



# **Fault Tolerance in the NonStop Cyclone System**

Scott Chan  
Robert L. Jardine

Technical Report 90.7  
June 1990  
Part Number: 48008



# **Fault Tolerance in the NonStop Cyclone System<sup>1</sup>**

**Scott Chan  
Robert L. Jardine**

**Tandem Computers Incorporated  
19333 Valco Parkway  
Cupertino, CA 95014**

## **ABSTRACT**

This paper surveys the methods used to achieve fault tolerance in the Tandem NonStop Cyclone system, a multiprocessor mainframe designed for high performance in simultaneous transaction, query, and batch processing. Fault detection and recovery are performed by various hardware and software techniques, ensuring data integrity and continuous system operation for critical commercial applications.

Tandem, NonStop, Cyclone, Dynabus, Dynabus+, Guardian-90, and VLX are trademarks of Tandem Computers Incorporated.

---

<sup>1</sup>A shorter version of this technical report was previously published in the proceedings of the Spring Conference of the Institute of Electronic, Information, and Communication Engineers, Chuo University, Tokyo, Japan, March 1990.



# Fault Tolerance in the NonStop Cyclone System

## 1. Introduction

The NonStop Cyclone system is a fault-tolerant multiprocessor mainframe designed for simultaneous transaction, query, and batch processing [1,2]. Each system (Figure 1) consists of four to sixteen processors loosely coupled by dual high-speed busses (Dynabus). Sections of four processors are interconnected by fiber optic cables (Dynabus+) and may be geographically distributed. Each processor has its own memory and controls two to four I/O channels. Fault detection is performed primarily by the hardware, and fault recovery is performed by the message-based operating system. The system can tolerate a single fault in a processor, peripheral controller, power supply, or cooling system. Failed components can be serviced on-line without disrupting processing.

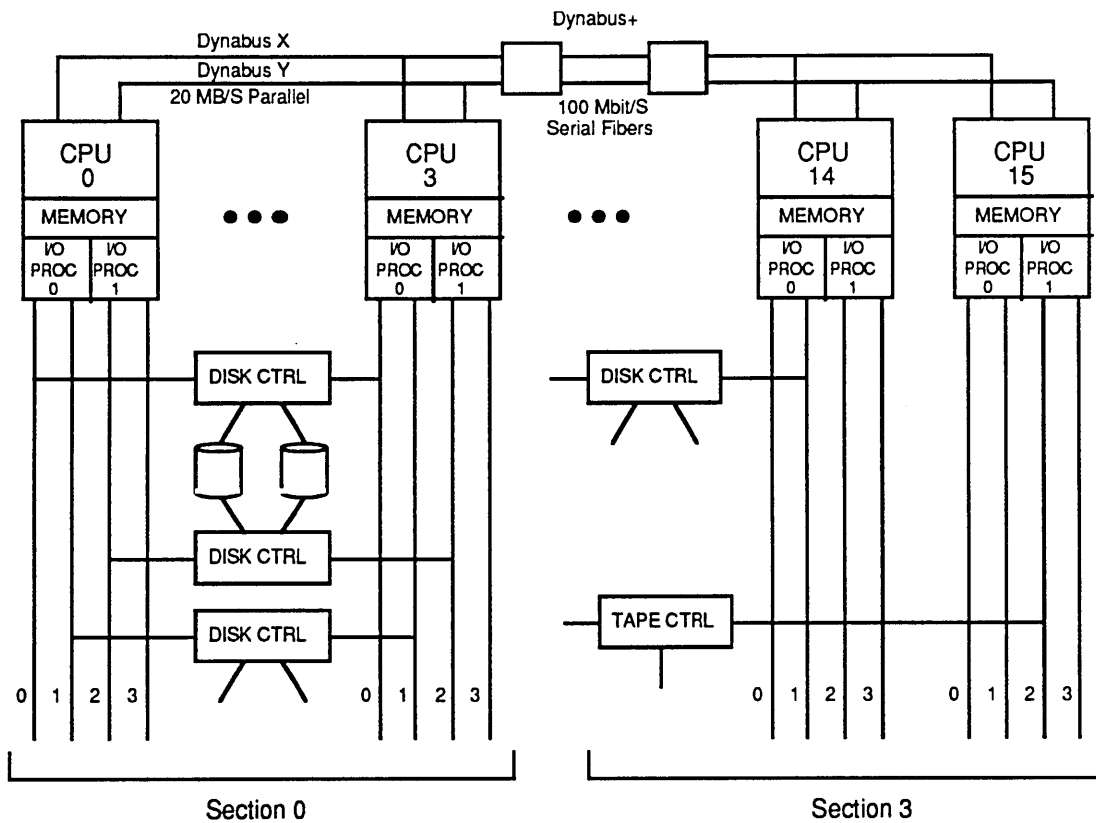


Figure 1. Cyclone System Architecture.



