

Roteiro da aula de Laboratório 01:  
**Projeto e teste utilizando SystemC**

## SystemC

*SystemC is a C++ library developed to support the modeling of hardware and software modules at the system level. The whole library is written in ISO/ANSI compliant C++ and therefore can be compiled with all standard compliant C++ compilers. It constitutes a domain specific language embodied in the library's data types and methods and can also profit of the object oriented standard constructs, which allows a larger flexibility in the modeling of embedded software and hardware modules through templates and inheritance features. The SystemC core language is built around an event-driven simulation kernel, which allows efficient simulation of compiled SystemC models. SystemC is an IEEE 1666-2005 standard and is further developed by the Open SystemC Initiative (OSCI).*

*The core language is constituted of abstract elements, like events, processes, modules, ports, interfaces and channels. In the following, structural, functional and communication aspects of SystemC are briefly defined:*

- The design structure is defined by means of modules (**sc\_module**) in SystemC. It may contain both functional description as well as further modules like in a hierarchical design.*
- The design functionality is specified by means of processes. SystemC has two types of process: methods (**sc\_method**) and threads (**sc\_thread**, **sc\_thead**). Methods are used to describe hardware at register transfer level (RTL) and its control flow cannot be suspended during the execution. Threads are used in the modeling of both software and hardware models at higher level of abstractions and they can be suspended by the **wait()** function. C++ language as well as SystemC specific data types can be used, such as **sc\_bit**, **sc\_int**, **sc\_bigint**, **sc\_bv**, **sc\_int<N>**, for instance, represents signed two's complement integers and *N* is the number of bits of an integer variable. **sc\_event** implements an event that allows to wake up a suspended process (i.e., thread) by means of the **notify()** function, which implements the immediate notification for the process.*
- The design communication is defined by means of **port**, **interface** and **channel** concepts, which are used specially in the modeling at system level.*

## Virtual-Box

Todas as ferramentas necessárias para desenvolver os projetos utilizando SystemC foram instalados em uma máquina virtual, como apresentado na Fig. 1.

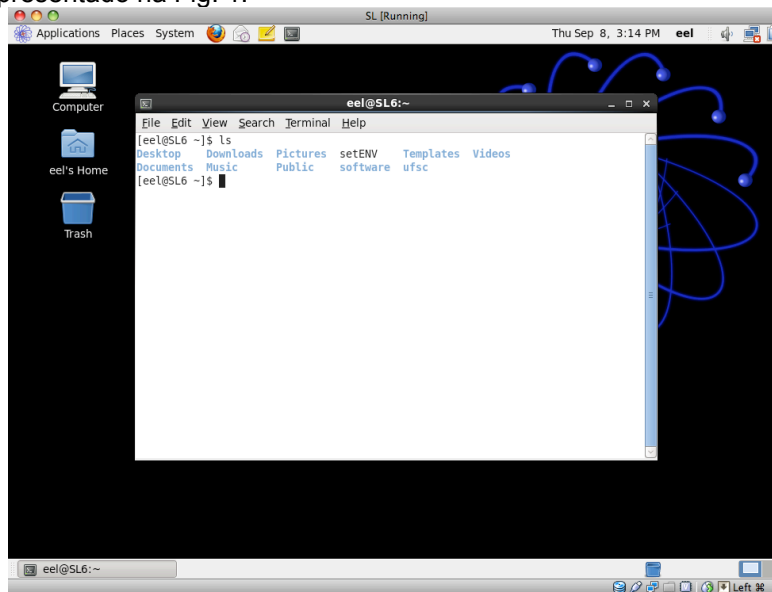




Fig 1: Janela principal da máquina virtual

Os seguintes passos deve ser realizados para a elaboração do Lab1:

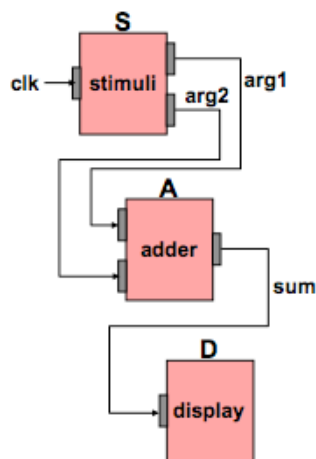
- 1) Abrir um terminal. 
- 2) \$ source setENV
- 3) \$ cd ufsc
- 4) \$ mkdir lab1
- 5) Ativar a internet na máquina virtual .
- 6) Abrir o navegador Firefox e acessar o endereço: <http://dlettin.paginas.ufsc.br>
- 7) Baixar os *slides* introdutórios da linguagem SystemC apresentados na última aula
- 8) Baixar o projeto. Pasta padrão do Firefox é **Downloads**
  - a) mv projeto.tgz.doc projeto.tgz
  - b) mv projeto.tgz ~/ufsc/.
  - c) cd ~/ufsc/
  - d) tar -zxvf projeto.tgz

### Passos para o projeto

Para os exercícios deste roteiro você deverá efetuar os seguintes passos.

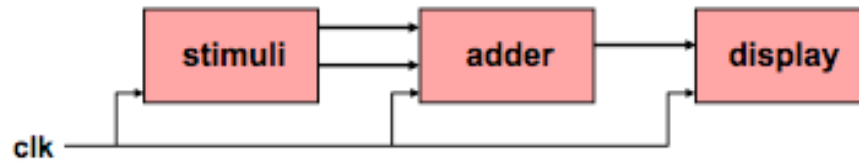
1. Editar os arquivos fontes utilizando um editor, por exemplo **kate**, **emacs**, etc.
2. Após a alteração o arquivo deve ser salvo
3. O projeto deve ser compilado utilizando o seguinte comando: \$ make all
4. (Opcional) Um arquivo no formato **vcd** pode ser criado para apresentar o conteúdo dos sinais em forma de onda
  - i) sc\_trace\_file \*tf;
  - ii) tf = sc\_create\_vcd\_trace\_file("trace");
  - iii) sc\_trace(tf, verif\_env.clk\_sig, "clk\_sig");
  - iv) Utilize o programa **gtkwave** para abrir o arquivo após a simulação

### Exercício 1: Projete e teste o seguinte sistema



Responda as Questões 1 e 2 do relatório.

**Exercício 2:** Faça uma copia do projeto anterior e projete / teste o seguinte sistema



Responda as Questões 3 e 4 do relatório.

**Exercício 3:** Adicione um segundo processo dentro do modulo “Adder” que deve ser responsável pelo incremento. Ou seja, um processo irá realizar a soma e o incremento + 1. A sincronização entre os processos deve ser implementada por meio de `sc_events`.

Responda a Questão 5 do relatório.