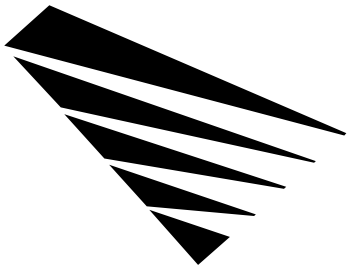

Chapter 2
NIOS Design

NIOS Architecture 6

Portable NIOS APIs 6
- Global Variables 7
- Configuration Services 8
- Debug Services 8
- Event Services 8
- Handle Management Services 8
- Hardware Interrupt Services 9
- Information Services 9
- Linked List Services 9
- Module Management Services 10
- Memory Management Services 10
- Popup Video Services 10
- Process Management Services 11
- Returnable Memory Management Services 11
- Statistists Services 11
- Time/Date Services 11
- Timer Services 12
- User Interface Services 12
- Utility Services 12
- Vxd Access Services 13

Supported NetWare OS API Calls 14

Overviews of Selected Services 17
- Queue Management Services Overview 17
- Handle Manager Services Overview 17

NIOS Architecture

NIOS is the backbone of the NetWare 32-bit client, functioning as the layer which isolates the client core (OS-independent) modules from the platform-specific modules and host operating system.

NIOS also serves as the core module manager, providing all the functionality necessary to load and unload modules as needed.

NIOS provides two major types of APIs: portable and OS-specific. The portable API is available on any NIOS Client-supported platform. Figure 2.1 shows an example of how OS-independent modules of the client use only the portable API. OS-specific modules may use either.

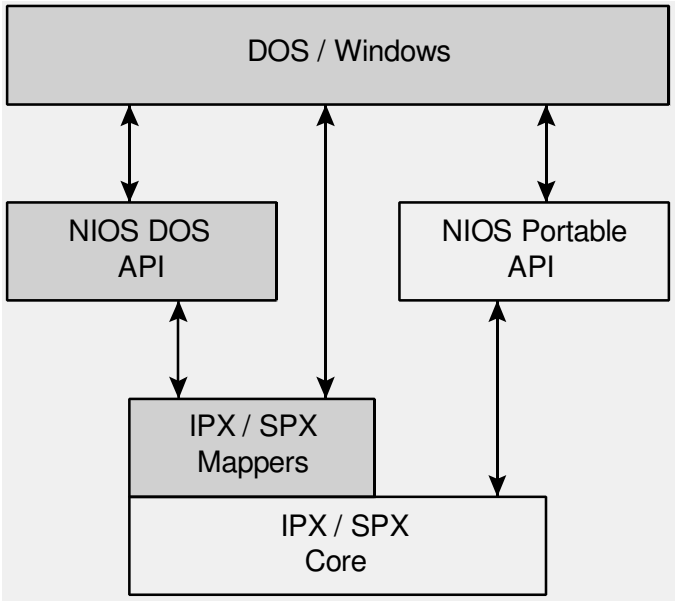

Figure 2.1: Example of NIOS portable and OS-specific APIs; shaded areas identify OS-specific parts.

Portable NIOS APIs

NIOS provides a robust set of APIs for client modules. These APIs are available on every platform supported by NIOS. Client NLMs that use these APIs exclusively are platform-independent.

Following is a summary of these OS-independent services grouped by functionality for easy reference and overview.

Note: If a Client NLM must make use of OS-dependent services, it is best to design source code modules such that OS-dependent and -independent components are easily separated. This is accomplished by designing well- defined and logical interfaces in one's modules so that OS-dependent and -independent components are logically divided in a way that facilitates porting.

Note that all NIOS functions that operate on characters or strings are fully double-byte character aware.

The API calls, described on the pages below, are divided into the following sets of services:

- Configuration Services
- Debug Services
- Event Services
- Handle Management Services
- Hardware Interrupt Services
- Information Services
- Linked List Services
- Module Management Services
- Memory Management Services
- Popup Video Services
- Process Management Services
- Returnable Memory Management Services
- Statistics Services
- Time/Date Services
- User Interface Services
- Utility Services

See Chapter 7 for a detailed description of each API and global variable listed below.

