

Index

Symbols

& operator 1-34
@ operator 2-25

Numerals

0 (memory location) 1-4, 1-35
0-length handles 1-34
24-bit addressing 3-5 to 3-7, 4-7 to 4-8
 defined 1-15
 setting with the Memory control panel 4-5
 stripping flag bits 4-21 to 4-23
32-bit addressing 3-7 to 3-9, 4-8
 defined 1-15
 machines that support 4-5
 setting with the Memory control panel 4-5
 using temporarily 4-20
32-bit clean 1-16

A

A5 register
 and A5 world 1-13, 4-5 to 4-6
 grow-zone functions saving and restoring 1-49, 4-14
 setting and restoring 1-78 to 1-79, 4-14, 4-24 to 4-25
 use of by Toolbox and Operating System routines 4-14
 using to access QuickDraw globals 4-18 to 4-19
A5 world
 accessing in completion routines 4-14 to 4-15
 accessing in interrupt tasks 4-16 to 4-17
 defined 1-12, 1-13
 setting 1-78 to 1-79, 4-24 to 4-25
addresses. *See* memory addresses
addressing modes
 24-bit 4-7
 32-bit 4-7 to 4-8
 current mode, getting 4-26
 switching 4-20 to 4-21, 4-26 to 4-27
Address Management Unit (AMU) 3-5
address space. *See* logical address space; physical address space
address-translation mode
 getting 4-26

 setting 4-26 to 4-27
 temporarily changing 4-20
AMU (Address Management Unit) 3-5
AND operator 1-34
AppleShare, and paging devices 3-5
application global variables 1-12
 accessing in completion routines 4-14
 accessing in interrupt tasks 4-17
application heap 1-9 to 1-11
 defined 1-10
 determining amount of free space 1-42 to 1-44
 maximizing space to prevent fragmentation 1-40
 setting up 1-38 to 1-42, 1-50 to 1-52, 2-27 to 2-29
application heap limit
 getting 1-53, 2-84
 setting 1-53 to 1-54, 2-84 to 2-85
application heap zone. *See also* heap zones
 defined 2-5
 getting a pointer to 2-81
 initializing 2-87 to 2-88
 maximizing size of 1-51, 2-27
 subdividing into multiple heap zones 2-14 to 2-16
application parameters 1-13
application partitions 1-4, 1-7 to 1-13
ApplicationZone function 2-81
ApplLimit global variable 1-8, 1-40, 1-53, 2-84
ApplZone global variable 2-81

B

backing-store files
 defined 3-5
 volume specified in Memory control panel 4-5
backing volume. *See* paging device
block contents 2-22
block headers 2-22 to 2-24
BlockMove procedure 1-74 to 1-75, 2-59 to 2-60
_BlockMove trap, flushing instruction cache 4-10
blocks, memory. *See also* nonrelocatable blocks;
 relocatable blocks
 allocating 1-44 to 1-46
 concatenating 2-64 to 2-66
 copying 1-74 to 1-75, 2-59 to 2-64
 defined 1-10
 how allocated 1-22
 manipulating 2-59 to 2-66
 releasing 1-44 to 1-46

size correction for 2-23, 2-24
 Boolean operators, short-circuit 1-34
 BufPtr global variable 2-14
 limitation on lowering during startup 2-85
 bus-error vectors 3-22
 Byte data type 2-25

C

caches. *See* data cache; disk cache; instruction cache
 callback routines
 and code segmentation 1-32 to 1-33
 maintaining the A5 register in 4-14 to 4-15
 click-loop routines, and the A5 register 4-15
 code resources, copying into system heap 2-13
 code segmenting
 and dangling pointers 1-31 to 1-32
 effect on callback routines 1-32 to 1-33
 compacting heap zones 2-71 to 2-73
 compaction. *See* heap compaction
 CompactMem function 2-71 to 2-72
 CompactMemSys function 2-72 to 2-73
 completion routines
 deferred under virtual memory 3-12
 maintaining the A5 register in 4-14 to 4-15
 concatenating memory blocks 2-64 to 2-66
 concurrent drivers 3-11
 control action procedures, and the A5 register 4-15
 control definition procedures, and the A5 register 4-15
 control panels, Memory. *See* Memory control panel
 copy-back cache 4-12
 copying memory blocks 1-74 to 1-75, 2-59 to 2-64
 CurrentA5 global variable 1-79, 4-25
 and callback routines 4-15
 defined 1-13
 getting value 1-79, 4-25
 current heap zone 2-5
 CurStackBase global variable 2-104
 cushions. *See* memory cushions

