The Proof of SIFT

P.M. Melliar-Smith, Richard L. Schwartz

Computer Science Laboratory
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025 USA

1 Introduction

SIFT (Software-Implemented Fault Tolerance)[Wen 78, Gol 80] is a reliable aircraft flight control computer system. SIFT uses five to eight Bendix BDX930 computers, each equipped with its own private storage. A broadcast communication mechanism allows each processor to transmit its results to a buffer area in each processor. The design is fully distributed, with no common buses, no common clocks, no common interrupt or synchronization registers, no common storage, and no physical means of isolating a faulty processor. The SIFT processors (physically) share only the ability to communicate with one another.

In SIFT, fault masking, detection, and reconfiguration are all managed by software. Safety-critical tasks are replicated on three or more processors, with all processors voting on the results of each redundant computation. A Global Executive task, which is itself replicated, is responsible for fault diagnosis on the basis of error reports from the voting software, and for selecting a new configuration excluding the processors deemed to be faulty. The result of this reconfiguration is that tasks are shifted from faulty processors to those still working. Every processor votes on the results of the Global Executive and adjusts its task schedule accordingly.

SIFT's processors run asynchronously; each contains its own private clock. The software must maintain a loose synchronization to within approximately 50 microseconds, and each processor runs a task periodically to resynchronize its clock with those of the other processors in the configuration. Care was taken in the design to ensure that, even under fault conditions, all working processors retain synchronization and remain consistent in their schedule and configuration.

2 The STP Specification and Verification System

STP [SSM 81] is an implemented system supporting specification and verification of theories expressed in an extension of multisorted first-order logic. The logic includes type parameterization and type hierarchies. STP support includes syntactic checking and proof components as part of an interactive environment for developing and managing theories in the logic. In formulating each new theory, the user begins with a certain *core theory* that comprises a set of primitive types and function symbols, and extends this theory by introducing new types and symbols, together with axioms that capture the intended semantics of the new concepts. The mechanical proof component of the system is predicated on a fast, complete decision procedure [Sho 82] for a certain syntactically characterizable subtheory. By providing aid to this component in the form of the selection of appropriate instances of axioms and lemmas, the user raises the level of competence of the prover to encompass the extended theory in its entirety. As a result of a successful proof attempt using STP, one obtains the sequence of intermediate lemmas, together with the axioms, auxiliary lemmas, and their necessary instantiations, which lead to the theorem. The system automatically keeps track of which formulas have been proved and which have

This research was supported in part by NASA contract NAS1-15528

not, so that the user is not forced to prove lemmas in advance of their application. The system also monitors the incremental introduction and modification of specifications to maintain soundness.

3 The Design Hierarchy of SIFT

The intent of formal specification and verification is to increase one's confidence that a system will function correctly. As such, the specification and proof of system conformance must be *believable*. The problem of specification credibility in the proof of SIFT is addressed through the use of hierarchical design specification and verification. This approach allows incremental introduction and verification of design aspects – making a step-by-step connection between the high-level, abstract view of the system to the detailed control and data structures employed in the implementation. The accompanying figure illustrates the SIFT design hierarchy.

The IO Specification, the most abstract functional description of the system, asserts that, *in a safe configuration*, the result of a task computation will be the effect of applying its mathematical function to the results of its designated set of input tasks, and that this result will be obtained within a real-time constraint. Each task of the system is defined to have been performed correctly, with no specification of how this is achieved. The model has no concept of processor (thus no representation of replication of tasks or voting on results), and of course no representation of asynchrony among processors. The specification of this model contains only 8 axioms.

The Replication Specification elaborates upon the IO Specification by introducing the concept of processor, and can therefore describe the replication of tasks and their allocation to processors, voting on the results of these replicated tasks, and reconfiguring to accommodate faulty processors. The specification defines the results of a task instance on a *working* processor based on voted inputs, without defining any schedule of execution or processor communication. This model is expressed in terms of a global system state and system time.

The Broadcast Specification develops the design into a fully distributed system in which each processor has access only to local information. Each processor has a local clock and a broadcast communication interface and buffers. The asynchrony among processors and its effect upon communication is modeled. The specification explicitly defines each processor's independent information about the configuration and the appropriate schedule of activities. The schedule of activities defines the sequence of task executions and votes necessary to generate task results within the required computation window. The Broadcast Specification is the lowest level description of the complete multiprocessor SIFT *system*.

The PrePost Specification consists of specifications for the operating system for a single processor. The specification, in terms of pre-condition/post-condition pairs, facilitates the use of sequential proof techniques to prove properties of the Pascal-based operating system as a sequential program. These specifications are very close to the Pascal programs, and essentially require the programs to "do what they do".

The Reliability Analysis is a conventional discrete semi-Markov analysis that calculates the probability that the system reaches an unsafe configuration from the rates of solid and transient faults and from the reconfiguration rates. Neither this Reliability Analysis nor the other Fault and Error Models will be described here.

A more detailed presentation of the SIFT specifications and their verification can be found in [MeS 81].

3.1 The Proof of the SIFT Design

Hierarchical proof of design involves axiomatically defining a mapping from functions and predicate symbols of a level of the design to terms of the level below. One then proves each axiom of the higher level as a theorem in the theory of the level below.

The most substantial portion of the verification of the IO Specification involved proof of the axioms that define the results of tasks in a safe configuration. To derive these from the Replication level involved a demonstration that task replication and majority voting suffice to mask errors in a safe configuration. To do so required approximately 22 proofs, with an average of 5 premises necessary per proof, and 106 instantiations of axioms and lemmas used overall.