Global Variables

NiosMemLockFlag	UINT8	Set to non-zero when memory that is going to be accessed at interrupt time must be locked.
NiosSystemFlags	UINT32	Global variable containing various system flags.

Configuration Services

NiosCfgRead	Gets the value of the first occurrence of the specified keyword.
NiosCfgReadSpecific	Gets the value of the specified occurrence of the specified keyword.
NiosCfgWrite	Writes a keyword and value at the first occurrence of section name.
NiosCfgWriteSpecific	Writes a keyword at the specified occurrence of the section name.
NiosKeywordDeRegister	Deregisters a keyword from the system.
NiosKeywordEnumerate	Retrieves configuration keyword information.
NiosKeywordRegister	Registers a callback invoked when a keyword's value changes.
NiosKeywordResetValue	Resets a keyword value to the default.
NiosKeywordSetValue	Sets a keyword value.
NiosKeywordUpdateNetCfg	Write a keyword value to the configuration file.

Debug Services

NiosBreak	C macro that executes an INT 1 instruction.
NiosBreak3	Executes an interrupt 03h instruction.
NiosDebugCharInWait	Waits for user input from a debugging console.
NiosDebugCharInNoWait	Tests for user input from a debugger console.
NiosDebugCharOut	Displays a character on a debugger console.
NiosDebugStringOut	Displays a string on a debugger console.
NiosDeregisterDebugger	Deregisters an external debugger.
NiosDprintf	Provides a debug trace-out function.
NiosDprintfDisablePause	Disables pausing while information is output with Printf or Dprintf.
NiosDprintfEnablePause	Enables pausing while information is output with Printf or Dprintf.
NiosDprintfGetPauseMode	Returns the current pause mode setting.
NiosDprintfReset	Determines when the output should be paused. Sets line count to 0.
NiosPrintf	General purpose Printf function.
NiosRegisterDebugger	Registers an external debugger.

Event Services

NiosCancelForegroundEvent	Cancels a previously scheduled foreground event.
NiosScheduleForegroundEvent	Schedules an event that fires in a foreground context.

Handle Management Services

NiosAddressToHandle	Returns the handle associated with a 32-bit linear address.
NiosDeregisterHandleClient	Deregisters handle manager client.
NiosFreeHandle	Deallocates a handle fro a given linear address.
NiosGetHandle	Gets a handle for a given linear address.
NiosHandleToAddress	Returns a linear address for a given handle.
NiosListHandles	Enumerates on given client's handles.

NiosRegisterHandleClient Registers handle manager client.

Hardware Interrupt Services

CheckHardwareInterrupt Determines if the specified IRQ is requesting service (IRR).
DisableHardwareInterrupt Masks off the specified IRQ.
DoEndOfInterrupt Issues End-Of-Interrupt (EOI) for the specified IRQ.
EnableHardwareInterrupt Masks on the specified IRQ.
NiosHookHardwareInt Installs a handler for the specified IRQ.
NiosUnHookHardwareInt Deinstalls an IRQ handler.

Information Services

NiosEnableLogging Enables and disables logging.
NiosGetVersion Returns the environment type and NIOS version information.
NiosGetMemInfo Returns information about the NIOS memory allocator.

Linked List Services

NiosDFindNode Searches for a given node in a doubly linked queue.
NiosDLinkFirst Inserta a node into the front of a doubly linked queue in LIFO order.
NiosDLinkLast Inserts a node at end of a doubly linked queue.
NiosDLinkNext Inserts a node into a doubly linked queue after a given node.
NisoDLinkPrevious Inserts a node into a doubly linked queue before a given node.
NiosDNext Returns the forward link for a node in a doubly linked list.
NiosDNextNode Returns the forward link for a node in a doubly linked list or zero for no link.
NiosDPreviousNode Returns the backward link for a node in a doubly linked list.
NiosDUnlinkFirst Removes the first node from a doubly linked list.
NiosDUnlinkLast Removes the last node from a doubly linked list.
NiosDUnlinkNode Removes a node from a doubly linked list.
NiosDQueueInit Initializes a doubly linked queue.
NiosFindNode Tests if queue entry key is a member of a specified queue.
NiosLinkFirst Inserts a node into the front of a singly linked list.
NiosLinkLast Inserts a node at the end of a singly linked list.
NiosLinkNext Inserts a node after the specified node in a singly linked list.
NiosNextNode Takes a queue entry and returns the next entry in the queue.
NiosUnlinkFirst Unlinks the first queue entry from a singly linked queue.
NiosUnlinkNext Removes a node into the front of a singly linked list.
NiosUnlinkNode Unlinks a specified node from the queue.

