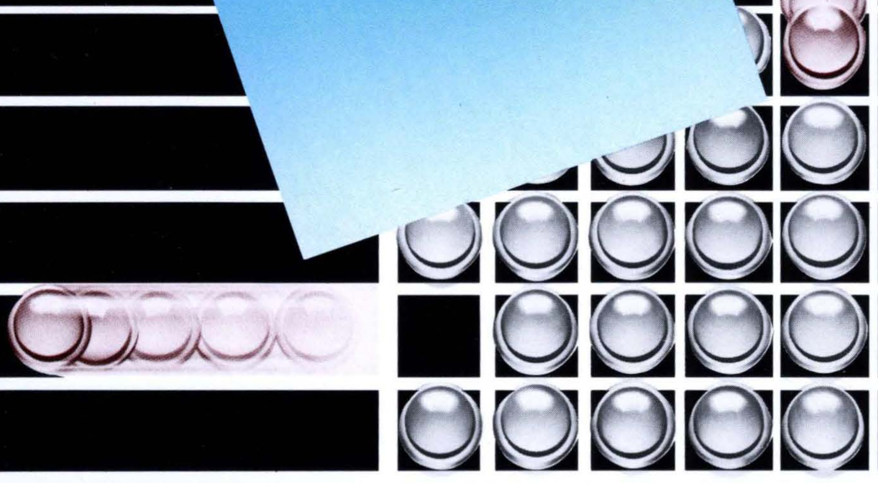
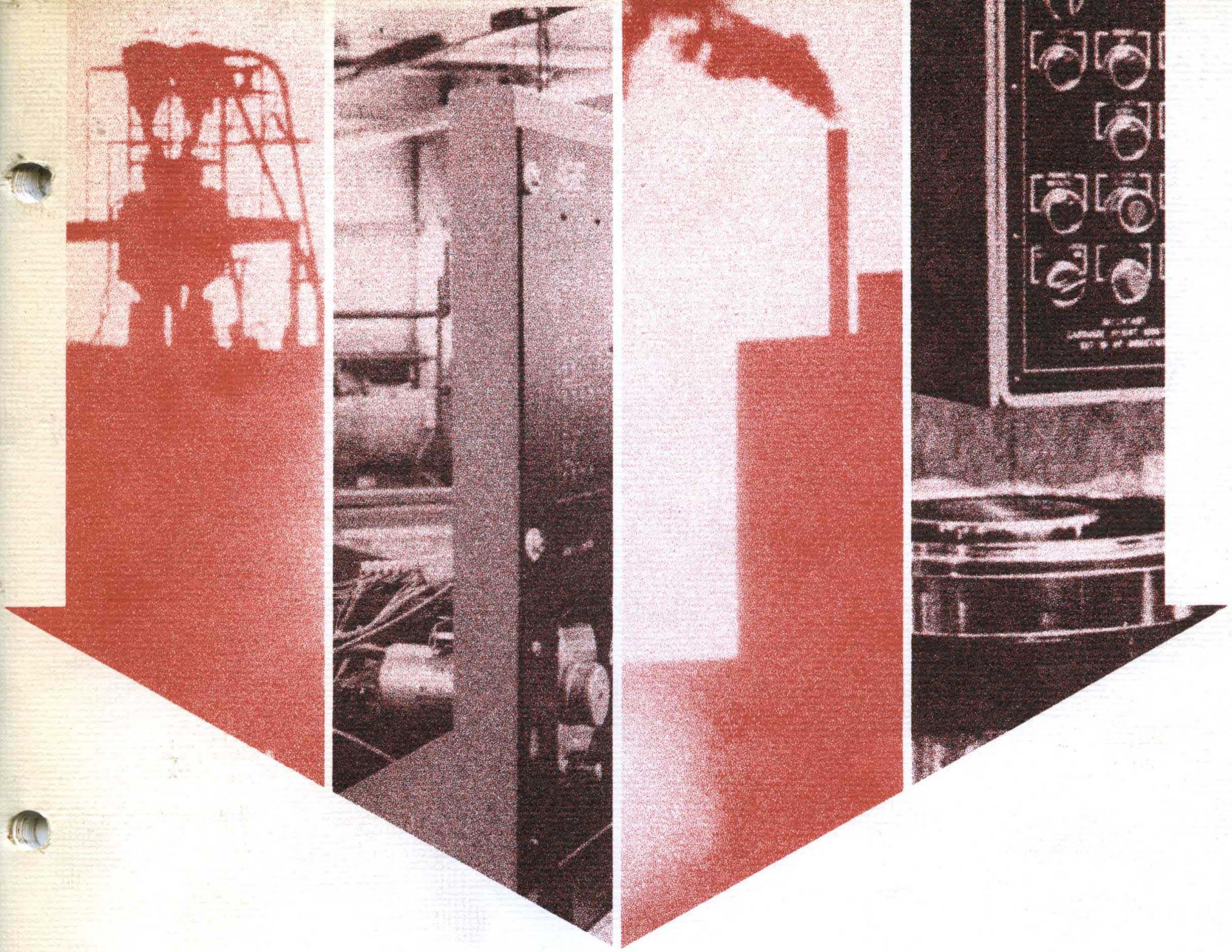
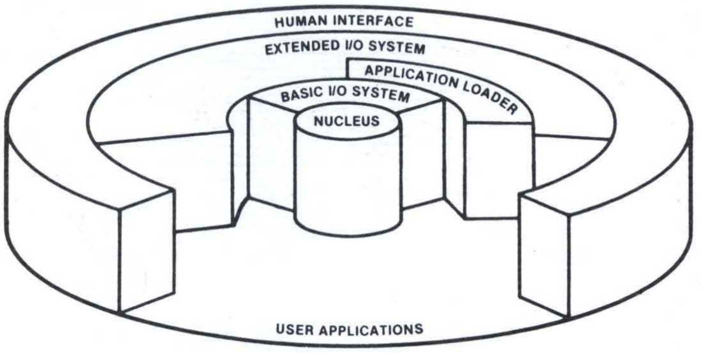
Intel

# iRMX™ OPERATING SYSTEM





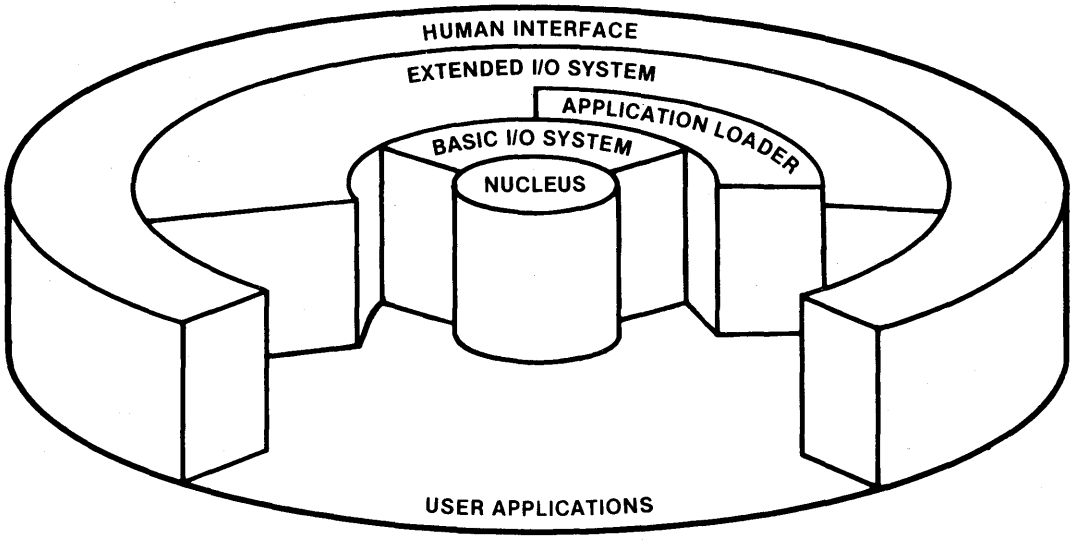


intel®

# OMS Field Sales Training Manual

## iRMX 86™ Operating System

# Table of Contents



**iRMX 86™ Operating System Layers**

# 1 Press Release

INTRODUCTION

The iRMX 86TM Operating System is a "real-time" modular operating system for the IntelR 8086 16-bit microprocessor. It extends the 8086 architecture, providing a structured, efficient environment for a wide range of applications including process control, intelligent terminals, office systems, medical electronics, and data communications.

The iRMX 86 software system provides OEMs with the most cost effective method yet devised to develop multiprogramming, multitasking systems that take advantage of the high-performance, megabyte capability of Intel's 8086. The iRMX 86 Operating System is the first to permit the development of simplified modular application coding. These systems help OEMs achieve large productivity gains by cutting labor costs when systems are first developed and later when new modules are added.

The iRMX 86 Operating System is easily customized. Not only is it divisable into powerful subsystems which can be included in an application, but each subsystem is also user-configurable and user-extensible. Four major benefits are derived from these features:

* the ability to quickly adapt to both new hardware and software technologies without rewriting application software is a key to reducing labor costs;

* application software is not burdened with the overhead of unneeded features;

* second-generation applications can take advantage of new, powerful iRMX 86 subsystems, enabling them to enter more sophisticated markets; and

* special application needs can be satisfied without the time and expense normally required to write an application-specific operating system.

## Operation

The iRMX Operating System architecture functions much like the Intel 8086 microprocessor. iRMX "operators," (system calls) manipulate "objects" (operating system data structures) in much the same way 8086 operators manipulate operands. iRMX 86 objects include tasks, memory segments, jobs, etc. The iRMX 86 data structures are treated symmetrically so users can create their own "objects" and write their own "operators" to achieve custom operating system functions. These user-defined objects and operators are no different in system operation than those provided with the original system.

Unlike earlier operating systems, the iRMX 86 is actually a library of functions which are used in conjunction with user-developed tasks or programs. This gives OEMs more flexibility in putting together custom systems. In effect, it gives OEMs control over features, size, and configuration, not of just the application programs, but of the operating system as well. Moreover, it allows the designer to enhance application software to meet new requirements.

The iRMX 86 Operating System gives the software designer a full-featured operating system based around a multiprogramming/-multitasking nucleus. Optional facilities include a device-independent input/output subsystem with an application loader; a human interface subsystem with a command language interpreter and a console interface; an interactive debugging subsystem; and a stand-alone terminal handler subsystem.

The iRMX 86 nucleus provides the foundation upon which application systems are built. Many multiprogramming, multitasking and real-time facilities are included, such as interprogram and intertask communication, interupt management, task synchronization and critical section management features.

The iRMX 86 priority-oriented schedular guarantees that the most important task is given the required system resources, allowing the application to be responsive to its external environment. Also, flexible error handling and reporting subsystems detect and trap errors such as those caused by incorrect coding of new programs and tasks. The error management system aids the user's development process by detecting software failures and allowing the user a variety of methods to handle the error, therefore reducing the debug time for new applications.

The iRMX 86 system also features a comprehensive I/O subsystem. This includes a standardized device-independent interface for application programs to communicate with an I/O device. Also included are both common and random access device driver support options that allow the user to easily write his own device driver. A single-density diskette driver, hard disk driver, and USART driver are provided with the system.

The iRMX 86 Human Interface Subsystem* provides interactive control over system resources and utilities. iRMX 86 system utilities include display file directories, copy files, rename files, etc. The Intel-supplied command-line interpreter is table driven, allowing easy modification by the user for application-specific requirements. An application command can be created. When using the command line interpreter, this applicaton language will be translated so that the appropriate user program will be invoked.

The iRMX 86 object-oriented debugging subsystem offers sophisticated facilities. iRMX 86 system lists, objects, and user-defined objects can be viewed and altered. Execution and task breakpointing facilities are also available.

## Recap of iRMX 86 Features

iRMX 86 nucleus:

* multiprogramming,
* multitasking,
* intertask communication and synchronization,
* interprogram communication and synchronization, etc.
* critical section manager,
* free space manager,
* error manager,
* user-extensible

iRMX 86 I/O subsystem:

* asynchronous I/O interface,
* device independence,
* file management system,
* standardized device/driver interface,
* hierarchical directories,
* stream I/O,
* user-selectible physical block granularity,
* file access rights,
* synchronous I/O interface with automatic buffering,*
* iSBC 204™ single-density diskette device driver,
* iSBC 206™ hard-disk device driver.

iRMX 86 terminal handler subsystem:

* line editing capabilities,
* keystroke control over out-start/stop/delete,
* echo input characters on console.

iRMX 86 debugger subsystem:

* view iRMX 86 system list,
* inspect objects--jobs, tasks, mailboxes, semaphores, regions, user-defined objects,
* inspect or alter absolute memory,
* execute iRMX 86 programs,
* set, change view, delete breakpoints

iRMX 86 human interface subsystem*:

* command line interpreter,
* utility programs,
* application parsing services,
* multiple-command sources (terminal or disk file);
* dynamic addition/deletion of commands,
* configurable syntax deliminters.

iRMX 86 application loader*:

* synchronous loading of both absolute and relocatable code.

iRMX 86 applications are developed using Intel's Intellec^R microcomputer development systems. The iRMX 86 operating system run on Intel's iSBC 86^TM single-board computers and on user 8086-based systems.

## Pricing

The license fee for the complete set of iRMX 86 features listed above is $7,500 with royalties as shown below. Delivery is 30 days ARO.

Royalty Schedule

| Yearly Purchase Quantities | Royalty Price per Use |
|---|---|
| 1-24 | $300 |
| 24-49 | $225 |
| 50+ | $160 |

# 2 Executive Overview

## THE iRMX 86 REAL-TIME EXECUTIVE OVERVIEW

Intel's iRMX 86 Real-Time Operating System is an easy-to-use, sophisticated software system that operates in Intel iAPX 86 and iAPX 88 microcomputer solutions.

Some of the key features of the operating system are listed below:

| TOPIC | iRMX 86 OPERATING SYSTEM |
|---|---|
| HARDWARE | Will run on any iSBC 86 or iSBC 88 board, or any board built around an 8086 or 8088 processor as long as the 8253 timer and 8259 interrupt controller are configured at the same addresses as in the iSBC products. (Note: Release 3 will remove this restriction) |
| COMPATIBILITY | Is upwared compatible with the operating systems designed for the iAPX 186 and the iAPX 286 processors. Through the use of the Universal Development Interface (UDI), the operating system also remains compatible with all other operating systems and language compilers and translators written to this specification. |
| REAL-TIME PERFORMANCE | Achieves the response time required by real-time systems by scheduling the system tasks according to an event-driven, priority-based scheme. Additional performance can be reached by incorporating the fully supported 8087 Math Coprocessor. |
| MARKET APPLICATIONS | Fits well into applications such as Business and Word Processing, as well as real-time applications such as process control and medical electronics. |
| "GOTCHAS" | -Provides real-time response in a multiprogramming environment.<br>-Supports all the new technology devices (iSBC 215, 218, 254, etc.) with device independent I/O.<br>-Entire operating system (or parts of it) can be (P)ROM'ed.<br>-8087 is completely supported in a multitasking environment.<br>-Configurable to include only those parts that are used. |
| CLOSE-THE-ORDER | Use the iRMX 86 Data Sheet and the tools provided in this manual to assist the customer in solving his problems. Get the Master Software License signed, so we can immediately ship the initial copy at $7,500. Royalties start at $300 for each incorporation into a derivative product. |

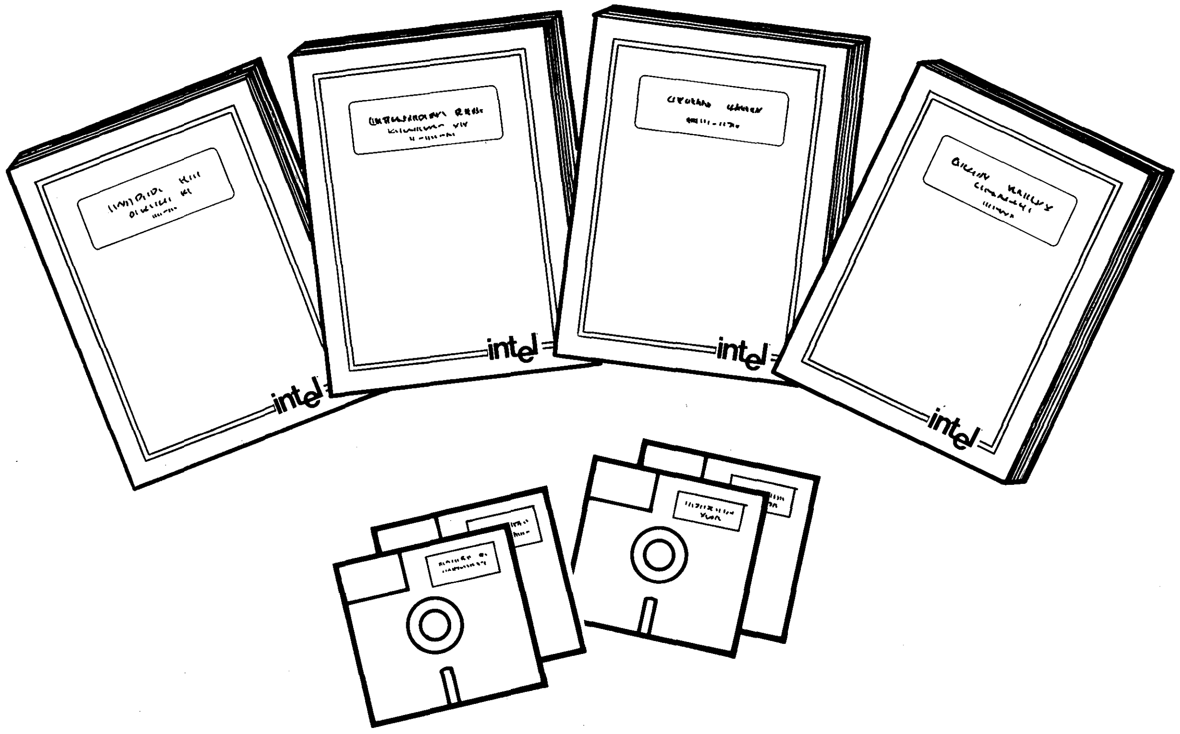
TOPIC iRMX 86 OPERATING SYSTEM

KEY REFERENCE DATA

| | |
|---|---|
| Size Data: | |
| Minimum Nucleus | 10 Kbytes |
| Full Nucleus | 22 Kbytes |
| Minimum Task RAM Overhead | 300 bytes |
| Miscellaneous Data: | |
| Timer Resolution | Configurable from 10 msec |
| Minimum Interrupt Latency | 50 usec |
| Average Context Switch Time | 700 usec |

# 3 Product Description

The Intel iRMX 86 Operating System is an easy to use, real time, sophisticated software system which operates on Intel iSBC 16-bit boards as well as user iAPX 86 and iAPX 88 based microcomputers.