D

dangling pointers
 avoiding 1-29 to 1-33
 causes of 1-29 to 1-33
 dangling procedure pointers 1-32 to 1-33
 defined 1-29
 detecting 1-29
 introduced 1-20
 locking blocks to prevent 1-29 to 1-30
 referencing callback routines 1-32 to 1-33

using local variables to prevent 1-31
 data cache 4-30 to 4-31
 and virtual memory 3-21
 defined 4-9
 flushing 4-9, 4-12
 DebuggerEnter procedure 3-23, 3-35
 DebuggerExit procedure 3-23, 3-35 to 3-36
 DebuggerGetMax function 3-34 to 3-35
 DebuggerLockMemory function 3-21, 3-23, 3-37
 DebuggerPoll procedure 3-23, 3-39
 debuggers, and virtual memory 3-21 to 3-24
 DebuggerUnlockMemory function 3-21, 3-23, 3-38
 _DebugUtil trap 3-22, 3-45
 deferred tasks, and the A5 register 4-16
 DeferUserFn function 3-33
 introduced 3-21
 using 3-20 to 3-21
 dereferenced handles 1-29
 DeskHook global variable
 clearing in Pascal 2-9
 and displaying windows during startup time 2-9
 DetachResource procedure 2-13
 device drivers, avoiding page faults 3-12
 dialog boxes, and low-memory situations 1-44
 direct memory access (DMA) 3-3, 3-13, 3-15, 3-16, 3-18,
 3-20, 3-21, 4-3, 4-10
 and stale data 4-12
 disk cache
 defined 4-4
 setting with the Memory control panel 4-4
 disposed handles
 checking for 1-33
 defined 1-33
 preventing dereferencing of 1-33
 problems using 1-33
 DisposeHandle procedure 1-46, 1-57, 2-34 to 2-35
 DisposePtr procedure 1-46, 1-60, 2-38 to 2-39
 DMA. *See* direct memory access
 double indirection 1-18
 double page faults 3-11 to 3-12, 3-14
 duplicating relocatable blocks 2-62 to 2-64

E

EmptyHandle procedure 1-67 to 1-68, 2-51 to 2-52
 used by a grow-zone function 1-49
 empty handles
 checking for 1-34
 defined 1-34

F

fake handles 1-35 to 1-36, 1-55, 2-30
 creating 1-35, 1-36
 defined 1-35
 problems using 1-35, 1-55, 2-30

Finder, allocation of memory for disk copying 2-9

flag bits
 master pointer 4-7
 stripping 4-7, 4-27

FlushCodeCache procedure 4-31 to 4-32

FlushCodeCacheRange function 4-32 to 4-33

FlushDataCache procedure 4-31

flushing
 data cache 4-9, 4-12, 4-31
 instruction cache 4-9 to 4-10, 4-29 to 4-30, 4-31 to 4-33

FlushInstructionCache procedure 4-30

fragmentation. *See* heap fragmentation

FreeMem function 2-66 to 2-67

FreeMemSys function 2-67

free space
 assessing 2-66 to 2-70
 assessing availability for temporary memory 2-79 to 2-80

G

gaps in heaps, danger of 1-25

GetApplLimit function 1-53, 2-84

GetHandleSize function 2-39 to 2-40

GetMMUMode function 4-26

GetNextEvent function, and temporary memory 2-10

GetPageState function 3-24, 3-39 to 3-40

GetPhysical function 3-31 to 3-33
 and discontinuous physical address space 3-11
 introduced 3-16
 using 3-16 to 3-20

GetPtrSize function 2-41 to 2-42

GetZone function 2-80

global variables. *See* application global variables;
 system global variables; QuickDraw global variables

grow-zone functions 1-48 to 1-49, 1-80 to 1-81, 2-89 to 2-90
 and the A5 register 4-15
 defined 1-38
 example of 1-49, 4-15
 finding protected block 1-78, 2-77
 setting 1-77 to 1-78, 2-76 to 2-77
 using SetA5 function 1-81, 2-90
 using SetCurrentA5 function 1-81, 2-90

