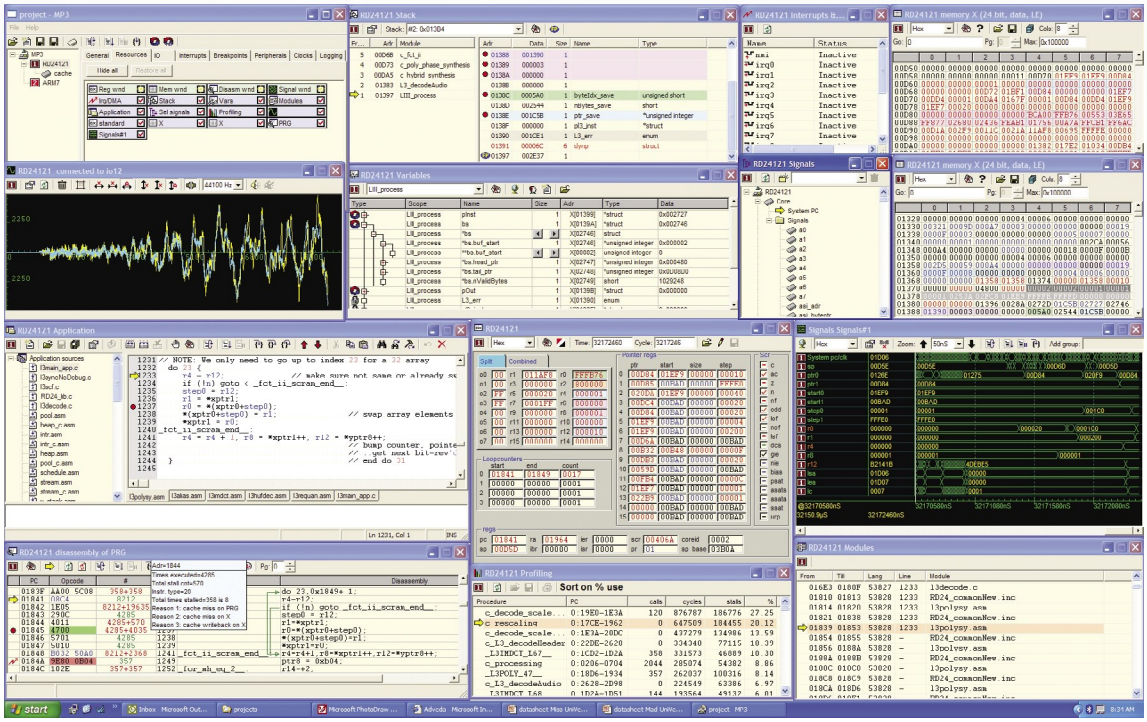
UNIVERS ISSIM

SW debugging with multiple different Instruction Set Simulators!

GENERAL DESCRIPTION

Univers® ISSIM is a complete Integrated Development Environment (IDE) for multi-processor SOC architectures, offering extensive debug capabilities, fast simulation and hardware target support. It is a multi-core IDE with most fascinating debug features and a configurable user-friendly graphical user interface. Adveda also provides fast cycle-accurate processor Instruction Set Simulators (ISS), which are the perfect match within this solution. Univers ISSIM is the Integrated System SW simulation tool of Adveda's UNIfied VERification Solutions.



Univers ISSIM's Integrated Development Environment offers the Software Engineer one central cockpit from which he can perform all his code generation and code verification. Univers ISSIM is a multi-processor debugger, which can handle most processors, accompanying tool chains and underlying targets, providing full debug capabilities. Here is a short summary of the main features:

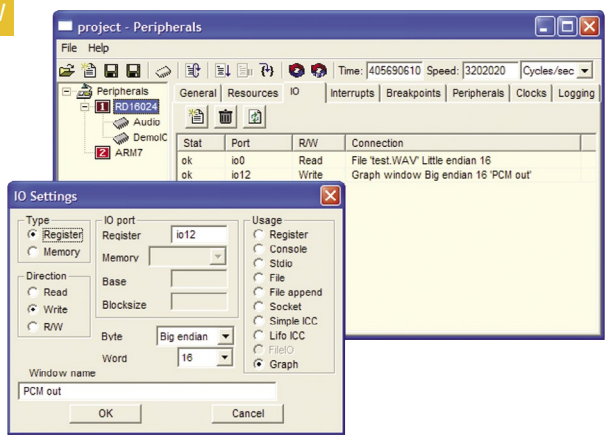
- Fast cycle-accurate Instruction Set Simulators
- Combines multiple different ISS's
- Complete Integrated Development Environment
- Automated flow for multi-core SOC projects
- Ultra Fast Verilog/VHDL simulation
- Extensive and unique debug features
- Graphical static and dynamic profiling
- Open debug-API for integration
- Automatic embedded OS support

UNIVERS FAMILY

Meet the family

Univers ISSIM is part of Adveda's Univers family. Univers RTSIM is a standalone RTL simulator with the same graphical user interface. Univers COVER is the combination of Univers RTSIM and Univers ISSIM and delivers a complete system-level cycle-accurate simulation of both hardware and software at an unsurpassed speed.
Univers ISSIM provides a multi-window interface with which developers can activate desired functionality and select system visibility as the debugging requirements evolve. It also integrates the development tools (C-compiler, assembler, linker) for each processor and manages the flow between these tools. Central to all debug sessions are the Project Window and the Application Window, which represents the application in source code, either in C-code and/or assembly language. The other windows are specifically designed to ease the complex debugging tasks faced by today's embedded software developers. All usual run-control functions are provided through menus and icon-buttons within the context of different windows. All settings of a project are saved in a XML-formatted project file.

PROJECT WINDOW

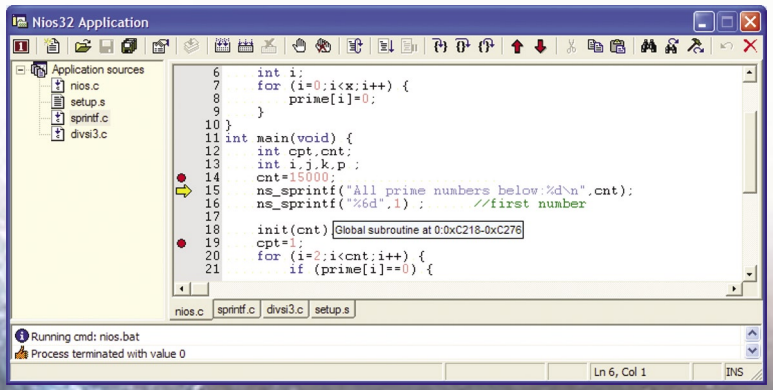


The Project Window instantiates the project and controls the overall debug process. Every processor core in the project is instantiated in the Project Window and has its own list of features and relevant debugging windows. These windows may be managed from the Project Window. The Project Window also contains a logging field as well as a command prompt. Furthermore, clocks, IO, DMA, breakpoint settings as well as inter-processor communication can be setup via this window.

APPLICATION WINDOW

Each processor has an Application Window, which contains the software source codes of the project for that processor. The object code can be built with all related files or can be individually compiled and assembled, all under menu or button control.
In the Application Window, the source code can be viewed, edited and (re-) compiled.
The source-code is context-sensitively colored for ease of reading. For easy debugging, the values of variables pop-up when the mouse is moved over the associated source code text.

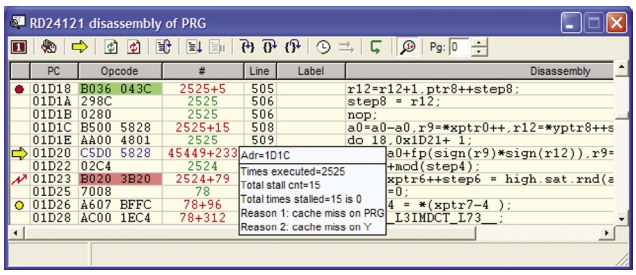
The Application Window also allows a walkthrough a function's backtrace, while simultaneously updating the Variables Window, Stack Window and the Application Window itself, thus facilitating easy tracing of function calls. Obviously, breakpoints in your source-code text can be placed as well.



