FlexOS 286 -- Flexible Automation Operating System

The FlexOS family of operating systems provides real-time, multi-tasking capabilities applicable to a wide variety of control and data processing applications. The family consists of separate versions developed for a specific microprocessor. Each version supports the chip's distinctive features while maintaining a common program and driver interface. This data sheet describes FlexOS 286, the version developed for the Intel 80286 microprocessor.

Synopsis

System Environment

- Intel 80286 or 80386 processor
- 16 Mbytes addressable memory
- Dynamically loadable and unloadable drivers
- Real-time response to hardware and software interrupts, event driven dispatcher
- Number of processes limited only by system memory size
- Number of I/O devices limited only by system memory size

Program Environment

- Separated user address spaces (protected mode)
- Asynchronous supervisor functions, up to 31 outstanding events per process
- Standard terminal interface, virtual consoles
- Virtual Device Interface for graphics devices
- Get, set and lookup access to comprehensive set of system, console, file and device parameters
- Logical device names and name substitution
- File and device locking by user and group identification
- Asynchronous record locking on disk files
- 1 Gbyte logical disk max. with no limit to number of logical disks
- DOS 3.x compatible disk media with hierarchical directories
- Built-in interfaces to network and graphics devices

User Interface

- Multiuser capability
- Logon and logoff with password protection
- MS-DOS front end
- Window manager
- Multitasking graphics services (FlexPanel)
- International character sets

FlexOS 286 is a general purpose operating system that supports Intel 80286 protected mode operation and provides unrestricted access to the processor's full, addressable memory space. FlexOS 286 components include the multitasking supervisor with real-time kernel and a set of resource managers for monitoring device and network I/O and interprocess communication. These features give the application programmer and system designer a multiuser, multitasking environment in which real-time control processes and general purpose, personal computer applications can run simultaneously.

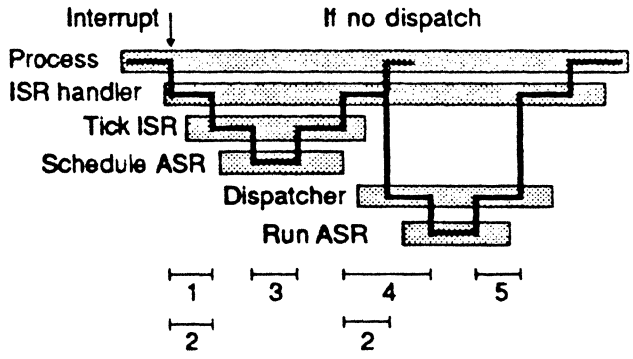
FlexOS 286 multitasking is based on an event-driven dispatcher allocating CPU time to prioritized processes. The OEM determines which events trigger a dispatch (for example, the illustration below shows a tick interrupt). On each dispatch, the dispatcher saves the registers, runs all outstanding Asynchronous Service Routines (routines within driver code that execute asynchronously to processes), and then transfers control to the highest priority, ready process. Process priority levels range from 0 to 255.

Performance Characteristics

Value in a real-time operating system derives from how quickly it responds to interrupts by external events. FlexOS 286 services interrupts immediately through its Interrupt Service Routine handler. This module saves the current process context, disables interrupts, and transfers control to an Interrupt Service Routine (ISR).

The figure below illustrates the transfers of control from process to ISR and back to a process in the sample tick interrupt handler provided with FlexOS 286. In this example, the Tick ISR schedules an Asynchronous Service Routine (ASR) and forces a dispatch--the full response cycle. ISRs do not, however, have to schedule an ASR nor force a dispatch. The table lists the timing for the periods shown.

