

# Tutorial básico de instalação do SESC – Superscalar Simulator

Matheus Alcântara Souza

matheusalcantarasouza@gmail.com

## 1. Informações preliminares

Este tutorial tem o objetivo de apresentar passos para instalação do simulador SESC – Superscalar Simulator, bem como apresentar noções básicas de configuração. Também é escopo deste tutorial exemplificar a execução de benchmarks no simulador e a leitura dos resultados obtidos, através da configuração de diferentes arquiteturas.

O trabalho foi feito baseando-se em uma instalação em sistema operacional Linux Ubuntu 14.04.1 LTS, versão desktop de 64 bits. O sistema operacional foi instalado em máquina virtual através do software Oracle Virtual Box. Considere que o usuário do sistema criado seja “**usuario**”.

### 1.1 Aplicativos e Bibliotecas

Inicialmente, é necessário instalar aplicativos e bibliotecas (CVS por exemplo) para posterior instalação e uso do SESC.

Acesse o terminal, e execute os seguintes comandos:

```
$ sudo apt-get install cvs build-essential flex bison zlibg-dev
```

## 2. Instalação

Para a instalação, o primeiro passo a ser realizado é o download do SESC, utilizando a ferramenta de controle de versões CVS, instalada anteriormente.

Acesse o terminal, e execute os seguintes comandos:

```
$ sudo cvs -d:pserver:anonymous@sesc.cvs.sourceforge.net:/cvsroot/sesc login
```

Será solicitada uma senha. Apenas confirme pressionando ENTER

```
$ sudo cvs -z3 -d:pserver:anonymous@sesc.cvs.sourceforge.net:/cvsroot/sesc co -P sesc
```

Com este comando, será criada uma pasta 'sesc' dentro do diretório atual. Considere, para entendimento do tutorial, a instalação no diretório **/home/usuario/sesc**.

Após o download completo do SESC, será necessário realizar a correção de algumas referências dentro dos arquivos fontes em linguagem C/C++ e para o build. Você deverá editar os arquivos, através dos comandos e instruções abaixo:

```
$ sudo gedit /home/usuario/sesc/src/libcore/FetchEngine.cpp
Adicione a linha #include <limits.h>
```

```
$ sudo gedit /home/usuario/sesc/src/libcore/Cluster.cpp
Adicione a linha #include <limits.h>
```

```
$ sudo gedit /home/usuario/sesc/src/libmint/subst.cpp
Altere a linha #include <linux/dirent.h> para #include <dirent.h>
```

```
$ sudo gedit /home/usuario/sesc/src/libsuc/Config.cpp
Adicione as linhas #include <limits.h> e #include <stdint.h>
```

```
$ sudo gedit /home/usuario/sesc/src/libsuc/conflex.y
```

Adicione, na linha 118, um ponto e vírgula (;) no final da linha, antes de fechar as chaves.

```
$ sudo gedit /home/usuario/sesc/src/Makefile.defs
```

Adicione, na linha 136, ao final, a opção '-fpermissive'

Para facilitar o trabalho, os arquivos estão disponíveis na seguinte pasta do Google Docs: [https://drive.google.com/folderview?id=0B75F\\_6H6mTTYXl0TDVXd9xWTQ&usp=sharing](https://drive.google.com/folderview?id=0B75F_6H6mTTYXl0TDVXd9xWTQ&usp=sharing) bastando copiá-los para os diretórios indicados, substituindo. Pode ser necessário fornecer permissões de administrador (use sudo).

Para compilar o SESC, vamos criar um novo diretório chamado **build**, e executar alguns comandos:

```
$ mkdir /home/usuario/build
$ cd /home/usuario/build
$ ../sesc/configure --help
```

O terceiro comando acima irá criar os arquivos para compilação do SESC baseado nas opções fornecidas. O parâmetro **--help** irá exibir essas opções. Por exemplo, a opção **--enable-smp** irá permitir a configuração e execução em SMP ou CMP. Vamos utilizar essa opção para compilar:

```
$ ../sesc/configure -enable-smp
$ make
```

Aguarde o final da configuração e arquivo **sesc.smp** será criado dentro da pasta build.

