

TANDEM
COMPUTERS
INCORPORATED

ONE MARKET PLAZA • SUITE 1516 • SPEAR STREET TOWER • SAN FRANCISCO, CALIF. 94105 • (415) 777-1230

February 20, 1980

Mr. Robert Brilliant
Purchasing Department
University of California
2405 Bowditch Street
Berkeley, California 94720

Dear Mr. Brilliant:

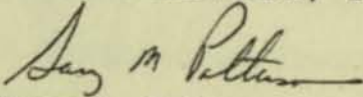
I am pleased to respond to the University of California's Request for Quotation for an Integrated Pest Management (IPM) System with the accompanying proposal. This proposal offers the University a complete functional solution to the Category A and B requirements set forth by the University.

The Tandem solution will not only address the immediate hardware and software requirements, but offers many other advantages that have immediate and long-term benefits to the IPM program. I encourage a complete review of all aspects of the Tandem solutions and encourage further discussion of any aspects of it.

My thanks to those University personnel who have assisted our understanding of your needs. Tandem Computers looks forward to a mutually beneficial relationship.

Sincerely,

TANDEM COMPUTERS, INC.


Gary M. Patterson
Marketing Representative

GMP:sl
Encl.

Executive
Overview

Pricing
Detail

Options

Questions

Tandem

TABLE OF CONTENTS

Request for Quotation Summary Sheet

UNIVERSITY OF CALIFORNIA

Executive Summary

System: INTEGRATED PEST MANAGEMENT

System Options

Answers to Specific RFQ Questions

About Tandem Computers, Inc.

System Detail

General

Hardware

Software

QUADRA Operating System

QUADRA/QUADRO Network

February 20, 1980

FACSIMILE Transaction Processing

IMMEDIATE Data Base Record Manager

Programming Language

RFQ Summary

Executive
Overview

Pricing
Detail

Options

Questions
Answers

Tandem

TABLE OF CONTENTS

Request for Quotation Summary Sheet

Executive Overview

System Pricing Detail

System Options

Answers to Specific RFQ Questions

About Tandem Computers, Inc.

System Detail

General

Hardware

Software

GUARDIAN Operating System

GUARDIAN/EXPAND Network

PATHWAY Transaction Processing

ENSCRIBE Data Base Record Manager

Programming Languages

RFQ Summary

No charge for
forms apply A
quotation B

**MUST
BE
SEPARATELY
TOTAL PRICE**

on
A
Pricing

S,

ishment under
at affirmatively
in all aspects of
arge, to provide
our affirmative

RETURN ONE COPY TO

UNIVERSITY OF CALIFORNIA
PURCHASING DEPARTMENT
2405 BOWDITCH STREET
BERKELEY, CALIFORNIA 94720

DATE 1/23/80

NUMBER LRQ/GS/RB

IS YOUR ADDRESS
INCLUDING ZIP
CODE CORRECT?

TANDEM COMPUTERS INCORPORATED
ONE MARKET PLAZA
SUITE 1516 - SPEAR ST. TOWER
SAN FRANCISCO, CA 94105
ATTN: GARY PATTERSON

TO BE CONSIDERED YOUR QUOTATION
MUST BE RECEIVED BY FEB. 20, 1980; 5:00 PM
IF FURTHER INFORMATION IS REQUIRED, PLEASE CONTACT
MR. R. BRILLIANT (415) 642-0881.

Please quote your lowest price for the material, and/or services, to be delivered as specified below. Any deviation from the specifications must be identified and fully described. No charge for package, for drayage, or for any other purpose will be allowed over and above the prices quoted on this sheet. University of California standard purchase order terms and conditions apply. A copy of these conditions will be furnished on request. The right is reserved to accept or reject quotations on each item separately, or as a whole, and to waive any irregularities in a quotation. If unable to quote, please return this form so marked.

ARTICLES SUPPLIED ON THIS FORM SHOULD BE PRODUCED OR MANUFACTURED IN THE UNITED STATES, IF OF FOREIGN MAKE, PLEASE SPECIFY ORIGIN.

UNLESS OTHERWISE SPECIFIED PRICES WILL BE F.O.B. UNIVERSITY OF CALIFORNIA

UNIT PRICE MUST
BE EXTENDED

SHOW ANY APPLICABLE TAX SEPARATELY

CASH DISCOUNT _____ PER CENT FOR PAYMENT WITHIN _____ DAYS FROM RECEIPT OF GOODS
OR INVOICE, WHICHEVER IS LATER.

UNIT PRICE TOTAL PRICE

THE UNIVERSITY OF CALIFORNIA, BERKELEY, DIVISION OF BIOLOGICAL CONTROL, REQUESTS YOUR BID FOR A COMPUTER SYSTEM, AS DESCRIBED IN THE ATTACHED SPECIFICATIONS.

For Pricing on
Categories A
and B, see Pricing
Section of
Proposal.

THE FOLLOWING CLAUSES ARE A PART OF THIS REQUEST:

1. IN ACCORDANCE WITH THE ATTACHED ELECTRICAL CLAUSE?
YES X NO _____
2. PLEASE STATE FULL WARRANTY: See Question ABC-3.
3. PARTS AND SERVICE MAY BE OBTAINED FROM: See Question ABC-2.
4. ARE THE PRICES YOU HAVE QUOTED HEREIN THE LOWEST OFFERED TO ANY FEDERAL, STATE, MUNICIPAL OR SIMILAR INSTITUTION ACCOUNT? YES _____ NO X IF NO, PLEASE EXPLAIN.

The GSA Schedule is lower.

5. PLEASE SUBMIT DESCRIPTIVE LITERATURE AND YOUR PUBLISHED PRICE LIST IF AVAILABLE.

Descriptive literature and specific pricing enclosed.

TABLE ELECTRICAL EQUIPMENT (120 VOLTS AND UP TO 15 AMPERES) REQUIRES A THREE-WIRE CORD NEMA CAP, WITH GROUND CONDUCTOR PERMANENTLY ATTACHED TO NON-CURRENT-CARRYING METAL PARTS, IN COMPLIANCE WITH THE CALIFORNIA ADMINISTRATIVE CODE, TITLE 24, PART 3, OR DOUBLE INSULATED, STATE OF CALIFORNIA TITLE 8, ARTICLE 11, SECTION 2395.45 AND MUST CONFORM TO ALL FEDERAL OSHA SAFETY STANDARDS. (REV. 2/1/77)

DELIVERY WILL BE MADE IN _____ DAYS AFTER RECEIPT OF ORDER.

SHIPPING WEIGHT _____ LBS. METHOD OF SHIPMENT _____

As a supplier of goods or services to the University of California I/we certify that racially segregated facilities will not be maintained nor provided for employees at any establishment under my/our control, and that I/we adhere to the principles set forth in Executive Orders 11246 and 11375, and undertake specifically to maintain employment policies and practices that affirmatively promote equality of opportunity for minority group persons and women, to take affirmative steps to hire and promote women and minority group persons at all job levels and in all aspects of employment, to communicate this policy in both English and Spanish to all persons concerned within the company, with outside recruiting services, and the minority community at large, to provide the University on request a breakdown of our total labor force by ethnic group, sex, and job category, and to discuss with the University our policies and practices relating to our affirmative action program.

2/20/80

(415) 777-1230

DATE

TELEPHONE NO

INCLUDING AREA CODE

SIGNATURE OF FIRM REPRESENTATIVE

Executive
Overview

EXECUTIVE OVERVIEW

The Tandem 16 Computer System proposed to the University of California for its Integrated Pest Management Program possesses specific strengths that will allow the University to meet its near and long-term expectations. Many of these capabilities are unique to Tandem, as your studies will undoubtedly conclude.

Tandem has proposed an initial network of 3 nodes, each with its exclusive NonStop™ dual processor system. The Central Site node has been configured to accommodate the activities set forth for both the Central and District (1) systems proposed at that site. The University will enjoy all the capabilities requested in the RFQ at a significant cost savings over other configurations and enjoy many benefits only possible from Tandem. For the Central Site, Tandem has proposed two processors, each with 768K/Bytes of memory, and 600 megabytes of removable disc storage. If the University so chooses, each processor can be dedicated to specific functions; i.e., development and operations.

The remote district sites each contain a dual processor, with 384K/Bytes of memory each and 50 megabytes of removable disc storage. With the appropriate communication links, the University will possess a complete fault tolerant system and enjoy true NonStop™ performance with optimum use of all resources.

The flexibility of such an approach leaves the University in a position to manage its total computer resource to meet daily needs that are ever changing and also adapts to any re-emphasis over the long term. The University's investment could not be better protected! Information contained in this proposal will add insight to Tandem's unique approach.

The Tandem System was developed during the mid-Seventies to address the emerging requirements of the Eighties. The IPM requirements might well have represented Tandem's design objectives.

A capsule look at some of these significant benefits as they may apply to the University is presented on the following pages.

IPM REQUIREMENT

- . High System Uptime
- . End User Satisfaction
- . Reliability

- . Ease of application software development and modification
- . Fast start-up

TANDEM SOLUTION

Tandem's exclusive NonStopTM, fault-tolerant hardware and software allows for continuous processing, which in turn protects critical data, increases end user confidence, and reduces many types of operational costs. User satisfaction is normally equated to system success.

Unlike some systems, which have been "patched" to provide a "hot standby," the T-16 utilizes fully and efficiently its multiple processors and memory under normal conditions.

Use of Tandem's software, PATHWAY, for screen development and transaction processing; ENSCRIBE for the development and maintenance of the data base; and GUARDIAN/EXPAND "off the shelf" networking with no programming required, will reduce the initial development effort significantly. Lesser skilled personnel can be used for these efforts. The net result will be faster development at a much lower initial cost, followed by much lower ongoing effort and cost.

Today, with reduced hardware costs providing a rapidly improving price/performance ratio, every data processing operation must address the sky rocketing software and personnel costs. This is best accomplished through their initial planning and commitment.

IPM REQUIREMENT

TANDEM SOLUTION

. Network

The IPM Program truly requires an effective Networking System. GUARDIAN/EXPAND will provide capability which is transparent to any application; it will allow development and control to be centered anywhere designated; and it is off the shelf.

In the University's initial network, Tandem would recommend a communications line between the two remote districts, thereby completing the network and protecting against node failure due to a specific line problem in the Network. The node would seek a secondary path to other nodes in the case of any problems.

. Modularity

The questions of "when" and "how much" growth of the IPM Program are easily answered as needed by Tandem's unequalled modularity - and with complete transparency to the end-user and system software. Incremental costs are very competitive; future upgrades can be very selective. The University's investment in hardware and software is protected. Staff efforts will contribute to new end user benefits, not re-systemitizing and reprogramming prior efforts.

. Data Base

Tandem's fast, efficient relational data base is implemented simply, is transparent to any application code, can be migrated to any system at will, and is modifiable on-line (no costly reorganizations).

Pricing
Detail

Options

Questions

Tandem

IPM REQUIREMENT

TANDEM SOLUTION

. The Vendor

Finally, the capability, interest and direction of the vendor must be considered. Tandem offers the University a relationship with a locally based computer company. Tandem offers a high-level of expertise in its local System Engineering and Customer Engineering staffs. A dedicated Education department and Regional and Corporate specialists further assure the University the consultation and support necessary for success.

Tandem believes the major near and long-term benefits that will be derived by use of Tandem hardware and software should result in a very competitive price performance ratio. We look forward to the careful analysis of all costs which will comprise such a program over a period of years.

Pricing
Detail

UNIVERSITY OF CALIFORNIA
 INTEGRATED PEST MANAGEMENT

HARDWARE AND SOFTWARE PRICING RECAP

Categories A and B

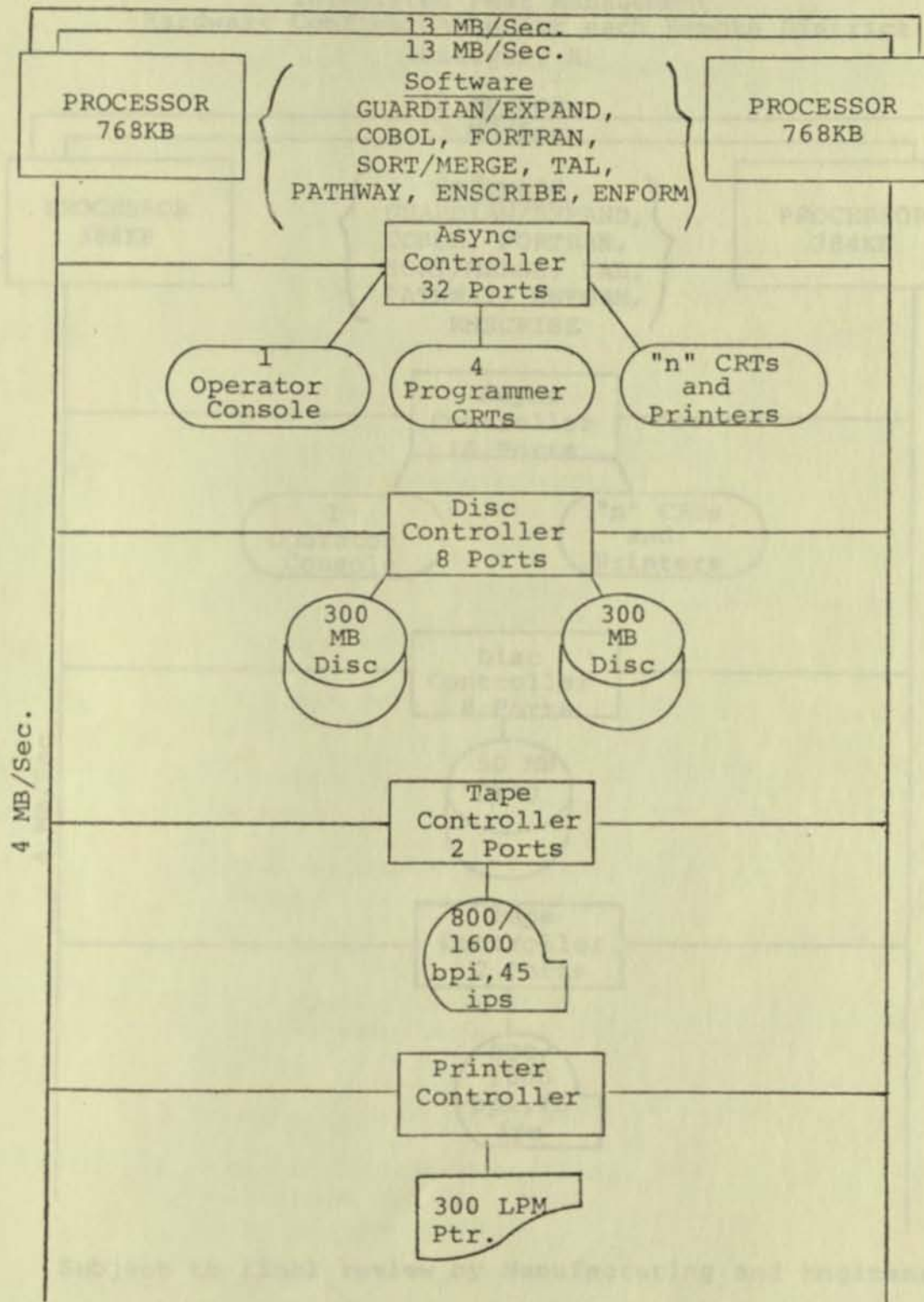
	<u>Totals</u>	
	<u>Purchase Price</u>	<u>Monthly Maintenance</u>
<u>Category A</u>		
Central and District #1 Systems at Central Site	\$262,025	\$2,157
District #2 System	136,550	980
District #3 System	136,550	980
Total, Category A	<u>\$535,125</u>	<u>\$4,117</u>
<u>Category B</u>		
Central System	\$ 59,600	\$ 560
District #1 System	44,700	420
District #2 System	44,700	420
Distirct #3 System	44,700	420
Total, Category B	<u>\$193,700</u>	<u>\$1,820</u>
TOTAL, CATEGORIES A and B	<u>\$728,825</u>	<u>\$5,937</u>

ESTIMATED FREIGHT CHARGES

All Category A Equipment	9000 lbs.	\$1,050
All Category B Equipment	2350 lbs.	225
		<u>\$1,275</u>

Subject to final review by Manufacturing and Engineering

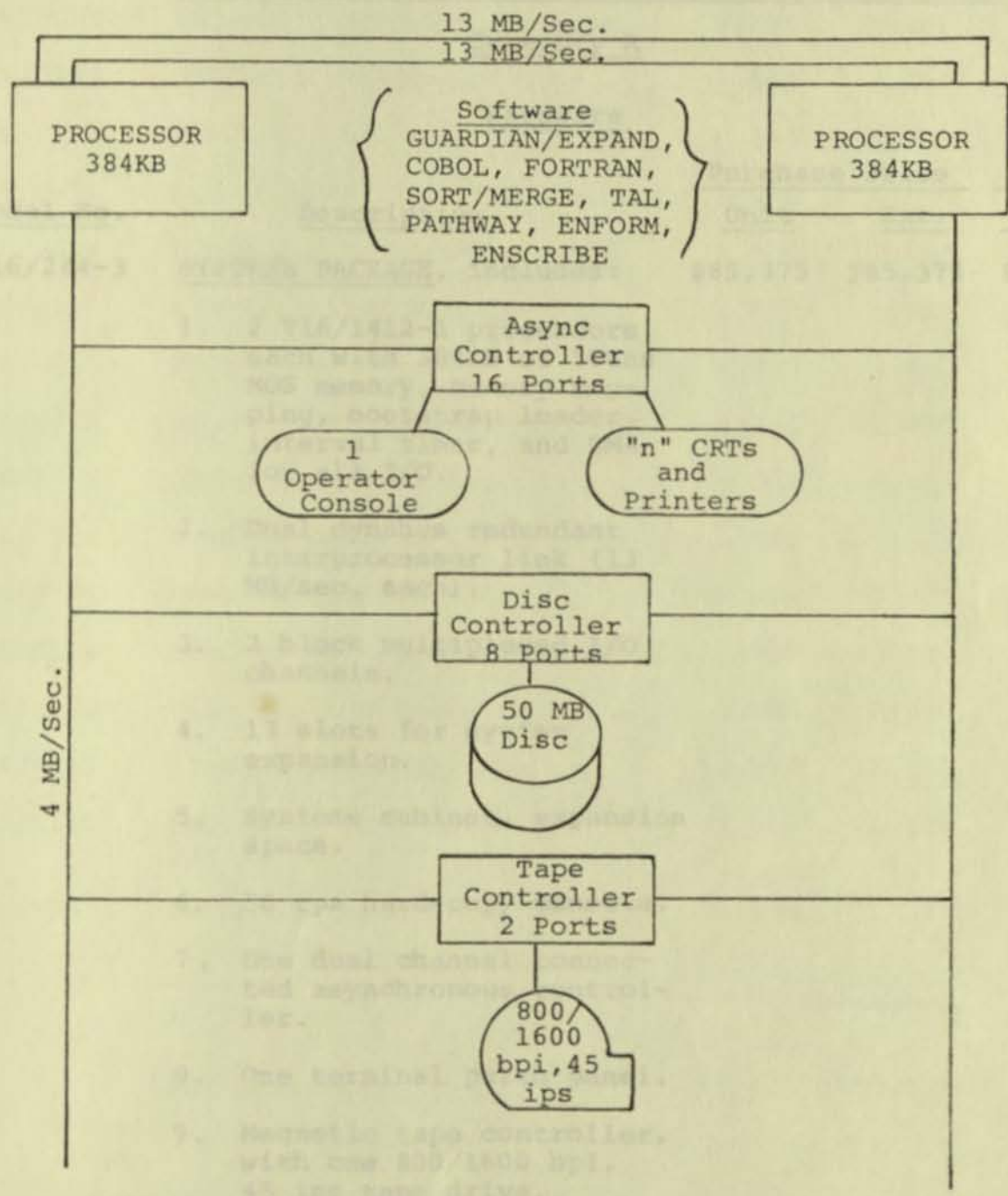
UNIVERSITY OF CALIFORNIA
 Integrated Pest Management
 Hardware Configuration
 for the Central and District Requirement at Central Site
 (Category A)



Subject to final review by Manufacturing and Engineering

UNIVERSITY OF CALIFORNIA

Integrated Pest Management
Hardware Configuration for each Remote District
(Category A)



Subject to final review by Manufacturing and Engineering

UNIVERSITY OF CALIFORNIA
INTEGRATED PEST MANAGEMENT

Hardware and Software Pricing Detail

Central and District Requirement at Central Site

Category A

Hardware

Qty.	Model No.	Description	Purchase Price		Monthly Maintenance	
			Unit	Ext.	Unit	Ext.
1	T16/244-3	SYSTEMS PACKAGE, includes:	\$85,375	\$85,375	\$ 652	\$ 652
		1. 2 T16/1412-1 processors, each with 384KB of 500ns MOS memory, memory mapping, bootstrap loader, interval timer, and DMA for all I/O.				
		2. Dual dynabus redundant interprocessor link (13 MB/sec. each).				
		3. 2 block multiplexed I/O channels.				
		4. 13 slots for system expansion.				
		5. Systems cabinet, expansion space.				
		6. 30 cps hard-copy console.				
		7. One dual channel connected asynchronous controller.				
		8. One terminal patch panel.				
		9. Magnetic tape controller, with one 800/1600 bpi, 45 ips tape drive.				
		10. Two battery packs for memory backup.				

Central and District

<u>Qty.</u>	<u>Model No.</u>	<u>Description</u>	<u>Purchase Price</u>		<u>Monthly Maintenance</u>	
			<u>Unit</u>	<u>Ext.</u>	<u>Unit</u>	<u>Ext.</u>
2	T16/2412	Memory Module, MOS, 384KB	\$12,800	\$25,600	\$ 120	\$ 240
2	T16/2001	DECIMAL ARITHMETIC Package	2,000	4,000	20	40
2	T16/2002	ENSCRIBE Microcode	1,500	3,000	N/C	N/C
2	T16/2003	COBOL Microcode	500	1,000	N/C	N/C
2	T16/2005	FLOATING POINT ARITHMETIC Pkg.	2,000	4,000	20	40
2	T16/2006	FORTRAN Microcode	500	1,000	N/C	N/C
2	T16/2007	GUARDIAN/EXPAND Microcode	2,000	4,000	N/C	N/C
2	T16/2010	PATHWAY Microcode	2,000	4,000	N/C	N/C
1	T16/6202	Byte Synchronous Controller	5,800	5,800	26	26
2	T16/6302	Async Extension Board	4,300	8,600	20	40
1	T16/7504	Disc Patch Panel, Std.	775	775	N/C	N/C
1	T16/3105	Disc Controller	10,500	10,500	53	53
2	T16/4104	Disc Drive, 300MB unformatted	26,500	53,000	189	378
1	T16/3302	Line Printer Controller	2,800	2,800	18	18
1	T16/5502	Line Printer, 300 lpm	11,500	11,500	140	140
4	T16/6520	Terminal, 2000 characters per screen	2,950	11,800	20	80
1	T16/7501	Terminal Patch Panel, Std.	775	775	5	5
TOTAL HARDWARE AND SOFTWARE				<u>237,525</u>		<u>1,712</u>
TOTAL HARDWARE				\$237,525		\$1,712

*One-time license fee per organization not applicable - fee already paid by University

Software

Qty.	Model No.	Description	One-Time License Fee		Monthly Maintenance	
			Unit	Ext.	Unit	Ext.
1	T16/9007	GUARDIAN/EXPAND Operating System, Network Operating System, includes TAL (Upgrade of existing University of California license)	\$10,000	\$ 10,000	\$ 100	\$ 100
1	T16/9002	ENSCRIBE, Data Base Manager	*	*	60	60
1	T16/9102	ENFORM, Query/Report Writer	*	*	70	70
1	T16/9103	PATHWAY, Transaction Process.	8,500	8,500	85	85
1	T16/9201	COBOL, ANSI '74	*	*	70	70
1	T16/9202	FORTTRAN, ANSI '78	6,000	6,000	60	60
TOTAL SOFTWARE				24,500		445

(1) Installation location may necessitate an additional charge

TOTAL HARDWARE AND SOFTWARE \$262,025 \$2,157

*One-time license fee per organization not applicable - fee already paid by University

UNIVERSITY OF CALIFORNIA
UNIVERSITY OF CALIFORNIA

INTEGRATED PEST MANAGEMENT

Hardware and Software Pricing Detail

Hardware Pricing Detail

For Central System Requirement

Category B

Hardware

<u>Qty.</u>	<u>Model No.</u>	<u>Description</u>	<u>Purchase Price</u>		<u>Monthly (1) Maintenance</u>	
			<u>Unit</u>	<u>Ext.</u>	<u>Unit</u>	<u>Ext.</u>
8	T16/6520	CRT terminal, 2000 characters per screen	\$2,950	\$23,600	\$20	\$160
8	T16/5508	Serial Character Printer, 200 cps	4,500	36,000	50	400
				<u>\$59,600</u>		<u>\$560</u>

(1) installation location may necessitate an additional charge

UNIVERSITY OF CALIFORNIA
INTEGRATED PEST MANAGEMENT

Hardware and Software Pricing Detail

For Each Remote (2) District Requirement

Category A

Hardware

Qty.	Model No.	Description	Purchase Price		Monthly Maintenance	
			Unit	Ext.	Unit	Ext.
1	T16/244-3	<u>SYSTEMS PACKAGE</u> , includes:	\$85,375	\$85,375	\$ 652	\$ 652
		1. 2 T16/1412-1 processors, each with 384KB of 500ns MOS memory, memory mapping, bootstrap loader, interval timer, and DMA for all I/O.				
		2. Dual dynabus redundant interprocessor link (13 MB/sec. each).				
		3. 2 block multiplexed I/O channels.				
		4. 13 slots for system expansion.				
		5. Systems cabinet, expansion space.				
		6. 30 cps hard-copy console.				
		7. One dual channel connected asynchronous controller.				
		8. One terminal patch panel.				
		9. Magnetic tape controller, with one 800/1600 bpi, 45 ips tape drive.				
		10. Two battery packs for memory backup.				

Remote (2) District

Qty.	Model No.	Description	Purchase Price		Monthly Maintenance	
			Unit	Ext.	Unit	Ext.
2	T16/2001	DECIMAL ARITHMETIC Package	\$ 2,000	\$ 4,000	\$ 20	\$ 40
2	T16/2002	ENSCRIBE Microcode	1,500	3,000	N/C	N/C
2	T16/2003	COBOL Microcode	500	1,000	N/C	N/C
2	T16/2005	FLOATING POINT ARITHMETIC Pkg.	2,000	4,000	20	40
2	T16/2006	FORTRAN Microcode	500	1,000	N/C	N/C
2	T16/2007	GUARDIAN/EXPAND Microcode	2,000	4,000	N/C	N/C
2	T16/2010	PATHWAY Microcode	2,000	4,000	N/C	N/C
1	T16/6202	Byte Synchronous Controller	5,800	5,800	26	26
1	T16/6302	Asynchronous Extension Board	4,300	4,300	20	20
1	T16/7504	Disc Patch Panel, Std.	775	775	N/C	N/C
1	T16/3102	Disc Controller (Single Port Drive)	4,800	4,800	22	22
1	T16/4102	Disc Drive, 50MB formatted	14,500	14,500	180	180
TOTAL HARDWARE AND SOFTWARE				\$136,550		\$980

UNIVERSITY OF CALIFORNIA
Software

Qty.	Model No.	Description	One-Time License Fee		Monthly Maintenance	
			Unit	Ext.	Unit	Ext.
1	T16/9007	GUARDIAN/EXPAND Operating System, Network Operating System, includes TAL (Upgrade of existing University of California license)	**	**	**	**
1	T16/9002	ENSCRIBE, Data Base Manager	**	**	**	**
1	T16/9102	ENFORM, Query/Report Writer	**	**	**	**
1	T16/9103	PATHWAY, Transaction Process.	**	**	**	**
1	T16/9201	COBOL, ANSI '74	**	**	**	**
1	T16/9202	FORTTRAN, ANSI '78	**	**	**	**

**License fee charged to University of California only once (see Central Site); no Maintenance required when Remote system is not development system.

TOTAL HARDWARE AND SOFTWARE

\$136,550

\$980

UNIVERSITY OF CALIFORNIA
INTEGRATED PEST MANAGEMENT

Hardware Pricing Detail

For Each District (3) Requirement

Category B

<u>Qty.</u>	<u>Model No.</u>	<u>Description</u>	<u>Purchase Price</u>		<u>Maintenance</u> ⁽¹⁾	
			<u>Unit</u>	<u>Ext.</u>	<u>Unit</u>	<u>Ext.</u>
6	T16/6520	CRT terminal, 2000 characters per screen	\$2,950 ⁽²⁾	\$17,700	\$20	\$120
6	T16/5508	Serial Character Printer, 200 cps	4,500	27,000	50	300
				<u>\$44,700</u>		<u>\$420</u>

(1) Installation location may necessitate an additional charge

(2) A quantity discount is available for 25 or more terminals

Options

OPTIONAL EQUIPMENT

The following items are listed under "Optional" in the RFQ. Their cost or other disposition is indicated. Also see "Responses to Questions."

<u>Item Requested</u>	<u>Tandem Response</u>	<u>Cost</u>
<u>Central Site</u>		
800/1600 Tape	One comes standard on each base system. Additional 800/1600 bpi, 45 ips Tape Drive (T16/5103).	\$8,000
2 ports and consoles	Ports available on system configured. Additional terminals (T16/6520).	2,950 ea.
<u>District Sites</u>		
Printer, 180 cps	Printer, 200 cps	4,500 ea.
FORTRAN	Only one license fee (see Central Site)	N/C
BASIC	BASIC not available at this time. Tandem recommends ENFORM.	N/C
PASCAL	Tandem recommends T/TAL.	N/C

Questions
& Answers

RESPONSE TO SPECIFIC QUESTIONS

- ABC-1. A delivery time of from 60 days after receipt of your Purchase Order is normally possible.
- ABC-2. Tandem's basic maintenance policy contractually provides for a maximum response time of 8 shift hours. Actual experience has normally provided response well under the allowable time frame. Of course, Tandem's NonStopTM hardware and software allow the system to stay up and continue with critical processing during such periods.
- In conjunction with the NonStopTM capability, Tandem has demonstrated a very high MTBF on the individual components utilized. Additionally, all systems are pre-staged and system tested prior to delivery to your site. You are invited to review this process at Tandem's Cupertino facility.
- Expanded periods and types of coverage are also offered at slightly higher costs.
- Field service offices are currently maintained in San Mateo and Los Angeles. A policy of maintenance and repair is normally sufficient to remedy hardware problems. Field replacement will be considered if the problem warrants it.
- ABC-3. The product warranty is 90 days for parts and labor.
- ABC-4. Tandem Computers now enjoys over 200 customers with 1,000 processors installed. The following are five of those customers located in the Bay Area and the contact at each. Each employs a tailored hardware and software configuration

but are generally applicable to the requirements of the RFQ. Further references can be provided upon request.

<u>Customer/Contact</u>	<u>Phone</u>
. University of California Mr. Bruce Benedict Associate Director	(415) 642-0494
. Crocker National Bank Mr. Al Schusterman Mgr., Mini Computer Ops.	(415) 477-2278
. Computer Systems Design Mr. Paul Kruger President	(415) 445-8600
. Patient Referral Services Mr. Terry Gilbert Vice President	(415) 946-3899
. Quantitative Medical Systems Dr. John Sargent President	(415) 654-9200

ABC-5. Tandem Computers offers and supports all the hardware and software bid in this Proposal.

A-1. Using the current configuration 16-disc controllers, each controlling eight 300MB disc drives, could be configured in the system. This would allow approximately 38 billion bytes of disc storage of possible storage.

A-2. The magnetic tape, included in each configuration, is one method of moving data and object files using a removable storage unit.

However, we are proposing a three-node network of Tandem Computers, and data or object file in any node can be accessed by any other node. This feature of the EXPAND Network eliminates the need to mail data between nodes. To copy a file from one node to another would be done as follows:

```
FUP COPY \BERK.FILE1,\RVR.FILE2
```

A-3. Default baud rates are configured in the Sysgen. The baud rate for asynchronous ports can range from 110 baud to 19,200 baud, and each port can have a separate baud rate. The synchronous controller has four lines and the sum of the baud rates cannot exceed 160,000 baud.

The baud rate for any port can be changed by calling a system function without re-Sysgening the Operating System, thus providing the University a great deal of future flexibility.

A-4. A Site Planning Guide is enclosed for planning site configurations. Tandem Computers' Customer Engineers are available to help the University in planning all computer sites as they occur.

All programs will be able to run in any CPU in the EXPAND network. All nodes of the EXPAND network operate the same Operating System. The use of EXPAND makes it very easy to maintain programs. All nodes have the same Operating System; all nodes use the same compilers. The application monitor, MUMPSY, can control all the resources in all the nodes.

- A-5. Software maintenance as it pertains to "bugs" is addressed as they occur. The Tandem software to date enjoys an excellent reputation for "settled" software, primarily due to the pre-release testing and control.
- Currently, software updates and new releases are provided by Tandem Computers three times each year. A site update tape is provided for each Tandem customer with the software the customer has purchased. Tandem also provides a program called INSTALL that reads the site update tape, writes the new software to disc, and creates a new Operating System. The INSTALL procedure takes less than three hours and the INSTALL for the district sites could be run from the central computer system.
- A-6. Any terminal on a Tandem system can be an operator console. One console is provided on each basic configuration. System security, of course, provides control as desired.
- A-7. Each node of the Tandem system can be expanded to 16 CPUs. Each processor can be expanded to 2M Bytes of memory. The EXPAND network can contain 255 nodes with a maximum of 4,080 processors. Additional CPUs, memory, controllers and I/O devices can be added to the system without changing any application design or recompiling any programs. This is of tremendous importance as the University expands its service to its end users.
- A-12. The modularity of the system proposed is unsurpassed. In addition, the power of each node is available to the others if desired.
- A-8. Tandem Computers will supply and maintain all software proposed in this bid.
- A-9. All programs will be able to run in any CPU in the EXPAND network. All nodes of the EXPAND network contain the same Operating System. The use of EXPAND makes it very easy to maintain programs. All nodes have the same Operating Systems; all nodes use the same compilers. The application monitor, PATHWAY, can control all the resources in all the nodes.

A-10. As stated in Question A-9., any program in a Tandem EXPAND network system can run in any CPU in the network. No changes would be required on any programs no matter which node they were run on. Interprocess communication is an integral part of the GUARDIAN Operating System. When a process wants to send information to another process in the EXPAND network, it is done by simple write and read statements.

A-11. All "future capability" items can be supplied today. Examples of specific items' costs are:

<u>Model No.</u>	<u>Description</u>	<u>Cost</u>
T16/2412	Memory Module, 384KB MOS	\$12,800
T16/1412-1	Processor with 384KB MOS	28,700
T16/4104	Disc Drive, 300MB	26,500
T16/5301	Card Reader, 600 cpm	4,800

A review of the proposed configuration will show excess capabilities in many areas where the IPM program expects to increase its requirements. For example, the present port capacity of proposed systems will accommodate nearly all suggested expansion.

Consequently, expansion hardware cost may be significantly reduced.

A-12. Documentation is enclosed for software products being bid. Further detail is available upon request.

A-13. Memory and processor configurations are based on past experiences of our senior analysts in designing on-line transaction systems. The configured system will run the GUARDIAN/EXPAND Operating System, PATHWAY, several Editors, and a few Compiles concurrently. A system sizer package is available to determine precise system resource needs as the application system becomes better defined.

The sizer package utilizes the following data:

1. Transaction size
2. Transaction frequency
3. Number of program modules
4. Disc I/Os per transaction
5. Data Base size and design
6. Computational requirements per program module

The future capabilities can be confirmed as the above data is gained. Experience suggests that some additional memory will be required in the remote district systems as the concurrent users increase significantly. The maximum memory capacity of each processor is currently 2 megabytes. Thus, each system as now configured with 2 processors has a capacity of 4 megabytes.

Tandem's modular hardware and software architecture allows for minimal effort and disruption as outlined elsewhere.

A-14.

All nodes of the proposed system contain multiple CPUs and all controllers are dual-ported. Therefore, any CPU can be disabled from the system, and diagnostics can be run without interrupting normal operations. Tandem has U. L. approval to fix components of the system while the rest of the system remains operational. Any components can be powered down, removed, repaired, and replaced without interrupting normal operations.

A-15.

Tandem does not supply any standard statistical packages. However, ANSI '77 FORTRAN is supplied and supported by Tandem. FORTRAN is a powerful numeric language in which statistical functions are easily written. Any statistical packages written in standard FORTRAN could be implemented on Tandem.

Two software products from Tandem that may be of interest to the University are:

- . MUMPS language interpreter - \$7,000
(see description under Programming Languages)
- . Users' Library - supported by Tandem Users' Group - \$40

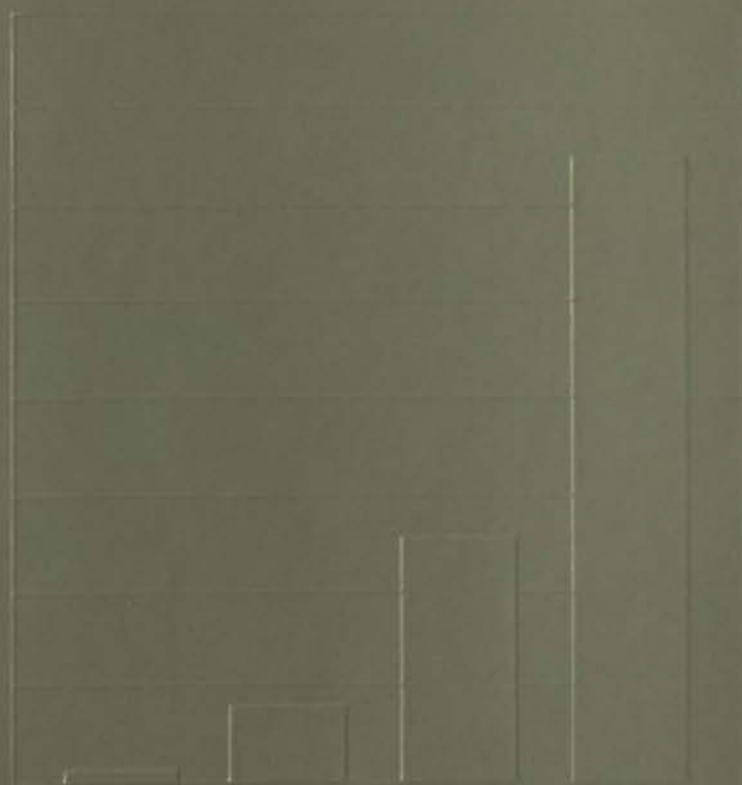
B-1. The terminals bid will run at baud rates from 110 to 19.2KB, and the rates are switch selectable.

B-2. The printer runs at a maximum of 9600 baud.

About
Tandem

TANDEM

annual report 1979



business review

Tandem Computers Incorporated was founded in 1974 to design, develop, manufacture, market and support a unique computer system which meets the critical needs of the on-line transaction processing marketplace. Called the Tandem NonStop System, its innovative architecture virtually eliminates the risk of system failures and protects the customer's data base from damage caused by electronic hardware malfunctions. It is also the only computer system that can be expanded modularly from a mid-size to a large-scale system—or expanded into a distributed data processing network of up to 255 geographically dispersed systems—without hardware or software conversions. Today, Tandem has manufacturing operations in two locations in the United States and one in West Germany, and supports customers' systems throughout North America and Europe from 41 offices.

About this report

As Tandem and the computer industry enter a new decade, the on-line transaction processing marketplace is emerging as a fast growing segment of a rapidly expanding industry. This fact—coupled with Tandem's unique position of leadership in this marketplace—gives rise to many questions. About the industry. And about Tandem, its products and its strategies.

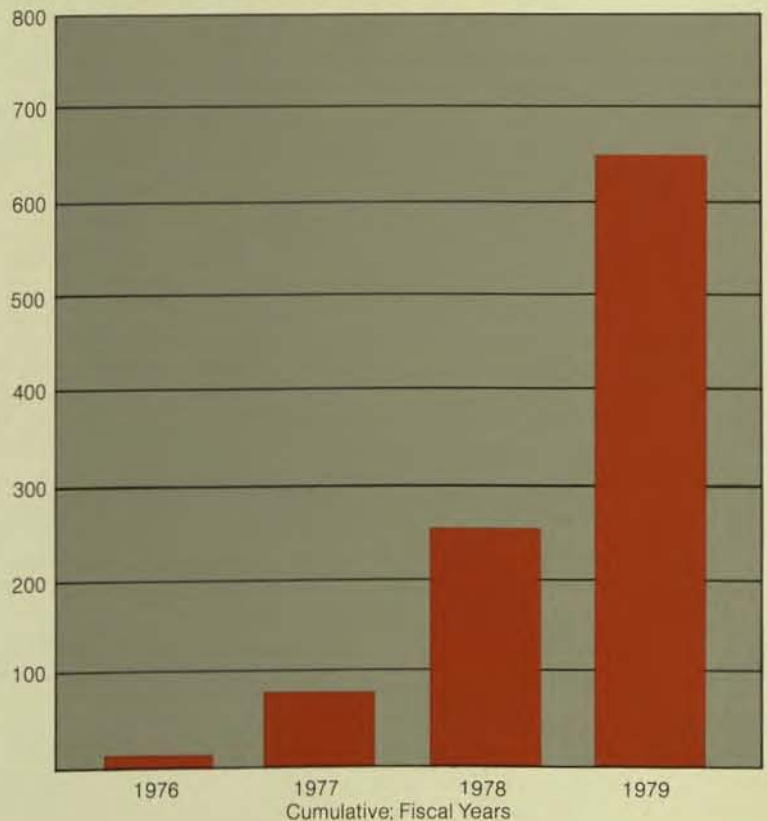
It is with those thoughts in mind that the forum for this year's Tandem annual report is constructed on questions from investment analysts.

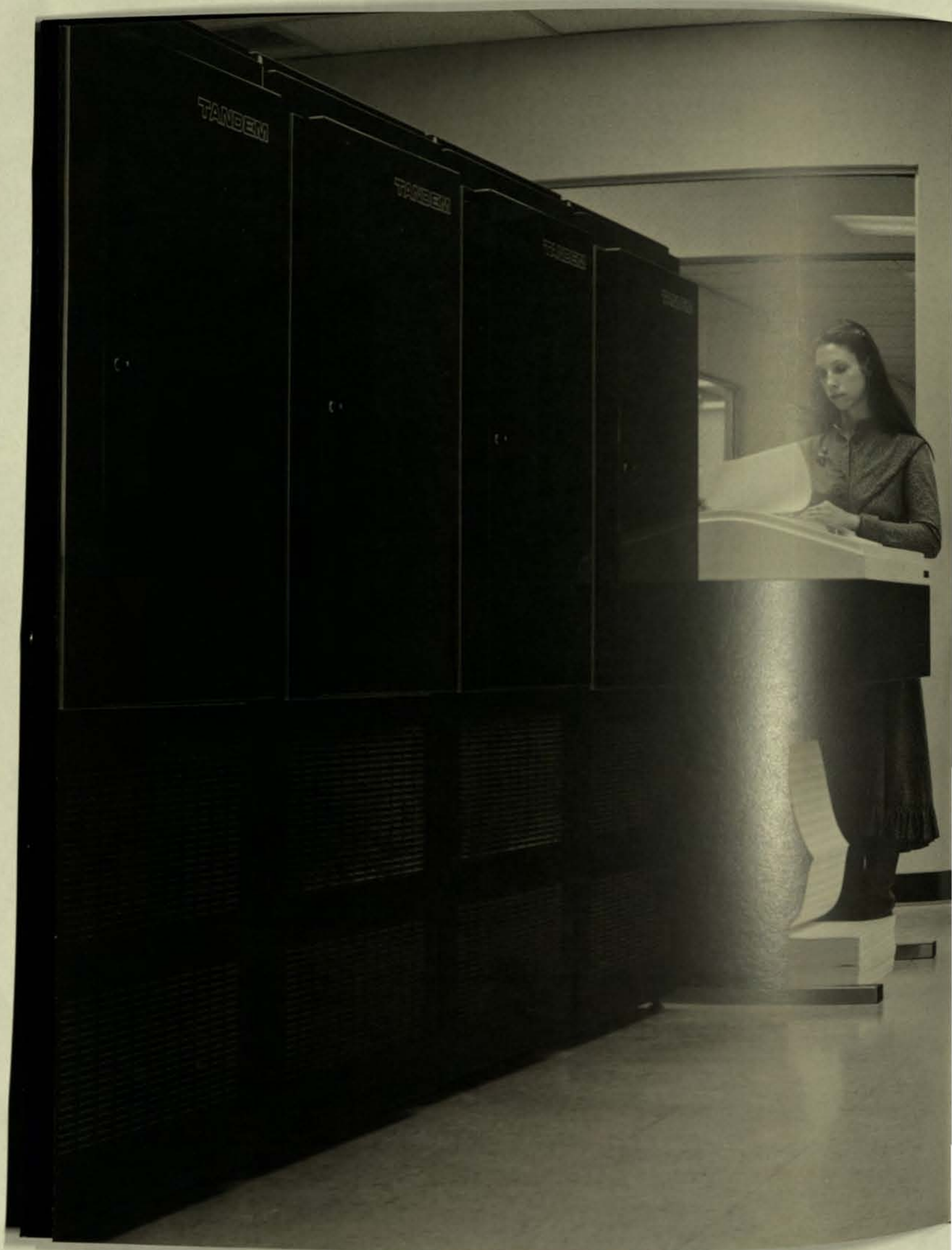
Fortunately, the computer industry—by virtue of its growing importance and the vast pool of public funds invested in the industry—has attracted an unusually large body of highly qualified, fulltime investment research analysts.

Not all analysts that follow the industry are represented in this report; to do so would be impractical because of their numbers. For the same reason, space limitations prohibit the answering of all questions offered. To preserve objectivity and best serve the need to communicate, questions about Tandem were solicited from researchers who do not write reports on the company for their firms' clients as well as those who do. The company is grateful for the time, effort and interest of the investment community in aiding this project.

The reader will also find in this report a number of thumbnail case studies of Tandem customers. These are intended to serve as a sampling of the ways Tandem systems are used, and to act as a primer for understanding the broad span of industries and businesses that have committed to on-line processing.

NUMBER OF PROCESSORS INSTALLED





Tandem alone enters the Eighties with a computer system strategically designed to meet the critical needs and economic requirements of on-line transaction processing, whether at a single site or within a distributed data processing network.

At the core of the emerging age of automation is a segment of the computer marketplace known as on-line transaction processing. It is here that computers are most visible—to the businesses that operate in an on-line environment, and to the customers of those businesses. Businesses like banks with on-line teller stations and automated teller machines. Travel reservation systems. Merchants that use on-line services to provide instant authorization for credit card purchases. Manufacturers and distributors that can immediately tell you the precise status of your order. And scores of other businesses, hundreds of other uses where the need to instantaneously and continuously access and update information is becoming vital.

In this on-line marketplace, the computer also finds its most demanding environment. As more and more businesses come to rely upon on-line transaction processing for better management control, greater productivity and improved customer service, the need for a computer that runs without interruption is essential. In these businesses, when the computer stops—or when a computer malfunction damages or destroys the data base—the business stops.

The Tandem NonStop System is the first general purpose, commercial computer system designed specifically to fulfill the critical needs of on-line transaction processing. The innovative, fault-tolerant Tandem architecture virtually eliminates the risk of system failures and protects the customers' data bases from damage caused by electronic malfunctions. The system is also the only one on the market that can be expanded modularly—without any programming changes and even while the system is running—from a two-processor, mid-sized system up to a 16-processor, large-scale system, creating a continuous range of models priced from approximately \$150,000 to over \$3,000,000.

And, only Tandem systems can be geographically dispersed in a distributed data processing network without modifications to the hardware and without reprogramming—extending the benefits of uninterrupted operations, data integrity and modular expansion to large, high-volume networks.

Tandem's many software products—all of which make the Tandem system easier to use and more productive—enable users to establish distributed data processing networks with much greater ease, speed and economy than ever before known in the industry.

With the EXPAND network operating system, users can easily build a distributed data processing network of up to 255 geographically dispersed Tandem systems without replacing hardware or changing applications software.

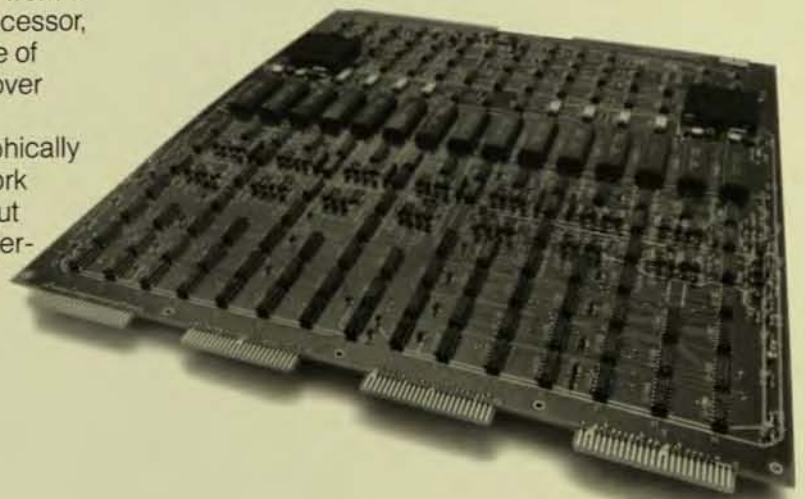
ENFORM is a powerful software tool that makes it easier for the user's programming and non-programming personnel to query multiple files in a data base—in a spoken language-like manner—and to write reports quickly in the user's specified format. The benefits of ENFORM are compounded when used with EXPAND in a distributed data processing network: Data located anywhere in the network can be accessed from any terminal if the user has the required security clearance.

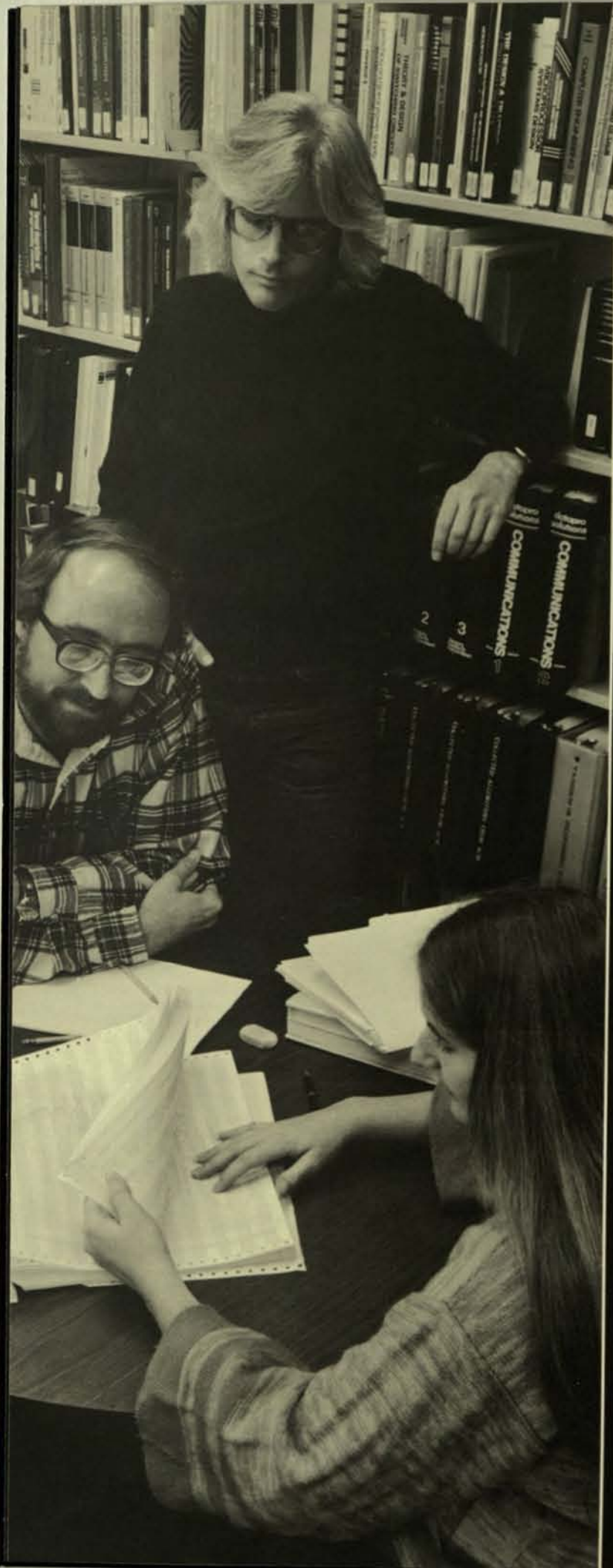
PATHWAY, announced in 1979, will further reduce user programming costs by making it easier and faster to develop on-line applications. PATHWAY handles many of the complex terminal and control functions that are an inherent part of any on-line transaction processing application. With PATHWAY, the user is free to concentrate on the applications themselves.

Also introduced in 1979 was the Tandem 6520, a new display terminal that provides high reliability and increased productivity in the on-line transaction processing environment.

As networks of dispersed Tandem systems grow, so will the importance of system serviceability. In addition to sophisticated self-diagnostic capability in hardware, DIAGLINK enables Tandem personnel to remotely diagnose a user's system via telephone link, anywhere in the world.

The Tandem NonStop System today stands unrivaled in its ability to meet the world's growing on-line transaction processing and distributed data processing needs.





◀ At Tandem customer training centers throughout North America and Europe, over two times as many representatives of customer organizations were enrolled during 1979 as in the preceding three years combined.

What is Tandem's niche in the market, that is, what is the company's expertise—should you be regarded as a hardware manufacturer or a predominantly software company?

—Howard W. Geiger, Jr., Securities Research Division, Merrill Lynch Pierce Fenner & Smith, Inc.

Tandem's niche is sole dedication to on-line transaction processing, and our successes to date are a result of our combined expertise in hardware and software directed at satisfying previously unfulfilled needs of the marketplace. In our focus on the on-line transaction user, we bring more to the marketplace than merely hardware and software. Like a mainframe manufacturer, we support our customer base with a large field service organization. For each salesman we have two systems analysts and two field service engineers. During fiscal 1979, Tandem's field service organization expanded 130%.

Tandem's original contribution was a state-of-the-art technology and architecture for maximizing multi-processing. The need for multi-processors existed before Tandem, but the only way it was being met was by the tremendously expensive, inefficient method of tying together two or more conventional computers.

Subsequent to the Tandem product introduction, we have continued to make the system more useful for customers by enhancing the hardware and by developing additional software tools that make the system easier to use and more productive. The effectiveness of the software—and Tandem does have a reputation for outstanding software—is only possible because of the unique architecture of the hardware.

Is Tandem's typical customer more interested in NonStop capabilities, the high transaction rates or the networking possibilities?

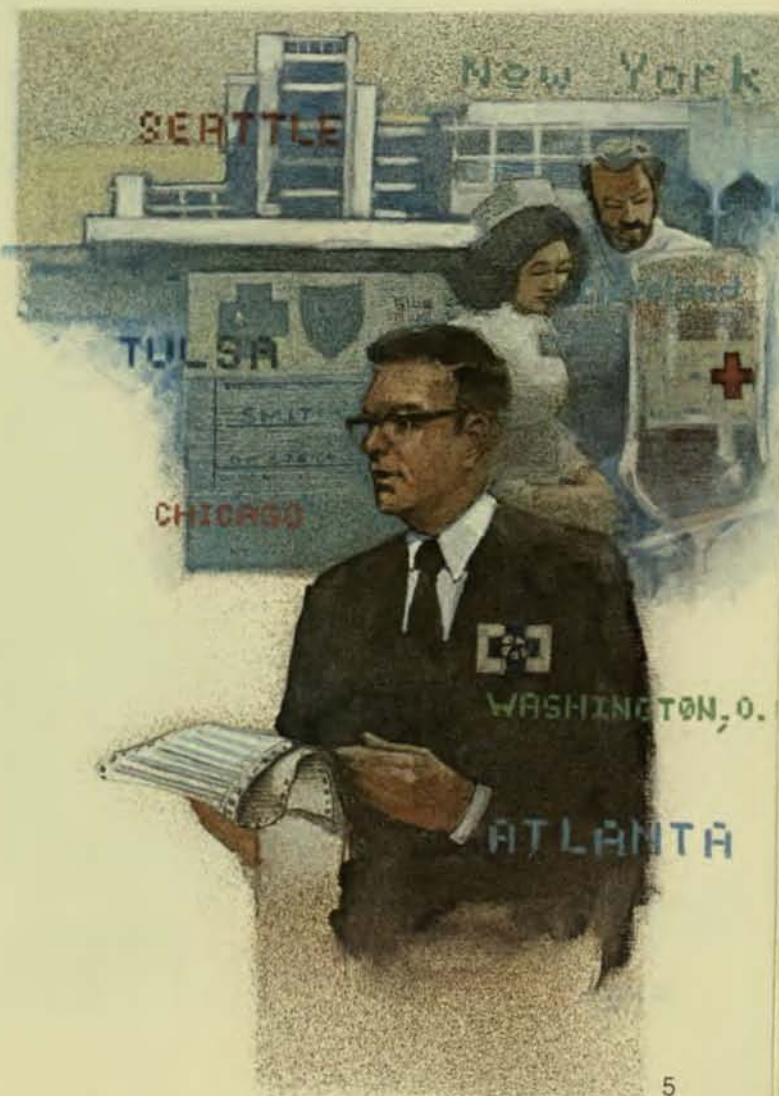
—James W. Reynolds, Vice President, Bateman Eichler, Hill Richards

Tandem's customers are interested in the productivity advantages of the on-line environment, and recognize that NonStop operation is essential. Before Tandem, there were a number of risks associated with committing vital aspects of a business to on-line processing. Tandem has alleviated those risks.

Our customers are attracted by the Tandem advantages in combination, but with varying degrees of emphasis on any given feature. For some, continuous system availability is critical. To others, it's data integrity. For most—those users with geographically dispersed systems and distributed data bases—networking is obviously important. For virtually all, modular expandability holds great attraction. And, it is universally appealing to Tandem

One hundred forty-five hospitals in two states depend on the Missouri-Illinois Regional Blood Program for some 185,000 pints of human blood annually, and the St. Louis-based Red Cross chapter now depends on its Tandem system to match supply with demand. The organization uses the Tandem system, acquired in early 1978, for donor record keeping, to aid in recruiting donors and, once the donations have been made, to screen for blood quality and verify blood type. Most importantly, the on-line inventory system tracks the 4,000 pints of blood kept on hand, directs inter-hospital transfers, and monitors the supply's limited shelf life. Additional applications including administrative and accounting systems are scheduled to come on-line in the near future.

When PLAN-NET, the new data communications network of the Blue Cross Assn. and Blue Shield Assn., becomes operational in early 1980, all 100 regional offices in the U.S., Canada and Puerto Rico will be linked together by some 200 terminals interacting with eight geographically dispersed Tandem systems. In all, the network will come on-line with 22 Tandem processors spread over Chicago, New York, Atlanta, Tulsa, Seattle, Cleveland and Washington, D.C. as a custom turnkey system developed by International Micor Systems, Inc., a wholly-owned subsidiary of Ramada Inns. PLAN-NET replaces a slower, over-taxed system that was handling 50,000,000 characters of information daily. One of the heaviest loads on the new, high-volume network will be daily data collection for all Medicare claims throughout the U.S. Other major functions include eligibility determination for transient claimants, and balancing accounts for the Blue Cross and Blue Shield Inter-Plan Bank. PLAN-NET will be on the air 24 hours every day.



◀ Far left photo: Tandem software professionals made major contributions to Tandem system efficiency, programming ease and productivity during 1979.

At the Canadian operations of Pilkington Glass Industries, Ltd., a subsidiary of the world's largest producer of flat glass, Tandem is the main computer. Acquired in mid-1979, Pilkington's Tandem system will take over all corporate data processing at the Toronto head office and main factory as additional processors are added. One of the company's principal reasons for converting to Tandem was the "smooth, painless upgrade path to a very high-powered system at relatively modest cost" essential to accommodating the workload. The Tandem system will be "intimately linked" with manufacturing in a company where, by necessity of the glass-making process, operations must run continuously around-the-clock, year after year. Currently, the system supports an on-line network of 35 terminals at the facility, which produces some 225,000,000 square feet of glass yearly for international markets.

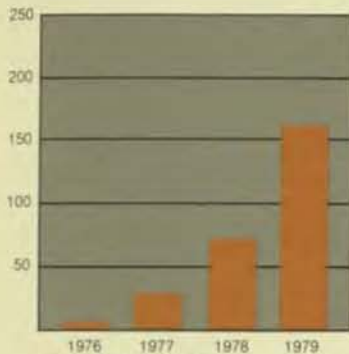
Girmes AG, one of Europe's principal manufacturers of textiles and carpets with annual revenues in excess of DM 600 million, acquired its Tandem system in the autumn of 1977 and has never experienced a hardware or operating system failure. The initial two-processor Tandem went on-line in June 1978 with completion of software development for an order processing system that now ties-in the company's three factories with some 50 terminals. The system size was doubled in the summer of 1979 to handle increased workloads. Girmes is now developing applications software for a new manufacturing data capture system that is scheduled to come on-line during 1980. The company employs 3,500 people in West Germany.

Jövan, the U.S. producer of fragrance products, went on-line with their new Tandem-powered warehouse control system in 1979 and quickly experience a near-tripling of productivity. By converting to the automated system to fill up to 4,000 orders daily for the company's 456 different products, Jövan has improved order delivery by more than a week with dramatically greater accuracy. Further cost savings emanate from automated freight consolidation of the typically small packages. The system receives all orders, allocates inventory, selects orders; generates the transportation plan, weighs each order; provides freight rates; and prints shipping labels, bills of lading, packing slips, shipping manifests and order confirmations.

Jövan selected Tandem because of the critical importance of continual operations and the ability to add computing power without interrupting shipping of products.

Unlike most Tandem installations, the system at the Ritepoint Pen Division of Penn Corp. is used primarily as a batch processing system. Ritepoint converted to Tandem in mid-1978 to achieve greater reliability, easier programming, lower costs and higher performance. Although not dedicated to on-line transaction processing, the company has found it enjoys the benefits of faster on-line data entry and quicker error detection. The bulk of the workload is tracking up to 20,000 orders at the St. Louis plant at any given time. Additionally, the system handles all accounting, production planning and other general data processing. The division, one of the largest manufacturers of advertising specialties, selected Tandem after a six-month competitive evaluation. Penn Corp. has annual sales of approximately \$30,000,000.





Number of Customers
(Cumulative, fiscal years)

customers that they enjoy all of these features without cost premium while also enjoying high throughput rates at low cost.

There are Tandem systems—and some very large ones—that are in environments where continuous operation is not critical to the function of the business, but where a hardware mal-

function that damages or destroys the data base represents a disaster. If you are a bank doing a multimillion-dollar electronic funds transfer when your computer malfunctions and sends the money to the wrong place and you can't get it back, the Tandem data integrity feature that prevents electronic interference with the data base is at least as important to you as the system's fault-tolerant capabilities.

The modular expansion capability is of great interest because most computer centers will need additional capacity within the foreseeable future. If you are in any business that is adding new applications or is growing, you know from past experience that it's going to cost you a lot of time, money, grief and disruption to your business to upgrade to a bigger model. The ability to easily and inexpensively add more modular processors to your Tandem system without changing a line of programming—without even shutting down the working processors—has appeal equal to the NonStop and data integrity advantages.

Modularity has its initial appeal when a customer is

When the U.S. Treasury Department's new Tandem system comes on-line in 1981, it will make electronic funds transfers of some \$100 billion annually between the Treasury and more than a hundred different government programs. In addition, the system will record and monitor the sale of U.S. government gold and service grant programs such as letters of credit. The Tandem system will be dedicated to the Treasury Financial Communications System—previously run on a shared computer—to improve security and reliability, and to provide for easy add-on of computer power.

converting to an on-line application for the first time. Instead of buying in advance the capacity that will be needed when the system goes into production, the customer can buy a minimum-size system to use in developing the application software, and then later add the necessary power painlessly. Once into production, the system can grow and grow and grow—and our customers know that, and know that they've bought a clear path to a mainframe-size system, and know that they will never again have to deal with the huge costs of new programming, new hardware and retraining people.

They also understand that the system's design inherently favors networking and that, with Tandem EXPAND software, the realization of a large, productive distributed data processing network is infinitely easier and faster, and that they can save millions of dollars over a network of conventional systems in software and communications costs.

Several studies and practical experience have shown that multi-processor architectures are inherently inefficient. Why shouldn't we assume that the Tandem NonStop approach will appeal only to a limited segment of the total marketplace, thereby placing a lid on long-term growth potential?

—L. Duane Kirkpatrick, Vice President, Research,
Dean Witter Reynolds, Inc.

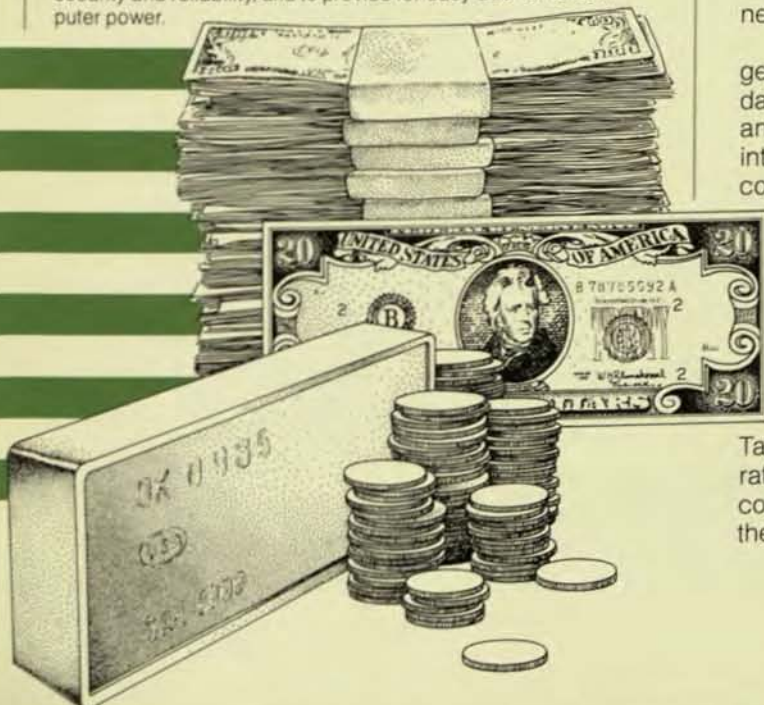
You will no doubt find on re-examination that those studies did not include Tandem. It is true that conventional computers, designed to stand alone, lose efficiency in a multi-processor configuration. It is also true, however, that the Tandem system was designed from scratch to be optimized for multi-processing. Tandem represents an entirely new architecture which is inherently efficient.

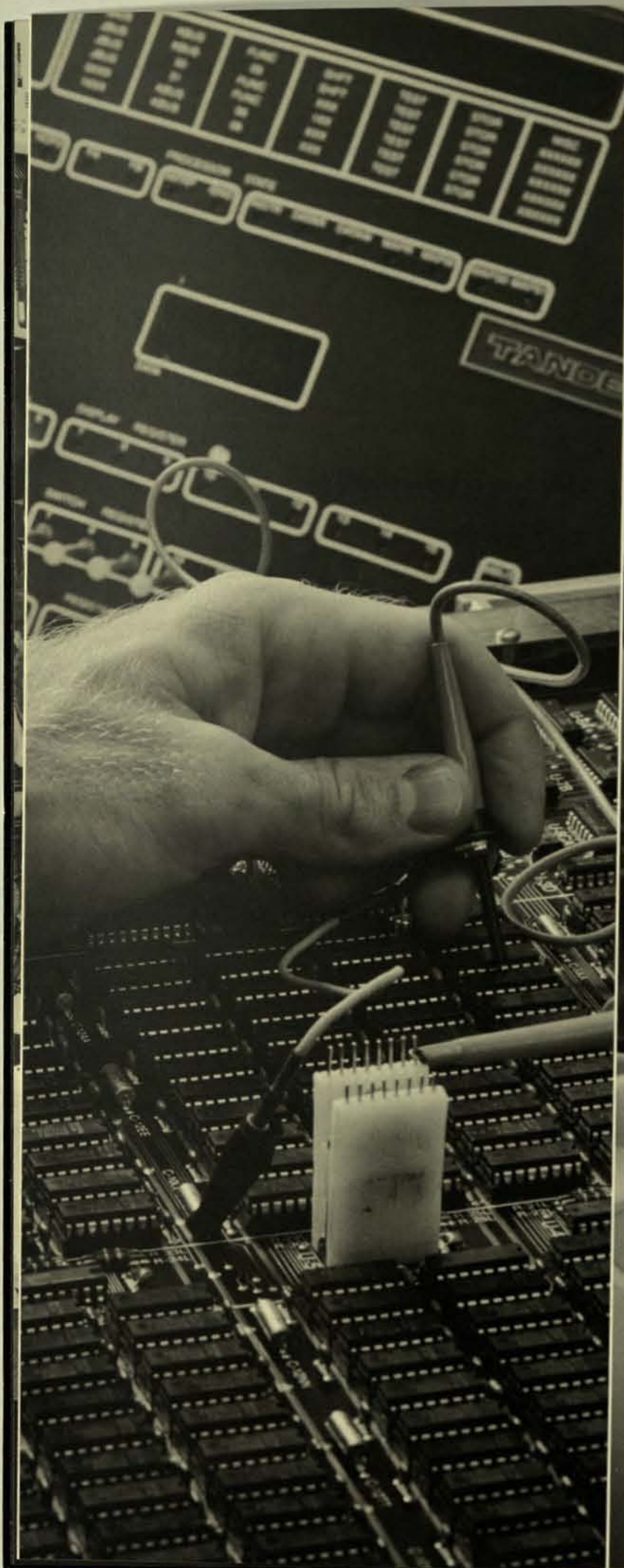
The Tandem system bears no relationship to the concept of merely coupling together two or more stand-alone computers, aside from the fact that both express a need for computers that do not fail.

Unlike the typical multi-processors of preceding generations, the Tandem system is not functionally redundant; all elements of the system are handling the workload, and no one processor is merely waiting for another to fail. This intentional efficiency of Tandem systems is proven out in competitive benchmarking—nearly always against uncoupled single processors—and by the resultant transaction rates, price-performance comparisons and by the hundreds of Tandem systems now in service.

In view of the inherent efficiency, and considering that customers get fault-tolerance, modular growth and data integrity capabilities without additional cost, Tandem perceives the system's appeal as universal within the broad, rapidly growing on-line transaction marketplace.

Tandem's participation in this marketplace is expanding at a rate consistent with the increased awareness in the business community of Tandem and of the productivity advantages of the on-line environment.





◀DIAGLINK enables Tandem personnel anywhere to remotely diagnose users' systems via telephone link worldwide.

Most Tandem customers have widespread operations and will establish networks of geographically dispersed systems. For them, Tandem's EXPAND software represents a major breakthrough that will save millions of dollars in communications and programming costs, and accelerate completion of distributed data processing networks.

Describe the strategic importance of EXPAND for Tandem's movement into distributed data processing and, relatedly, data communications. When will EXPAND begin to have a meaningful impact on income?

—Michael P. DeSantis, Partner,
Robertson, Colman, Stephens & Woodman

We are extremely enthusiastic about EXPAND. Our bringing a distributed data processing capability to the marketplace that eliminates reprogramming and hardware changes does much more than merely add another exclusive feature to Tandem systems. We view EXPAND as being as important to

Tandem and the on-line processing user as the original Tandem product offering.

Strategically, EXPAND broadens our market to the extent that it makes a significant contribution to our confidence in our ability to sustain a high rate of growth. A large portion of our customers are developing distributed data processing networks to better manage and control their geographically dispersed operations.

For example, one of our EXPAND customers—a major international bank—is implementing an integrated worldwide bank management system. Tandem computers are being installed at the bank's operating centers throughout the world, some of which are extremely remote. With EXPAND, the network of dispersed Tandem systems will handle all communications and processing functions between systems. Data will be processed and stored at the geographic location where it originates, or, for better response time, at a location where it is most used, and will be routinely accessed by any location as if the information was resident at that location.

To support the growing base of EXPAND users, Tandem had 41 marketing, field service and training centers throughout North America and Europe at the close of fiscal 1979. Additional centers will be opened during 1980.

The networking capability of Tandem systems was strategically planned at the product's original development stage. It is inherent to the system architecture. The same



First National Bank of Chicago, among the top ten U.S. banks with assets over \$25 billion, employs five separate Tandem systems in three different areas of the Bank's operations. In Chicago, one Tandem system handles over 200,000 transactions monthly through 42 automated tellers; this rate will double when 200 regular teller stations come on-line in early 1980. Three more Tandem systems—in London, Paris and Chicago—are being used to develop a set of international banking applications which will be deployed with Tandem's EXPAND software in an international network of the bank's medium and large overseas installations. And, back in Chicago, a fifth Tandem system is being used to develop new programs for future applications elsewhere within the bank.

Before Chase Manhattan Bank went on-line with its account locator and verification system, the bank's controller's office was manually responding to 1,500 telephone inquiries daily. To service these inquiries, the bank employed as many as fifty people to manually access, update and file 1 1/2 million cards on 750,000 customers. With the new Tandem system, the bank freed half of the employees for more productive work, and cost savings paid for the system within 18 months. Whereas it previously required a minute and a half to service each inquiry, it is now done in three or four seconds from any of 30 terminals located throughout the bank.

◀ Far-left photo:
Tandem's own design memory testing device is used in the manufacturing process and by the field service organization. Final system test—as with field service—is facilitated by integral self-diagnostics and the NonStop System feature.

engineering concepts that make Tandem's fault-tolerance and modular expandability work also facilitate the networking capability and make possible the precedent-setting proficiency of EXPAND software. This distributed data processing capability is a logical extension of Tandem system advantages in concert with the needs of the market.

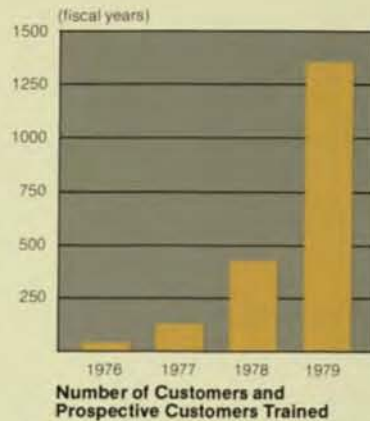
EXPAND software made immediate contribution to revenues upon availability and first deliveries in mid-year of fiscal 1979. What is important about EXPAND from the standpoint of revenues, however, is that EXPAND profoundly broadens the appeal of and demand for Tandem systems. All EXPAND buyers will be multiple system buyers over time.

It appears that one of your most significant products to date is EXPAND, your networking software. What percentage of your customer base has needed this capability? What percentage of your customer base five years hence will need EXPAND? And, how much of an edge does this software product give you over your potential competition?

—Steven P. Novak,
Assistant Vice President,
Harris Trust and Savings Bank

The majority of Tandem users are developing or planning distributed data processing networks. The reason for this is that most of Tandem's customers are large organizations with geographically dispersed operations. The early reception of EXPAND is indicative of the magnitude of the demand for the capability. First EXPAND deliveries were made near the end of our fiscal first half. By the end of the

Of all the inventory control problems known to modern business, those of a large railroad are among the most complex. To Illinois Central Gulf Railroad—one of the largest in the U.S. with nearly 9,000 miles of track—one object of inventory control is keeping some 50,000 freight cars continuously productive. The task involves the complexities of matching car orders with availability; assigning cars by commodity/class; generating "switch lists" to locate and move up to 180 cars from varying positions on up to 60 tracks in a yard to make up a train; blocking the cars in destination drop-off order; waybilling to conform to regulations and to assure billing to shippers; and then turning around and repeating the process at the other end of the line. ICG inaugurated its new waybilling and yard management system during 1979 on a Tandem system at its Baton Rouge, Louisiana, yard. Other ICG yards will come on-line with additional Tandem systems in 1980 and beyond in a program to upgrade car inventory control in all of the railroad's major yards.



fiscal year—that is, within just six months—15% of our customers had already ordered the package.

Inasmuch as we do not see a shift in the nature of our customer base in the future, it is likely to remain the case that most of our new customers will be distributed data processing candidates.

EXPAND further sharpens Tandem's competitive edge in that the user's total costs for developing a distributed data processing network—whether large or small—are dramatically reduced. The Tandem system is far ahead of the industry in its capacity and competitiveness as a distributed data processing system. We have functions, features, performance and cost advantages that are unmatched.

With EXPAND, there need be no host computer, as in other networks, that can fail and jeopardize the data or continued operation of an entire network. Each Tandem processor in a geographically dispersed network sustains its own data integrity and performance integrity. Under EXPAND, any Tandem processor in the network can communicate directly with any other without costly point-to-point communications between all systems. Tandem systems are also certified to communicate on X.25 public or private packet switched networks which can further reduce communications costs. And, in the event of a communications line failure, EXPAND automatically reroutes communications and the network stays on the air.

All of Tandem's product offerings in combination—including EXPAND—arm us with a substantial lead over any competitor who will have to develop hardware and software serially. Regardless of the potential competitor's resources, that process is time consuming.