FlexOS 286 imposes no limits on ISR length and allows the programmer to respond differently according to the urgency of the interrupt. For example, if the interrupt requires immediate servicing, the ISR can perform the entire action and



FlexOS Kernel Timings

All times are in microseconds and are based on a clock rate of 8 MHz with no wait states

From Illustration

1)	Interrupt to ISR	36
2)	ISR handler overhead (no dispatch)	48
3)	Schedule an ASR	42
4)	ISR to ASR	98
5)	ASR to same process	83
	ASR to different process	120

General

Process dispatch to different process	184
Process dispatch to same process	148
Get an event mask for a system flag	184
Complete a system flag event	98

FlexOS 286 -- Flexible Automation Operating System

return. On the other hand, the ISR can service a less urgent interrupt by scheduling an Aynchronous Service Routine to do the work. To ensure the timely execution of the ASR, ISRs can force a dispatch. FlexOS allows ISRs to reenable interrupts so less significant interrupts do not prevent the servicing of an important interrupt.

File Access and Security

Processes access disk files, consoles, printers, pipes (memory buffers for interprocess communication and synchronization, and other system resources through the file system using physical and/or logical device names. The supervisor translates logical to physical names according to configuration information defined at system boot and allows logical name substitution.

File access is governed by a security word setting world, group, and user privileges for read, write, execute, and delete access. Processes can open files for shared or exclusive access. Shared access can be regulated on a read or write basis with asynchronous record locking and shared or process-independent file pointers.

Application Services

FlexOS provides a rich environment for application development. Resources include the extensive set of supervisor calls listed in the FlexOS folder, a MS-DOS front end, and the FlexPanel graphics package. The MS-DOS front end emulates MS-DOS version 2.1 functions and conventions and runs many of the popular business applications (e.g., Lotus 1-2-3, MicroPro WordStar, Ashton-Tate dBase II) and programming tools developed for personal computers. To use the MS-DOS front end, the system must have the E2 stepping of the 80286 processor. Subsequent revisions of the front end will support newer MS-DOS versions.

FlexPanel is a graphics interface for displaying control data such as temperature, pressure, or flow-rate in forms familiar and meaningful to the operator. For example, furnace temperature can be shown as an actual thermometer under FlexPanel, rather than a difficult-to-decifer list of numbers. FlexPanel allows multiple meters and functions to be monitored simultaneously and updates to occur as frequently as the system clock tick (the tick rate is OEM-defined). FlexPanel uses the FlexOS 286 Virtual Device Interface so that it can be interfaced readily to any display device.

Developer Kits

FlexOS 286 is available in two packages, one designed for the application developer and the other for the system developer.

The Developer Kit provides a bootable system, utilities, and tools for application development for all of the FlexOS environments. The utilities include an editor and many Unix-like programs; tools include an assembler, linker, librarian, and standalone and symbolic debuggers

The FlexOS Developer Kit Supplement adds the FlexOS 286 object modules and driver sources to the Developer Kit contents. These resources allow the system developer to tailor FlexOS 286 to any 80286-based, hardware environment and for any purpose.

Related Products

FlexOS 286 is one member of the FlexOS family of products and the broad line Digital Research of system products. The other products in the FlexOS family are FlexOS 186 and FlexOS 386, FlexOS implementations for computers based on the Intel 80186 and 80386 processors, the FlexPanel graphics services with Virtual Device Interface, and the FlexNet network interface. The FlexOS architecture and application interface are also available for computers based on the Motorola 68000 family of processors in Concurrent DOS 68K. Implementations for other processor environments are under evaluation and development.

The FlexOS 286 Developer Kit includes an editor, assembler, linker, librarian, and debuggers. In addition, the MS-DOS front end provides access to many of the development tools available for that evironment. FlexOS 286 also supports the following high-level language compilers: Digital Research CBasic, Metaware High C (a full implementation of the proposed ANSI C standard), Metaware Professional Pascal, an ADA/ED-C interpreter developed at New York University, Ryan McFarland Fortran 77, and MicroFocus Level II Cobol.

For more information: Data sheets and manuals are available for all FlexOS and Digital Research products at Digital Research sales offices and corporate headquarters in Monterey, California.

