

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon



design & verification
conference & exhibition
February 14 - 16, 2005


SystemVerilog Interoperability Checklist


Stuart Sutherland
Sutherland HDL, Inc.

- ◆ Update (Feb 2006): This checklist was never finished (sorry!)
 - ◆ The standard kept evolving to fast
 - ◆ As a small company, we just didn't have time to finish
 - ◆ Major tools have implemented so much of SystemVerilog that there is less need for this checklist

© 2005, Sutherland HDL, Inc.

This presentation will...





- ◆ Discuss EDA tool support for SystemVerilog
- ◆ Discuss **obstacles** that are preventing companies from adopting SystemVerilog
- ◆ Provide a solution to these obstacles


The primary goal is enable design and verification engineers to begin using SystemVerilog ASAP!

© 2005, Sutherland HDL, Inc.DVCon-2005: SystemVerilog Interoperability Checklist3 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon



*Training engineers
to be HDL wizards*

SystemVerilog


verification design	assertions test program blocks clocking domains process control	mailboxes semaphores constrained random values direct C function calls	classes inheritance strings int shortint longint byte shortreal void alias	from C / C++ dynamic arrays associative arrays references globals break continue return do-while unions casting const
	interfaces nested hierarchy unrestricted ports automatic port connect enhanced literals time values and units specialized procedures	packages 2-state modeling packed arrays array assignments queues unique/priority case/lf compilation unit space	(* attributes *) configurations memory part selects variable part select	multi dimensional arrays signed types automatic ** (power operator)

Verilog-2001

ANSI C style ports generate localparam constant functions	standard file I/O \$value\$plusargs `ifdef `elsif `line @*	multi dimensional arrays signed types automatic ** (power operator)
--	---	--

Verilog-1995

modules parameters function/tasks always @ assign	\$finish \$open \$fclose \$display \$write \$monitor `define `ifdef `else `include `timescale	initial disable events wait # @ fork-join	wire reg integer real time packed arrays 2D memory	begin-end while for forever if-else repeat
---	---	---	--	--







*Training engineers
to be HDL wizards*




Which EDA Vendors CLAIM they are Supporting SystemVerilog?







- ◆ AT DAC-2004, **more than 50 EDA companies** claimed they support SV
- ◆ Just *a few* of the companies are...





































© 2005, Sutherland HDL, Inc. DVCon-2005: SystemVerilog Interoperability Checklist 5 of 30

SystemVerilog Interoperability Checklist


DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Which EDA Vendors **REALLY** Support SystemVerilog?



We support SystemVerilog (trust me!)




EDA sales rep

- ◆ At Sutherland HDL, we found...
 - EDA Marketing claims regarding SystemVerilog support are grossly exaggerated!***
- ◆ The truth (as observed by Sutherland HDL) is
 - ◆ At this paper date, no product has 100% SystemVerilog support
 - ◆ Every product supports different features of SystemVerilog
 - ◆ It is not easy to find a common subset that works with all tools

© 2005, Sutherland HDL, Inc.DVCon-2005: SystemVerilog Interoperability Checklist6 of 30

Top 10 ~~Excuses~~ Reasons That No EDA Vendor Supports 100% of SystemVerilog (yet)



- 1) We will support all of SystemVerilog, but it will take time
 - ◆ The SV LRM is 616 pages — just to describe extensions to Verilog!
- 2) Some SystemVerilog features are not needed for some products
 - ◆ E.g., a synthesis compiler should not support testbench features
- 3) Some features of SystemVerilog are too hard to implement
- 4) We have to first support Verilog-2001 before we can support SV
- 5) We are waiting for the IEEE SystemVerilog standard to be approved
- 6) Some SystemVerilog features came from our competitor
- 7) We have a better way to do certain SystemVerilog features
- 8) Our customers are not asking for some SystemVerilog features
- 9) We don't think anyone will ever use certain SystemVerilog features
- 10) SystemVerilog is not needed; use SystemC, VHDL, e, ...

© 2005, Sutherland HDL, Inc.DVCon-2005: SystemVerilog Interoperability Checklist7 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Obstacles to Adopting SystemVerilog

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

How can I, as a user, adopt SystemVerilog, when my EDA vendors have not implemented everything in SystemVerilog?

