Uso de Linux como Sistema Embarcado*

A. F. Zorzo[1]†, C. M. da Costa[1], D. Belz[1], C. S. Valvassori[1], L. C. M. Caruso[2], R. Scop[2]

[1]Faculdade de Informática — Pontifícia Universidade Católica do Rio Grande do Sul
{zorzo, celso, belz, vassoura}@inf.pucrs.br

[2]Digitel S.A.
{lcaruso, scop}@digitel.com.br

Abstract. *The goal of this paper is to present Linux as an embedded operating system for network devices. This study is based on the port of Linux to a hardware prototype, which was designed and manufactured by Digitel S.A. In this paper, we describe the main steps and experience acquired during the porting of the Linux operating system.*

Resumo. *Este artigo tem como objetivo, apresentar o uso do Linux como um sistema operacional embarcado para dispositivos de rede de computadores. Este estudo é baseado no porte do Linux para o protótipo de hardware, projetado e desenvolvido pela Digitel S.A. Descrevem-se os principais passos e as experiências obtidas nessa tarefa.*

1. Introdução

Muitos dispositivos eletrônicos usados atualmente possuem componentes de hardware desenvolvidos para executarem tarefas específicas. Porém, o uso desse hardware especializado traz sérias desvantagens: i) o hardware é estático, ou seja, não pode ser atualizado ou modificado; ii) o projeto de um hardware atualizado consome tempo e recursos; iii) a modificação desse hardware pode provocar a reestruturação de todo o projeto do dispositivo eletrônico.

Uma alternativa existente é substituir os componentes de hardware especializado por outros genéricos, mais baratos e já a disposição no mercado.

Porém, a tarefa que esses dispositivos executam não deixou de ser especializada. Isso requer a implementação de uma camada de software para atuar sobre esse hardware, efetuando as mesmas tarefas que o hardware especializado e eliminando algumas de suas principais desvantagens. Esse hardware, que pode ser um microcontrolador ou até mesmo um processador, compõe junto com a camada de software um sistema embarcado (*embedded system*) [1].

Assim, telefones celulares, televisores, video-cassetes e outros aparelhos eletrônicos tem dentro de si um ou mais sistemas embarcados. Sistemas mais simples (baseados em microcontroladores) são programados na forma de um laço de controle, já que sua função não é mais complexa que a leitura de sensores ou o envio e recebimento de comandos simples.

Os sistemas mais complexos, que usam processadores, são normalmente dependentes de um sistema operacional, muitos deles proprietários (ex. Cisco IOS [2]) e desenvolvidos para arquiteturas específicas de processadores.

Por possuir o seu código-fonte aberto, o Linux tem obtido uma posição de destaque no segmento de sistemas operacionais embarcados. Afinal, apresenta vantagens como a possibilidade de se modificar o seu código-fonte, pode ser copiado gratuitamente pela Internet e já foi portado para várias arquiteturas diferentes, e.g. Intel x86, Sparc, PowerPC.

Existem diversas distribuições do Linux específicas para sistemas embarcados (HardHat, TinyLinux, Lineo, Embedix, etc.). Porém, como é fácil a sua personalização, distribuições mais

*Financiado com recursos da Digitel S.A. – Digitel/PUCRS/FACIN/TA001/001.
†A. F. Zorzo recebe auxílio do CNPq através de bolsa produtividade em pesquisa.

conhecidas (RedHat, Slackware, YellowDog, etc.) também podem ser usadas em ambientes embarcados.

O objetivo deste artigo é apresentar o uso do Linux como um sistema operacional embarcado, atuando como um roteador, operando na camada de rede do modelo OSI (*Open Systems Interconnection*) da ISO (*International Standards Organization*) [3]. Assim, o sistema deverá ser capaz de receber pacotes, verificar o seu destino, reempacotar os dados com o protocolo e tamanho apropriados e reenviá-los para o seu destino através do caminho mais curto conhecido.

2. Linux Embarcado

O Linux é atualmente o sistema operacional de código-aberto com maior número de usuários no mundo [4]. Essa característica fez do Linux um sistema operacional estável e atualizado constantemente, levando-o além das fronteiras do meio acadêmico, onde surgiu.

Dependente apenas de um utilitário de carga (*bootloader*), um *kernel* e um processo de inicialização, o Linux não ocupa muito dos já escassos recursos de um sistema embarcado. E, mesmo assim, pode ser complementado com *drivers* de hardware e alguns aplicativos, conforme a necessidade. Além disso, sem aumentar muito o seu tamanho, pode-se adicionar suporte à sistema de arquivos (em memória RAM ou ROM) e à conexão em rede através de TCP/IP [5].

Dessa forma, supera-se um dos maiores problemas do uso do Linux em sistemas embarcados: o espaço disponível em memória. A primeira vista, o uso do Linux como sistema embarcado é impossível, pois somente o código-fonte de um *kernel* da versão 2.4.x ocupa aproximadamente 24 MB compactado. Uma distribuição Linux padrão ocupa pelo menos um CD-ROM cheio (640 MB aproximadamente). O que acontece é que um sistema operacional assume a existência de teclado, *mouse*, monitor, disco rígido, entre outros periféricos. Além disso, essas distribuições trazem consigo uma série de bibliotecas, sistemas de arquivos e aplicativos que são úteis somente para ambientes de servidores e estações de trabalho.