Four design principles contribute to fault-tolerant system operation: 1) fault detection by both hardware and software; 2) *fail-fast* and fault-containment designs that prevent corruption of user databases by faulty subsystems; 3) high reliability through safe design and error-recovery features within subsystems; and 4) continuous system operation achieved through mechanisms that recover from individual component failures. This paper describes how these principles apply through various levels of the NonStop Cyclone System.

## 2. System-level Fault Tolerance

As shown in Figure 1, all major components of a NonStop Cyclone system are replicated. The Guardian-90 operating system manages these components and, if one fails, arranges for its function to be taken over by another component. The replicated components are not merely redundant; components operate concurrently to enhance system performance while providing fault tolerance.

For example, in normal operation, both Dynabusses carry message traffic. In the event of a single Dynabus failure, all traffic is routed onto the remaining Dynabus. The Dynabus+, the fiber-optic connection between sections, is configured in a dual-ring arrangement, allowing tolerance of selected multiple failures. Power supply and cooling blower loads are distributed so that the effect of a single failure is confined to at most a single processor or I/O controller.

I/O controllers are checked by lock-stepped, duplicated microprocessors with self-checking comparison circuits. I/O controllers are dual-ported to separate processors, maintaining a path to the I/O device even when one processor or I/O channel is down. These replicated devices and busses ensure that no single failure will cause the entire system or any part of a database to be unavailable. Through mirroring, disk drives can also be configured to be tolerant of single faults.

At the software level, operating system processes are programmed as process pairs [3,4]-a primary process and a backup process, executing in different processors. The primary process performs the actual work of the process pair, occasionally sending the backup a checkpoint message containing its current state. If the primary fails (for example, if the processor in which it is executing fails to send its periodic *I'm Alive* message and is declared down), the backup process takes over and resumes processing at the point of the last checkpoint received. In addition to providing tolerance of single hardware faults, the message-based, process-pair structure of the software also provides tolerance of intermittent software faults, a feature not provided by hardware-only fault tolerant systems.





Finally, an even greater degree of protection and ease-of-programming is provided by the Transaction Monitoring Facility (TMF) [4]. TMF allows an application to package its computations and database updates into *atomic* units. The integrity of the database is then protected even in the face of multiple faults, prolonged power failures, and the like.

### 3. Fault Detection within a Cyclone Processor

Once system-level fault tolerance is provided for, the only requirement at the processor level is to halt quickly after detecting a failure. This *fail-fast* principle has two benefits: 1) it contains the error to the failing processor, preventing data corruption; and 2) it minimizes service delays while the backup processes take over. Faults within each processor are detected by a variety of hardware and microcode methods.

Parity checking is used extensively to detect single-bit errors. Parity is propagated through devices that do not alter data, such as memories, control signals, busses, and registers. Parity prediction is used on devices that alter data, such as arithmetic units and counters. Predicted parity is based strictly on a device's data and parity inputs; it does not rely on the device's outputs, which may be faulty. Thus, an adder might generate an erroneous sum, but the parity that accompanies the sum will correspond to the correct result. Parity checkers downstream will then detect the error.

The hardware multiplier is protected by a novel technique similar to Recomputation with Shifted Operands (RESO) [5]. After each multiplication, a second multiplication is initiated with the operands exchanged and one operand shifted. Microcode compares the two results whenever the multiplier is needed again or before any data leaves the processor. Unlike other implementations of RESO, these checking cycles incur almost no performance penalty because they occur concurrently with unrelated execution steps.

Invalid-state checking or duplication-and-comparison are used in sequential state machines. Checksums protect multiple-word transmissions, such as the inter-processor bus and I/O channel. Watch-dog timers and microcode polling monitor operations that take many cycles to complete.

If the processor hardware detects a fault from which it cannot recover, it shuts itself down within two clock cycles, before it can transmit any corrupt data along the inter-processor bus or I/O channel. The error is flagged in one or more of the approximately 300 error identification registers, allowing quick fault isolation to any of the 500 hardware error detectors in each processor (see Figure 2).



