AGNISYS

SYSTEM DEVELOPMENT WITH CERTAINTY

THE HSI SOLUTIONS NEEDED TO DESIGN AND VERIFY TODAY'S SoCs

What's Inside

- Design Entry using Word, Excel or OpenOffice Calc
- Import SystemRDL, IP-XACT, YAML, RALF, CSV
- Parameterize and generate RTL Code
- Parameterize and generate UVM Register Models
- Generate C/C++ Header Files, API and Classes
- Generate SoC Datasheet documentation HTML, PDF, Word
- Generate complete UVM test environment with sequences
- Specify portable sequences and generate for Verification, Firmware Development and Post-Silicon Validation

BENEFIT FROM THE INDUSTRY'S MOST COMPREHENSIVE AND POWERFUL HSI DESIGN AND VERIFICATION SOLUTION

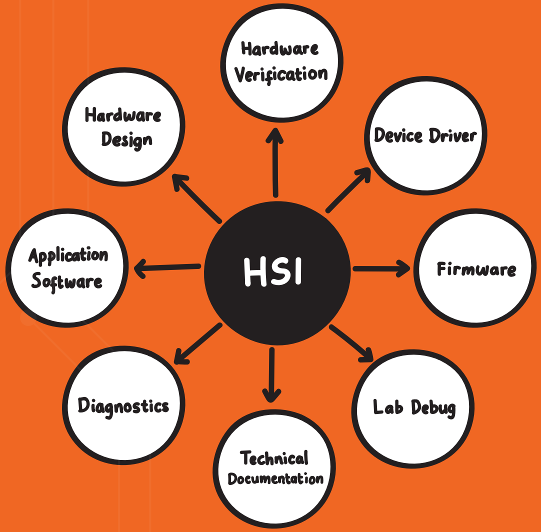
What's the Root Cause of Functional Flaws?

The functional specification remains the number one root cause of SoC functional flaws. It's also the main source of all design and verification activities. Any errors in the specification often lead to higher costs for fixing them and can even be the difference between success and failure for a given project.

The Hardware/Software Interface (HSI) data is a major component of the specification because it controls the configuration of peripherals and their communication with the software.

The HSI:

- includes the definition of addressable registers, interrupts, sequences, ports, API
- is often manually captured and maintained in Word™ or Excel™, and used in conjunction with various files such as IP-XACT or SystemRDL.
- is used by multiple isolated teams including system architects, designers, verification engineers, firmware developers, lab testers and SW developers
- registers need to be designed in SystemVerilog or VHDL, and the corresponding UVM models need to be created for verification



How much time do you spend creating register design files, and verification environments that could be saved by automation?

How do you bridge the gap between specification, design and verification?

With the amount of data and various SoC teams who create and rely on the HSI, it can easily get chaotic without a process in place – this is the main reason why the specification is the main culprit of functional flaws.

As a company established in 2007, we have been exclusively focused on solving the challenges associated with HSI. Our intuitive **Register Editors** are easy to use, and the powerful **Code Generators** are customizable for generating sign-off quality code in different formats for various SoC teams. You can specify a single test sequence and our **Portable Sequence Generator** generates sequences in various formats. Our **Automatic Register Verification** generates a complete UVM test environment that provides 100% register functional coverage without manually writing any UVM code.

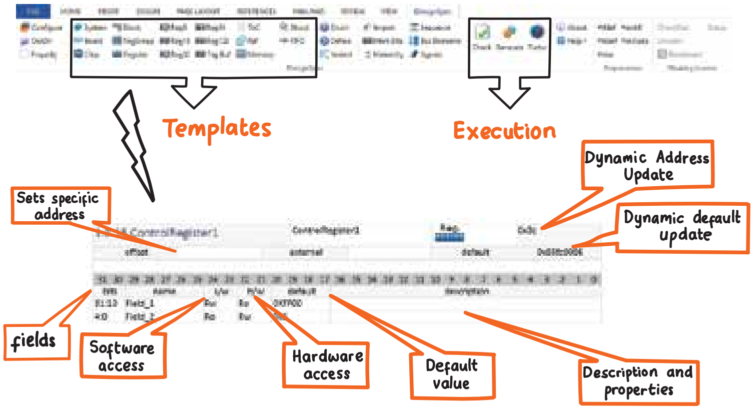
Partner with us. Partner with Agnisys!