Fast Feature-rich debugging

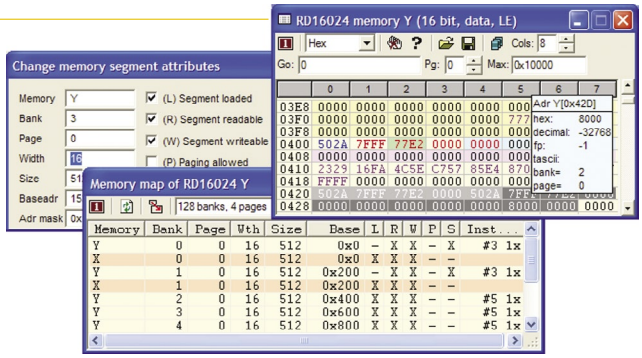
DISASSEMBLY WINDOW

Every processor has its own Disassembly Window containing a lot of information about the execution of the program. Besides the disassembled code, it also provides information how many times instructions are executed, how many stalls are introduced by each instruction and which effect caused the stall. This provides all basic information, which can be used to improve the overall performance of the application.



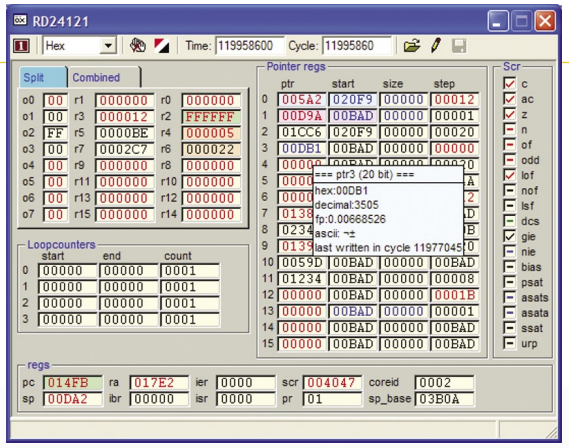
MEMORY WINDOW

Every memory space may be made visible in multiple Memory Windows. Within such a window one may set all types of breakpoints at every individual location or simultaneously at a range of locations. In every Memory Window one can define how a memory is instantiated and/or shared with other processors or peripheral blocks.



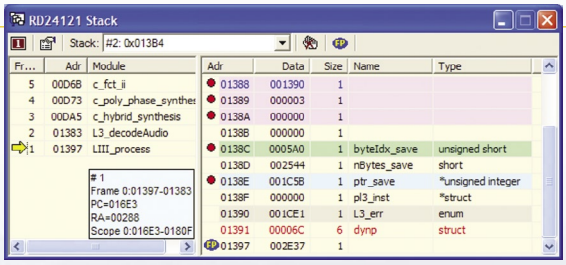
REGISTER WINDOW

Every processor has its own set of registers, which are displayed in a separate processor Register Window. In addition, the user may build his own customized Register Windows. Besides just viewing the content of registers, it is possible to overwrite these or set various types of breakpoints for each register. The register contents are color-coded indicating the type of usage of the registers, since last execution.



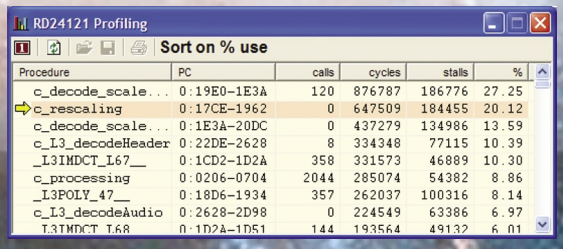
STACK WINDOW

Univers ISSIM automatically detects stack operations as well as different thread actions and will display these in the Stack Window. This allows Univers ISSIM to support any OS on any type of processor. This is one of the unique features of this integrated development environment.



PROFILE WINDOW

The Profile Window collects all data at run-time and it may be viewed at each moment in time. The, profiler displays execution cycles, stall cycles and types of stalls for each function in a very convenient way, highlighting the options for optimizing the application programs. The profiler supports graphical static profiling as well as graphical dynamic profiling.

