White Paper | February 2010

radisys®

Distributed Fault-Tolerant / High-Availability (DFT/HA) Systems

By Ravi Raj Bhat, Vice President of Engineering

Today's telecommunication platforms must provide "five nines" availability, meaning practically no loss of service due to hardware or software errors, nor any downtime for software upgrades or hardware maintenance. This expectation places an unprecedented burden on service providers to ensure that all the network elements needed to support a service are functioning whenever a user requests that particular service.

Five nines means 99.999 percent availability. Achieving 100 percent perfection, while desirable, may be impossible, and a fault-tolerant system is designed to function correctly given two predictable contingencies: one, the small (0.001 percent chance) but practically inevitable presence of a fault in the system; and two, the likely requirement for periodic maintenance or system improvements. We will return to this point.

Supporting five-nines service availability depends on the near-flawless interaction of software (application, OS and management middleware), hardware and network design, as well as on environmental and operational factors. Hardware fault tolerance typically relies on redundant processors, memory, buses, power supplies and databases. Software fault tolerance uses a combination of software redundancy and simple hardware redundancy to provide the necessary availability in the case of failure. Network fault tolerance uses redundant data link cross connects (T1s, DS3s, OC3s, etc.).

This paper provides an overview of Fault-Tolerant/High-Availability (FT/HA) components. Figure 1 shows where all the FT/HA components of a system can reside. Both the hardware and software solutions and the challenges addressed by these solutions are discussed. This paper also outlines the Trillium FT/HA and the patent-pending Distributed Fault-Tolerant/High-Availability (DFT/ HA) architectures and implementations, and how they are designed to overcome these limitations.

CONTENTS

FT/HA Concepts

High Availability

To understand availability, we first need to understand reliability. The reliability of an element is the conditional probability that the element will operate during a specified period of time. A system may be considered highly reliable (that is, it may fail very infrequently), but if it is out of service for a significant period of time as a result of a failure, it will not be considered highly available. One measure of an element's reliability is its failure rate, or Mean Time To Failure (MTTF), the interval in which the system or element can provide service without failure. Another measure of reliability is the Mean Time To Repair (MTTR), which represents the time it takes to resume service after a failure has been experienced. Systems may go out of service for any number of reasons, such as the occurrence of a fault, repair activities, software loading, hardware upgrading or periodic maintenance. In all these cases, MTTF and MTTR immediately become vitally important.

The availability of an element is the probability that the element is in service and available to a user at any instant in time. It can be expressed using these measures of reliability.

$$\textit{Availability} = \textit{MTTF} / (\textit{MTTF}+\textit{MTTR})$$

As the equation shows, the availability of systems can be increased by designing components that are highly reliable (high MTTF), and/or by shortening the time required to repair the system and to return it to service (low MTTR). Since it is impossible to create systems that never fail, the key to high availability is to make recovery time as brief as possible.

Network elements typically operate with a target of five nines availability. Such a level of availability in a telephone switch, for example, means that the switch is expected to be out of service only for about five minutes per year. These few minutes are all the time needed to repair faults, load software, upgrade hardware and perform periodic maintenance and any other necessary activities.

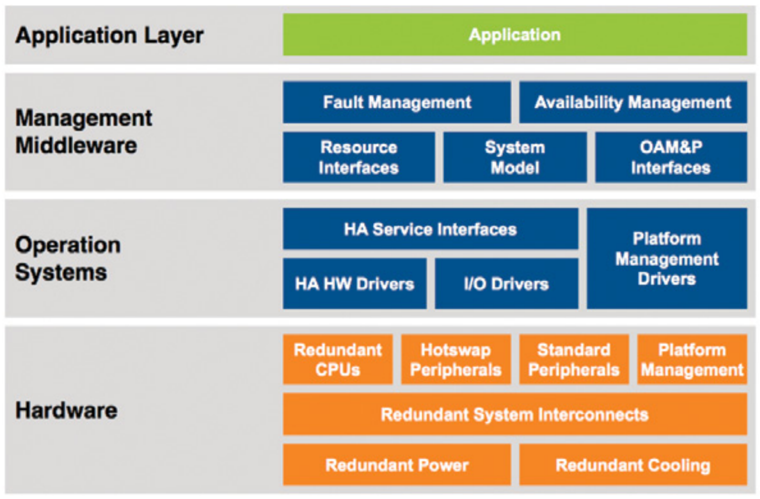

Figure 1. *FT/HA components within a system*

Fault Tolerance

A fault-tolerant system is available in the presence of faults. The design, development and testing of highly available systems is challenging and expensive. It generally proves more economical to assure high availability by detecting faults and avoiding service disruptions through redundancy in the system; that is, designing fault tolerance into the system so that it functions correctly and is continuously available in the presence of faults. Fault tolerance will increase the availability of a system, which is measured by the probability that the system will operate and be accessible when required for use, even during periods of preventative maintenance or repair.

