RASSP Definition and Role of Executable Requirement (ER-SPEC) and Design (ED-SPEC) Specifications Appnote

Abstract

An executable requirement specification communicates a detailed, unambiguous set of customer requirements to the contractor(s) who will bid on or fabricate the hardware/software elements of a project. Execution of the ER-spec corresponds to evaluation of conformance of a design to the requirements contained in the ER-spec. An executable design specification defines how project-specific design information is to be evaluated. Execution of the ED-spec corresponds to assignment of project parameter values which will be measured against the requirements in an ER-spec. This note describes the contents of hierarchical ER-specs and ED-specs and demonstrates how they capture and expresses both customer intent and detailed project requirements.

Purpose

This application note provides definitions for the RASSP view of executable requirements and specifications. It also describes the role each should play in the development of a new system and/or product. Benefits are described and a process is established for the use of these concepts. Though the LM/ATL RASSP program did not spend alot of time refining and using this technology, it matches well with the RASSP concept of a risk driven hierarchical virtual prototyping process. This work has also be complemented by the work done on the CEENSS program. A Case Study [CEENSS]reference document from this program has been incorporated in the reference section of this application note.

Roadmap

RASSP Definition and Role of Executable Requirement (ER-SPEC) and Design (ED-SPEC) Specifications Application Note

1.0 Introduction

Projects begin with a statement of requirements which describes values for selected physical properties and how the product must perform. At a minimum, system level requirements are supplied, but requirements that apply at lower packaging levels may also be included. Requirements are used to develop preliminary specifications at the outset of a project and then detailed specifications as the system hardware/software components and human factors evolve. The conventional approach to requirement statement has been a written document. This is typically generated by a customer and interpreted by the contractor. The interpretation process has been shown to be prone to errors. As a consequence, written requirements are being replaced by executable requirements and specifications. This application note describes the contents and usage of these executable elements in a digital systems design environment as developed under the Rapid Prototyping of Application-Specific Signal Processors (RASSP) program. This program relies heavily upon virtual prototyping with VHDL component modeling and simulation.

This document is intended for system engineers and design engineers developing signal processing systems using the RASSP methodology. Though RASSP concentrates mainly in signal processing design its concepts as well as the concepts presented here can be easily extended into the digital systems design environment. It defines how executable requirement specifications are generated and supplied to contractors and bidders for a RASSP project and how conformance to these requirements is established by use of executable design specifications. Contents of executable requirement and design specifications are examined to show the relationships of critical "documentation" components.

The process of distributing of requirements among all project phases from design through manufacturing are covered.

Three significant terms will be used in the text of this Application Note :

- Written Requirement Statement (WRS)
- Executable Requirement Specification (ER-spec)
- Executable Design Specification (ED-spec)

Some of these terms have been used previously in the literature and have received somewhat different definitions. To avoid semantic confusion, these terms will have the following meanings in this document :

Written Requirement Statement

- A detailed written (The term "written" means text or an equivalent electronic representation of the text material such as a database file set) statement set expressed in a natural language which attaches required quantitative values to selected system parameters and expresses qualitative factors such as goals, usage intent or critical influences such as cost (phase or life cycle) which a customer has determined to be significant for a particular project. Indeterminate values (TBDs) are also appropriately included in the Written Requirement Statement for place-holding requirement assignments.

(The term "written" means text or an equivalent electronic representation of the text material such as a database file set)

Executable Requirement Specification (ER-spec)

- A set of **conditional mathematical expressions** which attach measurable quantities and possibly acceptable/required measurement schemes to system parameters. The ER-spec is executable in the sense that a PASS/FAIL value or a quantitative rating factor will be associated with each parameter during system testing. Conformance of all system parameters to listed required values will be tested.

Executable Design Specification (ED-spec)

- A set of **values** or **computational methods** which expresses a specific **instance** of a consistent relationship among the parameters included in the executable requirement. The ED-spec is a set of **relationships among components** for a contractor-specific implementation of a system. It includes the effects of structure-dependent properties of the measurable quantities included in the ER-spec. An ED-spec is attached to each RASSP packaging level. The system is referred to as level 0. Other levels are assigned integer values which increase toward more detailed packaging levels. A level 0 ED-spec (ED-spec0) is a special case used to demonstrate that an ER-spec serves as the basis of a system which can be built.

The relationship among these entities is shown in Figure 1 - 1.

