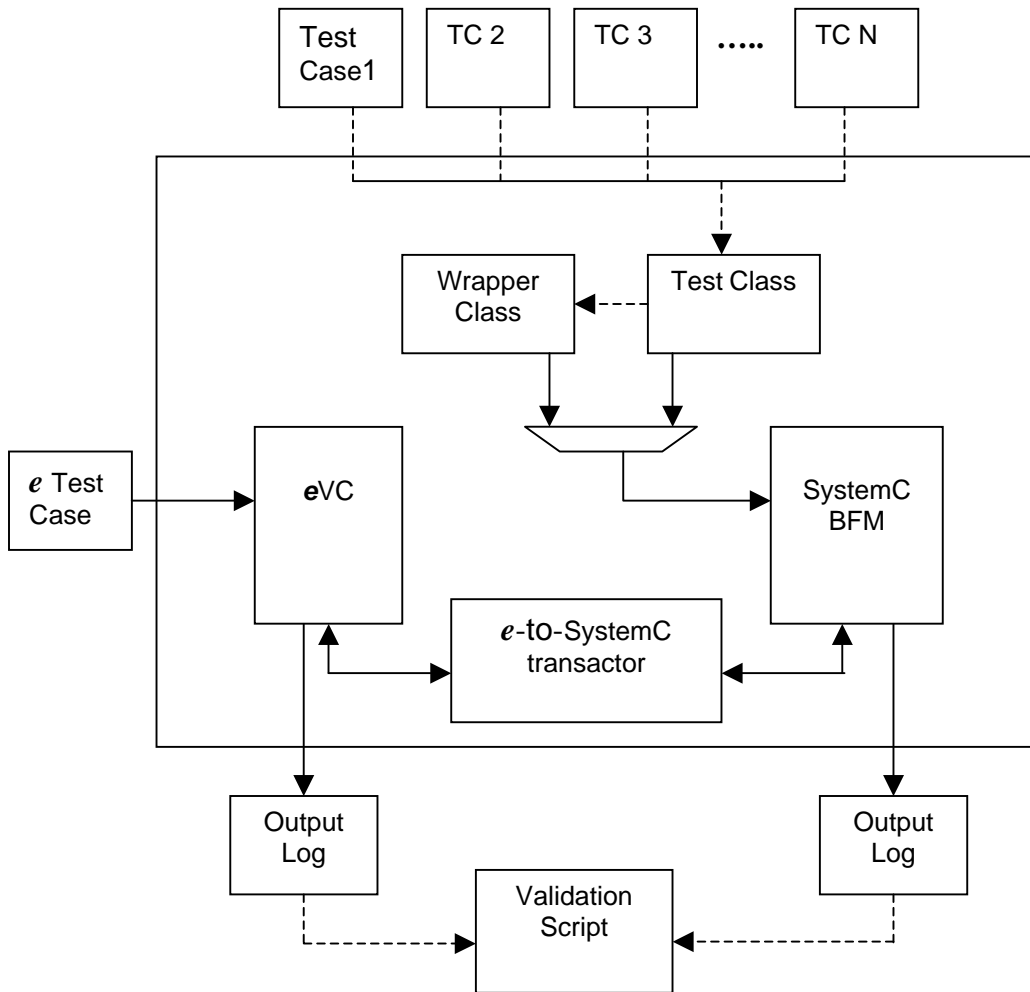


## PCI Express Endpoint *e*VC Verification

PCI Express, the 3<sup>rd</sup> generation I/O interface is an advanced serial version of conventional PCI/PCI-X. It is a high-speed serial bus that is expected to meet the future requirements of faster processors and advanced applications. It can support upto 80 Gbps data rate as per the current standard. PCI Express has a layered architecture containing Physical, Link and Transaction layers. PCI Express endpoint *e*VC is a ready-made, highly configurable *e* verification environment suitable for a PCI Express endpoint device DUT verification. The *e*VC can be configured for Upstream/Downstream component verification. The PCI Express endpoint *e*VC was verified against a SystemC BFM. Verification environment used is shown below.



The verification environment can be configured for Upstream/Downstream component checking. Various possible error conditions are simulated to verify *e*VC checkers. Bi-directional data transfer is possible through transactor. The transactor, written in SystemC, writes interface methods (PLI) between *e* language and SystemC language. On each clock cycle, the *e*VC and BFM interacts to each other through transactor. All test cases inherit the test class to write specific test conditions, which does basic configuration of SystemC BFM and verification environment as per requirement. It

interacts with BFM and injects stimuli as per the test case written. A wrapper class that combines multiple test cases using dynamic binding is used and fires them one by one to make regression testing. In this process, the wrapper class saves memory using dynamic binding and saves simulation time by avoiding re-initialization of model for each test condition. Self-checking test cases are written on top of CBFM to achieve more than 98% of coverage. Shell/Perl scripts are used for output validity. Makefile is used to run the test environment.

The verification environment is based on reusable methodology. It is designed and implemented using SystemC. Regression testing is carried out using dynamic binding. The verification environment supports extensive API for test case writing. Verification environment architecture is interoperable, object oriented and modular.

### **Conclusion:**

This methodology was selected to take benefits of both of the domains, *e* and SystemC. Modular architecture made layer-wise and system level verification possible using the same verification environment. It also proved that Specman Elite could be used to verify SystemC based designs too in an efficient way. Powerful features of C++ such as dynamic binding made the regression testing faster.