The iRMX 86 Operating System is designed to manage and extend the varied resources of an iAPX 86 or iAPX 88 based system. The architecture provides a structured, efficient environment for many applications, including process control, intelligent terminals, office systems, word processors, business data processing, medical electronics, and data communication.

The various layers of the operating system provide the management needed to utilize the resources pictured in Figure 3.1. By managing the processor time, the allocation and use of memory, the various devices attached to the system, and the data transfered between them, the iRMX 86 operating system provides the user with the tools needed to support almost any application.

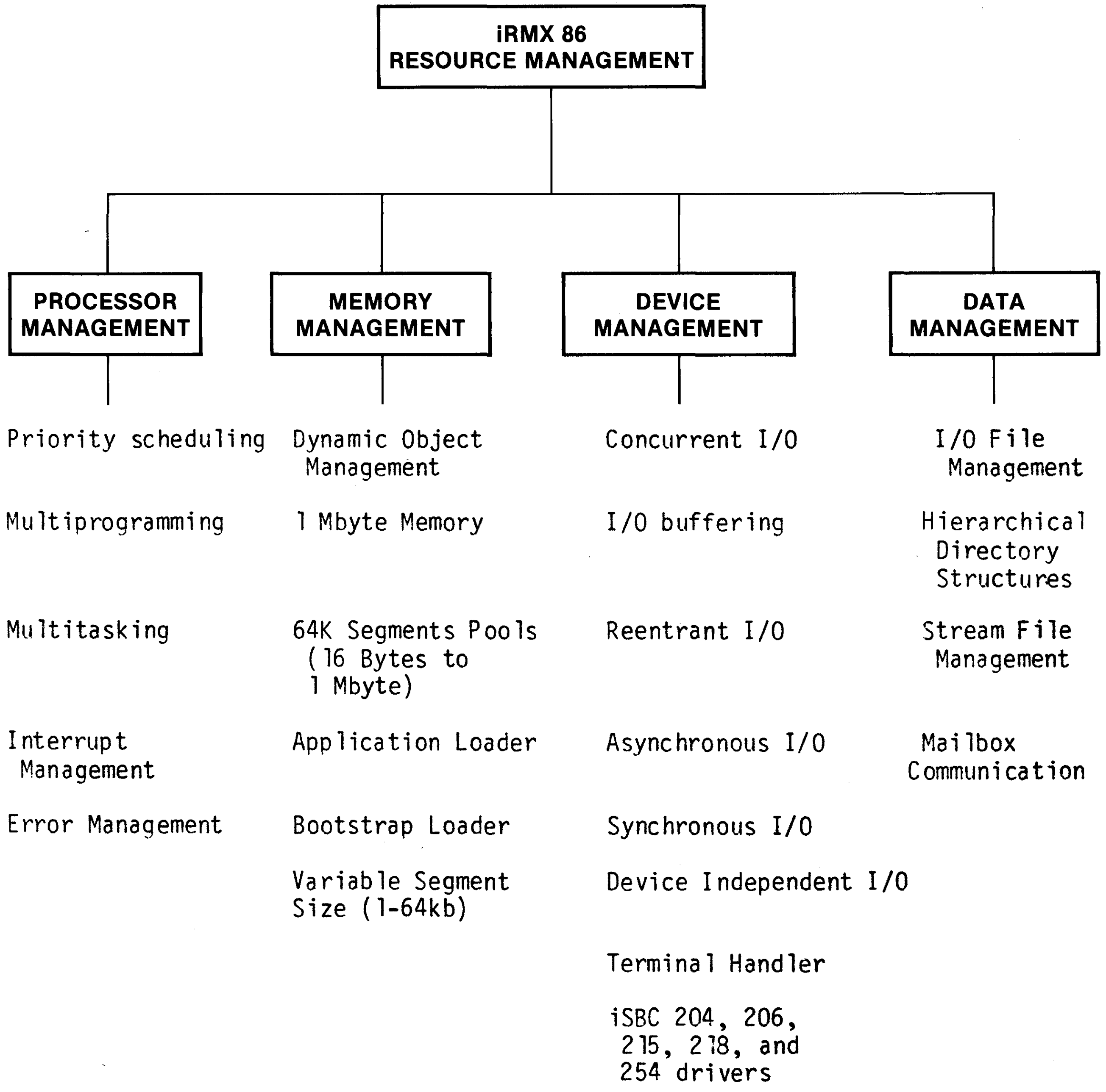


Figure 3.1

Resources Managed by the iRMX 86 Operating System

## PROCESSOR MANAGEMENT

Management of the processor time and resources is the responsibility of the inner-most layer of the operating system. The Nucleus, as shown in Figure 3.2 provides those functions neccessary to schedule the tasks operating in the different job environments of the system. The Nucleus also provides an interface to the outside world by acknowledging and processing interrupts and managing errors detected at various levels of the system.

## NUCLEUS

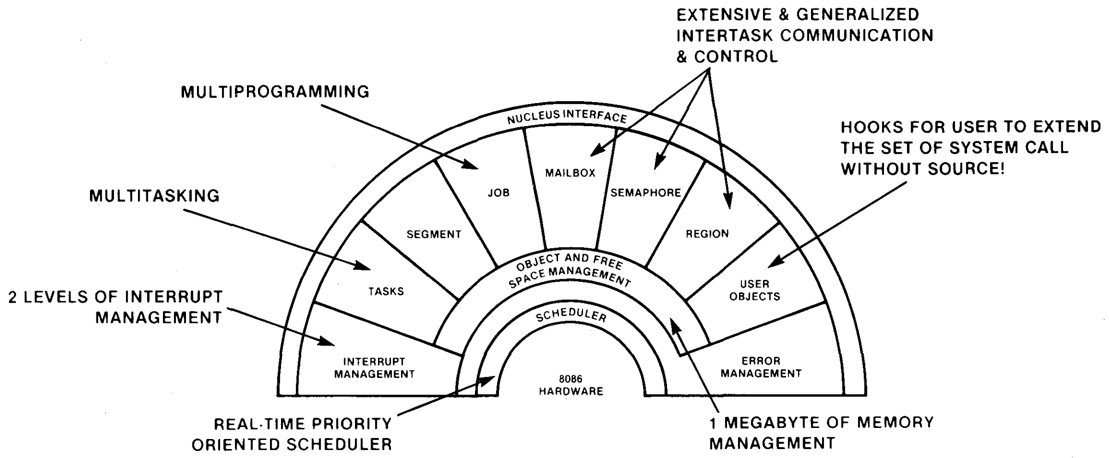

EXTENSIVE & GENERALIZED INTERTASK COMMUNICATION & CONTROL
MULTIPROGRAMMING
HOOKS FOR USER TO EXTEND THE SET OF SYSTEM CALL WITHOUT SOURCE!
MULTITASKING
2 LEVELS OF INTERRUPT MANAGEMENT
NUCLEUS INTERFACE
JOB
MAILBOX
SEMAPHORE
SEGMENT
REGION
TASKS
OBJECT AND FREE SPACE MANAGEMENT
USER OBJECTS
SCHEDULER
INTERRUPT MANAGEMENT
8086 HARDWARE
ERROR MANAGEMENT
REAL-TIME PRIORITY ORIENTED SCHEDULER
1 MEGABYTE OF MEMORY MANAGEMENT

Figure 3.2
The Nucleus provides support for the system and user objects.

The priority based, event-driven scheduling system provides the real-time response to external interrupts needed in time-critical, and dynamic systems. User selectable priorities guarantee this neccessary control. The scheduler provides the user with thousands of "virtual 8086's" by allowing many independent tasks to be scheduled with appropriate priorities and private execution environments.

Multiprogramming is supported by allowing the partitioning of any group of tasks into an isolated environment called a Job. This facility gives a user the opportunity to run more than one application at a time with less concern for possible affects on other users.

Two levels of both interrupt and error management are provided by the Nucleus. The user is allowed to make the trade off between speed and power as well as between global, system level control and more direct, local control.

NUCLEUS OBJECT MANAGEMENT SYSTEM CALLS

| System Calls for All Objects | O.S. Objects | Attributes | Object-Specific System Calls |
|---|---|---|---|
| | JOBS | Tasks<br>Memory pool<br>Object directory<br>Exception handler | CREATE$JOB<br>DELETES$JOB<br>SET$POOL$MIN<br>GET$POOL$ATTRIB<br>OFFSPRING |
| CATALOG$OBJECT<br>UNCATALOG$OBJECT<br>LOOKUP$OBJECT | TASKS | Priority<br>Stack<br>State<br>Exception handler<br>Machine registers<br>Instruction pointer | CREATE$TASK<br>DELETE$TASK<br>SUSPEND$TASK<br>RESUME$TASK<br>GET$EXCEPTION$HANDLER<br>SET$EXCEPTION$HANDLER<br>SLEEP<br>GET$TASK$TOKENS<br>GET$PRIORITY<br>SET$PRIORITY |
| ENABLE$DELETION | SEGMENTS | Base<br>Length | CREATE$SEGMENT<br>DELETE$SEGMENT<br>GET$SIZE |
| DISABLE$DELETION<br>FORCE$DELETE | MAILBOXES | List of objects<br>List of tasks waiting for objects | CREATE$MAILBOX<br>DELETES$MAILBOX<br>SEND$MESSAGE<br>RECEIVE$MESSAGE |
| GET$TYPE | SEMAPHORES | List of tasks waiting for units<br>Unit count | CREATE$SEMAPHORE<br>DELETE$SEMAPHORE<br>RECEIVE$UNITS<br>SEND$MESSAGE |
| | REGIONS | List of tasks waiting for critical section<br>Control state | CREATE$REGION<br>DELETE$REGION<br>RECEIVE$CONTROL<br>ACCEPT$CONTROL<br>SEND$CONTROL |
| | USER OBJECTS | Type code<br>Deletion Mailbox<br>Custom objects | CREATE$EXTENSION<br>DELETES$EXTENSION<br>CREATES$COMPOSITE<br>DELETE$COMPOSITE<br>INSPECT$COMPOSITE<br>ALTER$COMPOSITE |

Figure 3.3

System calls provided by the object oriented iRMX 86.
Nucleus show definite signs of symetry and consistency.

## MEMORY MANAGEMENT

System memory is also managed primarily by the Nucleus. By treating the various segments of memory as objects, the system hides the actual structures from the user, and makes all system objects easy to use. The partitions (Jobs) provided to allow multiprogramming, enable the user to focus on the segments of memory required for one particular application at a time. The memory management system is completely dynamic; able to reclaim any segment of the one Megabyte of RAM under its control when it is no longer needed.

Both the Application Loader, and the Bootstrap Loader are able to load program code into any section of the one Megabyte of available memory. The Bootstrap Loader takes only 800 bytes of (P)ROM, and may be used to load the rest of the operating system into RAM upon start-up of the system.

## DEVICE MANAGEMENT

Managment and control of the various devices attached to the system is provided primarily by the I/O levels of the system. The Nucleus forms the basic structure on top of which the Basic I/O system and the Extended I/O system are built. Figure 3.4 shows this relationship between these outer layers of the system. The rings in the diagram are built on top of the inner ring provided by the Nucleus. Figures 3.5 and 3.6 represent the outer-most rings, built on top of those provided by the Basic I/O system. The user is allowed to interface with the features of the iRMX 86 Operating System at any on these levels, through any of the interfaces pictured in the diagrams.

Basically, the features provided by the I/O system include the ability to communicate with an number of different devices in a buffered or non-buffered manner, synchronously, or asynchronously, in a concurrent fashion. The drivers provided to perform these functions are reentrant in nature, thereby allowing only one copy of the code to be used by many different connections to different devices.

A simple terminal handler provides a basic RS-232 interface to a system console device. This driver includes line editing features like rubout, supress output, resume output, repeat line, and signal the end of a line.
Other drivers provide the more complex connection to the iSBC 204 floppy disk controller, the iSBC 206, iSBC 215, and iSBX 218 hard disk and Winchester disk controllers, as well as the iSBC 254 magnetic bubble memory controller.

The synchronous and aynchronous I/O of the iRMX 86 device independent I/O facility provides the user with the basic tools needed to generate custom drivers to any iSBC product or any custom device controller. The standard READ, WRITE, and SEEK system calls can be issued by an application regardless of the type of device being used. To write a custom controller, the user need only write the basic device dependent portions and utilize the standard features provided by the I/O system remaining.

## BASIC I/O SYSTEM

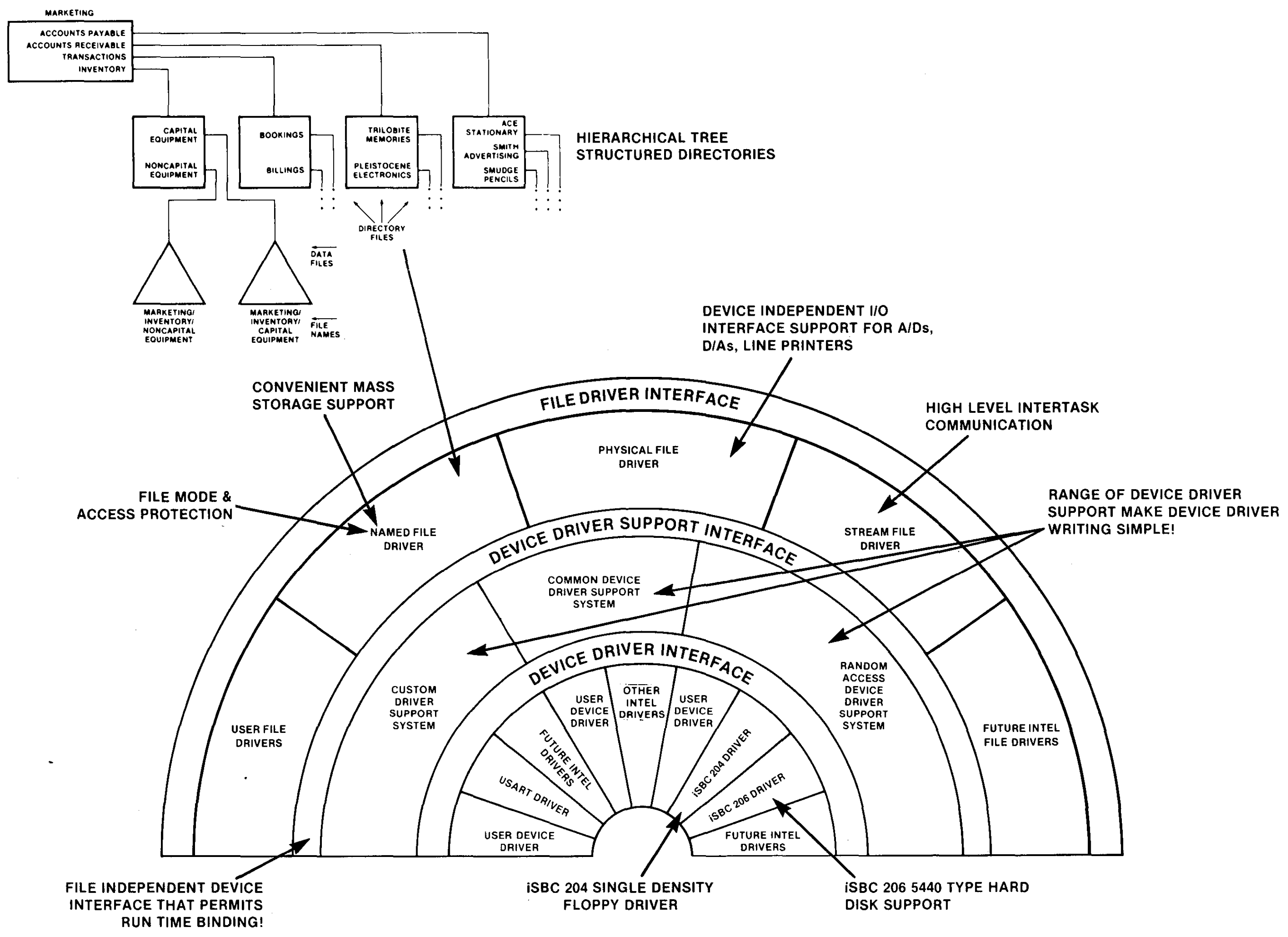

MARKETING
ACCOUNTS PAYABLE
ACCOUNTS RECEIVABLE
TRANSACTIONS
INVENTORY
CAPITAL EQUIPMENT
NONCAPITAL EQUIPMENT
BOOKINGS
BILLINGS
TRILOBITE MEMORIES
PLEISTOCENE ELECTRONICS
ACE STATIONARY
SMITH ADVERTISING
SMUDGE PENCILS
HIERARCHICAL TREE STRUCTURED DIRECTORIES
DIRECTORY FILES
DATA FILES
MARKETING/ INVENTORY/ NONCAPITAL EQUIPMENT
MARKETING/ INVENTORY/ CAPITAL EQUIPMENT
FILE NAMES
CONVENIENT MASS STORAGE SUPPORT
DEVICE INDEPENDENT I/O INTERFACE SUPPORT FOR A/Ds, D/As, LINE PRINTERS
FILE DRIVER INTERFACE
HIGH LEVEL INTERTASK COMMUNICATION
PHYSICAL FILE DRIVER
FILE MODE & ACCESS PROTECTION
NAMED FILE DRIVER
STREAM FILE DRIVER
RANGE OF DEVICE DRIVER SUPPORT MAKE DEVICE DRIVER WRITING SIMPLE!
DEVICE DRIVER SUPPORT INTERFACE
COMMON DEVICE DRIVER SUPPORT SYSTEM
CUSTOM DRIVER SUPPORT SYSTEM
RANDOM ACCESS DEVICE DRIVER SUPPORT SYSTEM
DEVICE DRIVER INTERFACE
USER FILE DRIVERS
FUTURE INTEL FILE DRIVERS
USER DEVICE DRIVER
OTHER INTEL DRIVERS
USER DEVICE DRIVER
FUTURE INTEL DRIVERS
iSBC 204 DRIVER
USART DRIVER
iSBC 206 DRIVER
USER DEVICE DRIVER
FUTURE INTEL DRIVERS
FILE INDEPENDENT DEVICE INTERFACE THAT PERMITS RUN TIME BINDING!
iSBC 204 SINGLE DENSITY FLOPPY DRIVER
iSBC 206 5440 TYPE HARD DISK SUPPORT

Figure 3.4

The Basic I/O System provides the interface between different devices and the objects managed by the Nucleus.

