2003 3

1992

1980

3 1989 3

1994 12

Windows 2000/XP

UNIX SVR4 Solaris Linux

CH1

; ;

Windows 2000/XP /

CH2

;

Window2000/XP Solaris

Linux UNIX SVR4 Windows2000/XP Linux

CH3

]

: Intel x86/Pentium Windows 2000/XP Linux

CH5 I/O I/O I/O I/O ; RAID / I/O Windows2000/XP I/O Linux

CH6

Windows2000/XP Linux

CH7 ;

Windows 2000/XP

CH8

Windows2000/XP

1999 98 99 00

ppt

ppt

ppt

!

email feixl@nju.edu.cn luobin@nju.edu.cn

2003 3

CH1

1.1

1.1.1

Operating System OS

- OS
- OS
- OS
- OS
- OS

1-1

/

1.1.2

Virtual Machine

I/O
I/O

Virtual
Machine

I/O I/O

1

process

thread

1

2 3 4 5 6

2

1

2

3

4

3

I/O I/O

I/O I/O

1 2

3 4) 5

6

4

1

2 3 4 5

6 7

5

1

2

3

6

1.1.3

1

concurrence

I/O I/O CPU

I/O CPU CPU I/O
CPU

?

multitasking system

CPU
CPU I/O I/O I/O

CPU

parallelism

CPU

I/O CPU

2 (sharing)

“ ”

3 (asynchronism)

“ ” CPU
CPU

4 (virtual)

CPU
CPU
CPU CPU CPU
Spooling

IBM VM

( ) (
)

1.2

1.2.1

“ ”

FORTRAN ALGOL COBOL 1956 1958
1959

1
2
3
4
5
6
7

-
-
-

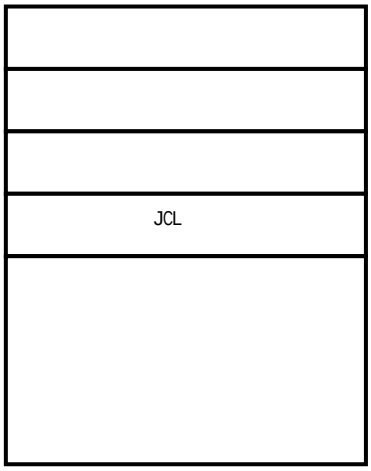
1 1
3 5%
10 6
3 50%
CPU I/O
CPU I/O

1.2.2

I/O

I/O CPU CPU
I/O I/O

I/O
I/O



1-2

resident monitor FMS FORTRAN
Monitor System IBSYS IBM 7094 Monitor System
1-2

●

Job Control Language

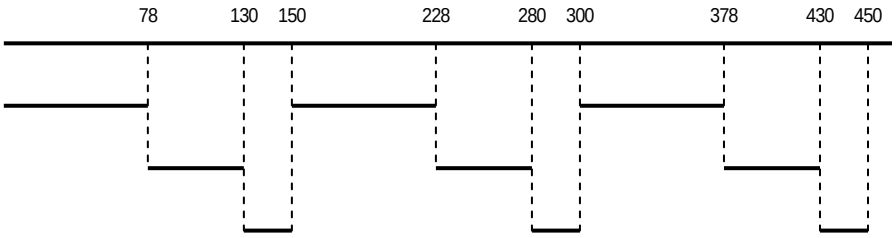
$JOB
$FTN FORTRAN $ASM $LOAD
$DATA $RUN $END

-
- I/O I/O
I/O
- FORTRAN
I/O
-

1.2.3

1

1-3 CPU
CPU



1-3

20 60
CPU I/O
multiprogramming **( )**

( )

CPU CPU

6400 500 52
2000 10
500
1-3

52/ 78 52 20 ≈ 35%

500

52 98

CPU

CPU

2000 42 1350

1-4 P1

CPU 500 52

98 CPU

2000 P2 I/O CPU

52+42 /150 ≈ 63

CPU

CPU

CPU I/O

I/O

10

CPU I/O
p n n I/O
p^n CPU
CPU $=1-p^n$
n (degree of multiprogramming) CPU
n 80% I/O CPU
10% 10
I/O 80% CPU
1MB 200KB
4 200KB 80% I/O CPU
( )=1-$(0.8)^4$=59% 1MB 4
9 CPU =1-$(0.8)^9$=87% 1MB
47% 1MB CPU 87% 96%
10%
CPU
I/O

multiprocessing system

CPU

CPU

2

I/O

- SPOOLING

1.2.4

1 “ ”

18 8 16 32

64 8 16

32 64

2

3

4

GUI

5

1

Batch Operating System

-
-
- /

2

Time Sharing Operting System

CPU

1959 MIT 1962
CTSS Compatible Time Sharing System IBM 7094 32
1965 8 IBM 360 TSS/360(Time Sharing System/360)
1965 MIT BELL GE

/

/

3

Real Time Operating System

1

2

3

4

“ ” “

”

” سبد

CP/M CDOS
Cromemco MDOS Motorola MS-DOS
Microsoft
MP/M XENIX MS-DOS
Windows OS2 MACOS Linux
GUI
API 1
CPU 2
3 64
CPU
SMP
4
( )
5

parallel processing

Stanford
V-Kernel Bell Meglos MACH
6

Network Operating System

/ Client/Server

Peer-to-Peer

1
/
2

3
4

UNIX Netware Windows NT UNIX
Windows NT Netware
C/S Netware UNIXware
Windows NT LAN Manager LAN Server

-

-
-
-
-

7

Distributed Operating System

Plan9 AT&T BELL 1987 Ken Thompson UNIX
Plan9
UNIX
15000 C
Plan9 CPU
Plan9 Naming
Space Process File System Windows System CPU
CPU Command
Amoeba
Amoeba Amoeba

Amoeba
CPU ي t `

-
-
-
-
-

8

Internet

3C Computer Communication Consumer Electronics

3C

( )

( )

( ) (embedded software)

( )

(1)

(8 16) ( )

84 ! Ã

в — υ Ŕ ὸ

HAL(Hardware Abstraction level) BSP(Board Support Package) HAL

BSP

Windows CE VxWorks( ) Linux plam OS Symbian

chorus Chorus Diba Sun Navio Oracle OS-9 Microsoft Psos ISI QNX QSSL VxWork WindRiver Windows CE Microsoft

HOPEN Windows CE

“ ” Windows CE 32 Personal Java SUN Java Java SUN Java OS for Consumers Embedded Java Hopen “ ” Hopen 10kB C Win32API Gb2312-80 Hopen Personal Java

1.3

1.3.1

1

2

3 **I/O** I/O I/O I/O

I/O

I/O

4

5

shared memory

message passing

6

1

CPU

I/O

CPU CPU CPU CPU

/

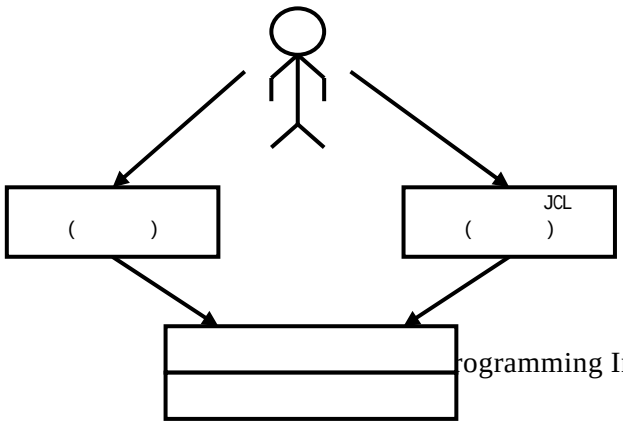
2

3

shell

1.3.2

1-5



rogramming Interface

system call ”

1-5

( )

( )

( )

1.3.3

1

(

)

“ UNIX ”

POSIX1003.1 Portable Operating System IX

UNIX

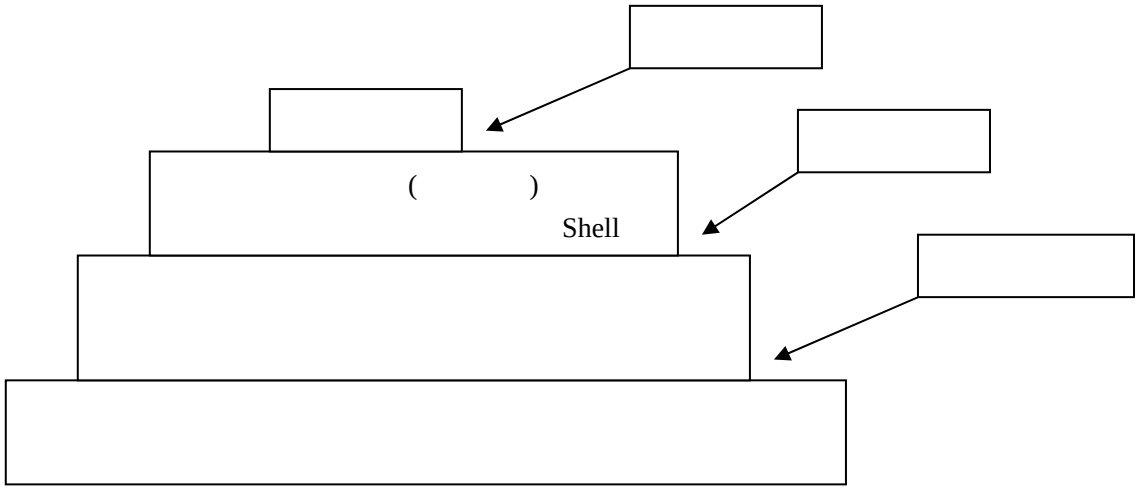
(C)

( )

UNIX Linux Windows OS 2 C

C 1-6

UNIX/Linux



1-6 Unix/Linux

1

2

3 I/O

4 5)

6

Windows API Kernel User GDI Kernel
User
GDI
Windows API

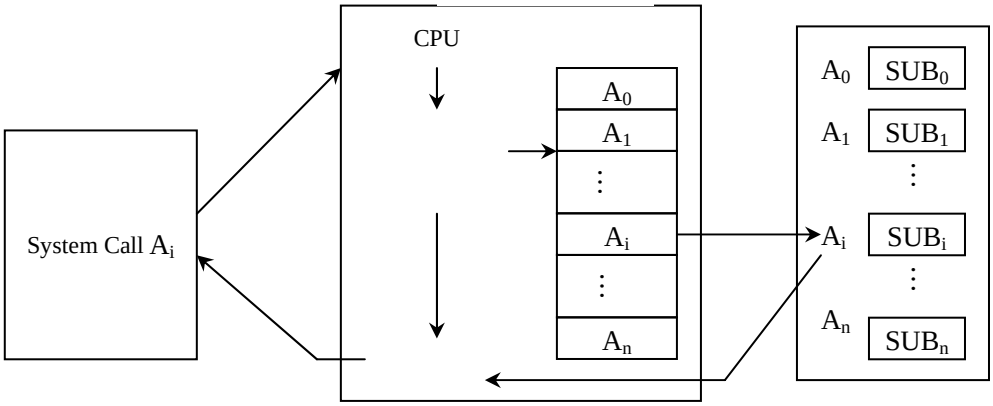
Windows DLL Dynamic Link Library

2

supervisor trap interrupt

1-7

CPU



1-7

3

(1)

(2)

(3)

(4)

4 Linux

Linux UNIX

Linux 190 Shell

Linux 1

C

2 API

Linux lib.a

Linux entry(sys_call_table)

include/asm/unistd.h

ENTRY(sys-call-table)

```
long SYMBOL_NAME(sys_ni_syscall)    0
long SYMBOL_NAME (sys_exit)         1
long SYMBOL_NAME (sys_fork)         2
long SYMBOL_NAME (sys_read)         3
long SYMBOL_NAME (sys_write)        4
long SYMBOL_NAME (sys_open)         5
long SYMBOL_NAME(sys_close)         6
  … … …
long SYMBOL_AME (sys_ni_syscall )
long SYMBOL_NAME (sys_ni_syscall )
long SYMBOL_NAME (sys-vfork )       190
```

Linux

Linux 0x80 int80h

1.3.4

1

( )

(1) ——

1)

Command arg1 arg2 argn
Command

Linux 1 cd chmod chown chgrp comm cp crypt diff file find ln ls mkdir mv od pr pwd rm rmdir 2 at kill mail nice nohup ps time write mesg 3 cat crypt grep norff uniq wc sort spell tail troff 4 cc f77 login logout size yacc vi emacs dbs lex make lint ld 5 date man passwd stty tty who

MS-DOS BAT BAT

UNIX Linux Shell
Shell
Shell Shell Shell

2

GUI Graphics User Interface

GUI WIMP (Window Icon Menu Pointing device)

GUI GUI Xerox Palo Aito Research Center 1981 Star8010 1983 Apple Apple Lisa Macintosh GUI Microsoft Windows IBM OS/2 UNIX Linux X-Window GUI GUI GUI MIT X-Windows Windows NT Visual C++ Visual Basic GUI

3)

(2) —— ( )

JCL Job Control Language

UNIX/Linux Shell JCL

JCL

JCL

JCL

2

command interpreter

( )

“ ”

“ ”

delete G
delete G
delete delete

ë ر ك م ك X ز لأ

1.4

MIT 1963 CTSS 32000
IBM OS/360 4000 5000
1975 MIT Bell Multics 2000
Windows 2000 2500 3200 Brooks
OS/360
OS/360
20 60

10 20
CPU

POSIX

1.4.1

1

(microkernel)

(monolithic kernel)

Linux

(module)

Linus

Linux

1

2

3

1

2

Solaris

3

Lniux

top half bottom half

Linux Windows 2000/XP

ISR ISR

DPC DPC

DPC

IRQL Dispatch/DPC IRQL

DPC

4

1)

2

3

2

1

2

3

3

CPU

multithreaded process

4

( ) 7 ÃÇ ¡ ÈÖ Ì 86wjØ @ , ¢ ìÊ@?ZØ@ÇZQ ” ÁÁ>ZÌ

-
-
-

-

5

1975 PDP 11/45
Solo

1.4.2

IBM S/360

1.4.3

A_0

A_1 A_1

A_1 A_2 “ ——

—— ”

(1)

(2) ( ) CPU

( )

(3

(4)

CPU

Edsger.W.Dijkstra 1968 THE

THE Electrologica X8

6 0

1

2

3 I/O I/O

4

I/O 5

1.4.4

IBM CP/CMS VM/370 Seawright and MacKinnon 1979 IBM S/390
1 2
VM/370
1-8(a)

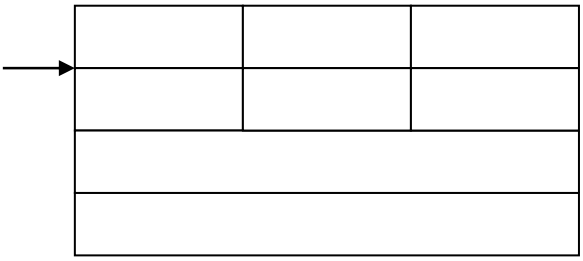
VM/370

CPU CPU
CPU CPU
SPOOLING

“ ”
/ CPU I/O

OS/360
CMS Conversational Monitor System

1-8(b) CMS
CMS VM/370 CMS
I/O I/O VM/370
VM/370



1.4.5 /

1 /

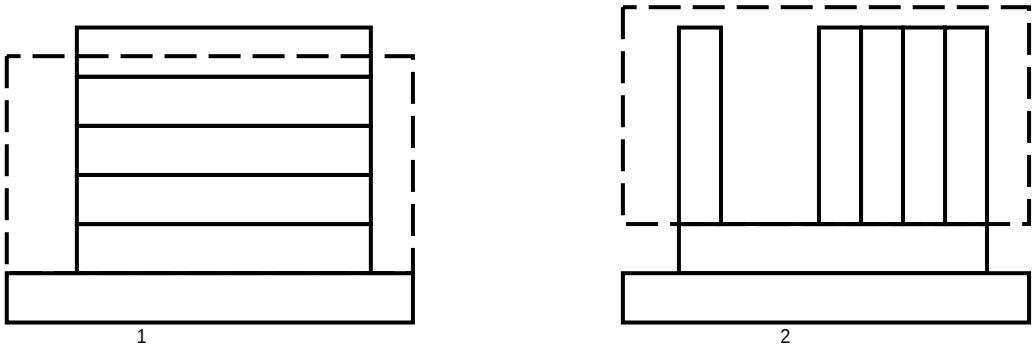
/ Client/Server
- Mach /

/
C/S ;

(
C/S)

/ ;

(microkernel) /



1-9

1-9

I/O

(1

(2

(3

CPU

CPU

(4

API

(5

(6

Windows2000/XP

/

Mach Chorus

300KB 140 L4
12KB 7

2

IPC I/O

(1

1-10

1-10

- grant
- map
- flush

(2

header

Ports

CPU

(3)I/O

I/O

```
driver thread;
do
      wait for (mhg      sender);
      if sender = my_hardware_interrupt
         {
              read/writer I/O ports;
              reset hardware interrut
         }
      else  …
while (true);
```

1.4.6

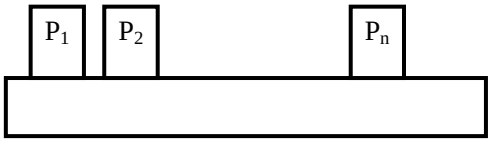
? ?

OS

OS

1

1-11



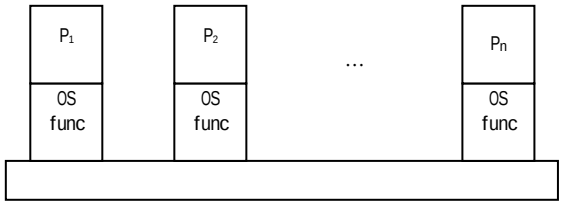
1-11

2 OS

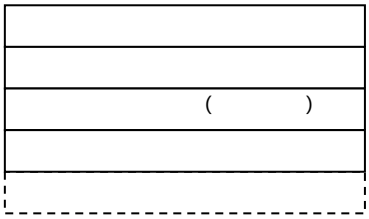
UNIX OS

1-12

1-13 OS



1-12 OS



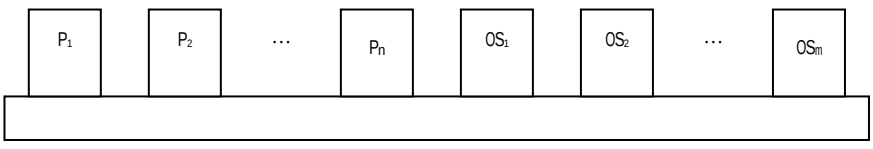
1-13 OS

3 OS

OS

Client/Server Window 2000/XP

1-14



1-14 OS

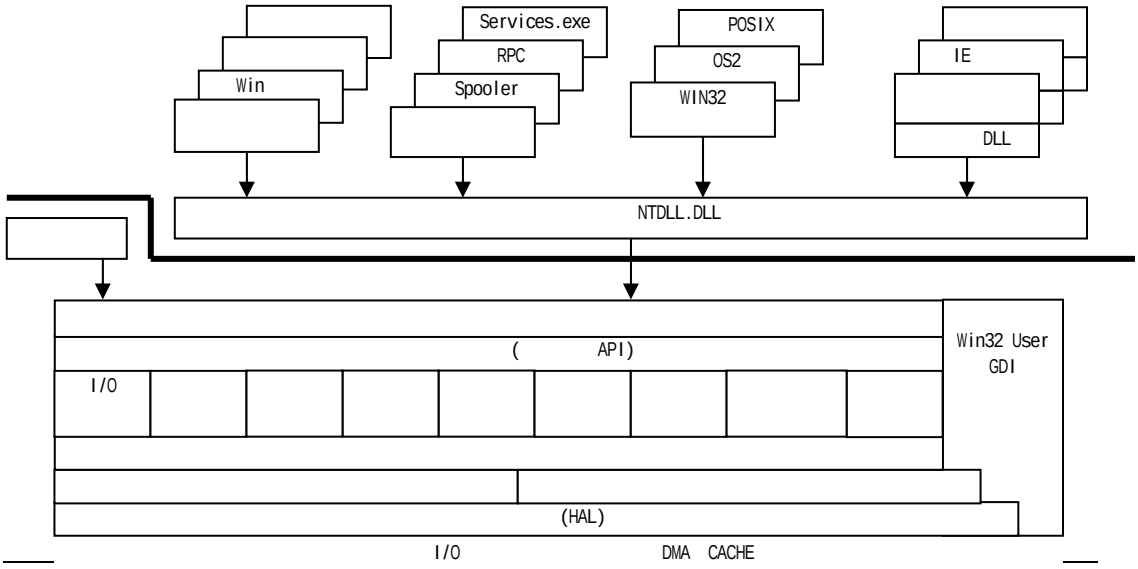
1.4.7 Windows 2000/XP /

Windows 2000/XP (1)
(2)
(3)
(4) DOS Windows
UNIX OS2 Netware (5)
Windows 2000/XP (1)
(2 Mach

Windows 2000/XP

Windows2000/XP /

1-15 Windows 2000/XP



1-15 Windows 2000/XP

I/O

Windows 2000/XP

(1 HAL

Windows 2000/XP HAL(Hardware Abstruct Level)

Windows 2000/XP

HAL

I/O DMA

HAL.DLL

(2

I/O

I/O I/O Windows2000/XP

HAL

I/O

I/O HAL

I/O I/O I/O

I/O

I/O

Windows 2000/XP WDM(Windows

Driver Modle) WDM

-

- -

I/O

WDM

(3)

NTOSKRNL.EXE

Windows

“ ”

“ ”

APC Asynchronous Procedure Call
DPC Deferred procedure Call I/O

“ ”

Windows 2000/XP

HAL
(4)
Windows 2000/XP NTOSKRNL.EXE

NTDLL.DLL WIN32 API

LPC NtQuerylnformationxxx
NtCreatePagingFile Windows 2000/XP DDK

Windows 2000/XP
“ ”

I/O
I/O I/O
I/O API

I/O

I/O

Windows 2000/XP

LPC

Local Procedure Call

/ LPC RPC

“ ”

(5 NTDLL.DLL

NTDLL.DLL NTDLL.DLL

Windows 2000/XP

Windows

200 NtCreateFile NtSetEvent

WIN32 API

NTDLL

NTOSKRNL.EXE

NTDLL

WIN32

APC

(6)

Windows2000/XP

- Idle ID 0 CPU Idle

CPU

Task Manager System Idle PSTAT.EXE

PVIEWER.EXE Idle PVIEW.EXE

TLIST.EXE QSLICE.EXE System

- System System System ID 2 System

NTOSKRNL.EXE

System

- SMSS.EXE System

LPC

WIN32

POSIX OS2 WinLogon

SMSS CSRSS WinLogon SMSS

- WIN32 CSRSS.EXE WIN32
- WinLogon.EXE
- LSASS.EXE WinLogon

(7)

SERVICES.EXE

UNIX

Win32

Win32

Windows2000

RPC

(8

“ ” Windows 2000/XP

WIN32 POSIX(POSIX.1) OS/2 1.2

DLL Win32

Windows2000/XP Win32 Win32

CSRSS

GDI

Windows

CPU OS Windows2000/XP

Windows 2000/XP 2000/XP i 2

DOS Windows OS/2 LAN Manager POSIX (9 ½

™

(9

WIN32 Windows 3.1 MS-DOS POSIX OS/2

1-13 “ ” “ ”

Windows 2000/XP ( ) Windows

2000/XP “ ”

( )

DOS DOS4.0
CPU DOS

DOS DOS PC
PC
DOS

Windows DOS
DOS

1.5.2 Windows

1 Windows

Microsoft 1975

1983 11 Microsoft Windows WindowsXP Windows
20
Windows 90% PC
Microsoft Xerox
GUI 1981 Apple Lisa 1983 Macintosh 1984
Microsoft 1983 11 Windows
11 1985 11 Windows
1.01 1987 Windows 2.0
1990 Windows 3.0
1992 4 Windows 3.1 Windows DOS

Windows 1.x Windows 3.x DOS

1995 8 Microsoft Windows 95
DOS Windows95
Microsoft Windows 97 Windows 98 Windows 98 SE Windows Me
Microsoft Windows Millennium Edition Windows3.x Windows 9x

Windows Windows NT
Windows 2000
PC

(7) (8) (9

3 Windows 9x

Windows 9x Windows95 Windows 97 Windows 98 Windows 98 SE Windows Me Windows

Windows 9x

(1) 32
16 (2) 32
(3) “ ” Windows
(4) USB(Universal Serial BUS) AGP(Accelerated Graphics Port) ACPI(Advanced Configuration and Power Interface) DVD (5) MPEG WVA MPEG AVI Apple Quiet Time (6)
Internet (7)
(8 FAT32

4 Windows NT

Windows RISC CPU /

1993 Windows NT New Technology Windows (1) SMP CPU
(2) (3)32
(4) API ψ ϕ ϖ خھ ϒ

1.5.3 UNIX

1 UNIX

UNIX

Kenneth Lane Thompson Dennis MacAlistair Ritchie 1969 DEC PDP-7 1971 PDP-11 1973 Ritchie BCPL(Basic Combined Programming Language M.Richard 1969) C UNIX C 3 UNIX 1974 7 ”The UNIX Time-Sharing System” Communication of ACM UNIX 1975 UNIX 6 1978 UNIX 7 UNIX UNIX 70 UNIX UNIX

BSD UNIX AT&T UNIX AT&T UNIX 1981 System 1983 System (PWB) 1984 System .2 1987 System .3 SCO 1978 PC UNIX XENIX UNIX AT&T UNIX SVR3.2 AT&T UNIX Berkeley UNIX BSD Berkelry Software Distribution BSD PDP VAX 1978 1BSD 2BSD 1979 3BSD 1980 4.0BSD 4.1BSD 4.2BSD 4.3BSD UNIX BSD 1993 4.4BSD TCP/IP UNIX Sun OS Solaris 4BSD 4.3BSD Sun OS AT&T UNIX SVR3.2 SVR4.0(System V Release 4) 1989 UNIX BSD UNIX

UNIX UNIX UINX

UNIX (1) C (2) (3) (4) I/O (5) CPU (6) Shell (7) (8

UNIX UNIX IBM AIX SUN Solaris Berkeley UNIX BSD DEC Digital UNIX Compaq Tru64 UNIX HP HP-UX SGI Irix SCO SCO UNIXWare Open Server AT&T SVR 20 90 UNIX 100 UNIX IEEE UNIX

POSIX UNIX
UNIX POSIX
UNIX SVID System Ⅴ Interface Definition XPG X-open Portability Guideline UNIX
C
UNIX
Ritcchie Thompson 1983 ACM (ACM Turing Award) (Software System Award)

2 Solaris

SunMicrosystem Solaris UNIX SVR4
32 Solaris2.x
SUN OEM SPARC Intel 80x86
Solaris SMP
C/S

SUN SUN OS 1982 SUN OS 1.0 BSD 4.1 MOTOROLA 680X0 1 MIPS 1M
20 80 UNIX SUN
SUN 1985 SUN OS 2.0
RPC NIS
NFS SUN NFS Network File System 1984
NFS
NFS C/S UDP RPC Remote Procedure Call (1)
(2)
(3
SUN 1988 SUN OS 4.0 MOTOROLA 680X0 SPARC Intel 1990 SUN OS 4.1 1992 SUN OS 4.1.3 ASMP

SUN

1992 SUN Solaris 2.0 Solaris 2.0 SVR4
SUN OS 4.X Solaris 1.X Solaris 2.X SUN OS 5.X 1992 Sun Solaris 2.1 Solaris 2.6 1998 Sun 64 Solaris2.7 2.8

1.5.4 Linux

1

Free Software or Freeware GPL General public License

free Richard Stallman 0 1 2

GPL ‘ ’ ‘ ’ Bill Gates “ ”

, , , ,

GNU GNU is not UNIX Richard Stallman GPL

1984 MIT Stallman GNU FSF Free Software Foundation GNU UNIX UNIX GNU gcc/gcc^{++} shell Linux GNU gcc/gcc++ Objective C Free BSD Open FORTRAN77 C BSD email BIND Perl Apache TCP/IP IP accounting HTTPserver Lynx Web GPL GPL

2 Linux

Linux Linus Torvalds 1991 Linus Internet Linux Internet

Linux Linux GNU Linux GNU Linux 1998 Linux Internet Windows NT IBM Intel Oracle Sun Compaq Linux Linux Linux Linux Linux

Linux UNIX UNIX

Linux
Linux GPL

Linux
Linux Internet Linux Internet
Linux Linux UNIX Internet
Linux
UNIX Windows
Linux 1 UNIX
2
Internet
3 TCP/IP

2^{52} UNIX
Internet
RS/6000 AIX FTP email Web
CAD CAE

3 OS/390 VM DOS/VSE

IBM S/390(System/390) CMOS OS/390
VM Virtual Machine DOS/VSE Disk Operating System/Virtual Storage Extended 2000 12 IBM Z900 2001 3 OS/390 ZOS
IBM S/360 S/370 S/390
IBM S/390
70% S/390

IT
— — S/390
OS/390 S/390 G6
1600MIPS
OS/390 MVS Multiple Virtual Storages 1996 IBM OS390 1.1 1998 IBM OS/390 2.5 OS/390 2.7
OS/390 X/Open UNIX
OS/390 S/370 TCP/IP
LAN
OS/390
Client/Server
OS/390 S/370 S/370 MVS/ESA390
Enterprise System Architecture ESA 10 240MB 256
ESA/390LPAR Logical Partitioning
CPU 20 LPAR CPU
OS/370 MVS/ESA370 MVS/ESA390
IBM VM DOS/VSE

4 OS/400

AS/400 64 RISC OS/400
SSP IBM System Support Program
AS/400 IBM
AS/400 OS/400
OS/400
OS/400
C
C++ Cobol RPG Java
Licensed Internal Code LIC IBM
AS/400
LIC CPU
OS/400 IBM AS/400
IBM DB2 4.4 AS/400 OS DB

OS/400

5 PC

PC DOS OS2 Windows
OS/2 Microsoft IBM 1987 PS/2
IBM OS/2 2.0 1994
Workplace Shell
32 OS/2 Warp Windows 95 1996
OS/2 OS/2 Warp Server4.0 OS/2 Warp Server SMP
TCP/IP 6 I/O Navigator Java

OS/2 1 2 16
32 CPU 3 4GB 4
5 6
API 7 MS-DOS
MS-DOS OS/2

1.5.6

1 Mach
Mach Carnegie-Mellon Richard Rashid Accent
1984 Accent Mach
Mach UNIX UNIX
ARPA Mach
Berkeley UNIX ARPA
Mach 1986 CPU VAX 11/784
IBM PC/RT Sun3 1987 Mach Encore
Sequent IBM DEC HP

2 Mac OS

Mac OS Apple Macintosh Apple Mac OS

Apple IBM PC Mac OS 1 2 3 4 5 6 7 Macintosh

3 Netware

Novell PC 1983 IBM Apple UNIX Dec Netware NOS Networking Operating System Netware DOS OS/2 MAC UNIX

Netware Netware5

4 Minix

UNIX

UNIX AT&T UNIX 7 UNIX

UNIX

Vrije A. S. Tanenbavm UNIX Minix Minix AT&T ‘ UNIX ’ Minix

Minix C

Minix “ Small is Beautiful ” Minix Minix2.0 TCP/IP 4GB 5 200 Minix Internet USENET Minix Minix Minix Linus Torvalds Minix Linux

1.6

( )

20

60

CPU

CPU

( ) ( )

/ /
(SMP)

/

Windows 2000/XP

/

DOS Windows UNIX
Solaris Linux IBM Mach Mac OS
Netware Minix

,

1.
2. ?
3. ? ?
4. ?
5. ?
6.
7. ? ?
8. ? ?
9.
10.
11.
12. I/O I/O
13. ?
14.
15. ? ?
16.
17.
?
18. ?
19. ? ?
20.
21.
22.
23.

24.
25.
26. ?
27.
28.
29. ? ?
30. ?
31. ?
32.
33.
34. ?
35. ?
36. / ?
37.
38. MS-DOS UNIX Windows 2000/XP VM/370 Mach
39. Windows 2000/XP
40. Windows 2000/XP
41. Windows 2000/XP ?
42. ? Windows 2000/XP ?
43. /
44.
45.
46.

1 1MB 200KB 200KB I/O 80% 1MB CPU

2 A B A 50ms 100ms 50ms 100ms B 50ms 80ms 100ms (1) CPU (2) A B CPU

3 A B C I/O

	A	B	C
	C_{11} 30ms	C_{21}=60ms	C_{31}=20ms
	I_{12} 40ms	I_{22}=30ms	I_{32}=40ms
	C_{13} 10ms	C_{23}=10ms	C_{33}=20ms

1ms

4 CPU I/O(I1,I2)

Job1 I2(30ms) CPU(10ms) I1(30ms) CPU(10ms) I2(20ms)
Job2 I1(20ms) CPU(20ms) I2(40ms)
Job3 CPU(30ms) I1(20ms) CPU(10ms) I2(10ms)
CPU I1 I2 Job1 Job2 Job3
CPU I1 I2 (1)
(2) CPU (3)I/O

5 CPU I/O(I1,I2)

Job1 I2(30ms) CPU(10ms) I1(30ms) CPU(10ms)
Job2 I1(20ms) CPU(20ms) I2(40ms)
Job3 CPU(30ms) I1(20ms)
CPU I1 I2 Job1 Job2 Job3
CPU (1)
(2) CPU (3)I/O

6 3 A B C A B C
A (20) I/O(30) (10)
B (40) I/O(20) (10)
C (10) I/O(30) (20)
I/O( )
CPU ?

7 3 A B C A B C
A (30) I/O(40) (10)
B (60) I/O(30) (10)
C (20) I/O(40) (20)
I/O( )
CPU ?

8 3 A B C A B C CPU I/O
A 60 20 30 10 40 20 20 (ms)
I/O2 CPU I/O1 CPU I/O1 CPU I/O1
B 30 40 70 30 30 (ms)
I/O1 CPU I/O2 CPU I/O2
C 40 60 30 70 (ms)
CPU I/O1 CPU I/O2

I/O

CPU ?

9 A (CPU)10 ( )5 (CPU)5 ( )10
(CPU)10 B ( )10 (CPU)10 ( )5 (CPU)5
( )10 A B CPU ?
10 2ms
60HZ CPU ?

CH2

——

“ ”

2.1

2.1.1

- SISD
- SIMD
- MISD
- MIMD

MIMD MIMD
MIMD
MIMD MSP Main/Slave
Multiprocessor SMP Symmetric Multi-Processor

SMP 64 SMP

I/O

Dec HP IBM SUN SGI SMP

MIMD

Cluster

MIMD

RAID

SMP

2.1.2

-
-
- /
- I/O I/O
- I/O I/O
-

PC Program Counter IR Instruction Register

I/O

2.1.3

PC IR PC

1 2
3

4
5) I/O

I/O

Privileged Instructions

PSW

2.1.4

Intel Pentium 4 0 3
4

- 0 I/O
- 1
- 2
- 3

Pentium Windows 0 3

2.1.5

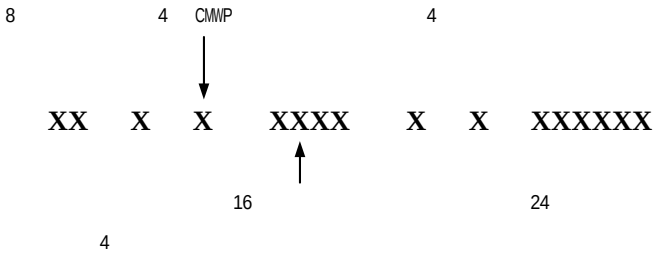
PSW Program Status Word 2-1
IBM360/370

- 8 0-7 0
  1 0-7 0-6
- 4 8-11 4 0
  PSW

- CMWP 12-15 PSW / /
  / /
- 16
- 2 32-33 01/10/11

- 2 34-35
- 4 36-39 1 0
  39

- 24 40-63



PSW
PSW

- (1) (2)
  (3

-
-

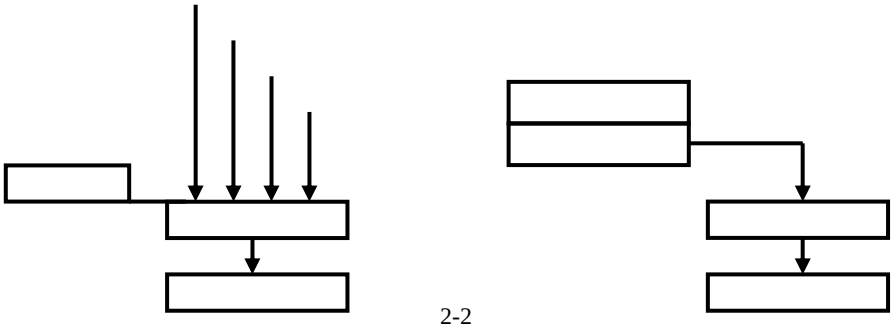
Intel Pentium EFLAGS EIP

32 EFLAGS 16 FLAGS

- OF SF ZF AF CF PF ZF
- DF DF 1 DF 0 VM 86 1 8086 TF 1 IF 1
- IOPL I/O NT RF EIP 16 IP 32 IP

-
- 0
-
-

2-2



2-2

-
- I/O
-

IBM Windows2000/XP

( )

CPU

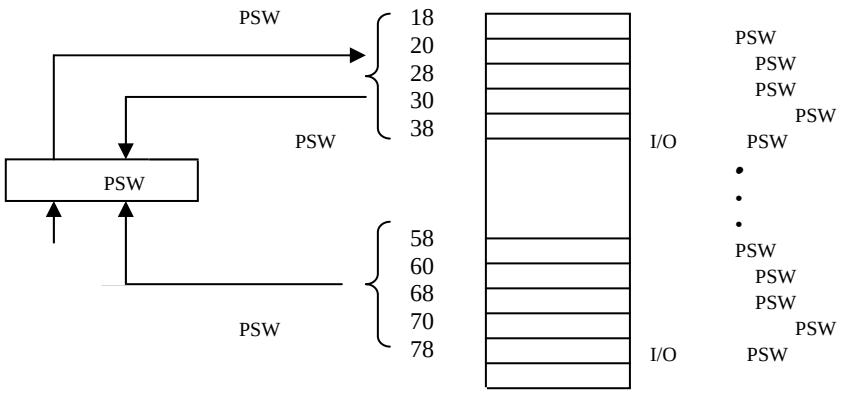
CH3

2.2.3

-
-
-

“ 0 ”

(16)



2-3 IBM

2-4 IBM

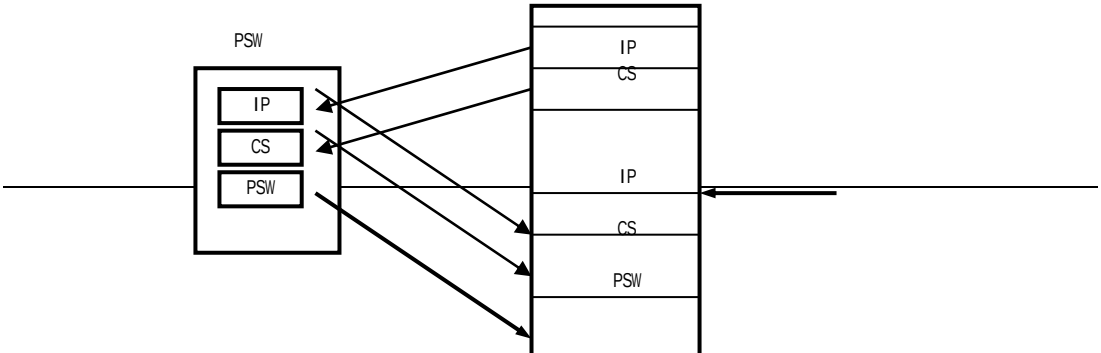
(16)

PSW(

I/O) CPU PSW

PSW PSW

PSW PSW PSW



-
-
-
-

2

2

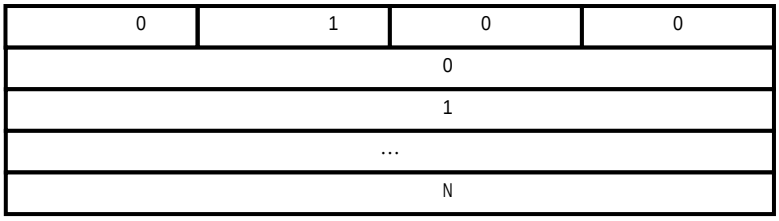
? on

on < > < >

on fixed overflow go to LA
LA

on fixed overflow go to LB
LB LA

2-5



2-5

0

?

“ 0 ”

“ 0 ”

2 0

1

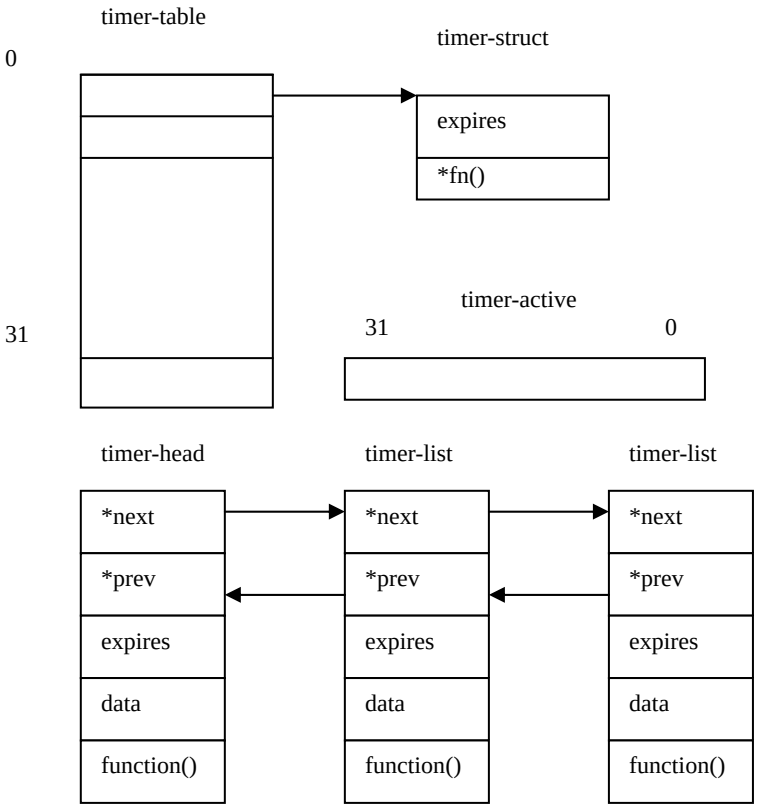
- on
- on
- on
-
-

3

1

ﻝ ﻟ

Linux (clock tick) jiffies Linux 2-6 32 timer-struct timer-active 32 timer-list



2-6

jiffies bottom half bottom half timer-active timer-list

Linux CPU jiffies (accounting time) Linux

Linux

- real tick SIGALRM
- virtual SIGVTALRM

- profile
  SIGROF

Linux
task-struct Virtual
profile tick
Real
timer-list real

SIGALRM

2

4 I/O

1 I/O

2 I/O

- n
-
-
-

2.2.6

1

?

I/O

I/O I/O

2

1 0

3

CPU

?