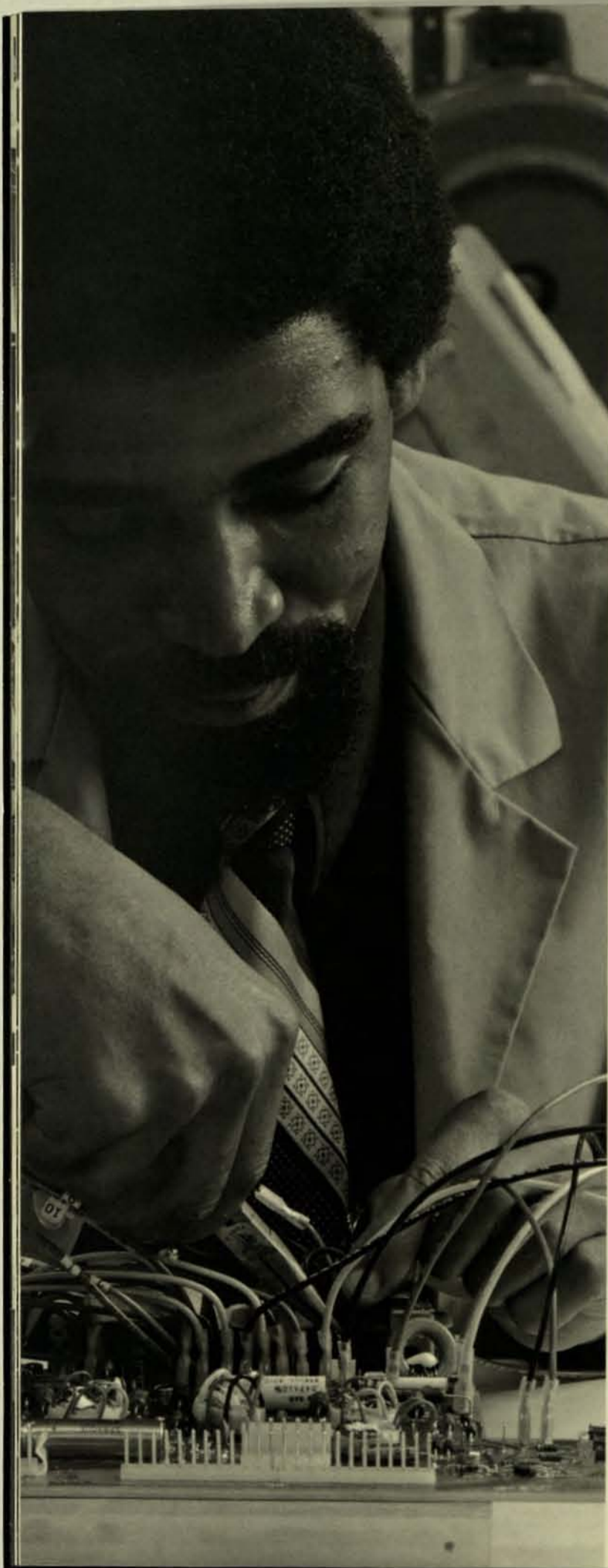




On opposite coasts of the U.S., seven days a week, thousands upon thousands of credit card purchases and personal checks are being instantaneously approved via Tandem systems at Telecredit, Inc., a national leader in check guarantee and credit card processing services. Since mid-1978 at Florida Service Center, a wholly-owned subsidiary of Telecredit, a Tandem system has been on-line providing complete bank credit card services involving some \$15,000,000 in purchases and 650,000 transactions weekly for Master Charge and Visa. During 1979, Telecredit's second Tandem system came on-line, and is currently servicing some 8,000 on-line point-of-sale terminals providing national credit card authorizations and check approvals for many major U.S. banks and over 70,000 merchants.



Two new buildings under construction in California will ► more than double headquarter's square footage when completed in early 1980. At left is James G. Treybig, president and chief executive officer, with Robert C. Marshall, vice president and chief operating officer.



Tandem has a sound base of satisfied customers: During fiscal 1979, half of all Tandem shipments went to previous customers.

In Europe, our main concern is the ability of Tandem to continue to grow. The decision making process in the European business community is slower than in the U.S., especially in accepting a new product that involves both a new technology and a new company. The first thing your European prospects will do is contact some of your customers and ask if they are satisfied, and if they have or are ordering more systems. Are they?

—*Pierre G. Mirabaud, Partner,
Mirabaud & Cie.*

We find the European prospect to be extremely thorough and analytical, especially with regard to cost consciousness. This factor has been a definite advantage to us in Europe because we do very well in the cost-competitive arena. We do not, however, find the decision making process any slower in Europe. Our European prospects contact our customers which, again, is advantageous to us because we have a broad base of satisfied customers as evidenced by our unusually high rate of repeat business—over 50% during fiscal 1979—and by the fact that we have never had a system returned to us.

Virtually all Tandem customers order more capacity within a year or so after their initial purchase for three basic reasons. First, the modular expandability of Tandem systems enables customers to install only the computer power they will need over the short term. With conventional systems, the user must project needs out over several years because of the high costs of reprogramming, retraining and bringing a new system up. Tandem users have none of these costs when upgrading, and can add power on relatively short notice.

Secondly, most new Tandem systems are used for new on-line applications which entail development of applications programming by the customer. Typically, this is done on a minimum-size, two-processor Tandem system, and later the system is enlarged when it goes into production.

Thirdly, the on-line environment is highly dynamic. Our customers are most often in growing businesses that demand regular increases in computer power. Most of our customers are continually developing new on-line applications, and many will be developing distributed data processing networks of geographically dispersed Tandem systems.

◀ Tandem has manufacturing operations in two locations in California and one in West Germany.



Europe's largest department store chain, Karstadt AG of West Germany, has 162 outlets, annual sales of over DM 10 billion, and a growing Tandem system committed to on-line management of the company's DM 560 million furniture business. Karstadt took delivery of its first Tandem system at company headquarters in Essen in mid-1977 to develop application software to service six stores with massive furniture departments. The inventory control system with some 50 terminals enables clerks to immediately verify warehouse stock, write the order, generate shipping papers and invoice the customer. A future software development will enable sales personnel to make entries to allow a customer to customize the furniture ordered. A second Tandem system was installed in late 1978.

Providing management with constant, on-line availability of a myriad of operational reports and comparative analysis on 170 stores in 13 divisions spread over 38 U.S. metropolitan areas is just part of the Tandem role at the May Department Stores Company headquarters. Corporate officers, using the easy, English-like language of Tandem's Enform, can also access the computer through 25 terminals for detailed, product-by-product merchandising statistics and for sophisticated researching of potential new store sites. Although Tandem's fault-tolerant capabilities are not considered by the company to be critical to the current application, the data processing group reasoned, "Given the choice, why shouldn't we buy a computer that keeps running?" May Stores, headquartered in St. Louis, occupy over 33 million square feet nationally.

What percentage of Tandem's processor placements are made to existing customers and what percentage to entirely new customers?

—George R. Balaschak,
Sr. Investment Research Officer,
The First National Bank of Boston

During fiscal 1979, Tandem delivered 389 processors to 118 customers (which, incidentally, is more processors than in all previous years combined.) Of those, over 50% went to existing customers and the remainder went to new customers. Our strategy of encouraging new customers to initially acquire minimum-size systems has benefits for both the customers and Tandem. For users—who are typically using new systems to initially develop software for new applications—it is advantageous because they do not have to pay in advance for the computer capacity they will later need when the application goes into production. For Tandem, it is beneficial because we can spread our production over a wider customer base, thereby building up a reserve of customers who will return to us with additional business. In virtually all cases, we are assured of additional business as long as we continue to serve and support a satisfied customer base.

What do you see as the potential market size for Tandem's products?

—Thomas J. Crotty, Vice President,
Gartner Group, Inc.

The market potential is equivalent to the size of the market for on-line transaction processing systems and networks made up of those systems. At present, we believe it to be a multi-billion dollar market, and growing at a rate in excess of 30% per annum.

We view ourselves as the premier contender in this marketplace inasmuch as (a) we are the only company that is dedicated solely to on-line transaction processing, and (b) we alone produce a system that is designed specifically to fulfill what we consider to be the inherent, essential needs of the on-line environment. These needs include continuous availability; data bases secure from electronic damage; built-in, painless growth potential; ease of programming; ease of operation; low cost per transaction, and systems that quickly and inexpensively become a distributed data processing network.

The ultimate, definitive size of the market—as well as its continued growth rate—is dependent upon the rate at which businesses discover the efficiency, customer service and profitability advantages of the on-line environment.

As a relatively young company, our strategy has been to

concentrate our marketing activities on prospects that have identified as critical their need for Tandem's features. This allows us to build our base faster. In theory, all computer users would opt for, say, a fault-tolerant system over one that fails—especially when it does not cost any more. But, not all users will buy from a five-year-old company, even if the product has immense design advantages. As Tandem and its reputation grow, the number of users who will buy from Tandem also grows.

Discuss the market segments where you are currently active and the potential for those segments that you see yourself participating in over the next 3-5 years.

—Irwin Lieber, Partner,
First Manhattan Company

Tandem systems are currently being used in many industries. We are active in industries that have been quick to regard the full integration of computers into their businesses by means of on-line transaction processing as a logical, competitive step forward. Organizations within those industries are converting and committing vital aspects of their operations to on-line transaction processing to control their businesses better; to improve the management of capital, and to offer better customer service in an increasingly competitive marketplace—all with an end objective of enhancing productivity and profitability. They are converting to the extent that on-line transaction processing has emerged as a major new market. It is quite possible, although perhaps presumptuous, that the advent of Tandem and our on-line-specific technology—hardware and software—will lend further impetus to the rate of conversions.

Much of the concern and burden associated with a company's decision to commit the vital aspects of its business to on-line automation has been obviated by the Tandem hardware and software.

During fiscal 1979, systems were shipped to customers in 25 industries. Banks and manufacturers each accounted for approximately 14% of shipments. Other major economic sectors that purchased Tandem systems included medical, service bureaus, non-bank financial institutions and national governments.

As to the future, we do not see a dramatic shift over the next few years in our customer mix. We do expect to see the market in which we are now strongest to remain strong. And we anticipate increased interest in many other segments where we have no presence or where we have just scratched the surface.

Most analysts expect to see competition for your exclusive NonStop capabilities. What is your strategy for competing against established, name-brand competitors?

—Jay Stevens, Securities Analyst,
Bear Stearns & Co.

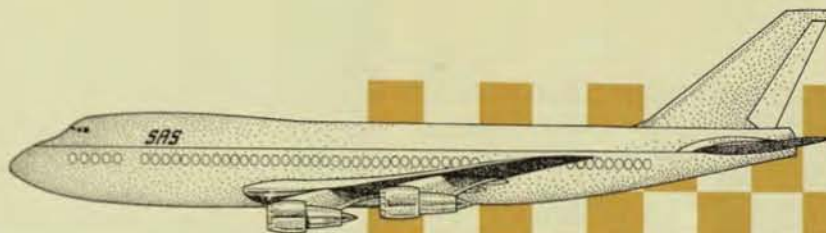
Eventually, all computers will have a much higher standard of reliability. The marketplace will demand it. Continuous availability, however, is just one of the Tandem capabilities. Our product incorporates other exclusive features that are equally innovative and broadly appealing to users. They are, of course, the Tandem features of data integrity, modular expandability and inherent networking ease.

Competition will lend credibility to the Tandem concepts. But, we do not see direct product competition on the immediate horizon. All computer manufacturers talk about high reliability, for instance, because of its obvious appeal to customers. We frequently hear of reliability in terms of 98% and 99% which sounds impressive.

Suppose, however, your business operates its computer



At the 1250-bed University of Alberta Hospital, an initial, two-processor Tandem system is taking over central registry, admission-transfer-discharge records and support of pharmacy and radiology functions. As the system grows during 1980 and beyond, some five processors interacting with over 150 terminals—including terminals at all nursing stations—are planned to monitor patient care services with a capacity for ordering and reporting all tests and procedures. Medical instructors at the institution, which is the leading research and teaching hospital of Alberta, will use the system to recreate past circumstances in evaluating performance of trainee staff. The system will also be used to control material management and equipment maintenance records, and will be expanded to incorporate patient-oriented services at out-patient clinics. The hospital's service domain spans thousands of miles of northern Canada, reaching virtually to the North Pole.



Some 400,000 packaged holiday tours by air to 60 destinations and 600 hotels are booked on-line annually by Vingressor AB, Sweden's largest tour operator and wholly-owned subsidiary of SAS (Scandinavian Airlines System.) Vingressor has 28 bureaus in Sweden, Norway, and England with 190 terminals connected via leased telephone lines to an on-line Tandem system. The rapidly growing tour operator computerized its business in 1972 and converted to Tandem in 1978 to overcome computer failures and to acquire the capability of easy expandability of computer power. "The luxury at no extra cost of Tandem NonStop has become second nature to us. It is hard to imagine how we lived without it, or how others are still living without it." Vingressor's original two-processor system was expanded to six in 1979 to accommodate peak loads.



24 hours a day. If your system fails once a month and is down for eight hours, that translates into 98.8% availability. But, it's also a full shift lost, and a lot of business lost.

Further suppose that you tie that computer into a distributed data processing network with a second, identical 98.8% availability computer. Now, your network is down twice a month; you have a failure, theoretically, every 15 days. Add a third system to the network and you're off the air once every ten days. Build a ten-system network and you're out of business every third day. Users are fully aware of this phenomenon. Once they are convinced in great numbers that the NonStop System technology really does exist, users will demand it in their systems. Ultimately, Tandem will have competition.

No manufacturer, however, can develop such a system quickly. There are basically two hurdles that have to be overcome. First, the development must progress serially. No matter how many people and dollars you put to the task, many aspects of the development cannot be undertaken in parallel. That takes time. Secondly, if you're to build a fault-tolerant system, your software has to have an unusually high level of integrity, and must undergo exhaustive quality assurance testing. Developing such software takes time. Then, having accomplished those things, a would-be competitor will have to catch up with all our other exclusive features, learn all we have learned about the on-line environment during the past five years, and then bring all of its talent to bear on translating that knowledge into meaningful products. That, too, takes time.

Our strategy toward meeting the eventual, direct competition is to continue to be a moving target. Tandem has added many unique capabilities since the original product introduction that are of significant benefit to users and effectively broaden the Tandem appeal. These new capabilities all focus on the same objective: make it easier and economically more favorable to bring an on-line system up and keep it running.

We will continue to broaden the appeal of Tandem

systems and remain a moving target for eventual competition by continually addressing ourselves to solving user problems.

Has the onset of recession and price-performance improvements by competitors made the selling of Tandem systems more difficult?

—Richard A. Goers, Investment Analyst,
Kemper Financial Services, Inc.

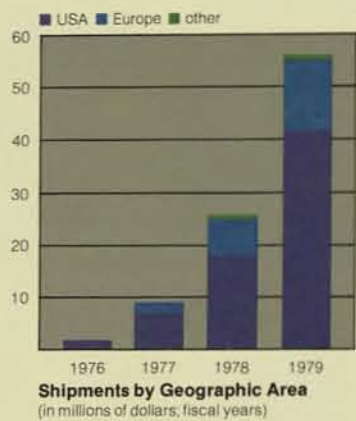
What might be the impact of a recession in view of your market which is mainly new applications that would be cut first if user data processing budgets are reduced?

—Stephen T. McClellan, Vice President,
Salomon Brothers

Despite price-performance improvements by others, Tandem systems are still highly price competitive. We have had major price-performance improvements ourselves in each of the last two years. Difficult as it may be to believe, Tandem buyers enjoy all of the unique Tandem features without cost premium as well as enjoying low costs per transaction—the true evaluation criterion.

At this writing—early in our first quarter of fiscal 1980—we have not felt the effects of the onset of a recession. It is possible that Tandem could benefit from the threat of a recession to the extent that buyers in a recession economy are more cost conscious. Any situation where critical cost analysis is paramount we believe will be favorable to Tandem.

There are opposing views as to what happens to data processing budgets in a recession. One view is that users tend to cancel new projects. In that case, Tandem would be



machines. In either case, our strategies for coping with a recession—should it come, and should it affect Tandem—have been addressed at length, and our contingency plans are in place.

vulnerable because most of our new-customer orders are for new applications. However, the opposing view holds that inasmuch as there is sentiment in a recession for cost reductions, products such as Tandem's could be immune from the severity of a recession by virtue of the fact that they are cost-cutting, productivity

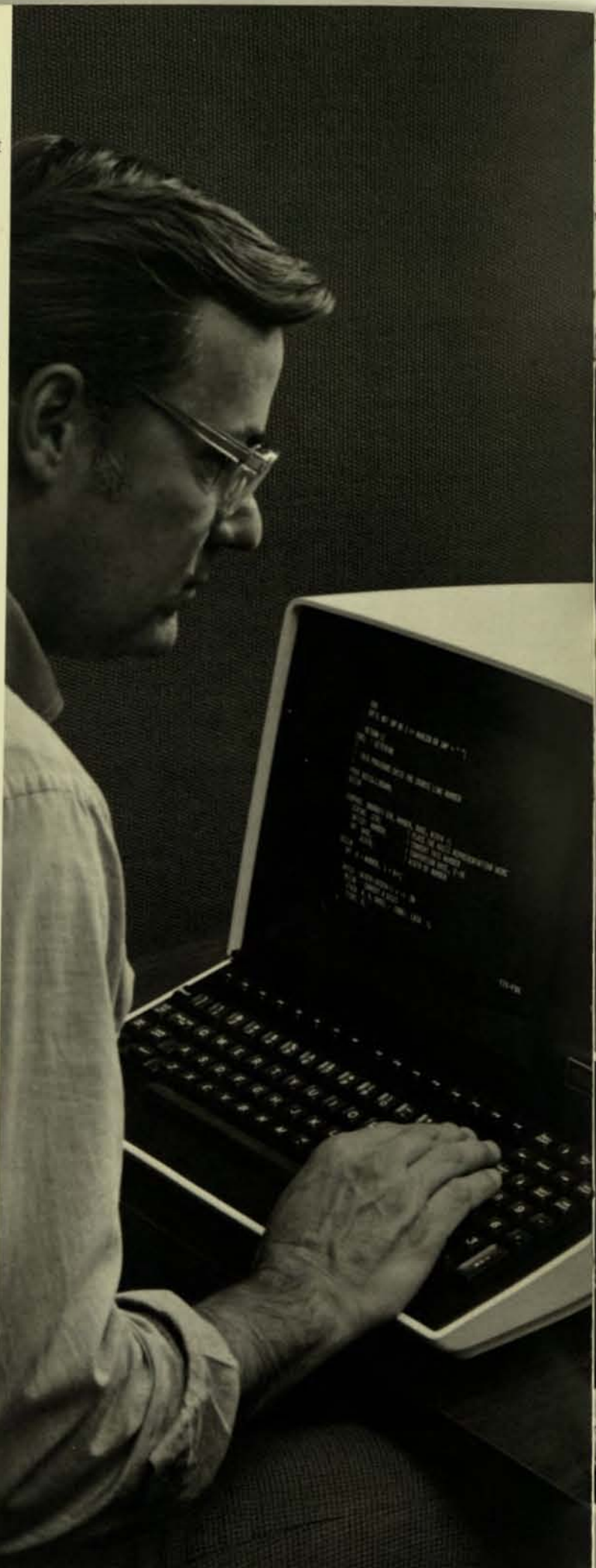
Can you tell me if the aggressive product and pricing announcements by IBM in 1979 had a meaningful impact on incoming order trends or the on-line processing market?

—Michael P. DeSantis, Partner,
Robertson, Colman, Stephens & Woodman

Tandem does not compete directly with IBM in that IBM products—and those of other manufacturers—do not have the functional capabilities of Tandem's. Any announcement by IBM, however, does impact the market as a whole; IBM is a powerful market force. And, certainly in a general sense, Tandem does compete with IBM for capital equipment expenditures. Consequently, many purchase decisions market-wide may be temporarily frozen while buyers investigate the new offering. Some manufacturers are directly and materially affected. The effect on Tandem was not appreciable as evidenced by our 1979 quarter-to-quarter shipment rates and net income.

Overall, we regard the implications of IBM's 1979 announcements as favorable to Tandem. They dramatically reconfirm our belief that the pricing trend of hardware is downward and that of software upward, reflecting more realistically the value added. This trend of charging more for software is immensely favorable to Tandem because we have great value-added in our proprietary software.

The company's new Tandem 6520 terminal, introduced during 1979, is on-line optimized for greater reliability and increased productivity.





The thrust of Tandem's commitment to the on-line transaction processing marketplace in the future will be an extension of that of the past: a continuing dedication to further reduce the user's *full costs* per transaction.

Isn't the ultimate limit of Tandem's marketplace the ability of customers to program complex on-line applications and, therefore, doesn't Tandem address narrow markets rather than large, broad markets? Given that, what company strategies can be brought to bear toward solving that market limitation?

—Peter Labé, First Vice President,
Smith Barney, Harris Upham & Co.

As discussed earlier, Tandem is addressing the broad market of on-line transaction processing in the stand-alone and distributed data processing environments. This market has been expanding rapidly because many businesses are strongly motivated by numerous economic factors to bring functions on-line.

The market existed before Tandem was founded, and the economic incentives of on-line automation have been compelling: Users have been willing to undertake massive on-line conversion programs to achieve larger economic objectives. Tandem, through its many contributions, has significantly lowered the total costs of on-line conversions and operations, and thereby expanded the on-line appeal.

Tandem has been cognizant of the complexity of on-line applications programming, and in October 1979 we announced a major step forward—a new software tool called PATHWAY. With first deliveries in February 1980, PATHWAY will significantly reduce the user's task and costs in developing on-line applications software.

For a number of reasons, we expect PATHWAY to profoundly broaden the appeal of on-line transaction processing and the market for Tandem systems.

Before PATHWAY, programmers had to achieve an extremely high skill level before undertaking the writing of on-line applications. Consequently, the supply of these high-level programmers has been relatively scarce. PATHWAY, in effect, dramatically increases the supply of on-line-competent programmers by simplifying application programming.

PATHWAY also opens up a whole new world of applications that can be brought on-line much faster—by many months. This factor will provide potential users with added impetus to convert to on-line processing, and provides Tandem with yet another competitive lead.

And, PATHWAY drastically reduces the costs of developing on-line applications, thereby expanding the cost-effectiveness appeal of on-line transaction processing.

It is the continuing strategy of Tandem to endow our products with capabilities that make them easier to use and more productive.

◀ PATHWAY, one of Tandem's major software announcements during 1979, will make it easier, faster and significantly less costly for customers to develop on-line applications.



Lieberman Enterprises acquired its first Tandem system in 1976 to enhance profitability by converting from batch processing to on-line processing, and to provide for continued growth without costly re-programming and interrupting operations. In the preceding years, the distributor of phonograph records had undergone a number of costly mainframe computer changes to keep pace with workload demands of rapid growth. Today, the Minneapolis-headquartered company's operations are critically dependent on Tandem's NonStop capability as well. Some 160 salesmen enter orders directly into the computer from telephones in the field, using hand-held acoustical couplers, while orders received directly from customers are simultaneously entered by terminal operators. On a peak day, some 30,000 orders are handled in this manner. The Tandem system generates warehouse picking lists, and additionally performs all accounting, inventory status, and sales analysis functions. Lieberman has sales of approximately \$150,000,000 annually.

Discuss the product evolution over the next 3-5 years necessary to make the company a healthy, fast-growing entity in an increasingly competitive market.

—Irwin Lieber, Partner,
First Manhattan Company

Tandem is now a "healthy, fast-growing entity", and the strategy to sustain that status is one of continuing to help customers solve the problems of the on-line environment by making Tandem systems easier to use, easier to maintain and functionally enhanced while simultaneously reducing users' total costs.

For all users, hardware costs are becoming relatively

less significant when compared to software costs. It follows, therefore, that anything Tandem can do to reduce the users' labor-intensive software costs will attract attention and business. That is exactly what we have been doing, and what we will continue to do.

System serviceability will also grow in importance to the user in direct relationship to the growth of networks. More self-diagnostics will be built into hardware, and more central site troubleshooting capabilities will be introduced to facilitate more rapid isolation of malfunctions within a network of systems. DIAGLINK, which enables Tandem support personnel to remotely diagnose users' systems, is an example of the trend of service sophistication.

Tandem has no need, however, to introduce a model "B" or "C" inasmuch as the Tandem system is, in fact, a

With automated wagering systems operating at over 50 pari-mutuel facilities, Delaware-based Autotote Limited placed its bets on Tandem's NonStop capabilities when going on-line at user sites with its new Autotrak system in 1979. At Autotrak-equipped facilities—the first three are at Harrisburg area's Penn National Race Course, Cleveland's Northfield Park and Miami's World Jai Alai—bettors need not go to separate windows for different denomination wagers or to cash-in winning tickets. All windows handle all types of bets of any dollar amount, and all windows are cashiers. The new Tandem-based system with four processors at each track continually updates odds and posts them; writes computerized tickets; validates winning tickets, and calculates payouts. The system also provides detailed analysis of every transaction at every window, automatically determines the state's share of revenues, and generates management reports. In another Autotote division, the company operates revenue control systems at a number of major airport and municipal parking facilities. That division's first Tandem-powered on-line system is controlling revenues from the 28 entry-exit lanes at Detroit Metropolitan Airport.





family of systems by virtue of modular expandability, with a continuous range of models from mid-size through mainframe-size. This does not imply, however, that we are not continuously investing in product development to improve the performance of our products and lower the cost per transaction to our customers.

Given Tandem's high-performance transaction orientation and inherent networking capabilities, how does the company see itself positioned to participate in the "office of the future?"

—James W. Reynolds, Vice President,
Bateman Eichler, Hill Richards

If we are talking about the office of the future as it relates to large organizations with multiple locations, Tandem is ideally positioned because the key to the office of the future is on-line computers connected together in a network. It is not a different marketplace, but rather a functional extension of on-line and distributed data processing networks.

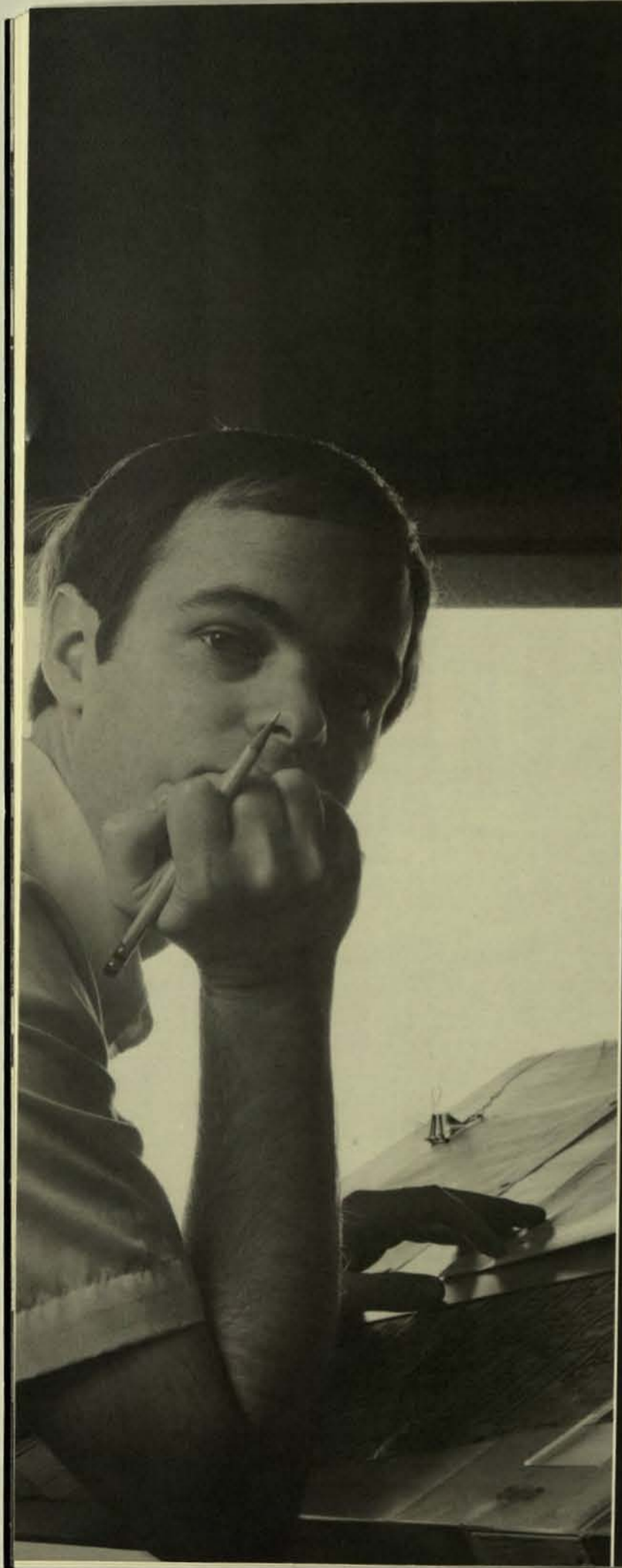
The concept of the future office, as we view it, is one of increasing the efficiency and productivity of management by having the ability to easily access and move day-to-day information. This cannot be done without on-line computers, and it cannot be done over great distances without on-line networks.

The realization of the office of the future will be an evolutionary process. First one location will be automated, then another, then another. Certain office functions will be brought up on-line—such as electronic mail, text processing, and various daily reports—and then the individually automated sites will be tied together in an on-line network where data can be accessed, edited and reports sent around the world. We are already doing many of these things at Tandem. We send memos out over our network from California that are received at specific individuals' terminals at any Tandem office in the world that is tied into our network. We also edit copy on-line in California that was generated at any of our offices on the network.

Once an on-line data processing system or network is in place, there is no limit to the number of functions that can be added. Functions will amount to being only specialized software and, in some cases, specialized terminals that are easy to use by non-technical individuals. The network user has already paid for the capability and for the communications lines; The incremental costs of adding office functions will be minimal. The foundation for the office of the future, for that user, is here today.



Pitney Bowes, a leading U.S. manufacturer of mailing equipment, came on-line in 1979 with a Tandem system which allows customers to buy postage without taking their postage meters to a post office for resetting. Known as the Remote Meter Resetting System (RMRS), this service is being operated six days per week for Pitney Bowes customers, and replaces a physical trip to the post office with a 90-second on-line toll-free phone call. Code information is requested by a computer-activated voice response, the user is issued a unique resetting number, and the meter is then credited with an amount of postage drawn from a pre-deposited trustee bank account. Pitney Bowes has installed more than 900,000 postage meters throughout the U.S., Canada and Great Britain.



4 Total employees nearly doubled during fiscal 1979. At fiscal year end, almost 60% of all Tandem people were in the marketing and field service organization.

Integral to Tandem's strategies for the Eighties is a framework for controlling high growth.

In a company enjoying rapid growth, it often happens that financial controls and organizational structure are insufficient to handle the growth. What steps are being taken by the company now to build a long-term growth enterprise and avoid those problems?

—Peter Labé, First Vice President,
Smith Barney, Harris Upham & Co.

As a preface, it should help to understand that the manner in which Tandem is managed is strongly influenced by the intent, since the inception of Tandem, to build a large company.

Tandem was not a "garage start-up". The original long-range plan was developed during the year prior to incorporation by several of the founders while on the staff of the venture capital firm that provided the seed money to launch the company. Tandem was then founded and went into operation five years ago with seasoned management—people who have had prior experience with cold starts that became large, successful companies.

Our current long-range plan, as with the original one, takes into account balanced growth of all functional areas of the company. We have designed control systems for each of the functional areas that envision future needs. These systems are computerized—and they are on-line. In fact, there is probably not another company of Tandem's size that uses computers in management control to the extent that we

In the early 1970s, Fred Meyer, the largest retailer in the northwest, reasoned that it was cashing so many checks that it should be in the banking business. Today, Fred Meyer Savings & Loan has over a half billion dollars on deposit at in-store facilities and free-standing branches. A Tandem System went on-line at Fred Meyer at the outset of 1979 to provide control and all data processing for the S&L's 30 automatic teller stations in Oregon that handle some 5,000 customer transactions daily. Another 5,000 daily transactions run through the system via 50 terminals to manage administrative and loan functions. In 1980, the growing Tandem system will begin driving all of the 150 regular teller terminals at the S&L's Oregon locations. The original applications software for Fred Meyer was developed by Applied Communications, Inc., of Omaha, Nebraska, which specializes in programs for financial institutions.



4 Far left photo: Tandem's engineering development group continues to work on programs to further improve the functional capabilities and the performance of the Tandem system.

do. These systems, by evolutionary process, are becoming more sophisticated as we grow.

The essential ingredients in our growth plan—now as in the past—are to be highly profitable, hire only good people, make sure our customers are satisfied, and have the organizational framework to handle high growth.

Financially, it is our objective to maintain a strong balance sheet and keep pretax margins in the 16% to 20% range. We do not lease our systems now and we do not intend to in the future. Approximately 10% of Tandem's systems are now financed by third-party lessors under full payout lease arrangements.

It is clear that our high growth rate cannot be financed solely by internally generated cash flow. Our options are to substantially slow the company's rate of growth or, within the self-imposed constraints of maintaining good profitability, to finance rapid growth with externally generated capital. We have chosen the latter course—using equity to maintain a strong balance sheet—to rapidly build the critical mass that will position Tandem as a long-term growth enterprise.

Explain the cost controls available with your forward, marketing/support integration versus backward, production integration strategy of traditional vendors.

—Barry F. Bosak, Vice President,
Research Division,
F. Eberstadt & Co., Inc.

It is my understanding that Tandem manufactures a smaller percentage of its total hardware than other computer manufacturers. Will this fact allow Tandem to grow faster?

—Thomas E. Mancino, Senior Research Officer,
Citibank

Tandem does not manufacture the peripherals portion of the Tandem system, and has no immediate plans to do so. Vertical integration of peripherals would not improve our margins at present, and would take resources away from our end-user marketing and support efforts. Our strategy is to concentrate resources on product development, marketing and support of our customers, and to build a sound base of satisfied customers as rapidly as possible while maintaining good profitability. At some point in the distant future, however, the economies of vertical integration could change. We are interested in optimizing peripherals for on-line transaction and distributed data processing efficiency, and are encouraging our OEM suppliers to add functional features toward

that end. Our new 6520 terminal, although proprietary and manufactured to our specifications, is produced for us by an OEM supplier.

The Tandem system architecture—the modularity, fault-tolerance and integral diagnostics—provide us with significant manufacturing advantages. The system design allows a much more rapid completion of systems integration and test than in most conventional computers.

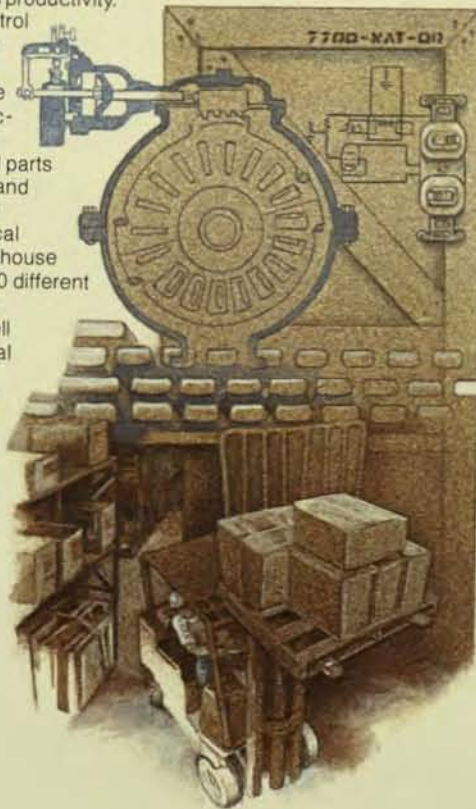
For our customers, this shortens the leadtime between initial order placement and system delivery. For Tandem, the comparatively short systems integration and test cycle means better control of inventories, more accurate manufacturing forecasting, and a higher degree of timely responsiveness to the marketplace.

Although the Tandem architecture is sophisticated, the modularity aspect greatly simplifies the manufacturing process. At the heart of each system, regardless of configuration, is some combination of just 17 standard, Tandem-designed, large-capacity printed circuit boards. Each modular cabinet can contain up to four processors, and any size system—from two to 16 processors—can be efficiently assembled and tested from the same standard boards, enabling identical software to be run on any size system.

This means that Tandem can build an entire family of systems—from a mid-size model through a large-scale system—at the same plant location, from the same standard modules, and at the same manufacturing station.

Busch-Jaeger-Elektro, the West German subsidiary of BBC, Brown Boveri Cie AG, automated its huge warehouse in Luedenscheid with a Tandem system in 1979 to significantly improve shipping rates and productivity.

The inventory control system manages the distribution throughout Europe of some 1,700 electrical products ranging from small parts such as switches and power breakers to all kinds of electrical devices. The warehouse stocks up to 13,000 different products, both finished goods as well as parts, with a total well in excess of 1,000,000 items in stock. The Tandem system automatically processes orders for 18,000 different locations in the warehouse.



Memphis Light, Gas & Water took delivery of its initial Tandem

system, a four-processor unit, during 1979 with the intent of standardizing its computer operations on a single manufacturer's system. The utility selected Tandem because of the system's unique ability to handle all mini-based functions while growing into a mainframe configuration. The first major usage of the Tandem system is an on-line, automated billing system for the utility's approximately 3,500 large industrial customers. During 1980, the system is scheduled to take over all customer billing operations. MLG&W has approximately 300,000 electric customers, 230,000 gas customers and 200,000 water customers.

What is Tandem's strategy for attracting and keeping people to achieve continued high growth?

*—William S. Deakyn, Vice President,
Jennison Associates Corp.*

Before discussing strategy, it is important to understand that Tandem is an interesting place to work, and a good place to work. We intend to keep it that way. We are on the leading edge of technology, we have a new way of looking at computers, and we are in an exciting market. Our high growth rate affords individuals the opportunity for career growth at a rate consistent with their abilities to manage more responsibility. All of these things attract good people to Tandem.

We have intentionally created a team spirit by having clear corporate objectives and a minimal structure. We function on individual responsibility and peer pressure—no one wants to let anyone else down. Everyone has well-defined goals and is delegated the authority to achieve those goals.

We are willing to take longer to find the right people, and then we take care of them. Because we really care. Our salaries are competitive, as are our benefits. In addition,

virtually all of our employees are shareholders or hold stock options—we want all of our people to share in the financial success of Tandem. After four years with Tandem, all North American employees are eligible for a fully-paid, six-week sabbatical in addition to regular vacation time.

Finally, we are willing to admit to our hiring mistakes by terminating individuals who do not live up to our high standards.

There are only a handful of companies that have enjoyed sustained success in the minicomputer or small computer system marketplace, and these include, most notably, broad-based, highly-integrated companies. There have been considerably more failures than successes over the years. What are the potential pitfalls that most concern Tandem? And, what are the ingredients that will make Tandem, a relatively recent entry into this marketplace now mainly populated by well-established companies, a long-term success?

—Barry Rosenberg, Senior Vice President,
G.S. Grumman/Cowan & Co.

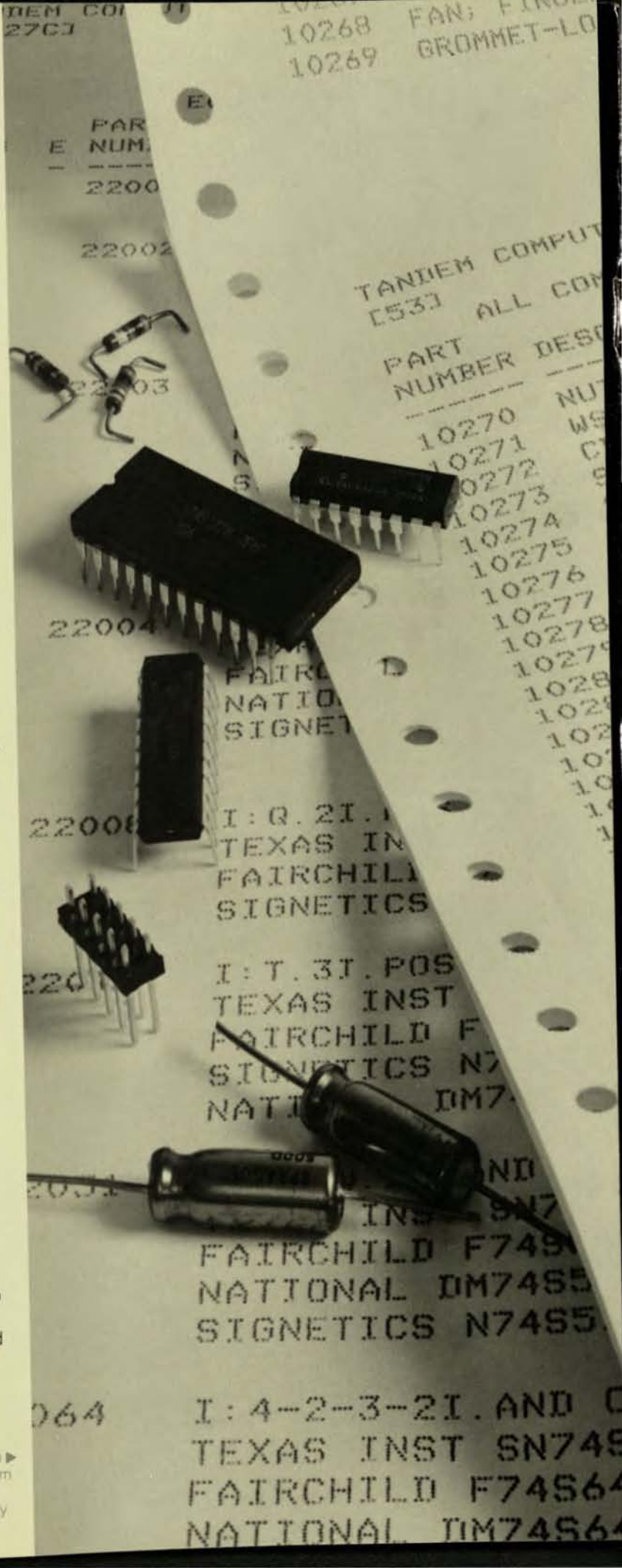
Tandem offers a continuous range of systems from mid-size through large, mainframe-size, and is, therefore, not in the "minicomputer or small computer system marketplace." Our list of success ingredients starts with this product line and all of the user benefits associated with it as previously discussed. In summary, those benefits are unique not only in themselves, but in that they are represented in a system that is the first developed specifically to fulfill the previously unfulfilled, critical needs of on-line transaction processing and distributed data processing networks.

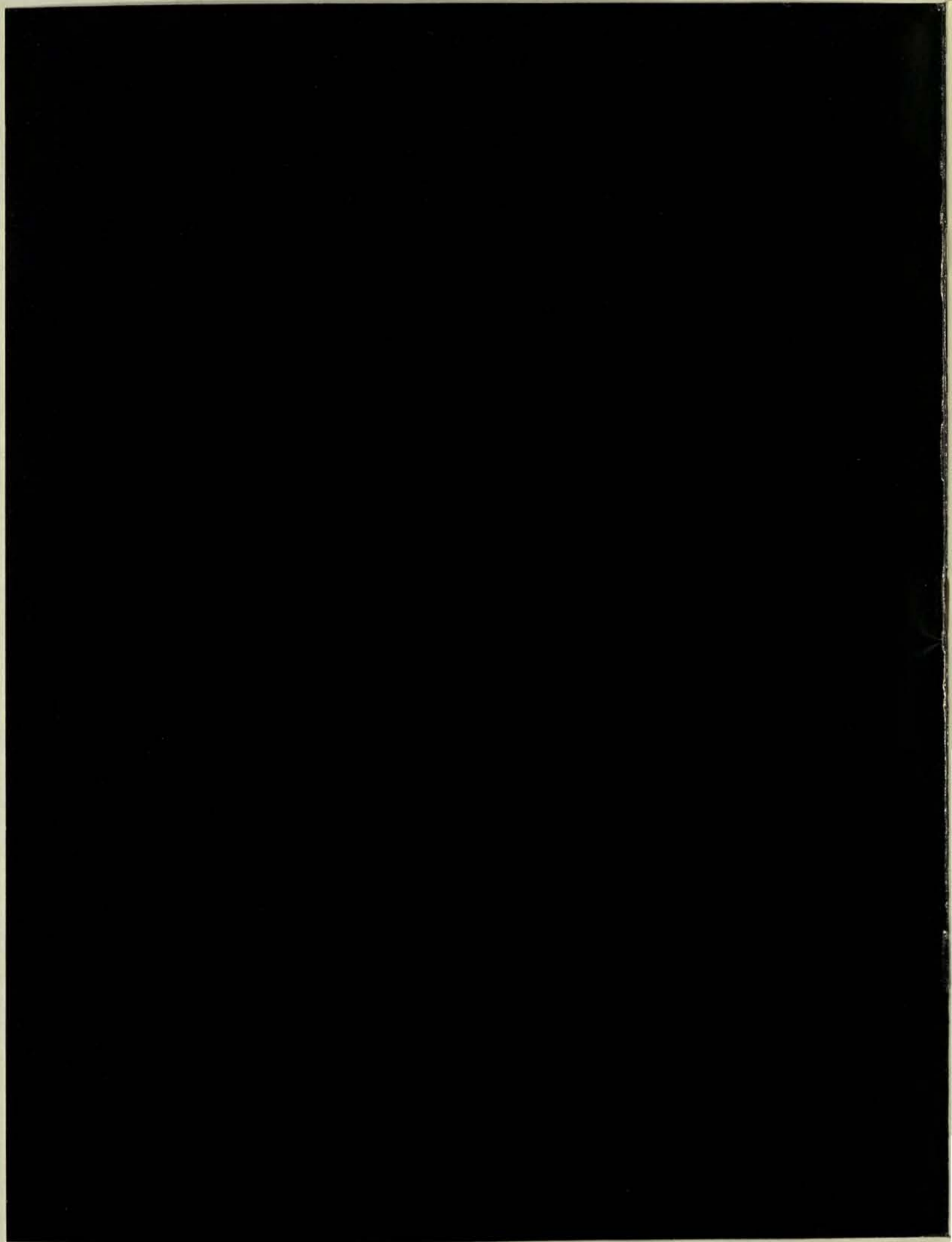
Tandem is not the first company to see a major new market that the "well-established companies" ignored. Early minicomputer vendors were quick to capitalize on just such a market opportunity. Tandem was founded five years ago to meet the unique needs of the emerging market for on-line transaction processing systems. We know that this market is large, and we believe that it is now among the fastest growing segments of the computer industry.

Our long-term success formula is based on five fundamentals: a strategic and superior product; quality people; satisfied customers; sustained profitability, and a framework for high growth. We will not allow the company to grow faster than the rate at which we can attract and productively use top talent.

We will not sacrifice continued customer satisfaction to achieve an arbitrarily established rate of growth. We will continue to maintain our high level of customer support, and continue to invest in the development of products and services that enhance the usefulness and productivity of Tandem systems.

Among the computerized control systems on-line at Tandem is an EXPAND-based manufacturing inventory control system that enables management to instantaneously determine the quantity of parts, sub-assemblies and finished product at any of the company's three manufacturing locations.





Highlights of the Year

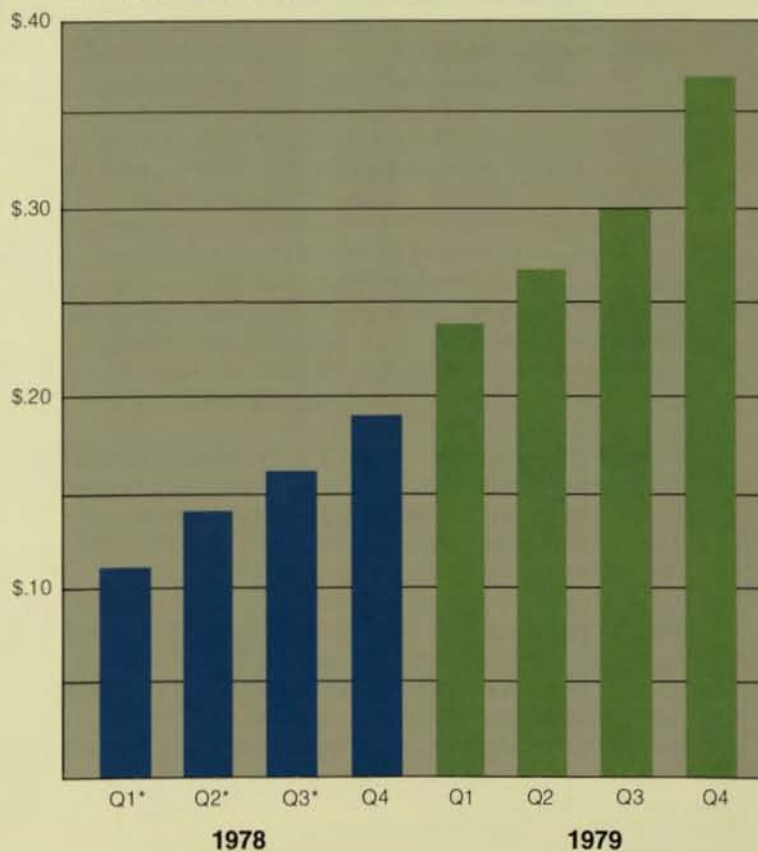
Fiscal year ended September 30	1978	1979
Revenues	\$24,305,000	\$55,974,000
Income Before Income Taxes	\$ 4,490,000*	\$10,104,000
Pre-Tax Return on Revenues	18.5%	18.1%
Net Income	\$ 2,153,000*	\$ 4,920,000
Income Per Share	\$.60*	\$1.18
Weighted Average Shares Outstanding	3,589,974	4,178,378
Working Capital	\$13,702,000	\$27,096,000
Total Assets	\$22,051,000	\$45,947,000
Shareholders' Equity	\$15,538,000	\$31,530,000
Number of Employees	446	828

Quarterly Results (Unaudited)

	1978				1979			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Revenues	\$ 3,909,000	\$ 5,259,000	\$ 6,699,000	\$ 8,438,000	\$10,398,000	\$12,471,000	\$14,992,000	\$18,113,000
Net Income	337,000*	532,000*	574,000*	710,000	951,000	1,123,000	1,295,000	1,551,000
Income Per Share	\$.11*	\$.14*	\$.16*	\$.19	\$.24	\$.27	\$.30	\$.37

*Before extraordinary credit

INCOME PER SHARE



TO OUR SHAREHOLDERS:

Tandem Computers, in the five years since its founding, has become a leader in the field of on-line transaction oriented data processing. The unique Tandem contributions of NonStop computing, modular expansion, data integrity and networking—together with a strong dedication to the support of our customers—have enabled us to grow very rapidly and successfully along with this expanding, major new market. We are pleased to report these results in fiscal 1979:

- continued excellent growth and operating profit
- major new product introductions which significantly expand our opportunities and enhance our already strong competitive position
- significant enlargement of our organization with outstanding people who are dedicated to Tandem's leadership role

Tandem's revenues grew 130% in fiscal 1979 to \$55,974,000. Our pretax margins of 18.1% were within our objective range of 16-20%. Income after tax advanced 129% to \$4,920,000 while income per share was \$1.18, up from \$.60 reported on a comparable basis in fiscal 1978. We ended fiscal 1979 in a sound financial position—with \$6.7 million in cash, a current ratio of 3.2:1 and a 5% debt to capitalization ratio. Our public offerings in December 1978 and November 1979 provided the company with \$34,428,000, which is being used to finance the working capital required to support our growth. With these resources plus our unused bank lines of credit for \$12,500,000 we believe that Tandem is in a strong and flexible financial position.

Fiscal 1979 was a year in which the marketplace increasingly came to understand Tandem's unique product features, and the importance of these features to the successful implementation of their on-line transaction processing applications. We have continued to build our end-user oriented marketing organization. At year end fiscal 1979, 474 of Tandem's 828 people were in our marketing and field service organization. Of these, approximately 15% were salespersons, and the remainder were systems analysts, field service engineers and training and headquarters marketing personnel, who together provide the high level of support that is essential to servicing our customers.

Consistent with our objective of providing a high level of customer support, we began decentralizing our manufacturing operations during the last year. A systems integration and test facility was opened on schedule in West Germany, with the objective being to bring the implicit high level of technical expertise of such an operation close to our users' sites. We have also commenced subassembly production in Watsonville, California.

We find that computer users in general are becoming increasingly sophisticated and are understanding that the *full cost* of any computer system includes not only the

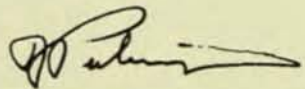
price of hardware, but also expenditures for system software, applications software, and maintenance of both hardware and software over the life of the system. Tandem's product thrust has been, and will continue to be in the 1980's, to offer customers the lowest *full cost* per transaction. Products announced during fiscal 1979 were illustrative of this corporate focus.

EXPAND, our network operating system, was delivered to 22 customers in the second half of the year. These customers are in the process of implementing their on-line distributed data processing applications on Tandem systems. ENFORM, Tandem's query and report writer language, which enables the user to access data using an English-like language no matter where the data might be located in the distributed data processing network, and to write reports, also enjoyed excellent initial acceptance in fiscal 1979. These two products greatly reduce the time and programming cost required for users to implement distributed data processing networks with distributed data bases. Other major product announcements over the last year include PATHWAY, a software product which performs many of the complex terminal and control functions that are an inherent part of any on-line transaction processing application, and the 6520 terminal, with unique design features that provide high reliability and improved user productivity in the on-line environment.

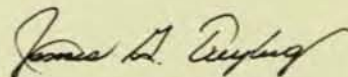
From a management viewpoint we believe that high growth requires a special structure of organization, one where at any point in time critical corporate resources—management, capital and control systems—are in place to handle volumes greatly in excess of what exists currently. We are committed to seeing that all individuals at Tandem have the opportunity for job advancement and, through option programs and the employee stock purchase plan, also have the opportunity to participate financially in the company's success. Virtually all of our employees are currently stock or option holders.

Our success over the last five years reflects the individual commitment, team effort, and pure hard work of all of our employees. We enter the 1980's in a strong position—in terms of our people, products, financial resources and base of satisfied customers—to take full advantage of the major market opportunity we identified five years ago.

Sincerely yours,



T.J. Perkins
Chairman of the Board



James G. Treybig
President and
Chief Executive Officer

December 5, 1979

CONSOLIDATED STATEMENT OF OPERATIONS

From date of incorporation to September 30, 1979

	(Dollars in Thousands Except for Per Share Data)				
	Year Ended September 30				
	1979	1978	1977	1976	1975*
Revenues	\$55,974	\$24,305	\$7,692	\$ 581	\$ —
Costs and Expenses:					
Cost of revenues	20,786	9,096	3,514	482	—
Product development	4,654	2,169	1,094	979	456
Marketing, general and administrative	20,828	8,808	2,719	1,327	192
Interest, net	(398)	(258)	36	(38)	(2)
	45,870	19,815	7,363	2,750	646
Income (loss) before income taxes and extraordinary credit	10,104	4,490	329	(2,169)	(646)
Provision for Income Taxes	5,184	2,337	171	—	—
Income (loss) before extraordinary credit	4,920	2,153	158	(2,169)	(646)
Extraordinary Credit—Tax benefit of net operating loss carryforwards	—	1,218	167	—	—
Net Income (Loss)	\$ 4,920	\$ 3,371	\$ 325	\$(2,169)	\$(646)
Income (Loss) Per Common Share:					
Income (loss) before extraordinary credit	\$ 1.18	\$.60	\$.06	\$ (4.33)	\$(1.49)
Extraordinary credit	—	.34	.06	—	—
Net income (loss)	\$ 1.18	\$.94	\$.12	\$ (4.33)	\$(1.49)
Weighted average outstanding shares	4,178,378	3,589,974	2,679,923	530,270	440,143

*From date of incorporation, November 29, 1974

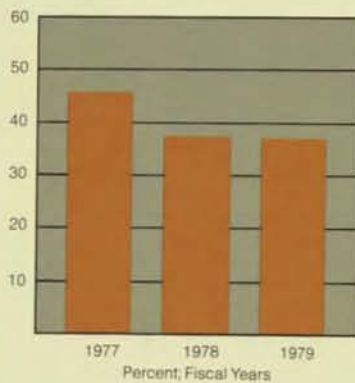
MANAGEMENT'S DISCUSSION AND ANALYSIS OF THE CONSOLIDATED STATEMENT OF OPERATIONS



Revenues

Revenues in fiscal 1979 increased 130% to \$55,974,000. This gain resulted primarily from increased shipments of systems and software to both new and existing customers and from sales of added processors, peripherals and software for existing systems. During fiscal 1979 the Company shipped 389 processors to 118 customers.

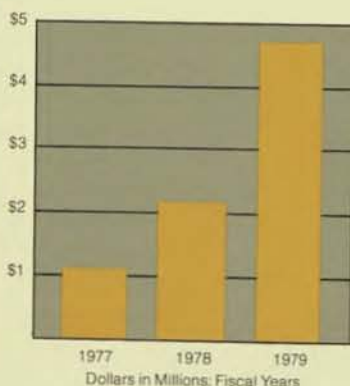
Revenues in fiscal 1978 increased 216% to \$24,305,000 from \$7,692,000 reported in fiscal 1977. This gain also resulted from a substantial increase in shipments of systems, processors and peripherals. During fiscal 1978 the Company shipped 176 processors to 59 customers.



Cost of Revenues

To meet rapidly increasing market demand for Tandem's products, manufacturing facilities were expanded during the year at the Company's Cupertino headquarters, and new plants were opened in Watsonville, California, for sub-assembly production and Neufahrn, West Germany, for systems integration and test. The Company's cost of revenues in fiscal 1979 increased 129% to \$20,786,000, while cost of revenues as a percentage of revenues remained essentially unchanged at 37.1%.

The cost of revenues in fiscal 1978 increased 159% to \$9,096,000. However, cost of revenues as a percentage of revenues declined to 37.4% from 45.7% in fiscal 1977, primarily because of per-unit price reductions and quantity discounts received by the Company due to the substantial increase in volume.



Product Development

The Company's product development effort is dedicated to meeting the needs of computer users who are implementing on-line transaction processing applications. Expenditures on product development in fiscal 1979 were \$4,654,000, up 115% over the fiscal 1978 level. In fiscal 1978 product development expenditures were \$2,169,000, up 98% over the prior year. These expenditures resulted in many new product introductions in both years and funded research on future products.

Product development expenditures as a percentage of revenues were 8.3% and 8.9% in fiscal 1979 and 1978, respectively. The Company has a long-term objective of maintaining development expenditures at approximately 9% of revenues.

Marketing, General and Administrative

The Company focuses its selling efforts on the end-user market, where providing a high level of customer support is essential. These support costs are encompassed in marketing, general and administrative expenditures, which increased 136% in fiscal 1979 to \$20,828,000. In fiscal 1978, marketing, general and administrative expenditures were \$8,808,000, up 224% over the prior year. In fiscal 1979 and 1978 such costs represented 37.2% and 36.2% of revenues, respectively. This expenditure level reflects not only the Company's end-user marketing orientation but also the significant geographical expansion of marketing operations over the last two years and the addition of marketing personnel in anticipation of future growth.



Pretax Income

Pretax income increased 125% to \$10,104,000 in fiscal 1979, while pretax margins (pretax income as a percentage of revenues) were 18.1% compared with 18.5% reported in fiscal 1978. The decline in pretax margins in fiscal 1979 resulted primarily from the reduction as a percentage of revenues in net interest income earned on cash equivalent investments. In fiscal 1978 pretax income was \$4,490,000, as compared to \$329,000 in fiscal 1977. The pretax margin improvement in fiscal 1978 resulted primarily from the reduction in cost of revenues and product development expenditures as a percentage of revenues.

The Company's effective tax rate declined slightly in fiscal 1979 to 51.3% from 52.0%, reported in both fiscal 1978 and 1977. Net income before extraordinary credit advanced 129% to \$4,920,000 in fiscal 1979. In fiscal 1978 and 1977 net income before extraordinary credit was \$2,153,000 and \$158,000, respectively.



Income Per Common Share

Income per common share* increased 97% to \$1.18 in fiscal 1979. Earnings* in fiscal 1978 and 1977 were \$.60 and \$.06 per share, respectively. Per share earnings have not increased as rapidly as net income before extraordinary credit because of increased shares outstanding. The Company completed public offerings of common stock in fiscal 1979 and 1978, and these offerings combined with the sale of stock to employees under the employee stock purchase plan and option plans resulted in increases in weighted average shares outstanding of 16% and 34%, respectively. Proceeds from the sale of these additional shares have been used to finance working capital expansion, which was necessary to support the Company's growth during this period.

*Before extraordinary credit.



CONSOLIDATED STATEMENT OF INCOME

For the Years Ended September 30, 1979 and 1978

(In Thousands Except for Per Share Data)

	1979	1978
Revenues (Notes 1 and 8)	\$55,974	\$24,305
Costs and Expenses:		
Cost of revenues	20,786	9,096
Product development	4,654	2,169
Marketing, general and administrative	20,828	8,808
Interest expense	84	65
Interest income	(482)	(323)
	45,870	19,815
Income before income taxes and extraordinary credit	10,104	4,490
Provision for Income Taxes (Note 2)	5,184	2,337
Income before extraordinary credit	4,920	2,153
Extraordinary credit—Tax benefit of net operating loss carryforwards	—	1,218
Net Income	\$ 4,920	\$ 3,371
Income Per Common Share (Note 7)		
Income before extraordinary credit	\$ 1.18	\$.60
Extraordinary credit	—	.34
Net income	\$ 1.18	\$.94

The accompanying notes are an integral part of this statement.

CONSOLIDATED BALANCE SHEETS

September 30, 1979 and 1978

	(In Thousands)	
ASSETS	1979	1978
Current Assets:		
Cash (Note 4)	\$ 2,198	\$ 1,063
Cash investments	4,560	3,384
Accounts receivable	19,881	8,115
Inventories (Note 1)	11,304	6,319
Prepaid expenses	1,385	619
Total current assets	39,328	19,500
Property and Equipment, at cost (Notes 1 and 3):		
Production and test equipment	1,982	506
Computer equipment	2,417	1,105
Office furniture and equipment	382	176
Systems spares	2,141	775
Leasehold improvements	1,597	606
	8,519	3,168
Less—Accumulated depreciation and amortization	1,900	617
	6,619	2,551
	<u>\$45,947</u>	<u>\$22,051</u>

	(In Thousands)	
LIABILITIES AND SHAREHOLDERS' INVESTMENT	1979	1978
Current Liabilities:		
Current maturities of capitalized lease obligation	\$ 375	\$ 203
Accounts payable	5,675	3,766
Accrued expenses	1,269	953
Accrued income taxes	4,913	876
Total current liabilities	12,232	5,798
Capitalized Lease Obligation, net of current maturities (Note 3)	1,144	715
Deferred Income Taxes	1,041	—
Commitments (Note 6)		
Shareholders' Investment (Note 5):		
Preferred stock—\$.10 par value, authorized 2,400,000 shares; none outstanding	—	—
Common stock—\$.05 par value, authorized 10,000,000 shares; outstanding 3,675,981 shares in 1978 and 4,169,749 shares in 1979	209	184
Additional paid-in capital	25,520	14,473
Retained earnings	5,801	881
Total shareholders' investment	31,530	15,538
	<u>\$45,947</u>	<u>\$22,051</u>

The accompanying notes are an integral part of these balance sheets.

CONSOLIDATED STATEMENT OF SHAREHOLDERS' INVESTMENT

For the Years Ended September 30, 1978 and 1979

(In Thousands)

	Preferred Stock		Additional Paid-in Capital	Common Stock		Retained Earnings (Deficit)	Total Share- holders' Investment
	Shares	Amount		Shares	Amount		
Balance, September 30, 1977	2,074	\$207	\$ 4,987	626	\$ 31	\$(2,490)	\$ 2,735
Sale of preferred stock	125	13	987	—	—	—	1,000
Conversion of preferred stock into common stock	(2,199)	(220)	110	2,199	110	—	—
Sale of common stock, net of related expenses	—	—	7,849	770	39	—	7,888
Sale of stock under stock option and stock pur- chase plans, net	—	—	306	81	4	—	310
Income tax benefit result- ing from exercises of non- qualified stock options and early disposition of shares acquired under qualified stock options	—	—	234	—	—	—	234
Net income	—	—	—	—	—	3,371	3,371
Balance, September 30, 1978	—	—	14,473	3,676	184	881	15,538
Sale of common stock, net of related expenses	—	—	10,054	420	21	—	10,075
Sale of stock under stock option and stock purchase plans, net	—	—	758	74	4	—	762
Income tax benefit resulting from exercises of non- qualified stock options and early disposition of shares acquired under qualified stock option and stock purchase plans	—	—	235	—	—	—	235
Net income	—	—	—	—	—	4,920	4,920
Balance, September 30, 1979	—	\$ —	\$25,520	4,170	\$209	\$5,801	\$31,530

The accompanying notes are an integral part of this statement.

CONSOLIDATED STATEMENT OF CHANGES IN FINANCIAL POSITION

For the Years Ended September 30, 1979 and 1978

	(In Thousands)	
	1979	1978
Working Capital Provided From (Used For):		
Net income before extraordinary credit	\$ 4,920	\$ 2,153
Add back:		
Depreciation and amortization	1,365	457
Deferred income taxes	737	—
Working capital provided from operations	7,022	2,610
Extraordinary credit	—	1,218
Acquisition of property and equipment	(5,770)	(2,387)
Net book value of equipment sold or retired	337	84
Increase in capitalized lease obligation, net of current maturities	429	399
Increase in deferred income taxes	304	—
Sale of preferred stock	—	1,000
Sale of common stock, net	10,837	8,198
Tax benefit of stock options	235	234
Net increase in working capital	\$13,394	\$11,356

Working Capital Increase Represented By:

Increase in current assets—		
Cash and cash investments	\$ 2,311	\$ 4,338
Accounts receivable	11,766	5,513
Inventories	4,985	4,457
Prepaid expenses	766	527
Decrease (Increase) in current liabilities—		
Notes payable to bank	—	800
Current portion of capitalized lease obligation	(172)	(112)
Accounts payable	(1,909)	(2,651)
Accrued expenses	(316)	(644)
Accrued income taxes	(4,037)	(872)
Net increase in working capital	\$13,394	\$11,356

The accompanying notes are an integral part of this statement.

NOTES TO CONSOLIDATED FINANCIAL STATEMENTS**1. Summary of Significant Accounting Policies****Consolidation**

The consolidated financial statements include the accounts of Tandem Computers Incorporated and its wholly owned subsidiaries after elimination of intercompany accounts and transactions. Foreign exchange gains and losses are not significant and are reflected in the results of operations.

Revenue Recognition

The Company generally recognizes revenues at the time of shipment.

Inventories

Inventories are stated at the lower of cost (first-in, first-out) or market and include material, labor, and manufacturing overhead. The components of inventory used to determine cost of revenues were:

(In Thousands)	September 30		
	1977	1978	1979
Purchased parts and subassemblies	\$1,185	\$4,196	\$ 6,207
Work-in-process and finished systems	678	2,123	5,097
	\$1,863	\$6,319	\$11,304

Income Taxes

The Company provides for income taxes on total DISC income and accounts for investment tax credits as a reduction of the provision for taxes on income in the year in which the related credits are realized.

Property and Equipment

Systems spares are depreciated using the double declining balance method. All other property and equipment are depreciated using the straight-line method. The estimated useful lives are:

	Years
Production and test equipment	5-10
Computer equipment	5
Office furniture and equipment	5-10
Systems spares	4
Leasehold improvements	Lease Term

Expenditures for maintenance and repairs are charged to operations as incurred. Expenditures for major betterments and renewals are capitalized and depreciated over the estimated remaining useful life of the asset. The net gain or loss on assets retired or otherwise disposed of is credited or charged to operations and the asset cost and related depreciation are removed from the accounts.

2. Income Taxes

The provision for income taxes for the years ended September 30, 1978 and 1979 is comprised of:

	1978	1979
Current:		
Federal	\$ 755,000	\$3,850,000
State	307,000	916,000
Foreign	580,000	326,000
	1,642,000	5,092,000
Prepaid:		
Federal	—	(638,000)
State	—	(7,000)
	—	(645,000)
Deferred:		
Federal	288,000	718,000
State	23,000	—
Foreign	384,000	19,000
	695,000	737,000
	<u>\$2,337,000</u>	<u>\$5,184,000</u>

The sources of prepaid and deferred taxes were as follows:

Prepaid:		
Revenues recognized for taxes, but not for financial statements	\$ —	\$ 503,000
Expenses recognized for financial statements, but not for taxes	—	142,000
	\$ —	\$ 645,000
Deferred:		
DISC income	\$ 304,000	\$ 449,000
Increase in revenues deferred for foreign tax purposes	384,000	19,000
Accelerated depreciation and other, net	7,000	269,000
	\$ 695,000	\$ 737,000

The provision for income taxes differs from the amount obtained by applying the statutory Federal income tax rate to income before taxes as follows:

	1978	1979
Federal tax provision at statutory rate	\$2,155,000	\$4,698,000
State income taxes net of Federal income tax benefit	172,000	486,000
Foreign income taxes in excess of Federal tax rate	179,000	294,000
Investment tax credit	(104,000)	(216,000)
Other	(65,000)	(78,000)
	<u>\$2,337,000</u>	<u>\$5,184,000</u>

3. Capitalized Lease Obligation

As of September 30, 1979, the Company had leased from a bank \$1,766,000 of equipment for the period through fiscal 1984 with an option to purchase the equipment at the fair market value at the end of the lease period.

The following summarizes the future minimum lease payments together with the present value of the minimum lease payments as of September 30, 1979:

Year Ending September 30	
1980	\$ 443,000
1981	428,000
1982	374,000
1983	300,000
1984	259,000
Total minimum lease payments	1,804,000
Less: Amount representing interest (8%)	285,000
Present value of minimum lease payments	\$1,519,000

4. Lines of Credit

In November 1979 the Company amended its revolving line of credit with a bank which provided for unsecured borrowings of up to \$5,000,000. The line of credit, as amended, increases the amount of borrowings available to \$10,000,000 at the bank's prime lending rate and expires on December 31, 1980. The agreement requires the Company to maintain certain financial covenants and also maintain a compensating balance of 2.5% of the commitment plus 5% of borrowings in excess of \$2,500,000.

The Company has a revolving line of credit with another bank providing for unsecured borrowings of up to \$2,500,000. The line of credit expires April 30, 1980, provides for borrowings at the bank's prime lending rate, and requires the company to maintain a compensating balance of 5% of the commitment plus 5% of the credit line utilized.

Borrowings in fiscal 1978 were outstanding from October 1, 1977 to December 21, 1977. The average interest rate on borrowings during this period was approximately 7.5%. The average month-end borrowing was \$400,000 and the maximum borrowing at any month end was \$500,000. There were no borrowings under either line of credit during fiscal 1979.

5. Stock Option and Stock Purchase Plans

Stock Option Plans

The Company has three stock option plans in effect for employees. Under all plans, the option price may not be less than 100% of the fair market value on the date of grant. Under the qualified plan, adopted in 1976, all options granted are exercisable upon the date of grant and expire five years from the date of grant or on May 20, 1981, if earlier. Under the two non-qualified plans, options are exercisable upon the date of grant and expire no later than seven years from the date of grant. The non-qualified plans were adopted in 1976 and 1979; however, options granted under the 1979 plan (42,250 shares) become exercisable only upon approval of the shareholders at the next annual meeting, to be held in early 1980.

In addition, two option plans covering a total of 4,000 shares have been adopted for two directors. Under these plans, options to purchase 2,000 shares are exercisable at \$23.50 per share and an additional 2,000 options are exercisable at \$30.25 per share.

At September 30, 1979 and 1978, there were options for 359,700 and 90,647 shares, respectively, available for future grant. Following is a summary of activity under the plans:

Options outstanding as of September 30, 1979:

Granted in Fiscal Year	Number of Shares	Option Price		Fair Market Value at Date of Grant	
		Per Share	Total	Per Share	Total
1976	3,000	\$.50- 1.00	\$ 2,000	\$.50- 1.00	\$ 2,000
1977	32,182	1.00- 3.50	52,000	1.00- 3.50	52,000
1978	151,909	3.50-30.25	2,909,000	3.50-30.25	2,909,000
1979	293,750	22.50-32.88	7,900,000	22.50-32.88	7,900,000
	<u>480,841</u>		<u>\$10,863,000</u>		<u>\$10,863,000</u>

Options became exercisable as follows:

Became Exercisable in Fiscal Year	Number of Shares	Option Price		Fair Market Value at Date Option Became Exercisable	
		Per Share	Total	Per Share	Total
1978	249,550	\$ 1.00-30.25	\$ 3,408,000	\$14.50-30.25	\$ 4,441,000
1979	292,900	22.50-31.00	7,612,000	22.50-31.00	7,612,000

Options were exercised as follows:

Exercised in Fiscal Year	Number of Shares	Option Price		Fair Market Value at Date Option Exercised	
		Per Share	Total	Per Share	Total
1978	83,653	\$.50-16.75	\$ 400,000	\$ 3.50-37.00	\$ 2,042,000
1979	59,119	.50-28.75	653,000	23.25-36.00	1,676,000

Stock Purchase Plan

As of September 30, 1979 and 1978, the Company has reserved 78,223 and 96,112 shares, respectively, of Common Stock for future issuance under its employee stock purchase plan adopted in fiscal 1978. Eligible employees may elect to purchase shares of Common Stock at 85% of the lower of the fair market value at the beginning or end of a three-month offering period. During 1979 and 1978, the Company issued 17,889 and 3,888 shares, respectively, of Common Stock pursuant to this plan.

Proceeds from the sale of common stock under the stock option plans and the stock purchase plan are credited to the common stock account to the extent of par value and the remainder to additional paid-in capital. No charges or credits are reflected in the income statement with respect to stock options or stock purchase plans.

6. Commitments

The Company leases its headquarters, operating facilities, field offices and automobiles under operating lease agreements which expire through fiscal 2003. Future lease payments are as follows:

Year Ending September 30	
1980	\$2,223,000
1981	2,273,000
1982	1,875,000
1983	1,645,000
1984	1,109,000
1985-89	3,726,000
1990-94	156,000
1995-99	156,000
2000-03	125,000
	\$13,288,000

Rent expenses included in the results of operations for the years ended September 30, 1978 and 1979, are \$726,000 and \$1,738,000, respectively.

The Company has entered into an operating lease for 165,000 additional square feet of office space, which is under construction. The lease term will be for 15 years, beginning with occupancy (approximately February 1980), at an initial annual rental rate of \$1,145,000. The annual rental rate will be increased by 15% at the end of 5 and 10 years, respectively.

7. Income Per Common Share

Net income per common share for the years ended September 30, 1978 and 1979, has been computed based upon the weighted average number of common and common equivalent shares outstanding. Common equivalent shares in 1978 and 1979 result from the assumed exercise of stock options outstanding which have a dilutive effect when applying the treasury stock method. Total shares used in the computation were 3,589,974 for 1978 and 4,178,378 for 1979. Fully diluted income per share is substantially the same as reported income per share.

8. Geographic Segment Information

The Company designs, develops, manufactures, markets and services multiple processor computer systems. The following table sets forth information about the company's operations in different geographic areas for the years ended September 30, 1978 and 1979:

Year Ended September 30, 1978						(In Thousands)
	Geographic Area			Adjustments and Eliminations	Consolidated	
	United States	Europe	Other			
Revenues—						
Customers	\$16,837	\$ 7,124	\$ 344	\$ —	\$24,305	
Intracompany	3,904	—	—	(3,904)	—	
Total revenues	\$20,741	\$ 7,124	\$ 344	\$(3,904)	\$24,305	
Income (loss) before taxes and extraordinary credit	\$ 3,122	\$ 1,473	\$ (69)	\$ (36)	\$ 4,490	
Identifiable assets	\$17,821	\$ 4,097	\$ 226	\$ (93)	\$22,051	

Year Ended September 30, 1979						(In Thousands)
	Geographic Area			Adjustments and Eliminations	Consolidated	
	United States	Europe	Other			
Revenues—						
Customer	\$41,292	\$13,501	\$1,181	\$ —	\$55,974	
Intracompany	8,846	102	—	(8,948)	—	
Total revenues	\$50,138	\$13,603	\$1,181	\$(8,948)	\$55,974	
Income (loss) before income taxes	\$11,127	\$ 230	\$ (173)	\$(1,080)	\$10,104	
Identifiable assets	\$35,667	\$10,113	\$1,319	\$(1,152)	\$45,947	

Intracompany transfers are made at approximately arm's length prices, which include manufacturing profits attributable to United States operations. Identifiable assets are those assets of the Company that are identified with the operation of each geographic area.

Revenues in 1978 include sales of approximately \$2,500,000 to one customer.

AUDITORS' REPORT

To Tandem Computers Incorporated:

We have examined the consolidated balance sheets of Tandem Computers Incorporated (a California corporation) and subsidiaries as of September 30, 1979 and 1978, and the related consolidated statements of income, shareholders' investment and changes in financial position for the years then ended. Our examinations were made in accordance with generally accepted auditing standards and, accordingly, included such tests of the accounting records and such other auditing procedures as we considered necessary in the circumstances.

In our opinion, the consolidated financial statements referred to above present fairly the financial position of Tandem Computers Incorporated and subsidiaries as of September 30, 1979, and 1978, and the results of their operations and the changes in their financial position for the years then ended, in conformity with generally accepted accounting principles applied on a consistent basis.

Arthur Andersen & Co.

San Jose, California

November 5, 1979

TANDEM STOCK PRICE

Calendar Quarter Price	High	Low
1977 4th Quarter*	\$15 $\frac{3}{4}$	\$13 $\frac{3}{4}$
1978 1st Quarter	\$16 $\frac{3}{4}$	\$13 $\frac{3}{4}$
2nd Quarter	\$24 $\frac{3}{4}$	\$15
3rd Quarter	\$36 $\frac{1}{2}$	\$23
4th Quarter	\$33 $\frac{1}{2}$	\$22
1979 1st Quarter	\$29 $\frac{3}{4}$	\$22 $\frac{3}{4}$
2nd Quarter	\$32 $\frac{1}{4}$	\$28 $\frac{1}{2}$
3rd Quarter	\$35 $\frac{3}{4}$	\$26 $\frac{1}{4}$

*December 14 and thereafter.

