Keysight Technologies
LTE Reference Vector

White Paper

KEYSIGHT TECHNOLOGIES

Introduction

The third-generation Universal Mobile Telecommunications System (UMTS), based on Wideband Code-Division Multiple Access (W-CDMA), has been deployed all over the world. To ensure that this system remains competitive in the future, the 3rd Generation Partnership Project (3GPP) began a project in November 2004 to define the long-term evolution of UMTS cellular technology. The specifications related to this effort are formally known as the Evolved UMTS Terrestrial Radio Access (E-UTRA) and Evolved UMTS Terrestrial Radio Access Network (E-EUTRAN) but are more commonly referred to by the project name LTE. The first version of LTE is documented in Release 8 of the 3GPP specifications.

Rapid evolution of new commercial wireless standards such as 3GPP Long Term Evolution (LTE) poses significant challenges for physical layer baseband design of commercial wireless products. This also creates challenges for Software-Defined Radio (SDR) development, in which a common hardware/software platform may be required to support multiple wireless standards such as LTE and Mobile WiMAX™.

Wireless developers are under increasingly demanding schedules to develop products before wireless standards finalize to meet product shipment schedules. This introduces product development risks to the baseband coding/decoding physical-layer design.

The physical-layer baseband design cycle often progresses in parallel with a wireless standard which is still rapidly evolving. Thus, algorithm definition is subject to the developer's interpretation of the preliminary standard, which may not yet be complete and may not yet be well-defined. Unfortunately, misinterpreting the wireless standard on the baseband implementation can result in costly time delays in re-working designs and potentially missing critical time-to-market windows.

Designing flexibility into hardware-software platforms is key to SDR development, where a common radio platform may need to be re-configurable to support multiple signal formats (LTE, WiMAX, WCDMA, custom/proprietary formats, etc.). The flexibility of FPGAs for the coding/decoding chain is attractive for these applications.

The breadth and flexibility of these new emerging wireless standards such as LTE and Mobile WiMAX present significant challenges for FPGA design. For example, LTE supports a number of modulation types (QPSK, 16QAM, 64QAM) with many different configurations in the transport block size, Resource Block (RB) allocation, different CRC lengths/types, etc. Traditional FPGA development methodologies of writing/creating independent test vector references (in addition to writing the actual HDL code needed for the implementation) may become problematic in terms of the development overhead needed to generate the many test vectors sets required to verify the many configurations supported by LTE and Mobile WiMAX.

A new approach is needed to facilitate rapid development and testing of FPGA-based physical-layer coding/decoding implementations for emerging standards such as LTE. This white paper outlines an improved approach to verify FPGA development using independent and configurable reference test vectors.

FPGA Development—Today's Existing Approach

FPGA developers typically start with system specifications that include behavioral requirements. These specifications will likely have some level of mathematical or pseudo-code representation, and may be part of a wireless standard which is still evolving and incomplete.

The FPGA development team will interpret these specifications and may develop reference models to generate stimulus inputs and expected outputs. These values, often called test vectors, will be used in testing of the synthesizable RTL (register transfer level) HDL code used to generate the actual design through synthesis and backend map, place and route tools.

These reference models are typically simulated at a higher level of abstraction than the simulation done at FPGA RTL level simulation, often using cycle based simulation models which run significantly faster than timing wheel simulations used for RTL development. These RTL simulations, while not gate-level timing accurate (timing closure is gained through static analysis at the backend point), still use fractions of the clock cycle to test for details not appropriate for behavioral high level simulation.

To summarize, today's approach to FPGA development highlights several key challenges:

- FPGA development can occur in parallel with an evolving wireless standard.
- Wireless standard may be incomplete, and subject to interpretation.
- Standards interpretation introduces risk in developing both the reference models and FPGA HDL code.

These items, combined with the many possible configurations supported by emerging standards such as LTE, highlight a need for an improved approach.

An Improved Approach--Configurable Reference Test Vectors

As mentioned in the previous section, FPGA developers need reference test vectors to verify that their design conforms to the wireless standard/specification. For LTE, this can present significant development time overhead, given the many configurations supported by the LTE standard. Complex blocks, such as the Turbo coder/decoder, may involve significant time to write and then create a behavioral test vector reference from scratch. Unfortunately, this can extend the development time already required for the primary task at hand--writing the HDL code for the actual FPGA implementation.

