DESENVOLVIMENTO DE SISTEMAS EMBARCADOS UTILIZANDO UCLINUX.

Alberto Alexandre Moura ALBUQUERQUE (1);André Luiz Carneiro ARAÚJO (2); Lucas Paulino AZEVEDO (3)

(1) Instituto de Telemática – CEFET-CE
Av. Treze de Maio, 1081 Benfica CEP 60.040-531 Fortaleza-CE, Telefone: 3307-3654 , e-mail: alberto.alexandre@gmail.com

(2) Instituto de Telemática – CEFET-CE, e-mail: andreluiz@cefetce.br

(3) Instituto de Telemática – CEFET-CE, e-mail: lucasazevedo9@gmail.com

RESUMO

O mercado de sistemas de controle e automação vem a cada dia exigindo mais funcionalidades e flexibilidade de operação dos sistemas embarcados. Isso exige dos desenvolvedores respostas mais rápidas no desenvolvimento das aplicações de Sistemas Embarcados (SEs). A utilização de Sistemas Operacionais Embarcados (SOs) aparece como solução viável, pois possibilita aos desenvolvedores uma maior produtividade. O *uClinux*, implementação do Linux para plataformas sem *MMU* (unidade de gerenciamento de memória), como por exemplo, a plataforma ARM, possibilita recursos de programação semelhantes aos já existentes em plataformas PC, tornando o desenvolvimento de aplicações bem mais simples e rápido. O objetivo deste trabalho é mostrar as vantagens de se trabalhar com *uClinux* em um sistema embarcado. A metodologia utilizada será utilizar comparações entre o desenvolvimento de um sistema, numa mesma plataforma (ARM), sem utilizar o *uClinux* e utilizando *uClinux*, onde serão observados o tempo de desenvolvimento e a complexidade da programação nos dois casos.Este trabalho visa mostrar como resultado as vantagens obtidas ao se desenvolver sistemas embarcados com *uClinux*, frente a sistemas que não utilizam um sistema operacional embarcado.

Palavras-chave: *uClinux*, sistemas embarcados

1. INTRODUÇÃO

Ao longo dos últimos anos, o número de aplicações com os ditos "Sistemas Embarcados" (SEs) explodiu. Os SEs podem ser vistos no mercado na forma de *mp3 players*, roteadores, telefones VoIP, *scanners*, leitores de cartão, *smartphones* , entre outros. Os microcontroladores, muito utilizados no desenvolvimento de SEs , também tem evoluído bastante, com capacidade de processamento elevadas e com unidades de armazenamento de dados com tamanho considerável, de modo que algumas plataformas, como ARM, Blackfin, MIPS, FreeScale, por exemplo, já possuem capacidade de abrigar um sistema operacional,ainda que reduzido, semelhante ao dos PC's.

Com um número crescente de produtos diferentes nascendo a cada dia, o mercado vem exigindo cada vez mais rapidez e confiabilidade no desenvolvimento de aplicações para as plataformas usadas em SEs.O *uClinux* , um sistema operacional para plataformas embarcadas, pode atuar como uma ferramenta para diminuir o tempo e melhorar a confiabilidade no desenvolvimento de aplicações para estas plataformas, visto que este torna o ambiente para o desenvolvimento bem próximo ao ambiente PC, onde permite a utilização de alguns recursos deste , facilitando assim , em muito, o trabalho do desenvolvedor.

No presente trabalho, estudamos as facilidades de se desenvolver aplicações em um ambiente de sistema embarcado utilizando um Sistema Operacional (SO) em contra partida ao mesmo sistema sem a utilização de um SO. Para tal, foi utilizado uma placa de desenvolvimento da *Embedded Artists* utilizando o microcontrolador ARM LPC2468 da NXP, e como sistema operacional uClinux (kernel 2.6).

