Index

A

ABORT$
  bit in I.STATE, 2–61
ABPND$
  bit in I.STATE, 2–61
Access control field, 3–91
ACF
  See Access control field
ACTIV$
  offset in channel status word, 2–13
Active page field, 3–92
Active page registers (APR)
  at bootstrap, 3–2
Active Page Registers (APR)
  and virtual addresses, correspondence, 3–44
  controlled by .CMAP request, 3–55
  control mapping, 3–29
  directly modifying (with caution), 3–83
  Supervisor mode controlled by .MSDS request, 3–56
Active page registers (APRs)
  defined, 3–1
  discussion, 3–87
  format, 3–90
  include PAR and PDR, 3–87
Addresses
  converting 16-bit To 18- or 22-bit, 3–92
Addressing
  18- and 22-bit, 3–84
Address space separation, 3–15
  completely virtual job, 3–18
  controlled by .CMAP request, 3–55
  effects from directly modifying APRs, 3–83
  writing a program for, 3–16
ALL64$
  with .SETTOP, 3–77
  with IOPAG$ and completely virtual jobs, 3–7
APF
  See Active page field
APR
APR (Cont.)
  See Active page registers
AS.CAR
  bit in AST word, 4–25
AS.CTC
  bit in AST word, 4–25
AS.HNG
  bit in AST word, 4–25
AS.INP
  bit in AST word, 4–25
AS.OUT
  bit in AST word, 4–25
AST Word
  See Asynchronous terminal status word
Asynchronous terminal status word
  defined, 4–2
  description, 4–25

B

Background job
  completely virtual, 3–18
  differences from foreground job, 1–23
  loading, 1–12
    RUN command, 1–12
BATCH$
  bit in $CNFG1, 2–47
BATRN$
  bit in I.STATE, 2–61
18-bit addressing
  discussion, 3–84
22-bit addressing
  discussion, 3–84
Bitmap
  for low memory protection, 2–48
BKSP$
  offset in terminal config word T.CNFG, 4–22
  SET options bit, 2–8
$BLKEY
  RMON fixed offset 256, 2–42
  used by USR, 1–31
Blocking conditions

Blocking conditions (Cont.)
defined, 2–24
discussion, 2–31
how the monitor blocks a job, 2–32
how the monitor unblocks a job, 2–35
list of bits in I.BLOK, 2–31
BPT instruction
with mapped systems, 3–96
BUS$
bit in $CNFG2, 2–51
using, 2–52

C

C.COMP
offset in timer queue element, 2–10
C.CSW
offset in I/O channel descriptor, 2–13
C.DEVQ
offset in I/O channel descriptor, 2–13
C.HOT
offset in timer queue element, 2–10
C.JNUM
offset in timer queue element, 2–10
C.LENG
offset in I/O channel descriptor, 2–13
C.LINK
offset in timer queue element, 2–10
C.LOT
offset in timer queue element, 2–10
C.SBLK
offset in I/O channel descriptor, 2–13
C.SEQ
offset in timer queue element, 2–10
C.SIZ
offset in I/O channel descriptor, 2–13
C.SYS
offset in timer queue element, 2–10
C.UNIT
offset in I/O channel descriptor, 2–13
C.USED
offset in I/O channel descriptor, 2–13
CACHE$
bit in $CNFG2, 2–51
CC$GBL
bit in $INDSTA, 2–54
CC$IND
bit in $INDSTA, 2–54
CCB
with background job loading, 1–14
CCL
adding new commands, 1–37
getting unaltered command, 1–40
CCL.ON
offset in CLI.FL, 2–57
CCL.RN
offset in CLI.TY, 2–57
.CDFN programmed request
when using in a mapped system, 3–95
CF3.1S
bit in $CNFG3, 2–56
with console terminal simulation, B–10
CF3.64
bit in $CNFG3, 2–56
CF3.AS
bit in $CNFG3, 2–56
CF3.AT
bit in $CNFG3, 2–56
CF3.DM
bit in $CNFG3, 2–56
CF3.HI
bit in $CNFG3, 2–56
CF3.IM
bit in $CNFG3, 2–56
CF3.OW
bit in $CNFG3, 2–56
CF3.SI
bit in $CNFG3, 2–56
CF3.UA
bit in $CNFG3, 2–56
CF3.UB
bit in $CNFG3, 2–56
CF3.UI
bit in $CNFG3, 2–56
CF3.US
bit in $CNFG3, 2–56
CF3.VB
bit in $CNFG3, 2–56
completely virtual environment, 3–15
.CGMAP programmed request
and I.CMAP, 3–55
with Supervisor and User APRs, 3–83
CHAIN$
bit in $JSW, 1–9
with background job loading, 1–14
Chain area
passing unaltered CCL command, 1–40
Chain bit, 1–9
Chaining

Chaining (Cont.)
completely virtual environment, 3–17
.CHAIN programmed request
with background job loading, 1–15
Channel status word
See CSW
Character case
handling by KMON, 1–40
.CHCOPY programmed request
applicable to system jobs, 2–39
$CHKEY
RMON fixed offset 260, 2–42
used by USR, 1–31
CHNIF$
bit in $JSW, 1–8
CHNWT$
bit in I.BLOK, 2–32, 2–62
CHRLN$
offset in second terminal config word T.CNF2, 4–22
CIS$
bit in $CNFG2, 2–51
CLI.FL
intro to and bit definitions, 2–57
CLI.TY
intro and bit definitions, 2–57
CLK50$
bit in $CNFG1, 2–47
Clock
support for, 2–9
CLOCK$
bit in $CNFG1, 2–48
CM.JAS
bit mask in I.CMAP, 2–63
CM.NOS
bit mask in I.CMAP, 2–63
CM.PR0
bit mask in I.CMAP, 2–63
CM.PR1
bit mask in I.CMAP, 2–63
CM.PR2
bit mask in I.CMAP, 2–63
CM.PR3
bit mask in I.CMAP, 2–63
CM.PR4
bit mask in I.CMAP, 2–63
CM.PR5
bit mask in I.CMAP, 2–63
CM.PR6
bit mask in I.CMAP, 2–63
CM.PR7
bit mask in I.CMAP, 2–63
CM.S
bit mask in I.CMAP, 2–63
CM.SID
bit mask in I.CMAP, 2–63
CM.SII
bit mask in I.CMAP, 2–63
CM.SUP
bit mask in I.CMAP, 2–63
CM.U
bit mask in I.CMAP, 2–63
CM.UID
bit mask in I.CMAP, 2–63
CM.UII
bit mask in I.CMAP, 2–63
.CMAP programmed request
aligns WCBs and APRs, 3–51
described, 3–54
relationship to I.CMAP, 3–54
with context switching, 3–34
with MCA region, 3–50, 3–54
with Supervisor and User APRs, 3–83
CMPLT$
bit in I.STATE, 2–62
$CNFG1
intro and bit definitions, 2–47
RMON fixed offset 300, 2–43
$CNFG2
intro to and bit definitions, 2–50
RMON fixed offset 370, 2–44
$CNFG3
into and bit definitions, 2–56
RMON fixed offset 466, 2–46
with console terminal simulation, B–8
.CNTXSW programmed request
data preserved in MCA region, 2–30
when using in a mapped system, 3–95
with context switching, 3–34
$CNTXT
RMON fixed offset 320, 2–43
Commands
getting CCL parameters, 1–40
getting unaltered CCL, 1–40
permanently inhibiting processors, 1–39
Completely virtual background job, 3–19
address separation and overlays, 3–22
address space separation, 3–20
extended memory overlays, 3–21

