Evan
An IDE for SystemC

Mazdak & Alborz Design Automation

http://www.mazdak-alborz.com

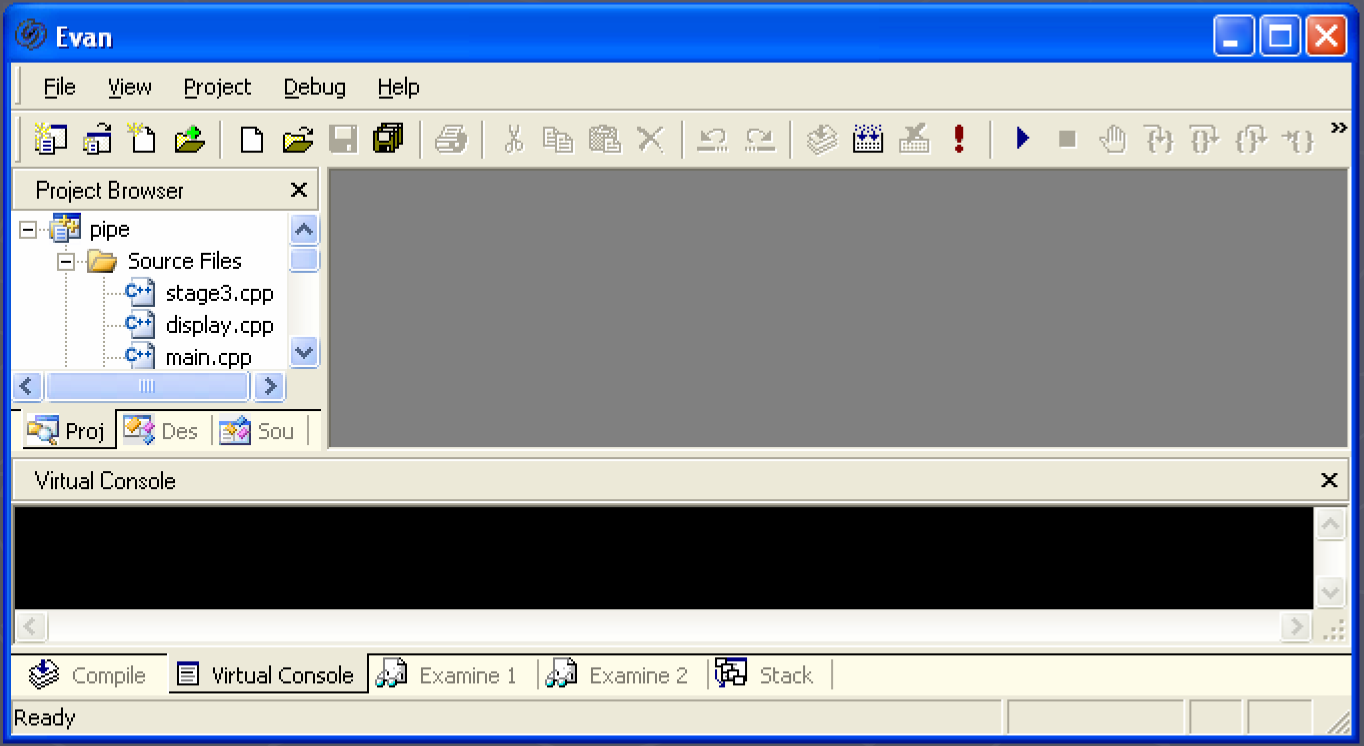
Evan 2007.04

Features

- Supports SystemC project creation/management
- Supports OSCI reference libraries including SystemC 2.1, SCV and TLM
- Creates SystemC executable models
- Provides simulation of SystemC models in context of Evan IDE through a virtual console
- Provides true SystemC debug facilities including breakpoint insertion, value examining and ...
- Provides waveform viewer
- Extracts design source details and design hierarchy
- Supports Mentor AVM for verification

Project Management

- Evan provides IDE style SystemC project creation and management.