The simplest failure-response strategy is to let a non-redundant system fail and then repair it off-line. This is the lowest-availability strategy, and it is unacceptable for mission- critical computer and communications applications in which high availability is a necessity. Such applications require systems that include fault- tolerant capabilities that make them highly available.

Fault tolerance can be achieved through a combination of system hardware and software. One way to achieve system fault tolerance is to use redundant hardware components within the system (e.g., multiple processors, buses or power supplies in a single system) that operate simultaneously and in parallel, comparing results of the operations performed. This is referred to as a hardware fault-tolerant or redundant system. Using software to handle the fault management of two or more subsystems, regardless of whether each subsystem has redundant hardware components, can enhance system fault tolerance. When one subsystem fails due to the presence of a fault, the other subsystem takes over, so that the overall system can continue to operate without disruption in service. Periodically causing a switchover between the software modules and restarting the passive software module can achieve software redundancy.

As we mentioned in defining "five nines," failure is just one of the contingencies that must be overcome to achieve 99.999% availability. The other and more likely event is anticipated maintenance to the hardware or software. Fault tolerance is necessary to enable the system manager to plan and execute "rolling" upgrades; that is, controlled, programmed improvements that avoid system shutdown. In a fault tolerant system, five- nines availability can be ensured because the system manager can execute a rolling upgrade at an optimal moment--an anticipated fix with no downtime instead of a remedial one with a minimized interruption.

Market and Technology Drivers

Following are some of the market drivers for FT/HA systems.

Converged and Decomposed Network

Telephone companies and their equipment manufacturers have set standards for five nines availability that other network elements in a converged and decomposed network must meet. Internet infrastructure and services will need to strive for near-perfection to provide services seamlessly across both the Internet and the Public Switched Telephone Network (PSTN).

These decomposed network elements (signaling gateways, media gateways, media gateway controllers, etc.) are built using commercial off-the-shelf (COTS) hardware and software solutions. Because of open standards and architecture, these solutions can be obtained from multiple vendors, and not, as before, from a single vendor that supplied everything-applications, hardware and software-in one box. These COTS solutions must meet the stringent availability expectations set by the PSTN.

Internet

The growing popularity of the Internet as a channel for business transactions and as an alternative to the traditional voice network is driving manufacturers to provide equally reliable and near-continuously available systems. Internet customers expect to be always-on, always-connected (AOAC), and they demand quality and reliability levels like those established by the telecom industry over the past 10-15 years in the PSTN. Convergence is driving these same requirements into data networks and mainstream infrastructure equipment.

In addition to mission-critical data-processing servers and carrier switches, Web servers, signaling gateways, media gateways, media gateway controllers, wireless base station controllers (BSCs)/mobile switching center (MSCs) and CT servers are expected to have higher levels of availability. Figure 2 shows many of these elements in a decomposed network.