2. FUNDAMENTAÇÃO TEÓRICA

2.1. Sistemas Embarcados

São ditos sistemas embarcados, sistemas em que o processador ou qualquer aparato computacional é inteiramente dedicado ao dispositivo ou sistema que controla. Os SEs são geralmente constituídos pelo: processador, memória e hardwares específicos. Alguns contam com interfaces para o usuário, que podem variar desde simples LED's indicadores e botões até monitores e teclados como os dos PC's. Como o sistema é dedicado à tarefas específicas, através de engenharia pode-se otimizar o projeto reduzindo tamanho, recursos computacionais e custo do produto. O *software* escrito para sistemas embarcados é muitas vezes chamado *firmware*, e armazenado em uma memória ROM ou memória flash ao invés de um disco rígido.

Como os sistemas embarcados são construídos para executar apenas uma tarefa pré-determinada, muitas vezes não têm flexibilidade (de software e de hardware) que lhes permita fazer outras tarefas quaisquer que não sejam aquelas para as quais foram desenhados e desenvolvidos.

2.2. O microcontrolador ARM

Os microcontroladores de 32 bits construídos com a tecnologia do *core* ARM, devido a sua simplicidade de arquitetura e baixo consumo de energia têm sido amplamente utilizados, para as mais diversas aplicações. A versão mais utilizada atualmente no mercado é o ARM7.

O ARM7 é um processador RISC, com *pipeline* de três estágios e arquitetura de registradores *load-and-store*. O processador estará sempre ocupado em executar três instruções em diferentes estágios. Enquanto busca a primeira, decodifica a segunda e executa a terceira. Essa arquitetura de *pipeline* é bem simples e evita os conflitos entre estágios de arquiteturas mais avançadas.

O processador ARM7 possui sete diferentes modos de execução. Partes desses modos são usadas para tratamento de exceções e de interrupções. Um sistema operacional pode utilizar os modos privilegiados para executar códigos de sistema e deixar as aplicações restritas no modo usuário.

2.3. O uClinux

Linux é um sistema operacional que descende do UNIX e adota a licença GPL de software livre, logo seu código-fonte está disponível para ser utilizado, estudado, alterado e distribuído de acordo com os termos da licença. O nome Linux também é usado como referência para o kernel do sistema, que teve sua primeira versão desenvolvida no começo dos anos 90 pelo finlandês Linus Torvalds, na época um estudante da Universidade de Helsinki. Desde então o kernel do Linux sofreu diversas modificações, tornando-se cada vez mais poderoso e tomando dimensões que durante sua criação não eram esperadas, tendo seu desenvolvimento coordenado por Linus com a ajuda de milhares de pessoas ao redor do mundo inteiro.

O kernel do Linux é híbrido monolítico, pois as funções do kernel (as quais gerenciam os recursos de hardware) são normalmente executadas em espaço de kernel, porém algumas podem rodar com permissões de usuário. Outra característica muito importante do kernel é que ele é um dos mais portáveis atualmente, podendo ser utilizado em diversas plataformas, como por exemplo, ARM, PowerPC, Intel, AMD, Alpha, Motorola M68k e BlackFin.

Essa propriedade do Linux que proporcionou a possibilidade de desenvolvimento de desenvolvimento de projetos de sistemas embarcados com o uso de programação de mais alto nível, aumentando a produtividade dos programadores. Atualmente existem várias distribuições (nome dado aos sistemas operacionais que se utilizam do kernel Linux) que visam essa área, como o PizzaBox Linux, BlueCat Linux, TASTE Linux, e o uCLinux que foi utilizado para a escrita do artigo.

O uCLinux original é um derivado do kernel Linux 2.0, com o intuito de ser usado para microcontroladores sem MMU (*Memory Management Unity* - Unidade de Gerenciamento de Memória). Com o crescimento do projeto, hoje o uCLinux abrange várias arquiteturas de processadores e é um sistema que tem uma gama de aplicações de usuário, bibliotecas e *toolchains*. A distribuição é otimizada para ter um tamanho reduzido, e usa alternativas compactas de bibliotecas como a uClibc, e software de administração de sistema, como o busybox.