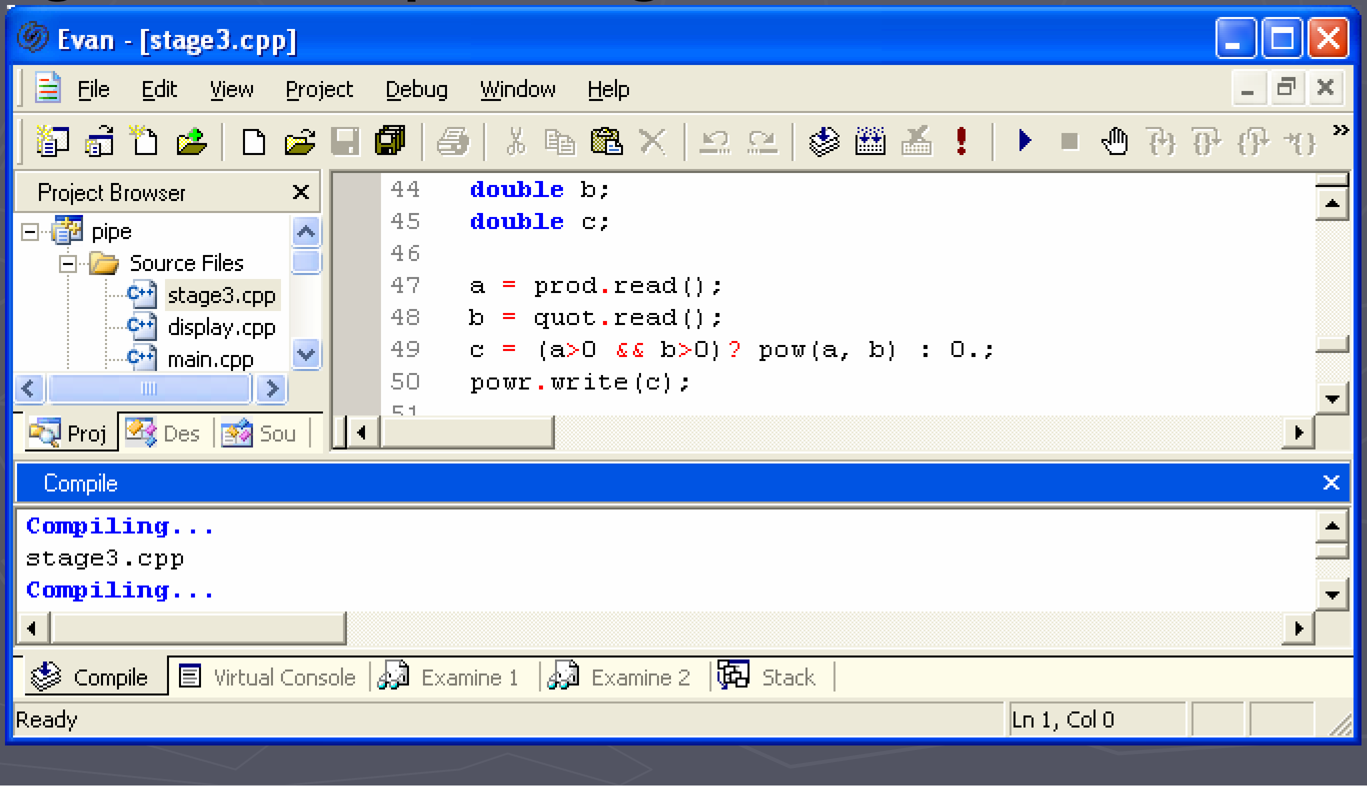


OSCI Libraries

- Evan ships with all of standard open source libraries of OSCI. All examples of these libraries also exist.
- Current release of Evan supports the following libraries
  - SystemC 2.1
  - SCV 1.0p2
  - TLM 2005-04-08

SystemC Executable Model

- Evan provides Compile and Build commands for creating SystemC executable models by using MinGW package.