Headquarters: Digital Research Inc., 60 Garden Court, Monterey, CA 93942 (408) 649-3896 TWX 910 360 5001

Western Sales Office: Digital Research Inc., 1860 Embarcadero Road, Suite 215, Palo Alto, CA 94303 (415) 856-4343
FAX (415) 856-2103

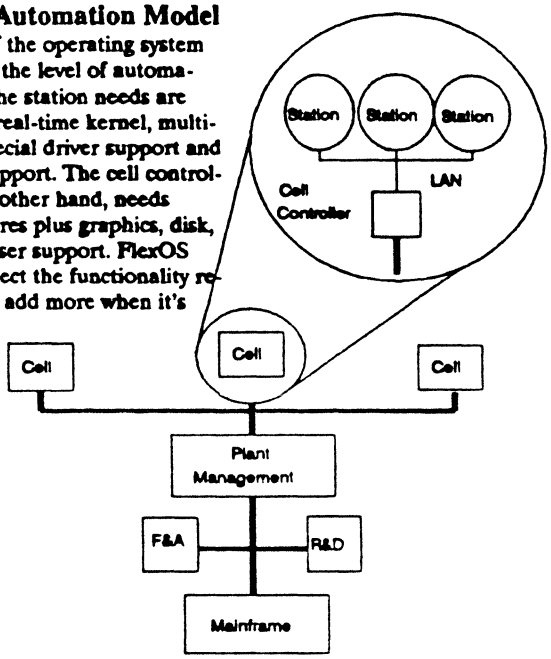
FlexOS System Software for Flexible Automation

Flexible automation, a concept that embraces environments ranging from computer integrated manufacturing (CIM) to home automation, requires the ability to integrate multiple programmable devices, each with a timely response to external events, through a common program interface. If the foundation for application software development, the operating system, does not provide a high degree of portability, perfor-

The flexible automation component that affects the cost-effective development of application software the most, and therefore impacts directly on the system's investment payback period and overall utility, is the operating system. Important operating system characteristics that affect application development and value are:

- functionality--the ability to provide real-time control, transaction processing, batch and interactive processing, and graphic user interface
- portability--the ability to run applications unaltered on different machines
- integratability--the ability to have different machines communicate with eachother
- flexibility--the ability to replace device drivers without affecting the other device drivers and the applications that use them.

Flexible Automation Model
The role of the operating system varies with the level of automation; e.g., the station needs are typically a real-time kernel, multitasking, special driver support and network support. The cell controller, on the other hand, needs these features plus graphics, disk, and multiuser support. FlexOS lets you select the functionality required and add more when it's needed



Since operating systems usually provide a unique programming environment, developing and maintaining applications gets quite complex and time-consuming when there are different operating systems running on the various machines. The system developer's challenge, then, is to minimize the number of operating systems required.

In answer to this challenge, Digital Research offers FlexOS, the flexible automation system software. FlexOS is a family of products comprised of processor-specific operating systems with a standard program interface, the FlexNet network interface, and the FlexPanel graphics display services. All the FlexOS operating systems are real-time and multitasking with design features that simplify and speed the development and maintenance of programmable controllers and computers.

FlexOS is the first true operating system to combine the features of

- memory and file protection associated with mainframe computers,
- the high-performance, real-time programmability found in programmable logic controllers,
- the cost-effective, general-purpose advantages of personal computers

into a highly configurable, independently-supplied environment. These features provide a logical migration path from small, high-speed, dedicated processing systems on the factory floor to the high-performance, DOS-compatible, memory intensive systems required in the workstation environment.

The Importance of System Flexibility

The FlexOS features described above provide the functionality, portability, and integratability necessary to implement flexible automation successfully. Flexibility is an equally important feature; for example, overall system flexibility lets the developer correct hardware and software mistakes without affecting the hardware and software successes. Flexibility must be inherent in two places: the application interface and the driver interface. Criteria for evaluating application interface flexibility are:

- How easily is the functional interface adapted to the system requirements.
- How easily are performance objectives met without writing direct to the hardware?
- How easily are special calls made directly to the hardware when necessary?

Criteria for evaluating the device driver interface are: How easily is new technology integrated into the system? and how easily are device drivers replaced without affecting existing applications?