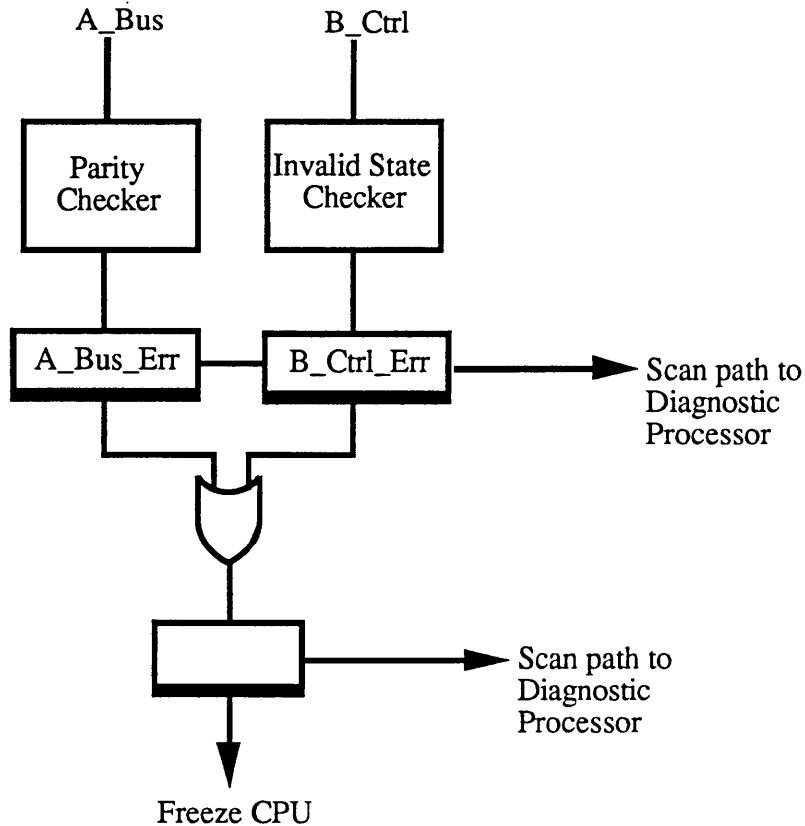


Figure 2. Cyclone Error Identification Registers.

The microcode and operating system both perform numerous consistency checks, such as invalid instruction detection and address bounds checking. Also, the microcode executes processor diagnostic routines during idle situations. If the microcode or operating system detects an unrecoverable error, it immediately executes a HALT instruction and transmits an error code to the Remote Maintenance Subsystem.

#### 4. Fault Recovery within a Cyclone Processor

System-level fault tolerance can be achieved without incorporating any fault tolerance capabilities within the processors themselves. However, each Cyclone processor has numerous on-line recovery mechanisms that allow it to withstand certain types of hardware faults, nearly doubling the calculated mean-time-to-failure of each processor and dramatically reducing customer service costs.



Large static RAM arrays, such as data and instruction caches, main control store, and I/O subsystem control store, can recover from intermittent ("soft") data errors by reloading from alternate copies. For example, a soft error in the data cache is corrected by refilling the block from main memory. In addition, the main control store and caches have spare RAMs that automatically replace hard-failed RAMs [6] (see Figure 3).

