetadas para aplicações de aquisição de dados e controle de processos, destacam-se as redes ASI, CAN, PROFIBUS-DP, DeviceNet, INTERBUS-S, WorldFip entre outras. Estas tecnologias foram desenvolvidas sob a premissa de garantir uma comunicação em tempo real confiável e robusta para aplicações específicas. No entanto, algumas delas possuem as características requisitadas em outras áreas de aplicação, como por exemplo, a rede CAN, a qual foi desenvolvida para indústria automotiva e atualmente é utilizada em várias áreas de aplicação. Em geral, estas redes diferem quanto a taxas de transmissão, comprimento das mensagens, controle de acesso ao meio, quantidade máxima de nós e mecanismos para detecção e correção de erros. Devido a alta qualidade destas tecnologias, um fator crucial na escolha está relacionado principalmente ao custo e disponibilidade de *hardware* e ferramentas para projeto e análise. CAN é uma das redes que apresenta concordância nestes critérios de escolha.

A arquitetura básica de um SADD é constituída por um ou vários nós sensores distribuídos no processo e um nó supervisor (computador), os quais possuem capacidade de processamento e trocam informações (mensagens) via um meio de transmissão compartilhado conforme um protocolo de comunicações em rede. Basicamente, os nós sensores adquirem periodicamente as informações provenientes dos seus respectivos sensores e transmitem para o nó supervisor, o qual processa e armazena estas informações.

Para determinar corretamente o comportamento do processo em análise, deve-se garantir que as mensagens transmitidas pelos nós sensores serão recebidas corretamente pelo nó supervisor de acordo com os requisitos de tempo da aplicação. No entanto, o fato de um SADD utilizar uma rede de comunicações com largura de banda limitada e meio de comunicação compartilhado, impõe limitações no projeto do sistema, de forma que é necessário realizar uma análise de escalo-

[1] Jadsonlee da Silva Sá é Bolsista CAPES - Brasil.

namento das mensagens para garantir o cumprimento destes requisitos.

Neste trabalho, apresenta-se a análise de escalonamento das mensagens de um SADD utilizando a rede de comunicações CAN. Este sistema foi desenvolvido para auxiliar nas atividades de pesquisa realizadas no Laboratório de Geração Termoelétrica instalado nas dependências do Departamento de Engenharia Elétrica na Universidade Federal de Campina Grande (Sá, 2007). Neste laboratório são desenvolvidas pesquisas com o objetivo de otimizar e controlar o uso de Gás Natural Veicular (GNV) em motores do ciclo diesel para geração de eletricidade. O principal componente deste laboratório é um sistema eletromecânico constituído por um motor do ciclo diesel de 250 HP acoplado a um gerador elétrico com capacidade de 150 KW.

Na seção 2, descreve-se a arquitetura e funcionamento do SADD. Na seção 3, apresentam-se as principais características de uma rede CAN. Na seção 4, descreve-se o modelo e a análise de escalonamento das mensagens CAN. Na seção 5, apresentam-se os resultados obtidos e na seção 6 as conclusões.

2 Arquitetura e Descrição do Sistema de Aquisição de Dados Distribuído

O SADD é composto por oito nós sensores (NS) e um nó supervisor conectados via um barramento CAN (dois fios) com comprimento de 25 m. O esquema da arquitetura do SADD está representado na Figura 1.

Cada nó sensor é constituído pelos seguintes componentes: sensores; circuitos para condicionamento; microcontrolador ADuC842 (Device, 2003); controlador CAN MCP2515 (Microchip, 2005); *transceiver* CAN MCP2551 (Microchip, 2003); e um *firmware* embarcado no microcontrolador para controle e gerenciamento do nó, desenvolvido em linguagem C. A comunicação entre o ADuC842 e o MCP2515 é realizada por meio da *interface* SPI (*Serial Peripheral Interface*) a uma taxa de transmissão de 8,33 M*bits*/s. O ADuC842 está configurado como mestre e o MCP2515 como escravo. O nó supervisor é constituído por um dispositivo CAN *Leaf Professional* HS (Kvaser, 2006) com *interface* USB conectado a um computador com sistema operacional Linux Ubuntu, e um *software* de aplicação desenvolvido em linguagem C.