In addition to the development time overhead, there is significant risk associated with this approach: the FPGA developer is writing/creating their own behavioral verification vectors to check their HDL code. This is analogous to an author writing their own spell checker. If an author is unsure about the spelling of a word, or convinced that a word is spelled in a particular manner, then spell checking with a self-written spell checker can be ineffective in verifying the spelling. The same holds true for FPGA development. If the LTE standard is misinterpreted when writing the HDL code, the same misinterpretation may be repeated when writing/creating the behavioral vector reference.

A commercial independent test vector reference which is parameterized and configurable for the many supported LTE configurations can help address these challenges.

This is conceptually illustrated in Figure 1, where vectors are generated with the independent test vector reference using the same stimulus that is applied to the HDL simulator. The HDL simulator output is then compared to the independent test vector reference to verify that they agree behaviorally.

If they do agree, the interpretation of the wireless standard is consistent. If they don't agree, further de-bugging can be performed with the intermediate nodes of the independent test vector reference. The simulated test vector reference algorithms can also be customized to achieve consistency, if needed to resolve differences in standards interpretation.

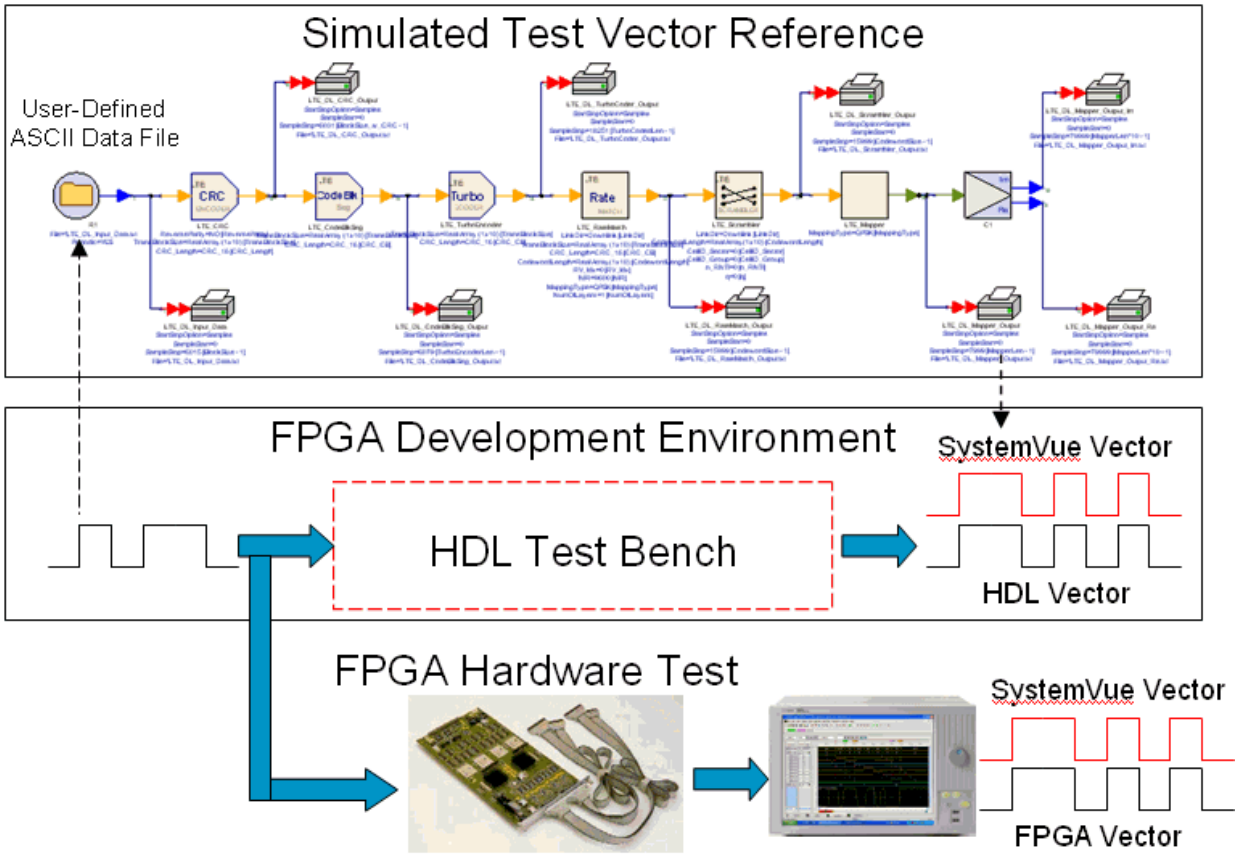

Figure 1. A conceptual illustration, using independent test vector references, which are parameterized and configurable for the many supported LTE configurations.