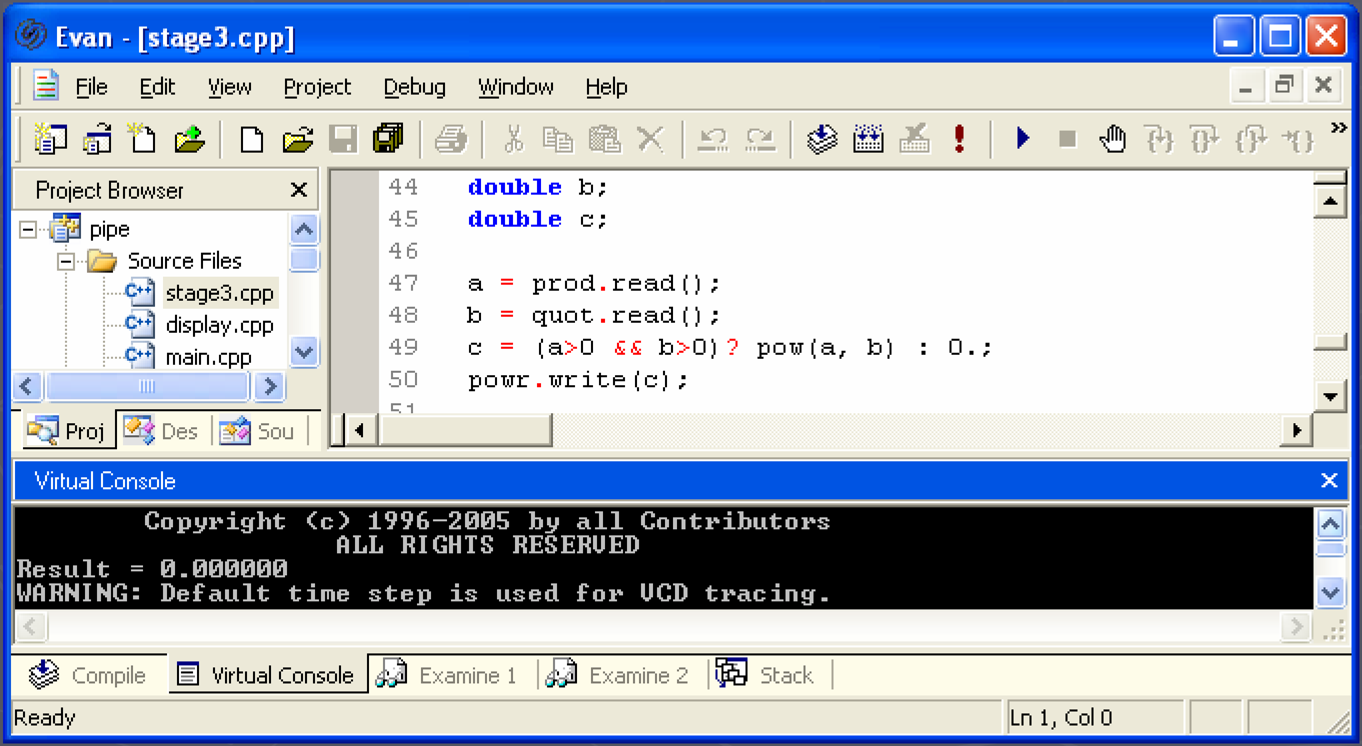
```
double b;
double c;

a = prod.read();
b = quot.read();
c = (a>0 && b>0) ? pow(a, b) : 0.;
powr.write(c);
```

```
Compiling...
stage3.cpp
Compiling...
```

SystemC Simulation

- SystemC models can be easily simulated in Evan platform. Simulation results appear in a Virtual Console.