*This nifty SystemVerilog feature would really make my life easier...BUT,
Do all the tools in my design flow support that feature?*

Tool A supports (or claims to):

- ✓ structures
- ✓ unions with real types
- ✓ importing from packages
- ✓ compilation unit declarations
- exporting tasks to interfaces
- Constrained random testing



Tool B supports (or claims to):

- ✓ structures
- unions with real types
- importing from packages
- ✓ compilation unit declarations
- exporting tasks to interfaces
- ✓ Constrained random testing

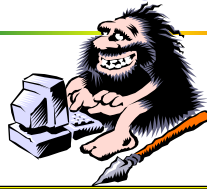
© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

8 of 30

Solution 1: Don't Use SystemVerilog, Stick with Trusty Old Verilog-1995

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*



John Cooley, moderator of DeepChip (www.deepchip.com) asked engineers:

Are you using SystemVerilog today?

"SystemVerilog looks promising. We like the concept. We won't use it until all of our tools in the flow support it...I don't think we will adopt it yet for a few years."

Maynard Hammond, Scientific Atlanta



How much design/verification productivity is lost by not taking advantage of what can be used in SystemVerilog today?

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

9 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Solution 2:

Use a subset of SystemVerilog Today (and a larger subset on the next project)

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

John Cooley, moderator of DeepChip (www.deepchip.com) asked engineers:

Are you using SystemVerilog today?

ANSWER:

- 1 in 5 respondents are already using some of SystemVerilog
- Many respondents indicated they would begin using some aspects of SystemVerilog very soon

"Using SystemVerilog on a current project...Today we are using the [design] extensions. In the future we will be using assertions."

Don Monroe, Enterasys Networks



A competitive company might say...

We hope all of our competitors wait to benefit from SystemVerilog!

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

10 of 30

Leap Over the Obstacles! You Do Not Need to Wait to Use SystemVerilog!

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ There are good reasons **NOT to wait** to use SystemVerilog
 - ◆ SV can help engineers create successful designs more quickly
 - ◆ SV can eliminate many types of subtle RTL coding errors
 - ◆ SV enables verifying complex designs using a single language
- ◆ It is not necessary to wait for 100% support of SystemVerilog to benefit from SystemVerilog
 - 1) Identify which SV constructs would be useful for a specific project
 - 2) Identify what EDA tools will be used in the project
 - 3) Identify what in-house tools will be used in the project
 - 4) Determine which SV constructs from the subset that you would like to use in the project, are supported by the tools to be used
 - 5) Re-evaluate what features can be used for each new project (or more often, if necessary and possible)

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

11 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Asking the Right Questions...

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ If you ask a general question, you will get a general answer!

*Does your XYZ tool
support SystemVerilog
structures?*



*Of course we support
structures — trust me!*



*Is the sales rep lying if the tool supports the structure syntax, but
does not yet support all of the ways a structure can be used?*

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

12 of 30

Digging Down to the Nitty-Gritty Details of SystemVerilog Support

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ It is not enough to ask: *Does your XYZ tool support SystemVerilog structures?*
- ◆ You must **ask questions on how you will use each construct**:
 - Are structure declarations supported **in modules**?
 - Are structure declarations supported **in interfaces**?
 - Are structure declarations supported **in packages**?
 - Are structure declarations supported **in the compilation unit space**?
 - Are **typedefs of structures** supported?
 - Are both **packed and unpacked** structure declarations supported?
 - Can **all (or specific) variables types** be declared within structures?
 - Can **net types** be declared as a structure type?
 - Can structures be **passed through module ports**?
 - Can structures be **passed to/from tasks and functions**?
 - Can structures be **initialized at declaration using an expression list**?
 - Can structures be **initialized at declaration using a default value**?
 - Can structures be **assigned a list of values**?
 - Can ...