- —

I/O

I/O

-

‘ ’

-

2.2.7 Windows 2000/XP

1 Windows 2000/XP

Pentium Windows

“ ”

I/O

I/O

2-7

“ Trap Frame ”

ISR

Interrupt Service Routine

2 Windows 2000/XP

1

Windows2000/XP

IRQL Interrupt Request

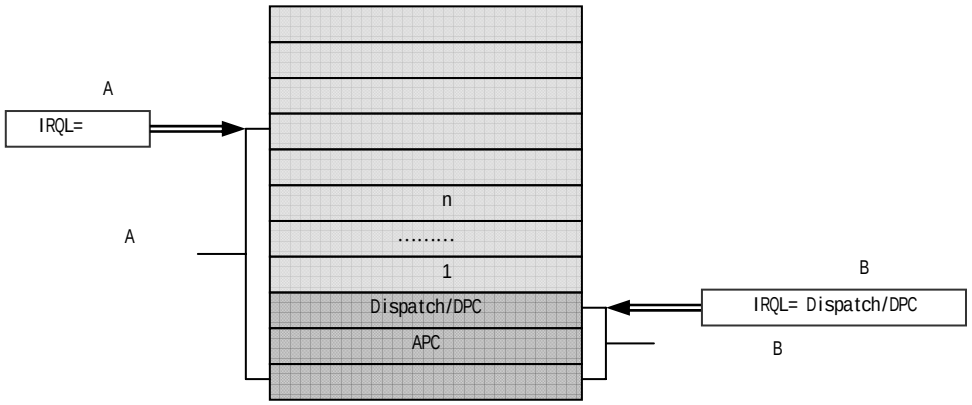
Level IRQL

IRQL IRQL

2-8 x86 IRQL IRQL

3

IRQL



2-9 Windows

IRQL

IRQL IRQL

IRQL IRQL

IRQL

3 Windows2000/XP

1)

IRQL

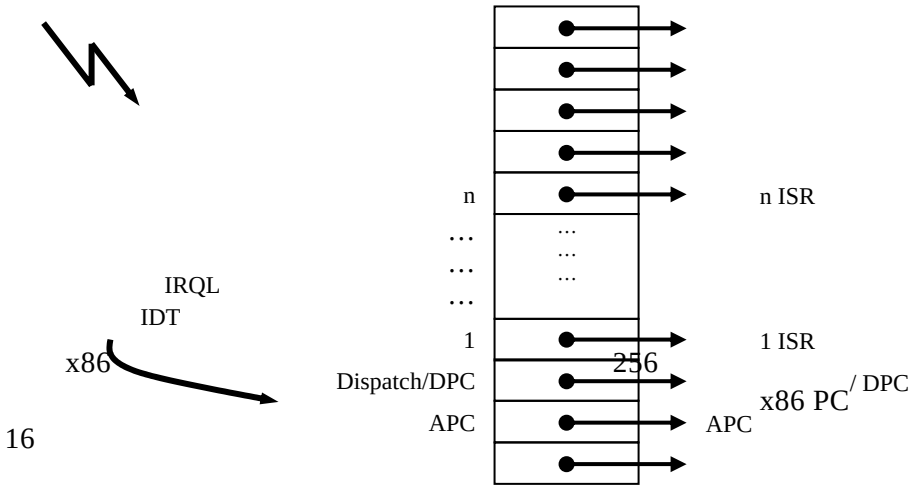
Windows2000/XP **IDT Interrupt Dispatch Table**

IRQL 2-10

x86 IDT PCR Processing Control Region

PCR —— PRCB

HAL



2-10 Windows

IDT

IRQL

IRQL

IDT

ISR

I/O

——

ISR

ISR ISR IRQL

ISR “ ”

ISR

ISR IDT ISR

ISR

ISR

IDT

ISR

IDT

ISR ISR

ISR

2)

Windows2000/XP

I/O

4

DPC

IRQL Dispatch/DPC

Dispatch/DPC

Dispatch/DPC IRQL Dispatch/DPC

IRQL Dispatch/DPC

IRQL DPC DPC

“ ”

DPC DPC

DPC I/O

DPC DPC

DPC

DPC DPC “ DPC

” DPC DPC

DPC

DPC DPC Dispatch/DPC

DPC IRQL IRQL

APC APC

IRQL DPC DPC

DPC

DPC

IRQL

0

Dispatch/DPC IRQL

DPC IRQL DPC

DPC

APC (

) APC

2 IRQL APC ( )

APC APC APC APC APC

DPC DPC APC

APC APC APC APC

APC APC APC

APC APC APC

“ ” “ ” APC

APC ( )

APC

I/O APC

POSIX APC POSIX

POSIX

APC I/O

I/O

I/O

Win32 API ReadiEX WriteFileEX QueueUserAPC APC

ReadiEX WriteFileEX I/O

APC I/O

5 Windows 2000/XP

WIN32

Structure Exception Handling

“ ”

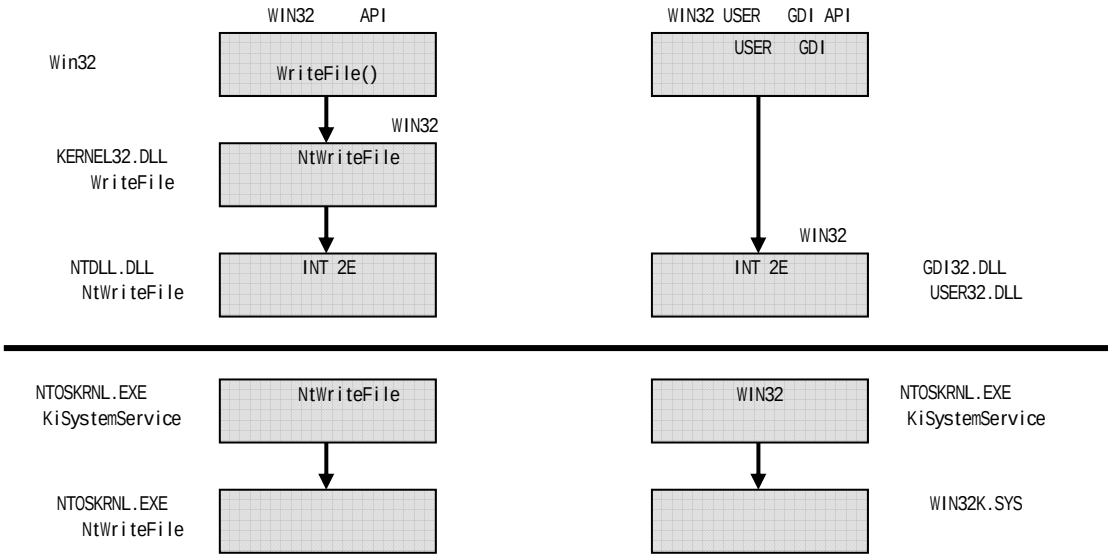
“ ”

6 Windows 2000/XP

x86 INT 2E System Service Dispatcher 2-11 (System Service Dispatch Table)

Windows2000/XP NTOSKRNL.EXE WIN32 WIN32K.SYS WIN32 USER GDI WIN32 WIN32 USER GDI WIN32 USER GDI Windows2000/XP NTDLL.DLL DLL NTDLL WIN32 USER GDI USER32.DLL GDI32.DLL

2-12 KERNEL32.DLL WIN32 WriteFile NTDLL.DLL NtWriteFile NtWriteFile NTOSKRNL.EXE KiSystemServic2afc>Tj/TT2 1 Tf2X



2-12Windows200

2.2.8 Solaris

1 Solaris

SPARC ULTRA SPARC Solaris

system call

interrupt

trap

2 ULTRA SPARC

SPARC ULTRA SPARC

SPARC

SPARC ULTRA SPARC

1

2

3

8

TBA Solaris

4M

SPARC ULTRA SPARC

SPARC ULTRA SPARC 1

2

3 4

5 6

SPARC ULTRA SPARC

PIL(Processor

Interrupt Level) PIL

ULTRA SPARC 5 0

4 3

3 Solaris

Solaris

Solaris

1 15 15

9

9 9

Solaris

R ȯ ¥

1 =

1)
C 2)
3)
(
) (
)
2

(bottom half handling) Linux
Linux
(top half) (bottom half)
top
half bottom half top half
bottom half top half
bottom half
bottom half bottom half
bh-active
---
bottom half
Linux
bottom half

bottom half Linux
32
top half bottom half
bottom half
bottom half
top half bottom
half bottom half
100 top half
100 bottom half 1 bottom half
” ”
3
2-13 32 bottom half
bh-base bottom half bh-mask bh-active bottom half
bottom half
bh-mask n bh-base n bottom half
bh-active n n bh-mask
bh-active “ AND ” “ ”
0 bottom
half 0 bottom half
1 bottom half
immediate bottom half

(tq-immediate)

4

(1) (TIMER-BH)

(2) (CONSOLE-BH) (3)
(NET-BH) (4) (IMMEDIATE-BH)

bh-active immediate
immediate bottom half
bh-active 1
bh-active
bottom half 0 1 31
bh-active bottom half
top half bottom half
( ) timer-interrupt top half do-timer
bottom half timer-bh timer-interrupt CPU
top half do-timer jiffies(1)
( ) W,X

2.3

2.3.1

?

I/O

I/O I/O

“ ” “ ”

“ ”

“ ” “ ”

“ ”

“ ”

“ ”

P A

B P B

P “ ”

P A

P ? B A ?

P

P P

P P P

P P

B P A

active

1978

-
-
-
-
- CPU
-
-

m P_1 P_2 … P_m

P_1 n_l

P_2 P_2 n_2 P_3 … P_m n_m

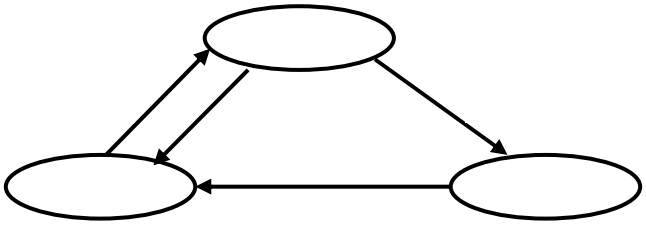
P_1 …

2.3.2

1

- running
- ready
- wait blocked sleep

2-14

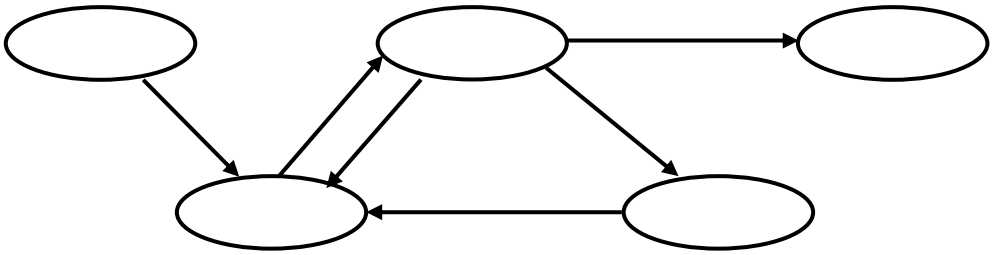


2-14

- —→
- —→
- —→
- —→ CPU

2

new exit 2-15



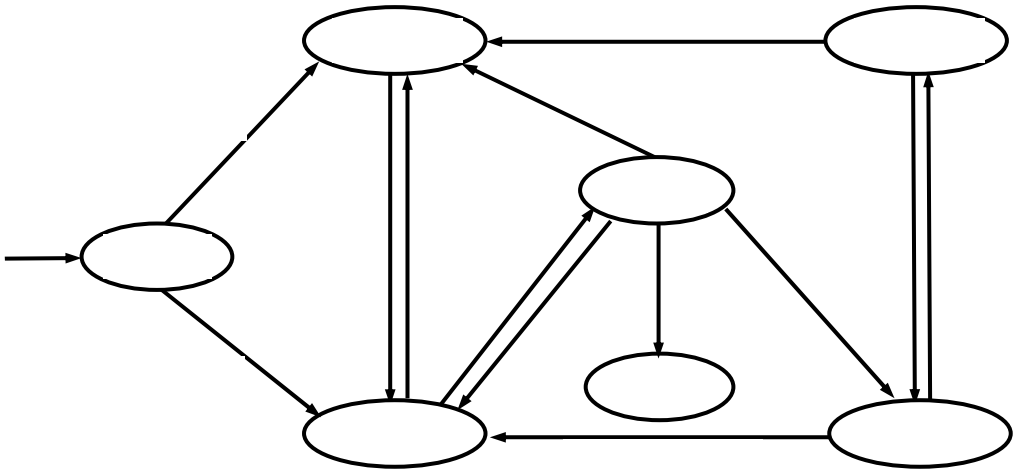
2-15

- NULL—→
- —→
- —→
- —→NULL
- —→
- —→

3

suspend

-
-
-
-
-
-



2-16

2-16

ready suspend blocked suspend

- —→
- —→
- —→
- —→
- —→

- —→
  CPU
- —→

-
-

-
-

2.3.3

1

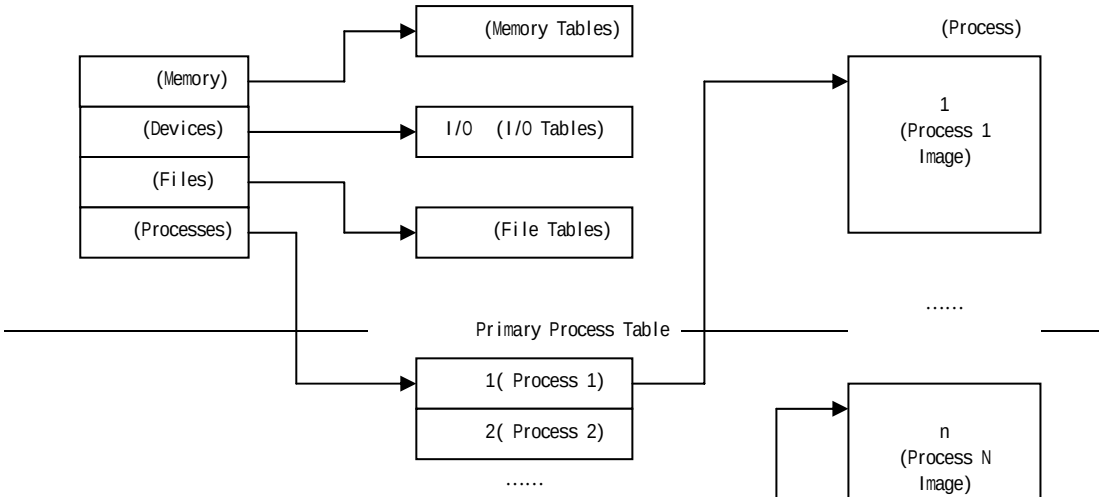
I/O

-
-

- I/O I/O I/O
  I/O I/O I/O

-

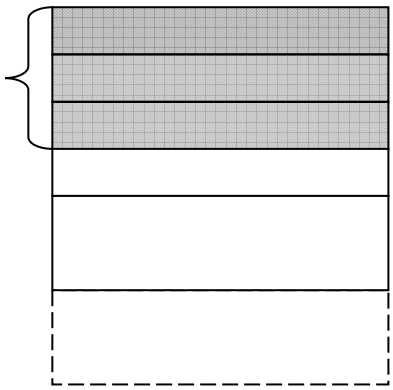
2-17



2

process context

- user -level context
- system -level context
- register context



2-18

Process Image

-
-
- / /
-

2-18

3

PCB Process Control Block

-

-

PSW

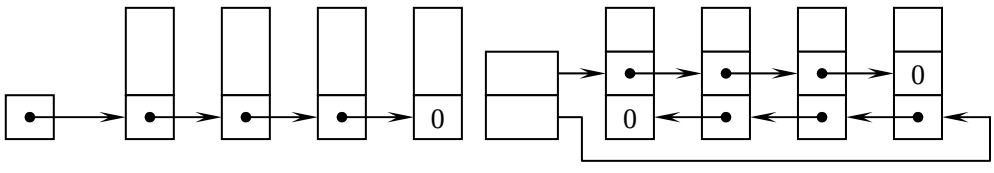
- 1
2
3
4 5 CPU
CPU
6
7
I/O

PCB
PCB PCB
PCB

4

PCB
PCB PCB

process queues PCB

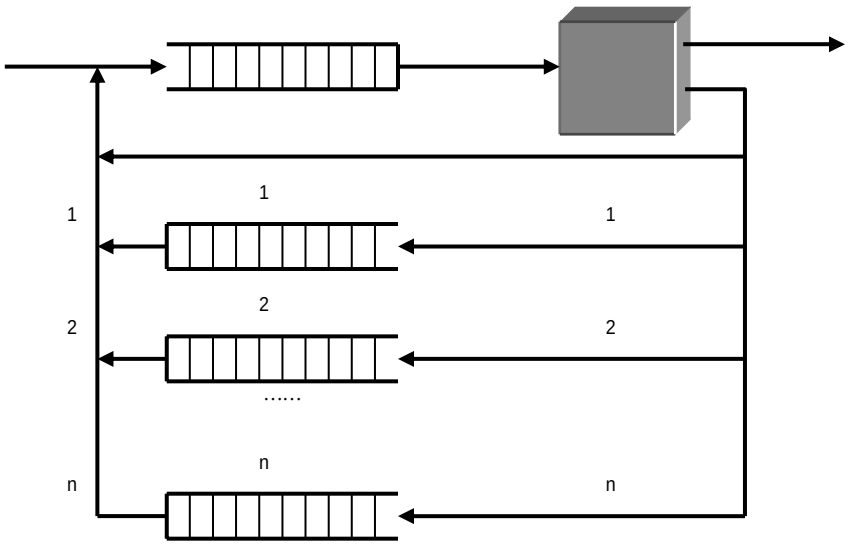


a b

2-19

2-19 a b

2-20



2-20

2-15 b

- 1 0
- 2 0
- 3

Q P
R Q Q R
Q P P R R
P Q Q

PCB PCB
PCB PCB

2.3.4

context layer

(1)
(2) CPU
(3) CPU
(4)

-
-
-
-
-
-
-

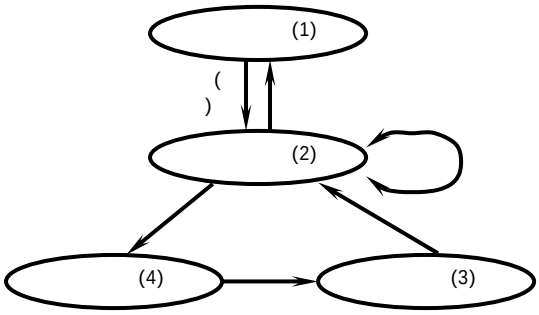
CPU

-
-
-

PCB

2-21

-
-
-
- I/O



2-21

1 2

4

3

2.3.5

(Primitive)

( )
( )

1

-
-
-
-

“ ”
parent process
child process
UNIX/Linux

- PCB PCB UNIX/Linux
task_struct
-
-
-
- PSW
-
-

Linux fork( )
clone fork( )

clone fork
Linux fork( ) clone( )
task_struct fs files sig

mm
count 1 count 0
copy on
write
mm
mm

UNIX Solaris
Linux2.4
size-of-memory-in-the-system/kernel-stack-size/2
512MB 512 × 1024 × 1024/8192/2=32768
Linux

2

I/O

I/O

- PCB
-
-
-

-
-
-

UNIX/Linux sleep( ) pause(
) wait( ) kill( ) Windows 2000/XP
SuspendThread ResumeThread

(suspend count) 1 1
0 1 1 0

3

-
-
-
-
-

-
-
-
-
-
-
-
-
-

- PCB
-
-
- PCB PCB

4

PCB

PCB

2.3.6 UNIX SVR4

UNIX SVR4

UNIX SVR4

1 UNIX SVR4

2-22 UNIX SVR4

kernel running user running

asleep in memory sleeping swapped

(preempted) (ready to run in memory) (ready to run swapped)

preempted ready to run in memory

2-19

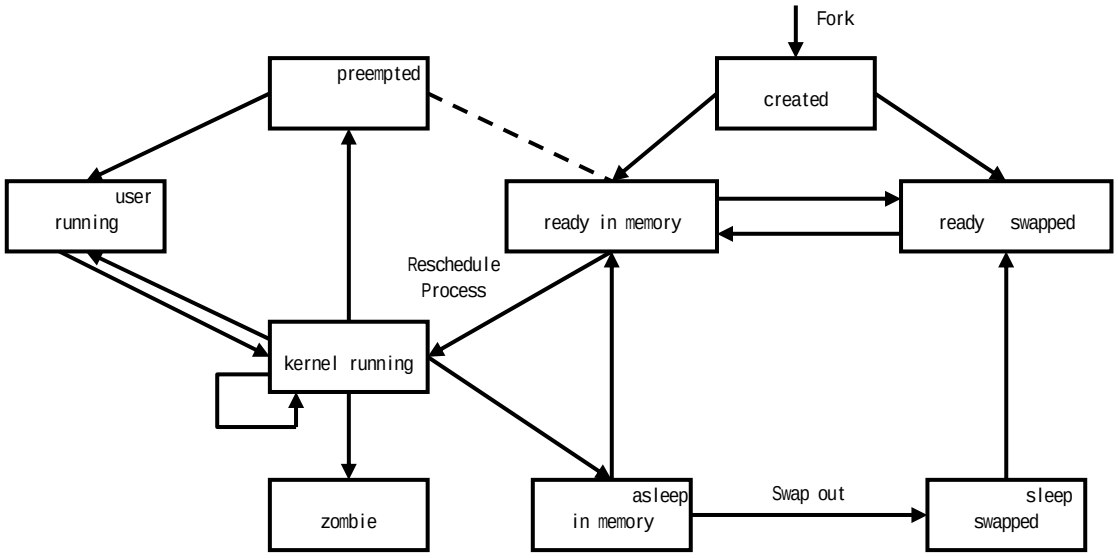
preempted ready to run in memory kernel running

user running

preempted

- user running
- kernel running
- preempted

- ready to run in memory
- asleep in memory
- ready to run swapped
- sleeping swapped
- created
- zombie



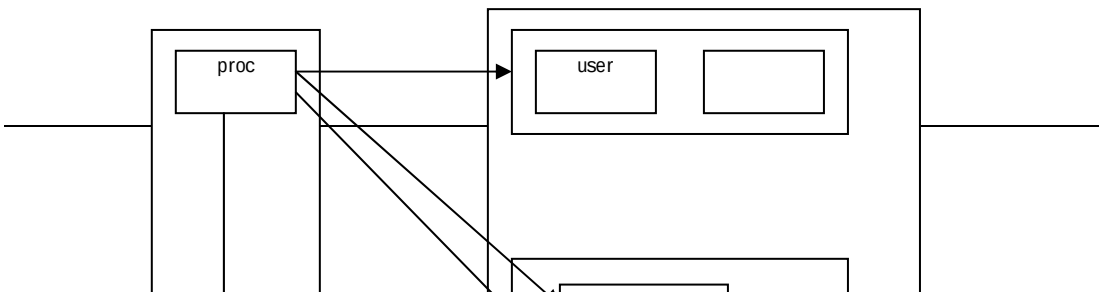
2-22 Unix SVR4

UNIX 0 swap
1 init 0 1
1

2 UNIX SVR4

UNIX SVR4 2-23 proc
UNIX PCB
proc user proc
user

user
/
proc proc
proc proc
proc proc



proc

user text/data/stack

/ /

proc

user proc

/ /

I/O

user

/ user

PCB

proc

1024

user

text

UNIX

text

text proc
text UNIX
proc user text

PPRT Per Process Region Table

/

3 UNIX SVR4

UNIX SVR4 fork( ) pid=fork( )

-
-
-
-
-
- 0

- fork( )
  Ready to Run
- fork( )
  Ready to Run
- ready to run
  fork( )

fork( )

? fork( ) pid 0
pid 0
pid

2.3.7 Linux

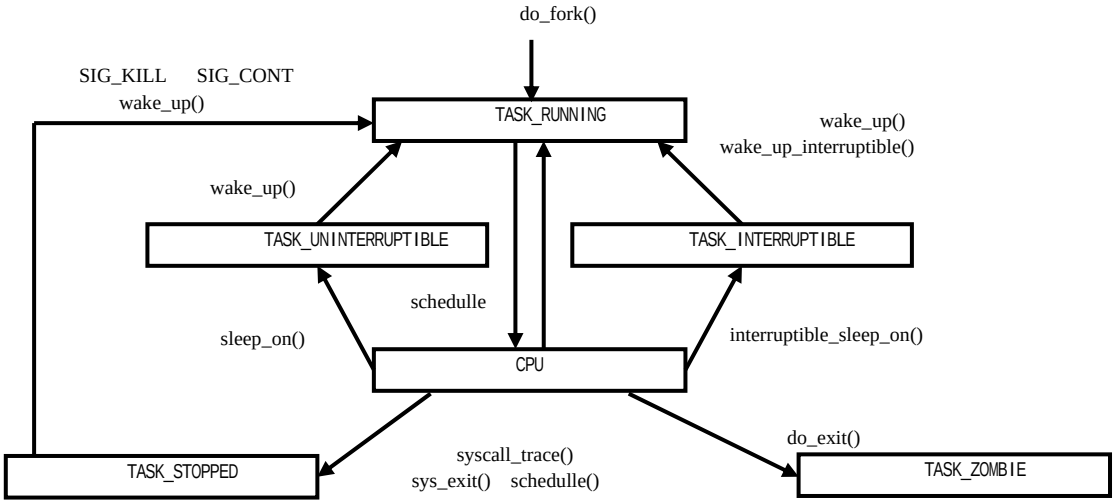
Linux

1

Linux
Linux 2.2.x SMP
2-24 Linux 6

- TASK_RUNNING CPU CPU
  current
- TASK_INTERRUPTIBLE
  SIGNAL
- TASK_UNINTERRUPTIBLE
  SIGNAL

- TASK_ZOMBIE
- TASK_STOPPED
- TASK_SWAPPING



2-24 Linux

sys_fork( ) sys_clone do_fork( )
do_fork( ) task_struct

pid
sys_clone( )

count 1
sys_exit( ) do_exit( ) do_exit( )
TASK_ZOMBIE exit_notify( )
schedule( )

2 Linux

Linux
process table NR-TASKS task_struct
/include/linux/sched.h Linux 2.2.10 512 x86 NR-TASKS
4096 Linux task-struct
PCB task-struct

task-struct

	state	(6)
	flags	(10)
	priority	
	rt_priority	
	counter	
	policy	0 1 2

[redacted]

32

CB

Linux

suid,sgid

uid/gid

uid/gid

	nswap	
	cmin_flt,cmaj_flt	
	cnswap	
	swap_cnt	
SMP	processor	SMP CPU
	last_processor	CPU
	lock_depth	
	used_math	FPU
	Comm[16]	
	rlim	
	errno	0
	Debugreg[8]	
	*exec_domain personality	iBCS2
	*binfmt	a.out,script,elf,java 4
	exit_code,exit_signal	
	dumpable	memory dump
	did_exec	POSIX
	tty_old_pgrp	
	*tty	
	*wait_chldexit	

Linux

(1)current SMP CPU CPU

(2)init-task 0 PCB

(3)*task[NR-TASKS] PCB (pid) task[0] 0 init-task tasks[] PCB for-each-task() next-task PCB

(4)jiffies Linux 0 10ms do-timer() 1

(5)need-resched 1 schedule() CPU

(6)intr-count

2.4

2.4.1

process

-
-
-
-
- / C/S

multiple threaded process

“
” “ ”

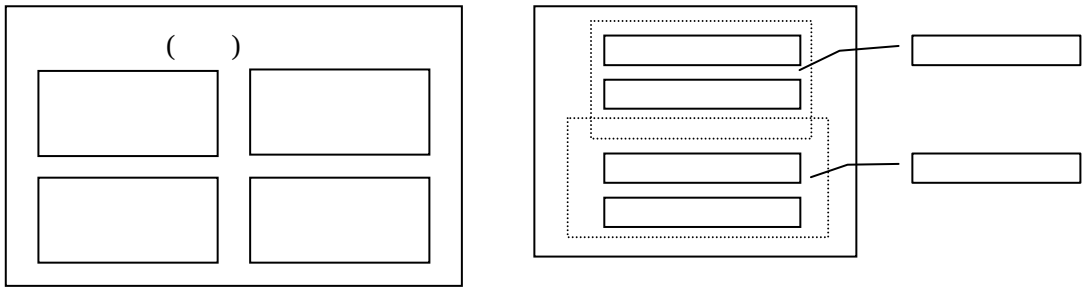
MS-DOS
UNIX
Solaris Mach SVR4 OS/390 OS/2 WindowNT Chorus JAVA
Solaris thread OS/2 thread Windows NT thread IEEE UNIX POSIX 1003.4a

2.4.2

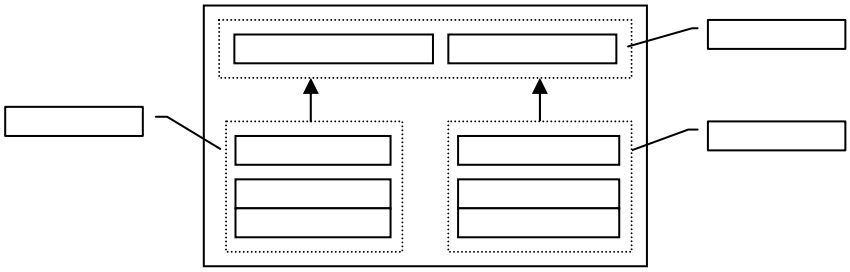
1

2-25

/



2-25



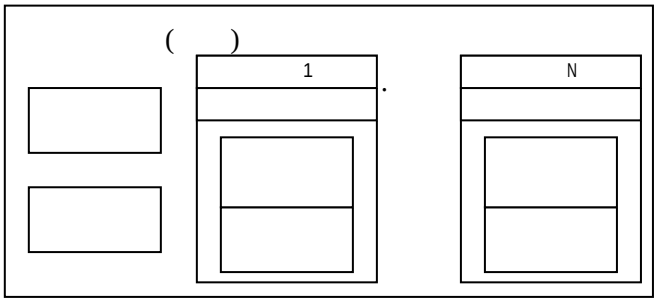
2-26

2-26

2-27

PCB

TCB Thread Control Block



2-27

-
- I/O
-

2

- …
-
-
-
- (1) /

- (2)

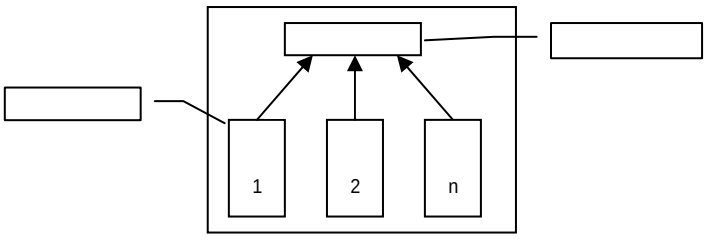
(3)

(4)

2-28

I/O CPU

LWP(Light-Weight Process)



2-28

“ ”

“ ” “ ” ( )

( )

( ) ( ) ( )

3

I/O

?

?

Windows

4 ( )

( )

( )

( ) ( )

( )

( ) (

) API

- spawn
- block
- unblock
- finish

TCB

( )

( ) U)

( )

( ) ( )

API

- -

( ) ULT(User Level
Thread) POSIX P-threads Java KLT(Kernel Level Thread)
Windows2000/XP OS/2 Mach C-thread Solaris
ULT KLT ( )

Intel 4

cache

- **5**

-

-

Mach
UNIX
10

-

-
- UNIX 50
-
-

6

/

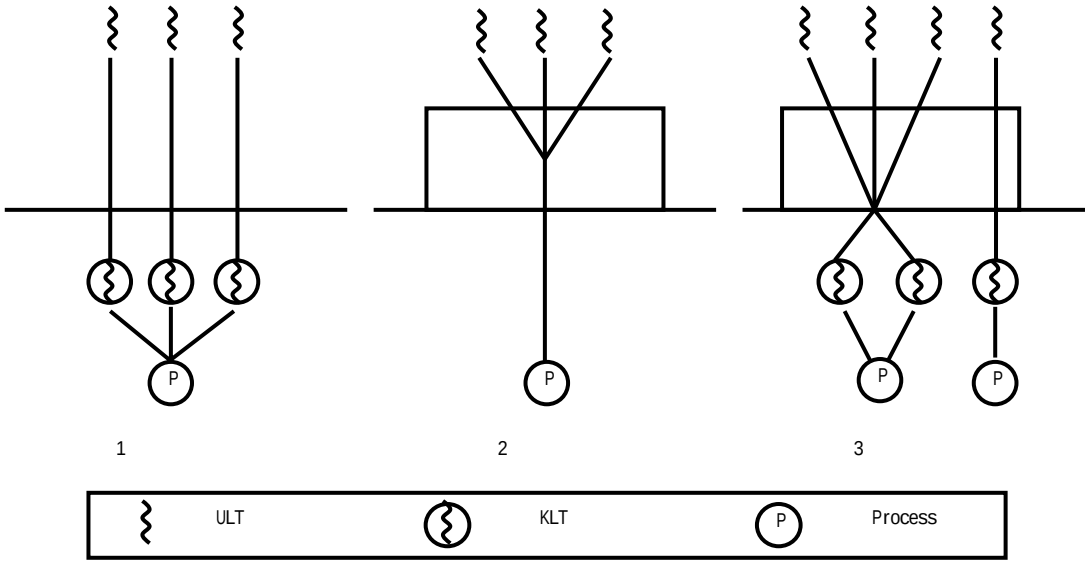
-
- C/S
  CPU
  CPU
-
-

-

GUI
GUI

2.4.3

ULT POSIX P-threads Java
KLT Windows2000/XP OS/2 Mach C-thread
Solaris
2-29



2-29

1

KLT Kernel Level Threads

KLT API

Windows 2000/XP OS/2

PCB

TCB

KLT

KLT

- -

2

ULT User Level Threads

ULT

“ ”

“ ”

TCB

…

B 2

- B 2 B
  I/O I/O B
  B
  2 2
  B
  B
- B
  B 2
  B 2 B
- 2 B 1 2
  1
  B
  2

ULT KLT

-
-

ULT ULT

UNIX UNIX
KLT ULT

-
  KLT
- ULT
  CPU

jacketing

jacketing I/O

jaketing I/O

3

ULT/KLT Solaris

KLT

ULT ULT KLT

KLT

2.4.4 Solaris

1 Solaris

Solaris 4 SMP

- Process UNIX PCB
- User-Level Threads

  ULT
- Light Weight Process LWP ULT KLT

  LWP ULT KLT LWP
- Kernel-Level Threads KLT

Solaris

LWP KLT LWP KLT

KLT LWP ULT LWP ULT KLT

ULT LWP

ULT

LWP(

LWP) LWP ULT ULT

LWP ULT LWP LWP

LWP LWP

KLT LWP ULT KLT ULT LWP

LWP ULT ULT

LWP ULT

KLT

LWP ULT LWP

ULT KLT

ULT LWP LWP

ULT LWP 5

ULT 5 LWP LWP

KLT KLT

4 LWP 4 ULT KLT

ULT

KLT LWP

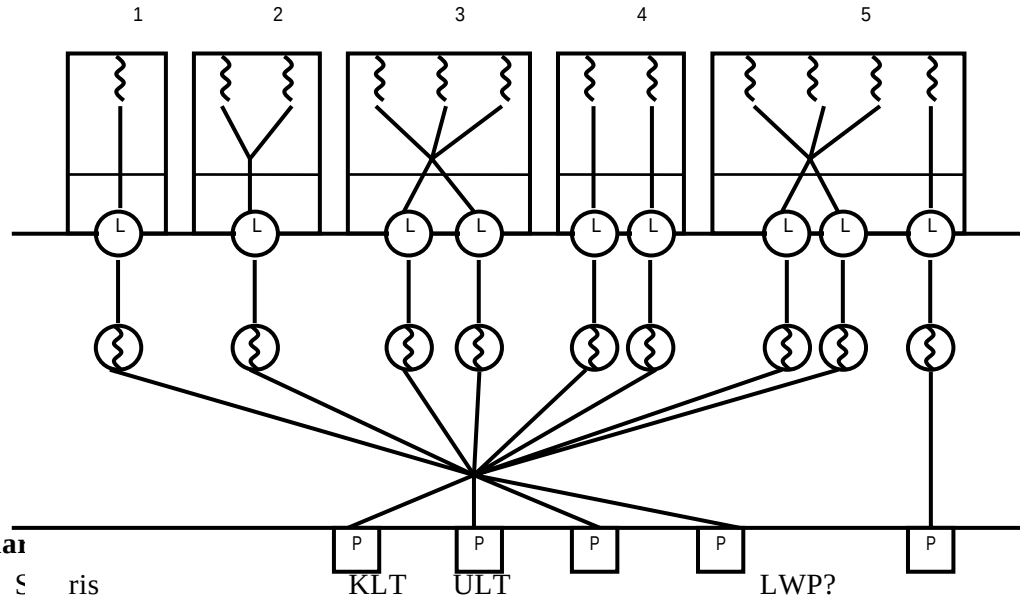
LWP LWP
LWP LWP ULT

LWP LWP
LWP

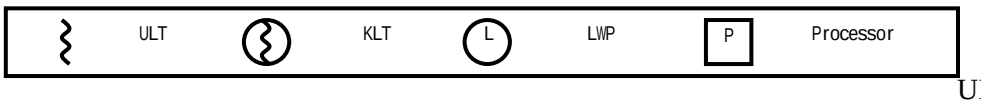
2-30 4 ULT LWP KLT LWP KLT
LWP LWP
LWP KLT
KLT
1
2 ULT ULT

3 LWP Solaris
ULT ULT LWP
4 LWP

5 ULT LWP ULT LWP
LWP



2 Solar
S ris KLT ULT LWP?



ULT

2-30 Solaris Г
LWP ULT
ULT LWP
ULT
CPU LWP LWP ULT CPU

ULT ULT
LWP

LWP
LWP

LWP
Solaris 2-31

( )
( ) Solaris
LWP UNIX
LWP
LWP LWP
( ) /
/

LWP KLT
LWP LWP

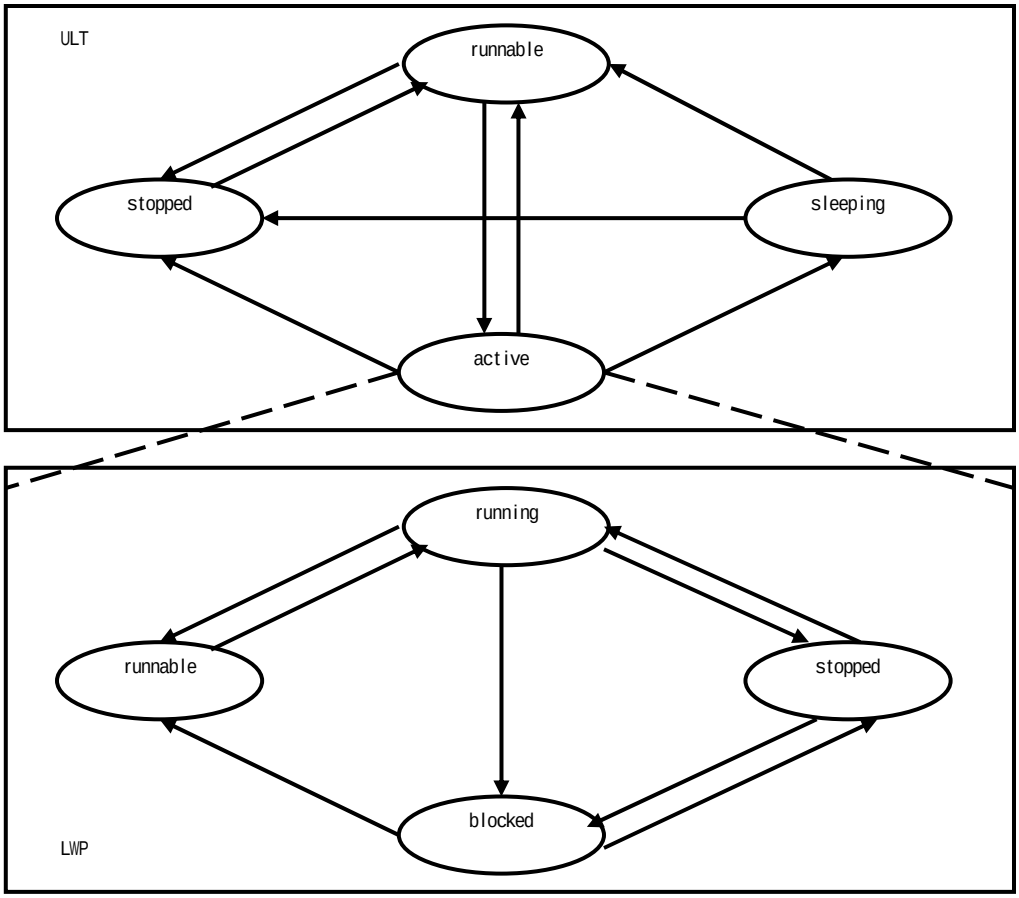
3 Solaris

2-32 ULT LWP
ULT LWP
ULT
LWP KLT ULT
ULT

- KLT א ULT

LWP

LWP



2-32 ULT LWP

2-32 LWP ULT
ULT LWP LWP
ULT
ULT LWP ULT
LWP
ULT LWP ULT LWP
ULT LWP
LWP
Solaris

thread_create( )	
thread_setconcurrency( )	LWP

thread _exit( )	
thread_wait( )	
thread_get_id( )	
thread_sigsetmask( ) thread_sigprocmask( )	
Thread_kill( )	
Thread_stop( )	
Thread_priority( )	
mutex_enter( ) mutex_exit( ) mutex_tryenter( )	
Sem_p( ) sema_v( ) sema_tryp( )	P V P
rw_enter( ) rw_exit( ) rw_tryenter( ) rw_downgrade( ) rw_tryupgrade( )	/ / / / / write lock read lock read lock write lock
cv_wait( ) cv_signal( ) cv_broadcast( )	 cv_wait( ) cv_wait( )

2.4.5 Windows 2000/XP

1 Windows 2000/XP

Windows2000/XP

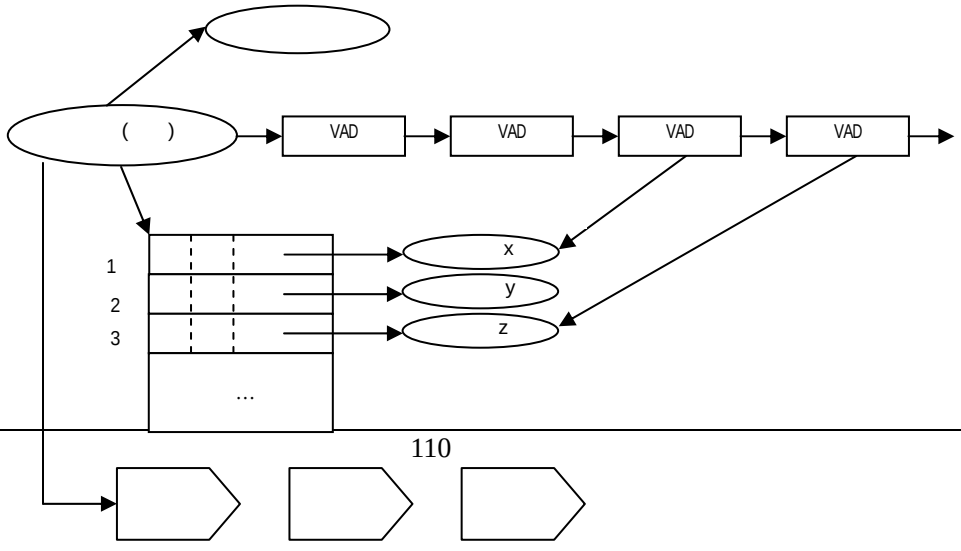
Windows2000 Windows NT4

CPU

Windows 2000/XP

SMP C/S CPU

-
-
-



2-33
Windows2000/XP access token

2-33

2

(object oriented)

( ) ( ) ( )

(object class)

(instance)

(encapsulati on)

(subclass) (superclass)

Windows 2000/XP (object-based)
Windows 2000/XP ?
Windows 2000/XP

(1)

(2)

Windows2000/XP

/

/

Window2000/XP

EPROCESS

(2-33)

3

EPROCESS EPROCESS

ETHREAD PEB

Process Environment Block

EPROCESS WIN32

CSRSS WIN32 WIN32

WIN32K.SYS WIN32 USER GDI

EPROCESS

- KRPROCESS
-
- CPU
- VAD
-
-
- /
-
-
- PEB
  GDI
- WIN32 WIN32
  WIN32
- CreateProcess
- CreateProcessAsUser
  EXE
- OpenProcess
- ExitProcess
- TerminateProcess
- FlushInstructionCache
- GetProcessTimes
- GetExitCodeProcess
- GetCommandLine
- GetCurrentProcessID ID
- GetProcessVersion Windows
- GetStartupInfo CreateProcess STARTUPINFO
- GetEnvironmentStrings
- GetEnvironmentVariable
- GetProcessShutdownParameters
- SetProcessShutdownParameters
  CreateProcess WIN32 WIN32

3 WIN32
KERNEL32.DLL Windows2000/XP WIN32 CSRSS
- .EXE
- Windows2000/XP
-

- GetExitCodeThread
- GetThreadTimes
- GetThreadSelectorEntry
- GetThreadContext CPU
- SetThreadContext CPU

CreateThread WIN32

-
-
- NtCreateThread

ID

TEB

WIN32 KeInitializeThread KTHREAD

- WIN32
- ID
- CREATE_SUSPEND

Windows2000/XP 2-35

7

- —— CPU
- ——
- ——
- —— 1

I/O 2 3

- ——

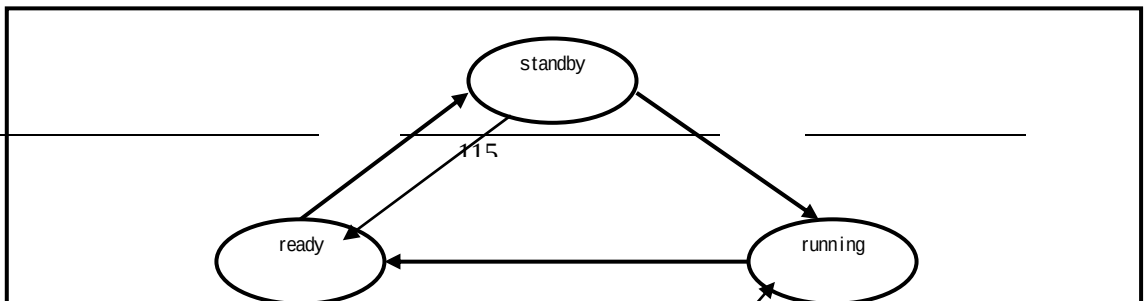
( )

- ——
- ——

5

Windows2000/XP “ ”

Windows NT4



-
- CPU
- CPU
-
-
-

ID SID

WIN32
SystemParameterInfo

WIN32

- CreateJobObject
- Open JobObject
- AssignProcessToJobObject
- TerminateJobObject
- SetInformationToJobObject
- QueryInformationToJobObject CPU
  ID

2.5

2.5.1

2-36

2-36

CPU

CPU

2-38

1 2

3

2.5.3

Medium Level Scheduling

Medium-term Scheduling

“ ”

2.5.4

Low Level Scheduling

Short_term Scheduling

CPU (dispatcher)

2.5.5

scheduler

scheduling algorithm

- —— CPU
  CPU =CPU /CPU CPU
  CPU +CPU
- —— ( )
- ——
- ——
- —— CPU

i t_s

t_f t_i

$t_i = t_f - t_s$

$T = \Sigma t_i / n$

i t_i t_k $w_i = t_i / t_k$

t_i 1

$W = \Sigma w_i / n$

2.6

2.6.1

JOB

Job Step “ ” “ ” “ ”

“ ” “
” “ ” CPU

UNIX

2.6.2

JCB Job Control Block JCB
Spooling
JCB JCB
CPU
/ /
/
(CPU I/O)

-
-
-
-

\ 9

2.6.3

T_i

T_i

T= $T_1+T_2+\ldots+T_n$ /n CPU

2.6.4

1

FCFS First Come First Served

FCFS

CPU

	CPU
1	9
2	4
3	10
4	8

SJF

2 4 1 3

T = 4+12+21+31 /4 = 17

W = 4/4+12/8+21/9+31/10 /4 = 1.98

FCFS

T = 9+13+23+31 /4 = 19

W = 9/9+13/4+23/10+31/8 /4 = 2.51

SJF FCFS FCFS

SJF

SJF

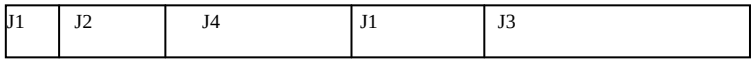
CPU

CPU

SRTF Shortest Remaining Time First

CPU

		CPU
1	0	8
2	1	4
3	2	9
4	3	5



0 1 5 10 17 26

Job1 0 Job2 1 Job1

7 JOB2 4 Job1 Job2

SRTF 10-1 + 1-1 + 17-2 + 5-3

/4=26/4=6.5 SJF 7.75

3 HRRF

Highest Response Ratio First

SJF

/ =1+ /

HRRF

HRRF

		CPU
1	0	20
2	5	15
3	10	5
4	15	10

SJF

1 3 4 2

T= 20+25+35+50 /4=32.5

W= 20/20+25/5+35/10+50/15 /4=3.2

FCFS 1 2 3 4

T= 20+35+40+50 /4=36.25

W= 20/20+35/15+40/5+50/10 /4=4.1

HRRF

- 1 1 20
- 1 1+15/15 1+10/5 1+5/10 3 5
- 3 1+20/15 1+10/10 2 15
- 2 4 10

T= 20+25+40+50 /4=33.75

W= 20/20+25/5+40/15+50/10 /4=3.4

HRRF SJF FCFS

4

I/O

5

6

2.7

2.7.1

dispatcher

-
-
-

-

2.7.2

1

I/O

2

CPU

100ms

—

10

100ms

100ms

300ms 3

3

?