GZRootHnd global variable 1-78, 2-77

GZSaveHnd function 1-49, 1-78, 2-77

H

HandAndHand function 2-64 to 2-65

Handle data type 1-18, 2-25

handles
 See also relocatable blocks
 checking validity of 1-34
 defined 1-18 to 1-19
 recovering 2-54 to 2-55
 relative 2-23

HandleZone function 2-82 to 2-83

HandToHand function 2-62 to 2-64

HClrRBit procedure 2-50 to 2-51

heap compaction
 defined 1-11, 1-23
 movement of relocatable blocks during 1-24
 routines for 2-71 to 2-73, 2-74 to 2-76

HeapEnd global variable 2-104

heap fragmentation
 causes of 1-25 to 1-28
 defined 1-10
 during memory reservation 1-25
 maximizing heap size to prevent 1-40
 preventing 1-24 to 1-28
 summary of prevention 1-28

heap purging 1-21 to 1-22
 routines for 2-73 to 2-76

heap. *See* application heap; system heap

heap zones
 accessing 2-80 to 2-83
 changing 2-81
 defined 2-5
 getting current zone 2-80
 initializing 2-86 to 2-87
 manipulating 2-83 to 2-89
 organization of 2-19 to 2-22
 See also zone headers; zone trailers
 subdividing into multiple heap zones 2-14 to 2-16

HFS RAM Cache panel 4-4

HGetState function 1-30, 1-61 to 1-62, 2-43 to 2-44

high memory, allocating at startup time 2-13 to 2-14

HLockHi procedure 1-73, 2-58 to 2-59

HLock procedure 1-30, 1-63 to 1-64, 2-45 to 2-46

HNoPurge procedure 1-66 to 1-67, 2-48 to 2-49

holding physical memory 3-14

HoldMemory function 3-14, 3-25 to 3-26

HPurge procedure 1-65 to 1-66, 2-47 to 2-48

HSetRBit procedure 2-49 to 2-50

HSetState procedure 1-30, 1-62 to 1-63, 2-44 to 2-45

HUnlock procedure 1-64 to 1-65, 2-46 to 2-47

_HWPriv trap macro 4-36

I

InitApplZone procedure 2-87 to 2-88
 initializing new heap zones within other heap zones 2-14 to 2-16
 InitZone procedure 1-81, 2-86 to 2-87, 2-90
 instruction cache
 defined 4-8
 flushing 4-9 to 4-10, 4-29 to 4-30, 4-31 to 4-33
 interprocess buffers, and temporary memory 2-10
 interrupts, nonmaskable 3-23
 interrupt tasks
 and Memory Manager routines 1-50, 2-26
 deferring under virtual memory 3-12
 maintaining the A5 register 4-16 to 4-17
 and temporary memory 2-10
 interrupt time
 avoiding Memory Manager routines at 1-50, 2-26
 deferring code execution under virtual memory 3-20
 I/O completion routines, and the A5 register 4-15
 ISP. *See* stack pointer, interrupt

J

jump table 1-13
 jump table entries
 and stale instructions 4-10
 for callback routines 1-32

L

linked lists, allocating new elements in 1-31
 loading code segments, and dangling pointers 1-31 to 1-32
 _LoadSeg trap, flushing instruction cache 4-10
 locking physical memory
 debugger routine 3-37
 defined 3-13
 routines for 3-28 to 3-30
 locking relocatable blocks 1-20 to 1-21, 1-63 to 1-64, 2-45 to 2-46
 LockMemoryContiguous function 3-16, 3-29 to 3-30
 LockMemory function 3-28
 and stale data 4-13
 introduced 3-15
 logical address space 3-5 to 3-9
 possible fragmentation of 3-7
 size of with 24-bit addressing 3-5
 size of with 32-bit addressing 3-7
 translating to physical address space 3-11
 logical sizes of blocks 2-22

LogicalToPhysicalTable data structure 3-17, 3-25
 logical-to-physical translation table. *See* translation table
 low-memory conditions 1-36 to 1-38
 low-memory global variables
 See system global variables