Completely virtual environment
automatic job running in, 3–15
stopping, 3–15
chaining in, 3–17
defined, 3–1
overlays in, 3–16
SET RUN VBGEXE, 3–15
VBGEX$ bit in $JSX word, 3–15
when address space separation, 3–16
Completely virtual foreground/system job, 3–24
address separation and overlays, 3–27
address space separation, 3–25
extended memory overlays, 3–26
Completely virtual jobs
See also Completely virtual environment
mapping characteristics, 3–6
static window with, 3–33
V and VRUN with, 3–14
Completion queue, 2–18
Completion queue element
format, 2–20
Completion routines
implications of a blocked main program, 2–36
Concise command language
See CCL
CONSL$
offset in terminal status word T.STAT, 4–24
CONSOL
using, 4–40
Console
See also Terminals
background or command, 4–4
boot-time, 4–4
definition of, 4–4
hardware, 4–4
private, 4–5
shared, 4–5
software simulation of, B–1
appear like DL11, B–3
input restoration, B–4
input setup, B–2
input simulation, B–3
interrupt source, B–3
multiterminal restoration, B–11
multiterminal setup, B–8
multiterminal simulation, B–10
Console
software simulation of (Cont.)
multiterminal system with, B–7
output restoration, B–7
output setup, B–4
output simulation, B–6
special characteristics, 4–30
switching, 4–40
Context switching, 3–33
defined, 2–24
discussion, 2–30
information saved, 2–30
MCA region with, 3–33
what information is saved, 3–33
Core control block
See CCB
CORPTR, 3–51
CORPTX, 3–51
CPEND$
bit in I.STATE, 2–61
.CRAW programmed request
aligns WCBs and APRs, 3–51
and window definition blocks, 3–48
described, 3–58
with global regions, procedure, 3–99
CRLF$
offset in terminal config word T.CNFG, 4–19
SET options bit, 2–8
.CRRG programmed request
described, 3–56
with global regions, procedure, 3–98
with Supervisor and User APRs, 3–83
CSCCA$
bit in $CNFG2, 2–51
CSIRN$
bit in I.STATE, 2–61
CSW
bits in, 2–13
$CSW
RMON fixed offset 4, 2–42
CTRL/B
discussion, 2–7
CTRL/C
discussion, 2–6
sets bit in AST word, 4–25
CTRL/C and no .SCCA
with .MTIN request, 4–28
CTRL/F
discussion, 2–7

CTRL/O
discussion, 2–7
with .MTRCTO request, 4–29
CTRL/Q
discussion, 2–7
CTRL/S
discussion, 2–7
CTRL/X
discussion, 2–7
CTRLC$
offset in terminal status word T.STAT, 4–24
CTRLU$
offset in terminal status word T.STAT, 4–24
Current queue element pointer, 2–15

D

Data structures
defining with SYSTEM.MLB, A–1
$DATE
RMON fixed offset 262, 2–42
DBLK$M
offset in channel status word, 2–13
DCL.ON
offset in CLI.FL, 2–57
DCL.RN
offset in CLI.TY, 2–57
ddCQE
described, 2–15
ddLQE
described, 2–15
Device handler block number table, 2–65
Device handler entry point table, 2–65
Device handlers
adding a device to tables, 2–67
adding to queue of I/O requests, 2–16
advantages of using, 5–6
block number table, 2–65
entry point table, 2–65
extended device-unit physical name table, 2–67
instead of in-line interrupt service, 5–4
logical name table, 2–66
ownership table, 2–66
physical name table, 2–66
protocol with multiterminal handler hooks, 4–7
relationship to RMON, 2–23
Device handlers (Cont.)
size (blocks) of device, table, 2–65
size (bytes) of device, table, 2–66
size of, 1–41
status table, 2–65
Device ownership table, 2–66
.DEVICE programmed request
use in an interrupt service routine, 5–13
Device status table, 2–65
Device status word, 2–65
Device tables, 2–63
adding a device to, 2–67
using, 2–63
$DFLG
RMON fixed offset 264, 2–42
DH11$
offset in terminal status word T.STAT, 4–24
DH11 interface
discussion, 4–2
requires multiterminal feature support, 4–38
with .MTSET request, 4–28
DHOIP$
offset in terminal status word T.STAT, 4–24
Displacement field, 3–92
DL11 interface
discussion, 4–2
multiple terminals without MTTY support, 4–38
time-out polling routine for, 4–32
.DSTATUS programmed request
size of device handler, 1–41
$STAT table with, 2–65
DTACH$
offset in terminal status word T.STAT, 4–24
with .MTDTCH request, 4–30
DVINT$
monitor psect, 1–18
DVREC$
monitor psect, 1–18
$DVREC table, 2–65
DVSIZ$
monitor psect, 1–18
$DVSIZ table, 2–65
DWRIT$
offset in channel status word, 2–13
Dynamic component

Dynamic component (Cont.)
defined, 1–1
Dynamic windows, 3–33
with virtual jobs, 3–33
DZ11$
offset in terminal status word T.STAT, 4–24
DZ11 interface
discussion, 4–2
line polling routine, 4–32
requires multiterminal feature support, 4–38
with .MTSET request, 4–28

E

$E16LS
RMON fixed offset 316, 2–43
$$EDIT
offset in $PROGD, 2–55
EDIT$
bit in $JSW, 1–9
EIS$
bit in $CNFG2, 2–51
.ELAW programmed request
aligns WCBs and APRs, 3–51
described, 3–63
.ELRG programmed request
described, 3–62
with global regions, 3–56, 3–57, 3–62
with global regions, procedure, 3–99
with Supervisor and User APRs, 3–83
$ELTIM
RMON fixed offset 422, 2–45
$EMTRT
RMON fixed offset 400, 2–45
$ENSYS monitor routine
called by $SYSWT, 2–33
discussion, 2–28
.ENTER programmed request
with USR, 1–31
ENTRY$
monitor psect, 1–18
$ENTRY table, 2–65
EOF$
offset in channel status word, 2–13
ERLG$
bit in $SYSGE, 2–53
$ERRBY
in SYSCOM area, 1–6
$ERRBY (Cont.)
SYSCOM area symbol, 1–6
with .CRAW request, 3–58
with .CRRB request, 3–56
$ERRCN
RMON fixed offset 356, 2–44
$ERRLE
RMON fixed offset 376, 2–44
ERROR$
error severity symbol, 1–7
Error byte
See $ERRBY
Errors
severity levels
error, 1–7
fatal, 1–7
success, 1–7
unconditionally fatal, 1–7
warning, 1–7
EXIT$
bit in I.BLOK, 2–31, 2–62
.EXIT programmed request
with USR, 1–31
Expansion direction field, 3–92
Extended device-unit support
$PNAM2 table with, 2–67
Extended memory
See also Mapping
See also Memory management
debugging applications
DBG–11 is first choice, 3–98
defined, 3–1
hardware support for, 3–84
interrupt service routines, 5–20
large buffers/arrays in, 3–66
maximizing use of, 3–64
use virtual programs, 3–64
use XHANDL overlay pseudohandler, 3–64
VRUN command, 3–64
memory management faults, 3–97
pages, 3–85
TRAPS,BPT,IOT instructions, 3–96
Extended memory data structures
global region control block, 3–41
MCA region, 3–29
region control block, 3–40
region definition block, 3–37
window control block, 3–48

Extended memory data structures (Cont.)
window definition block, 3–42
window status word, 3–45
Extended memory overlays
completely virtual background job, 3–18
address space separation, 3–18
Extended memory programmed requests
bypassing, 3–83
introduction to, 3–35
usage, 3–35
$EXTIN
RMON fixed offset 416, 2–45

F

F.BADR
offset in fork block, 5–16
F.BLNK
offset in fork block, 5–16
F.BR4
offset in fork block, 5–16
F.BR5
offset in fork block, 5–16
$$F77
offset in $PROGF, 2–55
FATAL$
error severity symbol, 1–7
FBMON$
bit in $CNFG1, 2–47
FBTTY$
offset in terminal config word T.CNFG, 4–20
SET options bit, 2–8
FILL$
offset in terminal status word T.STAT, 4–24
Fill character
in SYSCOM area, 1–6
Fill count
in SYSCOM area, 1–6
Fixed offsets
See RMON fixed offsets
FJOB$
bit in $CNFG1, 2–47
$FKPTR
setting up its value, 5–16
FMPUR
size of impure area in RMON, 2–57
Foreground impure area
definition, 1–28
Foreground job
differences from background job, 1–23
illustrated general structure, 1–25
starting, 1–26
Foreground stack
description, 1–27
Foreground/system jobs
completely virtual environment, 3–18
$FORK
RMON fixed offset 402, 2–45
Fork block
contents, 5–16
.FORK macro
changing processor priority with, 5–4
registers available, 5–17
setting up $FKPTR, 5–16
summary, 5–17
use in an interrupt service routine, 5–15
with .INTEN macro, 5–16
with real-time applications, 5–17
FORM$
offset in terminal config word T.CNFG, 4–19
SET options bit, 2–8
$$FORT
offset in $PROGF, 2–55
FPU11$
bit in $SYSGE, 2–53
Free memory list, 3–54
FRUN keyboard command
description, 1–24
relating to system jobs, 2–40

G

.GCMAP programmed request
described, 3–55
Global region control block
described, 3–41
status word, 3–42
structure of, 3–42
Global regions, 3–30
attaching to, 3–57
automatic elimination of, 3–63
creating, 3–56
creating and mapping, 3–98
detaching from, 3–57
eliminating, 3–62
for handlers, 3–31
with .CRRG request, 3–56