SystemVue Reference Test Vectors

An example of an LTE commercial test vector reference is shown in Figure 2.

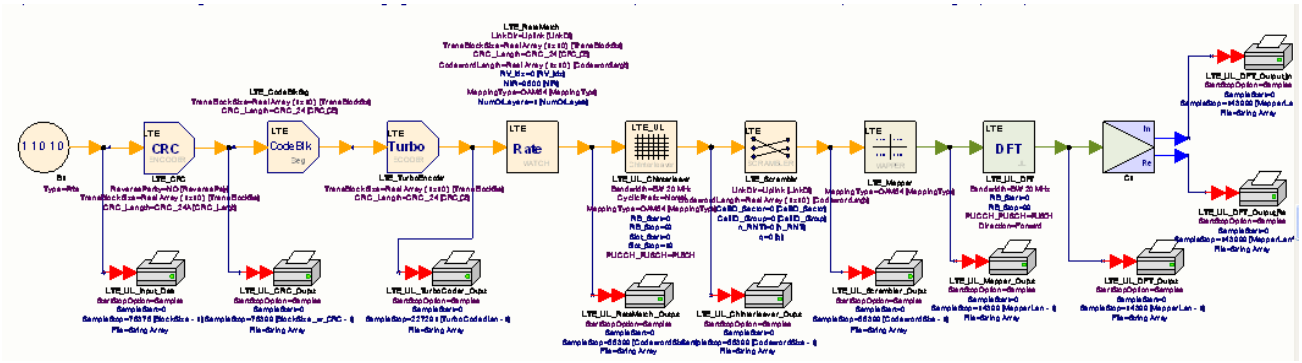

Figure 2. An example of SystemVue's LTE vector generation.

This is the LTE Uplink coding chain test vector reference from Agilent's SystemVue. This coding chain reflects the uplink coding defined in 3GPP specification TS36.212 and TS36.211.

When simulated, test vectors are generated at each stage of the coding chain and stored out as ASCII files for compatibility with other tools used in an FPGA flow. Test vectors can be stored with user-defined formats to fit into existing FPGA tool flows, such as:

- Floating point (default).
- Fixed point with user-defined precision.
- Integer.

The MathLang scripting capability in SystemVue also can be used to customize the test vectors into user-defined formats.

SystemVue vectors can be used for end-to-end HDL simulations and FPGA testing of LTE coding/decoding chains. Intermediate stage vectors can also be used for HDL simulations and FPGA testing of subsections or individual coding/decoding blocks. Intermediate stage vectors are particularly useful in debugging test vector discrepancies, as described in the next section.

There are pre-configured SystemVue reference designs for the LTE Uplink (UL) and Downlink (DL) channel coding, and UL and DL channel decoding. LTE test vector configurations can be easily modified by the user by changing parameters in the SystemVue workspace as shown in Figure 3. Changing parameters in the workspace as shown below automatically propagates the changes to the corresponding SystemVue LTE models.