CPU

CPU

CPU

CPU

UNIX

p-pri=min {127, (p-cpu/16+PUSER+p-nice)}

p-pri

-100~+127 p-nice

p-nice PUSER 100 p-cpu

p-pri 127 (p-cpu/16+PUSER+p-nice)

p-cpu 20ms

p-cpu 1 1

p-cpu p-cpu<10

200ms p-cpu 0 p-cpu>10

200ms p-cpu 10

(1) p-cpu

p-pri

(2)

200ms p-cpu 0 p-pri

UNIX 100

UNIX

p-pri=(p-cpu/2)+

4

2-40

(

)

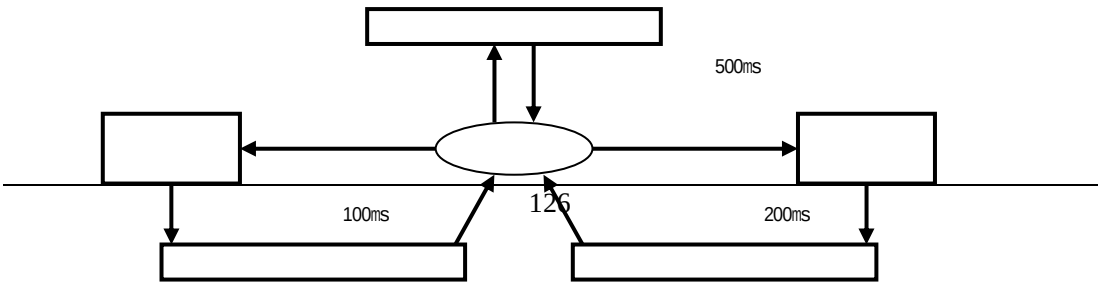
( )

XDS940

I/O

( )

I/O



5

n
CPU 1/n n
CPU 1/n
CPU
CPU n
CPU CPU
CPU
6

CPU
CPU
50 20ms

100
20 20% 20% CPU

40

10 20
25 / 10 20 25
CPU

2.7.3

1

CD-ROM
VCD

hard real time **soft real time**

m i P_i C_i CPU

$$C_1/P_1+C_2/P_2+\ldots+C_m/P_m \leq 1$$

schedulable

100ms 200ms
500ms 50ms 30ms 100ms

$$0.5 + 0.15 + 0.2 \leq 1$$

1 4 150ms

2

1

20ms 50 100ms
10

2

3

laxity
= -(+) ,

2.7.4

- cluster

I/O

-

1

master/slave

1

2 peer-to-peer

3

1

load sharing

-
-
-

Leutenegger S Vernon M 1

2

3

-

-
-

Mach

2

gang scheduling

-
-

N M

N N

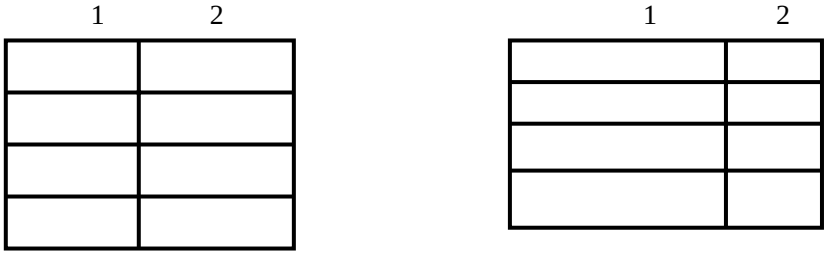
1/M 2-41

4 1 50%

CPU CPU

37.5%

4/5CPU 1/5 15%



37.5% 2-41

15%

3

dedicated processor assignment

4

dynamic scheduling

-
-
-

2.7.5 UNIX SVR4

UNIX SVR4 UNIX

( ) UNIX SVR4 3

160 UNIX

safe place

safe place

UNIX SVR4

- 159-100
- 99-60
- 59-0

UNIX SVR4 2-42

dispq

dqactmap (1)

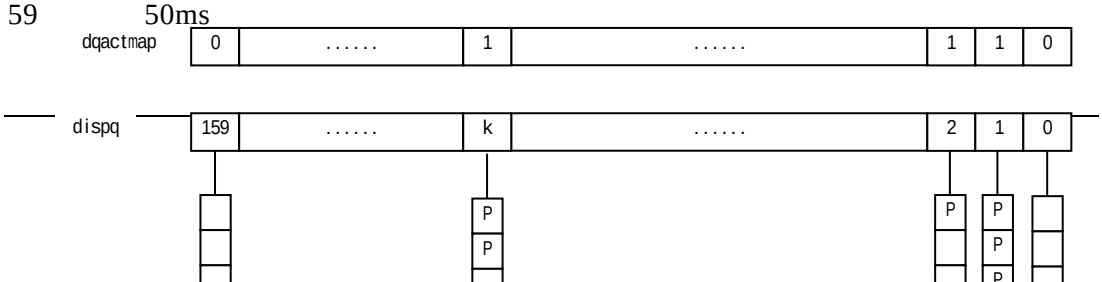
dqactmap

( )

kprunrun

0 100ms

59 50ms



2.7.6 Windows 2000/XP

1 Windows 2000/XP

Windows 2000/XP Windows 2000/XP

server

Window4 2000/XP

Windows 2000/XP

A 10 B 2

12 1/12

2 Windows2000/XP

Win32 API Win32 API

Win32 API

API	
Suspend/ResumeThread	
Get/SetPriorityClass	
Get/SetThreadPriority	
Get/SetProcessAffinityMask	
SetThreadAffinityMask	
Get/SetThreadPriorityBoost	
SetThreadIdealProcessor	
Get/SetProcessPriorityBoost	
SwitchToThread	
Sleep	0
SleepEx	I/O APC

3

Windows 2000/XP Windows 2000/XP

2-43 Windows 2000/XP 32 0 31

- 31-16
- 15-1

15

- 0

Win32

Windows 2000/XP Win32

Win32

Task Manager Win32 SetPriorityClass

quantum unit

1

Windows 2000 6 Windows 2000/XP 36

3

Windows 2000 3 2 Windows 2000/XP 12

x86 10 x86 15

2)

( ) Window2000/XP ?

Windows 2000

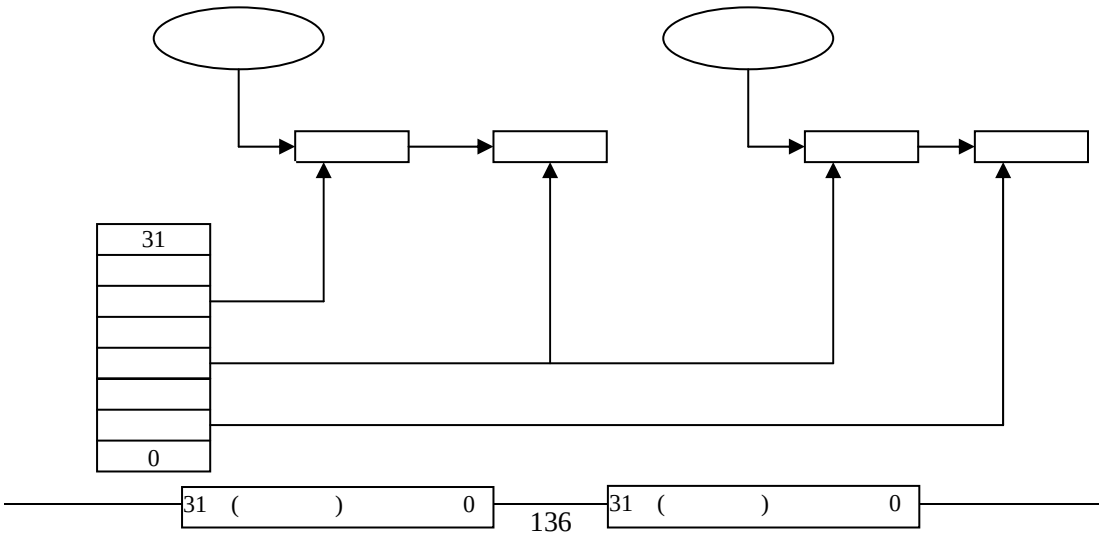
Windows 2000

Windows 2000 Windows 2000 (terminal services),

5

2-44 “ ”

KiDispatcherReadyListHead ,



Windows 2000/XP

KiReadySummary 32

0 0 1 1

KiIdleSummary 32

DPC/

KiDispatcherLock ,

6

Windows 2000/XP

Windows 2000/XP

≈

Ï*ıdVÄ5-1.4(

4)

ExitThread TerminateThread

7

5 Windows 2000/XP

1 I/O
2
3
4
5

16 31

I/O I/O

I/O

I/O

IoCompleteRequest

I/O

1 2 6

8

I/O

CPU

KiUnwaitThread

PsPrioritySeparation

2 KeSetEvent

Windows2000/XP (balance set manager)

300 (300 3 4) 15

2

300

16

10 10

CPU

8

Windows 2000/XP

1)

SetProcessAffinityMask SetThreadAffinityMask

2)

-
-

SetThreadIdealProcessor

3)

Windows 2000/XP

Windows 2000/XP

Windows 2000/XP

4)

Windows 2000/XP

-
-
- 2
- 24

5)

Windows 2000/XP

Windows 2000/XP

0 8
1 4
0 6
6 0 8 Windows 2000/XP
6 0 8 0
1 0 8 1
4

9

Windows 2000/XP

0

-
- DPC DPC
-
-

2.7.7 Linux

Linux schedule( ) CPU
CPU CPU
CPU
CPU
task_struct
schedule( )
1 policy policy

- #define SCHED_OTHER 0

- #define SCHED_FIFO 1 POSIX 1b FIFO

rt_priority CPU

- #define SCHED_RR 2 POSIX 1b RR

SCHED_FIFO

rt_priority

2 priority -20 20 1 40

CPU

CPU

3 rt_priority 0 99

rt_priority+1000

(

999 0 56)

4 counter

CPU counter 0

goodness( )

Linux

sched_setscheduler( )

sys_setpriority( )

Linux

CPU 0

Linux

task_struct counter

CPU

counter

10ms 60

600ms

counter priority

priority rt_priority policy

conuter priority

counter 1

- counter 0

0

0

0 0 (

0 0

) conuter

- 0

0 (priority)
0
(priority+counter/2)

I/O
CPU

0
Linux Linux
- CPU
-
- counter 0

Schedule( )
1
Schedule( ) bottom half

- TASK_RUNNING
- TASK_INTERRUPTIBLE
- counter=0

TASK_RUNNING
- TASK_RUNNING
- TASK_UNINTERRUPTIBLE

2
CPU

- goodness

counter 0 999 0 56
counter+1000 1001 1099

- counter
- counter

3
CPU task_struct

4 SMP
Linux2.0.0 SMP CPU
CPU SMP

task_struct CPU CPU
last_processor CPU
processor_mask CPU N
1 N CPU CPU (
) CPU
CPU cache CPU

CPU CPU
CPU CPU ( )
CPU

2.8

PSW

Window2000/XP Solaris Linux

—

—

( )

UNIX SVR4 Linux

) CPU

(ULT KLT)

/

Solaris Linux Windows2000/XP

FCFS SJF SRTF HRRF

FCFS

( ) (

)

1. PSW? ?
2. ?
3. CPU
4. CPU CPU
5. ?
6. ? ?
7. ?
8. ?
9. ?
10.
11. ? ?
12.
13. ?
14.
15.
16. ?

17.
18.
19. Windows 2000/XP
20. Windows 2000/XP ?
21. Linux
22. Linux
23. Linux
24. ? ?
25. ?
26. ? ?
27. ?
28. ?
29. 1) 2)

30. ?
31.
32.
33. PCB ? ?
34. ?
35.
36. ?
37. ?
38. ?
39. ?
40.
41.
42.
43. UNIX SVR4
44. UNIX PCB proc user ?
45.
46. ?
47.
48.
49. (TCB)? ?
50.
51. ULT KLT
52.
    a)
    b)
    c)
53.
54. ? ?

55.
56. ?
57. Solaris
58. Solaris ? ?
59. Solaris
60. Solaris
61. Solaris
62. Solaris
63. Windows 2000/XP
64. Windows 2000/XP
65. Windows 2000/XP
66. Windows 2000/XP
67. Windows 2000/XP
68.
69.
70.
71.
72.
73.
74. 1 2 (3) (4)
75.
76. JCB
77.
78.
79.
80.
81. ?
82.
83.
84.
85.
86. “ ”
87.
88. CPU
89. UNIX
90. UNIX SVR4
91. Windows 2000/XP
92.
93.
?
94.
95.

96. (scheduling mech anism)
(scheduling policy)
97. ( )

98.
0 /
99.
“ 0 ”
100.
101.

1 ?
(1) 2 3 4 PSW 5
(6) (7) I/O
2 “ ”
“ I/O ” “ ”
3 ?
(1) (2) (3) (4)
4
?
5 J1 J2 J3
a b c a<b<c

6 J_1 … J_n S_1 … S_n
CPU

7 0 1 2 3 4
5
1
( )
2

1	10	3
2	1	1
3	2	3
4	1	4
5	5	2

8 I/O T

S Q

CPU

1 Q ∞ 2 Q T 3 S Q T 4 Q S 5 Q 0

9 5 9 6 3 5 x

10 5 A E 2 4 6 8 10

1 2 3 4 5 5 1

2 3 4 (

C D B E A)

1 2 2 4

11 5 A E 10 6 2 4 8

3 5 2 1 4 5

1 FCFS(A B C D E) (2)

(3)

12

? ?

13

CPU ?

14

15

1	10 00	2 00				
2	10 10	1 00				
3	10 25	0 25				
W=						

16 FCFS SJF HRRF

(1)

(2)

20 100K 2 1
I/O

1	8:00	25	15K	1	1
2	8:20	10	30K	0	1
3	8:20	20	60K	1	0
4	8:30	20	20K	1	0
5	8:35	15	10K	1	1

FCFS
CPU (1) ?(2)
?(3) ?(4) ?

21 200K 5
I/O

A	8:30	40	30K	3
B	8:50	25	120K	1
C	9:00	35	100K	2
D	9:05	20	20K	3
E	9:10	10	60K	1

(1)FIFO ?(4) SJF
?

22 (1)FIFO
?(4) SJF ?

23 (1)
(2) (3) P1 20
40 P2 25 P3 35 35
P4 60

24 4 50 100 300 250ms
35 20 10 xms x

CH3

3.1

3.1.1

( )

Sequential Programming

-
-
-
-

( )

3.1.2

Concurrency

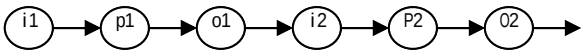
A B a1 a2 a3 b1 b2 b3
A B al a2 a3 bl b2 b3

al bl a2 b2 a3 b3 a1 bl a2 b2 b3 a3 A B

?

3-1 while(TRUE){input process output}

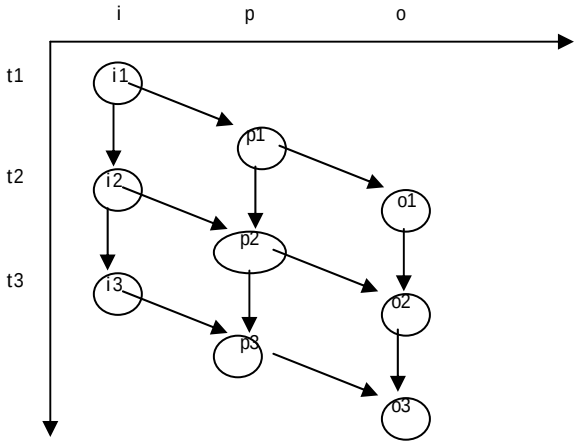
- -



3-1

while(TRUE) {input send}
while(TRUE) {receive process send}
while(TRUE) {receive output}
( ) send receive

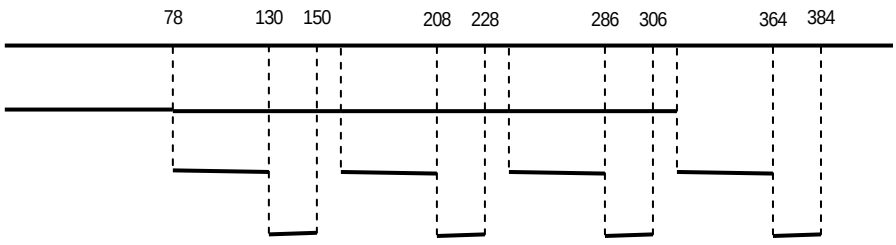
1 1
2 1 2
3 2



3-2

3-2 t3 i3 p2
o1 t4 t5 3-3
n

52 * n / 78 * n+52+20



3-3

n 67%

concurrent programming

1966

Bernstein Bernstein

$R(p_i)=\{a_1 \quad a_2 \quad \dots \quad a_n\}$ p_i

$W(p_i)=\{b_1 \quad b_2 \quad \dots \quad b_m\}$ p_i

p_1 p_2 Bernstein

$$R(p_1)\&W(p_2) \cup R(p_2)\&W(p_1) \cup W(p_1)\&W(p_2)=\{\ \}$$

S_1: a := x + y
S_2: b := z + 1
S_3: c := a - b
S_4: w := c + 1

$R(S_1)=\{x \ y\}$ $R(S_2)=\{z\}$ $R(S_3)=\{a \ b\}$ $R(S_4)=\{c\}$ $W(S_1)=\{a\}$ $W(S_2)=\{b\}$ $W(S_3)=\{c\}$ $W(S_4)=\{w\}$ S_1 S_2 **Bernstein**

-
-
-
- ( )

(1)

I/O I/O I/O (2)

(3)

( )

I/O

3.1.3

1

Tl T2

Aj(j=l 2 ...) xl x2

T1 T2

```
process Ti(i=1,2)
var Xi:integer;
begin
        {                  Aj};
        Xi:=Aj;
        if Xi>=1 then begin Xi:=Xi-1; Aj:=Xi;{          };end
            else {        “          ”};
    end;
```

T1 T2

T1: *X1 := Aj;* X1 = m (m>0)

T2: *X2 := Aj;* X2 (i= it ‘

coend

borrow return x
borrow B x
B>x { } return
return
{ } borrow

3.1.4 Interaction Among Processes

deadlock
starvation

( )

FCFS
“ ”

mutual exclusion ( )

input process output

Synchronization ( )

3.2

3.2.1

l

Aj **“ ” critical section “ ” critical resource**

T1

```
X1 := Aj;
if X1>=1 then begin X1:=X1-1; Aj:=X1;
```

T2

```
X2 := Aj;
if X2>=1 then begin X2:=X2-1; Aj:=X2;
```

Dijkstra 1965

shared

```
shared variable
region variable do statement
```

```
shared Aj
process Ti ( i = 1, 2 )
  var Xi:integer;
  begin
                  Aj;
   region Aj do begin
        Xi := Aj;
        if Xi>=1 then begin Xi:=Xi-1; Aj:=Xi;{          };end
              else {          };
    end;
```

```
end;
```

-
-
-

-
-
-

region x do begin … region y do … end;

region y do begin … region x do … end;

3.2.2

P1 P2 insidel inside2 true false P1(P2) inside2(insidel) insidel(inside2) true P2(P1) P1(P2) false

```
inside1,inside2:Boolean;
inside1 := false;    /* P1              */
inside2 := false;    /* P2              */
cobegin
process P1
begin
   while inside2 do begin  end;
   inside1:= true;
         ;
   inside1 := false;
end;
process P2
begin
   while inside1 do begin  end;
   inside2 = true;
         ;
   inside2 := false;
end;
coend.
```

P1(P2) inside2(insidel) insidel(inside2) P2(P1) insidel(inside2) false

inside2(insidel) true P1(P2)
P1(P2) inside2(insidel)
insidel(inside2) true P2(P1)
true

```
inside1,inside2:boolean;
inside1 := false;    /* P1              */
inside2 := false;    /* P2              */
cobegin
process P1
begin
    inside1 := true;
    while inside2 do begin  end;
          ;
    inside1 := false;
end;
process P2
begin
    inside2 := true;
    while inside1 do begin  end;
          ;
    inside2 := false;
end;
coend.
```

3.2.3

1 Dekker

T.Dekker
turn turn=1
P1 turn=2 P2 Dekker

```
var inside : array[1..2] of Boolean;
    Turn :integer;
    turn := 1 or 2;
    inside[1]:=false;
    inside[2]:=false;
cobegin
process P1
begin
    inside[1]:=true;
   while inside[2] do
                  if turn=2 then  begin
                  inside[1]:=false;
                  while turn=2 do begin  end;
                  inside[1]:=true;
                 end
          ;
    turn = 2
    inside[1]:=false;
end;
process P2
begin
     inside[2]:=true;
     while inside[1] do
                    if turn=1 then  begin
                    inside[2]:=false;
                    while turn=1 do begin  end;
                    inside[2]:=true;
                   end
          ;
    turn = 1;
    inside[2]:=false;
end;
coend.
```

Dekker P1(P2)
inside1 inside2 true
false P1 P2
turn turn 1 2 P1 P2
P2 P1 inside2(inside1) P2 P1
false P1 P2
P1 P2 false turn
2 1 P2 P1
turn= i(i = 1 2)
Pi (i = l 2)
turn
turn 1 2 while
Dekker peterson

2 Peterson

1981 G.L.Perterson

false

turn turn=i

Pi Perterson

```
var inside:array[1..2] of boolean;
    turn:integer;
    turn := 1 or 2;
    inside[1] := false;    /* P1              */
    inside[2] := false;    /* P2              */
cobegin
process P1
begin
    inside[1]:= true;
    turn := 2;
    while (inside[2] and turn=2)
         do begin  end;
          ;
    inside[1] := false;
end;
process P2
begin
    inside[2] := true;
    turn := 1;
    while (inside[1] and turn=1)
        do begin  end;
          ;
    inside[2] := false;
end;
coend.
```

turn while

inside[i]

Peterson while

“ inside[i] turn=1(2) ”

n

3.2.4

1

CPU

2

“ ” TS(Test and Set x TS(x) x true x false x TS

TS(x): x=true x:=false return true return false

TS s s

s true

TS

TS s

TS s true s false

s true TS

x

```
s : boolean;
s := true;
process Pi                          /* i = 1,2,...,n*/
   pi : boolean;
begin
   repeat pi := TS(s) until pi; /*    */
         ;
   s := true;                   /*    */
end;
```

3

Swap

Swap (a,b): temp:=a; a:=b; b:=temp;

Intel 80x86 XCHG

lock false

```
lock : boolean;
lock := false;
process Pi                                  /* i = 1,2,...,n*/
   pi : boolean;
begin
    pi := true;
```

```
        repeat Swap(lock, pi) until pi = false;/*    */
              ;
        lock := false;                          /*    */
    end;
```

3.3 PV

3.3.1

-

producer-consumer problem

--

-- n m k

pi cj

pi

cj -

```
var   k:integer;
      type item:any;
      buffer:array[0..k-1] of item;
      in,out:integer:=0;
      counter:integer:=0;
process producer
      while (TRUE)                          /*          */
      produce an item in nextp;             /*              */
      if (counter==k)   sleep( );           /*                  */
      buffer[in]:=nextp;                    /*                  */
      in:=(in+1) mod k;                     /*          */
      counter:=counter+1;                   /*              1*/
      if (counter==1)   wakeup( consumer);  /*
                                                  */
process consumer
      while (TRUE)                          /*          */
      if (counter==0) sleep ( );            /*                  */
      nextc:=buffer[out];                   /*              nextc*/
      out:=(out+1) mod k;                   /*          */
      counter:=counter-1;                   /*                  1*/
      if (counter==k-1) wakeup( producer);  /*
                                                  */
      consume thr item in nextc;            /*          */
```

sleep( ) wakeup( )

counter counter

counter counter

8 counter 1
counter 1 1
counter 9 7 8

counter 0
counter 1 counter 1 counter
0 wakeup
counter
0

pi cj

PV

3.3.2 PV

busy waiting CPU

1965 E.W.Dijkstra -- P V

semaphore P
V

Ń Ń ⍭ ⍯ 9

- 0 1
-

1

s P V P V

- P(s) s 0 s l P(s) s 0
- V(s) s 1

P(s) V(s)

P(s) while s ≤ 0 do null operation
s:=s-1

V(s) s:=s+1

P s<=0 “ ” s

2

s value queue value queue P V

- P(s) s l 0 P(s) s
- V(s) s 1 0 s P V

```
type  semaphore=record
      value:integer;
      queue: list of process;
      end
procedure  P(var s:semaphore);
begin
    s.value:= s.value – 1;          /*              1 */
    if s.value< 0 then W(s.queue); /*              0        P(s)
                                          W(s.queue)
                                          s                    queue*/
end;

procedure  V(var s:semaphore);
begin
    s.value:= s.value + 1;          /*            1 */
    if s.value≤ 0 then R(s.queue);/*                  0        R(s.queue)
                                          s      queue
                                          s                    */
end;
```

W(s.queue) s s CPU R(s.queue) s s

?

s 0 l

P V

1 s s P

s

2 s s

s P s

3 P V

P V

3

s value 0 1

queue (binary) P V BP BV

```
type binary semaphore=record
        value(0,1);
        queue: list of process
      end;
procedure BP(var s:semaphore);
        if s.value=1;
            then
              s.value=0;
        else begin
              w(s.queue);
          end;
procedure BV(var s:semaphore);
        if s.queue is empty;
            then
              s.value:=1;
        else begin
              R(s.queue);
          end;
```

0 1

```
var
    s1: binary-semaphore;
    s2: binary-semaphore;
    c:integer;
```

s1=1 s2=0 c s

s P V

```
P(s)                          V(s)
begin                         begin
  BP(s1);                       BP(s1);
  c:=c-1;                       c:=c+1;
  if c<0 then BP(s2);           if c ≤ 0 then BV(s2);
  BV(s1);                       BV(s1);
end                           end
```

s c P V s

c s s2

s1 1 c

3.3.3

P V TS P V
P V
TS P V

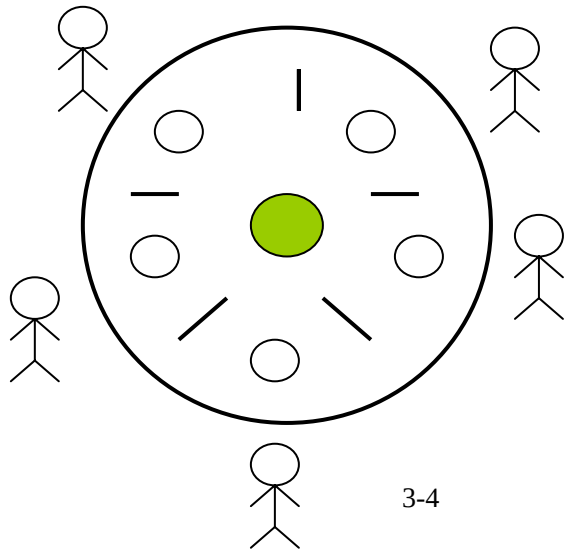
mutex l Tl
P P mutex l T1 T2
P P T2 mutex T1
V V mutex T2 T2
T1 T2
A[j]
s[j]

P V
else V

? P V
P V

P V

Dijkstra 1965



3-4

S_i (i=0,1,2,3,4) 1
P
V

```
var forki :array[0..4] of semaphore;
    forki := 1;
cobegin
  process Pi              // i=0,1,2,3,4
  begin
     L1:
         ;
     P(fork[i]);
```

```
        P(fork[i+1] mod 5);
                ;
        V(fork[i]);
        V(fork[i+1] mod 5);
      goto L1;
    end;
coend.
```

P(fork[4]) P(fork[0])

-
-

-

3.3.4 -

P V

P V
empty full 1 0 empty
full

```
var B : integer;
    empty:semaphore;          /*                    */
    full:semaphore;           /*                       */
    empty:= 1;                /*                       */
    full:= 0;                 /*                */
cobegin
process producer
begin
    L1:
    Produce a product;
    P(empty);
    B := product;
    V(full);
    Goto L1;
end;
process consumer
begin
    L2:
    P(full);
    Product:= B;
    V(empty);
    Consume a product;
```

```
        Goto L2;
    end;
    coend.
```

P V

m n k

mutex 1

empty k full 0

```
var B : array[0..k-1] of item;
    empty:semaphore:=k;      /*                    */
    full:semaphore:=0;       /*                        */
    mutex:semaphore:=1;      /*
    in :integer:= 0;  x      M
```

V P V
P
/
apple orange

```
var
    plate : integer;
    sp:semaphore;          /*                    */
    sg1:semaphore;         /*            */
    sg2:semaphore;         /*            */
    sp := 1;               /*                    */
    sg1:= 0;               /*              */
    sg2:= 0;               /*              */
cobegin                                process son
process father                         begin
begin                                      L3: P(sg1);
    L1:          ;                           plate       ;
    P(sp);                                 V(sp);
              plate;                             ;
    V(sg2);                                goto L3;
    goto L1;                           end;
end;                                   process daughter
process mother                         begin
begin                                      L4: P(sg2);
    L2:          ;                           plate       ;
    P(sp);                                 V(sp);
              plate;                             ;
    V(sg1);                                goto L4;
    goto L2;                           end;
end;                                   coend.
```

3.3.5 -

reader-writer problem Courtois 1971
F 1
2 3
4

rc
mutex rc W

```
var rc: integer;
```

```
    W  mutex: semaphore;
    rc := 0;      /*            */
    W := 1;
    mutex := 1;
procedure read;
begin
  P(mutex);
  rc := rc + 1;
  if rc=1 then P(W)
  V(mutex);

  P(mutex);
  rc := rc - 1;
  if rc = 0 then V(W)
  V(mutex);
end;
procedure write;
begin
  P(W);
       ;
  V(W);
end;

cobegin
  process readeri;
  process writerj;
coend.
process readeri;
 begin
  read;
 end.
process writerj
begin
   write;
  end.
```

/ /

Solaris reader/writer rw-enter( )( ) rw-exit( )( ) rw-tryenter( )( ) rw-downgrade( )(write lock read lock) rw-trygrade( )(read lock write lock)

3.3.6

n

3 waiting
0 customers
0 barbers
0 mutex 1

```
var waiting : integer;  /*                */
    CHAIRS:integer;     /*                  */
    customers, barbers  mutex : semaphore;
    customers := 0; barbers := 0;
    waiting := 0; mutex := 1;
procedure barber;
begin
while(TRUE);               /*                    ?*/
  P(customers);            /*                    */
  P(mutex);                /*        */
  waiting := waiting – 1;  /*                */
  V(barbers);              /*                      */
  V(mutex);                /*          */
  cut-hair( );             /*        */
end;
procedure customer
begin
  P(mutex);                    /*        */
  if waiting < CHAIRS          /*                */
     begin
        waiting := waiting + 1;/*             1*/
        V(customers);          /*                 */
        V(mutex);              /*          */
        P(barbers);           /*                   */
        get-haircut( );       /*                */
     end;
  else V(mutex);             /*           !*/
end.
```

3.4

3.4.1

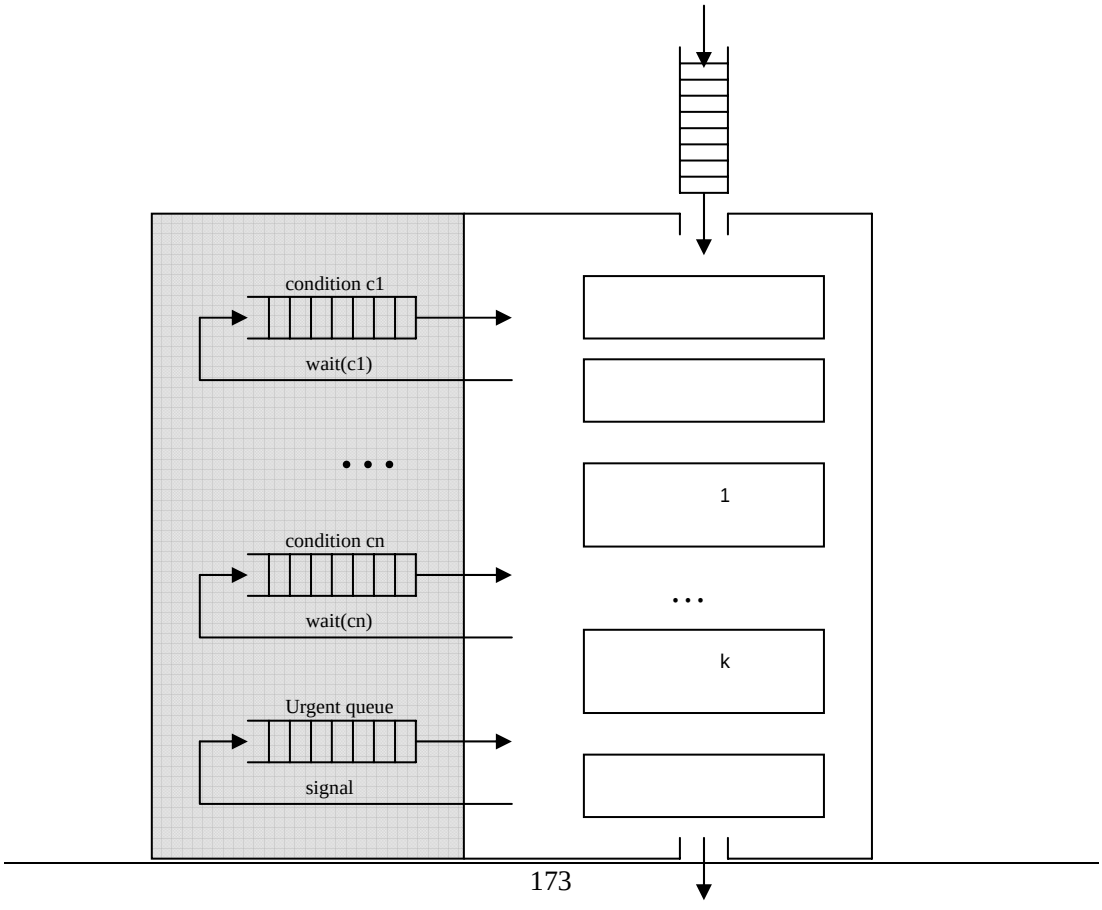
P V

1974 1975 Brinch Hansen Hoare
—— monitor

” ” ” ”
” ”

-
-
-

Pascal Modula-2 Java



```
TYPE <      > = MONITOR
  <           >;
  define <                              >;
  use <                                >;
  procedure <      >  <          >  ;
    begin
      <      >;
end;
  ......
  procedure <      >  <          >  ;
    begin
      <      >
end;
  begin
    <                        >
  end.
```

define

use

3-5

```
TYPE SSU = MONITOR
  var busy : boolean;
      nobusy : condition;
  define require, return;
  use wait, signal
  procedure require;
    begin
      if busy then wait(nobusy); /*                    */
      busy := ture
    end;
  procedure return;
    begin
      busy := false
      signal(nobusy);          /*                   */
    end;
  begin                        /*             */
    busy := false
  end;
```

SSU

define

wait signal use

condition variables
wait signal
condition wait

condition signal
P V

signal
wait signal

SSU busy
nobusy require require
busy busy true
nobusy wait
busy false busy true
return busy false
nobusy signal
busy true busy false
wait signal

wait
signal

wait
signal
signal

P signal Q P
signal Q
- P Q Q
- Q P P

signal P signal
Q

3.4.2 Hoare

P V

signal
signal
wait signal
mutex 1
P(mutex)

V(mutex)

wait V(mutex)

next 0 signal

P(next)

next

V(next) next next-count

0 next

x-sem

0 P(x-sem)

x-count 0

signal

V(x-sem)

```
TYPE interf = RECORD
  mutex:semaphore;     /*                          */
  next:semaphore;      /*   signal                 */
  next_count:integer;  /*   next             */
END;
```

wait signal

```
procedure wait(var x_sem:semaphore,x_count:integer, IM:interf);
begin
  x_count := x_count + 1;
  if IM.next_count > 0 then V(IM.next); else V(IM.mutex);
  P(x_sem);
  X_count := x_count – 1;
end;
procedure signal(var x_sem:semaphore,x_count:integer, IM:interf);
begin
  if x_count > 0 then begin
    IM.next_count := IM.next_count + 1;
    V(x_sem);
    P(IM.next);
    IM.next_count := IM.next_count - 1;
  end;
end;
```

```
P(IM.mutex);
  <      >;
if IM.next_count > 0 then V(IM.next);
                     else V(IM.mutex);
```

var state array[0…4] of (thinking hungry eating)

i state[i]=eating

state[(i-1)mod 5] ≠ eating state[(i+1)mod5] ≠ eating

var self array[0…4] of semaphore

i

self[ ] Hoare P V

condition semaphore

```
TYPE dining-philosophers = MONITOR
  var state : array[0..4] of (thinking, hungry, eating);
      self : array[0..4] of semaphore;
      s-count : array[0..4] of integer;
  define pickup, putdown;
  use wait, signal
procedure test(k : 0..4);
begin
  if state[(k-1) mod 5]<>eating and state[k]=hungry
     and state[(k+1) mod 5]<>eating then begin
        state[k] := eating;
        signal(self[k], s-count[k], IM);
  end;
end;
procedure pickup(i:0..4);
begin
  state[i] := hungry;
  test(i);
  if state[i]<>eating then wait(self[i],s-count[i],IM);
end;
procedure putdown(i:0..4);
begin
  state[i] := thinking;
  test((i-1) mod 5);
  test((i+1) mod 5);
end;
begin
  for i := 0 to 4 do state[i] := thinking;
end;
```

pickup putdown

```
cobegin
```

```
  process philosopher-i
begin
  ......
      P(IM.mutex);
      call dining-philosopher.pickup(i);
      if IM.next-count > 0 then V(IM.next);
                           else V(IM.mutex);

          ......
      P(IM.mutex);
      call dining-philosopher.putdown(i);
      if IM.next-count > 0 then V(IM.next);
                           else V(IM.mutex);
          ......
    end;
coend.
```

3.4.3 Hanson

signal
signal
wait signal check re1ease

- wait
- signal
- check
- release

check re1ease
check
release

```
TYPE interf = RECORD
      intsem : condition; /*                    */
      count1 : integer;  /*                 */
      count2 : integer;  /*
END;                                          */
```

intsem countl count2

```
procedure wait(var s:condition; var IM interf);
  begin
    s := s + 1;
    IM.count2 := IM.count2 – 1;
    if IM.count1 > 0 then
      begin
        IM.count1 := IM.count1 – 1;
        IM.count2 := IM.count2 + 1;
        R(IM.intsem);
      end;
    W(s);
  end;
procedure signal(var s:condition; var IM interf);
  begin
    if s > 0 then
      begin
        s := s – 1;
        IM.count2 := IM.count2 + 1;
        R(s);
      end;
  end;
procedure check(var IM interf);
  begin
    if IM.count2 = 0
    then IM.count2 := IM.count2 + 1;
    else
      begin
        IM.count1 := IM.count1 + 1;
        W(IM.intsem);
      end;
  end;
procedure release(var IM interf);
  begin
    IM.count2 := IM.count2 - 1;
    if IM.count2 = 0 and IM.count1 > 0 then
      begin
        IM.count1 := IM.count1 - 1;
        IM.count2 := IM.count2 + 1;
        R(IM.intsem);
      end;
  end;
```

signal wait s

W(s) s R(s)
s W(IM intsem) R(IM intsem)