Os nós sensores foram distribuídos no laboratório de acordo com a localização dos pontos de medição (sensores). O agrupamento dos sensores em cada nó foi baseado na localização dos sensores e nos respectivos requisitos de tempo das grandezas medidas. Por exemplo, no nó NS06 (Fig. 1), os pontos de medição de temperatura e pressão estão próximos fisicamente e possuem o mesmo requisito de tempo na amostragem.

Cada nó sensor executa uma tarefa periódica controlada pelo *firmware*, com instante de ativação igual ao período de amostragem das respectivas grandezas medidas. Todas as grandezas associadas a um nó possuem o mesmo período de amostragem. As instâncias da tarefa são ativadas por meio de uma interrupção de *hardware* gerada por um relógio no microcontrolador de cada nó. Esta tarefa realiza as seguintes atividades seqüencialmente: conversão analógica digital dos sinais provenientes dos sensores (resolução de 12 *bits*); envio destas informações para o *buffer* de transmissão do controlador CAN; e requisição de transmissão da mensagem. Cada nó transmite as informações de todas as suas grandezas em uma única mensagem CAN.

O nó supervisor executa uma tarefa esporádica controlada pelo *software* de aplicação, ativada pela chegada de alguma mensagem no *buffer* de recepção do dispositivo CAN. Esta tarefa faz a retirada das mensagens do *buffer* de recepção, processa e armazena as informações em um arquivo de texto.

3 Rede CAN

CAN (*Controller Area Network*) é uma rede de comunicações multi-mestre assíncrona que conecta nós via um barramento serial (Etschberger, 2001).

Os nós trocam informações em forma de mensagens (seqüência de *bits*) a uma taxa de transmissão de até 1 M*bits*/s. Os *bits* das mensagens são representados fisicamente pela codificação NRZ (*Non-Return to Zero*). Nesta codificação, o nível do sinal permanece constante durante o tempo de transmissão de um *bit*, chamado de tempo de *bit*. O nível lógico '0' é dominante, enquanto que o nível lógico '1' é recessivo. Para evitar a ausência de bordas de transição na seqüência de *bits* de uma mensagem, as quais são utilizadas no processo de sincronização durante a recepção das mensagens, CAN utiliza um mecanismo de inserção de *bits* que insere automaticamente após uma seqüência de cinco *bits* com mesma polaridade, um *bit* com polaridade oposta. Após a recepção da mensagem, os *bits* inseridos são retirados automaticamente. Além disto, este mecanismo é utilizado no processo de sinalização de erros nas mensagens.

Existem quatro diferentes tipos de mensagens: mensagem de dados, erro, requisição de dados e sobrecarga. Apenas as duas primeiras são usadas na maioria das aplicações. A mensagem de dados é usada na transmissão de dados de um nó para os outros, e a mensagem de erro é usada para indicar a existência de um erro de transmissão ou recepção em qualquer tipo de mensagem. Esta mensagem é transmitida automaticamente pelo nó que detectou o erro.

Uma mensagem de dados (Figura 2) é formada por cinco campos (arbitração, controle, da-