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

13 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

The Sutherland HDL SystemVerilog Interoperability Checklist

**SUTHERLAND
HDL**
*Training engineers
to be HDL wizards*

- ◆ To determine the nitty-gritty details on SystemVerilog support, at Sutherland HDL we have been developing the

SystemVerilog Interoperability Checklist

- ◆ A list of EVERY feature in SystemVerilog
- ◆ Each feature is broken down to detailed aspects of that feature
- ◆ Checkboxes to fill in for:
 - ◆ What features are important for a specific project
 - ◆ What features are supported in each of several tools
- ◆ The Checklist is in the form of a **Microsoft Excel spreadsheet**
 - ◆ Too large to include in the paper proceedings (8 page limit)
 - ◆ PDF would limit the Checklist usefulness

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

14 of 30

SystemVerilog Interoperability Checklist Example

**SUTHERLAND
HDL**
*Training engineers
to be HDL wizards*

SystemVerilog Construct	LRM Clause	Importance in my project	Tool A	Tool B	Tool C	Tool D
Enumerated types						
Structures						
Unpacked structures						
Integral types within	3.11					
Real types within	3.11					
Unpacked arrays within	3.11					
Packed arrays within	3.11					

◆ The Checklist is available for the Sutherland HDL web site

- ◆ Free to download, free to use
- ◆ No strings attached
- ◆ Can be adapted, modified, customized, butchered, ...

◆ Update (Feb 2006): This checklist was never finished (sorry!)

- ◆ The standard kept evolving to fast
- ◆ As a small company, we just didn't have time to finish
- ◆ Major tools have implemented so much of SystemVerilog that there is less need for this checklist

The spreadsheet is divided into major categories

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Who Should Use the SystemVerilog Interoperability Checklist

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

◆ Primary intent:



◆ For users (design engineers and verification engineers)

- To determine which SV features are important in a current project
- To determine which SV features are important for a future project
- To determine an SV subset common to the tools that can be used in a project

◆ Secondary intent:

- EDA vendors might find the checklist useful as they implement SystemVerilog



© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

16 of 30

SystemVerilog Interoperability Checklist Organization

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

◆ The Checklist divides SystemVerilog into major categories

- 1) Productivity enhancements
- 2) Data encapsulation enhancements
- 3) RTL enhancements
- 4) Abstract modeling enhancements
- 5) Assertions
- 6) Testbench enhancements
- 7) Object-oriented verification
- 8) API Enhancements



NOTE: The categories used in this Checklist in no way imply any subsets to the SystemVerilog standard, and should not be construed as such. The categories are merely for convenience in organizing a very large spreadsheet.

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

17 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

SystemVerilog Productivity Enhancements

**SUTHERLAND
HDL**
*Training engineers
to be HDL wizards*

◆ Productivity enhancements (in the author's opinion):

- ◆ Make it easier to model in Verilog
 - ◆ More functionality with fewer lines of code!
 - ◆ Eliminate common modeling errors
 - ◆ Do *not* add significant new functionality
- | | |
|--------------------------------------|---|
| ✓ Specifying time units & precision | ✓ Relaxed rules for using variables |
| ✓ Enhanced `define text substitution | ✓ Relaxed rules for module ports |
| ✓ Block names | ✓ Module instantiation shortcuts |
| ✓ Statement labels | ✓ Task/function default argument direction |
| ✓ Named end statements | ✓ Passing task/function args by name |
| ✓ Enhanced literal values | ✓ Passing task/function arguments by reference (pointers) |
| ✓ Replacement for "reg" keyword | ✓ C-like function returns |
| ✓ 2-state "bit" data type | ✓ Task/function implicit statement groups |
| ✓ Local for-loop variables | |

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

18 of 30

SystemVerilog Data Encapsulation Enhancements

**SUTHERLAND
HDL**
*Training engineers
to be HDL wizards*



◆ Data encapsulation enhancements

- ◆ Bundle many discrete signals or data together
 - ◆ Perform operations on bundles of signals or data
- | | |
|-----------------------|---|
| ✓ Interfaces | ✓ Vectors with subfields (packed arrays) |
| ✓ Packages | ✓ Initialize arrays with list of values |
| ✓ Nested modules | ✓ Initialize arrays to default values |
| ✓ Unpacked structures | ✓ Assign arrays to arrays |
| ✓ Packed structures | ✓ Select and assign slices of arrays |
| ✓ Unpacked unions | ✓ Assign list of values to arrays |
| ✓ Packed unions | ✓ Passing arrays/structures/unions through ports and to tasks/functions |
| ✓ Tagged unions | |