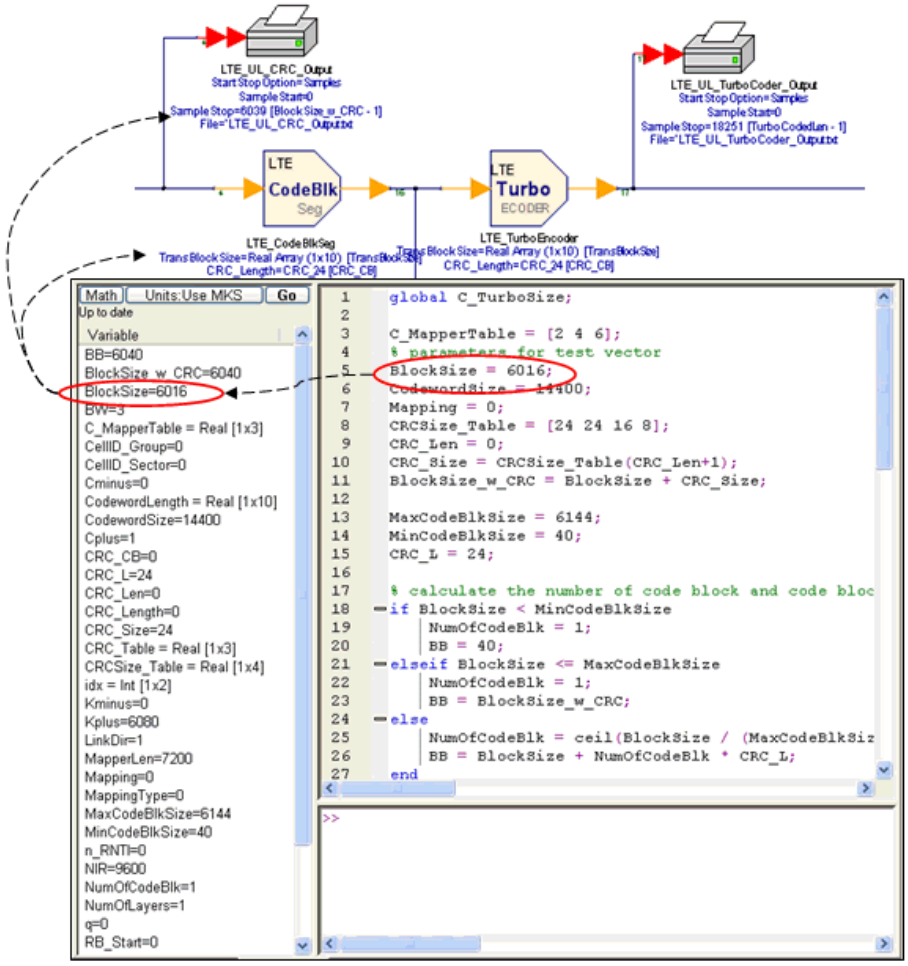

Figure 3. Changing SystemVue LTE test vector configurations.

Debugging a Discrepancy

There are several options available if a discrepancy is found between the HDL vectors and the SystemVue vectors.

It can be difficult to diagnose an issue when a discrepancy occurs in using end-to-end vectors. An example is having a bit reversal in the UL de-interleaver, which could propagate discrepancies along the entire decoding chain, including the de-rate matching, Turbo decoding, code block de-segmenting, and CRC decoding. In instances like these, it is helpful to use intermediate stage test vectors to provide more resolution and fidelity in debugging discrepancies, as illustrated in Figure 4.