Tandem Computers Incorporated common stock was offered to the public on December 14, 1977 at \$11.50 per share and thereafter has been traded in the over-the-counter market under NASDAQ symbol TNDM. High and low closing bid prices are shown above as reported by the National Quotation Bureau. These quotations represent prices between dealers, do not include markup, markdown or commissions, and may not represent actual transactions. No dividends have been declared on the common stock.

BOARD OF DIRECTORS

Thomas J. Perkins (1),
Chairman of the Board;
Partner, Kleiner, Perkins, Caufield & Byers
Morton Collins (2), Partner, DSV Associates
Thomas J. Davis, Jr. (1)(2),
Partner, Mayfield II
Franklin P. Johnson, Jr., President, Asset
Management Capital Company
Eugene Kleiner (2),
Partner, Kleiner, Perkins, Caufield & Byers
John C. Loustounou, Vice President, Chief
Financial Officer and Secretary, Tandem
Computers Incorporated
Alvin C. Rice, President and Chairman,
Imperial Bank
Robert G. Stone, Jr., Chairman of the Board,
West India Shipping Company
James G. Treybig (1), President and Chief
Executive Officer, Tandem Computers
Incorporated

(1) Member of Executive Committee
(2) Member of Audit Committee

OFFICERS

James G. Treybig, President and Chief
Executive Officer
Robert C. Marshall, Vice President and
Chief Operating Officer
Michael D. Green, Vice President—Software
Development
Lawrence A. Laurich, Vice President—
Engineering
John C. Loustounou, Vice President, Chief
Financial Officer and Secretary
David R. Mackie, Vice President—
Headquarters Marketing
Henry V. Morgan, Vice President
and Controller
Samuel J. Wiegand, Vice President—
Marketing
Jeanne D. Wohlers, Vice President,
Treasurer and Assistant Secretary

AUDITORS

Arthur Andersen & Co.,
San Jose, California

REGISTRAR AND TRANSFER AGENT

Bank of America N.T. & S.A.,
San Francisco, California

FORM 10-K

**A copy of the company's Form 10-K, as
filed with the Securities and Exchange
Commission, is available on written
request. Please direct request to:**

**Treasurer's Office
Tandem Computers Incorporated
19333 Vallco Parkway
Cupertino, California 95014**

ANNUAL MEETING

The annual meeting of stockholders will be
held at 10:00 a.m. on Friday, February 1,
1980, at the corporation's headquarters.

TANDEM

**Corporate Headquarters
19333 Vallco Parkway
Cupertino, CA 95014
(408) 996-6000**

DOMESTIC OFFICES

Headquarters Marketing

James A. Katzman, Vice President—
Marketing Support

Eastern Region

George Eckert, Vice President
Director, Eastern Region
One Penn Plaza
New York, NY 10001

Central Region

Michael Bateman, Vice President
Director, Central Region
1827 Walden Office Square
Schaumburg, IL 60195

Western Region

Charles W. Ryle, Vice President
Director, Western Region
1201 Watson Road
Arlington, TX 76011

Sales and Service Offices

Boston, Massachusetts
Chicago, Illinois
Cincinnati, Ohio
Cleveland, Ohio
Columbus, Ohio
Dallas, Texas
Denver, Colorado
Detroit, Michigan
Greensboro, North Carolina
Hasbrouck Heights, New Jersey
Houston, Texas
Las Vegas, Nevada
Long Beach, California
Milwaukee, Wisconsin
Minneapolis, Minnesota
New Orleans, Louisiana
Omaha, Nebraska
Philadelphia, Pennsylvania
Phoenix, Arizona
Pittsburgh, Pennsylvania
Rochester, New York
Salt Lake City, Utah
San Francisco, California
San Mateo, California
Seattle, Washington
St. Louis, Missouri
Tampa, Florida
Washington, DC

INTERNATIONAL OFFICES

Canada

Victor DeSouza
Managing Director, Tandem Computers
Canada Limited
180 Duncan Mill Road
Don Mills, Ontario M3B 1Z6
Offices also located in
Edmonton and Montreal

England

Jack Chapman
Managing Director, Tandem Computers Limited
187 High Street
Uxbridge, Middlesex UB8 1LA

France

Claude Raimond
Managing Director, Tandem Computers S.A.
2 Rue Nicolas Ledoux
Silic 255
94568 Rungis Cedex

Germany

Horst Enzelmueller, Vice President
Managing Director, Tandem Computers GmbH
Bernerstrasse 34
6000 Frankfurt/Main 56
Offices also located in
Dusseldorf, Hamburg, Munich
and Stuttgart

Sweden

Bengt Rindegard
Managing Director, Tandem Computers AB
Industrivagen 20 II
S-171 48 Solna

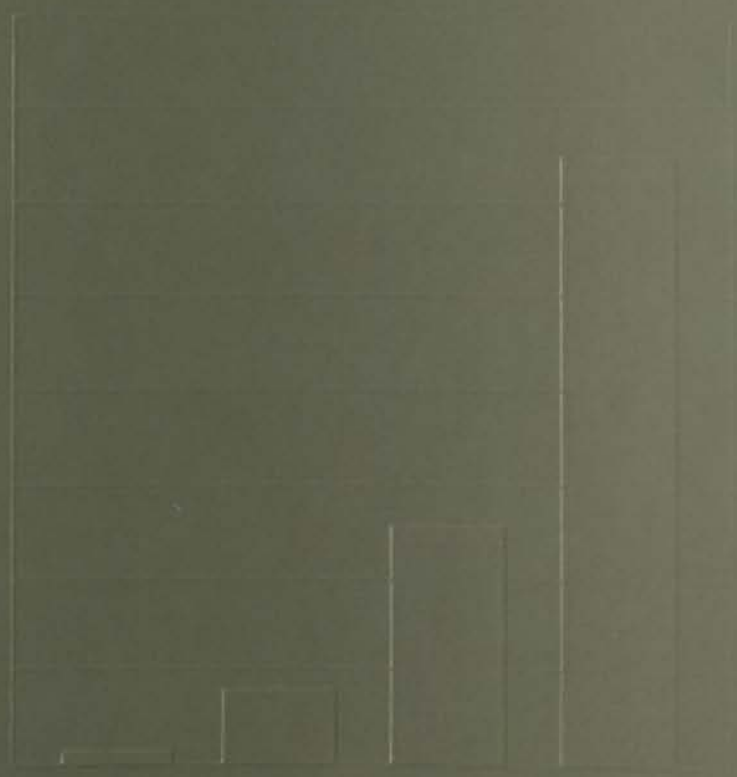
Switzerland

Heinz Studiger
Managing Director, Tandem Computers AG
Zweierstrasse 138
8003 Zurich

Tandem, NonStop, Guardian, Expand, Enform,
Enscribe and Pathway are trademarks and
service marks of Tandem Computers Incorporated.

TANDEM

annual report 1979



business review

System Detail

CONFIDENTIAL

General

Execution Environment

To best be able to understand the capabilities of the system and understand how "NonStop" works in a Tandem 16 System, a brief overview of the basic hardware/software components will be discussed first. Once the areas of responsibility have been defined, then we can address specific questions by relating to the area(s) responsible for that concern. The basic hardware structure of a Tandem 16 System is comprised of a network of 2 to 16 processor modules interconnected by a high-speed data bus. Each processor module contains:

- . A high-speed data bus control unit which can transfer data from one processor module to another at 13 mega bytes (Dynabus).
- . A central processing unit with 32 bit micro instructions and 100 nanosecond cycle time designed to execute 16 bit machine instruction.
- . Memory.
- . I/O channel control unit which can address up to 32 controller address with up to 8 devices per controller. The channel gives the controllers attached a direct path to memory at 4 megabytes.

All of the controllers attached to the I/O channel are Tandem manufactured and are dual ported (i.e., they are attached to the I/O channels of two processor modules). The ports of each controller are programmatically controlled, and only one is active at a time. In this environment the devices attached to a controller are accessed by one of a processor module pair. This allows the system to establish backup responsibility for device access. It also defines that up to 256 devices (e.g., 32 controllers x 8 devices) are accessible via a processor module pair. Any one processor module or both may have primary responsibility for the devices attached to the shared controllers.

The devices attached to the controllers are such peripherals as terminals, printers, mag tape, and disc drives.

The disc drives are manufactured as dual ported like the controllers (not by Tandem). This allows the disc drives to be attached to two controllers which are being shared between a processor module pair. This now gives two paths to the device on the same I/O channel as well as two backup paths to the device on the other I/O channel.

Using this architecture, we have established a means of interconnecting processor modules which, in some cases, share responsibility for devices attached to a common controller.

The next step is to establish within each processor module, an operating system capable of communicating between processor modules over the Dynabus and with a set of driver processes to drive the devices attached to the controllers on the I/O channel of that processor module. These operations and many more are accomplished with the implementation of Tandem's Guardian Operating System. The basic components of the Guardian Operating System are:

- . Kernel
- . I/O Drivers (with associated protocol handlers)
- . Message System
- . File System

Each of these areas will be further examined, and in doing so will define the basic capabilities of the Tandem 16 System.

Guardian Kernel:

The primary responsibilities of the Kernel of Guardian is the control of the processes which run within a processor module (a process is a program object module set up to execute in a processor module).

Process control responsibilities include:

- . Multiprocess control for up to 256 processes within a single processor module.
- . Priority scheduling of processes within a single processor module.

Processes can be assigned user defined priorities of value 1 to 255 for each process. As processes are ready to execute, the one with the highest priority goes first. Many processes may have the same

priority. In this situation, if more than one with the same priority is ready and that is the highest priority, they are processed in a round-robin fashion.

Memory management for ready processes.

The Tandem 16 machine uses a stack architecture. In the Tandem environment, a stack is a logically contiguous string of up to 64K words in length (128K bytes). A process is composed of two such stacks:

- . a non-modifiable code stack
- . a private data stack

For a total process size of 128K words (256K bytes).

The process, as it becomes ready for execution, does not get loaded totally into memory. Instead, a virtual memory management scheme has been implemented. This is accomplished by mapping the process stacks into two logical maps of 64 entries each. Thereby, each entry in the map represents 1K words (2K bytes) of logically contiguous code or data. The logical segments can then be loaded as required into locations of physical memory called pages (2K bytes in size) when they are required. This implies that no more than two pages of physical memory is actually necessary to execute a processor up to 256K bytes.

The object programs are stored and loaded from disc. Images of the code stack can be retrieved from here whenever the required page needs to be loaded. The data, on the other hand, is dynamic and an area on the disc must be set aside to store and retrieve any data generated by the process. This space is defined as the process' virtual memory.

The Tandem System philosophy is that there are no instructions to allow code modification; therefore, the code stack is shareable between multiple processes and is inherently re-entrant. That is to say that many processes started from the same object program actually use the same code map. But because the data each process may be working with may be different, each process has its own private data stack. As each process gets started, a physical page of memory for code and for data must be assigned to the logical pages to be worked with. If the process is very active, it may be that the previously used pages are still resident and have not been used by another process' execution; and then there is no requirement to get an image from the disc. If this is not the case, the operating system will pick a page to load code or data into. From all of the available pages, it will pick

one based on a "least recently used" algorithm. This minimizes the need for going to disc to get copies of virtual memory for subsequent processes' code or data pages. If the page picked is a code page, it is overlaid with the current process' code/data; but if the page holds data, it first checks to see if it had been modified after it had been loaded (via a "dirty bit" set in the page if modified when active). If not, it is overlaid; otherwise, it is written to the proper virtual space on disc before being overlaid.

There also exists in the system the option of forcing code/data to remain resident, therefore eliminating the need to go to disc once the page has been loaded into physical memory.

The operating system also maintains a set of code and data maps, as well as those set up for each user process.

The CPU in each processor module has four hardware registers to hold operating system maps (2) and user process maps (2). The operating system maps are only loaded once, at the time the processor module is loaded, but the user process maps must be loaded each time a process becomes ready for execution. Because there is only one set of registers for user process maps, it is impossible for one process to access another process' data. System data is also protected, but by convention, not physical limitations. To be able to access system data, the user process may execute privileged instructions. To be able to do this, the process must go through a licensing operation which is user controlled, therefore access to system data is not an inadvertent act.

Because of the fact that the system maps and user maps are resident at the same time, this allows for some shared resources within the system. This environment allows user processes to share operating system code because they can both be resident at the same time without having to load another set of maps. It is also possible to have user written code loaded with the operating system code so that it can be shared by all user processes, thereby making user code space larger than 64K words.

. Guardian File System, I/O Drivers and Message System

Within the Tandem 16 environment, it is the basic philosophy that a user process should not have to concern itself with the task of working with I/O on a physical level. The reason for this is many fold.

- . This allows Tandem to control all I/O so that we can optimize throughput on the system.
- . It removes the responsibility of the user to have to deal with the physical aspects of a device.
- . It allows the user to deal with I/O at a logical level and at the same time allows physical locations of devices to be transparent to the user.

To implement this philosophy, the File System of Guardian was designed.

The file system is the user interface for all I/O operations to be performed from the user process.

The types of resource available to the user is disc, tape, terminals, printers, and other processes. All of these are considered by the system to be "files." Files are considered by the system to have the same basic capabilities. That is the ability to be logically opened, closed, read and written. This gives these resources a degree of device independence. The physical location should also not be of concern to a user process. A process should have access ability to a system I/O resource no matter where it is in the system. To be able to implement this, it is necessary to define all devices to be attached to the system, the controller and unit on the controller of that device. Also, because the device is accessible via two processor modules, which one has primary responsibility for that device? Because the location of the device may change, a logical name is also associated with the device. This information is then maintained by the file system in each processor module so that it knows which processor has responsibility for a logical device. Through this vehicle of logical name association of physical device, the system maintains total geographic independence for user process access to any I/O resource.

A process is also considered an I/O resource, but just like devices to be accessible via the file system, it must have a logical name. This is accomplished by giving the process a name when it is loaded into a processor module. This name, like all the other devices, is then maintained by the file system in each processor module. Therefore, for two processes to communicate with each other, it is as simple as opening the process' logical name and doing reads and writes.

Files under the basic file system access are considered "unstructured." This defines an I/O resource with no specific format to the data processed. The record

sizes the system will handle in this mode is 0 to 4096 bytes on any file.

Most logical files within the system can be identified by assigning a logical name to a physical device, but the physical device for disc is the drive/pack. Therefore, a further qualification of a disc volume has been made. A fully qualified disc file is comprised of logical volume name.subvolume name.file name. Subvolume is a generic grouping of files resident on the same volume and the file name is a unique identifier within this grouping. On a logical volume, a directory is maintained for locating files on the volume. When a file is created (i.e., an entry made into a volume's directory), it may be defined as containing up to 16 parts or partitions, each residing on a separate disc volume. The maximum total size of a disc file is approximately 4 billion bytes. Files may be created by a system utility or programatically from a process. For other file types there is no associated maximum file size.

Although the file system is the interface to the user to access any system I/O resources, it does not actually perform the I/O itself. In the case of all I/O other than interprocess I/O, the actual interface to the resource is via a system process (I/O driver) designed to handle the unique characteristic of the resource.

Once the file system gets a request for a user process to do an I/O, it looks at a logical device table (LDT) where it locates where the device is physically and the name of the system process which is responsible for accessing it. The file system then routes a message to that process via a mechanism within Guardian called the Message System. The Message System delivers the I/O request to the required process which then performs the actual, physical I/O and returns a success or fail status to the file system, which then notifies the user process. In the case of inter-process I/O, the Message simply routes the information to the other user process. The information supplied to the Message System for message traffic is simply the processor number and the process number of the process with which it wants to communicate. It then sends the message over the interprocessor bus to the message system in the proper processor, which then routes it to the proper process. In some cases, it may actually send to itself. By use of this scheme, it makes it independent of which processor the drivers or named user processes reside. This allows user processes, in processors not in the same processor as the one with primary access to a disc, to access information from that disc.

Extensions to the Basic File System

The basic Guardian file system can be enhanced to allow special access capabilities in two areas.

Enscribe

Enscribe is an augmentation into the Guardian file system for disc file access. With this addition to the file system, Tandem supports structured disc file, as well as unstructured, and the access of these files.

Three file structures are supported:

- . Relative File - primary access by relative record number
- . Entry Sequence File - primary access by relative byte address
- . Key-Sequence File - primary access by unique key within record

Within all three structures, an alternate access is available to each record in a file by means of an alternate key file. Alternate key file(s) are a key sequence file which holds the alternate key field of each record and the primary key to access the records from the data file. Any file structure can have up to 255 alternate keys associated with the records in the file. This gives 256 total paths maximum of processing the data within the file.

Within a data path, records are accessed in ascending key sequence order starting at the first record based on one of three positioning modes.

. Approximate positioning:

Start at first record whose key is equal or greater than search key. Processing to end of file.

. Generic positioning:

Start at first record whose key is equal to search key. Processing to first record whose key is greater than search key.

. Exact positioning:

Search for record whose key matches the search key. Process that record.

Within the relative and keyed sequenced file structures, records may be added, deleted or updated. Entry sequence file will only allow add and update functions.

Record sizes for structured files are:

- . Relative and Entry Sequence - variable/fixed to 4096 bytes
- . Key Sequence - variable/fixed to 2048 bytes

Enscribe features also include record and file locking facilities and a cache buffer management. Cache is a means by which data blocks and index blocks (for key sequence files only) can be held in memory so that records can be accessed from memory without requiring disc access.

- . Envoy

Envoy is also an extension to the basic Guardian file system I/O access capabilities. Envoy is a data communications line handler. The basic file system will support only point-to-point asynchronous interfaces for terminals. The point-to-point connection may be local or switched direct dial. The Envoy line handler process will support both synchronous and asynchronous interface into the Tandem System in a point-to-point or multipoint environment.

- . System Availability

One of the primary objectives of the Tandem System is to establish a high degree of availability to all of its resources by maintaining a single fault-tolerant environment. That is, at least two paths to any resource within the system. Should the primary path to a resource be inaccessible, the system will automatically use an alternate path. This rerouting is totally transparent to the user processes running within the system.

Resources include such things as processor modules also. Therefore, a process running in a processor module which failed must have some facility for being started in another processor module without having to be reloaded. This capability is available through the facility of nonstop process pairs.

Processes run as process pairs establish a primary process that actively runs in one processor module and a backup process that maintains itself in a backup state in another processor module.

The primary process sends "checkpoints" to its backup process at critical points within its processing. A checkpoint is all or part of the primary's data stack and the current instruction location in the primary process. The backup process does nothing more than update his data stack with the checkpointed information. In the event of a failure of the primary process to be able to continue, for whatever reason, the backup process is notified and it then becomes fully active at the last checkpoint location of the primary process. This same technique is used by all the system I/O driver processes. The primary I/O process is located in the processor module which has the primary path to a device and the backup I/O process in the processor module which has the other path to the device. Should the primary path not be accessible (i.e., processor failure, channel failure, primary process failure, etc.), the backup process will take responsibility for access to the device. The Guardian Disc I/O processes always checkpoint each buffer to be written to the disc to its backup process. If the primary process is not successful at performing the I/O, the backup process will re-execute the operation to the device. This guarantees the integrity of the disc data base.

We can guarantee the integrity of the data base if the data base is accessible. If the disc was inaccessible for any reason, then the system availability is nearly zero. To account for this situation, Guardian supports a mirrored volume capability. Mirrored volumes define one logical volume as two physical volumes. This information is supplied to the system when you define all the devices and is maintained in the device table of the file system. When a logical write is issued to that logical unit, two physical writes occur, one to each disc volume. Should one device become disabled, the other will continue to be used for data base access.

Once the disabled disc is reusable, a system utility program can be run to execute a function called REVIVE. This process will bring the downed device up to the current status of the good disc while still processing against the live data. The operation goes on concurrently with all other data base access until the volume is fully restored.

If, in a system, a processor module was to fail, it would be necessary for backup responsibilities to be activated in other processor modules. The knowledge of the availability of all modules within a Tandem System must be monitored so that this backup can occur. This is accomplished by each

processor module maintaining a table of all processor modules within a system. At given intervals, each module sends out over the Dynabus an "I'm Alive" message to all other modules. Each module then updates its table. If a module fails to report within a given time, each module then checks itself to see if it can receive messages over the bus by sending a message to itself. If successful, it then assumes the non-reporting module is down and activates system I/O processes which have backup responsibility within this module, notifies any user backup processes so they can become active, and also notifies user processes that were communicating with processes in the downed module. Finally, it will update its device tables to find the backup processor modules for all devices that were being accessed by the downed module so that all routing of I/O request will go to the backup.

Any failure of resources within the system or error detection by the operating system is routed to a special Operator process. This process will then log these errors optionally at a console (any console) in a disc file on the system disc and/or to user written process.

. Process Creation

There are two means by which a process may be initiated into a system. The first of these is interactively through the use of a Tandem-supplied system interface program called the Command Interpreter. The Command Interpreter has many functions, one of which is to initiate the loading of object program module from disc. This is done through the use of the RUN command. To this command, the user supplies the name of the object file on disc and optionally other parameters such as which CPU to initiate the process in, a logical name (if it is to be a system I/O resource) the priority that the process is to run at, and parameters which can be passed to the process as a startup message that it can read over the message system. The parameter string may include such things as what data file the program should open to process. The second means by which a process can be initiated is under user program control via an operating system call to a routine called NEWPROCESS. Along with the call, the user will supply the same parameter as he would interactively via the Command Interpreter.

. Backup and Restore Capabilities

The facility for backing up and restoring of disc files is done via the system utilities BACKUP and RESTORE which can be initiated using the Command Interpreter program. These utilities allow for backing

up and restoring at the logical level from disc to mag tape.

A user may backup/restore:

- . a complete system (i.e., all logical volumes)
- . a single or multiple volumes
- . a single or multiple subvolume from/to one or more volumes
- . a single file or multiple files
- . any combination of the above.

. Security System

Security as implemented on the Tandem System is simple, yet powerful. It is designed in such a way so that users who require security can utilize all its features and those who do not are not hampered by it.

To be able to have access to the system, an individual must become a "user." A user is defined as someone who has been added to a user ID file maintained on the system disc. There are four classes of users:

. Standard User

A "standard" user is allowed to perform standard operations such as creating and purging disc files, running programs, displaying system status, etc. Additionally, a standard user is limited as to the processes it can stop or debug.

. Group Manager

A "group manager" user is permitted to perform the standard operations as well as designate new system users.

. System Operator

A "system operator" user is permitted to perform the standard operations as well as reload processor modules, set the system time-of-day clock, and alter the operating state of the interprocessor buses.

. Superid

The "superid" user has total freedom to perform any operation in the system. This includes debugging privileged programs, accessing any file, logging on as any user without knowing the user's password, adding new groups to the security system, running privileged programs which have not been "licensed."

A user is added to the system by either a group manager or the superid to belong to a user group and given a user name within that group. Along with the group name and user name is also associated a group number and user number within the group.

The user group name and user name is only used to gain access to the system via the Command Interpreter LOGON command. Once logged onto the system, the user group number and user number are used for security purposes. Users can protect others from using their logon by associating a password with the logon ID. To further aid in system security, the Command Interpreter has a hook in it to be able to pass all LOGONS to a process called CMON if it is running in the system. CMON can cancel the LOGON if so desired by passing a message back to the Command Interpreter.

Each file on the disc when created, either by a system utility or programmatically, has associated with it an owner. The owner is defined by group number and user number. Access to a file is controlled by four different criteria. These are read, write, purge and execute (this is for object program files only). At each of the four criteria, a security check can be imposed based on the following:

- . Any user may perform this operation to the file.
- . Only members within the same group as the owner may perform this operation to the file.
- . Only the owner of the file may perform this operation to the file.
- . Only the Superid may perform this operation to the file.

When a user process is initiated on the system, the ID that it assumes is the ID of the logged-on user. This ID is then used for security checks when access against a file is attempted. As a further security check, a user process can check which terminal the process was initiated from via an operating system call, and if not correct, cancel itself. This could allow only those persons who have access to a secured terminal the ability to run special programs.

. Networking Software

The addition of a networking capability to the Tandem System is accomplished by the installation of the GUARDIAN/EXPAND operating system. Expand is actually a subsystem extension to the message system within Guardian. The routing of messages between a network of processor modules was expanded to route these messages

out of the system over communications lines to other systems within the multiple system network. Each system within the network is defined as a node. The Expand subsystem will allow communication with up to 255 nodes within the network.

The implementation of Guardian/Expand will allow users of the Tandem System:

- NonStop Nodes

The fault tolerant hardware and software of the Tandem 16 System eliminates the computer as a source of network failures.

Note: Individual Tandem 16 systems within the network are called "nodes" to distinguish the computer system from the network system.

- Distributed System

The Guardian/Expand Network makes it possible to configure a network of Tandem 16 fault tolerant computer systems so that a user of any node in the network can access the resources of any other node (processors, files or physical devices) without regard for the physical location of the resource. To the user, the Guardian/Expand Network appears to be one large set of computer resources rather than a collection of separate systems.

- Dynamic Message Routing

The Guardian/Expand Network constantly monitors the communications paths. When a transmission fails, the system retries until the transmission succeeds or until the system determines that the communication path has been broken. When the communication path has been broken, the Guardian/Expand system automatically reroutes the message via a different communications path.

- Best Path Message Routing

The Guardian/Expand system monitors the communication lines and automatically selects the best path. The best path is the one that takes the least time. The system selects the fastest rather than the shortest path because this optimum end-to-end protocol can reduce communications costs. When a communications line fails, Guardian/Expand reroutes messages using the next fastest available path. When a new line is added, the system takes advantage of any new best paths created by the addition.

- Precisely Tailored Hardware

Different nodes within a network typically have different computing requirements. The Tandem 16 System allows users to place exactly the right amount of computing power at each site. Even though the nodes within the network may range from a basic two processor system to a sixteen processor system with billions of bytes of online disc storage, the systems retain total compatibility of data, software, and application programs.

- Logical Growth

The Tandem 16 architecture allows for incremental hardware expansion. A user can add memory, central processors, or peripheral devices as computing requirements grow. Similarly, the Guardian/Expand Network allows incremental expansion. Nodes can be added or removed and communication paths can be changed, all without reconfiguring existing systems.

Notice that the Guardian/Expand Network can forestall the need for hardware expansion since the resources of every system in the network are accessible.

- Data Integrity

The Guardian/Expand Network incorporates multiple safeguards to ensure that message packets are received correctly and that data cannot be lost in transmission.

- Human Engineering

Because the Expand Network is an extension of the Guardian Operating System, the user interface to the network is through the Guardian command interpreter. For example, to run the text editor program on the local system, the user enters the command EDIT at his terminal. To run the program on a remote system, the user simply enters the symbolic name of the remote system before the command: \OHIO.EDIT. In effect, this command connects the user terminal with the remote OHIO system and starts the text editor program on that system. The only difference the user may notice in running on the remote system rather than the local system is that response time may be slightly longer since the communication lines cannot match the performance of the local processor.

- Simple Programming Interface

The Expand Network relieves programmers of the need to deal with a cumbersome telecommunication access method. Since programs communicate with

each other via Guardian's message system, the programmer uses the same commands to communicate with a program in the same processor, another processor, or another system.

Within each node of the Tandem network, the Expand subsystem controls the networking control. This is accomplished by basically two different processes.

The Network Control Process maintains information about the network; i.e., which is the best path from one node to another, statistics, etc. The best path information is based on best time from one node to another. When the network is first brought up, Control processes (one in each node) talk to each other over the network to give information about what paths are available. Using this information and knowing the line speeds on each of the paths, it determines the best path. This information is stored in a Network Routing Table (NRT) which changes only if a path goes down and a new route must be determined. The links between nodes in the system are full duplex synchronous lines up to 56K bps. These lines may be leased or dial lines point-to-point (no auto dial). Between two nodes a dial backup capability may be established. Only one path between nodes is used even if multiple lines are available. If a Tandem network is established as a star or ring, there will always be an alternate path to a node. In this environment, a user can establish a nonstop network of nodes as the Tandem system can maintain between processors. To communicate between nodes in the network, the second network process, the Network Line Handler (NLH) is used. There is one NLH per network line out of a node.

The NLH communicates between nodes with packets of information using an X.25 like protocol to guarantee packet integrity. All packets will arrive at the destination node and be built back into a message using a Tandem end-to-end protocol which guarantees message integrity from the source node to the destination node.

In addition to the NLH process(es) between Tandem nodes there is an X.25 Line Handler process available to route messages to terminals which may be linked into a common carrier X.25 packet switch service. This would then allow these users full network access just as local or remote (over asynchronous point-to-point) terminals currently have.

The concept of central control in the Tandem network does not exist; however, it is possible that some users may want to designate one of the systems as a control center for monitoring the network. To provide this

capability, a Network Control Center program will be provided. The function of Network Control Center will provide for:

- . Centralized logging of:
 - . Changes in network status (line ready, not ready, connections ... etc.)
 - . Changes in CPU status at remote systems.
- . Centralized display of:
 - . All systems' CPU status.
 - . The time and distance between all systems.
 - . Individual systems' network maps.
 - . Individual line handler statistics.
 - . Probes from selected systems.

The Network Control Center will consist of:

- . An ADM-2 terminal
- . The NCC program
- . The network utilities at each system
- . An NCC/NCP interface

The NCC will not address the measurement of network performance since this will be handled by future XRAY enhancements.

. System Monitoring

System monitoring capabilities exist within a single Tandem System through the use of a utility package called XRAY.

With the XRAY package, a user can be provided information for doing:

- . Load Balancing
- . Growth Management
- . On-Line Monitoring
- . Application Tuning

XRAY will provide data for resolving performance and usage of:

- . Processors
- . Data Communication Lines
- . Tapes, Printers, etc.
- . Terminals
- . Disc and Disc File Opens
- . Processes
- . File Opens

The use of XRAY does not require any special hardware or display devices.

The XRAY programming language is a high level, Pascal-like block structured language designed for the easy implementation of transaction-oriented applications. XRAY is similar to Pascal, PL/I or Cobol in providing standard blocks and high level constructs such as IF, WHILE, REPEAT, GO, RETURN, etc. The programmer can write self-documenting programs with object generation and debugging facilities.

The XRAY programming language provides a set of standard subroutines for file handling, data base access, and other functions. These subroutines are designed to be used in a variety of applications. The language also provides a set of standard procedures for file handling, data base access, and other functions. These procedures are designed to be used in a variety of applications.

The XRAY programming language provides a set of standard subroutines for file handling, data base access, and other functions. These subroutines are designed to be used in a variety of applications.

All programs written in XRAY are portable, and, therefore, can be executed on a variety of computers. This portability is achieved by using standard blocks and procedures, thereby minimizing the amount of code required for different applications.

The XRAY programming language provides a set of standard subroutines for arithmetic, string and logical operations. These subroutines are designed to be used in a variety of applications. The language also provides a set of standard procedures for arithmetic, string and logical operations. These procedures are designed to be used in a variety of applications.

The XRAY programming language provides a set of standard subroutines for arithmetic, string and logical operations. These subroutines are designed to be used in a variety of applications.

The XRAY programming language provides a set of standard subroutines for arithmetic, string and logical operations. These subroutines are designed to be used in a variety of applications.

The XRAY programming language provides a set of standard subroutines for arithmetic, string and logical operations. These subroutines are designed to be used in a variety of applications.

Program Development Facilities

. TAL (Tandem Application Language)

The Tandem Application Language is a high level, Pascal like, block structured language designed for the easy implementation of transaction-oriented applications. T/TAL is similar to ALGOL, PL/1 or COBOL in providing procedure blocks and high level constructs such as IF...THEN...ELSE, DO...UNTIL, WHILE...DO, FOR and CASE. Programmers can write self documenting programs with object code generation as efficient as assembler code.

The procedure-oriented, block-structured approach allows a programmer to design procedure blocks for functional simplicity. Data can be assigned globally for all procedures to access or can be assigned locally - known only to the procedure in which it is declared. Local data is assigned dynamically and initialized each time the procedure is invoked. The local data feature also allows the use of recursive procedures.

One procedure can call another and pass parameters by value or by reference. The called procedure can optionally return a value through its name.

All procedure code is compiled as re-entrant, and, therefore, non-modifiable. This permits code to be shared by many different users, thereby minimizing the amount of memory required for multiple applications.

T/TAL provides data types for arithmetic, string and logical operations. Signed integer data can be specified with 15 bits or 31 bits of precision. Fixed data is a scaled decimal with up to 18 digits of precision. String data consists of one or more 8-bit bytes. Logical data consists of 16-bit words.

Data can be declared as simple, single element items or as arrays (multiple elements). In addition, a variable can be declared as a pointer variable to facilitate indirect addressing.

Type functions are an integral part of TAL and provide the ability to convert from one data type to another. Type functions are also available to determine if string data represents ASCII NUMERIC, ALPHA or SPECIAL values. Special string manipulation constructs are available to facilitate efficient processing of transaction data. Included are SCAN, MOVE and COMPARE string operations.

Data can be addressed at the bit level, if desirable. Operations exist in TAL to perform bit deposit, bit extraction, arithmetic shifting and logical shifting.

All I/O is performed via calls to operating system procedures. Procedures exist to open and close files, read and write data, perform control functions, create and purge files, lock and unlock files, and perform "wait" or "nowait" I/O. Extensive error checking is provided by the File System and error codes are returned to the calling procedure for inspection. All of the Tandem operating system is written in TAL.

ANSI COBOL X3.23 - 1974

Tandem's ANSI Standard COBOL -74 utilizes all the capabilities of Tandem's Guardian Operating System and Enscribe Data Base Record Manager - software designed to keep the system up and running while maintaining the integrity of the on-line data base. Thus Tandem/COBOL is compatible with programs written in T/TAL, Tandem's high level language for transaction processing, and is capable of running in a multi-language environment.

Tandem COBOL Extensions

Several extensions have been added to Tandem/COBOL to permit use of Tandem's Guardian Operating System.

NonStop Extensions

For NonStop programming in COBOL, you include a "?NONSTOP" line in your source program. STARTBACKUP and CHECKPOINT are the verbs that make the program nonstop. Normally STARTBACKUP is called once at the beginning of the program to set the nonstop mode. Thereafter the CHECKPOINT verb is used to pass information to the backup process at critical points in the processing. In a nonstop program, checkpoints will also occur automatically upon any OPEN or CLOSE executed after the backup is established. Both of these verbs will set the special register, PROGRAM-STATUS, to indicate the outcome of the checkpointing operation.

Extensions to the standard COBOL I/O facility

Three new verbs, LOCKFILE, UNLOCKFILE, and UNLOCKRECORD, have been introduced to allow the use of the corresponding system file and record locking routines. This addition allows separate processes to share a common data base. READ and REWRITE verbs have been extended to allow the specification of a LOCK or UNLOCK operation. The OPEN syntax has been extended to specify the file access, EXCLUSIVE, SHARED or PROTECTED, and to permit the SYNCDEPTH for files opened in the OUTPUT, I-O or EXTEND mode.

- . Calling TAL Procedures

The ENTER verb has been modified to call TAL procedures. This lets you access any TAL system external or your object library routines. These object library routines are assigned to the COBOL run unit at compile time.

- . Interprocess Communications

COBOL processes can communicate with one another or with processes written in other languages through the standard READ and WRITE statements. Communication to other processes is implemented using the interprocess communication facilities of the Guardian Operating System.

- . ANSI FORTRAN X3.9 - 1978

Tandem FORTRAN runs on the Tandem T16 Computer System - the only multi-processor architecture designed for NonStop, transaction-oriented, data base applications. Tandem FORTRAN utilizes all the facilities of the Guardian Operating System including NonStop operation, re-entrant code, interprocess communications, virtual memory, and Enscribe data base facilities for keyed, relative, and sequential access, multi-key data paths, and concurrent record access. Tandem FORTRAN is capable of running in a multi-language environment.

Tandem FORTRAN Extensions

Several extensions have been made to FORTRAN 77 to enhance its operation on Tandem systems.

- . NonStop Extensions

STARTBACKUP and CHECKPOINT functions allow a FORTRAN program to utilize the NonStop capabilities of Guardian. STARTBACKUP is called once at the beginning of a program to establish the nonstop mode. Thereafter CHECKPOINT is used to pass critical information to the backup process. Checkpoints will automatically occur upon any OPEN or CLOSE after the backup has been created.

- . Structures

Structures provide the ability to define records in FORTRAN. The constructs RECORD and END RECORD are used to define record structures. The Data Definition Language may also be used to transcribe a schema into FORTRAN RECORD structures.

- . ENSCRIBE Extensions

Extensions have been made to the FORTRAN READ and WRITE statements to permit the full use of ENSCRIBE facilities. Thus it is possible with FORTRAN statements to access key-sequenced, relative, and entry-sequenced files by primary or up to 255 alternate keys. Provision has been made to allow exact, approximate or generic positioning into an ENSCRIBE file structure using FORTRAN. Concurrent record access is supported with LOCK mechanisms at either the record or file level.

- . Interprocess Communications

FORTRAN processes can communicate with one another or with processes written in other languages through the standard FORTRAN READ and WRITE statements. Communication to other processes is implemented using the interprocess communication facilities of the Guardian Operating System.

- . Update

Tandem's Update Program is used to combine two or more object files and build them into one new object file. This allows users to have "libraries" of procedures which can be used by several programs (this can reduce source program size and compilation time).

- . Text Editor

The Tandem 16 Text Editor program - EDIT - is used for entering source text into the computer system and for adding to and modifying existing source text. The source text can be a T/TAL source language program, a SYSGEN configuration file - virtually any textual material desired.

The Editor is designed to be used interactively. Editor commands and text are typed into an online terminal. When a command is typed in, the appropriate action is taken by the Editor. Text is entered into the system by typing in an ADD command, followed by the text lines.

In a normal Editor command execution cycle, the user is prompted for a command on the terminal; the user enters a command; the Editor executes the command; then the cycle repeats.

- . Debug

The Guardian Debug facility provides a tool for interactively debugging a running process. Using the debug facility, a programmer can designate certain program code locations (called breakpoints) that, when executed,

cause the process to enter the debug state. While in the debug state, the programmer can interactively display and modify the contents of a program's variables, display and modify the contents of process' registers, and set other breakpoints.

The programmer specifies a location in the code area that, when executed, causes the process to enter the debug state. The operating system replaces the instruction in the specified location with a call to the DEBUG procedure. Then when the processor fetches and executes the contents of that location, the DEBUG procedure is invoked. Before leaving the DEBUG state, the operating system puts the replaced instruction back into its original location and adjusts the program counter so that it is executed. (In the next location after the breakpoint, the operating system puts another call to DEBUG. This is invisible to the user of DEBUG. Its purpose is to put a call to DEBUG back into the breakpoint location.)

When the same program file is run in the same processor module, the code area is shared. Therefore, if a breakpoint is specified, the first process to execute the breakpoint location goes into the debug state (not necessarily the process that set the breakpoint).

Development Testing

In order to test new system under development, it will be necessary to have multiple copies of whatever I/O resources are used by a production system. If the processes used in the system are parameterized for their I/O requirement (i.e., all file names passed at process startup dynamically), all that is required to switch over development to production is to pass the production parameters. Both development and production can run concurrently within the same system.

The network can access the resources of any other node (processors, files, or physical devices) without regard for the physical location of the resources. To the user, the Guardian/EPAS network appears to be one large set of computer resources rather than a collection of separate systems.

Network Resource Monitoring

The Guardian/EPAS network constantly monitors the communications paths. When a transmission fails, the system retries until the transmission succeeds or until the system determines that the communication path has been broken. When the communication path has been broken, the Guardian/EPAS system automatically reroutes the message via a different communications path.

GUARDIAN/EXPAND NETWORK SUBSYSTEM

The EXPAND NonStop Network is unique because it is an extension of an existing network operating system. Every Tandem/16 Computer System comprises from 2 to 16 separate central processors. Running in the NonStop mode requires constant communication among the processors. Consequently, the Guardian Operating System includes a sophisticated message handling system to control communications between processors and between processes (programs) running in one processor or running in separate processors. In effect, every Tandem/16 System is a local network controlled by the Guardian Operating System. The EXPAND NonStop Network expands the scope of the Guardian Operating System to allow communications among as many as 255 Tandem/16 systems.

The Guardian/EXPAND Network system, combined with the unique architecture of the Tandem/16 Computer System, provides network users with a number of features unequalled by other computer vendors:

- NonStop Nodes

The fault-tolerant hardware and software of the Tandem/16 System eliminates the computer as a source of network failures.

Note: Individual Tandem/16 Systems within the network are called "nodes" to distinguish the computer system from the network system.

- A Distributed System

The Guardian/EXPAND Network makes it possible to configure a network of Tandem/16 fault-tolerant computer systems so that a user of any node in the network can access the resources of any other node (processors, files, or physical devices) without regard for the physical location of the resource. To the user, the Guardian/EXPAND Network appears to be one large set of computer resources rather than a collection of separate systems.

- Dynamic Message Routing

The Guardian/EXPAND Network constantly monitors the communications paths. When a transmission fails, the system retries until the transmission succeeds or until the system determines that the communication path has been broken. When the communication path has been broken, the Guardian/EXPAND system automatically reroutes the message via a different communications path.

- . Best Path Message Routing

The Guardian/EXPAND system monitors the communication lines and automatically selects the best path. The best path is the one that takes the least time. The system selects the fastest rather than the shortest path because this optimum end-to-end protocol can reduce communications costs. When a communications line fails, Guardian/EXPAND reroutes messages using the next fastest available path. When a new line is added, the system takes advantage of any new best paths created by the addition.

- . Precisely Tailored Hardware

Different nodes within a network typically have different computing requirements. The Tandem/16 System allows users to place exactly the right amount of computing power at each site. Even though the nodes within the network may range from a basic two processor system to a sixteen processor system with billions of bytes of online disc storage, the systems retain total compatibility of data, software, and application programs.

- . Logical Growth

The Tandem/16 architecture allows for incremental hardware expansion. A user can add memory, central processors, or peripheral devices as computing requirements grow. Similarly, the Guardian/EXPAND Network allows incremental expansion. Nodes can be added or removed and communication paths can be changed, all without reconfiguring existing systems.

Notice that the Guardian/EXPAND Network can forestall the need for hardware expansion since the resources of every system in the network are accessible.

- . Data Integrity

The Guardian/EXPAND Network incorporates multiple safeguards to ensure that message packets are received correctly and that data cannot be lost in transmission.

- . Human Engineering

Because the EXPAND Network is an extension of the Guardian Operating System, the user interface to the network is through the Guardian Command Interpreter. For example, to run the text editor program on the local system, the user enters the command EDIT at his terminal. To run the program on a remote system,

the user simply enters the symbolic name of the remote system before the command: \OHIO EDIT. In effect, this command connects the user terminal with the remote OHIO system and starts the next editor program on that system. The only difference the user may notice in running on the remote system rather than the local system is that response time may be slightly longer since the communication lines cannot match the performance of the local processor.

. Simple Programming Interface

The EXPAND Network relieves programmers of the need to deal with a cumbersome telecommunication access method. Since programs communicate with each other via Guardian's message system, the programmer uses the same commands to communicate with a program in the same processor, another processor, or another system.

Full appreciation of the EXPAND Network System requires an understanding of how tightly the EXPAND Network System is integrated with the Guardian Operating System and the Tandem/16's architecture.

THE TANDEM/16 SYSTEM: A NETWORK IN A BOX

The Tandem/16 NonStop System is designed for the on-line, terminal and transaction, data base-oriented market where high availability is required.

This requires a multiple processor system that can tolerate the failure of any single component, including even a central processor unit. Figure 1 illustrates the architecture of a typical Tandem/16 System.

Notice that at least two paths connect any two components in the system, and everything in the system, including even individual disk files, can be duplicated. Normally, all resources in the system function as independent, parallel resources. However, when a component fails, the remaining system components automatically take over the workload of the failed component. A system user at a terminal is typically unaware of the failure.

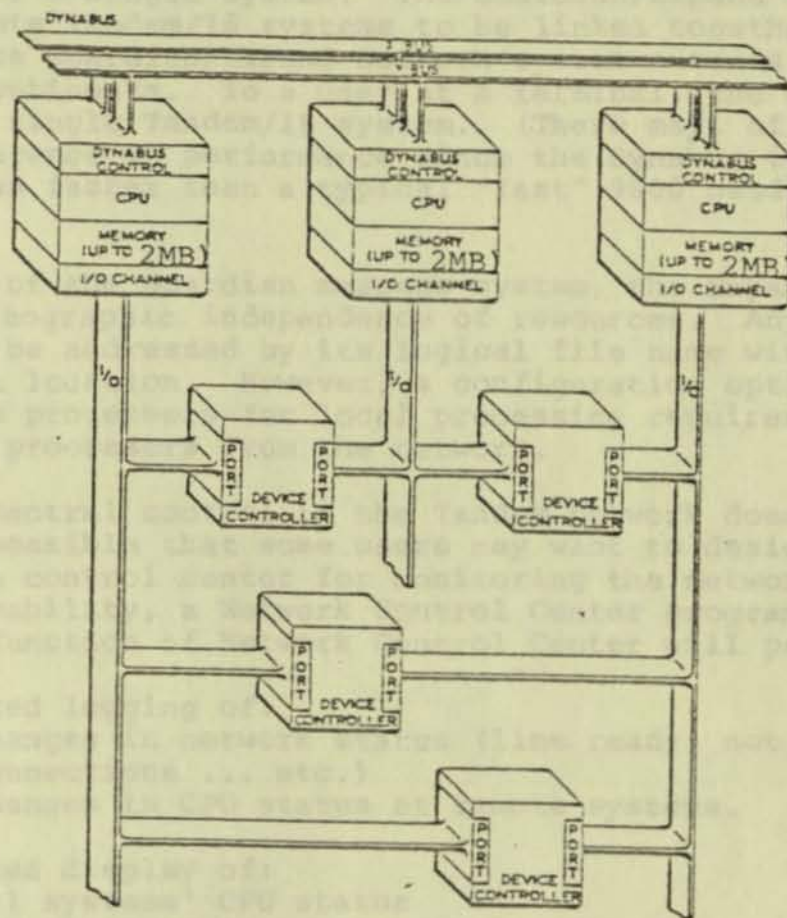


Figure 1. Typical Tandem/16 System

Notice that at least two paths connect any two components in the system, and everything in the system, including even individual disc files, can be duplicated. Normally, all resources in the system function as independent, parallel resources. However, when a component fails, the remaining system components automatically take over the workload of the failed component. A system user at a terminal is typically unaware of the failure.

The Expand Network system extends the Guardian message system beyond the boundaries of a single system. The Guardian/Expand Network allows up to 255 separate Tandem/16 systems to be linked together. Conceptually, the Guardian/Expand Network system extends the Dynabus over miles or continents. To a user at a terminal, the entire network appears to be a single Tandem/16 system. (There may, of course, be a noticeable difference in performance since the Dynabus itself is more than 10,000 times faster than a typical "fast" 9600 baud private line.)

As an extension of the Guardian message system, the Expand Network maintains the geographic independence of resources. Any resource in the network can be addressed by its logical file name without regard for its physical location. However, a configuration option allows users to reserve processors for local processing requirements, thereby excluding those processors from the network.

The concept of central control in the Tandem network does not exist; however, it is possible that some users may want to designate one of the systems as a control center for monitoring the network. To provide this capability, a Network Control Center program will be provided. The function of Network Control Center will provide for

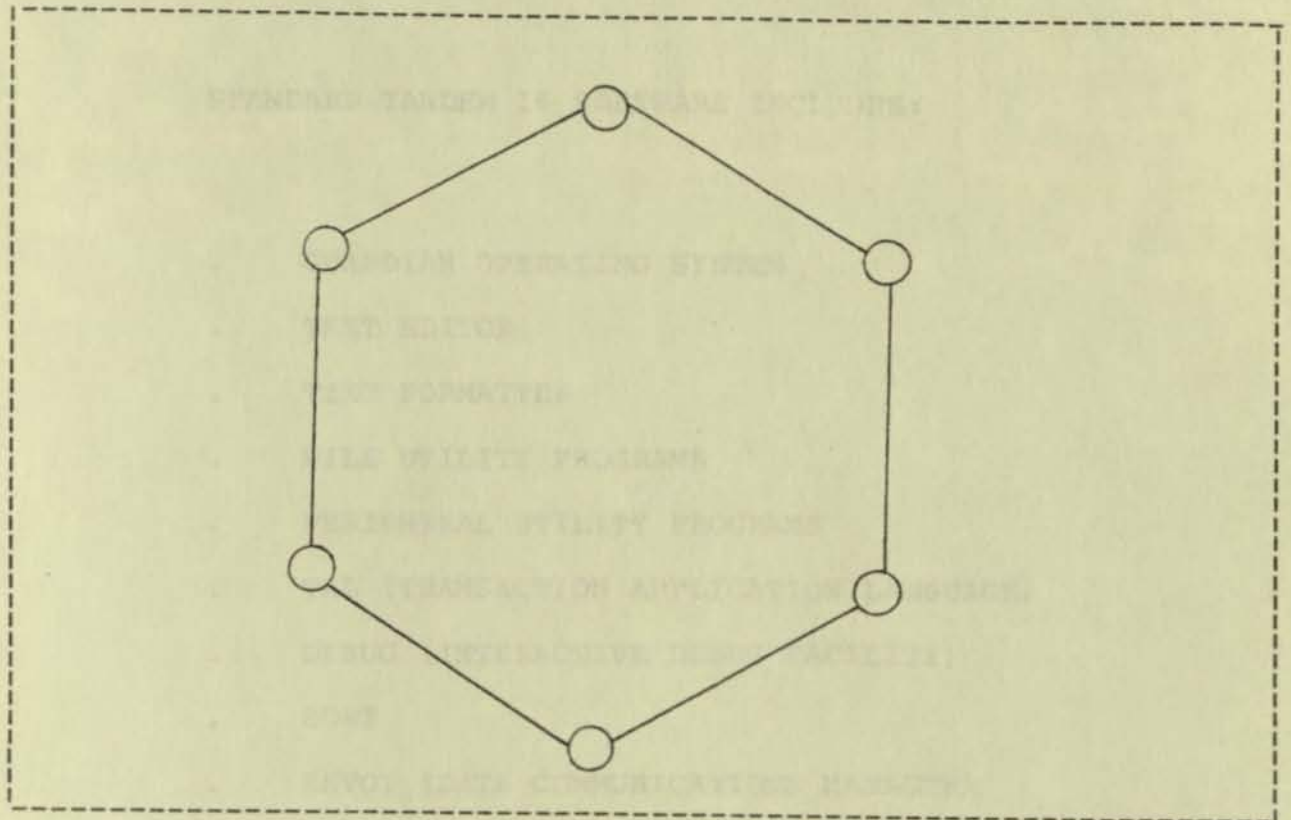
1. Centralized logging of:
 - a. changes in network status (line ready, not ready, connections ... etc.)
 - b. changes in CPU status at remote systems.
2. Centralized display of:
 - a. all systems' CPU status
 - b. the time and distance between all systems
 - c. individual systems' network maps
 - d. individual line handler statistics
 - e. probes from selected systems.

The Network Control Center will consist of:

1. An ADM-2 terminal
2. The NCC Program
3. The Network Utilities at each system
4. An NCC/NCP interface.

The NCC will not address the measurement of network performance since this will be handled by future XRAY enhancements.

Figure 6. A Typical Guardian/Expand Network



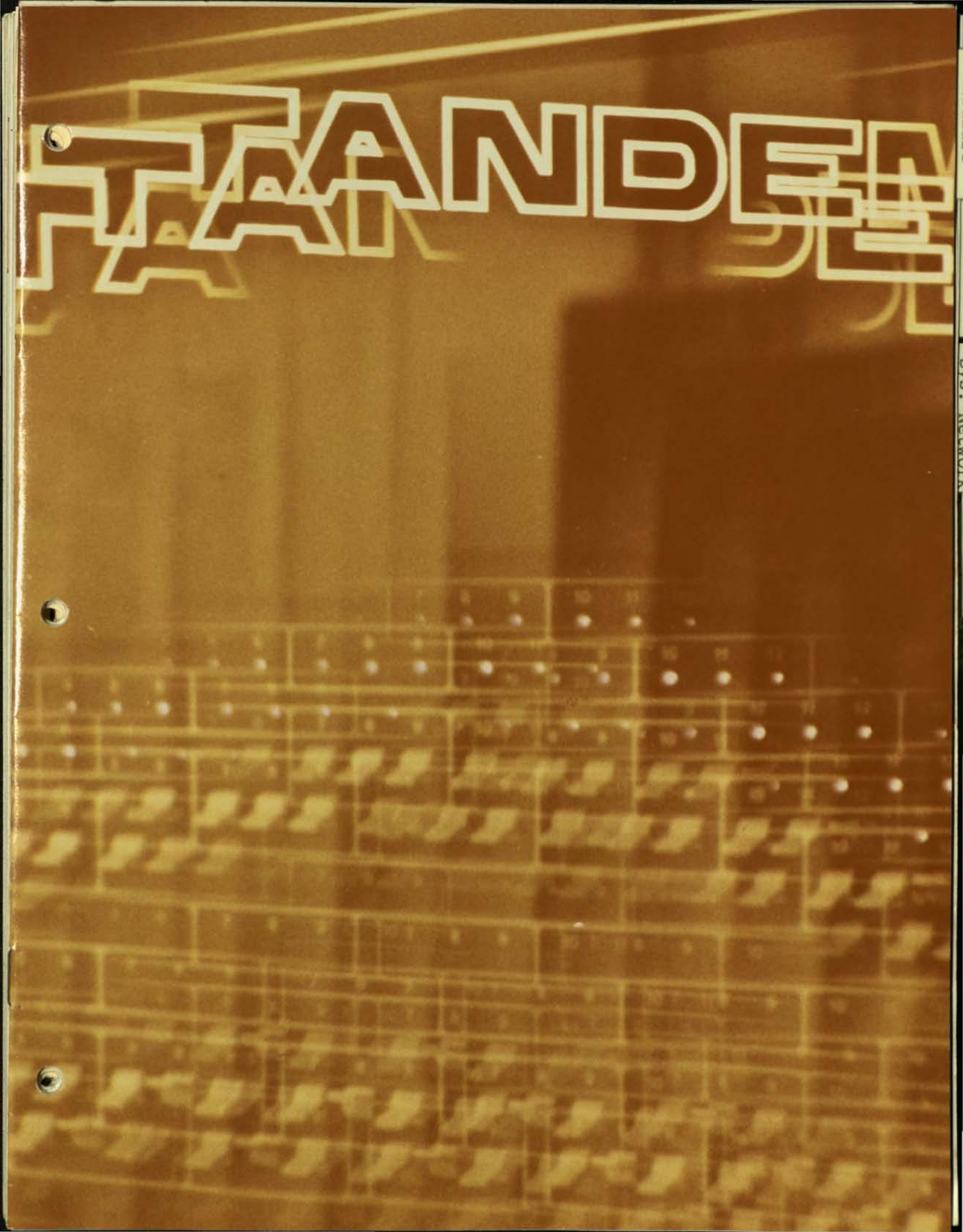
The Guardian/Expand Network reduces communications costs because its automatic message forwarding eliminates the need for point-to-point interconnection of the network. Most other commercially available networks require the user to write special applications programs to relay messages from node to node.

The Guardian Operating System is perhaps the only operating system designed as a network system. The multi-processor design of the Tandem 16 system necessarily requires an operating system structured as a local network-in-a-box. In effect, the Expand NonStop Network simply eliminates the box.

STANDARD TANDEM 16 SOFTWARE INCLUDES:

- . GUARDIAN OPERATING SYSTEM
- . TEXT EDITOR
- . TEXT FORMATTER
- . FILE UTILITY PROGRAMS
- . PERIPHERAL UTILITY PROGRAMS
- . TAL (TRANSACTION APPLICATION LANGUAGE)
- . DEBUG (INTERACTIVE DEBUG FACILITY)
- . SORT
- . ENVOY (DATA COMMUNICATIONS MANAGER)
- . DDL (DATA DEFINITION LANGUAGE)
- . COMMAND INTERPRETER
- . FILE MANAGEMENT SYSTEM
- . FILE SECURITY SYSTEM

FRANDEA



- 13:35 Processor 1 fails, messages are printed from both processor 0 and processor 2 noting this fact. Effect on system users — none.
- 15:20 Parity error during a disc read, retry was successful. Effect on system users — none.
- 17:30 Processor 1 repaired and re-incorporated into the system. Effect on system users — none.
- 19:10 Complete system power-interrupt and restart. Effect on system users — none.

```
20 13:35 24FEB77 FROM 00,000 CPU 01 IS DOWN
20 13:35 24FEB77 FROM 02,000 CPU 01 IS DOWN
05 15:20 24FEB77 FROM 00,005 LDEV 0001 RETRY X002000 X00001 X116000
21 17:30 24FEB77 FROM 01,000 PROCESSOR UP
20 19:10 24FEB77 FROM 01,000 $POWER ON$
20 19:10 24FEB77 FROM 00,000 $POWER ON$
20 19:10 24FEB77 FROM 02,000 $POWER ON$
█
```



TANDEM NonStop™ Systems

The age of on-line computers.

It's begun in earnest now and the reasons are sound. Dedicated, efficient and low-in-cost, today's on-line computers provide immediate, positive response in all types of applications.

Credit verification. Bank deposits and withdrawals. Funds transfers. Order processing and inventory control. Medical systems. Retail sales. Emergency services. Theater and sports events ticketing. Hotel and motel reservations. Communications networks. Manufacturing and materials control systems. Stock and commodity transactions. Off track betting systems and state lotteries. The list of potential applications is enormous. Wherever there's a high volume of transactions requiring speed and accuracy. Wherever there's a critical need for immediacy of information coupled with reliability and system expandability. Wherever efficient, low cost transaction processing is essential. There, Tandem's unique multiple processor architecture and multiple processor software offer a new and powerful solution. And it's proven in the field.

On-line means on-demand.

And there's the rub. For as reliable as the modern computer is, and it is remarkably reliable, it will on occasion fail. Which can cause lost business, or missed schedules, or unhappy customers, or costly errors, or worse. In live data base systems, a crash during processing can cause untold damage to, or even destruction of, the data base. And if that's a nightmare, consider how even worse a problem is creeping degradation of the data base. Where the failures aren't apparent except to your customers. That is totally unacceptable. And that is why Tandem came into business.

The search for alternatives.

Until now, the only way on-line computer users could protect themselves against the possibility of a computer failure was to install a "back-up" system, typically a

second processor strapped into the system, ready to come on line when a failure occurred. It required special interfaces and customized software; and there was the penalty of system inflexibility, loss of processing power, and uncertainty as to the status of transactions-in-process when a failure occurred. But aside from manual back-up systems, which went rapidly out-of-date, and were slow, inefficient and uncertain, it's all there was. Until Tandem.

NonStop Computing.

Tandem has designed and built the first multiple processor system designed from scratch to provide non-stop processing—even during a failure—with no penalties in the speed, capacity, throughput or memory utilization of the system. With no strapped-up interfaces and no customized software, and perhaps most important, no loss of system flexibility. The Tandem NonStop System puts a whole new meaning into the concept of expandability. Each individual system can expand from the basic two-processor system all the way to sixteen processors, without reprogramming, without customizing, and without one cent of penalty on the original investment. Even more, each individual system, whether minimal or fully expanded, can be treated as a distinct node in an overall system network with up to 255 nodes. File capacities are four billion bytes per file, and there is no limit on the number of files. A fully expanded network could support more than a million terminals interconnected/located around the world. And while not likely, that system could have begun with a simple two-processor system. And still be using the same software.

Data Base Integrity.

Probably no one aspect of an on-line system causes more concern, and rightly so. No one in the entire data processing industry offers anywhere near the protec-



tion offered by Tandem's NonStop System. And that protection extends not only against catastrophic failures, but also against the far more insidious creeping failure, the kind which doesn't show up as a stoppage in the computer room, but only as mis-billed customers, false inputs on a medical chart, funds transferred to a wrong account, a major purchase of the wrong stock, inventory shipped to a wrong location...who knows how much damage before detection.

Because of its unique transaction handling procedures, no transaction is ever lost in process nor is it ever duplicated. The integral redundancy of the Tandem system provides an unprecedented level of protection, not only at the transaction level, but at all levels of the data base. In almost every on-line application, this protection is significant. In some, it is essential. And again, there's only Tandem.

The three keys.

Non-stop operation. Data base integrity. Modular expansion. All in Tandem. And only from Tandem. The one and only answer to the opportunities and problems of today's computer applications: Data Base Systems. Distributed Systems. Data Communications Systems. Multi-terminal Systems. All of these have grown out of the enormous capabilities of the computer itself, but as of today, only one company has dedicated itself to the design and construction of efficient, low cost systems which meet their needs head-on. Tandem Computers. Up and running. NonStop.

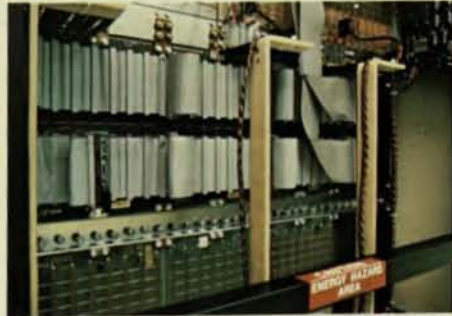
Tandem two processor system, expandable to four in same cabinetry, expandable to sixteen processors with additional cabinetry.



The benefits of NonStop Computing.

- Hardware and software are designed to work together continually, productively and efficiently.
- The system is designed around multiple independent processors to provide continuous operation regardless of a failure anywhere in the system.
- A failure in one part of the system will not contaminate the rest of the system.
- On-line maintenance and replacement of failed modules are completed without bringing the system down. Operations at terminals and programs in progress are unaffected by either the failure or its repair.
- The standard operating system software program is designed and tested to provide continuous operation. Single hardware failures are automatically dealt with instantaneously. This is in marked contrast to most operating systems which are designed to shut the system down in a failure, rather than actively respond to it.
- The software "hides" the independent nature of the separate processors and makes the system appear as an integral whole to the application programs.
- The system is expandable to meet changing or growing needs without changing hardware cost penalty, and without having to reprogram. Instead of having to invest initially for an anticipated later need, the user can start with the exact amount of computing power he needs, and add in increments, at mini prices, all the way to a full 16 processor system. Fully expanded, a Tandem system can support 2048 data communication lines, individual files of four billion bytes fully supported by the data base management system, providing continuous real-time operation, with one of the finest packages of software available in any computer system. And it's all NonStop. From Tandem Computers.

- Each system, whether minimal or fully expanded, can be treated as a distinct node in a network system with up to 255 nodes. Each node can, with proper security, access the entire data base as if it were resident in the local system, and there is no software penalty for expansion, not only within the nodes, but in implementing the full network of nodes.

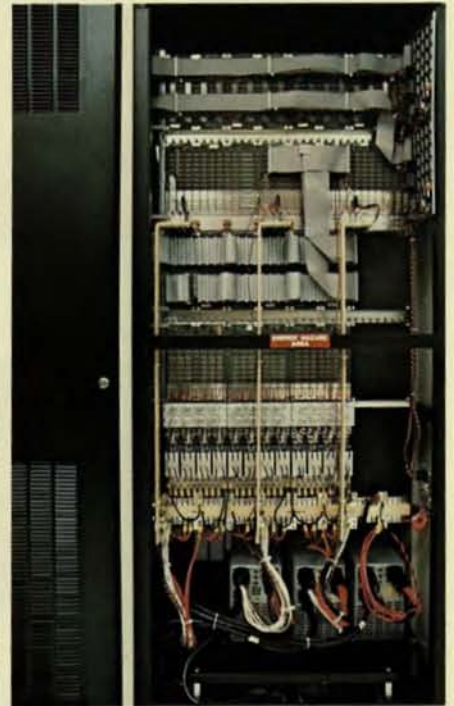


Clockwise from upper right:

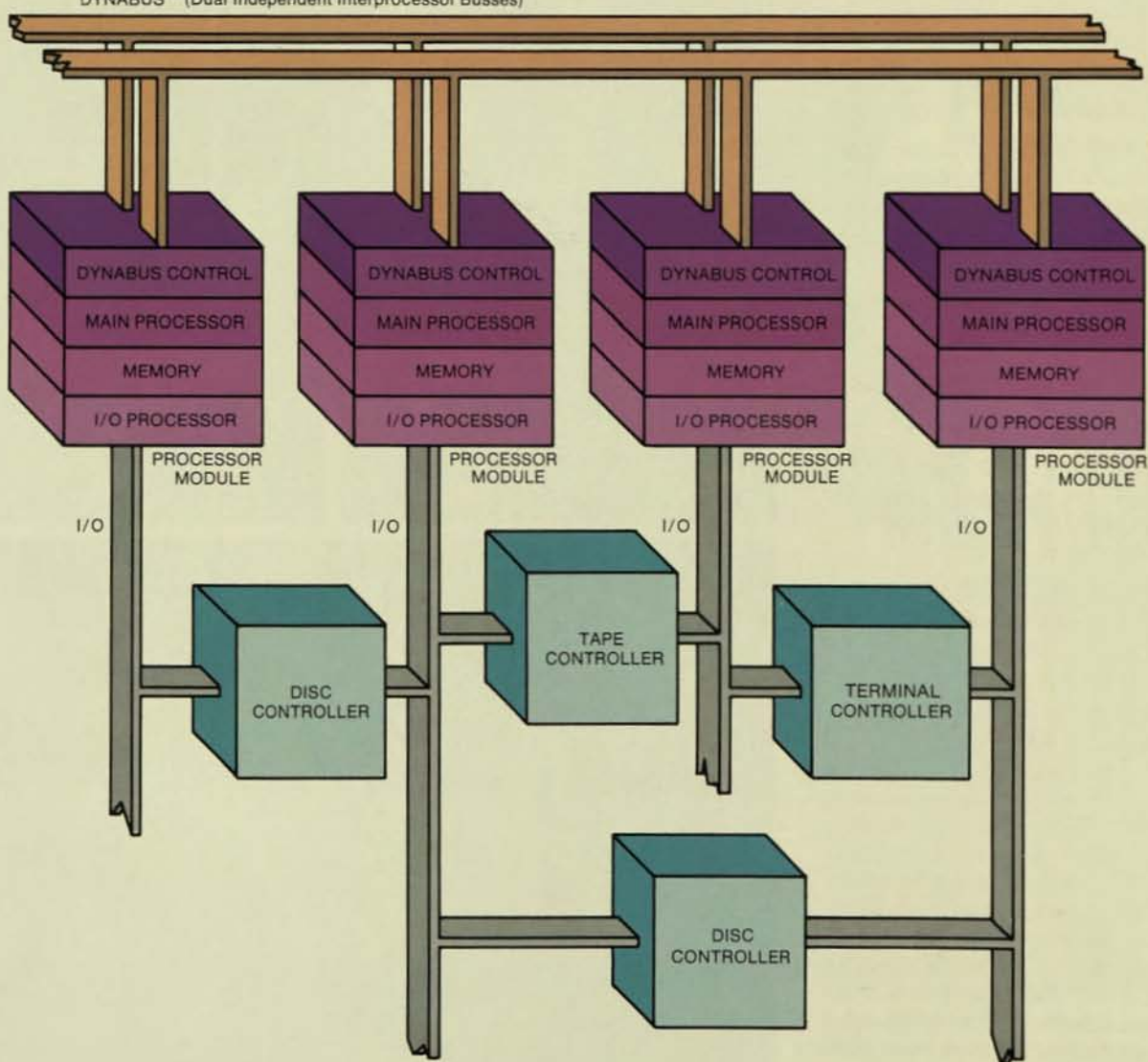
View of back of rack showing construction for a 3 processor system.

Dual independent, separately controlled interprocessor busses.

Independent I/O channels connected to separate independent ports of the I/O controllers.



DYNABUS™ (Dual Independent Interprocessor Buses)



NonStop Hardware

Multiple independent processors.

In order to insure that the loss of a processor will never completely isolate the peripheral devices attached to any controller, each Tandem controller module has two independent ports which allow it to be connected to two Tandem processor modules. In the event of a failure of that processor or I/O path, the other processor takes control automatically to insure that the system will always have a communications path to that controller.

All processor modules are interconnected through a unique, high-speed dual bus structure—Dynabus.™

Each Dynabus path is fully autonomous, operating independently of, but simultaneously with, the other bus to insure that two communications paths exist between all processor modules in the Tandem 16 NonStop System. All Dynabus interprocessor transfers are hardware controlled, independent of, but concurrent with, all normal I/O transfers and other CPU activities. Since Dynabus handles interprocessor transfers at 26 megabytes/second, this produces extremely low overhead interprocessor communications at speeds fast enough to simultaneously handle peak transaction and message loads even when the Tandem system is fully expanded to sixteen processors.

The Tandem processors are independently powerful.

Designed to increase transaction throughput and minimize system overhead, the Tandem processors separate I/O processing from interprocessor communications and CPU workloads. Each processor can handle up to 128 19.2K baud communications lines, a message handling capability miles ahead of other processors in their price category. The CPU segment of each processor is a pipelined microprogrammed unit designed specifically to handle business transac-

tion data applications. With a cycle time of 100 nanoseconds, each CPU can, for example, add two 18-digit numbers in under 2.4 microseconds.

Time-robbing I/O overhead is completely removed from the main processor.

Each processor contains an independent microprogrammed I/O processor which controls the 4 megabyte/second block multiplexed I/O channel. With truly large computer capability, each of these I/O channels can handle up to 32 device controllers. Each Tandem processor incorporates 2 megabytes of memory into a powerful system capable of complete single-error correction and multiple-error detection, using semiconductor memory at a cycle speed of 500 nanoseconds. Because of the virtual memory management feature, applications programmers need not worry about many of the constraints of physical memory.

NonStop Software

Guardian controls the traffic and watches over the system.

Guardian is Tandem's complete, transaction-oriented operating system. It resides in each processor, and has the capability to respond positively to a failure anywhere in the system. With Guardian, there is no need for customized or special operating system development by the user. Since programs are geographically independent, the task of applications programming is vastly simplified for the user. Further, Guardian carries the user-assigned priorities for applications programs, handles all communications among programs and between programs and the outside world, and extends the capability to start program execution in any available processor module from any processor.

Geographical independence of programs and data is a noteworthy feature of Guardian. Programs are not only unaware of which processor, or even which

node in a network, is running them; they may well be running simultaneously on all processors or in multiple nodes. And programs can access any device in the system, even those not physically connected to the processor(s) running the program. Because of this geographic independence, the Tandem NonStop system can be expanded all the way to sixteen processors and beyond via the network without any reprogramming.

Multiprocessor message system.

Guardian automatically handles all communications between Tandem processor modules, system processes and applications programs—routing messages to the correct processor, verifying correct receipt and deciding which program receives the message in the destination processor.

Guardian provides protection, performing comprehensive data validation on all transfers. It was designed to detect and isolate any faulty module, preventing corruption of any other module by a "mad" processor. Guardian has the inherent ability to detect an error or point of non-response anywhere in the system, log the failure and disallow access to the faulty module without disturbing system operation. The Guardian operating system, along with the system hardware architecture, is the basis of Tandem's NonStop operation.

Expand: global networks made easy.

Expand extends the basic, single node capabilities of Guardian to a whole network. Under Expand, the network can grow as large as 4,080 processors, each one capable of accessing a geographically distributed data base exactly as if it were located in its entirety in the local system. Each node of the network can be minimal or a fully expanded Tandem NonStop System; and there can be as many as 255 nodes in the network. Since Guardian treats all resources within the system, both hardware and software, as

files, expansion and perhaps reconfiguration of the system is accomplished without reprogramming, even without recompilation. Expand also extends the NonStop capabilities of the single node to the network. Packets of data are passed from node to node automatically and are rerouted automatically if a failure occurs in a communication line.

Tandem's Transaction Application Language.

T/TAL. A powerful block structured language designed for fast, flexible programming. T/TAL is self-documenting and is easy to read, modify and maintain. Developed by Tandem, T/TAL gives the programmer every vehicle to optimize the hardware potentials of its interactive, multi-processor environment. T/TAL promotes the use of structured programming principles.

Tandem/COBOL.

The only COBOL available for multiple processor systems, Tandem/COBOL (ANSI 1974) utilizes all the capabilities of Guardian and Enscribe. Under Guardian and Guardian/Expand, Tandem/COBOL features NonStop operation; shared, re-entrant code; virtual memory; geographic independence of I/O devices; and checkpoint/checkmonitor facilities. Under Enscribe, Tandem/COBOL provides key-sequenced, entry-sequenced and relative file structures; logical file size up to four billion bytes; up to 255 alternate keys per file; and optional mirror data base recording.

Tandem/FORTRAN.

This is the only FORTRAN available and optimized for multiple processor systems, Tandem/FORTRAN (ANSI 77). It utilizes all of our Guardian and Guardian/Expand Operating System features including NonStop operation, re-entrant code, interprocessor communications, virtual memory, and Enscribe data base record management facilities for keyed, relative and sequential access, multi-

keyed data paths, and concurrent record access. Benchmarks consistently show Tandem/FORTRAN to be exceptionally fast, and fully capable of running efficiently in a multi-language environment. This is comprehensive FORTRAN with a host of extensions utilizing and capitalizing on the special features of the Tandem NonStop System.

Tandem/MUMPS.

Like our COBOL and FORTRAN, this is the only MUMPS available for a multiple processor system. Tandem/MUMPS both complies with and vastly exceeds MUMPS ANSI Standard 1977. This is the only MUMPS which allows concurrent execution of COBOL, FORTRAN and T/TAL. It can access files created by other programs, and it can share global variables with other language systems. And since it operates under Guardian, Guardian/Expand and Enscribe, it offers all of Tandem's NonStop System features and support. Program development is made remarkably easy through the use of the Tandem Editor incorporated into the T/MUMPS capability. And no one else offers a MUMPS package with near the Tandem available memory.

Enscribe: versatile and efficient data base management system, providing high level access to and manipulation of records in data bases. Operating in distributed fashion across multiple processors as a part of Guardian or Guardian/Expand, Enscribe ensures the integrity of the data base in case a processor, an I/O channel, disc drive or communication line fails during transaction processing.

Enscribe protects the file structure.

During operations, Enscribe performs internal checkpointing to ensure integrity of the file structure and ensure that no user data is lost. Control information and data are maintained in two processors controlling a disc volume; if a failure occurs in one processor, Enscribe completes the operation using the alternate

processor. And Enscribe maintains all indices; when a new record is added to the file or a key value is changed, Enscribe automatically updates the indices to the affected records. The programmer need never worry about assignment or maintenance of indices.

Multiple file structures.

Key-sequenced, relative or entry-sequenced—all under Enscribe and all accessible by the primary key or any of the 255 possible alternate keys. Location of records may be by approximate (range of key values), generic (partial key matches) or exact key value. Enscribe maintains an index of all key values, providing rapid access and update whenever key values are supplied.

Four billion bytes per file.

With Enscribe, individual files may be partitioned in separate volumes, providing large system capacities and significant increase in throughput. Each partition can be under control of a separate processor, again transparent to the programmer. Enscribe includes a cache buffer management scheme, providing "look ahead" capability to optimize I/O. By increasing memory in the cache, throughput can be increased and there is no need to modify or recompile existing applications programs.

Mirrors to be sure.

Tandem is the only manufacturer which can provide this protection for Virtual Memory: if a failure occurs in the System Disc, Tandem's Mirror Capability prevents not only a shutdown, but prevents loss of any part of the operating system or application programs.

Enscribe offers an optional mirror volume technique whereby a disc volume's data can be physically recorded on two separate disc packs simultaneously. Reads may occur from the closest head in either pack, completely transparent to both the applications program and the user. If a failure occurs in one disc, all reads and

writes are automatically made from the other. As soon as the failed disc is restored, Enscribe automatically updates the failed device to exactly mirror the safe volume; this takes place concurrent with application updates. Again, completely transparent to both program and user and therefore access is never lost to the data even through something as potentially catastrophic as a head crash.

Locks, compression and utilities.

Enscribe allows for both record locking and file locking, providing flexibility of both access and security. For key-sequenced files, an optional set of techniques provides for both data and index compression, thereby reducing the number of head movements. A file update program provides for easy file creation and loading.

Data definition and manipulation.

Under Enscribe, the user can use Tandem's DDL, Data Definition Language, describing the data base as a "schema." Since all programs use the schema to access the data base, a correct view of the data is assured. It also defines which fields are to be used as access paths (keys) to retrieve records from the data base. Changes to data base record layout, additions of types of records or new alternate keys are accomplished with automated ease.

Enform: a major improvement in Queries and Reports.

Operating under Enscribe, and therefore within the framework of Guardian and Guardian/Expand, Tandem's Enform has two major advantages over any other Query/Report Writer. It operates over a distributed data base, invisibly and without special considerations. And, it defines relationships between separate records at the time of inquiry without affecting the data base. File relationships are defined by common codes, keys or fields and can be changed at will. Enform automatically takes advantage of all primary

and secondary keys to locate called data in the most efficient way. And the same English-like language is used for both queries and reports, which produces the results in a fraction of the time at a fraction of the cost. Formatting includes all appropriate signs and punctuation; you can even build in calculation of variable formulas, commissions and such. The same data independence, file structure independence and geographical independence available to the interactive users of Enform is also directly available to the application programmer in COBOL, FORTRAN, T/TAL or MUMPS. And of course you can change any aspect you wish—at will. Even convert into French, German or Spanish. It is incredibly flexible. Capable. And economical.

Envoy: multifaceted data communications system.

Operating as an integral part of Guardian, Envoy provides the interface between applications programs and data communications networks. Envoy supports both binary synchronous and asynchronous communications, with single or multi-drop lines on either a local or remote basis.

Data is transferred from terminals directly into main memory, which means processors are not interrupted until a complete message has been received. Binary synchronous terminal polling and character translation between ASCII and other communications codes are hardware executed, minimizing overhead. Asynchronous operations run at rates up to 19.2K baud per line; binary synchronous at rates up to 80K baud per line.

Entry: page mode forms creation, display and access.

One of the easiest-to-use programs of its kind. Users simply design the form on the screen to appear as it will be used. Delimiters, field names and validity checking are automatically recorded in a designated file on disc. In use, invalid data is

automatically checked and displayed as a flashing entry on the screen. Individual fields in any form may be accessed by name. This package is available for both Tandem's page mode terminals and IBM 3270 terminals.