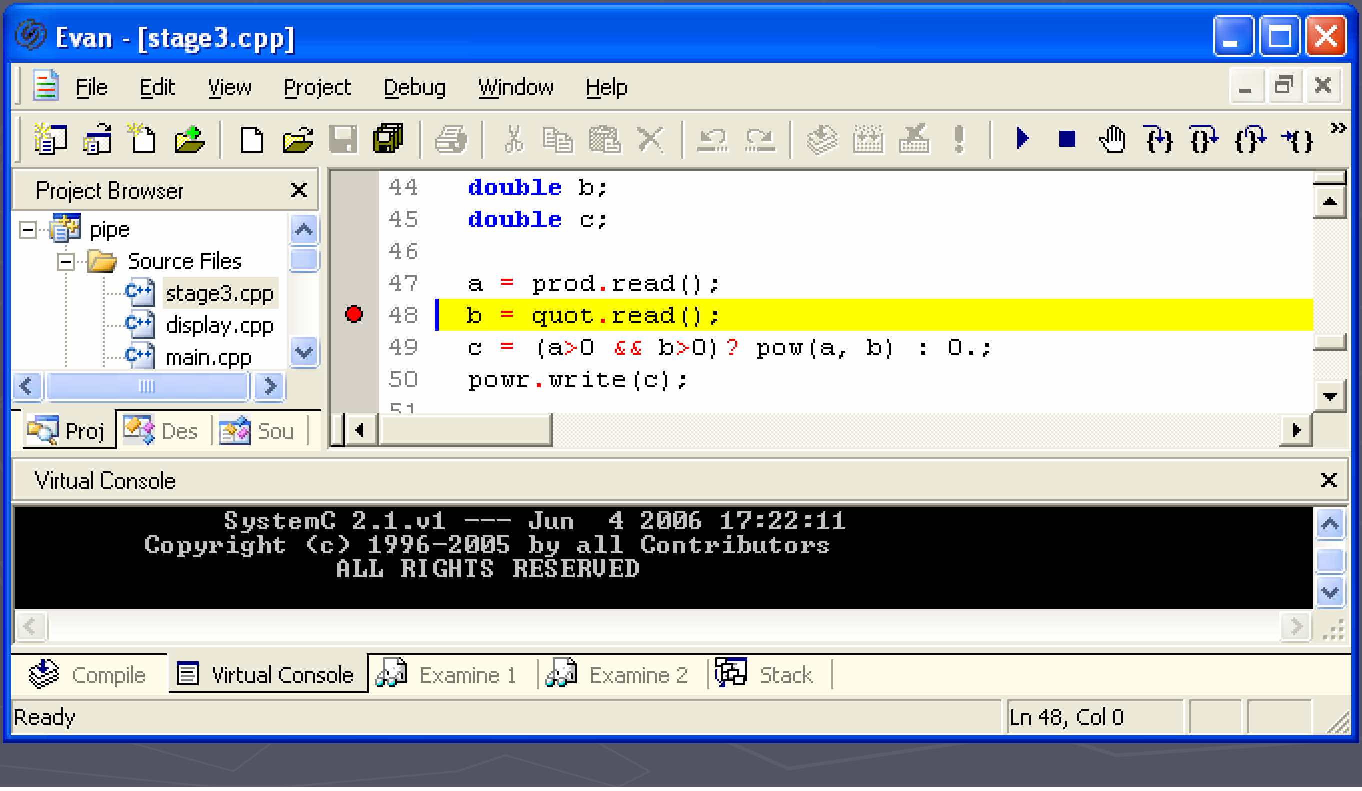
```
double b;
double c;

a = prod.read();
b = quot.read();
c = (a>0 && b>0)? pow(a, b) : 0.;
powr.write(c);
```

```
          Copyright (c) 1996-2005 by all Contributors
                          ALL RIGHTS RESERVED
Result = 0.000000
WARNING: Default time step is used for VCD tracing.
```

SystemC-Aware Debugger

- Evan provides all of the required SystemC debugging capabilities such as break point insertion, trace, examine and etc.