-
-
-

1

F 1 2
3

rc wc R W

```
type read-writer = MONITOR
  var rc, wc : integer;
      R, W : condition;
  define start-read, end-read, start-writer, end-writer;
  use wait, signal, check, release;
procedure start-read;
begin
  check(IM);
  if wc>0 then wait(R,IM);
  rc := rc + 1;
  signal(R, IM);
  release(IM);
end;
procedure end-read;
begin
  check(IM);
  rc := rc - 1;
  if rc=0 then signal(W,IM);
  release(IM);
end;
procedure start-write;
begin
  check(IM);
  wc := wc + 1;
  if rc>0 or wc>1 then wait(W,IM);
  release(IM);
end;
procedure end-write;
begin
  check(IM);
```

```
    wc := wc - 1;
    if wc>0 then signal(W,IM);
    else signal(R, IM);
    release(IM);
  end;
  begin
    rc := 0; wc := 0; R := 0; W := 0;
  end.
```

start-read start-write end-read end-write

```
  cobegin
    process reader
  begin
    ......
    call read-writer.start-read;
    ......
    read;
    ......
    call read-writer.end-read;
    ......
  end;
    process writer
  begin
    ......
    call read-writer.start-write;
    ......
    write;
    ......
    call rear-writer.end-write;
    ......
  end;
  coend.
```

2 apple orange

put get

```
  type FMSD = MONITOR
    var plate : (apple, orange);
        full : boolean;
        SP, SS, SD : condition;
```

```
  define put, get;
  use wait, signal, check, release;
procedure put(var fruit:(apple, orange));
begin
 check(IM);
  if full then wait(SP,IM);
  full := true;
  plate := fruit;
  if fruit=orange
     then signal(SS,IM);
  else signal(SD,IM);
  release(IM);
end;
procedure get(var fruit:(apple,orange),x:plate);
begin
  check(IM);
  if not full or plate<>fruit
  then begin
    if fruit = orange
      then wait(SS,IM);
    else wait(SD,IM);
  end;
  x := plate;
  full := false;
  signal(SP, IM);
  release(IM);
end;
begin
  full := false; SP := 0; SS := 0; SD := 0;
end;
```

put get

```
cobegin
  process father
begin
  ......
            ;
  call FMSD.put(apple);
  ......
end;
  process mother
begin
  ......
            ;
  call FMSD.put(orange);
  ......
end;
```

```
  process son
begin
  ......
  call FMSD.get(orange,x);
              ;
  ......
end;
  process daughter
begin
  ......
  call FMSD.get(apple,x);
              ;
  ......
end;
coend;
```

3 monitor

```
type producer-consumer = MONITOR
  var B:array[0..k-1] of item;   /*          */
      in,out:integer;            /*        */
      count:integer;             /*           */
      notfull,notempty:condition;/*        */
  define append,take;
  use wait, signal, check, release;
procedure append(x:item);
begin
  check(IM);
  if count=k then wait(notfull,IM);/*        */
    B[in]:=x;
    in:=(in+1) mod k;
    count:=count+1;               /*           */
    signal(notempty,IM);          /*         */
  release(IM);
end;
procedure take(x:item);
begin
  check(IM);
  if count=0 then wait(notempty,IM);/*        */
    x:=B[out];
    out:=(out+1) mod k;
    count:=count-1;               /*           */
    signal(notfull,IM);           /*         */
  release(IM);
end;
```

```
begin                                   /*      */
  in := 0; out := 0; count:= 0;
end;
cobegin                                 /*      */
process producer;
 var   x:item;
 begin
   produce(x);
   append(x);
 end;
 process consumer;
   var x:item;
  begin
   take(x);
   consume(x);
  end;
coend.
```

3.5

PV

IPC InterProcess Communication

- signal
- PV shared memory
- pipeline shared file
- / message passing

/ PV

3.5.1

signal

OS

del+ctrl+c

signal

PCB

UNIX

32

- SIGCLD SIGHUP SIGKILL SIGCHLD SIGSTOP
- SIGBUS SIGSEGV SIGPWR SIGFPE
- SIGPIPE SIGSYS SIGILL
- SIGINT SIGQUIT

delete break

- SIGTERM SIGALRM SIGUSR1 SIGUSR2
- SIGTRAP

UNIX UNIX

proc user p-clktim alarm

SIGALAM P-sigign 32

1 P-sig 32

1 U-signal[NSIG] 32

kill(pid,sig)

pid sig sig

kill signal(sig,func) sig

function

function=1 sig function=0

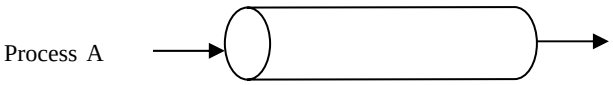
function (function

) signal

3.5.2 `

pipeline

3-6



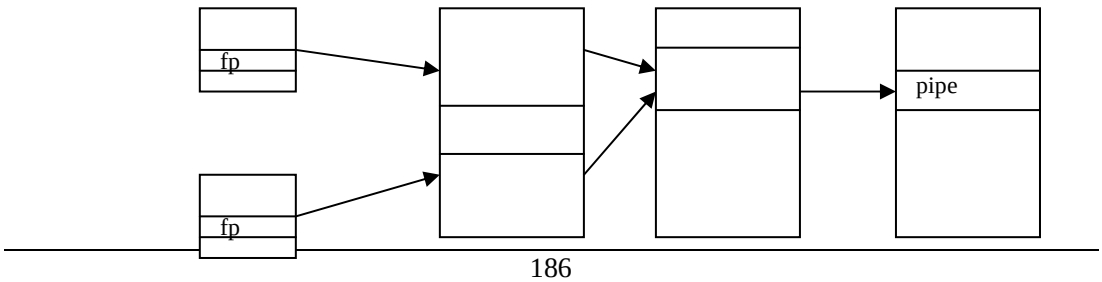
3-6 pipe

pipe UNIX C UNIX

( ) ( )

- i-node
  i-node
- SIGPIPE
- i-node K
  write read
  write
  write
  write

- UNIX
  int pipe(files);
  int files[2];
  pipe
  pipe
  i-node pipe pipe( )
  files[0] files[1] files[0]
  files[1] pipe 3-7



3-7 pipe

```
#include<stdio.h>
#define MSGSIZE 16
char *msg1=”hello,world#1”;
char *msg2=”hello,world#2”;
char *msg3=”hello,world#3”;
 main( )
  {
      char inbuf[MSGSIZE];
      int p[2],j,pid;
      /*open pipe*/
       if(pipe<0) {
            perror(“pipe call”);
            exit(1);
    }
       if((pid=fork( )<0) {
            perror(“fork call”);
            exit(2);
    }
   /*if parent, then close read file descriptor and write down pipe*/
      if(pid>0 {
            close(p[0]);
            write(p[1],msg1,MSGSIZE);
            write(p[1],msg2,MSGSIZE);
            write(p[1],msg3,MSGSIZE);
            wait((int*)0);
      }
 /*if child ,then close write file descriptor and read from pipe*/
       if (pid==0) {
            close(p[1]);
            for(j=0;j<3;j++)
               read(p[0],inbuf,MSGSIZE);
               printf(“%s\n,inbuf”);
      }
   }
  exit(0);
}
```

UNIX

FIFO

UNIX write read

mknod(pipename,S_FIFO+rw,0) pipe

open(pipename,O_RDONLY)

open(pipename,O_WRONLY)

read write

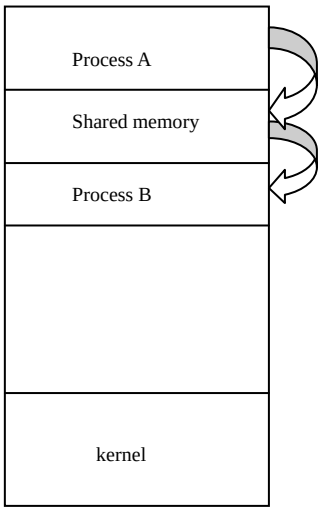
```
who|sort|more
```

10 100
1000 10

AT&T UNIX

3.5.3

3-8



3-8

UNIX/Linux

- shmget(key,size,permflags)
  size
  key permflags
- shmat(shm-id,daddr,shmflags)
  shm-id shmget daddr
  permflags
- Shmdt(memptr)
  memptr
- Shmctl(shm-id,command,&shm-stat) shm-id
  command &shm-stat
  shmget key
  key
  shmat

  shmdt shmat

3.5.4

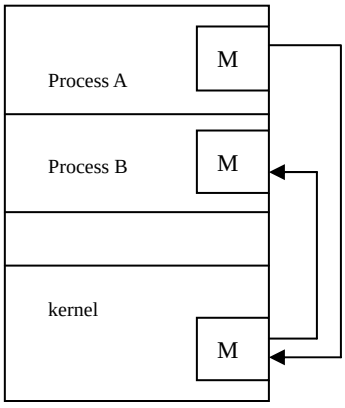
1

—— **message passing**

send receive

2

UNIX pipeline socket 3-9



3-9

1

send receive

- send P P
- receive Q Q
  P Q

2

——

/

port

“ ” “ ”

- send A A
- receive A A

“ ” “ ”

-
-

R( ) W( )

```
type box=record
        size:integer;                    /*          */
        count:integer;                   /*          */
        letter:array[1..n] of message;   /*      */
        S1,S2:semaphore;                 /*                  */
end

procedure send(varB:box,M:message)
    var i:integer;
    begin
      if B.count=B.size   then W(B.s1);
      i:=B.count+1;
      B.letter[i]:=M;
      B.count:=i;
      R(B.S2)
     end;

procedure receive(varB:box,x:message)
    var i:integer;
    begin
      if B.count=0   then W(B.s2);
      B.count:=B.count-1;
      x:=B.letter[1];
      if B.count  ≠ 0 then for i=1 to B.count do B.letter[i]:=B.letter[i+1];
```

3.5.5

0 send receive
receive receive
receive
send rendezvous

( ) (
)

P1 P2 P3 A P1 A P2
P3 A Pl ?

- Pl
-
-

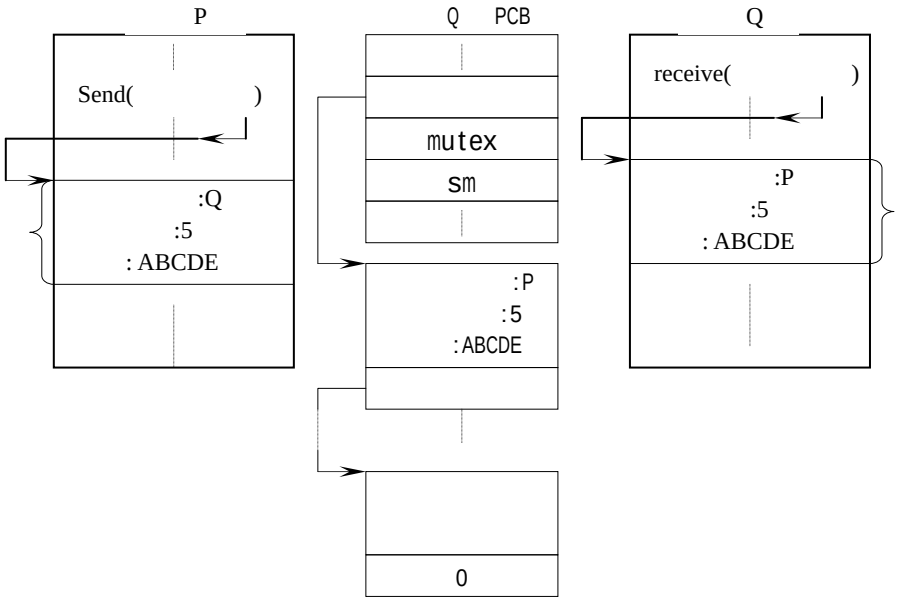
mailbox
mailbox mailbox

1973 P.B.Hansan RC4000

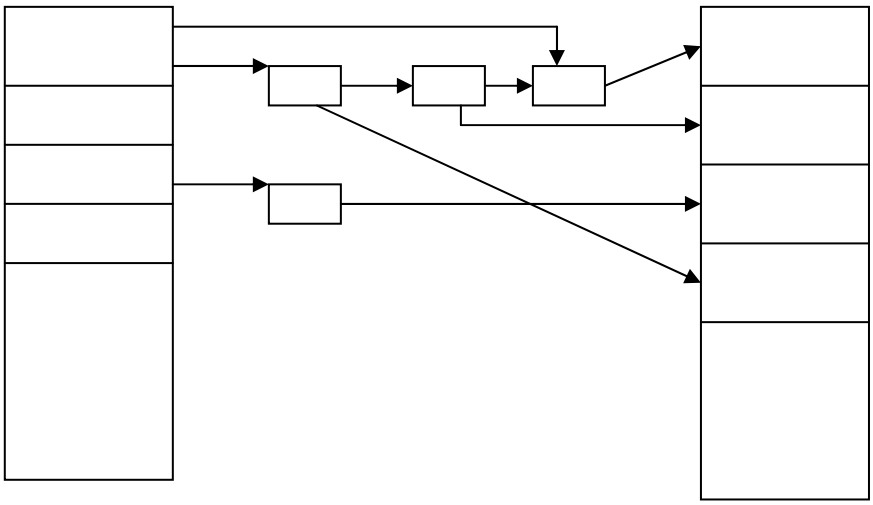
3-10

- sender
- size
- text
- next-ptr
  PCB
- mptr
- mutex 1
- sm 0

- send
  PCB P(mutex)
  V(sm) 1 V(mutex)
- receive P(sm) P(mutex)
  V(mutex)



3-10



3-11Unix/Linux

3-11 UNIX/Linux

- msgbuf
- 100
-

/

UNIX/Linux

- msgget
- msgsnd
- msgrcv
- msgctl

msgget

msgsnd

msgrcv

=0 <0 >0 (=0

>0 <0

)

msgctl

n

n

0

receive

3.6

3.6.1

-
-
-

“ ”

P Q

PV

Q1 Q2 r1 r2 s1 s2 r1 r2

s1 s2 1

```
        Q1                          Q2
    ……..                        ……..
    P(s1);                      P(s2);
    P(s2);                      P(s1);
    ……..                        ……..
        r1   r2;                    r1   r2
    ……..                        ……..
    V(s1);                      V(s2);
    V(s2);                      V(s1);
    ……..                        ……..
```

Q1 Q2 Q1 P(s1) P(s2) Q2 P(s2) Q1 P(s2) Q2 P(s1)

P

m n $m < n \cdot K$ m=5 n=5 k=2

P1 P3 S3 P2 S1 P2 P1 S1 P3 S2 P3 P2 S2 S3

(deadlock prevention) (deadlock avoidance) (deadlock detection and recovery)

3.6.2

n P1 P2 … Pn Pi i=1 … n Rj(j=1 … m)
Rj Pi+1 i=1 … n-1 Pn P1
n n

3.6.3

1

1971 Coffman

- **mutual exclusion**
- **hold and wait**
- **no preemption**
- **circular wait**

2

3

n m ri i m
r1,r2……,rm
ri(1 ≤ i ≤ m) rj(j<i)

t1 P1 rk1 rk1 P2
P2 P1
t2 P2 rk1 rk2
n tn Pn rkn
rkn Pi 1 ≤ i<n
P2 rk1 rk2
k1 k2

k2 k3 …<ki<… kn
Pi rkn rki
kn ki

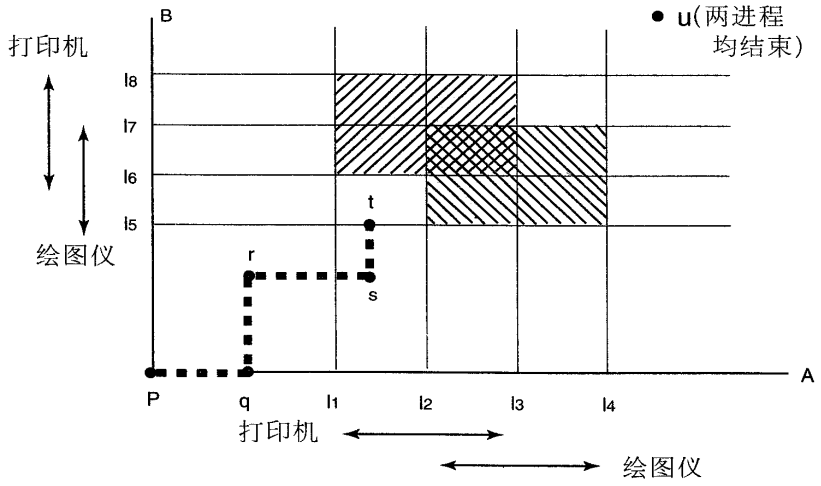
3.6.4

banker’s algorithm

1

3-12
A
B A I1 I3 I2
I4 B I5 I7 I6 I8

P A A
q q B



3-12

A r s I1 B
t

I1 I2 I5 I6
I2 I6 A
B

t A I4 I4
u

2

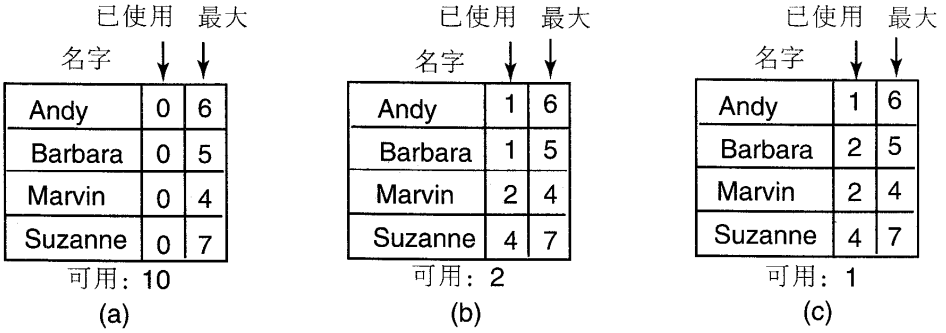
Dijkstra 1965

∑ N
●
●
●

∑

3-13 a 4

10 22 3-13 b



3-13 (a) (b) (c)

3-13 b

Marvin 4

Suzanne Barbara

Barbara 3-13 b 3-13 c

4

3-14

5

3-14 E P A

E 6 3 4 2 CD-ROM P

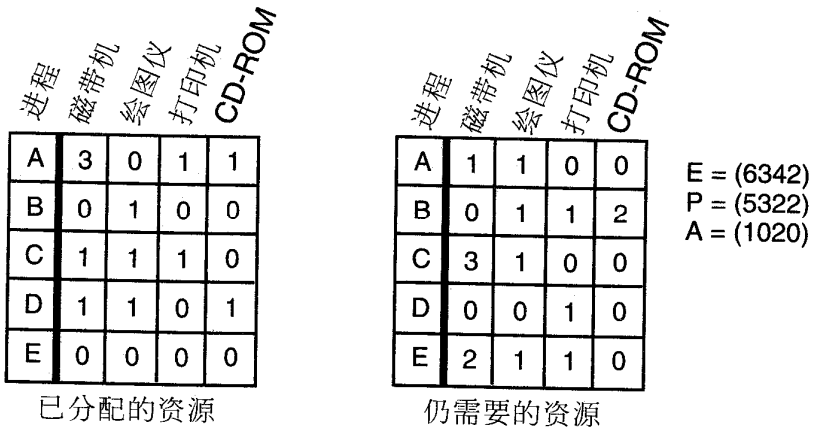
5 3 2 2 CD-ROM

1 A

2

A

3



3-14

1

B

D A E

B E E

1000 E

Dijkstra 1965

4

n m

● -- m

Resource=(R_1,R_2,…,R_m)

- -- m
- Available=(V_1,V_2,…,V_m)
- -- Cij Pi Rj

$$Claim=\begin{pmatrix} C_{11} & C_{12} & & C_{1m} \\ C_{21} & C_{22} & & C_{2m} \\ & & & \\ C_{n1} & C_{n2} & & C_{nm} \end{pmatrix}$$

- — Aij Pi Rj

$$Allocation=\begin{pmatrix} A_{11} & A_{12} & & A_{1m} \\ A_{21} & A_{22} & & A_{2m} \\ & & & \\ A_{n1} & A_{n2} & & A_{nm} \end{pmatrix}$$

- $Ri=Vi+\sum Aki$ i=1,..,m,k=1,..,n;
- $Cki \le Ri$ i=1,..,m,k=1,..,n;
- $Aki \le Cki$ i=1,..,m,k=1,..,n;

Ri

$$R_i \ge C_{(n+1)i}+\sum C_{ki} \qquad i=1,..,m,k=1,..,n;$$

R_i

“ ”

T0

$P_1,..,P_n$, $P_k(k=1,..,n)$

$C_{ki}-A_{ki} \le Available_i+\sum A_{ji}$ j=1,..,k-1;k=1,..,n;i=1,..,m;

P_k $Available_i$ T_0

P_k $\sum A_{ji}$ P_k

P_k

P_k $P_j(j=1,..,k-1)$

P_k P_k P_{k+1}

T_0

“ ”

(banker's algorithm)

```
type state= record                                          /*
              resource,available:array[0…m-1] of integer;
              claim,allocation:array[0…n-1,0…m-1] of integer;
            end
                                                            /*
if alloc[i,*]+request[*]>claim[i,*]  then  <error>      /*
else
    if request[*]>available[*]  then  <suspend process.>
    else                                                /*
        < define newstate by:
       allocation[i,*]:=allocation[i,*]+request[*]
       available[*]:=available[*]-request[*] >
    end;

    if safe(newstate)  then
        < carry out allocation>
    else
        <restore original state>
        < suspend process>
    end
end
                                                        /*
function safe(state:s):boolean;
var currentavail:array[0…m-1] of integer;
    rest:set of process;
begin
    currentavail:=available;
    rest:={all process};
    possible:=true;
while possible do
     find a P_k in rest such that
        claim[k,*]-allocation[k,*] ≤ currentavail;
    if  found  then
       currentavail:=currentavail+allocation[k,*];
       rest:=rest-[P_k];
    else
       possible:=false;
    end
  end;
  safe:= (rest=null);
end.
```

1 2

2 3

3 P_i

allocation[i,*]:=allocation[i,*]+request[*]

available[*]:=available[*]-request[*]

4 5

P_i

5

- currentavail possible
- currentavail:=available,possible:=true
- possible:=true rest claim[k,*]-allocation[k,*] ≤ currentavail Pk

currentavail:=currentavail+allocation[k,*] Pk

rest:=rest-[Pk] possible:=false

- rest

$\{P_0, P_1, P_2, P_3, P_4\}$ A B C A 10 B 5 C 7 T_0

Process	allocation			claim			available		
	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	5	3	3	3	2
P_1	2	0	0	3	2	2			
P_2	3	0	2	9	0	2			
P_3	2	1	1	2	2	2			
P_4	0	0	2	4	3	3			

$C_{ki}-A_{ki}$

Process	$C_{ki}-A_{ki}$		
	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

1 T0

T0 $\{P_1,P_3,P_4,P_2,P_0\}$

T0

	currentavil	$C_{ki}-A_{ki}$	allocation	currentavil+allocation	possible

	A	B	C	A	B	C	A	B	C	A	B	C	
P_1	3	3	2	1	2	2	2	0	0	5	3	2	TRUE
P_3	5	3	2	0	1	1	2	1	1	7	4	3	TRUE
P_4	7	4	3	4	3	1	0	0	2	7	4	5	TRUE
P_2	7	4	5	6	0	0	3	0	2	10	4	7	TRUE
P_0	10	4	7	7	4	3	0	1	0	10	5	7	TRUE

2 P_1

P1 1 A 2 C

- request1(1,0,2) ≤ C_{k1}-A_{k1}(1,2,2)
- request1(1,0,2) ≤ Available(3,3,2)

P_1 Available Allocation C_{ki}-A_{ki}

process	allocation			C_{ki}-A_{ki}			available		
	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	4	3	2	3	0
P_1	3	0	2	0	2	0			
P_2	3	0	2	6	0	0			
P_3	2	1	1	0	1	1			
P_4	0	0	2	4	3	1			

? P_1

	currentavil			C_{ki}-A_{ki}			allocation			currentavil+allocation			possible
	A	B	C	A	B	C	A	B	C	A	B	C	
P_1	2	3	0	0	2	0	3	0	2	5	3	2	TRUE
P_3	5	3	2	0	1	1	2	1	1	7	4	3	TRUE
P_4	7	4	3	4	3	1	0	0	2	7	4	5	TRUE
P_0	7	4	5	7	4	3	0	1	0	7	5	5	TRUE
P_2	7	5	5	6	0	0	3	0	2	10	5	7	TRUE

{P_1,P_3, P_4,P_0,P_2}

P1

3 P_4

P_4

- request4(3,3,0) ≤ C_{k4}-A_{k4} (4,3,1)
- request4(3,3,0) >Available(2,3,0)

P4

4 P0

P0

- request0(0,2,0) ≤ C_{k0}-A_{k0} (7,3,1)
- request4(0,2,0) ≤ Available(2,3,0)

P0

	allocation	Cki-Aki	available
	A B C	A B C	A B C
P0	0 3 0	7 2 3	2 1 0
P1	3 0 2	0 2 0	
P2	3 0 2	6 0 0	
P3	2 1 1	0 1 1	
P4	0 0 2	4 3 1	

P0 0,2,0

P0

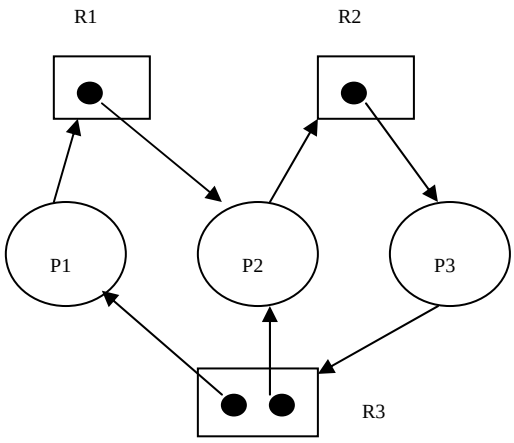
3.6.5

1

“ ”

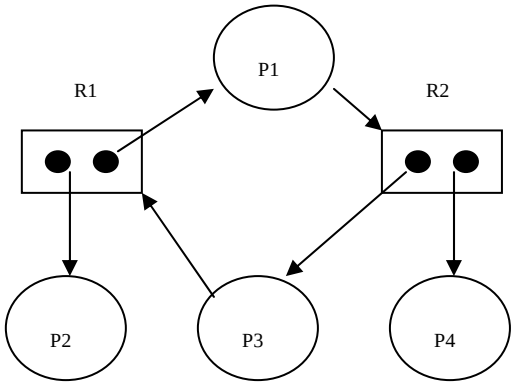
- -

Pi→Rj Pi Rj
Rj Pi
Rj Rj→Pi Rj Pi
Pi 3-15
-



-

3-16 P1
P3 R1 R2 R2 R1
P2 P4 R1 R2
P1 P3
-



3-16

“ ” -

(1) -
(2) -

(3) -
-

-

-

2

possible[k] k=1,..,n
1 currentavail:=available
2 rest Pk, claim[k,*]-alloc[k,*]=0 possible[k]:=true
possible[k]:=false k=1,..,n
3 rest Pk
possible[k]=false&(request[*] ≤ currentavail)
Pk 4 5
4 currentavail:=currentavail+allocation possible[k]:=true 3

(5) k=1,..,n possible[k]=true possible[k]=false Pk

$$A[b_{ij}]=\begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{pmatrix}$$

b_{ij} P_i P_j b_{ij} P_i P_j

“ ” warshall warshall A $[b_{ij}]$ b_{ij} $[b_{ij}]$

```
for k:=1 to n do
    for i:=1 to n do
        for j:= 1 to n do
            bij:=bij (bik bkj)
```

b_{ij} i j b_{ik} b_{ki} i P_k k j P_i j warshall

A A

b_{ii}=1(i= 2……,n)

P_1 P_2 P_3

$$A[b_{ij}]=\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

b_{12} b_{23} b_{31} 1 k=1 i j b_{31}=1 b_{12}=1 b_{32} 1 k=2 i j (b_{12} b_{23}) b_{31} 1 (b_{32} b_{23}) b_{33} 1 k=3 i j b_{11} b_{22} 1

P_1 P_2 P_3

-
-
-

? CPU

-
- checkpoint
-

3.7 Windows 2000/XP

3.7.1 Windows 2000/XP

Windows 2000/XP

API

- CreateMutex
- OpenMutex
- ReleaseMutex

0

API

- CreateSemaphore

- OpenSemaphore
- ReleaseSemaphore

API

- CreateEvent
- OpenEvent
- SetEvent PluseEvent
- ResetEvent

WaitForSingleObjecs WaitForMultipleObjects

Windows 2000/XP
API

CRITICAL_SECTION API InitializeCriticalSection( ) EnterCriticalSection( ) TryEnterCriticalSection(0) LeaveCriticalSection( ) DeleteCriticalSection( )

API

API InterlockedExchange(32
) InterlockedCompareExchange( )
InterlockedExchangeAdd( ) InterlockedDecrement(1
) Interlockedincrement(1)

3.7.2 Windows2000/XP

Windows2000/XP (signal)

(1)SetConsoleCtrlHandler
GenerateConsoleCrtlEvent
CTRL_C_EVENT CTRL+C
SetConsoleCtrlHandler CTRL+C
(2)signal raise
UNIX

Windows2000/XP (shared memory)

Windows2000/XP
(file mapping)
CreateFileMapping(
) OpenFileMapping( )
MapViewOfFile( ) FlushViewOfFile(
) UnmapViewOfFile( )
CloseHandle( )
Windows2000/XP (pipe)
I/O
UNIX CreatePipe
ReadFile WriteFile

C/S CreateNamePipe
ConnectNamePipe CallNamePipe
ReadFile WriteFile( ) ReadFileEx
WriteFileEx( )
Windows2000/XP (mailslot)
C/S

CreateMailslot(
) GetMailSlotInfo( ) SetMailslotInfo(
) ReadFile( ) CreateFile( )
WriteFile( )
Windows2000/XP (socket)

C/S
TCP/IP
Windows2000/XP ”Winsock” BSD
API

3.8 Linux

Linux

```
struct semaphore {
   atomic-t count;
   int waking;
   struct wait-queue *wait;
};
```

count

0 count sema-init

count 1( ) ( )

waking up

wait-queue

down(P) up(V) down-interruptible wake-up wake-up-interruptible Down up

down-interruptible

0

wake-up wake-up-interruptible TASK-INTERRUPTIBLE

1 1 Linux semary

semid-ds

```
static struct semid-ds *smeary[SEMMNI]

struct sem {
   int semval;
   int sempid;
};
```

struct sem semval 0

sys-semctl

Sempid pid

```
struct semid-ds {
   struct  ipc-perm sem-perm;
   -kernel-time-t sem-otime;      /*
   -kernel-time-t sem-ctime;      /*
   struct sem *sem-base;
   struct sem-queue *sem-pending;
   struct sem-queuq **sem-pending-last;
   struct sem-undo *undo;
   unsigned short sem-nsems;    /*
};
```

semid-ds

sem-base “ ”

SEMMSL sem-pending

sem-pending-last

undo

```
struct sem-queue { /*                    sem-queue
   struct sem-queue *next;      /*
```

```
    struct sem-queue **prev;      /*
    struct wait-queue *sleeper;   /*
    struct sem-undo *undo;        /*   sops
    int pid;                      /*
    int status;                   /*
    struct semid-ds *sma;         /*          struct sem-ds
    struct sembuf *sops;          /*
    int nsops;                    /*
    int alter;                    /*
};
struct sem-undo {/*      undo            sem-undo
    struct sem-undo *proc-next;   /*                 undo
    struct sem-undo *id-next;     /*                      undo
    int semid;                    /*                           semid-ds
    short *semadj;                /*
};
```

semget semop semctl Semget
semop ( )
semctl (
) Linux
0 0

sleeper sem-queue
sem-queue sem-pending sem-pending-last
Linux sem-pending
sem-queue

Linux

semid-ds task-struct sem-undo

3.9

CH2

(Dekker) CPU

Dijkstra P V Hansen Hoare

—— P V

PV

/

P V

1.
2.
3.
4.
5.

6.
7.
8.
9. Dekker
10. Peterson
11.
12.
13. P V
14. P V
15.
16.
17.
18. Hanson Hoare
19. P V ?
20.
21.
22. pipeline
23. ?
24.
25.
26. ? ?
27.
28.
29. ? ?
30. P V
?
31. /
32. 20 65
3
33.
?
34.
?
35. 8 N 3 N
36. ?
37.
38. ?
39. m n x (1 x m)

(n m x)

40. ?

41. ?

42.

43. P V

44.

45.

	¼û

1 R M P

1 K

2 K

P V

2 n

(1)

(2) m m ≤ n

3 P1 P2 S1 S2 0

P1 P2 x y z

P1	P2
begin	begin
y:=1;	x:=1;
y:=y+3;	x:=x+5;
V(S1);	P(S1);
z:=y+1;	x:=x+y;
P(S2);	V(S2);
y:=z+y	z:=z+x;
end.	End.

4

100 1 P V 2

5

P1 P2 P1 P2

P1 P2

6 Wait Signal

7

P V

8 4 1 2 3 4

9 S P V S S 0 S=0 S<0

10 A B C D E

```
Process P          Process Q
begin              begin
  A                  C
  B                  D
  C                end
end
```

11

1

2

12

(1)

(2)

13

1

2

14 (1) (2)

15 P V

16 Bakery Algorithm Lamport 1974

```
var choosing  array [0…n  1]  of  boolean
```

```
    number  array [0…n  1]  of  integer
repeat
    choosing [i]  =true
    number[i]  1  max ( number[0]  …  number[n  1])
    choosing[i]  =false
    for  j  0  to  n  1  do
    begin
    while choosing [j] do {nothing}
    while number[j] ≠ 0  number [j]  j  number[i]  i
    do {nothing}
end
    {critical section}
    number [i]:=0
    {remainder}
forever
```

choosing number false 0 i i

a b c d a c or a=c b d

1 2 3

17 (patil 1971)

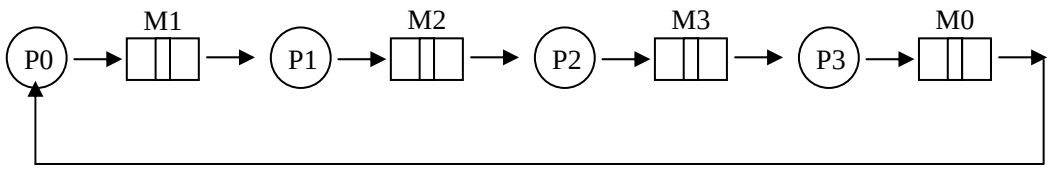
(1) P V (2)

18 Pi i=0…3 Mj j=0…3

Pi Mi $M_{(i+1)mod4}$ M0 M1 M2 M3

3 3 2 2 M0 PV

Pi i=0…3



19 P_i Q_j R_k P_i Q_j M1

buf1 Q_j R_k M2

buf2 P_i buf1 Q_j

buf2 R_k P V

20 P Q P 1

W Q 1

INPUT OUTPUT I/O Delay seconds

21 m n m n m ≤ n ?

22 N M / M M+N

23 100 A P B 200 ?

24 Jurassic m n n P V m n

25 k 1 2 … k file K 1) P V 2)

26 Available=(1 1 2)

,	Claim			Allocation		
	R1	R2	R3	R1	R2	R3
P1	3	2	2	1	0	0
P2	6	1	3	5	1	1
P3	3	1	4	2	1	1
P4	4	2	2	0	0	2

(1) Cki-Aki?

(2) ?

(3) P_1 request1(1 0 1)

(4) P_1 P_0 request0(1 0 1)

(5) P_0 P_2 request0(0 0 1)

27 A B C D 4 P0 P1 P2 P3 P4

Process	Allocation				Claim				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	3	2	0	0	4	4	1	6	2	2
P_1	1	0	0	0	2	7	5	0				
P_2	1	3	5	4	3	6	10	10				
P_3	0	3	3	2	0	9	8	4				
P_4	0	0	1	4	0	6	6	10				

(1)

(2) P1 request1(1 2 2 2)

28

Available=(1 0 2 0)

$$\text{Need}=\begin{pmatrix}1&1&0&0\\0&1&1&2\\3&1&0&0\\0&0&1&0\\2&1&1&0\end{pmatrix} \quad \text{Allocation}=\begin{pmatrix}3&0&1&1\\0&1&0&0\\1&1&1&0\\1&1&0&1\\0&0&0&0\end{pmatrix}$$

(1)

(2) request2(0 0 1 0)

(3) request5(0 0 1 0)

29 150 P1 70, 25 P2 60, 40 P3 60, 45 (1)P4 P4 60, 25 (2)P4 P4 60, 35

30 X Y (1) X Y (2) -N<X - Y <M N M P V X Y

31 A B m A B A B n(n<m) P V

32 A1 A2 … An1 m B1 B2 … Bn2

(1)

(2) B1 B2 … Bn2

(3) M PV

33 R1 3 R2 4 P1 P2 P3 P4 4

→ R1→ R2→ R1→ R1→ R2→ R1

(1)

(2) —

34 PV

35 -

(1) P V (2)

36 R1 R2

```
            c1:=1-c2
        until c2<>0;
        Critical  Section;    (*          *)
        c1:=1
          until false
    end;
procedure  p2;    (*              p2 *)
    begin
      repeat
        Remain  Section  2;
        repeat
            c2:=1-c1
        until c1<>0;
        Critical  Section;    (*          *)
        c2:=1
          until false
    end;
    begin    (*          *)
      c1:=1;
      c2:=1;
      cobegin
          p1;p2    (*          p1, p2          *)
      coend
    end.
```

42 ?

```
repeat
    key  =true
    repeat
        swap(lock  key)
    until  key=false
        Critical  Section;    (*          *)
    lock  =false
        other code
    until false
```

43 1

```
const  n=50
var tally    integer
procedure  total( )
var count  integer
begin
   for count  =1 to n do tally  =tally+1
end
```

```
begin (/*main program*/)
tally   =0
cobegin
   total()   total()
coend
writeln(tally)
end
```

(1) tally

(2) tally

44

```
var   balocked   array[0     1] of boolean
      turn   0     1
procedure P(id   integer)
 begin
    repeat
       blocked[id]   =true
       while turn ≠ id do
          begin
             while blocked[1-id] do skip
             turn   =id
          end
       {Critical Section}
      blocked[id]   =false
       {remainder}
      until false
    end
    begin
       blocked[0]   =blocked[1]   =false
       turn   =0
    cobegin
       P[0]   P[1]
    coend
 End
```

45 P1 P2 P3

P1

P2

P3

1) 2) P V

46 A B A B

1) P V 2)

47

. 1) P V 2) ,

48

49 P1 P2 P3 F P1 F P2 F P3 F
F (1) P V (2)

50 100 10
10
P V

51 10000

Monitor, Monitor

52 2m 2n
(1) (2)
(3)
P V

53

10 3
1

54 free[index] n
(index=0 … n-1) free[index]=true index free[index]=false
index acquire release

55 AND P " AND " "
"P V SP SV(Simultaneous P V) SP(s_1 s_2 s_n) VS(s_1 s_2 s_n)

```
procedure SP(var s1 ,sn semaphore)
begin
    if s1>=1 & & sn>=1 then begin
       for i:= 1 to n do
           si:= si-1;
    end
    else begin
           {                    si<1      si
                                    SP      };
    end
```

```
            procedure VP(var s1          ,sn   semaphore)
            begin
                for i:=1 to n do   begin
                    si:=si+1;
                    {          si                                    };
                end
            AND
56      AND            SP   SV                  -
57      AND            SP   SV
58      AND          SP
    AND          SV                ?
59                   (                                              )
        AND                                                  SP(s1,t1,d1;…;sn,tn,dn)
  SV(s1,d1;…sn,dn)
        procedure   SP(s1,t1,d1   …   sn,tn,dn)
            var s1,…,sn:semaphore;
                t1, …,tn:integer;
                d1,…,dn:integer;
        begin
            if s1>=t1& …&sn>=tn   then   begin
                for i :=1   to   n   do
                  si:=si-di;
            end
            else
                {                              si<ti          si
                                                    SP            };
            end
        end
        procedure   SV(s1,d1;…sn,dn)
            var s1,…sn:semaphore;
                d1,…dn:integer;
        begin
            for i:=1   to   n   do   begin
                si:=si+di;
                {          si                                    };
            end
        end
            ti                          di
```

60

- SP(s,d,d)
- SP(s,1,1)
- SP(s,1,0)

61 -

CH4

——

GB

-
-
-
-

Intel

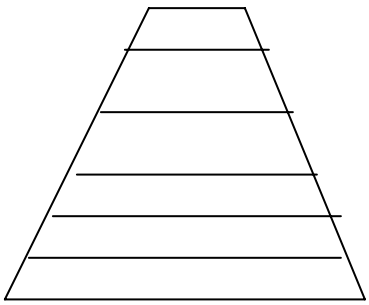
Pentium Windows2000/XP Linux

4.1

4.1.1

7

4-1



4-1

word

1 μ s 0.1
μ s 50% 0.55
μ s CPU 100
512KB 15ns 128MB 60ns 40GB
1TB
/

4.1.2 caching

caching

cache CPU
cache

cache CPU
cache
cache cache
cache CPU

cache cache CPU
cache 25ns 128KB 256KB
cache 80%-99%
cache cache

cache

cache
B A 1
B I/O A
A cache A CPU
A 1 CPU
A A A

A

CPU
A A

CPU CPU CPU
cache A cache CPU
A cache A cache
coherency

4.1.3

0
0

( )
CPU

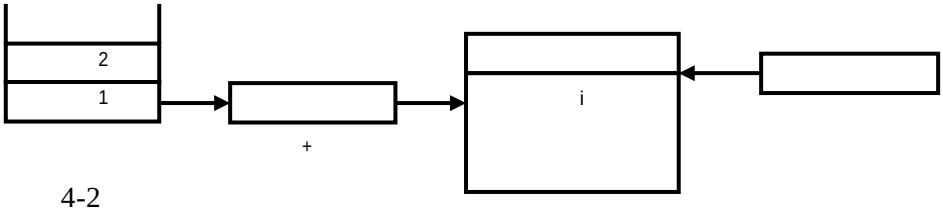
4.2

4.2.1

4-2

fence register

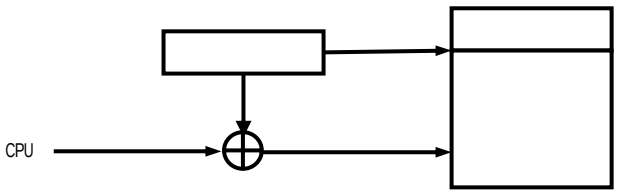
( )



4-2

CPU

4-3



4-3

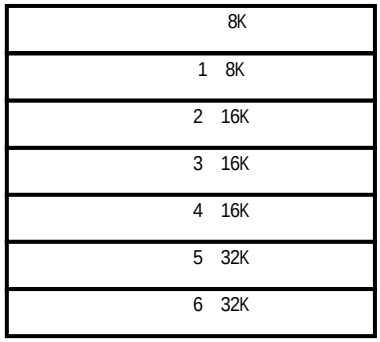
-
-
-

70

IBM7094 FORTRAN IBM1130

MIT CISS cromemco CDOS Digital Research Dyhabyte CP/M DJS0520 0520FDOS

4.2.2



4-4

fixed partition

IBM

OS/MFT Multiprogramming with a Fixed Number of Tasks

“ ”

4-5

1	8K	8K	0
2	16K	16K	Job1
3	32K	16K	0
4	48K	64K	0
5	64K	32K	Job2
6	96K	32K	0

4-5

“ 0 ”

“ 0 ”

4-5 2 5

Job1 Job2

“ 0 ”