M

master pointer blocks 1-18
 master pointer flag bits 4-7
 master pointers
 allocating manually 1-51 to 1-52, 2-28 to 2-29
 comparing 4-22
 defined 1-18
 determining how many to preallocate 1-41 to 1-42
 number per block in application zone 1-41
 running out of 1-41
 MaxApplZone procedure 1-51, 2-27
 and ApplLimit global variable 1-8
 automatic execution of 1-40, 2-16
 and heap fragmentation 1-40
 MaxBlock function 2-67 to 2-68
 MaxBlockSys function 2-68
 maximizing heap zone space 2-74 to 2-76
 MaxMem function 2-74 to 2-75
 MaxMemSys function 2-75 to 2-76
 maxSize constant 2-72
 MC680x0 microprocessor
 data cache 4-9
 instruction cache 4-8, 4-9
 size of memory blocks with 2-22
 MemErr global variable 1-50, 1-76, 2-26, 2-71
 MemError function 1-50, 1-76, 2-26, 2-70 to 2-71
 memory
 allocating and releasing 1-54 to 1-60, 2-29 to 2-39
 allocating during startup 2-13 to 2-14
 assessing 2-66 to 2-83
 changing sizes of blocks 2-39 to 2-43
 freeing 2-71 to 2-76
 holding 3-13, 3-14
 organization of 1-4 to 1-13, 2-19 to 2-24
 releasing 3-15
 See also temporary memory; virtual memory
 memory addresses
 comparing 4-8, 4-22
 converting to 32-bit mode 4-7, 4-21 to 4-24, 4-26 to 4-27
 mapping logical to physical 3-16 to 3-20
 stripping flag bits from 4-7, 4-21 to 4-23, 4-27
 translating 4-23 to 4-24, 4-28
 MemoryBlock data structure 3-17, 3-24
 memory-block record 3-17

memory blocks. *See* blocks, memory

memory configuration, obtaining information about 3-14

Memory control panel 3-4, 3-5, 4-3 to 4-5

- addressing mode controls 4-5
- disk cache controls 4-4
- illustrated 4-4
- introduced 4-3
- RAM disk controls 4-5
- virtual memory controls 4-5

memory cushions

- defined 1-37
- determining optimal size of 1-43
- maintaining 1-43 to 1-44

`_MemoryDispatchA0Result` trap macro 3-45

`_MemoryDispatch` trap macro 3-20, 3-44

memory management unit (MMU) 3-5

Memory Manager 2-3 to 2-105

- 24-bit 1-15
- 32-bit 1-15
- allocating master pointers 1-41
- and application heap 1-10 to 1-11
- application-defined routines 2-89 to 2-92
- calling grow-zone function 1-48
- capabilities of 2-4
- compacting heap 1-23 to 1-24
- data types 1-17 to 1-18, 2-24 to 2-26
- defined 2-3
- movement of blocks by 1-24
- purging heap 1-23 to 1-24
- reserving memory 1-22 to 1-23, 2-55 to 2-56
- returning result codes 1-50, 1-76, 2-26, 2-70 to 2-71
- routines 2-26 to 2-89
- testing for features 2-11 to 2-12

memory reservation. *See* reserving memory

memory reserves

- benefits of 1-37
- defined 1-37
- maintaining 1-46 to 1-48

`MemTop` global variable 2-14, 2-86

menu definition procedures, and the A5 register 4-15

MMU (memory management unit) 3-5

`MoreMasters` procedure 1-41 to 1-42, 1-51 to 1-52, 2-28 to 2-29

`MoveHHi` procedure 1-26 to 1-27, 1-71 to 1-72, 2-56 to 2-58

moving relocatable blocks high 1-26 to 1-27, 1-71 to 1-73, 2-56 to 2-59

multiple heap zones

- implementing 2-14 to 2-16
- uses for 2-6

N

`NewEmptyHandle` function 2-33

`NewEmptyHandleSys` function 2-34

`NewHandleClear` function 1-45, 1-56, 2-31 to 2-32

`NewHandle` function 1-44, 1-55 to 1-56, 2-29 to 2-31

`NewHandleSysClear` function 2-32

`NewHandleSys` function 2-31

`NewPtrClear` function 1-59, 2-37 to 2-38

`NewPtr` function 1-44, 1-58 to 1-59, 2-36 to 2-37

`NewPtrSysClear` function 2-38

`NewPtrSys` function 2-37

nonessential memory requests, checking whether to satisfy 1-43

nonmaskable interrupts 3-23

nonrelocatable blocks

- .See also* blocks, memory
- advantages of 1-20
- allocating 1-28, 1-58 to 1-59, 2-36 to 2-38
- allocating temporarily 1-28
- data type for 1-18
- defined 1-17
- disposal and reallocation of 1-25
- releasing 1-60, 2-38 to 2-39
- sizing 2-41 to 2-43
- when to allocate 1-27 to 1-28

