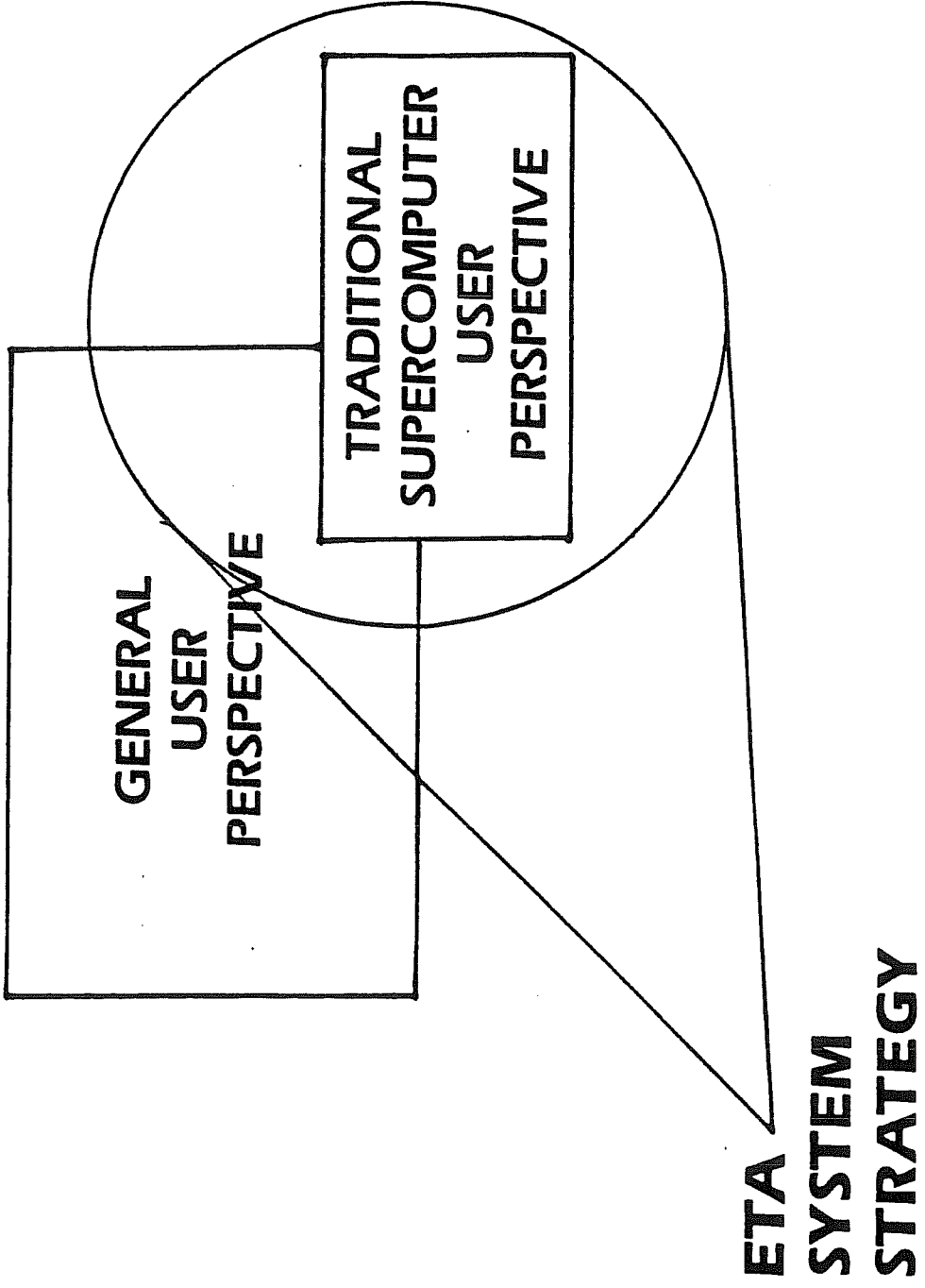


ETA SYSTEM STRATEGY

This slide addresses two markets that ETA supports:

- Traditional supercomputer users (bottom of slide)
 - Wrote own software and were close to the hardware so programs were specific to the architecture of that machine. The programs were not very portable.
 - Example: Livermore Lab.
- General users (top of slide)
 - Most users fall into this category.
 - Want to take program, port it to a machine, and have it run without making a lot of changes to the program.
 - Example: VAX, IBM, CYBER computers
 - The user community is moving toward this direction.

ETA wants to address both of these markets.



**GENERAL
USER
PERSPECTIVE**

**TRADITIONAL
SUPERCOMPUTER
USER
PERSPECTIVE**

**ETA
SYSTEM
STRATEGY**

ETA¹⁰ SOFTWARE FEATURES

Multiple User Interfaces

- Different ways for different users to access and use the ETA-10

High Performance Programming Tools

- Automatic vectorizing FORTRAN compiler
- Multitasking tools
- Debuggers

Network Connectivity

- ETA-10 is designed to be another node on the network.
- ETA-10 may be used interactively and/or as a back-end batch machine.

Flexible System Configuration

- System configurations are available to the site with multiple processors and various ways to use the machine
- The site may create several combinations of these operating methods to create a customized supercomputing environment.

Reliability

- Several reliability techniques are used by both the hardware and software. These will be discussed later in the presentation.

ETA¹⁰ SOFTWARE FEATURES

- **Multiple User Interfaces**
- **High Performance Programming Tools**
- **Network Connectivity**
- **Flexible System Configurations**
- **Reliability**

ETA USER ENVIRONMENTS

User Environments:

- The way the user accesses or sees the ETA-10
- Both System V (AT&T's UNIX) and VSOS provide a working context for command processing, etc.
- There is a set of commands, called session commands, which are common between UNIX System V and VSOS.
 - These commands allow users to submit and track jobs in either environment.
 - UNIX users may submit batch jobs to VSOS and vice versa.
- The user does not switch dynamically between user environments.
 - When you are using UNIX, you typically don't see VSOS and vice versa.
 - Files created under one environment may be used in another due to the common file system shared by both.
- A user will have a default user environment declared when assigned an account on the ETA-10.
 - This may be over-ridden at LOGIN time if the user has the privilege to use another environment.

ETA USER ENVIRONMENTS

SYSTEM V

VSOS

Working Context for:

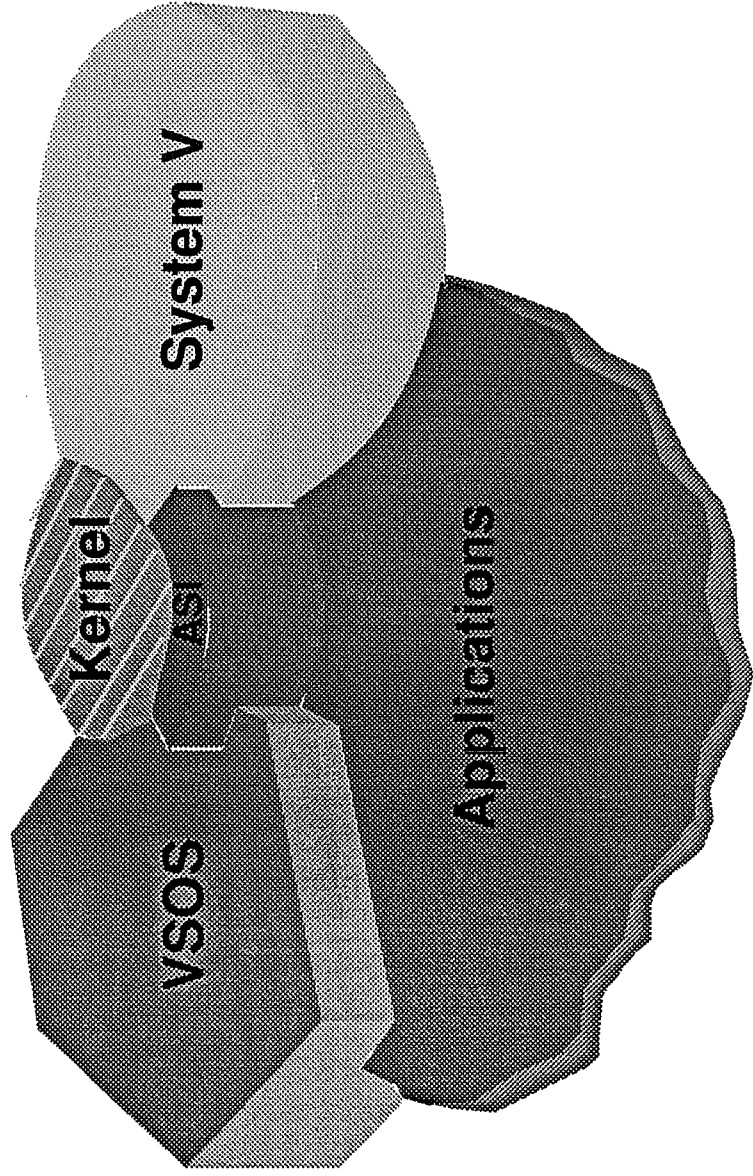
- **Command Processing**
- **Libraries, Utilities, and Tools**
- **Language Processors**