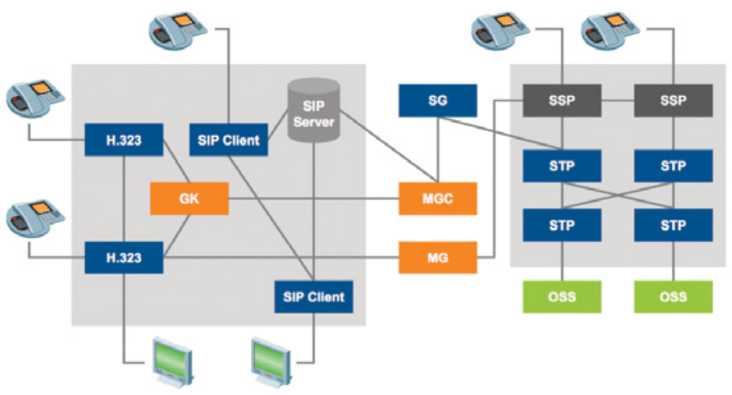

Figure 2. *A decomposed NextGen network*

FT/HA Hardware Solutions

Designers and developers can take several design approaches to ensure high availability for individual network elements. The simplest type of network element is non-redundant and must be repaired off-line if it fails. This type of element will have relatively low design complexity and low cost. (Depending on its reliability, it may also have low availability.) In contrast, high-availability elements require both the ability to support on-line repair (usually through the "hot swap" of components while the element is in service) and additional redundancy. Hardware fault tolerance typically relies on redundant processors, memory, buses, bus cross-connects, power supplies, cooling systems and disk storage. Figure 3 shows two extremes of these solutions.

Elements with additional redundancy typically use "retry" and "masking" for recovery. Retry-based elements attempt to ensure that there is a second attempt at the operation if an initial operation fails. If the second attempt succeeds, the fault was probably transient. If the second attempt fails, the fault is probably permanent. Masking-based elements attempt to ensure that only the results from the correct operating portion of the element are used if a component fails. In either case, if a component has failed, the system attempts to detect, diagnose, isolate, recover, repair and compensate for the fault.

System designers can enhance the fault tolerance of a system by combining simple hardware redundancy with fault-management software: replicating hardware subsystems to provide the necessary back-up hardware in the case of failure and executing a copy of the same software on each node provide simple hardware redundancy. These hardware subsystems, or "nodes," can either be entire computers complete with processor, memory, buses, power supplies, cooling systems, and disk storage (in practice this is called "clustering") or any of these components individually. The fault-management software manages the switchover from the failed node to another operational node, maintaining the node state information of the failed node so that it can be used by the system after the switchover. This eliminates the need for complex circuits, typical of more complicated hardware fault-tolerant approaches, and can significantly decrease the design complexity and cost of the system.

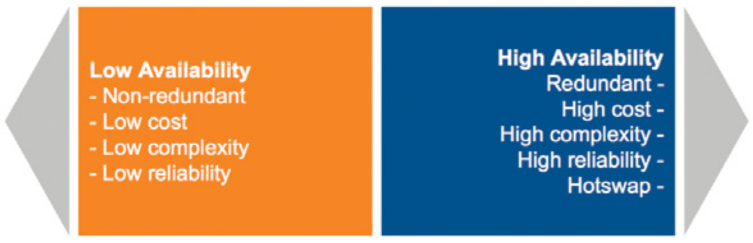

Figure 3. *Spectrum of FT/HA hardware solutions*

In telecommunication and networking applications, system designers typically achieve fault tolerance by using either dual-node or multi-node architecture. In dual-node architecture, two nodes are running simultaneously. In some systems, both nodes are active. In other systems, one node is assigned to be active and the other node is a standby. A more scalable multi-node architecture allows for the number of active and standby nodes to be configured and for the active nodes to share the system load. Dual-node architecture is more popular because of its lower cost.

A dual-node system typically achieves fault tolerance in one of two ways:

- Both nodes are active, sharing the load of the system by executing different tasks. If one active node fails, the other active node takes over the tasks of the failed node.
- Both nodes are capable of executing the same tasks, but one node is active and the other is standby. If the active node fails, the standby node becomes active and takes over the assigned tasks.

In either case, the switchover from one node to another is either controlled (operator- initiated) or forced (system-initiated). Generally, an operator-initiated switchover is performed for maintenance purposes, whereas a system-initiated switchover is carried out when a node fails. Application software is usually unaware of underlying HA hardware. These hardware nodes communicate with each other via current technologies such as cPCI buses or some upcoming technologies such as InfiniBand, RapidIO or switched Ethernet technologies.

Platform Management software provides the monitoring and controlling capabilities required for the detection, diagnosis, isolation, recovery and repair of nodes.

Three common redundancy schemes found in hardware-based solutions are:

- *2N:* one node in standby for every node in operation. For example, two independent cPCI chassis providing redundancy at the CPU and I/O level.
- *N+1:* one standby node for N operational nodes. For example, two cPCI chassis with crossover providing redundancy at the CPU and I/O level.
- *N+M:* A pool of N nodes working in normal operation with a pool of M nodes in standby. This is more typical of distributed systems where nodes are active and also the standby of each other.

