

Instalação do RTAI (Real-Time Application Interface) utilizando o Ubuntu 8.04

Elmo L. S. Libório, Tiago F. Machado
Universidade Federal do Vale do São Francisco (UNIVASF)
Sistemas de Tempo-Real – CCMP0073 – Turma C0
Juazeiro – Bahia - Brasil
{elmo.leo,tiago_otti}@hotmail.com

1. Introdução

Na medida em que o uso de sistemas computacionais prolifera na sociedade atual aplicações com requisitos de tempo real tornam-se cada vez mais comuns. Essas aplicações variam muito em relação à complexidade e às necessidades de garantia no atendimento de restrições temporais [Oliveira et al. 2000]. É comum encontrarmos desde os mais simples controladores de eletrodomésticos a aplicações que exijam uma restrição de tempo mais rigorosa como sistemas de monitoramento de pacientes em um hospital e sistemas embarcados nos veículos como Airbag.

O *RTAI (Real-Time Application Interface)* permite que seja criada uma camada de abstração sobre o sistema linux para execução de tarefas de tempo real, esse documento descreve todo o processo de construção do RTAI 3.7 no Ubuntu 8.04.

2. Desenvolvimento

2.1 Escolha dos componentes

Ao iniciarmos a implementação deste experimento, realizamos uma coleta de informações acerca de compatibilidade entre as ferramentas necessárias para a instalação do RTAI sobre o Ubuntu. Com a o ideal de criarmos uma espécie de tutorial de instalação

atualizado, com as mais modernas ferramentas, fizemos experimentos em três versões do Ubuntu, a 10.04, 9.04 e 8.04. Nas duas primeiras versões encontramos muitos problemas de incompatibilidade, e optamos então pela versão 8.04.

A instalação do RTAI, necessita da compilação de uma versão estável do kernel do linux, utilizamos a versão 2.6.28.7. A escolha da versão do RTAI no nosso experimento foi a versão 3.7 a mais citada nas referencias que consultamos. O gcc e g++ compatíveis são de extrema importância para construção da camada. Com o intuito de tentar usar ferramentas com versões atuais, utilizamos o gcc e g++ 4.1, pois versões maiores ou iguais a 4.3 são, de acordo com pesquisas e erros experimentais, incompatíveis.

De forma sucinta utilizamos para esse documento as versões:

- Ubuntu 8.04
- Kernel 2.6.28.7
- RTAI 3.7
- gcc e g++ 4.1

Após a escolha das versões há a necessidade de utilizarmos ferramentas que auxiliam na montagem desse quebra-cabeça. A seção seguinte descreve a instalação das ferramentas.

2.2 Instalação dos Componentes

Para o carregamento e gerenciamento dos módulos do kernel necessitamos instalar um módulo de inicialização de ferramentas através do comando:

```
# sudo -s
```

Que define o usuário root, onde é necessário o preenchimento da senha de administrador.

```
# apt-get install module-init-tools
```

Uma biblioteca necessária para o menu de configuração do Kernel, também foi instalada:

```
#apt-get install kernel-package linux-source libncurses5-dev
```

Faz-se necessário a instalação de softwares que criam arquivos .deb após a compilação do kernel:

```
#apt-get install kernel-package fakeroot
```

A ferramenta automake produz arquivos makefile portáveis para uso do programa make e é usada na compilação de softwares.

```
#apt-get install libtool automake
```

A versão do gcc instalada no Ubuntu 8.04 é a 4.2, realizamos testes sobre essa versão, mas após a compilação e instalação do kernel 2.6.28.7, o mesmo não conseguiu ser inicializado. Desta forma removemos a versão 4.2 e colocamos a versão 4.1 como já citado anteriormente, os passos seguintes descrevem o procedimento necessário para troca da versão.

```
# apt-get install gcc-4.1 g++-4.1
```

```
# cd /usr/bin
# cp gcc gcc-X.X (X.X é a versão atual)
# rm gcc
# cp gcc-4.1 gcc
# cp g++ g++-X.X (X.X é a versão atual)
# rm g++
# cp g++-4.1 g++
```

Com o sistema preparado o próximo passo foi o download da versão escolhida do kernel.

```
#cd /usr/src
#wget
http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.28.7.tar.bz2
```

Após o término do download, devemos realizar a descompactação, e criação de um link simbólico, que é um pequeno arquivo que aponta para outro arquivo no sistema de arquivos, uma espécie de ponteiro.

```
#tar xjvf linux-2.6.28.7.tar.bz2
#ln -s linux-2.6.28.7 linux
```

O passo posterior é o download do magma que é um branch de desenvolvimento de projeto:

```
#cd /opt
#cvcs -
d:pserver:anonymous@cvs.gna.org:/cvs/rtai
co magma
#ln -s magma rtai
```

O download, descompactação e criação de link simbólico do RTAI são realizadas na sequência:

```
#cd /opt
#wget --no-check-certificate
https://www.rtai.org/RTAI/rtai-3.7.tar.bz2
#tar xjvf rtai-3.7.tar.bz2
#ln -s rtai-3.7 rtai
```

Terminados os downloads e preparativos, seguimos para compilação, configuração e instalação do Kernel 2.6.28.7.