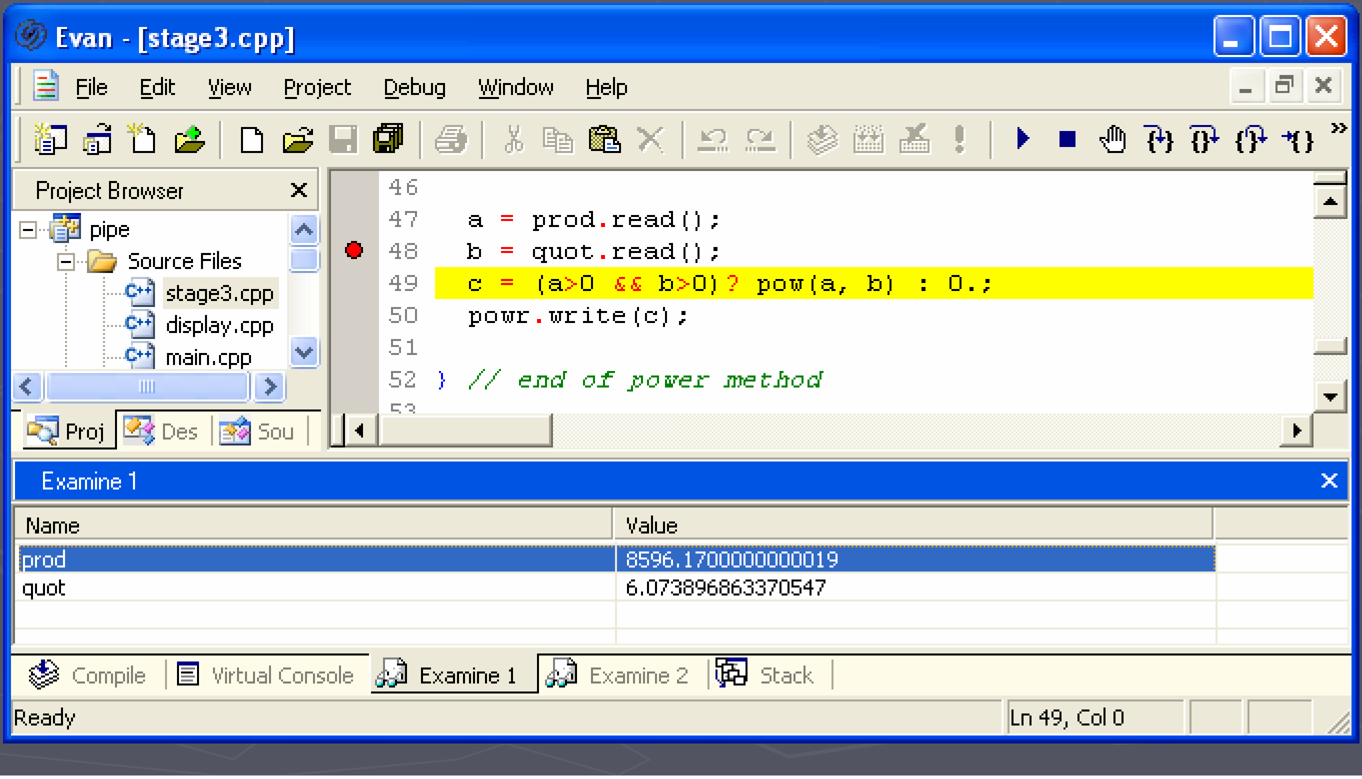
```
double b;
double c;

a = prod.read();
b = quot.read();
c = (a>0 && b>0)? pow(a, b) : 0.;
powr.write(c);
```

```
        SystemC 2.1.v1 --- Jun  4 2006 17:22:11
   Copyright (c) 1996-2005 by all Contributors
               ALL RIGHTS RESERVED
```

SystemC-Aware Debugger ‘Cont

- In Examine page user can check SystemC variables values. Evan extracts true values of signals and ports during simulation.