4-4

“ 0 ”

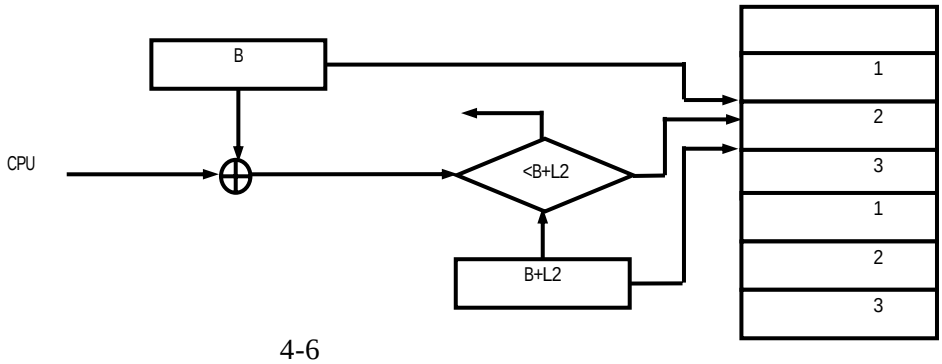
4-6

—— / CPU

/

B

/



4-6

CPU

4-5 Job1 Job2

10 18 16 32 20

4.2.3

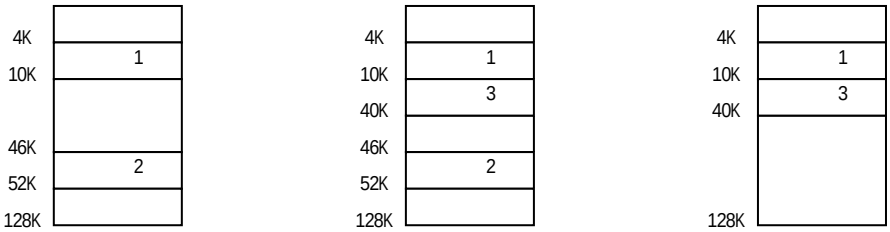
1

variable partition

IBM

OS/MVT Multiprogramming with a Variable Number of Tasks

4-7



4-7

“ ”

“ ” 4-8

1	4K	6K	Job1
2	46K	6K	Job2

a

1	10K	36K	
2	52K	76K	

b

4-8

4-7 30K

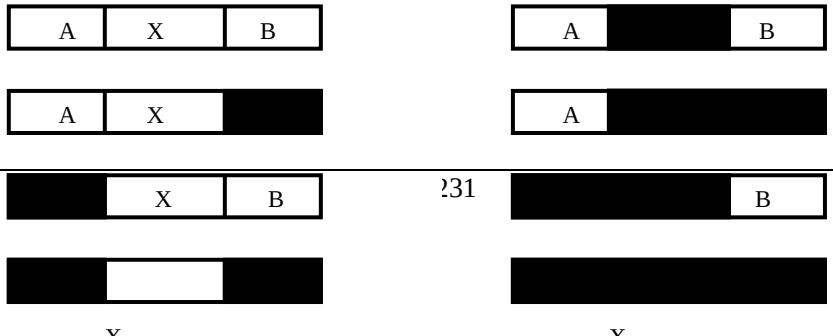
36K

30K 6K

“ ”

X A

B A B 4-9



1 (first fit)

“ ”

2) (next fit)

3 (best fit)

4 (worst fit)

5) (quick fit)

n 2KB

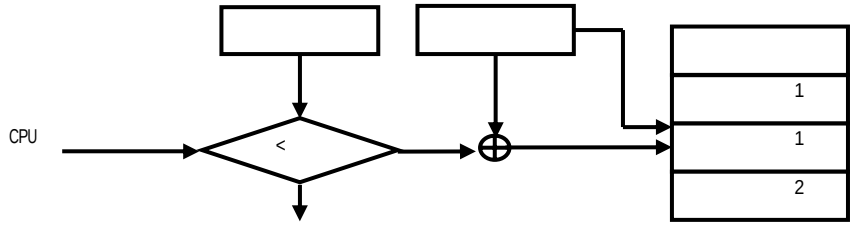
4KB 8KB

9KB 8KB

2

CPU

4-10



4-10

/

/

PSW

/ / CDC6600

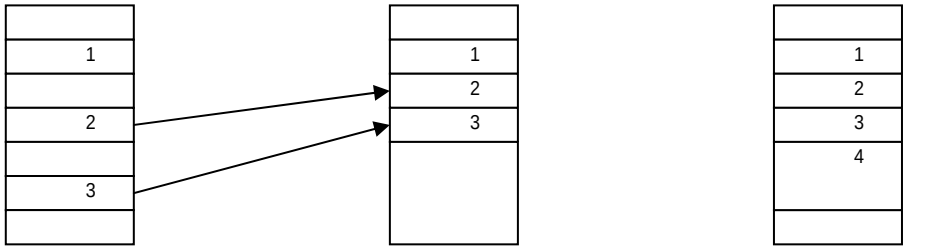
/

/

/

3

4-11



a b c 4

4-11

( )

i xK

1

2 xK 5

3 xK i

4

/

5 xK

i /

6

A B A

B

A B

n

1/2 + 1/3 + + 1/n · V

n /n × V

n n-1

4.3

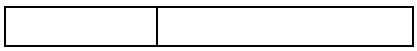
4.3.1

1 page frame

2

page

3



“ ”

4

( ) ( )

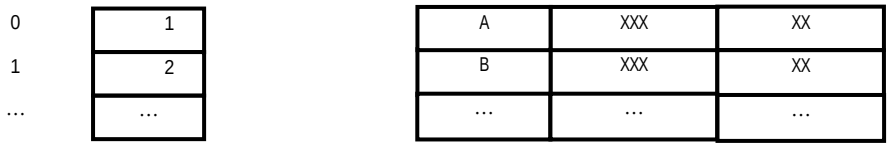
page

table

page table control register

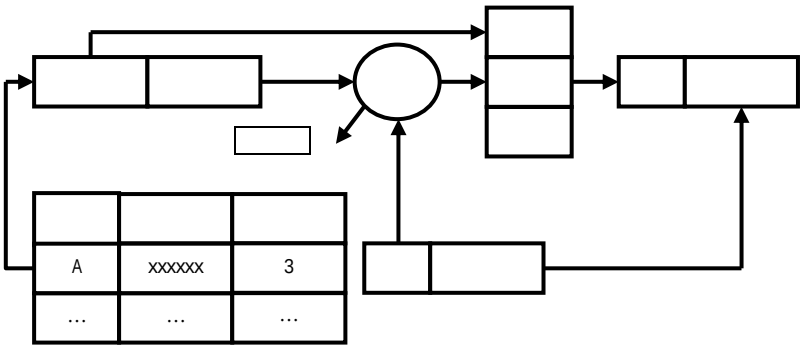
4-12

× +



4-12

4-13 ×



4-13

CPU

4.3.2

/

/

MMU( )

associative

memory TLB Translation Lookaside Buffer

Intel 80486 32

MMU

“ ”

(hit ratio)

100%

0

100 20 32

90%

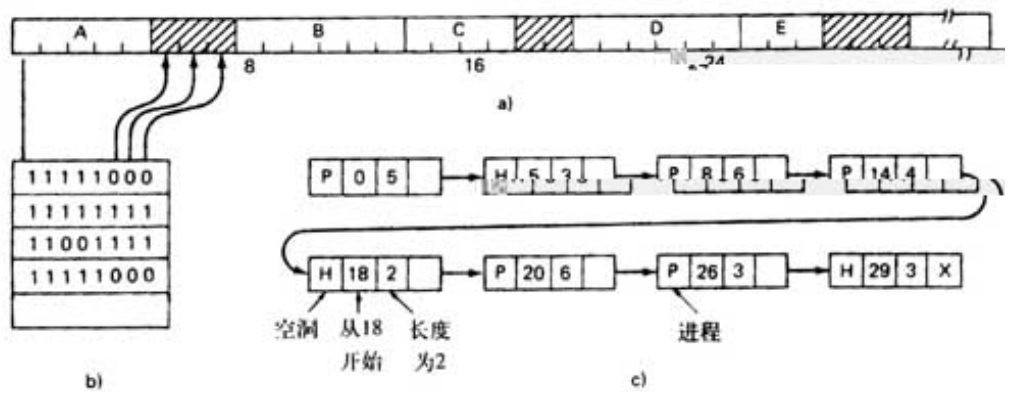
100 20 × 90% 100+100 × 1-90% 128

200

CPU

4.3.3

0/1 / 4-14



4-14

“ 0 ”

“ 0 ”

4-14 c P

H

4.3.4

30K 10 5K
10 350K 80K

EDIT
EDIT
3 5
EDIT
3 5

/

PSW “ ”
CH7 IBM System/370

4.3.5

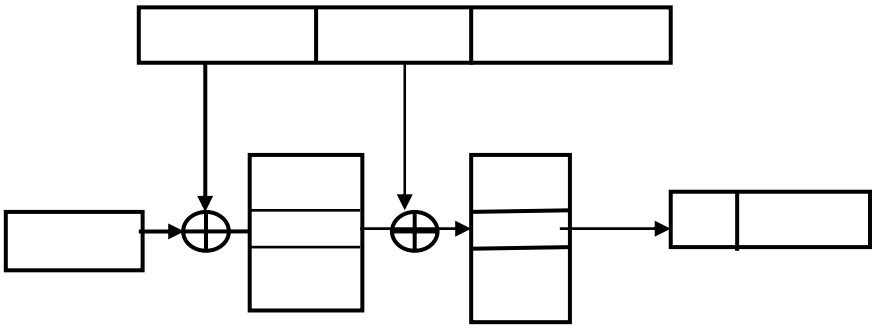
32 64 2^{32} 2^{64}
Windows 2000/XP Intel
x86 CPU 32 2^{32} 4KB 2^{12}
4GB 2^{32} 1 2^{20} 4
4MB 2^{22}

1K 2^{10}

page directory table

CPU

4-15



( ) ( )

4-15

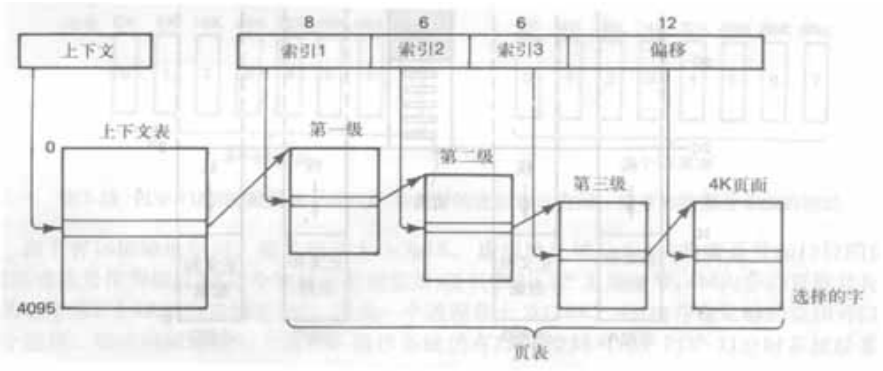
“ ” (
) “ ”
“ ”

“
” “ ”

64

SUN SPARC 4-16
4096

CPU MMU

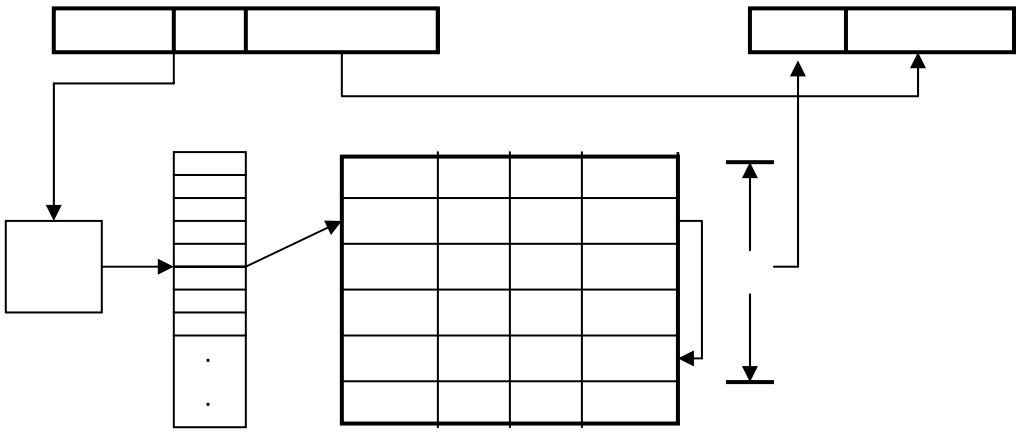


4-16 SPARC

4.3.6

IBM AS/400 Mac OS **IPT Inverted Page Table** IPT

4-17
MMU IPT
MMU
IPT



4-17

IPT 128MB 1KB
IPT 128KB IPT

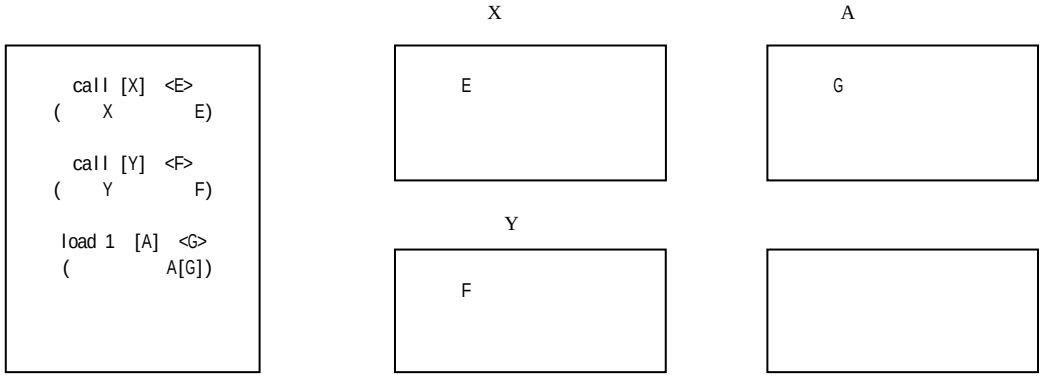
4.4

4.4.1

0

4-18

“ 0 ”



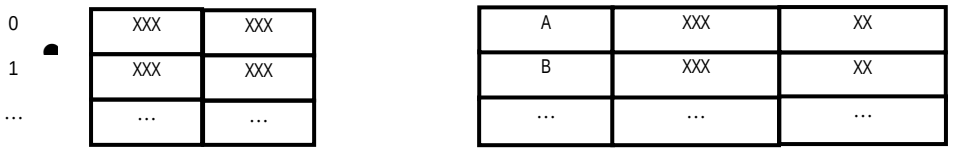
4-18

4.4.2

——

PDP-11/45 3 13
8 8K

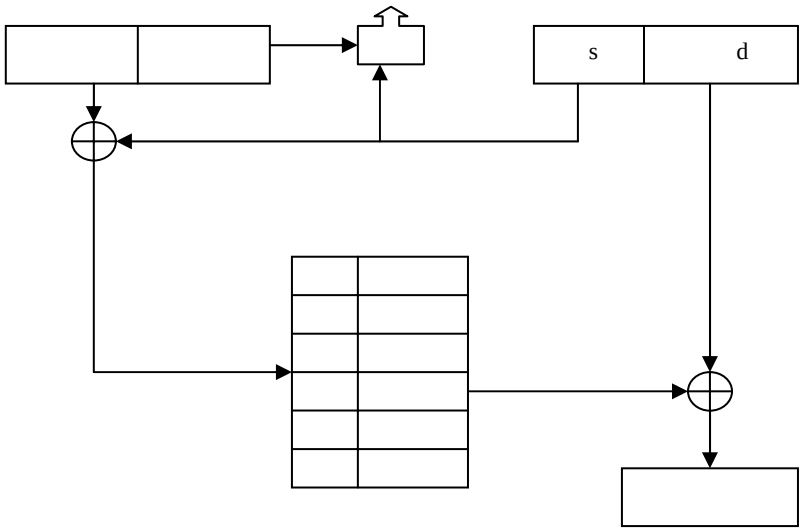
4-19



4-19

/

4-20



4-20

4.4.3

/

4.4.4

( )

( )

4.5

4.5.1

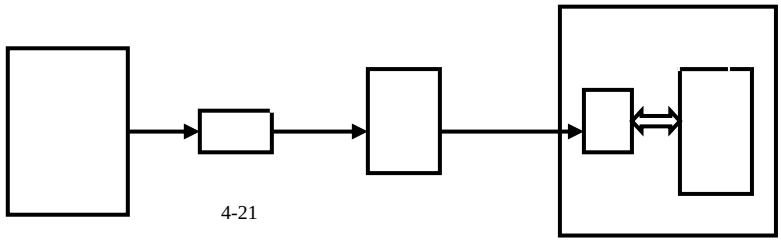
“ ”

“

” **virtual memory** (Fotheringham 1961)

“ ”

20 1MB Intel pentium
32 4GB Windows2000/XP
4GB 4-21



4-21

1968 P.Denning **principle of locality** Knuth Fortran Tanenbaum Huck

60 Atlas 60
MULTICS IBM
70

I/O

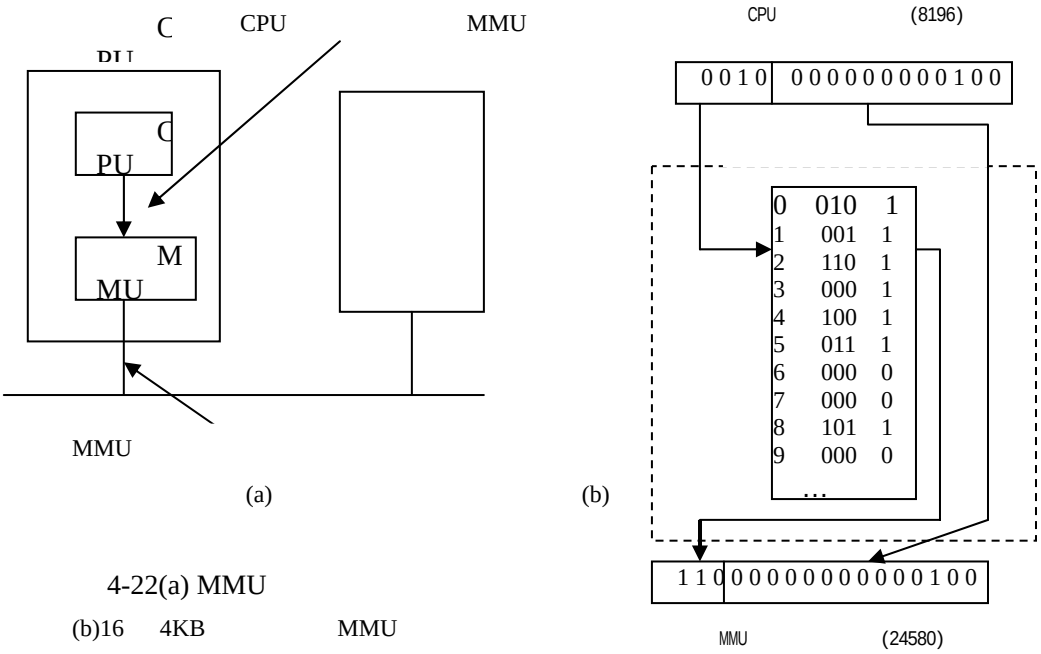
4.5.2

1

MMU Memory Management Unit MMU

4-22(a)

4-22(b)



4-22(a) MMU

(b)16 4KB MMU

- MMU
- TLB MMU TLB
  TLB

  TLB
- TLB
  TLB MMU
  TLB MMU
- MMU
- MMU

- MMU 4-22(b) 8196(
- 0010000000000100) MMU 16 4
  12
  ” ” 1
  12 15 (
  110000000000100) ” ” 0

2

demand
paging

4-23

4-23

0

CPU ( )
( )

Modified Renferenced

( )

( )

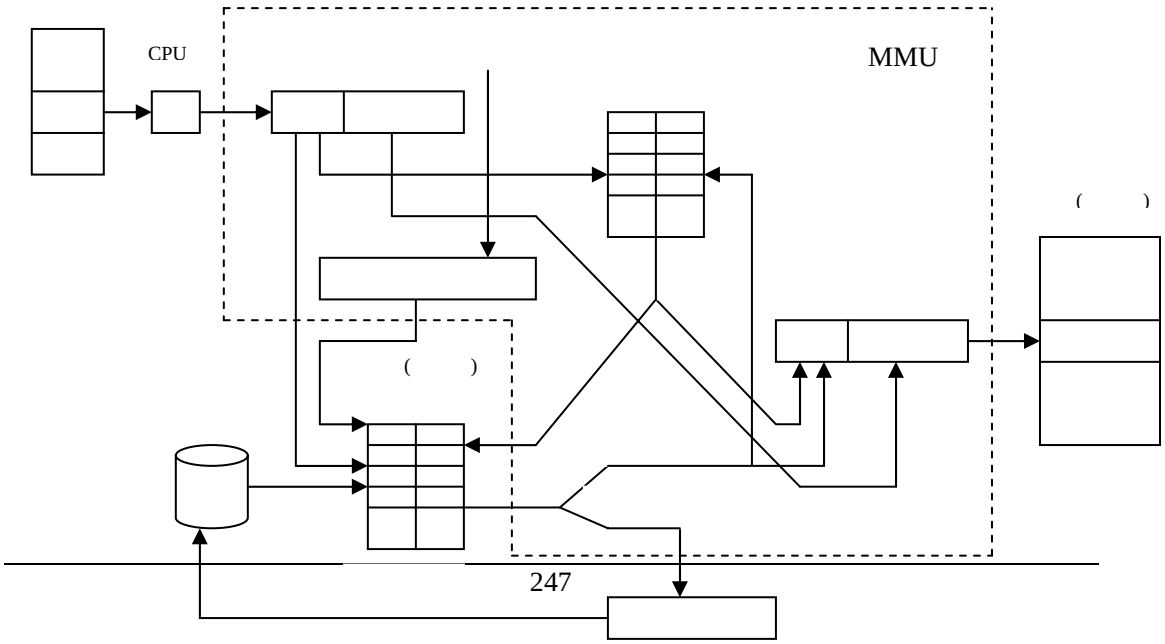
Linux -PAGE-PRESENT 1
-PAGE-RWFOE 0 1
-PAGE-USER 1 0
-PAGE-WT 1 (writethrough)
(writeback)
(Linux) -PAGE-PCD
I/O -PAGE-ACCESSED 1
-PAGE-DIRTY
-PAGE-4M pentiun 4MB
PFN

CPU
PCB
MMU 4-24
MMU CPU
TLB
TLB
TLB

TLB MMU
MMU

MMU

1
4
2
4 3
3 4
4



IBM/370 VS/1 VS/2
VM/370 Honeywell 6180 MULTICS UNIVAC 70/64 VMS

3

demand paging prepaging

I/O

I/O

50%

I/O CPU

4

I/O

5

“ ”

Thrashing “ ”

p n

m m n ≤ m ≤ n

m p S

F

A= S + F

f = F / A

f f

-
-
-
-

128 × 128 “ 0 ”

128

```
Var A     array[1..128] of array [1..128] of   integer;
      For   j     = 1 to 128
        do   for i     = 1 to 128
             do   A[i][j]   =0
```

[i][j] 128 × 128

1

```
Var A     array[1..128] of   array[1..128] of   integer;
      for i     = 1 to128
```

do for j = 1 to 128
do A[i][j] = 0
128 1

n n/2

Belady Belady

Optimal

Belady

1

2 **FIFO**

m

P[0] P[1] … P[m-1]
P[i] i=0 1 … m-1
k
P[k]
P[k] =
k = k+1 mod m;
m k 0 P[0]
P[0] k 1 … m P[m-1]
P[m-1] k 0 m+1
P[0]

FIFO

FIFO

2 3

3 LRU Least Recently Used

——

4 3 0 4 1 1 2 3 2

4	4	
3	4 3	
0	4 3 0	
4	3 0 4	
1	0 4 1	3
1	0 4 1	
2	4 1 2	0
3	1 2 3	4
2	1 3 2	

LRU

NUR(Not Recently Used) R
R 1 t R 0
R 0
R 0 t t
R 1 t R “ 0 ”

r
1 t r
r
r P0 P1 P2 T1 T2 T3 r

	T1	T2	T3
P0	1000	0100	1010
P1	1000	1100	0110
P2	0000	1000	0100

T3 P2 P0
P1 T3 T2

T1 P2

“ ”

LFU Least Frequently used

t

“ ”

4

FIFO FIFO FIFO

“ ” FIFO

“ ” 0

“ ” 1

“ ” 0

second chance

5 **Clock Policy**

FIFO

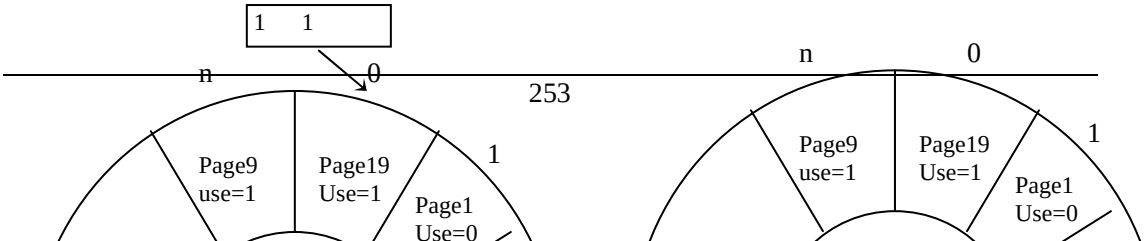
Clock

“ ”

- “ ” 1
- “ ” 1
- “ ” 1 “ ” 0 “ ” 0
- “ ” 1 “ ” 0

4-25

page727 page45 2 Clock page45 “ ” 1 “ ” 0 page191 3 “ ” 0 page556 4 “ ” 0 page556 page727 page727 “ ” 1 page13 5



“ ” “ ”

1 r=0 m=0
2 r=1 m=0
3 r=0 m=1
4 r=1 m=1

1
“ ” r=0 m=0
2 1 r=0 m=1
“ ” r
0
3 2 “ ”
r 0 1 2

Macintosh

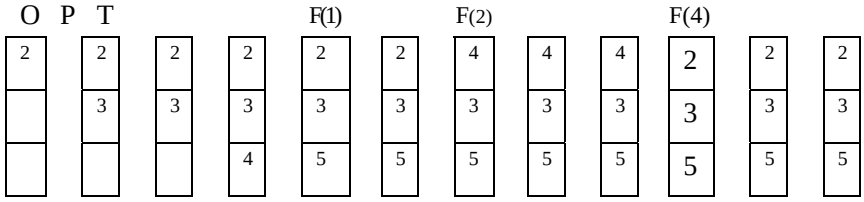
UNIX SVR4 clock clock
“ ”
“ ” 0 “ ” 1

“ ” 0
“ ” 1
“ ” 0
clock

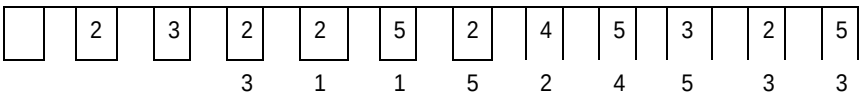
Opt FIFI LRU Clock

5 2 3 2 1 5 2 4 5
3 2 5 2 4-26
Opt 3 page1
page2 page4
LRU 4

page3 page1 page2 page4
FIFO
6
page2 page3 page1 page5 page2 page4
Clock 5
* “ ” 1 →
page5 “ ” 1
page2 page5 page2 page3 page1 “ ” 0
page2 page3 page2 page3
page4 page4 page1 page3
“ ” 1 page3 page5
page2 page3 “ ” 1 page2 “ ” 1 page4
“ ” 0 page2 page5 page4
FIFO Opt Clock LRU



L U F(3) F


2 3 2 2 5 2 4 5 3 2 5
3 1 1 5 2 4 5 3 3

2

3

+

I/O

4

512 4K

S P

S>>P e S/P Se/P

P/2

= + =Se/P+P/2

(Se/P) (

P/2) P P 0

$-Se/P^2+1/2=0$

P= 2Se

P S=128KB e=8B

1448 1KB

2KB 512B 4KB

Atlas 512 48 IBM370 2048

4096 VAX 512 IBM AS/400 512 Motorola 68040 Macintosh

4096 Pentium 4096 4 MIPS R4000 4096 16

7 CPU MMU

(2)

CPU

/ P.J.Denning

1968 working

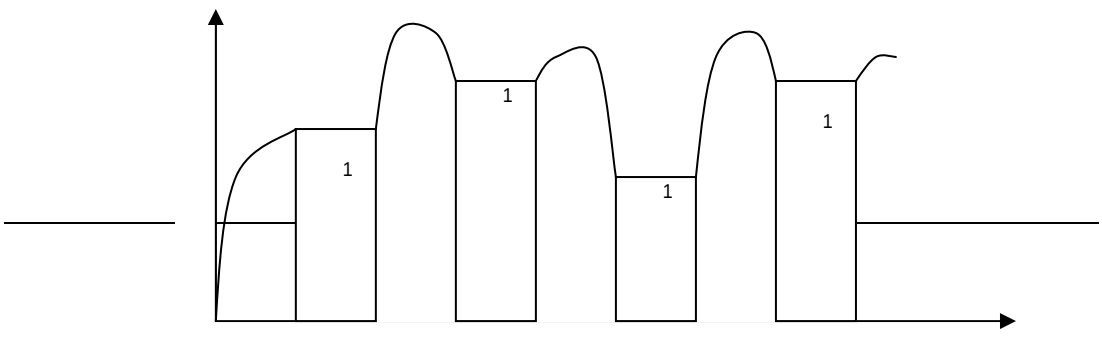
set Denning

“ ”

“

”

4-28



t- t

t W t “ ”

“ ”

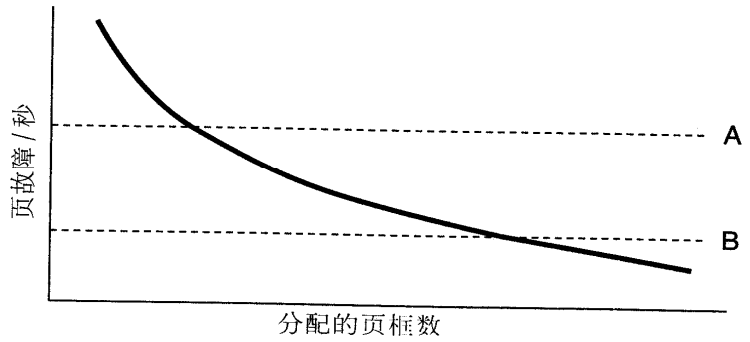
W t W t

	2	3	4	5
24 15 18 23 24 17 18 24 18 17 17 15 24 17 24 18	24 15 24 18 15 23 18 24 23 17 24 18 17 24 18 18 24 17 18 17 15 17 24 15 17 24 24 17 18 24	24 15 24 18 15 24 23 18 15 24 23 18 17 24 23 18 17 24 * * * * 15 17 18 24 15 17 * * 18 24 17	24 15 24 18 15 24 23 18 15 24 * 17 24 23 18 * * * * 15 17 18 24 * * * *	24 15 24 18 15 24 23 18 15 24 * 17 24 23 18 15 * * * * * * * * * *

W 4-29 W t

W W t

4-29 *



4-30

Page Fault Frequency PFF

LRU

4-30 A

B

PFF

A

(3)

UNIX/Linux

(4)

I/O

I/O I/O DMA

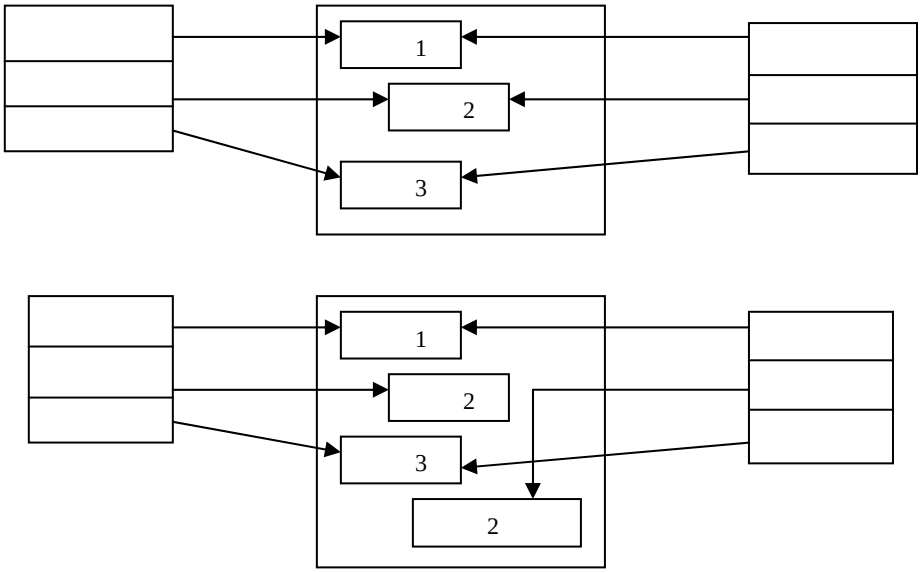
I/O

I/O

(5)

copy-on-write

UNIX Windows



4-31 ( ) ( )

“ ”

4-31

4.5.3

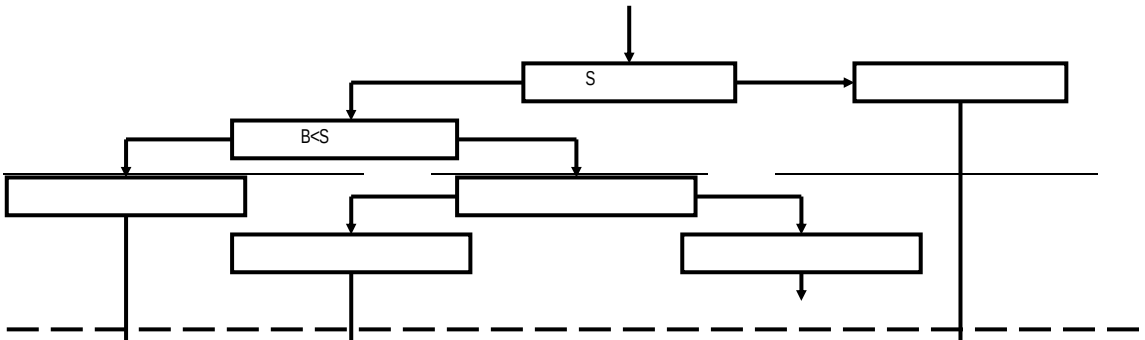
4-32

4-32

- 00 01 11
- 00 01 11
- 0 1
- 00 01 11
-
-

“ ”

4-33



1 “ ” 1

1

1

0

4.5.4

1
2

3

4

s d’

d’ p d d’=p × +d

5

6

s p p

p’ d

s d’8 ط ل

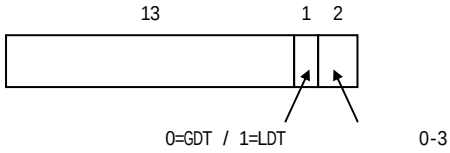
s s’ p s

8086
8086 386 CPU DOS
Windows Linux

4.6.1 Intel x86/Pentium

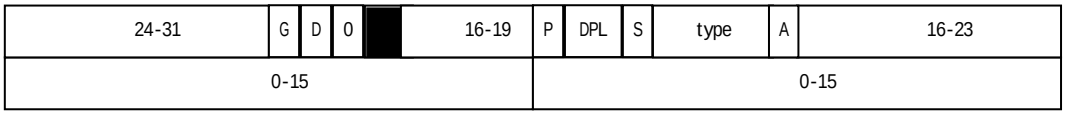
Intel x86/Pentium LDT local descriptor table GDT global descriptor table
LDT
GDT

Pentium
selector 6
CS DS
16 4-35
13 LDT GDT
16k GDT 8192
LDT 8192 LDT
LDT GDT



4-35 Pentium

GDT LDT
LDT GDT
8 4-36



4-36 Pentium

- 32 32
286 24—31 0 286
24
- 20
- G G=0
G=1 Pentium 4KB
2^{20} 2^{20} 4KB=2^{32}
- D D=1 32 D=0 16
- P P=1 P=0

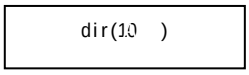
- Dpl 2 0—3 0 1
  2 3 Windows 95 0 3
- S S=1
  S=0
- type 3
- A

4.6.2 Intel x86/Pentium

Intel 286 MSW 1
0
386 CR0 0 PE PE 1
PE 0 CR0
31 PG PG 1 CR3
32 PG 0
32
(1) CR0 PE =1 PG =1
8192 2^{20} 4KB 2^{32}
(2) Intel “ ”

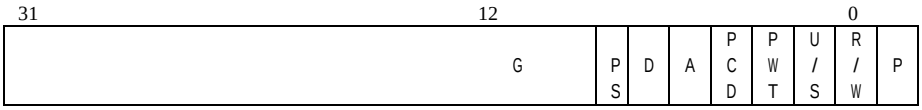
-
  LDT GDT
- P 0

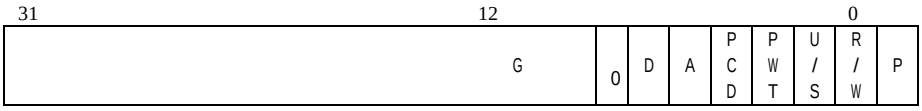
1024 4 32
10 0 1023
4KB page frame
32 12 0 4095
4-39 3 dir page
offset dir page
offset



(32)

4-40 32 20
/





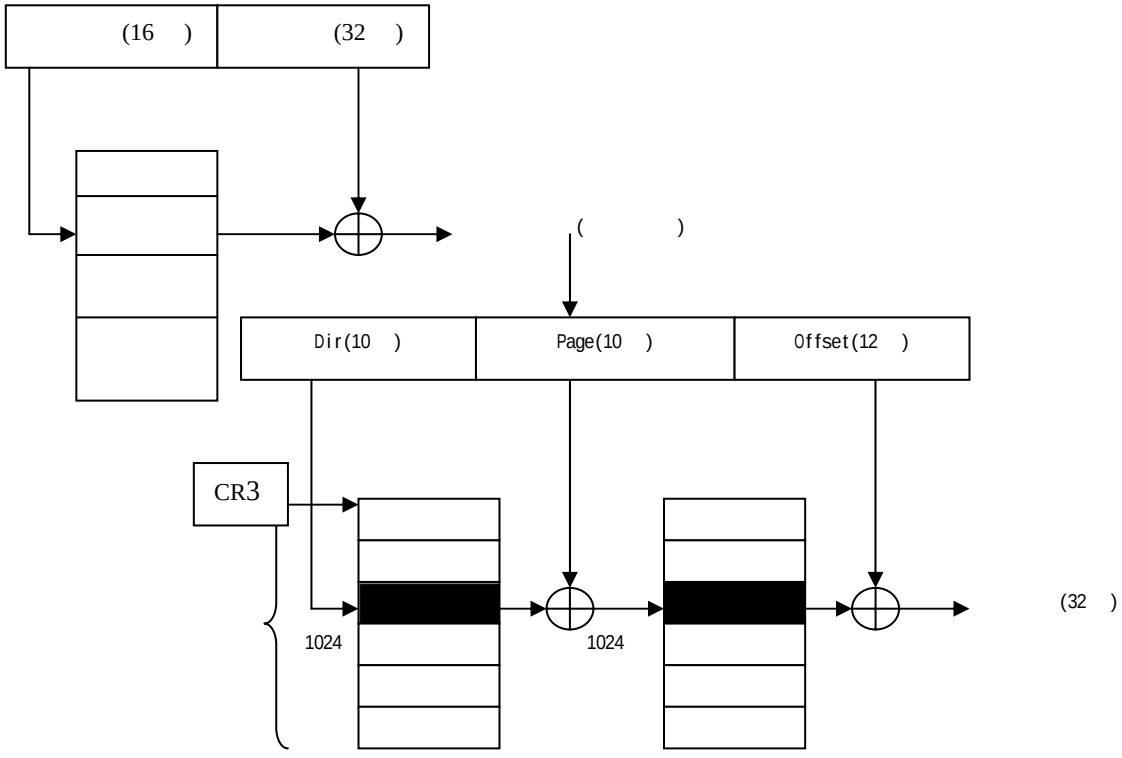
4-40

- G
- D
  0
- A / accessed
  A=0

- PCD cache
- PWT cache
- P P=1
- U/S / User/Supervisor U/S=0
  OS
- R/W / Read/Write R/W=1 R/W=0
  R/W=0
  1024 4K 4M
  4M

Intel x86/Pentium
4-41 Intel x86/Pentium
TLB dir/page

Intel x86/Pentium
80286



4-41Pentium CPU( )

4.7 Windows 2000/XP

4.7.1

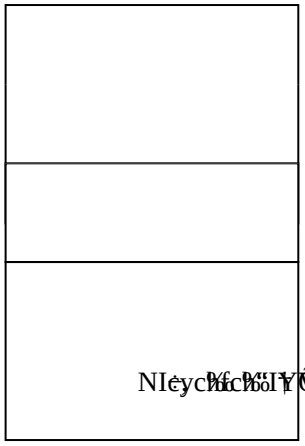
Windows 2000/XP VMM Virtual Memory Manager Ntoskrnl.exe Windows

——

VMM

Windows 2000/XP “ ” 386
32 4GB 2^{32}B 4 42
4GB 2GB
2GB Windows
3GB 1GB

FFFFFFFFH



NIeycñcbolTÕ8–Ebtc d4c

4.7.2

Windows 2000/XP

-
-
-

1

Windows 2000/XP 4GB

“ **” VAD** Virtual Address Descriptor
VAD
VAD
Self-balancing Binary Tree 4 43

Windows 2000/XP
VAD

VAD

VAD

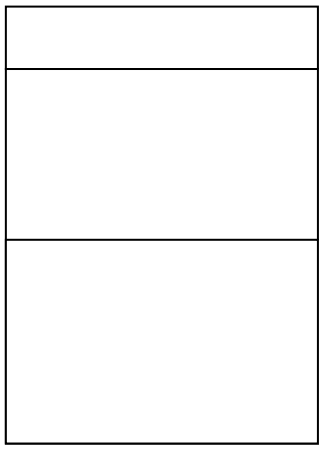
2

“ ” section object Win32 “ ”

-
-
-

4 44

-



4 44

3

2^{32}B 2^{64}B

Windows 2000/XP **AWE** Address Windowing Extension

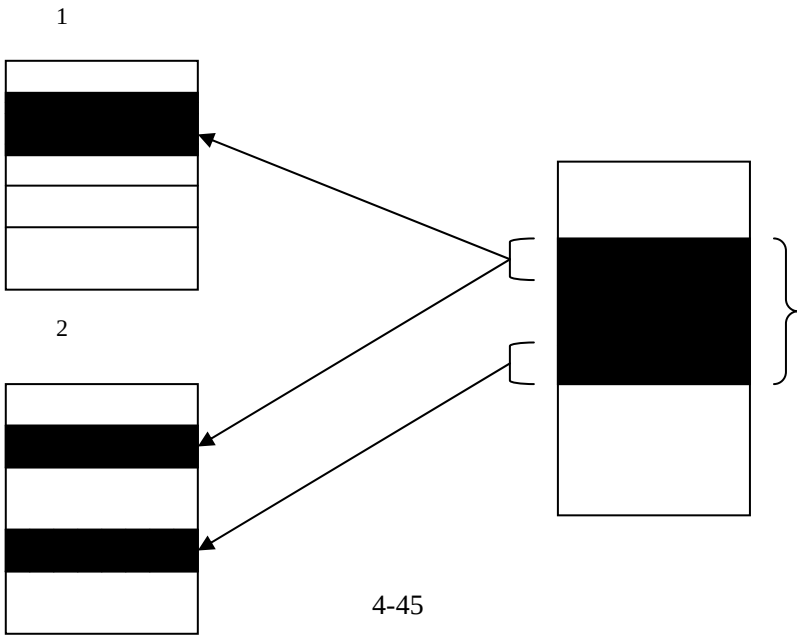
2^{32}B

2^{64}B 4 45 8GB

Windows2000/XP AWE

8GB AWE 2GB 3GB

32 2GB 3GB



4-45

AWE

1

2

3

-
-
-

Windows 2000/XP “ ”