Design Entry

Developed since 2007, our Editors are equipped with user-friendly register templates to assist you during register specification. Simple and complex registers can be created hierarchically such that large SoC designs are divided into manageable sub-blocks that are represented symbolically, designed and connected together. This methodology enables you to work on different parts of the design in parallel with a large team.

Using the Add-in to Word, Excel or OpenOffice™ you can:

- Specify the registers including register field names, widths, description and access types and various properties.
- Import and integrate IP-XACT, SystemRDL, YAML or RALF.
- Define RTL Properties for clocking, reset, special registers, counter, interrupt, multiple domains, memory mapping technology.
- Parameterize the auto-generated code using static values and expression with common operators
- Check for consistencies and errors



With our extensive support for **Parameterization**, you can specify various parameters and use them as macros in the specification. The value of the parameter can be static or based on expressions. Both string and numeric expressions are allowed and can be used with the following common operators:

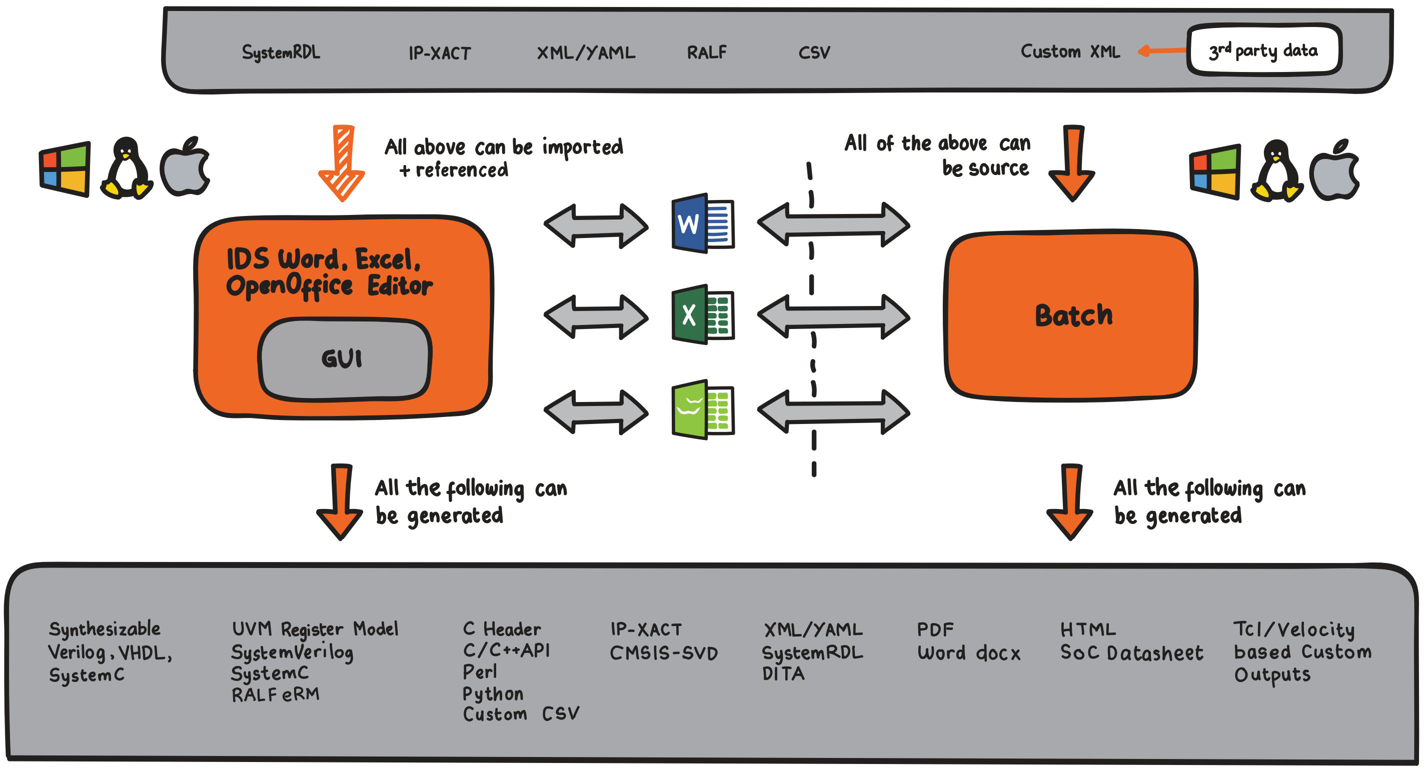
- Arithmetic operators : +, -, *, /, %, <<, >>, ()
- Comparison operators/relational operators : >,<, >=, <=, ==
- Conditional ternary operator: ?:
- Concatenation Operator : + (overloaded in string context)