Caso ocorra erros durante a compilação, verifique se a edição dos arquivos C/C++ foi feita corretamente, e em seguida repita os passos para compilar.

### 3. Simulações

Finalizada a instalação, será possível simular aplicações utilizando o SESC. Inicialmente vamos testar o benchmark Crafty, fornecido em conjunto com o código fonte do SESC na pasta **/sesc/tests**. Para isso, execute o seguinte comando:

```
$ ../sesc.smp -c ../sesc/confs/smp.conf ../sesc/tests/crafty < ../sesc/tests/tt.in
```

O comando **../sesc.smp**, executa o binário compilado.

O parâmetro **-c**, é o caminho do arquivo de configuração **.conf**, onde estão as configurações da arquitetura a ser simulada. Na pasta **/home/usuario/sesc/confs** existem alguns modelos (por exemplo **cmp.conf** e **smp.conf**) para análise. Escolha um arquivo e copie para o diretório **build**, podendo alterá-lo conforme a arquitetura que se deseja simular. É necessário lembrar que na simulação, será necessário alterar o parâmetro **-c** para o arquivo desejado.

O restante do comando se refere à aplicação que a executar. No caso do benchmark Crafty, **../sesc/tests/crafty** é o arquivo binário já compilado para a arquitetura MIPS. O arquivo **../sesc/tests/tt.in** é uma carga de dados para esse benchmark. Cada benchmark pode ter seu conjunto de parâmetros específicos.

### 3.1. Arquivos de configuração

Estes arquivos são os mais importantes para quem deseja utilizar o SESC para simulações. São os arquivos que irão orientar o SESC sobre qual arquitetura simular.

Anteriormente, fizemos a cópia de um dos arquivos no diretório **conf** para o diretório **build** (exemplo **smp.conf**). Para editar este arquivo, utilize um editor de texto comum. Vamos executar o comando:

```
$ gedit /home/usuario/builds/smp.conf
```

Os arquivos normalmente contém uma lista de parâmetros que serão lidos pelo SESC para simular arquitetura do processador. É possível alterar informações como quantidade de núcleos, clock, tamanho de cache, políticas, etc.

Recomenda-se a leitura dos demais arquivos e exemplos disponíveis na Internet, bem como alterações diversas nos parâmetros, para melhor entendimento das possibilidades de arquiteturas para simular.

### 3.2. Benchmark SPLASH-2

Um conhecido pacote de benchmarks para processadores é o SPLASH-2. Vamos utilizar uma versão pré-compilada para a arquitetura MIPS para alguns testes.

Crie a pasta **/home/usuario/build/splash**:

```
$ mkdir /home/usuario/build/splash
```

Faça o download do arquivo que contém o benchmark SPLASH-2 pré-compilado. O arquivo possui cerca de 2MB. Após o download, descompacte na pasta criada anteriormente.

Acesse [https://drive.google.com/folderview?id=0B75F\\_6H6mTTYXl0TDVXdM9xWTQ&usp=sharing](https://drive.google.com/folderview?id=0B75F_6H6mTTYXl0TDVXdM9xWTQ&usp=sharing) e faça o download do arquivo **SPLASH\_MIPS.zip**. Descompacte na pasta **/home/usuario/build/splash**

Para melhor desenvolvimento do tutorial e dos testes, vamos copiar os arquivos binários dos benchmarks:

```
$ cp thethem-mips-benchmarks-2a99416eed1e/splash-mips/* ./
$ cd /home/usuario/build
```

Feito isso, no diretório benchmark criado anteriormente, estarão alguns arquivos com extensão **.mips**, que são os binários de benchmarks do pacote SPLASH-2, pré-compilados para a arquitetura MIPS.