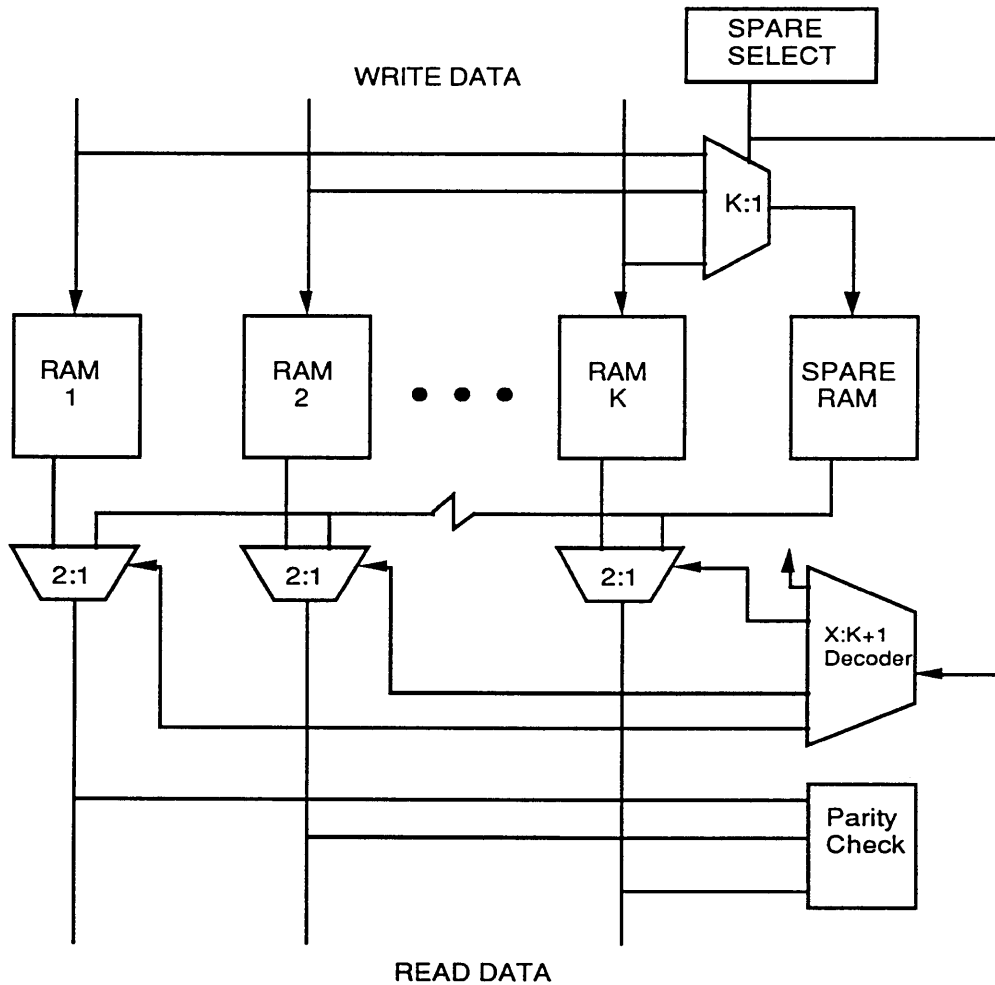


Figure 3. Cyclone Spare RAM Mechanism.

A single-error-correcting, double-error-detecting code protects dynamic RAMs in main memory. This code incorporates both data and address bits, so that addressing failures are detected as well as RAM failures. An asynchronous microcode process periodically checks for correctable memory errors; these errors are logged and the memory areas are scrubbed by the operating system. Inter-processor bus packets and I/O transfers are retried by the operating system if checksum errors occur.



## 5. Diagnostic Facilities

Each processor has a dedicated microprocessor that executes quick diagnostics, scans the initial state into the processor, loads bootstrap microcode into the writable control stores, and initiates system *cold load*. It can generate and collect pseudo-random scan-test signatures for quick fault detection and isolation, and it serves as the interface to a separate fault-tolerant Remote Maintenance Subsystem. This subsystem monitors and logs events in a running system, supports diagnosis of failures anywhere in the system, and optionally dials out to report problems to a Tandem customer support center.

## 6. Conclusions

Commercial data processing systems are moving increasingly toward on-line databases and applications. These systems require continuous system availability, secure transactions, and error-free databases. The NonStop Cyclone system provides the highest levels of performance, system availability, and data integrity for critical commercial processing needs.

## References

- [1] S. Chan and R. W. Horst, "Parallelism in the Instruction Pipeline", in *High Performance Systems*, Dec. 1989.
- [2] R. W. Horst, R. L. Harris, and R. L. Jardine, "Multiple Instruction Issue in the NonStop Cyclone Processor", *Seventeenth International Symposium on Computer Architecture*, Seattle, WA, May 1990; also Tandem Technical Report 90.6, Cupertino CA, June 1990.
- [3] J. Bartlett, "A NonStop Kernel", in *proc. Eighth Symposium on Operating System Principles*, pp. 22-29, Dec. 1981.
- [4] J. Bartlett, W. Bartlett, R. Carr, D. Garcia, J. Gray, R. Horst, R. Jardine, D. Lenoski, and D. McGuire, "Fault Tolerance in Tandem Computer Systems", Tandem Technical Report 90.5, Cupertino CA, May 1990.






- [5] G. S. Sohi, M. Franklin, and K. K. Saluja, "A Study of Time-Redundant Fault Tolerance Techniques for High-Performance Pipelined Computers", in *proc. Nineteenth International Symposium on Fault Tolerant Computing*, Chicago, IL, pp. 436-443, June 1989.
- [6] R. W. Horst, "Reliable Design of High-speed Cache and Control Store Memories", in *proc. Nineteenth International Symposium on Fault Tolerant Computing*, Chicago, IL, pp. 259-266, June 1989.





Distributed by  
 **TANDEM**  
Corporate Information Center  
10400 N. Tantau Ave., LOC 248-07  
Cupertino, CA 95014-0708