## EXTENDED I/O SYSTEM

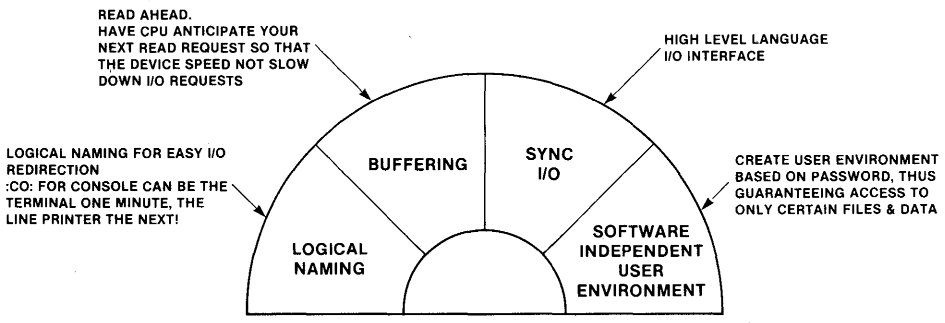

READ AHEAD.
HAVE CPU ANTICIPATE YOUR NEXT READ REQUEST SO THAT THE DEVICE SPEED NOT SLOW DOWN I/O REQUESTS
HIGH LEVEL LANGUAGE I/O INTERFACE
LOGICAL NAMING FOR EASY I/O REDIRECTION
:CO: FOR CONSOLE CAN BE THE TERMINAL ONE MINUTE, THE LINE PRINTER THE NEXT!
BUFFERING
SYNC I/O
CREATE USER ENVIRONMENT BASED ON PASSWORD, THUS GUARANTEEING ACCESS TO ONLY CERTAIN FILES & DATA
LOGICAL NAMING
SOFTWARE INDEPENDENT USER ENVIRONMENT

Figure 3.5

The EIOS builds a simpler interface upon the basics provided by the BIOs.

## HUMAN INTERFACE

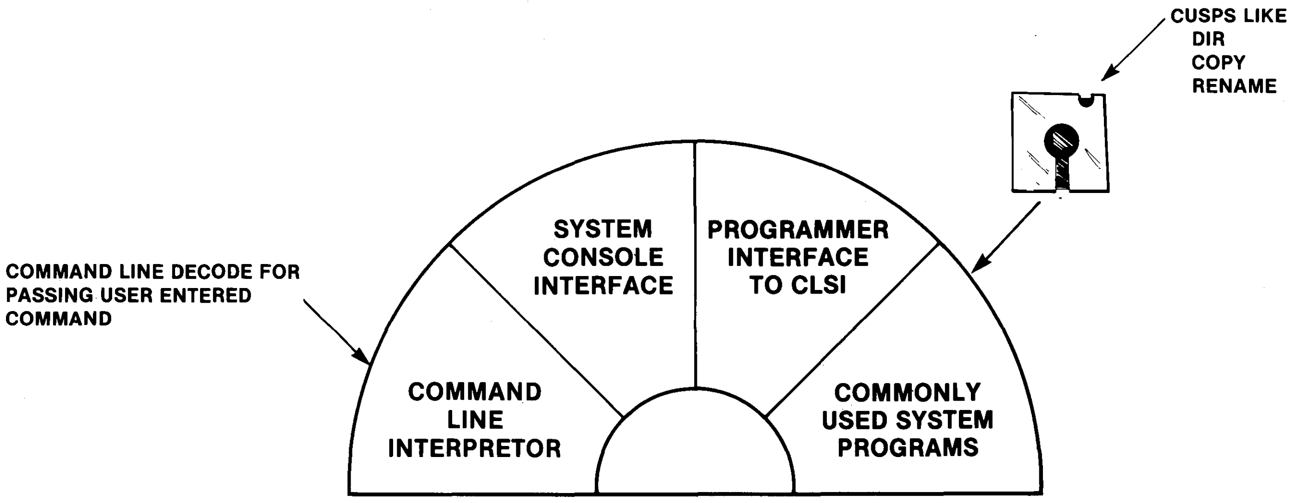

CUSPS LIKE
DIR
COPY
RENAME
COMMAND LINE DECODE FOR PASSING USER ENTERED COMMAND
SYSTEM CONSOLE INTERFACE
PROGRAMMER INTERFACE TO CLSI
COMMAND LINE INTERPRETOR
COMMONLY USED SYSTEM PROGRAMS

Figure 3.6
The Human Interface provides the user with easy access to the system.

## DATA MANAGEMENT

The I/O Systems provides the basic data managment facilities found in the most complete operating systems. Both Named and Physical files are supported. The Named files are supported by a hierarchical data directory structure that is consistent with large mass storage devices. The Physical files are geared more towards physical devices such as printers, A/D converters and D/A converters. Temporary, RAM based files, known as Stream Files, are also provided to enable applications such a translators and interpreters to store data from one pass to the next.

A directory service allows for functional grouping of data into a hierarchical structure. The location of any particular segment of data is aided by the use of a tree structure of directories located on the mass storage devices (Figure 3.4).

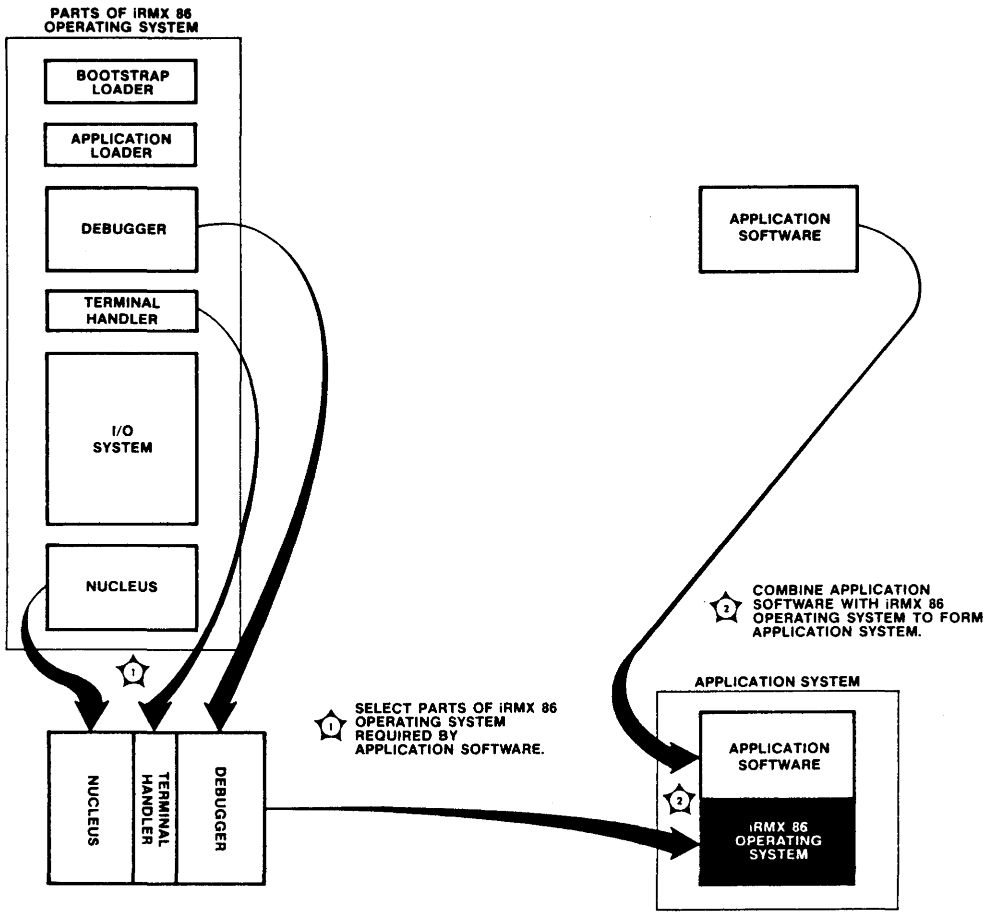
Data communication between tasks is managed via the basic file structures outlined above, or through a sophisticated Mailbox system provided by the Nucleus. This system offers message communication and system synchronization by permitting system and user created objects to be passed from one task to another in a well defined and controlled manner.

The Human interface (Figure 3.6) provides a basic and simple interface between the user and the system. It includes such features as a Command Line Interpreter (CLI) to accept user commands from the system console and cause appropriate action to take place. Included in the Human Interface are commands like Directory Copy, and Renaming of a File.

## CONFIGURATION

Not all of the resource management facilities discussed above are needed by all applications. In order for a user to tailor the iRMX 86 Operating System to the particular needs of an application, he must go through a process called configuration.

Configuration is simply taking the features you want, then combining them into the iRMX 86 system of your choice. Your "Black Box" or feature set is custom tailored to your needs! Pictorially, configuration looks like this:


PARTS OF iRMX 86 OPERATING SYSTEM
BOOTSTRAP LOADER
APPLICATION LOADER
DEBUGGER
TERMINAL HANDLER
I/O SYSTEM
NUCLEUS
NUCLEUS
TERMINAL HANDLER
DEBUGGER
1
SELECT PARTS OF iRMX 86 OPERATING SYSTEM REQUIRED BY APPLICATION SOFTWARE.
APPLICATION SOFTWARE
2
COMBINE APPLICATION SOFTWARE WITH iRMX 86 OPERATING SYSTEM TO FORM APPLICATION SYSTEM.
APPLICATION SYSTEM
APPLICATION SOFTWARE
2
iRMX 86 OPERATING SYSTEM

However, iRMX 86 goes one step further, each of the layers is configurable! The Nucleus, Basic I/O System, Extended I/O System, and Human Interface all have portions that can be included or excluded. For example, if your customer does not want semaphores in his Nucleus he can choose a Nucleus that looks like Figure 3.8 rather than like 3.7

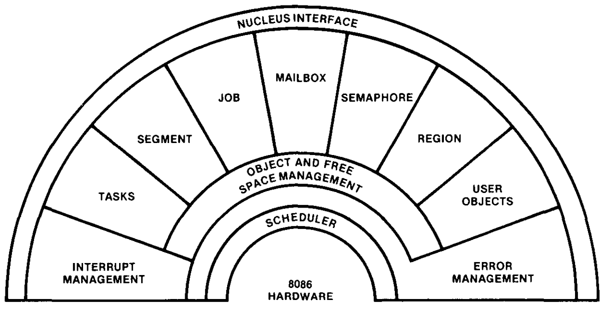


Figure 3.7

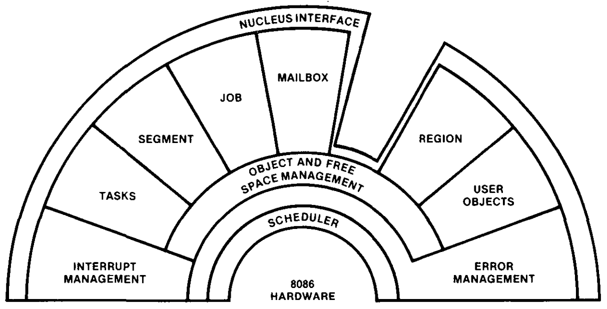


Figure 3.8

## THE HARD FACTS ABOUT RELEASE 2

### SYSTEM FEATURES

Number of tasks - 65,536
Number of priority levels - 256
Number of jobs (program) - 65,536
Number of segments - 65,536
Number of semaphores - 65,536
Number of regions - 65,536
Number of mailboxes - 65,536
Units per device - 255
Number of devices - 65,536
Internal clock capacity - 130 years
Minimum system clock resolution 10 milliseconds
Interrupt task latency - less than 900 usec
Interrupt handler latency - less than 50 usec

Note: These numbers are never reached in practice due to memory constraints.

### SIZES

Estimated code size reduction for Release 3 is 20% across the board!

| | | RELEASE 2 | | RELEASE 3 |
|---|---|---|---|---|
| NUCLEUS | | | | |
| | CODE | 13.5K standalone | → | 11K |
| | | 23.5K foundation for BIOS | | |
| | | 28.4K all primitive | → | 21K |
| | RAM | 1.6KB without memory scan | | |
| | | 9 KBytes with memory scan (used temporarily) | | |
| | | 80 Bytes per task | | |
| | | 64 Byes per job | | |
| | | 16 Bytes per segment | | |
| | | 32 Bytes per semaphore | | |
| | | 32 Bytes per region | | |
| | | 32 Bytes per mailbox | | |
| | | 32 Bytes per extension | | |
| | STACK | 500 bytes | → | 220 bytes |

### TERMINAL HANDLER

| | | |
|---|---|---|
| CODE | 1.1K | Output only |
| | 3.3K | Full System |

### APPLICATION LOADER

| | |
|---|---|
| CODE | 3.6K |
| RAM | 2K (1K reusable by loaded application) |

### BOOTSTRAP LOADER

| | |
|---|---|
| CODE | 500 bytes minimum |
| | 1.3K maximum |
| RAM | 3K |

## BASIC I/O SYSTEM

### CODE

1. PHYSICAL FILES ONLY
   13K - minimum - byte bucket driver only
   19K - 206 Driver Only (2.5K)
   38K - 204 Driver (1.5K) and 206 Driver

2. NAMED FILES ONLY
   38K - minimum - 206 Driver Only
   47K maximum - 204 and 206 Drivers

3. NAMED AND PHYSICAL FILES
   41K - 206 Driver Only

4. NAMED, PHYSICAL, AND STREAM FILES
   44K - 206 Driver Only

5. COMPLETE BIOS - including all BIDS system calls, TIMER, and Terminal Handler Interface.
   56K - byte bucket, 204, and 206 driver

### RAM

1. BIOS SYSTEM
   4.8K minimum - (no deblocking buffers)
   5.4K with four 512 Byte Deblocking Buffers

2. DRIVERS
   1.4K for 206 DRIVER
   1.4K for 204 DRIVER
   1.3K for Terminal Handler

3. PER DEVICE CONNECTION

| | |
|---|---|
| NAMED: | 2.4K without deblockng buffers |
| | 5.2 with four deblocking buffers |
| STREAM: | 900 |
| PHYSICAL: | 1K without deblocking buffers |
| | 3K with four 512 deblocking buffers |

4. PER FILE CONNECTION

| | |
|---|---|
| NAMED: | 1.3K without deblocking buffers |
| | 1.5K with four 512 byte deblocking |
| STREAM: | 800 Bytes |
| PHYSICAL: | 800 Bytes |
| STACK: | 200 Bytes |

# 4 Features and Benefits

THE TOP FIVE

| FEATURE | BENEFIT |
|---|---|
| Object-Oriented | "Ease-of-use" is the name of the game through higher levels of abstraction. The IBM system 38 mainframe is the only other commercially available state-of-the-art object-oriented system! |
| User Extensible | No source is required for users to add their own system calls and custom objects. |
| Multiprogramming | Resources can be divided into 64k independent multitasking partitions allowing independent applications to reside on one iRMX 86 system simultaniously. |
| Device Independent I/O | I/O read and write requests are device driver and file driver independent! |
| Priority Based Scheduler | The iRMX 86 Operating System offers realtime response to interrupts and allows programmers dynamic control over task execution. The system recognizes 256 separate task priorities. |

| FEATURE | BENEFIT |
|---|---|
| Configurable Family | The functionality provided by the iRMX 86 Operating System range from 14 Kbytes to 190 Kbytes allowing a family of different applications ranging from dedicated "turnkey" operations through multiterminal and multiprogramming system. |
| Multitasking | The iRMX 86 Operating System can appear to provide 65,536 separate "8086 processors", or tasks, because each task has its own code, data, register set, and task state. This allows an iRMX 86 system to be responsive to a number of asnynchronous process control applications while providing the foundation for multiterminal applications. |
| Extensive Intertask Communication and Control | Four mechaninsms: Mailboxes, Counting Semaphores, Regions, and Stream Files provide intertask communication, making the iRMX 86 Operating System the most powerful O.S. in this area of communication and control. |
| Portability | The iRMX Object oriented operating system will easily port to the hardware objects supported by future processors such as the iAPX 186 and iAPX 286. |

| FEATURE | BENEFIT |
|---|---|
| Hierarchical File Directory Structure | Tree structured directories offer improved access protection, so data can be organized by function, offering faster file location due to a limited directory search. |
| (P)ROM'able | No disk is required by the iRMX 86 Operating System! It can reside in (P)ROM!! However, if a RAM base system is desired, a bootloader is available as part of the system. |
| Minimum Hardware | The iRMX 86 Operating System will work with iSBC 86 and iSBC 88 boards, even though it makes no assumptions about the board or form factor. |
| Human Interface | The operator's interface to the system is aided by an automatic system start-up at power-on. |
| Interrupt Management | Two levels of interrupt management allows the user to trade between speed and power. |
| Error Management | Hierarchical Error Management provides flexible error servicing to protect application integrety and shorten debug time. |
| Memory Management | The full one Megabyte address range of the iAPX 86 and iAPX 88 processors is supported by the memory manager of the iRMX 86 Operating System. |

## iRMX 86 OPERATING SYSTEM

System software to support both your board and component accounts is here! And it's a major commitment by Intel to solve the software crisis!!!

The iRMX 86 Operating System is the super-star among microprocessor operating systems! It stands several notches above the competition in capabilities and in price/performance. Just consider that other operating systems in iRMX 86's class are a minimum of two times more expensive! iRMX 86 represents a gigantic step forward by offering mainframe functionaltiy at microcomputer prices!

Intel is the only computer manufacturer offering a full-featured operating system without imposing a system hardware architecture! iRMX 86 assumes no board form factor; assumes no mass storage devices, but supports your customers' iAPX 86 and iAPX 88-boards and is designed to move applications more gracefully to the iAPX 286 processor. Additionally, iRMX 86 supports the iSBC 86 and iSBC 88-based boards along with iSBC device controllers, as well as customer designed controllers. As a result, it's the first TRUE vendor supplied OEM software that shows responsibility for future processors and state of the art controllers!

iRMX 86 introduces some unique capabilities to the microprocessor world. It offers object-orientation, making iRMX 86 be to the operating system world what high level languages offered to the assembly language world-- "EASE OF USE"!. In addition, it is the first operating system that provides hooks allowing customers to add their own unique system calls to iRMX 86 without needing or being familiar with SOURCE code!

All of that combined with the iAPX 86 hardware family (including the 8087 floating point processor) and the growing Intel UDI standard software bus, should make your selling job easier and make you proud to be a member of the "Intel delivers solutions" team!

## SALES STRATEGY

Intel software forms part of an overall strategy to sell Intel hardware; Boards as well as components. Figure 5.1 indicates the different approaches that may be necessary to reach that overall objective. Any customer needing to solve a problem using microcomputers comes to the salesman fully equiped with a range of background experiences, personal bias and strong emotions. The customer usually wishes to enter the matrix shown in the figure at one of four different entry points. The strategy statements at the bottom of the figure will help to guide the customer to the highest Intel contribution node possible. Where it becomes necessary to use third party software, chassis, or application hardware, don't say "NO" - help the customer find the solution that takes the most advantage of Intel boards, components, and foundation software.