Module Management Services

NiosCreateModuleHandle Gets a NIOS-environment module handle for a non-NLM module.
NiosDeportNlmApi Deletes an anonymous reference to the specified NLM API function.
NiosDestroyModuleHandle Destroys a module handle created by NiosCreateModuleHandle.
NiosEnumLoadedModules Enumerates the currently loaded modules.
NiosGetAddressOwner Determines which NLM owns the specified memory.
NiosGetModHandleFromName Locates the module handle for the specified named module.
NiosHookExportedApi Installs a different handler for a given exported API.
NiosImportNlmApi Determines the linear address of the specified NLM API name.
NiosLoadModule Loads and installs an NLM.
NiosUnHookExportedApi Deinstalls a previously hooked exported API.
NiosUnloadModule Unloads an NLM from the system.
NiosUnloadSelf Allows an NLM to unload itself.
NiosValidateModuleHandle Determines if an NLM module handle is valid.

Memory Management Services

NiosFree Deallocates a previously allocated memory block.
NiosGetMemInfo Returns information about the NIOS memory allocator.
NiosGetPhysLinearStart Returns a linear range that maps the entire physical address range.
NiosIsPhysContig Determines if the specified linear range is physically contiguous.
NiosLinToPhys Returns the physical address of a specified linear address.
NiosLongTermAlloc Allocates a memory block for long-term usage.
NiosMapPhysMemory Allocates a linear address range for a non-system physical range.
NiosPageLock Locks the specified memory region, keeping it present and fixed.
NiosPageUnlock Unlocks the specified memory region so that it can be demand-paged.
NiosPhysContigAlloc Allocates a physically contiguous memory block.
NiosShortTermAlloc Allocates a memory block for short-term usage.

Popup Video Services

NiosVidCreateDialogBox Creates a modeless dialog box (status box).
NiosVidDestroyDialogBox Destroys the previously created dialog box referenced by the handle parameter.
NiosVidInputDialogBox Displays an input dialog and handles the user input.
NiosVidMessageBox Displays a message box and handles the user input.
NiosVidUpdateDialogBox Updates the title and the prompt of the status dialog.

Process Management Services

NiosCreateSemaphore	Allocates a new semaphore.
NiosDestroySemaphore	Destroys a previously allocated semaphore.
NiosExamineSemaphore	Examines the current token count of the specified semaphore.
NiosGetCurrProcessGroupId	Returns the ID assigned to the currently executing process group.
NiosGetCurrProcessId	Returns the ID assigned to the currently executing procees.
NiosGetProcessName	Returns a displayable description of the specified process.
NiosPoll	Yields to other waiting processes.
NiosSignalSemaphore	Performs an "up" operation on a semaphore.
NiosThreadArmId	Initializes for a subsequent call to **NiosThreadBlockOnId**.
NiosThreadBlockOnId	Blocks currently running thread of execution until specified id is signaled.
NiosThreadSignalId	Unblocks the thread currently blocked on the specified id.
NiosWaitSemaphore	Performs a "down" operation on a semaphore.

Returnable Memory Management Services

NiosMemPoolCheckAvail	Returns number of blocks in memory pool which are not allocated.
NiosMemPoolDeRegister	Deregisters a module with the memory pool manager.
NiosMemPoolEnum	Enumerates all memory blocks held by an application.
NiosMemPoolFindBlock	Look-up or allocate a block of memory.
NiosMemPoolFreeBlock	Releases a memory block.
NiosMemPoolGetSize	Determines how many blocks are available to the system or the application.
NiosMemPoolGetVersion	Retrieves version and memory option information.
NiosMenPoolHold	Increments the hold count on a memory block.
NiosMenPoolMakeMRU	Same as FindBlock function with MP_MAKE_MRU option.
NiosMenPoolRegister	Registers a module with the memory pool manager.
NiosMemPoolTestHold	Returns the number of holds placed on a memory block.
NiosMemPoolUnhold	Decrements the hold count on a memory block.