The proof of the relationship between the Replication Specification and the Broadcast Specification was more challenging. This proof required demonstrating the consistency of information present in each working processor (the set of processors still in the configuration, in particular), of the correct schedule to be used, and of the results of past task iterations. Furthermore, the proof required showing that the various processors, operating independently and asynchronously with only local information, can communicate with each other without mutual interference, and can cooperate to produce a single global system defined by the Replication Specification. It was also necessary to show that the task schedules were such that the task results are always available in other processors when required. The derivation of the Replication axioms involved 56 proofs, with an average of 7 premises each, and 410 instantiations of axioms and lemmas overall.

The proof of the relationship between the Broadcast Specification and the PrePost Specification was easier. Most of the interest centered on the mapping between the properties of the changing values of variables in the Pascal system and the properties of the Broadcast model's more abstract state functions which are explicitly parameterized by time and processor. A futher complication concerned mapping the functional representation of data structures in the Broadcast model to the (finite) Pascal data structures. Derivation of the necessary Broadcast axioms involved 17 proofs, with an average of 9 premises each, and 148 instantiations overall. The proof of the Pascal programs from the PrePost specifications used conventional verification condition generation.

References

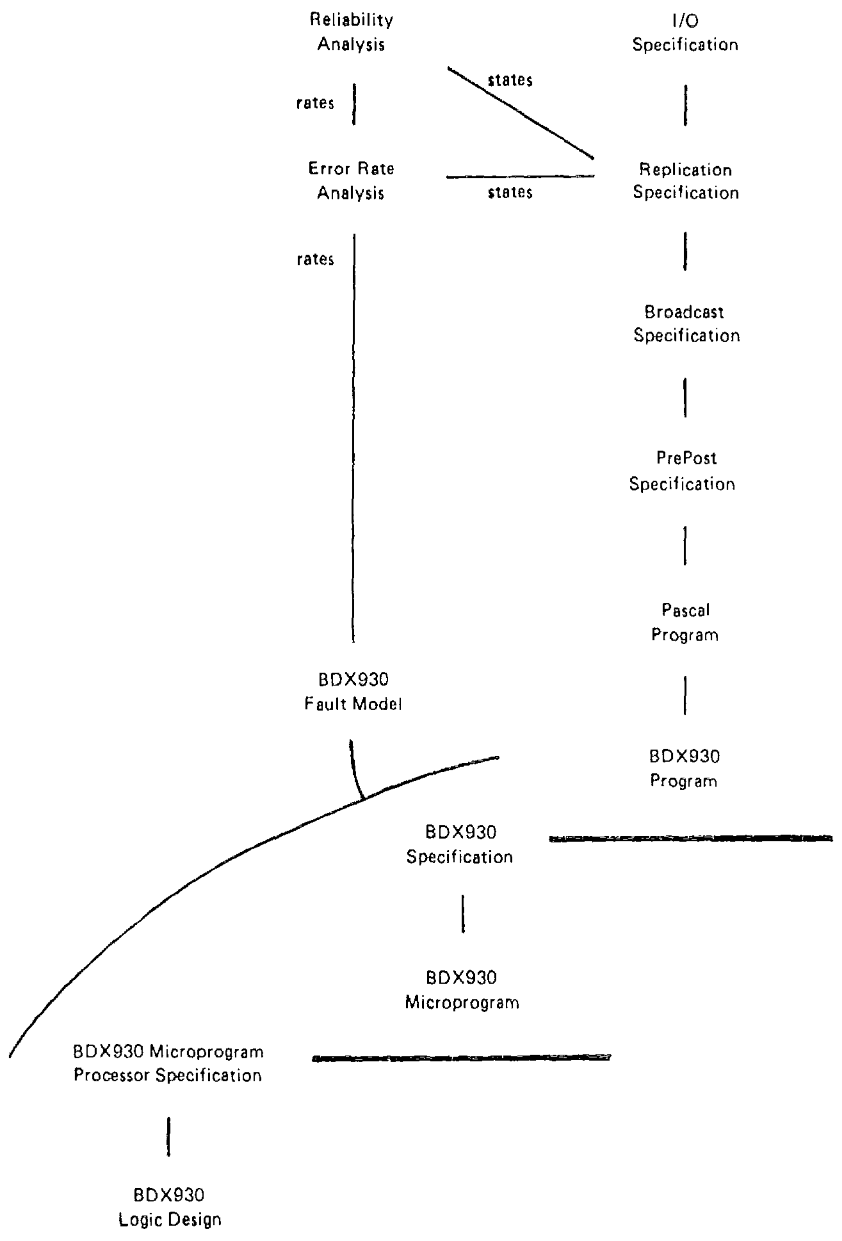
[Gol 80] Goldberg, J., "SIFT: A Provable Fault-Tolerant Computer for Aircraft Flight Control", IFIP, 1980.

[MeS 81] Melliar-Smith, P. M., R. L. Schwartz, "Hierarchical specification of the SIFT fault tolerant computer system", *Proc NATO/AGARD Conference*, 1981.

[Sho 82] Shostak, R., "Deciding Combinations of Theories", Technical Report 132, SRI International, January 1982.

[SSM 81] Shostak, R., R. Schwartz, P.M. Melliar-Smith, "STP: A Mechanized Logic for Specification and Verification", Technical Report 130, SRI International, December 1981.

[Wen 78] Wensley, J., et al., "SIFT: Design and Analysis of a Fault-tolerant Computer for Aircraft Control", *Proc IEEE*, Vol. 66, No. 10, Oct. 1978.



THE HIERARCHY OF SPECIFICATIONS AND ANALYSES USED TO SUBSTANTIATE THE RELIABILITY OF SIFT