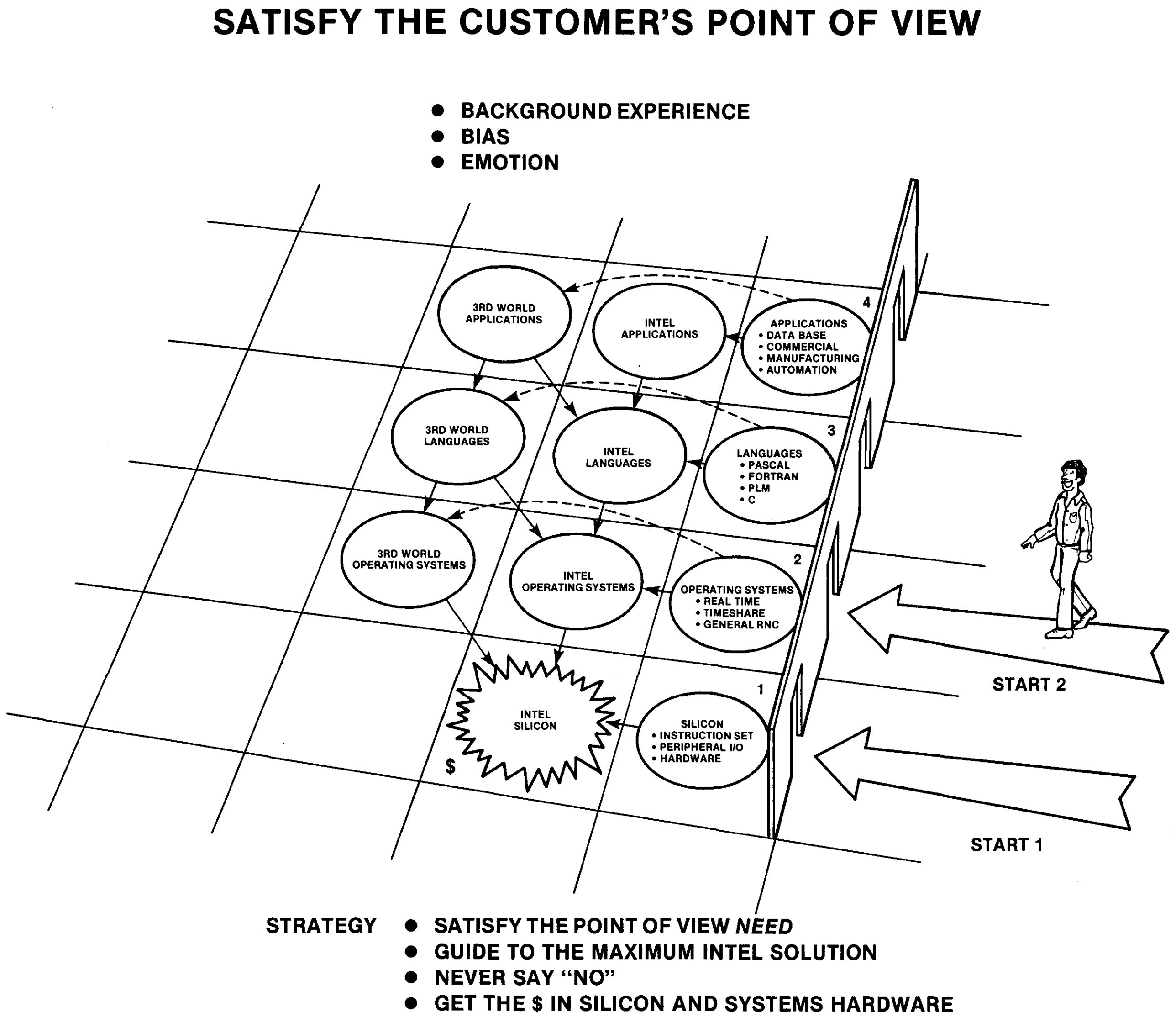

SATISFY THE CUSTOMER'S POINT OF VIEW
• BACKGROUND EXPERIENCE
• BIAS
• EMOTION
3RD WORLD APPLICATIONS
INTEL APPLICATIONS
APPLICATIONS • DATA BASE • COMMERCIAL • MANUFACTURING • AUTOMATION
4
3RD WORLD LANGUAGES
INTEL LANGUAGES
LANGUAGES • PASCAL • FORTRAN • PLM • C
3
3RD WORLD OPERATING SYSTEMS
INTEL OPERATING SYSTEMS
OPERATING SYSTEMS • REAL TIME • TIMESHARE • GENERAL RNC
2
INTEL SILICON
$
SILICON • INSTRUCTION SET • PERIPHERAL I/O • HARDWARE
1
START 2
START 1
STRATEGY
• SATISFY THE POINT OF VIEW NEED
• GUIDE TO THE MAXIMUM INTEL SOLUTION
• NEVER SAY "NO"
• GET THE $ IN SILICON AND SYSTEMS HARDWARE

Figure 5.1
GUIDE THE CUSTOMER TO INTEL

Advocate these advantages:

1. Lower cost

   - o off-the-shelf system
   - o does not require a predefined board set
   - o does not require a disk

2. Compatible product family

   - o configurable to satisfy "VW" through "Porsche Turbo Carrerra" products
   - o user extensible, at a system call level, without source code

3. "Easy to use"

   - o object-orientation
   - o application debugger
   - o iRMX 86 training course

## MARKET POSITION

Customers have been using microcomputers in many different applications, ranging from logic replacment to dedicated computers, to reprogrammable systems. Logic replacement refers to microprocessor based systems that perform a single, discrete function. Dedicated computers are microcomputer based solutions that operate on more than one function all within one application. Reprogrammable computers are a computer systems that contain language translator(s) in order to create or modify application software.

Our customers tell us their applications can be divided into two different categories: performance and functionality (see Figure 5.2) Performance is related to the number of I/O interrupts, the memory efficiency, and fundamental control code. The functionality category is the overall system capability, lots of features, extremely flexible, multiuser, multiprogram, hardware independence, and full protection.

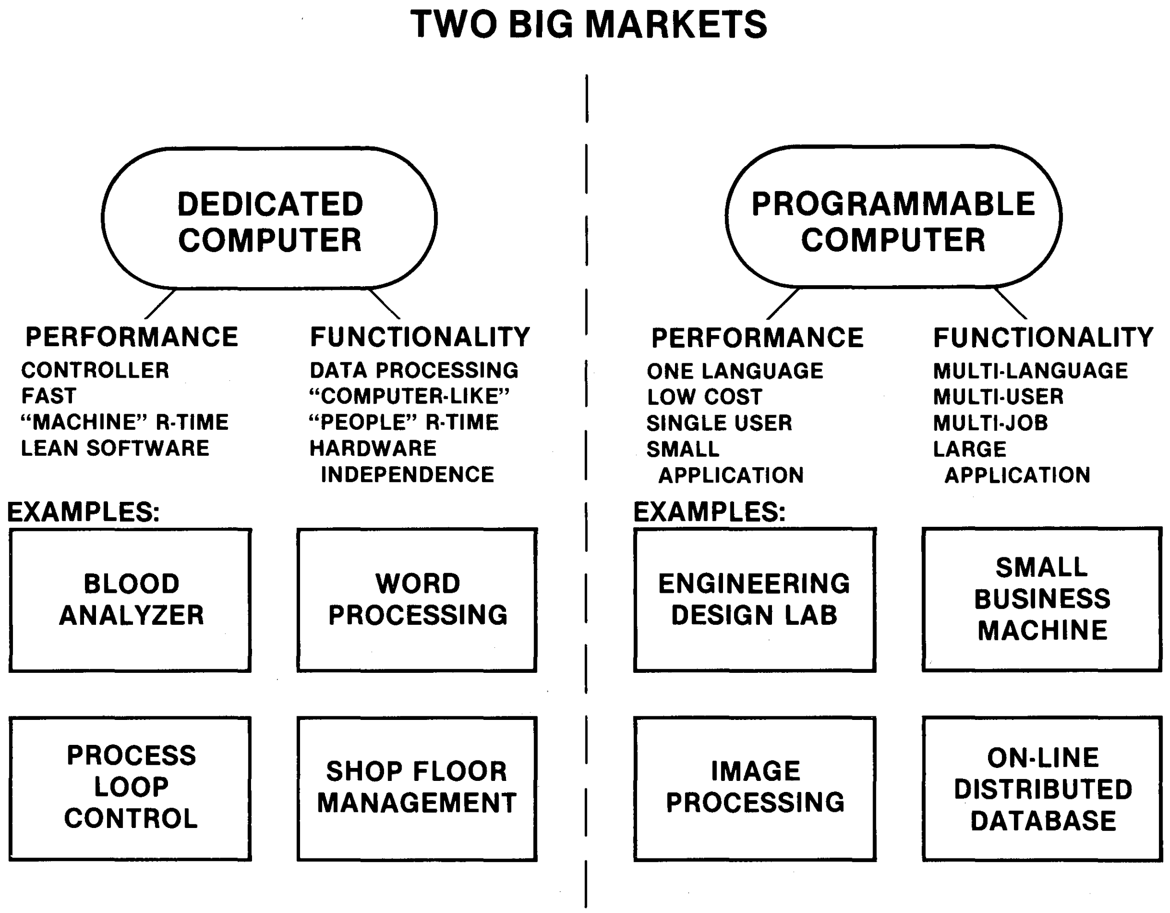


Figure 5.2
Performance / Functionality Markets

The VLSI curve for a single microcomputer would show a curve similar to that in Figure 5.3. The more functions to be provided, the less performance is to be expected. If you execute more code for a given function, it has to cost some performance! Imposed on this curve are the operating systems available from Intel. As you may have noticed, the iRMX 86 Operating system forms the basis for the iRMX 286 Operating system. Because of Intel's committment to help customers build their products, we stopped and listened as they described their dedicated computer applications. There are the fairly obvious differences as shown in Figure 5.3, but the migration story was just as important. Many of our 8080/8085 customers are looking for a growth path for previously designed dedicated computer applications. Many found the iRMX 80 Operating system to be sufficient for their multitasking operations in the past, but are looking now for a portable solution that offers the full performance of an iAPX 86 or 88 design (see Figure 5.2) The iRMX 88 Operating System simply offers this ported interface.

The sophisticated iAPX 86 and 88 customer is looking for the robust operating system capabilities of the iRMX 86 Operating System. This customer knows the advantages of the full functionality available with the O.S. They recognize the need to be compatible with Intel's next generation hardware and software.

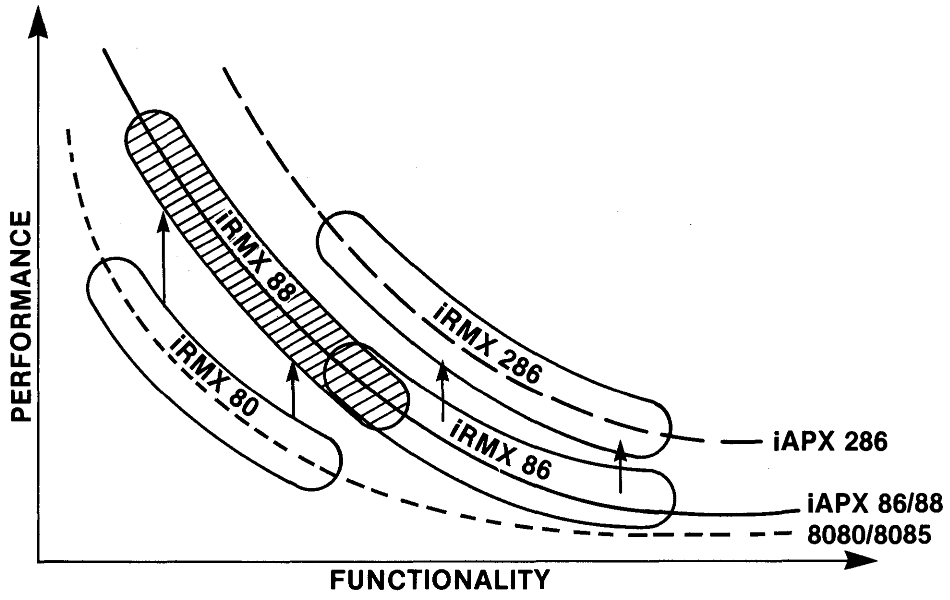


Figure 5.3
VLSI / Application Curve

Please examine Table 5.1 for the significant family distictions between the iRMX 80, iRMX 88, and iRMX 86 Operating Systems. The flow-diagram shown in Figure 5.4 provides a fundamental, thumb-nail sketch of determining customer needs. Customers can immediately prototype their application with iSBC products, be into production sooner, and move to their board-level design when sales volumes warrant the change.

| ATTRIBUTE | iRMX 80 | iRMX 88 | iRMX 86 |
|---|---|---|---|
| Sold to component customers | yes | yes | [✓ yes] |
| ✓ Object-oriented architecture | no | no | yes |
| Configurable | yes | yes | yes |
| Interactive Configuration | yes | yes | no |
| (P)ROM'able | yes | yes | yes |
| Memory size of O.S. (Kbytes) | .6-12 | 1.2-24 | 14-190 |
| Multiprogramming | no | no | yes |
| Multitasking | yes* | yes* | yes* |
| Memory management (Kbytes) | 64 | 128 | 1000 |
| Mailboxes | 255 | 255 | 64K* |
| Semaphores | none | none | 64K* |
| Regions | no | no | yes |
| Interrupt management | yes | yes | yes |
| Error management | direct | direct | hierarchical |
| Device independent I/O | no | no | yes |
| File directory structure | flat | flat | hierarchical |
| Double buffering | no | no | yes |
| Device driver support | iSBC 201,202, 204,206 | iSBC 208,220, 215/218 | iSBC 204,206, 215/218 |
| ✓ Human interface | no | no | [✓ yes] |
| Command line interpreter | yes | yes | yes |
| Instant-on | yes | yes | yes |
| Universal Development interface | no | no | yes |
| ✓ High-level language translators | no | no | [✓ yes] |
| Offered in silicon | no | no | yes |
| Future growth path for customers | iRMX 88 | iAPX 86/88 compatible | iRMX 286 |

* Number limited by avalilble RAM space

TABLE 5.1
iRMX FAMILY DISTINCTIONS

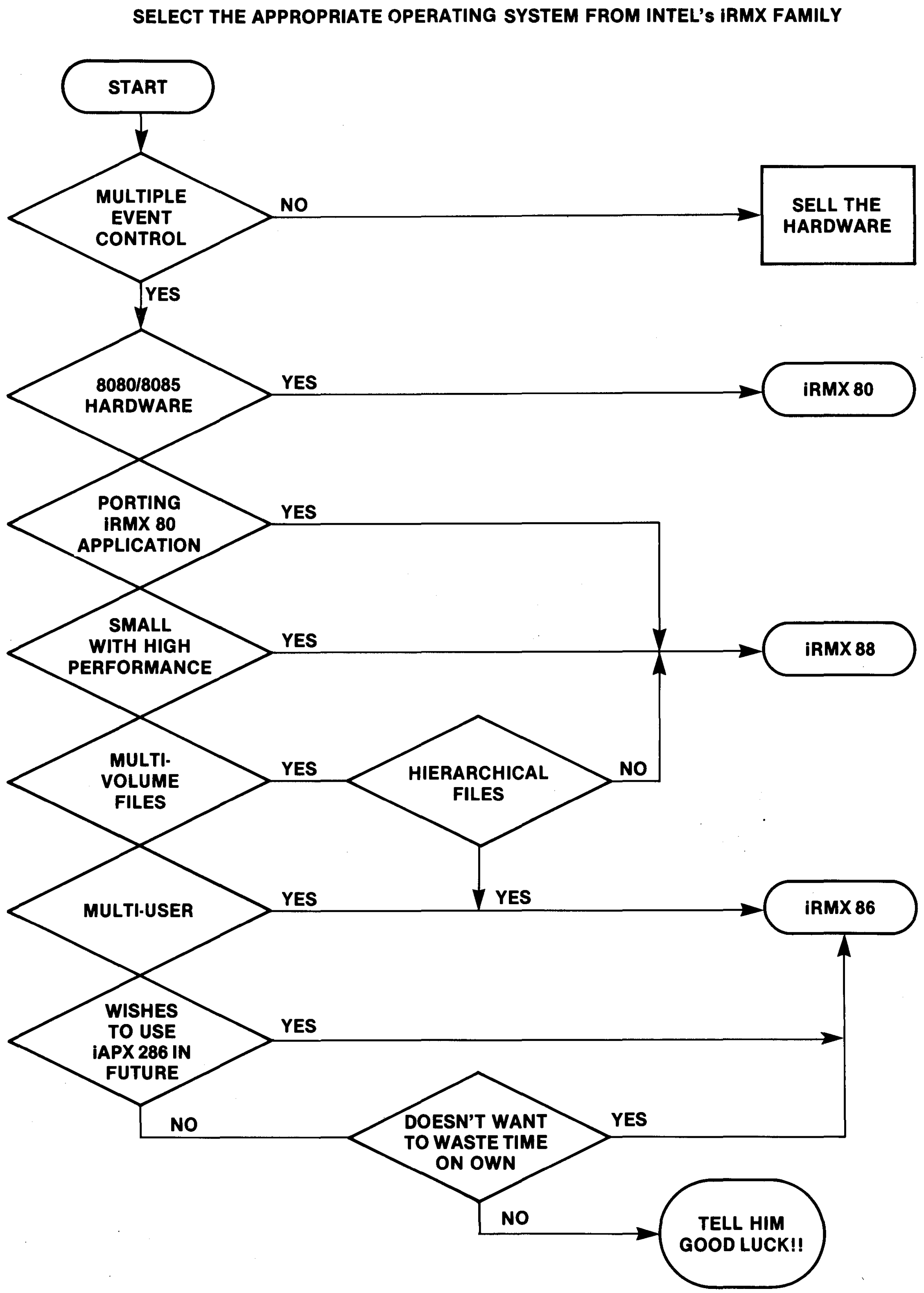


Figure 5.4
EXECUTIVE DECISION DIAGRAM

KEY APPLICATIONS

Applications for iRMX 86 cross the complete spectrum from Industrial Automation to Business DP. Industries and applications are sited to help you zero in on those opportunities!

iRMX 86 KEY APPLICATIONS

| MARKET SEGMENT | PRODUCT / APPLICATION |
|---|---|
| Telephone Networks | Stored Program PABX<br>Telephone Monitoring And Control<br>Network Testing |
| Process Control/Factory Automation | Data Acquisition Systems<br>Factory Floor Data Collection<br>Manufacturing Systems |
| Engineering And Scientific | Simulation Systems<br>Laboratory Data Acquisition |
| Medical Electronics | CAT Scanners<br>Ultrasound<br>Stress Test |
| Business Data Processing | Small Business Computers<br>Data Base Management Systems<br>Intelligent Terminal Clusters |
| Business Office Equipment | Word Processing Systems<br>Teleconferencing Products<br>Electronic Mail Systems |
| Financial Autotransaction Systems | Automatic Teller Central Unit<br>Electronic Funds Networks<br>Point Of Sale Systems |

# 6 Competition

## Overview

iRMX 86 stands several notches above the competition! Against the real-time systems it offers state-of-the art technology. Against the batch systems it satisfies both the real-time needs of your customers and in a background mode can satisfy his batch needs. Against the multiprogramming systems, it is comparitively inexpensive!

