Technique	Minimize the occurrence and effect of Built In Test (BIT) false alarms by applying principles and techniques that are intended to reduce the probability of false alarms and increase the reliability of BIT in avionics and other electronic equipment.

FALSE ALARM MITIGATION TECHNIQUES

Use techniques such as voting schemes, continuous monitoring, and decentralized architectural design to minimize the occurrence and effects of Built-in-Test (BIT) false alarms

Benefits	Effectively implementing BIT techniques automatically reduces the number of BIT false alarms. Decreasing the number of BIT false alarms increases a system's availability and decreases the maintenance man-hours required. The overall result is a reduction of the system's life cycle cost.
Key Words	Anomalies, Built-In-Test, False Alarms, Circuit Monitoring
Application Experience	International Space Station Program, National Space Transportation System
Technical Rationale	The reliability of a system's BIT can be determined in part by the number of false alarms it experiences. If the BIT can not accurately identify and report the occurrence of failures then the test has failed its mission. Testability must be treated with the same level of importance as other design disciplines. BIT reliability must be considered just as critical as any other performance requirement. A system can not perform its mission if its components are constantly being removed for false maintenance.
Contact Center	**Johnson Space Center**

False Alarm Mitigation Techniques
Technique DFE-2

To mitigate false alarms, a system's Built In Test (BIT) circuitry must be able to cope with a limited amount of anomalous performance. NASA Handbook 5300.4 (1E) defines a false alarm as "an indicated fault where no fault exists." Based on this definition, this technique is concerned only with BIT indications of system malfunction which cause unnecessary maintenance actions. The inability of a system to detect or report the occurrence of a failure, a "fails to alarm condition", is not a false alarm and is not addressed.

BIT should be designed to distinguish between actual failures and anomalies which must be tolerated due to adverse operating conditions or that are normal anomalies within acceptable limits. To accomplish this, the following principles and techniques must be mandated in the system specifications, requirement documents, and design policies and implemented in the system design.

Voting Scheme
One technique is called the "Voting Scheme." With the voting scheme, all test data are analyzed by three or more different computers. A failure is declared only when a majority of the computers detect the same failure. An example of this type of architecture is the Space Shuttle Orbiter Avionics System. The five General Purpose Computers (GPCs) are all interconnected to the same 28 serial data channels. The GPCs perform all system-level processing and require a majority agreement on all test signals. This technique requires an extensive use of resources but is extremely effective at mitigating false alarms. A less complicated version of this is the use of double or triple redundant monitors. Having two or more sensors in series increases the reliability of the test data reported while only requiring a single computer or processor.

Continuous Monitoring
Continuous monitoring with BIT filtering can be used in place of the voting scheme. With this technique, BIT results are based on a integration of successive measurements of a signal over a period of time instead of a single check of the signal. The monitoring of the signal does not have to be continuous but only sampled over the time period. The filtering involves comparing the current reading of a signal with past and future readings of the same signal. This filtering allows for the disregarding of sporadic out-of-limit measurements. Only when a signal is out-of-limits for a predefined time limit or a sequence of tests identify the same failure, should the BIT flag be set.

To maximize the effectiveness of continuous monitoring, the BIT data must be recorded. Once recorded, the data need to be summarized and evaluated so that trends can be tracked and weaknesses identified. Controls should be implemented to help manage all of this data. The number of signals monitored and the maximum sample rate can be limited. The time span over which data are collected should be set at a reasonable period, and the type of data accumulated should be restricted. Finally, computing techniques can be used that do not require the storage of old data. Once the information is gathered, a failure log should be created.

This failure log is the basis for future modifications to the system's BIT. To improve the BIT, every instant of anomalous performance not related to an identified failure mode should be analyzed and the root causes identified. Some form of corrective action must be taken to avoid recurrence. If a design change cannot be made, then the BIT must be modified to accommodate the non-failure causing anomaly.

The need for modification requires BIT to be flexible. Test parameters and limits must be easily changed. The operator should be able to control or even change the test sequence. This flexibility allows the necessary changes in the BIT to be made if false alarms start occurring. For example, the Space Station's Command and Data Handling System uses programmable Deadman Timers in the multiplexer/ demultiplexer (MDM's) and standard data processor (SDP's). The response intervals of the timers can be adjusted by the system controller to accommodate changes in system configuration or mode of operation. However, the BIT software must be changed without disturbing the system operation. For this to be possible, the BIT software must be independent of the operating software.

Decentralized Architecture

Another technique for mitigating false alarms is the use of a distributed or decentralized BIT architecture. With this approach the BIT is implemented so that a "NO GO" on a given test directly isolates the implied failure to a replaceable unit. Locating most of the BIT internal to a unit greatly reduces the possibility of incorrect isolation of a failure. Although the decentralized BIT concept consists primarily of unit level tests, some system level testing is still required.