Notification Manager, and the A5 world 4-16 to 4-17

notification response procedures, and the A5 register 4-16

O

`OpenResFile` function, calling `StripAddress` on filenames 4-22

`OpenRFPPerm` function, calling `StripAddress` on filenames 4-22

operating system queues, storing elements in system heap zone 2-12

ordered address comparisons 4-22

original application heap zone 2-5

`_OSDispatch` trap macro 2-104

P

Paged Memory Management Unit (PMMU) 3-5

`PageFaultFatal` function 3-22, 3-36

page faults

- defined 3-11
- handling 3-20
- intercepted by Virtual Memory Manager 3-11 to 3-12, 3-22

- protection against 3-12, 3-14
 - .See also* double page faults
- pages, memory
 - defined 3-4
 - holding 3-14, 3-25
 - locking 3-15, 3-28
 - locking contiguously 3-29
 - releasing 3-15, 3-27
 - unlocking 3-30
- PageState data type 3-24
- paging 3-4
- paging device 3-5
- partitions 1-4
 - .See also* application partitions; system partition
- patches, and stale instructions 4-10
- physical address space 3-9 to 3-11
 - discontiguous 3-9
- physical memory 3-14 to 3-20
 - holding pages in 3-14 to 3-15
 - locking pages in 3-15 to 3-16
 - releasing pages 3-15
 - unlocking pages 3-16
- physical sizes of blocks 2-22
- PMMU (Paged Memory Management Unit) 3-5
- pointers 1-17 to 1-18
 - .See also* nonrelocatable blocks; dangling pointers
- Process Manager, and callback routines 4-14
- processor caches 4-8 to 4-13, 4-29 to 4-33
 - .See also* data cache; instruction cache
- ProcPtr data type 2-25 to 2-26
 - and code segmentation 1-32 to 1-33
 - referencing code in code resources 2-13
- program counter, fixing before switching to 32-bit mode 4-21
- protected blocks
 - defined 1-49
 - determining which they are 1-81, 2-90
 - handle to returned by GZSaveHnd 1-78, 2-77
- PtrAndHand function 2-65 to 2-66
- Ptr data type 1-17, 2-25
- PtrToHand function 2-60 to 2-61
- PtrToXHand function 2-61 to 2-62
- PtrZone function 2-83
- PurgeMem procedure 2-73 to 2-74
- PurgeMemSys procedure 2-74
- PurgeSpace procedure 1-75, 2-68 to 2-69
- purge-warning procedures 2-16 to 2-18, 2-21, 2-90 to 2-92
 - defined 2-16
 - installed by SetResPurge 2-18, 2-91
 - restrictions on 2-91
 - sample 2-17
 - using SetA5 function 2-91
 - using SetCurrentA5 function 2-91
- purging heap zones 1-24, 2-73 to 2-74

purging relocatable blocks 1-21 to 1-22

Q

- QuickDraw global variables
 - defined 1-13
 - reading in stand-alone code 4-18 to 4-19
 - structure of 4-18
 - using in stand-alone code 4-18 to 4-19