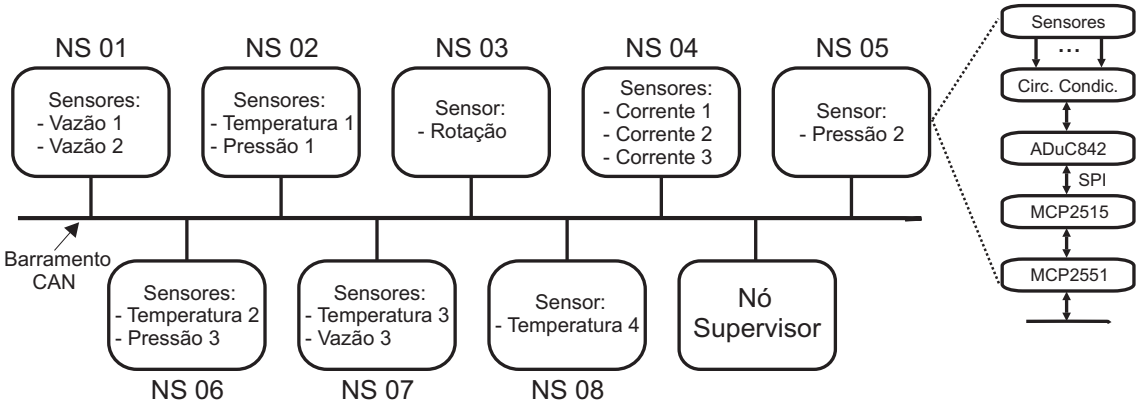

Figura 1: Arquitetura do SADD.

dos, CRC - *Cyclic Redundancy Code* e ACK - *Acknowledgement*) delimitados por um *bit* dominante (SOF - *Start Of Frame*) no início da mensagem e por sete *bits* recessivos (EOF - *End Of Frame*) no fim da mensagem. Os campos de SOF até CRC são submetidos ao mecanismo de inserção de *bits*. As informações são transmitidas no campo de dados e podem variar entre 0 e 8 *bytes*. A quantidade de *bytes* de dados é especificada no campo de controle por valores inteiros variando de 0 a 8. Cada mensagem possui um identificador único localizado no campo de arbitração, que pode ter 11 *bits* no formato padrão ou 29 *bits* no formato estendido. Apenas mensagens com formato padrão serão consideradas. O identificador tem duas funções: caracterizar o conteúdo da mensagem de forma que elas sejam filtradas/identificadas no nó receptor; e determinar o nível de prioridade quanto o acesso ao barramento.

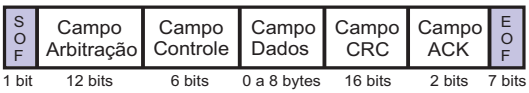

Figura 2: Esquema de uma mensagem de dados.

Mensagens de dados são separadas de mensagens precedentes por no mínimo três tempos de *bit*. Este espaço de tempo é chamado de intervalo entre mensagens. Os três *bits* são sempre recessivos.

CAN é uma rede *event-triggered*, ou seja, os nós acessam o barramento de forma aleatória. Portanto, é possível que mais de um nó tente transmitir uma mensagem simultaneamente. Para controlar as possíveis colisões durante o acesso ao barramento, CAN utiliza o mecanismo CSMA/CR (*Carrier-Sense Multiple Access/Collision Resolution*). Neste mecanismo, os nós transmitem mensagens (dados ou requisição de dados) somente se o barramento estiver livre. O barramento é considerado livre após o intervalo entre mensagens. Neste estado, o nível lógico lido pelos nós é sempre recessivo. Assim que o barramento estiver livre, todos os nós com mensagens prontas para serem transmitidas enviam o *bit* dominante SOF e entram em processo de arbitração. O resultado da arbitração é determinada pela comparação dos *bits* dos identificadores (campo de arbitração) das mensagens baseado no mecanismo *wired-AND*. Isto significa que a mensagem que tiver o identificador com menor valor numérico (maior prioridade) vencerá o processo de arbitração e transmitirá a mensagem completamente. Os nós que perderam a disputa e outros possíveis nós tentarão transmitir suas mensagens assim que o barramento estiver livre novamente.