.GMCX programmed request
described, 3–60
with Supervisor and User APRs, 3–83
GR.ADD
offset in global region control block, 3–42
GR.AGE
offset in global region control block status word, 3–42
GR.CAC
offset in global region control block status word, 3–42
GR.NAM
offset in global region control block, 3–42
GR.NRF
offset in global region control block status word, 3–42
GR.PRM
offset in global region control block status word, 3–42
GR.PRV
offset in GR.SHC, 3–42
with global regions, 3–56, 3–63
GR.SHC
offset in global region control block, 3–42
with global regions, 3–56, 3–57, 3–63
GR.SIZ
offset in global region control block, 3–42
GR.STA
offset in global region control block, 3–42
structure of, 3–42
.GTJB programmed request
applicable to system jobs, 2–39
GTLIN$
bit in $JSW, 1–10
.GTLIN bit, nonterminating, 1–10
.GTLIN programmed request
with .MTRCTO request, 4–29
GTLIN system subroutine
getting unaltered command, 1–40

H

H.SIZ
size of device handler, 1–41
$H2UB
RMON fixed offset 460, 2–46
Handlers
See Device handlers
HANMC$
HANMC$ (Cont.)
offset in terminal status word T.STAT, 4–24
with MTYCTL routine, 4–9
HANMT$
offset in terminal status word T.STAT, 4–24
with multiterminal handler hooks, 4–7
HDERR$
offset in channel status word, 2–13
High address limit
in SYSCOM area, 1–5
High-speed ring buffer
description, 2–5
HNGUP$
offset in terminal status word T.STAT, 4–24
HSIZE$
monitor psect, 1–18
$HSIZE table, 2–66
$HSUFF
RMON fixed offset 412, 2–45
HWFPU$
bit in $CNFG1, 2–47
HWTAB$
offset in terminal config word T.CNFG, 4–19
SET options bit, 2–8

I

I & D address separation
See Address space separation
I.BITM
impure area relative offset, 2–59
I.BLOK
checked for blocking conditions, 2–31
impure area offset 36, 2–59
job blocking word contents, 2–62
I.CHWT
impure area offset 10, 2–58
I.CLUN
impure area relative offset, 2–59
I.CMAP
and .GCMAP request, 3–55
completely virtual job default context, 3–7
impure area relative offset, 2–60
intro and bit definitions, 2–62
privileged job default context, 3–5

I.CMAP (Cont.)
relationship to .CMAP request, 3–54
virtual job default context, 3–6
with context switching, 3–34
I.CMPE
impure area offset 4, 2–58
pointer to end of completion queue, 2–18
I.CMPL
impure area offset 6, 2–58
pointer to list of completion queue elements, 2–18
I.CNSL
impure area offset 16, 2–58
with console terminal simulation, B–8
I.CNUM
impure area offset 26, 2–58
I.CSW
impure area offset 30, 2–58
I.CTLO
impure area relative offset, 2–59
I.DEVL
impure area relative offset, 2–60
I.ECTR
impure area relative offset, 2–61
I.FPP
impure area relative offset, 2–59
I.FPSA
impure area relative offset, 2–60
I.FSAV
impure area relative offset, 2–60
I.ICTR
impure area relative offset, 2–59
I.IGET
impure area relative offset, 2–59
I.IOCT
impure area offset 32, 2–58
I.IPUT
impure area relative offset, 2–59
I.IRNG
impure area relative offset, 2–59
I.ITOP
impure area relative offset, 2–59
I.JID
impure area relative offset, 2–59
I.JNUM
impure area offset 24, 2–58
I.LNAM
impure area relative offset, 2–59
stores system job name, 2–37
I.MPB
I.MPB (Cont.)
impure area relative offset, 2–60
I.MPTR
impure area relative offset, 2–60
I.MSG
impure area relative offset, 2–60
I.NAME
impure area relative offset, 2–59
I.OCTR
impure area relative offset, 2–59
I.OGET
impure area relative offset, 2–60
I.OPUT
impure area relative offset, 2–59
I.OTOP
impure area relative offset, 2–60
I.PCHW
impure area offset 12, 2–58
I.PERR
impure area offset 14, 2–58
I.PLSP
impure area relative offset, 2–60
I.PTTI
impure area offset 20, 2–58
I.QHDR
impure area offset 2, 2–58
list of free I/O queue elements, 2–15
I.QUE
impure area relative offset, 2–60
I.RSAV
impure area relative offset, 2–60
I.SCCA
impure area relative offset, 2–60
I.SCCI
impure area relative offset, 2–60
I.SCHP
impure area relative offset, 2–61
I.SCOM
impure area relative offset, 2–60
I.SCTR
impure area offset 34, 2–59
I.SERR
impure area relative offset, 2–60
I.SP
impure area relative offset, 2–59
I.SPC1
impure area relative offset, 2–61
I.SPCP
impure area relative offset, 2–61
I.SPLS

I.SPLS (Cont.)
impure area relative offset, 2–59
I.SPSV
impure area relative offset, 2–59
I.SSP
impure area relative offset, 2–60
I.STATE
checked by context switch, 2–31
impure area offset 0, 2–58
job state word
contents, 2–61
I.SWAP
impure area relative offset, 2–59
I.SYCH
impure area relative offset, 2–61
I.TERM
impure area relative offset, 2–60
I.TID
impure area offset 22, 2–58
I.TRAP
impure area relative offset, 2–59
I.TRM2
impure area relative offset, 2–60
I.TTLC
impure area offset 16, 2–58
impure area relative offset, 2–59
I.VHI
impure area relative offset, 2–61
I.VSTP
impure area relative offset, 2–61
I/O
Channel bits, 2–13
discussion of queued I/O, 2–11
using interrupts, 5–2
without using interrupts, 5–1
writing a routine, 5–6
I/O page
available to virtual programs, 3–64
described, 1–19
I/O processing
sequence of events, 2–21
I/O queue
operation, 2–13
I/O queue element
format, 2–14
I/O transfers
completing, 2–23
performing, 2–23
IFABT$
bit in $STATW, 2–50
IFACT$
bit in $STATW, 2–50
IFBEX$
bit in $STATW, 2–50
IFCHA$
bit in $STATW, 2–50
IFCTC$
bit in $STATW, 2–50
IFDAT$
bit in $STATW, 2–50
IFDOL$
bit in $STATW, 2–50
IFEKO$
bit in $STATW, 2–50
IFEOF$
bit in $STATW, 2–50
IFGTC$
bit in $STATW, 2–50
IFIND$
bit in $STATW, 2–50
IFINP$
bit in $STATW, 2–50
$IFMXN
RMON fixed offset 377, 2–45
IFRVT$
bit in $STATW, 2–50
IFSPC$
bit in $STATW, 2–50
IMP.SZ
length in bytes of impure area, 2–61
$IMPLO
discussion of, 2–55
RMON fixed offset 446, 2–46
with console terminal simulation, B–7
$IMPUR
pointer to impure area, 2–57
Impure area
contents, 2–58
defined, 2–24
determining size of (IMP.SZ), 2–61
discussion, 2–57
IN$IND
bit in $INDSTA, 2–55
IN$RUN
bit in $INDSTA, 2–54
$INDDV
RMON fixed offset 426, 2–45
$INDST
RMON fixed offset 417, 2–45
$INDSTA

$INDSTA (Cont.)
intro and bit definitions, 2–54
INDX$M
offset in channel status word, 2–13
INEXP$
offset in terminal status word T.STAT, 4–24
Input ring buffer
operation, 2–4
INTACT
used by $RQTSW monitor routine, 2–35
.INTEN macro
changing processor priority with, 5–4
discussion, 2–29
registers available, 5–17
summary, 5–17
use in an interrupt service routine, 5–13
with .SYNCH macro, 5–14
$INTEN monitor routine
discussion, 2–26, 5–13
Interrupt level counter
See INTLVL
Interrupt priority
discussion, 5–3
lowering with .INTEN, 5–13
Interrupts
described, 5–3
Interrupt service
for terminals, 4–30
Interrupt service routines, 5–1
advantages of in-line, 5–5
exiting, 5–17
in-line instead of device handlers, 5–4
in mapped systems, 5–20
registers available, 5–17
restricted in PAR1, 5–21
restrictions in PAR2, 5–22
skeleton outline, 5–17
structure, 5–11
use of .DEVICE, 5–13
use of .FORK, 5–16
setting up $FKPTR, 5–16
use of .INTEN, 5–13
use of .PROTECT, 5–11
use of .SYNCH, 5–14
writing a routine, 5–6
Interrupt vectors
list, 1–10
setting up the values, 5–11
INTLVL
INTLVL (Cont.)
interrupt level counter, 2–26
values, 2–26
IOPAG$
for virtual programs, 3–64
with .SETTOP, 3–77
with ALL64$ and completely virtual jobs, 3–7
IOPAGE global region, 3–64
IOT instruction
with mapped systems, 3–96