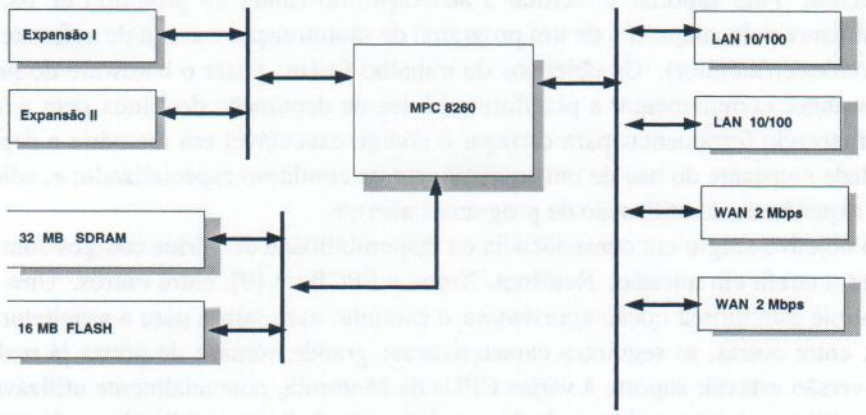

Figura 1: Arquitetura da placa roteadora.

Somente assumindo-se que todos esses *drivers* de dispositivos são desnecessários em um ambiente embarcado, pode-se reduzir significantemente o tamanho do *kernel* do Linux.

O que mais influi no tamanho do Linux em um sistema embarcado é a sua *ramdisk*, um disco virtual criado e manipulado em memória RAM, que implementa um sistema de arquivos.

Se essas aplicações forem específicas, podem executar diretamente em cima do *kernel*, efetuando chamadas diretamente à sua API (*Application Program Interface*). Porém, se existe a necessidade de usar mais aplicativos, principalmente aplicativos da distribuição Linux, necessita-se de pelo menos uma biblioteca *libc*, auxiliar para que o Linux possa funcionar [4]. Essa biblioteca ocupa aproximadamente 1.2MB na *ramdisk* [5].

3. Linux para a Plataforma de Roteamento

Após a determinação de todas as especificações que envolveram o software e o hardware alvo em questão (ver Figura 1), foi preparado um ambiente que possibilitou a execução dos variados testes na placa roteadora. Os componentes físicos principais que foram utilizados no projeto são visualizados na Figura 2, onde mostra-se o esquema montado para o porte de Linux embarcado.

O ambiente de desenvolvimento (Figura 2) possui os seguintes componentes: i) um *host* para acessar a placa alvo; ii) um cabo serial para vincular a placa ao *host*; iii) tanto o *host* como o protótipo da placa estão conectados a uma rede Ethernet pela porta LAN.

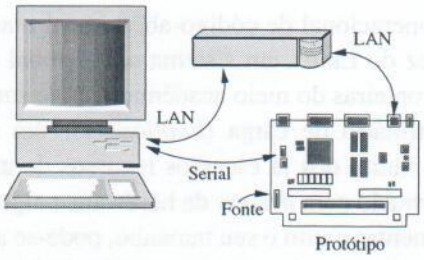

Figura 2: Ambiente de desenvolvimento.

Para que a comunicação ocorra entre a placa e o *host* foram necessárias as configurações dos seguintes recursos: i) *minicom*, que através do cabo serial acessa a placa roteadora e oferece um console no ambiente; ii) servidor de ftp (*tftp*); iii) conexão em rede dos equipamentos.

Um conjunto de variáveis de ambiente foi configurado através da console para informar dados sobre: i) a rede; ii) onde buscar a imagem do *kernel*; iii) onde buscar a imagem da *ramdisk*; iv) argumentos para o *boot* informando também o tamanho da *ramdisk*;

Carga do Sistema. Para suportar e facilitar a adaptação do Linux no protótipo de hardware, optou-se inicialmente pela adaptação de um programa de monitoração e carga de software para a placa alvo (*bootloader*/monitor). Os objetivos do trabalho foram: testar o hardware do primeiro protótipo do produto; complementar a plataforma básica de depuração do Linux para o sistema embarcado, fornecendo ferramentas para carregar o código executável em memória e depurá-lo, sem a necessidade constante do uso de um equipamento de emulação especializado; e, adicionalmente, ganhar experiência na utilização de programas abertos.

Este último objetivo surgiu em conseqüência da disponibilidade de vários códigos-fonte abertos para realizar a tarefa em questão: RedBoot, Xmon e PPCBoot [9], entre outros. Uma rápida análise revelou que esta última opção apresentava o caminho mais suave para a arquitetura alvo, por apresentar, entre outras, as seguintes características: grande número de portes já realizados (cerca de 50); versão estável; suporte à várias CPUs da Motorola, potencialmente utilizáveis por este fabricante; utiliza o mesmo ambiente de desenvolvimento do Linux; código baseado em Linux e NetBSD; boa documentação (README e fontes bem comentados); e, o mais importante, suporte especial para carregar o Linux sem necessidade de adaptação do *bootloader* embutido neste último, bem menos documentado e mais sujeito a modificações constantes.