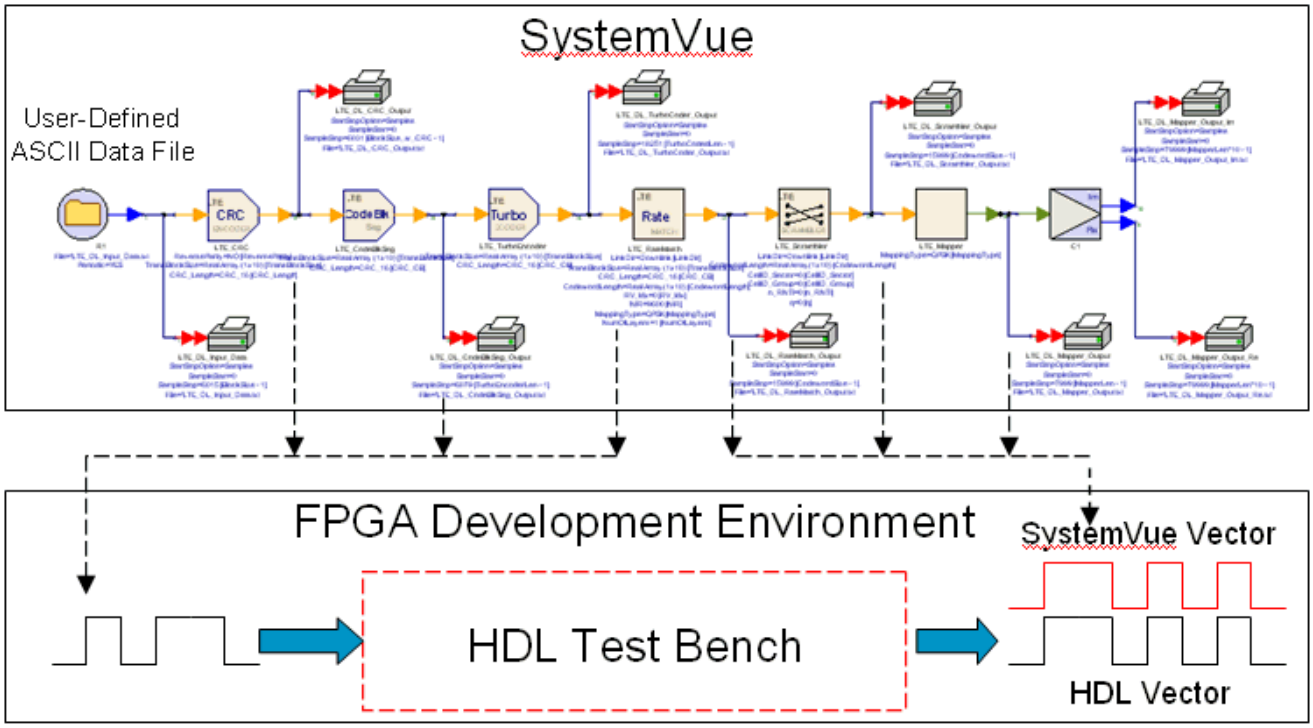

Figure 4. Using intermediate stage vectors for debugging.

Debugging a Discrepancy, continued

It can also be useful to bring the HDL into SystemVue for direct comparison to the simulated test vector references in the SystemVue simulation environment. Figure 5 shows an example where HDL co-simulation is used to directly compare vectors with the simulated model.

Several potential issues can result in a discrepancy between the two sets of vectors:

1. A difference in the interpretation of the standard when writing the HDL code
2. A difference in the interpretation of the standard in the SystemVue model
3. An error in the HDL code or HDL test bench

Once the block(s) causing the discrepancy has been identified, there are several options available to resolve it.

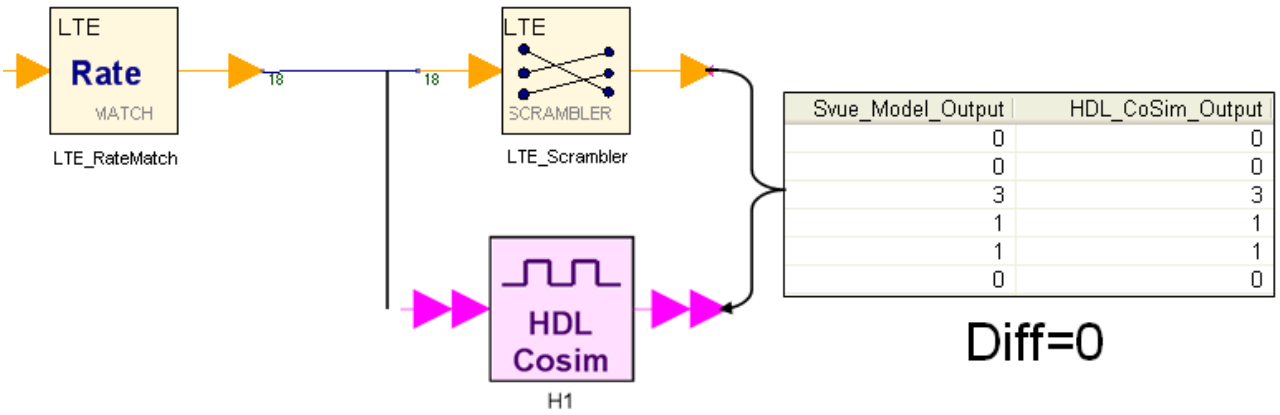

Figure 5. Using HDL co-simulation in SystemVue for debugging.

Debugging a Discrepancy, continued

Items 1 and 2 can be resolved by editing either the HDL code or modifying the SystemVue simulation model. This typically requires further visibility into the SystemVue model source code to determine what assumptions have been made in interpreting the standard and creating the simulation model.

The SystemVue LTE Baseband Exploration Library (BEL) utilizes switchable models, so that users can quick view and edit the underlying MathLang source code as shown in Figure 6.