J

Job chaining
See Chaining
$JOBNU
RMON fixed offset 322, 2–43
Job numbers, 2–37
Job priority, 2–37
Jobs
automatic completely virtual loading of, 3–15
completely virtual foreground/system, 3–18
$JOBS
intro and bit definitions, 2–55
RMON fixed offset 455, 2–46
Job status word
See $JSW
JSW
See $JSW
$JSW, 1–8
in SYSCOM area, 1–5
issue .MTRCTO or .RCTRLO after changing, 4–29
relationship with TCB, 4–30
SYSCOM area symbol, 1–5
VIRT$ restriction, 3–11
VIRT$ value for completely virtual jobs, 3–7
VIRT$ value for privileged jobs, 3–5
VIRT$ value for virtual jobs, 3–6
VRUNV$ bit explained, 3–14
with .MTIN request, 4–28
with .MTOUT request, 4–28
with .MTRCTO request, 4–29
$JSX
with .SETTOP, 3–77
with completely virtual job mapping, 3–7

K

$$K52
  offset in $PROGD, 2–55
$$KED
  offset in $PROGD, 2–55
KERNEL global region, 3–64
Kernel mapping
  at bootstrap, 3–2
Kernel mode, 3–23
  applies to .SYNCH, 5–22
  discussed, 3–23
$$KEX
  offset in $PROGD, 2–55
Keyboard commands
  expanded by KMON, 1–36
Keyboard monitor
  See KMON
KMON, 1–36
  communication with UCF, 1–39
  size of, 1–41
KMON commands
  See DCL and Keyboard commands
$KMONI
  RMON fixed offset 450, 2–46
KMON/UCF/UCL interface, 2–57
KSPND$
  bit in I.BLOK, 2–31, 2–62
KT11$
  bit in $CNFG1, 2–48
KW11P$
  bit in $CNFG1, 2–48
KXCPU$
  bit in $CNFG2, 2–51

L

$LAST$
  monitor psect, 1–18
Last queue element pointer, 2–15
LDREL$
  bit in $CNFG2, 2–51
LIGHT$
  bit in $CNFG2, 2–51
.LIMIT directive
  with .SETTOP (non-XM), 3–71
LINK keyboard command
  enables XM .SETTOP, 3–68
LINSP$
  offset in terminal config word T.CNFG, 4–21
LKCS$
  bit in $CNFG1, 2–48
LN$IND
  bit in $INDSTA, 2–54
LOAD$
  bit in I.STATE, 2–61
  with context switching, 3–34
LOAD keyboard command
  relating to system jobs, 2–40
Local lines, 4–3
Local regions, 3–30
  with .CRRG request, 3–56
.LOCK programmed request
  with USR, 1–34
Logical device name table, 2–66
Logical unit number
  defined, 4–2
  of a terminal, 4–3
.LOOKUP programmed request
  with USR, 1–31
Lowercase bit, 1–8
$LOWMA
  introduction to and bitmap, 2–48
  RMON fixed offset 326, 2–44
  when loading background job, 1–13
Low memory
  defined, 3–1
Low memory bitmap
  See Bitmap
LS handler
  can support multiterminal hooks, 4–6
  connecting as terminal in multiterminal system, 4–11
LSI11$
  bit in $CNFG1, 2–48
LUN
  See Logical unit number

M

M.APR
  offset in MCA region, 3–52
M.FCNT
  with .MTGET request, 4–27
  with .MTSET request, 4–28
M.MMR3
  offset in MCA region, 3–53
M.PASD
  offset in MCA region, 3–53
M.PASI

M.PASI (Cont.)
offset in MCA region, 3–53
M.PAUD
offset in MCA region, 3–53
M.PAUI
offset in MCA region, 3–52
M.PDSD
offset in MCA region, 3–53
M.PDSI
offset in MCA region, 3–53
M.PDUD
offset in MCA region, 3–52
M.PDUI
offset in MCA region, 3–52
M.RGN
offset in MCA region, 3–52
M.TFIL
with .MTGET request, 4–27
with .MTSET request, 4–28
M.TST2
with .MTGET request, 4–27
with .MTSET request, 4–28
M.TSTS
with .MTGET request, 4–27
with .MTSET request, 4–27
M.TSTW
with .MTGET request, 4–27
M.TWID
with .MTGET request, 4–27
with .MTSET request, 4–28
M.VHI, 3–52
M.WCB
offset in MCA region, 3–53
Mapped monitors
interrupt service routines, 5–20
Mapping
See also Extended memory
See also Memory management
between processor modes, 3–29
completely virtual background job, 3–18
address separation and overlays, 3–22
address space separation, 3–20
extended memory overlays, 3–21
completely virtual foreground/system job, 3–18
address separation and overlays, 3–27
address space separation, 3–25
extended memory overlays, 3–26
completely virtual job, 3–6, 3–13
Mapping (Cont.)
context switching, 3–33
controlled by programs, 3–29
default MCA region context, 3–55
defined, 3–2
for interrupt service routines, 5–20
information stored in MCA region, 3–50
privileged background job, illustrated, 3–9
privileged foreground/system jobs, illustrated, 3–10
privileged job environment, 3–7
privileged jobs, 3–5
selecting virtual, 3–11
support controlled by .CMAP request, 3–55
virtual address windows, 3–31
virtual background job, 3–11
virtual jobs, 3–6
virtual to physical, 3–29
Mapping context area
See MCA region
.MAP programmed request
aligns WCBs and APRs, 3–51
described, 3–59
with .CRAW request, 3–59
with global regions, procedure, 3–99
with Supervisor and User APRs, 3–83
$MAXBLK
RMON fixed offset 314, 2–43
MCA region
APR considerations, 3–51
.CMAP request with, 3–54
intro to, 2–62
structure of, 3–50
virtual address windows, 3–32
what is saved and preserved in, 2–30
with context switching, 3–34
with physical address regions, 3–29
with Supervisor mode windows, 3–51
Memory
use of extended memory, 3–1
Memory management
See also Extended memory
See also Mapping
relocation, 3–86
Memory management faults
discussion, 3–97
Memory management unit
status registers, 3–94
Memory mapping status registers, 3–29

Memory parity errors
discussion, 3–97
$MEMPT
RMON fixed offset 430, 2–45
with MCA region, 3–51
$MEMSZ
RMON fixed offset 420, 2–45
Message handler
See MQ handler
$MFPS
RMON fixed offset 362, 2–44
MMGT$
bit in $SYSGE, 2–53
MMR3 status register
used by memory management unit, 3–94
with context switching, 3–34
$MONAM
RMON fixed offset 406, 2–45
Monitor commands
relating to system jobs, 2–40
MPTY$
bit in $CNFG2, 2–51
MPTYS$
bit in $SYSGE, 2–53
MQH$P2
creates PAR2 restriction, 3–95
restricts interrupt service routines, 5–22
SYSGEN conditional for special MQ handler, 2–40
MQ handler
for inter-job messages, 2–39
MQH$P2 conditional, 2–39, 5–22
restricted in PAR2 under XM, 5–22
.MSDS programmed request
aligns WCBs and APRs, 3–51
described, 3–56
with Supervisor and User APRs, 3–83
.MTATCH programmed request
described, 4–26
.MTDTCH programmed request
described, 4–30
with .MTATCH, 4–26
MTEMT$
monitor psect, 1–18
.MTGET programmed request
described, 4–27
required before .MTSET, 4–28
.MTIN programmed request
described, 4–28
MTINT$
monitor psect, 1–18
MTOENB routine
multiterminal handler hooks, 4–8
.MTOUT programmed request
described, 4–28
.MTPRNT programmed request
described, 4–29
$MTPS
RMON fixed offset 360, 2–44
.MTRCTO programmed request
described, 4–29
issue after changing $JSW, 4–29
.MTSET programmed request
described, 4–27
requires previous .MTGET, 4–28
.MTSTAT programmed request
described, 4–29
MTTEMT.MAC
discussion, 4–1
MTTINT.MAC
discussion, 4–1
MTTY$
bit in $SYSGE, 2–53
monitor psect, 1–18
MTY$HK
with multiterminal handler hooks, 4–7
MTYBRK routine
multiterminal handler hooks, 4–9
MTYCTL routine
multiterminal handler hooks, 4–9
set HANMC$ before calling, 4–9
MTYSTA routine
multiterminal handler hooks, 4–10
Multiplexer
applying to DH11, 4–2
applying to DZ11, 4–2
Multiterminal feature
See also Multiterminal handler hooks
data structures, 4–14
terminal control block, 4–14
debugging application programs, 4–33
description of programmed requests, 4–26
DL11 time-out polling routine, 4–32
DZ11 line polling routine, 4–32
example program, 4–33
hardcopy boot-time console as printer, 4–11
interrupt service, 4–30
multiple terminals with, 4–12