XRAY: to balance loads and fine-tune applications.

With less than 1% overhead, Tandem's NonStop XRAY monitors total system performance and resource utilization. You can spot overloads immediately and fine-tune work distribution automatically, even as processing occurs. Bottlenecks in programs or files, processors, controllers or terminals, show up immediately on CRT display or in hard copy printouts and corrective balancing can be implemented without delay. One early application of XRAY allowed enhancement of our own system microcode so that two

Expansion, Service, Maintenance and Training — All NonStop.

processors could accomplish what previously required three.

Remote diagnostics: anywhere in the system.

Tandem's DIAGLINK provides interactive analysis of what's happening anywhere in the system, in hardware or software, from any terminal in the system (provided proper security clearances are handled) or, through a modem connection, from any Tandem Service Center. With this capability, our service personnel arrive at your DP center fully prepared to make the immediate replacement or adjustment required. And of course, the system is up and running during this whole time.

Software tools to work with.

Under the overall supervision of either Guardian/Expand or Guardian, each Tandem NonStop System has available a complete multiple processor and control communications system supporting a host of applications languages including industry standard ANSI 77 FORTRAN, ANSI 74 COBOL and ANSI 77 MUMPS. With our own T/TAL, EDITOR and its associated galley formatter TGAL, SORT/MERGE, DEBUG, ENFORM Query/Report Writer and complete diagnostic capabilities, the Tandem software support system is truly impressive. And best of all, it never requires modification as an individual system, node or the entire network expands or is modified to suit changing requirements. And any network node can communicate with IBM or any other mainframe using industry standard protocols.

We are where you are.

And not only in the development and production of advanced systems incorporating the three keys to success in the on-line environment: NonStop operation, data base integrity and expansion without reprogramming penalty.

On a more basic level, we are also where you are located—with full technical support in system evaluation, installation, service, maintenance and training. Most mini-based systems started their careers as components of others' systems in the world of OEM, and as a consequence required very little in field support. We on the other hand have been an end-user supplier since day one and have built an

incredibly strong and substantial representation in the field, both within the United States and in our overseas markets. Fully one half of our employees are directly involved in direct customer sales and support, service, maintenance and training. We have offices in every major computer market in the United States, Canada and Europe and have established strong representation in Mexico, South America and Australia. Tandem is there where and when needed.

Start with what you need.

Tandem's NonStop System is the only computer system on the market which allows you to start with only the computer



power you need right now, yet grow as your needs grow—easily, economically and without modification to your programs, either systems or applications.

Through incremental expansion all the way up to sixteen processors, virtually unlimited system growth is assured. Each additional processor module simply increases the system's processing power, input/output capability, and available memory. Allocation of processors to primary and secondary tasks (file editing, report writing, etc.) can be pre-programmed to take place at specified times with no need for user interaction.

Service during On-Line operations.

With any system a hardware failure must be repaired. But only with Tandem can the system keep right on operating, right through the failure and right through the repair too. With approval of all major international safety standards organizations. That's unique. Tandem's customer service representative can remove and replace any failed module in your system without interrupting service. The operators at terminals and the programs in process are unaffected by either the failure or replacement of the failed module. Without this feature, no system can truly be called NonStop.

Maintenance without shut-down.

Routine maintenance, too, can be accomplished without anyone using the system being aware of or inconvenienced by the process. All programs, the data

base, and system capability are maintained in operational status during the entire procedure.

Training for optimal results.

There has never been a system on the market like the Tandem NonStop System, and it offers both opportunities and challenges to the people who will make it perform to its capabilities. Because we are committed to helping our customers get the absolute maximum from their Tandem investment, we maintain extensive training facilities both in our Cupertino headquarters and in appropriate field offices. There is no aspect of the system which doesn't come in for the most intense scrutiny, and some of the enhancements of the system over the years have come from customer professionals probing the limits of its capabilities. The courses are comprehensive and low in cost; the return on investment is immediate and the benefits are enormous.

The Tandem 16 NonStop System and the Tandem NonStop Network—conceived for today's and tomorrow's needs, designed for efficiency, built for reliability, and serviced without interruption. A field-proven solution that works.



Clockwise from upper left:

Incoming inspection at the factory.

Maintenance panel facilitates on-line diagnosis and repair while the system is up and running.

Maintenance panel shown in use.

Regularly scheduled classes are held for both programmers and customer service engineers.

TANDEM COMPUTERS, INC.

Headquarters: Cupertino, California
(408) 996-6000

Regional Offices: New York (212) 594-2320;
Chicago (312) 397-5200; Dallas (817)
640-8771; Toronto (416) 863-0575; Branch
offices throughout the U.S.A., Canada and
Europe. Distributors in Mexico, South America
and Australia.

"Tandem," "NonStop" and "Dynabus" are trademarks and service marks of Tandem Computers Incorporated which have been registered in several states and for which Federal registration is pending.

TANDEM

Tandem Computers, Inc.
19333 Vallec Parkway
Cupertino, California 95014
Toll Free 800-538-9360 or
408-996-6000 in California
Telex: 352044

Tandem Computers GmbH
Bernerstrasse 50, 6000 Frankfurt/Main 56, West Germany
5072071-73, Telex: 416247-tacu-d

A Fault-Tolerant Computing System

James A. Katzman
Tandem Computers
19333 Vallco Parkway
Cupertino, California 95014

Copyright (C) 1977 Tandem Computers Inc.
First Revision January, 1979

Abstract

A fault-tolerant computer architecture is examined that is commercially available today and installed in many industries. The hardware is examined in this paper and the software is examined in a companion paper [4].

Introduction

The increasing need for businesses to go on-line is stimulating a requirement for cost effective computer systems having continuous availability [1,2]. Certain applications such as automatic toll billing for telephone systems lose money each minute the system is down and the losses are irrecoverable. Systems commercially available today have met a necessary requirement of multiprocessing but not the sufficient conditions for fault-tolerant computing.

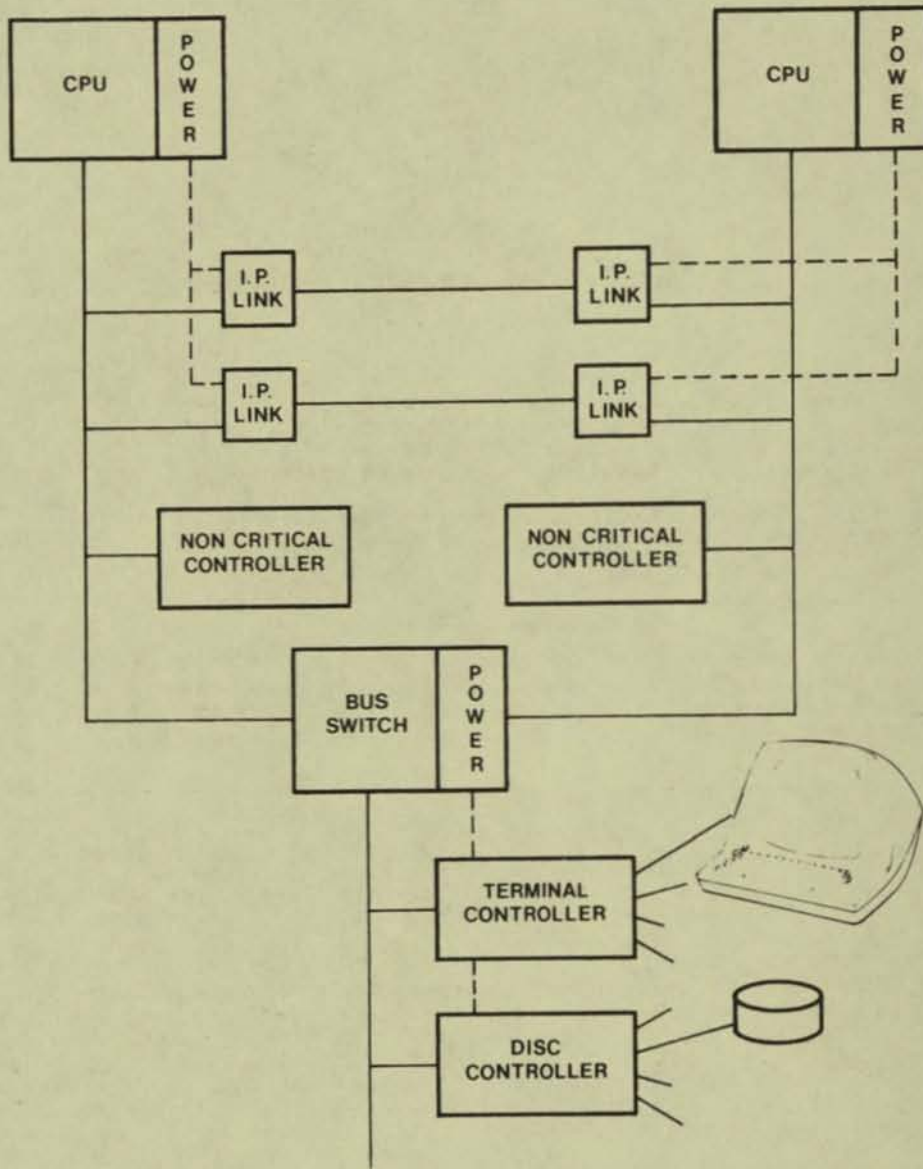
The greatest dollar volume spent on systems needing these fault-tolerant capabilities are in the commercial on-line, data base transaction, and terminal oriented applications. The design of the Tandem 16 NonStop* system was directed toward offering the commercial market an off-the-shelf, general purpose system with at least an order of magnitude better availability than existing off-the-shelf systems without charging a premium (see Appendix A). This was accomplished by using a top down system design approach, thus avoiding the shortcomings of the systems currently addressing the fault-tolerant market.

Except for some very expensive special systems developed by the military, universities, and some computer manufacturers in limited quantities, no

commercially available systems have been designed for continuous availability. Some systems such as the ones designed by ROLM have been designed for high MTBF by "ruggedizing," but typically computers have been designed to be in a monolithic, single processor environment. As certain applications demanded continuous availability, manufacturers recognized that a multiprocessor system was necessary to meet the demands for availability. In order to preserve previous development effort and compatibility, manufacturers invented awkward devices such as I/O channel switches and interprocessor communication adapters to retrofit existing hardware. The basic flaw in this effort is that only multiprocessing was achieved. While that is necessary for continuously available systems, it is far from sufficient.

Single points of failure flourish in these past architectures (Fig. 1). A power supply failure in the I/O bus switch or a single integrated circuit (IC) package failure in any I/O controller on the I/O channel emanating from the I/O bus switch will cause the entire system to fail. Other architectures have used a common memory for interprocessor communications, creating another single point of failure. Typically such systems have not even approached the problem of on-line maintenance, redundant cooling, or a power

* NonStop is a trademark of Tandem Computers



0199

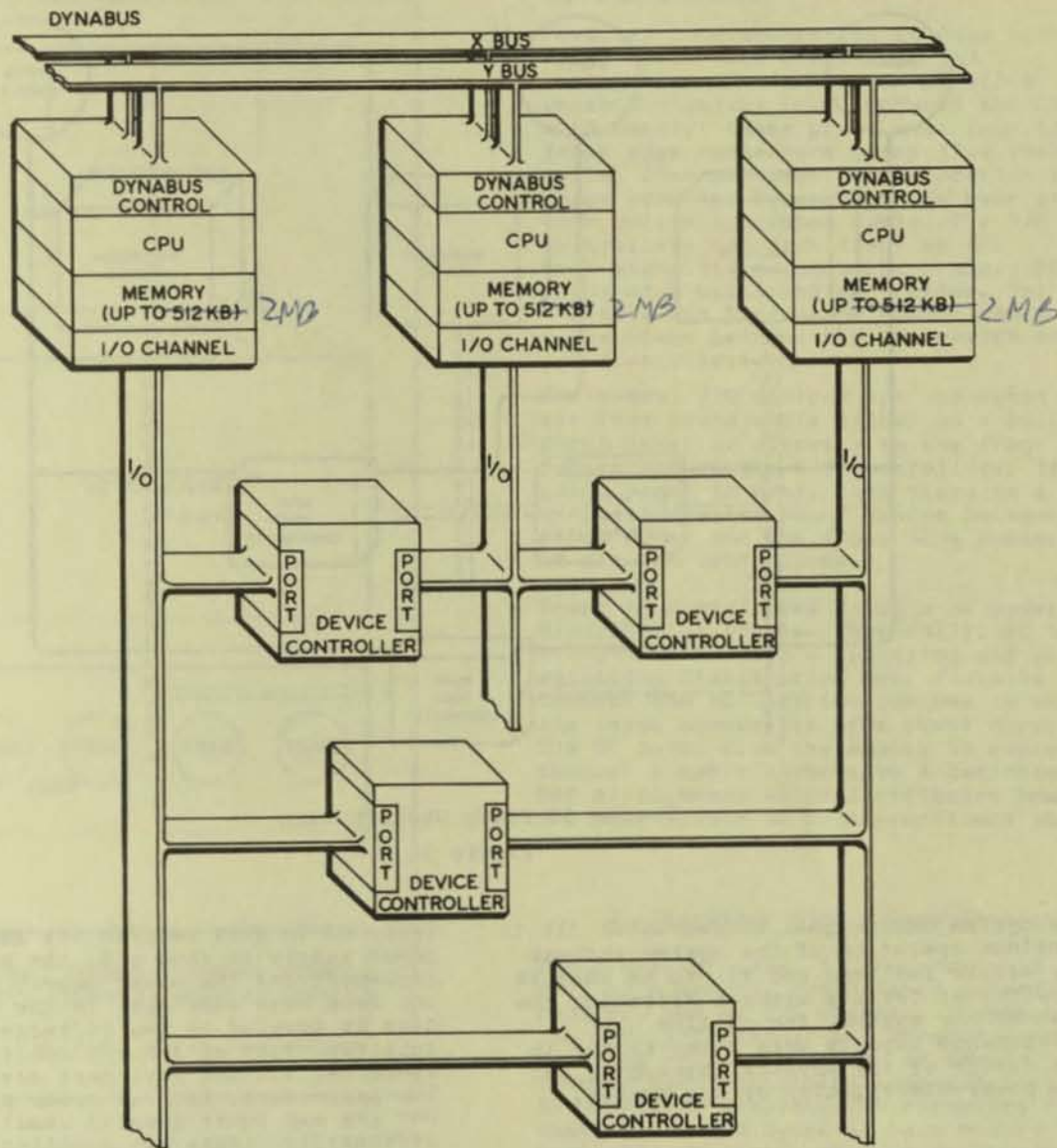
Example of Previous Fault-Tolerant Systems

Figure 1

distribution system that allows for brownout conditions. In today's marketplace, many of the applications of fault-tolerant systems do not allow any down time for repair.

Expansion of a system such as the one in figure 1 is prohibitively expensive. A three processor system, strongly connected in a redundant fashion, would require twelve interprocessor links on the I/O channels; five processors would need forty

links; for n processors, $2n(n-1)$ links are required. These links often consist of 100-200 IC packages and require entire circuit boards priced between \$6,000 and \$10,000 each. Using the I/O channel in this manner limits the I/O capabilities as a further undesirable side effect. The resulting hardware changes for expansion, if undertaken, are typically dwarfed in magnitude by the software changes needed when applications are to be geographically changed or expanded.



0200

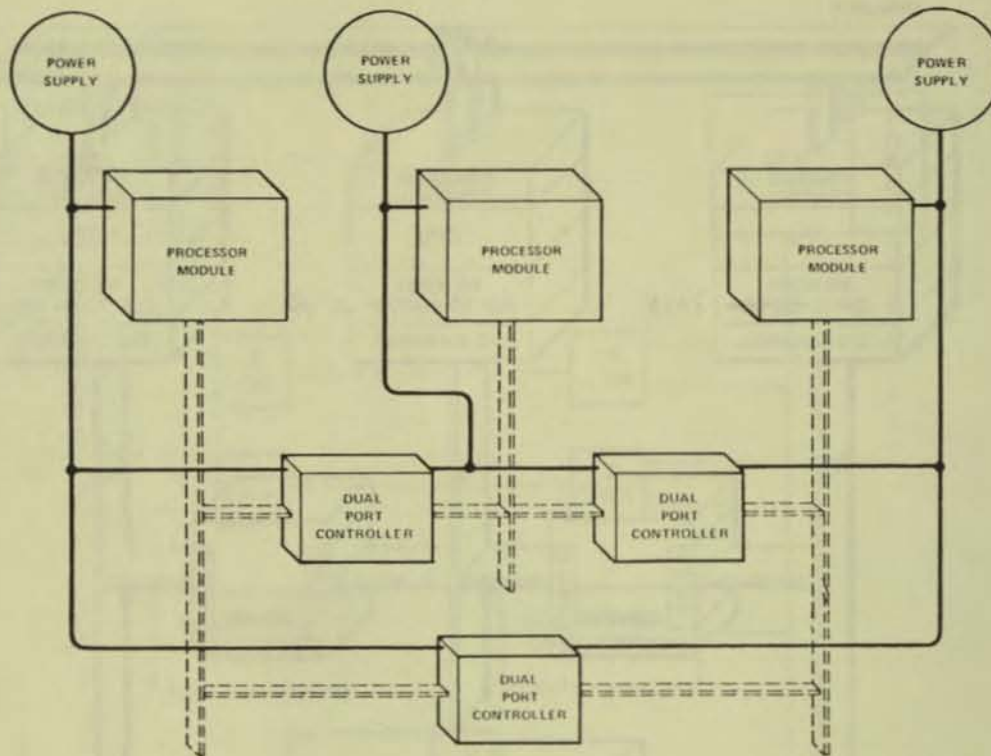
Tandem 16 System Architecture

Figure 2

1. System Organization

This paper describes the Tandem 16 architecture at the lowest level (the hardware). Section 1 deals with the overall system organization and packaging. Section 2 explains the processor module organization and its attachment to the interprocessor communications system. Section 3 discusses the I/O system organization. Section 4 discusses power, packaging, and on-line maintenance aspects that are not covered elsewhere in the paper.

The Tandem 16 NonStop system is organized around three basic elements: the processor module, dual-ported I/O controllers, and the DC power distribution system (Fig. 2,3). The processors are interconnected by a dual-interprocessor bus system: the Dynabus; the I/O controllers are each connected with two independent I/O channels, one to each port; and the power distribution system is integrated with the modular packaging of the system.



0201

Tandem 16 Power Distribution

Figure 3

The system design goal is two-fold: (1) to continue operation of the system through any single failure, and (2) to be able to repair that failure without affecting the rest of the system. The on-line maintenance aspects were a key factor in the design of the physical packaging and the power-distribution of the system.

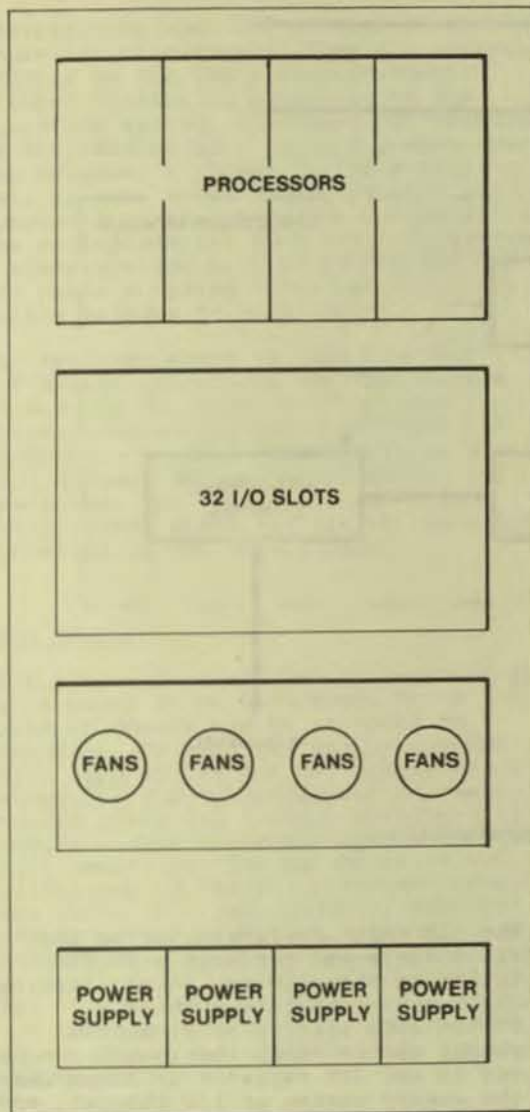
System Packaging

The cabinet (Fig. 4) is divided into 4 sections: the upper card cage, the lower card cage, cooling, and power supplies. The upper card cage contains up to 4 processors, each with up to 2 M bytes of independent main memory. The lower card cage contains up to 32 I/O controller printed circuit (PC) cards, where each controller consists of one to three PC cards. The cooling section consists of 4 fans and a plenum chamber that forces laminar air flow through the card cages. The power supply section contains up to 4 power supply modules. Multiple cabinets may be bolted together and the system has the capability to accommodate a maximum of 16 processors.

Each processor module, consisting of a CPU, memory, Dynabus control and I/O channel are powered by an associated power supply. If a failed module is to be

replaced in this section its associated power supply is shut off, the module is replaced, and the power supply is turned on. Each card cage slot in the I/O card cage is powered by two different power supplies. Each of the I/O controllers is connected via its dual-port arrangement to two processors. Each of those processors has its own power supply; usually, but not necessarily, those two supplies are the ones that power the I/O controller (Fig. 3). Each slot in the I/O card cage can be powered down by a corresponding switch disconnecting power from the slot from both supplies without affecting power to the remainder of the system. Therefore, if a power supply fails, or if one is shut down to repair a processor, no I/O controllers are affected.

The dual-power sourcing to the I/O controllers was originally designed using relay switching. This plan was abandoned for several reasons: a) to contend with relay failure modes is difficult; b) the number of contact bounces on a switch-over is neither uniform nor predictable making it difficult for the operating system to handle power-on interrupts from the I/O controllers; and c) during the switch-over, controllers do lose power, and while most controllers are software-restartable, communications controllers hang up their communications



Tandem 16 Physical Cabinet
Figure 4

0202

lines. We therefore devised a diode current sharing scheme whereby I/O controllers are constantly drawing current from two supplies simultaneously. If a power supply fails, all the current for a given controller is supplied by the second power supply. There is also circuitry to provide for a controlled ramping of current draw on turn-on and turn-off so there are no instantaneous power demands from a given supply causing a potential momentary dip in supply voltage.

Both fans and power supplies are electrically connected using quick disconnect connectors to speed replacement upon failure. No tools are required to replace a power supply. A screwdriver is all that is needed to replace a fan. Both replacements take less than 5 minutes.

Interconnections

Physical interconnection is done both using front edge connectors and back-planes. Communication within a processor module (e.g. between the CPU and main memory) takes place over four 50 pin front edge connectors using flat ribbon cable. Interprocessor communication takes place over the Dynabus on the back-plane also utilizing ribbon cable. The I/O controllers use etch trace on the back-plane for communication among PC cards of a multcard controller. The I/O channels are back-plane ribbon cable connections between the processors and the I/O controllers.

Peripheral I/O devices are connected via shielded round cable either to a bulk-head patch panel or directly to the front edge connectors of the I/O controllers. If a patch panel is used, then there is a connection using round cables between the patch panel and the front edge connectors of the I/O controllers.

Power is distributed using a DC power distribution scheme. Physically, AC is brought in through a filtering and phase splitting distribution box. Pigtails connect the AC distribution box to one of the input connectors of a power supply. The DC power from the supply is routed through a cable harness to a laminated bus bar arrangement which distributes power on the back-planes to both processors and I/O controllers.

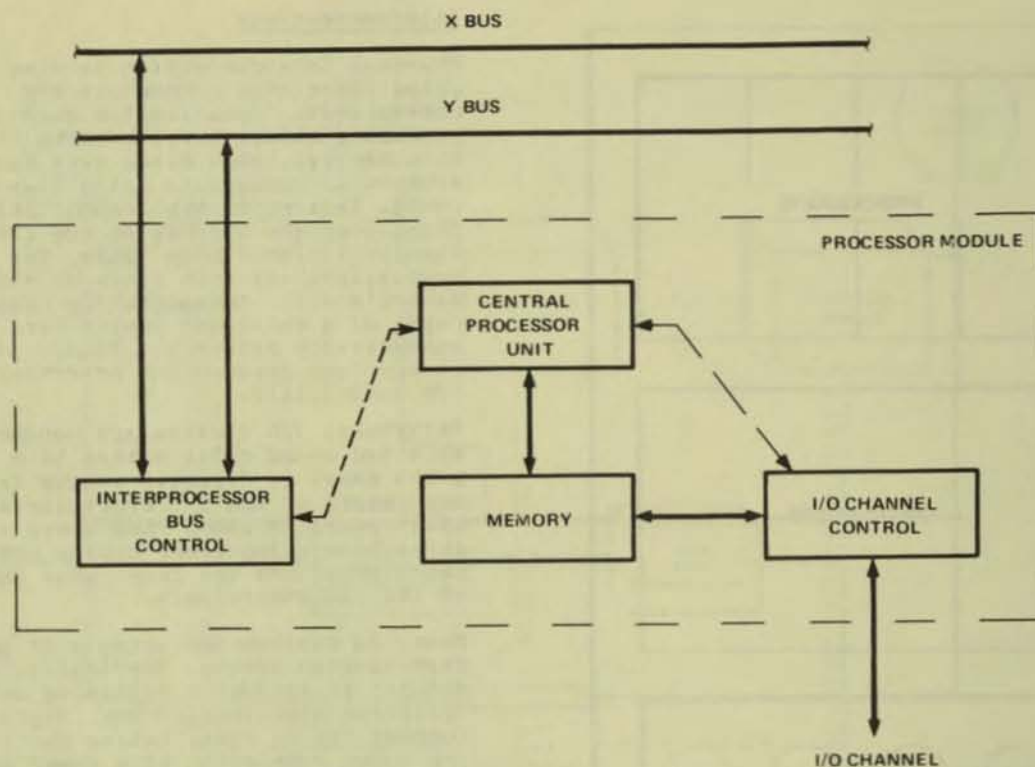
2. Processor Module Organization

The processor (Fig. 5) includes a 16 bit CPU, main memory, the Dynabus interface control, and an I/O channel. Physically the CPU, I/O channel and Dynabus control consists of two PC boards 16 inches by 18 inches, each containing approximately 300 IC packages. Schottky TTL circuitry is used. Up to 2 M bytes of main memory is available utilizing core or semiconductor technology. Core memory boards hold 32K or 128K 17-bit words and each occupy two card slots because of the height of the core stack. Semiconductor memory is implemented utilizing 16 pin, 4K or 16K dynamic RAMs. These memory boards contain 48K and 192K 22-bit words per board, respectively, and occupy only one card slot and are therefore 50% denser than core.

The processor module is viewed by the user as a 16-bit, stack-oriented processor, with a demand paging, virtual memory system capable of supporting multiprogramming.

The CPU

The CPU is a microprogrammed processor consisting of a bank of 8 registers which can be used as general purpose registers, as a LIFO register stack, or for indexing;



0110

Tandem 16 Processor Organization

Figure 5

an ALU; a shifter; two memory stack management registers; program control registers (e.g. program counter, instruction register, environment or status register, and a next instruction register for instruction prefetching); scratch pad registers available only to the microprogrammer; and several other miscellaneous flags and counters for the microprogrammer.

The microprogram is stored in read-only memory and is organized in 512-word sectors of 32-bit words. The microinstruction has different formats for branching, sequential functions, and immediate operand operations. The Tandem 16 instruction set occupies 1024 words with the decimal arithmetic and the floating point options each occupying another 512 words. The address space for the microprogram is 4K words.

The microprocessor has a 100 ns cycle time and is a two stage pipelined microprocessor, i.e., all microinstructions take two cycles to execute but one completes each cycle. In the first stage of the pipeline any two operands are selected by two source fields in the microinstruction for loading into the ALU input registers. In the second stage of the pipeline the ALU performs a primitive operation on the operands placed

in the ALU input registers during the previous cycle and performs a shift operation on the results. In parallel, a miscellaneous operation such as a condition code setting or a counter increment can be done, the result can be stored in any CPU register or dispatched to the memory system or I/O channel, and a condition test made on the results. Each of these parallel operations is controlled by a separate control field in the microinstruction.

The basic set of 173 machine instructions includes arithmetic operations (add, subtract, etc.), logical operations (and, or, exclusive or), bit deposit, block (multiple element) moves/compares/scans, procedure calls and exits, interprocessor SENDs, I/O operations, and operating system primitives. All instructions are 16 bits in length. The decimal instruction set provides an additional 32 instructions dealing with four-word operands while the floating point instruction set provides an additional 43 instructions.

The interrupt system has 16 major interrupt levels which include interprocessor bus data received, I/O transfer completion, memory error, interval timer, page fault, privileged instruction violation, etc.

Provision is made for several events to cause microinterrupts. They are entirely handled by the CPU's microprocessor without causing an interrupt to the operating system. One event for example, is the receipt of a 16 word packet over the Dynabus. A packet is the primitive unit of data which is transferred over the Dynabus for interprocessor communication. The microprocessor puts the information in a predetermined area of memory and does not cause a system interrupt until the entire message is received.

The register stack is used for most arithmetic operations and for holding parameters for block instructions (moves/comparisons/scans) which need the parameters updated dynamically so that the instructions may be interruptible and restarted. The 8-register stack is a "wraparound" stack and is not logically connected to the memory stack.

Main Memory

Main memory is organized in physical pages of 1K words of 16 bits/word. Up to 1 M words of memory may be attached to a processor. In the core memory systems there is a parity bit for single error detection, and in semiconductor memory systems there are 6 check bits/word to provide single error correction and double error detection. Due to the relative reliability of these two technologies, we have found that semiconductor memory, without error correction, is much less reliable than core, and that with error correction, it is somewhat more reliable than core. Battery backup provides short term non-volatility to the semiconductor memory system for utility power outage considerations.

It might be noted that there are some memory systems using a 21 bit error correction scheme (5 check bits on a 16 bit data word instead of 6). While 5 bits are enough to correct all single bit errors, it does not detect approximately 1/3 of the possible double bit error combinations. In these conditions, this 5 check bit scheme will incorrectly deduce that some bit (neither of the bits actually in error) is incorrect and correctable. The scheme will then correct this bit (actually causing 3 bits to be in error), and deliver it to the system as "good" reporting a correctable memory error.

Memory is logically divided into 4 address spaces (Fig. 6). These are the virtual address spaces of the machine; both the system and the user have a code space and a data space. The code space is unmodifiable and the data space can be viewed either as a stack or a random access memory, depending on the addressing mode used. Each of these virtual address

spaces are 64K words long addressed by a 16 bit virtual address.

The physical memory address is 20 bits with conversion from the virtual address to physical address accomplished through a mapping scheme. Four maps are provided, one for each logical address space; each map consists of 64 entries one for each page in the virtual address space. The maps are implemented in 50 ns access bipolar static RAM. The map access and main memory error correction is included in the 500 ns cycle time for semiconductor memory systems.

The unmodifiable code area provides reentrant, recursive, and sharable code. The data space (Fig. 7) can be referenced relative to address 0 (global data or G+ addressing), or relative to the memory stack management registers in the CPU.

The lowest level language provided on the Tandem 16 system is T/TAL, a high-level, block-structured, ALGOL-like language which provides structures to get at the more efficient machine instructions. The basic program unit in T/TAL is the PROCEDURE. Unlike ALGOL, there is no outer block, but rather a main PROCEDURE. T/TAL has the ability to declare certain variables as global. PROCEDURES cannot be nested in T/TAL, but a SUBPROCEDURE can be nested in a PROCEDURE and only in a PROCEDURE. A SUBPROCEDURE is limited in local variable access capabilities.

The memory stack, defined by two registers in the CPU, is used for efficient linkage to and from procedures, parameter passing, and dynamic storage allocation and deallocation for variables local to the procedure.

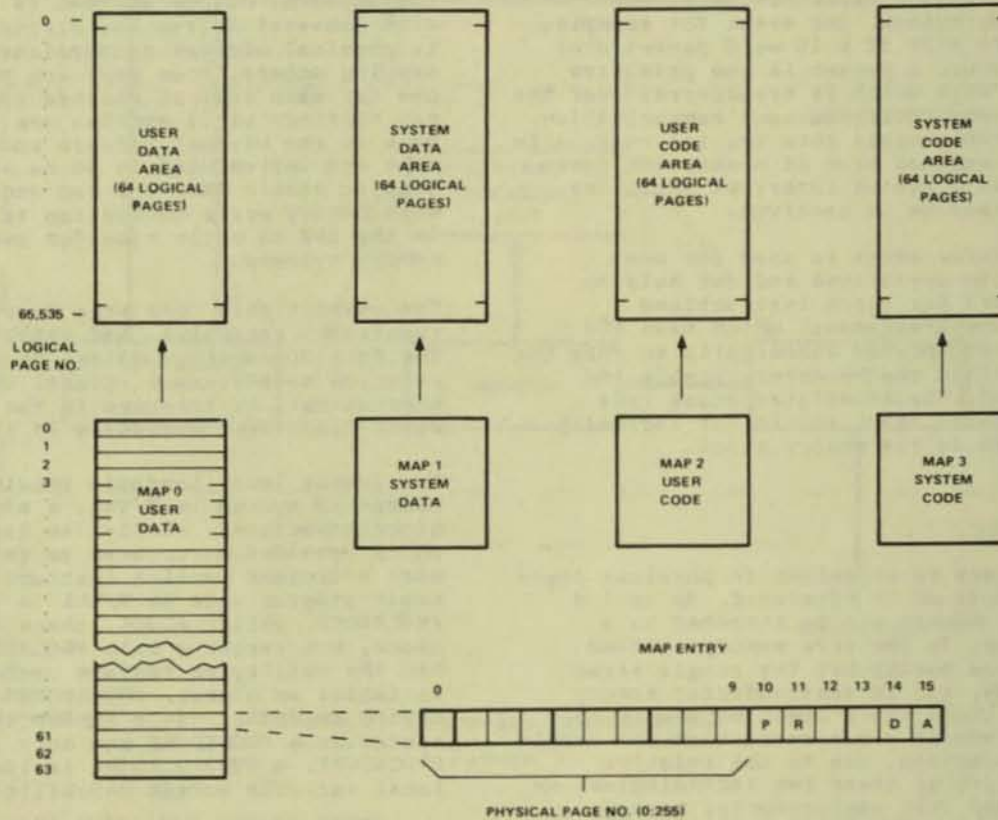
The L register (Local variables) points to the last stack marker placed on the stack. This marker contains return information about the caller such as the return address and the previous location of the L register. The contents of the L register are primarily changed by the procedure call and exit instructions.

Addressing relative to the L register provides access to parameters passed to a procedure (L-) and local variables of the procedure (L+). Parameters may be passed either by value (using direct addressing) or by reference (using indirect addressing).

The S register (stack top pointer) points to the last element placed on the stack. It is used for a SUBPROCEDURE's sublocal data area when S relative addressing (S-) is used.

There is a special mode of addressing used by the operating system, called System Global (SG+) addressing. It is used by the operating system while it is working in a

LOGICAL ADDRESS



P - PARITY
 R - REFERENCE BITS - USED BY OPERATING SYSTEM TO SELECT A PAGE FOR OVERLAY
 D - DIRTY BIT - SET WHENEVER A WRITE ACCESS IS MADE TO THE PAGE
 A - ABSENT - "1" INDICATES THAT THE PAGE IS NOT PRESENT IN PHYSICAL MEMORY

0203

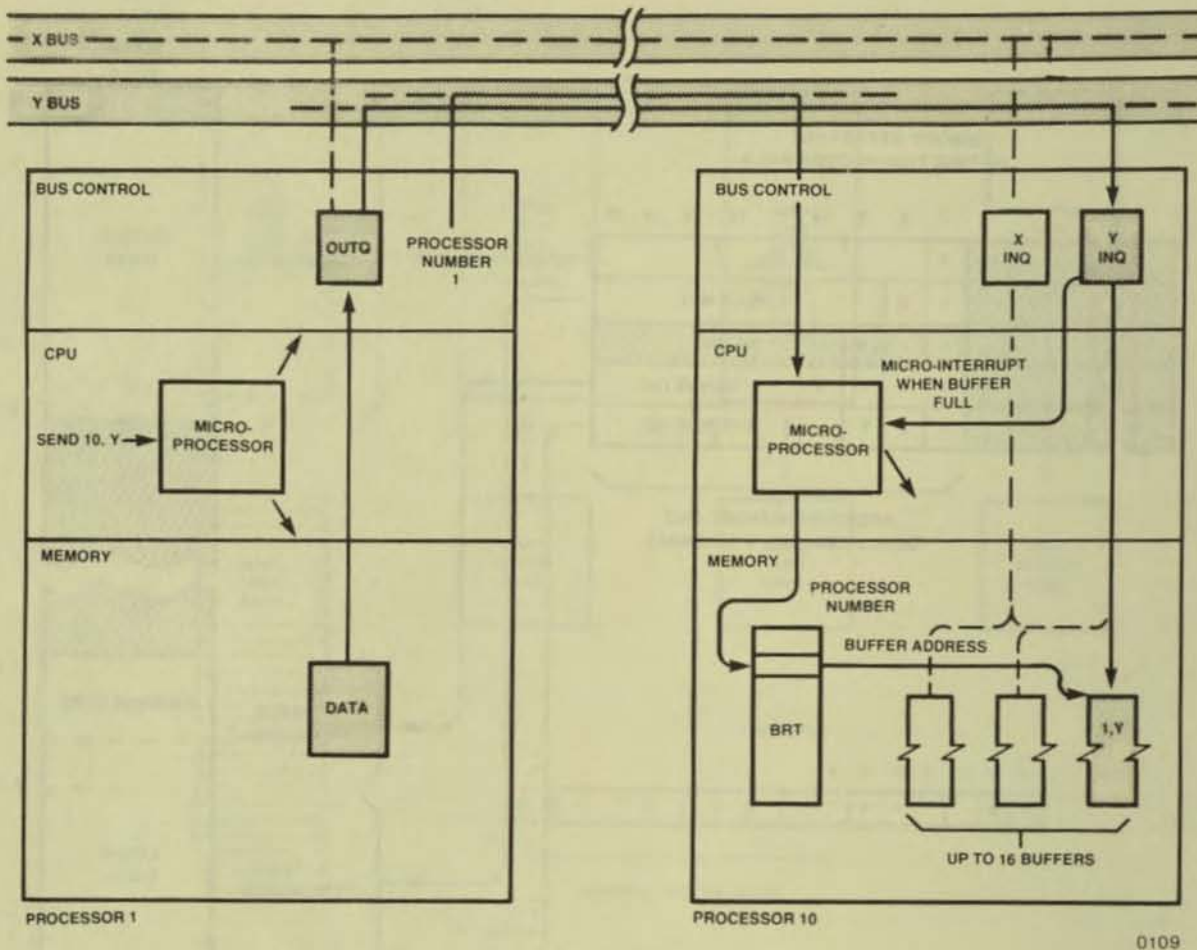
Tandem 16 Logical Memory Address Spaces
 Figure 6

user's virtual data space (on his behalf) and needs to address the system data space. The system data space contains many resource tables and buffers and the need to access them quickly justifies the existence of this addressing mode.

There are three tables known to the operating system, the microprogram and the hardware: the system interrupt vector (SIV), the I/O Control (IOC) table, and the Bus Receive Table (BRT). These tables will be explained in later sections as appropriate.

The Dynabus

The Dynabus is a set of two independent interprocessor buses. Bus access is determined by two independent interprocessor bus controllers. Each of these controllers is dual-powered, in the same manner as an I/O controller. The Dynabus controllers are very small, approximately 30 IC packages, and are not associated with, nor physically a part of any processor. Each bus has a two byte data path and control lines associated with it. There are two sets of radial

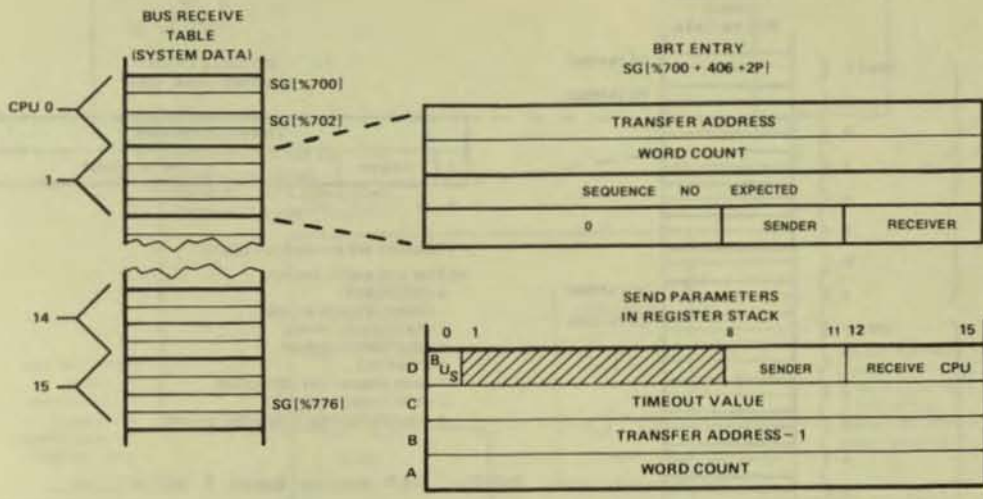


Tandem 16 Dynabus Interface & Control
Figure 8

Each processor in the system attaches to both interprocessor buses. The Dynabus interface control section (Fig. 8) consists of 3 high speed caches: an incoming queue associated with each interprocessor bus, and a single outgoing queue that can be switched to either of the buses. All caches are 16 words in length and all bus transfers are cache to cache. All components that attach to either of the buses are kept physically distinct, so that no single component failure can contaminate both buses simultaneously. Also in this section are clock synchronization and interlock circuitry. All processors communicate in a point to point manner using this redundant direct shared bus (DSB) configuration [3].

For any given interprocessor data transfer, one processor is the sender and the other the receiver. Before a processor can receive data over an interprocessor

bus, the operating system must configure an entry in a table (Fig. 9) known as the Bus Receive Table (BRT). Each BRT entry contains the address where the incoming data is to be stored, the sequence number of the next packet, the processor number of the sender and receiver, and the number of words expected. To transfer data over a bus, a SEND instruction is executed in the sending processor, which specifies the bus to be used, the intended receiver, and the number of words to be sent. The sending processor's CPU stays in the SEND instruction until the data transfer is completed. Up to 65,535 words can be sent in a single SEND instruction. While the sending processor is executing the SEND instruction, the Dynabus interface control logic in the receiving processor is storing the data away according to the appropriate BRT entry. In the receiving processor this occurs simultaneously with program execution.



BUS = X OR Y (0 = X BUS)
 CPU = PROCESSOR MODULE 0-15
 32768 - TIMEOUT VALUE IS THE NUMBER OF 0.8 μSEC
 UNITS ALLOCATED TO COMPLETING THE SEND EXAMPLE.
 TIMEOUT VALUE = 0 THEN
 32768 * 0.8 = 0.026 SECONDS

NOTE: "%#" means base 8 notation. 0205

Bus Receive Table
Figure 9

The message is divided into packets of 14 information words, a sequence number word, and an LRC check word. The sending processor first fills its outgoing queue with these packets, requests a bus transfer, and transmits upon grant of the bus by the interprocessor bus controller. The receiving processor fills the incoming queue associated with the bus over which the packet is received, and issues a microinterrupt to its own CPU. The microprocessor of the CPU checks the BRT entry, stores the packet away, verifies the LRC check word, and updates the BRT entry accordingly. If the count is exhausted the currently executing program is interrupted, otherwise program execution continues.

The BRT entries are four words that include a transfer count buffer address, sequence number expected and the sender and receiver CPU numbers. The SEND instruction has as parameters the designation of the bus to be used, the intended receiver, the data buffer address in the system data space, the word count to be transferred, and a timeout value. Error recovery action is to be taken in case the transfer is not completed within the timeout interval. These parameters are placed on the register stack and are dynamically updated so that the SEND instruction is interruptible on packet boundaries.

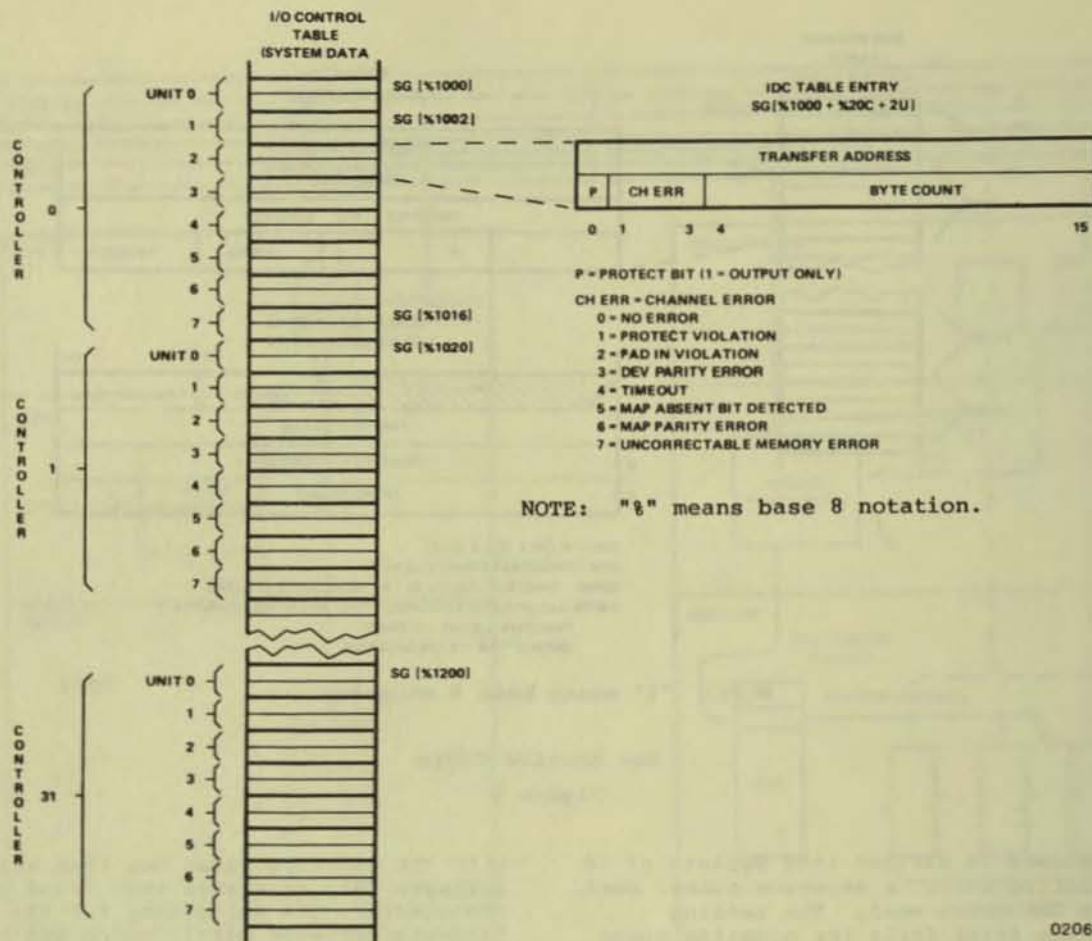
There are several levels of protocol, beyond the scope of this paper, dealing

with the interprocessor bus that exist in software [4], to assure that valid data is transferred. The philosophy for the hardware/software partitioning was to leave the more esoteric decisions to the software, e.g., alternate path routing, and error recovery procedures, with fault detection and reporting implemented in the hardware. Fault detection was designed in those areas having the highest anticipated probability of error.

The Input/Output Channel

The heart of the Tandem 16 I/O System is the I/O channel. All I/O is done on a direct memory access (DMA) basis. The channel is a microprogrammed, block multiplexed channel with the block size determined by the individual controllers. All the controllers are buffered to some degree so that all transfers over the I/O channel are at memory speed (4 M Bytes/Second) and never wait for mechanical motion since the transfers always come from a buffer in the controller, rather than from the actual I/O device.

There exists a table in the system data space of each processor called the IOC (I/O Control) table that contains a two word entry (Fig. 10) for each of the 256 possible I/O devices attached to the I/O channel. These entries contain a byte count and virtual address in the system data space for data transfers from the I/O system.



I/O Control Table
Figure 10

The I/O channel moves the IOC entry to active registers during connection of an I/O controller and restores the updated values to the IOC upon disconnection. The I/O channel alerts the I/O controller when the count has been exhausted and that causes the controller to interrupt the processor.

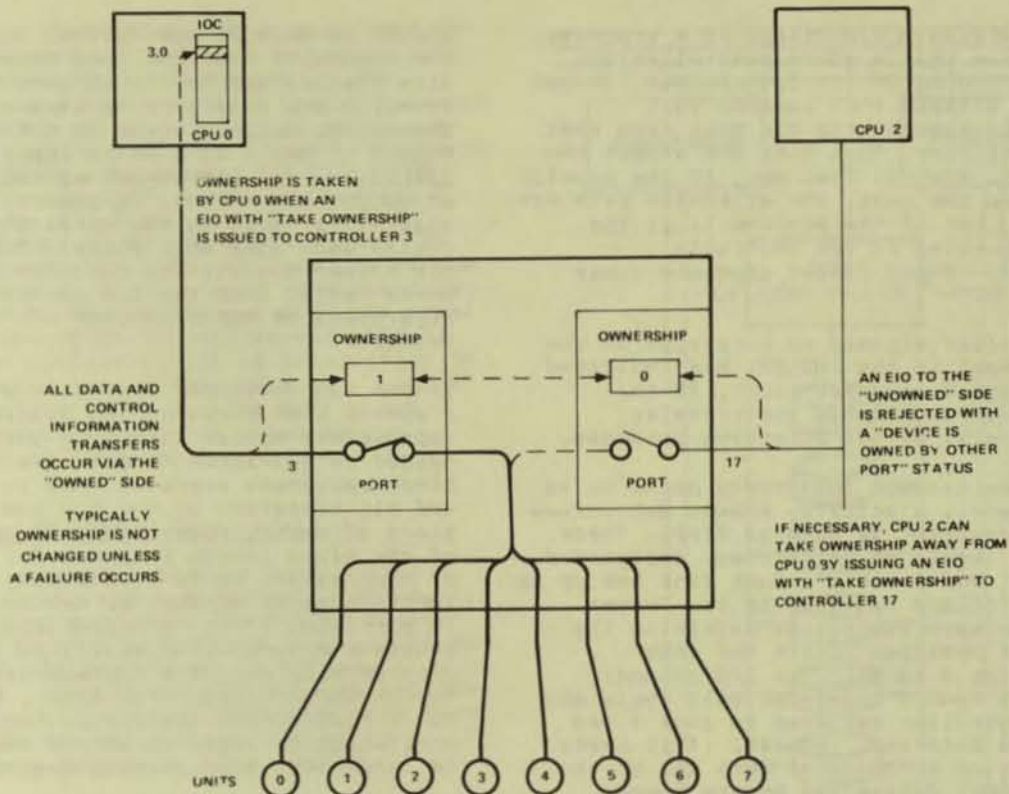
The channel does not execute channel programs as on many systems but it does do data transfer in parallel with program execution. The memory system priority always permits I/O accesses to be handled before CPU or Dynabus accesses (in an on-line, transaction oriented environment, it is rare that a system is not I/O bound). The maximum I/O transfer is 4K bytes.

3. I/O System Organization

The I/O system had a design goal of being very efficient in a transaction, on-line oriented environment. This environment has constraints different from those of a batch environment. The figure of merit in

an on-line system is the number of transactions/second/dollar that can be handled by the system. We also wanted an I/O system that had low overhead, fast transfer rates, no overruns, and no interrupts to the system until a logical entity of work was completed (e.g., no character by character interrupts from the terminals). The resulting design satisfied these goals by implementing an I/O system that was extremely simple.

I/O controllers reconnect to the channel when their buffers are stressed past a configurable threshold, transfer data in a burst mode until their buffer stress is zero (buffer empty on input operations, full on output operations), and disconnect from the channel. When the transfer terminates the I/O controller interrupts the processor. Controllers may interrupt for other reasons than an exhausted byte count, e.g., a terminal controller receiving an end-of-page character from a page mode terminal, or I/O channel error condition, or a disc pack being mounted.



0207

Ownership Circuitry

Figure 11

Dual-Port Controllers

The dual-ported I/O device controllers provide the interface between the Tandem 16 standard I/O channel and a variety of peripheral devices using distinct interfaces. While the I/O controllers are vastly different, there is a commonality among them that folds them into the Tandem 16 NonStop architecture.

Each controller contains two independent I/O channel ports implemented by IC packages which are physically separate from each other so that no interface chip can simultaneously cause failure of both ports. Each port of each controller has a 5-bit configurable controller number, and interrupt priority setting. These settings can be different on each port. The only requirement is that each port attached to an I/O channel must be assigned a controller number and priority distinct from controller numbers and priorities of other ports attached to the same I/O channel.

Each controller has a PON (power-on) circuit which clamps its output to ground whenever the controller's DC supply voltage is not within regulation. The PON circuit has hysteresis in it so that it will not oscillate if the power should

hover near the limit of regulation. When the power is within regulation, the output of the PON circuit is at a TTL "1" level. A power-on condition causes a controller reset and also gives an interrupt to one of the two processors to which it is attached. The output of the PON circuit is also used to enable all the I/O channel bus transceivers so that a controller being powered down will not cause interference on the I/O channels during the power transient. This is possible because the PON circuit operates with the supply voltage as low as .2 volts and special transceivers are used which correctly stay in a high impedance state as long as the control enable is at a logical "0".

Logically only one of the two ports of an I/O controller is active and the other port is utilized only in the event of a path failure to the primary port. There is an "ownership" bit (Fig. 11) indicating to each port if it is the primary port or the alternate. Ownership is changed only by the operating system issuing a TAKE OWNERSHIP I/O command. Executing this special command causes the I/O controller to swap its primary and alternate port designation and to do a controller reset. Any attempt to use a controller which is not owned by a given processor will result

in an ownership violation. If a processor determines that a given controller is malfunctioning on its I/O channel, it can issue a DISABLE PORT command that logically disconnects the port from that I/O controller. This does not affect the ownership status. That way, if the problem is within the port, the alternate path can be used, but if the problem is in the common portion of the controller, ownership is not forced upon the other processor.

A controller signals an interrupt on the I/O channel if the channel has indicated an exhausted transfer count, if the controller terminates the transfer prematurely, or for attention purposes.

When simultaneous interrupts occur on an I/O channel, a priority scheme determines which interrupt is handled first. There are two levels of priorities, designated "rank 0" and "rank 1". Each rank has up to 16 controllers assigned to it. Jumper wires on each controller determine the rank and position within the rank (positions 0 to 15). The I/O channel issues a rank 0 interrupt poll cycle and each controller assigned to rank 0 can place an interrupt request, if it needs service, on a dedicated data bit of the I/O channel determined by the jumper wires. If there are no controllers on rank 0 requiring service, the I/O channel issues the interrupt poll cycle for rank 1. Note, only 32 controllers can be assigned to a given channel and each one has a unique rank and position designation. The highest priority controller is granted access to the interrupt system. Thus a radial polling technique allows the processor to resolve 32 different controller priorities in just two poll cycles. Each port of a controller has a separate set of configuration jumpers so that a controller can have different priorities on its primary and alternate path.

Controller Buffer Considerations

In the design of the Tandem 16 I/O system, a lot of attention was paid to the overrun problem. While overruns are possible on this system, they have been made a rare occurrence. Each I/O controller has 3 configurable settings: the I/O controller number, the interrupt priority, and buffer stress threshold reconnect setting.

Each I/O controller is buffered to some extent. The asynchronous terminal controller has 2 bytes of buffering, while the disc controller has 4K bytes of buffering. Considerations of device transfer rate, channel transfer rate, the individual controller's buffer depth, the controller's reconnect priority, and a given channel's I/O complement can be used to determine the buffer's depth (stress threshold) at which a reconnect request

should be made to the channel to minimize the chance of overrun. Each controller with significant buffering (more than 32 bytes) has a configurable stress threshold. Buffer stress is defined as the number of cells full on an input operation, and the number of cells empty on output operations. In general, the I/O channel relieves stress while the I/O device generates more stress. Therefore the higher the stress, the more the buffer needs relief from the I/O channel, regardless of the direction of data transfer.

Tandem has developed a program which takes a system configuration and determines the appropriate stress threshold settings needed to guarantee no data overruns. Since reconnect overhead time is known, and all transfers on the I/O bus take place at memory speed, and the upper bound of the block length is known for each type of controller, it is a deterministic function as to whether or not an overrun is possible. If it is impossible to generate a no-overrun configuration, the program will output a minimum-overrun threshold settings. Most times, however, it is possible to iterate on the configuration until threshold settings can be determined that prevent overruns.

Disc Controller Considerations

The greatest fear that an on-line system user has is that "the data base is down" [5]. Many of these users are willing to pay the premium of having duplicated or "mirrored" data bases in case a disc drive fails. To meet this requirement, Tandem provides automatic mirroring of data bases.

A disc volume is a set of data contained on one spindle or one removable disc pack. A user may declare any of the disc volumes as mirrored pairs at system generation time (Fig. 12). The system then maintains these pairs so they always contain identical data. Thus protection is achieved for a single drive failure. Each disc drive in the system may be dual-ported. Each port of a disc drive is connected to an independent disc controller. Each of the disc controllers are also dual-ported and connected between two processors. A string of up to 8 drives (4 mirrored pairs) can be supported by a pair of controllers in this manner.

Note that in this configuration there are many paths to any given data and that data can be retrieved regardless of any single disc drive failure, disc controller failure, power supply failure, processor failure, or I/O channel failure.

The disc controller is buffered for a maximum length record which provides several features important in an on-line system. For example, the disc controller is absolutely immune to overruns.

This disc controller uses a Fire code [6] for burst error correction and detection. It can correct 11 bit bursts in the controller's buffer before transmission to the channel. Since overlapped seeks are allowed by the controller, when data is to be read from a mirrored pair it can be read from the drive which has its arm closest to the data cylinder. This is accomplished by using "split seeks," a SYSGEN parameter that requires one of the mirrored pair to only read from the first half of the disc cylinders with the other disc responsible for the second half of the disc cylinders. It is interesting to note that since the majority of transactions in an on-line system are reads, mirrored volumes actually can increase performance.

NonStop I/O System Considerations

The I/O channel interface consists of a two byte data bus and control signals. All data transferred over the bus is parity checked in both directions, and errors are reported via the interrupt system. A watchdog timer in the I/O channel detects if a non-existent I/O controller has been addressed, or if a controller stops responding during an I/O sequence.

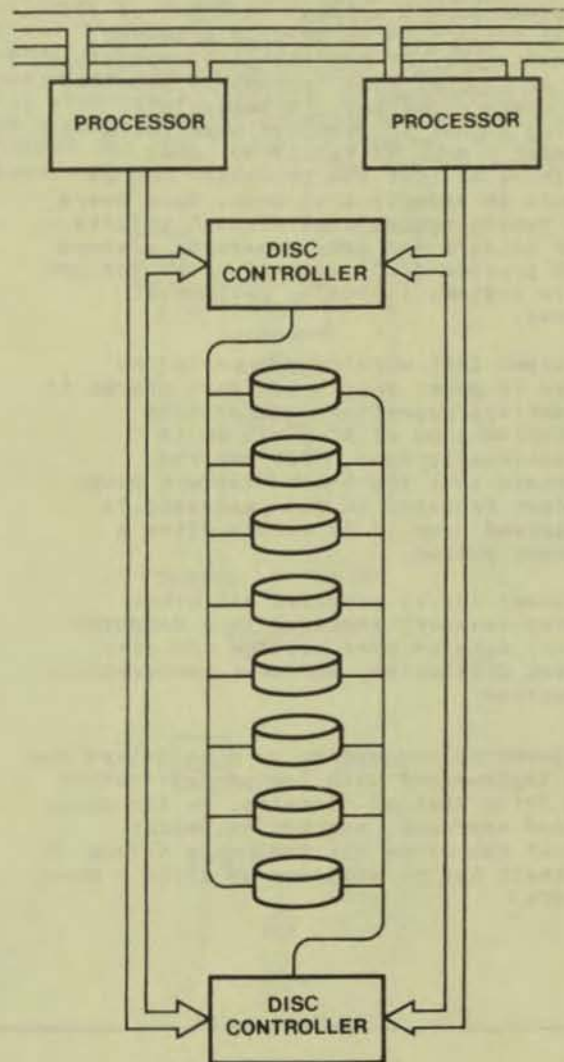
The data transfer byte count word in the IOC entry contains four status bits including a protect bit. When this bit is set to "1" only output transfers are permitted to this device.

Because I/O controllers are connected between two independent I/O channels, it is very important that word count, buffer address, and direction of transfer are controlled by the processor instead of within the controller. If that information were to be kept in the controller, a single failure could cause both processors to which it was attached to fail. Consider what would happen if a byte count register was located in the controller and was stuck in such a situation such that the count could not decrement to zero on an input transfer. It would be possible to overwrite the buffer and cause system tables to become meaningless. The error would propagate to the other processor upon discovery that the first processor was no longer operating.

Other error conditions that the channel checks for are violations of I/O protocol, attempts to transfer to absent pages (it is the operating system's responsibility to "tack down" the virtual pages used for I/O buffering), uncorrectable memory errors, and map parity errors.

4. Power, Packaging, On-line maintenance

The Tandem 16 power supply has 3 sections: a 5 volt interruptible section, a 5 volt uninterruptible section, and a 12-15 volt



0208

Tandem 16 Disc Subsystem Organization
Figure 12

uninterruptible section. The interruptible section will stop supplying DC power when AC is lost while the uninterruptible sections will continue to supply DC power. The interruptible section powers I/O controllers and that portion of a processor which is not related to memory refresh operation. The uninterruptible sections provide power for the memory array and refresh circuitry. The 5 volt sections are switching regulated supplies while the 12-15 volt section is linearly regulated. The uninterruptible sections have a provision for a battery attachment so that in case of utility power failure, memory contents are kept for 1.5 to 4 hours, depending on the amount of memory attached to the supply.

The power supply accepts AC input of 110 or 220 volts +20% to provide brownout insensitivity. At nominal line conditions, over 30 msec of ride through is provided by storage capacitors. A power-fail warning signal is provided when there is at least 5 msec of regulated power remaining so that the processor can go through an orderly shut down. Some users must remain operational through utility power failure and have generator systems which provide continuous AC power for the entire system, including peripheral devices.

The power-fail warning scheme in the Tandem 16 power supply monitors charge in the storage capacitors rather than monitoring loss of AC peaks as is conventionally done. This has the advantage that the 5 msec to do a power shutdown sequence in the processor is guaranteed even if it occurs after a brownout period.

The power supply provides all other prudent features required in a computer system, such as over voltage and over current protection, and over temperature protection.

The power-up sequencing on disc drives has been implemented with independent rather than daisy chained circuits. In the daisy chained approach, one bad sequencer circuit can cause the remaining drives in the chain not to sequence up after a power failure.

Further Packaging and On-line Maintenance Considerations

Modularity is a key concept in the Tandem 16 system. The maintenance philosophy is to make all repair by module replacement at the user site without making the system unavailable to the user. Therefore the backplanes, power supplies, fans, I/O channels, as well as the PC cards are modular and easily replaceable. Thumb screws are used when they can be so that a minimum of tools are needed for repair. The package is designed so that there is easy access to all modules.

Processors and I/O controllers not only can be replaced on-line, but added on-line without system interruption if expansion is planned, all without application software being changed.

Summary

The contribution of the Tandem 16 system lies in the synthesis of a system to directly address the need of the NonStop application marketplace. By avoiding the "onus of compatibility" to any previous system, an architecture could be designed from "scratch" that was "clean" and efficient.

The system goals have been met to a large degree. Systems have been installed containing two to twelve processors. Many application programs are on-line and running. They recover from failures, and stay up continuously.

Biography

James A. Katzman is a founder and Vice President of Marketing Support for Tandem Computers. From Tandem's start in November of 1974 through mid 1978, Mr. Katzman held the post of Vice President of Engineering and is one of the principal architects of the Tandem 16 NonStop system.

Previously Katzman was responsible for the design of the integrated I/O channel for the Amdahl 470V/6 system while at Amdahl Corp. from 1971 to 1974. While at

Hewlett-Packard Company from 1968 to 1971 he was one of the principal architects of the H-P 3000 computer system.

Mr. Katzman holds patents on all of the above machines. He is a member of the ACM and IEEE. Academically he holds a BSEE from Purdue University and a MSEE from Stanford University. He is a member of Tau Beta Pi, Eta Kappa Nu, and Omicron Delta Kappa honorary societies. He is listed in the 1978-1979 edition of Who's Who in the West.

APPENDIX A

The Tandem 16 system provides its high availability through architecture. In the literature [7,8] we find that availability ranges between 0 and 1 and is defined as:

$$A = \frac{MTBF}{MTBF+MTTR} \quad (1)$$

where

A = Availability
 MTBF = Mean Time Between Failure
 MTTR = Mean Time To Repair

The availability of two redundant systems where only one is required is represented by:

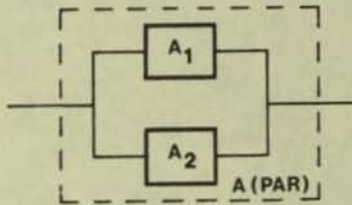


Figure 13

and the parallel system, A(PAR), has an availability of

$$A(PAR) = A_1 + A_2 - A_1 A_2 \quad (2)$$

If $A_1 = A_2 = A$ then,

$$A(PAR) = 2A - A^2 \quad (3)$$

When subsystems in series are required for operation, the system is represented by:

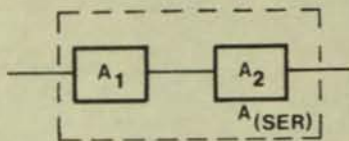
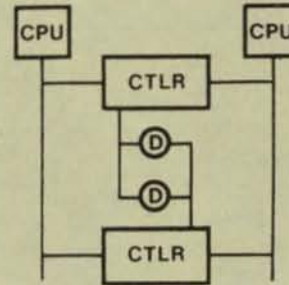


Figure 14

and the series systems, A(SER), has an availability of

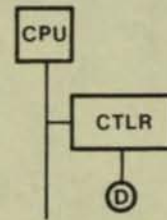
$$A(SER) = A_1 A_2 \quad (4)$$

While it is not the intention of the author to give any more than these basics of the theory of Availability, a comparison of 3 architectures of disc subsystems connected to host computers will serve as an example to demonstrate the order-of-magnitude more availability claimed for the Tandem 16 system. The three architectures will be the following:



Tandem 16 System

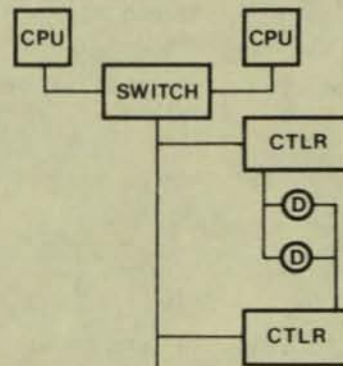
Figure 15



Batch System

Figure 16

and the typical "fault-tolerant" system



"f-t" System

Figure 17

0209

Hardware

STANDARD PAPER

SYS Network

INTRODUCTION

TANDEM HARDWARE

The Tandem 16 System represents a major departure from existing computer architecture. For the first time, a complete system has been designed to meet the growing demand for on-line transaction processing systems with failure tolerance capabilities. The basic design philosophy of the Tandem 16 Computer System is that no single module failure will stop or contaminate the system. Using standard Tandem hardware and software modules, the user may build a system to match necessary requirements exactly, both in throughput and reliability. No special or custom designed hardware or software is necessary. Equally important, the Tandem 16 System can grow to meet increasing demands with no change in operating system, applications software, or existing hardware. This growth is possible even without loss of the system during expansion.

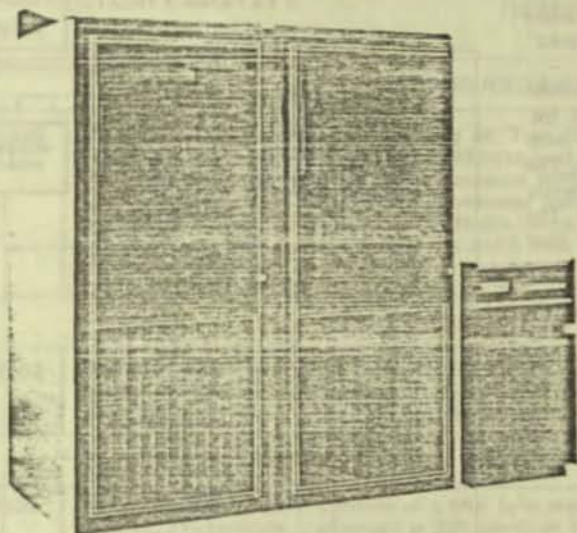
One Tandem 16 processor module is a powerful computer. The Tandem 16 Computer System consists of two to sixteen processor modules. Processor modules are interfaced to one another by means of the two interprocessor buses. A processor module may interface to I/O devices by means of its input/output channel.

Each processor module contains the functions that normally comprise a complete computer system: central processing unit, memory and input/output channel. Therefore, each processor module is capable of operating independently of, and simultaneous with, all other processor modules in the system.

The interprocessor buses (which are always under control of the Tandem Operating System - GUARDIAN) are used to transfer data between the memories of processor modules.

Data is transferred between an input/output device (i.e., discs, terminals, line printers, etc.) and a processor module by means of an input/output channel. Each processor module has one I/O channel that is capable of communicating with up to 256 I/O devices. I/O devices are interfaced to I/O channels by dual-port controllers. Each dual-port controller is connected to the I/O channels of any two processor modules. Therefore, each I/O device can be controlled by either of two processor modules. (In actual practice, an I/O device is controlled exclusively by one processor module until a failure occurs such that the processor module can no longer communicate with the I/O device. If such a failure occurs, the other processor module takes control of the I/O device.)

EM NonStop™ STANDARD PACKAGED SYSTEMS



TANDEM

NonStop™ STANDARD PACKAGED SYSTEMS

- NonStop™ Multi-Processor Systems for On-Line Transaction-Oriented Applications
- Autonomous/Dual DYNABUS™ for Inter-Processor Communications (13 Mega Bytes/second each)
- Fast (800 or 500 nSec) Core or Semiconductor Memory with Virtual Memory Control and Automatic/Dynamic Program Allocation
- High-Speed Redundant Dual-Port I/O Channels with Dedicated Microprocessed Interrupt and Block-Multiplexed DMA (2.5 or 4.0 Mega Bytes/Second)
- Versatile System Applications Support with Tandem/Guardian and Tandem/Transaction Applications Language (T/TAL)
- 100 ns cycle time Processors

SYSTEM CONFIGURATIONS

The Tandem NonStop systems (illustrated on the reverse side of this page) are packaged especially for critical on-line transaction-oriented applications where high reliability and low-program overhead are essential. All systems feature dual processors with Tandem's unique DYNABUS for high-speed inter-processor communications, and redundant dual-port I/O channels for input/output resiliency. Each individual processor module contains two microprogrammed processors: one dedicated to programs and one dedicated to input/output control and data transfers. In addition, each central processor provides Power Fail/Auto Restart, an Interval Timer, Hardware Multiply/Divide, DMA, Dynamic Memory Mapping, Virtual Memory control, and a Bootstrap Loader.

TANDEM T16/212-1 SYSTEM

The Tandem T16/212-1 System consists of a dual processor package utilizing core memory modules. Each of the two processors contains 192K bytes of 800 nanosecond core memory with parity. The memory is arranged in 64K byte modules (32K words of 17 bits each). The T16/212-1 may be easily expanded in the field to 448K bytes of memory on each processor. The T16/212-1 I/O System provides high speed dual-port I/O Channels with block-multiplexed DMA. The data rate of each I/O Channel is 2.5 MBytes/second. The system cabinet provides 14 vacant slots for expansion of I/O Controllers.

TANDEM T16/244-1 SYSTEM

The Tandem T16/244-1 System consists of a dual processor package utilizing semiconductor (MOS) memory modules. Each of the two processors contains 192K bytes of 500 nanosecond semiconductor memory with error detection and correction. The memory is arranged in 96K byte modules (48K words of 22 bits each, 16 for data, 6 for ERCC). The T16/244-1 may easily be expanded in the field to contain two additional processors for a total of four processors and each processor may be expanded to 512K bytes of memory. Battery backup is supplied as a standard feature on semiconductor memory. The T16/244-1 I/O System provides high speed dual-port I/O Channels with block multiplexed DMA. The data rate of each I/O Channel is 4.0 MBytes/second. The System Cabinet provides 14 vacant slots for expansion of I/O Controllers.

SUBSYSTEM CONFIGURATIONS

All Tandem packaged systems are supported with a wide selection of peripheral subsystems to meet the heavy demands of diversified applications and high-volume data bases.

Disc Subsystems — Because disc requirements are highly application dependent the user will add disc controllers and drives as needed. The user can choose from a wide variety of disc controllers and drives including 10MB, 50MB and

The following system modules may be added to TANDEM packaged systems. The modules may also be combined to configure systems other than the standard packaged systems if desired.

PRODUCT NUMBER	DESCRIPTION
SEMICONDUCTOR MEMORY PROCESSORS	
T16/1403	General Purpose Processor consisting of: Two (2) pipelined microprocessors, one for programs and one for I/O. Complete DMA only I/O system (4.0M bytes/sec). Virtual Memory control. Memory mapping and protection for up to 512K bytes of main memory. Hardware MPY/DIV. Power-fail/auto-restart. Bootstrap Loader. Interval Timer. Control Panel. Dual interprocessor message hardware. Provision for up to 32 I/O controllers. 122 instructions including string manipulation and double word arithmetic. 96K bytes (48K words) of semiconductor memory arranged as 22 bit words (16 data bits and 6 error detection/correction bits) <i>all</i> single bit errors are corrected and <i>all</i> double bit errors are detected. Memory cycle time of 500 nsecs. Battery Backup for semiconductor memory.
CORE MEMORY PROCESSORS	
T16/1102	Same as T16/1402 with the following exceptions: 1) Core memory (cycle time 800 nsecs) is used in place of semiconductor memory. Core memory utilizes one parity bit per two bytes. 2) The I/O channel speed is reduced to 2.5 Megabytes/sec, memory capacity is 64K bytes (32K words).
MEMORY MODULES	
T16/2102	Core Memory module, consists of a 64K byte memory plane with a read access time of 500 ns and a cycle time of 800 ns. The module is arranged as 32K words of 17 bits (16 data bits one parity bit) up to 7 of the modules may be controlled by a single processor.
T16/2402	Semiconductor memory module, consists of 64K byte memory module with a cycle time of 500 nsecs. The module is arranged as 32K words of 22 bits, 16 data bits and 6 error detection/correction bits. The module enables detection and correction of <i>all</i> single bit errors and detection of <i>all</i> double bit errors. Up to 8 of these modules may be controlled by a single processor.
T16/2403	Semiconductor memory module, consists of 96K byte memory module with a cycle time of 500 nsecs. The module is arranged as 48K words of 22 bits, 16 data bits and 6 error detection/correction bits. The module enables detection and correction of <i>all</i> single bit errors and detection of <i>all</i> double bit errors. Up to 5 of these modules may be controlled by a single processor.
TERMINAL SUBSYSTEMS	
T16/5202	Synchronous Controller, dual channel connected and may be powered from either processor to which it is connected. Will control from 1 to 4 synchronous communication lines, either point-to-point or multidrop. A single line can run up to 56K bps. The aggregate data rate for all four lines cannot exceed 160K bps. The controller performs all character translation (ASCII or EBCDIC), Block Check Character generation, and autopolling. Includes loop-back test module, down-line loading capability.
T16/6301	Asynchronous Controller, dual channel connected and may be powered from either processor to which it is connected. Will control up to two terminal lines either hard-wired or modem connected. Line speed is programmable from 50 to 19.2K bps. Accommodates up to two extensions (see T16/6302). Includes loop-back test cable.
T16/6302	Asynchronous Extension Board, provides additional control for 15 asynchronous lines. Speed of each line is programmable from 50 to 19.2K bps. Each line can be hard-wired or modem connected. Pre-requisite: T16/6301.
T16/6001	Console Subsystem, consists of: 1. Dual channel connected controller, T16/6301 with (1) port for a console device; 2. One (1) T16/6604 30 cps, 132 column hard-copy console, C/L connected.
T16/6603	Terminal, Hard Copy, 30 cps, 132 columns, includes 25' cable, RS232 interface.
T16/6604	Terminal, Hard Copy, same as T16/6603 except 20mA current loop interface.
T16/6401	Terminal, CRT, character mode, 24 lines x 80 characters, speed switch selectable 110-19.2K bps, local or modem attachment, includes 25' cable, RS232 interface.
T16/6402	Terminal, CRT, character mode, 24 lines x 80 characters, speed switch selectable 110-19.2K bps, includes 25' cable, 20mA current loop interface.
T16/6511	Terminal, CRT, character or page mode, local editing and function keys, 24 lines x 80 characters, protected, unprotected, full/half bright reverse, blinking, speed switch selectable 110-19.2K bps, 25' cable, RS232 interface, local or modem attachment.
T16/6512	Terminal, CRT, character or page mode, local editing and function keys, 24 lines x 80 characters, protected, unprotected, full/half bright, reverse, blinking, speed switch selectable 110-19.2K bps, 25' cable, 20mA current loop interface.
T16/6552	Terminal, CRT, same as T16/6511 except uses Polling Protocol for Multidrop Communications Lines.
DISC SUBSYSTEMS	
T16/3102	Disc Controller (Small Discs), dual channel connected, can be powered from either processor, can control 1 to 4 drives (any mix of T16/4101 or T16/4102), each with separate connection (signal & data cable for each drive), uses 2314 recording technique.