4

1

Windows 2000/XP reserved memory committed memory (release memory)

1

Win32 virtual Alloc

	lpAddress	NULL

(bInheritHandle) (lpName)
5 UnmapViewOfFile
(lpBaseAddress)

3

heap

Ntdll Ntoskrnl.exe
Win32API Ntdll
Ntoskrnl
1KB
HeapCreate Heap Destroy

GetprocessHeap
HeapAlloc HeapFree

4.7.3

1

Windows 2000/XP x86
32 10
10 12 4KB

KPROCESS

CPU CR3

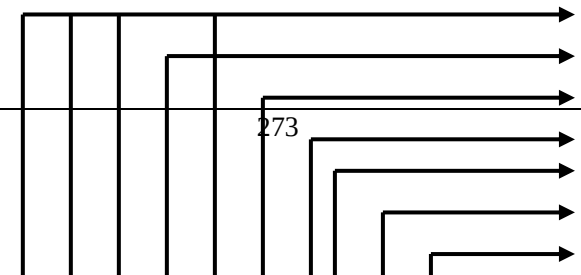
PDE(page directory entry) PDE 4
x86 CPU
(1024 PDE) 1024
1024 4GB 512
512 PTE(page table entry)
4-46

2

“ ” prototype PTE prototype page table Entry

PTE
PTE 4 47 PTE
PTE

PTE
PTE PTE

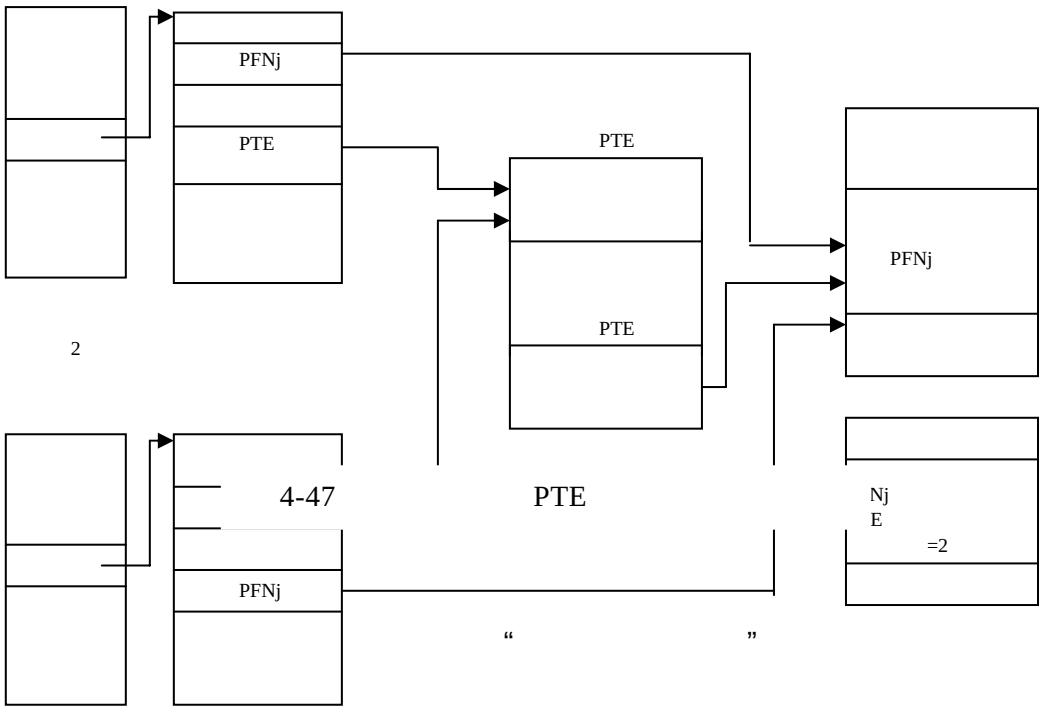


PTE 32

PTE

PTE

1



PTE 6

- active/valid
- transition

- modified-no-write
- demand zeropage
- page tile
- mapped file

3

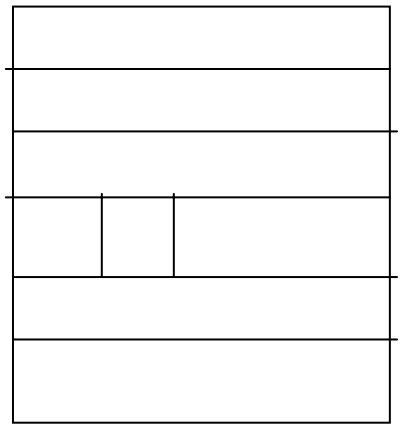
Windows 2000/XP Page Frame Database PFN Page Frame Number

4-48

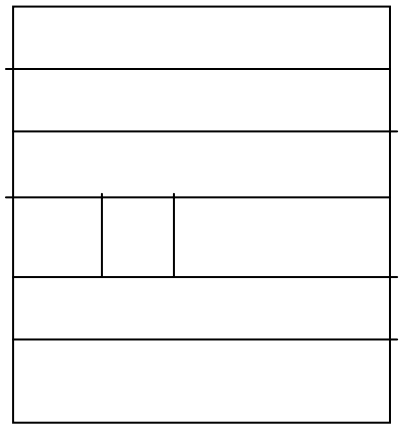
-
-

0

- PNF
- PTE



PFN



PFN

PFN 8

1 **Valid**

2 **transition**

I O

3 **Stand by**

4 **Modified**

5 **modified no write**

6 **free** (0)

7 **zeroed**

8 **bad**

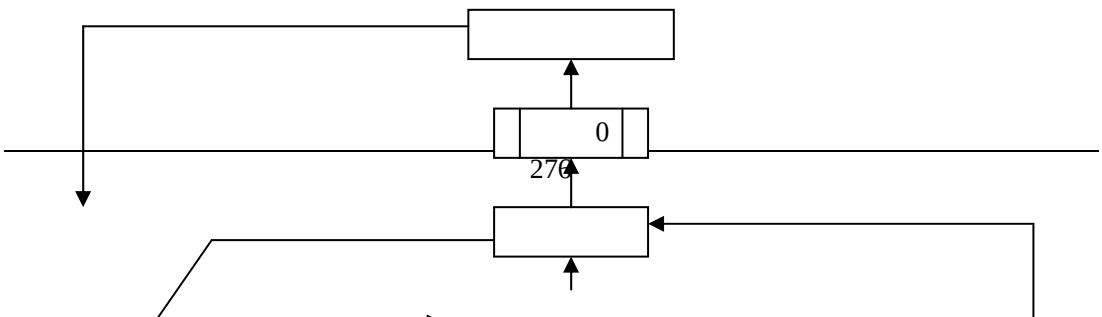
6

4 49

4

| > > >

|



MmMinimum Freepages “ ”

“ ”

PTE
PTE

5

Windows 2000/XP

I O
19MB 8
4 8
Windows 2000/XP FIFO
FIFO

Windows 2000/XP “
”

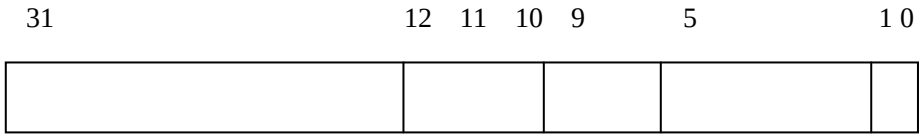
32MB Windows 2000/XP server
64MB 50 345

1
Windows 2000/XP

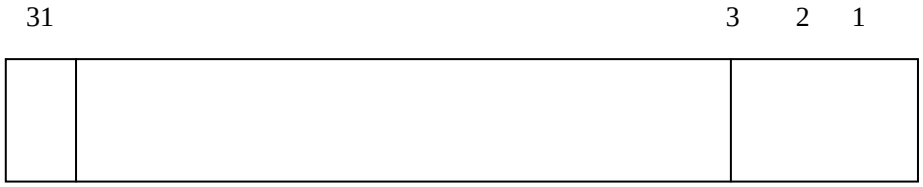
Win32 Set process working
Set

6

Windows 2000/XP 16
4-50 a
4-50 b



(a)



b

4-50

7

1

-
- Win32

8

2

allocation granularity
64KB
x86 4KB 18KB
x86 20KB

3

Windows 2000/XP
4

-

-
-
- ACL
  Access Control List ACL

4.8 Linux

4.8.1 Linux

Pentium Linux
Linux Unix Pentium Linux
4GB 0 3GB
3GB 4GB

3 0
Linux 3GB 4GB

3GB 3GB+4MB 0 4MB
3GB 3GB+4MB 0
4MB
Unix Linux PGD Page
Directory PMD Page Middle Directory PT

4GB 4MB

Linux vma virtual memory area

vma

vma

vm_area_struct

```
struct vm_area_struct{
  struct mm-struct *vm_mm          /*          */
  unsigned long vm_start           /*          */
  unsigned long vm_end             /*          */
  pgprot_t vm_page_prot
  unsigned short vm_flags
  /*              vma   AVL   *
  shrot vm_avl_height
  struct vm_area_struct *vm_avl_left
  struct vm_area_struct *vm_avl_right
  /*              vma          *
  struct vm_area_struct *vm_next
  /*                                                    *
  struct vm_area_struct *vm_next_share
  struct vm_area_struct *vm_prev_share
  /*              *
  struct vm_operations_struct vm_ops    *          open   close*
  unsigned long vm_offset          /*                  *
  struct inode *vm_inode           /*          inode   NULL*
  unsigned long vm_pte
}
```

vma Linux vma PCB task_struct mm_struct

mm_struct

vma mmap vm_next vma

pgd Linux

“ ”

4.8.3

linux

men-map free-area-init( )

men-map men-map-t men-map-t

```
typedef struct page{
    struct page *next   *prev        /*                    */
  struct inode *inode
    unsigned long offset
```

```
/*                                          inode          */
      struct page *next_hash       /*page          hash                    */
      atomic_t    count            /*                          */
      unsigned flags               /*              */
      unsigned dirty               /*                */
      unsigned age                 /*                          */
      struct wait_queue *wait
      struct page *prev_hash       /*pager         hash                    */
      struct buffer_head *buffers  /*                                  */
  unsigned long swap_unlock_entry
  unsigned long map_nr         /*          mem_map              */
}mem_map_t
```

mem_map bitmap free_area_init( )

bitmap NR_MEM_LISTS 6

0 end_mem−start_mem/PAGE_SIZE/2^{0+3} 2^0

end_mem−start_mem/PAGE_SIZE/2^{1+3} 2^1 1 2

i end_mem−start_mem/PAGE_SIZE/2^{i+3} 2^i 1 1

Linux free_area NR_MEM_LISTS free_area_struct

```
struct free_area_struct{
    struct page *next    *prev     /*          next    prev          struct page      */
    unsigned int *map            /*      bitmap    */
}
static struct free_area_struct    free_area[NR_MEM_LISTS]
```

bitmap free_area 0 2^i free-area i Linux buddy 2^i $0 \le i<NR\_MEM\_LISTS$ _get_free_pages( ) free_pages( ) 2^i free_area i i+1 free_area free_area bitmap i NR_MEM_LISTS bit 1 change_bit( ) bitmap bitmap free_area free_area

4.8.4

vmalloc( ) vfree( )

3GB high_memory HOLE_8M vmlist
3GB high_memory
3GB high_memory
HOLE_8M 8MB “ ” vmlist
0 3GB ∃ :

mlock munlock mlockall munlockall

vma

vma vm_flags PROT_READ PROT_WRITE PROT_EXEC vm_page_prot

mprotect

4.8.6

1

GB

“ ”

CPU CPU

70

CPU

Linux

Linux

CPU

Linux

4096

“ SWAP_SPACE ”

4096

4096-10 *8-1=32687

Linux MAX_SWAPFILES 8

inode

swapon

2

Linux

kswapd

kswapd

kswapd init init_swap_timer( )

10ms swap_tick( )

free_pages_high free_pages_low

kswapd kswapd

Kswapd

page cache buffer cache

3

CR2

Linux

do_page_fault( ) CR2

Linux vm_area_struct

P

P=0

Linux clock

8 age age 1

age

1 Age 0 “ ”

Age

4.8.7

Linux kmalloc cache swap cache page cache

1 swap cache

Linux swap-cache swap cache swap cache

Vnsigned long *swap-cache

swap-cache

0 0

swap cache

2 page cache

page cache Linux hash page-hash-table page cache

page

page cache

mem-map-t

Linux

page cache

page cache page cache

3 kmalloc cache

kmalloc( ) kfree( )
kmalloc cache

4.9

,

Optimal LRU Clock LFU Optimal
LRU

-
-
-
-

( )

,

1
2
3 ?
4
5
6
7
8
9
10
11
12 ?
13
14
15
16 /

17
18
19
/
20

25 (1) (2)
26 “ ”
27

28

29
30
31
32
33
FIFO LRU ( )
34
35 CPU20% 97.7%
50% CPU
36 (1)
(2) (3) goto (4) (5)
37

?

1
1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
FIFO OPT LRU 3 4 5 6

2 5
(1)1 4 3 1 2 5 1 4 2 1 4 5
(2)3 2 1 4 4 5 5 3 4 3 2 1 5
FIFO LRU

3 FIFO OPT LRU

(1)2 3 2 1 5 2 4 5 3 2 5 2
(2)4 3 2 1 4 3 5 4 3 2 1 5
(3)1 2 3 4 1 2 5 1 2 3 4 5
3 4

4 10K 4K 20K 18K 7K 9K
12K 15K (1)12K 10K 9K (2)12K 10K 15K 18K

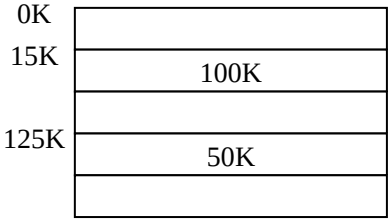
5 100K 500K 200K 300K 600K
212K 417K 112K 426K (1) first-fit best-fit worst-fit
?(2) ?
6 32 9 11

7 5 A B C D A B E A B C D E
FIFO 3 4
8 Ans
Bns
Cns Bns n-1
/n m-1 /m
9 cache cache
20ns cache 60ns
12ms 60ns cache
cache 0.9 0.6 (ns)
10 (1) 1.2
(2 80 × %

11

0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

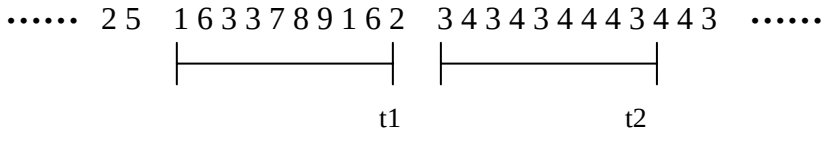
1 [0 430] 2 [3 400] 3 [1 1] 4 [2 500] 5 [4
42]
12 24 2^{18}B
1KB 11123456 100(
)
13



Job1 30K Job2 70K Job3 50K

14 16 2048
8
15 16 4096
2F6AH 0 1 2 15 18 21

16 VAR A ARRAY[1 100,1 100] OF integer
LRU 3 200
1
A
FOR i 1 TO 100 DO
FOR j 1 TO 100 DO
A[i,j] 0
B
FOR j 1 TO 100 DO
FOR i 1 TO 100 DO
A[i,j] 0
A B
17 48 32 8KB
? ?
18



9 t1 t2
19 CPU
(1)CPU 13% 97% (2)CPU
87% 3% (3)CPU 13% 3%
20 32 1KB 16KB
10 0 1 2 3 8 7 4 10
0AC5H 1AC5H
21 4 R D
( )

page	loaded	last ref	R	D
0	126	279	0	0
1	230	260	1	0
2	120	272	1	1
3	160	280	1	1

FIFO LRU CLOCK ?
22
for (i=0;i<20,i++)

for(j=0;j<10;j++)
a[i]=a[i] × j

23 1
5ms 75% 10%

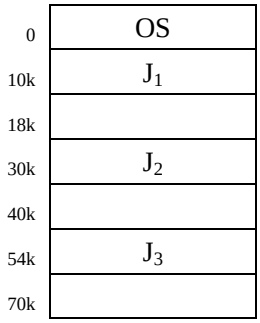
24 1024KB 4KB 512KB
10 11 12 13 62 78 25 36 0BEBC(16)
(16) ?

25
1) 120ns ?
2) 20ns 80%
?

26 J_1 J_2 J_3 20k J_4

(1) J_4

(2) J_4



27

0	200	500
1	890	30
2	120	100
3	1250	600
4	1800	88

1) <0,480>2)<1,25> 3)<1,14> 4)<2,200> 5)
<3,500>6)<4,100>

28 10,11,104,170,73,305,180,240,244,445,467
366 1) 100 2) 3 FIFO
LRU ?

29 2M 512K 512K
70% I/O 1M ?

30 4 I/O
CPU ?

31 1 n k

32 1024
500ns 32 100ns
200ns ?
33 1 2001
60 15000
?
34 FIFO

35 4 8
2KB (a) ?(b)
?(c) 5ABC
36 4KB 4B 64
1 ?
37 m p
n (n>m) FIFO LRU

38
2 8
1 60% 2
?
39 P1 P2 C1 C2
M1 M2

C1	C2	M1	M2
4KB	4KB	2MB	2MB
60ns	80ns	1μs	0.9μs

P 0.7
P=0.9

CH5

I/O

CPU

I/O

I/O

-
-
-
-
-

5.1 I/O

I/O I/O

I/O

I/O

5.1.1 I/O

I/O **I/O**

I/O

I/O

I/O

1

- KB
-
-
-

I/O

5.1.2 I/O

I/O

CPU I/O

I/O CPU

I/O

1

5-1 CPU I/O

CPU I/O

I/O CPU I/O

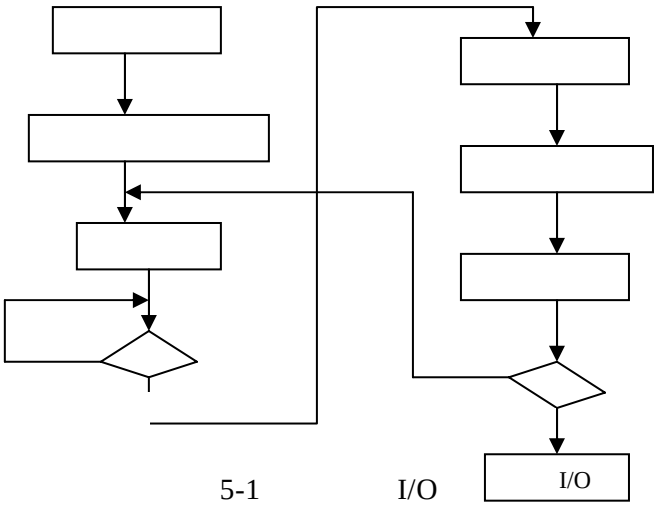
I/O CPU

I/O

(1) (2)

(3)

CPU



5-1 I/O

CPU I/O I/O

CPU CPU I/O

CPU CPU CPU

DMA 1

- 1
- DMA

DMA DMA

DMA

DMA

- I/O
- CPU I/O CPU I/O

DMA

DMA DMA DMA

CPU CPU DMA DMA “

”

DMA CPU CPU CPU

CPU

DMA CPU DMA

I/O CPU

CPU DMA

DMA

I O

4

DMA CPU I/O

CPU

I/O I/O

I/O

CPU I/O

I/O DMA CPU

I/O

CCW

-

IBM 370 256

-

-

5.1.3

I/O

2 4 8
ANSI IEEE ISO
IDE
SCSI

CPU CPU
CPU

16 512
4096 ECC

CRT
CRT

CPU
I/O I/O
I/O I/O
CPU IBM
I/O 15
IBM PC
IRQ
IRQ IRQ
PC I/O

I/O PC
15
CPU
I/O
CPU
CPU

PC I/O

I/O	I/O		
	040--043	0	8
	060--063	1	9
	1F0—1F7	14	118
	3F0-3F7	6	14
LPT1	378—37F	7	15
COM1	3F8—3FF	4	12
COM2	2F8-2FF	3	11

CPU

CPU I/O

CPU

CPU

CPU

I/O

5.2 I/O

5.2.1 I/O

I/O I/O

I/O

I/O

4

-

CD-ROM

-

- —— I/O CPU

I/O

READ

-

I/O

- I/O
- I/O
- I/O
- I/O

5.2.2 I/O

I/O

I/O

-
-

- I/O

-

PSW

1

“ ” “ ”

2

- “ ”

-
-
- CAW
- 1000
500 “ ”

3

“ ”

“ ”

4 “ ”

5.2.3

I/O I/O

n

I/O

5.2.4 I/O

I/O

-
-
-
-
-
-
-
-

I/O

(1)I/O

UNIX

/dec/tty00 i- major device number i- minor device number

I/O

UNIX I/O

rwx

(2)

ǐ

5.2.5 I/O

(1)

I/O

( ) I/O

C

```
count = write  fd  buffer  nbytes
```

write

I/O

C printf

write ASCII

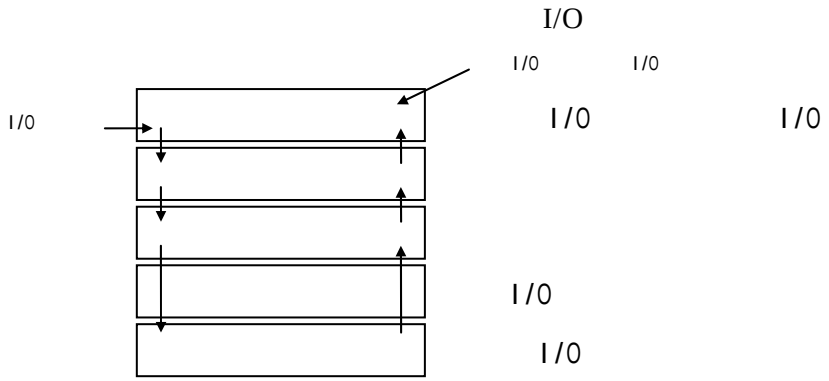
scanf printf I/O I/O

2 spooling

I/O spooling

I/O

5-4 I/O



5-4 I/O

5.3 I/O

CPU I/O

CPU

5.3.1

1

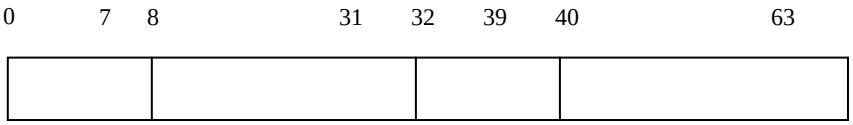
I/O I/O

CCW Channel Command Word I/O

I/O

I/O

IBM370 5-5



5-5 IBM370

-
-
“ ”
- 32 36

32 33 0 01

32 1

34 1

35 1 36 1

-
0

2

```
CCW   X’02’   inarea        X’40’   80
CCW   X’02’     *           X’50’   80
CCW   X’02’   inarea +80     X’40’   80
CCW   X’02’     *           X’50’   80
CCW   X’02’   inarea +160    X’40’   80
```

inarea DS CL240

*

X’50’ “ ”

3

I/O CAW Channel Address Word CSW Channel Status Word

8

I/O IBM

- 8
-
-
-

5.3.2 I/O I/O

IBM I/O I/O I/O I/O Start I/O SIO I/O Test I/O TIO Test Channel TCH I/O Halt I/O HIO Halt Device HDV

I/O

SIO X’00E’

0 0E

CPU I/O I/O

SIO 0 CAW

CPU 1

CSW 2

3

I/O I/O CPU

CAW

CPU I/O I/O

- I/O
-
- I/O
- I/O

IBM

```
START
        BALR    11    0
        USING    *    11
        SSM = X’00’                /*
        LA    8    READ0
        ST    8    CAW
        SIO    X’0182’             /*
        BC    7    *-4             /*
        TIO    X’0182’
        BC    7    *-4             /*
LOOP LA    8    READ1
```

```
        ST   8   CAW
        SIO   X'0182'          /*                  1
        BC   7   *-4
        TIO   X'0182'
        BC   7   *-4           /*
        LA   8   PRINT1
        ST   8   CAW
        TIO   X'00E'
        BC   7   *-4           /*              2
        SIO   X'00E'           /*                  1
        LA   8   READ2
        ST   8   CAW
        SIO   X'0182'          /*                  2
        BC   7   *-4
        TIO   X'0182'
        BC   7   *-4          /*
        LA   8   PRINT2
        ST   8   CAW
        TIO   X'00E'           /*
        BC   7   *-4           /*              1
        SIO   X'00E'           /*                  1
        B   LOOP

READ0   CCW   X'07'     *          X'20'    1
READ1   CCW   X'02'   BUFFER1     X'00'    512
READ2   CCW   X'02'   BUFFER2     X'00'    512
```

5.3.3 I/O

CPU CPU
I/O CPU CPU
I/O

- I/O I/O

CAW CPU SIO
CPU
I/O CPU

- I/O CAW
I/O

CPU I/O

CCW
CPU I/O
CPU I/O CPU

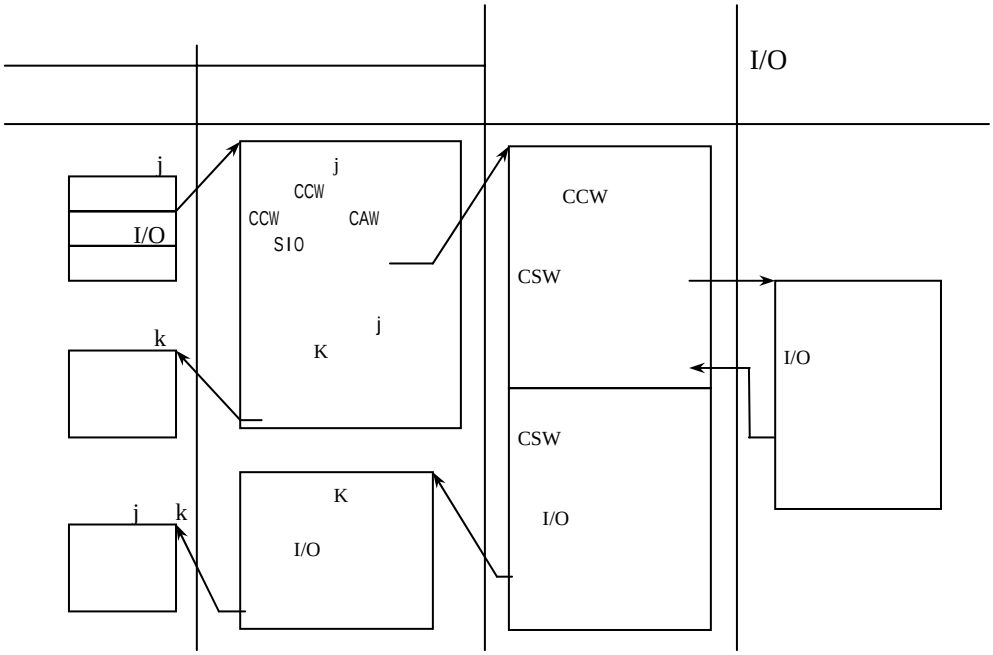
- I/O

CPU I/O

CSW CPU

I/O I/O 5-6

I/O



● 5-6 I/O

5.4

CPU I/O I/O CPU

CPU

I/O

I/O

I/O

CPU I/O

I/O I/O

I/O

5.4.1

I/O

T M

C

T+C max[C

T]+M M C T

I/O

5.4.2

I/O

buffer swapping 1 1

2 1 1

2 2

CPU I/O
UNIX 15 512
100 8

“ ” “ ” “ ” CPU
I/O

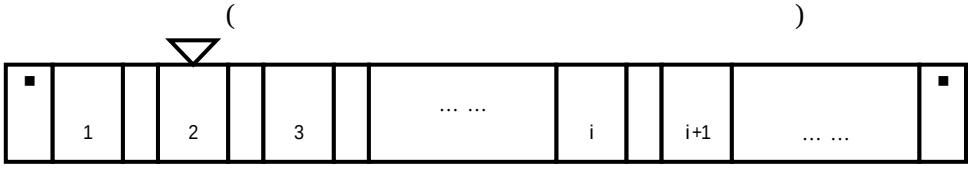
5-7 UNIX I/O

5-7 UNIX I/O
/ c-cc
c-cf c-cl
c-cf
e f c-cc c-cf c-cf
0 c-cf
c-next c-cf c-cl
c-cl
0
I/O
c-cl

5.5

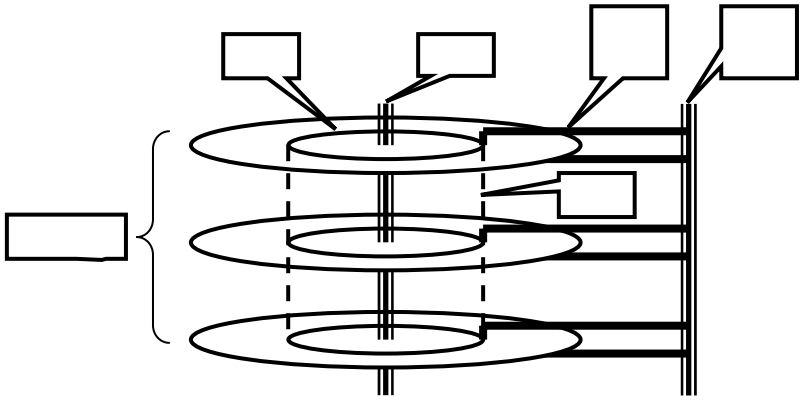
5.5.1

5-8



5-8

100 99
/



5-9

5-9

“
” 20
“ ”
10

5.5.2

4

1	4
2	3
3	2
4	1

- 1 4 3 2 1
1/2 1/4 4 3/4
3 1/2+1/4+3 × 3/4=3 60

- 2 1 2 3 4
  1/2+1/4+3 × 1/4=1.5 30
- 3 3 4 1
  2 3 20
  3

1 1 2
2 1 3
3 1 1
4 6 3
5 4 2

1 5 2 4
1 5

5.5.3

10 A
B …… J 10

1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	I
10	J

20
4 A D
B 10 10 A
+2 A +4 A +9 × [16 +2
+4] 214

1	A

2	H
3	E
4	B
5	I
6	F
7	C
8	J
9	G
10	D

A B
10 10 A +10 ×
[2 × 4]=70 3

5.5.4

8 20 A 1
1 A 10 A 1 1
1 5 “ ”

n “ ” n

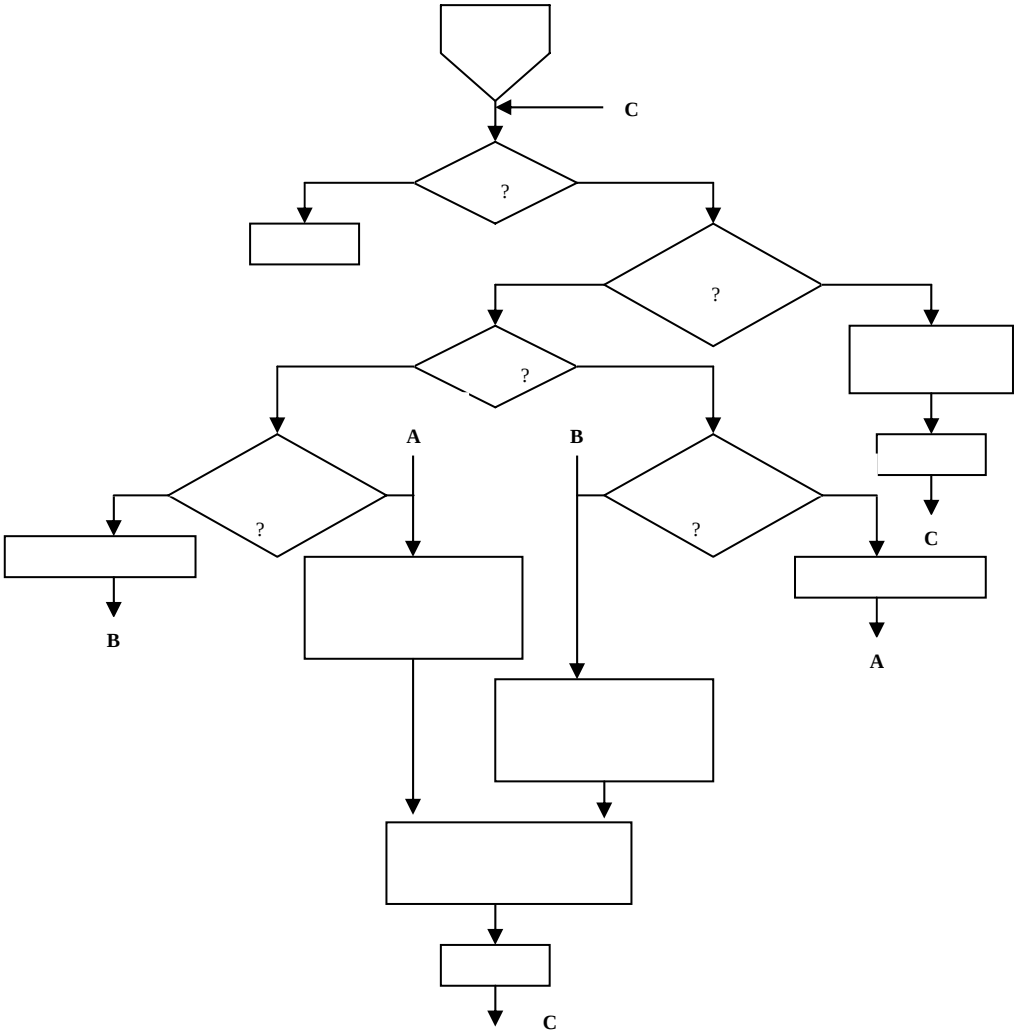
5.5.5

7 4 1
7 4 8
7 4 5
40 6 4
2 7 7

first-come first-served 0
0 7 40
2 2 7 7
7 40 7

7 4 1
7 4 5
7 4 8

7



1) “ ” 5-10 5-10

I/O

I/O

“ ”

I/O

I/O

2 “ ” **shortest seek time first algorithm**

“
”

78 40

“ ”

3 “ ” **scan algorithm**

I/O

“
” I/O “
” “ ”

4 “ ” **N-steps scan algorithm**

“ ” I/O

N

N “
” N=1 “ ”

5 “ ” **circular scan algorithm**

0

0

I/O

1 2

1 “ ”

“ ”

“ ” “ ”

5.5.6

RAID Redundant Array of Independent Disks 1987

Berkeley

scheme

RAID CPU

I/O

I/O

I/O

RAID 6 RAID0 RAID5 RAID6 RAID7

RAID

I/O

RAID

RAID0

strip

RAID1

RAID2 RAID3

Hamming code bit interleaved

I/O CAD RAID4 RAID5

block interleaved block

interleaved distributed I/O

1 RAID level 0

I/O RAID

level0 RAID

2 RAID level 1

RAID1 RAID2 RAID5 RAID1

RAID0

RAID1

3 RAID level 2

RAID2 RAID3

I/O

RAID RAID2 RAID3

RAID2

RAID2 RAID1

14 4

4 RAID level 3

RAID3 RAID2

RAID3

X0 X3

X4 i

X4 i =X3 i X2 i X1 i X0 i

X1 X4 i X1 i 2

X1 i =X4 i X3 i X2 i X0 i

RAID3 RAID4

RAID5

RAID

5 RAID level 4

RAID4 RAID5

I/O

I/O RAID4 RAID5

bit-by-bit

X0 X3

X4 X1

i

X4 i =X3 i X2 i X1 i X0 i

,

X4’ i =X3 i X2 i X1’ i X0 i

=X3 i X2 i X1’ i X0 i X1 i X1 i

=X4 i X1 i X1’ i

6 RAID level 5

RAID5 RAID4

n n

n

RAID4

7 RAID level 6 RAID level 7

RAID RAID6

RAID RAID7 RAID6

“ ”

( )

( )

5.6.2

I/O

I/O

I/O

( ) /

/ I/O /

( ) (/)

I/O

5.7

5.7.1

4

-
-

-

“ ”

“ ”

-
-

1

n n

80 100 80 20
2000 2000

0 2 101 1
n
└ n /100┘
n mod 100
n 100
2

1

3
CPU
?

5.7.3 SPOOLING

SPOOLING

1 spooling

daemon
spooling
spooling

2 spooling
spooling

Internet USENET

email send
spooling email

5.8 Windows 2000/XP I/O

5.8.1 Windows 2000/XP I/O

Windows 2000/XP I/O Windows 2000/XP
NTOSKRNL EXE I/O
I/O Windows 2000/XP I/O
I/O Win32 OS/2 POSIX
I/O

FAT CD-ROM (CDFS) UDF Windows 2000/XP (NTFS)

Windows 2000/XP I/O Windows 2000/XP I/O
I/O IRP
I O I/O I/O

Windows 2000/XP I/O
I/O

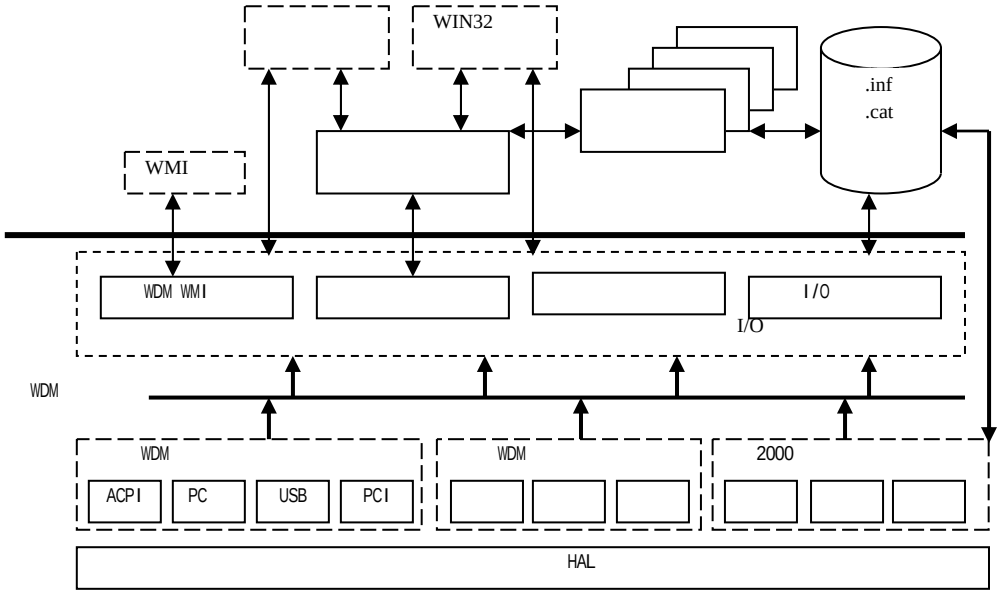
I/0

Windows 2000/XP

Windows 2000/XP I/O

I/O

I/O



5-12 I/O

Windows 2000/XP I/O 5-12

- API
- I O

  I/O

  I/O IRP
- I/O I/O

  I/O

  I/O
- PnP(plug and play) I/O
- I/O
- WMI(Windows Management lnstrumentation) Windows

  WDM(Windows Driver Model)WMI

  WMI
- WDM I/O

  WDM WDM

  I/O

  I/O
-
- (HAL) I/O

Windows 2000/XP

I/0 I/O I/O
I/O

2 PnP

PnP (Plug and Play) I/O
PnP

PnP
PnP PnP I/O
PnP
I/O PnP

I/O PnP
PnP “ ” “ ”
PnP PnP
(PnP
PnP BIOS) PnP I/O

PnP Windows 2000/XP
PnP
PnP USB USB
USB Windows 2000/XP PnP

- PnP
- PnP (resource arbitrating)
  (I/O)
  PnP
- PnP
  I/O
- PnP
  Windows 2000/XP

Windows 2000/XP PnP PnP
PnP
PnP PnP
PnP NT4
Windows 2000/XP PnP

PnP PnP
PnP PnP
PnP

PnP PnP
(start-device) PnP

PnP PnP

PnP query

remove remove PnP query—stop stop start—device

3

Windows 2000/XP PnP

ACPI(AdvancedConfigurationandPowerInterface)

BIOS(BasiclnputandOutputSystem) ACPI 1998

x86 ACPI

ACPI S0( ) S1 S3 S4 S5( )

- ●
- ●
- ●

S1 S4 S0 S1- S5

S0 S0 S1 S5

ACPI

DO D3 DO D3 Dl D2 D1 D2 D2 D3

Windows 2000 XP

S0 S1 DO D1

PoRegisterDeviceForldleDetection

PoRegisterDeviceForldleDetection PoSetDeviceBusy

5.8.2 Windows 2000/XP I/O

I/O I O (IRP)

1

Windows 2000/XP

Windows 2000/XP I/O

I/O Windows 2000/XP

( ) Windows 2000/XP I/O

	I/O I/O

I/O

(ACL) I/0

ACL

Win32 LockFile

2

I/O

I/O

- I/O

I/O ( )

-

I/O

I/O

Device

Floppy0 myfile.dat A myfile.dat kDevice
Floppy0 Windows 2000 XP
myfile.dat I/O Floppy0

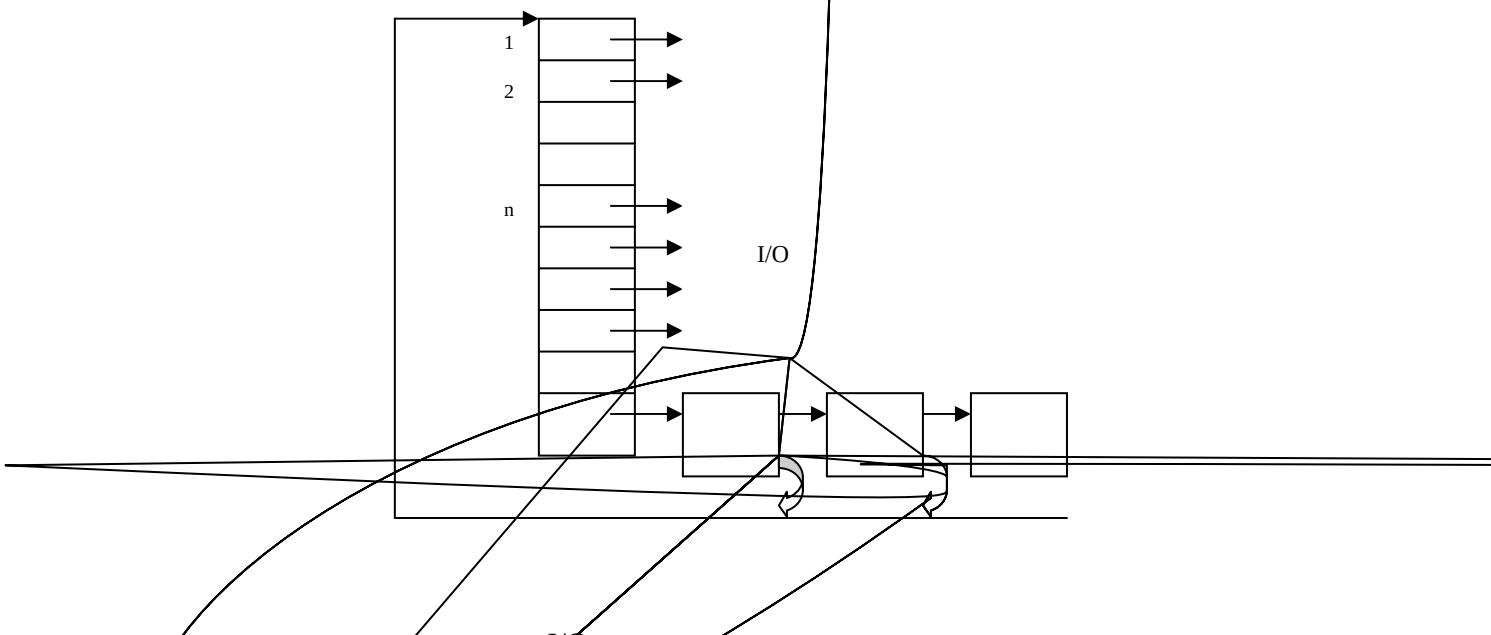
I/O Floppy0 Win32
Windows 2000/XP

Windows 2000/XP

Win32

Win32 QueryDosDevice DefineDosDevice

5-14 I O
I/O



I/O

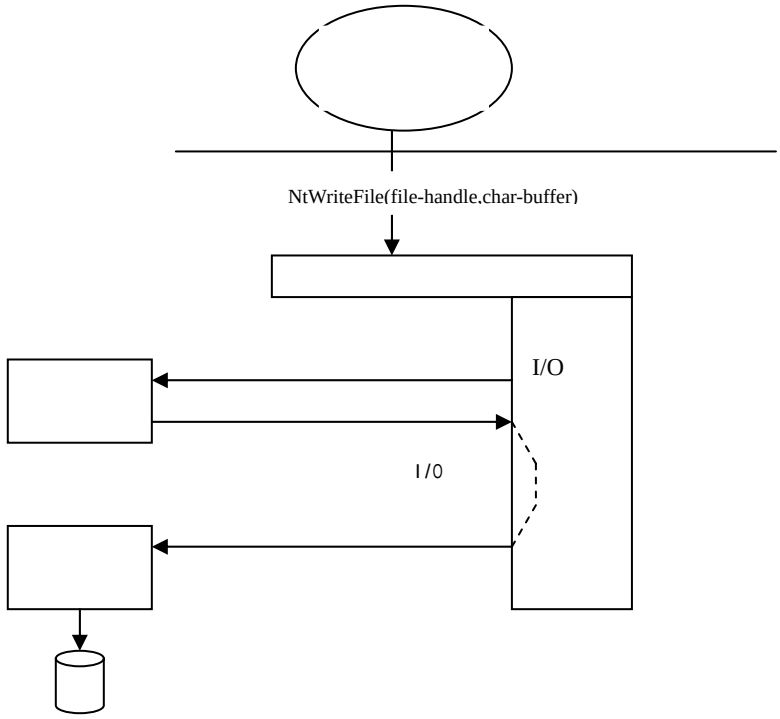
3 I/O

IRP(I/O Request Packet) I/O I O
I/O I/O IRP I/O
I/O IRP
IRP ( )