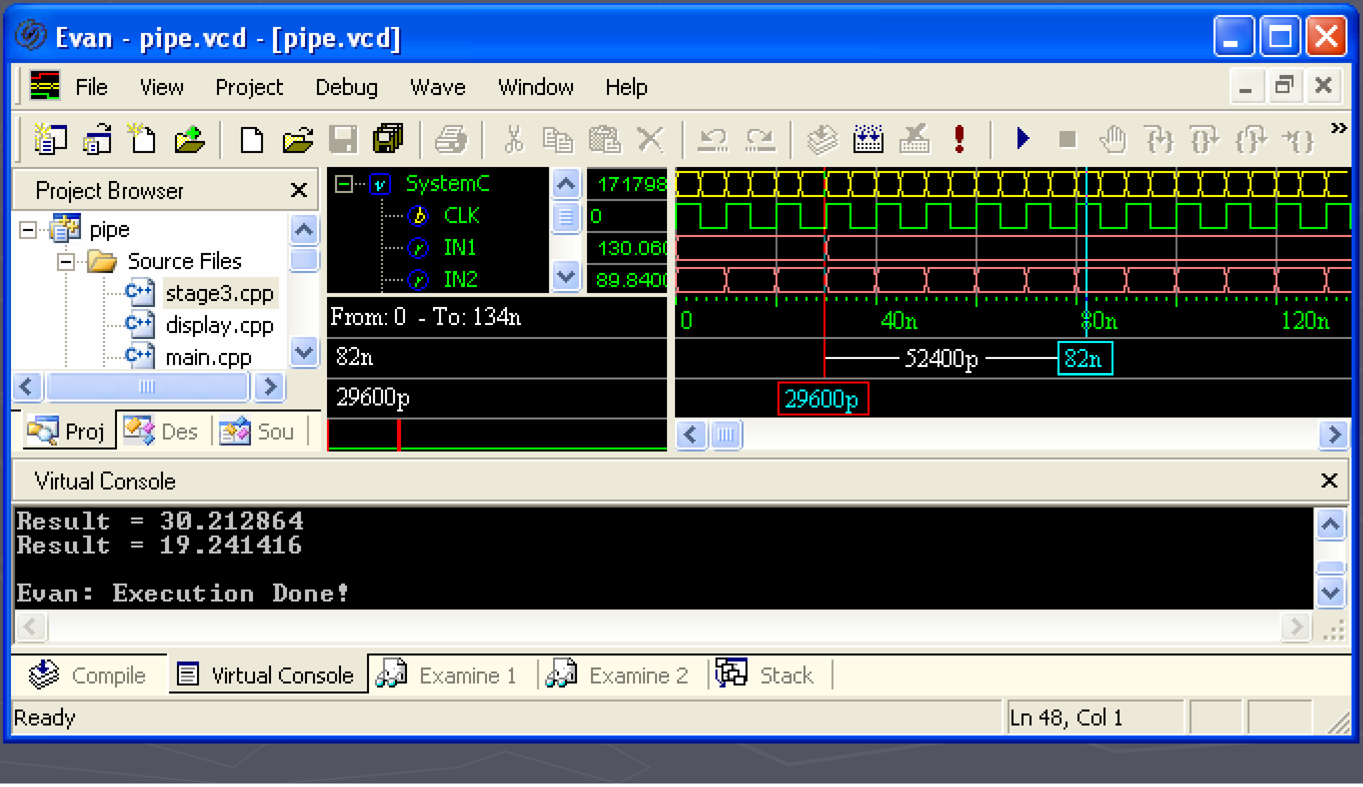
```
  a = prod.read();
  b = quot.read();
  c = (a>0 && b>0)? pow(a, b) : 0.;
  powr.write(c);

} // end of power method
```

Name	Value
prod	8596.1700000000019
quot	6.073896863370547

Waveform Viewer

- Evan is equipped with a powerful waveform viewer with zoom, cursor and bookmark features.

