LABORATÓRIO TPSE I

Laboratório 01: Usando um toolchain pré-compilado

Prof. Francisco Helder

8 de agosto de 2022

Nesta atividade iremos instalar, estudar e testar um toolchain pré-compilado, com objetivo de gerar o bootloader U-boot, o istema operacional Linux e aplicações para a placa BeagleBone Black.

Utilizaremos o toolchain da Linaro, que pode ser baixado em http://www.linaro.org/downloads/. Para facilitar nosso trabalho, baixe e descompacte no ambiente de laboratório como mostrado no exemplo ao lado “ex: /home/XXX/labs/”.

1 Instalando o toolchain

Abra um novo terminal e entre no diretório de exercícios do treinamento:

```
$ cd /home/XXX/labs/toolchain
```

Descompacte o toolchain:

```
$ tar -Jxvf  ~/Downloads/gcc-linaro-13.0.0-2022.08-x86_64_arm-linux-
   gnueabihf.tar.xz
```

Adicione à variável de ambiente PATH o caminho dos binários do toolchain. Desta forma, teremos acesso às ferramentas de compilação do toolchain no terminal corrente.

```
$ export PATH=/home/XXX/labs/toolchain/gcc-linaro-arm/bin:$PATH
```

Pronto, seu toolchain está instalado!

Dica: sempre que você alterar ou criar com o comando export uma variável de ambiente, esta alteração se perderá quando você abrir um outro terminal ou reiniciar sua máquina. Se você quiser manter as alterações feitas em uma variável de ambiente, pode incluir estas alterações no arquivo “ /.bashrc”. Este arquivo é um shell script que é executado sempre que você inicia uma nova seção de terminal.

1.1 Testando o toolchain

Vamos criar uma aplicação simples em linguagem C para testar o toolchain. Entre no diretório de exercícios do treinamento:

```
$ cd /home/XXX/labs/01
```

E implemente a aplicação:

```
$ gvim main.c
```

```
#include <stdio.h>
int main() {
        printf("Hello embedded world!\n");
        return(0);
}
```

Compile a aplicação para ARM com o toolchain gerado pelo crosstool-ng:

```
$ arm-linux-gcc main.c -o main
```

Use o comando file para verificar as informações do arquivo. Perceba que a aplicação foi compilada para a arquitetura ARM:

```
$ file main
```

main: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, not stripped

1.1.1 Linkando a Aplicação Estaticamente

Compile novamente a aplicação para ARM, porém linkando com as bibliotecas do sistema de forma estática. Depois compare o tamanho do binário gerado com a mesma aplicação compilada e linkada dinamicamente.

Dica: para identificar os parâmetros necessários para compilar uma aplicação estaticamente, dê uma olhada na página de manual do gcc:

```
$ man gcc
```

Para fazer uma busca dentro de uma página de manual, pressione "/" e digite a(s) palavra(s) que deseja procurar. Pressionando a letra 'n' (next) você pode avançar para a próxima palavra na busca.

2 Atividades Práticas

pratica 1:

Cross-compile uma aplicação "Hello World!" com a lib *stdio* lincado dinâmica e estaticamente, envie e execute no Linux que está rodando na BleagleBone Black.

Laboratório 02: Gerando seu Próprio toolchain

Nesta atividade iremos configurar e compilar nosso próprio toolchain usando a ferramenta crosstool-ng. Este será o toolchain que depois utilizaremos para compilar o bootloader, o kernel, as bibliotecas e aplicações que desenvolveremos no decorrer do curso. O crosstoo-ng pode ser baixado da página do projeto em http://crosstool-ng.org/.

Pacotes necessários para realização deste laboratório:

```
$ sudo apt install build-essential git autoconf bison flex texinfo
   help2man gawk libtool-bin libncurses5-dev unzip
```

2.1 Baixando e compilando o toolchain

Abra um novo terminal e entre no diretório de exercícios do treinamento, baixe os fontes do Crosstool-ng, através de seu repositório git, e mude para um commit desejado:

```
$ cd /home/XXX/labs/02
$ git clone https://github.com/crosstool-ng/crosstool-ng
$ cd crosstool-ng/
```

Como não estamos construindo o Crosstool-ng a partir de um arquivo específico, mas de um git commit, primeiro precisamos gerar um script de configuração e, de maneira mais geral, todos os arquivos gerados que são enviados no arquivo de origem para um específico:

```
$ ./configure --enable-local
$ make
```

Pronto! Agora a ferramenta está pronta para uso. Uma lista de parâmetros disponíveis pode ser exibida com o comando abaixo:

```
$ ct-ng help
```

O crosstool-ng vem com algumas configurações de toolchain por padrão, que podem ser visualizadas com o comando abaixo:

```
$ ct-ng list-samples
```

Você pode se aventurar a criar uma configuração do zero, porém o mais comum é utilizar uma das configurações pré-definidas mais próximas das características do seu hardware (arquitetura da CPU, variante da arquitetura, suporte à ponto flutuante, suporte à biblioteca do sistema, etc). Depois, se necessário, você pode abrir o menu de configuração do crosstoolng e customizar o toolchain. Execute então o comando abaixo para carregar a configuração de toolchain mais próxima das características de hardware do target:

```
$ ct-ng arm-cortex_a8-linux-gnueabi
```

Vamos agora abrir o menu de configuração do crosstoolng e fazer alguns ajustes:

```
$ ct-ng menuconfig
```

Entre na opção “Paths and misc options” e configure o diretório de instalação do toolchain:

1) Altere o nível máximo de log para ver DEBUG (procure LOG_DEBUG na interface, usando a tecla /) para que possamos ter mais detalhes sobre o que aconteceu durante a compilação caso algo dê errado.

2) (/home/XXX/labs/02/toolchain) Prefix directory

Na opção Toolchain:

1) Defina a string do fornecedor (TARGET_VENDOR) para treinamento.

2) Defina o alias (TARGET_ALIAS) para arm-linux. Desta forma, poderemos usar o compilador como arm-linux-gcc em vez do arm-training linux-uclibcgnueabihf-gcc, que é muito mais longo para digitar.

Em C-Library:

1) Se ainda não estiver definido, defina a biblioteca C como uClibc (LIBC_UCLIBC).

2) Mantenha a versão padrão que é proposta.

3) Se necessário, habilite Add support for IPv6 (LIBC_UCLIBC_IPV6).

Em C compiler:

- Certifique-se de que C++ (CC_LANG_CXX) esteja habilitado.

Salve e saia do menu de configuração. Agora é só iniciar a geração do toolchain com o comando abaixo:

```
$ ct-ng build
```

No fim do processo, verifique se o toolchain foi gerado corretamente:

```
$ ls /home/XXX/labs/02/toolchain/bin
```

3 Atividades Práticas

pratica 2:

Cross-compile uma aplicação “Hello World!” com a lib *stdio* lincado dinâmica e estaticamente, envie e execute no Linux que está rodando na BleagleBone Black.