ETA¹⁰ SOFTWARE ARCHITECTURE

This slide shows some important features of the ETA-10.

- UNIX and VSOS can be run at the same time, but they are independent of each other. (You can run UNIX without running VSOS or vice versa.)
- In either case, both VSOS and UNIX System V are supported by a common underlying kernel.
- The kernel is the traditional underlying operating system in the computer.
 - Supports both UNIX and VSOS
 - Provides file system capabilities, memory management, scheduling, accounting, user validation on the system.
- Application Support Interface
 - This is the mechanism that allows the users from both UNIX and VSOS to make operating system (kernel) calls.
 - If we change or update the operating system (kernel), the users will not see that because it will be reflected in the application support interface.
- Applications:
 - Designed to be run either from VSOS or UNIX
 - Can take existing applications from the CYBER 205, recompile them on the ETA-10.
 - Can take those existing applications that run under UNIX today, C programs, and recompile them on the ETA-10 also.
 - ETA has a large suite of applications ready to run on the ETA-10.
- The kernel itself has no user interface. This is provided by the user environment.

ETA SOFTWARE ARCHITECTURE



ETA SYSTEM V FEATURES

- Bourne
 - Comes standard with AT&T's System V UNIX
 - Shells are the command interpreters for UNIX. They are very powerful procedural facilities.
- C Shell
 - Berkeley bsd4.2 shell
 - It resembles the "C" programming language, hence the name C Shell.
- Berkeley Networking Extensions
 - The socket manager from Berkeley UNIX provides a standard calling interface to network functions.
 - It allows us to support routines like FTP which lets users transfer programs from one UNIX machine to another UNIX machine.
- FTP will also run under VSOS.

ETA SYSTEM V FEATURES

- **Derived from AT&T's UNIX[®] System V Release 2 Product**
- **Bourne & C Shells**
- **Berkeley Networking Extensions**
- **File Transport Protocol (FTP)**
- **ETA¹⁰ Enhancements**

ETA SYSTEM V ENHANCEMENTS

NOTE:

Do not use this slide unless your audience is interested in an in-depth UNIX discussion.

UNIX was designed on a PDP 7 Computer (Digital Equipment Corp.)

- This computer had a single CPU, single memory, and simple hardware.
- ETA-10:
 - Multiple CPUs, multiple memories, high performance distributed I/O, and the ability to handle thousands of processes per CPU.
- ETA Systems had to make changes to UNIX in order to run it on the ETA-10.
 - These changes are in the kernel.
 - Kernel-based implementation that supports the architecture of the ETA-10
 - ETA added security controls to UNIX as an option at the site when the system is set up so that users cannot see other users' files, cannot issue certain commands, etc.
 - Many of those controls come in the area of the super user.
 - Super user rights have been distributed so the site may create many super users, each with specific privileges.

ETA SYSTEM V ENHANCEMENTS

- **Multiple CPUs**
- **Memory Hierarchy**
- **High Performance Distributed I/O**
- **Thousands of Processes**
- **Security Controls**

ETA VSOS FEATURES

VSOS:

- Can be used interactively or batch
- Typically, VSOS is used in a batch environment with a front-end computer. We are enhancing the interactive capabilities of VSOS.
- Easy to learn operating system because it is simple and straightforward.
- Multi-host Network:
 - Loosely Coupled Network, Ethernet, and HYPERchannel
- FORTRAN 200:
 - Currently runs on CYBER 205
- ETA FORTRAN
 - Superset of FTN 200

ETA VSOS FEATURES

- **Interactive and Batch Access**
- **Easy to Learn**
- **Mature Environment**
- **Multi-host Network Support**
- **FTN200 and ETA FORTRAN**

ETA VSOS ENHANCEMENTS

NOTE:

Use this slide only with audiences familiar with CYBER 205.

- Enhancements made to VSOS are largely the result of the rewritten kernel (underlying operating system on ETA-10), which allowed VSOS to take advantage of a number of capabilities in the kernel.
- ETA VSOS is based on VSOS 2.2 from Control Data.
- ETA VSOS supports multiple sessions per user as well as the ability to share files between users. (This is something you could not do with the CYBER 205 and results from the newly created file system on the ETA-10.)
- The user has the option to retain file position across processes.
- Refers to disks by device classes instead of by pack names. (see Devices slide)

ETA VSOS ENHANCEMENTS

- **Based on VSOS 2.2**
- **Multiple Sessions per User**
- **Shared Files**
- **File Position Retained across Processes**
- **Device Classes vs Pack Names**

KERNEL

3 basic components on the ETA-10 kernel: resource managers, system overseers, system connectors.

Resource managers

- Manage a particular function or piece of hardware
- Example: Shared Memory Manager
 - Controls all access to shared memory on ETA-10.
- Example: Process Management
 - Handles the creation and termination of processes.

System Overseers

- Make sure system runs as the site sets it up to run
- Example: Global Scheduler
 - Looks like a single global scheduler on the machine but there is a scheduler running on each CPU. (This is also true for the resource managers.)
- Example: System Monitor
 - Runs primarily in the service unit but also runs partly in each CPU
 - Communicates with CPU to maintain integrity of the system.
 - If a processor were to go down, system monitor in service unit would detect that fact.
 - It would notify the global scheduler not to schedule jobs in the CPU that was down.
 - It would also demote the CPU from the service unit.
- Example: System Configuration Control
 - Keeps track of the hardware and software configuration of the ETA-10