FlexOS flexibility derives from its modular architecture. FlexOS is composed of a supervisor, a set of independent resource managers, and device drivers. The supervisor (itself composed of kernel, memory manager, and loader) provides a standard function interface to the system's

resources, such as consoles, disks, and sensors, and directs calls to the appropriate resource manager. The resource managers govern the device drivers. FlexOS has five resource managers:

- Disk--Controls access to floppy and fixed disk drives.
- Console--Controls monitor, keyboard, and pointer I/O and provides window management and graphics functions.
- Network--Provides a transparent network user interface and connection or connectionless application interface to individually loadable network drivers.
- Pipe--Creates and controls access to memory files and shared memory regions for interprocess communication.
- Miscellaneous--Controls I/O with all other devices, including printers, plotters, sensors, communication devices, and so forth.

Figure 2 illustrates the relationship of components in a FlexOS system.

A core FlexOS system consists of the supervisor, the Miscellaneous Resource Manager, and the device drivers needed to load the system and software. The other resource managers are optional and totally independent of eachother. If required, they are integrated into the system at system link time. Device drivers are also independent of eachother. They can be integrated in any of three ways:

- Linked into the system with the supervisor and resource managers.
- Loaded with the configuration boot script (an ASCII file read automatically by FlexOS during initialization)
- Loaded under program control.

FlexOS also supports dynamically loadable subdrivers. These are individual components driven by the same driver code (for example, two serial printers).

The supervisor and resource managers are designed for processor and device independence. (See the data sheets for the description of the FlexOS implementations on various processors.) The only place the code must be hardware specific is in the drivers. This provides application portability between the different machines and offers two other advantages:

- Target machine applications can be written on a development system with a simulator in a protected environment and installed on the target machine without modification.
- Physical devices can be replaced without affecting the application software.

The FlexOS combination of modular architecture, portable code development, and real-time processing helps to avoid many of the pitfalls associated with flexible automation development. For example:

- You don't get locked into superseded technology. FlexOS allows you to add or replace drivers without affecting other system components.
- You don't need software kludges to get the job done. The FlexOS supervisor and function interface provide the real-time and multitasking features needed to service devices either under supervisor or program control--whichever is more appropriate to the task.
- You don't have a system with multiple and conflicting program interfaces. FlexOS is available on a variety of processors so that programmers have a single system-wide interface to learn rather than a different interface for each machine.
- You won't get overwhelmed by data. Often the problem isn't acquiring the information; it's presenting it in a digestible form. The FlexOS multitasking capability lets the user run multiple applications simultaneously, with separate windows for each application. The graphics support allows the programmer to display complex information with simple graphic representations.
- You won't end up with isolated islands of automation. The FlexOS network interface lets you incorporate individual drivers for multiple network protocols. This allows the machine to talk different protocols and to change protocols without affecting existing programs.

Other benefits realized from FlexOS flexibility include:

- Reduced system overhead--only the system modules required need be used and maintained.
- Application portability--applications developed for one FlexOS environment can run on others.
- Fewer programming tools required--a single set of compilers, assemblers, linkers, and run-time libraries can be used for all machine programming.

FlexOS Component Features

FlexOS provides the broad services characteristic of general purpose operating systems with performance characteristics features usually attributed to dedicated operating systems. The program interface consists of a comprehensive set of functions supported on a per process basis. A subset of the functions support asynchronous operation. (Asynchronous function calls allow program code to make a device call and continue executing without waiting for the call to return.) Table 1 lists the FlexOS supervisor functions.

All function calls go through the supervisor interface. The supervisor manages the file system and controls access to all resources. The file system has the following characteristics:

- Dynamic driver installation and removal
- Named devices and files
- Logical name substitution
- Concurrent file access protection
- Protected access by user and group IDs
- Table information accessible for each type of file (console, disk, pipe, port, printer, etc.).