The comparison charts that follow are broken up into the various resource management facilities.

COMPARATIVE CHARTS

PROCESSOR MANAGEMENT:

| | iRMX 86 | UNIX | CP/M | RSX-11M | RT-11 | MTOS-86 |
|---|---|---|---|---|---|---|
| SCHEDULING | Real-time | Multi-program | Batch | Real-time | Real-time | Real-time |
| MULTITASKING | Yes (64K) | No | No | Yes (64K) | No | Yes (4K) |
| PRIORITY LEVELS | 255 | None | None | 250 | None | 255 |
| MULTIPROGRAMMING | Yes (64K) | Yes (64K) | NO | Yes (64K) | Foreground /Background | No |
| MULTIPROCESSOR | No | No | No | No | No | Yes (16) |
| MULTIUSER | No | Yes (4) | No | Yes (4) | No | No |
| INTERRUPT MANAGEMENT | Yes | No | No | Yes | Yes | Yes |
| ERROR MANAGEMENT | 4 levels | Yes | No | Yes | Yes | No |
| POWERFAIL PROTECT | No | No | No | Yes | No | No |

MEMORY MANAGEMENT:

| | iRMX 86 | UNIX | CP/M | RSX-11M | RT-11 | MTOS-86 |
|---|---|---|---|---|---|---|
| DYNAMIC | Yes (1MB) | Yes (64K) | No | Optional | No | Yes (64K) |
| # of MEMORY POOLS | 64K | None | None | 8 | None | 32 |
| COMPLETELY MEMORY RESIDENT | Yes | Yes | Yes | Yes | No | Yes |
| APPLICATION LOADER | Yes | Yes | Yes | Yes (11S: No) | Yes (RT$^2$: No) | No |
| BOOTSTRAP LOADER | Yes | Yes | No | Yes | Yes | No |
| (P)ROM'able | Yes | No | No | No | No | Yes |

DEVICE MANAGEMENT:

| | iRMX 86 | UNIX | CP/M | RSX-11M | RT-11 | MTOS-86 |
|---|---|---|---|---|---|---|
| CONCURRENT I/O | Yes | Yes | No | Yes | Yes | Yes |
| I/O BUFFERING | Yes | Yes | No | Yes | Yes | No |
| REENTRANT I/O | Yes | Yes | No | Yes | Yes | Yes |
| ASYNCHRONOUS I/O | Yes | No | No | Yes | Yes | Yes |
| SYNCHRONOUS I/O | Yes | Yes | Yes | Yes | Yes | No |
| DEVICE INDEPENDENT I/O | Yes | Yes | Yes | Yes | Yes | Yes |
| DEVICE ALLOCATION | PLANNED | | No | Yes | No | No |
| Max. # of DRIVERS | 64K | | 256 | 256 | 16 | 256 |

DATA MANAGEMENT:

| | iRMX 86 | UNIX | CP/M | RSX-11M | RT-11 | MTOS-86 |
|---|---|---|---|---|---|---|
| FILE SUPPORT: | | | | | | |
| 1. SEQUENTIAL | Yes | Yes | Yes | Yes | Yes | Yes |
| 2. INDEXED SEQ. | No | | No | Yes | No | No |
| 3. DIRECT ACCESS | Yes | Yes | Yes | Yes | Yes | Yes |
| SWAPPING | No | | No | Yes | No | No |
| OVERLAYS | Yes | Yes | Yes | Yes | Yes | No |
| HIERARCHICAL DIRECTORIES | Yes | Yes | No | No | No | No |
| STREAM FILES | Yes | Yes | No | No | No | No |
| MAILBOXES | Yes | No | No | No | No | No |
| CRITICAL REGIONS | Yes | Yes | No | No | No | Yes |
| HOST SYSTEM FOR DEVELOPMENT | Yes (With UDI) | Yes | Yes | Yes (11S: No) | Yes ($RT^2$: No) | No |

## SPECIFICS ABOUT THE COMPETITION

### UNIX (XENIX) vs iRMX 86

o UNIX is expensive! From Western Electric, OEMs pay initially 78,000 which includes no support and is coded only for PDP 11s! No royalty buyout agreement has ever been signed, but the negociations start in the millions according to the Unix Western Electric Product Manager. From Microsoft the pricing is as follows:

o XENIX pricing is on a sliding scale such that on every $500,000 paid in royalties, you get a new price break! The pricing is also adjusted to the number of users that UNIX is preconfigured for. Four is more that three, etc.

o The initial license fee for a 4-user system is $2,000.

| UNITS | 11-526 | 527-1240 | 1241-2240 | 2241-4740 | 4741+ |
|---|---|---|---|---|---|
| ROYALTY | $950 | $700 | $500 | $200 | $80 |

o UNIX is for timesharing environments only! It cannot be used for communications, scientific applications or process control! No interrupt management is provided!

o UNIX is not (P)ROM'able! It must have a mass storage device.

o UNIX is not configurable! For the 8086 it is estimated to be 55K - 60K bytes.

o UNIX is developed by Bell Labs and supported by Microsoft, a software house with '79 revenues of $2.5 million.

o UNIX will not support multitasking nor task to task communication.

o UNIX will support up to 8 terminals (4 is suggested on the 8086,Z8000, and 68000), but it is bought from Microsoft already configured. (If you go from 4 terminals to 5, Microsoft has to configure it!).

o UNIX requires special memory protection hardware for the 8086 or the MMU for the Z8000.

o UNIX is well known in the college environment and runs on VAX, all PDP-11s including LSI-11/23, and will be offered by Microsoft for 68000 (Q281), 8086 (Q181), and Z8000 (Q181).

XENIX is offered by
Microsoft
Bellevue, Washington

UNIX is offered by
Western Electric

## CP/M VS RMX 86

- o CP/M is a file management system not an operating system. Digital Research markets CP/M as "a monitor control program for system development." It is equivalent to ISIS-II, the Intel development system monitor.
- o CP/M is a single user, single program system. Only one program can execute at once.
- o CP/M is not Real-Time. It cannot be used for process control, scientific applications, or communications.
- o CP/M is not prommable. It requires a mass storage device.
- o CP/M will not support multitasking nor task to task communication.
- o CP/M offers an attractive ISIS II compatible mass storage file management system for a one user system running on 8080, and Z80.
- o CP/M, due to its low single usage price, has become the "defacto" 8-bit standard. Many small software houses have written 8-bit software packages that run on top of CP/M 80 including high level languages (FORTRAN, COBOL, PASCAL, BASIC), applications packages (accounting, database management), and utility packages (text editors).

  How long it will take these vendors to convert to CP/M 86 and to the 8086 processor is not clear! Digital research expects to be ready by December. Remember that the file system is not easily expanded to support new device controllers.
- o Initial license including 50 royalties: $3,000

| | |
|---|---|
| UNITS | 1 - infinity |
| ROYALTY | $60 |

  Buyout $50,000

  CP/M for a single user is $150
- o Digital Research plans to introduce MP/M, a multiple user version of the same operating system, some time in 1981.
- o CP/M has a good track record as is known to be a reliable package with good vendor "HOTLINE" support.

CP/M is offered by
Digital Research
Pacific Grove, CA

## MTOS 86 VS RMX 86

- o MTOS 86 is in a class comparable to iRMX 88, it is an 8-bit system ported to a 16-bit system.

- o Pricing is such that support is explicitly excluded. As a result, IPI, whose annual revenues are less than $1 million, are only addressing the 1-time buyer.

| Pricing (including source) | |
|---|---|
| MTOS 86 | $4500 |
| FLOPPY DRIVER | 750 |
| LINE PRINTER DRIVER | 750 |
| ISIS-II FILE SYSTEM | 1500 |
| NETWORKING | 4000 |
| MULTIPROCESSOR | 4000 |

- o Simple, "static" low-end real-time system. The configuration process is difficult and time consuming!

- o Does not offer a man-machine interface! Can not be used in an interactive environment, just dedicated computer controller applications.

- o Does not support memory management of memory pools greater than 64K.

- o Does not offer an application loader, therefore, there is no way of loading tasks from a mass storage device and executing them.

- o File system can never support new peripheral technology including 8272, iSBC 208, iSBC 215, iSBX 218, and iSBC 220.

- o Multiprocessing support for 16 iAPX 86 or iAPX 88 processor boards running on the MULTIBUS.

- o Network support for 255 MTOS 86 nodes.

MTOS 86 is offered by
Industrial Programming Incorporation
Jericho, N.Y. 11753

## RSX-11M/RSX-11S VS RMX 86

o DEC's RSX-11M requires a PDP-11 system with a disk! It's the only solution for a "high roller" OEM!

o DEC's value added is not the silicon like Intel's! To maintain high revenues, their software prices are steep! - First product RSX-11M CAT A, full support $3,450.

o RSX-11M is a tried and true architecture just like our 8080. Innovative OEMs that require new solutions and need the room for product growth cannot afford an old solution.

o System extensions requires source that is not offered in the basic product.

o RSX-11M supports multitasking, multiprogramming, and does offer a rich set of languages including BASIC, COBOL, FORTRAN, and APL.

o RSX-11S is a smaller version than RSX-11M. It does not support application development. It still requires a disk, and a PDP-11 system, but the price is still a factor of 2 higher than a subset of the iRMX 86 Operating System that will support languages!

o Some of our customers have said that context switch times in the order of 5 msec are not uncommon.

## RT-11 VS RMX 86

o RT-11 is not multitasking! It supports 2 tasks with 2 priorities. (The iRMX 86 Nucleus supports 64K tasks and 256 priorities). It is difficult to use RT-11 in asynchronous applications like communications, multi-terminal applications, and process control.

o Only 1 foreground task responsive to interrupts can run. If you want to load a new foreground task, it can only be loaded by issuing the request on the system console which is blocked until the background task has completed!

o RT-11 is a stable product that has no planned additions. It runs on LSI-11s thru PDP 11/70s.

o RT-11 is a one user system with high level language support including FORTRAN, BASIC, and APL.

o RT-11 is more expensive than iRMX 86! - First license is "CAT A" full support at $2,760 the cheapest other 2+ licenses are "CAT D" no support options at $486/copy.

Pricing on next page

PRICING

o Using the discount schedule, on "CAT D" for both products:

| License only | 1-9 | 10-24 | 24-49 | 50-99 | 100-199 | 200-399 | 400-599 |
|---|---|---|---|---|---|---|---|
| Price RSX-11M | $2200 | $1496 | $1298 | $1056 | $968 | $902 | $858 |
| Price RT-11 | $489 | $333 | $289 | $234 | $215 | $200 | $190 |

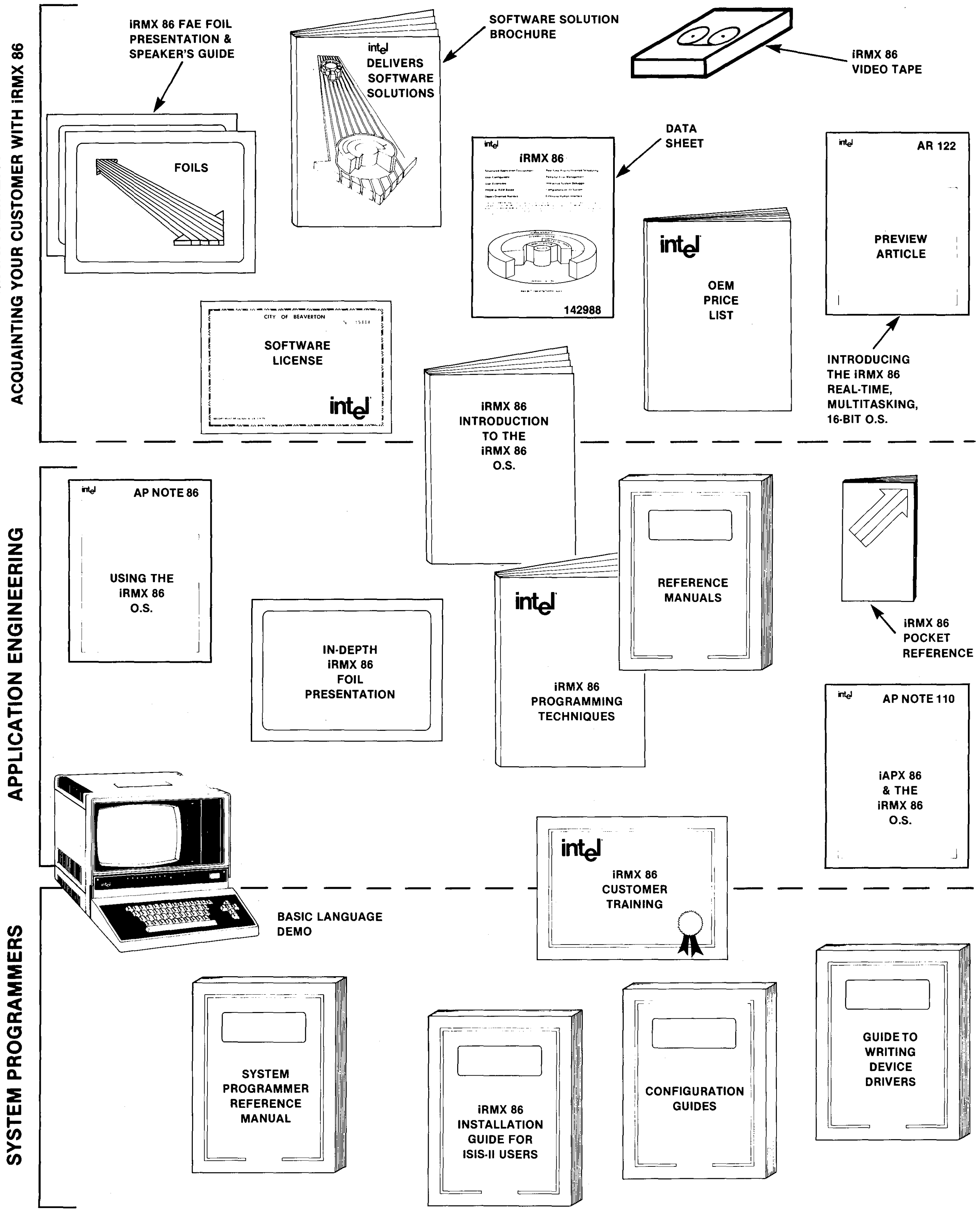
RSX-11M, RSX-11S, and RT-11 is offered by Digital Equipment Corp. Maynard, Mass.

## COMPETITIVE SUMMARY

"GOTCHAS"

| | | iRMX 86 OPERATING SYSTEM | COMPETITION |
|---|---|---|---|
| ✓ | 1. | Tasks can schedule other tasks | Nobody here!! |
| ✓ | 2. | Dynamic task creation and deletion | Only MTOS-86 comes close |
| ✓ | 3. | Multiprogramming - Job isolation | Only the minis do it |
| ✓ | 4. | Math coprocessor support | DEC has a slower one that is not IEEE compatible |
| ✓ | 5. | Disk file support for the iSBC 204, 206, and 215/218 | Not even considered by MTOS-86 |
| ✓ | 6. | (P)ROM'able / Bootstrap support | DEC has bootstrap, not (P)ROM'able; MTOS-86 has no bootstrap |
| ✓ | 7. | Real-Time system | CP/M is batch only |
| ✓ | 8. | Command line interpreter | User extendable |
| ✓ | 9. | iSBC and Component support | Flexibility designed into standard package |

## DOCUMENTATION AVAILABLE


MANAGEMENT
ACQUAINTING YOUR CUSTOMER WITH iRMX 86
iRMX 86 FAE FOIL PRESENTATION & SPEAKER'S GUIDE
FOILS
SOFTWARE SOLUTION BROCHURE
intel DELIVERS SOFTWARE SOLUTIONS
iRMX 86 VIDEO TAPE
DATA SHEET
iRMX 86
142988
AR 122
PREVIEW ARTICLE
INTRODUCING THE iRMX 86 REAL-TIME, MULTITASKING, 16-BIT O.S.
OEM PRICE LIST
SOFTWARE LICENSE
iRMX 86 INTRODUCTION TO THE iRMX 86 O.S.
APPLICATION ENGINEERING
AP NOTE 86
USING THE iRMX 86 O.S.
IN-DEPTH iRMX 86 FOIL PRESENTATION
iRMX 86 PROGRAMMING TECHNIQUES
REFERENCE MANUALS
iRMX 86 POCKET REFERENCE
AP NOTE 110
iAPX 86 & THE iRMX 86 O.S.
BASIC LANGUAGE DEMO
iRMX 86 CUSTOMER TRAINING
SYSTEM PROGRAMMERS
SYSTEM PROGRAMMER REFERENCE MANUAL
iRMX 86 INSTALLATION GUIDE FOR ISIS-II USERS
CONFIGURATION GUIDES
GUIDE TO WRITING DEVICE DRIVERS

## DEMOS

Tiny Basic Demo: Shipped with the product. This demo contains a complete configuration using the Nucleus, Debugger, and the Terminal Handler.

## INTRODUCTION PLANS

| | |
|---|---|
| November, 1980: | Release 2 Shipment. This release will provide a configurable Nucleus, an Application Loader, and a Bootstrap Loader. In addition, many of the problems detected in the first release will be corrected. |
| May, 1980: | Release 3 Shipment. This release will complete the product by providing the remaining system layers: the Extended I/O System, and the Human Interface. In addition, major improvements will be made to the system throughput and system memory requirements. In general, most memory and timing parameters will be improved by about 20%. Added features include I/O port configurability and cascaded interrupts. |
| December, 1981: | Most of the functions of the Nucleus will be integrated into a new silicon device along with the various timers and interrupt controllers required by the operating system. This new 'operating system component' will offer the component customers a new opportunity to improve their performance while decreasing their costs and PCB realestate. |

## KEY FACTORY CONTACTS

| | | |
|---|---|---|
| PRODUCT MARKETING | | |
| iRMX 86 Product Manager | Peter Wolochow | (503) 640-7423 |
| Software Marketing Manager | Dave Cloutier | (503) 640-7133 |
| Product Marketing Manager | Sam Bosch | (503) 640-7131 |
| Marketing Manager | Bob Brannon | (503) 640-7082 |
| TECHNICAL SUPPORT | | |
| Software Product Support Manager | Roger Jennings | (503) 640-7259 |
| Applications Engineer | George Heider | (503) 640-7140 |
| iRMX 86 HOT LINE !!!!! | | (503) 640-7650 |
| MARKETING SERVICES | | |
| Contract Adminstration | Mike Morrison | (602) 869-4596 |
| Royalty Adminstration | Sherri Morningstar | (408) 987-8810 |
| Marketing Product Administrator | Dorothy Tannahill-Moran | (503) 640-7105 |
| Product Scheduler | Patty Patzke | (503) 640-7107 |
| CUSTOMER MARKETING | | |
| U.S. Sales Development Manager | Vinod Mahendroo | (503) 640-7113 |
| Southwest Marketing Coordinator | Bob Wilkinson | (503) 640-7109 |
| North/Mid-West Marketing Coordinator | Fred Langner | (503) 640-7124 |
| Atlantic/Eastern Marketing | Bob McFadden | (503) 640-7565 |
| New England Marketing Coordinator | Bob Critchlow | (503) 640-7524 |
| Distributor Marketing Coordinator | Jim Alexander | (503) 640-7116 |
| International Marketing | Steve Chase | (503) 640-7158 |

# 8 Ordering Information

The iRMX 86 Operating System package is a licensed software product that is Drop-Shipped only.