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

19 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

SystemVerilog RTL Modeling Enhancements



**SUTHERLAND
HDL**
*Training engineers
to be HDL wizards*

◆ RTL modeling enhancements

- ◆ Make it easier to model for synthesis
- ◆ Remove model ambiguity
- ◆ Reduce pre- and post-simulation mismatches
- ◆ **Automatic warnings if model does not match designer's intent**

- ✓ Specialized procedural blocks for
 - Combinational logic
 - Sequential logic
 - Latched logic
- ✓ "unique" decision modifier
- ✓ "priority" decision modifier
- ✓ User-defined types
- ✓ Enumerated types
- ✓ Increment/decrement operators
- ✓ Assignment operators
- ✓ "Don't care" comparison operators
- ✓ Void functions

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

20 of 30

SystemVerilog Abstract Modeling Enhancements

**SUTHERLAND
HDL**
*Training engineers
to be HDL wizards*

◆ Abstract modeling enhancements

- ◆ Model more functionality with fewer lines of code
- ◆ More like programming than register transfer level code
- ◆ Enables higher level behavioral and bus functional modeling
- ◆ Type compatibility with C models and SystemC models

- ✓ C-like data types (int, longint, etc.)
- ✓ Unsigned type modifier
- ✓ type casting
- ✓ vector size casting
- ✓ signedness casting
- ✓ "const" variables
- ✓ Redefinable data types
- ✓ C-like jump statements
 - break
 - continue
 - return
- ✓ Bottom testing do-while loop
- ✓ Array iteration for-each loop

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

21 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

SystemVerilog Assertions and Coverage

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*



No more doing things
the hard way!

◆ Assertions and coverage enhancements

- ◆ Enable verification of complex logic sequences
 - ◆ Provide control over pass/fail message generation
 - ◆ Simplifies white box and black box testing
 - ◆ Automatic reporting of verification coverage
- | | |
|------------------------------------|----------------------------------|
| ✓ Immediate assertions | ✓ Assertion severity levels |
| ✓ Concurrent assertions | ✓ Assertion control |
| ✓ PSL-like property specifications | ▪ disable iff |
| ▪ property blocks | ▪ \$assertoff/\$asserton |
| ▪ sequence blocks | ✓ assert/assume/cover directives |
| ✓ Local variables in properties | ✓ Coverage grouping and bins |
| ✓ Multi-clock sequences | ✓ Coverage reporting |

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

22 of 30

SystemVerilog Testbench Enhancements

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

◆ Testbench enhancements

- ◆ Make it easier to represent test programs
 - ◆ Provide special event scheduling for testing events
 - ◆ Prevent common test-to-design race conditions
 - ◆ Cycle-based test timing
- | | |
|-------------------------------------|-------------------------------|
| ✓ Extended Verilog event scheduling | ✓ Dynamic arrays |
| ▪ Preponed region | ✓ Associative arrays |
| ▪ Observe region | ✓ String arrays |
| ▪ Reactive region | ✓ String methods |
| ✓ Program blocks | ✓ Final blocks |
| ✓ Clocking blocks | ✓ fork...join_any/join_none |
| ✓ Cycle delays (##) | ✓ Event data type persistence |

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

23 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

SystemVerilog Object Oriented Verification

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ Object-Oriented verification
 - ◆ Adds true verification language capabilities to Verilog
 - ◆ Based on the Open-VERA verification language
 - ◆ Enables advanced, high-level verification methodologies
 - ◆ Enables modularized, re-usable verification programming
- ✓ C++ like class objects
 - With Java-like automatic garbage collection
- ✓ Object construction using "new"
- ✓ Class inheritance
 - Enables polymorphic testing
- ✓ Public, private and protected classes
- ✓ Built-in semaphore class objects
- ✓ Built-in mailbox class objects
- ✓ Constrained random value generation

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

24 of 30

SystemVerilog API Enhancements

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ Application Programming Interface (API) enhancements
 - ◆ Increase Verilog's PLI/VPI capabilities
 - ◆ Extend Verilog's VPI to support SystemVerilog constructs
 - ◆ Simplify interacting with C/C++ using a direct interface
- ✓ VPI extensions
- ✓ Assertions API
- ✓ Coverage API
- ✓ Extended VCD files
- ✓ Direct Programming Interface
 - Import C functions into Verilog
 - Export Verilog functions to C
 - Export Verilog tasks to C

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

25 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

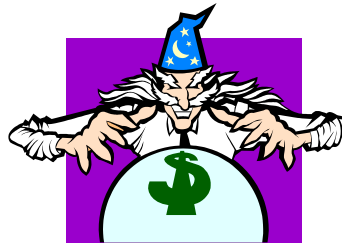
by Sutherland HDL, Inc., Portland, Oregon

The Big Question: What are the Checklist Results?