System Connectors

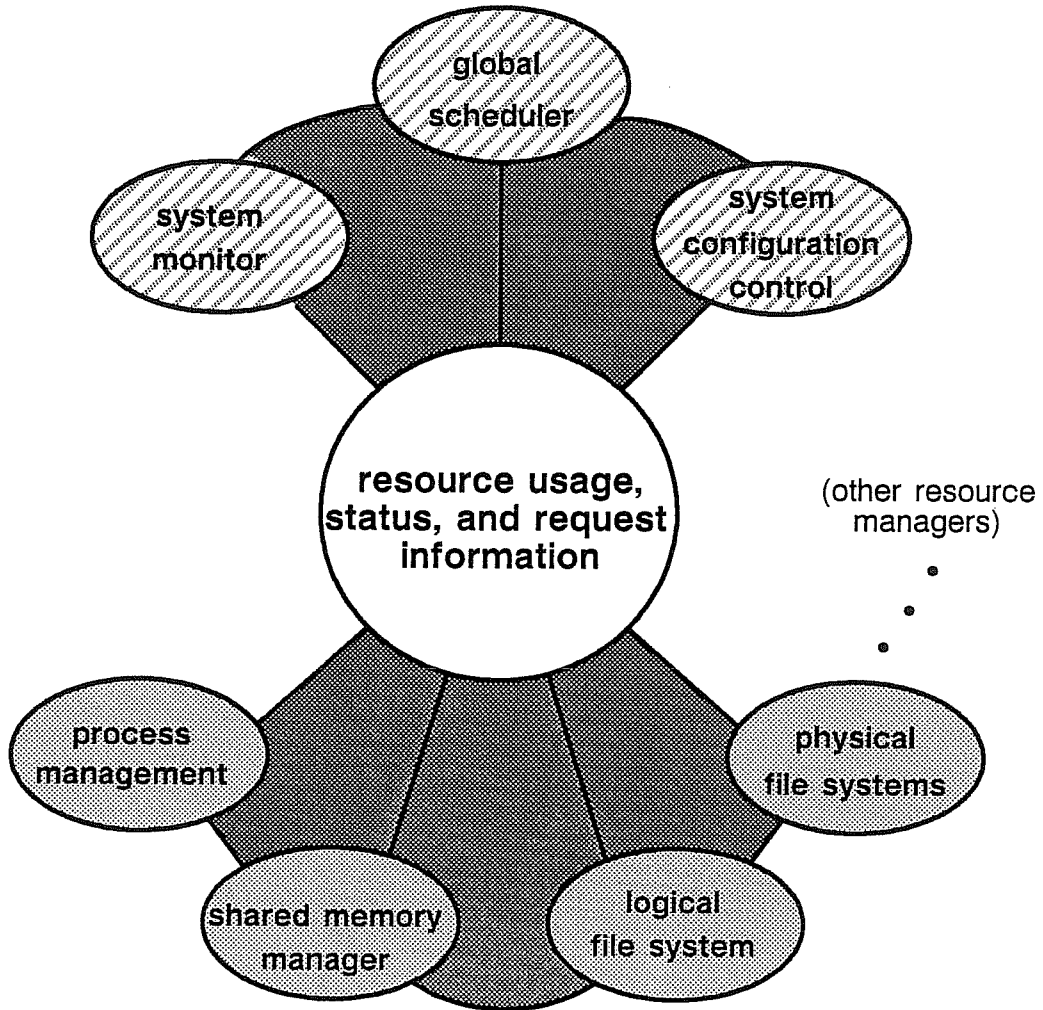
- Allow different resource managers and system overseers to communicate with one another, using standard calling procedures.
- Done through interprocess communication facility which uses facilities in the Communication Buffer of the ETA-10.

The reason different resource managers and system overseers are in separate boxes or components is because that is the way they are implemented by the operating system. (Some people call them domains)

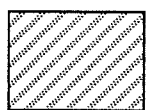
- Example: Physical file system actually runs on the hardware in the IOU.
 - Logical file system runs in the CPU and must be able to communicate with the physical file system.
 - That is done through the Communications Buffer using the interprocess communications facility.


Different resource managers reside or constitute their own domains in the operating system. These domains have protection capabilities within them (Example: The Shared Memory manager is the only domain in the operating system that can issue Shared Memory instructions on the ETA-10.)

KERNEL



 Resource Managers

 System Overseers

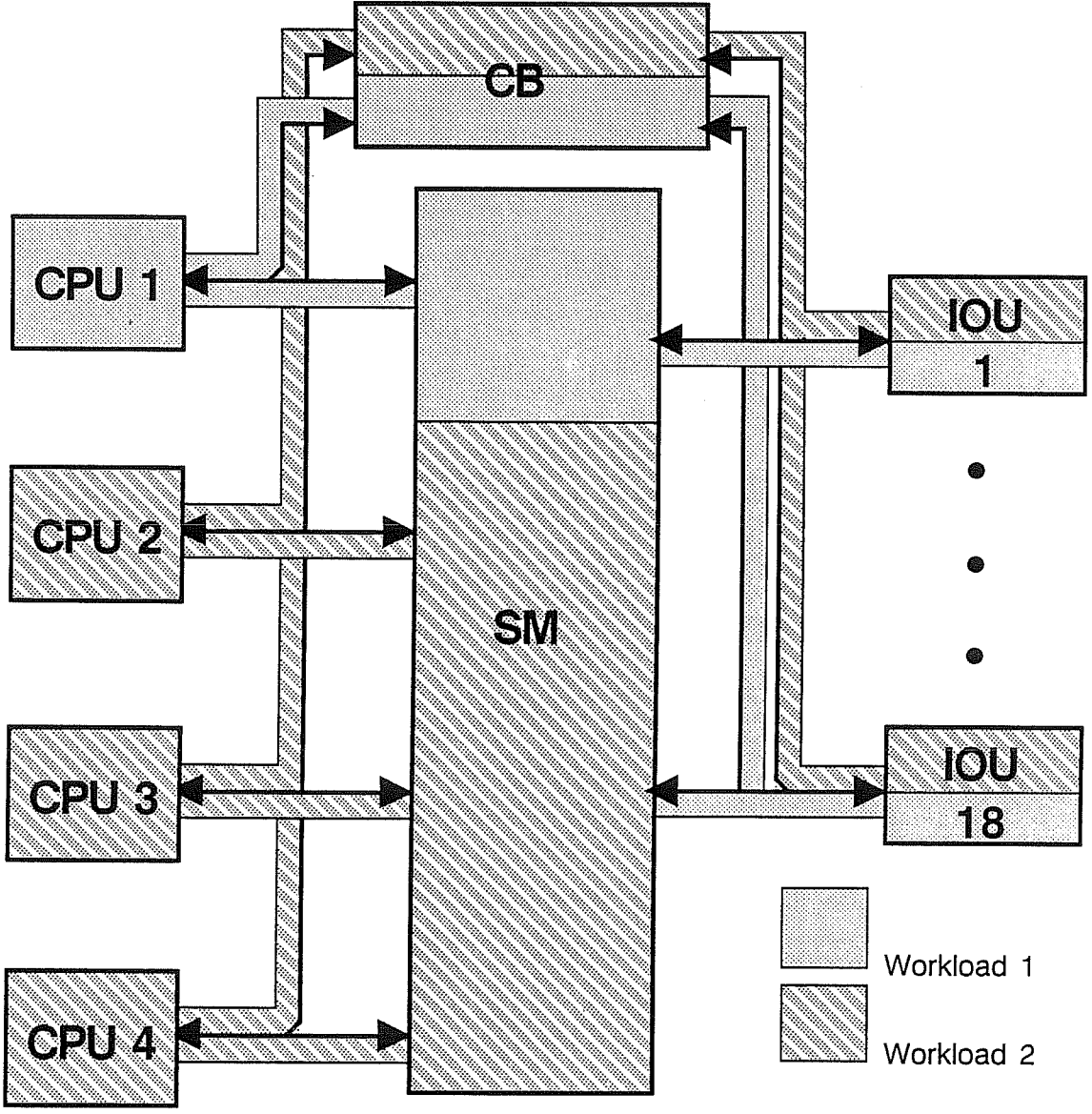
 System Connectors

ETA¹⁰ SYSTEM PARTITIONING

This slide illustrates that on the ETA-10 you can:

- Have all CPUs run as one large CPU
- Divide up the resources and have some of the CPUs run one type of workload and others run another type
- These choices can be changed automatically and dynamically by the operator on the site based on time of day or day of the week.
 - Example: At 7:00 p.m. the CPU may become part of a partition which runs large batch jobs, and 8:00 a.m., it can be changed to run interactively.
- Workloads consist of:
 - Session categories defined by the site to classify groups of users based on system resources to which they have access.
 - This may include the right to access tape drives, high performance disk devices, user environments, large pages, Shared Memory, etc.
 - The user may specify a session category; otherwise, he will run under his default category.
 - Users must be validated for each session category they wish to run under.
- CPU partitions:
 - Groups of CPUs, set up by the site, that have been tuned for a specific class of sessions.
- Tuning parameters
 - Include page size, interactive or batch access, scheduling algorithms, UNIX or VSOS, and development or production runs.
- Sessions are assigned to CPU partitions at run time based on the CPU partitions available.
- A list of allowed session categories is associated with each CPU partition.