O tempo gasto no pior caso para um nó transmitir fisicamente uma mensagem m, denotado por C_m, é derivado pela Eq. 1. C_m é determinado em função do comprimento da mensagem (em *bits*) e do tempo de transmissão de um *bit* (τ_b). O comprimento da mensagem depende do número de *bytes* de dados (b_m) e da quantidade de *bits* inseridos pelo mecanismo de inserção de *bits*. O operador $\lfloor x/y \rfloor$ retorna o maior inteiro menor que o valor x/y. Este termo equivale aos *bits* inseridos pelo mecanismo de inserção de *bits* considerando o pior caso. Os três tempos de *bit* equivalentes ao intervalo entre mensagens estão inclusos na Eq. 1.

$$C_m = \left(47 + (8b_m) + \left\lfloor \frac{34 + (8b_m) - 1}{4} \right\rfloor\right) \tau_b \quad (1)$$

CAN foi projetado para garantir uma comunicação confiável e robusta. Para garantir a integridade das informações transmitidas nas mensagens, o protocolo implementa cinco mecanismos para detecção de erro. A detecção de um erro na mensagem é automaticamente sinalizada por algum nó da rede e a mensagem será re-transmitida conforme o processo de arbitração. O processo de sinalização de erro resulta em um tempo extra ocasionado pelo envio da mensagem de erro. O comprimento da mensagem de erro pode variar entre 17 e 31 tempos de *bit*.

4 Análise de Escalonamento das Mensagens CAN

A análise de escalonamento das mensagens em uma rede CAN foi desenvolvida recentemente por Davis et al. (2007) baseada nos trabalhos de Tindell et al. (1994) e George et al. (1996). Esta análise fornece um método para calcular os tempos de resposta no pior caso das mensagens CAN. Deste modo, é possível verificar se todas as mensagens do sistema cumprirão seus requisitos de tempo. Esta análise foi utilizada como parte do projeto de desenvolvimento do SADD.

4.1 Notação e Modelo de Escalonamento

Assume-se que o controlador de cada nó tem a capacidade de garantir que a sua mensagem com maior prioridade armazenada no *buffer* de transmissão, entrará no processo de arbitração assim que o barramento estiver livre.

Uma mensagem m possui uma prioridade fixa (p_m), um comprimento fixo de *bytes* de dados (b_m) e um tempo máximo de transmissão (C_m). As variáveis m e p_m assumem valores de 1 à n, onde n é igual a quantidade de mensagens do sistema.

Toda mensagem é armazenada no *buffer* de transmissão do seu respectivo controlador por uma tarefa periódica liberada a cada período T_m. Esta tarefa gasta uma quantidade de tempo limitada, entre 0 e J_m, para armazenar a mensagem no *buffer* de transmissão, onde J_m é chamado de *jitter* de armazenamento.

Cada mensagem possui um requisito de tempo chamado de *deadline* (D_m). Deve-se garantir que uma instância da mensagem m será transmitida antes que a próxima seja armazenada no *buffer* de transmissão. Caso contrário, a mensagem anterior será sobrescrita e perdida. Portanto, esta restrição impõe que $D_m \leq T_m + J_m$.

4.2 Análise dos Tempos de Resposta das Mensagens

A análise dos tempos de resposta fornece um método para calcular o tempo de resposta no pior caso das mensagens do sistema.

O tempo de resposta no pior caso de uma mensagem m, denotado por R_m, corresponde ao maior tempo gasto desde a ativação da tarefa até o instante em que o último *bit* da mensagem chega ao *buffer* de recepção do nó destino. Uma mensagem é escalonável, se e somente se, $R_m \leq D_m$. Portanto, o sistema é escalonável, se e somente se, todas as mensagens são escalonáveis. O valor de R_m é determinado pela Equação 2.

$$R_m = J_m + W_m + C_m \tag{2}$$

Conforme descrito anteriormente, J_m corresponde ao *jitter* de armazenamento, e o seu valor pode ser adquirido experimentalmente para cada mensagem. C_m (Equação 1) corresponde ao tempo máximo gasto para transmitir fisicamente uma mensagem m no barramento. W_m chamado de atraso de enfileiramento, corresponde ao tempo máximo que a mensagem m poderá esperar no *buffer* de transmissão do controlador antes de iniciar a transmissão, devido outras mensagens estarem ocupando o barramento.