Multiterminal feature (Cont.)
multiple terminals without, 4–38
programmed requests summary, 4–13
terminals
assuring output enable, 4–29
attaching to, 4–26
detaching from, 4–30
getting information about, 4–29
obtaining status of, 4–27
printing characters on, 4–28
printing character strings on, 4–29
receiving characters from, 4–28
setting characteristics for, 4–27
Multiterminal handler hooks, 4–5
MTOENB routine, 4–8
protocol with device handlers, 4–7
terminal line break routine, 4–9
terminal modem control routine, 4–9
terminal output enable routine, 4–8
terminal status routine, 4–10
THOOKS data structure, 4–8
Multiterminal support, 4–1
hardware, 4–2

N

Next free address, 3–69
NORUN$
bit in I.BLOK, 2–32, 2–62
NOVBG$
completely virtual chaining, 3–17
NUM.RG, 3–52
NUM.WN, 3–52

O

ODDPR$
offset in second terminal config word T.CNF2, 4–23
ODT
use VDT in extended memory, 3–98
use VDT to debug multiterminal applications, 4–33
OFS.AP, 3–52
OFS.MC, 3–51
OFS.RG, 3–52
OFS.WN, 3–52
Output ring buffer
operation, 2–2
Overlay bit, 1–9
Overlay regions
completely virtual background job, 3–18
completely virtual environment, 3–16
Overlay segments
completely virtual environment, 3–16
OVLY$
bit in $JSW, 1–9
OVLYnn
monitor psect, 1–18
OWNER$
monitor psect, 1–18
$OWNER table, 2–66

P

P1$EXT
RMON fixed offset 432, 2–46
Page
defined, 3–1
PAGE$
offset in terminal config word T.CNFG, 4–20
offset in terminal status word T.STAT, 4–24
SET options bit, 2–8
Page address registers (PARs)
defined, 3–2
discussion, 3–88
format, 3–91
Page Address Registers (PARs)
with context switching, 3–34
Page descriptor registers (PDRs)
discussion, 3–89
format, 3–91
Page Descriptor Registers (PDRs)
with context switching, 3–34
Page length field, 3–92
Pages
correspondence between pages and APRs, 3–91
in memory management unit definition, 3–85
PAR
See Page address registers
PAR1
restricted for interrupt service routines, 5–21
restrictions for using, 3–94
with MQ handler, 2–40
PAR2

PAR2 (Cont.)
restrictions for interrupt service routines, 5–22
restrictions for using, 3–95
with MQ handler (MQH$P2=1), 2–40
PAREN$
offset in second terminal config word T.CNF2, 4–23
Pass line to KMON bit, 1–8
PATCH$
monitor psect, 1–18
PDP60$
bit in $CNFG2, 2–51
PDP70$
bit in $CNFG2, 2–51
PDR
See Page descriptor registers
Permanent name table, 2–64
Physical address
defined, 3–2
discussion, 3–84
Physical address regions, 3–29
Physical device name table, 2–66
Physical unit number
of a terminal, 4–3
PLAS
defined, 3–33
PNAM2
monitor psect, 1–18
$PNAM2 table, 2–67
as relates to $PNAME, 2–67
PNAME$
monitor psect, 1–18
$PNAME table, 2–64
as relates to $PNAM2, 2–67
$PNPTR
RMON fixed offset 404, 2–45
.PRINT programmed request
with ring buffers, 2–2
Priority
device and processor discussion, 5–3
lowering with .INTEN, 5–13
of jobs, 2–37
Privileged jobs
background mapping, illustrated, 3–9
foreground/system mapping, illustrated, 3–10
mapping characteristics, 3–5
mapping environment, 3–7
Privileged jobs (Cont.)
static window restricted, 3–32
windows with, 3–8
compared to virtual, 3–8
Processor modes, 3–23
controlled by .CMAP request, 3–55
Processors
determining bus structure of, 2–52
Processor status word
See PSW
$PROGD
intro and bit definitions, 2–55
RMON fixed offset 452, 2–46
$PROGF
intro and bit definitions, 2–55
RMON fixed offset 453, 2–46
Program's logical address space
See PLAS
Program high limit, 3–69
Programmed requests
defining EMT request block symbols, A–2
extended memory, 3–35
summary of, 3–35
for multiterminal support, 4–26
multiterminal summary, 4–13
PROS$
bit in $CNFG2, 2–51
using, 2–52
.PROTECT programmed request
use in an interrupt service routine, 5–11
use of bitmap, 2–49
PS
See PSW
PSCLKH
pseudoclock, 2–10
PSCLOK
pseudoclock, 2–10
Pseudoclock, 2–10
PSW
description, 5–4
processor modes, 3–23

Q

Q.2UNI
offset in I/O queue element Q.BLKN, 2–14
Q.BLKN
offset in I/O queue element, 2–14
Q.BUFF
offset in I/O queue element, 2–14

Q.COMP
offset in I/O queue element, 2–14
Q.CSW
offset in I/O queue element, 2–14
Q.ELGH
offset in I/O queue element, 2–14
Q.FUNC
offset in I/O queue element Q.BLKN, 2–14
Q.JNUM
offset in I/O queue element Q.BLKN, 2–14
Q.LINK
offset in I/O queue element, 2–14
Q.MEM
offset in I/O queue element, 2–14
Q.PAR
offset in I/O queue element, 2–14
Q.TYPE
offset in I/O queue element Q.BLKN, 2–14
Q.UNIT
offset in I/O queue element Q.BLKN, 2–14
Q.WCNT
offset in I/O queue element, 2–14
QC.CMP
offset in completion queue element, 2–20
QC.CSW
offset in completion queue element, 2–20
QC.LINK
offset in completion queue element, 2–20
QC.OFT
offset in completion queue element, 2–20
$QCOMP
RMON fixed offset 270, 2–42
$QHOOK
RMON fixed offset 456, 2–46
QS.CMP
offset in .SYNCH block, 5–15
offset in synch queue element, 2–20
QS.ID
offset in .SYNCH block, 5–15
offset in synch queue element, 2–20
QS.JOB
offset in .SYNCH block, 5–15
offset in synch queue element, 2–20
QS.LNK
offset in .SYNCH block, 5–15
offset in synch queue element, 2–20
QS.SYN
offset in .SYNCH block, 5–15
offset in synch queue element, 2–20
.QSET programmed request
processed by USR, 1–29
when using in a mapped system, 3–95
QUEUE$
bit in $CNFG1, 2–48
Queued I/O
discussion, 2–11

R

R.BADD
offset in regions control block, 3–41
with .CRRB request, 3–56
R.BNWD
offset in regions control block, 3–41
with .CRRB request, 3–56
with .ELAW request, 3–63
with .MAP request, 3–60
with .UNMAP request, 3–61
R.BSIZ
offset in regions control block, 3–41
with .CRRB request, 3–56
R.BSTA
offset in regions control block, 3–41
with .CRRB request, 3–56
R.GBAS
offset in region definition block, 3–38
R.GID
offset in region definition block, 3–38
with .CRRB request, 3–56
R.GNAM
offset in region definition block, 3–38
R.GSIZ
offset in region definition block, 3–38
structure of, 3–38
with .CRRG request, 3–56
with global regions, 3–57
with global regions, procedure, 3–98
R.GSTS
offset in region definition block, 3–38
with global regions, 3–57
with global regions, procedure, 3–99
RCB
See Region control block
RDB
See Region definition block
.RDBBK macro
described, 3–40
with global regions, 3–57
with global regions, procedure, 3–98