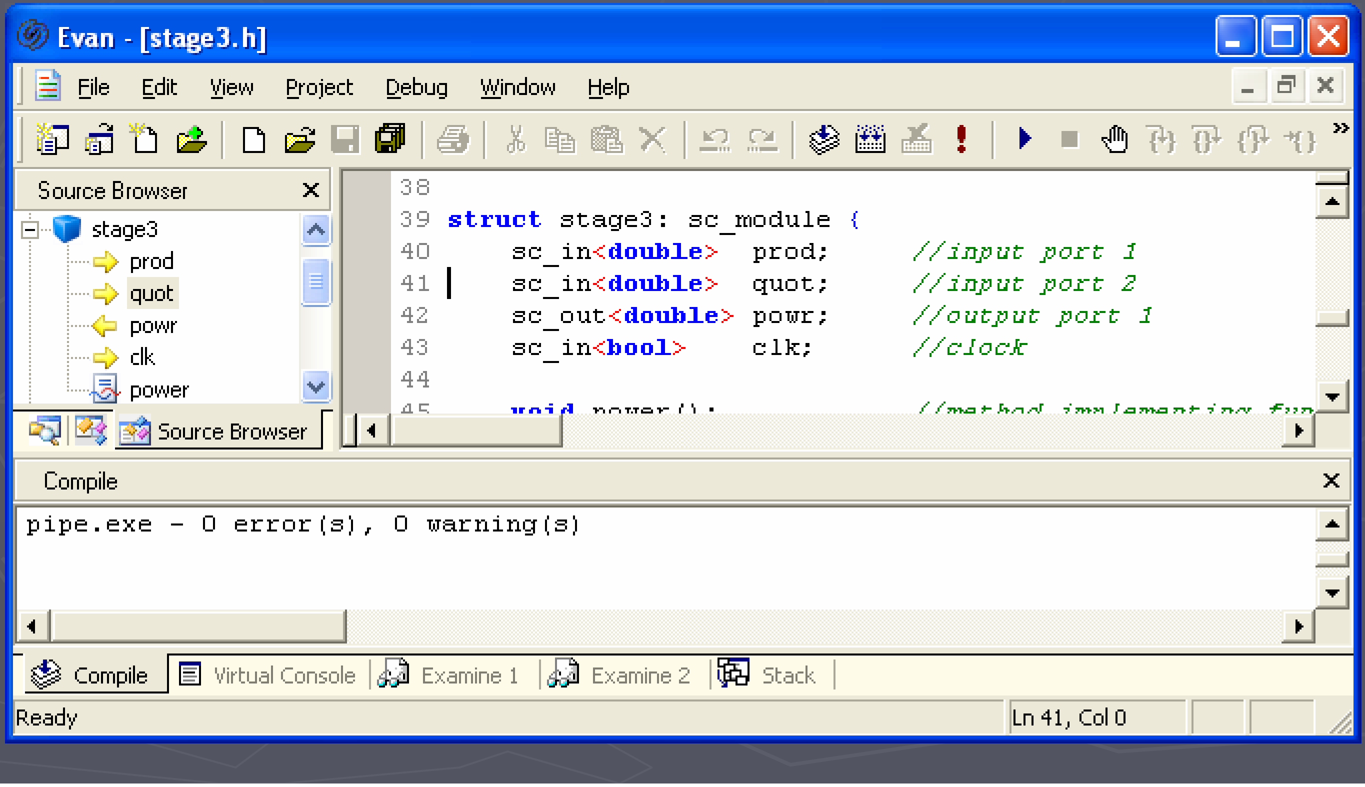


```
Result = 30.212864
Result = 19.241416

Evan: Execution Done!
```

Design Details

- Evan extracts details of SystemC models including signals, ports, processes, etc. and shows them in Source Browser page.



```
struct stage3: sc_module {
    sc_in<double>  prod;     //input port 1
    sc_in<double>  quot;     //input port 2
    sc_out<double> powr;     //output port 1
    sc_in<bool>    clk;      //clock

    void power();            //method implementing fun
```