FT/HA Software Solutions

As discussed before, hardware fault tolerance usually relies on redundant processors, memory, buses, power supplies and disk storage. Software fault tolerance, on the other hand, uses a combination of software redundancy and simple hardware redundancy to provide the necessary availability in the case of failure. Unlike hardware fault tolerance, software fault tolerance is absolutely necessary to provide system-wide redundancy. A major technology differentiation of high-availability systems is the inclusion of configuration and fault management software functions, enabling the detection, diagnosis, isolation, recovery and repair of faults anywhere that could result in the failure of the high-availability system. This section discusses how the operating system (OS), management middleware (MM) and application software can provide FT/HA functionality.

Operating System

High-availability systems put certain requirements on the OS. Some of the important requirements are:

- Memory address space protection for kernel and applications: support for virtual address space. With this type of memory protection, the active process cannot corrupt the memory space of other kernel processes and applications in the system.
- Robust kernel: the OS should be able to isolate, prevent the propagation of or mask the impact of, potential hardware and software faults.
- Ability to deal with a dynamic hardware configuration: an HA-aware OS must be able to support dynamic reconfiguration of applications and drivers as faulty hardware is replaced with healthy hardware. This “hot swap” support should include fast restarting of applications and dynamic installation and update of device drivers.
- Support for fail over mechanisms: the OS should be able to respond rapidly to faults and fault recovery procedures. The OS should be able to support fault recovery as dictated by policy management.
- Interface to MM: for the management of the OS itself and ability to report faults externally.
- Support for online upgrades.

Management Middleware

Management Middleware (MM) provides a set of configuration and fault management capabilities. MM typically contains direct interfaces to the operating system, to hardware devices and to the applications, and it adds additional capabilities such as messaging or state replication services to an operating system.

The following are some other important capabilities of MM:

- Fault management: detects, diagnoses, isolates, recovers, assigns the role and repairs the faults anywhere in the high-availability system. MM may also support prediction of faults by looking at changes taking place in the system.
- Availability management: collects data, does role assignment and performs rapid recovery in conjunction with fault-management.
- Resource interfaces: provides interfaces to the OS and application.

- Systems model: manages configuration and dependencies among system-wide components. This also maintains a state-aware model of the total system.
- Operations, Administration, Maintenance and Provisioning (OAM&P) interfaces: provides local and remote interfaces to view and manage the system or its components, for control and intervention.
- Messaging services: supports checkpointing and inter-application messaging.
- Support for online upgrades.

Application

The application software may or may not be aware of the underlying HA system infrastructure. Application software will typically interact with the OS and MM no differently than in non-HA systems. In case the application needs to be made aware of the underlying HA system infrastructure, there must be an interface between the application software and the MM. This interface may provide access to services unique to the high-availability system such as checkpointing application state or heartbeating.

Current Challenges

Legacy fault-tolerant systems were built using special fault-tolerant hardware platforms that were, in many cases, designed specifically for the application/service that they were to host. In addition, fault-tolerant applications were constructed from scratch and were expensive to design, produce and maintain. Along with specific hardware and applications, the system management software was also designed for the specific system that it was supposed to manage. In such cases, the FT/HA software might be available in binary form only, rather than source form. This would limit the available platform choices. For the system designer to be able to select a different board, processor and operating system for a specific application, the availability of portable software-written in a popular programming language-is paramount. Otherwise, system designers and engineers will be forced to make a platform choice that may not be suited to their needs: they will be precluded from choosing alternative platforms that would work better for their application, or provide cost or other competitive advantages. Moreover, they will not be able to modify the software to provide their own proprietary value to the end product.

Some other important limitations of current software solutions include:

- They execute the same software simultaneously on both active and standby nodes. Using this solution, the entire system could fail since the same fault could occur in the software on both nodes at the same time.
- Some systems are unable to protect individual software instances separately, if both the instances are on the same node (either active or standby). This amounts to an “all or nothing” solution. A more flexible approach is to distribute the protected instances on separate nodes as appropriate for their application. The benefit of this approach is easier fault detection.
- Some systems are unable to detect and resolve sequential faults. It is possible for an error to occur during the recovery period of another fault. The second error must also be considered and resolved in a graceful manner (e.g., first queued, and then processed after the first error has been resolved).
- Some systems are incapable of rolling upgrades, therefore are to be taken offline to perform planned maintenance.