O atraso de enfileiramento consiste de dois tipos de atraso: atraso de bloqueio, originado por uma mensagem com menor prioridade, que poderá estar no processo de transmissão quando a mensagem m foi armazenada no *buffer* de transmissão do controlador; e o atraso de interferência, originado pelas mensagens com maior prioridade, as quais poderão ser transmitidas antes da mensagem m.

O máximo atraso de bloqueio ocorre quando a maior mensagem com menor prioridade inicia a transmissão imediatamente antes da mensagem m ser armazenada no *buffer* de transmissão. A mensagem m deverá esperar até que o barramento esteja livre, antes de entrar no processo de arbitração. O máximo atraso de bloqueio (B_m) é determinado pela Equação 3, onde $lp(m)$ corresponde ao conjunto de mensagens com menor prioridade que a mensagem m.

$$B_m = \max_{k \in lp(m)} (C_k) \tag{3}$$

A derivação do atraso de enfileiramento de uma mensagem m é baseada na definição de período ocupado com nível de prioridade - p_m (Lehoczky, 1990)(Harbour et al., 1991). Este período é definido como um intervalo semi-aberto à direita, que inicia em algum instante t^i quando uma mensagem com prioridade p_m ou maior é enfileirada no *buffer* de transmissão do seu respectivo controlador CAN, e não existem mensagens com prioridade p_m ou maior esperando para serem transmitidas, as quais foram enfileiradas antes do instante t^i. O período é finalizado no instante t^f quando o barramento torna-se livre e não existem mensagens com prioridade p_m ou maior esperando para serem transmitidas, as quais foram enfileiradas antes do instante t^f. Durante este intervalo nenhuma mensagem com prioridade menor que p_m é capaz de vencer a arbitração e ser transmitida. O atraso de enfileiramento no pior caso para uma mensagem m com prioridade p_m, ocorre para alguma instância desta mensagem enfileirada durante o período ocupado que inicia imediatamente após o início da transmissão da maior mensagem com menor prioridade que p_m. O maior período ocupado com nível de prioridade - p_m inicia no chamado instante crítico (Liu and Layland, 1973), instante onde a mensagem m é armazenada simultaneamente com todas as men-

sagens com maior prioridade, em seus respectivos *buffers* de transmissão.

O comprimento t_m do maior período ocupado com nível de prioridade - p_m é determinado pela seguinte relação recursiva (4), onde $hep(m)$ é o conjunto de mensagens com prioridade maior ou igual à mensagem m. O operador $\lceil x/y \rceil$ retorna o menor inteiro maior ou igual a x/y.

$$t_m^{n+1} = B_m + \sum_{\forall k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k \qquad (4)$$

O procedimento de iteração começa com um valor inicial $t_m^0 = C_m$, e é finalizado quando $t_m^{n+1} = t_m^n$. Devido o lado direito da Eq. 4 ser uma função monotônica não-decrescente de t_m, a relação de recorrência convergirá se o fator de utilização U_m para mensagens com prioridade maior ou igual a m for menor ou igual a 1 (Eq. 5).

$$U_m = \sum_{\forall k \in hep(m)} \left(\frac{C_k}{T_k} \right) \leq 1 \qquad (5)$$

Para determinar o maior atraso de enfileiramento de uma mensagem m é necessário calcular e verificar qual é o maior atraso de todas as suas instâncias enfileiradas e transmitidas durante o maior período ocupado com nível de prioridade - p_m. O número de instâncias Q_m é derivado de (6).

$$Q_m = \left\lceil \frac{t_m + J_m}{T_m} \right\rceil \qquad (6)$$

Na análise seguinte, a variável q será usada para representar uma instância da mensagem m. A primeira instância do período ocupado corresponde a $q = 0$, e a última instância, $q = Q_m - 1$. O atraso de enfileiramento da instância q da mensagem m é determinado pela Eq. 7, onde $hp(m)$ corresponde ao conjunto de mensagens com prioridade maior que a mensagem m e a variável τ_b (tempo de *bit*) representa as diferenças de tempo para o início da arbitração nos diferentes nós, devido aos atrasos de propagação e tolerâncias do protocolo.