ETA¹⁰ SYSTEM PARTITIONING



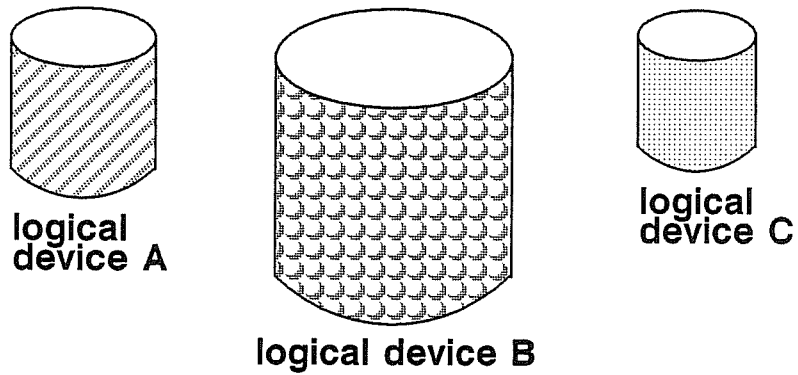
USER'S LOGICAL DEVICES

This slide shows how users access disk drives.

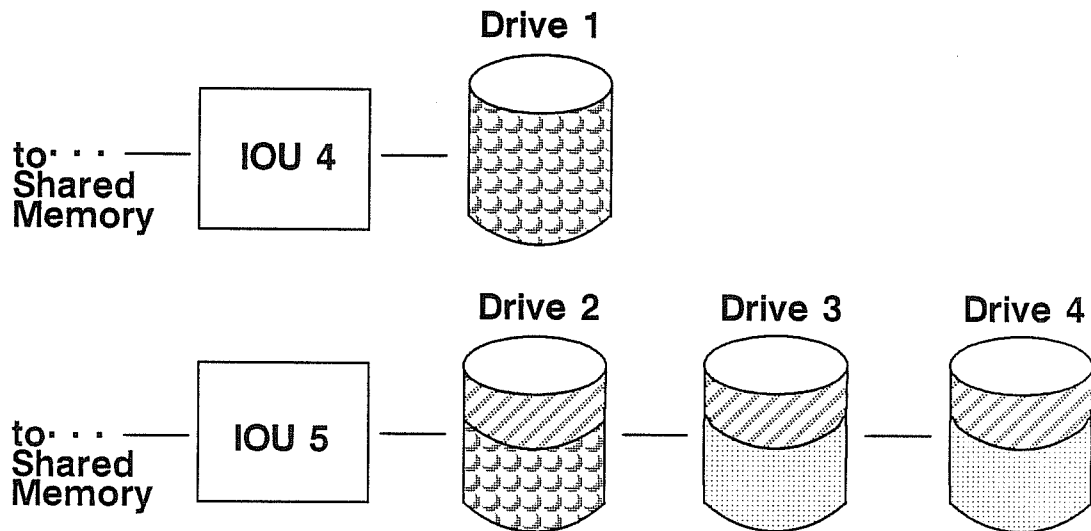
- Example:
 - The top portion of the slide shows that logical device B (blue) looks like a large disk drive to the user.
 - The bottom portion shows that logical device B is actually more than one physical disk drive, all connected together by the operating system.
- Example:
 - The top portion of the slide shows that logical device A looks like a fast disk drive to the user.
 - The bottom portion shows that it is actually a striped or high performance disk which is managed by the operating system that splits the user's file across multiple disk drives and writes each block onto a different disk drive at the same time.
 - This is done by the operating system and can be done at the fourth level or four disk drives at the same time.

The reason we can have up to four disk drives at one time is because each IOU can have up to four disk channel controllers.

USER'S LOGICAL DEVICES



SYSTEM'S PHYSICAL DEVICES



DATA MOVEMENT

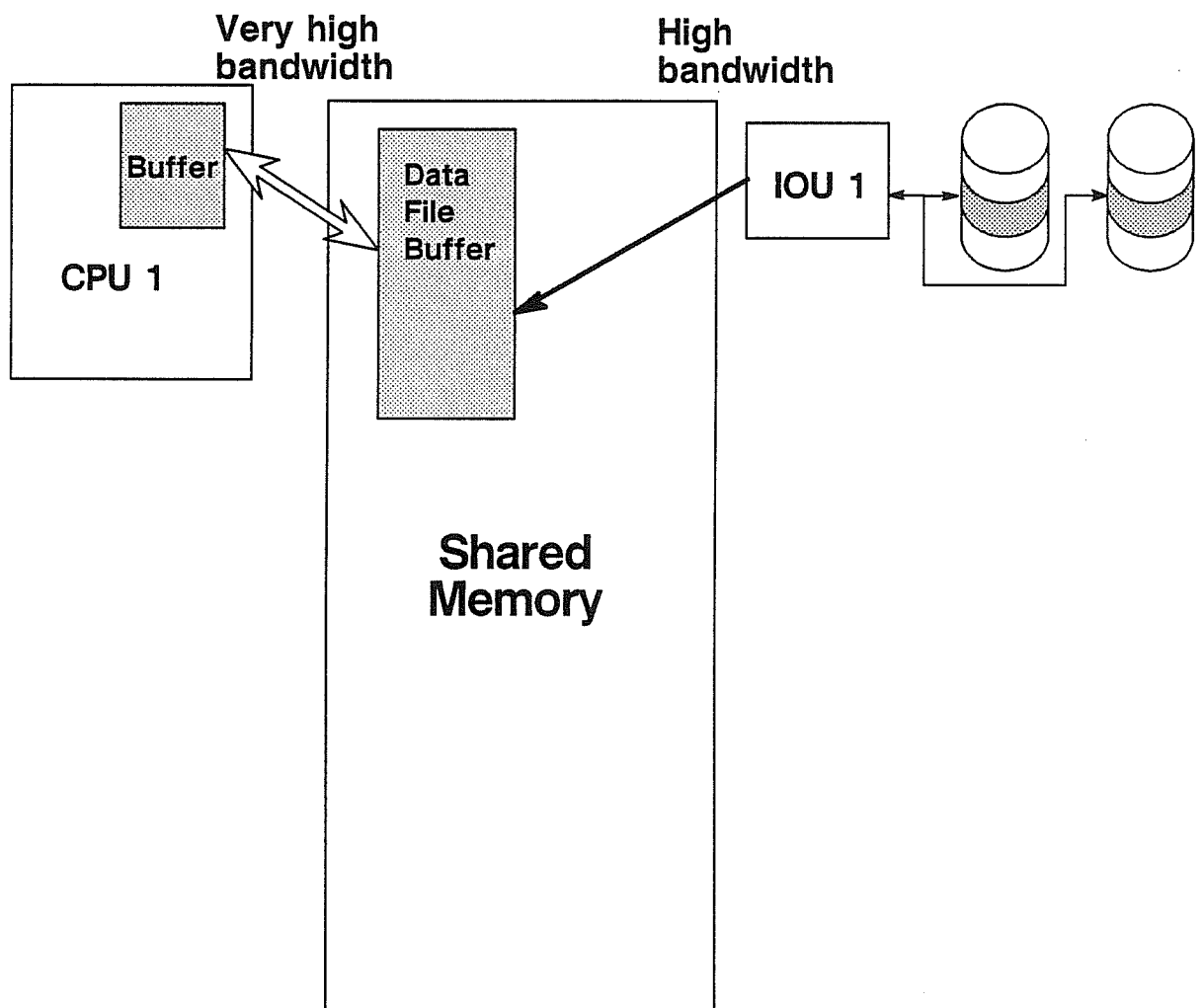
This slide shows how data moves through the ETA-10 system.