TANDEM SYSTEM MODULES

TANDEM COMMUNICATION CONTROLLERS

SYNCHRONOUS CONTROLLER

FEATURES

- Supports four lines per controller with speeds up to 56K Bps per line
- Full or Half Duplex synchronous operation
- Automatic Generation and Detection of Block Check Characters with support for VRC, LRC and CRC16 Modes of Operation
- Automatic Code Translation to ASCII & EBCDIC
- Automatic Polling Capability provided by the Controller for Multipoint Environments
- DMA Access to Main Memory
- Supports Transparency and Auto-insertion of "DLE" and "SYN" characters
- Supports Bell-type 201, 203, 208 and 209 modems

INTRODUCTION

The T16/6201 Synchronous Communications Adapter, utilizing microprogrammed technology, provides a high speed, intelligent interface for synchronous communications environments. When used with Tandem's ENVOY Data Communication Manager a powerful, yet flexible data communications system can be designed with a minimum of effort.

All input/output is directly between main memory and the controller. CPU processing is interrupted only when a transfer of a message completes or a line error condition is encountered.

The controller generates and appends check character information upon transmission and performs error checking upon reception.

The controller automatically recognizes the transmission of transparent text. If the control sequence [DLE-STX] is encountered at the beginning of a message transfer, the controller enters the transparent text mode. In this mode, the controller will insert a control [DLE] character in front of a data [DLE] character when transmitting. Conversely, when receiving transparent text, the controller will delete a control [DLE] character in front a data [DLE] character. The controller remains in the transparent text mode until an [ETB], [ETX], or [EOT] control sequence is encountered.

The controller's internal character code, as far as detecting control character sequences is concerned, is ASCII. The controller has the capability to translate

ASCII code to EBCDIC code upon transmission and to translate EBCDIC code to ASCII code upon reception. Because of this capability, application processes need deal only with the ASCII character set.

The polling of multipoint stations is, for the most part, handled by the controller. ENVOY formats a polling list (on behalf of the application process) for the controller to use to poll the multipoint tributary stations. ENVOY then commands the controller to begin polling. CPU processing is interrupted only when a polled station responds.

The controller has the capability to recognize if a line is being polled or selected. For each line, the controller stores the first byte of the station's polling address and the first byte of the station's selection address. Only when the line is polled or selected and the corresponding poll or select byte matches is CPU processing interrupted.

OPERATION

Each line can be configured dynamically for

- Translate Enable
- Transparent Text Capability
- Full or Half Duplex Operation
- Polling Address
- Selection Address

In addition, instructions are available to;

- Answer/Hang Up the Phone
Sets or Clears the modem control signal Data Terminal Ready (DTR).
- Initiate Write
Initiates a Write operation
- Initialize Read Control
Sets up a Read Operation which will follow an automatic line turnaround
- Initiate Read
Initiates a Read operation
- Terminate Read
Terminates a Read Continuous operation
- Stop Polling
Terminates auto polling for a line and interrupts

ASYNCHRONOUS CONTROLLER

FEATURES

- 2 to 32 asynchronous lines per controller with line speeds from 50 to 19.2K Bps
- Point-to-Point or Multipoint Modes of Operation
- DMA Access to Main Memory on all I/O Transfers
- EIA (RS232) or Current Loop Interfaces
- Modem support for Bell-type 103 and 202 (including reverse channel)
- Each line individually programmable with respect to;
 - Baud Rate.
 - Character Size.
 - Computer Parity Generation.
 - Computer Parity Checking.
 - Connection Type.
 - Enable/disable Checking for Signal Characters.
 - Half-Duplex Modem Turn-around Character(s).
 - Read Completion on ETX Character.
 - Default Transfer Mode.
 - Conversational Mode Line Termination Character.
 - Conversational Mode Automatic Linefeed on Input.
 - Conversational Mode Backspace Type.
 - Conversational Mode Carriage Return/Line Feed Delay.
 - Conversational Mode Forms Control Delay.
 - Page Mode Page Termination Character.
 - Page Mode Psuedo-Polling Trigger Character.

INTRODUCTION

The T16/6301/6302 Asynchronous Controller provides a flexible interface for all types of asynchronous communications. For point-to-point applications the standard GUARDIAN I/O Subsystem will support any RS232 or Current Loop Terminal by merely configuring the line in SYSGEN. Standard I/O

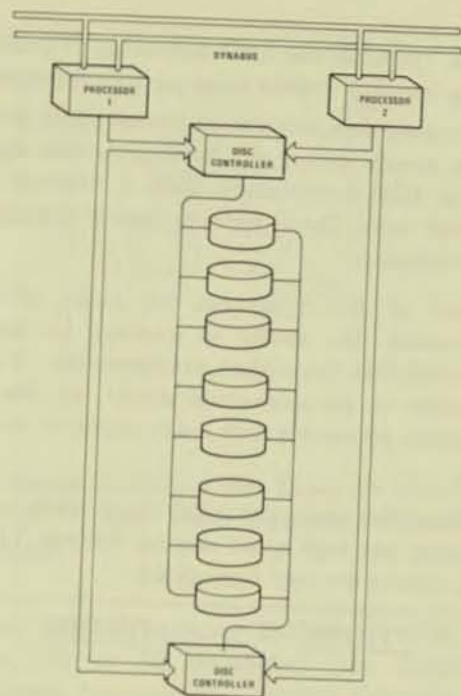
calls (READ, WRITE, WRITEREAD) provide access to the terminal. For multipoint applications ENVOY provides the ability to interface polling terminals such as Tandem's T16/6552 CRT Polling Terminal or TI's TINET Data Entry Pads. The Asynchronous Controller is sufficiently fast to support 32 lines all running at 19.2K Bps.

Both conversational and page mode terminals are supported. Conversational terminals can specify, as a SYSGEN parameter, the characters which will be interpreted as line termination, line cancel, backspace and end-of-file. Page mode terminals can specify, as a SYSGEN parameter, the characters which will be interpreted as Page Termination and Pseudo-poll Triggers.

OPERATION

Instructions are available to;

- Initiate a Read
- Initiate a Write
- Initiate a Write Read
 - Writes data to a terminal, then waits for data to be read back from terminal
- Setmode
 - Sets or clears
 - single spacing
 - auto linefeed
 - conversation/page mode
 - signal characters
 - parity checking
 - break ownership
 - access mode
 - read termination on ETX
 - read termination on signal character
- Control
 - Used for form control and modem connect/disconnect



TANDEM DUAL-PORT DISC SUBSYSTEMS

- Field Proven Disc Drives for Transaction-oriented Systems
- High Capacity
- High Performance
- Dual-Ported Disc Drives for Data Integrity
- Reliable Servo Track Positioning for Accuracy
- Intelligent, microprogrammed Disc Controller
- Compact, Stand-alone Packaging

SUBSYSTEM CONFIGURATION

The Tandem Dual-port Disc Subsystem provides high capacity, high performance, high availability and high data integrity for on-line, transaction-oriented systems. When used on the Tandem NonStop (tm) Computer System these disc subsystems provide the most cost effective solution for transaction-oriented applications where data integrity is of paramount importance.

The Model T16/3105 Disc Controller provides a high degree of flexibility and expandability. The disc controller is actually an intelligent, "backend" disc processor capable of supporting up to eight disc drives per controller. Because the controller is microprogrammed a wide variety of disc technologies and data base functions can be supported.

The T16/3105 Disc Controller connects to two (2) I/O channels simultaneously and may be powered from either processor in the event of a single processor failure. Thus, high availability is insured to the controller.

The T16/4103, T16/4104 and T16/4105 Disc Drives provide high capacity and performance through the use of new disc technologies. In addition high availability and data integrity are provided through dual-port connections. Each disc drive can connect to two (2) controllers simultaneously.

T16/3105 DISC CONTROLLER

The T16/3105 Disc Controller is used to interface up to eight disc drives consisting of any mix of T16/4103, T16/4104 or T16/4105 Disc Drives. The controller provides high availability to the drives through its dual-channel connection pioneered by Tandem in its other controllers.

Significant features of the controller include:

- (1) Dual-channel connection
- (2) 4K RAM Buffer for buffering logical records
- (3) Dual recording or the ability to transfer the buffer to one or more disc units without having to re-transmit on the I/O bus
- (4) Support for dual-access disc drives
- (5) Read without I/O transfer which allows one disc to be copied to another without involving the I/O bus

The disc controller can be configured so that two controllers are connected to the same disc unit. Up to eight drives can be daisy-chained on one controller. With the dual-ported disc option daisy-chain failures will not cause a loss of the data base, a loss of data base integrity, or a loss of an access path to the data base.

T16/4103, 4104 AND 4105 DISC DRIVES

The T16/4103, 4104 and 4105 are stand-alone disc drives using the latest in disc technologies to provide high capacity and performance.

The T16/4103 uses 3330 technology to provide a storage capacity of 160M bytes per pack (formatted). Performance data includes an average access time of 28 ms, average latency of 8.35 ms, and a data transfer rate of 806K bytes/second. Data is recorded on an eleven-high pack. Servo tracking insures accuracy of track following.

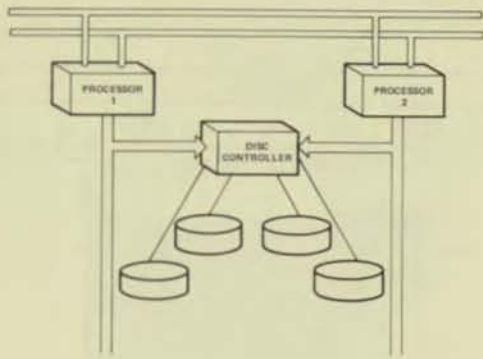
The T16/4104 uses SMD technology to provide a storage capacity of 240M bytes per pack (formatted). Performance data includes an average access time of 28 ms, average latency of 8.35 ms, and a data transfer rate of 1.2M bytes/second. Data is recorded on an eleven-high pack. Servo tracking insures accuracy of track following.

The T16/4105 uses SMD technology to provide a storage capacity of 64M bytes per pack (formatted). Performance data includes an average access time of 30 ms, average latency of 8.35 ms, and a data transfer rate of 1.2M bytes/second. Data is recorded on a five-high pack. Servo tracking insures accuracy of track following.

Each of the drives has an option of being dual-ported, the ability to connect to two (2) T16/3105 Disc Controllers simultaneously. If a disc controller or the daisy-chain should fail, the other controller connection will insure access to the data base.

Each drive uses a linear DC motor (voice coil) for accurate and high speed seeking. Reliable TTL and ECL circuits are used throughout.

SPECIFICATIONS	T16/4103	T16/4104	T16/4105
RECORDING CAPACITY			
Capacity (Unformatted)	200M Bytes	300M Bytes	80M Bytes
Capacity (Formatted)	160M Bytes	240M Bytes	64M Bytes
Recording Mode	MFM	MFM	MFM
Recording Density	4040 BPI	6060 BPI	6038 BPI
Tracks per Surface	808 + 7 alt	808 + 7 alt	808 + 15 alt
Tracks per Inch	384	384	384
Data Surfaces	19	19	5
Servo Surfaces	1	1	1
PROCESSING SPEED			
Data Transfer Rate	806K Bytes/Sec	1.2M Bytes/Sec	1.2M Bytes/Sec
Spindle Speed	3600 RPM	3600 RPM	3600 RPM
Latency (average)	8.35 ms	8.35 ms	8.35 ms
ACCESS SPEEDS			
Minimum (one track)	10 ms	10 ms	6 ms
Average	28 ms	28 ms	30 ms
Maximum	55 ms	55 ms	55 ms
DISC PACKS			
(all packs must be flag free)	3M 936/11 CDC 9882 Memorex Mark XI Dysan	Dysan (300MB)	CDC 9877
PHYSICAL			
Dimensions (D x W x H)	34 x 19 x 38	34 x 19 x 38	34 x 19 x 34
Weight	465 lbs	465 lbs	243 lbs
ELECTRICAL			
Voltage	208V, 60 Hz 220/240V, 50 Hz	208V, 60 Hz 220/240V, 50 Hz	120V, 60 Hz 220/240V, 50 Hz
Phases	Three	Three	Single
Operating Amps	6.4	6.4	8.2
ENVIRONMENTAL			
Temperature (operating)	60-90 F	60-90 F	59-90 F
Humidity (operating)	20-80%	20-80%	20-80%
Heat Dissipation	5800 BTU/hr	5800 BTU/hr	1800 BTU/hr



TANDEM SINGLE - PORT DISC SUBSYSTEMS

- **Tough, Service-Engineered Disc Drives for Critical NonStop™ Real Time Transaction-Oriented Applications**
- **Low-Cost High Performance Medium- to High-Capacity (10 to 50 Megabytes) Disc Drives with Proven Dependability, System Versatility and Modular Maintainability**
- **Economical High-Speed Reliability with Advanced Hybrid and CMOS Technology**
- **Fast Data Transfer Rate (312K Bytes/Second) for Efficient and Economical Data Processing**
- **Wide Choice of Optional Features to Facilitate NonStop™ System Configuration or Expansion in the Field.**

SUBSYSTEM CONFIGURATION

The Tandem Disc Subsystems were particularly selected to provide low-cost, high reliability, high data capacity, fast data transfer and easy as well as economical maintainability. All these parameters are crucial to the exacting requirements of a non-stop transaction-oriented environment.

A standard Disc Subsystem consists of a Disc Controller and one or more Moving Head Disc Drives. The Controller features two independent I/O Channel ports and may be powered from either system processor in the event of a single processor failure. In addition, a disc failure cannot corrupt the system processor's memory since the address and count words are held in the I/O processor — not the disc controller.

The Model T16/4101 is a 10MB capacity drive using moving heads while the Model T16/4102 is a 50MB capacity drive using moving heads. The Model T16/3101 Disc Controller can accommodate 1 to 4 disc drives in any mix of T16/4101 and T16/4102 Disc Drive configurations. Each drive connected to the controller has its own signal and data cable. No daisy chain is used.

MODEL T16/4101 DISC DRIVE

The T16/4101 Disc Drive is a low-cost, medium capacity (10M Byte) disc drive featuring a 312KB transfer rate, high-speed random-access and pedestal-mounted cartridge disc drive. The unit is comprised of one fixed platter of 5MB and one removable top-loading platter of 5MB with average seek and latency times of 35 and 12.5 milliseconds, respectively. This rugged unit also features: a service-oriented modular design which eliminates all field adjustments, and an automatic cleaning system which purges and cleans the disc area to reduce costly preventive maintenance.

MODEL T16/4102 DISC DRIVE

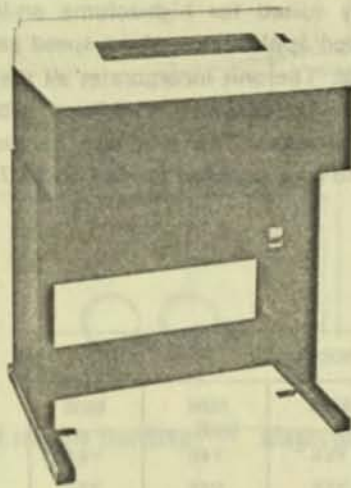
The Model T16/4102 Disc Drive is a moderately-priced, high-capacity (50M Bytes) disc drive ideally suited for large data bases. This high-performance unit is a random-access, pedestal-mounted disc pack drive and is directly I/O interchangeable with the Model T16/4101 cartridge disc drive (described above) when the Model T16/3102 Controller is installed. The unit uses CDC^R 9873 Disk Packs, or equivalent, for the storage and retrieval of data. Each removable pack contains 11 discs. Average seek time is 35 ms. Average latency is 12.5 ms. The transfer rate is 312KB.

SPECIFICATIONS

PARAMETER	MODEL	
	4101	4102
RECORDING CAPACITY		
Capacity (Maximum Megabytes, Formatted)	10MB	50MB
Recording Mode	Double Freq.	Double Freq.
Recording Density (BPI - Nominal)	2200	2200
Tracks Per Surface (Maximum)	408	406
PROCESSING SPEED		
Data Transfer Rate (K Bytes/Second)	312	312
Spindle Speed (RPM)	2400	2400
ACCESS TIME (Direct Seek)		
Full Stroke (Milliseconds)	60	70
Average (Milliseconds)	35	35
One-Track (Milliseconds)	7	10
DISC CARTRIDGE/PACK		
Number of Discs	1	11
Usable Surfaces	2	20
Pack Type	5440 equiv.	2314-2 equiv.

PARAMETER	MODEL	
	4101	4102
RECORDING HEADS	4	20
PHYSICAL:		
Dimensions (L x W x H Inches)	29-3/4 x 18-1/2 x 34	32-1/2 x 27-1/2 x 38
Weight (Pounds)	252	700
ELECTRICAL:		
Voltage	1	2
	50 ±1.0	50 ±1.0
Frequency (Hz) Disc	60 ±1.0	60 ±1.0
Power Rating (Operating AMPS)	4.6	5.0
Power Rating (Surge AMPS)	9.2	22.0
ENVIRONMENTAL:		
Operating Temperature (°F)	60 - 90	60 - 90
Operating Humidity (Non-Condensing)	10 - 90%	10 - 80%
Non-Operating Temperature (°F)	-30 to 150	-30 to 150
Non-Operating Humidity (Non-Condensing)	5 - 95%	5 - 95%

- Notes: (1) Integral 100 - 250 VAC ±10% at 50 or 60 Hz (Standard), single phase or two phase.
 (2) 208 VAC ±10% at 60 Hz or 220 VAC ±10% at 50 Hz (Standard). 200, 230, 240, or 250 VAC ±10% at 50 or 60 Hz (Optional), three phase.



TANDEM LINEPRINTER SUBSYSTEMS

- Economical Medium to High-Speed (300 to 1500 lpm) Field-Proven Performance
- Wide Selection of Vertical and Horizontal Form Formats
- Versatile ASCII or OCR Compatible Character Sets (64 or 96 Characters in 132 columns)
- Rugged Solid-State Reliability and Easy Modularized Maintainability with Self-Contained Service Aids

SUBSYSTEM CONFIGURATION

The Tandem Lineprinter Subsystems were especially selected to offer the user a wide choice in price/performance trade-offs to meet individual transaction-oriented application needs. The standard subsystems consist of one or more Series T16/5500 Lineprinters and a Model T16/3302 Lineprinter Controller. The controller incorporates Tandem's NonStop™ feature of two independent I/O ports. In the event of a single processor failure, the controller automatically switches over to the second processor. In addition, a Lineprinter failure cannot corrupt a processor since the critical control parameters are held in the I/O processor — not the Lineprinter Controller.

The Tandem Lineprinter Subsystems were human-engineered to provide easy on-line non-stop maintenance and expandability in the field. The rugged

modularity concept employed ensures that these subsystems perform their function with a minimum of preventive and/or corrective maintenance downtime.

MODEL T16/5502 LINEPRINTER

The Model T16/5502 Lineprinter is a medium-priced, low-speed (300 lpm) drum unit featuring a 12-channel Vertical Format Unit (VFU). The VFU utilizes IBM-compatible carriage tapes. The unit provides 132 columns of either 64 or 96 (optional) ASCII characters with a spacing of 10 characters/inch horizontally and 6 or 8 lines/inch vertically. Optionally, the unit features OCR character set with appropriate fonts. For service and maintenance, an optional self-test code generator is also available. This unit accepts up to 6-part fanfold edge-punched forms 4 to 16.75 inches wide.

MODEL T16/5503 LINEPRINTER

The Model T16/5503 Lineprinter is a low-priced, medium speed (600 lpm) drum unit incorporating the same features as the Model T16/5502 described above. The minimum printing rate is maintained at 436 lpm even when the optional 96 character set is incorporated.

MODEL T16/5504 LINEPRINTER

The Model T16/5504 Lineprinter is a moderately-priced, medium-speed (900 lpm) drum unit identical to the Model T16/5502 described above. This rugged, compact unit also accepts up to 6-part forms.

MODEL T16/5505 LINEPRINTER

The Model T16/5505 Lineprinter is a high-performance, high-speed (1500 lpm) train unit ideally suited for high-volume on-line transaction-oriented applications where speed and reliability are critical. The unit incorporates all the features of the Model T16/5502 and, in addition, provides a powered paper stacker. This unit also handles up to 6-part fanfold edge-punched forms 4 to 16.75 inches wide.

OPTIONS

DESCRIPTION	MODEL			
	5502	5503	5504	5505
ASCII 96-character Set	YES	YES	YES	YES
OCR Character Set	YES	YES	YES	YES
Vertical Format Unit (VFU)	YES	YES	YES	YES
Elapsed Time Meter	YES	YES	YES	YES
Self-Test Feature	YES	YES	YES	YES
Static Eliminator	YES	YES	YES	YES
Pedestal Mount	STD	STD	STD	STD

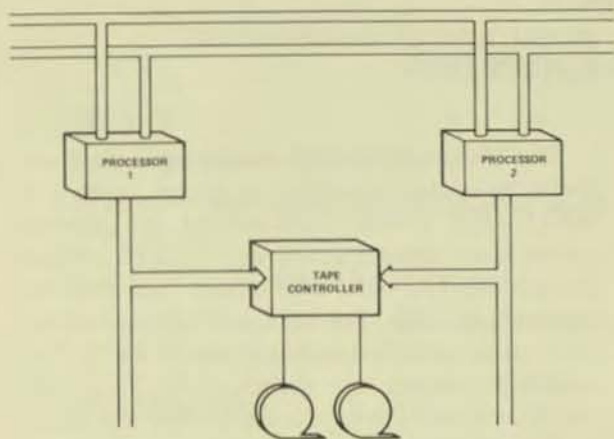
Notes: ① Standard

SPECIFICATIONS

PARAMETER	MODEL			
	5502	5503	5504	5505
PHYSICAL:				
Dimensions (L x W x H)	26 x 33 x 45	26 x 33 x 45	36 x 33 x 45	24 x 48 x 46
Weight (Pounds)	340	370	420	800
ELECTRICAL:				
Voltage (Single Phase VAC)	1	1	1	230 ± 10%
Frequency (Hz)	60 ± 2.0	60 ± 2.0	60 ± 2.0	60 ± 3.0
Power Rating (Watts)	525	680	825	50 ± 3.0
ENVIRONMENTAL:				
Temperature (°C)	10 - 37	10 - 37	10 - 37	10 - 43
Humidity (Non-Condensing)	30 - 80%	30 - 80%	30 - 80%	10 - 90%
PRINTING:				
Type	Drum	Drum	Drum	Drum
Speed (64 Character Set lpm)	340	600	900	1500
Speed (96 Character Set lpm)	240	436	660	1220
Columns (Standard)	132	132	132	132
Horizontal Spacing (Characters/Inch)	10	10	10	10
Vertical Spacing (Lines/Inch)	6/8	6/8	6/8	6/8
Paper Widths (Inches)	4 - 16.75	4 - 16.75	4 - 16.75	4 - 16.75
TRANSMISSION:				
Code (Standard)	ASCII	ASCII	ASCII	ASCII

Note: ① 100, 115, 125, 200, 220 or 240 ± 10%

TANDEM MAGNETIC TAPE SUBSYSTEMS



- Flexible and reliable NonStop^(tm) Magnetic Tape Controllers
- 800 bpi NRZI and/or 1600 bpi PE Magnetic Tape Drives
- 45 ips or 125 ips Magnetic Tape Drives
- 9 track recording, IBM and ANSI compatible
- Ease of serviceability

SUBSYSTEM CONFIGURATION

The Tandem Magnetic Tape Subsystem provides reliability and performance for Non-Stop transaction-oriented systems requiring magnetic tape capability. A basic Magnetic Tape Subsystem consists of a NonStop magnetic tape controller and one or more magnetic tape drives. Two controllers are available — the T16/3201 Magnetic Tape Controller handles 800 bpi NRZI formats and the T16/3202 Magnetic Tape Controller handles both 800 bpi NRZI and 1600 bpi Phase Encoded (PE) formats.

Both controllers are dual-ported. Each connects to two I/O channels simultaneously so that in the event of a single failure (processor, power supply or I/O channel) the other port can maintain a data and control path to the magnetic tape drives.

Each controller can control one or two drives.

Three tape drives are available. They can be configured in 800 or 1600 bpi, NRZI or PE recording formats, and have tape speeds of 45 or 125 ips.

The drives were designed to provide ease of maintenance. All major service functions can be performed from the front door opening.

MODEL T16/5101 MAGNETIC TAPE DRIVE

The Model T16/5101 Magnetic Tape Drive is a moderately priced, medium speed direct drive unit. Tape speed is 45 ips using 800 bpi NRZI recording formats. The effective data rate is 36KB.

The drive uses 9 track recording and is compatible with IBM and ANSI standards. The drive uses standard 1/2" tape and can accommodate reels up to 10.5 inches in diameter using standard IBM hubs.

Data integrity is maintained by the use of VRC, LRC and CRCC checking.

MODEL T16/5103 MAGNETIC TAPE DRIVE

The Model T16/5103 Magnetic Tape Drive is a moderately priced, medium speed direct drive unit. Tape speed is 45 ips. Recording mode is dual-density using either 1600 bpi Phase Encoded (PE) recording formats or 800 bpi NRZI recording formats. The effective data rate is 72KB or 36KB depending on the operator-settable switch.

TANDEM MAGNETIC TAPE SUBSYSTEMS

The drive uses 9 track recording and is compatible with IBM and ANSI standards. The drive uses standard 1/2" tape and can accommodate reels up to 10.5 inches in diameter using standard IBM hubs.

Data integrity is maintained by the use of VRC, LRC and CRCC checking.

MODEL T16/5104 MAGNETIC TAPE DRIVE

The Model T16/5104 Magnetic Tape Drive is a moderately priced, high speed vacuum column unit. Tape speed is 125 ips. Recording mode is dual-density using either 1600 bpi Phase Encoded (PE) recording formats or 800 bpi NRZI recording formats. The effective data rate is 200KB or 100KB depending on the operator-settable switch.

The drive uses 9 track recording and is compatible with IBM and ANSI standards. The drive uses standard 1/2" tape and can accommodate reels up to 10.5 inches in diameter using standard IBM hubs.

Data integrity is maintained by the use of VRC, LRC and CRCC checking

SPECIFICATIONS

Characteristics	5101	5103	5104
Tape Speed (ips)	45	45	125
Recording Density (bpi)	800	800/1600	800/1600
Transfer Rate (cps)	36KB	36/72KB	100/200KB
Recording Format	9-track NRZI	9-track NRZI/PE	9-track NRZI/PE
Tape Reels	1/2" std IBM hub	1/2" std IBM hubs	1/2" std IBM hubs
Rewind Time (ips)	300	300	375
Transport	Tension Arm	Tension Arm	Vacuum Column
Checking	LRC, VRC, CRCC	LRC, VRC, CRCC	LRC, VRC, CRCC

MODEL T16/3101 MAGNETIC TAPE CONTROLLER

The Model T16/3101 Magnetic Tape Controller provides the ability to attach up to two NRZI tape drives. Connections are starred rather than daisy-chained to insure maximum utilization. The controller allows each drive to operate at 800 bpi using NRZI recording formats as specified in ANSI X3.22-1973. The controller allows tape speeds of 45, 75 or 125 ips. The controller conforms to IBM standards for 9 track digital recording.

MODEL T16/3102 MAGNETIC TAPE CONTROLLER

The Model T16/3102 Magnetic Tape Controller provides the ability to attach up to two NRZI and/or PE tape drives. Connections are starred rather than daisy-chained to insure maximum utilization. The controller allows each drive to operate at 800 bpi using NRZI recording formats as specified in ANSI X3.22-1973 and/or 1600 bpi using Phase Encoded recording formats as specified in ANSI X3.39-1973. The controller allows tape speeds of 45, 75 or 125 ips. The controller conforms to IBM standards for 9 track digital recording.

TANDEM UNIVERSAL INTERFACE

FEATURES

- Provides 8/16 line parallel interface
- Operates half-duplex, bi-directional at up to 4 MBytes/sec
- Uses both TTL and Differential logic
- Contains dual-channel connected logic for fault-tolerant operations

INTRODUCTION

The T16/3401 Universal Interface (UI) provides the ability to interface custom equipment to the T16 Computer System. The UI is capable of attaching two devices that have 8 or 16 line parallel data interfaces to the Tandem 16 Computer. The Universal Interface (UI) provides a device data path that is buffered (16 words deep), bi-directional, and capable of operating in half-duplex mode at a sustained data transfer rate of up to 4 Mbyte per second (depending on the channel configuration). It interfaces to one device over positive or ground true TTL lines for short distances (up to 25 ft) and to the second device over differential lines for longer distances (up to 500 ft). The data path between either or both of the two devices and the UI can be either one byte (8 bits) or one word (16 bits) wide.

Configuration of the UI is accomplished by software and by configuration jumpers in the connector hood.

DATA PATH

The UI can be used to interface both input and output devices requiring a data path width of either one byte (8 bits) or one word (16 bits) in half-duplex mode. Data is transferred between the UI and the channel in bursts consisting of several words. For input or output devices that operate in word mode, the UI passes data words directly to and from the device on the DATA 0:15 interface lines (Figure 1). For output devices that operate in byte mode, the UI disassembles data words from the channel and transfers data bytes to the device. For input devices that operate in byte mode, the UI assembles data bytes from the device and transfers data words to the channel. In byte mode, data is transferred to and from the device on the DATA 8:15 interface lines (Figure 1). The width of the

device data path is program controlled, such that a device with a one byte wide data path can have additional control signals sent to it on the remaining eight data lines. For example, some terminals have a one byte data path but require 12 bits for cursor addressing.

DATA TRANSFER

Data transferred between the UI and a device may be strobed in a handshake sequence or pulsed without a handshake. In Handshake Mode, the device response can either lead or follow the UI data strobe. In Pulse Mode, the UI data strobe is used to pulse data to the device on write operations with no response from the device, while for read operations, the device response is used to pulse data to the UI with no response from the UI.

DATA PARITY

Odd parity is generated and checked for each data word that is transferred between the channel and the UI. The parity that exists between the UI and each device is defined by configuration jumpers in the connector hood. The jumpers select odd, even or no parity.

DATA TRANSFER TERMINATION

A data transfer sequence between a device and the I/O channel is initiated with an EIO instruction executed by the CPU program. When a specified number of bytes have been transferred, the sequence is terminated in one of three ways: 1) by the I/O channel, 2) by the device, or 3) by the UI.

UI STROBES

The UI is capable of transmitting up to six strobe signals to a device, one of which is the Data Strobe. The other five strobes can be transmitted individually or in combinations under program control.

Two strobe signals are provided that can be pulsed before and after data transfer. The "before" strobe (Strobe 4) is 1 μ s in duration and the "after" strobe (Strobe 5) is 10 μ s in duration.

Two strobe signals (Strobe 2 and Strobe 3) can be held active during the entire data transfer.

One strobe signal (Strobe 1) can be gated during data transfer by an external trigger from the device.

In a ground-true TTL interface, Strobes 1 through 5 can be wired together in the connector hood in any combination to "or" their respective functions.

UI STATUS

Device status is presented to the UI on eight (8) interface lines, STATUS 8:15. These signal lines can be jumpered to the STATRES 0:8 pins in the connector hood to form the desired status results.

Controller (UI) and device status words are transferred to the CPU program in the CPU's register stack in response to EIO, IIO and HIO instructions.

UI DEVICE ADDRESSING

All devices attached to a processor I/O channel are assigned unique Physical Unit Addresses. The Physical Unit Address is 8 bits in length and consists of a 5 bit Controller Number and a 3 bit Unit Number. The Controller Number is defined by switches on the UI board; Unit Number 0 is the device attached to the TTL interface and Unit Number 1 is the device attached to the differential interface.

UI COMMANDS

Commands are issued to the UI by executing EIO (Execute Input/Output) instructions in the CPU program. The UI command set is comprised of the following commands:

- SENSE
- TAKE OWNERSHIP
- DISABLE PORT
- SET COUNT
- READ
- WRITE

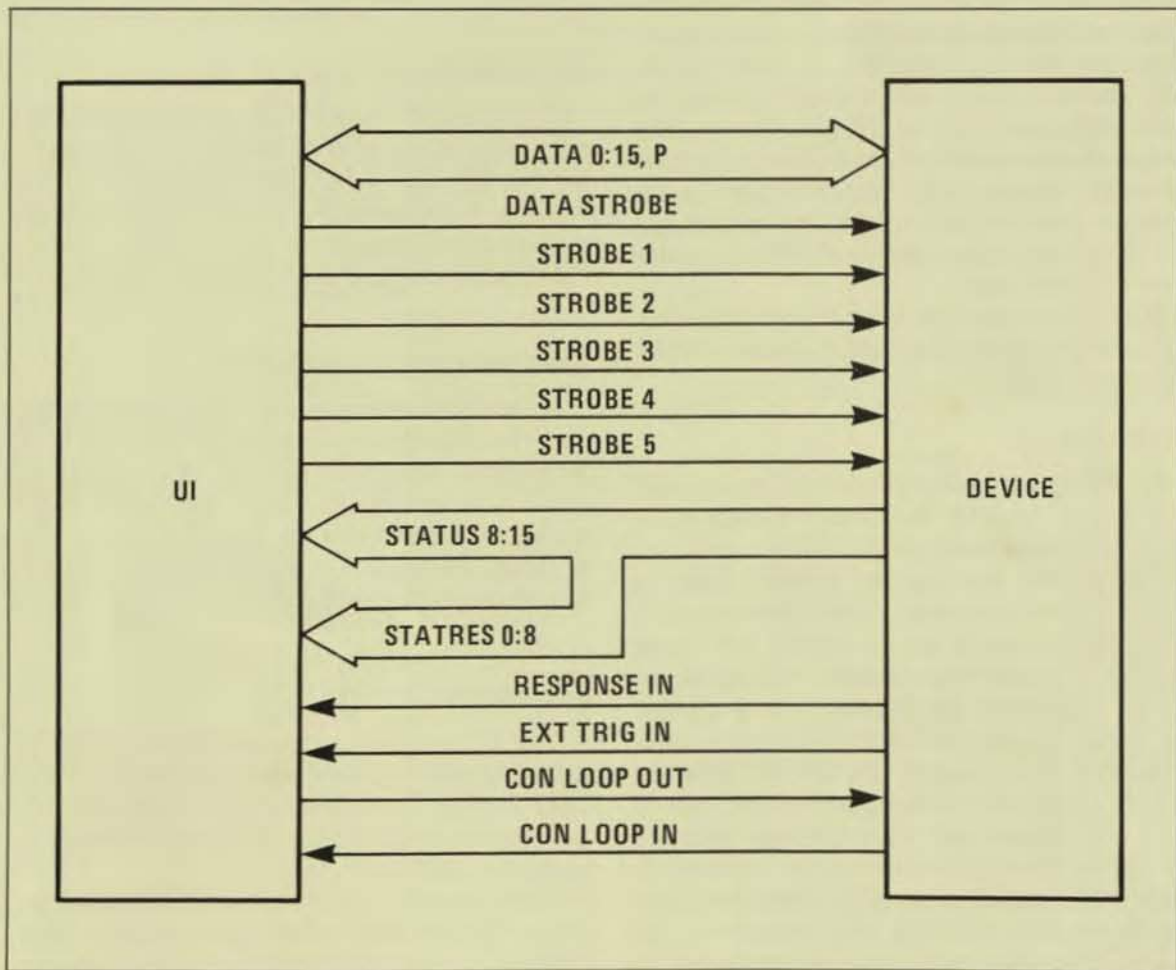


Figure 1. UI/Device Interface

TANDEM

6520

MULTI-PAGE DISPLAY TERMINAL

The Tandem 6520 block mode terminal represents a significant advance in state-of-the-art technology. Designed specifically for use in the on-line transaction environment, the 6520 underscores Tandem's commitment to making transaction processing easier and more reliable.

The Tandem 6520 features include:

- MEMORY PARITY for data integrity
- MULTIPLE DISPLAY PAGES for high data throughput and reduced line utilization
- FULL COMPLEMENT OF VIDEO AND DATA ATTRIBUTES, EDITING AND PROGRAM FUNCTION KEYS for ease of operation
- CONVERSATIONAL AND BLOCK MODES
- SYNCHRONOUS AND ASYNCHRONOUS PROTOCOLS for simplified communications
- POINT TO POINT AND MULTIPOINT AT SPEEDS UP TO 19.2 K BPS for increased flexibility in data transmissions
- RS-232 AND CURRENT LOOP for ease in communication interface.

Increased Data Integrity

Tandem's commitment to data integrity has now been extended to the 6520 terminal through the use of parity on all display memory. No other terminal configuration can as reliably transmit to and receive data from the host processor, monitoring it for accuracy. Nor can other terminals provide as much assurance as the 6520 that the data displayed on the screen is the data actually received by the terminal.

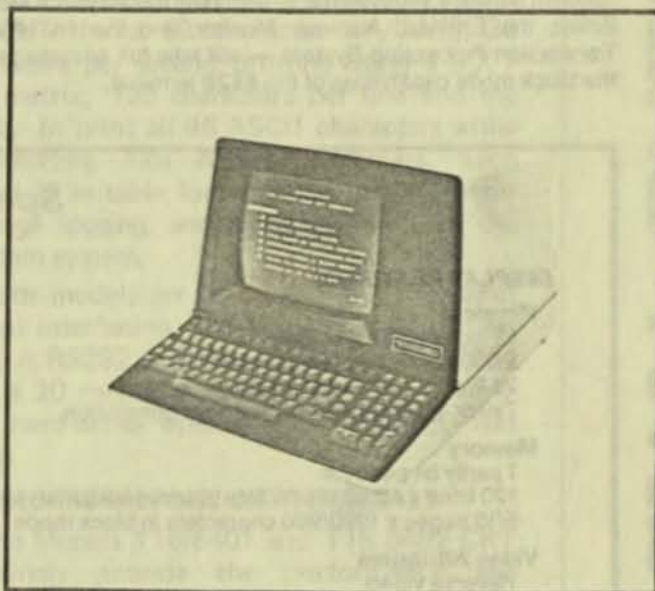
The Tandem 6520 provides immediate feedback to the operator if a memory parity error occurs, by displaying an error message on the 25th line of the screen. It also terminates any I/O transaction in process to prevent contamination of the data base.

High Data Throughput

The Tandem 6520 has an impressive multiple display paging facility which significantly enhances throughput — a critical factor in the on-line transaction environment.

While interacting with an operator on a screen display, the 6520 terminal can be transmitting from, or receiving into, any one of the five 1920 character pages at the same time.

The multiple display paging feature gives the host processor the option of writing to, or reading from, any page of the 6520 terminal at its discretion.



Ease of Operation

The Tandem 6520 provides a full complement of video and data attributes for ease of operation. Video attributes include: reverse video, blinking, non-display, underscore and two programmable brightness levels. A 25th display line is provided for status information.

Data attributes include: protected and unprotected fields, alpha/numeric or alphanumeric only fields, upshifted fields, modified field indication and auto-tab disable fields.

The 6520 provides an impressive array of editing and function controls for ease of use. Local and remote editing functions are an integral part of the terminal design, continuing Tandem's commitment to fully distributed processing.

Editing keys include: cursor up, down, left, right, home and return. Screen control keys include erase character/line/field/page, character/line insert and delete, set/clear tabs, forward/back tabulation, roll up and down, next/previous page and numeric key pad.

There are 16 function keys, providing 32 user-programmable functions, for greater flexibility in defining application dependent functions.

Increased Flexibility

The Tandem 6520 allows a high degree of flexibility in the ways it may be utilized by Tandem 16 systems, allowing increased ease of configuration and expansion, a very important factor in the on-line transaction oriented environment.

Conversational mode allows the terminal to interact with the host processor on a character-at-a-time basis. Display memory is organized as 120 lines by 40 or 80 characters wide, 24 lines of which are displayable at a time. The screen may be rolled up or down to view all 120 lines.

Block mode operation allows the terminal to transmit and received blocks of characters. Display memory is referenced as five pages of 1920 characters or 10 pages of 960 characters. Fields may be designated as PROTECTED, which will not allow entry into those fields from the keyboard. Through the use of the READ MODIFIED command, only fields which have been modified from the keyboard will be transmitted by the terminal.

Tandem software subsystems — including the VS Block Mode Editor, the EXPAND Network Monitor, and the PATHWAY Transaction Processing System — will take full advantage of the block mode capabilities of the 6520 terminal.

Simplified Communications Interface

Synchronous operation allows attachment of the 6520 to the Tandem T16/6202 Byte Synchronous Controller in block mode via a half or full duplex RS-232 communication link. The 6520 is connected multipoint with one to 63 Tandem 6520 terminals on the same communication link.

Asynchronous operation allows attachment of the 6520 to the Tandem T16/6303 or 6304 Asynchronous Communications Controller via a half or full duplex RS-232 communication link either point to point in conversational or block modes, or multipoint in block mode. The 6520 may also be connected to the 6303 or 6304 controller via a 20 ma current loop link in point to point conversational or block modes.

Both synchronous and asynchronous communications protocols allow the 6520 to operate at nine different speeds between 110 and 19.2 K BPS.

Tandem also offers the 6524 terminal which has the same impressive features as the 6520, plus the printer port option.

Specifications

DISPLAY FEATURES

Physical

- 12 inch diagonal screen
- 2,000 characters per screen
- 25 lines x 40/80 characters per screen
- 7x9 or 14x9 dot matrix character generation

Memory

- 1 parity bit per byte
- 120 lines x 40/80 characters in conversational mode
- 5/10 pages x 1920/960 characters in block mode

Video Attributes

- Reverse video
- 2 brightness levels
- Blinking
- Underscore
- Non-display
- Blinking underscore/reverse video cursor
- Addressable/readable cursor

Data Attributes

- Protected/unprotected fields
- Modified data tag/partial screen transmit
- Alpha/Numeric/Alphanumeric only fields
- Upshifted fields
- Auto-tab disabled fields

KEYBOARD FEATURES

Style

- Typewriter
- 32 program function keys
- 10 key pad
- Repeating keys
- Audible alarm

Editing Functions

- Roll up/down
- Set/clear tabs
- Forward/reverse tabulation
- Character/line insert and delete
- Line/page erase
- Up/down/left/right/home/return cursor controls
- Previous/next page

COMMUNICATION FEATURES

- Conversational Mode
- Block Mode
- ASCII code
- Synchronous/Asynchronous protocols
- Half/full duplex
- Point to point/multipoint
- RS-232/20ma current loop
- Speeds of 110, 150, 300, 1200, 1800, 2400, 4800, 9600, 19,200 bps

ENVIRONMENTAL FEATURES

Physical

- 13.3" x 17.0" x 22.0" (HxWxL)
- 40 lbs.

Electrical

- 100/120 VAC, 60 HZ, 2 Amps
- 220/240 VAC, 50 HZ, 1 Amp

Temperature/Humidity

- 40F to 95F (5C to 35C)
- 20% to 80% relative humidity (non-condensing)

Mounting

- Table top

TANDEM

TANDEM COMPUTERS INCORPORATED, 19333 Vallco Parkway, Cupertino, CA 95014. Toll Free (800) 538-9360 or (408) 996-6000 in California. Offices throughout the United States, Canada, Europe and the United Kingdom. Distributors in Australia, Finland, Mexico and Venezuela.



TANDEM

TERMINAL SUBSYSTEMS

- High Performance Terminals for NonStop^(tm) Transaction-Oriented Applications
- Select from Keyboard/Printer, CRTs or Receive-Only Printer
- Asynchronous RS232 or 20mA Current Loop Interfaces
- CRT Speeds to 19.2K bps
- Serial Printers Speeds to 200 cps
- Field Proven Reliability

SUBSYSTEM CONFIGURATIONS

The Tandem Terminal Subsystems offer the user a wide selection of operational benefits to maximize the Non-Stop features of the system and minimize program development overhead. All terminals meet or exceed ANSI, EIA and ECMA standards in addition to being UL certified. A wide range of options are also offered to meet the individual needs of a customized or diversified application.

A basic Terminal Subsystem consists of a Multiplexer, a System Console and up to 16 additional terminal lines. The Multiplexer is comprised of a Model T16/6301 Asynchronous Controller and one or two Model T16/6302 Asynchronous Extension Interfaces. The dual-port Controller handles up to 32 independent communication lines for modem or direct-wire connection compatible with single- or multi-drop terminals. One line is dedicated to the hard- or soft-copy system console. Each Asynchronous Extension provides an interface for up to 15 additional terminal lines.

MODELS T16/6603-6604 KEYBOARD/PRINTER TERMINALS

The Models T16/6603 and T16/6604 terminals provide hard-copy output combined

with keyboard input at a low cost. Both models offer a 59 character keyboard, 30 characters per second printing using a 5 X 7 dot matrix, 132 characters per line and the ability to print all 96 ASCII characters while transmitting 128 ASCII characters. Each model is suitable for operator logging, error message logging, and operator input to the Tandem system.

Both models are identical in every respect except interfacing. The Model T16/6603 uses an EIA RS232 interface while the T16/6604 uses a 20 mA current loop interface permitting hard-wired operation up to 1500 feet away.

MODELS T16/6401-6402 CRT TERMINALS

The Models T16/6401 and T16/6402 CRT Terminals provide the performance and reliability of cathode ray tubes at less cost than keyboard/printer terminals. Both models incorporate a compact 12 inch CRT which displays 24 lines by 80 characters, or 1920 characters per screen. The standard keyboard provides 59 keys for ease of data entry. Upper Case/Lower Case may be chosen as an option. Transmission rates are switch selectable from 75 to 19,200 bps. Thus display rates up to 1920 characters per second are attainable.

Both models are identical in every respect except interfacing. The Model T16/6401 uses an EIA RS232 interface while the T16/6402 uses a 20mA current loop interface permitting hard-wired operation up to 1500 feet away.

MODELS T16/6511-6512 PAGE-MODE CRT TERMINALS

The Models T16/6511 and T16/6512 CRT Terminals provide increased capability over the T16/6401-6402 CRT Terminals by the use of page-mode display screens and added keyboard functions. Both models incorporate a compact 12 inch CRT which displays 24

lines by 80 characters, or 1920 characters per screen. The standard screen provides the ability to define protected and data entry fields for page-mode operation. The standard keyboard provides 68 keys for ease of data entry. In addition to the standard keys a ten-key pad, 16 function keys, edit function keys, and cursor control keys are provided. Upper Case/Lower Case is switch selectable. Transmission rate for both models is 9600 bps. Thus display rates up to 960 characters per second are attainable.

Both models are identical in every respect except interfacing. The Model T16/6511 uses an EIA RS232 interface while the T16/6512 uses a 20mA current loop interface permitting hard-wired operation up to 1500 feet away.

MODEL 5508 SERIAL PRINTER

The Model T16/5508 Serial Printer provides high performance serial printing at a modest cost. Using a 9-wire ballistic head, the T16/5508 generates 7 X 9 dot matrix characters at speeds of 200 characters per second. Printing is bi-directional to provide maximum throughput for NonStop applications. Both upper and lower case characters can be printed. A versatile electronic VFU provides flexible forms control.

The T16/5508 uses a 20mA current loop interface. No special line printer interface is required since a current loop port on the Asynchronous Multiplexer can be used. The T16/5508 can be hard-wired up to 1500 feet from the Tandem system.

SPECIFICATIONS

	T16/6603-6604	T16/6401-6402	T16/6511-6512	T16/5508
Device Type	KYBD/PTR	CRT	CRT	R/O PTR
PRINT/DISPLAY/KYBD				
Type	Impact	CRT	CRT	Impact
Character Gen (Dot Matrix)	5 X 7	5 X 7	5 X 9	7 X 9
Columns	132	80	80	132
Lines	n/a	24	24	n/a
Character Display	n/a	1920	1920	n/a
Horizontal Spacing	10	n/a	n/a	10
Diagonal Display	n/a	12"	12"	n/a
VFU	no	n/a	n/a	yes
Keyboard	59 keys	59 keys	116 keys	n/a
Printing	Unidirectional	n/a	n/a	Bidirectional
Print Speed	30 cps	1920 cps	960 cps	200 cps
Transmission				
Code	ASCII	ASCII	ASCII	ASCII
Connection				
RS232	6603	6401	6511	n/a
20mA CL	6604	6402	6512	yes
Line Speed (bps)	300	75 to 19.2K	9600	9600
Mounting	Pedestal	Table-top	Table-top	Table-Top
Electrical				
Voltage	110V, 60Hz 220V, 50Hz	110V, 60Hz 230V, 50Hz	110V, 60Hz 230V, 50Hz	110V, 60Hz 220V, 50Hz
Operating Amps	3.1	0.8	2.0	4.0
Physical				
Dimensions (LxWxH)	24x28x34	20x16x13	24x21x13	16x23x13
Weight (lbs)	120	32	50	45
Environmental				
Temperature	50-104F	41-122F	41-122F	39-95F
Humidity	10-90%	5-95%	5-95%	20-90%

Software

Operating
Sys. Network

Processing

Labware

GUARDIAN OPERATING SYSTEM SOFTWARE

Overseeing Tandem 16 System operation is the Tandem 16 Guardian Operating System. Guardian provides the multi-processing (parallel processing in separate processor modules), multiprogramming (interleaved processing in one processor module), and NonStop capabilities of the Tandem 16 Computer System.

In a typical system, master copies of the Guardian Operating System, configured for the specific application, are kept in a "system" area on a disc volume. Critical and frequently used parts of Guardian are resident in each processor module's memory. As such, the system's capabilities are maintained even if a module fails. Non-critical or less frequently used parts of Guardian are virtual and are brought into a processor module's memory from disc only when needed.

Several functions of Guardian are transparent to application programs. These include:

- . The preparation of a program for execution in virtual memory when a request is made to run a program.
- . The capability for processes to communicate with each other regardless of the processor module where they are executing.
- . Providing the virtual memory function by automatically bringing absent memory pages in from disc when needed.
- . Scheduling processor module time among multiple processes according to their application-assigned priorities (a "process" is an executing program).

Guardian provides an additional and extremely important function. Concurrent with application program execution, Guardian continually checks the integrity of the system. This is accomplished as follows: Guardian in each processor module, at a predefined interval, transmits "I'm alive" messages to Guardian in every processor module (this interval is typically one second). Following this transmission, Guardian in each processor module checks for receipt of an "I'm alive" message from every other processor module. If Guardian in one processor module finds that a message has not been received from another processor module, it first verifies that it can transmit a message to its own processor module; if it can, it assumes that the non-transmitting processor module is inoperative; if it can't, it takes action to insure that its own module does not impair the operation of other processor modules. In either case, Guardian then informs system processes and interested application processes of the failure.

An important service provided by Guardian is file management. File management is the means by which application programs perform input/output operations in the Tandem 16 computer system. A "file" can be all or a portion of a disc pack, a non-disc device such as a terminal or line printer, a process (i.e., running program), or the operator console. Files are identified by symbolic "file names." This frees the programmer from needing to know the physical addresses of I/O devices and permits addition and reconfiguration of input/output devices without the need to rewrite or recompile programs.

File management operations are performed by calling procedures that are part of the operating system. All files are accessed through this same set of procedures, thereby providing a single, uniform access method. Additionally, the file management procedures are designed so as to eliminate the peculiarities of various devices.

An application program "sees" operating system services as a set of "library" procedures. The library procedures have names such as "READ," "WRITE," "OPEN," etc. To request an operating system service (e.g., input), a call to the appropriate operating system procedure is written in the application program (e.g., "READ"). (The operating system library procedures exist in the System Code area and therefore are shared by all processes.)

Another Guardian feature is "mirrored" disc volumes. A "mirrored" disc volume consists of a pair of physically independent disc devices. The purpose of a mirror volume is to insure that the information on a disc volume (i.e., operating system, programs, virtual-swapping-area, and application data) will be available even if one of the disc drives fail. When a write is made to a mirror volume, Guardian records the data to be written on the packs of both disc devices. As a result, both disc packs contain identical information. If one of the disc devices becomes inoperable, Guardian performs all reads and writes to the other operable disc volume. The operation of a mirror volume is entirely transparent to both application programs and system users.

Process control is another important service provided by Guardian. Through the process control functions, an application process can run and stop processes in any processor module in the system and can monitor the operation of any processor module or any process running in the system. If a module fails or a process stops, or if a failed module becomes operable, Guardian will notify the application process. Process control functions are invoked by making call to operating system procedures.

All Tandem System programs and operating system procedures are designed to operate with standard character-mode terminals, with or without communications modems. No special programming is required. Tandem also provides the ENVOY Data Communications Manager as part of its standard operating system. ENVOY supplies data communications services for asynchronous and binary synchronous communications networks in both point-to-point and multipoint networks. For remote job entry with batch data transmission, the Tandem EXCHANGE system can be used by an operator or programmatically. The EXCHANGE system is an extremely flexible 2780/3780 emulator software package that has capability beyond the classical IBM product.

2. Prevent unauthorized access to sensitive data files by programmers or operations personnel.
3. Prevent unauthorized interference with executing programs (processes).

Additionally, the Guardian Security System is designed so as not to interfere with application design in systems where security is not desired.

Additional security may be provided by the application program. Some examples of application program security checks are:

Limitation of Capability at a Terminal

It is not necessary to have a Guardian Command Interpreter executing at an application terminal. Therefore, the application program has control over what the terminal operator sees. The application program can limit the functions that the terminal operator can perform.

Physical Security

Programs which alter or produce reports of sensitive data may include routines which check the terminal from which they are run. This allows the application to restrict the running of the program to a specific terminal which is physically secure (e.g., in a locked office for which there is only one key).

Special Devices

These include authorization terminals such as badge readers, fingerprint readers, etc.

Individuals that have access to the system are called "users." In general, there are four classes of system users:

SECURITY

The Guardian Security System is designed to fulfill three objectives:

1. Prevent inadvertent destruction of files through purging or overwriting.
2. Prevent unauthorized access to sensitive data files by programmers or operations personnel.
3. Prevent unauthorized interference with executing programs (processes).

Additionally, the Guardian Security System is designed so as not to interfere with application design in systems where security is not desired.

Additional security may be provided by the application program. Some examples of application program security checks are:

- Limitation of Capability at a Terminal

It is not necessary to have a Guardian Command Interpreter executing at an application terminal. Therefore, the application program has control over what the terminal operator sees. The application program can limit the functions that the terminal operator can perform.

- Physical Security

Programs which alter or produce reports of sensitive data may include routines which check the terminal from which they are run. This allows the application to restrict the running of the program to a specific terminal which is physically secure (e.g., in a locked office for which there is only one key).

- Special Devices

These include authorization terminals such as badge readers, fingerprint readers, etc.

Individuals that have access to the system are called "users." In general, there are four classes of system users:

. Standard User

A "standard" user is allowed to perform standard operations such as creating and purging disc files, running programs, displaying system status, etc. Additionally, a standard user is limited as to the processes it can stop or debug.

. Group Manager

A "group manager" user is permitted to perform the standard operations as well as designate new system users.

. System Operator

A "system operator" user is permitted to perform the standard operations as well as reload processor modules, set the system time-of-day clock, and alter the operating state of the interprocessor buses.

. Superid

The "superid" user has total freedom to perform any operation in the system. This includes debugging privileged programs, accessing any file, logging on as any user without knowing the user's password, adding new groups to the security system, running privileged programs which have not been "licensed."

Additionally, for systems where security is not desired, all standard users can be defined as "null" users. In a system such as this, all users have equal access to all files in the system. However, such a system must still have a "superid" user and perhaps a "system operator" user so that their related functions can be performed. Another alternative is to have all users access the system as the "superid."

Before a user can access the system, the user must "log on." Log on is accomplished by supplying an application-predefined user name to the system by means of the Command Interpreter LOGON Command. The user name supplied to LOGON is of the form

<group name> . <user name>

<group name> identifies an individual as a member of a group (e.g., a department).

<user name> identifies the individual within the group.

GUARDIAN/EXPAND NETWORK SUBSYSTEM

The EXPAND NonStop Network is unique because it is an extension of an existing network operating system. Every Tandem/16 Computer System comprises from 2 to 16 separate central processors. Running in the NonStop mode requires constant communication among the processors. Consequently, the Guardian Operating System includes a sophisticated message handling system to control communications between processors and between processes (programs) running in one processor or running in separate processors. In effect, every Tandem/16 System is a local network controlled by the Guardian Operating System. The EXPAND NonStop Network expands the scope of the Guardian Operating System to allow communications among as many as 255 Tandem/16 systems.

The Guardian/EXPAND Network system, combined with the unique architecture of the Tandem/16 Computer System, provides network users with a number of features unequalled by other computer vendors:

- NonStop Nodes

The fault-tolerant hardware and software of the Tandem/16 System eliminates the computer as a source of network failures.

Note: Individual Tandem/16 Systems within the network are called "nodes" to distinguish the computer system from the network system.

- A Distributed System

The Guardian/EXPAND Network makes it possible to configure a network of Tandem/16 fault-tolerant computer systems so that a user of any node in the network can access the resources of any other node (processors, files, or physical devices) without regard for the physical location of the resource. To the user, the Guardian/EXPAND Network appears to be one large set of computer resources rather than a collection of separate systems.

- Dynamic Message Routing

The Guardian/EXPAND Network constantly monitors the communications paths. When a transmission fails, the system retries until the transmission succeeds or until the system determines that the communication path has been broken. When the communication path has been broken, the Guardian/EXPAND system automatically reroutes the message via a different communications path.

. Best Path Message Routing

The Guardian/EXPAND system monitors the communication lines and automatically selects the best path. The best path is the one that takes the least time. The system selects the fastest rather than the shortest path because this optimum end-to-end protocol can reduce communications costs. When a communications line fails, Guardian/EXPAND reroutes messages using the next fastest available path. When a new line is added, the system takes advantage of any new best paths created by the addition.

. Precisely Tailored Hardware

Different nodes within a network typically have different computing requirements. The Tandem/16 System allows users to place exactly the right amount of computing power at each site. Even though the nodes within the network may range from a basic two processor system to a sixteen processor system with billions of bytes of online disc storage, the systems retain total compatibility of data, software, and application programs.

. Logical Growth

The Tandem/16 architecture allows for incremental hardware expansion. A user can add memory, central processors, or peripheral devices as computing requirements grow. Similarly, the Guardian/EXPAND Network allows incremental expansion. Nodes can be added or removed and communication paths can be changed, all without reconfiguring existing systems.

Notice that the Guardian/EXPAND Network can forestall the need for hardware expansion since the resources of every system in the network are accessible.

. Data Integrity

The Guardian/EXPAND Network incorporates multiple safeguards to ensure that message packets are received correctly and that data cannot be lost in transmission.

. Human Engineering

Because the EXPAND Network is an extension of the Guardian Operating System, the user interface to the network is through the Guardian Command Interpreter. For example, to run the text editor program on the local system, the user enters the command EDIT at his terminal. To run the program on a remote system,

the user simply enters the symbolic name of the remote system before the command: \OHIO EDIT. In effect, this command connects the user terminal with the remote OHIO system and starts the next editor program on that system. The only difference the user may notice in running on the remote system rather than the local system is that response time may be slightly longer since the communication lines cannot match the performance of the local processor.

Simple Programming Interface

The EXPAND Network relieves programmers of the need to deal with a cumbersome telecommunication access method. Since programs communicate with each other via Guardian's message system, the programmer uses the same commands to communicate with a program in the same processor, another processor, or another system.

Full appreciation of the EXPAND Network System requires an understanding of how tightly the EXPAND Network System is integrated with the Guardian Operating System and the Tandem/16's architecture.

FEATURES

- Full-Service Message System
- Self-Programming Message Management
- Remote File Access
- Full-Service Editor and Translator
- System-Independent Translator Application
- Command Language
- System-Independent Translator Application

- Programmers can write programs that communicate with distributed systems and other programs without actually knowing in which processor their program is executed or in which processor other programs are located.
- The hardware choice of multi-processor systems is transparent to application programs.
- The system can be reconfigured automatically.
- For application-dependent error recovery, the system provides an error report not specifically designed for error recovery, using an in-built-out approach.

INTRODUCTION

The GUARDIAN Operating System is a multi-processor system designed for reliability, performance, and security. It has been specifically architected to support a wide range of applications and to provide a high level of security. Many of the responsibilities normally handled

The full-service features of GUARDIAN make it a complete solution for the user, providing security, reliability, and performance. It is designed to be integrated with other systems, such as databases, and to share the responsibility of the application program.

TANDEM

GUARDIAN OPERATING SYSTEM

FEATURES

- Fail-Safe GUARDIAN Operating System for Continuous Non-Stop™ Running of Transaction-Oriented Applications
- Multiprogramming/Multiprocessing Virtual Memory Management System to Support High Transaction Rates from Numerous High-Speed Terminals
- Redundant File Management System with Symbolic File Access, File Security, File and Record Locking, Concurrent Input/Output and Disc Volume Interchangeability without Reprogramming
- Fail-Safe Message System with Checkpointing for Fault-Tolerant Programs, Process Control to Establish, Change, Suspend or Delete Processes, and Automatic Resource Allocation and Memory Mapping
- Fail-Safe Utility Procedures for Automatic Data Conversion, Time and Date Logging, and Calling the Debug Facility
- Efficient High-Level Transaction Application Language (TAL) Compiler for Easy Implementation of Application Programs
- Interactive Command Interpreter (COMINT), Text File Editor (EDIT), Object File Editor (UPDATE), and Debugger (DEBUG) for Fast and Economical Program Development
- On-Line Program-Concurrent Diagnostics for System Security and Program Integrity.

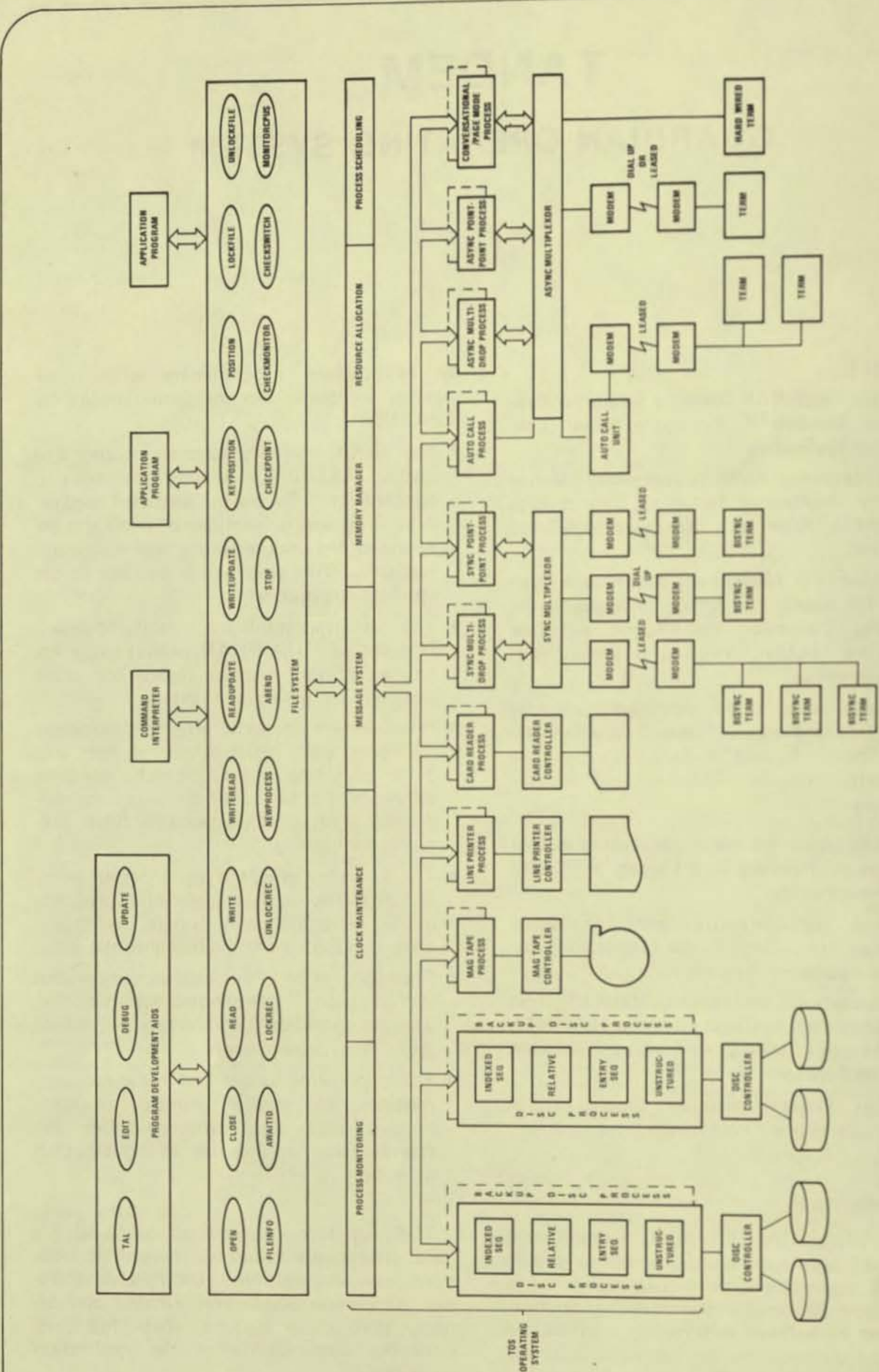
INTRODUCTION

The GUARDIAN Operating System is a true multi-processor, multi-programming *fault tolerant* operating system. GUARDIAN has been specifically architected so that applications can be defined and implemented to run continuously (even during hardware failures). Many of the responsibilities normally handled

by applications programmers with other systems are taken care of automatically by GUARDIAN:

- The virtual memory scheme incorporated into GUARDIAN enables programmers to concentrate fully on the intended application. Programs or portions of programs are swapped between memory and disc automatically. This swapping is invisible to the applications program.
- The multiprogramming, multiprocessing features of GUARDIAN permit programs to be written without regard for other programs running in the system.
- Programmers can write programs that communicate with input/output devices and other programs without actually knowing to which processors these devices are connected or in which processors other programs are located.
- The hardware aspect of input/output transfers is transparent to application programmers. Interrupts and system dependent error conditions are handled automatically.
- Transfers over the interprocessor bus (DYNABUS)™ are handled automatically. The bus operation is completely invisible to application programmers.
- For application dependent error recovery routines, the system provides an error number that specifically describes any errors encountered during an input/output operation.

The *Fail-Safe* features of GUARDIAN make it uniquely suited for the fast development, easy expandability, and reliable operation of on-line transaction-oriented applications. With other systems, these functions were the responsibility of the application programmer.



Guardian Operating System

VIRTUAL MEMORY MANAGEMENT

Within each processor the operating system is responsible for allocation of execution time to multiple programs on a priority basis; allocation of buffer space and control blocks; process synchronization; fault and trap handling; and interval clock maintenance. In addition, GUARDIAN isolates the application from physical memory constraints by providing an efficient virtual memory management system. Paging hardware is provided in the form of four 128K byte memory maps (user code user data, system code and system data). All code is both sharable by multiple programs and non-modifiable, two features which reduce overlay and swapping overhead. In addition, hardware is used to record frequency of access to all memory pages and modification of data pages, thus providing a low overhead method of determining the correct page to be replaced.

FILE SYSTEM

Provided as part of GUARDIAN, the *File Manager* is a flexible, easy-to-use device-independent interface. It allows an application program to communicate with disc files, serial I/O devices, conversational page mode and multidrop terminals, and other application programs all through one standard set of routines. The File Manager allows program execution to continue concurrent with execution of the file request. In fact, the input/output may be taking place in a different processor module.

The File Manager also includes a number of features unique to the *NonStop* environment. When a processor or I/O port fails, the File Manager automatically reroutes subsequent requests to a back-up processor module. In addition, through a very efficient program-to-program communication mechanism, the File Manager provides the application with a simple method of keeping back-up programs informed of current operations so that a smooth transition may be made.

File Access — For I/O devices normally dedicated to a single process, such as Terminals or Line Printers, the device is assigned a *symbolic* name. The programmer need not know the physical address of the device. This provides a simple means to add and/or reconfigure I/O devices without reprogramming the application. In the case of disc devices, a file name represents a user-specified portion of the disc storage space. The File Manager makes no distinction between sequential and random access to a disc file. A file pointer (relative byte address) determines where a data transfer is to begin. The file pointer is normally automatically incremented for sequential access but may also be set explicitly for random access. An access mode specifies the operations to be performed on a file:

- Read/Write (Default Mode)

- Read Only
- Write Only

File Procedures — To implement file operations, calls are made to file management procedures. All files are accessed through the same set of procedures which provides a single uniform access method. These procedures include:

- CREATE a new file
- OPEN a file for access
- READ data from a file
- WRITE data to a file
- WRITEREAD write/read data to/from a file
- READUPDATE read data from a file for subsequent update
- WRITEUPDATE write updated data to a file
- CLOSE a file to access
- PURGE a file from a disc

Numerous other procedures are provided for device-dependent operations.

File Security — The versatile File Manager provides the ability to limit file access to an individual, a group of individuals, or an individual within a group. This limitation is *password* protected. Four types of file operations (read, write, execute and purge) may be separately limited to either the individual (owner) who created the file, the owner's group, or any individual within the group. A program uses an *exclusion mode* to limit file access. The exclusion modes are:

- Shared Access (Default Mode)
- Exclusive Access
- Protected Access

File locking is provided so that cooperative application programs may share file operations as a disc file. Also, disc volumes may be removed and replaced without reprogramming the application and without loss of file security.

MESSAGE MANAGEMENT SYSTEM

The GUARDIAN message system handles all communications between Tandem 16 processor modules, system processes and application programs. It frees the user of the responsibility of routing messages to the correct processor, verifying that it got there correctly, and deciding which program is to receive it in the destination processor. A program need not be aware of which of the 16 Tandem processors will ultimately run it. In fact, the same program may be executing simultaneously on all processors. In addition, a program may access any device on the system, even if the device is not physically connected to the processor in which the program is running. This allows the system to be expanded without reprogramming the application.

Process Control — A *process* is the execution of a program under control of the GUARDIAN Operating System. Process control procedures are used to interactively call processes. These processes are:

- **NEWPROCESS** — create a new process (run a program)
- **DELAY** — suspend the calling process for a specified interval of time
- **PRIORITY** — change the process execution priority
- **STOP** — delete a process with a normal indication
- **ABEND** — delete a process with an abnormal indication

For example, to have a process delete itself and close its files, the procedure CALL STOP is used.

System Messages — GUARDIAN sends messages directly to application processes to inform the application of certain system conditions. Some of these messages are:

- **Processor Module Failed**
- **Process Stopped Execution**
- **Processor Module Reloaded**

Certain critical error conditions prevent normal execution of a process. These errors cause traps to GUARDIAN Trap Handlers.

Checkpointing — Fail-safe, non-stop operating environments require one or more primary/backup process pairs. The primary process executes in one processor while the backup process monitors in another processor. With this type structure, the backup process is kept informed of the primary's execution state of the primary process via periodic *checkpoint* messages sent to the backup process. Each process in a process pair has the same set of files open. This ensures that the backup process has immediate access to the files in the event of a primary processor's failure. GUARDIAN provides several procedures to aid in the development of fault-tolerant programs:

- **CHECKOPEN** — is called by a primary process to open a file in its backup process
- **CHECKPOINT** — is called by a primary process to checkpoint its current state to its backup process
- **CHECKMONITOR** — is called by a backup process to monitor its primary and take appropriate action in the event of the primary's failure
- **CHECKSWITCH** — is called by a primary process to switch control to its backup process

- **CHECKCLOSE** —

is called by a primary process to close a file in its backup process

COMMAND INTERPRETER (COMINT)

Tandem's COMINT is a high-level man-machine interface which allows the user to converse with the system. The user may obtain or alter the current operational status of the system; create, verify and purge disc files; and run programs (both Tandem-supplied and application programs). Normally, the user initially executes COMINT on a system console. From this point on, COMINT may be specified to be run on any other terminals connected to the system.

PROGRAM DEVELOPMENT AIDS

Included as part of the standard software provided with every Tandem 16 system is a comprehensive set of program development tools. These programs, which run under control of the Command Interpreter, allow the user to develop application programs with a minimum of effort. Included are a high-level compiler, a source file editor, object file editor and interactive debugging facility.

Transaction Application Language (T/TAL)

Tandem's *Transaction Application Language* is a high-level, block structured language designed for the easy implementation of transaction-oriented applications. T/TAL provides many high level constructs, including IF THEN, FOR, DO UNTIL, WHILE and CASE. It allows the programmer to write self-documenting programs and eliminates the time consuming errors inherent in assembly language programming. Special string manipulation operations are included to facilitate fast processing of transaction data; among them are MOVE, COMPARE and SCAN strings. While providing all these flexible features, T/TAL does not sacrifice execution efficiency. A highly optimized compiler, it produces object programs as efficient as those written in an assembly language.

Edit — The *Text Editor* is a very flexible, interactive text file editor that can be used to prepare both program source files and documentation. EDIT can be run from either conversational or page-mode terminals, and provides an additional interface to allow it to be driven by other programs.

Update — The *Object File Editor* allows the user to make changes to previously compiled programs. In addition, the output of multiple compilations can be combined through the facilities of UPDATE.

Debug — An interactive program debugging facility supported by GUARDIAN enables the user to test application programs. It provides program breakpoints, tracing of variables, and access to all of the code and data of the program, all from an interactive terminal.

TANDEM

Guardian/Expand

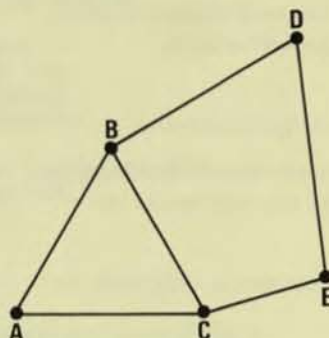
The Expand extension to the Guardian operating system extends the concept of fault-tolerant computing to networks of geographically distributed computer systems. A network of Tandem 16's, running under the Guardian/Expand operating system, features

- exceptional reliability
- ease of programming
- X.25 compatibility

Reliability

The reliability of a network of Tandem 16's derives from the basic architecture of the Tandem 16 itself.

- The hardware of the Tandem 16 is immune to the failure of any single component. Therefore a network of Tandem 16's is built with nodes that are designed to stay up
- The Expand software, as well as application programs, runs NonStop (TM), providing immunity to a processor failure
- Expand *best path routing*, which always chooses the fastest path between nodes, allows the configuration of fault tolerant networks
- The Guardian/Enscribe fail-safe database record manager protects your distributed database



A Fault Tolerant Computer Network

In the event of a failure of path AC, the Network Control Process automatically reroutes messages from A to C via ABC.

Ease of Use

A network of Tandem 16's is not much more complicated than a single Tandem 16, because the Tandem 16 is already a "network" of processors.

The Guardian operating system allows the Tandem 16 to be viewed as a set of inter-communicating processes distributed over several processors. Each process communicates with others without regard for where the others are running.

The expand extension generalizes this point of view to a set of processes distributed over a network. Thus, no special programming is necessary; most programs that run on a Tandem 16 will run on a network of Tandem 16's, because an application's environment looks the same whether it's running in a network or in a single Tandem 16 system.

- Networks of Tandem 16's are simple to configure; moreover, once a network is configured, nodes can be added or removed without the need for re-configuration.

55
77888899

X.25 Compatibility

A network of Tandem 16's can be interfaced to other hardware via a public X.25 packet network.

Components of Expand

Expand consists of the following components:

- Tandem's *End to End protocol* ensures data integrity not just from node to node, but from sender to receiver.
- The *Network Control Process* runs at each node. Its responsibilities include:
 - establishment of communication paths to other nodes
 - maintenance of routing information and determination of the best path to other nodes
 - logging of network status
- A *Network Line Handler* manages one full-duplex communication line of one X.25 virtual circuit, and implements the End to End protocol.

The Network Control Process and Line Handlers run NonStop: i.e., they are unaffected by a processor failure.

- The *Network Utilities* include programs used to monitor the network. The Utilities allow you to trace the path of data through the network, monitor network events (such as line connections), display network-related statistics, and determine the status of individual processors throughout the network.

The reliability and simplicity of a network of Tandem 16's derive from basic Tandem 16 architecture

	<u>Tandem 16</u>	<u>Network of Tandem 16's</u>
reliability:	fault tolerant hardware and software	each node of the network is designed to stay up
simplicity:	The Guardian operating system lets programs communicate without regard for their location in the system	The Guardian/Expand operating system lets programs communicate without regard for their location in the network

TANDEM

TANDEM COMPUTERS INC., 19333 Vallco Parkway, Cupertino, California 95014 • Toll Free 800-538-9360 or (408)996-6000 in California. Branch offices throughout the United States. Foreign Offices in Frankfurt, Dusseldorf and Munich; West Germany • Uxbridge, Middlesex; England • Zurich; Switzerland • Toronto; Canada.

A "NonStop"* Operating System

Joel P. Bartlett
Tandem Computers Inc.
19333 Vallco Parkway
Cupertino, California

Copyright (C) 1977, Tandem Computers Inc.
All Rights Reserved

ABSTRACT

The Tandem/16 computer system is an attempt at providing a general-purpose, multiple-computer system which is at least one order of magnitude more reliable than conventional commercial offerings. Through software abstractions a multiple-computer structure, desirable for failure tolerance, is transformed into something approaching a symmetric multiprocessor, desirable for programming ease. Section 1 of this paper provides an overview of the hardware structure. In section 2 are found the design goals for the operating system, "Guardian". Section 3 provides a bottom-up view of Guardian. The user-level interface is then discussed in section 4. Section 5 provides an introduction to the mechanism used to provide failure tolerance at the application level and to application structuring. Finally, section 6 contains a few comments on system reliability and implementation.

1. INTRODUCTION

1.1 Background

On-line computer processing has become a way of life for many businesses. As they make the transition from manual or batch methods to on-line systems, they become increasingly vulnerable to computer failures. Whereas in a batch system the direct costs of a failure might simply be increased overtime for the operations staff, a failure of an on-line system results in immediate business losses.