$$W_m^{n+1}(q) = B_m + qC_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{W_m^n(q) + J_k + \tau_b}{T_k} \right\rceil C_k \qquad (7)$$

A relação de recorrência começa com um valor de $W_m^0(q) = B_m + qC_m$, e termina quando $W_m^{n+1}(q) = W_m^n(q)$ ou $J_m + W_m^{n+1}(q) - qT_m + C_m > D_m$, onde no último caso, a mensagem é não-escalonável. A relação de recorrência sempre convergirá se a condição do fator de utilização do sistema $U = \sum_{k=1}^{n} \left(\frac{C_k}{T_k} \right) \leq 1$ for satisfeita (Spuri, 1996). Onde n equivale ao número de mensagens do sistema.

A instância q da mensagem m ocorre no instante $qT_m - J_m$ relativo ao início do período ocupado. Então, o tempo de resposta da instância q da mensagem m é determinado na Eq. 8.

$$R_m(q) = J_m + W_m(q) - qT_m + C_m \qquad (8)$$

Observe que o termo W_m em (2) equivale ao termo $W_m(q) - qT_m$ em (8). Finalmente, o tempo de resposta no pior caso da mensagem m é derivado da Eq. 9.

$$R_m = \max_{q=0 \ldots Q_m - 1} (R_m(q)) \qquad (9)$$

5 Resultados Obtidos

Nesta seção apresenta-se a análise de escalonamento das mensagens do SADD apresentado na seção 2, utilizando várias taxas de transmissão. A política de atribuição de prioridade utilizada foi a *deadline* monotônico (Leung and Whitehead, 1982). Foi desenvolvido um programa em linguagem C para resolver as equações apresentadas na seção anterior, as quais determinam os tempos de resposta no pior caso das mensagens do sistema.

As informações das mensagens dos nós sensores e os tempos de resposta de cada mensagem para diferentes taxas de transmissão estão indicadas na Tabela 1. Cada nó transmite uma única mensagem m. As mensagens estão organizadas na tabela em ordem decrescente de prioridade. A mensagem enviada pelo nó 04 ($m = 1$) possui a maior prioridade, enquanto que a mensagem enviada pelo nó 08 ($m = 8$) possui a menor prioridade. Os períodos de amostragem das grandezas físicas e conseqüentemente das mensagens foram especificados de acordo com os requisitos da aplicação. Para as taxas de transmissão de 1 M$bits$/s e 500 K$bits$/s o sistema é escalonável, visto que $R_m < D_m$ para todas as mensagens do sistema. Para estas taxas de transmissão, o fator de utilização $U = 23,08\%$ e $46,12\%$, respectivamente. Na taxa de transmissão de 417 K$bits$/s, $U = 55,34\%$. Nesta taxa, a mensagem ($m = 1$) do nó 4 tem um $R_m = 554\ \mu s$, que é maior do que $D_m = 550\ \mu s$. Portanto o sistema é não-escalonável. Devido o tempo de resposta ser diretamente proporcional a taxa de transmissão do barramento, o sistema será não-escalonável para todas as taxas abaixo de 417 K$bits$/s. Para taxas de transmissão abaixo de 227 K$bits$/s, $U > 100\%$.

6 Conclusões

Neste trabalho foi realizada a análise de escalonamento das mensagens de uma rede CAN aplicada a um sistema de aquisição de dados distribuído.

Tabela 1: Informações da análise de escalonamento das mensagens.