- Data moves from the outside world, such as disks, tapes or networks, through the IOU into Shared Memory.
- Shared Memory has the capacity to handle all 18 IOUs at the full rate, simultaneously.
- While that is occurring, we can also move data from Shared Memory to all CPUs at its full rate.
- The wider arrow on the left shows that this transfer is occurring much faster than the transfer of data over the fiber optics link.

This slide also shows the asynchronous input/output in the ETA-10.

- While you are moving data from the IOU to Shared Memory, you're moving data from Shared Memory to the CPU at the same time the CPU is executing other instructions.
- The reason for this is there are different hardware components that control access from the IOU and Shared Memory and from Shared Memory to the CPU.

DATA MOVEMENT



MULTI-ENVIRONMENT PRODUCTS

This slide shows some products that will run in either environment on the ETA-10.

Three languages (ETA FORTRAN, Pascal, C) are common or compatible:

- They all have a common object text format.
- This means you can write a program partially in FORTRAN, partially in Pascal, and partially in C and link them all together and run them as a single program.
- We also support a common argument passing procedure within the different languages.

Network Interface

- Only applies to FTP
- Runs on both UNIX and VSOS on ETA-10

Multitasking Libraries

- Can be called from any of the three languages from either UNIX or VSOS.

MULTI-ENVIRONMENT PRODUCTS

- **ETA FORTRAN**
- **Pascal**
- **C**
- **Network Interfaces**
- **Multitasking Libraries**

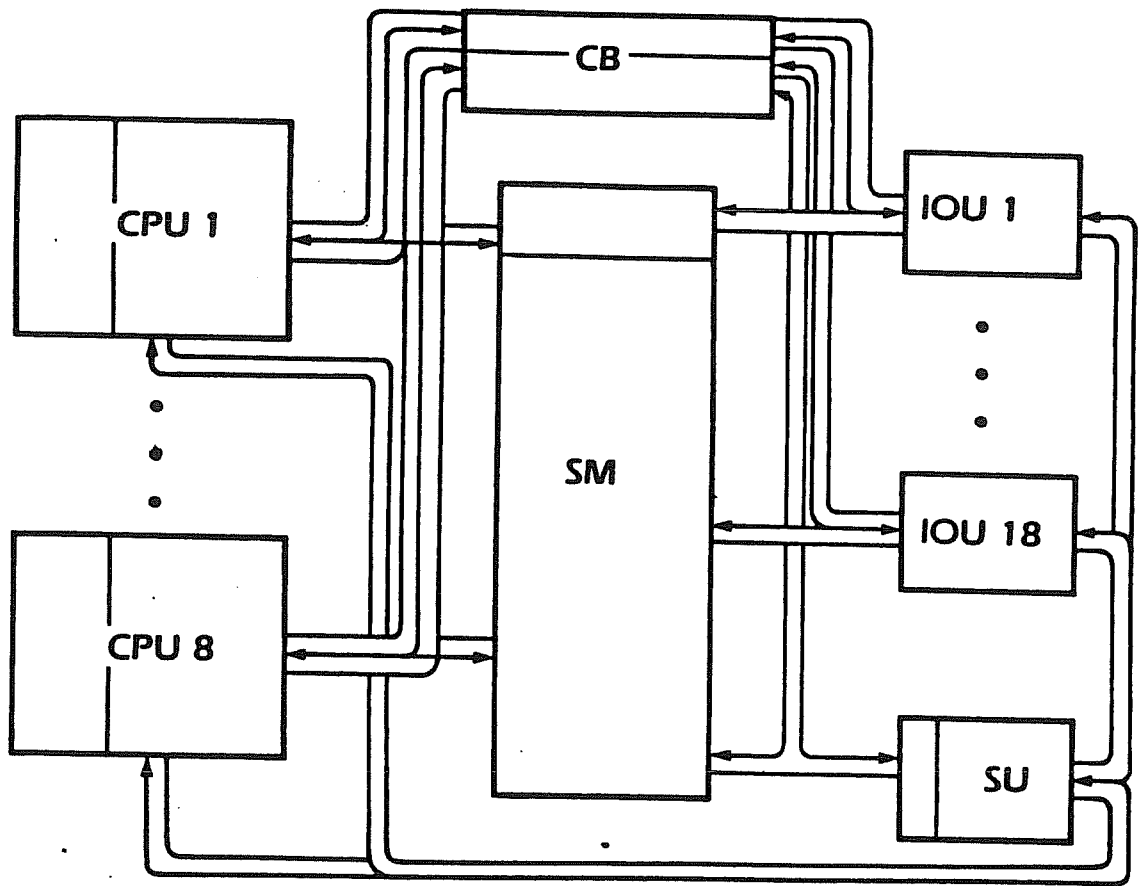
KERNELS/USER ENVIRONMENTS & APPLICATIONS

This slide shows the distribution of the operating system software on the different hardware components of the ETA-10.

- Green portion: kernel or operating system software
- Blue portion: user environment or application

The ETA-10 is a fully distributed operating system.

- Parts of the operating system run in the input/output unit, Service Unit, and CPUs.
- Tables are shared in Communication Buffer and Shared Memory across processors.
- Each processor has its own intact version of the operating system running.
 - There is no master/slave.
 - If one CPU goes down as detected by the Service Unit, the other CPUs may pick up the additional work.



□ KERNELS

□ USER ENVIRONMENTS
& APPLICATIONS

ETA FORTRAN

This slide discusses the new ETA FORTRAN compiler.

- It is based on the 77 ANSI standard and continues from there.
- ETA is adding the first cut at the 8x array notation from the new 8x FORTRAN standard (8x refers to FORTRAN 88 or FORTRAN 89, etc.).
- The VAST II vectorizer, from Pacific Sierra Research, is incorporated into the compiler and is automatically invoked if automatic vectorization was selected on the user's FORTRAN command line.
- ETA FORTRAN includes a number of user directives that gives the user control over the compilation of his program (some of which are Cray-compatible).
- Performance analysis is built in and is very similar to the CFT flow trace option.
 - It may be invoked per subroutine or by user directive anywhere in the program.
 - It reads the contents of the instrumentation counters (as part of the domain) which yield information on scalar, vector, memory and I/O usage.
- It is basically a superset of the FORTRAN 200 compiler that runs on the CYBER 205.

ETA FORTRAN

- **77 ANSI Standard**
- **Anticipated 8X Array Notation**
- **State-of-the-art Vectorizer**
- **User Directives**
- **Performance Analysis**
- **FORTRAN 200 Compatibility**

VECTOR PROCESSING

NOTE:

Use this slide only for audiences that don't know anything about vectorization.

SCALAR:

- Traditional FORTRAN DO-loop on the scalar machine
- Adding 500 elements from A to B and storing them in C
- Code generated by a typical scalar machine looks something like this:
 - Must load each element of A
 - Load B
 - Add A and B
 - Store the result in C
 - Update or index your counter (I)
 - Test it to see if you have added all 500 elements
 - Branch if you have more additions to perform
 - Net result: get one result (addition) for every several clock cycles.

VECTOR

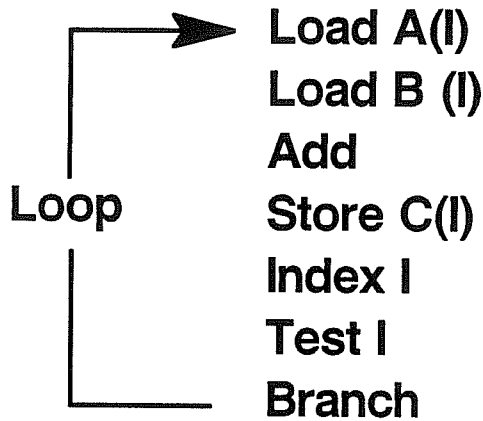
- Can write same DO loop using vector notation or we can generate vector instructions from the DO loop itself
 - In either case, the compiler generates a vector add instruction which adds A plus B and stores the result in C for 500 calculations.
 - Net result: You get up to 8 results per single clock cycle per CPU.