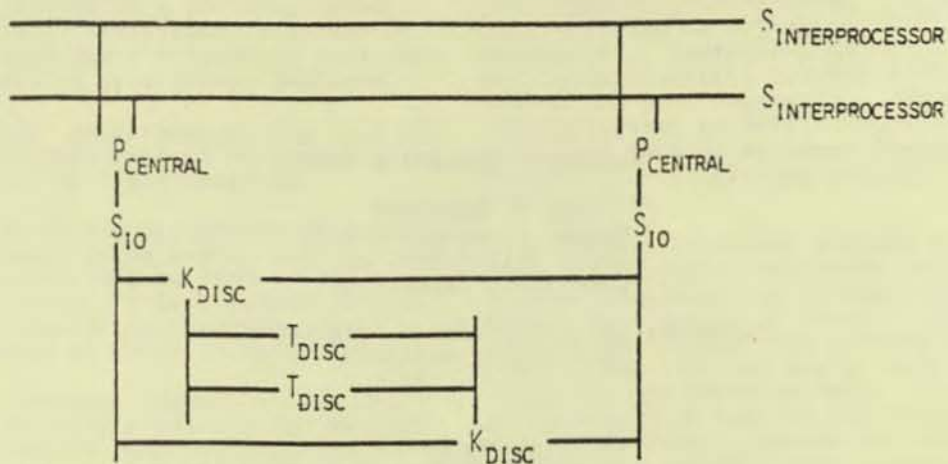
1.2 System Overview

The Tandem/16 (1,2) was designed to provide a system for on-line applications that would be significantly more reliable than currently available commercial computer systems. The hardware structure consists of multiple processor modules interconnected by redundant interprocessor

buses. A PMS (3) definition of the hardware is found in Figure 1.

Each processor has its own power supply, memory, and I/O channel and is connected to all other processors by redundant interprocessor buses. Each I/O controller is redundantly powered and connected to two different I/O channels. As a result, any interprocessor bus failure does not affect the ability of a processor to communicate with any other processor. The failure of an I/O channel or of a processor does not cause the loss of an I/O device. Likewise, the failure of a module (processor or I/O controller) does not disable any other module or disable any inter-module communication. Finally, certain I/O devices such as disc drives may be connected to two different I/O controllers, and disc drives may in turn be duplicated such that the failure of an I/O controller or disc drive will not result in loss of data.

* "NonStop" is a trademark of Tandem Computers Inc.



Hardware Structure
Figure 1

The system is not a true multiprocessor (4), but rather a "multiple computer" system. The multiple computer approach is preferable for several reasons. First, since no module is shared by the entire system, it increases the system's reliability. Second, a multiple computer system does not require the complex hardware needed to handle multiple access paths to a common memory. In smaller systems, the cost of such a multiported memory is undesirable; and in larger systems, performance suffers because of memory access interference.

On-line repair is as necessary as reliability in assuring system availability. The modular structure of the Tandem/16 system allows processors, I/O controllers, or buses to be repaired or replaced while the rest of the system continues to operate. Once repaired, they may then be reintegrated into the system.

The system structure allows a wide range of system sizes to be supported. As many as sixteen processors, each with up to 512k bytes of memory, may be connected into one system. Each processor may also have up to 256 I/O devices connected to it. This provides for tremendous growth of application programs and processing loads without the requirement that the application be reimplemented on a larger system with a different architecture.

Finally, the system is meant to provide a general solution to the problem of providing a failure-tolerant, on-line environment suitable for commercial use. As such, the system supports conventional programming languages and peripherals and is oriented toward providing large numbers of terminals with access to large data bases.

2. SYSTEM DESIGN GOALS

2.1 Integrated Hardware/Software Design

The Tandem/16 system was designed to solve a specific problem. This problem was not stated in terms of hardware and software requirements, but rather in terms of system requirements. The hardware and software designs then proceeded in tandem to provide a unified solution. The hardware design concerned itself with the contents of each module, their interconnections to the common buses, and error detection and correction within modules and on the communication paths. The software design was given the problem of control; that is, selection of which modules to use and which buses to use to communicate with them. Furthermore, as errors are detected, it was the responsibility of the software to control recovery actions.

2.2 Operating System Design Goals

The first and foremost goal of the operating system, Guardian, was to provide a failure-tolerant system. This translated into the following design "axioms":

- the operating system should be able to remain operational after any single detected module or bus failure
- the operating system should allow any module or bus to be repaired on-line and then reintegrated into the system.

- the operating system should be implemented in a reliable manner. Increased reliability provided by the hardware architecture must not be negated by software problems.

A second set of requirements came from the great numbers and sizes of hardware configurations that are possible:

- the operating system should support all possible hardware configurations, ranging from a two-processor, discless system through a sixteen-processor system with billions of bytes of disc storage.
- the operating system should hide the physical configuration as much as possible such that applications could be written to run on a great variety of system configurations.

3. OPERATING SYSTEM STRUCTURE

To satisfy these requirements, the operating system was designed to have the appearance of a true multiprocessor at the user level. The design of the system was strongly influenced by Dijkstra's work on the "THE" system (5), and Brinch Hansen's implementation of an operating system nucleus for a single-processor system (6). The primary abstractions are processes, which do work, and messages, which allow interprocess communication.

3.1 Processes

At the lowest level of the system is the basic hardware as earlier described. It provides the capability for redundant modules, i.e. I/O controllers, I/O devices, and processor modules consisting of a processor, memory, and a power supply. These redundant modules are in turn interconnected by redundant buses. Error detection is provided on all communication paths and error correction is provided within each processor's memory. The hardware does not concern itself with the selection of communication paths or the assignment of tasks to specific modules.

The first abstraction provided is that of the process. Each processor module may have one or more processes residing in it. A process is initially created in a specific processor and may not execute in another processor. Each process has an execution priority assigned to it. Processor time is allocated on a strict priority basis to the highest priority ready process.

Process synchronization primitives include "counting semaphores" and process local

"event" flags. Semaphore operations are performed via the functions PSEM and VSEM, corresponding to Dijkstra's P and V operations. Semaphores may only be used for synchronization between processes within the same processor. They are typically used to control access to resources such as resident memory buffers, message control blocks, and I/O controllers.

When certain low-level actions such as device interrupts, processor power-on, message completion or message arrival occur, they result in "event" flags being set for the appropriate process. A process may wait for one or more events to occur via the function WAIT. The process is activated as soon as the first WAITed for event occurs. Events are signaled via the function AWAKE. Event signals are queued using a "wake up waiting" mechanism so that they are not lost if the event is signaled when the process is not waiting on it. Like semaphores, event signals may not be passed between processors. Event flags are predefined for eight different events and may not be redefined.

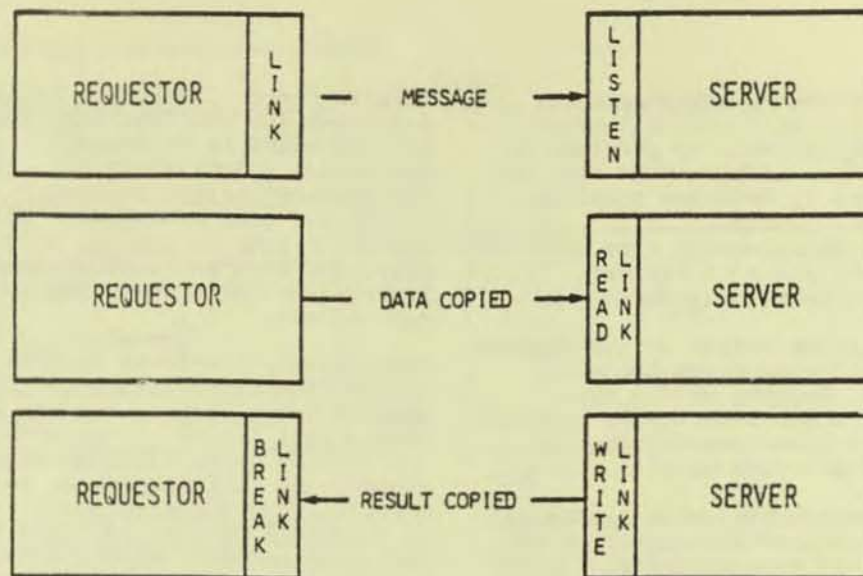
When a process blocks itself to wait for some event to occur or for a semaphore to be allocated to it, it may specify a maximum time to block. If the time limit expires and the event has not occurred or the resource has not been obtained, then the process will continue execution but an error condition will be returned to it. This timeout allows "watch dog" timers to be easily placed on device interrupts or on resource allocations where a failure may occur.

Each process in the system has a unique identifier or "processid" in the form: <cpu #, process #>, which allows it to be referenced on a system-wide basis. This leads to the next abstraction, the message system, which provides a processor-independent, failure-tolerant method for interprocess communication.

3.2 Messages

The message system provides five primitive operations which can be illustrated in the context of a process making a request to some server process, Figure 2. The process' request for service will send a message to the appropriate server process via the procedure LINK. The message will consist of parameters denoting the type of request and any needed data. The message will be queued for the server process, setting an event flag, and then the requestor process may continue executing.

When the server process wishes to check for any messages, it calls LISTEN. LISTEN



Message System Primitive Operations
Figure 2

returns the first message queued or an indication that no messages are queued. The server process will then obtain a copy of the requestor's data by calling the procedure READLINK.

Next, the server process will process the request. The status of the operation and any result will then be returned by the WRITELINK procedure, which will signal the requestor process via another event flag. Finally, the requestor process will complete its end of the transaction by calling BREAKLINK.

A communications protocol was defined for the interprocessor buses that would tolerate any single bus error during the execution of any message system primitive. This design assures that a communications failure will occur if and only if the sender or receiver processes or their processors fail. Any bus errors which occur during a message system operation will be automatically corrected in a manner transparent to the communicating processes and logged on the system console. The interprocessor buses are not used for communication between processes in the same processor, which can be done faster in memory. However, the processes involved in the message transfer are unable to detect this difference.

The message system is designed such that resources needed for message transmission (control blocks) are obtained at the start of a message transfer request. Once LINK has been successfully completed, both processes are assured that sufficient resources are in hand to be able to

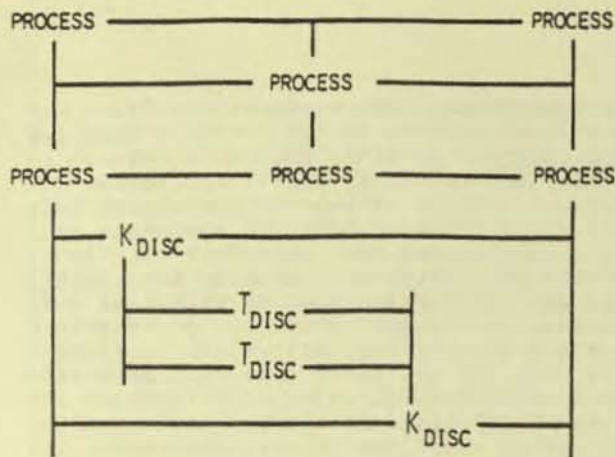
complete the message transfer. Furthermore, a process may reserve control blocks to guarantee that it will always be able to send messages to process a request that it picks up from its message queue. Such resource controls assure that deadlocks can be prevented in complex producer/consumer interactions, if the programmer correctly analyzes and anticipates potential deadlocks within the application.

3.3 Process-pairs

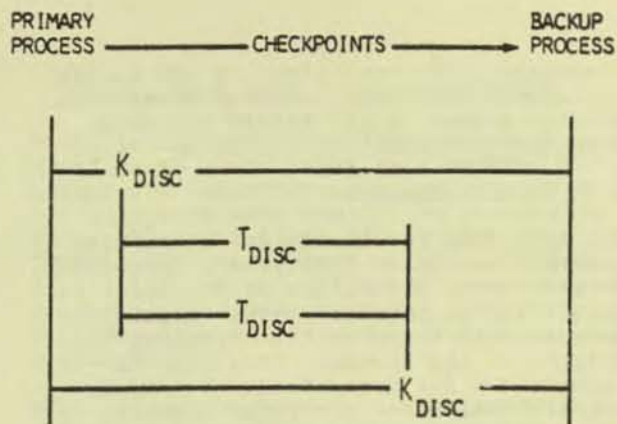
With the implementation of processes and messages, the system is no longer seen as separate modules. Instead, the system can be viewed as a set of processes which may interact via messages in any arbitrary manner, as shown in Figure 3.

By defining messages as the only legitimate method for process-to-process interaction, interprocess communication is not limited by the multiple-computer organization of the system. The system then starts to take on the appearance of a true multiprocessor. Processor boundaries have been blurred, but I/O devices are still not accessible to all processes.

System-wide access to I/O devices is provided by the mechanism of "process-pairs". An I/O process-pair consists of two cooperating processes located in two different processors that control a particular I/O device. One of the processes will be considered the "primary" and one will be considered the "backup". The primary process handles requests sent to it and controls the I/O device. When a request for an operation such as a file



System Structure After the Addition of Processes and Messages
Figure 3



Process-pair for a Redundantly-Recorded Disc Volume
Figure 4

open or close occurs, the primary will send this information to the backup process via the message system. These "checkpoints" assure that the backup process will have all information needed to take over control of the device in the event of an I/O channel error or a failure of the primary process' processor. A process-pair for a redundantly-recorded disc volume is illustrated in Figure 4.

Because of the distributed nature of the system, it is not possible to provide a block of "driver" code that could be called directly to access the device. While potentially more efficient, such an approach would preclude access to every device in the system by every process in the system.

The I/O process-pair and associated I/O device(s) are known by a logical device name such as "\$DISC1" or by a logical device number rather than by the processid of either process. I/O device names are mapped to the appropriate processes via the logical device table (LDT) in every processor, which supplies two processids for each device. A message request made on the basis of a device name or number results in the message being sent to the first process in the table. If the message cannot be sent or if the message is sent to the backup process, an error indication will be returned. The processid entries in the LDT will then be reversed and the message resent. Note two things: first, the error recovery can be done in an automatic manner; and second, the requestor is not concerned with what process actually handled the request. Error recovery cannot always be done automatically. For example, the primary process of a pair controlling a line

printer fails while handling a request to print a line on a check. The application process would prefer to see the process failure as an error rather than have the request automatically retried, which might result in two checks being printed.

The two primitives, processes and messages, blur the boundaries between processors and provide a failure-tolerant method for interprocess communication. By defining a method of grouping processes (process-pairs), a mechanism for uniform access to an I/O device or other system-wide resource is provided. This access method is independent of the functions performed within the processes, their locations, or their implementations. Within the process-pair, the message system is used to checkpoint state changes so that the backup process may take over in the event of a failure. This checkpoint mechanism is in turn independent of all other processes and messages in the system.

The system structure can be summarized as follows. Guardian is constructed of processes which communicate using messages. Fault tolerance is provided by duplication of components in both the hardware and the software. Access to I/O devices is provided by process-pairs consisting of a primary process and a backup process. The primary process must checkpoint state information to the backup process so that the backup may take over on a failure. Requests to these devices are routed using the logical device name or number so that the request is always routed to the current primary process. The result is a set of primitives and protocols which allow recovery and continued processing in spite of bus,

processor, I/O controller, or I/O device failures. Furthermore, these primitives provide access to all system resources from every process in the system.

3.4 System Processes

The next step in structuring the system comes in assigning functions to processes. As previously shown, I/O devices are controlled by process-pairs. Another process-pair known as the "operator" is present in the system. This pair is responsible for formatting and printing error messages on the system console. Here is an example of where Guardian has not followed a strict level structure. The operator makes requests to a terminal process to print the messages, yet the terminal process wishes to send messages to the operator to report I/O channel errors. An infinite cycle is prevented by having the terminal process not send messages for errors on the operator terminal and having I/O processes never wait for message completions when sending errors to the operator. While it may be preferable to prevent cycles of any type in system design, they have been allowed in Guardian when it can be shown that they will terminate. The ability to reserve message control blocks assures that no cycle will be blocked because of resource problems.

Each processor has a "system monitor" process which handles such functions as process creation and deletion, setting time of day, and processor failure and reload cleanup operations.

A memory management process is also resident in each processor. This process is responsible for allocating a page of physical memory and then sending messages to the appropriate disc processes to do the actual disc I/O. Pages are brought in on a demand basis and pages to overlay are selected on a "least recently used" basis over the entire memory of the processor.

The choice of relatively unsophisticated algorithms for scheduling and memory management was a result of the fact that the system was not intended to be a general-purpose timeshare system. Rather, it was to be a system which supported multiple processes and terminals in an extremely flexible manner.

3.5 Application Process Interface

Above the process and communication structure there exists a library of procedures which are used to access system resources. These procedures run in the calling process' environment and may or may not send messages to other processes

in the system. For example, the file system procedures do not do the actual I/O operations. Instead, they check the caller's parameters, and if all is in order a message is sent to the appropriate I/O process-pair. Likewise, process creation is seen as a procedure call to NEWPROCESS, which does nothing but check the caller's parameters and then send a message to the system monitor process in the processor where the process is to be created. On the other hand, a procedure such as TIME which returns the current time of day does not send any messages. In either case, the access to system resources appears simply as procedure calls, effectively hiding the process structure, message system, hardware organization, and associated failure recovery mechanisms.

3.6 Initialization and Processor Reload

System initialization starts with one processor being cold loaded from some disc on the system. The load file contains a memory image of the operating system resident code and data, with all system processes in existence and at their initial states. The system monitor process then creates a command interpreter process.

Guardian may be brought up even though a processor or peripheral device is down. This is possible because operating system disc images may be kept on multiple disc drives, I/O controllers may be accessed by two different processors, and the terminal that has the initial command interpreter on it is selected by using the processor's switch register.

After a cold load, the system logically consists of one processor and any peripherals attached to it. More processors and peripherals may be added to the system via the command interpreter command:

```
:RELOAD 1,$DISC
```

This command will read the disc image for processor 1 from the disc \$DISC and send it over either interprocessor bus to processor 1. Once it is loaded, all processes residing in other processors in the system will be notified that processor 1 is up.

This command is also used to reload a processor after it has been repaired. Guardian does not differentiate between an initial load of a processor and a later reload. In each case, resources are being logically added to the system and processes must be notified so that they may make use of them.

The previous example of a reload message being sent to all processes is an example of how functions are split in Guardian. A mechanism is provided for informing a process of a system status change. It may then take some unspecified action (including doing nothing). Similarly, a system power-on simply sets the PON event flag for all processes. The operating system kernel must only insure that the process structure and message system are correctly saved and restored. It is then the responsibility of individual processes to do such things as reinitialize their I/O controllers.

3.7 Operating System Error Detection

Besides the hardware-provided single error detection and correction on memory, and single error detection on the inter-processor and I/O buses, additional software error checks are provided. The first of these is the detection of a down processor. Every second, each processor in the system sends a special "I'm alive" message over each bus to all processors in the system. Every two seconds, each processor checks to see that it has received one of these messages from each processor. If a message has not been received, then it assumes that that processor is down.

Additionally, the operating system makes checks on the correctness of data structures such as linked lists when operations are done on them. Any processor detecting such an error will halt.

All I/O interrupts are bracketed by a "watch dog" timer such that the system will not hang up if an I/O operation does not complete with the expected interrupt. If an I/O bus error occurs then the backup process will take over control of the device using the second I/O bus.

As previously noted, the interprocessor bus protocol is designed to correct single bus errors. In addition to this, extensive checks are made on the control information received over the buses to verify that it is consistent with the state of the receiving processor.

Power-fail/automatic restart is provided within each processor. A power-failure is detected independently by each processor module and as a result is not a system-wide, synchronous event. The system was designed to recover from either a complete system power-fail, or a transient which will cause some of the processors to power-fail and then immediately restart.

4. USER-LEVEL SYSTEM INTERFACE

Tools are provided for interactive program development using COBOL or a block-structured implementation language, T/TAL. A file system with facilities comparable to or exceeding those offered by other "midi" computer systems allows access to disc files and other I/O devices. Process creation, intercommunication, and checkpointing primitives are also implemented.

The application process level facilities and the interactive program development tools have been heavily influenced by the HP 3000 (7) and by UNIX (8).

4.1 Interactive System Access

General-purpose, interactive access to the system is provided by the command interpreter, COMINT, similar in many ways to the Shell of UNIX. Normally a command interpreter is run interactively from a terminal, but commands may be read from any type of file. The command interpreter is seen by the operating system as simply another type of application process.

Commands are read from the terminal, prompted by a colon (":"):

```
: command / process parameters / arguments
```

If the command is recognized, it will be directly executed. A command of this type is:

```
:LOGON SOFTWARE.JOEL
```

which is used to gain access to the system. If the command is not recognized, then a process will be created using the program file "\$SYSTEM.SYSTEM.command" and the arguments for the command will be sent to this new process. The command interpreter will then suspend itself until a message is received indicating that the process has stopped. If this process cannot be created, then an error message is printed. For example, the text editor is accessed by typing EDIT followed by any command string:

```
:EDIT FILE
```

This will result in a process being created using the program file \$SYSTEM.SYSTEM.EDIT and the command string, "FILE", being sent to it. Also a part of this command string message are the names of the files that are being used for input and output by the command interpreter. These are then used by the process for its input and output. If the previous command was typed at a terminal, the input and

output files would be the device name of the terminal. Alternative names for the input and output files may be specified. For example:

```
:EDIT /IN COMMANDS/
```

will create an editor process and pass it the file name "COMMANDS" for the input file and the terminal's file name, the default, for the output file. Finally, the processor to use and the priority at which to run the process may also be specified:

```
:EDIT /PRI 100, CPU 3/
```

This will create an editor process in processor three with a priority of 100.

Additional features allow multiple processes to be started from one command interpreter and allow the previously typed command line to be edited.

4.2 Programming Languages

Compilers have been implemented for two languages, T/TAL, and ANSI 74 COBOL. T/TAL is a block-structured implementation language. Its capabilities are similar to those offered by C on UNIX or SPL on the HP3000. All Tandem software is written in T/TAL as are most user applications.

Code generated by either compiler may be shared by multiple processes in the same processor. Both compilers generate an object file which may be immediately run without any intervening link edit operation. However, the object file also contains enough information so that an object editor, UPDATE, may combine the objects produced by several compilations or selectively replace procedures in an object file.

4.3 Tools

Program development tools include an interactive text editor, object file editor, text formatter, and interactive debugger. A screen generation program and access routines are provided to facilitate application interaction with page mode CRT terminals. File utilities exist which allow file backup and restore, file copying and dumping, and initial loading of key-sequenced files. A peripheral utility is provided to do such operations as disc formatting, disc track sparing, and mounting or demounting disc volumes.

4.4 Process Creation and Deletion

Processes are created by the command interpreter or by an application process

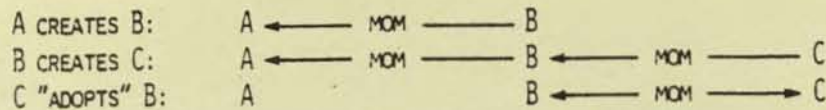
call to the procedure NEWPROCESS. Parameters supplied include the name of the file holding the object code for the process, the processor number to use, and the priority at which to run the process. The parameters will be checked and then sent to the system monitor process in the appropriate processor. The system monitor will then create the process and return a "creationid" identifying the new process to the calling process. Part of this value is the processid previously defined, and the rest is the value of the processor clock at the time of process creation. The clock is kept as a 48 bit value which is the number of 10ms intervals since 12 a.m. on December 31, 1975, which assures that creationid's will be unique over the life of the system.

Processes are not grouped in classical ancestry trees. No process is considered subservient to any other process on the basis of parentage. Two processes, one created by the other, will be treated as equals by the system. When a process, A, creates another process, B, no record of B is attached to A. The only record kept is in process B where the creationid of A is saved. This creationid is known as B's "mom". When process B stops, process A is sent a stop message indicating that process B no longer exists. A process's mom is flexible and a process may adopt another process. For example, (Figure 5), process A creates process B. Process B in turn creates a cooperating process, C. Since C would like to know if B stops, C will adopt B.

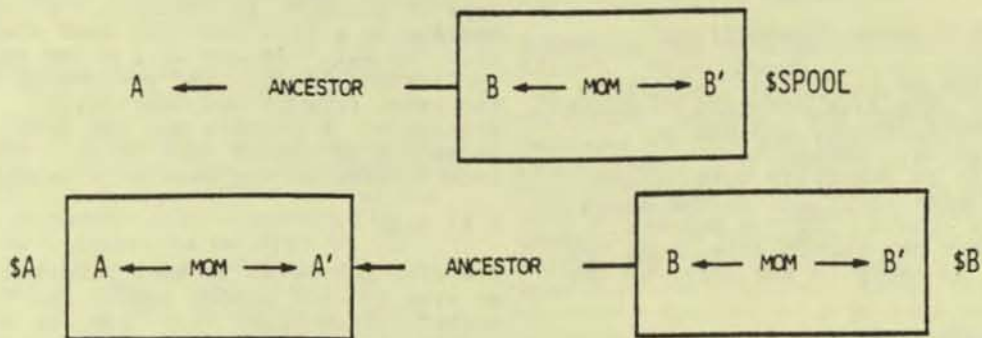
A process may stop itself or some other process by calling STOP. Process deletion is again a function of the system monitor process. Resources will be released and a stop message will be sent to the process' mom. If the mom process does not exist, then no message will be sent.

4.5 Application Process-pairs

The process-pair concept introduced earlier is a powerful method for making some resource available to all processes in the system in a fault-tolerant manner. It is extended to the application processes as follows. When a process is created via NEWPROCESS, a process-pair name may be supplied. The creationid returned for this process consists of the processid and the process name rather than the processor clock value. For example, (Figure 6), process A wishes to create a process with the name "\$SPOOL". Once B has been created, any process in the system may send a message to that process via the name "\$SPOOL".



Flexible Process Relationships
 Figure 5



Application Process-pairs
 Figure 6

Process B may now wish to create a process B' in another processor to be its backup. B would then call NEWPROCESS, supplying the name "\$SPOOL". Process B will keep B' updated via checkpoints so that B' may become the primary if B fails. B and B' each wish to receive an indication if the other process is deleted. Therefore, B and B' will be automatically set to be each other's moms.

When the last process with the name "\$SPOOL" is deleted, process A will be sent a message. Process A is known as the "ancestor" by the fact that this process was the one which created the first process by the name of "\$SPOOL". Process A in turn may be a named process, in which case A's name will be sent the termination message. This allows a process-pair, "\$A" consisting of processes A and A', to create a named process, "\$B" consisting of B and optionally B', and guarantee that it will be sent a message when the process name \$B no longer exists. This will occur even if the process which first created \$B no longer exists.

4.6 File System

The Guardian file system provides a uniform method for access to disc files, unit record devices, and processes. All files are named: disc files have names such as "\$DISC1.SUBVOL.FILE" and unit record devices and processes have names such as "\$LP". Access by name allows any process running in any processor to access any file in the system. Direction to the appropriate process of the process-pair is

handled by the file system in a manner transparent to the requesting process.

Files of all types are opened by calling:

```
CALL OPEN(filename,filenum,...)
```

The calling process supplies the file name. Security will be checked and then a file number will be returned to the calling process. This file number is then used for all further accesses to the file. A file may be opened for "wait" or "no-wait" access. If "wait" access is chosen, the process will be suspended until the requested operation on the file has been completed. On the other hand, if the "no-wait" access is requested then once the operation has been initiated, the requesting process may continue processing.

4.7 Disc Files

Each disc file is composed of between one and sixteen partitions. Each partition resides on a specific disc volume and is in turn composed of up to sixteen extents. Each extent is one or more contiguous disc pages of 2048 bytes each. Disc files comes in several types. The first is "unstructured", similar to UNIX, where the file is treated as a contiguous set of bytes. A current file pointer is kept which is the byte address of the beginning of the next transfer. After each read or write operation:

```
CALL READ(filenum,buffer,cnt,transfercnt)
CALL WRITE(filenum,buffer,cnt,transfercnt)
```

the file pointer is incremented by the number of bytes transferred. The file pointer may be moved explicitly by:

```
CALL POSITION(filenum,fileposition)
```

The second type of file is a "relative-record" file. The file consists of fixed or variable-size records and may be randomly accessed. Rather than positioning to an arbitrary byte in the file, the process positions to the start of a specific record. If the process reads less than the record size, then the file pointer advances to the start of the next record.

The third type of file is "entry-sequenced". Records written to this file may be of varying lengths and are always appended to the end of the file. This type of file is normally used as a log file.

The final type of disc file is "key-sequenced". A file of this type may have a unique primary key and up to 255 alternate keys. Entry-sequenced and relative-record files may also have alternate keys. Each key may be up to 255 bytes long. Access to this file may be done on any key using the procedure:

```
CALL KEYPOSITION(filenum,key,keytag,
                 keylen,positionmode)
```

The parameter "keytag" identifies which key is being used. The pointer "key" designates the value of the key which is "keylen" bytes long. The "positionmode" describes what type of access is to be made to the file. The first type of access is "approximate". Using this, successive reads to the file will return all records whose key values are greater than or equal to the "key" for "keylen". The second type of positioning is "generic". Here, successive reads will return all records whose key value is equal to "key" for "keylen". The final type of positioning is "exact". Successive reads will return all records whose keys are "keylen" long and equal to "key".

Files or individual records may be locked by:

```
CALL LOCKFILE(filenum)
CALL LOCKRECORD(filenum)
```

Record locking and unlocking may be combined with the actual I/O operation desired for increased efficiency:

```
CALL READUPDATELOCK(filenum,...)
CALL WRITEUPDATEUNLOCK(filenum,...)
```

The distributed nature of the system does not allow efficient detection of deadlocks caused by file locking. As a result, this type of checking is not done. A lock request on a file that has been opened with "no-wait" access will allow the application to do other processing if the requested file is not immediately available. A process may use this mechanism to assure that it will not wait indefinitely in the case of a deadlock.

4.8 Disc I/O

The disc processes in each processor share an area of main memory called the "disc cache". Each block read from the disc is placed in this area. Space is reused on a weighted "least recently used" basis. Thus, such items as index blocks for key sequenced files are kept available in memory so that successive accesses do not require that they be reread.

A logical disc volume, "\$DISC1", may be recorded onto two different disc drives using two different I/O controllers. This second, or "mirror" volume provides a transparent duplication of data which protects a data base against loss via a failed disc drive or controller. All file writes are performed on both disc drives and file reads may be done from either drive. When a failed drive has been repaired, it may be "revived" while the application continues accesses and updates to files on that logical device.

4.9 Device I/O

I/O operations are done to unit record devices in a similar manner to disc file accesses. Here, Guardian does not support a record-structured, device-independent mode of access and as a result operations such as unblocking tape records must be done by the application program. While this lack of device-independent I/O can be considered a liability in some applications, it allows easy addition of new types of I/O devices to the system without requiring changes to the file system and allows device-dependent control by the application program.

Read and write operations are done in an identical manner for all files. Device-dependent operations such as skipping on vertical form channels on a line printer may be done by:

```
CALL CONTROL(filenum,control#,parameter)
```

Enabling or disabling terminal parity checking or other such access options are done by:

```
CALL SETMODE(filenum,modetype,...)
```


Guardian provides an extremely general purpose interface to asynchronous RS-232 or current loop devices. The file system and asynchronous terminal process provide a read after write operation:

```
CALL WRITEREAD(filename,buffer,writesnt,
               readcnt,countread)
```

which allows a character sequence to be output to a device followed immediately by a read from the device. This allows the character sequence which causes a CRT terminal to transmit to be sent to it. The line will then be turned around and the terminal's buffer read into the processor. Since this write/read turn-around is done in the device controller, no data is lost because the read could not be started soon enough.

Normally, operating systems wish to enforce certain terminal characteristics such as inserting a carriage return and linefeed after each line written or interpreting certain characters on input for such operations as line and character delete. While Guardian provides such facilities, they may be disabled at the time the system is configured or after the file is opened. Other characteristics such as type of connection, character size, parity, speed, and character echoing are completely configurable. This allows arbitrary character sequences to be input and output without any interpretation or character editing being done by the operating system.

Communication software is also provided to handle multi-point asynchronous terminals. Point-to-point and multi-point Bisync software is also provided. Rather than attempting to emulate specific devices, the application program is allowed to specify the line control used.

4.10 Interprocess I/O

Each process in the system may have messages from other processes queued for it. Access to this message queue is provided via the file "\$RECEIVE". A read on this file will return the first message. A process may check to see if any messages are queued and then continue processing if none are present. A process may ascertain the identity of the sending process via the procedure:

```
CALL LASTRECEIVE(sender)
```

This returns the "creationid" of the sending process. It is supplied by the operating system and thus may not be forged by the sending process. A process will receive indication of such events as a process being stopped, a processor

failing or being reloaded, or the break key being pressed on a terminal that this process has open in the form of messages read from this file.

A process may open another process as a "file". Once opened, the process may use the file system procedures WRITE, WRITEREAD, SETMODE, and CONTROL to send messages to that process. The receiving process will read these requests from its "\$RECEIVE" file. It will then process them and possibly return an error indication to the sending process. This allows the "server" process to simulate some arbitrary device. Using these tools, an output spooler or a process which could allow access to labeled magnetic tapes written on some other system can be constructed. The requesting process believes that it is communicating with a device, and the server process is able to simulate that device without requiring special privileged "hooks" in the file system.

5. APPLICATION PROGRAMS

5.1 Application Program Checkpointing

Application process-pairs are used to provide some service on a failure-tolerant basis. Requests are processed by the primary process and results are returned to the requestor process. On a failure of the primary process, the backup must be able to continue offering this service. This requires that any state changes in the primary process be sent (checkpointed) to the backup process. While such checkpoints could be sent on an instruction-by-instruction basis, this is clearly not feasible because of the overhead involved. Instead, a process need only checkpoint its state when it is about to make a non-retryable request to another process.

For example, at time T1, when the primary process and its backup are in the same state, the primary process starts some operation. Later, at time T2, when it is ready to write the result to a disc file, it will checkpoint changes made since time T1 to its backup. The processes will then again be in the same state. If the primary process failed at any point before T2, the backup process could restart at the last checkpoint, made at T1. The selection of states to checkpoint is analogous to defining restart points for jobs in a batch processing system. In a batch environment, these checkpoints are saved in a disc file; in process-pairs they are saved in a backup process.

Guardian provides system functions for checkpointing process state information

between processes of a process-pair. The first type of item checkpointed is portions of the process' data space. This includes global data and/or the current stack, holding procedure return addresses, procedure local variables, and procedure parameters. Consider the following program segment, written in T/TAL, whose purpose is to output to a terminal the first 100 items of an array, "buffer":

```
FOR i := 1 TO 100 DO
  BEGIN
    CALL WRITE(terminal,buffer[i],itemlen);
  END;
```

This operation could be made failure-tolerant by two calls to the CHECKPOINT procedure. The first checkpoint copies the entire buffer to the backup process. This need only be done once as the data is not changed by later processing. The second checkpoint, before each write, saves the current process state, including the variable "i". This allows the backup process to take over the operation, duplicating at most one line of output.

```
CALL CHECKPOINT(,buffer[1],buffersize);
FOR i := 1 TO 100 DO
  BEGIN
    CALL CHECKPOINT(stackbase);
    CALL WRITE(outfile,buffer[i],itemlen);
  END;
```

When the primary process fails, the backup will take over at the last checkpoint. The next logical extension to the original segment would be if the process were copying the one hundred values to be output from some disc file:

```
FOR i := 1 TO 100 DO
  BEGIN
    CALL READ(infile,buffer,itemlen);
    CALL WRITE(outfile,buffer,itemlen);
  END;
```

In this case, not only would the process' data space contents need to be checkpointed as before, but so would the current file pointers for the input and output files. This ensures that they are correctly set when the backup process takes over. In order for file pointers to be checkpointed, both processes of the process-pair must have the files open. Special functions are provided which allow the primary process to cause a file to be opened or closed by the backup process:

```
CALL CHECKOPEN(filename,...)
CALL CHECKCLOSE(filename,...)
```

In the sample program, CHECKOPEN would be called following the call to OPEN when the primary process started. The program segment would now look like:

```
CALL CHECKPOINT(,buffer[1],buffersize);
FOR i := 1 TO 100 DO
  BEGIN
    CALL CHECKPOINT(stackbase,,infile,,
      outfile);
    CALL READ(infile,buffer,itemlen);
    CALL WRITE(outfile,buffer,itemlen);
  END;
```

If a failure occurred after the read but before the write, the backup would take over and repeat the read using the same file pointer as was used by the primary. In both of these examples, a failure following the write but preceding the next checkpoint could result in a record being written twice. This would cause no problem if the record was being written to some absolute position in the file; however, an error would occur when writing to a key-sequenced disc file. In this case, the primary would successfully write the record to the file, but its backup process would get a "duplicate key" error when repeating the write. This problem is solved by having Guardian automatically

ACTION	SEQUENCE VALUES AFTER ACTION:		
	PRIMARY	BACKUP	DISC PROCESS
CHECKPOINT(stackbase, ,infile, , outfile) sequence # of primary sent to backup	0	0	0
CALL WRITE(outfile, buffer, itemlen) sequence #'s match, operation is done, sequence #'s advanced	1	0	1
*** PRIMARY PROCESS FAILS, BACKUP TAKES OVER ***			
CALL WRITE(outfile, buffer, itemlen) sequence #'s don't match, operation is not done, backup's sequence # is advanced		1	1

File System Sequence Numbers
Figure 7

generate an optional sequence number for disc file writes.

A part of the information copied to the backup process when a file is checkpointed is the sequence number for the next write to the file. When a write is done to a file that has been opened with this option, the sequence number passed by the file system is compared with the copy held by the disc process. If it is the same, then the operation is done and the status (error indication and transfer count) is returned to the application process and a copy is saved by the disc process. On the other hand, if sequence numbers do not agree, then no operation is done and a copy of the previous operation's status is returned. Using the previous example, the use of file sequence numbers is shown in Figure 7.

When a process-pair has a file open, any records locked in the file will be considered locked by the process-pair. When the primary fails, the backup may finish file modifications with locks still in effect, preserving the integrity of the data base.

While the primary process operates, the backup process receives the checkpoint information via a call to the procedure CHECKMONITOR. When the primary process sends a checkpoint message via a call to CHECKPOINT, this procedure moves checkpointed portions of the primary process' data space into the backup's data space and saves the latest file information. If a message is directed to the backup process and the primary process still exists, it is rejected with a "ownership" error which informs the sender that the message is to be sent to the other member of the process-pair. Finally, when the primary process fails, CHECKMONITOR will transfer control to the correct restart point.

The Tandem implementation of COBOL provides a similar checkpointing facility. In each case, checkpointing is not an automatic operation. Careful attention during the application design phase will result in fewer checkpoints and will yield a checkpoint scheme that can be analyzed for correctness. Consideration must also be given to how the application will recover from failures occurring while a write operation is in progress to non-disc devices. Recovery when accessing a CRT terminal could be automatically done by rewriting the entire screen. Recovery while printing checks on a line printer would require some manual intervention and operator interaction with the application program.

5.1 Application Structuring

The process, process-pair, and inter-process communication primitives of Guardian provide extremely general tools for application structuring. For example, consider an inquiry application such as hotel reservations. Requests come in from various types of terminals for reservations, cancellations, and hotel registration. Other requests come in for items such as management reports showing the number of rooms available at some hotel on some date. The application could be structured as follows.

Process-pairs will be defined for each type of terminal to handle actual terminal I/O (including any required line protocol and character conversion) and initial request verification. Each process-pair will be designed to handle some number of terminals. When a valid request has been received from a terminal, the terminal process-pair will route the message to the appropriate server process-pair.

Each server process-pair will be assigned a certain part of the application. In some cases, multiple copies of a server program will be run to allow multiple requests to be processed in parallel.

There are several advantages to this approach. First, the handling of terminals and processing of requests have been cleanly separated. New types of terminals can be added by simply adding a type of terminal control process-pair. New types of requests can be handled by adding another type of server process-pair. Likewise, software modifications and testing can be done on a modular basis. Finally, nowhere in this structure is there any requirement for a specific number of processors in the system or for the relative locations of processes. As the system load or the application changes, the number of processors, amount of memory, or physical location of processes may be changed without disturbing the application's internal structure.

6. SOFTWARE RELIABILITY

When design of the operating system was started, we hoped to eliminate as much as possible the archetypal system crash. That is, once or twice a day, or week, the system crashes in a non-repeatable fashion. Our experience on an in-house system used primarily for software development and manual writing shows that we have achieved that goal. During a three-month period in the summer of 1977, a processor failed because of a software

problem on two occasions: In each case, the problem was found at that time and the failure could be repeated by running a particular program.

I propose the following explanation of this reliability. First, the system was very carefully structured and much time was spent in initially specifying primitives. As experience was gained in trying to apply these primitives at higher levels, problems were found. This resulted in design changes at lower levels rather than "kludges" at a higher level. Implementation was also forced to stay within the designed structures because of the distributed nature of the hardware. If a problem could not be solved using processes interacting via messages, then it could not be "kludged" by turning off interrupts and changing some flag in memory. Given a single processor system, there is a very strong temptation to solve difficult problems in this manner.

Second, as the operating system and hardware were developed at the same time, another vendor's system was used to provide interactive text editing, a cross T/TAL compiler, a Tandem/16 processor simulator, and a downloader for the Tandem/16 prototypes. Implementation and checkout were not impeded by unreliable prototypes and as each level of the system was implemented, it could be extensively checked. These tools allowed initial implementation and checkout of all functions of the system through the level of the command interpreter. The wisdom of this approach can best be shown by the fact that when the first prototype processors were made available to the operating systems group, all operating system functions which ran on the simulator ran on the prototypes.

Third, debugging tools were built into the operating system from the start. A low-level interactive debugger was implemented which allowed breakpoints to be set at any level of the operating system, including interrupt handlers. Once this low-level debugger is entered in one processor, clocks in all other processors in the system are stopped so that they will not decide that the first processor is down. When the first processor continues, so will the rest of the system. A full maintenance panel only had to be used to track problems that managed to destroy the low-level debugger. Consistency checks were also coded into low-level routines. For example, before an element is inserted in a doubly-linked list, the list links of the element that the new element is being inserted behind are verified. These checks have proved to be extremely valuable in tracking problems or when

implementing new features in the system. Even when extensive changes are being made to the system, it has the property that it will stop at one of these consistency checks very soon after something has gone wrong, allowing the problem to be rapidly found.

Fourth, formal testing was carried out at all levels of the system as they were implemented. A third person, whose only job was testing, was added to the initial project well before completion. By testing not just the external specifications of the system but also the underlying system primitives, it was assured that all system functions at all levels could be checked.

Finally, the primary design goal of the entire system was reliability. When the system design goals are clearly defined and understood by all involved, they can control implementation on a daily basis. Implied goals on the other hand are often forgotten when seemingly small decisions are made.

7. CONCLUSIONS

The innovative aspects of Guardian lie not in any new concepts introduced, but rather in the synthesis of pre-existing ideas. Of particular note are the low-level abstractions, process and message. By using these, all processor boundaries can be hidden from both the application programs and most of the operating system. These initial abstractions are the key to the system's ability to tolerate failures. They also provide the configuration independence that is necessary in order for the system and applications to run over a wide range of system sizes.

Guardian provides the application programmer with extremely general approaches to process structuring, interprocess communication, and failure tolerance. Much has been said about structuring programs using multiple communicating processes, but few operating systems are able to support such structures.

Finally, the design goals of the system have been met to a large degree. Systems, with between two and ten processors, have been installed and are running on-line applications. They are recovering from failures and failures are being repaired on-line.

8. ACKNOWLEDGEMENTS

An operating system is the work of many people. In particular I would like to acknowledge the contributions of Dennis

generate an optional sequence number for disc file writes.

A part of the information copied to the backup process when a file is checkpointed is the sequence number for the next write to the file. When a write is done to a file that has been opened with this option, the sequence number passed by the file system is compared with the copy held by the disc process. If it is the same, then the operation is done and the status (error indication and transfer count) is returned to the application process and a copy is saved by the disc process. On the other hand, if sequence numbers do not agree, then no operation is done and a copy of the previous operation's status is returned. Using the previous example, the use of file sequence numbers is shown in Figure 7.

When a process-pair has a file open, any records locked in the file will be considered locked by the process-pair. When the primary fails, the backup may finish file modifications with locks still in effect, preserving the integrity of the data base.

While the primary process operates, the backup process receives the checkpoint information via a call to the procedure CHECKMONITOR. When the primary process sends a checkpoint message via a call to CHECKPOINT, this procedure moves checkpointed portions of the primary process' data space into the backup's data space and saves the latest file information. If a message is directed to the backup process and the primary process still exists, it is rejected with a "ownership" error which informs the sender that the message is to be sent to the other member of the process-pair. Finally, when the primary process fails, CHECKMONITOR will transfer control to the correct restart point.

The Tandem implementation of COBOL provides a similar checkpointing facility. In each case, checkpointing is not an automatic operation. Careful attention during the application design phase will result in fewer checkpoints and will yield a checkpoint scheme that can be analyzed for correctness. Consideration must also be given to how the application will recover from failures occurring while a write operation is in progress to non-disc devices. Recovery when accessing a CRT terminal could be automatically done by rewriting the entire screen. Recovery while printing checks on a line printer would require some manual intervention and operator interaction with the application program.

5.1 Application Structuring

The process, process-pair, and inter-process communication primitives of Guardian provide extremely general tools for application structuring. For example, consider an inquiry application such as hotel reservations. Requests come in from various types of terminals for reservations, cancellations, and hotel registration. Other requests come in for items such as management reports showing the number of rooms available at some hotel on some date. The application could be structured as follows.

Process-pairs will be defined for each type of terminal to handle actual terminal I/O (including any required line protocol and character conversion) and initial request verification. Each process-pair will be designed to handle some number of terminals. When a valid request has been received from a terminal, the terminal process-pair will route the message to the appropriate server process-pair.

Each server process-pair will be assigned a certain part of the application. In some cases, multiple copies of a server program will be run to allow multiple requests to be processed in parallel.

There are several advantages to this approach. First, the handling of terminals and processing of requests have been cleanly separated. New types of terminals can be added by simply adding a type of terminal control process-pair. New types of requests can be handled by adding another type of server process-pair. Likewise, software modifications and testing can be done on a modular basis. Finally, nowhere in this structure is there any requirement for a specific number of processors in the system or for the relative locations of processes. As the system load or the application changes, the number of processors, amount of memory, or physical location of processes may be changed without disturbing the application's internal structure.

6. SOFTWARE RELIABILITY

When design of the operating system was started, we hoped to eliminate as much as possible the archetypal system crash. That is, once or twice a day, or week, the system crashes in a non-repeatable fashion. Our experience on an in-house system used primarily for software development and manual writing shows that we have achieved that goal. During a three-month period in the summer of 1977, a processor failed because of a software

problem on two occasions: In each case, the problem was found at that time and the failure could be repeated by running a particular program.

I propose the following explanation of this reliability. First, the system was very carefully structured and much time was spent in initially specifying primitives. As experience was gained in trying to apply these primitives at higher levels, problems were found. This resulted in design changes at lower levels rather than "kludges" at a higher level. Implementation was also forced to stay within the designed structures because of the distributed nature of the hardware. If a problem could not be solved using processes interacting via messages, then it could not be "kludged" by turning off interrupts and changing some flag in memory. Given a single processor system, there is a very strong temptation to solve difficult problems in this manner.

Second, as the operating system and hardware were developed at the same time, another vendor's system was used to provide interactive text editing, a cross T/TAL compiler, a Tandem/16 processor simulator, and a downloader for the Tandem/16 prototypes. Implementation and checkout were not impeded by unreliable prototypes and as each level of the system was implemented, it could be extensively checked. These tools allowed initial implementation and checkout of all functions of the system through the level of the command interpreter. The wisdom of this approach can best be shown by the fact that when the first prototype processors were made available to the operating systems group, all operating system functions which ran on the simulator ran on the prototypes.

Third, debugging tools were built into the operating system from the start. A low-level interactive debugger was implemented which allowed breakpoints to be set at any level of the operating system, including interrupt handlers. Once this low-level debugger is entered in one processor, clocks in all other processors in the system are stopped so that they will not decide that the first processor is down. When the first processor continues, so will the rest of the system. A full maintenance panel only had to be used to track problems that managed to destroy the low-level debugger. Consistency checks were also coded into low-level routines. For example, before an element is inserted in a doubly-linked list, the list links of the element that the new element is being inserted behind are verified. These checks have proved to be extremely valuable in tracking problems or when

implementing new features in the system. Even when extensive changes are being made to the system, it has the property that it will stop at one of these consistency checks very soon after something has gone wrong, allowing the problem to be rapidly found.

Fourth, formal testing was carried out at all levels of the system as they were implemented. A third person, whose only job was testing, was added to the initial project well before completion. By testing not just the external specifications of the system but also the underlying system primitives, it was assured that all system functions at all levels could be checked.

Finally, the primary design goal of the entire system was reliability. When the system design goals are clearly defined and understood by all involved, they can control implementation on a daily basis. Implied goals on the other hand are often forgotten when seemingly small decisions are made.

7. CONCLUSIONS

The innovative aspects of Guardian lie not in any new concepts introduced, but rather in the synthesis of pre-existing ideas. Of particular note are the low-level abstractions, process and message. By using these, all processor boundaries can be hidden from both the application programs and most of the operating system. These initial abstractions are the key to the system's ability to tolerate failures. They also provide the configuration independence that is necessary in order for the system and applications to run over a wide range of system sizes.

Guardian provides the application programmer with extremely general approaches to process structuring, interprocess communication, and failure tolerance. Much has been said about structuring programs using multiple communicating processes, but few operating systems are able to support such structures.

Finally, the design goals of the system have been met to a large degree. Systems, with between two and ten processors, have been installed and are running on-line applications. They are recovering from failures and failures are being repaired on-line.

8. ACKNOWLEDGEMENTS

An operating system is the work of many people. In particular I would like to acknowledge the contributions of Dennis

McEvoy, Dave Hinders, Jerry Held, and Robert Shaw in its design, implementation, and testing.

9. REFERENCES

1. Katzman, J. A., System Architecture for NonStop Computing, Compcon, (February 1977), pp 77-80.
2. Tandem Computers Inc., Tandem/16 System Description, 1976.
3. Bell, C. G. and Newell, A., Computer Structures: Readings and Examples, McGraw-Hill, Inc., (1971), pp 15-36.
4. Enslow, P. H. Jr., Multiprocessor Organization - a Survey, Computing Surveys 9, (March 1977), pp 103-129.
5. Dijkstra, E. W., The Structure of the "THE" Multiprogramming System, Comm. ACM 11, (May 1968), pp 341-346.
6. Brinch Hansen, P., The Nucleus of a Multi-programming System, Comm. ACM 13, (April 1970), pp 238-241, 250.
7. Hewlett-Packard Journal, January January 1973.
8. Thompson, K. and Ritchie, D. M., The UNIX Time-Sharing System, Comm. ACM 17, (July 1974), pp 365-375.

Biography. The author received his M.S. degree in Computer Science : Computer Engineering and B.S. degree in Statistics at Stanford University in 1972. Before joining Tandem Computers in 1974, he was employed by Hewlett-Packard. Member of the ACM.

TANDEM SORT

FEATURES

- Three modes of operation
 - Conversational
 - Via command files
 - Programmatic
- Record Input and Output
 - From/To an External File
 - From/To a User Application Program
- Support for multiple file types
 - Unstructured files
 - ENSCRIBE structured files
 - EDIT files
- SORT keys may be;
 - Ascending or descending
 - STRING or INTEGER
 - Multiple key fields per sort
- SORT Output can be;
 - Complete record
 - Record Sequence Numbers
 - Keys only

INTRODUCTION

The Tandem SORT program reorders a set of records according to the values of sort key fields defined within the records. SORT may be driven by a set of commands conversationally, or by a text file containing the commands, or it may be called from your program.

Records may be passed to SORT from a file, or sent one by one through procedure calls from your program. Similarly, the sorted set of records may be written to a file, or your program may call a procedure to retrieve the records, one per call.

Actual sorting runs as a separate process from the host program. Standard interface procedures are present in the Sort Command Interpreter program or called from your program, which handle process creation, control, and communication.

CONVERSATIONAL MODE

For this mode simply type,

```
:SORT
```

and wait for the prompt "<". Six commands are allowed:

FROM – names the file of unsorted records and describes the records;

TO – names the file for the sorted records and selects output options;

ASCENDING – describe the location and attributes of the sorting keys within the records;
DESCENDING

RUN – starts the actual sort.

EXIT – exits from the Sort Command Interpreter.

Example:

```
:SORT/OUT $lp/  
<FROM insort,RECORD 56  
<TO outsort  
<ASC 6:12 INTEGER  
<RUN  
<EXIT  
:
```

TANDEM SORT

TANDEM SORT

COMMAND FILE MODE

For this mode, use EDIT to put your sort commands, one to a line, in a command file

```
:SORT [/IN <command file>]
      [, OUT <list file>] /]
```

SORT reads the commands from <command file>. When a RUN command is found, and the previous commands describe a valid sort, SORT begins. Once the SORT completes, <command file> is read for more commands. End-of-file on <command file> terminates SORT.

Example:

```
:SORT/IN myfile,OUT $lp/
```

Contents of "myfile";

```
FROM datain
TO dataout
ASC 1:3 INTEGER
DESC 10:16 STRING
RUN
ASC 88:89
FROM names
DESC 22:35
TO sortout
RUN
```

PROGRAM SORTS

There are four ways to use SORT in your program.

1. SORT Input from External File
SORT Output to External File
2. SORT Input from External File
SORT Output to Program, one record at a time
3. SORT Input from Program
SORT Output to External File

4. SORT Input from Program
SORT Output to Program, one record at a time.

Five SORT Interface Procedures are provided.

SORTSTART — Initiate a sort.

SORTSEND — Send input record to SORT

SORTRECEIVE — Retrieve a sorted record

SORTFINISH — Complete the sort

SORTERROR — Format a SORT error message

GENERAL METHOD OF OPERATION

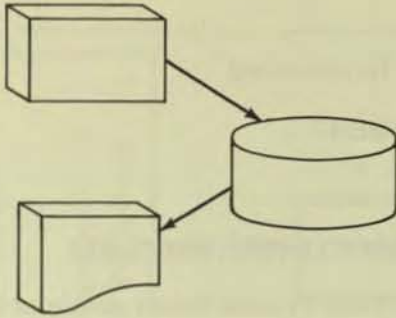
Two things determine how the sort functions internally:

- the amount of data to be sorted,
- the amount of memory you give.

If the amount of data is small enough, the data is sorted within memory.

In most large sorts, the memory is insufficient to sort all the data at once, so SORT splits the input data into sorted pieces it can handle, and puts them in a scratch file. Each piece is referred to as a *run* since the records in each one "run" in sequence.

When there is more than one run formed from the initial data, the sorted output is produced by merging the runs together. This method of sorting is known as "Replacement/Selection".



TANDEM SPOOLER

- SPOOLER runs NonStop^(tm)
- User can supply own Print Processes
- SPOOLCOM allows operator inspection and/or alteration of job parameters
- Routing structure permits individual or broadcast locations
- SPOOLER library procedures allow blocking and compression
- Multiple files can be spooled from one application
- SPOOLER parameters can be specified programmatically
- Forms Alignment

INTRODUCTION

The Tandem SPOOLER provides a means of storing application output in holding areas for later retrieval. Output may be passed to other processes or printed on one or more devices.

The SPOOLER is actually many processes working in unison to provide spooling facilities. These processes include SPOOLER Supervisor, SPOOLER Collectors, Print Processes, and SPOOLCOM.

SPOOLER Supervisor functions as the SPOOLER monitor and communicates with the other SPOOLER processes to determine which tasks to perform or schedule. SPOOLER Control is actually a server process interfacing with 1) SPOOLCOM, 2) applications calling SPOOLERCOMMAND or SPOOLERSTATUS procedures, 3) SPOOLER Collection Processes, and 4) SPOOL Print Processes.

SPOOLER Collectors accept output from application processes and store it on disc. There can be one or more SPOOLER Collectors.

Print Processes retrieve spooled data and print it. The Tandem supplied print processes are capable of handling multiple jobs and devices. Users may supply their own Print Processes.

SPOOLCOM is an operator/user interface with the SPOOLER subsystem. It can be run interactively on a terminal or can be passed commands from an application process. SPOOLCOM performs such functions as downing a device or ordering extra copies of a report.

APPLICATION PROCESS INTERFACE

At the simplest level an application process can open the SPOOLER one or more times to perform spooled output. The standard file management procedures WRITE, CONTROL, and SETMODE are used.

The application may also use the SPOOLER library procedures to implement more advanced features of the SPOOLER. The library procedures are:

- SPOOLSTART
 - start a job and specify spooling attributes
- SPOOLWRITE
 - write a print line
- SPOOLEND
 - end the spool job
- SPOOLCONTROL
 - control functions
- SPOOLSETMODE
 - setmode functions

Through SPOOLSTART the application can specify location, form name, priority of printing, number of copies, report name, and hold before/after printing.

TANDEM SPOOLER

TANDEM SPOOLER

SPOOLCOM

SPOOLCOM is used to initiate and control the operation of the spooling system by accepting commands interactively from an operator at a terminal or programmatically from a user application process. SPOOLCOM commands include:

- DEV
 - controls devices
- JOB
 - control jobs
- PRINT
 - control Print Processes
- COLLECT
 - control collection processes
- LOC
 - set up and modify destination structure
- SPOOLER
 - control the Spooling System
- HELP
 - list SPOOLCOM Commands
- EXIT
 - terminate SPOOLCOM
- SPOOLERCOMMAND
 - issue a control command
- SPOOLERSTATUS
 - retrieve status information
- SPOOLERREQUEST
 - Request information for a specific job necessary to start printing.

FC

- fix command

COMMENT

USER-WRITTEN PRINT PROCESSES

The Print Process library procedures allow user-written processes for devices not supported by SPOOLER or the retrieval of spooled data by an application process. The procedures are:

PRINTINIT

- Initialize communication with the SPOOLER control process

PRINTCOMPLETE

- Accepts messages from the SPOOLER control process

PRINTREADCOMMAND

- Interprets control messages from the control process

PRINTSTART

- Initialize data required to print a job

PRINTREAD

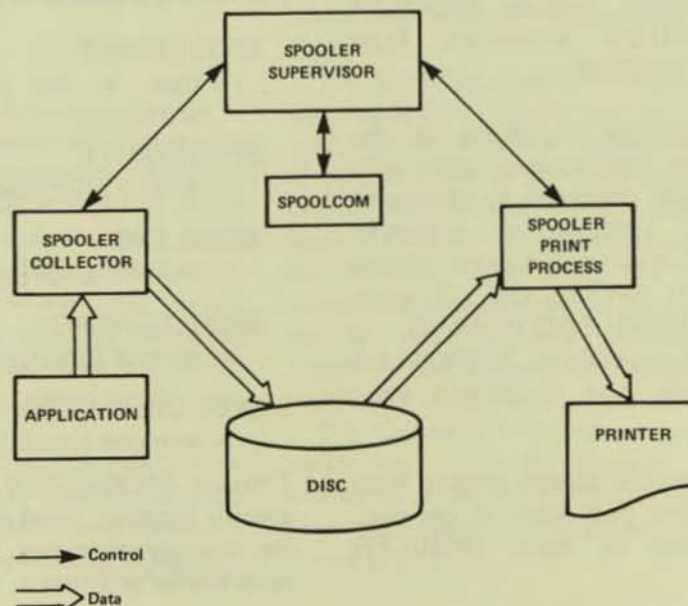
- Read a line of spooled data

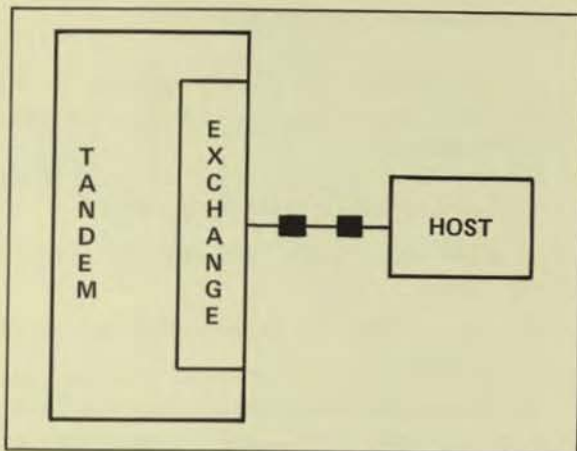
PRINTINFO

- Obtain information on an active job

PRINTSTATUS

- Inform SPOOLER control process of an event such as end of file or error on the device.





TANDEM

EXCHANGE REMOTE

JOB ENTRY

SUBSYSTEM

FEATURES

- Supports 2780 and 3780 Emulation
- Batch input from any media
 - terminal
 - magnetic tape
 - disc
 - card reader
 - another process
- Batch output to any media
 - terminal
 - magnetic tape
 - disc
 - line printer
 - another process
- Three Modes of Operation
 - Conversational
 - Command File
 - Programmatic
- File transfer ability between Tandem Systems
- NonStop Operation Optional

INTRODUCTION

EXCHANGE is a Tandem Subsystem designed to allow a Tandem System to emulate the functions of a 2780 or 3780 remote batch workstation. Input and output can be from/to any media supported by the Tandem System including disc, magnetic tape, terminals, card readers, line printers, and other processes. Commands to define the input, list and punch files, and characteristics about the connection can be accepted conversationally from a terminal, from a Command File, or programmatically from another process.

General capabilities of EXCHANGE include transmitting and receiving in ASCII or EBCDIC, accepting horizontal tab codes, accepting vertical forms control codes, transmitting or receiving EBCDIC transparent

data, short record truncation, blank field compression, transmitting and receiving blocked data link messages, and generation of WACK and TTD control codes when temporarily unable to transmit or receive.

Another useful feature is the ability to transfer files between two workstations. An EXCHANGE subsystem on one Tandem can perform remote file transfers to another Tandem which is also running the EXCHANGE subsystem.

EXCHANGE, at the user option, may be run in a NonStop mode.

EXCHANGE STRUCTURE

EXCHANGE is composed of two program modules; a Server Process and a Command Interpreter.

The Server has all data link handling responsibility and responds to requests to send or accept data to/from a remote system. The Server is typically run in conjunction with the Command Interpreter but can be called directly from an application process. Send and receive data and initial connection parameters are passed to the Server. The Server handles all message assembly/disassembly, blank compression/decompression, horizontal tab expansion, record truncation, and character set translation. The Server accepts and delivers data on a record-by-record basis.

The Command Interpreter provides a conversational interface to the operator. Commands are entered to control the connection type, specify the receive file names and parameters, and identify the files to be sent.

EXCHANGE COMMAND INTERPRETER

The EXCHANGE Command Interpreter allows the user to send or receive files by entering commands at a terminal or through a command file. Commands which can be entered are:

- CONNECT – defines remote terminal connection
- SEND – defines file to be sent
- RECEIVE – defines the receiving file
- ABORT – stops any transfer in progress
- DISCONNECT – ends connection to host
- EXIT – exits from EXCHANGE Command Interpreter
- STATUS – displays line status and statistics

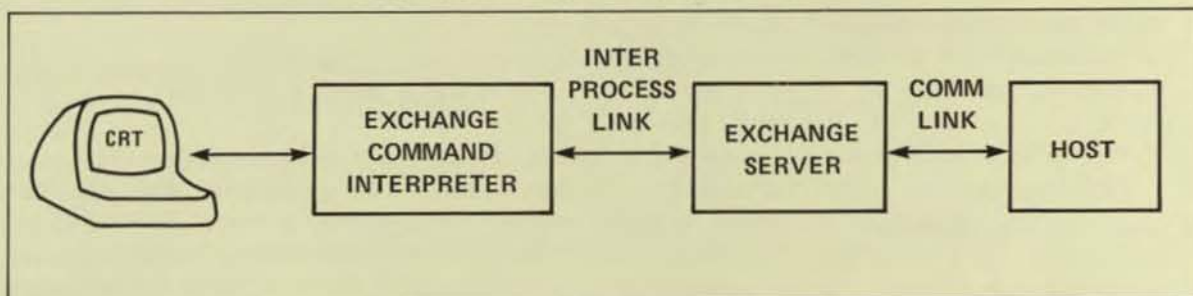
PROGRAMMATIC INTERFACE

A user process may directly interface to the EXCHANGE Server. The required steps in the application process are:

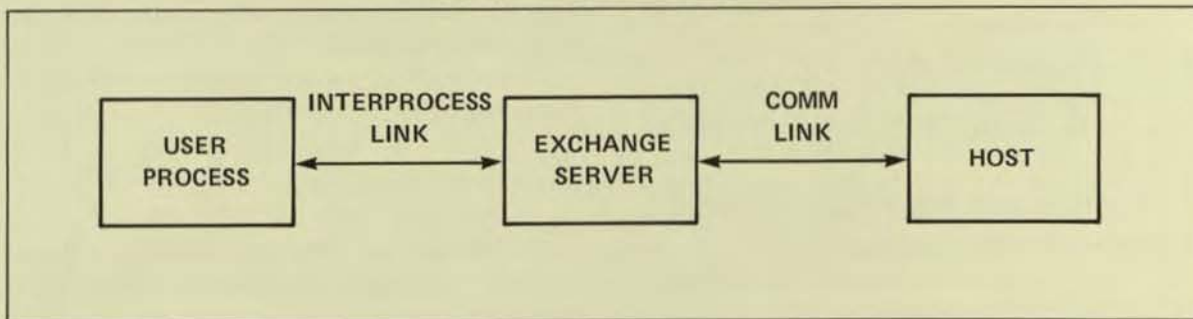
1. Open an interprocess file to the Server.
2. Establish the receive file character set via SETMODE.
3. Establish a data link to the remote system via SETMODE.

At this point the application may send/receive records via simple READ/WRITE type statements. Data link message formatting is handled by the Server, and compressed blanks and embedded horizontal tabs are expanded.

COMMAND INTERPRETER INTERFACE



PROGRAMMATIC INTERFACE



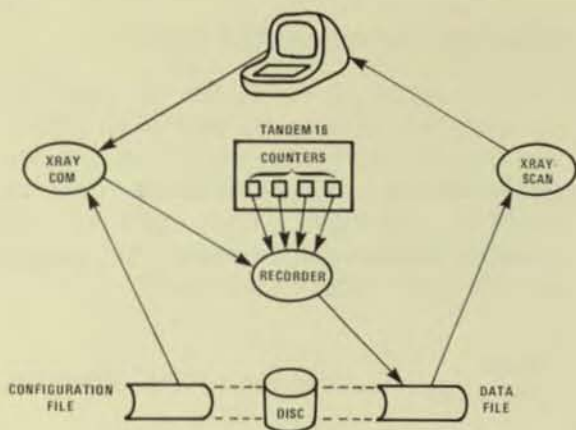
NONSTOP FEATURES

The EXCHANGE Server can be made to run Non-Stop. In the NonStop mode of operation EXCHANGE will:

1. Maintain a connection between the User, the EXCHANGE Command Interpreter, the EXCHANGE Server, and the Communication Data Link.
2. Maintain a major state; connected, disconnected, connecting.

In the NonStop mode of operation each process checkpoints all major state changes. This implies that:

1. Connected or disconnected status is maintained after a failure of either process.
2. Connection attempts are retried if a failure occurs during a connect.
3. If a failure occurs in either process SEND and RECEIVE commands are aborted, but the communication link is maintained.



TANDEM XRAY

PERFORMANCE MONITOR FOR TANDEM/16 COMPUTER SYSTEM

- Diverse applications include Mix Balancing, Growth Management, Online Monitoring and Application Tuning
- Provides data usable for resolving a variety of performance issues
- Does not require any special hardware or display devices
- Includes 2 programs: XRAYCOM for measurement control, and XRAYSCAN for data reduction and analysis
- Self-measuring capability shows the exact amount of perturbation XRAY's measurements are causing in the system

INTRODUCTION

XRAY is a tool for monitoring the performance of a TANDEM/16 computer system. The applications of XRAY span the following areas:

Mix Balancing — the distribution of applications across system hardware so as to eliminate bottlenecks.

Growth Management — the long-term appraisal of system component usage for planning, budgeting, and control purposes.

On Line Monitoring — permitting immediate detection of performance difficulties and providing continuous feedback on system usage.

Application Tuning — highlighting where the application program should be restructured to increase transaction throughput.

The data provided by XRAYSCAN in the form of time plots and reports can be used to resolve a variety of performance issues, for example:

- The usage of the system components (i.e., cps, memories, discs, communication lines) can be monitored on a long-term basis, permitting an orderly management of growth.

- Programs can be distributed among the processors for the most effective use of cpu and memory resources.
- Data base files can be distributed among the system's disc volumes to avoid bottlenecks at a single spindle.
- Programs responsible for excessive cpu, message, or virtual memory activity can be pinpointed.
- Data base files can be restructured for speedy access, and the optimum disc cache size can be determined.
- Response times can be tracked to provide an objective measure of system performance.

ADDITIONAL EQUIPMENT

XRAY will run on a minimum TANDEM/16 with the GUARDIAN Version C operating system and firmware. XRAY does not require ENSCRIBE OR ENVOY, but is fully integrated with both so that all data base and data communications activity can be measured or monitored. XRAY does not require any special hardware or display devices: it can be operated from any asynchronous, point-to-point terminal with a line width of at least 80 characters.

PROGRAMS

The program XRAYCOM is used to control measurements. Data is collected in a disc file. System performance is analyzed by running the program XRAYSCAN against the collected data; thus XRAYSCAN is used for data reduction and analysis.

Online performance monitoring is achieved by simply running XRAYSCAN against the currently active disc file. Any item or set of items in the measurement can be plotted on a terminal, as they are observed.

TANDEM XRAY

TANDEM XRAY

An important feature of XRAY is that it can measure itself, indicating exactly how much the system was perturbed by the measurements. This is possible because XRAY allows measurement of individual processes on the system. When measurements are being taken, XRAYCOM installs a data collection process, called the RECORDER, in each processor. XRAY can be directed to measure each RECORDER, letting the user know exactly how much overhead XRAY entails. (Typically, this is less than 1% of the processor's workload during the time of the measurement).

XRAYCOM has five commands:

- DATA designates the disc file in which the RECORDERS store measurement data
- CONF specifies a configuration file, which lists the system components to be measured
- GO starts the measurement and specifies the time interval at which the RECORDERS write measurement data to the data file
- EXIT exit from XRAYCOM. The form "EXIT!" stops the measurement
- LIGHTS displays processor utilization on the cpu panel lights

A typical sequence of commands used to start a measurement would be:

```
: XRAYCOM
+ CONF xrayconf
+ DATA xraydata
+ GO 10
+ EXIT
```

Any of the commands can be issued at any time during a measurement, permitting the measurement configuration, the data file, and the time interval to be changed.

The items to be measured can be extended beyond basic device utilizations by configuring particular sets of files and programs for measurement. The configuration is placed in a EDIT-type file for input to XRAYCOM. A typical configuration file looks like this:

```
PROGRAMS: SYSPROCS
          $SYSTEM.SYSTEM.*
          $DEVELOP.TESTPROG.*
FILES:    $ORDERS.*.*
          $ADMIN.BUDGET.*
```

EXAMINING THE COLLECTED DATA

The program XRAYSCAN is used for exploring and filtering the data in an XRAY data file. Any particular part of the data file can be selected for examination with the WINDOW command. Then, sets of the measured entities can be chosen for perusal with the entity selection commands.

Report Command	Measured Entities Reported
CPU	processors
LINE	data communication lines
DEVICE	tapes, printers, etc.
TERMINAL	terminals
DISC	discs, and disc file opens
PROCESS	processes
FILE	file opens

For each of the above report commands, a set of items is displayed. The items depend on the nature of the component being reported on. As an example, the items associated with the CPU report command include CPU BUSY, SWAP RATE, DISC RATE, CHIT RATE, SEND RATE, CPU QLEN, DISP RATE, TRAN RATE and RESP TIME.

The XRAY report can be restricted to entities having values in particular ranges. For example, the command

```
+PROCESS 1, IF CPU BUSY > 1.5
```

restricts the report to those processes in cpu 1 which have used 1.5% of the cpu or more during the portion of the measurement being examined.

The current report can be displayed with the entities sorted on any item. This is done by naming the item in a BY clause:

```
+FILE $ORDERS.*.* ,BY FILE RATE
```

The files accessed on \$ORDERS during the WINDOW will be displayed, in order of most active file to least active.

Many reports and plots will routinely be printed on a hard copy printer. At a CRT, a hard copy of the last report or plot can be made with the COPY command.

The ability to plot any item over time is an integral function of XRAY. Plots are used to find exception conditions, detect counter overflows, and associate correlated counters. It is simple to generate a plot, and a user may elect to plot any set of items on the same set of axes.

Transaction
Processing

PATHWAY TRANSACTION PROCESSING SYSTEM

The PATHWAY Transaction Processing System combines a set of special Terminal Control Processes, a new screen formatting language, a user-controlled application monitor, and an interactive screen definition facility to provide for Tandem users' significant reductions in programming requirements and simplicity in development requirements for on-line transaction processing applications.

PATHWAY takes full advantage of Tandem's fault-tolerant capabilities to assure data integrity and continuous availability.

Other features include:

- . the ability to access multiple applications from the same terminal
- . the capability to perform on-line addition, modification or deletion of transaction types, screen definitions, applications and terminals

Until now, a programmer typically has had to spend a large portion of development time addressing terminal characteristics. PATHWAY software provides the necessary procedures, programs, and structures to relieve him of these tasks, thereby increasing productivity and also making on-line application development more accessible to a broader base of programmers. On-line applications become as easy to write as batch applications, opening up on-line transaction processing to a whole new world of users and applications.

Simplifying application design and programming even further, PATHWAY divides terminal control and file manipulation into separate programs, which means the user need only be concerned with "single-threaded processing." The Tandem supplied software performs all necessary data checking and format validation.

With the PATHWAY system, all terminal oriented functions are isolated within the terminal control processes (TCPs) supplied by Tandem. Although a single TCP can control multiple terminals, each terminal is logically independent of the others, even to the extent of maintaining distinct data areas and control information.

PATHWAY

NonStop™ TRANSACTION PROCESSING SYSTEM

PATHWAY software gives users the only transaction processing system capable of NonStop™ operation, assuring data integrity and continuous availability. Specifically designed to take full advantage of unique multiprocessing capabilities of the TANDEM 16 computer, **PATHWAY** software eases the task of developing applications for the on-line transaction processing environment.

With the **PATHWAY** system, programmers no longer need to be concerned about terminal characteristics when writing applications programs. TANDEM supplies all the programs, procedures and application structures necessary to get user-written applications up and running in less time.

PATHWAY software features:

- The only NonStop Transaction Processing System assuring continuous availability and data integrity
- Impressive software development tools designed to significantly reduce the cost of applications development
- Application structures that ease the task of designing and maintaining programs
- Increased transaction throughput by utilizing the full advantages of the unique fault-tolerant multiprocessing capabilities of TANDEM
- True distributed processing through the TANDEM EXPAND Network, allowing applications to run in any CPU in any system, regardless of the physical location of terminals and data
- Access to multiple applications from the same terminal for increased flexibility
- All the necessary terminal handling software allowing the user to concentrate on application design
- On-line addition, modification or deletion of transaction types, screen characteristics, applications and terminals.

NonStop Transaction Processing System

Designed to simplify the way TANDEM 16 users develop on-line transaction processing applications, the **PATHWAY** system is an important addition to the total TANDEM product line. **PATHWAY** software is a key development tool which assists in bringing up new applications faster and easier.

The **PATHWAY** software package supplies all the procedures, programs and application structures necessary to allow users to write single-threaded application program modules. Furthermore, the user-written application modules can be designed without concern for terminal characteristics and communications protocols.

TANDEM's **PATHWAY** software furnishes all the major components necessary to implement a NonStop Transaction Processing System: a terminal control process, a COBOL-like screen language, an application monitor, and an interactive screen definition facility.

An Impressive Terminal Control Process

All terminal oriented functions are isolated into TANDEM supplied Terminal Control Processes (TCPs). Each TCP interacts with one or more terminals, providing each terminal with a "center of control" where the overall processing flow of that terminal is supported.

The TCP performs four major application functions: terminal interface (multi-terminal I/O handler), field validation (data consistency checks), data mapping (data conversion and formatting), and transaction control (application scheduling and transaction flow).

Although a single TCP can control multiple terminals, each terminal is logically independent of the others. The TCP maintains distinct data areas and control information for each terminal, and several terminals may be performing the same or different application functions at any one time. The TCP automatically allocates the use of shared resources.

Users of **PATHWAY** software may run multiple TCPs for better distribution of available resources. In addition, users may start and stop TCPs on-line in response to changes in the transaction environment.

A Powerful, COBOL-Like Screen Formatting Language

The handling of each terminal is defined by the user in a high-level language known as Screen COBOL. With Screen COBOL (a subset of ANSI '74 COBOL with extensions optimized for screen handling), the user defines the screen formats, input and output data mapping, data validation and consistency checks, transaction routing and overall application control of the **PATHWAY** system. The Screen COBOL compiler produces an intermediate code file, which is executed by the Terminal Control Process.

A Screen COBOL program may be simple, defining one or two screen formats and handling a single application. Or, it may increase in complexity, defining many screen formats and handling many transaction types. However, a Screen COBOL program is written once for a single terminal type and may be executed multiple times to support multiple terminals of the same type.

A User-Controlled Application Monitor

Overall control of the **PATHWAY** system is carried out by the Application Monitor, a TANDEM-supplied program which is used to supervise and control all the working processes in the transaction processing system.

The Application Monitor is a multi-terminal control process which manages load sharing to eliminate bottlenecks. The Application Monitor performs on-line addition, modification or deletion of transaction types, screen formats and terminals — all under user control.

The Application Monitor is the first program that is run to bring up the transaction processing system. From that point, the Application Monitor controls the start-up of all other working processes of the system. The user first defines the Application Monitor parameters, characteristics of the terminals, terminal control processes, and user-written application processes of the system. The user can then start, stop, and alter the operation of the processes and devices. The Application Monitor reports error conditions in the system, and can display on command the status of the various system components.