This enables the source code to be easily evaluated and compared to the HDL code. Once the source of the discrepancy has been identified, corrective action can be taken to either edit the HDL code or edit the SystemVue source code to resolve the discrepancy.

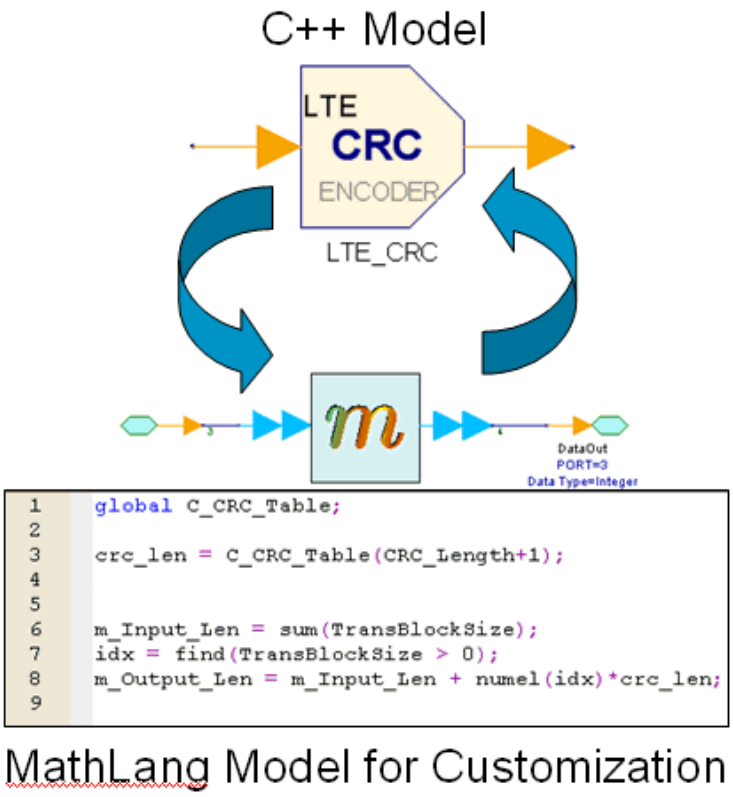

Figure 6. Customization of the SystemVue simulation model.

Summary

This white paper outlines an improved approach to facilitate rapid development and testing of LTE FPGA-based physical-layer coding/decoding implementations by using the SystemVue LTE Baseband Exploration Library to generate reference test vectors. For further information, please visit www.agilent.com/find/eesof-systemvue, or contact your local Agilent Technologies representative.

For more information about Agilent EEsof EDA, visit:
www.keysight.com/find/eesof

myKeysight

myKeysight

www.keysight.com/find/mykeysight
A personalized view into the information most relevant to you.

Keysight Channel Partners
www.keysight.com/find/channelpartners
Get the best of both worlds: Keysight's measurement expertise and product breadth, combined with channel partner convenience.

www.agilent.com/find/eesof-systemvue

For more information on Keysight Technologies' products, applications or services, please contact your local Keysight office. The complete list is available at:
www.keysight.com/find/contactus

Americas

Canada	(877) 894 4414
Brazil	55 11 3351 7010
Mexico	001 800 254 2440
United States	(800) 829 4444

Asia Pacific

Australia	1 800 629 485
China	800 810 0189
Hong Kong	800 938 693
India	1 800 112 929
Japan	0120 (421) 345
Korea	080 769 0800
Malaysia	1 800 888 848
Singapore	1 800 375 8100
Taiwan	0800 047 866
Other AP Countries	(65) 6375 8100

Europe & Middle East

Austria	0800 001122
Belgium	0800 58580
Finland	0800 523252
France	0805 980333
Germany	0800 6270999
Ireland	1800 832700
Israel	1 809 343051
Italy	800 599100
Luxembourg	+32 800 58580
Netherlands	0800 0233200
Russia	8800 5009286
Spain	800 000154
Sweden	0200 882255
Switzerland	0800 805353 Opt. 1 (DE) Opt. 2 (FR) Opt. 3 (IT)
United Kingdom	0800 0260637

For other unlisted countries:
www.keysight.com/find/contactus
(BP-09-23-14)

KEYSIGHT TECHNOLOGIES

Published in USA, January 20, 2015
5990-3357EN
www.keysight.com