.RDBDF macro
described, 3–39
invoked by .RDBBK, 3–40
Reenter bit, 1–8
Region control block (RCB)
described, 3–40
structure of, 3–41
with context switching, 3–34
with physical address regions, 3–30
Region definition block (RDB)
discussed, 3–37
structure of, 3–38
Region status word, 3–38
Registers available
after .FORK, 5–17
after .INTEN, 5–17
after .SYNCH, 5–17
after interrupt, 5–17
Relocation
by memory management unit definition, 3–86
defined, 3–2
REMOT$
offset in terminal config word T.CNFG, 4–22
with .MTSET request, 4–27
Remote lines, 4–3
RENAM$
offset in channel status word, 2–13
Resident monitor
See RMON
RESUME keyboard command
relating to system jobs, 2–41
Ring buffers
for terminal service, 2–1
high-speed, 2–5
operation, 2–3
RMNUSR
monitor psect, 1–18
RMON, 2–1
description, 1–17
monitor psect, 1–18
relationship to device handlers, 2–23
size of, 1–41
$RMON
RMON fixed offset 0, 2–42
RMON base address
in SYSCOM area, 1–6
RMON fixed offsets, 2–41
RMON fixed offsets (Cont.)
discussion, 2–41
values, 2–41
RONLY$
offset in channel status word, 2–13
RPALL$
offset in second terminal config word T.CNF2, 4–23
$RQTSW monitor routine
discussion, 2–35
to request a scheduling pass, 2–35
RS.AGE
offset in regions status word, 3–39
with global regions, 3–63
with global regions, procedure, 3–98
RS.BAS
offset in regions status word, 3–39
RS.CAC
offset in regions status word, 3–39
RS.CGR
offset in regions status word, 3–38
with global regions, 3–57
with global regions, procedure, 3–98
RS.CRR
offset in regions status word, 3–38
with .CRRB request, 3–56
with .ELRG request, 3–62
with global regions, 3–57
with global regions, procedure, 3–99
RS.DSP
offset in regions status word, 3–39
RS.EGR
eliminating global regions, 3–62
offset in regions status word, 3–39
RS.EXI
offset in regions status word, 3–39
RS.GBL
offset in regions status word, 3–38
with global regions, procedure, 3–98
RS.NAL
offset in regions status word, 3–38
RS.NEW
offset in regions status word, 3–38
with global regions, procedure, 3–99
RS.NSM
offset in regions status word, 3–39
RS.PVT
offset in regions status word, 3–39
RS.UNM
offset in regions status word, 3–38

RS.UNM (Cont.)
with .ELRG request, 3–62
RSTRT$
bit in $JSW, 1–8
RT11
monitor psect, 1–18
RTDATA
monitor psect, 1–18
RTEM$
bit in $SYSGE, 2–53
$RTSPC
RMON fixed offset 464, 2–46
RUN keyboard command
description, 1–12

S

Scheduler
discussion, 2–35
how it works, 2–36
Scheduling
defined, 2–24
Separated I and D addresses
See Address space separation
SET CLI condition bits, 2–57
SET keyboard commands
SET ERROR
effect, 1–7
SET TT options
status word bit definitions, 2–8
SET LS LINE=n, 4–11
SET RUN VBGEXE
completely virtual environment, 3–15
.SETTOP programmed request
aligns WCBs and APRs, 3–51
automatically enabling, 3–68
Non-XM type, 3–69
regular and XM type, 3–68
restricted with privileged foreground jobs, 3–73
with background jobs (summary), 3–80
with completely virtual jobs
setting a limit, 3–76
with completely virtual jobs (summary), 3–82
with foreground jobs (summary), 3–82
with privileged background jobs, 3–72
with privileged jobs, 3–72
with Supervisor and User APRs, 3–83
with virtual background jobs, 3–75
.SETTOP programmed request (Cont.)
with virtual foreground jobs, 3–75
with virtual jobs, 3–74
XM type described, 3–71
SET TT: CRLF bit, 2–8
SET TT: FB bit, 2–8
SET TT: FORM bit, 2–8
SET TT: PAGE bit, 2–8
SET TT: SCOPE bit, 2–8
SET TT: TAB bit, 2–8
SET TT CONSOL=n, 4–11
SET XL LINE=n, 4–12
SHARE$
offset in terminal status word T.STAT, 4–24
SHOW keyboard commands
SHOW JOBS
relating to system jobs, 2–41
SHOW MEMORY
to get size and base of RMON, 1–41
to get size of loaded handlers, 1–41
to get size of USR, 1–41
Simulated console terminal
see Console terminal, software simulation of
Single-line editor bit, disable, 1–9
SLEDI$
bit in $CNFG1, 2–47
SLKMO$
bit in $CNFG1, 2–47
$SLOT2
RMON fixed offset 502, 2–47
to find size of device tables, 2–63
SP$0
bit in $SPSTA word SP$NFL, 2–54
SP$ABT
bit in $SPSTA word SP$OPR, 2–54
SP$ACT
bit in $SPSTA, 2–53
SP$DEF
bit in $SPSTA word SP$NFL, 2–54
SP$ERR
bit in $SPSTA, 2–54
SP$EXI
bit in $SPSTA word SP$OPR, 2–54
SP$GTM
bit in $SPSTA, 2–54
SP$IEN
bit in $SPSTA, 2–54
SP$KIL

SP$KIL (Cont.)
bit in $SPSTA word SP$OPR, 2–54
SP$NFL
bit in $SPSTA, 2–53
SP$NXT
bit in $SPSTA word SP$OPR, 2–54
SP$OFF
bit in $SPSTA word SP$OPR, 2–54
SP$ON
bit in $SPSTA word SP$OPR, 2–54
SP$OPR
bit in $SPSTA, 2–53
SP$SCN
bit in $SPSTA, 2–53
SP$SHO
bit in $SPSTA, 2–53
SP$UNI
bit in $SPSTA, 2–53
Special chain exit bit, 1–9
Special mode terminal bit, 1–8
SPND$
bit in I.BLOK, 2–32, 2–62
$SPSIZ
RMON fixed offset 504, 2–47
$SPSTA
intro to and bit definitions, 2–53
RMON fixed offset 414, 2–45
SPUSR
RMON fixed offset 272, 2–42
SPXIT$
bit in $JSW, 1–9
SRUN keyboard command
description, 1–24
relating to system jobs, 2–40
STACK$
monitor psect, 1–18
Stack pointer
in SYSCOM area, 1–5
Starting address
in SYSCOM area, 1–5
STASK$
bit in $SYSGE, 2–53
STAT$
monitor psect, 1–18
Static component
defined, 1–1
Static window, 3–32
MCA region with, 3–51
restricted with privileged jobs, 3–32
with completely virtual jobs, 3–33
Static window (Cont.)
with virtual jobs, 3–33
$STAT table, 2–65
Status registers
defined, 3–2
in memory management unit, 3–94
$STATW
intro to and bit definitions, 2–50
RMON fixed offset 366, 2–44
SUCCS$
error severity symbol, 1–7
Supervisor mode, 3–23
discussed, 3–23
initial relationship to User mode, 3–55
SUSPEND keyboard command
interaction with scheduler, 2–36
relating to system jobs, 2–41
SWREG$
bit in $CNFG2, 2–51
$SYIND
RMON fixed offset 364, 2–44
$SYNCH
RMON fixed offset 324, 2–43
.SYNCH block
contents, 5–15
.SYNCH macro
does not use an I/O queue element, 2–20
executes with kernel mapping, 5–22
how errors returned by, 5–15
registers available, 5–17
summary, 5–17
use in an interrupt service routine, 5–14
uses completion queue, 2–20
with .INTEN macro, 5–14
with USR, 5–14
SYNCH queue element
format, 2–20
Synchronous system traps, 3–95
$SYPTR
SYSCOM area symbol, 1–6
$SYSCH
RMON fixed offset 244, 2–42
SYSCOM area, 1–2
$SYSGE
intro to and bit definitions, 2–52
RMON fixed offset 372, 2–44
with console terminal simulation, B–1
SYSMAC.SML
defining structures with SYSTEM.MLB, A–1

SYSTEM.MLB
listing of macros in, A–5
using, A–1
System communication area
See SYSCOM area
System device handler
described, 1–18
System jobs
applicable programmed requests, 2–39
design in writing, 2–38
equivalent to foreground job, 1–22
memory space considerations if writing, 2–38
name stored in I.LNAM, 2–37
RMON size from support for, 2–37
scheduling considerations if writing, 2–38
starting, 1–26
System state
conditions requiring, 2–25
in I/O processing, 2–21
switching asynchronously, 2–26
switching synchronously, 2–28
$SYSUPD
RMON fixed offset 277, 2–43
$SYSVE
RMON fixed offset 276, 2–43
$SYSWT monitor routine
discusssion of, 2–33
run after all completion routines, 2–36
to check job blocking, 2–33
$SYUNI
RMON fixed offset 274, 2–42