Nó	m	b_m ($bytes$)	J_m (μs)	T_m (ms)	D_m (ms)	R_m (μs) 1 M$bits$/s	R_m (μs) 500 K$bits$/s	R_m (μs) 417 K$bits$/s
4	1	6	50	0,5	550 μs	260	470	**554**
1	2	4	36	1000	$\approx$ 1000	341	646	1044
2	3	4	36	1000	$\approx$ 1000	436	1066	1548
6	4	4	36	1000	$\approx$ 1000	531	1486	2052
7	5	4	36	1000	$\approx$ 1000	721	1636	2508
3	6	2	22	1000	$\approx$ 1000	782	2002	2950
5	7	2	22	1000	$\approx$ 1000	857	2382	3130
8	8	2	22	1000	$\approx$ 1000	857	2382	3130

Por meio desta análise foi possível determinar em quais taxas de transmissão os requisitos de tempo das mensagens serão satisfeitos. Verificou-se que o conjunto de mensagens do sistema é escalonável somente quando a taxa de transmissão é maior do que 417 K$bits$/s.

A análise considerou apenas o comportamento ideal da rede, visto que não foi considerado a presença de erros de transmissão nas mensagens, as quais aumentam os tempos de resposta das mensagens, devido ao processo de sinalização e recuperação do erro.

Pretende-se em trabalhos futuros: estender a análise sob condições reais a partir de um modelo matemático capaz de descrever o comportamento de erros na rede para o ambiente do laboratório; verificar analiticamente que todas as mensagens serão processadas pelo *software* de aplicação antes de ocorrer um estouro de capacidade na fila de recepção do nó supervisor; e utilizar o sistema em outras aplicações, por exemplo, em sistemas de controle em rede.

Agradecimentos

Os autores agradecem o apoio financeiro fornecido pela GEBRA/BRASYMPE, CAPES e CNPq durante a realização deste projeto.

Referências

Broster, I. (2003). *Flexibility in Dependable Real-Time Communication*, PhD thesis, University of York, England.

Davis, R. I., Burns, A., Bril, R. J. and Lukkien, J. J. (2007). Controller area network schedulability analysis: Refuted, revisited and revised, *Real-Time Systems* **35**(3): 239–272.

Device, A. (2003). *ADuC842, microconverter, 12-bit ADCs with embedded 62 kB flash MCU*, Analog Device, USA.

Etschberger, K. (2001). *Controller Area Network: Basics, Protocols, Chips and Application*, first edn, IXXAT Press, Weingarten, Germany.

George, L., Rivierre, N. and Spuri, M. (1996). Preemptive and non-preemptive real-time uniprocessor scheduling, *Technical Report RR-2966*, INRIA.

Harbour, M. G., Klein, M. H. and Lehoczky, J. P. (1991). Fixed priority scheduling of periodic task with varying execution priority, *Proceedings 12th IEEE Real-Time Systems Symposium - IEEE Computer Society Press* pp. 116–128.

Kvaser (2006). *Kvaser Leaf User Guide*, Kvaser, Sweden.

Lehoczky, J. P. (1990). Fixed priority scheduling of periodic task sets with arbitrary deadlines, *Proceedings 11th IEEE Real-Time Systems Symposium - IEEE Computer Society Press* pp. 201–209.

Leung, J.-T. and Whitehead, J. (1982). On the complexity of fixed-priority scheduling of periodic, real-time tasks, *Performance Evaluation* **2**(4): 237–250.

Liu, C. L. and Layland, J. W. (1973). Scheduling algorithms for multiprogramming in a hard real-time environment, *Journal of The ACM* **20**(1): 46–61.

Microchip (2003). *High-Speed CAN Transceiver - MCP2551*, Microchip, USA.

Microchip (2005). *Stand-Alone CAN Controller with SPI - MCP2515*, Microchip, USA.

Sá, J. S. (2007). *Projeto e análise de um sistema de aquisição de dados distribuído para aplicações em tempo real*, Master's thesis, Universidade Federal de Campina Grande - UFCG, Brasil.

Spuri, M. (1996). Analysis of deadline schedule real-time systems, *Technical Report RR-2772*, INRIA.

Tindell, K. W., Burns, A. and Wellings, A. J. (1994). An extendible approach for analysing fixed priority hard real-time systems, *Journal of Real-Time Systems* **6**(2): 133–152.