An excellent technology for combining unit level testing with system level testing is boundary scan. Boundary scan is the application of a partitioning scan ring at the boundary of integrated circuit (IC) designs to provide controllability and observability access via scan operations. In Figure 1, an IC is shown with an application logic section, related input and output, and a boundary scan path consisting of a series of boundary scan cells (BSC), one BSC per IC function pin.

.

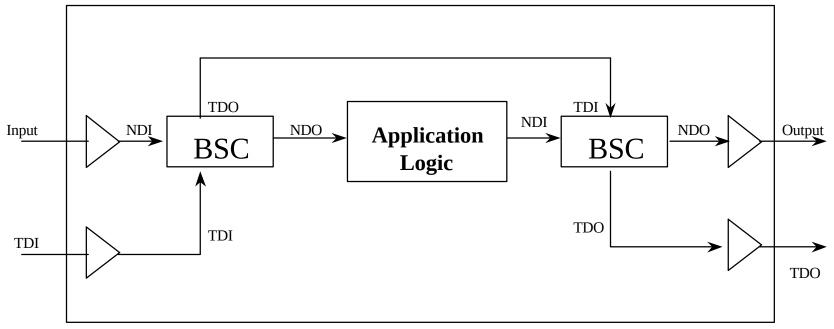

Figure 1. Integrated Circuit with Boundary Scan Paths

The BSCs are interconnected to form a scan path between the host IC's Test Data Input (TDI) pin and Test Data Output (TDO) pin, for serial access

During normal IC operation, input and output signals pass freely through each BSC, from the Normal Data Input (NDI) to the Normal Data Output (NDO). However, when the boundary test mode is entered, the IC's boundary is partitioned in such a way that test stimulus can be shifted in and applied from each BSC output (NDO). The test response can then be captured at each BSC input (NDI) and shifted out for inspection. Internal testing of the application logic is accomplished by applying test stimulus from the input BSCs and capturing test response at the output BSCs. External testing of wiring interconnects and neighboring ICs on a board assembly is accomplished by applying test stimulus from the output BSCs and capturing test response at the input BSCs. This application of a scan path at the boundary of IC designs provides an embedded testing capability that can overcome test access problems. The unit level tests can also be combined for a subsystem or system level verification (Figure 2). More details on applying these techniques are in IEEE Standards 1149.1 "Boundary Scan" and 1149.5 "System and Maintenance Bus."

Finally, high-reliability components should be used in the design. The reliability of the BIT hardware should at least equal or exceed that of the hardware it is testing. The BIT software also needs to be thoroughly tested and verified to ensure that it will not be a source of false alarms. Accordingly, adequate amounts of effort and resources must be allocated during the design phase. The designer should not be unduly limited by memory size, component count, or any other allocated resource.

These guidelines are not all inclusive. The false alarm problem is very complex. Each system is unique and must be approached differently. The best approach is simply to eliminate each factor as it is identified.

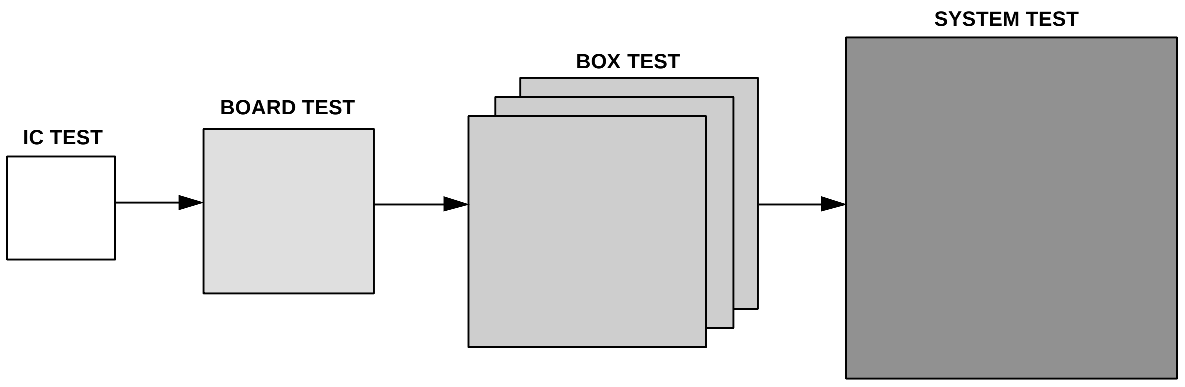

Figure 2: Typical Test Regimen for Space Systems

References

1. Coppola, Anthony, *A Design Guide for Built-In-Test (BIT)*, RADC-TR-78-224, April 1979.

2. Malcolm, John G., Highland, Richard W., *Analysis of Built-In-Test False Alarm Conditions*, RADC-TR-81-220, August 1981.

3. NASA Handbook 5300.4 (1E), *Maintainability Program Requirements for Space Systems*, March 10, 1987.

4. Texas Instruments Inc., *TESTABILITY, Test and Emulation* Primer, 1989.

This page intentionally blank