I/O

IRP

—



5-15

5-15 I/0

( )

I/0

( )

I/0

Windows 2000/XP

1

I/O 5-16

- I/O

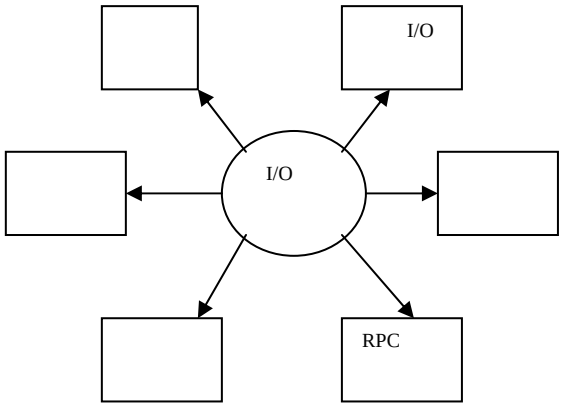
I/O

- PnP
-

I/O I/O IRP

- I/O I/O

- (ISR)
  Windows 2000/XP I/O ISR
  (IRQL)
  ISR IRQL (DPC)
  (ISR
  ISR)
- DPC DPC ISR
  DPC ISR IRQ
  DPC I/O I/O



5-16

- IRP
  I/O
- I/O I/O
  I/O I/O I/O
- I/O
-
- ( )
  I/O I/0

2

- ( )
- Windows 2000/XP

IRQL I/0

ISR

ISR

ISR

Windows 2000/XP

ISR

(CPU)

ISR Windows 2000/XP

ISR CPU IRQL

“ ” CPU

ISR

( )

ISR

5.8.4 Windows 2000/XP I/O

I/0 I/O

I/O

I/0 I/O I/O

1 I/O

I/0 I/0 I/O

I/O

1) I/O I/O

I O “ ”

I/O ReadFile

WriteFile

I/O

“ I/0 ” I/O

I/0 I/O

I/O Win32 CreateFile

FILE_FLAG_OVERLAPPED I/0

I/0

(I/O)

I/O I/O

I/O IRP I/O

I/O I/O I/0

I/O Win32 HasOverlappedToCompleted

I/O

2) I/O

I/O I/0 IRP

I/O

3) I/O

I/O I/O I/0

“ I/O ”

I/O

Win32 CreateFileMapping Map ViewOfFile I/0

I/O

( ) I/0

I/O

Windows 2000/XP

(

)

4) / I/O

Windows 2000 XP I/0 “ / ”

(scatter/gather) Win32 ReadFileScatter WriteFileScatter

/ I/O I/O

I/O

2 I/O

I/O

(1)I/O DLL

(2) DLL I/0 NtWriteFile

(3)I/O IRP

(4) I/O

(5) CPU

(6)I/O I/O

I/0

1)

I/O Windows 2000/XP

I/O

I/O

ISR Windows 2000/XP ISR

ISR IRQL

DPC

DPC

I/O I/O IRP

I/O

DPC IRQL

Dispatch/DPC IRQL DPC

2) I/O

DPC I/0

I/O “ I/O ” (I/O completion) I/0
I/O “ I/O
” (I/O status block) I/O I/O

I/O
I/O

I/O (APC)

APC DPC
DPC DPC APC

IRQL APC
I/O APC ( )
I/O IRP (
I/O) I/O (
)

I/O I/O ReadFileEx
APC I/O I/O
APC I/0
Platform SDK APC

5.8.5 Windows 2000 XP

1 Windows 2000 XP

Windows 2000 XP
Windows 2000 XP (
) Windows 2000 XP

Win32 CreateFile
Windows 2000 XP
I O Windows 2000 XP

I 0
Windows 2000 XP

Windows 2000 XP
(1)
Windows 2000 XP
CD-ROM
( )
(metadata)( ) Windows 2000 XP

(2)
Windows 2000 XP

(section object)

I O (1RP)

(3)

( )
(Win32 MapViewOfFile)
Windows 2000 XP

1 2
1 2
(4)
(NetWare OpenVMS OS 2 UNIX
) (logical block)
Windows 2000 XP
(virtual block caching)
256KB

-
- I O (I O)

I O
(5)
Windows 2000 XP
NTFS
NTFS
Windows
2000 XP

(6)
(recoverablefile system) NTFS
I O
I O
(log
file) ( )

-
-

-
-

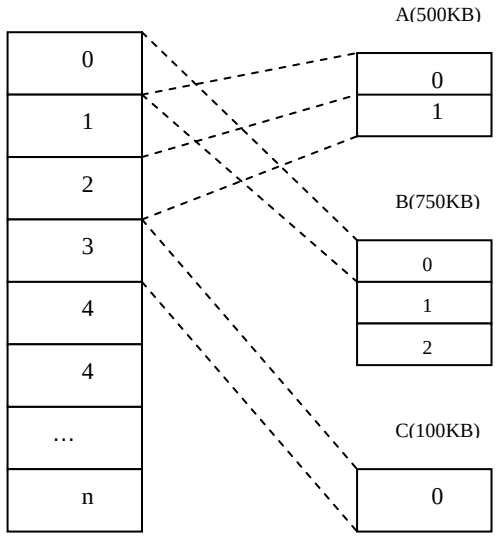
2

Windows 2000 XP

256KB (slot) (view)
I O( )
256KB 256KB
300000 10
262144 (256KB) 256KB

256KB 256KB
5-17 B A C
B B 256KB
C 100KB(
) 256KB

FILE-FLAG-RANDOM-ACCESS CreatFile



— I O

5-17

3

Windows 2000 XP (
) ?
Windows 2000 XP

1)

64MB
4032 (16MB) 128MB
16MB 4MB 64MB 64MB
128MB+(64MB-16MB) 4MB∗64MB=896MB
x86 2GB (MmSystemCacheStart)

(MmSystemCacheEnd) 64/960MB(
0xC1000000-E0FFFFFF) 512MB

2)

Windows 2000 XP

“ ”(
)

4

(1) 256KB VACB
(2)

(3)

1)

VACB(virtualaddresscontrolblock)

VACB CcVacbs VACB
VACB 256KB VACB
5-18
VACB

(256KB)
VACB I
O VACB I O

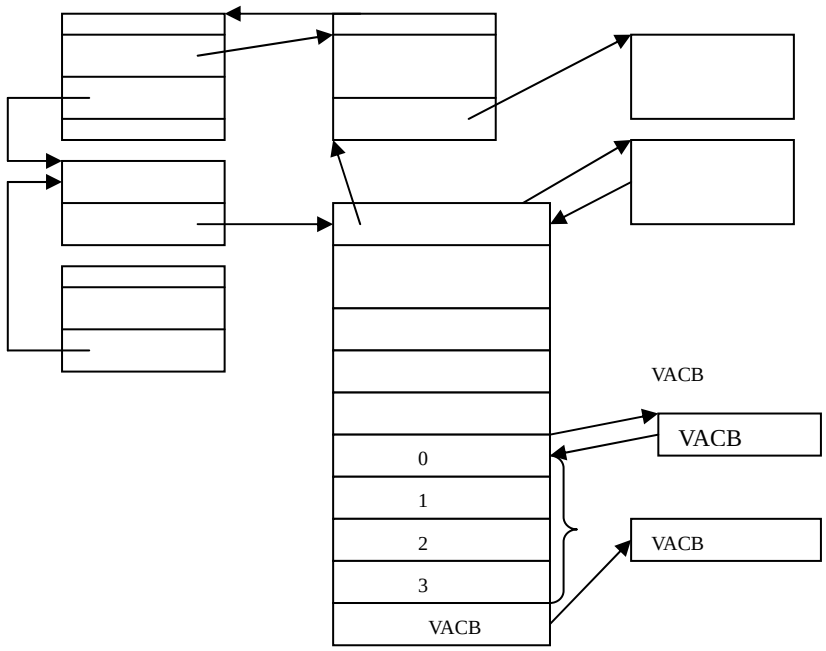
5-18 VACB

2)

(

)

VACB 5-19



5-19

? VACB ?

VACB

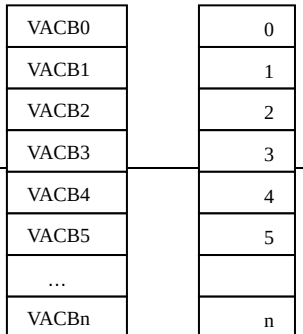
VACB “ VACB ” VACB

256KB 256 KB 5-20

A

VACB

VACB



VACB

(VACB) VACB

VACB

4 VACB
VACB 256KB 1 MB
1MB VACB
256KB(1)

32GB VACB
(sparse multilevel index array)
128
(-18) 7
18 VACB 256KB 256KB 2^{18}B
7 128 2^{7} 128 2^{63}(
) 7
VACB

32GB 256KB
VACB 3
32GB 2^{35} 3
VACB
128000 1000
5

I O (Win32 CreateFile)
FILE_FLAG_NO_BUFFERING

1)
Windows 2000 XP (lazy writing)

(write-back)

I O

I O

(

)

—— —

— 1 8 “ ” dirty ( )

2)

(threshold)

HKLM SYSTEM CurrentControlSet Control SeSSiOnManager MemoryManagement LargeSystemCashe Windows 2000 XP Professional 0 Windows 2000 XP Server 1 Windows 2000 XP Server

Windows2000 XP Professional

8

4 4MB ( )

2MB

3)

Win32 CreateFile FILE_ATTRIBUTE_TEMPORARY

——

4)

CreateFile

FILE_FLAG_WRITE_THROUGH

Win32FlushFileBuffers

5)

(

)

(Win32FlushFileBuffers)

“ ”

-
-

6)

Windows 2000 XP

(intelligent read-ahead)

—— Cache

ReadAheads Sec CcReadAheadlos

7)

(virtual address read-ahead)

30Tc<02c[(300)-5.200

8)

I O

“ ” (asynchronous read-ahead with history)

4000 3000

9)

I 0

( )

( )

-
- ( )

32 64 Windows 2000 XP Professional 128 Windows 2000 XP Server 256

10) I O

I O(fast I O)

I O I O (1RP) I O I O I O

IRP

Windows 2000 XP

IRP

I O

( )

I O

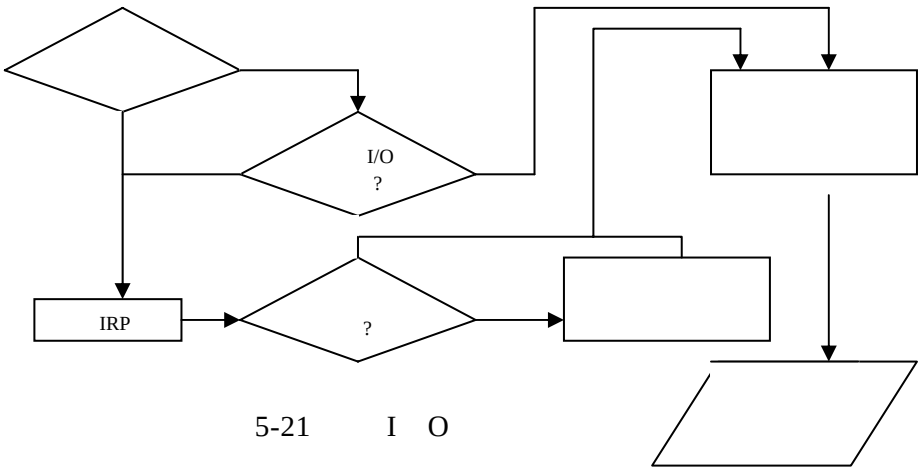
I O I O

I O I O

(Win32 LockFile UnlockFile)

IRP I O

5-21



5-21 I O

I O

1)
2) I O I O
I O
I O
3) I O I O
(I/O
I/O I O IRP
)
4)
5)

6)
- ●
- ● “ ”
- ●

6

CclnitializeCacheMap

(1) “ ”

(2) “ ”
(3) “ ”
——

( ) IRP “
”

1)

( )

CcCopyRead CcFastCopyRead CcCopyWrite CcFastCopyWrite	CcCopyRead 32 NTFS FAT CcCopyWrite 32

	NTFS FAT

2)

( )

CcMapData CcPinRead CcPreparePinWrite CcPinMappedData CcSetDirtyPinncdData CcUnpinData	/

(
)

3)

(direct memory access DMA)
DMA

DMA (
)
(1KB 2KB)
DMA

(MDL) 4
DMA
DMA

CcMdlRead	MDL
CcMdlReadComplete	MDL
CcMdlWrite	MDL 0
CCMdlWriteComplete	MDL “ ”

7

Windows 2000 XP

9600

CcSetDirtyPageThreshold —

5.9 Linux

5.9.1 Linux

Linux

Linux

Linux

mknod

Linux

Linux I/O

I/O

- Linux
-
-
-
- Linux Linux

●

Linux polling

Linux

Linux

I/O Linux

I/O Linux

/proc/interrupts

SCSI I/O Linux DMA I/O

7 DMA

16 8 16M DMA

DMA

DMA Linux dma_chan DMA

cat/proc/dma dma_chan

Linux

Linux

Linux

device_struct fs/devices.h chrdev

Linux chrdev device_struct

Linux blk_devs

chrdev blk_devs blk_dev_struct

blk_dev_struct

request request

5.9.2 Linux

Linux DOS EXT2 Linux

Linux Linux

gendisk

/include/linux/genhd

Linux IDE Inergrated Disk Electronic SCSI

Small Computer System Interface I/O Linux

IDE IDE IDE

IDE IDE

IDE Linux 4 IDE

ide_hwif_t ide_hwifs

ide_hwif_t ide_drive_t IDE IDE

Linux CMOS

Linux IDE

/dev/hda /dev/hdb /dev/hdc … IDE Linux

IDE IDE 3 IDE

22 IDE blk_devs
3 22
SCSI SCSI SCSI

32
40MB/ SCSI
8
1 BUS FREE
2 ARBITRATION SCSI SCSI
SCSI SCSI
3 SELECTION SCSI
SCSI SCSI

4 RESELECTION SCSI
SCSI
5 COMMAND 6B 10B 12B
6 DATA IN DATA OUT
7 STATUS

8 MESSAGE IN MESSAGE OUT
Linux SCSI host device
Host SCSI SCSI
SCSI host Device SCSI SCSI
Device

5.9.3 Linux

Linux device

Device
1
0 /dev/ethN /dev/seN SLIP /dev/pppN PPP
/dev/lo
2 irq
base address DMA DMA
channel
3
IP
4 mtu
internet AF_INET
Linux
X.25 SLIP PPP Apple Localtalk
5 sk_buff

6

5.9.4 Linux

Linux

UNIX SVR4 DDI/DKI(Device-Driver Interface/Driver-Kernel Interface -
/ -) Linux
DDI/DKI

- I/O
  file-operations
  lseek() read()
  write() select() mmap()
  open() release() fsync()
-
-

sys-setup() sys-setup() device-setup()

xxx-init()

open() lp-open() hd-open()
( )
release()
read() write() block-read()
block-write()

ioctl() cdrom-ioctl()

5.10

CPU I/O
CPU I/O CPU
I/O

I/O
I/O I/O CPU
DMA

CPU CPU CPU

DMA
CPU

CPU I/O DMA CPU
I/O
CPU I/O I/O
I/O
I/O I/O

I/O I/O

I/O
I/O
I/O I/O
Spooling
I/O
CPU
CPU I/O CPU
CPU I/O
CAW CSW I/O I/O
CPU I/O
I/O

I/O I/O
cache

cache
cache

I/O
RAID

I/O
I/O I/O

spooling ( )

spooling

1
2 I/O
3 DMA
4

5 I/O
6 I/O
7 I/O
8 CPU ?
9
10
11 I/O
12
13
14
15
16 “ ” “ ”
17 “ ”
18
19
20
21 Spooling
22 Spooling
23 Spooling ?
24 Spooling ?
25 Windows 2000/XP I/O
26 Windows 2000/XP
27 Windows 2000/XP
28 T
M C
max C T +M
29
30 IBM370 I/O
31
32
33 ?
34
35

1 I/O 20
20 1
2 1
1 …… 20 20 2 20

2 8 18 27 129 110 186 78 147 41 10 64 12

100

3

4 5

512 50 121 75 80 63 1569

?

5

1	7	2	8
2	7	2	5
3	7	1	2
4	30	5	3
5	3	6	6

1

6 40 0~39 11

1 36 16 34 9 12 1 FCFS 2

SSTF 3 SCAN

7 200 0~199 143

125 86 147 91 177 94

150 102 175 130

(1) FCFS

(2) SSTF

(3) SCAN

(4)

8 FCFS (1)

(2) (3)

9 100

190 10 160 80 90 125 30 20 29 140 25

10 100

23 376 205 132 19 61 190 398 29 4 18 40

11 L B (1)

?(2) K I/O ?

12 200 20 8 1024B

606

13 800 160 0.6
1500 50

14 800 200 0.6
3200 8

15 40 6ms
13 2
100ms 25ms 100

16 10 22 20 2 40 6 38
20 6ms 1)FCFS 2)
3) ( )

17 75 80

	1	2	3	4	5	6	7	8
	160	40	190	188	90	58	32	102

(1)
(2)

18 640 × 480 256

19 n 0 1 2 … n-1 m 0 1
2 … m 1 k 1 1 2 … k x
a,b,c
x a,b,c
x=k × m × a+k × b+c
a=(x-1) DIV (k × m)
b=((x-1) MOD (k × m)) DIV k
c=((x-1) MOD (k × m)) MOD k+1
n 200 m 20 k 10
(1) 185 12 5
(2) 1200
(3) 0 0 1 … k-1 x
a b c

CH6

“ ”

( )

( ) (
)

-
-
-
-
-
-
-

6.1

6.1.1

“ ”

;

6.1.2

“ ”

“ ? ”

“ * ”

MS-DOS 1 8
0 3

Windows
\ / < > ”
COM EXE
LIB BAT OBJ

MS-DOS
Windows-98 UNIX
“ asm ” C
“ c ” 255
Windows UNIX

6.1.3

UNIX
●

-
- I/O
- I/O

ASCII ASCII
DOS Windows CTRL+Z
UNIX CTRL+D ASCII

I/O

6.1.4

-
-

ASC

-
-
-

DOS XCOPY
BACKUP RESTORE Windows UNIX compress tar
bar arj lha winzip

-
-

socket

-

UNIX/Linux

r w x 10

-rwxrwxrwx

- 1 (-) (d) (l) (b/c)
- 2-4
- 5-7
- 8-10

-rwxr-x--x

6.1.5

IBM

1

i

× i

2

1024 4096
22 48 9

3

6.1.6

JCL UNIX cat cd cp find mv rm mkdir rmdir —

- ‘ ’
- / /
  /
-
-
-

6.2

6.2.1

“ ”

“ ”

FCB(File Control Block)

-
-

Ń z z

-

- di-size
- di-add[8]
- di-atime
- di-mtime
- di-ctime

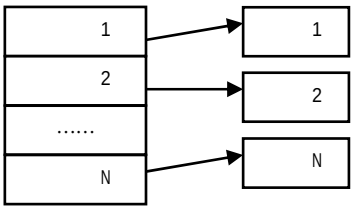
inode

UNIX/Linux —— inode
( ) 100
inode inode
inode inode inode inode
inode
inode inode inode

6.2.2

6-1

CP/M
32
0
CP/M

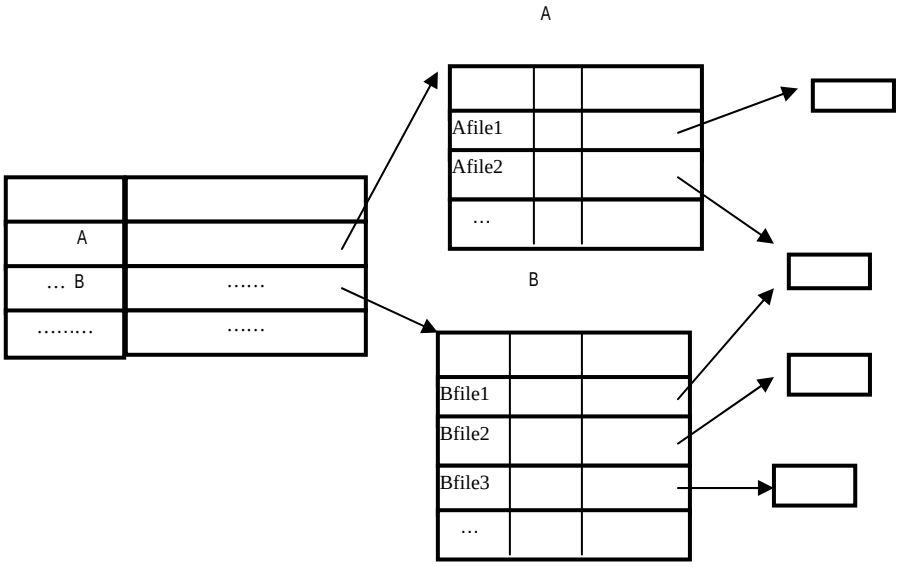


6-1

‘ ’

6.2.3

6-2



6-2

6.2.4

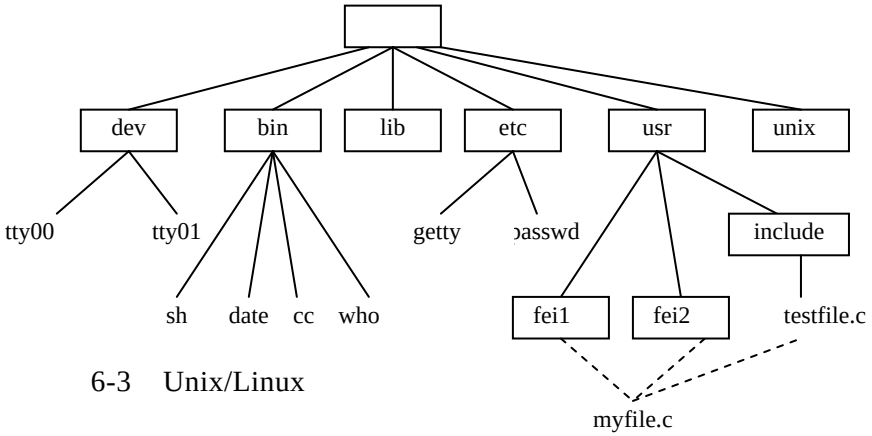
6-3

/ /

MS-DOS2.0

112

UNIX/Linux dev
bin lib etc
tmp usr usr
bin tmp lib include
UNIX/Linux UNIX/Linux
6-3 UNIX



6-3 Unix/Linux

/usr/fei1/myfile.c

“ / ” UNIX/Linux inode
inode
usr
… usr inode
inode inode inode inode
usr inode
usr inode usr
fei1
usr fei1 inode
inode inode fei1 inode (
inode) fei1
myfile.c fei1
myfile inode inode
inode myfile.c inode (
inode)
myfile.c inode

“ . ” inode “ . . ” inode
. . /fei1/myfile.c “ . . ” inode
fei1

6.3

6.3.1

6.3.2

1

COBOL

…… COBOL

-
-
- “ ” 1
  2 1 2
  ‘ M ’ ‘ F ’ ‘ ’ ‘ ’
-
- FORTRAN
  COBOL

2

80 80
80

800

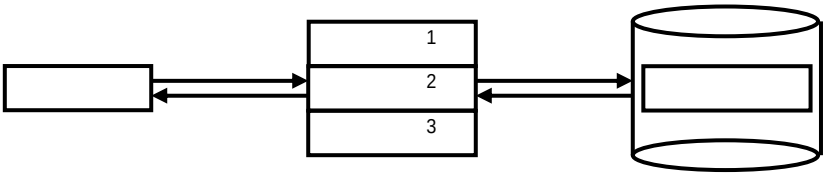
10 10

1600

20 20

6-4

I/O

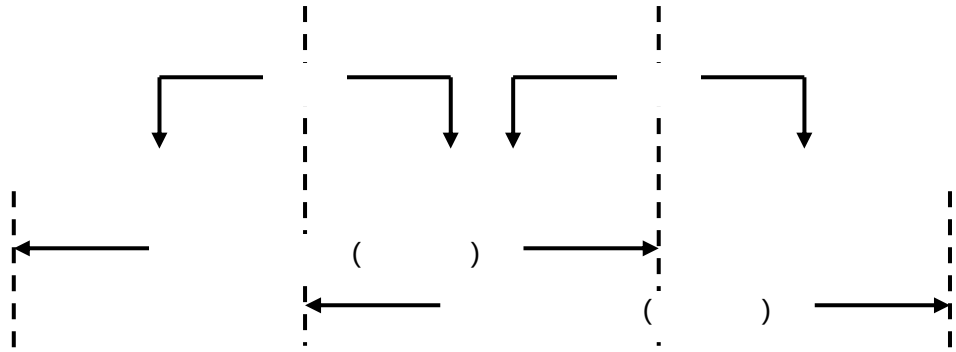


6-4

3

- – –
- – –
- – –

6-5



6-5

- F
- V
- S

F

V

-
-

RL

RL BL

S

-
-
-
-

1000

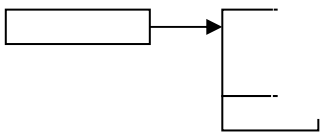
6.3.3

1

2

6-6

0



- – –
‘ ’

- – – ‘ ’

- – –

3

(hash)

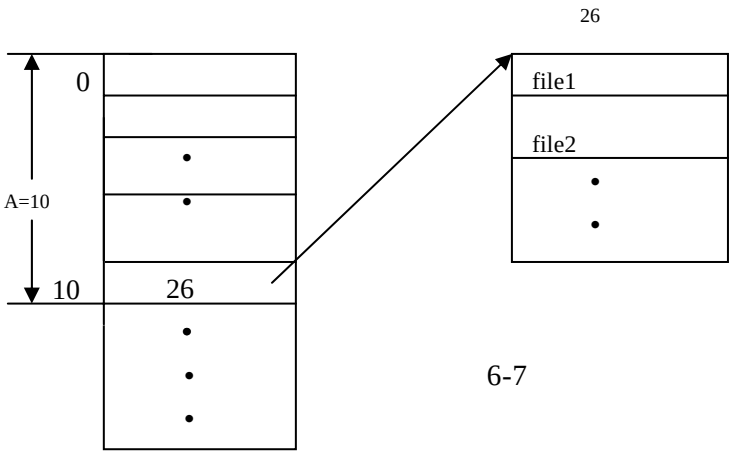
Linux hash cache
cache Linux
hash inode cache directory cache buffer cache
‘ ’

‘ ’

hash

hash hash

1 hash 8 ASC 8
8 hash 2 “ ”
ASC 2
a_i ASC
A= (a_1 a_2 … a_8)
2 6-7 1
hash A A
A



6-7

3 1
A A

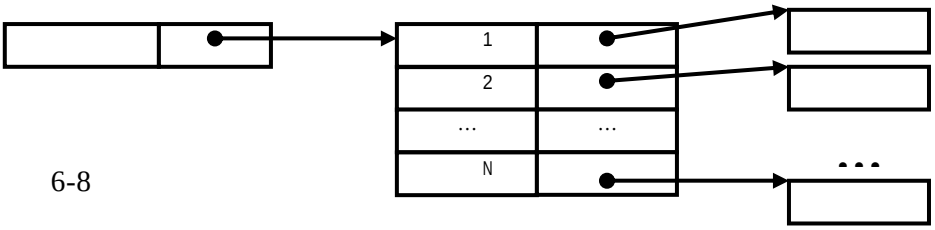
4
A
A+k k

A+2 × k A+k

4

6-8
( )

( )



6-8

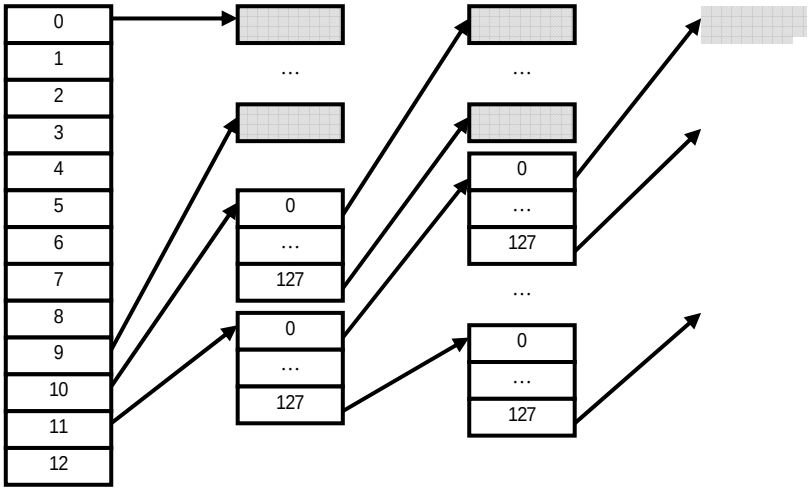
( )

( )

n (n+1)

n

( )



UNIX/Linux 6-9
13 4
UNIX/Linux (
) 10 0 9
10 11
128
12 13 512
UNIX/Linux 11
10
80% 20%
10

6.4

6.4.1

‘ ’

‘ ’

1

‘ ’

-
-
-
-
-

‘ ’

2

‘ ’

‘ ’

-
-
-
-

3

-
-
-
- I/O

4

- ‘ ’
-
-

5

-
-
-

6.4.2 UNIX/Linux

UNIX/Linux

1

1

inode C

```
int fd, mode;
char *filenamep;
fd = creat (filenamep, mode);
```

filenamep mode

i_mode fd

creat “ ”

fd

/usr/lib/d2 C

creat

```
int fdlib;
```

fdlib = creat (“/usr/lib/d2”, 0775);

d2

d2 /usr/lib

d2 i_mode 0775

i_nlink “ 1 ”

f_flag “ ” f_offset “ 0 ”

d2

“ ”

“ ”

2)

i_link “ 1 ”

unlink (filenamep) creat

“ ”

2

“ ”

“ ”

1

int fd, mode;
char * filenamep;
fd = open (filenamep, mode);

mode 0 1

2 creat open

mode

“ ”

i_count “ 1 ” i_count

(2)

```
int fd;
close (fd);
```

fd close

fd

f_count “ 1 ” “ 0 ”

i_count “ 1 ” “ 0 ”

f_count i_count

f_offset f_offset

3

“ ”

“ ”

f_offset

1)

```
int nr, fd, count;
char buf [ ]
nr = read (fd, buf, count);
```

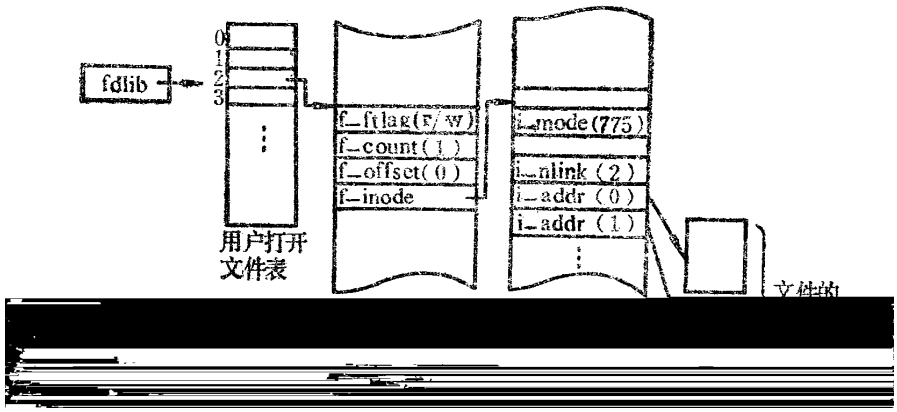
fd buf

count nr

nr count

read

“ 0 ”



6-10

/usr/lib/d2

6-10

d2 1500 bufp number

read

```
number = read (fdlib, bufp, 1500);
```

read f_flag

f_offset

i_addr

bufp read

2)

```
nw = write (fd, buf, count);
```

fd, count nw read buf buf

nw count

4

“ ” f_offset

offset

lseek f_offset

```
long lseek;
long offset;
int whence, fd;
lseek (fd, offset, whence);
```

fd whence “ 0 ”

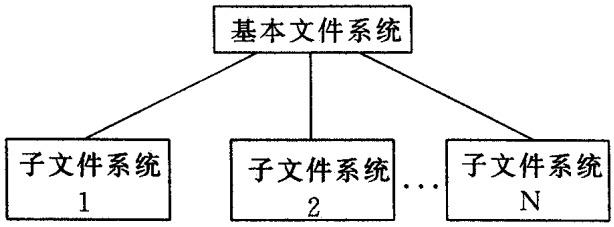
f_offset offset whence “ 1 ” f_offset

offset

6.4.3

( )

(Windows Linux)



6-11 Unix/Linux

Windows/DOS

Windows/DOS

UNIX/Linux UNIX/Linux

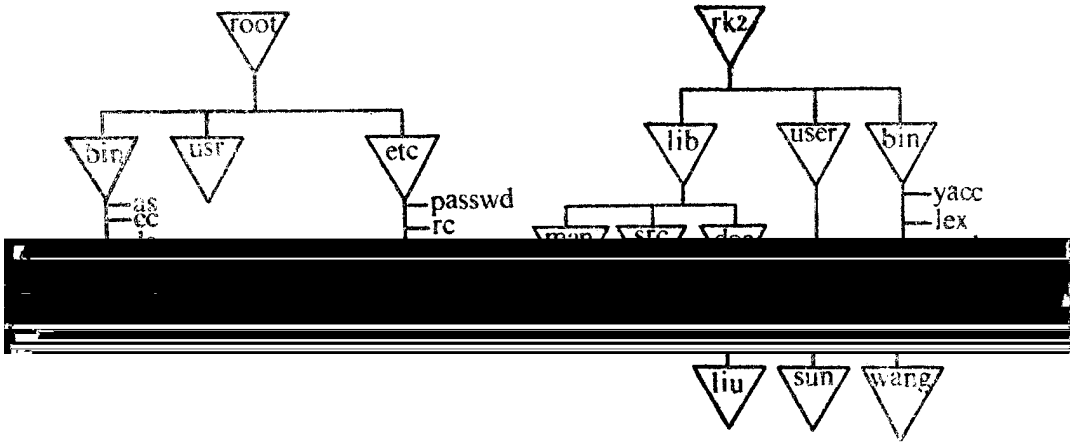
6-11

(1)

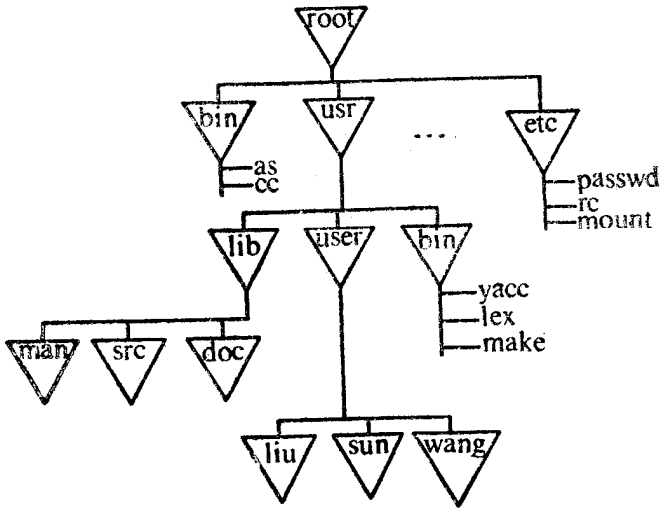
(2)

UNIX/Linux

C



(b) 待安装的文件卷 rk2



(c) 把文件卷 rk2安装到 usr节点之后

6-12

“ ”

“

”

“ ”

6-12

(a) rk2 usr (b)
rk2( ) (c)
( )mount
mount(“ /dev/rk2” “ /usr” 0)
rk2 usr
(mount point)
“ ” “ ” “ ”

inode (/dev/rk2)
inode (/usr)

(1)

(2)

6.4.4

( )

UNIX/Linux

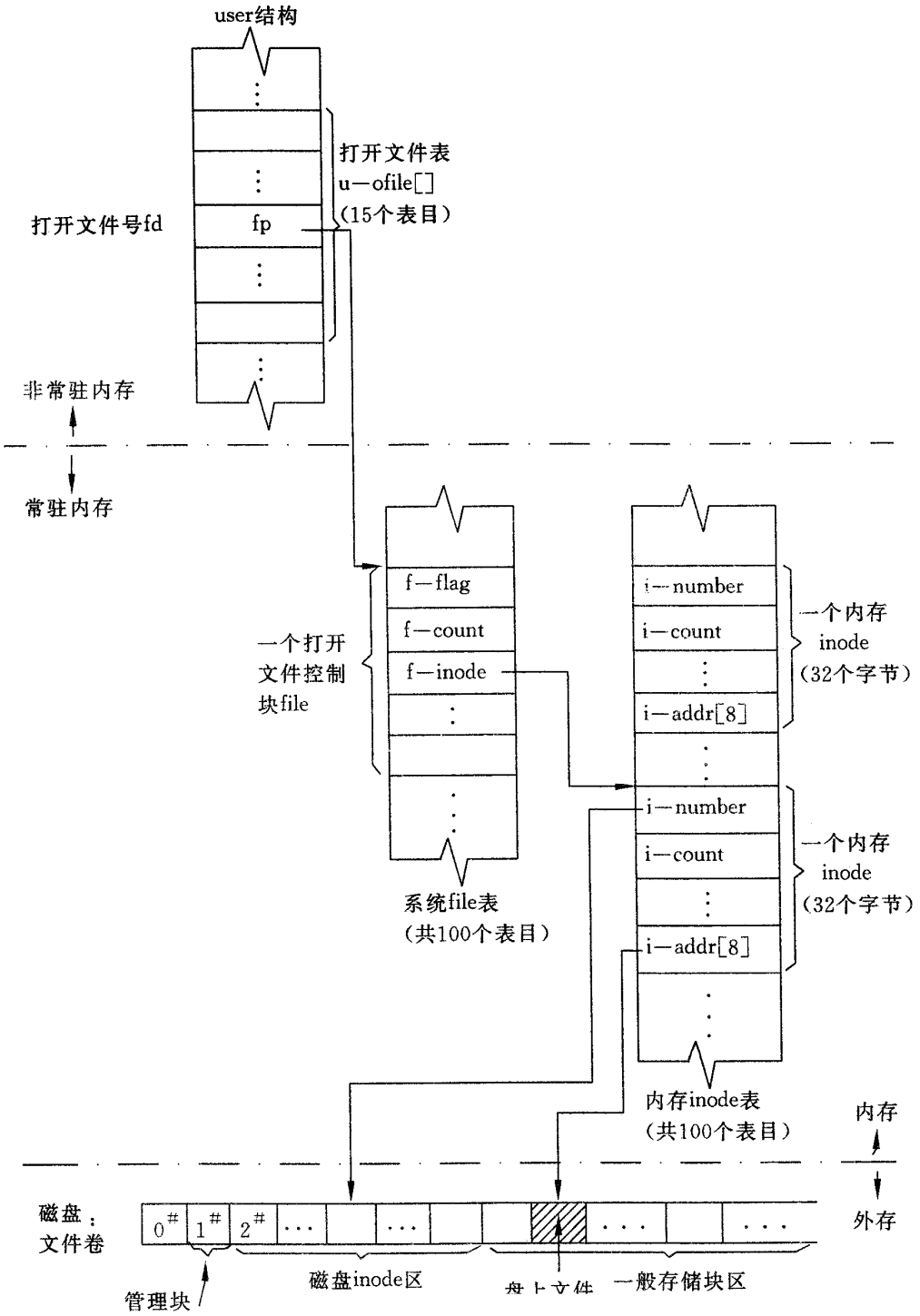
UNIX./Linux

(file link)

UNIX/Linux

( )
( ) 6-3 fei1 fei2
myfile.c include testfile.c
UNIX/Linux inode

```
        chat * oldnamep, * newnamep;
        link (oldnamep, newnamep);
oldnamep      newnamep
```