Interactive Screen Builder

The **PATHWAY** Transaction Processing System includes an interactive screen builder facility that allows for the design of screen formats directly on the terminal screen. The screen builder program then generates the appropriate Screen COBOL source statements that describe the screen format. The new screen description may then be added to an existing Screen COBOL program — to add an extension to an existing application or to become the basis for a new Screen COBOL program.

The screen builder will also take an existing Screen COBOL screen description, display the screen, and allow the designer to make modifications directly at the terminal. This procedure allows the designer and the ultimate terminal user to prepare screens together, and further reduces the time required to bring an on-line application into production.

Writing Applications with PATHWAY Software

The **PATHWAY** system makes the task of writing on-line applications as simple as writing batch programs. Under control of the Screen COBOL program, the TCP passes a transaction record to a user-written application program which deals with the record very straightforwardly. It receives the record, processes it in a single-threaded manner (making accesses to the ENSCRIBE data base as required) and replies to the TCP. The application program then waits for the next request.

The transaction record sent to the application has already passed the data validation checks, and is independent of the type of terminal that originated the transaction. The applications can therefore handle several terminal types.

For ease of design and programming flexibility, these user applications may be written in COBOL, FORTRAN, MUMPS or T/TAL (TANDEM/Transaction Application Language.)

The **PATHWAY** Transaction Processing System allows multiple copies of an application to be run for higher throughput. In addition, **PATHWAY** software contains facilities to dynamically start and stop application modules in response to changes in the transaction environment.

PATHWAY software also allows many different application programs to be present in the same system — even accessible from the same terminal — allowing a highly modular approach to application design and providing phased, on-line application development.

TANDEM's **PATHWAY** Transaction Processing System can support terminals of several different types including the TANDEM 6520 multi-page display terminal and 3270-compatible terminals.

The goal of the **PATHWAY** software is to simplify the design and programming of applications used in on-line transaction processing environments, where predefined transactions originate at terminals to access and update the data base. The **PATHWAY** system eases the burden of developing these applications on a TANDEM computer by providing complete terminal handling and application monitoring software.

TANDEM

TANDEM COMPUTERS INCORPORATED, 19333 Vallco Parkway, Cupertino, CA 95014. Toll Free (800) 538-9360 or (408) 996-6000 in California. Offices throughout the United States, Canada, Europe and the United Kingdom. Distributors in Australia, Finland, Mexico and Venezuela.

TANDEM

ENTRY SCREEN FORMATTER

FEATURES

- Interactive Page-Mode Form Creation on Video Display Terminal for Non-Stop Transaction-Oriented Applications
- Efficient and Economical Forms Creation with Automatic or Manual Cursor Positioning, Field Delimiters, Automatic Validity Checking, and Easy Transfer to Disc Files
- Fast and Error-Free Forms Display with Automatic Data Entry Validity Checking
- Individual Field Access with Length and Validity Check Attributes for User-Defined Error Checking
- Tandem-Supplied SCREEN Program to Create New or Modify Old Forms and to Insert Multiple Forms in a Single Disc File

INTRODUCTION

The ENTRY Screen Formatter provides a simple easy-to-use method of creating and displaying user-defined forms on an interactive page mode video terminal. Extensive validity and error checking is accomplished via the Tandem-supplied ENTRY procedures. When an error has been made in the creation or data entry processes, a flashing cursor is positioned over the field in error and an error message is printed to specifically flag the error.

Entry essentially performs four tasks:

- Creates Forms
- Programmatically Displays Forms
- Programmatically Accesses Individual Fields
- Tests Forms Independent of Application Program

In addition, ENTRY provides the facilities to: modify or update old forms, insert multiple forms in a single disc file, and provide length and field validity checking attributes for user-defined error checking.

FORMS CREATION

Forms creation is performed on a page mode terminal. The user simply designs the form on the screen as it is to appear when used. Fields on the form where terminal operator entries are to be made are indicated by delimiters. Once the form image is created, the user specifies a field name and validity checking attribute for each field. When completed, the form image is written to a designated file on disc.

FORMS DISPLAY

Forms display, forms read and field checking are all performed by application programs through calls to ENTRY procedures. When a form is displayed on a terminal, the operator fills in the fields on the form and presses a function key to transmit the fields into the computer where ENTRY performs the validity check on each field. If a field contains invalid data, the form can be programmatically redisplayed with the invalid entry flashing on the screen.

FIELD ACCESS

Individual fields in a form may be referenced by an application program. A field is referenced by the name assigned when the form was defined. Additionally each field has a length attribute and a validity checking attribute for user-defined error checking.

SCREEN PROGRAM PROCEDURES

ENTRY provides the following procedures to aid terminal I/O with forms created by the SCREEN program:

- EXPAND ^ SCREEN – initializes the applications I/O buffer with the control and data characters required to output a screen to the terminal
- READ ^ SCREEN – fills the applications I/O buffer with the control sequence required to input the operator entry fields from a screen
- CHECK ^ SCREEN – moves the operator entry fields from the applications I/O buffer into the correct fields and does the required validity checking
- BLINK ^ SCREEN – fills the applications I/O buffer with the control sequence required to turn the "blinking" on or off for a specific operator entry field
- POSITION ^ SCREEN – fills the applications I/O buffer with the control sequence required to position the cursor over a specific operator entry field
- FL ^ SCREEN – is used to compute the actual length of the field input by the operator

ERROR MESSAGES

When a user makes an error during form creation and testing, the error is flagged as follows: the flashing cursor is automatically positioned over the field in error and an error message is displayed on the bottom line of the screen. The user may then correct the field in error and strike any function key to send the corrected form back to the computer. Some typical error messages are:

- **FIELD NOT TERMINATED:** the operator entry field does not have a terminating delimiter or is greater than 255 characters in length
- **ILLEGAL FIELD NAME:** the field name was not an alphanumeric string of 8 or fewer characters, starting with an alphabetic character
- **ILLEGAL ATTRIBUTE:** the checking attribute is either missing or not a legal integer value
- **NOT ENOUGH FIELDS DEFINED:** the user did not provide enough field names
- **TOO MANY FIELDS DEFINED:** the user provided too many field names or attempted to define more than 255 fields

Example: Creating a Screen Format

```
NEW ACCOUNT FORMAT

PLEASE FILL IN THE FOLLOWING INFORMATION. IF YOUR DATA DOES NOT
COMPLETELY FILL IN THE SPACE PROVIDED YOU CAN USE THE TAB KEY TO GET
TO THE NEXT ITEM.

LAST NAME [ ]
FIRST NAME, M.I. [ ]
STREET ADDRESS [ ]
CITY [ ]
STATE [ ]
ZIP CODE [ ]

ACCOUNT NUMBER [ ]

INITIAL DEPOSIT AMOUNT (SNNNNN.NN) $[ ]

TO ENTER THE DATA PLEASE TYPE FUNCTION KEY F1. IF DATA IS IN ERROR THE
ITEM WILL BE SET BLINKING AND AN ERROR MESSAGE WILL APPEAR AT THE
BOTTOM OF THE SCREEN. PLEASE CORRECT THE DATA AND RE-ENTER. A
CORRECT, COMPLETED FORM WILL BE INDICATED BY A MESSAGE. TYPE FUNCTION
KEY 16 TO RETURN TO THE MENU.

|

Note: All operator entry fields are defined with [...]. All other
data is set protected on the screen.
```

Example: Assignment of Field Names & Checking Attributes

NAME	1	<--- Any Characters Valid
FIRST	1	<--- Any Characters Valid
STREET	1	<--- Any Characters Valid
CITY	1	<--- Any Characters Valid
STATE	2	<--- Alphabetic Only
ZIP	3	<--- Numeric Only
ACCT	3	<--- Numeric Only
INITIAL	8	<--- Financial Numeric Only
MSG	0	<--- No Checking Performed

Example: Coding the Entry Procedures

```
PROC PAGE^OPEN:
BEGIN

! Initialize the I/O Buffer with Screen Format
WRTCNT := EXPAND^SCREEN(@FORMOPEN,SBUFFER,1):
CALL WRITE(CRT,BUFFER,WRTCNT,CNT):
IF < THEN BEGIN ...
.
.

! Turn Blinking Field Off
WRTCNT := BLINK^SCREEN(@FORMOPEN,SCREEN,SBUFFER,FORMOPEN^MSG,0):
CALL WRITE(CRT,BUFFER,WRTCNT,CNT):
IF < THEN BEGIN ...

! Prepare a buffer for Reading a Screen
WRTCNT := READ^SCREEN(@FORMOPEN,SBUFFER):
CALL WRITEREAD(CRT,BUFFER,WRTCNT,FORMOPEN^IOBUF,CNT):
IF < THEN BEGIN ...
.
.

! Perform Validity Checking on Operator Entry Fields
DO ... UNTIL CHECK^SCREEN(@FORMOPEN,SCREEN,SBUFFER,OPEN^CHK):

! Turn on Blink, Position Cursor
WRTCNT := BLINK^SCREEN(@FORMOPEN,SCREEN,SBUFFER,FORMOPEN^MSG,1):
WRTCNT := POSITION^SCREEN(@FORMOPEN,
SCREEN,
SBUFFER[WRTCNT],
FORMOPEN^ACCT) + WRTCNT:
CALL WRITE(CRT,BUFFER,WRTCNT,CNT):
IF ERR THEN BEGIN ...
```

DESIGNER'S OVERVIEW OF TRANSACTION PROCESSING

Lloyd Smith
Tandem Computers
19333 Vallco Parkway
Cupertino, California 95014

Copyright © 1979 Tandem Computers Inc.

Abstract

This paper presents information for the individual responsible for designing a transaction oriented system. It covers the major design considerations that should be taken into account. The paper is divided into the following topics:

- I. Introduction
- II. A Transaction Processing System Model
- III. System Control
- IV. Implementation Considerations
- V. TANDEM's Transaction Processing System

I. Introduction

One of the important points in any design, and the most important in a transaction environment, is that the system as a whole should be viewed as a "SERVICE". In any service organization the first goal is to find a service that people need. The second goal is to offer a service that people can depend on, and the third goal is to respond to the changing demands of those who use the service. If all three goals are not established from the beginning the success of

NOTE: TANDEM, GUARDIAN, EXPAND, NonStop, and PATHWAY are trademarks of Tandem Computers Incorporated.

the service may be negligible. The one goal most neglected in the past has been responding to change. The combination of user needs, building a reliable system, and the ability to react to change quickly are explored in this paper. The intent of this paper is to offer a better understanding of transaction processing and suggest general guidelines for successful implementations in the future, by examining these three points.

The first design consideration must be to fulfill a user defined need. This need or service is referred to as a "USER FUNCTION". Once a function is offered, a user gains confidence in the service based on reliability and the system's responsiveness to change as the needs of the business change. This ability to change and evolve is referred to as "REACTION TIME". The following design considerations have a direct impact on reaction time.

INSTALLABILITY:

A system can succeed only if it can be installed in a reasonable amount of time. Ease of installation also applies to any major enhancements or user functions.

FLEXIBILITY:

Determines how well the system reacts to change. If a system is designed properly with change in mind, changes can be applied quickly. Reaction time is reduced significantly.

EXPANDABILITY:

Allows the system to accommodate new users and new functions. After a system's initial success, two events typically occur:

1. More users want to use the system. A well-designed system must incorporate the ability to handle an increase in the number of users.
2. New functions are requested. The ability to handle new user functions without affecting the current user functions must also be considered.

MAINTAINABILITY:

The ability to correct problems and "tune" a running application. Too often, this consideration is overlooked in the original design. The problems that occur not only affect the existing functions but future functions as well. If significant resources are required to maintain existing functions, the ability to provide changes and enhancements is reduced. The total reaction time increases and the probability of overall success decreases.

RELIABILITY:

To ensure "service" to the user, the system must be reliable. The best implementations fail if the user cannot access his system. In addition to being constantly available, the system must ensure the integrity of its data base.

Note: Performance considerations are important; however, the above mentioned considerations should not be sacrificed for efficiency. The key to performance is through a good design.

II. A Transaction Processing System Model

Figure 1 shows the components of a typical transaction oriented system. A video display unit provides the human interface to the system. The remainder of the diagram describes the software components of a computer system including a Data Base stored on a direct access device external to the computer. Notice that the flow of control in this diagram is bi-directional. Notice also that operator request may not need the services of all five components. For example, if an operator enters non-numeric data into an entry field defined as numeric data only, the terminal I/O and the field validation routines are the only two components exercised. The request for a new function might involve the display of a new format. In this case four components are exercised: terminal I/O, field validation, data mapping, and transaction control.

This diagram is used as the foundation for the design of a transaction oriented system. The components describe the functional activities that are common to all transaction oriented applications. A brief definition of each component follows the diagram.

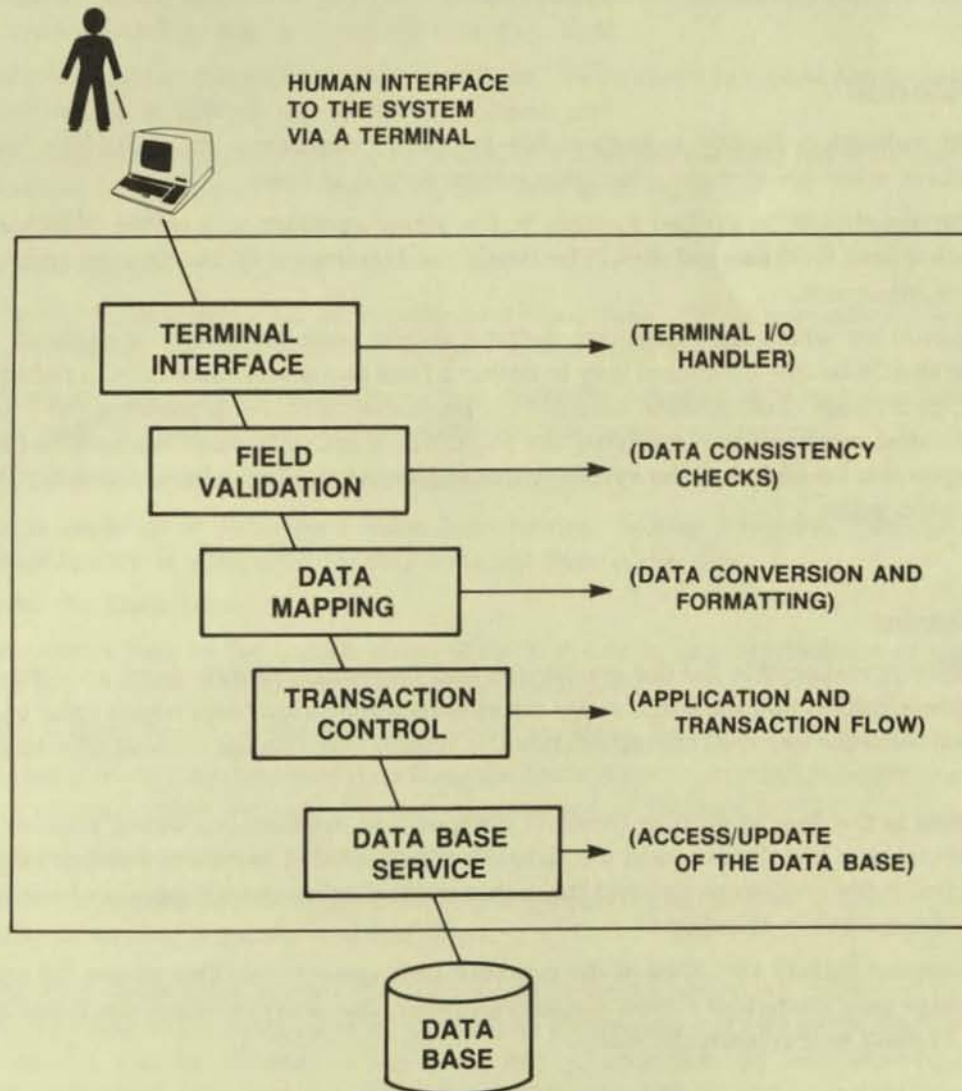


FIGURE 1. TRANSACTION PROCESSING SYSTEM MODEL

Major Program Components

1. Terminal Interface

The terminal interface is responsible for the following general functions:

- All physical terminal I/O
- Control of device-dependent characteristics

Because the physical I/O to various terminal types often involves different protocols, it is advantageous to isolate the code that actually communicates with terminals into one place. This approach enhances the capability to test and install new terminal types, isolate and fix problems, and easily take advantage of new features that may become available on existing terminal types.

The data transmitted by the physical I/O also requires different handling by different terminal types, e.g., the codes to delimit a field may differ.

2. Field Validation

The field validation facility is responsible for data consistency on a field-by-field level. Normally these edits are defined when the screen format is built.

Field validations should be applied as close to the actual operator as possible. Notification of a problem with a data field entered should be timely, and its impact on the running system should be kept to a minimum.

The mechanism by which field edits are defined should be independent of physical terminal type. There should be one consistent way to define a field as numeric data only, a field that must be entered, or a range of acceptable values for a particular field. By separating the type of edit from the physical mechanism of applying the edits, which are defined at the terminal interface, terminal types can be added to the system without altering the logical view of entry fields and their associated edits.

3. Data Mapping

This facility is responsible for the conversion and formatting of data from an external to an internal representation and back again. By developing application tasks which refer to the data in its internal form the external characteristics of a system may change without affecting existing applications.

Data mapping is the key to writing terminal independent applications which process requests with no concern as to how that request was actually constructed. Therefore, whether requests are stored in a batch file on disc or entered through a video display should have no bearing on the application design below this point.

The data mapping facility should be at the symbolic field name level. This allows the ordering of fields to change on a particular screen display; however, the order in which the fields appear in the logical request will remain the same.

4. *Transaction Control*

This facility is responsible for the overall application flow. It is analogous to the top level implementation of a structured program which:

- Initiates all logical terminal I/O;
- Interprets and validates the request received from the data mapping facility;
- If Data Base access is required to service the request, the appropriate data along with the proper control information is passed to the appropriate Data Base routine; and
- Interprets and validates the Data Base routine replies.

The main function of transaction control is application flow, along with the routing of requests to one or more Data Base routines. It is a relatively small portion of actual application code; however, it is the heart of any application. The major benefits of this approach are:

- Since the actual amount of code necessary to control any one function is relatively small, it is a simple task to add and change user functions;
- Since the approach is structured in a modular fashion, new functions can be integrated and tested easily as part of the whole application; and
- Application flow and control can be tested as a separate piece of the total application and therefore have little or no impact on the Data Base services.

5. *Data Base Service*

This facility is responsible for all activity on a Data Base. This is normally a very important application function because it alters the state of the Data Base.

A Data Base service routine should be written using the simplest approach possible. The most straightforward and simple approach contains the following components:

- Get a request from a transaction control facility:
This is the point of entry for a Data Base service. Getting a request from the transaction control facility is similar to reading a record from a disc file,
- Access the Data Base:
The request may be for a read, write, update, delete or any combination of the four. The specified requests are applied against the Data Base.
- Build a reply based on the results of the Data Base access:
The reply could contain actual data from the Data Base, control information describing any error condition that occurred, or any combination of the two.
- Reply to the transaction control facility:
This is the exit point of the Data Base service. Replying to the transaction control facility is similar to writing a record to a disc file.

Moreover, each Data Base service should process requests uniformly from one or more user functions. By doing so the Data Base service will be independent of any particular user function, and the service can be viewed as a general utility, accessible by any user function. This eliminates redundant code and simplifies the implementation of new user functions requiring Data Base services already established.

The Data Base service must be written in a context free environment. The Data Base service should not be responsible for the retention of data between requests. Once a request is received, the Data Base service should be able to process the complete request and then forget about it. This approach simplifies the code significantly and creates an environment that is easy to understand and maintain.

One critical part of any transaction oriented system is input validation. There are three major types of input validation:

TYPE OF VALIDATION	EXAMPLE
FIELD	NUMERIC MUST FILL RANGE 100 THRU 199
REQUEST	INTER-FIELD RELATIONSHIPS (IF PAYMENT-METHOD = "CASH" THE AMOUNT ENCLOSED MUST BE > 0) (IF CITY WAS ENTERED STATE AND ZIP CODE MUST ALSO BE ENTERED)
DATA BASE	DOES ACCOUNT # 12345 EXIST IN THE DATA BASE? DOES THIS SALES TRANSACTION IN THE AMOUNT \$245.00 FOR ACCOUNT # 12345 EXCEED THE ESTABLISHED CREDIT LIMIT?

Figure 2 shows each level of edit within the transaction processing system model.

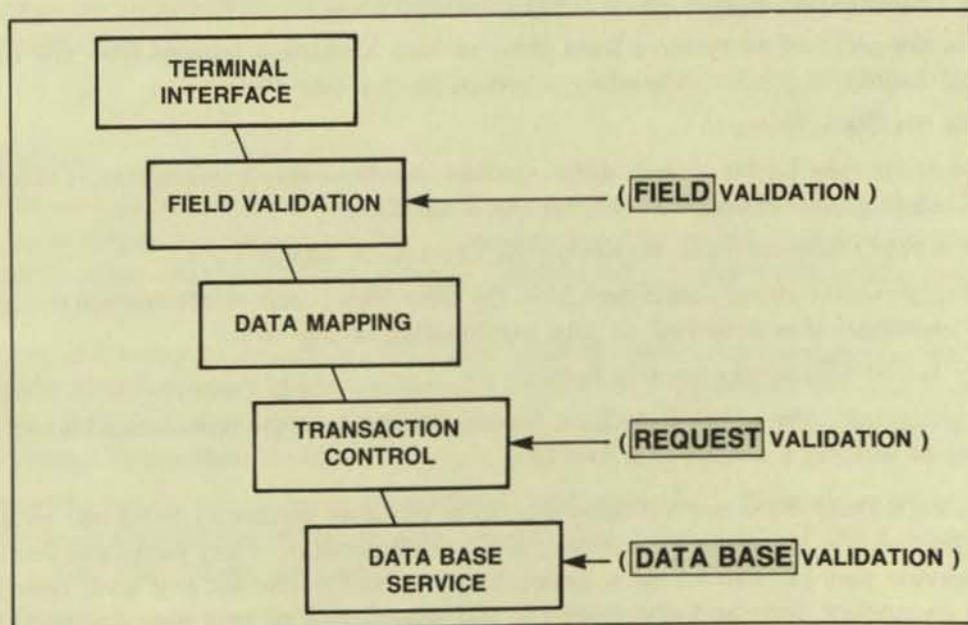


FIGURE 2. EDIT LEVELS

At this point the model is defined. The following illustrations show the use of the model in an application environment.

The internal components of the transaction processing system model can be grouped into the following categories:

Request oriented:

Components 1 through 4 have the combined responsibility for gathering, interpreting and responding to requests. From this point on the combination of components 1 through 4 will be referred to as the "REQUESTOR" portion of our model.

Service oriented:

Component 5, the Data Base services, are written as general utility functions accessible by any user function within a REQUESTOR. The Data Base services will be referred to as "SERVERS".

Figure 3 shows the REQUESTOR/SERVER relationships:

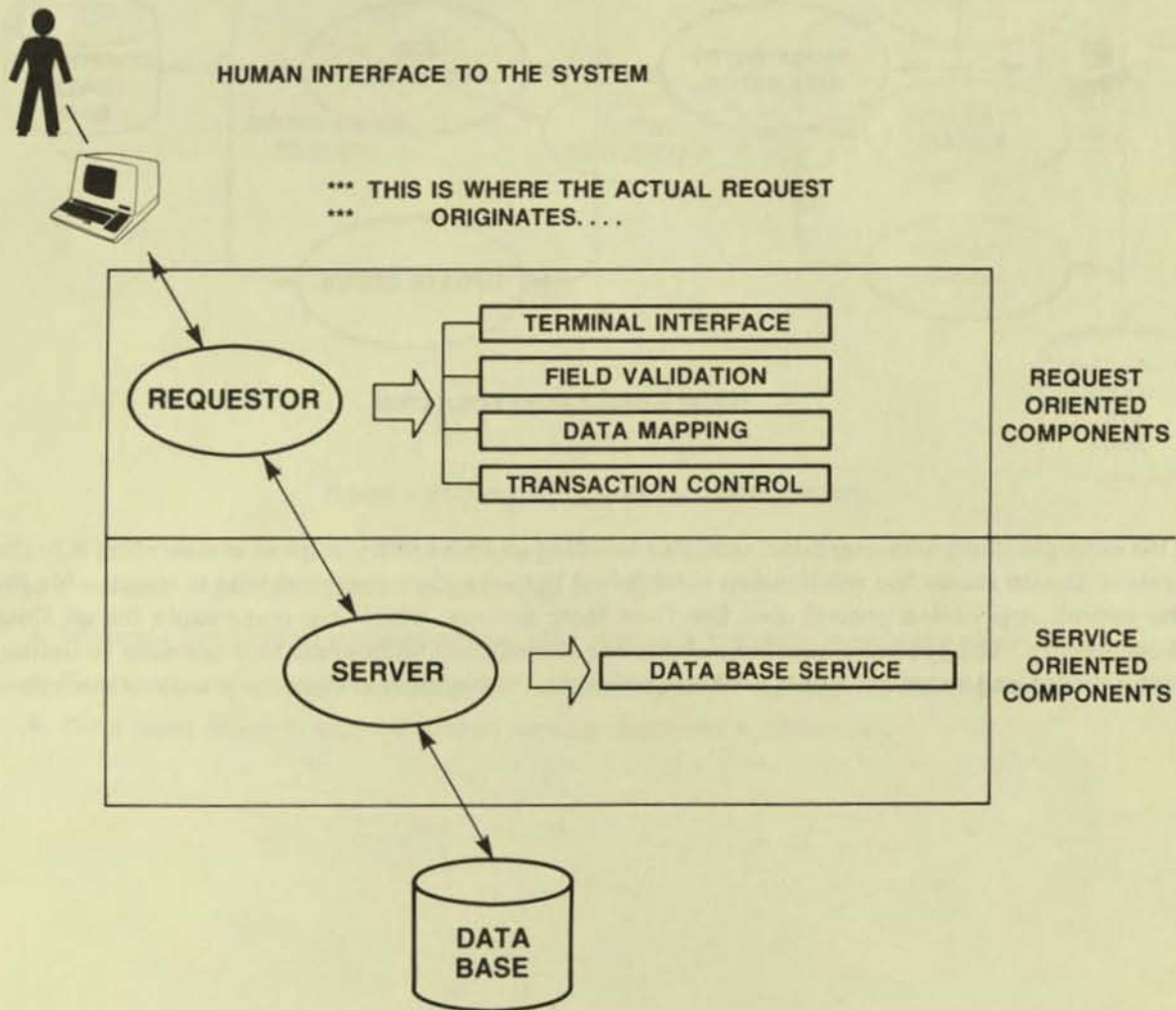


FIGURE 3. REQUESTOR/SERVER RELATIONSHIPS

Figure 4 illustrates an order entry application with three basic functions: credit checking a customer, adding a new order, and updating an existing order.

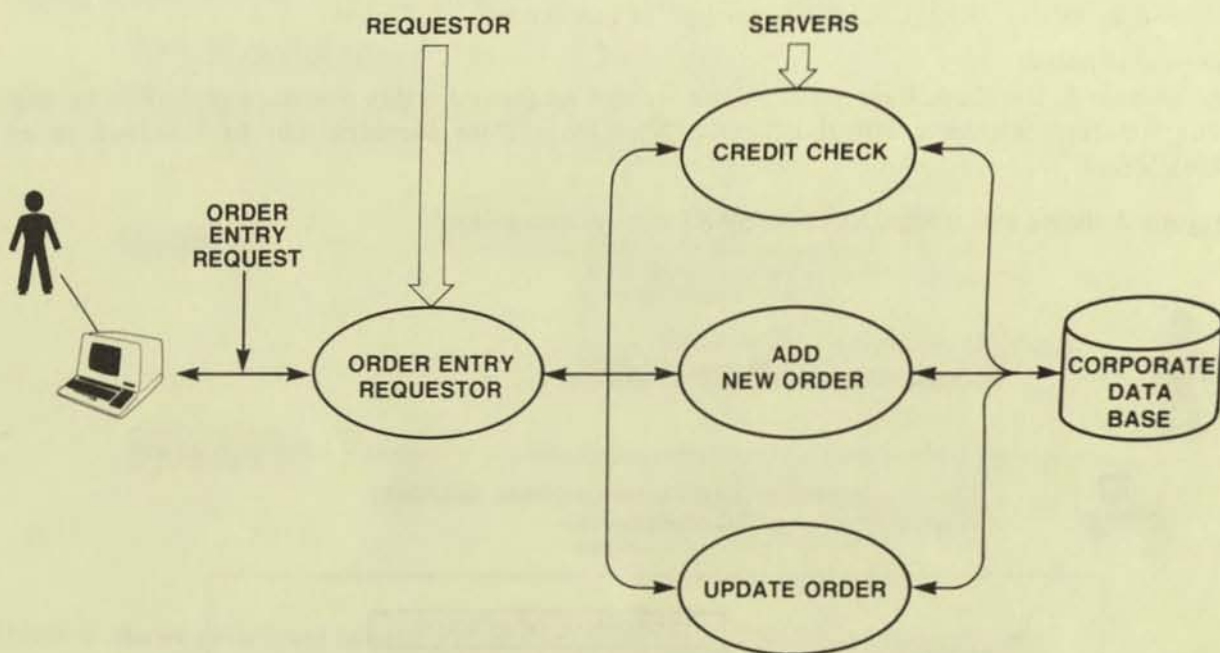


FIGURE 4. ORDER ENTRY APPLICATION

This example illustrates a terminal operator building an order entry request and sending it to the system. It also shows the relationship established between the requestor, who is responsible for the overall application control, and the Data Base servers, which are responsible for all Data Base activity. The system is partitioned into small modular components that are easy to define, write, debug and enhance. Because of its modularity, the system is extremely easy to maintain.

Figure 5 shows the relationships established between multiple applications running under the control of one transaction processing system.

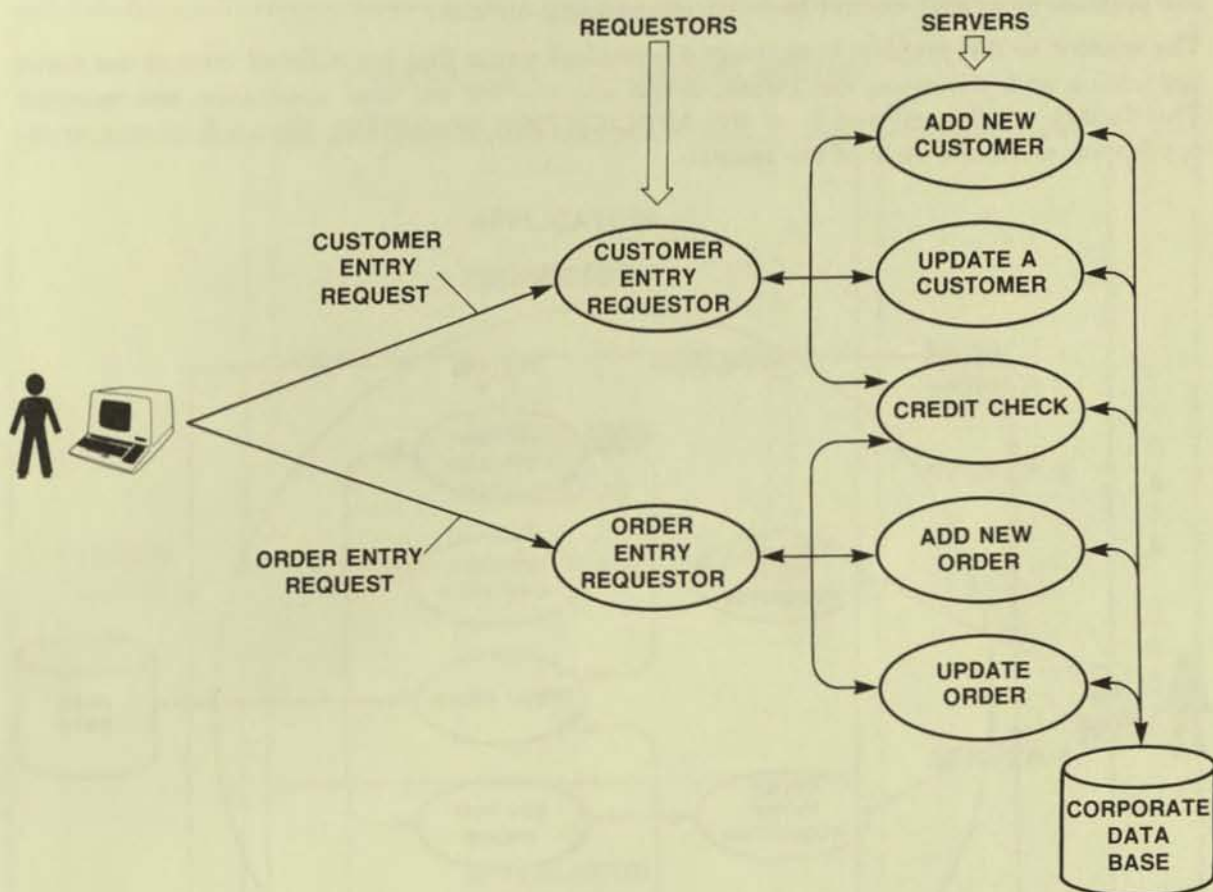


FIGURE 5. MULTIPLE APPLICATIONS IN A SINGLE SYSTEM

The above example illustrates the following points:

- Multiple application requestors can exist within a transaction processing system.
- Each terminal operator should have access to the number of application requestors needed.
- Data Base Servers may be shared among requestor applications.

III. System Control

Taking advantage of modular design techniques improves the overall quality of the system. However, as the components of a system are divided into smaller, more manageable segments, the problem of overall control becomes increasingly difficult.

The solution to this problem is to create a command center that has a global view of the entire application and, therefore, can create, delete and monitor the total application environment. This facility will be referred to as the **APPLICATION MONITOR**. Figure 6 illustrates the application monitor's view of the system:

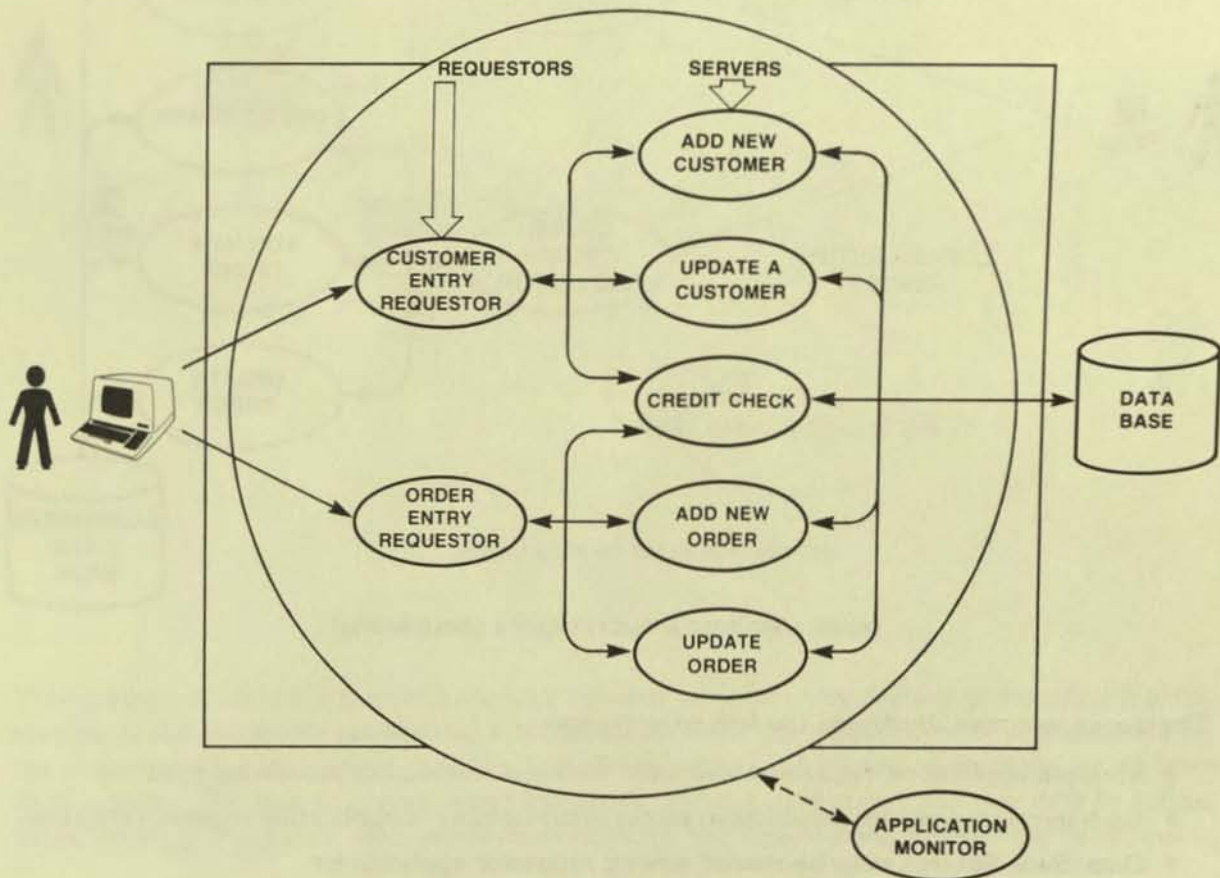


FIGURE 6. SYSTEM CONTROLLED BY APPLICATION MONITOR

The transaction processing system model established previously can be viewed in the following way:

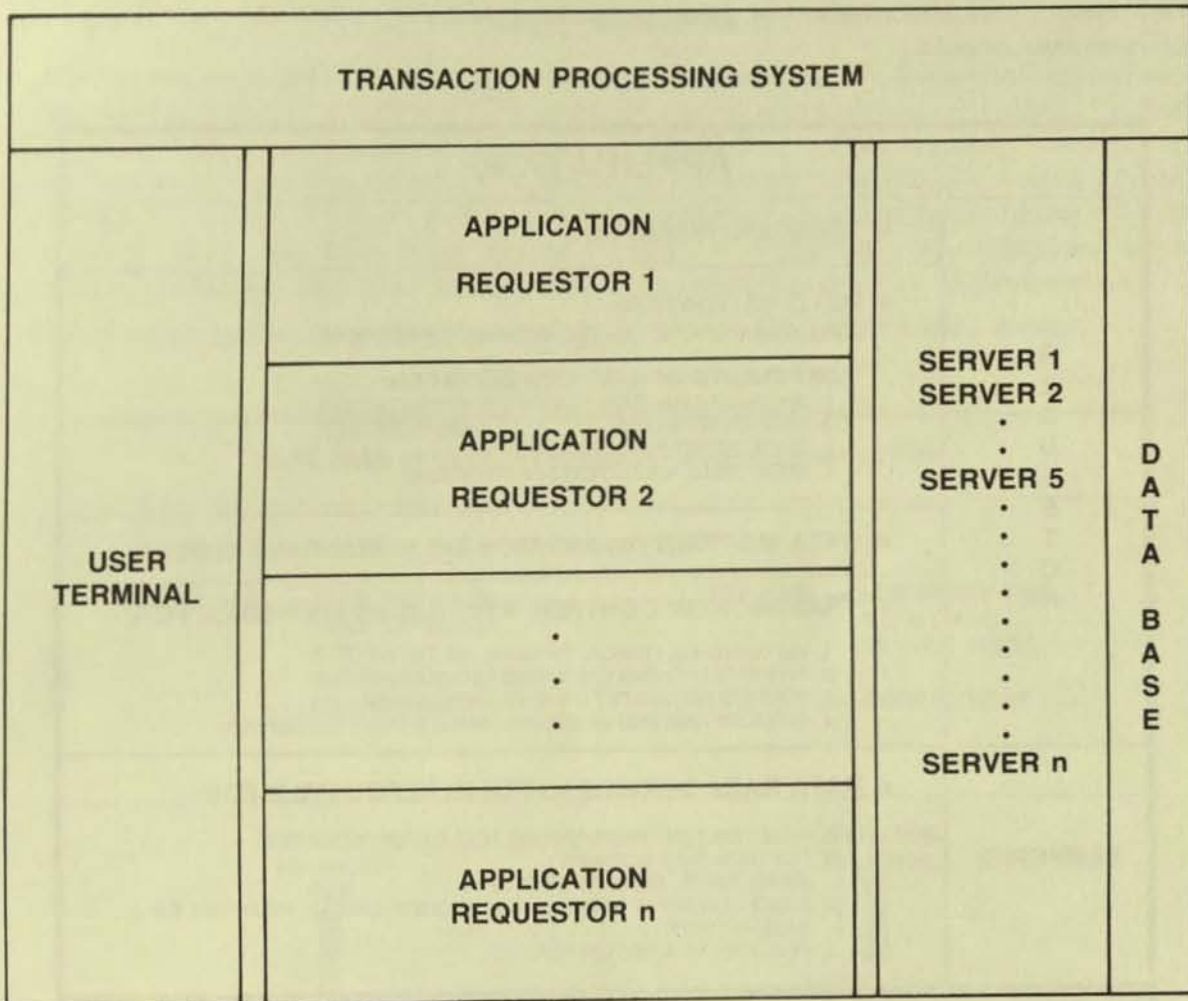


FIGURE 7. TRANSACTION PROCESSING SYSTEM OVERVIEW

The application monitor's view of a transaction processing system is analogous to the view a system operator might have at his console. The system operator controls the hardware environment and the application monitor controls the application environment.

The application monitor should maintain control over the following functions within a transaction processing system:

- Overall Transaction Processing System
- Each Application REQUESTOR
- Each Data Base SERVER
- Each Operator TERMINAL

IV. Implementation Considerations

Once the system is designed, its timely implementation, plus its ability to change as the needs of the business change, will be the deciding factors that determine its ultimate success or failure. Each system may be composed of many applications and each application will require the following components:

APPLICATION	
R E Q U E S T O R	<ul style="list-style-type: none"> • TERMINAL INTERFACE
	<ul style="list-style-type: none"> • FIELD VALIDATION (ONE COMPONENT OF THE SCREEN DEFINITION) <p>COMPONENTS OF A SCREEN DEFINITION:</p> <ol style="list-style-type: none"> 1. INFORMATIONAL DATA FOR PROMPTS (PROTECTED) 2. DATA ENTRY FIELDS (UNPROTECTED) 3. INITIAL ENTRY FIELD VALUES 4. ENTRY FIELD VALIDATION SPECIFICATIONS
	<ul style="list-style-type: none"> • DATA MAPPING (TO AND FROM THE SCREENS AND MEMORY)
	<ul style="list-style-type: none"> • TRANSACTION CONTROL WHICH IS RESPONSIBLE FOR: <ol style="list-style-type: none"> 1. INITIATING ALL LOGICAL TERMINAL I/O 2. INTERPRETING AND VALIDATING REQUESTS 3. ROUTING REQUESTS TO THE PROPER SERVER 4. INTERPRETING AND VALIDATING REPLIES FROM THE SERVER
SERVER(S)	<ul style="list-style-type: none"> • DATA BASE SERVICE WHICH IS RESPONSIBLE FOR: <ol style="list-style-type: none"> 1. ACCEPTING AND INTERPRETING REQUESTOR MESSAGES 2. ANY DATA BASE ACTIVITY: (READ, WRITE, REWRITE OR DELETE) 3. BUILDING A REPLY BASED ON THE SUCCESS OR FAILURE OF THE DATA BASE ACTIVITY 4. REPLYING TO A REQUESTOR

FIGURE 8. FUNCTIONS OF REQUESTORS VERSUS SERVERS WITHIN AN APPLICATION

Requestor Procedures

Because the requestor provides the logic that communicates with the end user, it must be the most flexible component of the system. A means of designing, changing and deleting screen formats is essential. Internal record formats produced through data mapping should be kept and maintained in a data definition library similar to those libraries associated with record definitions on a Data Base Management System. The actual transaction control might be written in a procedural language that is easy to use but flexible enough to handle total application flow. All the above facilities should be maintained in a library accessible at run time. This allows smooth integration of each function within a requestor. It also allows modular expansion of functions within an application with little or no impact on current running functions.

This approach to implementing an application requestor ensures the reaction time necessary to effectively handle user demand. Moreover, it allows the system to expand in small, well-controlled increments, thus increasing the integrity of the overall system.

Server Procedures

The back-end component of any application is the Data Base server. Back-end functions must be handled with care, for they maintain the most critical aspect of any application, the Data Base. One of the principal advantages of this design concept is that it greatly simplifies the implementation. Server procedures can be designed and tested in the familiar — read a record, update the Data Base and Write a reply — fashion. Input transactions can be read from a disc file or magnetic tape. Test Data Bases can be created for the purposes of testing, and server procedure testing can take place independent of the overall application implementation.

The following diagram illustrates the two step integration of any Data Base server:

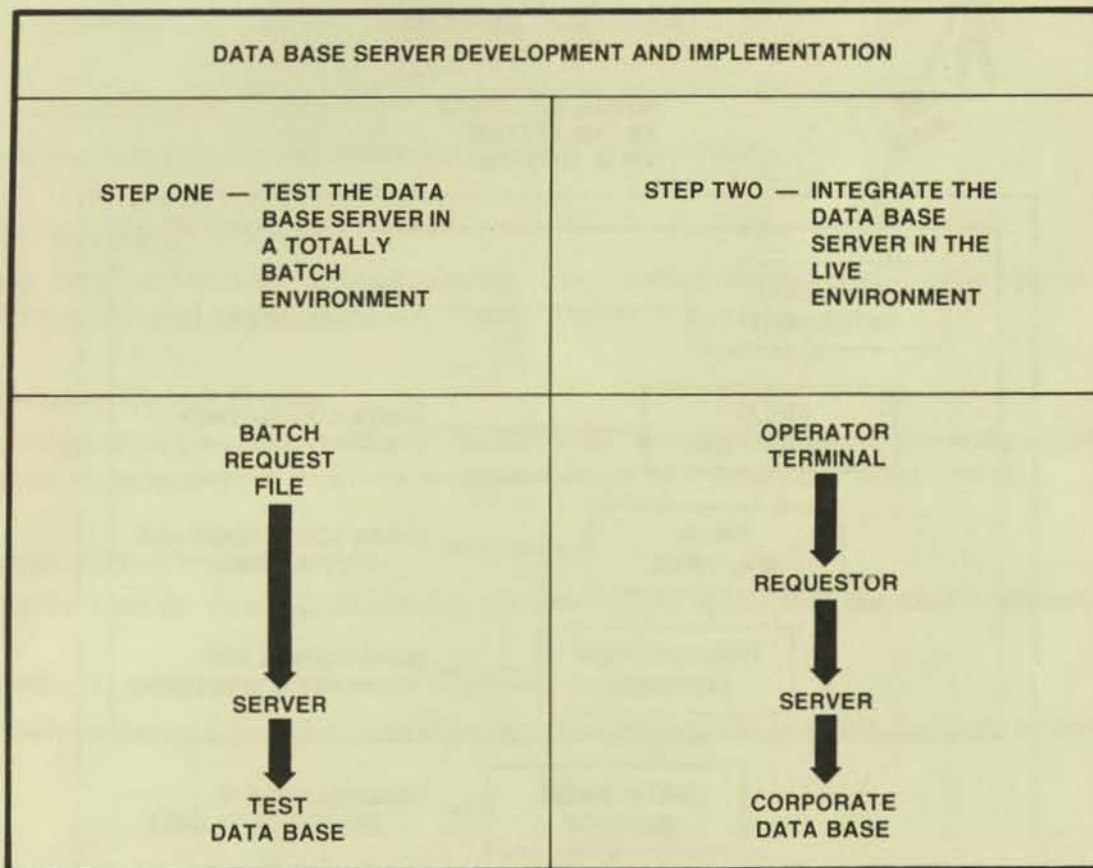


FIGURE 9. INSTALLING A DATA BASE SERVER

The integration of both REQUESTORS and SERVERS into the live system should be handled via the APPLICATION MONITOR. The application monitor should be able to logically start and stop any component within the system.

V. The TANDEM Transaction Processing System

TANDEM offers a total environment for transaction processing. The GUARDIAN OPERATING SYSTEM was specifically designed with NonStop transaction processing in mind. The FILE SYSTEM within Guardian allows separately running processes (in the same CPU, different CPUs within a single system or different systems with an EXPAND network) to communicate with each other at a simple READ/WRITE level. Guardian allows one logical computer system to incorporate up to 16 processors. The EXPAND network allows the interconnection of as many as 255 logical systems within a network and still maintains the simple READ/WRITE level communications between application processes. With this as a base, TANDEM has introduced a new product called PATHWAY. PATHWAY allows a user to take advantage of the unique TANDEM architecture, and it significantly reduces the time necessary to develop a transaction processing system. The functions enclosed within the inner box are addressed by the PATHWAY product.

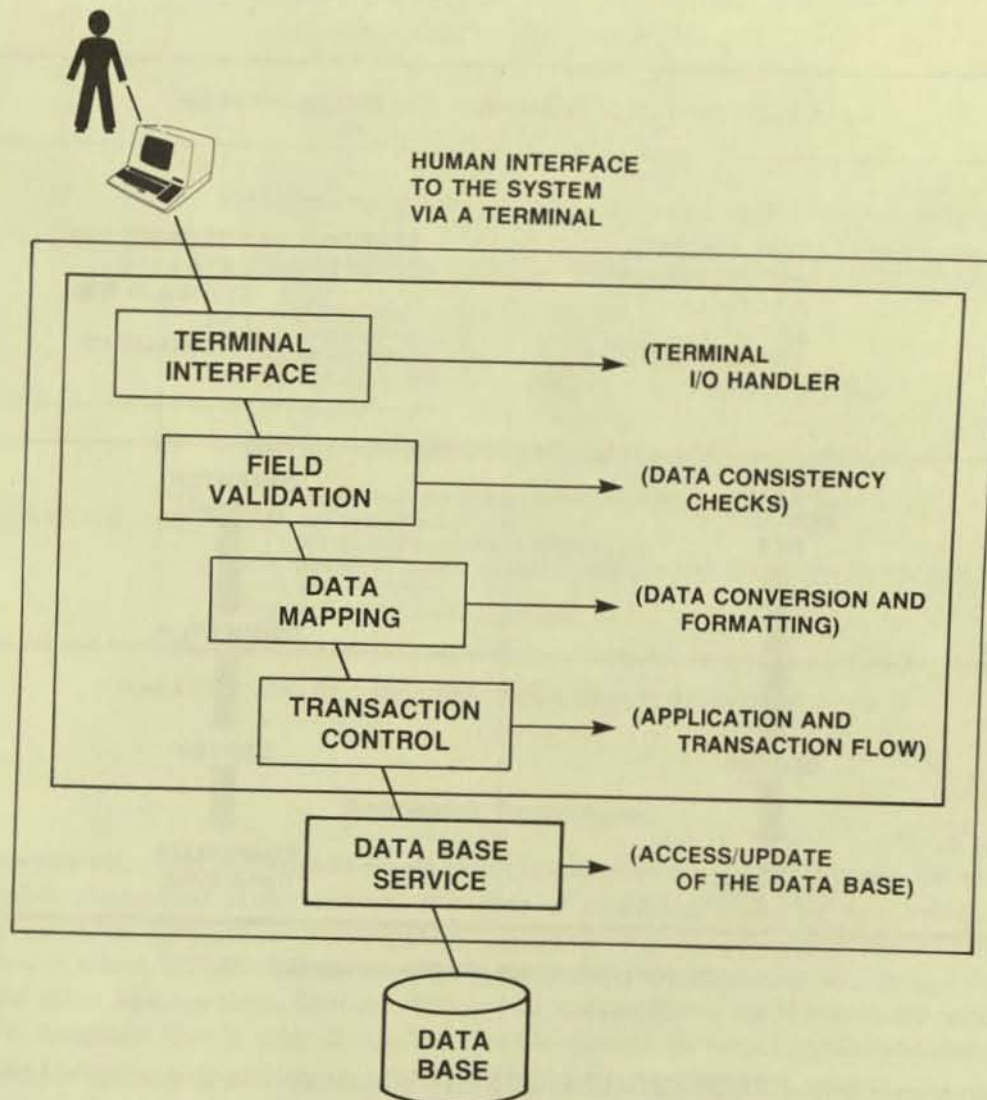


FIGURE 10. THE PATHWAY SYSTEM WITHIN THE TRANSACTION PROCESSING SYSTEM MODEL

PATHWAY Product Overview

The goal of the PATHWAY product is to simplify the design and development of transaction oriented applications. The PATHWAY product addresses four of the major components necessary to implement a transaction oriented application.

- | | | |
|------------------------|----------------------------------|---|
| 1. Terminal Interface | (Multi-terminal I/O handler |) |
| 2. Field Validation | (Data consistency checks |) |
| 3. Data Mapping | (Data conversion & formatting |) |
| 4. Transaction Control | (Application & transaction flow |) |

The fifth component (Data Base Server) can be implemented using any of the TANDEM standard languages — COBOL, FORTRAN, TAL, or MUMPS.

The PATHWAY product has the following components:

- *Interactive Screen Builder*

Allows the user to build screens interactively at a terminal.

- *Screen COBOL compiler*

A COBOL-like terminal oriented language. The compiler creates and maintains a pseudo code library accessed by the Terminal Control Process at run time.

- *Terminal Control Process*

Interprets the pseudo code-library created by the Screen COBOL compiler and performs the four major application functions mentioned above in a NONSTOP environment.

- *Application Monitor*

Responsible for creating, monitoring and altering the application run time environment.

- *AMCOM – Application Monitor Command Language*

The mechanism by which an operator may communicate with an active Application Monitor.

If we assume a successful installation of a transaction oriented system, we now must deal with EXPANDABILITY. By using the unique TANDEM architecture, an application can be written and then expand smoothly as the demands placed on the system increase. Most successful systems first expand because of an increase in the number of users who need to use it. Figure 11 shows the addition of new users and interjects a new question: "Do I run more than one copy of

the total application or do the users share the application?" Figure 11 shows users sharing the application. It should be noted that TANDEM's terminal control process, a part of PATHWAY, handles all the multi-tasking between more than one terminal of the same type. In Figure 11, notice that the application remains unchanged even though the number of users increases.

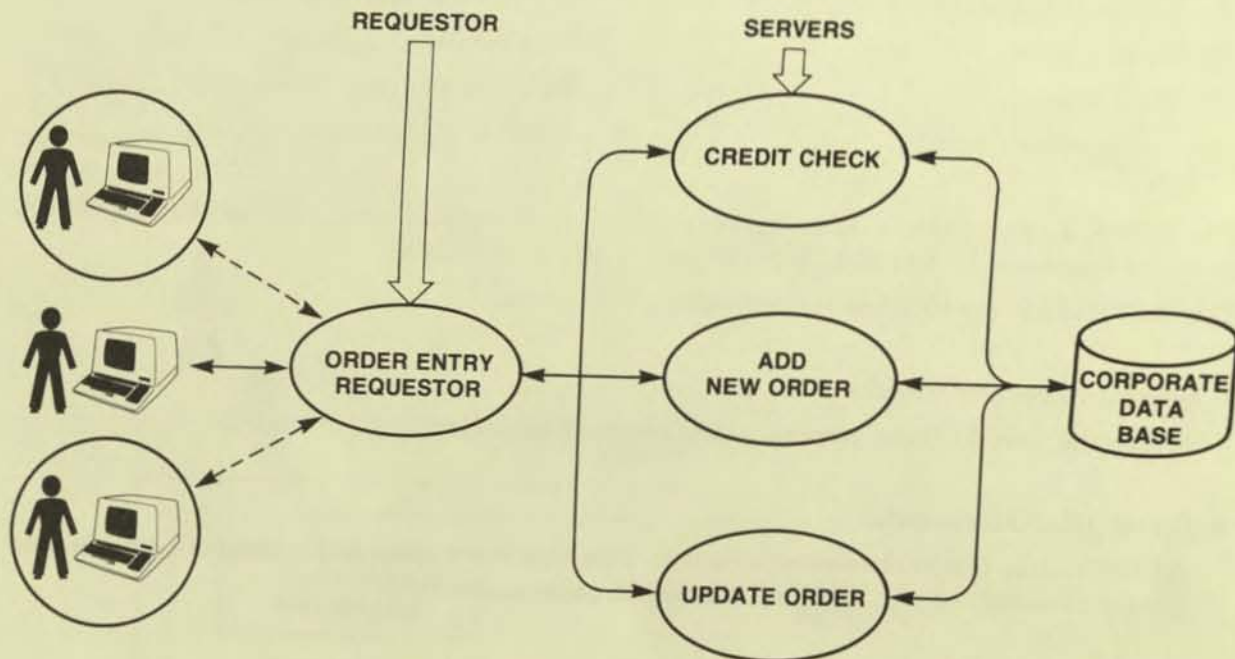


FIGURE 11. ADDING USERS TO PATHWAY SYSTEM

In a successful application, the addition of users can degrade system performance. At that point, most systems go through either a major change or a rewrite. The unique TANDEM approach offers an alternative.

To alleviate the problem of increased user load, the system must be able to distribute the application into more than one physical process. This is the key to expandability.

The description of the system to this point views the system as one self-contained application. However, all expansion limitations can be eliminated if one logical application can comprise multiple physical processes or running programs. This leaves the original design of the system unchanged, but increases system throughput by expanding the modularity of the application beyond the physical boundaries of a single program unit.

The following diagrams illustrate the various ways of functionally distributing the application to allow for expansion:

- Terminal operators can be connected to multiple requestors running the same application.
- Multiple copies of the same server can be created to increase throughput.
- Sharing a server between more than one application requestor.

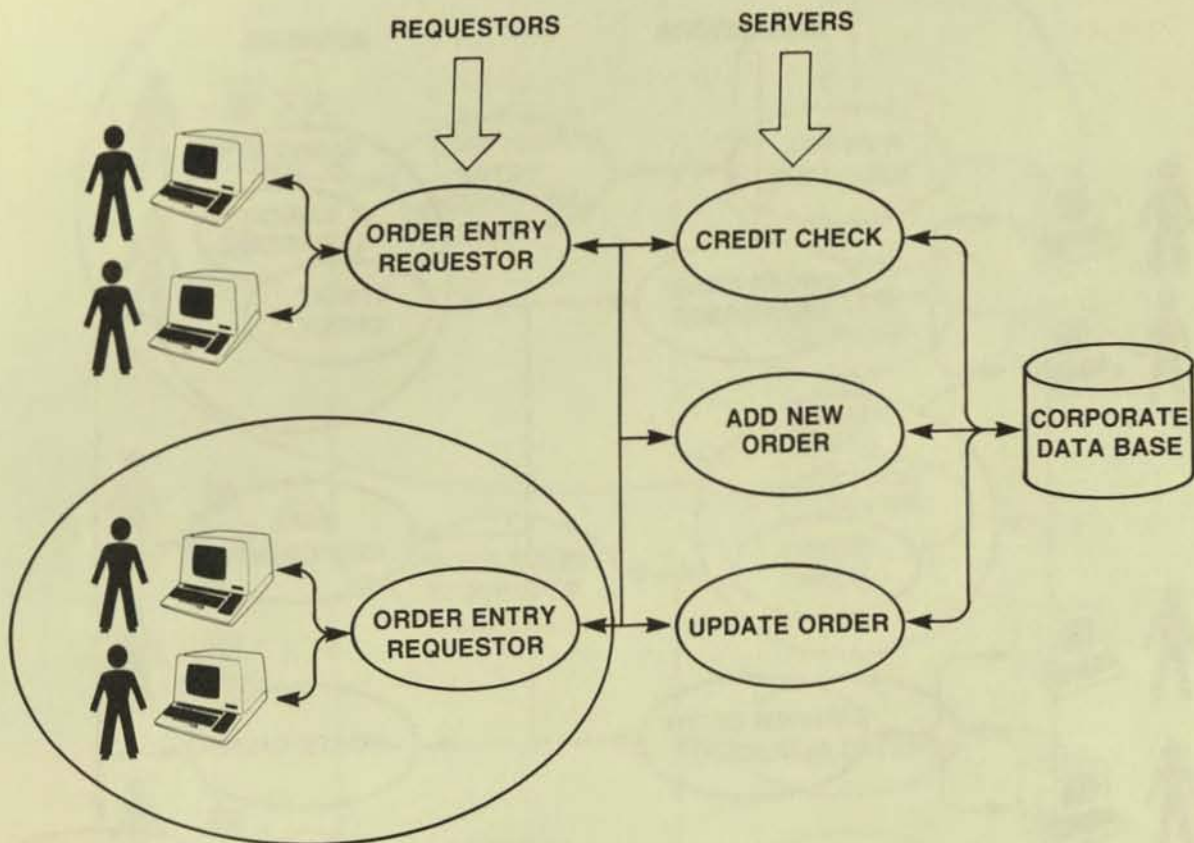


FIGURE 12. SYSTEM EXPANSION BY ADDING REQUESTOR

Notice that the users are distributed between two requestors and that the requestors share the server functions. Therefore, the total number of servers remains constant even though the system includes multiple requestors.

Another expansion problem is caused by increased demand on any one server. This problem can be overcome by duplicating a particular server to create what is known as a "SERVER CLASS". For example, if the majority of user demand on the system is to run credit checks, using a single server for credit checking could create a bottleneck and impact the total application. This bottleneck can be eliminated by creating another copy of the credit check server and distributing the requests between the two copies. Figure 13 illustrates a server class:

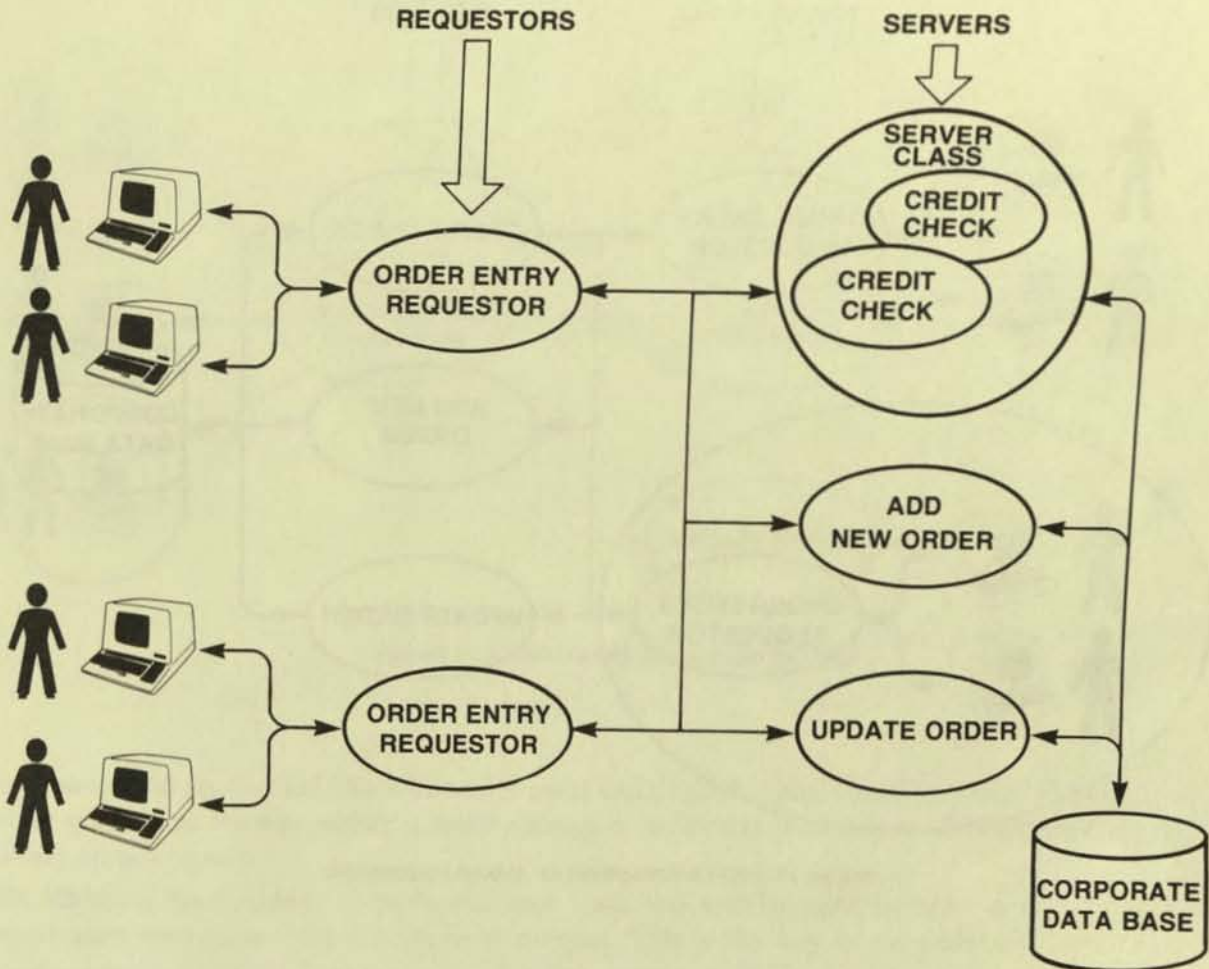


FIGURE 13. SERVER CLASSES

Figure 14 illustrates the addition of a new application. Notice that even though there are two unique applications, the requestors can still share servers or server classes.

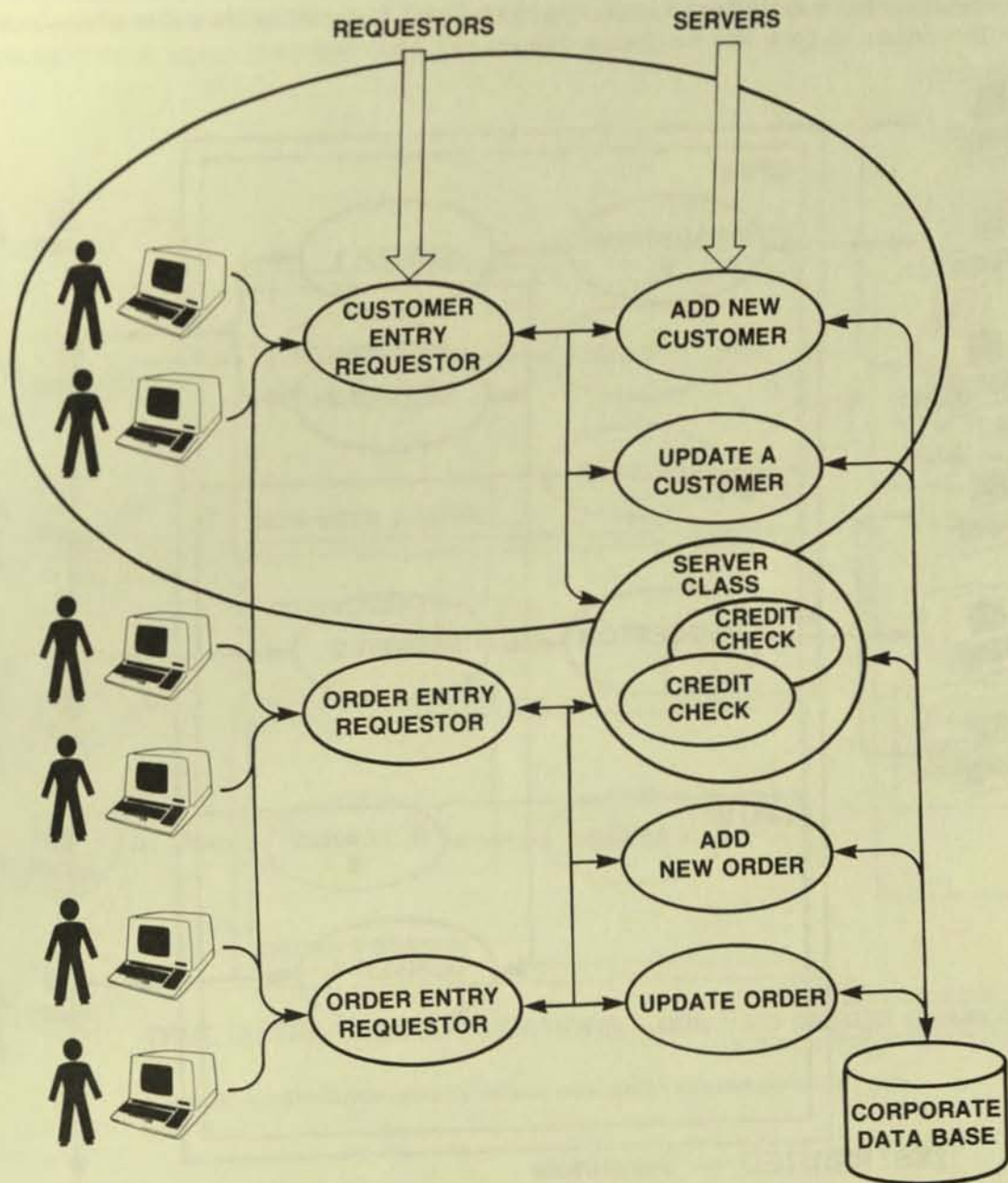


FIGURE 14. ADDING A NEW APPLICATION REQUESTOR

Figure 14 shows that both the order entry and customer entry applications need to be able to check credit. The credit check server class is therefore shared by both requirements of a particular application.

One of the major reasons for distributing an application into multiple processes was for expandability. Figure 15 illustrates the distribution of application functions within a local TANDEM system. Terminals, Requestors and Servers may be distributed among multiple CPU's, maximizing parallel operations and increasing throughput. It should be clear that when demand forces the system to grow that no design changes will be necessary.

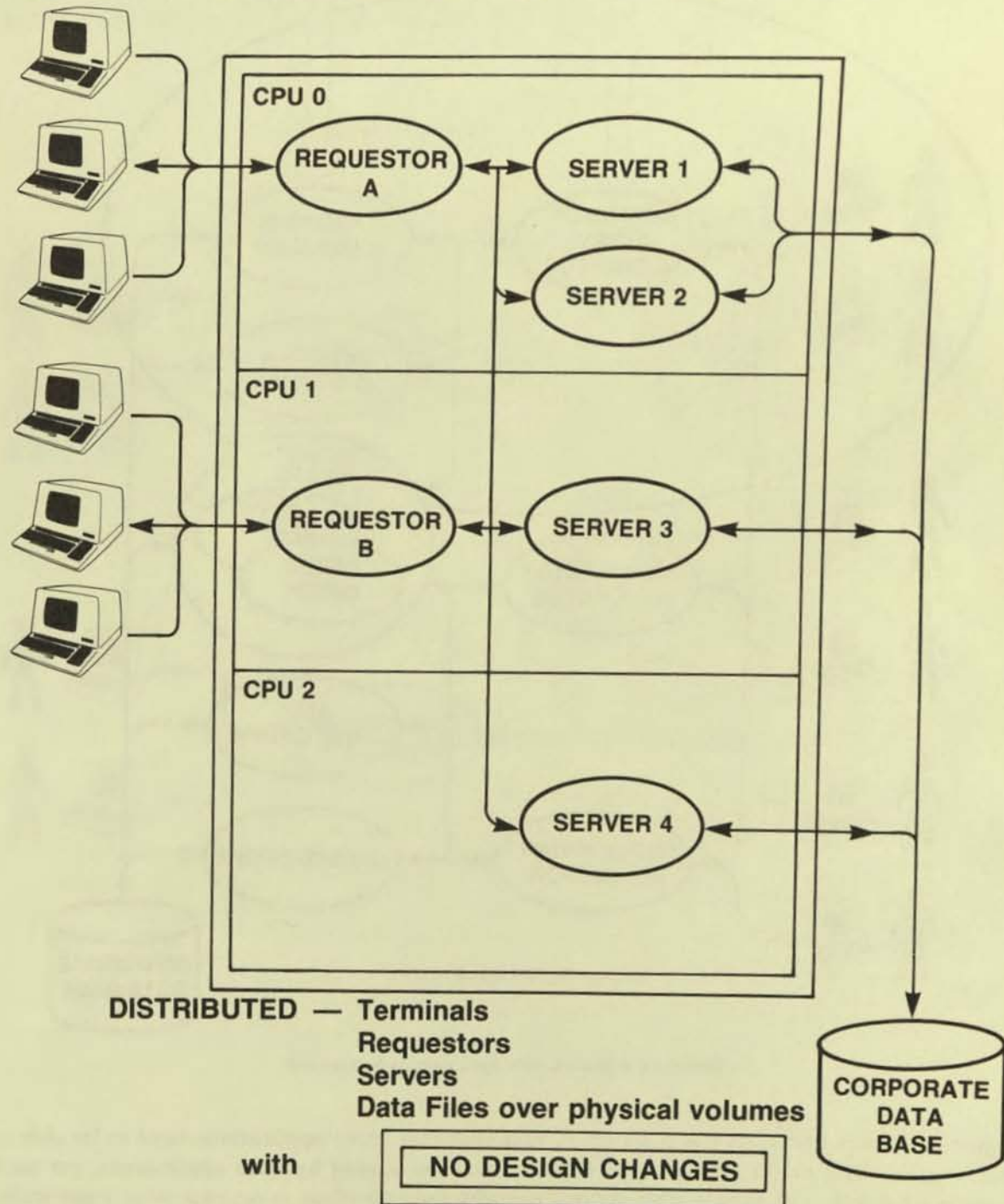


FIGURE 15. DISTRIBUTED LOAD BALANCING ON A LOCAL TANDEM SYSTEM

The next step in distributing application functions is over a network. The mechanism for communicating between processes within the TANDEM environment is consistent (Read / Write). Therefore, distributing application functions over a TANDEM network using EXPAND does not impact the original design. Figure 16 illustrates the distribution of application functions over a simple EXPAND network.

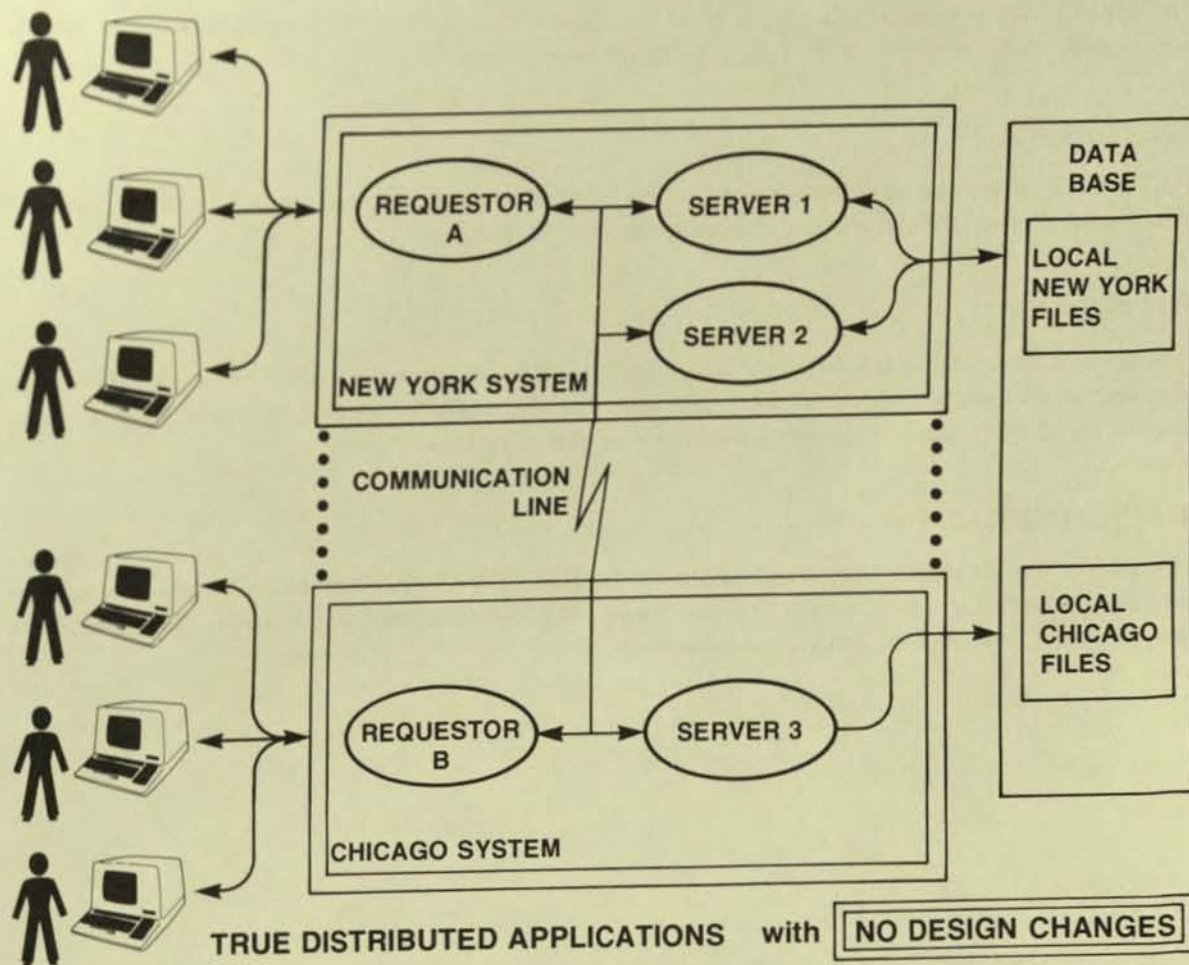


FIGURE 16. DISTRIBUTED LOAD BALANCING ON A SIMPLE EXPAND NETWORK

In summary TANDEM offers a total solution to transaction processing:

RELIABILITY:

NonStop transaction processing assures continuous system availability and data integrity

INSTALLABILITY:

PATHWAY offers a quick and easy way of developing transaction oriented applications, which significantly reduces the cost of applications development.

FLEXIBILITY:

PATHWAY offers the ability to make on-line additions, modifications, or deletions of transaction types, screen characteristics, applications, and terminals.

EXPANDABILITY:

The combination of the GUARDIAN Operating system and the EXPAND network offers true distributed processing, which allows applications to run in any processor in any system without regard for the physical location of terminals or the data base.