The **Editor Compiler** helps create correct-by-construction specification by checking the following data:

- Address calculation & back-annotation on the specification
- Inserts Register Map – Table of Content (address, default values)
- Overlaps (Bit, Register, RegGroup and Block)
- Incomplete data or incorrect data
- Duplicate or illegal names
- Invalid access or incompatible access

Our **Document Generator** can output file formats such as HTML alt1/alt2, .doc, .PDF, .xls, IP-XACT or SystemRDL.

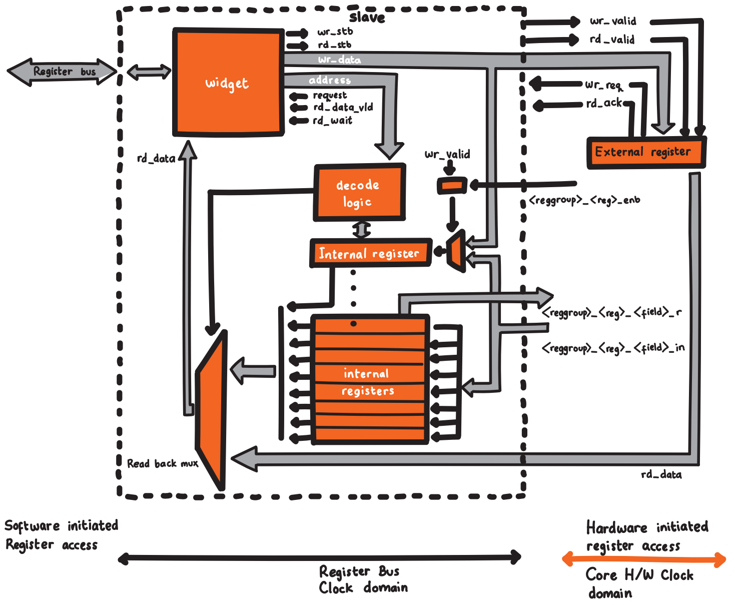
You can also customize the outputs using our **Velocity Template** based on a very simple architecture with two components – a data model and template. The data model is basically the input which describes the design specification, and the template is a plain text file that describes the static output and dynamic elements from the data model. This enables you to generate specialized code and documents required by your customers.


SystemRDL
IP-XACT
XML/YAML
RALF
CSV
Custom XML
3rd party data
All above can be imported + referenced
All of the above can be source
IDS Word, Excel, OpenOffice Editor
GUI
Batch
All the following can be generated
All the following can be generated
Synthesizable Verilog, VHDL, SystemC
UVM Register Model SystemVerilog SystemC RALF eRM
C Header C/C++API Perl Python Custom CSV
IP-XACT CMSIS-SVD
XML/YAML SystemRDL DITA
PDF Word docx
HTML SoC Datasheet
Tcl/Velocity based Custom Outputs

Code Generation

Based on the golden specification, various SoC teams can use the high-performance code generators via GUI or command line.

The generated **RTL Code** (VHDL, Verilog, SystemVerilog or SystemC) for the registers is human-readable with easy-to-follow comments. The RTL also includes a bus slave and a decode logic specific to the bus protocol (AHB, APB, AXI, AXI-Lite, or proprietary), ensuring instant connection of the application logic to the register bus.