The FlexOS kernel manages processes and memory. Both the kernel's timer facility and memory manager are OEM-modifiable. Other features of the kernel are:

- Event driven dispatcher that allows a high priority process to immediately react to important external events.
- Optional time slicing to reschedule CPU bound tasks on a process priority basis
- Single or multiple user capabilities that can be added or removed at run time without interrupting on-going processing
- Protected memory
- Loadable from ROM or via network
- Process table information providing parent process ID, process state, current priority, outstanding events, etc.

The supervisor routes device access calls (e.g., open, read, and write) to the associated resource manager. The resource managers provide the following features:

Disk Resource Manager

- PC DOS 3.x media compatible
- Hierarchical directories
- Asynchronous record locking
- File ownership by user and group IDs
- Directory label
- Fixed length records
- Optional mixed case media
- Special function support

Console Resource Manager

- Multiple physical consoles
- Dynamic virtual console creation, selection, and deletion
- Dynamic window creation, selection, and deletion
- Standard keyboard and screen
- Optional Digital Research GEM™ (Graphic Environment Manager) and Virtual Device Interface (VDI) graphic functions.
- 8- or 16-bit modes for input and/or output
- 8-bit PC, 8-bit ISO ASCII, 8-bit KANA, Shift-JIS KANJI, and 16- bit KANJI support

Pipe Resource Manager

- Dynamic, named pipe and shared memory creation and deletion, (up to 64K)
- Memory-based message pipes for interprocess communication
- Semaphore pipes for interprocess synchronization
- Shared memory regions
- Separate security (by user and group ID) on each pipe and shared memory region
- Non-destructive reads on message pipes