## LICENSED PRODUCT

An order for either iRMX 86 KIT ARO or iRMX 86 KIT BRO (The ARO is for single density diskette versions, and the BRO is for double density diskette versions) requires the license number and quote number. The license number comes from the prenumbered Master Software License Agreement as outlined in the next sections. Licensing instructions also appear in the back of current OEM and DISTI price books. This new Master Software License supercedes all previous license agreements, and need be signed by the customer only once.

## PRODUCT PRICING

KEY
N = NEW PRODUCT
↓ = PRICE DECREASE
↑ = PRICE INCREASE

**OEM MICROCOMPUTER SYSTEMS**

| PROD. RESP. | PROD. CODE/ PART NO. | DESCRIPTION | SUGGESTED RESALE PRICE QUANTITY 1-9 | 10-24 | DISTRIBUTOR(7) UNIT PRICE |
|---|---|---|---|---|---|
| B | RMX 86-Kit | Order replacement part RMX 86-Kit ARO or RMX 86-Kit BRO. | | | |
| B | RMX 86-Kit ARO | Realtime multiprogramming operating system for Intel iSBC-86 Single Board Computers contained on single density and double density ISIS-II compatible diskettes. iRMX 86 contains a Nucleus, Basic I/O System, Terminal handler, Debugger as well as Device Drivers for the iSBC 204 and iSBC 206 Disk Controllers. A comprehensive Documentation Package includes iRMX 86 Nucleus, Terminal Handler, and Debugger Reference Manual; iRMX 86 I/O System and Loader Reference Manual; iRMX 86 System Programmer's Reference Manual; iRMX 86 Pocket Reference; Guide to Writing Device Drivers for the iRMX 86 I/O System; iRMX 86 Configuration Guide for ISIS-II Users; iRMX 86 Installation Guide for ISIS-II Users; Introduction to the iRMX 86 Operating System; and iRMX 86 Programming Techniques. Price includes software license fee.<br><br>iRMX 86 is provided with an iSBC-957A Execution Vehicle, and credit for one iRMX 86 customer training course. (Valid for 6 months.) | 7500.00N | — | 5250.00N |
| | | | (DROP-SHIP ONLY) (Requires Master Software License)* (Class II) Royalties payable directly to Intel. Royalty Price: 1-24 300.00; 24-49 225.00; 50+ 160.00 | | |
| B | RMX 86-Kit BRO | Same as RMX 86-Kit ARO except BRO is double density. Price includes software license fee. | 7500.00N | — | 5250.00N |
| | | | (DROP-SHIP ONLY) (Requires Master Software License)* (Class II) Royalties payable directly to Intel. Royalty Price: 1-24 300.00; 24-49 225.00; 50+ 160.00 | | |
| B | RMX 86-Kit AWX | Warranty extension program for the iRMX 86 Operating System. Warranty coverage includes one year of updates, software problem report service, periodic newsletter, and technical support. | 2000.00N | — | 1850.00N |
| | | | (DROP-SHIP ONLY) Requires verification of licensed purchase of RMX 86 from Intel. | | |
| B | RMX 86-Kit BWX | Same as RMX 86-Kit AWX except BWX is double density. | 2000.00N | — | 1850.00N |
| | | | (DROP-SHIP ONLY) Requires verification of licensed purchase of RMX 86 from Intel. | | |

## LICENSE AGREEMENT PROCEDURES

Here are some of the details you will need to know:

**CLASSIFY**

The price book gives all the information you need about each of the software products; including --

-Class I = end user
-Class II = OEM
-Class III = OEM (Pass-through)
-Initial license fee
-Additional royalties required (if any)

Become familiar with your customer's rights for each class of software

**SOURCE**

Source code licenses are handled with a separate license agreement. Contact the product marketing manager with any inquiries.

**BLANKET AGREEMENT**

Existing blanket agreements should be converted over to the new form the next time your customer orders a software product. All new customers or new orders from existing customers not already on a blanket license, will require the new agreement.

**SHIPMENT**

Software will not be shipped until we have a signed license agreement from the customer. (The quote number procedure will still be used and is available from Mike Morrison, in Deer Valley.) Once we have a signed Master Software Licence Agreement, subsequent orders from the same customer will ship without delay.

**CUSTOM BOARDS**

The license agreement requires our software products to be used only on Intel products such as Intellec$^{TM}$ systems, and iSBC$^{TM}$ boards. If your customer wants to build custom boards using Intel components, we will issue a letter designating his board as a "licensed Product".

**ADDITIONAL COPIES**

The now obsolete, 'additional copy' price for software, is no longer authorized or applicable to any of the software products. Customers are now authorized to make seven copies, at a location, for each purchased product.

**NON-INTEL**

If your customer really requires using our software on equipment not manufactured or licensed by Intel, or on putting our software into OEM equipment which does not use Intel boards or components, contact the product marketing manager. We will require a different license agreement and we cannot provide warranty or support.

**DISTI'S**

Distributors (domestic) are not requested to monitor or assist in collecting royalties. They are only required to get software licenses signed, and registration certificates completed.

**ROYALTY PAYMENTS**

Royalty payments do <u>NOT</u> go through the normal sales system. Purchase orders and royalty payments should be sent to Royalty Administration, c/o Sherri Morningstar, SCIV - 105. All paperwork should identify the customer's license number and the respoonsible FSE's name.

Special considerations for licensing the iRMX 86 Operating System:

STEP PRICING

The iRMX 86 Operating System requires the customer to pay royalties for passing portions of the program through as part of an OEM product. The royalty schedule is based on step pricing so that no volume forecasts are required from the customer. Once every six months, we will ask the customer to report the total number of incorporations into OEM products (or products used internally). All portions incorporated into a single product count as one incorporation. We will maintain totals for each customer.

The first 24 units will be at $300 each. The next 25 units will be at $225 each. All subsequent incorportations will be at $160 each.

For example, if a customer has incorportated the selected portions of iRMX 86 into an OEM product and sold 58 of them, he would have paid:

| | | |
|---|---|---|
| 24 x 300 | = | 7,200 |
| 25 x 225 | = | 5,625 |
| 9 x 160 | = | 1,440 |
| TOTAL | | $14,265 |

The same formula would apply when the customer incorporates selected portions of iRMX 86 into their software for internal use.

NO DISCOUNT

We will not discount royalties. If you are negotiating a volume agreement, we will consider discounting the boards to be competitive. For example, consider 24 incorporations into an iSBC 86/12A based product:

| | |
|---|---|
| Board price | 2,405 |
| Royalty | 300 |
| TOTAL | $2,705 |

Assuming a competitive board quote of $2,200, we should bid:

| | |
|---|---|
| Board price | 2,250 |
| Royalty | 300 |
| TOTAL | $2,550. |

DPA's

Royalty payments will be counted for discount purposes in the DPA's (Discount Purchase Agreements).

ROYALTY

Royalty Administration and Collection will not be handled by the distributors. Our software license agreement is between Intel and the Customer.

# 9 Questions and Answers

1. When should I sell iRMX 86 features?

ANSWER: The iRMX 86 Operating System is ideal for those applications that require a stron, and complete set of features needed to support applications that depend on real-time performance and state of the art design and device support. The device independent I/O, and the multiple job environment provided by iRMX 86 allow applications like word processing and data processing to co-exist with applications like data collection and individual device control.

2. Can I sell the iRMX 86 Operating System to component customers?

ANSWER: YES!!! Any part of the operating system may be put in (P)ROM, makint it very easy for the customer to adapt his application to a custom configuration. Remeber: The best idea is to use an 8086 based iSBC product for prototyping, and move to the component design when the software has been tested.

3. What are the restrictions for component customers?

ANSWER: Releases 1 and 2 require that the system contain an 8253 (timer) and an 8259A (interrupt controller) configured at the same locations as specified on the iSBC 86/12A Single Board Computer. If the Terminal Handler is used in the system, an 8251A (USART) must also be part of the system, and be located as it is on the iSBC 86/12A. Release 3 will allow the customers to configure the addresses of the 8253 and the 8259A

4. Should I attempt to sell iRMX 88 to component customers before trying to sell iRMX 86?

ANSWER: ABSOLUTELY NOT!! The iRMX ** Operating System is well suited for those applications that do not require device independent I/O, and a multiple job environment. Without iRMX 86, the customer has no way of expanding beyond 128K of O.S. controlled memory, or designing ahead for the next generation of processors and device drivers. Furthermore, remember that component software is just around the corner, and will be based on the iRMX 86 Operating System Nucleus.

5. What is the UDI? -- What does it do for iRMX 86?

ANSWER: The Universal Development Interface provides a "software bus" or interface on top of which any standard compiler or translator can be built. By providing this interface, iRMX 86 (and several other Intel operating systems) can use all the programs written to conform with the interface. This will allow Intel and other software houses to provide a multitiude of languages that all run on all the new operating systems and in a very short period of time.

6. What languages can be used with iRMX 86?

ANSWER: PLM 86 and the 8086 Assembler can be used now. The only real restriction is that the procedure calls must conform to the PLM parameter passing conventions for the particular model of computation selected by the user. Other languages (including COBOL, FORTRAN, PASCAL, and BASIC) will be built to run on the UDI by Q2'81. These languages will be provided with an editor and the various linking and locating programs needed to do acutal development on a system using the iRMX 86 Operating System.

7. Can I sell source to all customers that ask for it?

ANSWER: TRY NOT TO SELL SOURCE. We beleive that the sofware that Intel has developed is VERY expensive and valuable. The standard Master License described in this manual does not include source. If the customer demands to have listings, we are prepared to discuss the possibility. Machine readable source is treated in a different manner. The point to remember is: Intel supports its software; Customers need to have source to ensure proper support or to relieve anxiety.

8. What is planned to follow Release 3 of the iRMX 86 Operating System?

ANSWER: The iRMX 86 Operating System will continue to be maintained and updated for many years. The current plan is to update the system at least once per year to remove any problems and to add new device drivers. A Release 4 is being planned to be introduced before the end of 1981 and to include new device drivers (iSBC 220, ect.), multiprocessing capabilities, and further performance and size improvements.

9. How do customers maintain their iRMX 86 Operating System?

ANSWER: The iRMX 86 Operating System is sold with one year of updates and newsletters included in the base price. Once this warranty period is over, the customer must exted the warranty by purchasing the appropriate 'WX' (Warranty Extension) product (ex: iRMX 86 AWX). Customers not under warranty will not receive new updates or factory support.

# 10 Self Evaluation

Can you answer these questions? If you can, you are doing well, and are ready to help solve your customer's problems with the iRMX 86 Operating System.

1. What are the major advantages of using the iRMX 86 Operating System?
2. What are the advantages of an object-oriented architecture?
3. What are the advantages of a device independent I/O system?
4. What are the major advantages of the layered structure of the iRMX 86 Operating system?
5. What are the major differences between iRMX 86 and the other multitasking operating system available?
6. How can iRMX 86 work for component customers?
7. How is the iRMX 86 Operating System configured for each application?
8. How can the operating system be loaded into the application hardware?
9. Which Intel iSBC boards and devices are directly supported by the iRMX 86 Operating System?
10. How is Intel software licensed?
11. Give one "GOTCHA" for iRMX 86 against each of: MTOS-86; UNIX; CPM-86; and RSX11.
12. What are the two ways an application can respond to interrupts using the iRMX 86 interrupt mechanisms?
13. What are the six objects supported by the iRMX 86 Nucleus?
14. How are user defined objects handled by the operating system?
15. What is the range of the interrupt latency imposed by the iRMX 86 Operating System?
16. What is the maximum size of a segment controlled by the iRMX Operating System?
17. What is the current price for all layers of the operating system?
18. What is the royalty schedule for iRMX 86 derivative works?
19. How does the iRMX 86 Nucleus provide application independence?

# Appendix A

Understanding some of the fundamental concepts will help you answer this question in your own words. This section identifies the resource management required for real-time, batch, and multiprogramming. The management of CPU, I/O, and memory resources is critical to the efficient execution of programs.

A real-time operating system can manage the asynchronous events in a timely fashion so that control can be effective. TTL or logic controllers do not manage independent I/O and are not part of this discussion. The following tables provide significant detail as to the batch and multiprogramming environments found in the reprogrammable market place. In a simple batch system, the memory resource is allocated totally to a single program. Thus, if a program does not need all of memory, a portion of that resource is wasted. There are many batch operating systems still in use today, however, for microcomputer applications the multiprogramming operating system will be the better choice. Multiprogramming allows more than one job to reside in memory and execution is overlapped with I/O.

Operating systems do nothing more than manage system resources.

Resource Management In A Nutshell

1. Processor management to determine what code gets CPU time.
2. Memory management to allocate and deallocate which memory the system is using.
3. Device management to facilitate peripheral communication between A/D and D/A, consoles, mass storage devices, etc.
4. Data management to aid in data manipulation.

Now these resources are managed leads to operating system classifications. In trying to summarize operating systems, the best distinction can be made using target markets. John Zarella's book Operating Systems Concepts And Principles published by Microcomputer Applications, P.O. box E, Suisun City, CA 94585 goes into much more detail, however his market identification and O.S. classification, as shown in Table A.1, is one of the best.

| O.S. TYPE | MARKETS SERVED | NEEDS TO BE SATISFIED |
|---|---|---|
| REAL-TIME | Industrial Control<br>Communications PABXs<br>Telecommunications<br>Military Command Control<br>Interactive Graphics<br>Simulation (Including Flight) | Ability in real-time to accept parameters that affect output.<br>Timely response to external stimulus |
| BATCH | University Computer Center<br>Payroll<br>Home Computer System<br>Filter Design<br>Large Scientific # Crunching<br>Forecasting<br>Statistical Analysis | 1 program to execute at once.<br>Overlapping program execution with input of the next program. |
| MULTI PROGRAMMING | On-Line Database (airline reservation)<br>Time-Sharing Systems<br>Word Processing / Text Editing<br>Modeling<br>CAD (computer aided design) | Multiple programs to appear to execute simultaneously<br>Quick console response |

Table A.1 - Markets and Operating Systems

## A Real-Time Application

The heart of the real-time, multitasking system is the Nucleus which controls and manages all system resources, e.g. memory, I/O, and CPU.

Real-time means that the executive is capable of responding to outside events in such a manner as to control or observe the phenomena with the events. Consider the events that occur as chemicals are mixed in the correct proportions to create a final product. There are valves opening and closing as the flow of material is measured. Motors are started, accelerated, and stopped when the reaction is complete, usually monitored through temperature probes (see Figure A.1).

Even a simple mixing process may have more than one event occurring at the same time. Many customers have found it worthwhile to use microcomputers to control such processes. They need software which can perform several tasks or functions at the same time; i.e. multitasking.

As you may have noticed in Figure A.1; you want the right proportion of ingredients to be introduced as the stirrer turns steadily. The temperature measurement is important for determining the final reaction point. The mixing-vat valve can not open until the reaction is completed. If the valve were unable to open, then an alarm must be sounded and the process halted. As can be visualized, a pattern of events must occur in a proper sequence with exceptions (alarms) occuring as needed. This leads to prioritization of the work (tasks) to be completed.

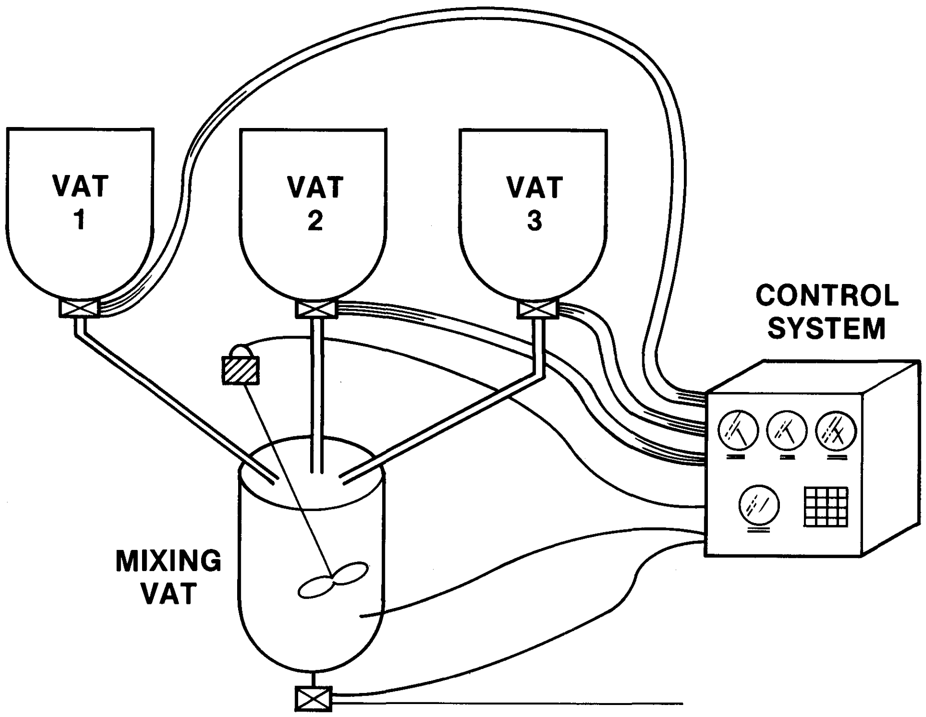


Figure A.1

Simple Batch Process

The previous discussion represents the fundamental control software a customer must provide for this microcomputer application. The foundation software provides a structured environment that is sufficiently flexible to meet the real-time requirements. This structured environment permits the user to segment his problem into tasks related to a function to be performed or a device to be sampled. The most efficient use of the CPU is when the executive can manage the concurrent execution of tasks.

Tasks are ranked (prioritized) as to importance so that the highest-ranked, ready-to-run task is executing. Alarm conditions, as described previously, are prioritized such that they preempt tasks appropriately. Since events (interrupts) occur randomly, the executive must synchronize the event (see Figure A.2) with a task and consider its priority relationship. When one task completes its operation, e.g. closes the valves in the example, it may want to start the temperature monitoring task. This is a form of task to task communication.