Trillium FT/HA Solutions

Trillium software solutions provide a flexible framework that is platform-independent and cost-effective. Trillium provides FT/HA communication layers and stacks that can be classified as applications that provide service to a system user. This FT/HA solution maintains active calls during software and hardware failures. Provided in source code, the Trillium solution gives systems designers and engineers total freedom when choosing their hardware platform and operating system (kernel or user space). Developed primarily for telecommunications products, the Trillium solution is also applicable to other types of products requiring high availability.

The Trillium solution does not employ the standby node to execute the protected application at the same time as the active node, thereby protecting the system from "mirror- image" errors that could occur on both nodes. Instead, the active node updates the standby node's internal states, so that when the standby node becomes "active," it will start executing the software based on the most recent update. This approach does not allow the propagation of a software fault from the active node to the standby, since the standby is not executing the same segment of the code at the same time as the active node. Therefore, the system can potentially bypass a software problem that may occur in the active node. Trillium's solution includes a FT/HA Core product, plus a series of technology-specific products, each called a FT/HA Protocol Specific Function (PSF). Trillium's FT/HA Core software product can be used on systems consisting of either hardware-fault-tolerant nodes or non-hardware-fault-tolerant nodes.

The FT/HA Core enables system designers and engineers to replicate a node and turn those nodes into a FT/HA system. Trillium's solution can be used for open service platforms, which in turn can be used as a basis to create other products. For example, it can be used for a Service Control Point (SCP), which accesses databases for telephone number translations. The Trillium-enabled SCP is able to use the dual- node architecture using either a telecommunications board manufactured by any of the major hardware providers or a board developed in-house, and it is able to use any multi-tasking operating system. For other systems, such as an Intelligent Peripheral (IP), the multi-node approach using the Trillium solution can provide the high level of availability that is required by such systems.

For specific telecommunications applications, Trillium develops the FT/HA PSF for particular technologies providing a complete solution for FT/HA systems. The user need only place an application, such as a Mobile Application Part (MAP), on top of this stack and configure the protocol stack for its particular requirement specifications.

The Trillium FT/HA solution provides the following functionality:

- Maintenance of all active calls during software or hardware failures by updating the states of the standby node without execution of its state machine.
- Alarming the stack manager software when a fault occurs within the protected layer and carrying out a forced switchover. The stack manager is a system-dependent function and must be implemented by the system designer. Once the stack manager has received a failure indication, it can initiate corrective action by invoking Trillium's system manager to carry out a forced switchover.
- Protection of a single application (layer) or multiple applications (stack) with each layer residing in separate nodes or multiple nodes. When a node fails, Trillium's FT/HA software will re-route the messages destined for the failed node to another operational node.
- Functionality within systems in which the stack manager carries out a controlled switchover for operator- initiated routine maintenance. The stack manager "requests" the switchover and the "request" is subsequently carried out by the software.
- Functionality within systems designed to detect and resolve sequential errors. The stack manager must queue the errors and notify the software.
- Support for rolling upgrades, thus ensuring five nines of availability even during planned maintenance.

Most equipment manufacturers use a dual-processor architecture with active and standby subsystems. Trillium's FT/HA solution enables the standby subsystem to maintain state information through state updates from the active subsystem. These updates prevent the loss of state information, enabling an orderly switchover procedure from the failed subsystem to the standby subsystem. Figure 4 illustrates this functionality.

Trillium DFT/HA Solutions

The Trillium portable FT/HA solution has been extended to include distributing the processing load across multiple processors. These solutions, called DFT/HA, provide the high-performance and scalability demanded by network infrastructure manufacturers. The patent-pending DFT/HA software, coupled with Trillium's broad suite of communications software solutions, will enable manufacturers to build products meeting today's stringent carrier-grade telecommunications requirements. The Trillium's DFT/HA products are designed for the active/standby dual-node architecture and scaleable multi- node architecture, which allows for the number of active and standby nodes to be configured and for the active nodes to share the system load. This way, different processors can be active and standby of each other, providing an N+M configuration. Figure 5 shows this functionality. This N+M configuration goes beyond the pure distributed solution with no standby nodes in the system. In the non-redundant solution, if the active node goes down, the services provided by that node are no longer available.

The Trillium solution includes the DFT/HA Core product, and a series of technology- specific products, each called a FT/HA Protocol Specific Function (PSF) and Load Distribution Function (LDF). The LDF distributes the traffic of a protocol layer onto multiple hardware nodes to increase the traffic handling capacity of the protocol layer.