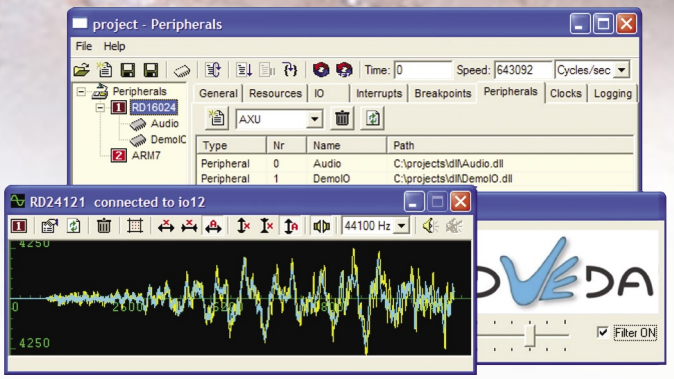


UNIVERS ISSIM

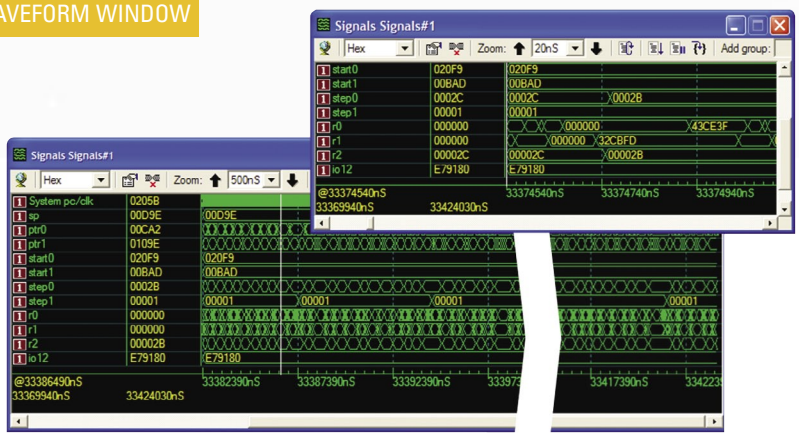
UNIVERS ISSIM

PERIPHERAL SUPPORT

Univers ISSIM provides extensive standard IO support from the Project Window. Additional peripherals can be described as 'C' models and included in the project. These peripherals may have their own custom windows. Data of the peripheral can be passed on through the sophisticated debug API. Adveda encourages engineers to use the real RTL peripheral description within Univers COVER, which saves a lot of work in writing C models.



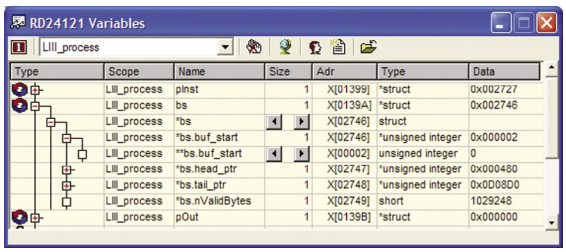
WAVEFORM WINDOW



No longer is the Waveform Window restricted to hardware engineers. With Madam Univers, software engineers can now also trace the signal history. In the Waveform Window any value within the ISS of each processor can be traced. This enables the display of the history of selected signals and may give a very good understanding of detailed cycle aspects. It allows to debug and fine-tune the application code.

VARIABLES WINDOW

All variables are displayed in the Variables Window. Variables may be selected per procedure or from a user-defined list. The window shows the value of a variable as well as his absolute address in one of the associated memory spaces.



ISS AND HW-TARGET

Univers ISSIM's Integrated Development Environment is a complete multi-core development. It supports interfaces to real hardware implementations of the processors. Some debug features may not be available in such mode, since not all data, available from the ISS models, can be obtained from real hardware implementations. The amount of debug features will vary per processor implementation.

ADVEDA

VERIFICATION SOLUTIONS FOR CHIP DESIGN

ADVEDA B.V. • FINISTÈRELAAN 18 • 5627 TD EINDHOVEN • THE NETHERLANDS • TEL: +31 40 291 23 85 • FAX: +31 40 291 23 86 • WWW.ADVEDA.COM • INFO@ADVEDA.COM