Why all this discussion? For this reason, multitasking executives require some thought and can be error-prone user solutions. The iRMX 88 Executive provides the foundation software for the entire application. Based on the field-proven and reliable architecture of the iRMX 80 Executive system, the iRMX 88 Executive minimizes the risk and provides a firm foundation for the application development. This can save at least 25% of the software development and maintenance costs.

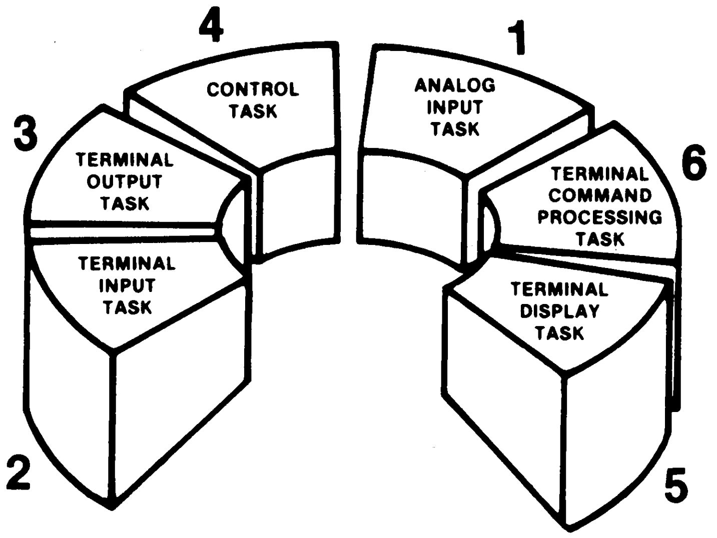


Figure A.2
Prioritized Tasks

## A MULTIPROGRAMMING EXAMPLE.

A system which is built for multiprogramming must consider the same resource management problems as a real-time nucleus does for a dedicated computer application. The multiprogramming system is more "people" oriented than machine real-time control, and performs more than one job at a time. The system is more complex depending upon the functions to be performed, see Table A.2. For the discussion we will use the system depicted in Figure A.3.

| RESOURCE | DEDICATED COMPUTER | REPROGRAMMABLE COMPUTER | | | |
|---|---|---|---|---|---|
| | REAL-TIME | BATCH | MULTIPROGRAMMING | | |
| | | | SYSTEM CONTROL | MULTI TERMINAL | MULTIUSER TIMESHARE |
| PROCESSOR SCHEDULING | External premptive priority scheduling.<br><br>Generally multitasking<br><br>Scheduler passive causes events to change task execution. | NONE - only <u>one</u> program runs at a time.<br><br>Assures maximum turnaround. | Assures maximum thruput.<br><br>Program idle causes system to switch program execution. | Scheduling assures interactive programs have highest priority.<br><br>One terminal cannot block another. | Scheduling assures interactive programs have highest priority.<br><br>One user cannot block another. |
| DEVICES | Time critical external devices supported through direct interupt management. | No time critical device management.<br><br>Often supports I/O spooling. | Supports buffered I/O.<br><br>Device independent I/O. | No time critical device management.<br><br>System geared to support fast man-machine console service. | |
| MEMORY | Minimal memory clean-up. | Load next program over previous one! | Memory clean-up guarenteed to load next program. | | Drives memory protection hardware for user isolation.<br><br>Memory clean-up guarenteed to load next program. |
| DATA | Generally task to task communication. | Generally supports file management. | Good file management. | Generally supports data base.<br><br>Good file management. | Good file management system.<br><br>File protection from user to user.<br><br>Usually data base support. |

Table A.2 - Operating System Resource Management

The business office is one application area that is growing by leaps and bounds. Many OEM customers have found it worthwhile to use microcomputers to control word processing equipment. In general, it is a multitasking environment where multiple terminals are providing input or displaying interactive information. As can be seen, there are communications to be controlled, input strings to be inserted into files, and print-outs to be obtained. All of these can be represented as real-time events. One way to manage this work is thru time-slicing as shown by Figure A.4, but the eventdriven system is actually more effective in managing resource alloction.

The clever thing about the flexibility in the iRMX 86 real-time operating system is that it could be extended to support time-slicing. However, a time-slicing, or batch system can not be converted to process real-time events readily.

The Figure A.3 actually shows more than word processing occurring. The OEM who built this system has expanded it to include a back-office business programming environment. The OEM has added more terminal control, a COBOL compiler for report generation, and some general business application programs, e.g. general ledger. This required top notch resource management with functionality found in the iRMX 86 Operating System. The OEM started the reprogrammable need, and may move to the iAPX 286 version when it becomes profitable.

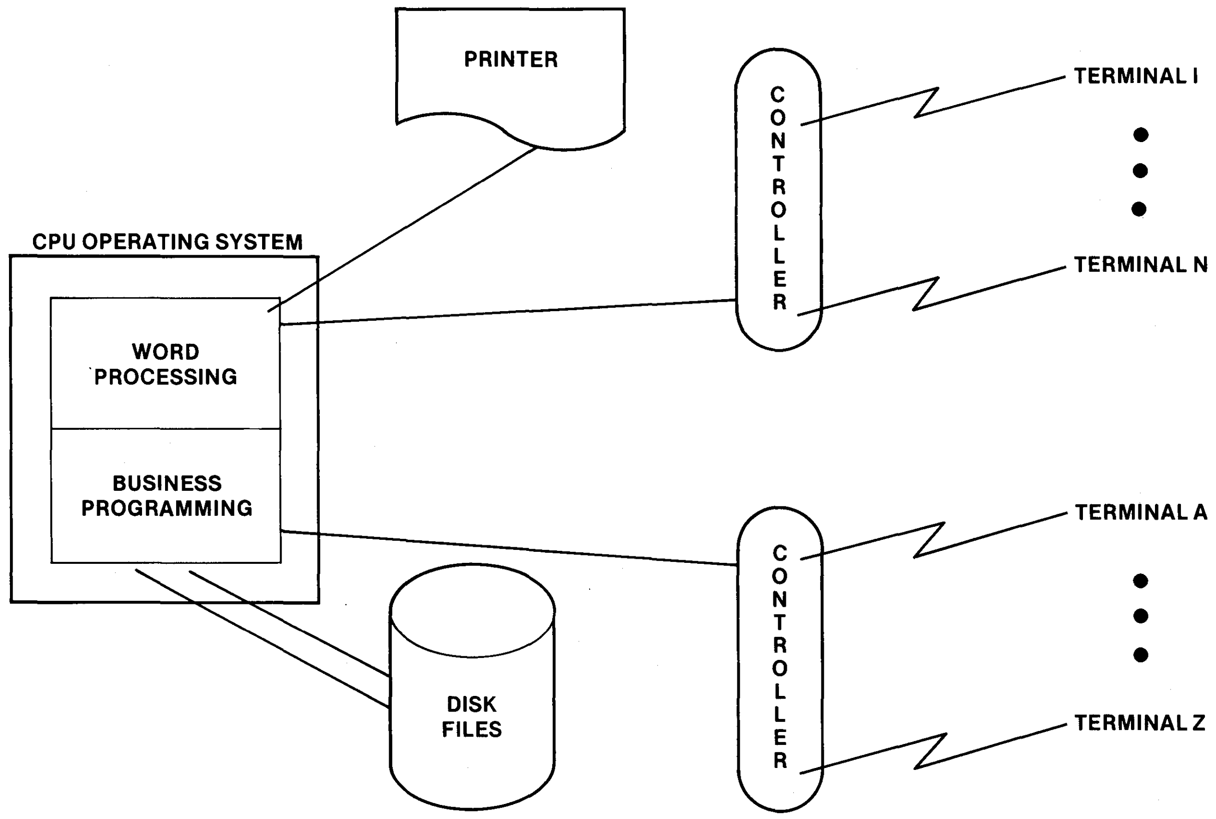


Figure A.3
Multiprogramming Application

## A SHORT QUIP

**TIME SLICING** **EVENT DRIVEN**

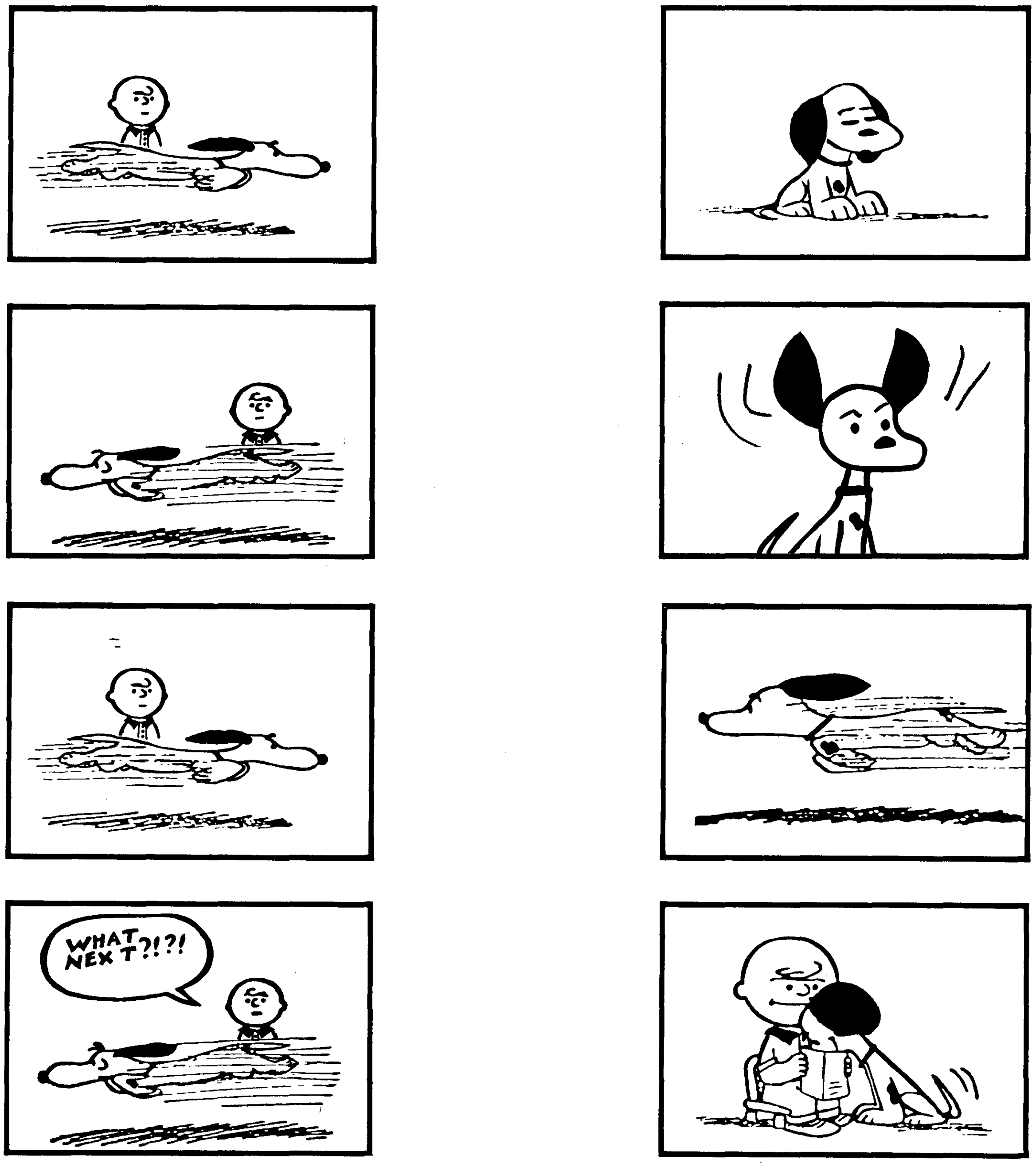


Figure A.4 - Timeslicing vs Event-Driven

In our timeslicing cartoon, poor iSnoopy is continuously torn between one thing and the next. It seems that most of his energy is devoted to switching between the tasks, rather than to getting the job done. When event-driven, our iSnoopy switches activities only when alerted by a need to do something now. What satisfaction.

An operating system is the foundation for all code written for the application. In a word processing application, this foundation includes a human-interface console plus mass storage support. The high-level language (COBOL) assumes an operating system because the language allows the user to reference data or files stored on mass storage devices like disk, or bubbles.

Operating systems are thus nothing more than a black box (see Figure A.5) extension to the computer architecture. Operating systems "systems calls" are viewed in a similar manner to processor instructions! Migration is significantly improved because of the standard "software bus" approach used by Intel. The software bus provides a mechanism to interface applications, or compilers in the same manner. Even if the hardware changes interfaces, the bus remains the same. It is the software connection, or bridge, to future operatng systems and hardware. The iRMX 86 Operating System is tomorrow's technology available today!

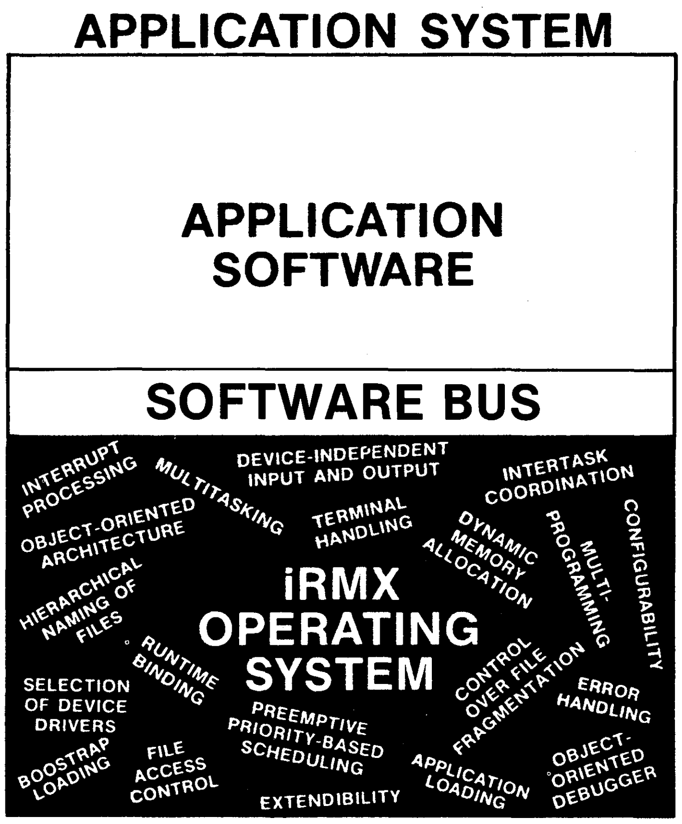


Figure A.5
Operating System - "BLACK BOX" With Intel "SOFTWARE BUS"

# Appendix B

It seems impossible to talk about operating systems without getting deep into the jargon. The following list contains many of the terms used throughout this and other Intel manuals. Many of the definitions have been used with permission from "Operating Systems", John Zarrella; Microcomputer Applications, Suisun City, 1979.

Access Time - The amount of time required to fetch a given data element. This time may be as short as 25 nanoseconds for data in a cache memory buffer or as long as hundreds of milliseconds for a flexible diskette sector access.

Allocate - A system service used by a task to request memory for messages or I/O buffers.

Attach - A system service of the I/O subsystem that connects a physical device to a logical device. Thus, tasks may be written to be independent of the physical devices and the user or another task may attach any legal device when the task executes.

Batch - A type of operating system where jobs are processed one at a time and a job must complete its execution before the next is begun. In an attempt to maximize the throughput of the system, batch operating systems will often overlap the loading of the next job, and the output of the previous job, with the execution of the current job.

Bit Map - A data structure utilized by some file systems to assist with file space allocation. A bit in the bit map is set aside for each sector on the media - a one indicates that the corresponding sector is allocated, and a zero indicates that the corresponding sector is free. The bit map is placed physically close to the directory in order to minimize disk head movement.

Breakpoint - The address at which task execution is to be suspended. Many systems feature the capability to suspend task execution before or after a breakpoint address is executed, when a read or write to a breakpoint address is attempted, or when an I/O operation to a breakpoint address is executed.

Buffer - A temporary storage area, usually used for message and I/O transfers.

Busy Wait - A situation in which a task spins, or waits for access to a resource by continuously testing a flag. Thus, even though the task is not performing any useful work during its wait, the processor that it runs on is busy executing test instructions, and no other task can execute on that processor.

Communication - The facility whereby one task can pass data and commands to another. For example, an application task may pass an output buffer to a printer control task. In some operating systems, this communication facility is generalized to permit tasks to reside in separate physical locations and to transmit data across a communication link such as a standard telephone line, leased line, or microwave link.

Compiler - A language translator that translates the text of a high level language such as FORTRAN or COBOL into machine code. Compilers automatically assign, save, and restore registers and generate subroutine linkages. Thus, a programmer does not need to worry about system housekeeping and can concentrate on the application.

Concurrent processes - Processes or tasks whose execution overlaps in time. They may be cooperating or interacting (communicationg with one another) or completely independent.

Consumable Resource - A resource, such as an I/O buffer or message, that is produced by one task and consumed and freed by another.

Contention - A situation that occurs when more than one task vies for a single resource.

Context Switch Time - The absolute time required for the operating system to switch from one task's environment to that of another.

Copy - A system support utility that copies data blocks from one storage device or location to another.

Create - A system service that initializes a structure by entering information such as its name, size, etc., into system tables. At this point the operating system knows the structure exists, hence the term creation. Other functions may also be performed depending on the type of structure to be created. For instance, creation of a file may mean default allocation of a prespecified number of disk blocks.

Critical Region - A portion of software that accesses a shared resource and must be protected so that while one task is performing the access (executing the software), no other task is permitted to access the same resource. For example, when updating a 32-bit floating point number, all four bytes must be written before another task may read the data. In some cases, interrupts must be disabled when this code is executed if a higher priority task may require access to the same resource as a result of the interrupt.

Data Base - A large and complete collection of information that covers a variety of subject areas. For instance, a company's payroll data base contains employee number, social security number, classifaction, age, wage rate, salary history, etc., about each employee. A medical diagnostic data base might contain symptoms for all common deseases and injuries.

Data Base Management - Facility to manage and maintain a data base in a manner designed to promote fast data access.

Data Structure - A mechanism, including both storage layout and access rules, by which information may be stored and retreived within a computer system.

Date - A system service that supplies the current time and date to the requesting task.

Deadlock - A situation that occurs when all tasks within a system are suspended waiting for resources that have already been assigned to other tasks that are also waiting for additional resources. This situation can typically be avoided through the use of semaphores.

Deblocking Buffer - When a read or write is issued, generally it is in byte increments, however, transfers from mass storage devices are usually in sector increments. The deblocking buffer is used to store such a sector and read or write only the bytes requested.

Debugger - A system software utility that aids a protrammer in removing errors from software.