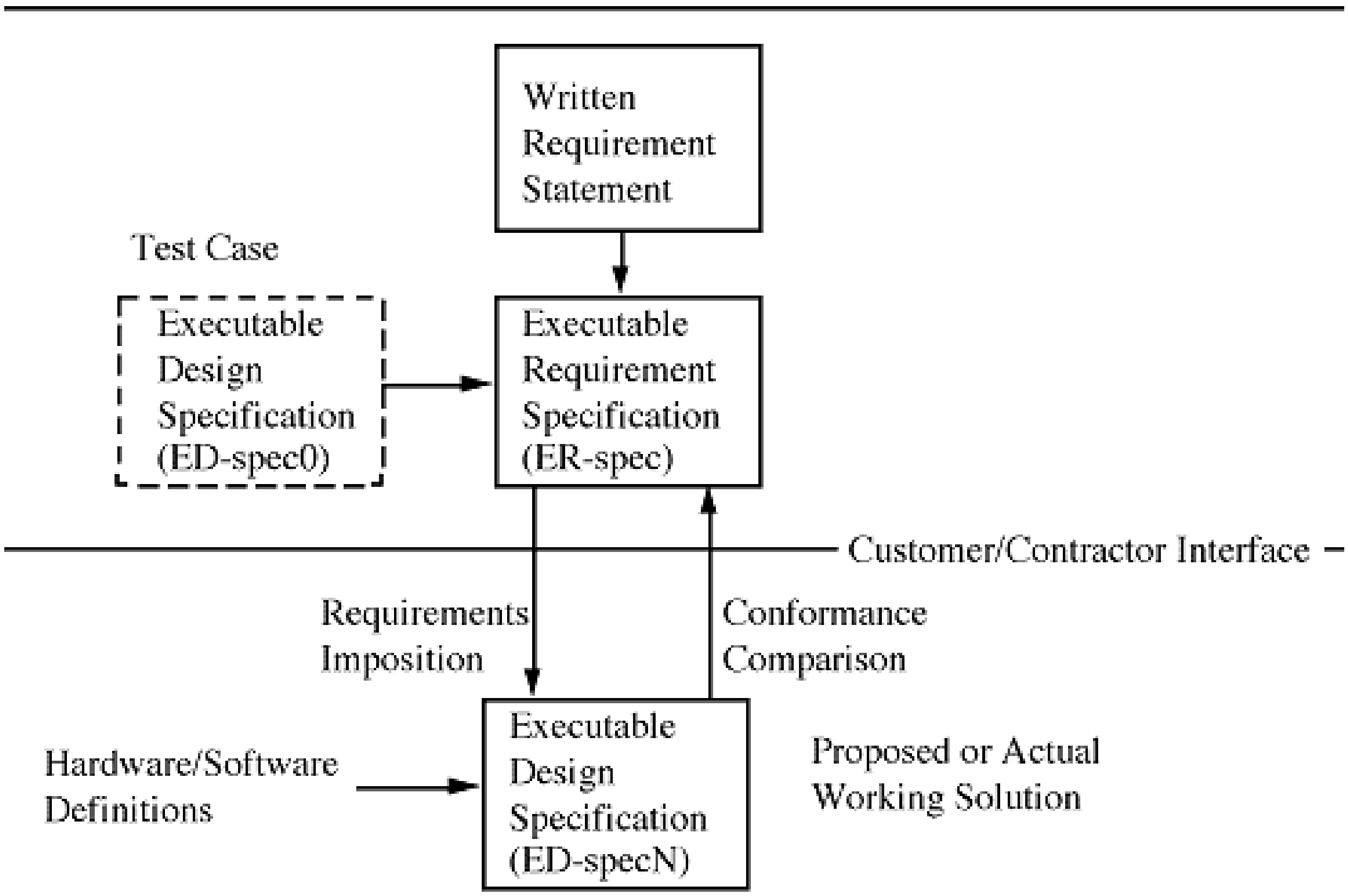

Written Requirement Statement
Test Case
Executable Design Specification (ED-spec0)
Executable Requirement Specification (ER-spec)
Customer/Contractor Interface
Requirements Imposition
Conformance Comparison
Hardware/Software Definitions
Executable Design Specification (ED-specN)
Proposed or Actual Working Solution

Figure 1 - 1: Relationships Among Requirement and Design Specifications

The executable requirement specification is always supplemented with the human readable Written Requirement Statement which serves to "comment" the executable requirements in the manner in which "comments" are used in computer programs to explain source code. For this reason, **the term executable requirement specification can refer to the combination of the formal ER-spec and a written requirements statement.** The sections below describe the concepts of executable requirements and specifications and their roles in a RASSP project. First a methodology description is presented. Requirements, a rationale for introducing ER-specs into RASSP, and the constituents of executable requirement and design specifications are discussed. Then an examination of the RASSP Benchmark 4 (SAIP) highlighting requirements and defining executable requirements and specifications is performed to illustrate a current attempt at an ER-spec for an ongoing RASSP project.

Next Up Previous Contents

Next: 2 Methodology Description **Up:** Appnotes Index **Previous:**Appnote ERED Index

RASSP Definition and Role of Executable Requirement (ER-SPEC) and Design (ED-SPEC) Specifications Application Note

2.0 Methodology Description

2.1 Requirements and Their Properties

A **requirement** is a performance or physical need established by a customer which is attached to a procured hardware/software system to ensure that the acquired system meets a particular criterion. For RASSP, target systems are signal processing systems which consist of hardware and software components. Typically, many computational algorithm-based requirements are set for a given system.

Many types of requirements can exist for a given project. They can establish minimum technical performance standards, component attributes, goals, conformance to existing generic specifications, the need for project-specific deliverables such as prototypes and documentation, and cost or time limitations. The relative importance of requirements is determined by the customer according to the type of project. For example, a weight limit would be a significant requirement for an airborne equipment, or a minimum battery amp-hour rating would be an important requirement for equipment intended for unserviceable operation. Regardless of their type, requirements must pass certain tests before they can be attached to a system. The kind of test depends upon the system characteristic to which the requirement is assigned.

2.1.1 Quantitative or Qualitative Requirements Classification

Quantitative requirements are requirements which can be expressed in terms of a measurable numerical value are called quantitative, e.g., signal-to noise ratio shall be greater than 20 dB.

Qualitative requirements are requirements which state system needs or goals which cannot be a priori assigned a value, e.g., design shall be reproducible by any competent manufacturer, or, design shall include a self-test capability which will test functionality of each circuit board separately. A special case is the merit requirement which defines a set of rules by which a numerical quantity (figure of merit - FOM) is to be assigned to some combination of quality factors.

2.1.2 Measurable Requirements

All requirements must be measured according to three evaluation criteria before they can be associated with a real system. In particular, requirements must be realizable, consistent, and valid.

- Realizable requirement are achievable and measurable with current simulation and test technology.
- Consistent requirements are derivable from higher packaging (Level refers to the RASSP hierarchy of packaging : System, Subsystem, Board, MCM, Chip. Requirements set for any given level must support established requirements for the next higher level of packaging) level requirements and do not contradict other requirements at their own or higher levels.
- Valid requirements are predictable, verifiable by simulation, and measurable.

The importance of these properties was established in the RASSP Design For Testability (DFT) Methodology V1.0 document, which addressed requirements independently of their format (executable or otherwise). Each proposed requirement must meet the three criteria for it to be included as any part of a requirement document. Satisfaction of these criteria ensures that tests will be available to validate conformance to requirements established at the outset of a project.

2.1.3 Requirement Source

A third class of requirement characteristics is the requirement source. A complete set of verifiable requirements includes customer or conformance requirements based upon system characteristics (operational and physical) and derived requirements (not explicitly stated by the customer) necessary to fabricate reliable, manufacturable equipment. Conformance requirements consist of the typical explicit requirement list that is given to the contractor(s). Derived requirements are not (necessarily) explicitly supplied but are nonetheless "understood" and based upon the experiences of contractors in the building of similar products. The importance of derived requirements is not diminished, however, since they always must be applied to produce working, reliable, fieldable equipment. Both conformance and derived requirements need test means applied in a manner to demonstrate that project goals will be achieved. Conformance requirements are usually tested during product qualification testing. Many derived requirements are tested during simulation exercises and quality control evaluation of production facilities.

Requirement specifications are the "documents" that a contractor uses to define system requirements. One example is the classic MIL-Spec which details what conformance to selected standards means and how it is measured. The definitions are generic and are to be applied to any proposed hardware/software implementation. Such specifications are assigned by the customer at the start of the contract. Design specifications quantify how the collection of specific system components will meet the requirement targets.

2.2 Rationale For Executable Requirement and Specification Methodology

The following sections provide rationale for using executable requirement and design specifications.

2.2.1 Executable documents support methodology automation at all packaging levels

Requirements are statements of the customer's needs which form the basis for a system design. In the any design methodology, requirements are first developed at system level and then these system level requirements are selectively flowed down to lower levels of packaging. In the RASSP methodology, flow down is implemented during application of a top-down spiral development model. Requirements established at high levels are adapted and consistently applied at lower packaging levels as a design and its implementation develop to ensure that all components of a system support the system level requirements defined early in the project life cycle. Consistency with the use of a managed set of tools to automate system development necessitates that automation also be applied to the specification and measurement of requirements at all levels. Thus a rationale for adoption of an executable requirement as a standard for requirements delivery is established.

RASSP methodology uses VHDL-based simulation to design and evaluate signal processors prior to commitment to hardware. The goal of first pass success is supported by many tools which produce and analyze virtual hardware designs prior to hardware fabrication. With tool-based hardware/software codesign, requirements can be introduced into the RASSP methodology effectively by using an executable requirement. This will not only support requirement conformance measurement during simulations but also after physical hardware fabrication.