There is a big turnaround from scalar to vector.

VECTOR PROCESSING

SCALAR

DO 5 I = 1,500
5 C(I) = A(I) + B(I)



**1 Result / Multiple
Clocks**

VECTOR

$C(1;500)=A(1;500)+B(1;500)$

**Vector Add
A + B = C**

Multiple Results / Clock

AUTOMATIC VECTORIZATION

This slide discusses the automatic vectorization capabilities.

DO and IF Loops:

- Basically, the vectorizer is scanning DO and IF loops to see if it may vectorize part of the loop or the entire loop.

Scalar Promotion

- When a scalar element is added to a vector element, the scalar element is promoted to a temporary array so the operation can be vectorized .
- This occurs when the scalar element is modified within the loop. Otherwise the scalar element may be broadcast.

If Then Else Constructs

- Uses bit vector hardware on ETA-10.

User Feedback and Directives

- Allows user, vectorizer, and compiler to communicate with one another
- Compiler and vectorizer will tell you which parts of your code were vectorized and why they were vectorized.
- If some portions of your code could not be vectorized, it may ask you for more information about that piece of code and will tell you why it couldn't be vectorized.
- User can insert certain directives, giving the vectorizer more information about its code and answering any questions asked by the vectorizer.

Four Levels of Vectorization

- May be selected by the user
- Depends upon how much work you want the vectorizer to perform on your code.

AUTOMATIC VECTORIZATION

- **Do and If Loops**
- **Scalar Promotion**
- **If Then Else Constructs**
- **User Feedback and Directives**
- **Four Levels of Vectorization**

AUTOMATIC VECTORIZATION

Note:

Use this slide only if the audience is advanced in its knowledge of vectorization.

- Compiler has ability to do automatic strip mining.
- The compiler will generate multiple 64K temporary vector arrays if you have a vector greater than 64K.
- The compiler also allows the user to select the length of temporary vectors for memory and speed trade-offs.

Array Bounds Checking

- Performs array bound checking and round-off error control which is under the control of the user directives.

Nested Loop Collapse

- Example: Traversing through a multi-dimensional array.

Alternate Complex Storage

- Ability to store all reals in one array and all imaginaries in another array
- This helps you perform efficient vector calculations on complex numbers.

AUTOMATIC VECTORIZATION

- **Automatic Strip Mining**
- **Array Bounds Checking**
- **Round-off Error Control**
- **Nested Loop Collapse**
- **Alternate Complex Storage**

ETA¹⁰ SYMBOLIC DEBUGGER

- Symbolic debugger supports FORTRAN, Pascal, and C.
- This debugger is more advanced than the CYBER 205.
- It includes the ability to do both source and machine or object level debugging.
- It has built-in multitasking support.
- It has the ability to keep track of optimized and vectorized code that is generated by ETA FORTRAN.
- There are a number of facilities to manage windows and source text display for the user, either on a work station or on a normal ASCII terminal.
- The user may look at source code in one portion of the screen, debugger output in another portion of the screen, and set debugger parameters in another.

ETA¹⁰ SYMBOLIC DEBUGGER

- **Source and Machine Level Debug**
- **Multitasking Support**
- **Optimized and Vectorized Code**
- **Source Text Display**
- **Window Oriented User Interface**

ETA¹⁰ MULTITASKING

- ETA's approach to multitasking is unique because we are trying to provide a total solution environment for multitasking.
- ETA selected programming models that encourage correct programming.
 - We did not change the definition of FORTRAN or of any other language in order to support multitasking. It is all handled by routines that are part of the multitasking library.
 - We use a data private model. That is, all data is assumed to be private or local to a task unless specifically declared global. Data assumed to be global may cause difficult bugs when the programmer thinks it is local.
- ETA uses top-down approach.

Active Experimentation

- Experiments are running on the ETA-10 simulator which runs on the CYBER 205.

ETA¹⁰ MULTITASKING

- **Total Solution Environment**
- **Encourages Correct Programming**
- **Top Down Approach**
- **Active Experimentation**

ETA SYSTEMS' STRATEGY FOR MULTIPROCESSING

This slide discusses the different strategies and techniques that ETA will be using for multiitasking.

Top portion of the slide: The three boxes represent the programming level.

Three options available to user:

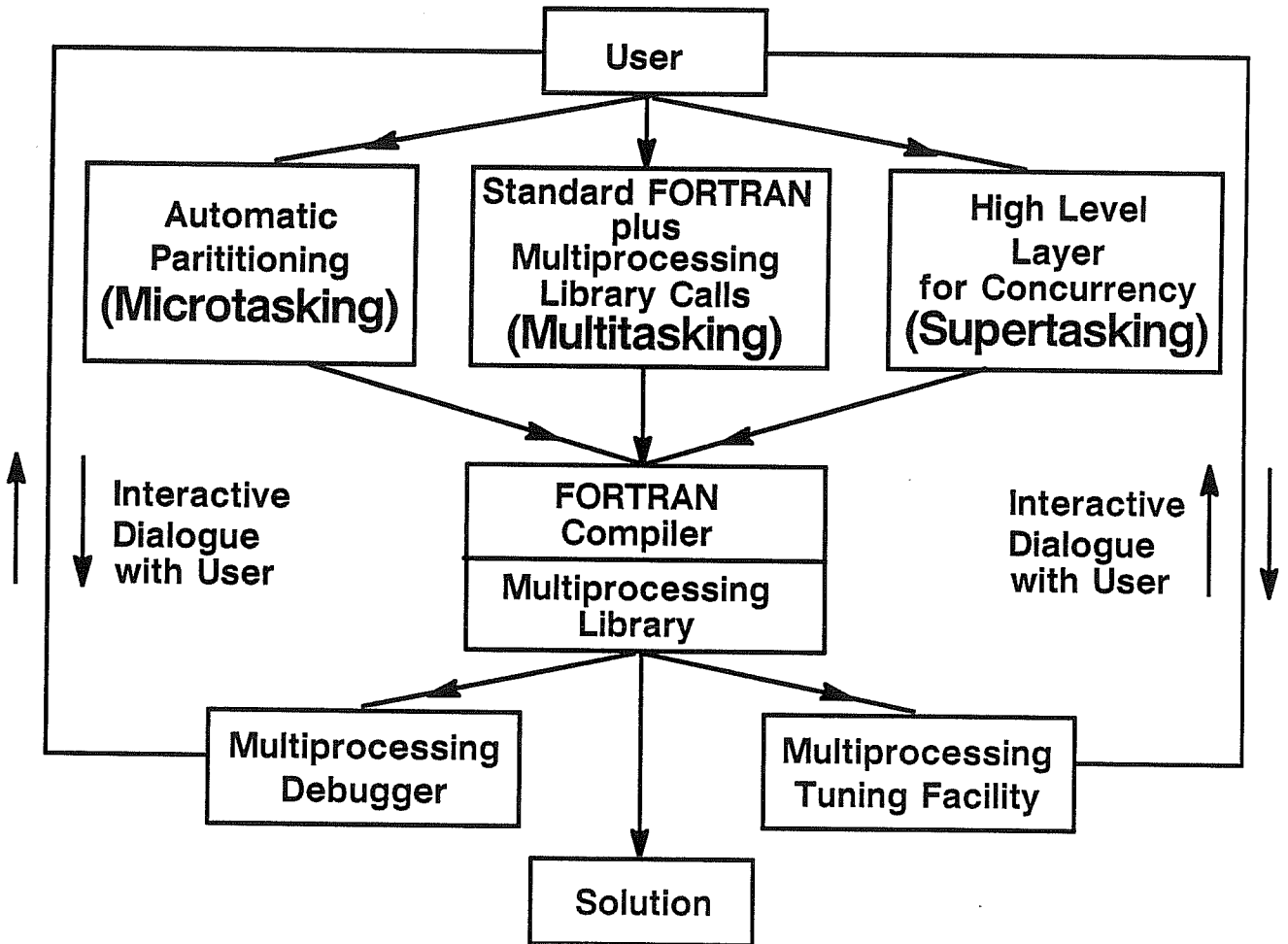
- Microtasking
 - Done automatically by the FORTRAN compiler with the aid of user directives
- Multitasking
 - User goes in and inserts calls to the multiprocessing library himself.
 - Standard FORTRAN calls to multitasking library.
 - User looks at code and says it is so big he can break it up into 2, 3, 4, etc. parts.
 - He can make calls to the multiprocessing library to divide up his code and start up different processes on different CPUs.
- Supertasking:
 - New wave approach to multitasking
 - Ability to have separate language level that understands scheduling.
 - FORTRAN doesn't understand scheduling of multiple CPUs so you can use a language (e.g. ADA) to do scheduling and to express the dependency between tasks in the scheduling language.
 - Can then use FORTRAN to do the actual calculations.