These solutions add the following functionality to the existing FT/HA solutions:

- *Flexible Architecture*
  DFT/HA allows manufacturers to design distributed FT applications, pure distributed applications and pure FT applications. A DFT/HA solution supports 2N, N+1 and N+M (same as N+1 when M is 1) redundancy schemes.
- *Load Distribution and Scalability*
  DFT/HA supports multiple active processors simultaneously with multiple standby processors, thus avoiding the need to reconfigure the system's architecture when it needs to be expanded to handle the extra traffic.
- *Smooth migration path*
  DFT/HA allows the co-existence of both distributed and non-distributed software layers within a protocol stack.

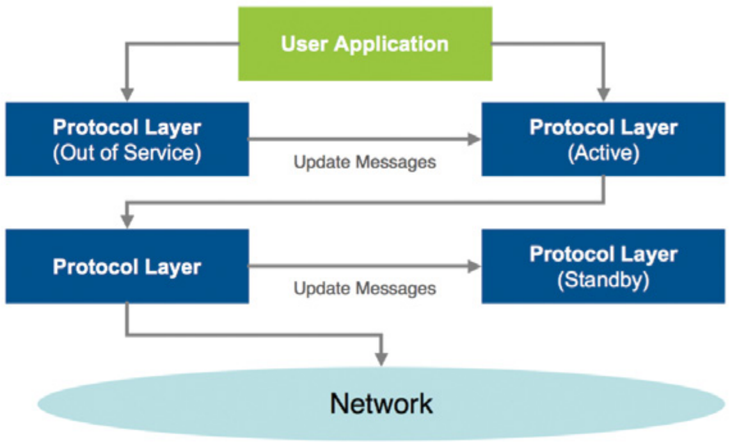

Figure 4. *Trillium's FT/HA solutions*

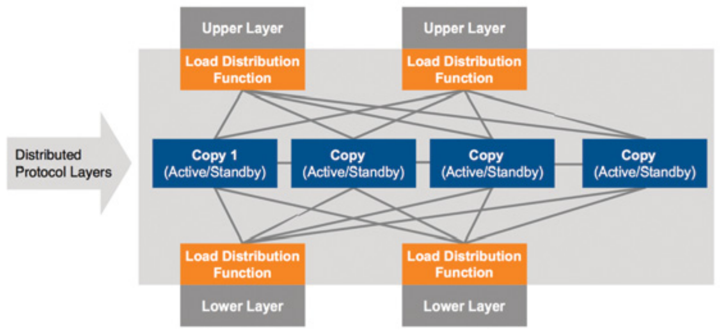
Upper Layer
Load Distribution Function
Upper Layer
Load Distribution Function
Distributed Protocol Layers
Copy 1 (Active/Standby)
Copy (Active/Standby)
Copy (Active/Standby)
Copy (Active/Standby)
Load Distribution Function
Lower Layer
Load Distribution Function
Lower Layer

Figure 5. *Trillium's DFT/HA Solution*

Summary

High-availability products and services will play an increasingly important role in the network of the future. The availability of services is a complicated function of the equipment design and also of the network design. Different design choices will result in different levels of complexity, cost, performance, potential information loss and (of course) availability. The reliability and availability of the public telephone network has set the standard against which future services from an integrated Internet and PSTN infrastructure will be measured.

For mission-critical applications such as telephony, FT/HA systems are vital. To achieve FT/HA, Trillium has undertaken a substantial development effort in recent years to enable it to offer a FT/HA solution that is flexible and platform-independent. Because it is offered in portable source form, Trillium's FT/HA software solution does not force users to limit their choice of hardware platform and operating system (unlike binary solutions). Trillium's solution gives system designers and engineers the freedom to select the optimal platform for their specific application and to focus on ways in which they can add proprietary value to their products with an optimized investment of time. In addition to supporting Trillium's existing fault-tolerant capabilities, the new patent-pending DFT/HA architecture enables high performance and scalability in a system by distributing a protocol layer across multiple processors while maintaining fault tolerance support.

Corporate Headquarters

5435 NE Dawson Creek Drive
Hillsboro, OR 97124 USA
503-615-1100 | Fax 503-615-1121
Toll-Free 800-950-0044
www.radisys.com | info@radisys.com