UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA MECÂNICA
Curso de Graduação em Engenharia Mecatrônica

Sistemas Digitais para Mecatrônica
FEELT49081

Tarefa da Semana 10 - Compilação Cruzada

Prof. Éder Alves de Moura

Hericles Felipe Ferraz - 11811EMT022

Uberlândia, 5 de Março de 2022

1. Faça um resumo do processo de compilação cruzada descrito no vídeo

A compilação cruzada é o ato de compilar código para um sistema de computador (geralmente conhecido como destino) em um sistema diferente, chamado host. É uma técnica muito útil, por exemplo, quando o sistema de destino é muito pequeno para hospedar o compilador e todos os arquivos relevantes.

Como funciona a compilação cruzada?

Um compilador cruzado é um compilador capaz de criar código executável para uma plataforma diferente daquela em que o compilador está sendo executado. Na paravirtualização, um computador executa vários sistemas operacionais e um compilador cruzado pode gerar um executável para cada um deles a partir de uma fonte principal.

Onde o compilador cruzado é usado?

O compilador cruzado é usado no Bootstrapping. Explicação: Bootstrapping para uma nova plataforma. Quando o software é desenvolvido para uma nova plataforma, um compilador cruzado é usado para compilar as ferramentas necessárias, como o sistema operacional e um compilador nativo.

O que é compilação cruzada para ARM?

Um compilador cruzado é aquele que compila binários para arquiteturas diferentes da sua, como compilar binários ARM em um processador x86 da Intel. Um “compilador cruzado” é executado em um ambiente e gera código para outro. Um “compilador nativo” gera código para seu próprio ambiente de execução.

Por que a compilação cruzada pode ser complexa?

Construir um compilador cruzado é significativamente mais difícil do que construir um compilador que tem como alvo a plataforma em que é executado. O problema existe devido à forma como as bibliotecas são construídas e acessadas. Na situação normal, todas as bibliotecas estão localizadas em um local específico e são usadas por todos os aplicativos desse sistema.

Quais são as fases do compilador?

- Principais fases do compilador
- Análise Lexical.
- Análise sintática (ou seja, análise)
- Geração de código intermediário (e análise semântica)
- Otimização (opcional)
- Geração de Código.

Quais são os diferentes tipos de compilador?

- Compiladores de passagem única.
- Compiladores de duas passagens.
- Compiladores Multipass.
- Qual é o processo de compilação?

A compilação é um processo de conversão do código fonte em código objeto. O processo de compilação pode ser dividido em quatro etapas, ou seja, Pré-processamento, Compilação, Montagem e Linking. O pré-processador recebe o código-fonte como entrada e remove todos os comentários do código-fonte.

2. Faça um resumo do processo de compilação cruzada descrito no vídeo

Instalação do crosstool

```
$> git clone https://github.com/crosstool-ng/crosstool-ng.git
$> cd crosstool-ng
$> git checkout crosstool-ng-[latest_version]
#
# Check the dependencies for your build system.
#
$> ./bootstrap
$> ./configure --enable-local
$> make
$> make install
```

Como usar o crosstool-NG para construir para o emulador QEMU

```
$> ./ct-ng distclean
# List all available configurations
$> ./ct-ng list-samples
# Choose the correct configuration file. Here the one for QEMU is chosen
$> ./ct-ng arm-unknown-linux-gnueabi
# In "Paths and misc options" untick "Render the toolchain read-only"
$> ./ct-ng menuconfig
$> ./ct-ng build
$>export PATH=${HOME}/x-tools/arm-unknown-linux-gnueabi/bin/:$PATH
```

Como criar um executável com static libraries

```
*-gcc -c mystaticlib1.c
*-gcc -c mystaticlib2.c
*-ar rc libmystaticlib.a mystaticlib1.o mystaticlib2.o
*-gcc myprog.c -lmystaticlib -I../usr/include -L../libs -o myprog
```

Como criar um executável com dinamic libraries

```
*-gcc -fPIC -c mydynlib1.c
*-gcc -fPIC -c mydynlib2.c
*-gcc -shared -o libmydynlib.so mydynlib1.o mydynlib2.o
*-gcc myprog.c -lmydynlib -I../usr/include -L../libs -o myprog
```