R

- RAM cache. *See* disk cache
- RAM disks
 - defined 4-5
 - setting size of with Memory control panel 4-5
- _Read trap, flushing instruction cache 4-10
- ReallocateHandle procedure 1-68 to 1-69, 2-52 to 2-53
- reallocating relocatable blocks 1-21 to 1-22
- RecoverHandle function 2-54 to 2-55
- reference constant fields
 - using to store A5 value 4-17
- relative handles 2-23
- releasing held pages 3-15
- relocatable blocks
 - .See also* blocks, memory; handles
 - allocating 1-55 to 1-56, 2-29 to 2-34
 - changing properties 1-60 to 1-67, 2-43 to 2-51
 - clearing resource bit 2-50 to 2-51
 - concatenating 2-64 to 2-65
 - data type for 1-17
 - defined 1-17
 - disadvantages of 1-20
 - duplicating 2-62 to 2-64
 - emptying 1-67 to 1-68, 2-51 to 2-52
 - getting properties 1-61 to 1-62, 2-43 to 2-44
 - in bottom of heap zone 1-25
 - locking 1-20 to 1-21, 1-63 to 1-64, 2-45 to 2-46
 - for long periods of time 1-28
 - for short periods of time 1-28
 - making purgeable 1-65 to 1-66, 2-47 to 2-48
 - making un-purgeable 1-66 to 1-67, 2-48 to 2-49
 - managing 1-67 to 1-73, 2-51 to 2-59
 - master pointers after disposing 1-33
 - master pointers for 1-41
 - moving around nonrelocatable blocks 1-24
 - moving high 1-26 to 1-27, 1-71 to 1-73, 2-56 to 2-59
 - properties of 1-20 to 1-22
 - purging 1-21 to 1-22
 - reallocating 1-21 to 1-22, 1-68 to 1-69, 2-52 to 2-53

- releasing 1-57, 2-34 to 2-35
- restrictions on locked blocks 1-27
- setting properties 1-62 to 1-67, 2-44 to 2-51
- setting resource bit 2-49 to 2-50
- sizing 2-39 to 2-41
 - movement during 1-24
- unlocking 1-20 to 1-21, 1-64 to 1-65, 2-46 to 2-47
- when to lock 1-28
- removable disks, and virtual memory 3-5
- ReserveMem procedure 1-70 to 1-71, 2-55 to 2-56
- ReserveMemSys procedure 2-56
- reserves. *See* memory reserves
- reserving memory 1-22 to 1-23
 - and heap fragmentation 1-25
 - defined 1-22
 - for relocatable blocks 1-26
 - limitation of 1-25
 - routines 2-55 to 2-56
- resource bit
 - clearing 2-50 to 2-51
 - setting 2-49 to 2-50
- Resource Manager, installing purge-warning procedures 2-18, 2-91
- resource types
 - 'SIZE' 1-13
 - 'sysz' 2-13
- result codes for Memory Manager routines 1-50, 1-76, 2-26, 2-70 to 2-71

S

- self-modifying code, and stale instructions 4-10
- SetA5 function 1-79, 4-14, 4-25
 - used in a grow-zone function 1-81, 2-90
 - used in a purge-warning procedure 2-91
- SetApplBase procedure 2-88 to 2-89
- SetApplLimit procedure 1-53 to 1-54, 2-84 to 2-85
 - using to increase size of stack 1-40
- SetCurrentA5 function 1-79, 4-25
 - used in a grow-zone function 1-81, 2-90
 - used in a purge-warning procedure 2-91
- SetGrowZone procedure 1-77 to 1-78, 1-81, 2-76 to 2-77, 2-90
- SetHandleSize procedure 2-40 to 2-41
- SetPtrSize procedure 2-42 to 2-43
- SetResPurge procedure, installing purge-warning procedures 2-18
- SetZone procedure 2-81
- short-circuit Boolean operators 1-34
- SignedByte data type 1-17, 2-25
- size correction for blocks 2-23, 2-24
- Size data type 2-26
- 'SIZE' resource type, specifying partition size 1-13

- slot-based VBL tasks, deferred under virtual memory 3-12
- stack
 - collisions with the heap 1-8
 - default size of 1-40
 - defined 1-8
 - determining available space 2-69
 - increasing size of 1-39 to 1-40
- stack frame 1-9
- stack pointer
 - interrupt (ISP) 3-23
 - user (USP) 3-23
- stack sniffer 1-8
- StackSpace function 2-69 to 2-70
- stale data
 - avoiding problems with 4-13
 - defined 4-10
- stale instructions
 - avoiding problems with 4-9
 - defined 4-9
- stand-alone code resources, changing address-translation mode in 4-20
- startup process
 - allocating memory during 2-13 to 2-14
 - displaying windows during 2-9
- Str255 data type 2-25
- StringHandle data type 2-25
- StringPtr data type 2-25
- StripAddress function 4-21 to 4-23, 4-27 to 4-28
- supervisor mode 3-23
- SwapDataCache function 4-30 to 4-31
- SwapInstructionCache function 4-29
- SwapMMUMode procedure 4-26 to 4-27
 - calling from stand-alone code 4-20
- SysEqu.p interface file 2-7
- system extensions, allocating memory at startup time 2-13
- system global variables
 - changing 2-9
 - defined 1-6 to 1-7, 2-6
 - reading 2-8 to 2-9
 - uses of 2-6 to 2-7
- system heap 1-6
 - defined 1-6
 - held in RAM under virtual memory 3-12
- system heap zone
 - allocating memory in 2-12
 - creating new heap zones within 2-16
 - defined 2-5
 - getting a pointer to 2-82
 - installing interrupt code into 2-13
 - uses for 2-5
- system partition 1-4 to 1-7
 - .See also* system heap; system global variables
- SystemZone function 2-82