2.3 Configuração e compilação do Kernel

Antes de configurarmos o kernel devemos aplicar o patch da versão escolhida 2.6.28.7, através dos comandos:

```
#cd /usr/src/linux
#patch -p1 <
/opt/rtai/base/arch/x86/patches/hal-linux-
2.6.28.7-x86-2.2.06.patch
```

Aplicado o patch, através do comando:

```
#make menuconfig
```

Podemos configurar o kernel para receber o RTAI, o menu de configuração é aberto na tela como mostra a Figura 1, e devem ser editadas algumas configurações:

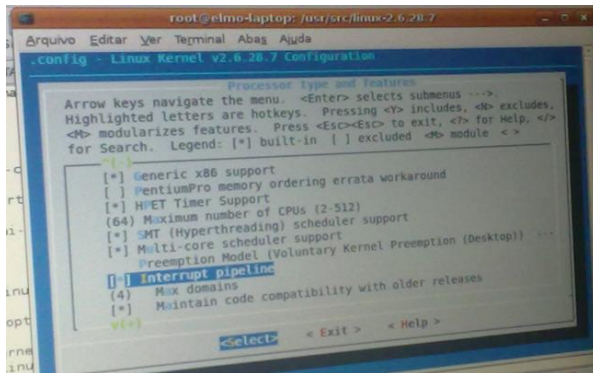


Figura 1: Menu de configuração do Kernel

General Setup

- Prompt for development and/or incomplete code/drivers: (Y)
- Local version - append to kernel release: (-rtai)

Enable loadable module support: (Y)

- Module versioning support: (N)

Processor Type and Features

- Interrupt Pipeline: (Y)
- Timer frequency: (1000 Hz)
- Preemption model: Low Latency Desktop

Kernel hacking

- Compile the kernel with debug info: (N)
- Compile kernel with frame pointers: (N)

Após inúmeros experimentos e tentativas de modificações nas configurações essas foram as que possibilitaram a compilação e inicialização do kernel, porém pode ser necessário e é extremamente recomendável, segundo o manual do RTAI, desabilitar as opções relacionadas ao Gerenciamento de Energia: ACPI, APM e Graduação da Frequência da CPU.

Além dessas modificações sugeridas pelo RTAI, pode se configurar outros pontos tais como tipo de processador.

Processor Type and Features

- Processor Family: "O seu processador"

Finalizadas as configurações do kernel o comando abaixo o compila.

```
#make-kpkg --initrd kernel_image
kernel_headers
```

O processo demora cerca de 40 minutos, no laptop utilizado no experimento, com as seguintes configurações:

- Processador Intel Core 2 duo
- 4GB memória RAM

Após a conclusão devemos realizar os procedimentos abaixo:

```
#cd ..
```

Em seguida é necessário criar uma pasta de firmware para esta instalação, devido a um bug no pacote criado ao qual gera uma mensagem de erro durante a posterior instalação do pacote, por isso a necessidade deste passo:

```
#mkdir /lib/firmware/2.6.23-rtai
```

Em seguida deve ser instalado os pacotes .deb criados na compilação.

```
#dpkg -i *.deb
```

Deve-se colocar o rtai-3.7 no diretório /usr/src/rtai usando o comando:

```
#cp -R rtai-3.7 /usr/src/rtai
```

Feito isso o reboot com o kernel novo é necessário, a Figura 2 mostra como fica sua tela de boot após esse procedimento.

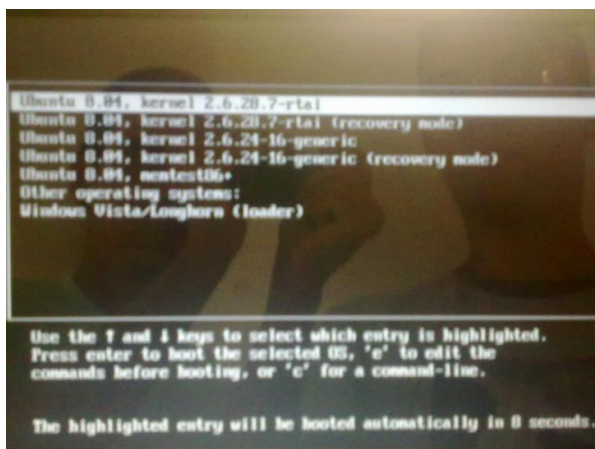


Figura 2: Tela de boot.