Whichever approach you use, it will always make calls to ETA's multiprocessing library.

Bottom portion of slide:

- Multiprocessing debugger is a workstation-based tool that allows the users to track different events using different windows on the workstation which are actually events that may be taking place in different processors on the machine.
- Users can look at different tasks according to different CPUs all on the same screen.
- Multiprocessing tuning facility is a workstation-based tool where the user can graph out (Gant chart or historical timing) to see what benefit he's gained from multiprocessing on the ETA-10.

ETA SYSTEMS' STRATEGY FOR MULTIPROCESSING



NETWORK OBJECTIVES

Minimal CPU Overhead

- Objective is to minimize the overhead on the CPU.
 - ETA offloaded about 80% of the software in the IOU.

Industry Standard

- Ethernet and TCP/IP

Multiple Connections

- ETA-10 has the ability to make multiple connections.

NETWORK OBJECTIVES

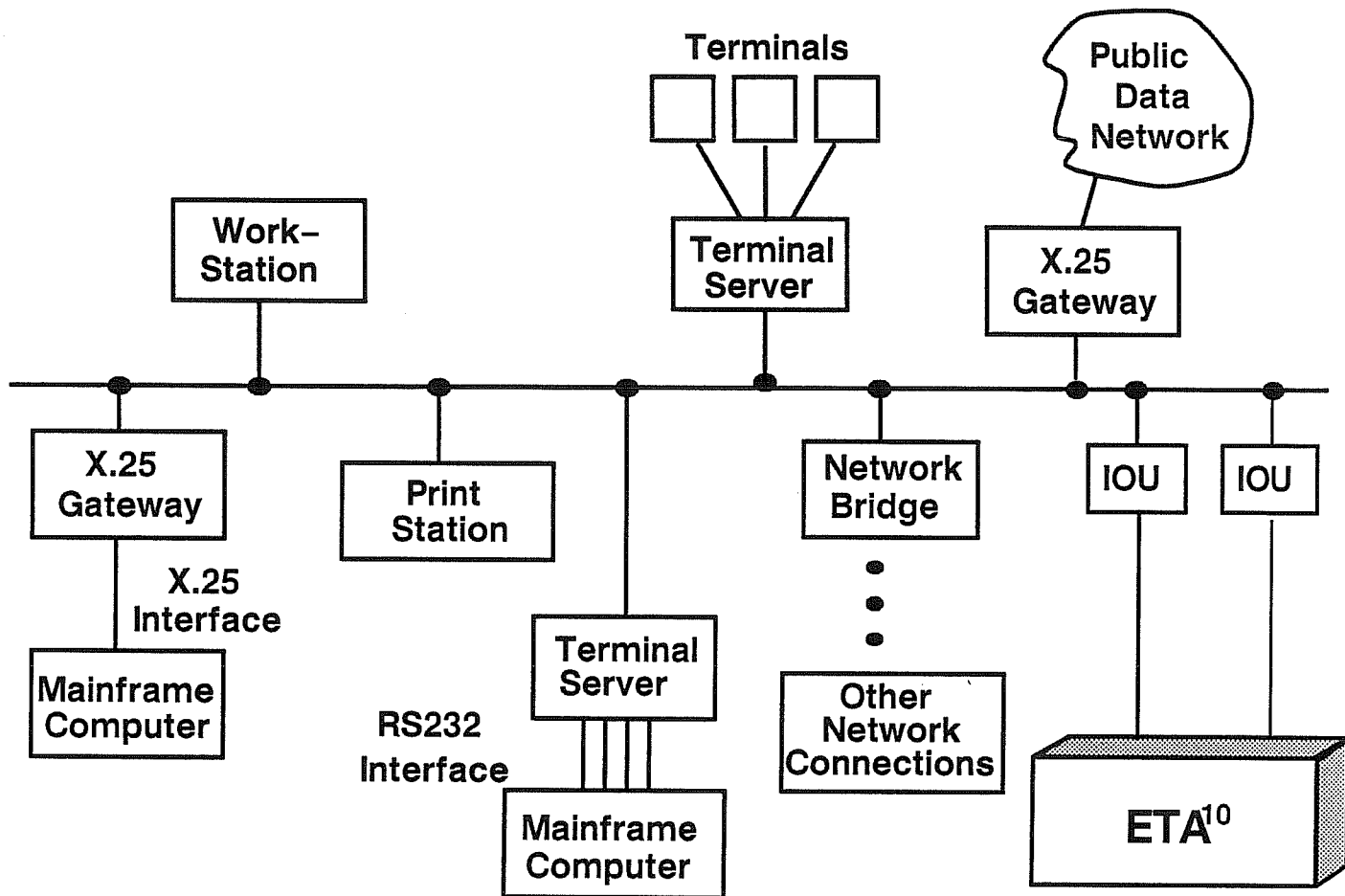
- **Minimal CPU Overhead**
- **Industry Standard**
- **Multiple Connections**

OPEN INTERCONNECTION NETWORK

This slide shows how you connect the ETA-10 to Ethernet network (Open Interconnection Network).

- White horizontal line is a coaxial cable.
 - Runs at 10 megabits/sec
 - Connects to Input/Output units on ETA-10 through VME interface
- IOUs can be up to 1000 ft away from ETA-10.
- Once you connect the ETA-10 to the Ethernet coax, it becomes a standard Ethernet network.
 - Allows you to connect terminals such as IBM PCs and VT-100 terminals via a terminal server to ETA-10.
 - The user makes a simple network connection and then logs into the machine normally.
- Supports X.25 gateways that allows you to access public data networks as well as multiple vendor networks
- Workstations have the ability to connect directly to Ethernet to make a direct connection to the ETA-10.
- Via bridges to other networks and/or to multiple Ethernet segments which follow the standard Ethernet rules.