SysZone global variable 2-82
 'sysz' resource type 2-13

T

tag bytes 2-23
 TempFreeMem function 2-79
 TempMaxMem function 2-79 to 2-80
 TempNewHandle function 2-78
 temporary memory
 allocating 2-10 to 2-11
 confirming success of allocation 2-10
 defined 1-13, 2-4
 determining zone of 2-10
 limitation on locking 2-10
 operating on blocks 2-5
 optimal usage of 2-5
 release of during application termination 2-10
 routines 2-77 to 2-80
 testing for features of 2-11 to 2-12
 tracking of 2-10
 using as a heap zone 2-16
 TheZone global variable 2-80
 32-bit addressing 3-7 to 3-9, 4-8
 defined 1-15
 machines that support 4-5
 setting with the Memory control panel 4-5
 using temporarily 4-20
 32-bit clean 1-16
 THz data type 2-20
 Time Manager tasks
 and the A5 register 4-16
 deferred under virtual memory 3-12
 TopMem function 2-14, 2-85 to 2-86
 Translate24To32 function 4-23 to 4-24, 4-28 to 4-29
 translating logical to physical addresses 3-16 to 3-20,
 3-31 to 3-33
 translation tables 3-17, 3-25
 trap patches, and the A5 register 4-15
 24-bit addressing 3-5 to 3-7, 4-7 to 4-8
 defined 1-15
 setting with the Memory control panel 4-5
 stripping flag bits 4-21 to 4-23

U

UnholdMemory function 3-15, 3-27
 _UnloadSeg trap, flushing instruction cache 4-10
 unlocking physical memory 3-16, 3-30 to 3-31
 debugger routine 3-38

unlocking relocatable blocks 1-20 to 1-21, 1-64 to 1-65,
 2-46 to 2-47
 UnlockMemory function 3-16, 3-30 to 3-31
 updating windows, saving memory space for 1-44
 USP. *See* stack pointer, user

V

VBL tasks
 and the A5 register 4-16
 deferred under virtual memory 3-12
 Vector Base Register (VBR) 3-22
 virtual memory
 and AppleShare volumes 3-5
 and removable disks 3-5
 and user interrupts 3-21
 backing-store file 4-5
 bus-error vectors under 3-22
 CPU data caching 3-15
 debugger routines 3-34 to 3-40
 debugger support for 3-21 to 3-24
 deferring interrupt code execution 3-12, 3-20
 introduced 1-15
 management routines 3-25 to 3-33
 mapping information, getting 3-16 to 3-18
 requirements for running 3-5
 setting with the Memory control panel 4-5
 testing for availability 3-14
 Virtual Memory Manager 3-3 to 3-45. *See also* virtual
 memory
 data structures 3-24 to 3-25
 defined 3-3 to 3-4
 routines 3-25 to 3-40

W, X, Y

WaitNextEvent function, and temporary
 memory 2-10
 window definition procedures, and the A5
 register 4-15
 WITH statement (Pascal), and dangling pointers 1-29
 word-break routines, and the A5 register 4-15
 write-through cache 4-11

Z

zero (memory location) *See* 0 (memory location)
 zero-length handles *See* 0-length handles
 Zone data structure 2-20

I N D E X

zone headers 2-5, 2-20 to 2-21
zone pointers 2-20
zone records 2-20, 2-20 to 2-21
zone trailer blocks 2-20
zone trailers 2-5