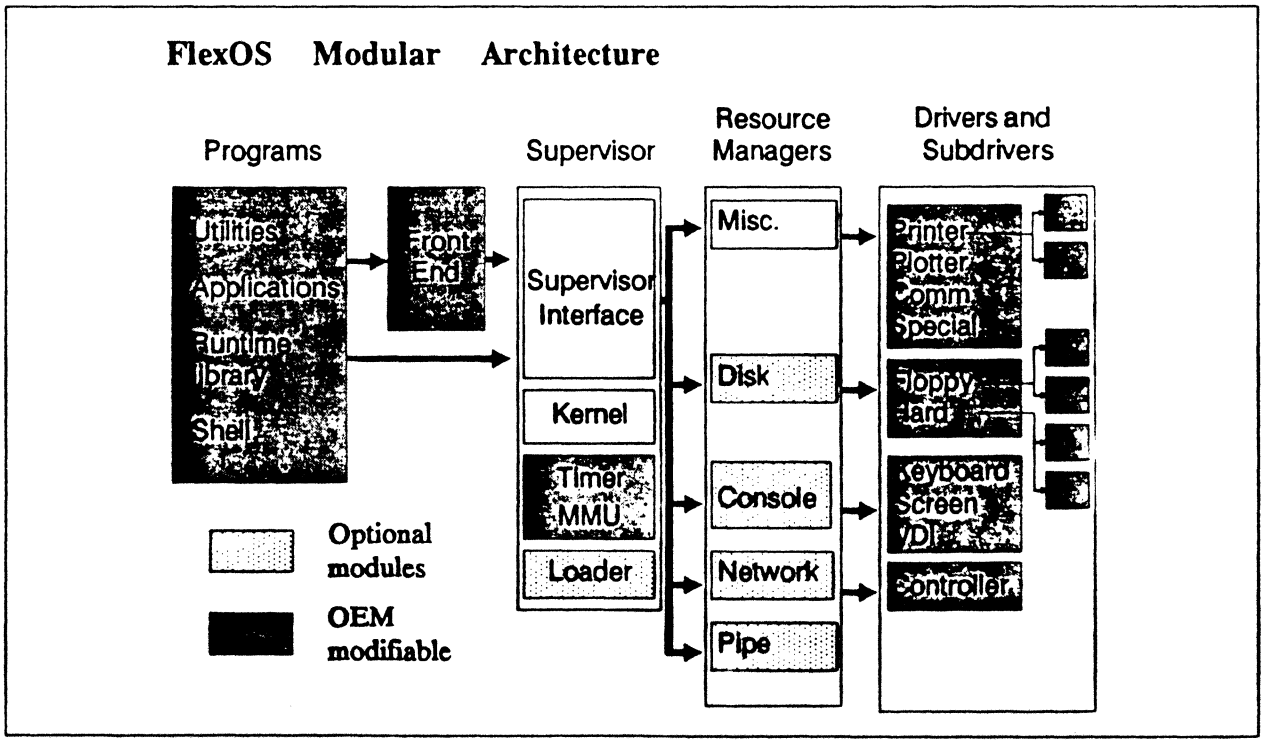
Miscellaneous Resource Manager

- OEM-defined function support
- Special table support for OEM-defined device information
- Automatic network support for special devices

Driver Services

The FlexOS supervisor provides over 40 driver functions for event, process, and memory management. You can also call the supervisor functions from drivers. Table 2 lists the driver functions.

FlexOS allows a single driver to support multiple units through device subdrivers. Each unit is a logical device. For example, a single disk driver can be written to control several floppy or hard disk units. Unit access can be controlled by the resource manager or the driver. In addition, driver installation options allow unit access synchronization at the driver or unit level.



FlexOS Supervisor Function Summary

File Management

DEFINE	Define logical name for a path
CREATE	Create a file
DELETE	Delete a file
OPEN	Open a file
CLOSE	Close a file
READ*	Read from a file
WRITE*	Write to a file
SEEK	Modify or obtain current file pointer
LOCK*	Lock/unlock an area of a file
RENAME	Rename or move a file

Console Management

KCTRL	Obtain keyboard ownership
ORDER	Order windows on parent screen
XLAT	Specify keystroke translation
GIVE	Give keyboard to child process
COPY	Copy screen rectangle
ALTER	Alter a screen rectangle
GSX	Call Virtual Device Interface

Event Management

CANCEL	Cancel asynchronous events
WAIT	Wait for one or more events to complete
STATUS	Get status of asynchronous events
RETURN	Get return code of completed event

Real-time and Process Management

TIMER*	Set and wait for timer interrupt
ABORT*	Abort specified process
COMMAND*	Perform command
EXCEPTION	Set software interrupts on exceptions
MALLOC	Allocate memory to heap
MFREE	Free memory from heap
EXIT	Terminate with return code
ENABLE	Enable software interrupts
DISABLE	Disable software interrupts
SWIRET	Return from software interrupt
CONTROL*	Control a process for debugging
OVERLAY	Load overlay from command file

Device Management

SPECIAL*	Perform special device function
DEVLOCK	Lock or unlock device for user/group
INSTALL	Install, replace, and associate drivers

Table Management

GET	Get a table
SET	Set table values
LOOKUP	Scan and retrieve tables

* These functions can be called asynchronously.

FlexOS Driver Services Summary

Flag System

FLAGCLR	Clear a flag
FLAGEVENT	Return an event mask
FLAGGET	Allocate a system flag number
FLAGREL	Release a system flag
FLAGSET	Set a system flag (completing an event)

Asynchronous Service Routines

ASRWAIT	Wait for an event to complete in an ASR
DOASR	Schedule an ASR
DSPTC	Force a dispatch
EVASR	Schedule an ASR to run upon process event completion
NEXTASR	Schedule ASR to run upon ASR event completion

Device Polling

POLLEVENT	Poll device for event completion

System Memory Management

MAPU	Map another process's User Memory
MAPPHYS	Map Physical Memory
MLOCK	Lock User Memory
MRANGE	Check ranges
MUNLOCK	Unlock User Memoyr
PADDR	Convert System Address to Physical Address
SADDR	Convert User Address to System Address
SALLOC	Allocate System Memory
UADDR	Convert System Address to User Address
UNMAPU	Restore User Memory

Critical Regions

ASRMX	Obtain ASR mutual exclusion (MX) region ownership
MXEVENT	Obtain process MX region ownership
MXINIT	Create an MX region
MXREL	Release an MX region
MXUNINIT	Remove an MX region from the system
NOABORT	Enter a "no abort" region
NODISP	Enter a "no dispatch" region
OKABORT	Exit a "no abort" region
OKDISP	Exit a "no dispatch" region