3. PROGRAMAÇÃO DE UM SISTEMA EMBARCADO COM UCLINUX

No desenvolvimento de aplicações para SEs, temos entre as principais motivações de se utilizar o uCLinux como plataforma de desenvolvimento:

3.1. Portabilidade

Portabilidade é a capacidade de uma aplicação ser escrita, compilada e executada em diferentes ambientes de *hardware* e/ou *software*, e o que define o quão uma aplicação é portável (ou portátil) é a facilidade com a qual ela é utilizada em outro ambiente. O número de CPU's diferentes e sistemas operacionais usados em computadores pessoais (*desktops*) atualmente é muito pequeno, com predominância da arquitetura x86 e de poucos SO's. Mas no mercado de sistemas embarcados, a portabilidade é um problema significante. Ao se desenvolver uma ferramenta direto para uma arquitetura específica, é provável que a aplicação fique restrita ao *hardware* utilizado. Porém muitas das aplicações escritas para uClinux têm seu código fonte feito independentemente da arquitetura, sendo executáveis em todas que o suportam ou é facilmente re-compilado para a ambiente em questão ou que sofra pequenas alterações para se adequar ao novo *hardware*.

3.2. Abstração de Hardware

Dispositivos comuns, como uma porta serial, um teclado ou um mouse utilizam-se de interfaces de driver que são acessadas seguindo um padrão, logo mesmo que os drivers sejam desenvolvidos para uma arquitetura específica, o modo como o a aplicação acessa o driver deverá ser exatamente o mesmo para as diferentes arquiteturas existentes no mercado. Do ponto de vista do desenvolvedor, o SO poupa o trabalho

deste em ter que fazer a configuração básica de periféricos, mantendo-se sempre num nível mais alto e inteligível de programação.

3.3. Reutilização de código

Reutilização de código é o conceito que pode ser definido como o aproveitamento de experiências passadas em desenvolvimento de *software*, seja utilizando partes do próprio código em projetos futuros ou em utilizar partes do próprio projeto em outros. O desenvolvedor pode utilizar aplicações básicas desenvolvidas por ele em vários projetos diferentes, não havendo a necessidade de "reinventar a roda", além do que há uma comunidade de projetistas e usuários tanto de Linux quanto do próprio uCLinux que interagem entre si com contribuições mútuas.

3.4. Facilidade de Manutenção da Aplicação

Uma aplicação desenvolvida em cima de um sistema operacional é mais fácil de ser escrita, testada, debugada e mantida. O desenvolvedor além de abstrair o *hardware*, pode utilizar seus conhecimentos em Linux para tornar sua aplicação mais eficiente, pode fazer os testes da aplicação facilmente utilizando-se apenas de um servidor NFS conectado à rede e o mesmo vale para testar todo o sistema com mudanças cruciais no kernel sem a necessidade de gravar a imagem binária no hardware, utilizando-se da capacidade do u-boot de iniciar o sistema em um servidor de mesmo tipo.

3.5. Facilidade de interação com estações Linux

As aplicações desenvolvidas para o uCLinux podem se comunicar de modo simples com estações que utilizem Linux ou sistemas operacionais derivados do UNIX que usem programas compatíveis, podendo monitorar ou ser monitorado por eles. O uCLinux tem suporte a vários protocolos de rede, e também pode ser utilizado como um *web service* embarcado, podendo ser um *web-server*, um servidor DNS, um servidor de *e-mail*, um roteador, um limitador de uso de banda.

4. ANÁLISE E INTERPRETAÇÃO DOS DADOS

Para uma análise qualitativa foi feita uma experiência onde dois programadores (com conhecimentos semelhantes), tiveram que desenvolver uma aplicação onde o primeiro utilizaria uma plataforma ARM com o uClinux embarcado para o desenvolvimento da aplicação, e o segundo a mesma plataforma ARM programando diretamente para a plataforma sem auxílio de um sistema operacional embarcado.