Abaixo 2 exemplos, sendo 1 para o benchmark FFT e outro para o benchmark Ocean com dados contíguos:

```
$ ./sesc.smp -c smp.conf ../benchmarks/fft.mips --help
$ ./sesc.smp -c smp.conf ../benchmarks/fft.mips -p2 -m4
$ ./sesc.smp -c smp.conf ../benchmarks/fft.mips -p4 -m8

$ ./sesc.smp -c ../sesc/confs/smp.conf ../benchmarks/ocean-con.mips --help
$ ./sesc.smp -c ../sesc/confs/smp.conf ../benchmarks/ocean-con.mips -p2 -n10
$ ./sesc.smp -c ../sesc/confs/smp.conf ../benchmarks/ocean-con.mips -p1 -n6
```

O parâmetro **--help** irá exibir as opções de entrada do benchmark, que podem variar. Nos exemplos acima, o parâmetro **-p** representa o número de processadores utilizados na

execução do benchmark. Mas é importante lembrar que neste momento não estamos definindo a arquitetura, mas sim como o benchmark irá utilizá-la. Sendo assim, poderíamos ter um arquivo **conf** com 16 processadores, e nas linhas de comando acima, executar com no máximo 4 processadores.

Alguns benchmarks necessitam de um arquivo de entrada de dados, como o Crafty. Caso escolha, para seu trabalho, algum benchmark nesse estilo, será necessário pesquisar o referido arquivo com a carga de dados desejada.

### 3.3. Leitura dos resultados

Após cada execução de uma simulação, um arquivo no padrão **sesc\_benchmark.abcde** será gerado, sendo **abcde** uma extensão aleatória. Este arquivo está em formato texto, e pode ser visualizado com um editor de texto comum. Entretanto, não é de fácil interpretação.

Para melhor visualização dos resultados, o SESC disponibiliza um script Perl de formatação. Execute o seguinte comando:

```
$ ../sesc/scripts/report.pl sesc_benchmark.abcde
```

Lembre-se que o nome do arquivo pode variar conforme o benchmark, e que a extensão sempre será diferente.

Se desejar ver o relatório da última execução diretamente, troque o nome do arquivo pelo parâmetro **-last**:

```
$ ../sesc/scripts/report.pl -last
```

### 3.4. Erro ao executar com mais de 4 processadores

Caso você deseje executar os benchmarks com mais de 4 processadores em sua configuração, poderá ser apresentado um erro.

Se isso acontecer, troque o arquivo **/home/usuario/sesc/src/libmint/subst.cpp** pelo disponível neste link:

[https://drive.google.com/folderview?id=0B75F\\_6H6mTTYXl0TDVXdm9xWTQ&usp=sharing](https://drive.google.com/folderview?id=0B75F_6H6mTTYXl0TDVXdm9xWTQ&usp=sharing)

## 4. Considerações finais

Espero que o tutorial tenha colaborado com a instalação e execução do SESC em seu trabalho. Em caso de dúvidas no conteúdo aqui escrito, não hesite em me comunicar por e-mail. Boa sorte e bons estudos!

### Algumas Referências

<http://www.oracle.com/technetwork/server-storage/virtualbox/downloads/index.html>

<http://releases.ubuntu.com/precise/ubuntu-12.04.3-desktop-i386.iso>

<http://sesc.sourceforge.com/index.html>

<http://khawajahashim.blogspot.com.br/2011/11/sesc-simulator-first-time-installation.html>

<https://bitbucket.org/thethem/mips-benchmarks>

[http://www.ann.ece.ufl.edu/courses/eel6935\\_11fal/EEL6935\\_SESC\\_F11.pdf](http://www.ann.ece.ufl.edu/courses/eel6935_11fal/EEL6935_SESC_F11.pdf)

S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations", *Proc. Int. Symp. Computer Architecture*, pp.24 -36 1995