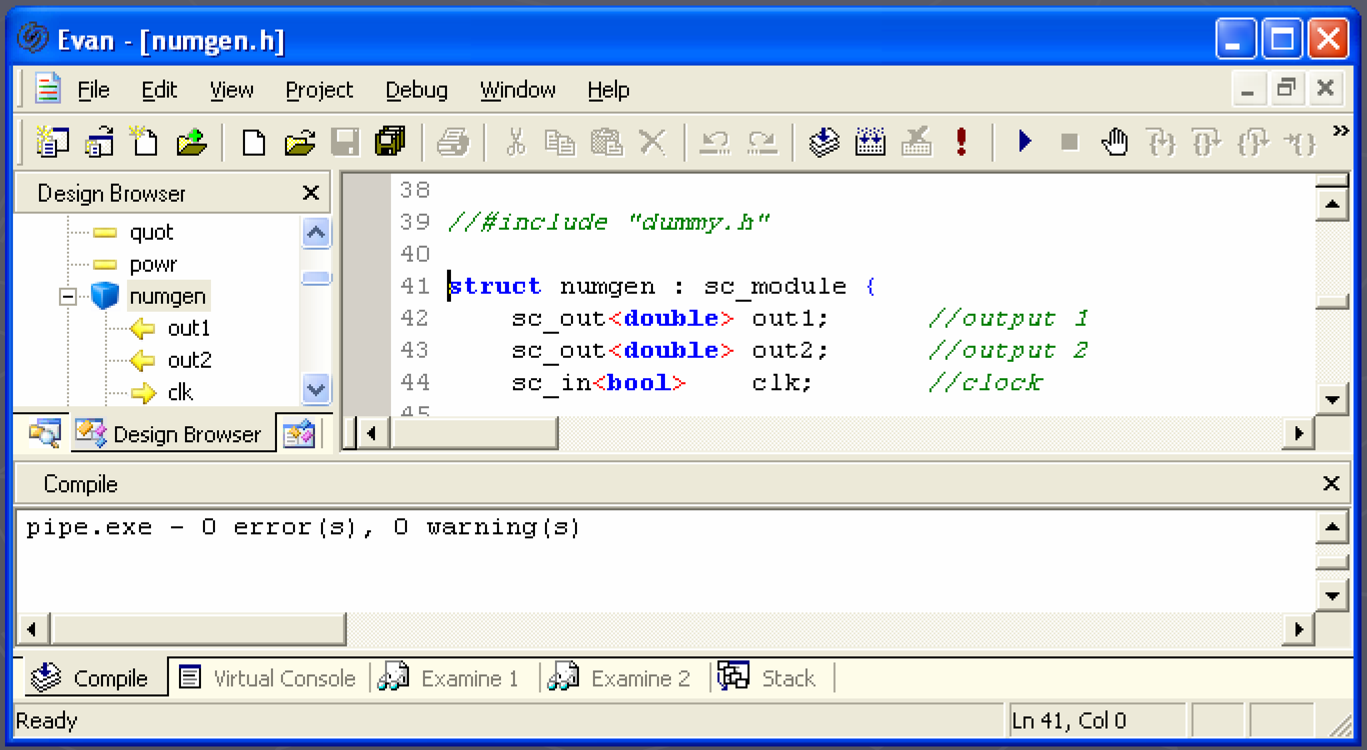
```
pipe.exe - 0 error(s), 0 warning(s)
```

Ready

Ln 41, Col 0

Design Hierarchy

- Evan extracts design hierarchy and shows it at Design Browser page.



```
//#include "dummy.h"

struct numgen : sc_module {
    sc_out<double> out1;      //output 1
    sc_out<double> out2;      //output 2
    sc_in<bool>    clk;       //clock
```

```
pipe.exe - 0 error(s), 0 warning(s)
```

AVM for verification

- Evan is adapted with Advanced Verification Methodology (AVM) from Mentor Graphics. All SystemC examples of this package are shipped with Evan.
- Current version of Evan supports AVM 2.0