A comprehensive list of **RTL Properties** can be defined hierarchically in the golden specification and reflected in the generated RTL code. This leads to a more customizable RTL to meet your design requirements. For example, you can easily specify clock_edge = posedge, reset_type = async or reset_level = high, and the generated RTL code reflects them.

The RTL Properties can be used to specify special registers such as **Shadow, Interrupt, Counter, Alias, Indirect, Lock, FIFO or Trigger Buffer, Wide, Multi-Dimensional or RO-WO Pair.**

The generated RTL code supports the following:

- **External Registers** - a register or regGroup implemented by the user outside the generated RTL.
- **Pipeline Stages** - in order to meet special timing requirements.
- **Special Control Signals** - examples include Write Pulse, Write One Pulse, Write Zero Pulse, Read Pulse, Clearing field on remote signals.
- **Low Power Output** - eliminates assigning the same value at every clock edge or eliminating write operations all together.

The generated **UVM Register Model** is also human-readable with easy-to-follow comments, and includes Register Arrays, Memories, Indirect Access Registers, FIFO Registers and Coverage, Constraints Models and hdl_path.

By defining a **UVM Property** called 'coverage', you can control the coverage for any particular register, reggroup, memory or block. This property is hierarchical so if applied at the chip level then it automatically sets the coverage to the rest of the element. You can set 'coverage' as follows:

- 'a' – Coverage code for the addresses read or written in an address map.
- 'b' – Coverage code for the bits read or written in registers.
- 'f' – Coverage code for the values of fields.
- 'on' – Coverage code for all the above mentioned types
- 'off' – No coverage code is generated

You can add the **hdl_path** using a property named 'hdl_path' with a value set to the hierarchical path. hdl_path is a mechanism by which each individual element in a UVM model is connected to the RTL model of the element.

Other project critical files can be generated such as SystemRDL, CMSIS-SVD, IP-XACT, XML and as well as C/C++ Header Files and API needed by the software team.

Portable Sequences

Sequences are a powerful tool for stimulus generation used by multiple teams involved in verification and post-silicon validation stages. But most teams lack the unified flow for creating sequences.

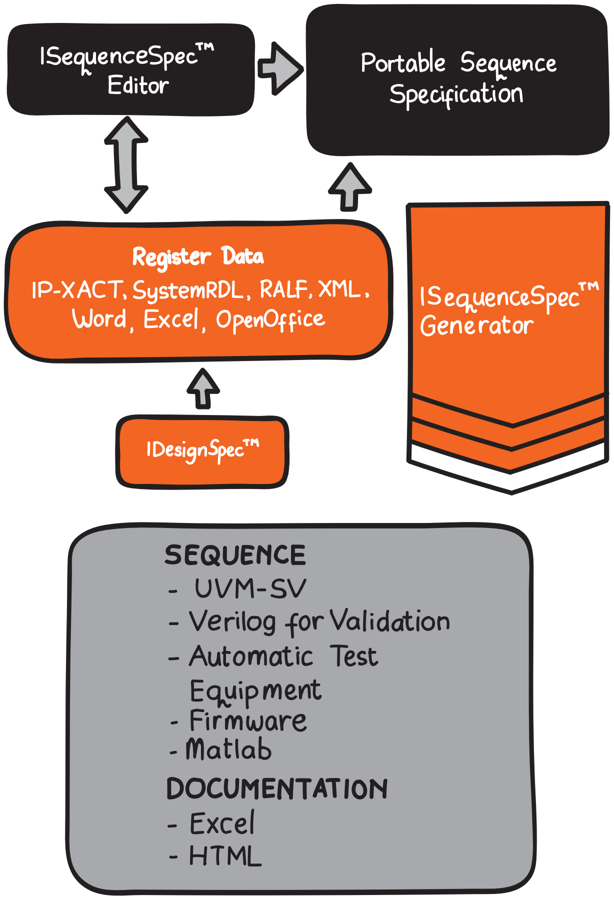
Our solution unifies the creation of portable sequences from a golden specification. By using the register information from IP-XACT, SystemRDL, XML, Word or Excel, you can create specific test sequences in a spreadsheet and generate multiple output formats for a variety of domains:

- UVM sequences for verification
- SystemVerilog sequences for validation
- C code for firmware & Device driver development
- Specialized formats for Automatic Test Equipment

The sequence constructs include loops, if-else, wait, arguments, constant or in-line functions. It supports constrained variables for randomized sequences and handling of indirect and interrupt register. The sequences support the following:

- Structures in arguments - refers to the datatype of the argument and all the elements inside the structure can be read or write in the sequence
- Signals in Sequence steps - values can be write/read to Reg/Reg Field and also signals can be written in one another.
- Repeat on Register - signifies how many times a particular element of Register/Register Group/Block/Chip can be repeated

The tool flow on how to generate portable sequences from a single specification is shown below.

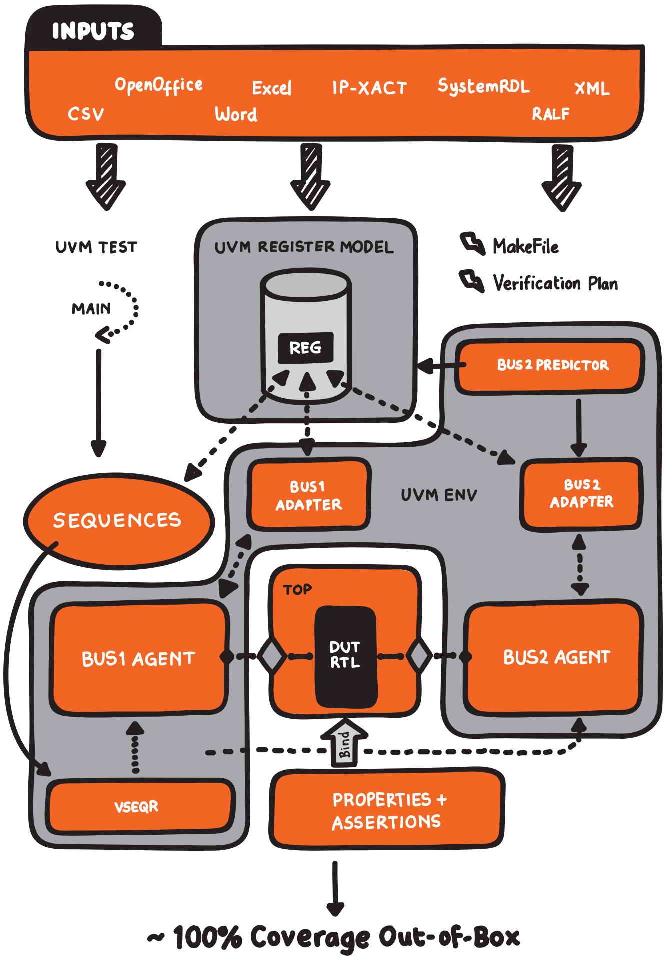


Automatic Register Verification

Our solution automates the tedious effort of creating a UVM based verification environment that provides **100% functional coverage** for registers. You can obtain faster coverage closure with the usage of constraint random stimulus generation, auto-generation of illegal bins for different register behavioral scenarios.

Based on the golden specification, you can generate the following for verification via **Simulation**:

- Complete and fully-connected UVM test environment including components, hdl_paths, covergroups, constraints and illegal bins
- Sequences for positive and negative functionality
- "Makefile" (for VCS, Incisive or Questa) to run the simulations and collect results from simulation database appropriate for the simulator being used
- The **Verification Plan** with ability to back annotate the simulation and coverage results.