2.2.2 Executable Requirements and Design Specification Independence from the Physical System

Use of separate executable requirement and design specifications allows independence between requirements and the physical system that proposes to conform to the requirements. There is considerable merit to the concept of separation between requirement and design specifications based on the separation on **what is to be done** and **how to do it**. Independent requirements allow definition of a problem without a forced solution. The problem **definition** is formed by those who understand the problem best and solutions are proposed by those who understand hardware/software system solutions best.

2.2.3 Removal of Ambiguity

Executable documents control ambiguity by removing opportunities for interpretation of requirements and specifications. It is critical that requirement specifications be correct and unambiguous at all packaging levels, starting at the system level. Both of these qualities are enhanced with the use of an executable requirement. Correctness ensures that the requirements imposed are appropriate to the intended system mission goals. Absence of ambiguity ensures that requirement flow down through the system hierarchy will be achieved without confusion of function or performance at any packaging level. A correct executable requirement established at system level supports correctness at lower levels by maximizing measurability of system parameters. The quantitative aspect of the ER-spec is effective in removing ambiguity as a design progresses toward lower packaging levels.

The **executable requirement specification** is a preferred format for requirements which can be transmitted to the contractor(s) for both bidding and fabrication purposes. Interpretation of the requirements is minimal since the ER-spec establishes a standard basis for validation by anyone with appropriate testing means. Furthermore, the ER-spec can be backed up with a test case executable design specification (ED-spec) which serves as a reference to establish that the ER-spec has a solution, that is, it defines a realizable system. This generality represents significantly improved ambiguity control relative to conventional written requirement sets which may be open to interpretation.

2.2.4 Permanent Records Provide Life-Cycle Support

Executable documents produce permanent records which support modifications and updates which occur during model year upgrades.

Another driver for the use of executable requirements and specifications is the compatibility of these formats with reuse in the RASSP model year architecture concept. Projects maintain currency with technology by admitting updates to the system components as technology advances. There is no obsolescence caused by rigid commitment to the technology in existence at the time of project inception. The ER-spec and ED-spec provide a complete definitive reference to previous work which can be readily updated to accept new or additional component characteristics while preserving the bulk of the work which has been previously verified. The need to create new requirements and assign new specifications is minimized and advancement of capabilities for a system becomes an evolutionary process rather than a new design. The economic benefits for this approach are significant since only new elements of the design must be validated.

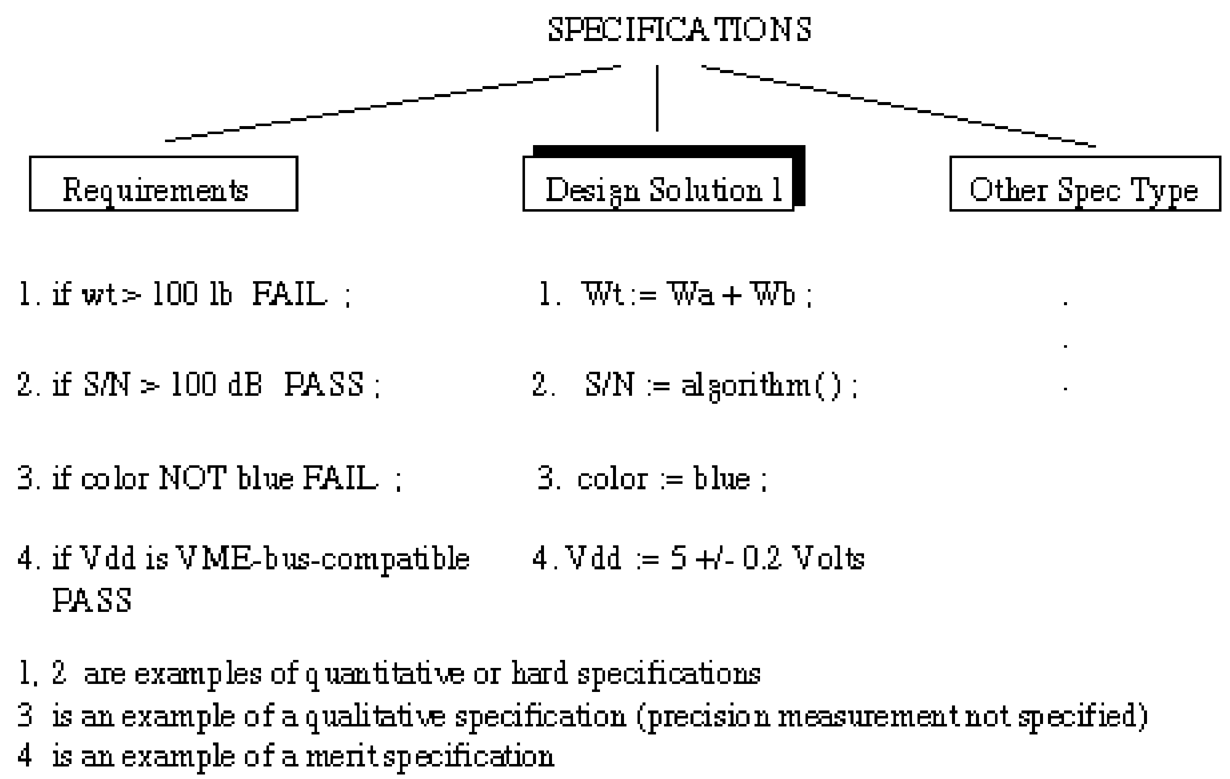
2.2.5 Readily Distributed Standalone References

Executable requirement and design specifications are standalone references which are readily distributed. Executable requirement and design specifications are intended to replace conventional written-only requirements and the typical SOW. They offer completeness and a better definition of the requirement and design specifications. Before they are accepted as replacements, however, they must offer better accessibility to the information contained within them, and they must be genuinely useful. They must be human readable and/or executable by personnel with a wide range of differing project interests. Specifically, **no extensive training to access** the documents should be required. (Personnel executing documentation are assumed to have required special training.) Also, there **must be value within the documents** if they are to be continually used as primary references throughout a project's life cycle. Requirements entered into a data base which is difficult to access or not available everywhere it is needed are likely to remain unused or ignored. Otherwise, it is likely that some conventional form of documentation will be substituted.

Accessibility must be accompanied by ease of distribution and update. The electronic storage nature of ER-spec components, files, etc., supports electronic distribution from a central source. If used properly, a single source for requirements can limit references to outdated documentation. This is a common problem with large projects and paper documentation where design/fabrication/integration are carried out in separated geographical regions. On line availability and version control are available to manage distribution for executable documentation.

2.3 Relationship Between Executable Requirement and Design Specifications

Executable requirement and design specifications are **instances of specifications** in the same sense that objects are instances of the classes of an object oriented design methodology. This relationship is illustrated in Figure 2 - 1.



```
1. if wt > 100 lb  FAIL ;            1.  Wt := Wa + Wb ;              .
                                                                      .
2. if S/N > 100 dB  PASS ;           2.   S/N := algorithm() ;        .

3. if color NOT blue FAIL ;          3.  color := blue ;

4. if Vdd is VME-bus-compatible      4. Vdd := 5 +/- 0.2 Volts
   PASS
```

1, 2 are examples of quantitative or hard specifications
3 is an example of a qualitative specification (precision measurement not specified)
4 is an example of a merit specification

Figure 2 - 1: Executable Specification Class With Requirement and Design Instances

The requirement set consists of a set of executable statements that express the customer desired needs of a project in terms of its characteristics. Included are quantitative requirements such as weight and signal-to-noise ratio as well as qualitative requirements such as ease of use. Both quantitative and qualitative requirements are conveniently expressed as conditional statements that are easy to evaluate. In some cases, qualitative statements may be preprocessed to yield a figure of merit (FOM) value before they are tested by a conditional. The FOM is typically an accumulated sum obtained from a qualitative/semi-quantitative evaluation of a specific set of system parameters. Such preprocessing is appropriate for comparing the relative effectiveness of different design solutions. Both requirement types are conveniently written as pseudocode statements which are readily converted to any desired language suited to a specific application. Requirements are generic and do not necessarily suggest a design solution.