System Process Creation

PCREATE	Create a system process

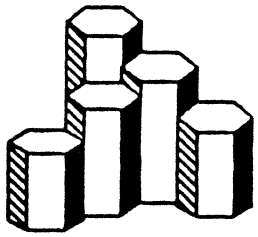
Interrupt Service Routines

SETVEC	Set interrupt vector to interrupt service routine (ISR)

For more information: Data sheets and manuals are available for all FlexOS and Digital Research products at Digital Research sales offices and corporate headquarters.

Headquarters: Digital Research Inc., 60 Garden Court, Monterey, CA 93942, (408) 649-3896 TWX: 910 360 5001

Western Sales Office: Digital Research Inc., 1860 Embarcadero Road, Suite 215, Palo Alto CA 94303 (415) 856-4343 FAX: (415) 856-2103



FLEXIBLE AUTOMATION

"Integrating Man & Machine"

Software Products for Industry

The FlexOS™ technology represents an integration of popular features found in competitive operating systems.

UNIX™	RMX™	VRTX™	DOS
Portable software architecture	Full use of hardware architecture	Fast response time	Application interface
"C" code interface	Strong sponsorship	Reliability	File system
Tools	Migration path	Modular design	Familiar feel
		Real – time services	
		ROMable	

FlexOS supports all of these features and more

Graphics – Device independent
Networking – Protocol independent

O/S PRODUCT FAMILY

Key Facts:

- > 75 + Man Years in development & testing
- > Real – time, vertical market focus
- > 45 + current customers
- > 186 Beta 2/87
- > 386 Beta 5/87