Se ocorrer erro na inicialização do novo Kernel, as configurações podem ser refeitas usando os seguintes passos:

```
#sudo -s  
#cd /usr/src/linux  
#make clean  
#make mrproper
```

Em seguida usando o comando abaixo podem ser realizadas as alterações necessárias.

```
#make menuconfig
```

2.3 Configuração e compilação do RTAI

No kernel novo deve ser compilado o RTAI. Ao abrir o terminal usamos os comandos:

```
# sudo -s  
#cd /usr/src/rtai
```

Em seguida criamos a pasta /build para a compilação.

```
#mkdir build  
#cd build
```

O comando abaixo abre o menu de configuração do RTAI

```
#make -f ../makefile
```

Devem ser editadas as seguintes configurações para que tenha suporte a RTnet:

Base System

- Supported Services
 - Use RTnet: Y

Add-ons

- Real-Time Driver Model over RTAI: (Y)

Após a configuração o comando abaixo instala o RTAI.

```
#make install
```

Para testarmos se o RTAI está instalado corretamente seguimos os seguintes passos

```
#cd /usr/realtime/modules/
```

```
#insmod rtai_hal.ko
```

```
#cd /usr/realtime/testsuite/kern/latency
```

```
#./run
```

Se nenhum erro ocorrer o RTAI estará instalado corretamente. Como na Figura 3.

```
RTAI Testsuite - USER latency (all data in nanoseconds)
2007/08/15 13:30:48
RTH|  lat min|  ovl min|  lat avg|  lat max|  ovl max|  overruns
RTD| -1229| -1229| 3436| 8852| 8852| 0
RTD| -869| -1229| 3443| 6812| 8852| 0
RTD| -1220| -1229| 3393| 6708| 8852| 0
RTD| -1092| -1229| 3436| 10219| 10219| 0
RTD| -1073| -1229| 3445| 8541| 10219| 0

Agora, testaremos em modo kernel:
# cd /usr/realtime/testsuite/kern/latency
# ./run

Temos uma saída parecida no modo kernel:
RTAI Testsuite - KERNEL latency (all data in nanoseconds)
RTH|  lat min|  ovl min|  lat avg|  lat max|  ovl max|  overruns
RTD| -2084| -2084| 2418| 4891| 4891| 0
RTD| -2130| -2130| 2424| 5779| 5779| 0
RTD| -2097| -2130| 2436| 6094| 6094| 0
RTD| -2090| -2130| 2434| 8015| 8015| 0
RTD| -2088| -2130| 2455| 9054| 9054| 0
```

Figura 3: Tela de teste esperada.

No nosso experimento o RTAI foi compilado corretamente porém na tela de testes Figura 4. Após o comando `insmod rtai_hal.ko`, nos deparamos com um erro na instalação, foram realizadas inúmeras tentativas de corrigi-lo, porém nenhuma com sucesso.

```
root@elmo-laptop: /usr/realtime/modules
Arquivo Editar Ver Terminal Abas Ajuda
elmo@elmo-laptop:~$ sudo -s
[sudo] password for elmo:
root@elmo-laptop:~# cd /usr/realtime/modules/
root@elmo-laptop:/usr/realtime/modules# insmod rtai_hal.ko
insmod: error inserting 'rtai_hal.ko': -1 Operation not permitted
root@elmo-laptop:/usr/realtime/modules#
```

Figura 4: Tela de teste com erro.

3. Conclusão

Através do experimento fomos capazes de identificar a importância da construção correta do RTAI para obter uma camada abstrata que possibilita o escalonamento prioritário de tarefas de Tempo Real, garantindo que o Linux não interfira nessas tarefas, lançando todo o sistema operacional como uma tarefa de baixa prioridade.

Foi possível perceber o quão simples é a obtenção dos recursos utilizados no experimento (*Open Sources*), por outro lado ficou clara a dificuldade em se construir um ambiente propício aos testes e exigências rigorosas de um Sistema de Tempo Real.

Fica a perspectiva para que nos trabalhos futuros consigamos fazer a compilação com novas versões do Kernel e RTAI além de execução de exemplos no RTAI, não possível nesse trabalho devido ao erro apresentado na construção do mesmo.

4. Referencias

Cavalcante, A., *Sobre Open Source Software*. Available in: <http://sobreoss.blogspot.com/2009/05/compilando-o-rtai-no-ubuntu.html>

Oliveira, S. R., Farines, J. M., Fraga, S. J. (2000). *Sistema de Tempo Real* - UFSC – Universidade Federal de Santa Catarina.

RTAI. *The RealTime Application Interface for Linux from DIAPM*. Available in: <https://www.rtai.org/>