The design specifications are shown in Design Solution N boxes. There can and will likely be more than one design solution, especially during the bidding phase of a project. Each set of design specifications labeled 0-N consists of a set of pseudocode statements that indicate how a proposed solution will conform to the requirement it supports. The example case shown in Figure 2 indicates that the system weight W will be the sum of the weights of two subsystems, Wa and Wb. The example solution also states that S/N shall be calculated as the results of a call to the function representing an algorithm which is exercised during simulation. The results of both calculations along with others can be supplied as inputs during execution of the requirement statements to determine whether the requirement is satisfied. The requirements in the ER-spec are specified as conditional expressions and if results for a proposed system design instance (represented by an ED-spec) are computed according to the rules in the ED-spec and entered into the conditional expression as compare parameters, execution of the ER-spec can generate an evaluation list of satisfied and non-satisfied requirements. Only those items which are true requirements are included in the ER-spec code. Peripheral issues such as additional capabilities or conditions are not included as requirement specifications. The ER-spec can serve as a testbench for the requirement set in the manner shown in Fig. 2 - 2. The ER-spec is the stimulus for the test (ED-spec) and the response of the proposed solution to the stimulus is measured in the ER-spec execution.

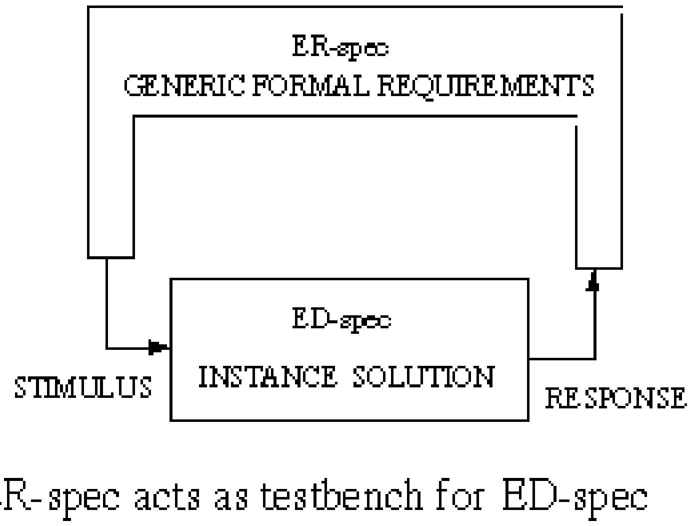

ER-spec
GENERIC FORMAL REQUIREMENTS
ED-spec
INSTANCE SOLUTION
STIMULUS
RESPONSE
ER-spec acts as testbench for ED-spec

Figure 2 - 2: ER-spec to ED-spec Relationship

In addition to the requirement and design specifications, the box Other Spec Type is included to illustrate extensibility of the **specification** concept. The sections below will explore in more detail the range of contents that can be included/used as parts of executable requirements and design specifications.

2.4 Constituents of An Executable Requirement Specification

As delivered to the contractor, a complete executable requirement specification consists of :

- the Written Requirement Statement (WRS)
- the formal executable requirement specification consisting of supporting data and/or executable files which are to be used during execution of the ER-spec by the contractor

The content of these requirement documents is subject to a great deal of product-dependent variability. For large systems, both can be quite extensive in detailing a long list of requirements. For smaller systems, the content can be significantly less. Because of the wide range of systems covered, a preferred approach to the design of an ER-spec is to define a set of **core components** and to allow enhancement of this core set according to the needs of a specific project. Use of a core set of components allows valuable standardization which can be applied from project to project. Users of any executable requirement could expect critical components to be presented in a known format using common language. The discussions below will identify elements which should be considered as core elements of an ER-spec.

For both RASSP and non-RASSP projects, the basic Written Requirement Statement (WRS), functionally equivalent to the classic requirements list containing a Statement Of Work (SOW) for a project, is typically created by the customer. It is often developed in stages through a refinement process. Initially, the contractor will likely contribute little, if any, contents during its drafting, unless the Statement is prepared as a result of a contractor's unsolicited proposal for a project and the customer agrees to use the contractor's proposal as the basis for a procurement. The WRS will be the first expression of system requirements and will likely be updated as needed to become a reference for the **current** quantitative and qualitative requirement descriptions for a project. It will contain as core elements technical details of project hardware, software, and simulation code descriptions including the execution sequence of code supplied with the executable requirement. Directions for compiling supplied source code will also be included. Another core element of the WRS is the expression of the intent of the customer in selected areas where definitive specifications may not yet be assignable. "TBD" items are frequently present in requirements statements. The basic Written Requirement Statement will contain a Consolidated Test Requirement document generated for the testability analysis and a CAD tool based requirements database file, i.e. RDD-100 or RTM, which is developed by system engineering.

The formal executable requirement specification will contain mathematical representations of the written requirements. Its **core elements**will typically be processing algorithm definitions and expected performance capabilities during execution of these algorithms for test cases, physical characteristics, and component models and testbenches for environments in which components are to be tested. Other representations such as detailed performance level vs. operational mode specifications can be substituted as an alternative. Execution of the ER-spec corresponds to demonstrating the **results expected** of the project hardware/software suite. Satisfaction of requirements for a project can then be validated by measuring the quantities assigned to system properties and comparing them to the ER-spec values. The formal ER-spec can be generated solely by the customer or by interaction with contractor(s) actually (post-award) or potentially (pre-award) involved with the project. In either case, it represents a standard against which conformance can be unambiguously established.

Typical core executable components of an ER-spec are :

- code which when executed demonstrates expected system or component performance for a set of values assigned to system parameters, e.g., initialized high level representations of algorithms. The purpose of this code is to generate reproducible results which define performance and the effects of parameter variations. The code itself is not necessarily a definitive representation of the implementation of the process it represents.
- software which will execute on the project hardware during normal system operation
- simulation testbench for verifying performance of a design
- structural and behavioral models of system components such as logic devices or memories
- structural and behavioral models of test-related support components such as FPGA-based memory BIST subsystems
- FPGA designs and/or synthesizable models of FPGA circuitry

Support components for an ER-spec for a signal processor can assume existence of data files in specific formats. Where these are necessary to the core executable components, they become core components also. Typical data sets will consist of :

- test program sets (TPS) consisting of test vectors and instructions/expected results
- operational test input and result sets for demonstration of operation in selected scenarios (test cases)
- drawing sets (CAD mechanical and schematic) and parts lists

The ER-spec is typically supported by a customer-generated reference system-level executable design specification (ED-spec0), which is **one solution instance**of the executable requirement specification that serves the purpose of proving by example that the ER-spec will admit a solution to the signal processing problem being addressed. The contractor will develop a hardware/software-specific executable generic design specification (ED-spec0) to provide a **working solution instance**. ED-spec0 is not intended to be hardware or software definitive. This is done in order to allow design freedom to contractors. It is Also used primarily as a consistency and realizability check for the executable requirement specification. ED-spec0 is developed by the customer as part of ER-spec generation process, and, depending on circumstances, may or may not be supplied to the contractor(s).