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ **Cadence**
 - Simulation
 - Synthesis
 - Formal
 - Hardware acceleration
- ◆ **Mentor Graphics**
 - Simulation
 - Synthesis
 - Formal
 - Hardware acceleration
- ◆ **Synopsys**
 - Simulation
 - Synthesis
 - Formal
 - Hardware acceleration

The answer is...



© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

26 of 30

How Do Your Tools Measure Up?

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*

- ◆ **This paper does not answer that question!**
- ◆ Why not?
 - 1) How big of a fool do you think I am?
 - 2) No vendor supports 100% of SystemVerilog today (16 Feb 2005)
 - 3) No design/testbench needs 100% of SystemVerilog
 - ◆ Some projects can best from one set of SV constructs
 - ◆ Other projects can best benefit for a different set of SV constructs
 - ◆ Few, if any, engineers will master everything in SV all at once
 - 4) There are 75+ EDA companies, which ones are your vendors?
 - 5) Many companies have in-house tools that parse Verilog/SystemVerilog code
- ◆ This paper gives you a tool, so **YOU** can answer the questions
 - ◆ The checklist is free to download, free to modify, free to use



© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

27 of 30

SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Caveats



- ◆ **The SystemVerilog Interoperability Checklist:**
 - ◆ Developed by Sutherland HDL, Inc. for use at Sutherland HDL
 - ◆ The Checklist is detailed, but not exhaustive (too many corner cases)
 - ◆ The Checklist may have errors, but we think it is accurate
 - ◆ Checklist categories are for convenience in organization
 - ◆ The categories do not imply language subsets
 - ◆ Many constructs could have fit into other categories
 - ◆ **Has no guarantees or maintenance, expressed or implied**
- ◆ **The Checklist is available**
 - ◆ The official P1800 SystemVerilog ballot draft renumbered sections
 - ◆ The original checklist focused on design constructs
 - ◆ Sutherland HDL is in the process of adding verification constructs

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

28 of 30

Conclusion



- ◆ **SystemVerilog is critical to successful designs**
 - ◆ Enhanced modeling capabilities to handle large designs
 - ◆ Enhanced, and unified, verification capabilities to test large designs
- ◆ **Much of SystemVerilog can be used TODAY!**
 - ◆ Many EDA vendors have partial SystemVerilog support
 - ◆ No vendor has 100% SystemVerilog support (as February 2005)
- ◆ **To benefit from SystemVerilog right away, users must:**
 - ◆ Identify which portions of SystemVerilog are needed
 - ◆ Identify which portions of the SV subset are currently supported
- ◆ **The SystemVerilog Interoperability Checklist is intended to help companies begin benefiting from SystemVerilog TODAY**
 - ◆ Free to download and customize from www.sutherland-hdl.com

© 2005, Sutherland HDL, Inc.

DVCon-2005: SystemVerilog Interoperability Checklist

29 of 30


SystemVerilog Interoperability Checklist

DVCon-2005 Paper

by Sutherland HDL, Inc., Portland, Oregon

Questions?

SUTHERLAND
HDL
*Training engineers
to be HDL wizards*



© 2005, Sutherland HDL, Inc. DVCon-2005: SystemVerilog Interoperability Checklist 30 of 30

The slide features a cartoon wizard with a long grey beard, wearing a red pointed hat and a red robe decorated with yellow stars and crescent moons. The wizard is positioned in the center of the slide. The text 'Questions?' is in the top left, and the Sutherland HDL logo is in the top right. A horizontal line with a rainbow gradient is located below the 'Questions?' text. The footer contains copyright information and a page number.