MAINTAINABILITY:

Because of the structured approach taken by the PATHWAY product, modules may be written and implemented in small, single threaded, and easy to understand components. Therefore, the maintenance task will be kept to a minimum.

Data Base

DATA MANAGEMENT SOFTWARE

The Enscribe Data Base Record Manager provides high level access to, and manipulation of, records in a data base. Enscribe operates as an integral part of the Guardian Operating System in a distributed fashion across multiple processors. As such, Enscribe ensures the integrity of the application's data in the event that a processor module, I/O channel, or disc drive fails.

All data transfers between an application process and an Enscribe disc file are done in terms of logical records. The placement of and access to records in a disc file is determined by the file structure (a file's structure is specified at file creation time).

For all file structures, the maximum length of a logical record (i.e., the maximum number of bytes that can be inserted in a single operation) is specified for each file at file creation time. The actual number of bytes comprising a logical record can be variable (up to the specified record length); the minimum number of bytes that can be inserted depends on the file structure.

Each record has a length attribute. The length attribute is a count of the number of bytes inserted when the record was written. A record's length is returned when the record is read.

Key-Sequenced File Structure

Records are stored in ascending sequence according to the value of a field within each record called the "primary key field." The primary key field is designated when a key-sequenced file is created and may be any set of contiguous bytes within the data record. Physical and logical record lengths can be variable; a record occupies only the amount of space specified for it when inserted into the file.

Relative File Structure

Records are stored in a position relative to the beginning of the file according to a record number supplied by the application program. A record number is an ordinal value and corresponds directly to a physical record position in a file. Each physical record position in a relative file occupies a fixed amount of space (although logical record lengths may be variable).

Entry-Sequenced File Structure

Records are appended to the end of an entry-sequenced file in the order in which they are presented to the system. Once added to a file, a record's contents may be updated but the record's size may not be changed and the record may not be deleted (although an application program may use a field within the record to indicate that it has been logically deleted). Physical and logical record lengths can be variable; a record occupies only the amount of space specified for it when inserted into the file.

Each record in a file is uniquely identified among other records in that file by the value of its primary key. For key-sequenced files, the primary key is a byte field within a record; for relative files, the primary key is a "record number"; for entry-sequenced files, the primary key is a "record address." Records in a file are physically ordered by ascending value of the primary key.

One or more byte fields within a record may be designated "alternate keys." Any Enscribe file structure may have up to 255 alternate key fields. Values in alternate key fields need not be unique. Several associated records of the same type may be located by their entries in an alternate key field. Each key in an Enscribe file provides a separate access path through records in that file. Records in an access path are logically ordered by ascending access path key values.

A subset of records in a designated access path can be described by a "positioning mode" and a key value. The positioning modes are: "approximate," "generic," or "exact." Approximate means that the subset is comprised of all records whose access path key value is equal to or greater than the supplied key value; generic means that the subset is comprised of all records whose access path key value matches a supplied partial value; exact means that the subset is comprised of only those records whose access path key value matches the supplied key value exactly.

Relational access among files in a data base is accomplished by obtaining a value from a field in a record in one file, then using that value to locate a record in another file.

Sequential Access Buffering Option

For a program that sequentially reads a file, the access time to individual records can be greatly reduced by means of the Sequential Access Buffering Option (this option, if desired, is specified at

file open). Basically, this option allows the record deblocking buffer to be located in the application process's data area (rather than in a system I/O process). This buffer is then used by Enscribe to deblock the file's records. The advantage to this buffering is that it eliminates the request to the system I/O process to retrieve each record in a block (instead, a request retrieves an entire block of records). This option is allowed only if the file is opened by the requesting process with protected or exclusive access.

Automatic Maintenance of All Keys

When a new record is added to a file or a value in an alternate key field is changed, Enscribe automatically updates the indices to the record (the value of a record's primary key cannot be changed). This operation is entirely transparent to the application program.

If more key fields are later added to a file, but existing fields in that file are not relocated, existing programs that access the file need not be rewritten or recompiled.

Data and Index Compression

For key-sequenced files, an optional data compression technique permits storing more data in a given disc area, thereby reducing the number of head repositionings.

Similarly, an optional index compression technique is provided for key indices to data records.

Both data and index compression may be specified for a file when the file is created.

Multiple-Volume (Partitioned) Files

At file creation time, a file can be designated to reside entirely on a single volume or may be partitioned to reside on separate volumes. Up to sixteen partitions are permitted; each partition can have up to sixteen extents.

In addition to providing a maximum file size of approximately four billion bytes, the use of multi-volume files provides for simultaneous access to a file's records:

- . If the file resides on seeking volumes connected to the same control device, seeking (disc head repositioning) can be occurring on all volumes simultaneously.
- . If each file resides on a volume that is connected to a different control device, several data transfers (as well as seeks) with the file can occur concurrently.
- . If each volume's control device is connected to a different processor module, simultaneous processing of the file's data can occur, as well as simultaneous seeking and data transfers.

Cache

The "Cache" is an area of main memory, whose size is specified at system generation time, that is reserved for buffering blocks read from disc.

When a request is made to read a record from disc, Enscribe first checks the Cache for the block that contains the record. If the block containing the record is already in the cache, the record is transferred from the Cache to the application process. If the Cache does not contain the block, the block is read from the disc into the Cache, then the requested record is transferred to the application process.

If no space is available in the Cache when a block must be read in, a weighted, least-recently-used algorithm (LRU) determines which block to overlay. The purpose of the LRU is to, whenever possible, keep the most recently referenced blocks in main memory. (For key-sequenced files, this weighting favors index blocks.)

When a request is made to write a record to disc, the block in the Cache that contains the record is modified then immediately written to disc (if the block to be modified is not in the Cache, it is first read from the disc). However, the modified block remains in the Cache until it is needed for overlay.

Enscribe is a powerful English-like relational query/report language easily used by non-programming personnel where key words may be easily used in a different language (German, French, Spanish). It automatically develops the most efficient structure to search data files your data base and produce reports at a fraction of the programming cost and time of a conventional language.

Support for data base record formats is provided by Tandem's Data Definition Language. The purpose of the Data Definition Language is twofold: First, to provide a uniform method of describing record formats, regardless of the programming language which will be used to access the record; second, to provide a global (i.e., system-wide) definition of record formats so that all programs will have a consistent definition of a given record format.

Record formats are described by means of a "schema." The schema is written using the Tandem-supplied EDIT program, then compiled by the Tandem-supplied SCHEMA program. The output of the SCHEMA compiler is a file which contains record definitions in a form compatible with a designated programming language (i.e., T/TAL or COBOL). Additionally, the SCHEMA compiler will provide a file which contains FUP commands for creating the files described by the record formats.

ENFORM

The Query-Report Writer (ENFORM) provides the user with a capability to perform queries upon a data base composed of one or more structured or unstructured files. The data retrieved can be qualified by an arbitrary boolean condition, and can span several files, which are conceptually "linked" by an equality condition on fields in their respective records. The retrieved information can be used to produce one or more reports or saved in a file specified by the user. Reporting options enable the user to sort and summarize retrieved data items; to evaluate standard or user-defined aggregate functions, over all records or by break in sort key value; to specify actions to be taken at breaks; and to control report format specifications, including titles, headings, spacing, and data formatting.

The general syntax for selecting items to be listed is:

```
LIST <target-list> ( <qualification-clause> )
```

The target-list is composed of the names of data base items required in a particular query or report. Each item in the list may have associated with it one or more attributes which specify such things as its heading, display-format and type.

The optional qualification-clause contains one or more instructions on how to select items for the target-list.

ENFORM is a powerful English-like relational query/report language easily used by non-programming personnel where key words may be easily redefined in a different language (German, French, Spanish). It automatically develops the most efficient strategy to extract data from your data base and produces reports at a fraction of the programming cost and time of a conventional language.

Tandem's Form Creation and Access Utility offers an easy way to make application-defined forms displayed on a Tandem T16/6511 or T16/6512 page mode terminal, and a simple method of accessing the fields within the forms.

The use of the utility is separated into three parts:

- Form Creation

Designing a form becomes a simple task with Tandem's SCREEN program and a page mode terminal. You arrange the fields on the screen and delimit the places where data entries are made. Once you are satisfied with the form, SCREEN asks for a field name, data attribute, and optionally, a default value for each field. Once the form is written in a file, you may test it with SCRNTST, which means you can test the efficiency and completeness of your form before the application program is ready.

- Forms Display

Calls to the ENTRY procedures display a form and read input data from it. The name of the form to be displayed is passed to the procedure, EXPAND^SCREEN, which fills the application program's I/O buffer with control characters and screen fields. Then a call to a file management procedure writes the buffer to the terminal.

After entries are made at the terminal, READ^SCREEN is called to fill the application program's I/O buffer with the terminal control sequence, then WRITEREAD, a file management procedure, transfers the screen data to the program. CHECK^SCREEN is called to check the data entries. Incorrect entries may be redisplayed, one at a time, with the blinking feature to make them stand out.

In all cases, the application program is responsible for doing the actual I/O through Tandem's file system, while the ENTRY procedures create buffers and process input buffers. This means you have full use of the file system features, plus ease of communication with the page mode terminal.

Field Access

Each field of a form has a name, length and data attribute associated with it. Using the naming conventions of SCREEN, a program has access to all three. For example, if your form "CARD" has a field "NAME," then your application program would reference it as "CARD^NAME," its length as "CARD NAME^LEN" and its data attribute as "CARD^NAME CHK."

TANDEM
ENSCRIBE™

FEATURES

- Full Site File and Record Management for Time-Share™ Transaction-Oriented Applications
- Multiple-File File Organization (File Organization, Retrieval, and Query Capabilities) with Exact Query or Approximate Multi-Key Access to All Records
- Approximate Record and Mechanism of Multi-Volume Files
- Merge Volume to Provide Full-Site Data Base Protection
- Data and Index Compression for Key-Organization Files to Optimize Disc Storage Space
- Main Memory Cache Buffering to Increase and Enhance Throughput
- Separate Simultaneous Record and File Locks to Facilitate and Expedite Concurrent Record Access and Provide System Security
- Full Compliance of Interactive High-Level Utilities or Relaxed Applications Programming Overhead and Cost

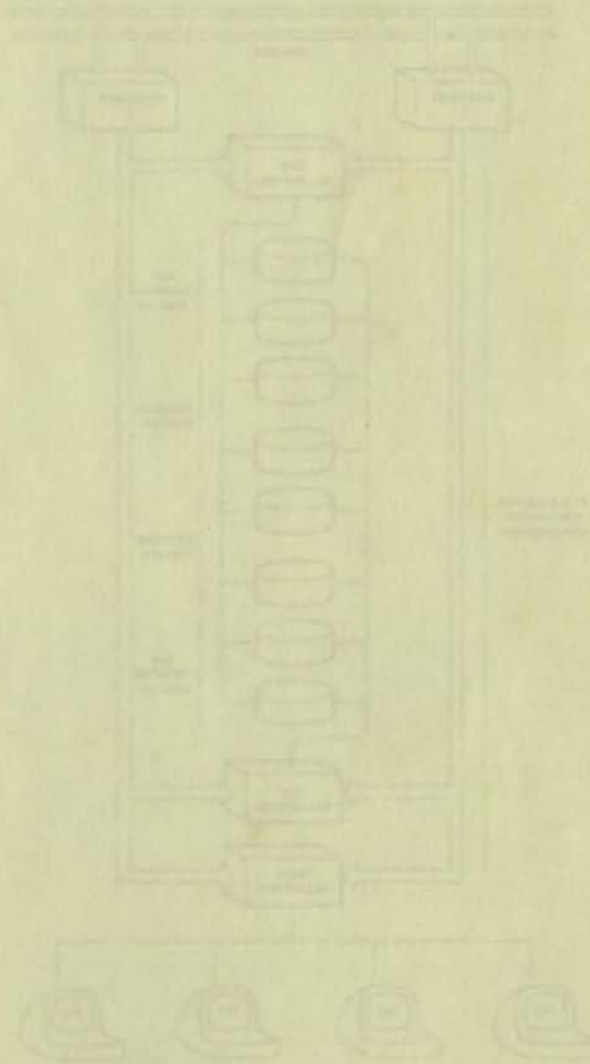


Figure 1. Multi-Volume File Organization

TANDEM ENSCRIBE™
Languages

TANDEM ENSCRIBE™

FEATURES

- Fail Safe File and Record Management for Non-Stop™ Transaction-Oriented Applications
- Multiple Disc File Organizations (Key-Sequenced, Relative and Entry-Sequenced) with Exact, Generic or Approximate Multi-Key Access to All Records
- Automatic Management and Maintenance of Multi-Volume Files
- Mirror Volumes to Provide Fail-Safe Data Base Protection
- Data and Index Compression for Key-Sequenced Files to Optimize Disc Storage Space
- Main Memory Cache Buffering to Increase and Enhance Throughput
- Separate or Simultaneous Record and/or File Lock to Facilitate and Expedite Concurrent Record Access and Provide System Security
- Full Complement of Interactive High-Level Utilities to Reduce Applications Programming Overhead and Costs

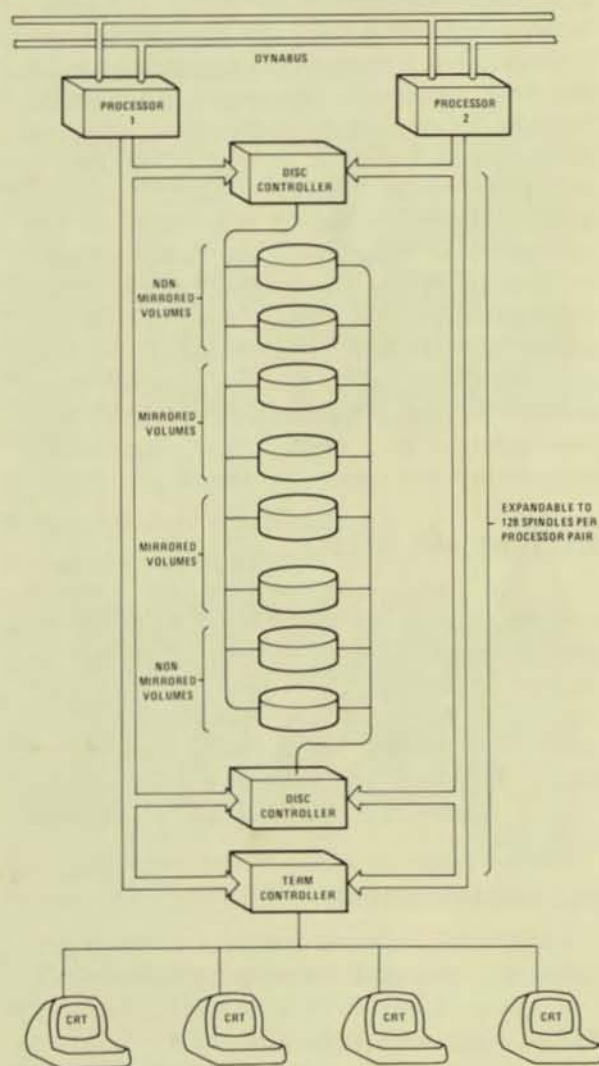


Figure 1. Multi-Volume File Configuration

INTRODUCTION TO ENSCRIBE

The unique and evolutionary ENSCRIBE software package extends Tandem's Non-Stop concept of hardware and operating system failure immunity to fail-safe file and record management. Since ENSCRIBE operates as an integral part of the GUARDIAN Operating System across multiple processors, data base integrity is ensured in the event of a Processor, I/O Controller or Disc Drive failure. ENSCRIBE brings many advanced features not found even in traditional mid- and large-scale computer systems into the mini-computer price range.

DATA BASE INTEGRITY

When a conventional processor or disc drive fails, it is most often necessary to reconstruct the data base from archival tapes or manual records to ensure the integrity of the data base. Conversely, in a Tandem T16 system, when a processor fails, all file management functions are automatically transferred to an alternate processor. Optionally, ENSCRIBE manages a data base using the mirror principle illustrated in Figure 1. In this mode, data is written onto two disc volumes simultaneously. When a disc fails, all file management functions are automatically transferred to the mirror disc volume. Upon restoration of the failed disc, ENSCRIBE copies the data back onto the failed disc. This copying operation is done concurrent with requests to read or update data and is entirely transparent to both the application programmer and user.

MULTI-VOLUME FILES

A file may reside entirely on a single volume or may be partitioned to reside on several volumes. With system expansion to sixteen processors, and assuming that each volume is mirrored, the maximum size for each multi-volume file is nearly four billion bytes. And each partition of a multi-volume file is under the control of a separate processor.

FILE ORGANIZATIONS

ENSCRIBE provides a variety of file organizations for the applications programmer:

- Key-Sequenced (Indexed)
- Relative
- Entry-Sequenced

The programmer need not be burdened with structuring the file. He simply chooses the file organization best suited to a particular application and ENSCRIBE automatically structures the file. This allows a programmer to create, access and maintain data files with efficiency and economy. Moreover, since records are logically positioned, random records may be accessed sequentially. ENSCRIBE also provides Multi-Key Access to data records. Location of records in a Key-Sequenced file may be by approximate, generic or exact key value. ENSCRIBE maintains an index of key values which provides rapid access to data records. When a new record is added to a file, or when a key value has been changed, ENSCRIBE automatically updates the index including all secondary keys. This operation is entirely transparent to the application program.

DATA AND INDEX COMPRESSION

For key-sequenced files, an optional data compression technique may be used to pack more data into a particular disc area. Similarly, an optional index compression is provided for key indices.

MAIN MEMORY CACHE BUFFER

ENSCRIBE provides a Cache Buffer which resides in main memory. The purpose of the cache is, whenever possible, to keep the most recently accessed blocks in main memory speeding up access. System transaction rates may, therefore, be increased by simply allocating more memory to the cache. The cache resides in a separate area from the operating system and application programs.

RECORD AND FILE LOCKING

ENSCRIBE provides both record and file locking for system security. Record locking provides a greater degree of concurrent access to a file. A file lock request must wait for the record to be unlocked before it is granted.

SUMMARY

ENSCRIBE is complemented by Tandem's *Data Definition Language* (DDL) for data base management. ENSCRIBE may be used alone for those applications which do not require centralized data base administration, or in conjunction with DDL to provide full data base access. Thus, the user is provided with an economical growth path from file management to NonStop data base access.

ENSCRIBE FACILITIES

GENERAL SERVICES

ENSCRIBE provides file and record management services for the Tandem GUARDIAN Operating System. These services are interfaced to the application program via T/TAL or COBOL.

ENSCRIBE facilities may be used separately for handling file and record management tasks, or alternatively, may be used as an extension to Tandem's *Data Definition Language* (DDL); a *schema* for centralized data base management. Once a file organization has been established, programs may store, retrieve and/or modify logical data records without concern for file structure. ENSCRIBE facilities provide all necessary access control, data buffering, blocking/deblocking and file structure maintenance.

FILE AND RECORD STRUCTURE SERVICES

ENSCRIBE offers the application programmer a choice of three file organizations: *Key-Sequenced (Indexed)*, *Relative*, or *Entry-Sequenced*. In addition, both fixed-length and variable length record formats are supported under ENSCRIBE. A major advantage of fixed-length records is high performance during data access. Variable-length records optimize storage space on the disc. Both the file organization and record format must be specified at file definition time.

Key-Sequenced (Indexed) Files — In key-sequenced file organizations, records are stored in ascending order according to the value assigned a primary key field within the record. The primary key field may be defined as any contiguous set of bytes within the record. For example, a customer's name in an invoice file may be defined as the primary key. All records in an indexed file are variable-length. Primary keys must be unique. Up to 255 secondary keys can be specified.

Relative Files — Records are stored in a relative file according to their relative position within the file. The primary key for a relative file is the record number. These record numbers are ordinal values. Each record in a relative file is a fixed-length record. Relative files can have secondary keys.

Entry-Sequenced Files — In an entry-sequenced file, records are stored according to the sequential order in which they are presented to the system. These records may be either fixed or variable length — but once entered, the record's size may not be changed. The primary key for an entry-sequenced file is the record's logical address. Entry-sequenced files can have alternate keys.

Multi-Keyed Records — Under ENSCRIBE, a single file may have up to 255 alternate key fields, and the values contained in these alternate keys need not be unique. Thus, each file may have a primary key and alternate keys. These key fields are used to construct a secondary key index. This scheme is commonly called *Multi-Indexing*. The user need only specify the key values to access records. Moreover, ENSCRIBE automatically keeps all indices up-to-date during record updates and record insertions.

Multi-Indexing — Indexing provides the greatest flexibility in accessing records. Records may be accessed randomly by specifying a key or keys, or sequentially by consecutively accessing the records in the collated order of an index. Since multiple key fields can be defined, multiple indices allows an indexed file to appear as sequential. Moreover, ENSCRIBE provides three indexing options:

- Exact Key Match
- Approximate Key Match
- Generic Key Match

Exact key match means that the record's key field must exactly match the specified key. Approximate match means that the record key may be equal or greater than the search key. This allows a user program to access records without knowing the exact key. Generic key match means that only the initial portion (partial key) of a key need be specified (such as the prefix to a part number in a vendor's record).

Cache Buffering — Transparent to the application program, ENSCRIBE transfers one or more physical data blocks to main memory. Records within these blocks are then made available to the program. The purpose of this cache buffering is to keep the most recently accessed blocks in main

memory. Thus, index blocks for a Key-Sequenced File are likely to remain in the cache. ENSCRIBE automatically ensures that enough buffer space is available via overlaying the least recently accessed blocks, and also ensures that extraneous physical accesses are avoided.

FILE AND RECORD OPERATIONS

At program execution time, the user program may issue requests for various ENSCRIBE services. These services fall into two categories: *file* operations and *record* operations. Figure 2 illustrates a few of the available operations.

File Operations — File operations may be categorized as follows:

- Definition of an ENSCRIBE file organization.

- Opening of an existing ENSCRIBE file for processing.
- Closing of an ENSCRIBE file and termination of processing on that file.
- Examination of the attribute information stored within an ENSCRIBE file.
- Extension of the allocated space in an ENSCRIBE file.

Record Operations — Record operations may be summarized as follows:

- Read a record.
- Find a record.
- Insert a record.
- Update an existing record.
- Delete a record.
- Lock/Unlock a record.

```

INT .filename[0:11] := "$SYSTEM USER  FILE  ",
    .prim^key[0:2] := [5,0,1024],
    .buffer[0:49];
INT file^number, read^cnt, wrt^cnt, cnt^read, cnt^wrt;

STRING key[0:5];

LITERAL prim^ext = 5,
        file^code = 888,
        sec^ext = 2,
        file^type = 3,
        rec^len = 100,
        blk^len = 1000;

! Create a new keyed file
CALL CREATE(file^name,
            prim^ext,
            file^code,
            sec^ext,
            file^type,
            rec^len,
            blk^len,
            prim^key);

! Insert a new record into a keyed file
CALL WRITE(file^number,buffer,wrt^cnt,cnt^wrt);

! Find a record by key value
CALL KEYPOSITION(file^number,key);

! Read a record from a file
CALL READ(file^number,buffer,read^cnt,cnt^read);

! Update a record
CALL READUPDATE(file^number,buffer,read^cnt,cnt^read);
.
.
Update the record in buffer
.
CALL WRITEUPDATE(file^number,buffer,wrt^cnt,cnt^wrt);

! Delete a record from a file
CALL WRITEUPDATE(file^number,buffer,0,cnt^wrt);

! Lock a record
CALL LOCKREC(file^number);
.
.
Next Record Read will be Locked
.

```

Figure 2. File and Record Management Examples

TANDEM ENVOY™

DATA COMMUNICATIONS MANAGER

FEATURES

- Fail-Safe Data Communications Manager for Non-Stop™ Transaction-Oriented Applications
- Multiple Communications Protocols
 - IBM Binary Synchronous (BSC)
 - ADM-2 Asynchronous
 - TINET Asynchronous
 - Burroughs Synchronous
- Multiple Line Types (Data Links)
 - Point-to-Point (BSC only)
 - Centralized Multipoint Supervisor
 - Centralized Multipoint Tributary
- Trace Facility, Line Usage & Error Statistics, On-line Testing
- Support for Auto Call Facility
- Support for Multiple Modem Types
 - Bell type 103, 202 Asynchronous
 - Bell type 201, 208 Synchronous
- Simplified Applications Programming with Calls to GUARDIAN Operating System Procedures

INTRODUCTION

The ENVOY *Data Communications Manager* provides an efficient, easy-to-use interface between transaction-oriented application programs and the telecommunications network. And since ENVOY is under control of the GUARDIAN *Operating System*, it has the same inherent *fail-safe* features of all other Tandem systems software. As such, ENVOY ensures that data communications are maintained even in the event of a processor or I/O channel failure. ENVOY provides all the control necessary for switched or leased point-to-point and leased multi-point telecommunication networks. An automatic calling option allows unattended stations on switched networks to communicate and exchange data without operator intervention. The ENVOY Data Communications Manager supports Synchronous transmission at speeds up to 56K Bps and Asynchronous transmission at speeds up to 19.2K Bps.

POINT-TO-POINT DATA LINK

ENVOY supports a Binary Synchronous point-to-point data link over either a switched

or leased (non-switched) lines. In the switched network mode, each of two stations has a unique telephone number. The originating station dials the answering (remote) station and establishes a connection. After the connection has been established, system security messages may be interchanged to ensure the integrity of the two stations. Information interchange may then proceed. Once the interchange has been completed, the calling station automatically disconnects. The answering station detects the disconnection and then it also automatically disconnects. In the leased (non-switched) mode, the two stations are permanently connected (no number is required), but the data exchange sequence is the same as that described above for switched lines.

MULTIPOINT DATA LINK

A multipoint data link consists of two or more stations communicating over a leased (non-switched) line. With this type of data link, one station is designated the *supervisor* and controls all communications over the link. All other stations are designated *tributaries*. In a *centralized* multipoint link, communications can occur only between the supervisor and a tributary (tributary-to-tributary communication is not allowed). Each tributary station in a multipoint link is identified by a *polling* address and a *selection* address.

To solicit data transfers from the tributary stations, the supervisor station periodically *polls* each tributary station in the multipoint link by transmitting each tributary's polling address. When a tributary that has data to send is polled, further polling of the data link stops. At this point, the tributary responds by transmitting its data to the supervising station.

To transfer data to a tributary station, the supervisor station first *selects* the desired tributary station by transmitting the tributary's selection address. For selection to occur, the tributary station must be monitoring the line and be willing to accept the forthcoming data. Following selection, the supervisor station transmits the data to the tributary station.

TANDEM ENVOY™

ENVOY PROCEDURES

ENVOY functions are implemented via making calls to the *GUARDIAN Operating System's File Management Procedures*. These procedures are as follows:

- The **OPEN** Procedure is used to gain access to a data communications line.
- The **DEFINELIST** Procedure is used by application processes operating as a multi-point supervisory or tributary station. For a supervisory station, **DEFINELIST** specifies the polling address and selection address of each station. For a tributary station, **DEFINELIST** specifies the polling address(s) and selection address(s) that identify the tributary when the multi-point line is polled.
- The **WRITE** Procedure is used to transmit data to a remote station.
- The **READ** Procedure is used to accept data from a remote station.
- The **WRITEREAD** Procedure is used to transmit data to a remote station and then await a reply.
- The **HALTPOLL** Procedure is used by a station operating as a multipoint supervisor to stop continuous polling of all stations on a line.
- The **CHANGELIST** Procedure is used by application processes operating as a multipoint supervisory or tributary station. For a supervisory station, **CHANGELIST** is used to enable/disable polling of a particular station. Additionally, **CHANGELIST** is used to restore polling of non-responding units.
- The **CLOSE** Procedure is used to terminate access to a line.

TRACE FACILITY

ENVOY provides an aid in checkout of communications applications with the Trace Facility. The Trace Facility works with all line types except auto call units.

Trace Facility records line "events" in a Trace Table. Each Trace Table entry provides;

- Sequence Number
- Controller and Line Number
- Line State
- Event
- Miscellaneous Information
- Timestamp

LINE STATISTICS

Line statistics are maintained by ENVOY for each line from the time a line is opened until the line is closed. The statistics are printed on the operator console when a predetermined error threshold is exceeded, when a path switch occurs, or when the line is closed.

ON-LINE TESTING

ENVOYTST (Envoy Test) is used to verify the operation of the data communications process and the synchronous controller operating as the following line types:

- Point-to-point non-switched primary
- Point-to-point non-switched secondary
- Point-to-point switched primary
- Point-to-point switched secondary
- Multipoint supervisor
- Multipoint tributary

TANDEM

Enform - Query/Report Writing Language

Features

Query

- Powerful English-like relational query language easily used by non-programming personnel
- Retrieves data from multiple files which may be related in ways not anticipated during database design
- Used with Expand, allows queries on a distributed database
- Automatically develops the most efficient strategy to extract data from your database
- Keywords may be easily redefined to a different language, such as German, Spanish or French
- Uses Tandem's DDL statements to define database

Report

- Produces reports at a fraction of the programming cost and time of conventional languages, such as COBOL
- Reporting options allow sorting and summarizing of retrieved data as well as evaluation of built-in or user-defined functions
- Automatically spaces information on a page and supplies headings
- Accumulates information with your own formula, such as calculating commissions for sales people
- Formats numbers with commas, decimal point, currency sign
- Can be used from TAL, COBOL or FORTRAN programs to replace tedious report-formatting coding

ENFORM will make a difference in the time and energy required to extract a report from your database because you don't have to write a program, or be a programmer, to use it. ENFORM automatically writes the information gathered into rows and columns with headings and page numbers. If you want something different from the automatic settings, you may override them.

Enform Queries

Gathering the data from your database is done with the query portion of ENFORM. If your database was designed with specific relationships in mind, then you can take full advantage of them with ENFORM. However, what if you want the files related in a different manner, a manner not originally planned into the database? ENFORM handles this with no trouble; just tell it how you want the files related through a LINK statement.

Furthermore, ENFORM automatically selects an efficient method of searching for the data you requested. It uses existing keys and alternate keys to speed up the search (and cut down the cost of producing the report).

Defining the Database

The only task you must do before using ENFORM is define the database records with Tandem's Data Definition Language (DDL):

```
RECORD order.
05 order-number; PIC "9(4)".
05 customer; PIC "X(35)".
05 part-number; PIC "9(5)".
05 quantity; PIC "999".
RECORD parts.
05 part-number; PIC "9(5)".
05 cost; PIC "9(5) V99".
05 part-name; PIC "X(30)".
05 stock-amount; PIC "9(4)".
```



Ease of Use

ENFORM's statements and commands look like English words. For example:

```
OPEN order, parts
LINK order to parts VIA part-number;
LIST BY order-number,
customer, part-number, quantity,
(quantity * cost) HEADING "TOTAL COST",
subtotal;
```

Keywords don't have to look like English — they may resemble any language you prefer, such as German or Spanish, by redefining them.

Entering the Statements

You can use ENFORM interactively by typing in the statements, one at a time, or put the commands into a file for later execution. You can save the "compiled" output from an ENFORM session so the ENFORM statements don't have to be checked and processed each time the report is created.

Choosing the Report Data

Criteria for selecting the data you want is defined with a WHERE statement:

```
OPEN parts
LIST part-number, part-name
WHERE stock-amount LESS THAN 5
AND part-number GREATER THAN 250;
```

Enform Reports

ENFORM's report phase automates many of the formatting details most reports require, such as centering titles and headings, spacing columns of data, underlining headings, skipping to a new page, writing headings. Special keywords further enhance these report duties, such as totaling, sub-totaling, centering data, overriding headings, formatting numbers with commas, periods or currency sign.

Accumulation Operations

Common accumulation operations — percentage, cumulative

sum, sum, average, count, maximum, minimum — are built into ENFORM:

```
LIST BY part-number
MIN (cost)
MAX (cost)
WHERE part-number EQUAL TO 5000 THRU 5999;
```

If you have an accumulation scheme not satisfied by these operations, you can build an expression to meet your needs.

Sample Report

10/04/78 — 10:39:35 AM

ORDER DETAIL REPORT

OCT 04TH 1978

ORDER-NUMBER	PART-NUMBER	COST	QUANTITY	TOTAL-COST
35	244	87,000.00	1	87,000.00
	2001	1,500.00	2	3,000.00
	2403	9,600.00	4	38,400.00
	3103	10,500.00	2	21,000.00
	3302	2,800.00	1	2,800.00
	4103	24,500.00	2	49,000.00
	6301	2,900.00	1	2,900.00
	6302	4,300.00	2	8,600.00
				<u>212,700.00</u>

TO: FRESNO STATE BANK FRESNO, CALIFORNIA
ORDER-DATE: 03/03/78 DELIVERY-DATE: 08/10/78

38	244	87,000.00	1	87,000.00
	2402	7,500.00	3	22,500.00
	3102	4,800.00	1	4,800.00
	4102	14,500.00	1	14,500.00
	5502	11,500.00	1	11,500.00
	6201	5,800.00	1	5,800.00
	6302	4,300.00	1	4,300.00
	6402	1,500.00	2	3,000.00
				<u>153,400.00</u>

TO: BROWN MEDICAL CO., SAN FRANCISCO, CA
ORDER-DATE: 03/19/78 DELIVERY-DATE: 08/20/78

TANDEM

TANDEM COMPUTERS INC., 19333 Vallco Parkway, Cupertino, California 95014 • Toll Free 800-538-9360 or (408)996-6000 in California. Branch offices throughout the United States. Foreign Offices in Frankfurt, Dusseldorf and Munich; West Germany • Uxbridge, Middlesex; England • Zurich; Switzerland • Toronto; Canada.

Programming
Languages

PROGRAMMING LANGUAGES

The basic unit of information in the Tandem 16 is the 16-bit word. The word defines the Tandem 16's machine instruction length and its logical addressing range.

A Tandem 16 program executing in the processor module consists of: 1) a code area that contains the executable machine instruction codes, and 2) a separate data area that contains the program's variables and buffers. The code area for a given program consists of a maximum of 65,536 words. Likewise, the maximum size of the data area for a given program is 65,536 words.

A code area is comprised of one or more procedures. A procedure is a block of machine instructions that can be called into execution to perform some specific task. A procedure (i.e., the block of instructions that a procedure represents) can be invoked (called) from any point in the program. When a procedure is called, the current environment is automatically saved by the hardware; when the procedure finishes, the previous environment is automatically restored. The procedure itself executes in an environment separate from other procedures.

The code part of a program is not modifiable. Therefore, all code is inherently sharable and reentrant.

T/TAL

Programs for the Tandem 16 are written in Tandem's Transaction Application Language (T/TAL). T/TAL is a high-level, block-structured, procedure-oriented language designed for ease of programming and efficient use of the many architectural features of the Tandem 16 (such as separate code and data).

A typical statement written in T/TAL looks like this:

```
IF item = taxable THEN payment := price
+ (price *tax rate); the upper case
elements are reserved words in T/TAL;
the lower case elements are data variables,
:= means "is assigned the value of," and
* means multiply.
```

Some characteristics of T/TAL are:

- Free-Form Structure

The free-form structure of T/TAL permits programmers to format their programs in a manner providing readability and self documentation.

- Machine Independent

T/TAL programs are written using such high-level constructs as: IF THEN, WHILE DO, DO UNTIL, CASE, etc. The T/TAL compiler generates optimum code taking advantage of the Tandem 16's hardware characteristics.

Features are provided in T/TAL for programmers desiring to explicitly operate on hardware registers (STACK and STORE statements) and program at an assembly language level (CODE statement).

- Identifiers

Program elements such as constants, variables, labels, and procedures are identified throughout a source program by use of symbolic, program-assigned identifiers. This eliminates the need for a programmer to keep track of specific memory addresses. An identifier can contain up to 31 alpha-numeric characters.

- Data Types

INT (integer), INT (32) (double word integer), STRING (byte), and FIXED (18-digit fixed point).

- Block (Multiple Element) Operations

T/TAL provides multiple element operations such as move block, compare blocks, and scan block.

- Bit Operations

T/TAL provides bit operations such as bit deposit, bit extraction, and bit shifts.

Procedures

As previously mentioned, a procedure is a block of machine instructions that exists only once in a program but can be called into execution from any point in the program. Procedures, as implemented in the T/TAL language, have special properties that make them nearly as versatile as complete programs. A program has a global data area that is accessible only by statements within that program; a procedure has its own (local) data area that is accessible only by statements within that procedure. Unlike the program's global data area, however, a procedure's local data area is allocated and initialized only when the procedure executes. This provides two major benefits: 1) storage is not allocated except when needed (keeping the total amount of storage required by a program to a dynamic minimum) and 2) the data area is initialized when the procedure is entered.

Recursive Procedures

Because a procedure has its own local data area and the data area is initialized each time the procedure is entered, a procedure can call itself.

Subprocedures

Subprocedures are similar to procedures in that they can have their own variables and can be called recursively. However, a subprocedure is a part of a procedure and therefore can be called only from the procedure in which it resides.

COBOL ANSI '74, Level 2

COBOL is a procedural language designed to read much like ordinary English, and can automatically provide much of its own documentation. Except where noted, Tandem COBOL conforms to American National Standard COBOL (Standard COBOL). Standard COBOL is basically a machine-independent language, yet it promotes efficiency across a wide variety of computer systems by accommodating their differences. Some of the ways Standard COBOL cooperates with an operating system do not apply to the Tandem 16 system. Nonetheless, the Tandem COBOL compiler accepts the syntax for those language conventions. Where possible, they are viewed as comments.

FORTRAN ANSI '77

Tandem FORTRAN conforms to the full language as specified in the document "American National Standard Programming Language FORTRAN, X3.9-1977." In addition, Tandem has extended the language to include NonStop features consistent with the Tandem philosophy. FORTRAN object code utilizes the new Tandem Floating Point microcode.

MUMPS - ANSI X11.1-1977 (Mgh Utility Multi-Programming System)

Language developed at Massachusetts General Hospital (MGH) in Boston to meet the needs of information handling by computer in the extremely diverse health care environment. It is oriented towards the storage and retrieval of character string information, such as names, addresses and text rather than towards performing complex calculations with numbers.

DDL (Data Definition Language)

Allows centralized administration of a data base to accommodate any number of diversified application programs. All data field references are defined in a schema (Codyl-based compiler); changes to record layout, and additions and/or changes to record types and/or fields may be accomplished without code modifications to existing programs. It provides for easy-to-use EDIT and SCHEMA Programs to facilitate compilation of application programs with high-level languages, TAL, COBOL and FORTRAN.

GUARDIAN features handled by Tandem/COBOL include:

- * NonStop Operation
- * Shared, Re-entrant Code
- * Virtual Memory System
- * Single-line Independence of I/O Devices
- * Checkpoint/Recovery Facilities

ENCIPHER features handled by Tandem/COBOL include:

- * Key-encrypted, Entry-encrypted and Relative file structures
- * Logical file size to 4 BILLION bytes
- * One primary and up to 31 alternate keys
- * Optional mirror data (aka mirroring)

SEQUENTIAL FILE - Level 2

Sequential files consist of records of a file one after the other. In the code they were written, memory addresses are shared, and the file pointers for the FILE CONTROL, IO CONTROL, and FD entries are included. Within the procedure DIVISION, CLOSE, OPEN, READ, REWRITE, and WRITE are assigned.

RELATIVE FILE - Level 1

Relative files consist of records in order of their sequential nature. Each record in a relative file is uniquely identified by an integer value greater than zero which specifies the record's logical position in a file. It provides for FILE CONTROL, IO CONTROL, and FD entries.

TANDEM COBOL

TANDEM COBOL

FEATURES

- Conforms to ANSI COBOL X3.23-1974
- Full compatibility with ENSCRIBE file structures
- Runs in mixed language environment with T/TAL
- Includes Tandem Extension for NonStop^(TM) Programming

INTRODUCTION

Tandem/COBOL runs on the Tandem T16 Computer System — the only multi-processor system designed for NonStop, transaction-oriented, data base applications. Tandem/COBOL utilizes all the capabilities of Tandem's GUARDIAN Operating System and ENSCRIBE Data Base Record Manager — software designed to keep the system up and running while maintaining the integrity of the on-line data base. Thus Tandem/COBOL is compatible with programs written in T/TAL, Tandem's high level language for transaction processing, and is capable of running in a multi-language environment.

GUARDIAN features handled by Tandem/COBOL include;

- NonStop Operation
- Shared, Re-entrant Code
- Virtual Memory System
- Geographic Independence of I/O Devices
- Checkpoint/Checkmonitor Facilities

ENSCRIBE features handled by Tandem/COBOL include;

- Key-sequenced, Entry-sequenced and Relative file structures
- Logical file size to 4 BILLION bytes
- One primary and up to 31 alternate keys
- Optional mirror data base recording

LEVEL OF SUPPORT

NUCLEUS — Level 2

The nucleus gives a basic language for the internal processing of data within the four divisions of a program. Qualification, punctuation characters, data-name formation, connectives, and figurative constants are supported as well as ACCEPT, ADD, ALTER, COMPUTE, DISPLAY, DIVIDE, EXIT, GO, IF, INSPECT, MOVE, MULTIPLY, PERFORM, STOP, STRING, SUBTRACT, and UNSTRING statements.

Extensions to the language include the ability to call T/TAL procedures via the ENTER verb.

TABLE HANDLING — Level 2

Table Handling lets you define three dimensional fixed length tables of contiguous data items. Each item is accessed relative to its position in the table. Each item may be identified through use of a subscript or an index. This level also provides series options and the ability to vary the contents of indices by an increment or decrement. Table handling includes the use of the SET and SEARCH statements.

SEQUENTIAL I-O — Level 2

Sequential I-O reads records of a file one after the other, in the order they were written. Memory areas among files are shared, and the full facilities for the FILE-CONTROL, I-O-CONTROL, and FD entries are included. Within the Procedure Division, CLOSE, OPEN, READ, REWRITE, USE, and WRITE are recognized.

RELATIVE I-O — Level 2

Relative I-O accesses records in either random or sequential manner. Each record in a relative file is uniquely identified by an integer value greater than zero which specifies the record's logical position in a file. It provides full facilities for the FILE-CONTROL, I-O-CONTROL and FD entries.

TANDEM COBOL

TANDEM COBOL

Within the Procedure Division it provides full capabilities for the CLOSE, DELETE, OPEN, REWRITE, START, USE, and WRITE statements.

INDEXED I-O – Level 2

Indexed I-O accesses records in either random or sequential manner. Each record in an indexed file is identified by the value of one or more keys within that record. It provides full facilities for the FILE-CONTROL, I-O-CONTROL and FD entries. Within the Procedure Division it provides full capabilities for the CLOSE, DELETE, OPEN, READ, REWRITE, START, USE, and WRITE statements.

SORT-MERGE – Level 2

Sort-Merge orders one or more files of records, or combines two or more identically ordered files of records, according to a set of keys within each record. Optionally, you may apply some special processing to each of the individual records by input or output procedures using RELEASE and RETURN. This special processing may be applied before and/or after the records are ordered by SORT or after the records have been combined by MERGE. Also, you may sort one or more files, or combine two or more files, one or more times within a given execution of a COBOL program.

REPORT-WRITER – Null Level

SEGMENTATION – Null Level

Segmentation is automatic via Tandem's GUARDIAN Operating System.

LIBRARY – Level 1

Text from a library is copied into the source program using COPY. The copied text is not changed.

DEBUG – Level 1

Debug provides a basic debugging capability, including the ability to specify (1) selective or full procedure monitoring, and (2) optionally compiled debugging statements.

INTER-PROGRAM COMMUNICATION – Level 1

Inter-program Communication means a program can communicate with one or more programs using CALL and CANCEL. This

communication is done by (1) transferring control from one program to another within a run unit, and (2) letting both programs have access to the same data items.

COMMUNICATION – Null Level

TANDEM COBOL EXTENSIONS

Several extensions have been added to Tandem/COBOL to permit use of Tandem's GUARDIAN Operating System.

1. Nonstop Extensions

For Nonstop programming in COBOL, you include a "?NONSTOP" line in your source program. STARTBACKUP and CHECKPOINT are the verbs that make the program nonstop. Normally STARTBACKUP is called once at the beginning of the program to set the nonstop mode. Thereafter the CHECKPOINT verb is used to pass information to the backup process at critical points in the processing. In a nonstop program, checkpoints will also occur automatically upon any OPEN or CLOSE executed after the backup is established. Both of these verbs will set the special register, PROGRAM-STATUS, to indicate the outcome of the checkpointing operation.

2. Extensions to the standard COBOL I/O facility

Three new verbs, LOCKFILE, UNLOCKFILE, and UNLOCKRECORD, have been introduced to allow the use of the corresponding system file and record locking routines. This addition allows separate processes to share a common data base. READ and REWRITE verbs have been extended to allow the specification of a LOCK or UNLOCK operation. The OPEN syntax has been extended to specify the file access, EXCLUSIVE, SHARED or PROTECTED, and to permit the SYNCDEPTH for files opened in the OUTPUT, I-O or EXTEND mode.

3. Calling TAL procedures

The ENTER verb has been modified to call TAL procedures. This lets you access any TAL system external or your object library routines. These object library routines are assigned to the COBOL run unit at compile time.

DIMENSION M (10,20)
 READ 100, I, J, K
 DO 200, IS = I, 10
 DO 200, JS = J, 20
 200 M (IS, JS) = K
 IF (I .EQ. J) GO TO 300

TANDEM FORTRAN

- Conforms to full language specifications of ANSI FORTRAN, X3.9-1977
- Full use of all ENSCRIBE facilities including multi-key access and locking
- RECORD Structures for DDL compatibility
- Extensions for NonStop^(tm) Programming
- Facilities for Interprocess Communications
- Support of Floating Point Arithmetic

INTRODUCTION

Tandem FORTRAN runs on the Tandem T16 Computer System — the only multi-processor architecture designed for NonStop, transaction-oriented, data base applications. Tandem FORTRAN utilizes all the facilities of the GUARDIAN Operating System including Non-Stop operation, re-entrant code, interprocess communications, virtual memory, and ENSCRIBE data base facilities for keyed, relative, and sequential access, multi-key data paths, and concurrent record access. Tandem FORTRAN is capable of running in a multi-language environment.

LEVEL OF SUPPORT

Tandem FORTRAN conforms to the specifications described in the American National Standards Institute for Programming Languages — FORTRAN, X3.9-1977.

Data Types

Six data types are supported; INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL and CHARACTER. In addition data may be arranged as arrays or substrings. Dimensioned arrays can have up to seven subscripts.

Expressions

Expression types can be arithmetic, character, relational or logical. Operators allowed in expressions include:

Arithmetic	**	(Exponentiation)
	/	(Division)
	*	(Multiplication)
	-	(Subtraction)
	+	(Addition)
Character	//	(Concatenation)
Relational	.LT.	(Less Than)
	.LE.	(Less Than or Equal)
	.EQ.	(Equal)
	.NE.	(Not Equal)
	.GT.	(Greater Than)
	.GE.	(Greater Than or Equal)
Logical	.NOT.	
	.AND.	
	.OR.	
	.EQV.	
	.NEQV.	

Specification Statements

Support is included for DIMENSION, COMMON, EQUIVALENCE, IMPLICIT, PARAMETER, EXTERNAL, INTRINSIC, and SAVE statements. In addition, type statements are allowed for INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL, and CHARACTER.

DATA Statement

DATA Statements can be used for initialization of variables, arrays, array elements, and substrings.

Assignment Statements

There are four kinds of Assignment statements: Arithmetic, Logical, Statement label (ASSIGN), and Character.

TANDEM FORTRAN

Control Statements

Sixteen Control statements are used to control the execution of a program.

Unconditional GO TO	DO
Computed GO TO	CONTINUE
Assigned GO TO	STOP
Arithmetic IF	PAUSE
Logical IF	END
Block IF	CALL
ELSE IF	RETURN
ELSE	
END IF	

Input/Output Statements

Nine Input/Output statements are supported.

READ	OPEN	BACKSPACE
WRITE	CLOSE	ENDFILE
PRINT	INQUIRE	REWIND

Tandem FORTRAN utilizes the UNIT, FMT, REC, IOSTAT, ERR, and END control parameters for READ and WRITE.

For OPEN the UNIT, IOSTAT, ERR, FILE, STATUS, ACCESS, FORM, RECL, and BLANK open parameters can be used. For CLOSE the UNIT, IOSTAT, ERR, and STATUS parameters can be used.

For INQUIRE specifiers may be used for IOSTAT, ERR, EXIST, OPENED, NUMBER, NAMED, NAME, ACCESS, SEQUENTIAL, DIRECT, FORM, FORMATTED, UNFORMATTED, RECL, NEXTREC, and BLANK.

For BACKSPACE, REWIND, and ENDFILE the UNIT, IOSTAT, and ERR specifiers can be used.

FORMAT Specification

A complete range of FORMAT specifiers is available. Both explicit and list-directed editing are allowed. Editing descriptors allowed are:

Apostrophe	P
H	BN and BZ
Positional	Numeric (I, F, E, D, G)
T, TL, and TR	L
X	A
Slash	List-Directed
Colon	S, SP, and SS

Functions and Subroutines

Four categories of procedures are supported:

- Intrinsic Functions
- Statement Functions
- External Functions
- Subroutines

External functions and subroutines may be specified in either FORTRAN or other programming language subprograms.

An ENTRY statement may be used to permit specification of the entry point into a subprogram.

BLOCK DATA Subprogram

Block Data subprograms are allowed for initializing variables and array elements in named common blocks.

TANDEM FORTRAN EXTENSIONS

Several extensions have been made to FORTRAN 77 to enhance its operation on Tandem systems.

1. NonStop Extensions

STARTBACKUP and CHECKPOINT functions allow a FORTRAN program to utilize the Non-Stop capabilities of GUARDIAN. STARTBACKUP is called once at the beginning of a program to establish the nonstop mode. Thereafter CHECKPOINT is used to pass critical information to the backup process. Checkpoints will automatically occur upon any OPEN or CLOSE after the backup has been created.

2. Structures

Structures provide the ability to define records in FORTRAN. The constructs RECORD and END RECORD are used to define record structures. The Data Definition Language may also be used to transcribe a schema into FORTRAN RECORD structures.

3. ENSCRIBE Extensions

Extensions have been made to the FORTRAN READ and WRITE statements to permit the full use of ENSCRIBE facilities. Thus it is possible with FORTRAN statements to access key-sequenced, relative, and entry-sequenced files by primary or up to 255 alternate keys. Provision has been made to allow exact, approximate or generic positioning into an ENSCRIBE file structure using FORTRAN. Concurrent record access is supported with LOCK mechanisms at either the record or file level.

4. Interprocess Communications

FORTRAN processes can communicate with one another or with processes written in other languages through the standard FORTRAN READ and WRITE statements. Communication to other processes is implemented using the interprocess communication facilities of the GUARDIAN Operating System.

TAL

TANDEM APPLICATION LANGUAGE

- HIGH LEVEL LANGUAGE FOR APPLICATIONS AND SYSTEMS PROGRAMMING
- PROCEDURE-ORIENTED LANGUAGE FOR TRANSACTION PROCESSING
- EXTENSIVE DATA TYPES FOR ARITHMETIC AND STRING DATA TYPES
- SPECIAL STRING AND BIT MANIPULATION FUNCTIONS
- OBJECT CODE OPTIMIZATION COMPARABLE TO ASSEMBLER CODING
- FREE-FORM STRUCTURE
- CONSTRUCTS AMENABLE TO STRUCTURED PROGRAMMING

TRANSACTION APPLICATION LANGUAGE (T/TAL)

The Tandem Transaction Application Language is a high level, block structured language designed for the easy implementation of transaction-oriented applications. T/TAL is similar to ALGOL, PL/I or COBOL in providing procedure blocks and high level constructs such as IF...THEN...ELSE, DO...UNTIL, WHILE...DO, FOR, and CASE. Programmers can write self documenting programs with object code generation as efficient as assembler code.

PROCEDURES

The procedure-oriented, block-structured approach allows a programmer to design procedure blocks for functional simplicity. Data can be assigned globally for all procedures to access or can be assigned locally — known only to the procedure in which it is declared. Local data is assigned dynamically and initialized each time the procedure is invoked. The local data feature also allows the use of recursive procedures.

One procedure can call another and pass parameters by value or by reference. The called procedure can optionally return a value through its name.

All procedure code is compiled as re-entrant and, therefore, non-modifiable. This permits code to be shared by many different users thereby minimizing the amount of memory required for multiple applications.

DATA DECLARATIONS

T/TAL provides data types for arithmetic, string and logical operations. Signed integer data can be specified with 15 bits or 31 bits of precision. Fixed data is a scaled decimal with up to 18 digits of precision. String data consists of one or more 8-bit bytes. Logical data consists of 16-bit words.

Data can be declared as simple, single element items or as arrays (multiple elements). In addition a variable can be declared as a pointer variable to facilitate indirect addressing.

Type functions are an integral part of TAL and provide the ability to convert from one data type to another. Type functions are also available to determine if string data represents ASCII NUMERIC, ALPHA or SPECIAL values.

SPECIAL FEATURES

Special string manipulation constructs are available to facilitate efficient processing of transaction data. Included are SCAN, MOVE and COMPARE string operations.

Data can be addressed at the bit level, if desirable. Operations exist in TAL to perform bit deposit, bit extraction, arithmetic shifting and logical shifting.

I/O

All I/O is performed via calls to operating system procedures. Procedures exist to open and close files, read and write data, perform control functions, create and purge files, lock and unlock files, and perform "wait" or "nowait" I/O. Extensive error checking is provided by the File System and error codes are returned to the calling procedure for inspection.

TAL CONSTRUCTS

General Form

```
[label:] statement;
```

Specific Form

Assignment

```
variable := [variable:=...] arithexp;
```

Compound

```
BEGIN  
statement1;  
statement2;  
.  
.  
statementn;  
END;
```

Unconditional Branching

```
GOTO label;
```

Invoking Procedures

```
CALL procedurename [(parameterlist)];  
RETURN [arithexp];
```

Conditional

```
IF condexp THEN statement [ELSE statement];  
CASE index OF BEGIN
```

```
statement0;  
statement1;
```

```
..  
..  
statementn;
```

```
OTHERWISE statement;
```

```
END;
```

Iteration

```
FOR variable := initvalue TO limit [BY step] DO statement;  
WHILE condexp DO statement;  
DO statement UNTIL condexp;
```

String Manipulation

```
destarray { :=* } { sourcearray FOR count } { --nextaddr } ;  
          { :=* } { constantlist }  
  
{ SCAN } array { WHILE } testchar { --addr } ;  
{ RSCAN } array { UNTIL }
```

The listing on the next page shows a simple TAL program including GLOBAL and LOCAL data declarations and a procedure block. Also included is the variable map and procedure map which are used when debugging the program.

TANDEM APPLICATION LANGUAGE

TAL - TANDEM COMPUTERS VERSION A01 (6/25/76 - 10 AM)
 DATE - TIME : 6/28/76 - 11:56:41

SOURCE LANGUAGE: TAL - TARGET MACHINE TANDEM/16
 OPTIONS: ON (LIST, CODE, MAP, WARN, LMAP) - OFF (ICODE, INNERLIST)

```

1. 000000 0 0 ?NOCODE
2. 000000 0 0 ! GLOBAL DATA DECLARATIONS
3. 000000 0 0 INT ,HOMETERM[0:11], ,LINE[0:35];
4. 000060 0 0 STRING ,LINES := @LINE '<<' 1;
5. 000060 0 0 DEFINE NULL = %0%, CRLF = [%15,%12]#;
6. 000060 0 0 ?NOLIST
9. 000000 0 0
10. 000000 0 0 PROC CHANGE*DATE MAIN;
11. 000000 1 0 BEGIN
12. 000000 1 1 ! LOCAL DATA DECLARATIONS
13. 000000 1 1 INT SAVE*ADDR1, SAVE*ADDR2, SAVE*ADDR3, MONTH*IN*DECIMAL, TERM;
14. 000000 1 1 INT COUNT, STATUS, INPUT*DATA*POINTER;
15. 000000 1 1 INT LENGTH*YEAR, LENGTH*MONTH, LENGTH*DAY;
16. 000000 1 1 STRING ,YEAR[0:3], ,MONTH[0:2], ,DAY[0:1], ,TEMPS[0:3];
17. 000000 1 1 STRING ,MONTH*TABLE[0:35] := "JANFEBMARAPRMAJUNJULAUGSEPCTNOVDEC";
18. 000022 1 1
19. 000022 1 1 CALL MYTERM(HOMETERM); ! GET TERM ID AND OPEN TERMINAL FILE
20. 000050 1 1 CALL OPEN(HOMETERM,TERM,0);
21. 000055 1 1 NEXT: LINES := "ENTER DATE YY/MM/DD" & CRLF;
22. 000072 1 1 CALL WRITEREAD(TERM,LINE,21,9,COUNT); ! GET DATE AS YY/MM/DD
23. 000103 1 1 IF COUNT = 0 THEN CALL STOP; ! IF NO DATA THEN QUIT
24. 000111 1 1 LINES(COUNT) := " "; ! INITIALIZE FOR SCAN
25. 000121 1 1 ! GET THE YEAR AND EXPAND IT
26. 000121 1 1 SCAN LINES UNTIL "/" -> SAVE*ADDR1; ! SCAN FOR FIRST SLASH
27. 000125 1 1 LENGTH*YEAR := SAVE*ADDR1 - @LINES; ! CALCULATE LENGTH OF YEAR FIELD
28. 000131 1 1 YEAR := "19" & LINES[0] FOR LENGTH*YEAR; ! CONCATENATE "19" TO YEAR FOR A 4 DIGIT VALUE
29. 000143 1 1 INPUT*DATA*POINTER := SAVE*ADDR1 - @LINES + 1; ! CALCULATE POINTER TO NEXT FIELD IN DATE
30. 000150 1 1 ! GET THE MONTH, CHECK FOR VALIDITY, AND CONVERT TO ALPHA
31. 000150 1 1 SCAN LINES[INPUT*DATA*POINTER] UNTIL "/" -> SAVE*ADDR2; ! SCAN FOR SECOND SLASH
32. 000155 1 1 LENGTH*MONTH := SAVE*ADDR2 - SAVE*ADDR1 - 1; ! CALCULATE LENGTH OF MONTH
33. 000162 1 1 TEMPS := LINES[INPUT*DATA*POINTER] FOR LENGTH*MONTH & NULL;
34. 000174 1 1 ! SAVE MONTH NUMERIC VALUE FOR VALIDITY CHECK AND ALPHA LOOKUP
35. 000174 1 1 CALL NUMIN(TEMPS,MONTH*IN*DECIMAL,10,STATUS); ! CONVERT ASCII MONTH TO INTEGER
36. 000203 1 1 IF STATUS <= 0 THEN BEGIN ! IF CONVERSION IS NOT GOOD
37. 000206 1 2 ERROR: LINES := "INPUT ERROR";
38. 000215 1 2 CALL WRITE(TERM,LINE,11,COUNT);
39. 000225 1 2 GOTO NEXT;
40. 000234 1 2 END;
41. 000234 1 1 IF MONTH*IN*DECIMAL > 12 THEN GOTO ERROR;
42. 000240 1 1 ! FIND ALPHABETIC VALUE FOR MONTH*IN*DECIMAL
43. 000240 1 1 MONTH := MONTH*TABLE[3 * (MONTH*IN*DECIMAL - 1)] FOR 3;
44. 000251 1 1 INPUT*DATA*POINTER := SAVE*ADDR2 - @LINES + 1; ! CALCULATE POINTER TO NEXT FIELD IN DATE
45. 000256 1 1 ! GET THE DAY AND SAVE
46. 000256 1 1 SCAN LINES[INPUT*DATA*POINTER] UNTIL NULL -> SAVE*ADDR3;
47. 000263 1 1 LENGTH*DAY := SAVE*ADDR3 - SAVE*ADDR2 - 1;
48. 000270 1 1 DAY := LINES[INPUT*DATA*POINTER] FOR LENGTH*DAY;
49. 000274 1 1 ! FORMAT AND OUTPUT THE NEW DATE
50. 000274 1 1 LINES := CRLF & MONTH FOR 3 & " " & DAY FOR LENGTH*DAY & " " & YEAR FOR LENGTH*YEAR+2;
51. 000331 1 1 CALL WRITE(TERM,LINE,10+LENGTH*DAY+LENGTH*YEAR,COUNT);
52. 000344 1 1 GOTO NEXT;
53. 000351 1 1 END: ! END OF PROC
    
```

COUNT	VARIABLE	INT	L+006	DIRECT
DAY	VARIABLE	STRING	L+016	INDIRECT
ERROR	LABEL			
INPUT*DATA*POINTER	VARIABLE	INT	L+010	DIRECT
LENGTH*DAY	VARIABLE	INT	L+013	DIRECT
LENGTH*MONTH	VARIABLE	INT	L+012	DIRECT
LENGTH*YEAR	VARIABLE	INT	L+011	DIRECT
MONTH	VARIABLE	STRING	L+015	INDIRECT
MONTH*IN*DECIMAL	VARIABLE	INT	L+004	DIRECT
MONTH*TABLE	VARIABLE	STRING	L+020	INDIRECT
NEXT	LABEL			
SAVE*ADDR1	VARIABLE	INT	L+001	DIRECT
SAVE*ADDR2	VARIABLE	INT	L+002	DIRECT
SAVE*ADDR3	VARIABLE	INT	L+003	DIRECT
STATUS	VARIABLE	INT	L+007	DIRECT
TEMPS	VARIABLE	STRING	L+017	INDIRECT
TERM	VARIABLE	INT	L+005	DIRECT
YEAR	VARIABLE	STRING	L+014	INDIRECT
CHANGE*DATE	PROC			
CRLF	DEFINE			[%15,%12]
HOMETERM	VARIABLE	INT	G+000	INDIRECT
LINE	VARIABLE	INT	G+001	INDIRECT
LINES	VARIABLE	STRING	G+002	INDIRECT
MYTERM	PROC			
NULL	DEFINE			%0%
NUMIN	PROC	INT		
OPEN	PROC			
STOP	PROC			
WRITE	PROC			
WRITEREAD	PROC			

PEP	BASE	LIMIT	ENTRY	ATTRIBUTES	NAME
002	000003	000402	000025	M	CHANGE*DATE

OBJECT FILE NAME IS \$SCR,SYSTEM,SAMPLE
 NO. ERRORS=0 ; NO. WARNINGS=0
 PRIMARY GLOBAL STORAGE=3
 SECONDARY GLOBAL STORAGE=4A
 CODE SIZE=256
 DATA AREA SIZE=2 PAGES
 CODE AREA SIZE=1 PAGES
 MAXIMUM SYMBOL TABLE SIZE=3A3
 ELAPSED TIME - 01 0:19

TANDEM

MUMPS

ANSI '77 Standard MUMPS Interpreter

TANDEM MUMPS is a high-level, interpreted computer language designed for data management. MUMPS is used extensively in commercial and medical applications because of its ease of use, richness of features, ease of learning and superior string manipulation capabilities.

TANDEM MUMPS features include:

- Program structures that are easy to code, debug and modify;
- Modular expansion of the system without reprogramming applications, allowing the most cost-effective use of your investment;
- A vendor-supported MUMPS implementation fully integrated with the multifunctional GUARDIAN Operating System;
- High availability and reliable operation using **NonStop™** System facilities;
- Concurrent execution of COBOL, FORTRAN or TAL programs for increased programming flexibility;
- Easy access to non-MUMPS files for maximum utilization of system resources;
- Versatile development tools to cut programming costs.

TANDEM MUMPS supports all features of the ANSI 1977 MUMPS Language Standard. In addition, **TANDEM MUMPS** differs from other implementations in that it runs under the vendor-supported GUARDIAN Operating System. This means that **TANDEM MUMPS** has the unique benefits of the **NonStop** System which allows applications to continue to run even if a processor, disc, controller or I/O channel should fail during operation.

TANDEM MUMPS Extensions

TANDEM MUMPS provides the most powerful extensions of MUMPS available to date. In addition to all the benefits available with Standard MUMPS, TANDEM supplies an interface to COBOL, FORTRAN, TAL and PATHWAY programs and the

reliability of **NonStop** computer systems, plus the following extensions:

1. **String subscripting**, which lets you give variables meaningful names such as NAME("LAST") rather than the cryptic numeric identifiers such as NAME(4). String subscripts can be up to 250 characters long.
2. **Negative and decimal number subscripting**, which lets you subscript variables with negative numbers, TIME(-12), and non-integer numbers, VARIANCE(-.75), MEAN(-25.67).
3. The **\$ORDER** function as proposed by the MUMPS Development Committee, which returns all subscript values including non-numeric strings. The \$ORDER is supplied in addition to the \$NEXT function.
4. The **\$ZFINFO** function, which returns information on all system files (e.g., key sequenced, entry sequenced and relative disc files).
5. The **ZTRAP** command, **\$ZTRAP** function and **\$ZTRAP special variable** which together provide a complete error trapping facility that can be tailored to individual applications.
6. **Commands and utilities** that create, edit and delete application and utility routines.
7. Commands that give access to **information on the state of the GUARDIAN Operating System** environment in which MUMPS is running.
8. **READ and WRITE** commands which provide the capability of accessing sequential and key-sequenced files with record sizes up to 4 k bytes.
9. **Extension to the MUMPS READ and WRITE commands**, providing for interprocess communication with COBOL, FORTRAN, and TAL and other MUMPS processes.
10. **Simple interface** to the PATHWAY Transaction Processing System, which allows CRT terminals to interact with applications written in MUMPS, as well as other TANDEM languages.

Versatile Operating Environment

TANDEM MUMPS runs under the GUARDIAN Operating System with full system resources at its disposal. **TANDEM MUMPS** users are able to concurrently execute COBOL, FORTRAN and TAL programs, as well as take advantage of the unique features available with a *NonStop* System, including a fault-tolerant mode. With **TANDEM MUMPS** you have:

- The capability of initiating any system or user process directly from MUMPS.
- The ability to communicate with COBOL, FORTRAN and TAL programs through the ENSCRIBE Data Base Manager for increased efficiency and optimum utilization of system resources. Through this facility users can read and write disc files created by any of these high-level languages, as well as other MUMPS processes.
- Access to the GUARDIAN Operating System facilities and resources, including the versatile TANDEM-supplied Spooler.
- The ability to utilize the ENFORM Query/Report Writer for fast and easy report generation.
- Access to the EXPAND Fault-Tolerant Communication Network capabilities for full utilization of distributed data bases.
- Access to the impressive text editing facility for program entry and maintenance.

Reliability

The reliability of a **TANDEM MUMPS** application is inherent in the basic architecture of the TANDEM 16 itself:

- TANDEM *NonStop* Computers are the only commercially available systems that are able to continue to operate despite the failure of a major component (i.e., CPU, controller, disc).
- Automatic *NonStop* operation of MUMPS routines can be selected with the use of the ZNONSTOP command.

Expandability

TANDEM provides systems that can grow with you. Modular expansion is easy with TANDEM Computers; there is no need for costly reprogramming. TANDEM provides the capability of dynamically expanding your system from 2 to 16 processors. Up to 255 system nodes can be joined in an EXPAND Network — all without additional costs for applications software reprogramming. With proper security, data can be accessed from any node in a network without special programming.

TANDEM MUMPS Applications

MUMPS is a specialized computer language that is intended for applications that emphasize interactive data entry and inspection, manipulation of text and encoded data, and data sharing. MUMPS is most useful and efficient when used for the following applications and conditions:

- When rapid program development is a requirement;
- For dynamic, evolving applications;
- For applications handling data that map into hierarchical, sparse data structures;
- For text manipulation;
- For applications involving interactive data base query and update operations;
- For applications involving extensive error checking of user-entered data.

TANDEM

TANDEM COMPUTERS INCORPORATED, 19333 Vallco Parkway, Cupertino, CA 95014. Toll Free (800) 538-9360 or (408) 996-6000 in California. Offices throughout the United States, Canada, Europe and the United Kingdom. Distributors in Australia, Finland, Mexico and Venezuela.

TANDEM MUMPS

MUMPS, an acronym for Massachusetts General Hospital Utility Multi-Programming System, is a high-level, interpreted computer language designed for easy manipulation of string data and of hierarchically-structured data bases.

MUMPS is used extensively in commercial and medical applications because of its richness of features, ease of learning, ease of use, and superior string manipulation capabilities.

Tandem MUMPS differs from most current implementations in that it is not running under its own operating system, rather it runs under the Tandem Guardian operating system. As a result, users are able to execute COBOL, FORTRAN or TAL programs concurrently. And of course MUMPS has the benefits of running on the Tandem NonStop™ system so that applications can stay up and running even if a processor, disc, disc controller or I/O channel should fail during operation.

Standard MUMPS

Tandem Standard MUMPS supports all features of the ANSI 1977 MUMPS Language standard.

Tandem MUMPS Extensions

Tandem supplies an interface to COBOL, FORTRAN, and T/TAL programs, the reliability of NonStop computing, and the following extensions:

- Negative and decimal number subscripting, which lets you subscript variables with negative numbers (PERS(-12)) and non-integer numbers (MATR(23.75), MATR(-25.67)).
- The \$ORDER function proposed by the MUMPS Development Committee, which returns all subscript values including non-numeric strings, \$ORDER is supplied in addition to the \$NEXT function.
- The \$ZINSPECT and \$ZRINSPECT functions, which perform a "scan until" capability from the front and back, respectively, of a string. These capabilities speed and simplify the stripping of leading zeros and trailing blanks.
- The \$ZFINFO function, which returns information on all Tandem files, all of which can be accessed from MUMPS.
- The ZTRAP command, the \$ZTRAP function, and \$ZTRAP special variable, which provide a complete error trapping facility that can be tailored to individual applications.
- Commands and utilities that create, edit, and delete application and library routines.
- Commands that give access to Tandem environmental information.
- READ and WRITE commands which provide the capability of accessing sequential and key-sequenced files with record sizes up to 4KB.
- Extensions to the MUMPS READ and WRITE commands, which provide interprocess communication between MUMPS, COBOL, FORTRAN, TAL and other MUMPS processes.
- String subscripting, which lets you give variables meaningful names such as NAME("LAST") rather than the cryptic numeric names such as NAME(4). String subscripts can be up to 254 characters long.

The System MUMPS

Tandem MUMPS runs under the Tandem GUARDIAN operating system with full system resources at its disposal. With Tandem MUMPS you have:

- Access to the power of the Tandem Text Editor for code entry and maintenance.
- The capability of initiating any system or user process directly from MUMPS.
- The capability of communicating with COBOL, FORTRAN, and T/TAL as well as with other MUMPS processes.
- The capability of reading and writing disc files created by COBOL, FORTRAN and T/TAL programs through Tandem's ENSCRIBE data base manager.
- Access to the Tandem GUARDIAN/EXPAND communications network capabilities.
- Access to the GUARDIAN operating system facilities and resources, including the versatile Tandem spooler.

Reliability

The reliability of a Tandem MUMPS application is derived from the architecture of the Tandem 16 itself:

- The Tandem 16 is immune to the failure of any single component.
- The Tandem software, including the MUMPS interpreter, complements the NonStop™ environment.
- The hardware and software NonStop™ functions are transparent to the MUMPS application programmer and user.

Expandability

The Tandem system provides the capability of dynamically expanding your system from 2 to 16 processors. It also provides the capability of dynamically expanding into a Tandem network.

MUMPS Applications

MUMPS is a specialized computer language that is most useful and efficient when used for the following applications and conditions:

- When rapid program development is a requirement.
- For dynamic, evolving applications.
- For applications handling data that map into hierarchical, sparse files.
- For text manipulation.
- For applications involving interactive data base query and update operations.
- For applications involving extensive error checking of user-entered data.

TANDEM

TANDEM COMPUTERS, 19333 Vallco Parkway, Cupertino, CA 95014
—800-538-9360—

TANDEM

DATA DEFINITION LANGUAGE

FEATURES

- Centralized Data Base Provides Common Resource for Numerous Application Programs, Ensures Data Consistency, Reduces Redundant Data, and Allows Easy Maintainability without Sacrificing System Security
- Automatic Space Management Allocates and Optimizes Storage Resources Enhancing File and Record Activities
- Data Definition Language (DDL) to Facilitate the Definition of a Centralized Data Base Schema
- Easy-to-Use EDIT and SCHEMA Programs to Facilitate Compilation of Application Programs with High-Level Language TAL Compiler and ENSCRIBE Data Base Record Manager Procedures

INTRODUCTION

Tandem's *Data Definition Language* allows centralized administration of a data base to accommodate any number of diversified application programs. And the applications programmer need not know where files associated with a specific application are located. Since the data base is described as a *schema*, and all data field references are defined in the schema, changes to record layout, and additions and/or changes to record types and/or fields may be accomplished *without* code modifications to existing programs. The schema also provides the necessary information for query and reporting programs. The *Data Definition Language* (DDL) is used to describe the schema.

DATA BASE DEFINITION

As illustrated on the next page, the programmer defines a data base in a *Schema*

Definition File using the Tandem-supplied EDIT program. This example is used to define records for a customer data base. The *name* assigned to the record type is CUSTOMER. The fields are named ACCTNO, NAME, FIRST, STREET, CITY, STATE, ZIP, BAL. CHARACTER means that the field contains a string of ASCII characters and the number defines the length of each field in bytes. BINARY indicates arithmetic data. A O following KEYTAG means that the field ACCTNO is the record's primary key and "NM" means that the field NAME is an alternate key (other fields are not defined as key fields).

APPLICATION PROGRAM COMPILATION

The *Schema Definition File* is used as the input to the Tandem-supplied SCHEMA program. This program generates a T/TAL library file that, when compiled along with the application program, produces an object application program that is tailored to the particular data base. (See example on the next page).

DATA BASE ACCESS

Record types are accessed through ENSCRIBE provided with the GUARDIAN Operating System. Fields in a record type are accessed by *field identifiers*. As shown in the examples, a field identifier is a concatenation of the name of the record and the name of the field. For example, the field identifiers for the CUSTOMER record defined in the example are:

<field name>	<field identifier>
ACCTNO	CUSTOMER ^ ACCTNO
NAME	CUSTOMER ^ NAME
STATE	CUSTOMER ^ STATE

Each field defined as type CHARACTER also has a corresponding *field length identifier* that can be used to determine the length in bytes of a field. A *field length identifier* is the concatenation of the *record name*, the *field name*, and the string LEN. For example, the *field length identifier* for the NAME field is:

CUSTOMER^NAME^LEN

CUSTOMER^NAME^LEN has the value "25" when referenced in the program.

Each field which was defined to have a key tag will have a *key tag identifier* that can be used to pass the key tag to the file system. The form of this identifier is a concatenation of the *field identifier* and KEY. For example, the *key tag identifier* for the field NAME is:

CUSTOMER^NAME^KEY

CUSTOMER^NAME^KEY has the value of "null" when referenced in a program.

Defining the Data Base

```
* SCHEMA DEFINITION
*
* CONSISTS OF TWO TYPES OF RECORDS
* 1. CUSTOMER MASTER RECORD
* 2. TRANSACTION LOG RECORD
*
RECORD CUSTOMER.
03 ACCTNO: TYPE CHARACTER 5; KEYTAG 0.
03 NAME: TYPE CHARACTER 25; KEYTAG "NM".
03 FIRST: TYPE CHARACTER 25.
03 STREET: TYPE CHARACTER 56.
03 CITY: TYPE CHARACTER 25.
03 STATE: TYPE CHARACTER 25.
03 ZIP: TYPE CHARACTER 5.
03 BAL: TYPE BINARY 64,2.
*
RECORD TRANSACT.
03 ACCTNO: TYPE CHARACTER 5; KEYTAG 0.
03 NAME: TYPE CHARACTER 25; KEYTAG "NM".
03 DATE: TYPE CHARACTER 10; KEYTAG "DT".
05 YEAR: TYPE CHARACTER 2.
05 MONTH: TYPE CHARACTER 2.
05 DAY: TYPE CHARACTER 2.
03 CODE: TYPE CHARACTER 1.
    "+" = TRANSACTION IS DEPOSIT
    "-" = TRANSACTION IS WITHDRAWL
03 AMT: TYPE BINARY 64,2.
03 TIME: TYPE CHARACTER 5; KEYTAG "TM".
05 HOUR: TYPE CHARACTER 2.
05 MINUTE: TYPE CHARACTER 2.
```

Compiling the Schema

```
SCHEMA /IN DDLFILE, OUT SLP/ APPLIB
```

Example: Accessing the Data Base

```
Application Global Declarations
.
.
! Including the schema source in the application program
?SOURCE SCHEMA(CUSTOMER,TRANSACT)
ALLOCATE CUSTOMER;
ALLOCATE TRANSACT;
PROC USER APPLICATION;
.
! Using the Schema Definitions
CUSTOMER^ACCTNO := FORMOPEN^ACCT FOR CUSTOMER^ACCTNO^LEN;
CUSTOMER^NAME := FORMOPEN^NAME FOR CUSTOMER^NAME^LEN;
CUSTOMER^FIRST := FORMOPEN^FIRST FOR CUSTOMER^FIRST^LEN;
CUSTOMER^STREET := FORMOPEN^STREET FOR CUSTOMER^STREET^LEN;
CUSTOMER^CITY := FORMOPEN^CITY FOR CUSTOMER^CITY^LEN;
CUSTOMER^STATE := FORMOPEN^STATE FOR CUSTOMER^STATE^LEN;
CUSTOMER^ZIP := FORMOPEN^ZIP FOR CUSTOMER^ZIP^LEN;
.
! Insert a new Customer Record into the File
CALL WRITE(CUST,CUSTOMER,CUSTOMER^LEN,CNT);
.
.
! Position to Customer Record for Subsequent Update
CALL KEYPOSITION(CUST,FORMDEP^ACCT,,,EXACT);
IF < THEN BEGIN ...
.
.
! Get Customer Record and Lock It
CALL READUPDATELOCK(CUST,CUSTOMER,CUSTOMER^LEN,CNT);
IF < THEN BEGIN
CALL FILEINFO(CUST,FILE^ERR);
IF FILE^ERR=11
THEN BEGIN ! RECORD NOT FOUND
.
.
.
```