

INtime™

REAL-TIME FOR WINDOWS NT®

INtime® and the Windows NT Blue-Screen

Real-time Recovery After a
Windows NT Crash

Application Note

May, 1998

RadiSys Corporation

RadiSys.

5445 NE Dawson Creek Drive, Hillsboro, OR 97124 USA
+1 (503) 615-1100 fax +1 (503) 615-1150 www.radsys.com

This document is protected by US and international copyright laws.

RadiSys and INtime are registered trademarks of RadiSys Corporation.

All trademarks, registered trademarks, service marks, and trade names are the property of their respective owners.

Products described in this document may be protected by patents, or pending patent applications.

Information regarding products other than those from RadiSys has been compiled from available manufacturers' material. RadiSys cannot be held responsible for inaccuracies in such material.

RadiSys reserves the right to make changes to specifications and product descriptions at any time, without notice. Contact your local RadiSys sales office or distributor to obtain the latest specifications and product descriptions.

Copies of this document and other RadiSys literature may be obtained from:

RadiSys Corporation
5445 NE Dawson Creek Drive
Hillsboro, OR 97124

or call +1 (503) 615-1100
or visit the RadiSys website at <http://www.radisys.com>

INtime® and the Windows NT Blue-Screen

Introduction

INtime, RadiSys' real-time extension for Microsoft Windows NT, provides protection for real-time applications when a Windows NT blue-screen event (i.e., an NT crash) occurs. The INtime kernel and real-time processes maintain the ability to execute—even after the Windows NT operating system ceases to execute.

The purpose of this application note is to:

- Describe the protection provided by the INtime kernel for real-time applications in the event of a Windows NT blue-screen crash.
- Review the sample real-time applications included with INtime that provide the foundation for implementing a blue-screen recovery application. A working recovery application (including source code) is provided with the INtime product.
- List some considerations that a real-time application must take into account in order to provide its own blue-screen recovery strategy.

What Happens When an NT Blue-Screen Event Occurs

Windows NT “blue-screens” when a system fault has occurred which Windows NT deems to be unrecoverable. At this point Windows NT ceases all normal activities in progress and begins its shutdown processing. Without the INtime kernel present Windows NT would proceed to perform the processing specified by the system recovery options in effect. These options can include dumping the contents of the Windows NT operating system memory to a file and automatically restarting the system.

When the INtime kernel is present a Windows NT system fault instead results in an immediate context switch to the INtime kernel. There is no opportunity for the Windows NT kernel and its applications to over-write anything in INtime's protected real-time address space. Even if the Windows NT system tables (page tables, descriptor tables, etc.) have been corrupted, INtime's system tables will be left intact because they exist in a separate hardware task. The INtime kernel and its real-time applications will continue to execute properly.

At the time of the context switch to the INtime kernel INtime suspends Windows NT and all applications and drivers that reside within the Windows NT environment. This is possible because the Windows NT environment executes as an INtime process. The handle for the Windows NT process is cataloged in the INtime root process directory so the NT process may be resumed later, if desired. Thus, it is possible, if desired, to allow Windows NT to proceed with its normal shutdown processing activities.

Blue-Screen Console Application

INtime's blue-screen console application (*bluecon.rta*) provides access to console input and output for real-time applications following a blue-screen event. This is necessary because real-time applications' console IO—which was provided via NT—is no longer available after the crash. When an NT crash occurs *bluecon.rta* takes over the system display and keyboard. All console output will be directed to the

INtime controlled single system console. All console input will occur on the **single** input stream established by the first real-time application that performs console input after *bluecon.rta* sets up the stream.¹ *bluecon.rta* announces its setup completion by cataloging the entry “BLUECONREADY” in the root process directory. Blue-screen recovery applications need to wait for this entry to be cataloged before proceeding with any console IO.

Sample Blue-Screen Recovery Application

A working sample blue-screen recovery application (*bluesamp.rta*), with source code, is provided with INtime. This application provides real-time application developers with an example of one simple recovery strategy. The processing performed by *bluesamp.rta* is as follows:

- 1) Sets the priority of its executing thread so that it is the highest priority (non-interrupt) thread. This is important to assure that it will gain ownership of the single console input stream following an NT crash.
- 2) Waits for the root process directory entry “NT_CRASHED” indicating that NT has blue-screened.
- 3) Waits for the root process directory entry “BLUECONREADY” which indicates that *bluecon.rta* has finished setup.
- 4) Displays the recovery option menu to the system console for user selection. The options provided are:
 - a) Continue with the NT shutdown procedure by resuming the NT process.
 - b) Immediately reset the system.
- 5) Starts a time-out thread which will reset the system unless the user enters a valid recovery option before the (30 second) time-out occurs.
- 6) Waits for user to select one of the recovery options.

Some Considerations for Blue-Screen Recovery Applications

The sample blue-screen recovery application provided with INtime is useful for demonstrating how to establish console IO and how to continue with the Windows NT shutdown procedure. This is simply a starting point for the design of more sophisticated recovery scenarios.

Some other considerations for designing a recovery strategy follow:

- It may be important for some critical real-time processes to be notified when an NT crash occurs. These critical real-time processes might need to include a thread which receives notification from the blue-screen recovery process—or they might wait on the BLUECONREADY catalog entry (as described in the sample application above). These critical processes might, in turn, notify the recovery process regarding if and when they can be safely terminated, etc.

¹ Note that any console input calls that are pending when NT crashes on an INtime 1.x system will hang the calling real-time threads. This behavior may be changed in a future INtime release so that pending console input calls will be returned with an error condition when NT crashes.

- Although the sample recovery application performs console input and output, this is not a requirement. The recovery process might just communicate with other real-time processes, as needed, and then continue with an NT shutdown when it is deemed safe.
- Console output performed after the blue-screen will be funneled through INtime's blue-screen console application. Console input is available only to the first real-time thread that performs console input after the crash. That is why the recovery application should set its priority higher than all non-interrupt threads.
- When an NT crash occurs real-time processes will lose their ability to communicate with NT processes. These real-time processes might want to be notified when an NT crash occurs so that they can take the appropriate action.
- Any communication channels (i.e., serial, Ethernet, etc.) controlled directly by the INtime kernel and its real-time applications will continue to operate properly and can be utilized to send a "service me" message.

It is important to design the real-time side of your application so that those parts which depend on the Windows NT subsystem being present do not impede any critical threads in your application which must continue to run in the event of a blue-screen crash. Critical threads should not be designed to wait indefinitely on threads that communicate directly with a Windows NT process.

Conclusion

The INtime kernel runs in a completely separate x86 hardware task, which is similar to running on a second independent processor. This method of operation allows the INtime kernel to define distinct and separate data structures for controlling real-time interrupts, virtual memory, and the hardware task state (the machine context). INtime maintains separate GDT, IDT, LDTs, and page table entries in order insure the INtime environment is safe and secure in the event of a Windows NT blue-screen crash.

Since INtime systems are "NT-centric," it is necessary to take into account the degradation in the system functionality that will result when the Windows NT operating system "blue-screens." Those elements of your real-time system which share information with an NT process will, unfortunately, be stalled. Critical real-time processes which must continue to run following a blue-screen event cannot be designed to rely on guaranteed access to the Windows NT environment. This means you should make a design decision to accommodate a "crash scenario" that will result in gracefully degrading some aspects of the system without impeding any critical functionality. The system can then be restored to full functionality at a time that is convenient for the system operator and machine function, avoiding the reactionary situation caused by an unexpected or inconvenient crash of the Windows NT operating system.