T

T.AST
two words in terminal control block, 4–19
with .MTATCH request, 4–26
with .MTDTCH request, 4–30
T.CNF2
offset in terminal control block, 4–17
structure of, 4–22
with .MTGET request, 4–27
with .MTSET request, 4–28
T.CNFG
offset in terminal control block, 4–16
structure of, 4–19
with .MTATCH request, 4–26
with .MTGET request, 4–27
with .MTIN request, 4–28
T.CNFG (Cont.)
with .MTOUT request, 4–28
with .MTPRNT request, 4–29
with .MTRCTO request, 4–29
with .MTSET request, 4–27
T.CNT
offset in terminal control block, 4–19
T.CSR
offset in terminal control block, 4–17
with console terminal simulation, B–8
T.FCNT
byte offset in terminal control block, 4–17
with .MTGET request, 4–27
with .MTSET request, 4–28
T.ICTR
offset in terminal control block, 4–18
T.IGET
offset in terminal control block, 4–18
T.IPUT
offset in terminal control block, 4–18
T.IRNG
offset in terminal control block, 4–18
T.ITOP
offset in terminal control block, 4–18
T.JOB
byte offset in terminal control block, 4–17
T.LPOS
byte offset in terminal control block, 4–17
T.NFIL
byte offset in terminal control block, 4–17
T.OCHR
byte offset in terminal control block, 4–17
T.OCTR
byte offset in terminal control block, 4–18
with .MTDTCH request, 4–30
T.OGET
offset in terminal control block, 4–18
T.OPUT
offset in terminal control block, 4–18
T.OTOP
offset in terminal control block, 4–18
T.OWNR
offset in terminal control block, 4–17
with .MTATCH request, 4–26
with .MTDTCH request, 4–30
with multiterminal handler hooks, 4–7
T.PRI
offset in terminal control block, 4–17
T.PTTI
byte offset in terminal control block, 4–17

T.PUN
byte offset in terminal control block, 4–17
T.RTRY
offset in terminal control block, 4–18
T.STAT
configuration of, 4–24
offset in terminal control block, 4–17
with .MTGET request, 4–27
with console terminal simulation, B–8
with multiterminal handler hooks, 4–7
T.TBLK
seven words in terminal control block, 4–18
T.TCTF
byte offset in terminal control block, 4–18
T.TFIL
byte offset in terminal control block, 4–17
with .MTATCH request, 4–26
with .MTGET request, 4–27
with .MTIN request, 4–28
with .MTOUT request, 4–28
with .MTRCTO request, 4–29
with .MTSET request, 4–28
T.TID
offset in terminal control block, 4–18
T.TNFL
byte offset in terminal control block, 4–18
T.TTLC
offset in terminal control block, 4–18
T.TWID
with .MTSET request, 4–28
T.VEC
offset in terminal control block, 4–17
T.WID
offset in terminal control block, 4–17
with .MTGET request, 4–27
T.XBUF
three words in terminal control block, 4–19
T.XCNT
byte offset in terminal control block, 4–19
T.XFLG
byte offset in terminal control block, 4–19
T.XRE
offset terminal control block, 4–19
TCB
See Terminal control block
TCBIT$
bit in $JSW, 1–9
offset in terminal config word T.CNFG, 4–20
TCBIT$ (Cont.)
with .MTATCH request, 4–26
with .MTIN request, 4–28
with .MTOUT request, 4–28
with .MTPRNT request, 4–29
with .MTRCTO request, 4–29
with .MTSET request, 4–27
$TCFIG
RMON fixed offset 424, 2–45
$$TECO
offset in $PROGD, 2–55
Terminal (second) configuration word (T.CNF2), 4–22
Terminal configuration word (T.CNFG), 4–19
Terminal control block (TCB)
defined, 4–1
description, 4–14
format, 4–14
getting information about, 4–29
getting information from, 4–27
patching, 4–24
relationship with $JSW, 4–30
Terminal I/O
control characters, 2–6
limitations, 2–6
sets bit in AST word when input available, 4–25
sets bit in AST word when output buffer empty, 4–25
Terminal line break (MTYBRK) routine, 4–9
Terminal modem control (MTYCTL) routine, 4–9
Terminal modem status (MTYSTA) routine, 4–10
Terminal output enable (MTOENB) routine, 4–8
Terminals
See also Console
connecting handlers to, 4–5
console
software simulation of, B–1
different types defined, 4–4
interrupt service
local, 4–31
remote, 4–31
switching the console, 4–40
using more than one, 4–5
video console and printer terminal, 4–38
Terminals, multiple without multiterminal feature, 4–38

Terminal service
input ring buffer, 2–3
in RMON, 2–1
output ring buffer, 2–1
remote terminal sets bits in AST word, 4–26
ring buffers, 2–1
Terminal status word (T.STAT), 4–24
Terminal wait bit, inhibit, 1–9
THK.BK
offset in multiterminal THOOKS, 4–8
THK.CT
offset in multiterminal THOOKS, 4–8
THK.LE
offset in multiterminal THOOKS, 4–8
THK.NU
offset in multiterminal THOOKS, 4–8
THK.OE
offset in multiterminal THOOKS, 4–8
THK.ST
offset in multiterminal THOOKS, 4–8
THK.SZ
offset in multiterminal THOOKS, 4–8
THK.TC
offset in multiterminal THOOKS, 4–8
$THKPT
RMON fixed offset 472, 2–47
with multiterminal handler hooks, 4–7
THOOKS
See Terminal hooks
structure of, 4–8
Time
maintained by system clock support, 2–10
$TIME
system time, 2–10
TIMER$
bit in $SYSGE, 2–53
Timer queue element
format, 2–10
Timer service, 2–9
TIMIT$
bit in $SYSGE, 2–53
Transparent spooler (SPOOL) status word
see $SPSTA
TRAP instruction
with mapped systems, 3–96
Trap vectors, 1–2
list of, 1–2
TRMTBL.MAC
TRMTBL.MAC (Cont.)
discussion, 4–1
TSXP$
bit in $SYSGE, 2–53
TTCL$
with .MTRCTO request, 4–29
$TTCNF
bit definitions, 2–8
SET TT status word, 2–8
$TTFIL
SYSCOM area symbol, 1–6
TTIUSR
used by terminal interrupt service, 2–7
TTIWT$
bit in I.BLOK, 2–32, 2–62
$TTKB
RMON fixed offset 306, 2–43
with console terminal simulation, B–1
$TTKS
RMON fixed offset 304, 2–43
with console terminal simulation, B–1
TTLC$
bit in $JSW, 1–8
offset in terminal config word T.CNFG, 4–22
with .MTATCH request, 4–26
with .MTIN request, 4–28
with .MTOUT request, 4–28
with .MTSET request, 4–27
TTNFI
SYSCOM area symbol, 1–6
TTOEM$
bit in I.BLOK, 2–32, 2–62
TTOENB routine
called by MTOENB, 4–8
TTOUSR
used by terminal interrupt service, 2–7
.TTOUTR programmed request
with ring buffers, 2–2
TTOWT$
bit in I.BLOK, 2–32, 2–62
$TTPB
RMON fixed offset 312, 2–43
with console terminal simulation, B–1
$TTPS
RMON fixed offset 310, 2–43
with console terminal simulation, B–1
TTSPC$
bit in $JSW, 1–8

TTSPC$ (Cont.)
offset in terminal config word T.CNFG, 4–22
with .MTIN request, 4–28
with .MTOUT request, 4–28
with .MTRCTO request, 4–29
with .MTSET request, 4–27
.TTYOUT programmed request
with ring buffers, 2–2
.TWAIT programmed request
with global regions, procedure, 3–99