2.5 Customer-Generated Executable Design Specification Level Zero (ED-spec0)

As mentioned above, it is essential that the requirements assigned to a project are realizable, i.e., they represent a system which can be built, and consistent, i.e., there exists no conflict among the separate requirements. It cannot be taken for granted that any system can be practically made since the complexity of systems admits the possibility of many combinations of factors of which the system designer may be unaware. Likewise, complexity dictates that the effects of individual requirements upon others be examined.

Use of a reference executable design specification (ED-spec0) offers control over potential problems arising during the requirements generating phase of a project by serving as a reference solution to the requirements set. For signal processing systems, for example, ED-spec0 could contain a high level source code implementation of an algorithm which will accept a set of behavioral system parameter values written into the requirements. If the parameters are realizable for a real system, and the code demonstrates that use of these values produces a required performance level, then ED-spec0 serves as proof that the desired performance is achievable with at least one set of system parameters. Requirement practicality is thus established.

Since its major purpose is proof of concept, the ED-spec0 may or may not be supplied to the contractor(s). It can serve as a means of clarifying concern or explaining details in some cases but it is not intended to recommend preferred approaches to contractors. It is constructed by the creators of the requirement documentation and is not a replacement for the system definition task normally performed by the contractor. This is the role of the contractor executable design specifications (ED-spec).

2.6 The Contractor Executable Design Specification (ED-specN)

The collection of system specific parameter computation formulas which defines the relationships among the contributors to a requirement value is called an executable design specification. An ED-spec is a complete project- and packaging-level-specific executable document which associates a particular design with an executable requirement. The relationship of ER-spec and ED-spec can be illustrated with an example of how one requirement is handled. Consider that a requirement in the ER-spec could demand that total power for a system be less than 100 Watts. The stated value is correct by assumption, unambiguous and applies at the system level. What remains lacking before the requirement can be associated with some version of the system, however, is a reference as to how conformance to the requirement is to be verified. Suppose this particular system consists of three subsystems. In such a case a reasonable verification method is to measure and sum the powers consumed by the subsystems and then compare the sum with the value 100 Watts. The expression PTOTAL = PSS1 + PSS2 + PSS3 completely defines how the system power is to be computed. The formula for PTOTAL becomes a **specification** of **how** the system power consumption is to be calculated.

The formal executable design specification is constructed by converting generic mathematical statements from the ER-spec into system-specific mathematical expressions which can be evaluated. The result will typically take the form of compilable source code (HLL such as C, C++, JAVA, Matlab, or VHDL) which expresses desired relationships among system parameter values. The optimum base format for executable requirement statements is yet to be determined but work in the area of formal methods suggests that Boolean expressions are equivalent to formal specifications (This conclusion is proposed in "Specifications, Programs, and Total Correctness" by Eric Hehner, which gives an indication of the type of effort occurring in this area. Others have advanced predicate statements and mathematical expressions.) Boolean expressions also appeal on the non-theoretical level and make conversion from a natural language like English relatively straightforward.

An ED-spec is executable in the sense that it defines an executable procedure for calculating values for system properties which will be compared to system requirements. The procedure will typically consist of a collection of high level language compilable and executable files for computing system performance characteristics. For the power example above, the (Boolean) expression :

```
if (PTOTAL < 100) PASS else FAIL ;
```

could express the result of an executable test. The ED-spec files can be standalone expressions of relationships or simulations which can be extrapolated for use during system design. The latter case will become more prevalent when reuse becomes more integrated with the design process. For signal processors, execution will typically take the form of running a sequence of programs with parameter inputs selected by the contractor to represent his version of the system. Unless dictated by the customer (unusual circumstance), the ED-spec is assembled by the contractor who is responsible for details of hardware fabrication and software development. Typically, the methods for calculating system properties are supplied by the contractor to the customer as part of a proposal or design review package.

2.7 Procedures for Creating Executable Requirements and Design Specifications

To date, no universal step by step procedure which will lead to an ER-spec and ED-spec for the full range of possible signal processing projects has been found. The same can be said for generation of **non-executable** requirement statements. However, experience with projects performed under RASSP and tool-based requirement generation for other projects shows that a common approach based on development of a database which contains a hierarchy of requirements based upon packaging levels is workable. Results from application of this approach in particular cases has differed according to the tools applied.

The discussion below describes how **executable** requirements and specifications can be created and used within the RASSP methodology at all levels of packaging. The intent is to show the value of the ER-spec and ED-spec and how their distribution among packaging levels provides control of a project throughout its life cycle. The target system is a generic RASSP signal processor to be built by a single contractor. Two bidders will be assumed to illustrate the role of the ER-spec in the competition process. This section should be compared with the **Application Example** described in the SAIP Benchmark 4 case study to contrast recommendations with an actual application that used an ER-spec in a **limited** manner. The purpose of the present discussion is to demonstrate the range of applicability of the ER-spec and ED-spec to **all phases** of a project.

A project begins with a system definition and a generation of a set of system requirements. The Written Requirement Statement is the first document generated by the customer. When this has been refined to a sufficient level of detail, the first version of the ER-spec is created. Refinement continues and an ED-spec0 is developed by the customer to validate the parameter values which populate the ER-spec. The ED-spec0 becomes a proof of concept when it substantiates the system parameter values in the ER-spec. When this occurs, the ER-spec becomes available for formal release to the contractors bidding on the job.

When the contractors receive the bid package, each responds with a system design (ED-spec) which attempts to satisfy requirements received in the ER-spec (including the Written Requirement Statement). Each potential contractor has the opportunity to demonstrate their ability to satisfy the requirements of the system by creating an ED-spec specialized to their respective proposed design. The ED-specs and supplemental material are supplied to the customer, who awards a contract following possible discussions with the bidders. ED-specs at this time include system level specifications and present architecture level details only as required to indicate a likely hardware/software implementation of the system. Extensive architecture trades cannot be presumed at this point.

The successful bidder now begins the system analysis phase which leads to a complete system level specification. The ER-spec and ED-spec are updated and made more complete as system definition moves to the architecture selection and verification phases. During these phases, a RASSP-designed system will produce its virtual prototype along with its variants which represent architecture trades which can be used to verify conformance with requirements. The architecture level VP will consist of behavioral VHDL models and other high level code which will be tested on a system testbench. Execution of the testbench code along with execution of the ER-spec and ED-spec demonstrate whether the system will conform to the requirements.