FlexOS 386

FlexOS 286 superset
More power
Virtual 86
Full addressability
Mini replacement

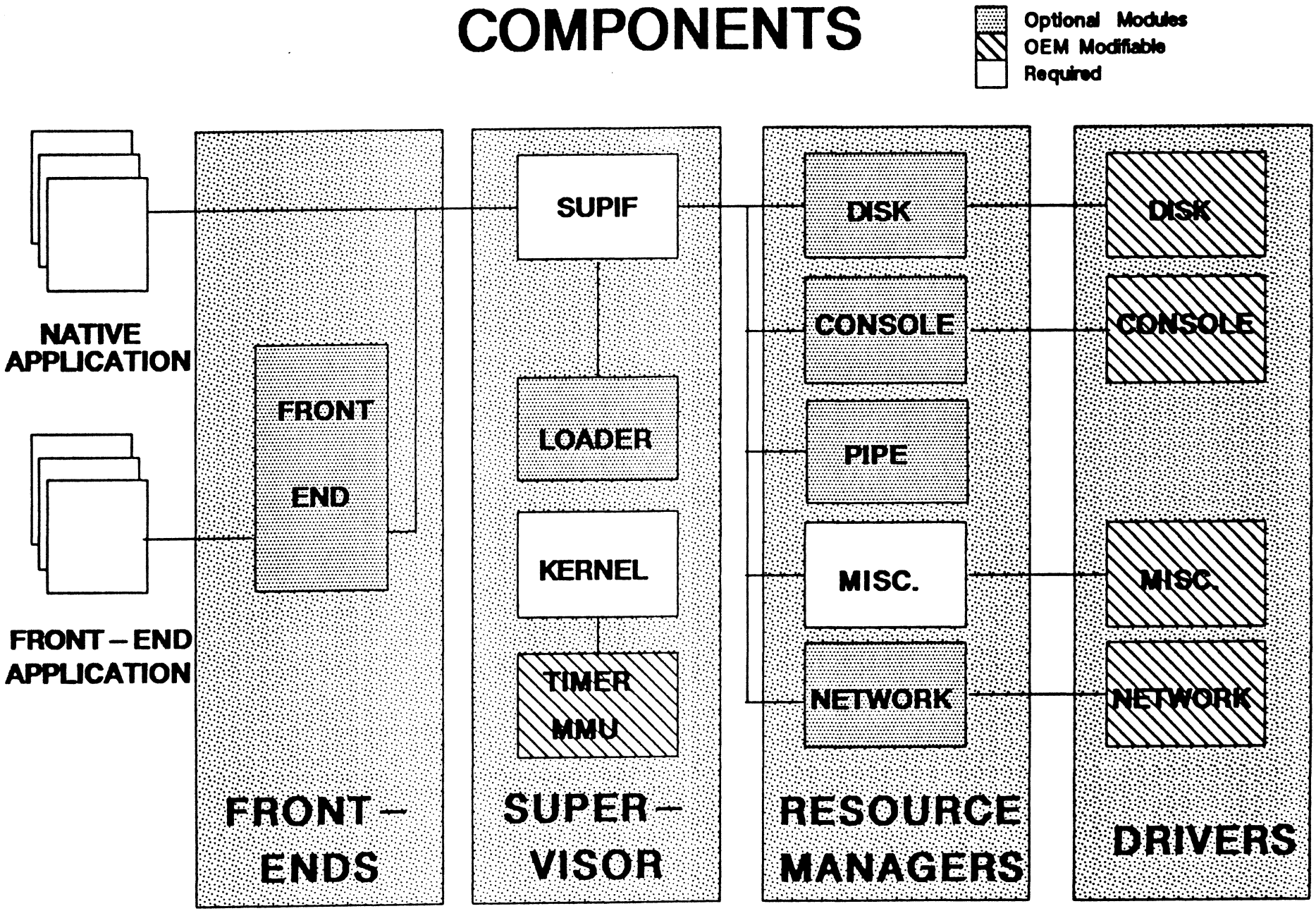
FlexOS 286

Protected Mode
Multi – tasking Graphics
Protocol Independent Network
DOS compatible interface
Migration to 386
Supervisor workstation

FlexOS 186

Kernel only
ROM/Download
Network node
Embedded control

COMPONENTS


Optional Modules
OEM Modifiable
Required
NATIVE APPLICATION
FRONT – END APPLICATION
FRONT END
SUPIF
LOADER
KERNEL
TIMER
MMU
DISK
CONSOLE
PIPE
MISC.
NETWORK
DISK
CONSOLE
MISC.
NETWORK
FRONT – ENDS
SUPER – VISOR
RESOURCE MANAGERS
DRIVERS

Features

Process Management

- Real – Time
- Multi – Tasking
- Single or Multi – User

Protection

- Protected File Access
- Protected Memory

Async I/O System

- 31 Events per process
- Wait for multiple events
- Software Interrupts on event completion

Configuration

- Dynamically Loadable Device Drivers
- Modular

Console System

- Virtual Consoles
- 16 – bit/8 – bit Character Sets
- Standard Terminal I/F

Graphics

- Standard VDI Interface

Disk System

- Shared File System
- Hierarchical Directories
- DOS 3.x Compatible
- File ownership

Networking

- IBM PC Networking Compatible (SMB)
- Transparent Access

Portability

- CPU and Device Independent Interface

Front Ends

- PC DOS 2.1 ⟶ 3.2

International Considerations

- International Character Sets
- Customizable Messages

Kernel Timings

Measured on IBM PC/AT @6 MHz w/1 wait state
estimated timing (x .6) @8 MHz w/ 0 wait states

	FlexOS 286	FlexOS 186	Flex OS 386
Process Dispatch (Different Process)	.185 msec	.098 msec	
Process Dispatch (Same Process)	.148	.098	
Tick Overhead (w/out Tick ISR)	.284	.275	
ISR Handler Overhead (no dispatch)	.048	.063	
DOASR	.042	.052	Expect on average a 2 – 2.5 times performance improvement
Interrupt to ISR	.036	.034	
ISR to ASR	.098	.090	
Last ASR to Process (Diff. Process)	.120	.074	
Last ASR to Process (Same Process)	.083	.074	
FLAGEVENT	.185	.173	
FLAGSET	.098	.094	