A aplicação a ser desenvolvida pelos dois programadores, consiste em utilizar os recursos de comunicação serial da plataforma para fazer uma interface com um modem GSM/GPRS, através de comandos AT no objetivo de fazer uma conexão com a internet via GPRS e estabelecer uma comunicação TCP\IP via *socket* com um servidor remoto.

As aplicações foram escritas em linguagem C, sendo que para a plataforma com o Uclinux embarcado foi utilizado o C padrão Unix, usando o compilador GCC e para a plataforma sem uClinux foi utilizado o C com padrão específico para os ARM da família LPC2xxx do compilador IAR.Para ambas as aplicações foram utilizada a placa de desenvolvimento da *Embbedded Artists* LPC2468 OEM Board + OEM *Base Board Basic*.Analisaremos adiante as dificuldades encontradas, por ambos os programadores para o desenvolvimento da aplicação,

4.1. Caso 1: Programação sem o uClinux

Para o inicio do desenvolvimento da aplicação, o programador teve que estudar primeiramente as mudanças e instruções específicas da linguagem C utilizada na programação do microcontrolador LPC2468 pelo compilador IAR. O compilador oferece algumas bibliotecas, para utilização e configuração dos periféricos do microcontrolador, neste caso a serial. Sendo que esta tem que ser configurada em seu nível mais básico acessando os registros especiais do processador para programar os atributos como a taxa de transferência sendo para isso necessário um conhecimento anterior da freqüência do cristal do circuito oscilador do

microcontrolador, e a realização de cálculos seguindo orientações vindas na folha de dados do microcontrolador.

Para a rotina de recepção foi usada uma interrupção do microcontrolador, algo que exigiu certo tempo para o entendimento de sua funcionalidade e como deveria ser realizada. O programador A teve o tempo total de 10 horas no desenvolvimento desta aplicação.

4.2. Caso2: Programação com o uClinux

A aplicação foi desenvolvida utilizando-se uma *toolchain* (arm-elf-gcc) do compilador *gcc* no Linux, para a escrita do código não foi necessário um conhecimento adicional além daquele da linguagem C padrão Unix, neste padrão o C "enxerga" a serial como um simples ponteiro de arquivo enviando e recebendo dados como se estivesse lendo e escrevendo em um arquivo. Foi utilizado pelo programador o recurso de *pthreads* para se utilizar da recepção de dados da porta serial enquanto o programa principal cuidava da transmissão dos dados pela mesma. O conhecimento a mais exigido além daquele necessário para programação em PC foi, no caso desta aplicação, mínimo, limitando-se apenas ao conhecimento de como utilizar corretamente a toolchain do compilador *gcc*. O programador B teve o tempo total de 6 horas no desenvolvimento desta aplicação.

5. CONCLUSÃO

Observamos que a análise realizada neste trabalho mostrou que o uClinux pode atuar como uma ferramenta poderosa na ótica de facilitar e agilizar o desenvolvimento de sistema embarcados em plataformas que suportam o um sistema operacional como o uClinux. Como visto, para uma aplicação simples tivemos um ganho de tempo da ordem de cinqüenta por cento, frente a um desenvolvimento na mesma plataforma sem o uso do uClinux, este ganho de tempo tende a ser menor com o aumento da complexidade das aplicações e na utilização de hardwares mais específicos.

REFERÊNCIAS

http://opensrc.sec.samsung.com/document/Getting_Familiar_with_uClinuxARM2_6.html | acessado dia 16/08/2008

http://docs.blackfin.uclinux.org/doku.php?id=why_use_uclinux | acessado dia 16/08/2008

http://docs.blackfin.uclinux.org/doku.php?id=operating_systems | acessado dia 16/08/2008

http://www.uclinux.org/ | acessado dia 16/08/2008

http://www.linux.org/info/ | acessado dia 16/08/2008

http://br-linux.org/faq-linux/?q=faq-linux | acessado dia 16/08/2008

http://www.linuxdevices.com/articles/AT2760742655.html | acessado dia 16/08/2008

AGRADECIMENTOS

A Deus.