Organização do Linux. Atualmente, tem surgido várias distribuições Linux para ambientes com poucos recursos e embarcados. Monta Vista Hard Hat Linux [6], Lineo [7] e BlueCat [8] são algumas dessas distribuições. A opção foi pelo Hard Hat Linux, principalmente por oferecer suporte a várias arquiteturas de processadores (PowerPC, StrongARM, x86, IA32, MIPS e SH). Já a escolha do *kernel* (versão 2.4.4) fornecido pela Denx Software Engineering [9] pode ser justificada pelo fato de que o *bootloader* (PPCBoot) usado é fornecido pela mesma empresa.

Por tratar-se de um projeto de hardware totalmente novo, alguns *drivers* precisaram ser adaptados, principalmente os responsáveis pela E/S da placa. Essas adaptações permitiram que as portas

LAN e serial pudessem ser acessadas. Assim, várias configurações do *kernel* puderam ser testadas, com a inclusão e remoção de alguns recursos, conforme a necessidade.

A partir do momento que foi alcançada uma configuração básica estável para o *kernel*, iniciou-se a montagem da *ramdisk*. Realizada no computador *host* (Figura 2), a montagem da *ramdisk* é um processo relativamente demorado, pois existe um limite de espaço (na placa, iniciou com 4 MB e alcançou seu valor ótimo em 12 MB), sendo que os programas têm de ser selecionados manualmente conforme a utilização que pretende-se dar ao sistema.

A *ramdisk* está organizada da seguinte forma: contém um sistema de arquivos, *ext2*, do Linux com um conjunto de programas básicos para os objetivos do sistema. Entre os programas pode-se citar os seguintes: *login*, *telnet*, *ping*, *init*, *bash*, *cp*, além de bibliotecas e protocolos. Os arquivos de configuração são armazenados na memória Flash.

Um *driver* foi escrito para permitir o particionamento da Flash. Com esse driver, a configuração da Flash no *kernel* ficou mais simples. Além disso, evitou que fossem necessárias modificações em outros arquivos do *kernel*. Cada partição, após a criação da imagem do *kernel* no computador *host*, é estática, ou seja, não pode ser redimensionada. Porém, é possível mudar seu tamanho alterando-se o código-fonte.

Após a Flash ter sido detectada e particionada, tratou-se da implementação do sistema de arquivos sobre ela. O sistema de arquivos mais indicado é o JFFS, pois foi desenvolvido para ser usado exatamente em memórias Flash. Uma nova versão desse sistema de arquivos (JFFS2) traz algumas melhorias, mas por ainda estar em desenvolvimento, alguns erros podem ocorrer [10].

4. Conclusões

No trabalho realizado, três pontos fundamentais são: a construção da placa, o *boot* do sistema, e o porte do Linux para a placa. O primeiro ponto foi concluído com êxito. O protótipo se tornou operacional dentro do prazo previsto sendo que a depuração transcorreu sem grandes imprevistos. Para o *boot* do sistema, foi escolhido o PPCBoot para servir como base de desenvolvimento, o qual foi modificado de maneira a atender as especificações da placa.

Duas vantagens adicionais de uso do PPCBoot foram observadas: o trabalho de adaptação do *kernel* do Linux para a placa alvo reduziu-se drasticamente (em apenas dois dias, a partir do término da depuração do PPCBoot e da placa alvo, já havia um *kernel* portado em depuração, pelo mesmo desenvolvedor); e o ambiente criado revelou-se um bom substrato para implementação e teste de rotina de diagnóstico de problemas de hardware, necessárias tanto na área de produção como para depuração do produto em campo.

Quanto ao porte do Linux para a placa roteadora está de acordo com as espectativas: i) a memória Flash é detectada normalmente, todas as suas partições estão prontas, executa leitura e gravação; ii) a imagem do *kernel* e a *ramdisk* são gerados e carregados para a placa; iii) diversos protocolos são carregados e estão operacionais, e.g. TCP/IP, I PSec, RIP, RIP2, BGP, EGP, PPP.

Referências

[1] W. Wolf. *Computers as Components: Princ. of Embedded Computer Syst. Design*, Morgan 2000.

[2] CISCO. *CISCO Web Site*, www.cisco.com.

[3] L. F. G. Soares, G. Lemos, S. Colcher. *Redes de Computadores*, Editora Campus, 1995.

[4] J. Lombardo. *Embedded Linux*, Ed. New Riders, 2001.

[5] J. R. Williams. *Embedding Linux in a Commercial Product*, Linux Journal, 1994.

[6] MontaVista. *MontaVista Web Site*, www.mvista.com.

[7] Lineo inc. *Lineo Web Site*, www.lineo.com.

[8] LynuxWorks. *BlueCat Linux Web Site*, www.bluecat.com.

[9] Denx Software Engineering. *Denx Web Site*, www.denx.de.

[10] Axis Communications. *Axis Web Site*, www.axis.com.