As the system is broken down into subsystems, boards, etc., requirements assigned at the system level must be interpreted and allocated to the more detailed lower packaging levels. A corresponding level specific ED-spec development parallels generation of the levels of ER-specs. Each level must be assigned its own specific ER-spec (ED-spec) which must be consistent with and be a contributor to the ER-spec of the level above it. Each ER-spec is executable and contains an accompanying Written Requirement Statement like the system ER-spec. The ER-specs become more detailed as the packaging level proceeds toward board and/or MCM level where fabrication will occur. The process of generating ER-specs and ED-specs for lower packaging levels is a **flow down process** similar to software functional decomposition. A similar consistency requirement exists also since each lower level requirement is a contributor to a higher level requirement and each high level requirement is made up from a collection of lower level requirements.

Several tool-specific commercial alternatives have been used for distributing requirements among the different packaging levels for a project. The RDD-100 data base tool by Ascent Logic has been used to capture and distribute requirements for the RASSP benchmark projects. RTM (Requirements and traceability Management) by Integrated Chipware is another tool which formulates requirements as objects and manages them as elements of a data base. It has been used by Lockheed Martin on the F22 Program to provide requirement traceability and flowdown. **Neither of these tools creates executable requirements or specifications**, and both require training for the user and development of a degree of skill to make effective use of the requirement management features of the products. This skill must be acquired by all project team members if requirements are to be distributed and readily accessible to concerned parties working at all packaging levels. Acquisition of this skill base can represent a considerable training investment dedicated to a specific tool. In addition, use of specific commercial tools for requirement distribution presumes **accessibility** and continued vendor **support** for the tool used by any current or future activity (e.g., model year update by another vendor) using the ER-spec.

An alternative to strong coupling to a specialized requirement and design specification management tool is use of a general purpose multiple-spreadsheet-based application. In the area of test, RASSP used test strategy diagrams (TSDs) implemented with EXCEL workbooks and worksheets to describe the flow down of test requirements through the packaging hierarchy. This approach uses an "open" , relatively simple, environment to flow testing requirements to all packaging levels. Data entry is natural language based and the level of skill required for linking requirements hierarchically is moderate and widespread. Success of the multilevel TSD structure suggests that such an approach could serve as a model for a requirements hierarchy. Both Written and executable requirements can be accommodated in this scheme. Reference to an executable file is included in the same manner as a written statement. A consequence of the simplicity of this scheme is that no storage intelligence exists, that is, what a viewer sees is exactly and only what was entered.

An example of a coupled worksheet hierarchy of requirements is shown in Figure 2 - 3.

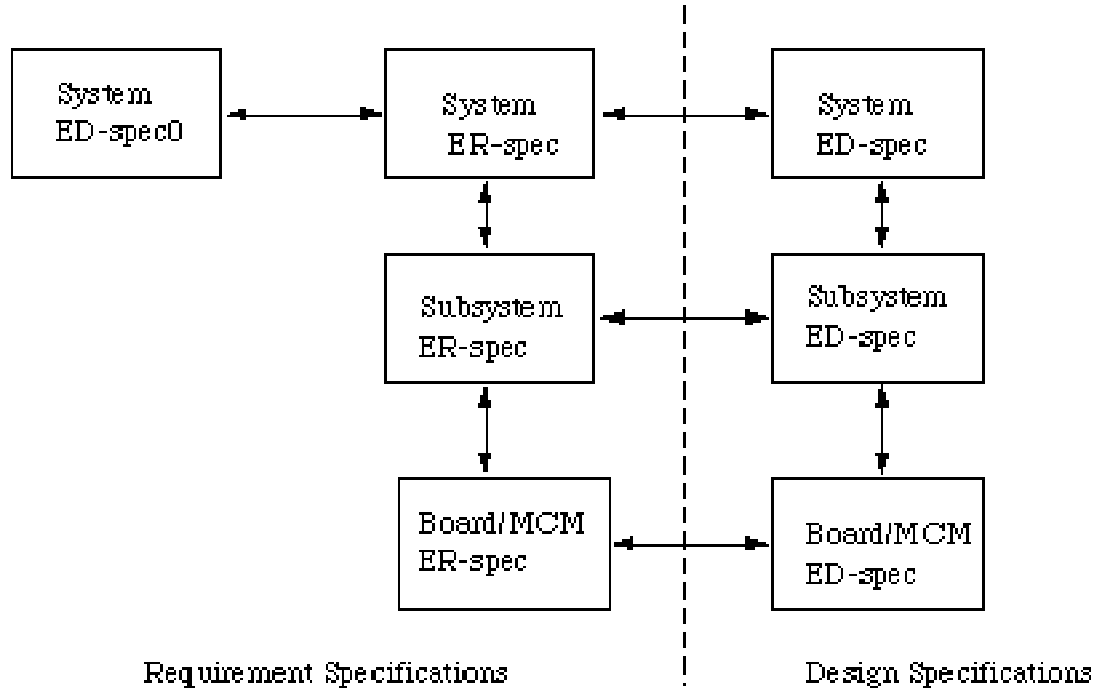


Figure 2 - 3: ER-spec/ED-spec Hierarchy Implemented as Multilevel Spreadsheet

The top row shows the system level components of an executable requirement and executable design specification as a set of spreadsheets. Other rows show the relationships of other spreadsheets at subsystem level and below. Actual entries on the spreadsheets can consist of data available directly or references to external data and executable files - any of the possible components of a requirement or specification set. Written requirements statements can be implemented as an external reference.

Horizontal arrows indicate a conformity evaluation path among the requirements and design specifications at any given level. A listing of level-specific requirements is available and a design specification defining methods by which the hardware and software will satisfy the requirements has been established. Both requirements and design specifications can be placed into separate regions on the same spreadsheet. Conformance at any level can be determined by simple cell-based comparison of the requirements and design specifications at that level.

Vertical arrows indicate a derivation relationship which covers flow down of requirements or design specifications from the next higher level and the mapping of lower level requirements at any level to the next higher upward level. Entries for multiple packaging levels can be displayed on the same spreadsheet or, more likely, because of complexity and number of entries, each packaging level can be displayed on a separate spreadsheet. In either case, relationships among entries can be implemented via cell-based or inter-sheet comparison. Thus, a mechanism for requirement distribution and traceability is created.

2.8 Roles of Customer and Contractor

Requirements and specifications are developed and applied to some degree by both the customer and the contractor. Usually, the following chain of events illustrates their respective roles in the process :

- Customer establishes need for a project - converts requests from field units into requirements documentation and an ER-spec which can be passed onto a fabricator. (An ED-spec0 may also be created.)

- Contractor accepts and verifies requirement specification (reality check) - generates cost - Interacts with customer as needed to create final set of requirement specifications - amendments to the original customer requirements list are made if necessary - generates ED-specs
- Contractor monitors conformance to requirements and reports to customer - uses flowed down version of ER-specs and ED-specs as basis for a design at all packaging levels and fabricates product - customer verifies performance of contractor design
- After fabrication, contractor and customer perform acceptance testing to verify conformance to requirements
- Customer becomes user and verifies performance in field deployment

---

Next Up Previous Contents

Next: 3 Application Example **Up:** Appnotes Index **Previous:**1 Introduction

1 Level refers to the RASSP hierarchy of packaging : System, Subsystem, Board, MCM, Chip. Requirements set for any given level must support established requirements for the next higher level of packaging,

RASSP Definition and Role of Executable Requirement (ER-SPEC) and Design (ED-SPEC) Specifications Application Note

3.0 Application Example

The RASSP Benchmark 4 project was selected to illustrate the state of development of executable requirement and design specifications. The efforts indicate the value of progressing to executable expressions of requirement and design specifications, but show that there is considerable room for improvement. The requirement list for the project contained executable elements but was not a true ER-spec as described in this document. Likewise, the design specification was not executable in the sense of the ED-spec discussed above.