Based on the golden specification, you can also generate the following code for **Formal Verification**:

- **SystemVerilog Properties and Assertions** to check the register access policies and compliance to bus protocols
- Top-level file to bind the DUT RTL as well as third-party design IP with the assertions.
- Makefile or TCL command scripts
- A Verification Plan with ability to back-annotate these formal results so that engineers can analyze the results
- Comprehensive C/C++ tests for Firmware/Validation

OUR CUSTOMERS

“ I was looking for a flexible tool that will generate quality RTL code while enabling our ASIC team to scale-up in a rapidly changing environment. I was also looking for a "one stop shop" that will give a good solution for design, verification, Firmware, and Architects.
I found IDesignSpec-Batch to be the tool that provides us with all of the above - and more.
I was also very happy with the great level of support and commitment I received from the Agnisys team.
Thanks Agnisys team.” - **Alon Shtepel, Micron**

“ **IDesignSpec is a great tool** that bridges the gap of register specifications to the design and verification of the actual logic. ”

- **Allegro MicroSystems**

“ We use IDesignSpec for ASIC design, generating our UVM register models from Word. It provides a massive productivity increase over hand-editing these files, with over 50,000 lines of code being generated in just a few short minutes.
We are able to have significant control over the UVM generation, adding specific coverage items and even incorporating our own register classes.
We’re currently expanding the use of the tool to integrate even more tightly with software development, to help keep the register descriptions and header generation in sync with the actual hardware. In all, it’s been the best kind of tool: one that removes useless drudgery and facilitates our workflow. Clearly a great tool to have in the toolbox.” - **Scott Reedstrom, Boston Scientific**

“ **Our experience with IDesignSpec has been extremely positive.**
We were looking for a tool that could describe our registers in a user-friendly input format.
This input acts as our golden reference to all our downstream consumers to include design, validation, verification, software, and modeling. IDesignSpec indeed satisfies all those requirements. ”

- **Bahaa Osman, FABU America’s Verification Team Leader**

AQUANTIA · Boston Scientific · facebook · Google · alphawave

decaWave · HARRIS · yellowbrick · violin MEMORY · dji

BARCLAYS · SanDisk · Rockwell Collins · icron · Leica

mercury systems · amazon · WD Western Digital · Allegro · IDEX

INTRINSIX · BOEING · JOHN DEERE · Micron · America FABU

IDT · ARM · ROSS · NETINT

AGNISYS
SYSTEM DEVELOPMENT WITH CERTAINTY

About Agnisys

Agnisys, Inc. is the leading supplier of Electronic Design Automation (EDA) software for solving complex design and verification problems for system development. Its products provide a common specification-driven development flow to describe registers and sequences for system-on-chip (SoC), Field Programmable Gate Array (FPGA) and Intellectual Property (IP) enabling faster design, verification, firmware, and validation. Based on patented technologies and intuitive user interfaces, its products increase productivity and efficiency while eliminating system design and verification errors. Founded in 2007, Agnisys is based in Boston, Massachusetts with R&D centers in the United States and India.

Corporate Office
75 Arlington St. Suite 500
Boston, MA 02116
1 (855) VERIFYY (837-4399)
1 (866) 927-3653 (fax)

www.agnisys.com

IDesignSpec™

An end-to-end solution for design and verification of all SoC addressable registers and interrupts, enabling IP, HW, Firmware and SW teams to collaborate around a single specification.

- Available as command line batch utility or as an add-in to Word™, Excel™ or OpenOffice
- Imports SystemRDL 1.0/2.0, IP-XACT, XML, YAML, CSV, RALF
- Auto-generates sign-off quality register RTL models, UVM models, C/C++ models, SystemC models and Documentation

ARV™

An add-on for IDesignSpec that auto-generates a complete UVM environment with sequences for RTL simulation, C-based tests, and assertions for formal verification, providing 100% functional coverage.

- Provides traceability between register design and verification elements
- Integrated with commercial RTL simulation and formal tools
- Generates verification plan with back-annotation of test results
- Supports special registers including Lock, Indirect, Contraint, FIFO or Multiple-Bus Domains

ISequenceSpec™

An add-on for IDesignSpec that auto-generates user-defined and portable UVM sequences for simulation and C-based firmware sequences for HW/SW co-simulation and post-silicon validation, bridging the gap between specification and silicon validation.

- Sequence Editor available in Excel™ or batch utility
- Capture sequences at a higher level in-sync with register specification
- Sequence constructs include loops, if-else, wait, arguments, constant, in-line functions
- Hooks to the latest Portable Stimulus Standard (PSS)