OPEN INTERCONNECTION NETWORK



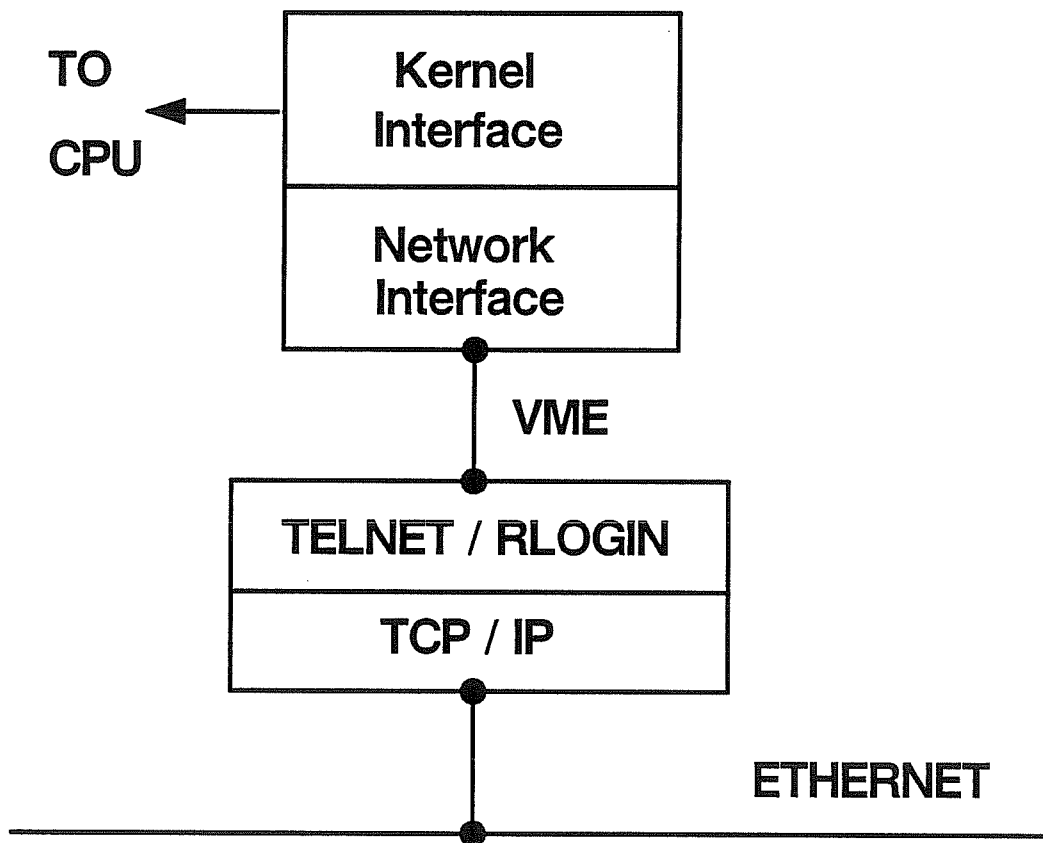
NETWORK ARCHITECTURE

This slide shows how we make the connection from the ETA-10 to the Ethernet network.

- Software runs on a single board computer inside the Input/Output unit.
- The kernel interface is the interface back to the kernel which runs on the CPU.
- Network Interface
 - Contains the drivers and transmission managers that allow you to connect to the VME Ethernet card.
 - The VME card runs the TCP/IP protocols as well as Telnet/RLogin and makes the connection to Ethernet peers.

NETWORK ARCHITECTURE

SINGLE BOARD COMPUTER (SBC)

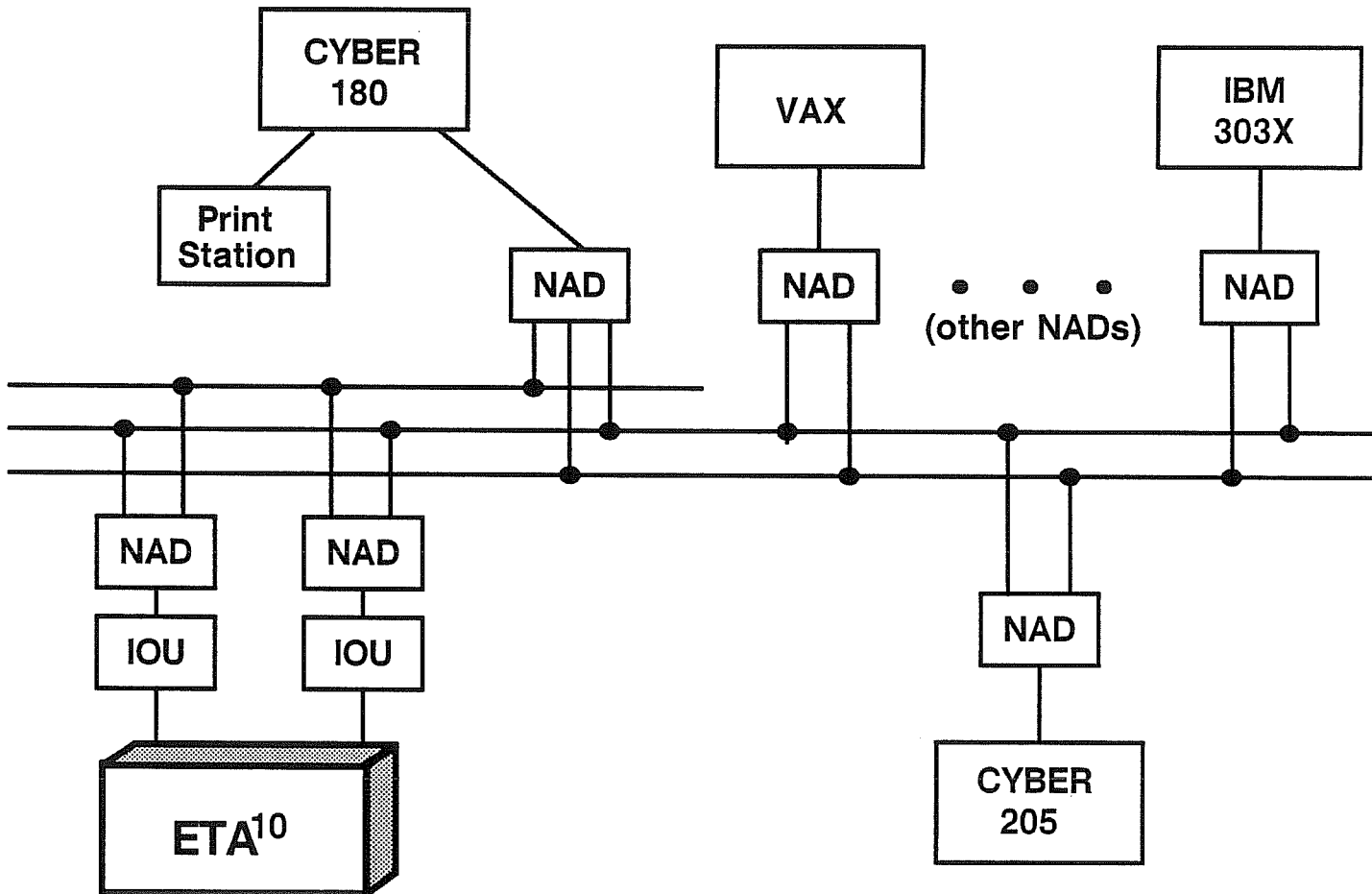


LOOSELY COUPLED NETWORK

This slide shows how you can connect the ETA-10 to a high speed network for host-to-host file transfers for large files.

- You may transfer files using Ethernet but it isn't very efficient.
 - Ethernet uses small packet sizes.
- ETA-10 is connected to the Loosely Coupled Network via IOU, which connects to NAD (network access device) on the FIPs channel (IBM channel) on the IOU.
- ETA-10 can connect up to four 50-megabit coaxial cables.
- RHF software is running on all computers (VAXs, IBMs, CYBERs) that allow users to transfer files or submit batch jobs from a VAX or other machine to the ETA-10 or transfer a file from one of these machines to an ETA-10.

LOOSELY COUPLED NETWORK



SOFTWARE RELIABILITY

Domains

- The main interprocess address protection mechanism in the machine is provided by the local address spaces and keys in the domains.

Capabilities

- Used by the operating system to control access to system resources.

Other Standard Techniques

- Redundant data means we have redundant hardware in the machine.
- Atomic actions in critical regions keep track of different actions or different pieces of software that are running on a multiple CPU system.

SOFTWARE RELIABILITY

- **Domains**
 - Intraprocess Address
- **Capabilities**
 - Controlled Access to System Resources
- **Other Standard Techniques**
 - Redundant Data
 - Atomic Actions
 - Critical Regions