3.1 Benchmark 4 (SAIP) Project Description and Specification of Requirements

The RASSP Benchmark 4 (BM4) project involved construction of two hardware subsystems that would execute a specified set of image data processing algorithms. The subsystems are called High Definition Imaging (HDI) subsystem and Mean Square Error (MSE) classifier subsystem.

In support of the two subsystems to be fabricated, MIT Lincoln Laboratory supplied an "executable requirement" to Lockheed Martin ATL as the basis for design of hardware and the integration of software for the HDI and MSE subsystems. **The MIT-LL "executable requirement" was not the same as the ER-spec as described in this application note.** A Technical Description document was also provided. These documents were titled :

RASSP Benchmark 4 Technical Description (BTD) (January 9, 1998)
RASSP Benchmark 4 Executable Requirement User Manual (ERUM)

The BTD described the HDI and MSE in their roles within a semiautomated image processing system (SAIP). This document was not explicitly given a requirements-related name but it did contain a significant number of requirement descriptions that made it an indispensable part of the overall requirements description. Requirements contained in the BTD were not "executable", but rather supplied in conventional written format. Section 2 is called "Processor Requirements" and describes the processing algorithms and the image processing modes. Included are :

physical requirements (size, weight)
image accuracy (comparison with supplied standard code)
algorithm requirements (which algorithms shall be executed)
power supply characteristics
Fault detection, isolation and testing, including acceptance testing documentation
reporting (project management)

Other sections were devoted to describing metrics for cost, performance and quality as well as the project deliverables. The BTD thus assumed the role of a traditional requirements document including a statement of work (SOW). It however, was not a complete requirements document, since it did not contain detailed descriptions of the algorithms which were to be used which would be needed to ensure conformance to expected algorithm execution. Detailed descriptions were contained in the ERUM document. For BM4, separate requirements documentation sources were used to isolate restricted design data, primarily in the algorithm area, from general requirements information. Nonetheless, the BTD was de facto a major vehicle for requirements delivery. Its non-executable nature indicates that the overall requirement documentation for BM4 was a hybrid between traditional and executable requirements.

The ERUM included a very detailed description of the algorithms to be used for processing images. Mathematical details as well as versions of source code implementing the algorithms was provided down to the level of C structures for the system parameters and the C functions which manipulate these structures. The ERUM was accompanied by a collection of files which defined and demonstrated by execution the required operation of the HDI and MSE subsystems. The ERUM was the Written requirements statement for the executable algorithmic portion of the "executable requirement" for the project and the file set contained or could generate (after appropriate compilation) the "executable" requirements. The variety of executable information supplied is illustrated by the directory listings for the files. The files *.c and *.h are C language source code files which will demonstrate algorithm behavior when used with instructions supplied in the ERUM.

File Directory For HDI

Makefile, README_LL, bool.h, boolean.h, calldgesvd.f, callsgesvd.f, center_data.c, cfftb.f, cfftf.f, cffti.f, chip_gate.c, dgesvd.f, do_image_fcn.c, est_centers2.c, est_rwin.c, est_xrwin.c, filtr_and_dec2.c, gauss_gate.c, gen_vecs.c, hdi.c, hdi.h, invert_image.c, iostream.h, linalg.h, main.C, main.jou, make-970305.log, ake_filter.c, make_look_index.c, make_real_looks.c, make_real_vecs.c, make_syn_looks.c, make_vecs.c, mlm_clipm.c, mlm_coher.c, mlm_confm.c, music.c, param, param.jou, refocus_iqdata.c, sgesvd.f, svdr.c, tools.C, tools.c, tools.h, tools.jou, unwindow.c, unwindow_squint.c, utils.c, utils.jou, zfftb.f, zfftf.f, zffti.f

File Directory For MSE

README_LL, algo.h, array.h, array.jou, bool.h, ds.h, ds_vars.h, function.h, h_files_diff.out, hdi.h, iostream.h, iterator.h, main-1.log, main-2.log, main-3.log, main-4.log, main.C, main.bak, main.jou, make-970306.log, makefile, mse.C, mse.h, mse.jou, pair.h, stringClass.h, tools.C, tools.c, tools.h, tools.jou, vector.h, vector.jou

The number and variety of file types for the BM4 "executable requirement" suggests that an ER-spec, even a partial ER-spec, can have many components, and that a clear detailed description of the files supplied and how to use them is essential.

3.2 Stepwise Application of Methodology

3.2.1 Creation of the Written Requirement Specification

As mentioned above, original requirements for a project come from field experience and needs expressed by personnel in the field. For Benchmark 4, requirements are based upon execution of selected image processing algorithms in a semi-automated image processing system. Written requirements were established by setting physical and performance goals for equipment which matched the SAIP hardware interface and were consistent with hardware/software fabrication and integration capabilities and economic limits. These goals and expected levels of achievement were refined and traded off until a preliminary design concept emerged. At this point, algorithms to solve the image processing problem and generic hardware/software solutions were known, but no specific implementation was recommended. In particular, no decision was made to choose custom hardware or to choose commercial off-the-shelf (COTS) hardware. Documentation of the trades and technical examinations resulted in the Benchmark 4 Technical Description, the primary component of the Written Requirement Statement. Detailed studies of algorithm performance were also conducted and described in text (RASSP Benchmark 4 Executable Requirement User Manual) to accompany the instructions for using the executable portion of the requirements.

3.2.2 Extraction of Executable Requirements

Once a project concept is defined, the Written Requirement Statement can be analyzed and an executable requirement can be generated. Benchmark 4 had requirements in the following categories - typical to signal processor system hardware and software :

physical
performance
environmental
general performance and acceptance testing

Physical and environmental requirements are determined by the mission of the project. Equipment intended for aircraft installation will have characteristics different from those of shipboard equipment. Benchmark 4 hardware consisting of signal processing circuit boards was to be installed in a van environment (ground, benign) to accompany the SAIP equipment it is used with. Operating temperature and physical size are typical requirements in this class.

Examination of the intent of a project and the nature of the tasks to be performed identify the performance items that should be included within an ER-spec. Mathematical descriptions of these algorithms are translatable into the code which can be entered into an executable requirement. For RASSP BM4, performance requirements were supplied as a set of files containing C source code initialized with system parameters which exhibited the performance required when the HDI and MSE algorithms executed.

The category of testing requirements establishes methods for determining whether all other requirements conform to the assigned values or level of performance and also establish a mechanism for ensuring reliability of produced equipment. A separate RASSP study, Benchmark 3 DFT Shadow Report, examined testing and testability and has established a methodology for handling testing requirements which optimizes testability of a product over its life cycle. Part of that study defined a method for consolidating test requirements.

