Protocolo Ponto-a-Ponto sem-fio usando kernel de tempo real

Resumo Expandido - Disciplina de TCC290009

Bruno Antônio de Pinho
Estudante do Curso de Engenharia de Telecomunicações
Macelo Maia Sobral
Professor Orientador

Semestre 2019-1

Resumo- *Redes sem fio são extremamente dependentes do tempo de resposta a já que quando o tempo limite de resposta de um pacote é ultrapassado o mesmo é descartado. Sistemas operacionais em tempo real são especializados no tempo de resposta sendo mais preciso que sistemas operacionais de propósito geral. Este trabalho tem como objetivo a implementação de um device driver para protocolo um protocolo ponto-a-ponto para redes sem fio em um sistema operacional em tempo real a fim de utilizar suas características em tempo de resposta de modo a se obter um melhor desempenho do protocolo.*

Palavras-chave: RTOS, Xenomai, portocolo ponto-a-ponto, Tempo Real, Device Driver.

1 Introdução

Sistemas operacionais tradicionais dão impressão de ser multitarefas dividindo o seu tempo de processamento em fatias de tempo, cada processo tem uma ou mais fatias de tempo alocadas conforme sua prioridade, após esse tempo acabar passa-se para o próximo processo dando a impressão que múltiplas aplicações estão rodando simultaneamente (MAZIERO, 2019). Esta abordagem tem algumas desvantagens no entanto como, por exemplo uma aplicação com uma prioridade alta e com um longo tempo de processamento pode fazer com que aplicações menos prioritárias fiquem longos períodos sem ser processadas.

Sistema operacionais em tempo real do inglês *Real Time Operational Systems (RTOS)* é um sistema operacional mais especializado onde o tempo de resposta é mais importante do que executar centenas de tarefas simultaneamente. O tempo de resposta não precisa necessariamente ser o mais rápido possível, como o nome pode sugerir, mas deve ser previsível, logo, *Real-Time* pode ser uma resposta de vários minutos ou nano segundos, dependendo da forma que seu sistema funciona. As tarefas do *RTOS* contam com tempo limite para serem executadas, então é comumente chamado de *Time Critical* (MORAIS, 2018).

O Xenomai é um *RTOS* baseado em Linux, possuindo um *kernel* Linux tradicional e um *kernel* para aplicações em tempo real chamado Cobalt, o qual trata aplicações que precisam de um curto tempo de resposta de maneira independente do *kernel* Linux. (XENOMAI, 2019).

Este trabalho tem como objetivo a implementação de um protocolo de transmissão ponto-a-ponto sem fio utilizando um *RTOS* (Xenomai), tendo-se em vista as características desse sistema espera-se que as chamadas do protocolo sejam atendidas com um menor atraso, causado pelo processamento de outras aplicações, e consequentemente mais rapidamente fazendo com que se tenha um ganho no desempenho. Como objetivos específicos tem-se:

- Estudo sobre o desenvolvimento de *drivers* em *RTOS*.

- Planejamento e execução de testes para avaliação do desempenho da implementação em *RTOS*.
- Obter-se um desempenho melhor do que a implementação em sistema operacional Linux.

2 Metodologia

2.1 *Application Programming Interface (API's)*

O Xenomai possui diversas *API's* diferentes que podem ser usadas para o desenvolvimento de software podendo ser usado tanto no *kernel Linux* quanto no *kernel Cobalt*, cada *API's* possui sintaxe, funcionamento distintos ou seja diferentes *API's* são melhores para diferentes casos. As *API's* oferecidas pelo Xenomai são:

- *Portable Operating System Interface (POSIX)*: interface padrão dos sistemas Unix.
- Alchemy: É a *API's* nativa do Xenomai e matém compatibilidade com versões anteriores do sistema operacional.
- VxWorks: Tem o intuito de ser compatível com o *RTOS* de mesmo nome porém não foi completamente implementada ainda.
- Smokey: É uma *API* especificamente para automatização de teste.
- *Real Time Driver Model (RTDM)*: É uma interface para desenvolvimento de *drivers*, tendo como criar um modelo padrão para os vários *RTOS* existentes.

