

Instalando o cross compilador

Um cross compilador nada mais é do que um compilador para uma plataforma diferente do computador usado. Como por exemplo um computador com a arquitetura x86 compilando para uma placa ARM. O cross compilador que será usado será o arm-none-eabi.

Baixando o arm-none-eabi

O pacote com o arm-none-eabi pode ser encontrado no site da GNU e no site da própria ARM. Acesse esse link <https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads> e baixe diretamente do site da ARM.

Ou use os seguintes comandos para fazer o download.

```
$ mkdir Labs  
$ cd Labs/  
$ wget https://bit.ly/2KYUqcF -O compiler.tar.bz2
```

Após baixado o cross compilador, será preciso criar uma pasta na pasta da disciplina, nesse caso siga os seguintes passos no terminal:

```
$ mkdir compilador
```

Após a criação da pasta é preciso cópiar o arquivo baixado para dentro da pasta compilador/, então para a copia do arquivo, siga os seguintes passos:

```
$ cp compiler.tar.bz2 compilador/  
$ cd embarcados/Labs/compilador
```

Onde o primeiro parametro é o nome do arquivo (você tem que tá no diretório do arquivo) e compilador/ é o caminho da pasta criada anteriormente (o caminho da sua pasta pode ser diferente). Veja que no meu caso baixei o compilador na pasta labs. Já copiado, agora dentro da pasta compilador/ é preciso descompactar o arquivo, digite o seguinte comando para isso:

```
$ tar -xf compiler.tar.bz2
```

Lembre-se de verificar o nome do arquivo antes de copiar e descompactar, pois obviamente se tiver um nome diferente, o linux não vai encontrar!!!

Com isso, já temos o compilador, porém para que ele seja acessado de qualquer parte do computador é preciso que seja mudada uma variavel de ambiente chamada \$PATH, para isso é preciso modificar o arquivo .bashrc que está no diretório do usuário, siga os seguintes passos para a modificação da variavel.

```
$ cd ~  
$ gedit .bashrc
```

Onde a pasta "~" é onde está todas as informações do usuário e gedit é um editor de textos (pode ser usado qualquer outro). Após abrir esse arquivo é preciso adicionar a seguinte linha no final do arquivo:

```
PATH=$PATH:~/UFC/Monitoria/embarcados2/Labs/compilador/gcc-arm-none-eabi-7-2018-q2-update/bin
```

Veja que ~/UFC/Monitoria/embarcados2/Labs/compilador/gcc-arm-none-eabi-7-2018-q2-update/bin é o caminho da pasta que criei, portanto adicione o caminho da sua pasta, que pode ser visto no terminal com o comando "pwd". Caso o caminho contenha espaços, como por exemplo "Área de Trabalho", é necessário uma barra invertida (\), portanto, ficaria "Área de Trabalho". Verifique seu caminho e cole corretamente no arquivo bashrc.

Após configurar a variável \$PATH, o compilador já pode ser usado normalmente.