Statistists Services

NiosStatDeRegister	Removes an entry from the registry.
NiosStatEnumerate	Enumerates through available statistics tables.
NiosStatGetTable	Retrieves specific statistics table in condensed form.
NiosStatRegister	Creates an entry in the statistics registry.
NiosStatResetTable	Sets all UINT32 and UINT64 counters to zero for the requested table.

Time/Date Services

NiosGetDateTime	Returns the current date and time.
NiosSetDateTime	Sets the system date and time.

Timer Services

NiosCancelAESEvent Cancels the specified outstanding AES event.
NiosCancelAllModuleAESEvents Cancels all outstanding AES events for the specified module.
NiosGetHighResIntervalMarker Returns a high resolution timer marker accurate to 1 microsecond.
NiosGetIntervalMarker Returns a timer marker accurate to 55ms. Units are in milliseconds.
NiosGetTickCount Returns a timer marker accurate to 55ms. Units are in 1/18 s.
NiosScheduleAESEvent Schedules an event to fire after a specified amount of time.

User Interface Services

NiosDeregisterStdOutHandler Deregisters a StdOut handler.
NiosPrintf Formats and displays strings.
NiosRegisterStdOutHandler Registers a handler to receive StdOut message notifications.

Utility Services

NiosChar Returns the size of a character.
NiosCli C macro that clears the interrupt flag.
NiosEatWhite Removes leading white space from a string.
NiosHexCharToByte Converts hex alpha numeric character into a byte.
NiosMemCmp Case-sensitive memory compare.
NiosMemCmpi Case-insensitive memory compare.
NiosMemCpy Copies the contents of one memory buffer to another.
NiosMemSet Initializes a memory buffer to a given value.
NiosNextChar Advances a string pointer by one character.
NiosPopfd C macro restores the Eflags register to specified value.
NiosPrevChar Backs up a string pointer by one character.
NiosPrintf A double-byte-character-aware printf function.
NiosPushfd C macro that returns the current value of the Eflags register.
NiosPushfdCli C macro that returns the current Eflags register.
NiosSprintf String formatting service.
NiosSti C macro that sets the interrupt flag.
NiosStrChr Searches a string for a given character.
NiosStrCmp Case-sensitive string compare.
NiosStrCmpi Case-insensitive string compare.
NiosStrCpy Copies the contents of one string to another.
NiosStrLwr Converts all uppercase characters in a specified string to lowercase.
NiosStrtoByteArray Converts ASCIIZ numeric string into a byte array.
NiosStrtoul Converts a string to a value in the given radix.
NiosStrUpr Converts all lowercase characters in a specified string to uppercase.
NiosTestCharBoundary Determines if a string pointer bisects a double-byte character.

NiosToLower	Converts an uppercase character to lowercase.
NiosToUpper	Converts a lowercase character to uppercase.
NiosUltoa	Converts a value to a displayable string in the given radix.

Vxd Access Services

NiosVxdBeginNlmUse	Determines if the specified NLM is present and builds a Vxd/NLM dependency.
NiosVxdEndNlmUse	Destroys the Vxd/NLM dependency allowing the NLM to be unloaded.
NiosVxdGetVersion	Returns NIOS version information and signals initialization complete.

Supported NetWare OS API Calls

The following list of NetWare OS API function calls are supported by the 32-bit client architecture:

Note: Entries preceded by "&" are data variables.