Para esse projeto será usado a *API RTDM* pois é feita especificamente para a criação de *drivers* que sera o modo como o protocolo ponto-a-ponto sera implementado.

2.2 Protocolo

Um protocolo ponto-a-ponto é usado para estabelecer uma conexão direta entre dois nós, esse tipo protocolo foi escolhido devido a fato de ser mais simplificado quando comparado a protocolos ponto-multi-ponto que embora ofereçam a vantagem de aceitar mais de uma conexão simultânea implicam em complexidades adicionais. Decidiu-se ainda por desenvolver o protocolo para redes sem fio onde a questão do temporização é mais criticos que em redes cabeadas e a vantagem de se utilizar um *RTOS* seriam melhor aproveitadas e se tornam mais evidentes.

2.3 *Driver*

Programas em espaço de usuário não podem se comunicar diretamente com hardware devido ao fato do usuário não possuir certas permissões e a utilização de interrupções. Os *drivers* assumem esse papel fazendo comunicado entre o hardware e o restante do kernel e espaço de usuário (VENKATESWARAN, 2008). Para o desenvolvimento do protocolo será feito um *driver* o qual ficara responsável pela transmissão de dados através da interface de rede sem fio ao invés do padrão IEEE802.11n o qual seria o utilizado pelo sistema operacional normalmente. Como a implementação do *driver* depende do *hardware* já que ha uma comunicação direta entre os dois optou-se por utilizar o chipset AR9271 o qual é um componente bastante consolidado e o qual se tem acesso facilmente através de *dongles* disponíveis.

2.4 Testes

Para fim de testes pretende-se usar a ferramenta *iperf* a qual funciona como uma aplicação cliente servidor, na qual esse gera um tráfico de dados conforme os parâmetros passados para o servidor. Dessa forma é possível usar essa aplicação de forma a simular uma carga no canal de comunicação ponto-a-ponto e averiguar o desempenho do sistema com uma implementação em um sistema Linux normal e no *software* desenvolvida no Xenomai é possível averiguar se houve algum ganho devido à implantação

Figura 1 - Cenário de teste.

em *RTOS*, para fim de testes pretende se usar a princípio o cenário mostrado na figura 1 onde teremos um link ponto-a-ponto sem fio entre dois computadores. Para testes inicialmente pretende se testar a taxa de vazão de *upload* e *download* com e sem trafego no sentido contrário de comunicação, além da latência para qual será usado o *ping* e o tempo para estabelecimento da conexão, caso haja necessidade ou perceba que são necessárias novas informações puderam ser feitos outros testes a fim de se obter os dados desejados.

3 Resultados Esperados

Os objetivos iniciais propostos para o trabalho foram alcançados. Foi realizado um estudo aprofundado sobre o desenvolvimento de *drivers* para *RTOS*, foram realizados testes usando a ferramenta *iperf* afim de se averiguar a taxa de transmissão em *upstream* e *downstream*, com e sem trafego no sentido inverso, a latência e tempo para estabelecimento da conexão. Quando comprado com a implementação do protocolo em um sistema operacional tradicional a versão implementada no Xenomai apresentou um aumento considerável da taxa de transmissão.

Como sugestões de trabalhos futuros sugere-se:

- Implementação de um protocolo ponto-multiponto utilizando o Xenomai.
- Melhorar o desempenho do programa atual.

Referências

MAZIERO, C. *Sistemas Operacionais: Conceitos e Mecanismos*. [S.l.: s.n.], 2019.

MORAIS, J. *RTOS: Um ambiente multi-tarefas para Sistemas Embarcados*. 2018. Disponível em: <https://www.embarcados.com.br/rtos-sistema-operacional-de-tempo-real/>.

VENKATESWARAN, S. *Essential Linux Device Drivers*. First. Upper Saddle River, NJ, USA: Prentice Hall Press, 2008. ISBN 9780132396554.

XENOMAI. *Home*. 2019. Disponível em: <https://www.xenomai.org/>.