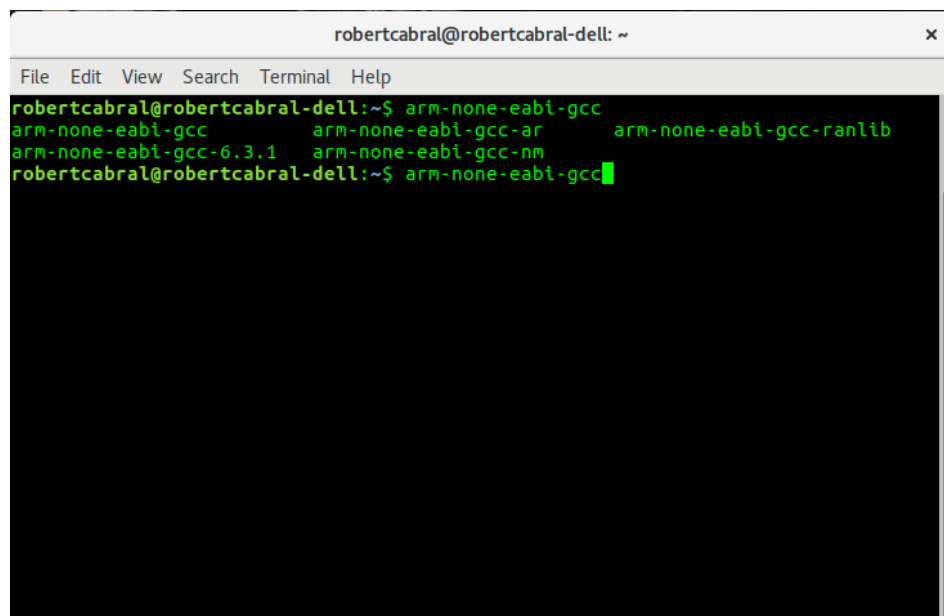


Figura 1: Compilador instalado.

Após a instalação do compilador, compile um programa qualquer e tente rodar no próprio computador, o computador não conseguirá rodar a aplicação, pois foi compilado para outra plataforma.

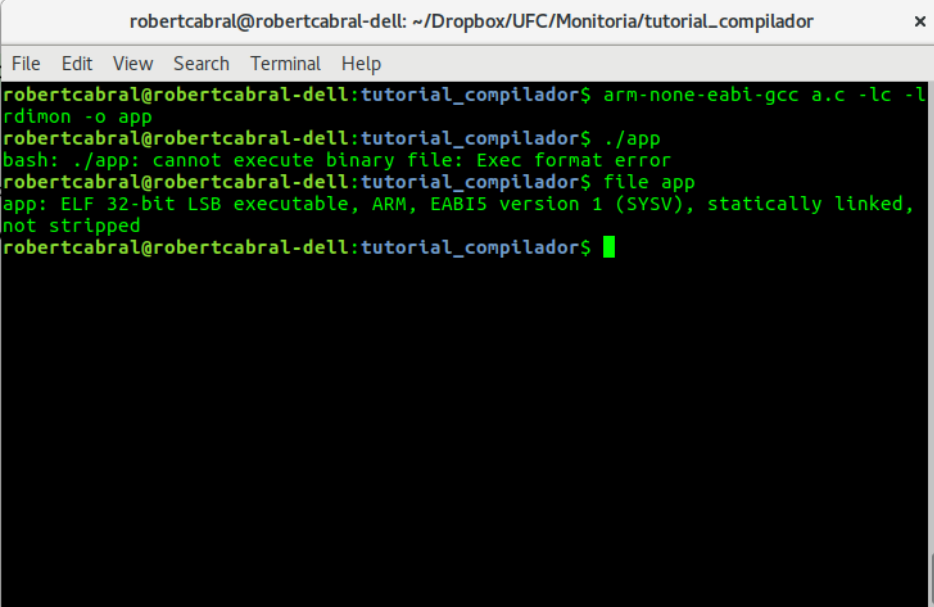
```
#include <stdio.h>

int main() {
    printf("Hello World");
    return 0;
}
```

Para compilar programas como esse, é preciso alguns parâmetros de compilação:

```
$ arm-none-eabi-gcc main.c -lc -lrdimon -o app
```

Uma vez gerado o executável, ao tentar executar, o computador não vai conseguir executar. Para saber para qual arquitetura o executável tá compilado, simplesmente use o "file" no terminal.



```
robertcabral@robertcabral-dell: ~/Dropbox/UFC/Monitoria/tutorial_compilador
File Edit View Search Terminal Help
robertcabral@robertcabral-dell:tutorial_compilador$ arm-none-eabi-gcc a.c -lc -l
rdimon -o app
robertcabral@robertcabral-dell:tutorial_compilador$ ./app
bash: ./app: cannot execute binary file: Exec format error
robertcabral@robertcabral-dell:tutorial_compilador$ file app
app: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), statically linked,
not stripped
robertcabral@robertcabral-dell:tutorial_compilador$
```

Figura 2: Tentando rodar o executavel para o ARM.