Dedicated Computer (DC) - A computer system whose hardware and/or software limit it to a particular application. With this restriction, the computer can be optimized for one specific concern.

Delete - A system service used to remove a currently active data structure from the system tables.

Device - A unit of peripheral hardware, usually for Input and/or Output such as a printer, terminal, or card reader.

Device Driver - A system software module that directly controls the data transfer to and from I/O peripherals. Some operating systems also include the capability to add custom device drivers for hardware that is unique to a given system or installation.

Directory File - A named file containing entries for each file in the file system. Each directory entry contains information about the file name, owner, access rights, size, etc. Directory entries on some systems may be pointers to other directories.

Distributed Processing - A multiprocessing technique where each processor has a specific task or set of tasks to perform. These processors transfer commands and data via a standard communication interface, often SDLC or IEEE-488. In some cases, programs are transferred between processing units. However, program transfers are normally utilized for power-on loading or control algorithm changes rather than general load sharing.

DMA - Process used by an I/O device to transfer data to or from memory at the maximum memory data rate by stealing memory cycles from the processor. When the I/O transfer is complete, the processor is given an interrupt signal

Dual Port Memory - A memory subsystem designed to provide two complete paths (address, data, and control) to memory.

Dynamic Priority - A form of scheduling in that a task's execution priority varies depending on the system environment. For example, it is common for time sharing systems to initially give large tasks low priority, but to raise the priority periodically as the task waits to execute. This assures that low priority tasks will not be locked out of execution time by small, high priority tasks.

Editor - A system utility that permits a programmer to create, edit, concatenate, and delete complete files and portions of files on secondary storage media. Editors operate almost exclusively on text files. Three types of editors are common: Character editors, line editors, and screen editors. Line editors treat a text file on a line by line basis, while character editors treat a text file as a series of characters, where line delimiters such as carriage returns and line feeds, are treated like all other characters. Screen editors present an entire screen for review at once. The operator uses the cursor controls to manipulate entire sections of the screen at once.

Event - An event may be a timer or I/O interrupt, the arrival of a message at a task's mailbox, or the occurrence of an exception condition.

Event Driven - A scheduling algorithm where the occurance of certain events (usually interrupts) trigger the execution of the tasks in the system.

Exception - A condition which is out of the ordinary in normal task execution. For example, an arithmetic overflow, or the untimely removal of a disk pack.

Exchange - A system data structure that handles task communication. As with mailboxes, tasks may send to and receive messages at exchange points.

File - A collection of data, normally stored on a secondary storage device such as magnetic tape or disk.

File System - System software modules that manage files on secondary storage media. This software usually provides functions to create, delete, or rename files, permit reading and writing of existing files, and enforce system protection strategies.

Foreground/Background - A type of operat ing system with a combination of real time and either batch or multitasking capabilities. This type of system is used to permit time critical programs to operate in the foreground and execute with high priority while background assemblies, compilations, etc., run at a lower priority. Foreground programs always have priority over background work. This type of operating system usually has the capability to protect the fore- ground from the background, and is used in applications such as process control.

Format - A system utility that initializes secondary storage media with information necessary to ensure that data can subsequently be read or written without error. The information is usually closely related to the read/write hardware. An example of a popular formatting technique is the IBM soft-sectored format for single density diskettes.

Fragmentation - The division of contiguous storage areas such as main memory or secondary storage in a way that causes areas to be wasted. In segmented systems or in a system with dynamic memory al- location, continually loading and swap- ping new tasks will almost always cause small areas of memory to become un- usable. At the point where this wasted space impacts system performance, most operating systems perform garbage col- lection.

Free - A system service that returns allocated memory blocks to the memory manager for reuse.

Free List - A list of memory locations that are currently unused and may be allocated by the memory manager to requesting tasks. Free lists are usually organized as linked lists of memory blocks, where each block contains the size of the block and a pointer to the next block in the list.

Garbage Collection - A system function contained in most operating systems whose memory (main or secondary) is subject to fragmentation. This function reallocates the affected memory in such a way as to reclaim all wasted space. This is usually a drastic and time consuming step, and is only invoked on operator command or when memory utilization is very low and seriously affecting system efficiency. This function is sometimes known as Squeezing, Copmpressing, or Compacting.

Hard Error - An error in magnetic media, electromechanical device, or electronics that is repeatable.

Hierarchical Directory - A means of organizing secondary storage file directories where directory entries may describe either files or other directories. Some multiuser systems have a master disk directory with one subdirectory for each user.

Hierarchical Error Management - The processing of errors based on a scheme that allows global treatment of some errors and local, direct control of others.

High Level Language - A soiphisticated, yet easy to use, language (written words and special punctuation) that allows a programmer to write software without being concerned with housekeeping functions (eg. register allocation, parameter passing) or optimization. Common high level languages are FORTRAN, COBOL, BASIC, ALGOL, APL, PL/1, PL/M, and PASCAL.

Human Interface - A level of system software designed to reduce the level of confrontation between the operator and the machine. This type of facility usually contains basic functions like COPY, RUN, SUBMIT, FORMAT, CREATE, etc.

Indirect I/O - A facility, usually specified as part of the computer system architecture, whereby input from or output to peripheral devices is performed by a unit separate from the main processor. This unit is commonly called a Channel or I/O Processor.

Interactive Debugger - A system software utility that permits a user to examine his task while it executes by stopping it at given points (usually called breakpoints) and displaying and changing memory/register contents.

Interleaving - A track formatting technique utilized with moving and fixed head disk media in which sectors are not stored sequentially. Using this technique, multiple sectors may be read or written sequentially with a minimum of disk latency. This is made possible by placing sectors on a track in such a way that the time required to process a single sector is slightly less than the time required for the disk to rotate to the start of the next logical sector..

Interpreter - A language translator that accepts high level language input text and translates this text into a special intermediate code that is simulated (interpreted) by a system program. Usually this intermediate code cannot be directly executed on a general purpose processor.

Interrupt - A signal from an external source such as a timer or I/O device that causes a processor to interrupt executuion of the current task. Most operatiing systems use this occurrence to reschedule tasks for execution and therefore dedicate a timer interrupt for this purpose.

Interrupt Latency - The period of time imposed by the system hardware and software to acknowledge an interrupt signal and begin execution of a procedure associated with the particular interrupt.

Interval Timer - A hardware or software clock which generates an interrupt after a specified period of time.

I/O Subsystem - A set of system software modules that control all Input and Output within a system. An I/O subsystem normally contains device drivers, performs buffer administration, handles error conditions, and provides a uniform interfacing structure by which tasks may request I/O transfers.

ISAM - An abbreviation for Indexed Sequential Access Method. This technique for finding records within a file assigns a number or key to each record and creates a separate file index. To access a record, the record number or key is looked up in the index to find the actual record location within the file. This organization is useful for sorting files according to its content.

Job - A collection of tasks, grouped and run together in order to perform a specific function.

Job Control Language (JCL) - A special computer command language designed for use in batch systems to inform the system's software and the computer operator of unique requirements for the running of a computer program.

Kernel - The most basic portion of an operating system, usually supporting only task synchronization, scheduling, communication, and the most rudimentary of memory allocation capabilities.

Language Translator - A system program that translates text written in one language to another language. Assemblers, interpreters, and compilers are examples of language translators.

Librarian - A system program that is responsible for creating, editing, and deleting software libraries.

Library - A collection of system and/or user programs which may executed by other tasks in the system.

Linker - A system software module that connects previously assembled or compiled programs or program segments into a unit that can be loaded into memory and executed.

Loader - A system software module that moves user tasks from secondary storage into memory for execution. It may also perform relocation computations as required.

Load Module - A software module that is ready to be loaded into memory and executed by the system loader. A load module has all static relocation and linkage operations completed.

Load Sharing - A scheduling technique in multiprocessing systems whereby a task is executed by the next available processor. In order to make this technique operate successfully, all processors must behave identically, and have identical memory addressing capabilities.

Logical Device - A reference to an I/O device by name or number without regard to the exact nature of the I/O characteristics of the device.

Log On - A mechanism by which a computer system user identifies himself and gains access to system facilities.

Mailbox - A system data structure that handles task communication. Tasks send messages to and receive messages from Mailboxes.

Memory Dump - A system utility used in software debugging that prints the contents of all or a portion of main memory on the printer.

Memory Manager - A system software module that manages the physical memory resource and controls memory allocation for tasks in the system. In a virutal memory system, this module also translates logical addresses into physical addresses and implements the page fetch, placement, and replacement algorithms.

Memory Mapped - A method of implementing system input and output that provides I/O ports that are accessed as if they were memory locations. However, the term memory mapping does not imply that all memory operations (eg. increment memory contents) may be performed on a memory mapped I/O port. This is due to the fact that some peripheral control registers may be either read only or write only in nature. In fact, some designs mate write only control data with read only status data in order to save memory addresses.

Microcomputer Development System - A system (like Intel's MDS 230) designed exclusively to aid in the development of microprocessor hardware and software systems.

Module - A section of software that has well defined inputs and outputs and may be developed and tested independently of other software.

Multiprocessing - The ability of an operating system to support multiple processors. Generally, in order to make the operating system task manageable, all processors are equivalent and interchangeable, although this is not an absolute requirement.

Multitasking - The ability of an operating system to permit multiple tasks to run on a single processor concurrently.

Multiterminal - The ability of an operating system task to support multiple terminals connected to the same system. Note that this capability does not imply Multiuser capability, since on some systems, multiple terminals must be under control of the same task or cooperating tasks, and all users are therefore forced to use the same software. For example, order entry processing is usually a single task controlling many terminals.

Multiuser - The ability of an operating system to support more than one independent user. A true multiuser system allows different users to independently use the system resources.

Mutual Exclusion - A process synchronization rule which prohibits more than one task from using the same resource at the same time.

Named File - A collection of bytes residing on a random access storage device and possessing a name by which the entire collection can be refered to.

Non-Preemptive Scheduling - A scheduling algorithm where a task does not stop executing until it is complete. Non-preemptive scheduling techniques attempt to pick the tasks for execution which keep average turnaround time to a minimum. However, once a long task begins execution, all other tasks must wait until it completes, regardless of relative priorities.

Nucleus - The kernel, or innermost layer of an operating system executive.

Object - Objects are data structures with a fixed set of attributes. Just as a floating point number is a data structure with operators, Object-oriented operating systems provide basic functions to operate on the varies objects defined in its environment.

Object Code - The output of an assembler or compiler that will execute on the target processor. Note that linking, locating, or loading may be required before this code can execute directly on the processor.

Object Oriented - A characteristic of an operating system that provides system calls to operate on and manipulate certain data structures called objects. By grouping many features into a generic "object", the O.S. is able to maintain a consistent interface while dealing with different structures.

Open - A system service used with disk files and other shared resources. This service informs the operating system of the manner in which a task intends to use a given resource.

Operating System - A collection of system software that permits user written tasks to interface to the machine hardware and interact with other tasks in a straightforward, efficient, and safe manner.

Overhead - The amount of processing time required by the operating system to perform housekeeping tasks such as paging, swapping, and scheduling. This time is usually expressed as a percentage of total available time. If system overhead is quoted as 20%, user programs can only utilize 48 seconds (80%) of each minute of processor execution time.

Overlay - A technique used to execute programs which are larger than the available memory size in systems without paging or segmentation capabilities. To utilize this method, a program must be manually divided into a number of mutually exclusive groups of software modules. Any software common to more than one overlay is included in the 'root' overlay. The root must remain in system memory at all times and other overlays are loaded and executed one at a time in a designated memory area. This loading is normally performed by the operating system at the request of the root overlay.

Position Independent Code - Executable code with runs independent of the physical memory location at which it is loaded. This usually implies that the code contains only relative transfer instructions, or that all addressing is done via base registers.

Preemptive Scheduling - A means of scheduling where a high priority task will preempt execution of a lower priority task as soon as the high priority task is ready to run, usually after an interrupt.

Priority Scheduling - A scheduling algorithm where each task is given a priority, and the next task to execute is the one in the Ready List with the highest priority. Priorities may be static or dynamic. With static priorities, it is possible that some low priority tasks will never run due to the frequency and execution duration of higher priority tasks. This is very undesirable in a Time Sharing System environment where it is necessary to guarantee that all tasks will receive some processor time on a regular basis. For these systems, the task priority is often changes as a function of time, so that the longer a task waits to execute, the higher its priority is elevated.

Privileged - Instructions that may only be executed by system software. Set Interrupt Mask, Load Base Register, Load Protection Register, and Halt are all examples of privileged operations. Some computer systems execute in one of two operational modes: normal and privileged. System software executes in the privileged mode and may execute all processor instructions. User tasks operate in the normal mode and the system is protected against many task malfunctions.

Protection - An operating system service that prevents a task from interfering with the execution of and the data belonging to onther task.

Queue - A data structure in which the elements form an ordered list while waiting for some activity. Queues may be implemented as First In First Out (FIFO), or using some other scheme.

Random Access File - A type of file structure in which data may be accessed in a random manner, regardless of its position within the file.

Ready - A task state in which a given task is prepared to execute.

Ready List - A system list of tasks that are in the Ready state; available for execution on the processor.

Real Time - A type of operating system that supports online equipment having critical time constraints. Events must be handled promptly, and within strict timing limits. Most process control systems are real time in nature.

Record - A set of data elements that are logically accessed together. On a secondary storage medium, this can also be used to indicate the smallest set of data elements that are stored contiguously on the medium.

Reentrant Code - Code that may be executed simultaneously by more than one task. Thus, the code cannot be self-modifying, and each task must maintain its own data area.

Register - A high speed memory location associated with a processor. Registers are normally used to store intermediate data and addresses within a program in order to speed overall execution time. Due to the high speed requirements and the cost, the number of registers within a processor is usually limited to 8 or 16 storage locations, although some processors have a few hundred.

Relocation - The process of moving the location of a program in system memory. Relocation may either be static or dynamic. Static relocation means that all program addresses are assigned and fixed before the program is loaded into memory. Dynamic relocation refers to a technique that requires all program addresses to be relative to some base address and programs may be relocated at any time by moving the program code and changing the base address.

Rename - The act of changing the name of a data structure. This function is commonly supplied as a system service of the file system in order to change file names.

Reprogrammable Computer (RMC) - A computer system containing a language translator or interpreter that allows its user to create and modify application software without the use of a separate development system.

Resource - Assets of a computer system that the operating system can use and/or allocate to tasks for their use. Assets such as memory, disk storage space, printers, and terminals, as well as processors in a multiprocessing system, are typical system resources.

Resource Allocation - The operating system function of assigning resources to tasks that request them. Resources such a peripherals (printers, terminals, and card readers) and main memory are allocated to the tasks according the different algorithms that depend on the type of operating system.

Response Time - The elapsed time from the entry of a command until its execution is complete. This term is usually used to represent the time it takes for a Time Sharing System to respond to a user command line from a user's terminal.

Running - A task state in which the task is executing on a processor.

Save - A system utility that writes a file from secondary storage to a backup volume such as magnetic or paper tape for protection or archival purposes. Often, save also refers to storing a program currently in memory on a disk for later reference.

Scheduler - A system service that determines which task within the system should be running at any given instant.

Scheduling Algorithm - The rule(s) by which the scheduler determines which task in the Ready List should be granted processor time next.

Sector - The smallest contiguous storage area on a rotating memory device. For example, flexible diskette drives use a typical sector size of 128 bytes.

Segment - An arbitrary user defined block of memory (containing data or instructions) that functions as an independent unit. A segment may be placed anywhere in memory and its contents accessed by segment name and relative location within the segment. Some computer systems implement segmentation in hardware via a Base Register (to contain the starting address of the segment), with all segment accesses occurreing relative to the segment starting address.

Semaphore - A 'gating' variable that is used to synchronize task operations on shared data. A semaphore permits only a single task to access shared data at any given time. All other tasks are locked out until the first task unlocks the shared data. By operating on semaphores with two well defined functions a task may request and release exclusive access to shared data.

Send - A system service used for task communication and synchronization that places a message in the appropriate mailbox or exchange.

Software Bus - A layer of software whose interface is standardized and serves as a common ground on which many different compilers and applications may be built.

Spooler - A system program which permits I/O transfers to be queued for an I/O device, thereby per- mitting the requesting task to continue executing even when it cannot im- mediately use the device. Spoolers are commonly used with very slow sequential output only devices such as printers.

Suspended - A task state in which the task has lost control of the processor.

Swapping - A feature of an operating system which permits suspended tasks to be moved to secondary storage in order to generate enough memory space so that the next task on the Ready List can be loaded into memory (if it is not already resident) and executed. The suspended task will be swapped back into system memory when it is again scheduled for execution. This movement to and from secondary storage may occur many times to a given task before its execution is complete.

Synchronization - The process of coordinating the execution of tasks within an operating system environment. Access to shared data as well as the transmission and reception of messages requires this coordination.

System Generation - The process of generating, linking and loading all required system modules together in order to build a new operating system or to update configuration tables in an existing system. Because of installation environment variations, it is often necessary to make modifications to an operating system (as shipped by the vendor) through this process.

System Software - Software that is intimately associated with the operating system. For example: Kernel routines, system services and system support software.

System Support - Functions such as language translators, debugging tools, diagnostics, and libraries which enable a system user or programmer to write and test tasks in an efficient manner.

Task - A software module in which code is executed in a sequential manner.

Thrashing - The point of system collapse in which operating system overhead is so large that no useful task execution can be accomplished. This occurs often in virtual storage systems when so many users are on the system that every time the system attempts to service a user, a page fault occurs, and the system is constantly attempting to load or reload pages from secondary storage.

Time - A system service that supplies the current time of day to the requesting task.

Timer Resolution - The minimum period of time discernable by the timing function of the operating system clock. All timers in a system have no more accuracy than plus or minus one of these periods.

Time Sharing - A type of operating system that supports more than one user concurrently, and allows each user to interact with his job. This is done by allocating processor and resources to each user in turn for a small period of time, thus assuring each user a system response within a few seconds. This philosophy is almost the exact opposite of a Batch System.

Time Slice - That period of time, normally between 1 and 50 milliseconds, allocated to each task in a time sharing system before another task is allowed to run.

Token - A 16-bit item used to identify an object. The operating system and its users manipulate objects only when in control of the associated tokens.

TSS - An abbreviation for a Time Sharing System.

Turn-Around Time - The interval between the time a program or job is submitted and the time it is completed. This term is usually restricted to refer to Batch Systems.

Virtual Memory - A technique that separates the logical address space from the physical address space. This permits an application program to be written and executed independent of the available physical memory.

Volume - A unit of secondary storage media such as a magnetic tape, disk pack, or flexible diskette.

Wait - A system service that causes a task to be suspended pending the occurrence of an event. It is also used to refer to the task state in which execution is suspended pending the occurence of an event.

## Notes

intel®

5200 NE Elam Young Parkway   Hillsboro, OR 97123   (503) 640-7112