&IOConfigurationList
AddPollingProcedureRTag
Alloc
AllocateResourceTag
CancelInterruptTimeCallBack
CancelNoSleepAESProcessEvent
CancelSleepAESProcessEvent
CFindResourceTag
CheckHardwareInterrupt
ClearHardwareInterrupt
CloseFile
CPSemaphore
CRescheduleLast
CVSemaphore
DeRegisterHardwareOptions
DisableHardwareInterrupt
DoEndOfInterrupt
DoRealModeInterrupt
EnableHardwareInterrupt
Free
GetCurrentTime
GetDebuggerActiveCount
GetFileServerMajorVersionNumber
GetFileServerMinorVerionsNumber
GetFileSize
GetHardwareBusType
GetNestedInterruptLevel
GetNLMNames
GetNLMVersionInfo
GetProcessorSpeedRating
GetRealModeWorkSpace
GetServerConfigurationType
GetServerPhysicalOffset
GetSuperHighResolutionTimer
GetSystemMemoryMap
ImportPublicSymbol
INWDOSClose

lNWDOSLSeek
KillMe
OpenFileUsingSearchPath
OutputToScreen
ParseDriverParameters
ReadEISAConfig
RegisterForEventNotification
RegisterHardwareOptions
RemovePollingProcedure
ReturnMessageInformation
ScheduleInterruptTimeCallBack
ScheduleNoSleepAESProcessEvent
ScheduleSleepAESProcessEvent
SetHardwareInterrupt
UnImportPublicSymbol
UnRegisterEventNotification

NetWare 3.11 Only Publics

AllocBufferBelow16Meg
AllocSemiPermMemory
FreeBufferBelow16Meg
FreeSemiPermMemory
MapAbsoluteAddressToDataOffset
MapDataOffsetToAbsoluteAddress
MapCodeOffsetToAbsoluteAddress
QueueSystemAlert

Supported NSI/NBI Calls

CDoBusEndOfInterrupt
ClearBusInterrupt
DMACleanup
DMAStart
DMAStatus
FreeBusMemory
GetAlignment
GetBusInfo
GetBusName
GetBusTag
GetBusType
GetCardConfigInfo
In8

In16
In32
In64
InBuff8
InBuff16
InBuff32
InBuff64
MapBusMemory
MaskBusInterrupt
MovFastFromBus
MovFastToBus
MovFromBus8
MovFromBus16
MovFromBus32
MovFromBus64
MovToBus8
MovToBus16
MovToBus32
MovToBus64
Out8
Out16
Out32
Out64
OutBuff8
OutBuff16
OutBuff32
OutBuff64
Rd8
Rd16
Rd32
Rd64
ReadPhysical
ScanBusInfo
ScanInterruptInfo
SearchAdapter
Set8
Set16
Set32
Set64
SetBusInterrupt
Slow
UnMaskBusInterrupt
WritePhysical
Wrt8
Wrt16

Wrt32
Wrt64

Overviews of Selected Services

Some of the chapters in this document focus on specific sets of services, such as Chapter 4 "Memory Pool Services" and Chapter 5 "Popup Video Services", because these services require extensive explaination. Other services need very little explaination.

In this section, the focus is on two sets of services which need only a brief overview: Queue Management Services and Handle Management Services.

Queue Management Services Overview

NIOS provides a set of linked list management routines that supports both singly and doubly linked lists that are linear (non-circular). Routines are provided for inserting elements at the start and end of a specified queue. There are also routines that provide standard insert, traverse and removal operations within a list.

A set of helper macros are provided in the include file NIOSQ.H that support standard link list operations via macros to minimize latency in making a direct call to perform these operations.

Queue nodes must have a forward link for singly linked lists and forward and backward links for doubly link lists. The offsets to these fields must be provided when using a queueing call. A queueing structure is required to maintain the queue pointers (first and last node) of the list that is requiring the link list operation.

Use of these functions in time critical code is not suggested.

Handle Manager Services Overview

The handle manager provides a mechanism for providing handle dereferencing of linear address space. These API are useful when a service provider needs to supply information to a client, but does not (or cannot) supply the true linear address of the information.

The handle manager supplies routines for:

- Allocating handles
- Freeing handles
- Listing handles associated with a given service provider
- Finding the linear address associated with a given handle
- Finding a handle associated with a given linear address

To utilize the handle manager the service provider (client of the handle manager) must register for the service and when the service is no longer required a de-registration mechanism is provided.

The following 7 APIs which are used to read and write to a configuration file:

NiosRegisterHandleClient
NiosDeregisterHandleClient
NiosGetHandle
NiosFreeHandle
NiosListHandles
NiosHandleToAddress
NiosAddressToHandle