U

UCF, 1–39
communication with KMON, 1–39
permanently enabling UCF, 1–39
processing order, 1–40
UCF.KM
offset in CLI.FL, 2–57
UCF data structure, 1–40
UCF.ON
offset in CLI.FL, 2–57
UCF data structure, 1–40
UCF.RN
offset in CLI.TY, 2–57
UCF/UCL/KMON interface, 2–57
UCL
adding new commands, 1–38
default device, 1–38
default filename, 1–38
UCL.ON
offset in CLI.FL, 2–57
UCL.RN
offset in CLI.TY, 2–57
UCL/UCF/KMON interface, 2–57
$UFLOA
SYSCOM area symbol, 1–5
UNAM1$
monitor psect, 1–18
$UNAM1 table, 2–66
UNAM2$
monitor psect, 1–18
$UNAM2 table, 2–66
UNBLOK monitor routine
to unblock a job, 2–34
UNCON$
error severity symbol, 1–7
UNLOAD keyboard command
relating to system jobs, 2–40
.UNLOCK programmed request
with USR, 1–34
.UNMAP programmed request
aligns WCBs and APRs, 3–51
described, 3–61
with Supervisor and User APRs, 3–83
User command linkage
See UCL
User Commands First (UCF) feature, 1–39
User error byte
See $USRRB
User mapping
at bootstrap, 3–2
User mode, 3–23
discussed, 3–23
initial relationship to Supervisor mode, 3–55
USERRB
See $USRRB
User service routine
See USR
User state
discussion, 2–25
in I/O processing, 2–21
returning from system state, 2–29
USR, 1–29
execution, 1–31
forcing a directory segment read, 1–31
handling directory segments, 1–31
keeping resident in FB, 1–34
load address
in SYSCOM area, 1–5
sharing between jobs, 1–34
size of, 1–41
structure, 1–29
swapping by background job, 1–35
swapping by foreground job, 1–35
swapping considerations, 1–32
swapping over MACRO programs recommendations, 1–32
with .SYNCH macro, 5–14
USR$
bit in $CNFG1, 2–48
$USRAR
RMON fixed offset 374, 2–44
size of USR, 1–41
USRAREA
See $USRAR
$USRLC

$USRLC (Cont.)
RMON fixed offset 266, 2–42
$USRLO
RMON fixed offset 352, 2–44
$USRPC
SYSCOM area symbol, 1–5
$USRRB
discussed, 1–6
error severity levels, 1–7
setting, 1–7
SYSCOM area symbol, 1–6
USRRMN
monitor psect, 1–18
USRRN$
bit in I.STATE, 2–61
$USRSP
SYSCOM area symbol, 1–5
$USRTO
SYSCOM area symbol, 1–5
with background job loading, 1–13
with foreground/system job loading, 1–25
with separated address space programs, 3–16
USRWT$
bit in I.BLOK, 2–31, 2–62
USTOP$
offset in second terminal config word T.CNF2, 4–23

V

/V
LINK option
enables XM .SETTOP, 3–68
VBGEX$
bit in $JSW, 1–9
completely virtual chaining, 3–17
completely virtual environment, 3–15
VBGEXE is executing bit, 1–9
$VBGTO
with .SETTOP, 3–77
VDT
for debugging in extended memory, 3–98
use to debug multiterminal applications, 4–33
VIRT$
bit in $JSW, 1–9
restriction, 3–11
value for completely virtual jobs, 3–7
value for privileged jobs, 3–5
VIRT$ (Cont.)
value for virtual jobs, 3–6
Virtual address
defined, 3–2
discussion, 3–84
Virtual addresses
defining, 3–31
explained, 3–84
Virtual address space
gaps in, 3–72
Virtual address window
eliminating, 3–63
Virtual address windows, 3–31
dynamic windows, 3–33
static window, 3–32
Virtual high limit, 3–69
Virtual image bit, 1–9
Virtual jobs
and interrupt service routines, 5–20
background mapping, 3–11
dynamic windows with, 3–33
mapping characteristics, 3–6
selecting virtual mapping, 3–11
static window with, 3–33
USR sliding with, 3–12
windows compared with privileged jobs, 3–8
VLOAD$
bit in I.STATE, 2–62
VMON
defined, 3–2
VRUN$
bit in I.STATE, 2–62
VRUN keyboard command
explained, 3–14
VRUN or V job running bit, 1–10
VRUNV$
bit in $JSW, 1–10
completely virtual chaining, 3–17
with VRUN command, explained, 3–14

W

W.BFPD
offset in window control block, 3–50
with .CRAW request, 3–58
W.BHVR
offset in window control block, 3–50
with .MAP request, 3–60
W.BLPD

W.BLPD (Cont.)
offset in window control block, 3–50
with .MAP request, 3–60
W.BLVR
offset in window control block, 3–50
with .CRAW request, 3–58
W.BNPD
offset in window control block, 3–50
with .MAP request, 3–60
W.BOFF
offset in window control block, 3–50
with .MAP request, 3–60
W.BRCB
offset in window control block, 3–50
with .ELAW request, 3–63
with .MAP request, 3–60
with .UNMAP request, 3–61
W.BSIZ
offset in window control block, 3–50
with .CRAW request, 3–58
with .ELAW request, 3–64
W.NAPR
offset in window definition block, 3–43
with .CRAW request, 3–58
with .GMCX request, 3–60
W.NBAS
offset in window definition block, 3–43
with .GMCX request, 3–61
W.NID
offset in window definition block, 3–43
with .CRAW request, 3–58
with .ELAW request, 3–63
with .MAP request, 3–59
W.NLEN
offset in window definition block, 3–44
with .GMCX request, 3–61
with .MAP request, 3–59
W.NOFF
offset in window definition block, 3–44
with .GMCX request, 3–61
with .MAP request, 3–59
W.NRID
offset in window definition block, 3–44
with .GMCX request, 3–61
W.NSIZ
offset in window definition block, 3–44
with .GMCX request, 3–61
W.NSTS
offset in window definition block, 3–44
structure of, 3–45
W.WCSD
offset in MCA region, 3–53
W.WCSI
offset in MCA region, 3–53
W.WCUD
offset in MCA region, 3–53
W.WCUI
offset in MCA region, 3–53
WARN$
error severity symbol, 1–7
WCB
See Window control block
WDB
See Windown definition block
.WDBBK macro
described, 3–47
.WDBDF macro
described, 3–46
invoked by .WDBBK, 3–47
$WILDD
RMON fixed offset 454, 2–46
Window
virtual address, 3–31
Window control block (WCB)
controlled by .CMAP request, 3–55
described, 3–48
structure of, 3–50
with context switching, 3–34
Window definition block (WDB)
described, 3–42
structure of, 3–43
Window status word
described, 3–45
structure of, 3–45
WINDW$
bit in I.STATE, 2–61
WPALL$
offset in second terminal config word T.CNF2, 4–23
Written into field, 3–92
WRWT$
offset in terminal status word T.STAT, 4–24
WS.C
offset in window status word, 3–45
WS.CRW
offset in window status word, 3–46
with .CRAW request, 3–58
with .ELAW request, 3–63

WS.CRW (Cont.)
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61
WS.D
offset in window status word, 3–45
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61
WS.DEF
offset in window status word, 3–45
WS. DSI
offset in window status word, 3–46
with .CRAW request, 3–58
with .ELAW request, 3–63
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61
WS.ELW
offset in window status word, 3–46
with .CRAW request, 3–58
with .ELAW request, 3–63
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61
WS.I
offset in window status word, 3–45
with .CRAW request, 3–58
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61
WS.IDD
offset in window status word, 3–45
with .CRAW request, 3–58
with .ELAW request, 3–63
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61
WS.MAP
offset in window status word, 3–45
with .GMCX request, 3–61
WS.MOD
offset in window status word, 3–45
with .GMCX request, 3–61
WS.OVR
offset in window status word, 3–45
WS.RO
offset in window status word, 3–45
with .GMCX request, 3–61
WS.S
WS.S (Cont.)
offset in window status word, 3–45
WS.SPA
offset in window status word, 3–45
with .GMCX request, 3–61
WS.U
offset in window status word, 3–45
WS.UNM
offset in window status word, 3–46
with .CRAW request, 3–58
with .ELAW request, 3–63
with .GMCX request, 3–60
with .MAP request, 3–59
with .UNMAP request, 3–61

X

XEDOFF
RMON XON/XOFF flag, 2–7
XITSW$
bit in $CNFG2, 2–51
XL handler
can support multiterminal hooks, 4–6
connecting as a terminal in multiterminal system, 4–12
XM .SETTOP
defined for all mapped monitors, 3–2
/XM option
LINK keyboard command
enables XM .SETTOP, 3–68
$XMSIZ
to of free memory list table, 3–54
XMSUBS
monitor psect, 1–18
XOFF
equivalent to CTRL/S, 2–7
$XOFF
RMON fixed offset 462, 2–46
XON
equivalent to CTRL/Q, 2–7
$XTTNR
RMON fixed offset 470, 2–46
with console terminal simulation, B–1
$XTTPB
RMON fixed offset 500, 2–47
with console terminal simulation, B–1
$XTTPS
RMON fixed offset 476, 2–47
with console terminal simulation, B–1