3.2.3 Derivation of Consolidated Test Requirements

From the test and testability viewpoint, each assigned requirement must have an associated conformance test. In addition, production of reliable equipment necessitates introduction of requirements which are based upon known fabrication and software construction capabilities. These derived requirements also have associated tests. Test requirement consolidation refers to the definition of a testing program which uses specific available test means to perform all required testing throughout the life cycle of the product. The capability of each test means to detect, isolate, and correct the identified fault classes is quantified and a recommended test strategy is defined. This strategy applies test means in a specific sequence to achieve the required coverage for each identified fault.

The test requirements consolidation process forces consideration of the practicality of assigned requirements and contributes primarily to the Written Requirement Statement. The test requirements can be included in the executable requirement list in cases where simulations and use of test vectors are involved.

The Benchmark 4 Consolidated Test Requirements document consists of a collection of requirement templates for each of the derived requirements and a listing of the conformance-related test requirements.

Each template identifies a fault (including a fault model), test means applied to the fault, a definition of a fault coverage metric, and a quantitative value for the fault coverage. The templates are placed into three categories to cover design, manufacturing and field testing. Conformance test requirements expand the requirements listed in the Written Requirement Statement by identifying the type of test that will be assigned to verify conformance to a requirement.

Both classes of requirements are treated as equally necessary based on the philosophy that execution of algorithms on unreliable or untestable equipment is not acceptable.

A interesting conformance requirement for Benchmark 4 was inclusion of built in self test (BIST) capabilities.

The IEEE 1149.1 (JTAG) bus was to be the primary BIST means. This is a good example of a requirement which is not simple to quantify without a thorough examination of the consequences of its inclusion. Assignment of percentage of coverage can have a significant impact upon the cost of a project as well as the technical detailed design. For example, COTS boards could be eliminated by imposition of a substantial JTAG-based BIST requirement, and custom design freedom could be restricted by the need to use only JTAG-compliant components. For BM4, the original version of the Written Requirement Statement did require BIST as the primary performance monitoring means. A consequence of flow down of this requirement was possible elimination of COTS boards as hardware candidates. The lower cost of an existing board design could not be used to advantage. The undesirability of this restriction forced reconsideration of the imposition of the original requirement and a change to allow exclusion of COTS boards from the JTAG capability was suggested.

3.2.4 Definition of an Executable Design Specification

As mentioned above, the requirements for Benchmark 4 were a hybrid between written and executable requirements. Likewise, development of a design specification was based upon a hybrid approach. At the time of this writing, several alternatives to the HDI and MSE hardware were under consideration. In particular, the use of custom boards is being compared to use of COTS boards. Virtual prototyping and performance analyses are being conducted to select an optimized design with significant attention being paid to cost. Requirements will be applied to the hardware/software combination and in effect, a limited form of ED-spec will be created for each candidate. Development of ED-specs for the architecture candidates must be carried sufficiently far to demonstrate a clear path to meeting all requirements. More information regarding the Benchmark 4 SAIP effort can be found in the case study.

3.2.5 Economics of Executable Requirements

Limited application experience for executable requirements and specifications has demonstrated the utility of the concept. Before it will be considered for general applications, however, the economic feasibility of the concept must be validated. How does the cost for using ER-specs and ED-specs compare with the cost of conventional requirement and specification generation ?

Next Up Previous Contents

Next: 4 Application Example **Up:** Appnotes Index **Previous:**2 Introduction

Approved for Public Release; Distribution Unlimited *Dennis Basara*

1 The rationale for test requirement templates as a part of the RASSP DFT methodology and a full description of their contents are discussed in “RASSP Design for Testability (DFT) Methodology” Version 1.0.

RASSP Definition and Role of Executable Requirement (ER-SPEC) and Design (ED-SPEC) Specifications Application Note

4.0 Conclusions and Recommendations

4.1 Conclusions

- Executable requirements add value to a RASSP project by providing quantitative, unambiguous statements of requirements which are independent of implementation.
- Executable specifications describe equipment specific solutions to a requirements statement and can be used to demonstrate conformance to requirements during the virtual prototyping phase of a project.
- The RASSP methodology is designed to manage a project over its complete life cycle, including model year upgrades. The ER-spec provides a detailed record of original requirements which can be modified as needed to adapt to a system update. Effects of modification of original requirements at any system packaging level can be readily evaluated by creating an updated ED-spec to match the updated ER-spec.
- Use of ER-specs and ED-specs will add to the quality of designs and produce products with a high probability of first pass success by reducing ambiguity and its associated probability of error.
- A hierarchical ER-spec/ED-spec system will facilitate correct requirements flow down through levels of packaging and monitoring of conformance throughout the life cycle of a project.
- Use of executable requirements and design specifications is very likely to result in economic savings for a project over its life cycle similiar to those expected to be achieved using the automation in the RASSP methodology

4.2 Recommendations

- Incorporate an ER-spec to a new RASSP project and verify its effectiveness in conveying requirements to various levels of packaging. Generate the Written Requirement Statement and convert it to an executable form. Prefer a system which is likely to have model year upgrades.
- Perform a detailed economic analysis to perform a quantitative comparison of the results of using an ER-spec relative to the current conventional approach.
- Develop ER-spec and ED-specs for existing project to determine efficiency of generation of executable counterparts from existing requirements and specifications listings.

RASSP Definition and Role of Executable Requirement (ER-SPEC) and Design (ED-SPEC) Specifications Application Note

5.0 References

Anderson, A. H., G. A. Shaw, "Executable Requirements and Specification," Journal of VLSI Signal Processing 15 (1997). [paper not available]

Anderson, A. H., G. A. Shaw, "Executable Requirements: Opportunities and Impediments," ICASSP 1996 [ANDERSON_A96]

Anderson, A. H., G. A. Shaw, and C. T. Sung, "Vhdl Executable Requirement," Proceedings 1st Annual RASSP Conference, Arlington, Va., August, 1994, pp 87 - 90. [ANDERSON_A94]

Benchmark 4 Consolidated Test Requirements, April 30, 1997. (Note: Under review for release approval.)

Hehner, Eric C. R., "Specifications, Programs, and Total Correctness," University of Toronto, January, 1997. [paper not available]

"RASSP Benchmark 4 Processor Requirements (Draft V0.5)," March 18, 1997. [document not available, see Gary Shaw for release]

RASSP Design For Testability (DFT) Methodology document, Version 1.0, Lockheed Martin Advanced Technology Laboratories, September 1995. [METHODOLOGY_DFT95]

RASSP DFT Summary Benchmark 3 (BM3) Shadow Program, Lockheed Martin Advanced Technology Laboratories, November 1997 [BM3_DFT_Shadow_97] (Note: Under review for release approval.)

RASSP Methodology Document, Version 2.0, Volume 1, Lockheed Martin Advanced Technology Laboratories, October 1995. [METHODOLOGY_95]

Shaw, G. A., Anderson, A. H., and Anderson, J. C., "RASSP Benchmark 4 Technical Description," MIT Lincoln Laboratory, January 9, 1998 [BM4TD_98]

Thulen, M.,"A Case Study of the Application of Simulatable Specifications in the Design of Electronic Sustems," (CEENSS) Program, April 1998. [CEENSS]