6-13 Unix/linux

UNIX/Linux

/

/

UNIX/Linux

user

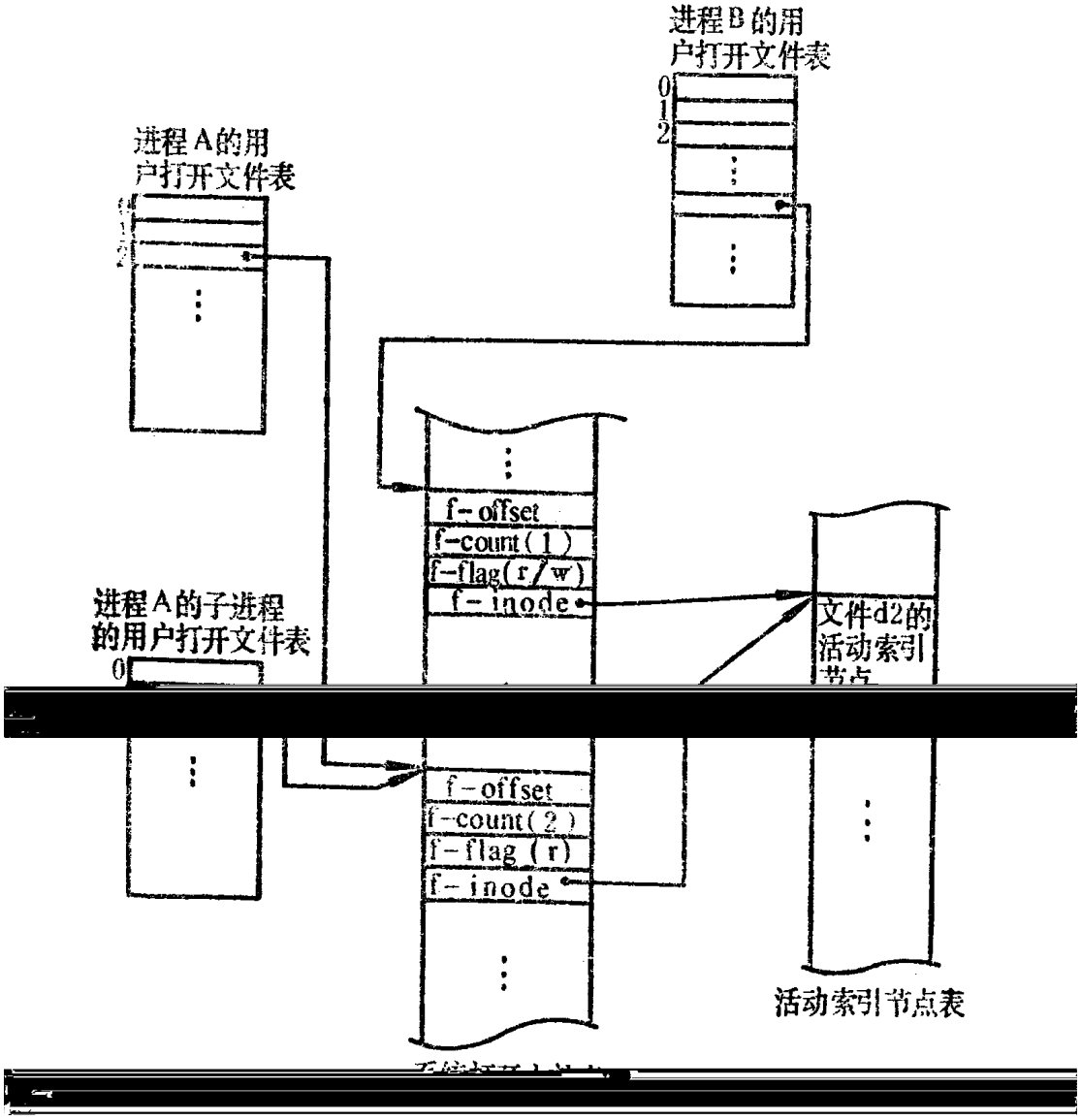
/

fork

user user

6-14 UNIX/Linux

6-14 UNIX/Linux



6-15 UNIX

“

”　　100

f_offset　　f_count　　f_flag

f_inode　　f_count

6-14　　6-15

6-15　　A　　d2

f_count　“ 2 ”　　B

f_count　“ 1 ”　　A　　d2

‘ ’

●

1

’1’ ’0’

CP/M VM/SP Windows Macintosh

2

3

DOS

4

6-17 UNIX/Linux 512

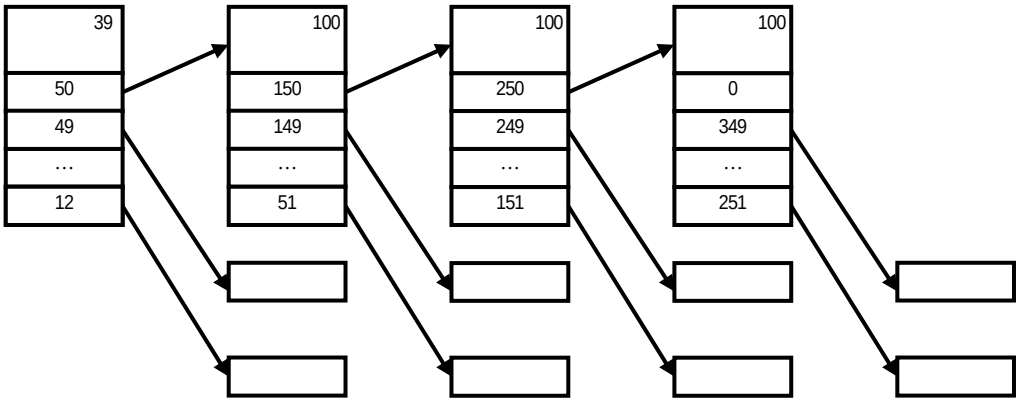
438 12 349

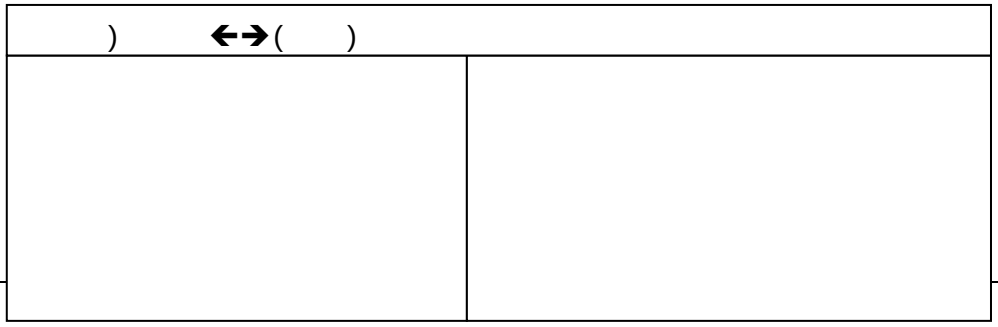
100

50#-12#— 50# 100 150#-51#

150# 100 250#-151#

250# 1 0





6-17 UNIX

0

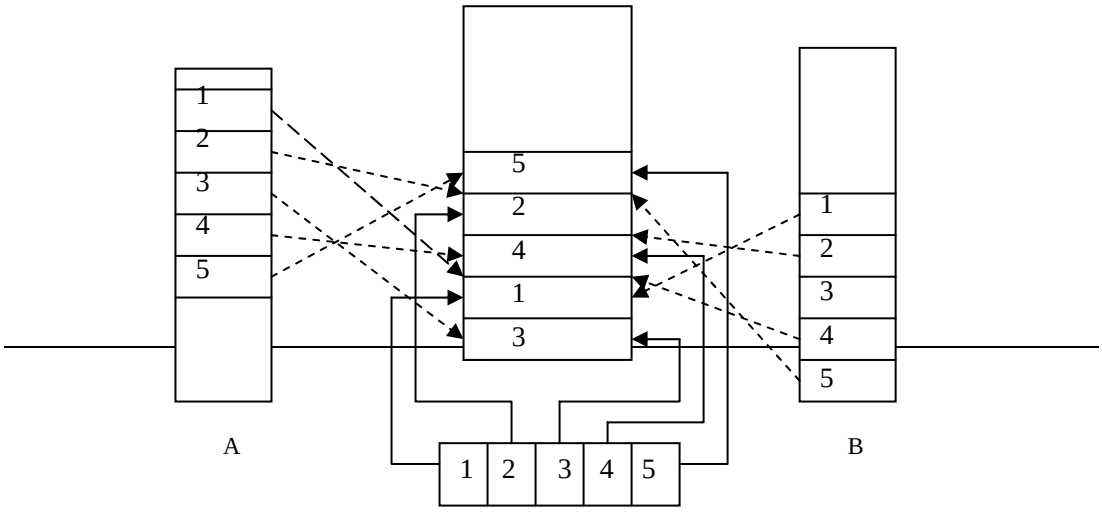
6.4.7

MULTICS
Windows
2000/XP
memory-mapped file

I/O
I/0

64KB
512KB 512KB
0 512KB+100 100
512KB
0
512KB+100

6-18



I/O

0

6.4.8

UNIX UNIX
Windows2000/XP FAT
Linux

AT&T Sun vnode/vfs NFS UNIX SVR4

(1)

(2)

i

i

v ( ) vfs( ) v
vfs

v (struct vnode)
v-count v-data v-type v-op (struct vnodeops) vnodeops
vop-open vop-read vop-write vop-mkdir vop-lookup
vn-open vn-link vfs (struct vfs)
vfs-next vfs-data vfs-op vfs-fstype (struct vfsops) vfsops
vfs-mount vfs-root vfs-unmount vfs-sync vfs-statvfs
v vfs
v vfs UNIX v i
i
v i
v i
UNIX SVR4 v

```
Struct vnode {
    u-short    v-flag;                  /*          */
    u-short    v-count;                 /*          */
    struct vfs    *vfsmountedhere;      /*                    */
    struct vnodeops    *v-op;           /*v                   */
    struct vfs    *vfs p;               /*    v                */
    struct stdata    *v-stream;         /*                    */
```

```
    struct page    *v-page;          /*                */
    enum vtype     v-type;           /*          */
    dev-t    v-rdev;                 /*                id*/
    caddr-t    v-data;               /*                    */
}
```

UNIX SVR4 v

vfs v

v-op

( )

vfs vfs

```
struct vnode {
    struct vfs    *vfs-next;         /*vfs            */
    struct vfsops    *vfs-op;        /*              */
    struct vnode    vfs-vnode covered;  /*                    v      */
    int vfs-fs    type;             /*                */
    caddr-t    vfs-data;            /*          */
    dev-t    vfs-dev;               /*              id*/
 }
```

v v

v vfs

UNIX SVR4

( )

vfs mount

mount lookuppn() v v

mount

( )

vfs

vfs v

vfs-mount mount

6.5 Linux

6.5.1 Linux

Linux EXT EXT2 MINIX UMSDOS NCP ISO9660 HPFS MSDOS NTFS XIA VFAT PROC NFS SMB SYSV AFFS UFS

Linux

Linux VFS(Virtual File System) VFS

Linux

VFS inode

Linux hash

I/O EXT2 Linux

inode

6.5.2 Linux

Linux

inode

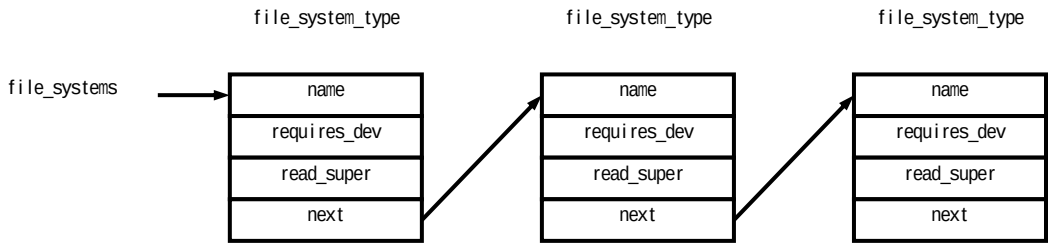
inode

inode

Linux

VFS

kerneld insmod



6-19 Linux

6-19 file_systems

name

requires_dev read_super

next register_filesystem

unregister_filesystem

link

Linux mount

umount

Linux /etc/fstab mount

mount umount

/ add_vfsmnt/remove_vfsmnt

/ lookup_vfsmnt

vfsmntlist vfsmnttail

vfsmount

```
static struct vfsmount vfsmntlist = (static struct vfsmount )NULL;      /*      */
static struct vfsmount *vfsmnttail = (static struct vfsmount *)NULL;    /*      */
static struct vfsmount *mru_vfsmnt = (static struct vfsmount *)NULL;    /*      */

struct vfsmount {
   kdev_t    mnt_dev;                  /*                                */
   char    *mnt_devname;               /*                               /dev/hda1 */
```

```
    char    *mnt_dirname;           /*              */
    unsigned int    mnt_flags;      /*               ro */
    struct semaphore    mnt_sem;    /*                  */
    struct super_block    *mnt_sb;  /*              */
    struct file    *mnt_quotas[MAXQUOTAS];/*                    */
    time_t    mnt_iexp[MAXQUOTAS];  /* expiretime for inodes */
    time_t    mnt_bexp[MAXQUOTAS];  /* expiretime for blocks */
    struct vfsmount    *mnt_next;   /*              */
};
```

mount

mount

1 3

2

3

4

5 super_blocks[]

6 read_super

7 vfsmount

umount

umount

6.5.3 VFS

VFS

Linux

VFS

inode VFS

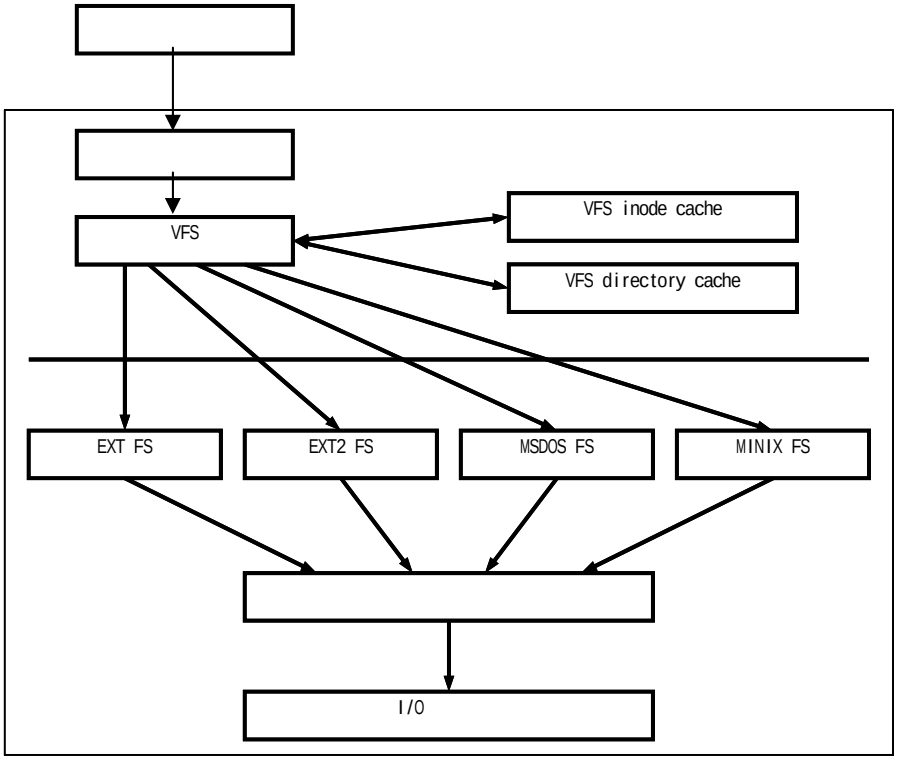
VFS

-
-
-
-

inode

VFS Linux 6-20

VFS



6-20 Linux

VFS inode

VFS read_super

VFS

```
struct super_block {
   struct list-head s-list;
   kdev_t    s_dev;                  /*                          */
   unsigned long    s_blocksize;     /*          */
   unsigned char    s_blocksize_bits;     /*      2                */
   unsigned char    s_lock;          /*                                   */
   unsigned char    s_rd_only;       /*              */
   unsigned char    s_dirt;          /*                */
   struct file_system_type    *s_type; /*                                */
   struct super_operations    *s_op;  /*                                 */
   struct dquot_operations      dq_op;
   unsigned long    s_flags,s_magic,s_time;
   struct dentry    *s-root;
   struct wait_queue    *s_wait;     /*                          */
   struct inode    *s-ibasket;
   short int s-ibssket-max,s-ibasket-count;
   struct list-head s-dirty;
   union { /*                                    */
            struct minix-sb-info minix-sb;
            struct ext2-sb-info ext2-sb;
            struct hpfs-sb-info hpfs-sb;
```

```
        struct ntfs-sb-info ntfs-sb;
        struct msdos-sb-info msdos-sb;
        struct isofs-sb-info isofs-sb;
        struct nfs-sb-info nfs-sb;
        struct sysv-sb-info sysv-sb;
        struct ufs-sb-info ufs-sb;
        struct romfs-sb-info romfs-sb;
        struct smb-sb-info smb-sb;
        struct hfs-sb-info hfs-sb;
        struct adfs-sb-info adfs-sb;
        struct qnx4-sb-info qnx4-sb;
        void *generic-sbp;
    } u;
};
```

super_block.u VFS
inode s-mounted inode
inode Linux
VFS inode
inode_cache
inode

```
struct inode {
  struct list-head i-hash,i-list,i-dentry;
  unsigned long i-ino;
  unsigned int i-count;
  kdev_t  i_dev;                  /*                  */
  umode_t  i_mode;                /*                 */
  nlink_t  i_nlink;               /*            link  */
  uid_t  i_uid;
  gid_t  i_gid;
  kdev_t  i_rdev;                 /*                  */
  off_t  i_size;                  /*          */
  time_t  i_atime  i_mtime  i_ctime;
  unsigned long  i_blksize;        /*   (1KB)          */
  unsigned long  i_blocks;         /*        */
  unsigned long  i_version;
  unsigned long  i_nrpages;        /*                */
  struct semaphore  i_sem  i-atomic-write;
  struct inode_operations  *i_op;  /*                  */
  struct super_block  *i_sb;       /*          VFS     */
  struct wait_queue  *i_wait;      /*                */
  struct file_lock  *i_flock;      /*                  */
  struct vm_area_struct  *i_mmap;
  struct page  *i_pages;           /*                 */
  struct dquot  *i_dquot[MAXQUOTAS];
  struct inode  i_next, i_prev;    /*inode      */
  struct inode  i_hash_next, i_hash_prev;/*inode cache     */
  struct inode  *i_bound_to, *i_bound_by;
  struct inode  *i_mount;          /*              inode     */
```

```
        unsigned long    i_count;          /*              0              */
        unsigned short    i_flags, i_writecount;
        unsigned char    i_lock;           /* inode                */
        unsigned char    i_dirt;           /* inode                */
        unsigned char    i_pipe    i_sock    i_seek    i_update    i_condemned;
        union { /*                        inode              */
                struct pipe-inode-info pipe-i;
                struct minix-inode-info minix-i;
                struct ext2-inode-info ext2-i;
                struct hpfs-inode-info hpfs-i;
                struct ntfs-inode-info ntfs-i;
                struct msdos-inode-info msdos-i;
                struct umsdos-inode-info umsdos-i;
                struct isofs-inode-info isofs-i;
                struct nfs-inode-info nfs-i;
                struct sysv-inode-info sysv-i;
                struct affs-inode-info affs-i;
                struct ufs-inode-info ufs-i;
                struct romfs-inode-info romfs-i;
                struct coda-inode-info coda-i;
                struct smb-inode-info smb-i;
                struct adfs-inode-info adfs-i;
                struct qnx4-inode-info qnx4-i;
                struct socket socket-i;
                void *generic-ip;
        } u;
        };
```

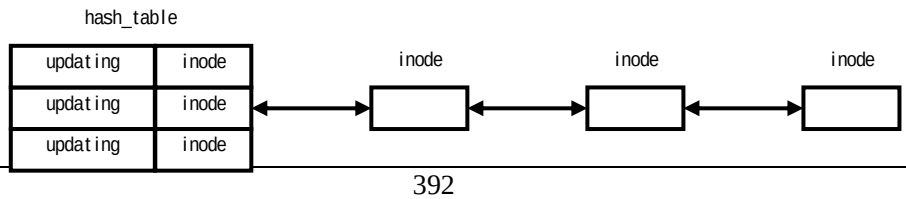
inode.u inode
inode i_prev i_next first_inode inode i_dev
i_ino i_count 0 inode
inode first_inode inode VFS
grow_inodes
inode first_inode first_inode VFS
insert_inode_free( ) remove_inode_free( ) put_last_free( ) insert_inode_hash( )
remove_inode_hash( ) clear_inode( ) get_empty_inode( ) lock_inode( ) unlock_inode( )
write_inode( )

6.5.4

Linux VFS
cache

1 VFS inode cache

VFS inode inode
VFS inode cache hash 6-21
VFS inode hash h
hash_table[h] i_hash_next i_hash_prev cache
inode inode i_count 1 inode



6-21VFS inode cache

2 VFS directory cache

inode

Linux inode VFS directory cache directory cache Linux namei()

```
struct hash_list { struct dir_cache_entry *next  *prev; };
struct dir_cache_entry {
   struct hash_list  h;
   kdev_t  dc_dev;
   unsigned long  dir;          /*          inode */
   unsigned long  version;
   unsigned long  ino;          /*          inode */
   unsigned char  name_len;
   char  name[DCACHE_NAME_LEN];
   struct dir_cache_entry  *lru_head;
   struct dir_cache_entry  *next_lru    *prev_lru;
}
```

VFS directory cache level1_cache level2_cache cache level1_head level2_head level1_cache level2_cache dir_cache_entry.next_lru dir_cache_entry.prev_lru 128 LRU level1_cache level2_cache level1_cache level2_cache Linux level1_cache level2_cache hash

3 buffer cache

Linux buffer cache buffer cache

buffer cache

Linux struct buffer_head buffer_head

buffer_head

buffer cache hash buffer

- hash

```
static struct buffer_head ** hash_table;
```

- 4 lru_list[0] lru_list[1] lru_list[2] inode lru_list[3] buffer_head

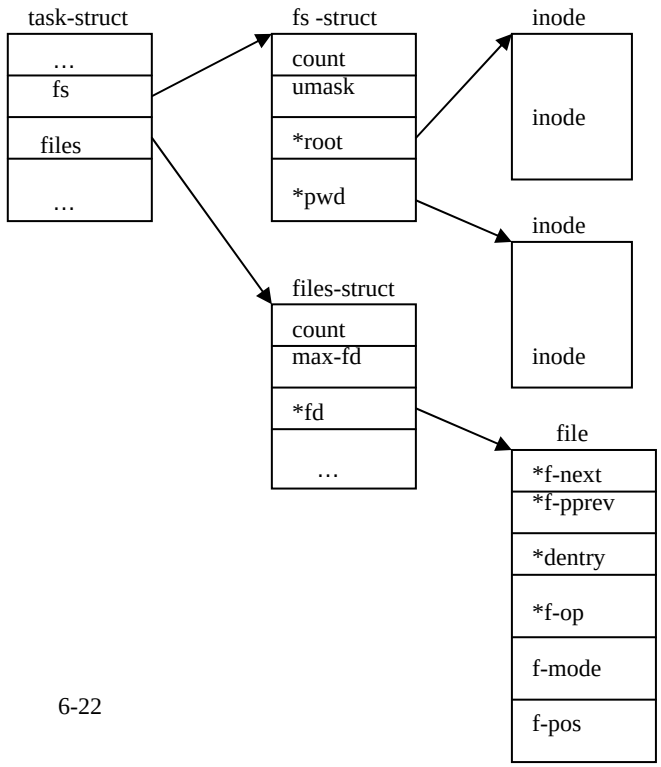
```
static struct buffer_head *lru_list[NR_LIST];
```

- buffer 512 1024 2048 4096 8192 5 buffer_head

```
static struct buffer_head *free_list[NR_SIZES];
```

- buffer_head

Linux
include/linux/fs.h



6-22

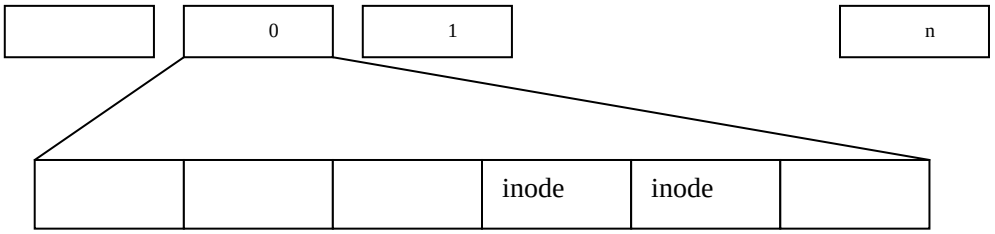
6.5.6 EXT2

EXT 1992 EXT2 1994 Linux
EXT2
6-23 EXT2 (boot block)
(block group) (super group)
(group descriptors) (block bitmap) inode (inode bitmap) inode (inode table) (data blocks) inode
inode inode inode inode inode
inode inode

inode inode
inode

1 EXT2

EXT2
0
EXT2 inode
inode
inode
Linux (fragment)
EXT2 EXT2
include/linux/ext2_fs.h ext2_super_block ext2_sb_info



6-23 EXT2

2 EXT2

inode inode
inode
include/linux/ext2_fs.h ext2_group_desc

3 EXT2 inode

inode inode
inode inode
/ / / link
include/linux/ext2_fs.h ext2_inode

4 EXT2

“ . ” “ .. ”

```
struct ext2_dir_entry {
   _u32    inode;                  /*          inode    */
   _u16    rec_len;                /*          */
   _u16    name_len;               /*          */
   char    name[EXT2_NAME_LEN];    /*       */
}
```

5

“ ”

Linux defrag(defragmentation program)
EXT2

-

64
8

- EXT2 (EXT2_PREALLOCATE)

EXT2 inode ext2_inode_info
prealloc_block prealloc_count

6.6 Windows 2000/XP

6.6.1 Windows 2000/XP

Windows 2000/XP FAT FAT DOS
Windows3.x Windows95

FAT 2^{12} 2^{16} FAT
2GB
Windows9x Windows Me FAT 32 FAT32
FAT16 4GB
FAT Windows98
FAT32 Windows NT FAT32 NT Windows
2000/XP FAT32 CDFS UDF
HPFS
Microsoft Windows NT
NTFS(New Technology File System) NTFS FAT

/ Windows 2000/XP
NT4 NTFS NT4 NTFS4
NTFS5 NTFS

- NTFS
- NTFS
  Windows2000/XP (ACL)
- Windows2000 EFS(Encrpyting File System)
  NTFS
- NTFS RAID
- NTFS 64
- NTFS
  NTFS
- Unicode NTFS 16 Unicode
  255
  Unicode
- NTFS
  Windows 2000
  ID (OLE)
  NTFS
- Windows2000
  NTFS
- NTFS

- Windows2000 NTFS
- Windows2000
- Windows2000
  50% 40%
- Windows2000 NTFS ID
  ID ID
  (OLE
)

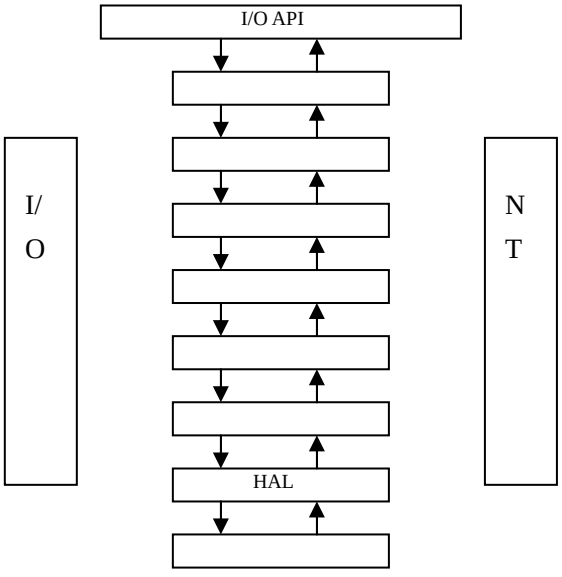
- POSIX POSIX
  Windows2000
  “ (HSM) ”

Windows 2000/XP (DFS) Windows 2000/XP NT4
Windows 2000/XP DFS
DFS
DFS

“ ” DFS
DFS

6.6.2 Windows2000/XP FSD

Windowx2000/XP I/O I/O
6-24



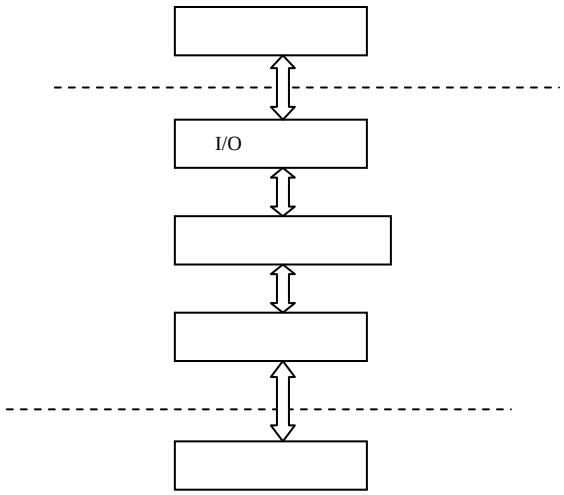
6-24 Windows

- I/O I/O
- I/O

- FSD(File System Driver)
  (NTFS)
-
  I/O
  API

FSD
FSD I/O
FSD Ntoskrnl
IFS(Installable File System)
FSD FSD FSD

1. FSD
   FSD Ntfs.sys Fastfat.sys Udfs.sys CDfs.sys Raw FSD
6-25 FSD I/O I/O
FSD



6-25 FSD

Windows2000/XP

FSD I/O
VPB(Volumn Parammeter Block)
FSD VPB I/O
I/O FSD
FSD
FSD

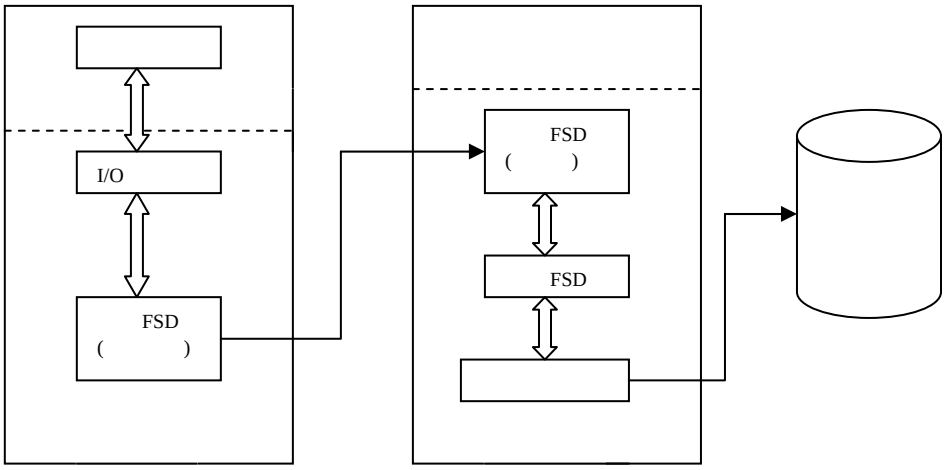
2. FSD

FSD FSD FSD

FSD I/O

FSD FSD

FSD 6-26 FSD



6-26 FSD

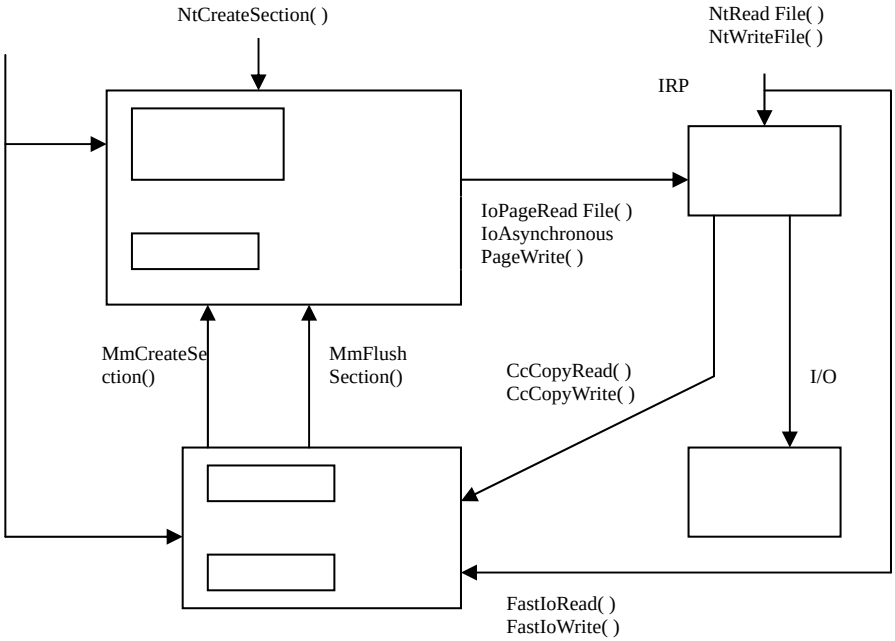
Windows2000/XP FSD LANMan (LANMan Redirector) FSD LANMan (LANMan Server)
/

CIFS(Common Internet File System)

3. FSD

Windows FSD 6-27

FSD



6-27 FSD

I/O Win32 I/O CreateFile ReadFile WriteFile

MmFlushSection
MmFlushSection IoAsychronousPageWrite FSD

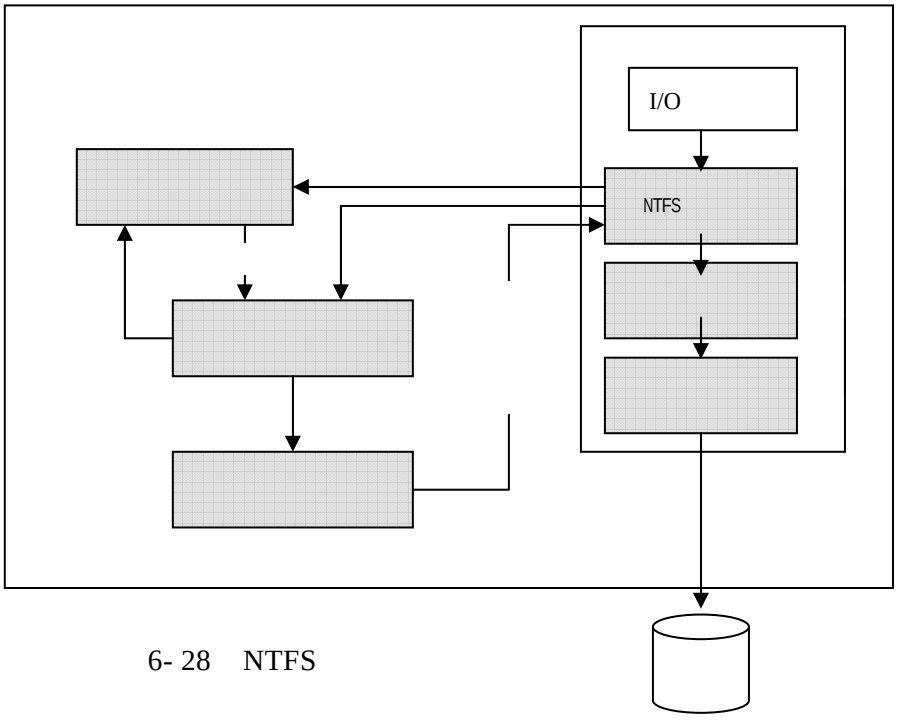
IoAsychronousPageWrite IRP IRP
FSD
I/O
MmAccessFault IoPageRead IRP

6.6.3 NTFS

Windows2000/XP NTFS FAT HPFS POSIX
I/O

I/O
Window2000/XP
6-28 Windows2000/XP I/O
NTFS
I/O I/O I/O

NTFS



6- 28 NTFS

LFS(log file server)

NTFS
NTFS
Windows 2000 NTFS
( )
Windows2000

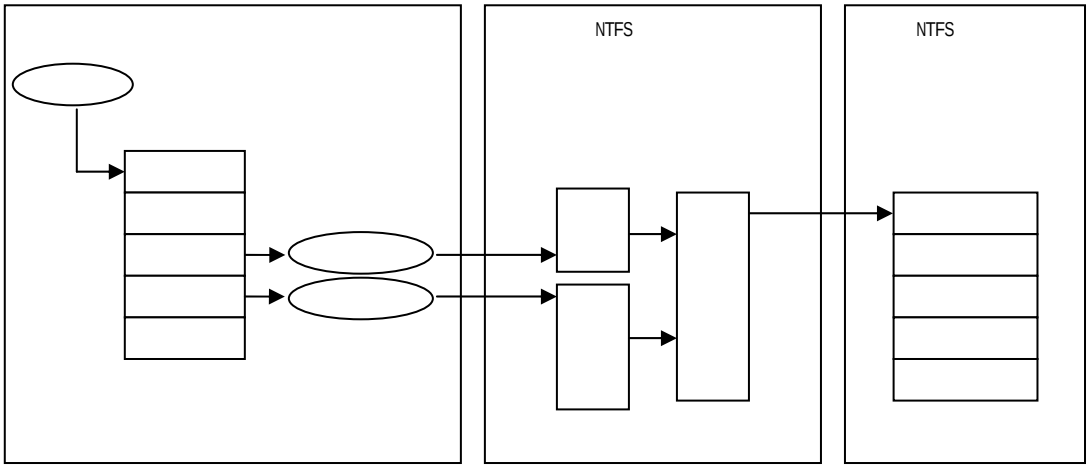
Windows2000
NTFS
“ ” (lazy writer) I/O
(
)
NTFS
Windows2000/XP
Windows2000/XP —— I/O NTFS
Windows 2000/XP

I/O NTFS

6-29 I/O
NTFS NTFS

Stream Control
Block SCB SCB

SCB File Control Block FCB FCB
MFT
NTFS



6-29 NTFS

6.6.4 NTFS

NTFS
NTFS FAT NTFS
512
NTFS

512B 1KB 2KB … 64KB
1 2 128
NTFS LCN(Logical Cluster Number) VLN(Virtual Cluster Number) LCN VCN
LCN
VCN LCN

NTFS NTFS

MFT(master file table) NTFS NTFS
1KB MFT
MFT MFT NTFS “
” NTFS
($) NTFS
6-30

MFT($Mft) /*
MFT ($MftMirr) /*MFT 9
($Logfile) /*
($Volume) /* NTFS
($AttrDef) /*
($/) /*
($Bitmap) /*
($Boot) /*Win2000/XP
($BadClus) /*
($Secure) /*
($UpCase) /*
($Ext. metadata Directory)
…
…

NTFS
MFT MFT
“ MFT ” MFT

NTFS MFT
VCN LCN NTFS
MFT NTFS
MFT NTFS NTFS

NTFS —— (log fi1e)
NTFS NTFS
NTFS
MFT NTFS
NTFS
NTFS MFT
MFT
NTFS (bitmap file)

(bootfile) Windows2000

Format

NTFS NTFS

Windows 2000 “

”

I/O

NTFS “ ” (bad-cluster)

“ ” (volume fi1e) NTFS

Chkdsk

NTFS (attribute definition table)

NTFS “ ” 64

MFT 1 (

1)

MFT NTFS

NTFS / (

)

NTFS —— NTFS

( ) ( )

NTFS FAT 255 Unicode5IX-6.4(e)]TJ/TT33

“ ” (index buffers)

6.6.5 NTFS

NTFS (logging)

NTFS

“ (Write-ahead logging) ”

- LFS(Log File Service) NTFS
  NTFS LFS LFS (restart area) " NTFS Y 5 Y ý

-
-

NTFS ---
EFS(Encrpyted File System)
EFS FEK(File
Encryption Key) DESX(Data Encryption Standard)
DESX
(Symmetric Encryption Algorithm) EFS
RSA(Rivest Shamir Adleman) FEK
EFS --- DDF(Data
Decryption Field) DDF FEK
FEK

6.7

IBM

i-node

( )

ch7

(1)

(2)

i

1.
2. ? ?
3.
4. ? ?
5. ? ?
6.
7. ? ?
8.
( ) ( )
( )
(.)
2
9. ?
10. Windows 2000 NTFS
11. Windows 2000 NTFS
12. Windows 2000 NTFS

13. Windows 2000 NTFS
14.
(1)
(2)
(3)
15. 3
? ?
16.
17. ? ?
18. hash ?
19. hash ?
20. hash
21. ?
22. ?
23. ‘ ’ ? ?
24. ? ?
25. UNIX
26. UNIX/Linux
27. UNIX/Linux i
28. UNIX/Linux 0
29. UNIX/Linux
30. OPEN CLOSE
1
2 OPEN COLSE
31.
?
32. rename
33.
36 UNIX/Linux 2 1 1
2

1 j
i i
2 B= R= F= (
) F
3 500 32

(1) ? (2) i j ? (3) /

4 A B C D E
A D E F A A
D E

5 UNIX 1KB 4 256
(1)9999 (2)18000 (3)420000

6 UNIX/Linux /usr/wang /ast/xxx

7 UNIX F rwxr-x---, uid 12 gid=1,
uid=6 gid=1 F

8 5
512 50 121 75 80 63
1569

9 UNIX/Linux 1KB 4
263168

10 13 0 9
3
n 512

11 ABCD 18 512B
1024B 12 15

12
?

13 1ms
40ms h h
0 1

14 10 100 16
(1) ?(2)
5 ?

15 100 8 4
0 16 200 (0-199)
(1) 15 7 0
?(2) 56 6 3
0?

16 128B 4B 68B 8KB

17 100
I/O ( ) (1)
(2) (51) (3) (4)

(5) (51) (6)

CH7

7.1

——

-
-

-

(Authentication)
(Authorization) (Encryption) (Audit)
1985 “ ”

- D
- C1
- C2
- B1
- B2
- B3
- A1

DOS D Windows NT Solaris
C2 OSF/1 B1 UNIX Ware 2.1 B2

7.2

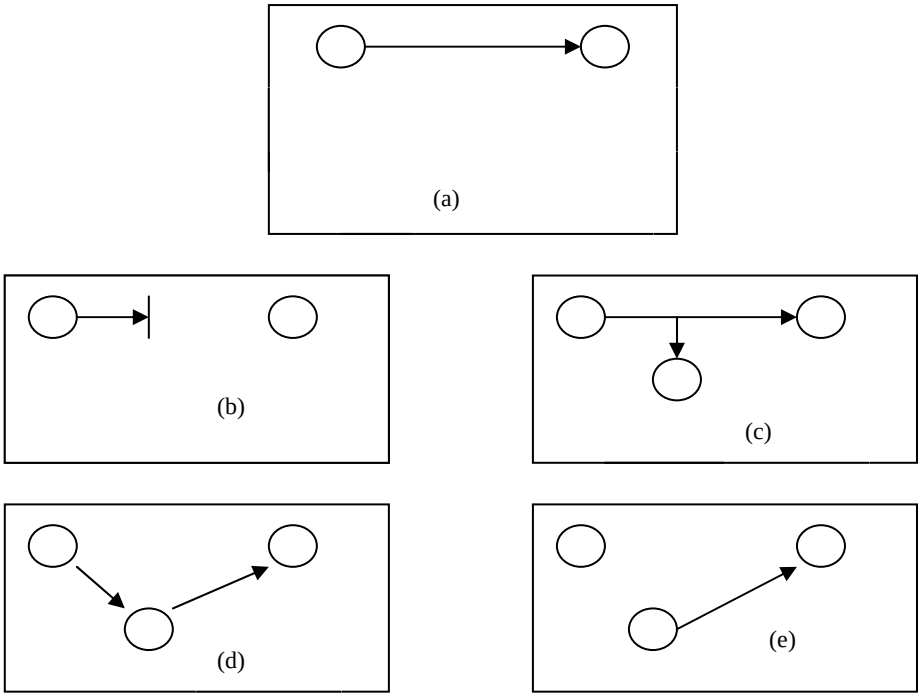
( )

- confidentiality

- integrity
- availability
- authenticity

7-1a

-
-
-
-



7-1

			/

1

2

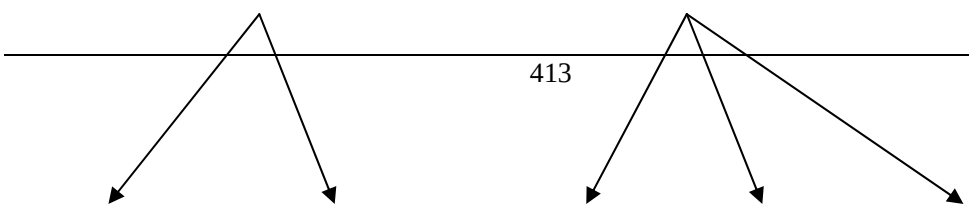
3

A B C D A B C D E

E

4

7-2



7.3

7.3.1

I/O

-
-
-
-
-
-

7.3.2

7.3.3

ID ID
ID ID/

ID ID
ID/

7.3.4

(access satrix)

- **(subject)**
- **(object)**
- **(access authority)**

7.4

7.4.1

hacker
cracker Anderson

- masquerader
- misfeasor
- clandestine user

-

-

1
2
3

4
5
6
7
8
6

7

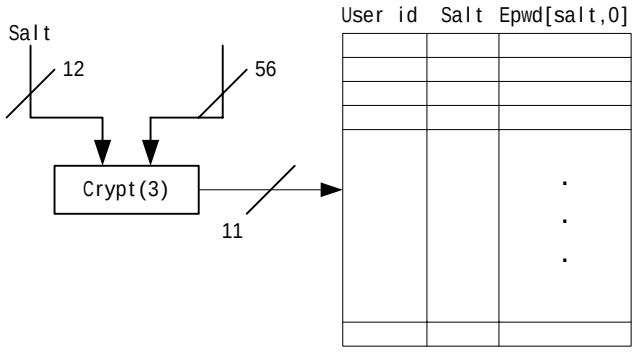
8

7.4.2

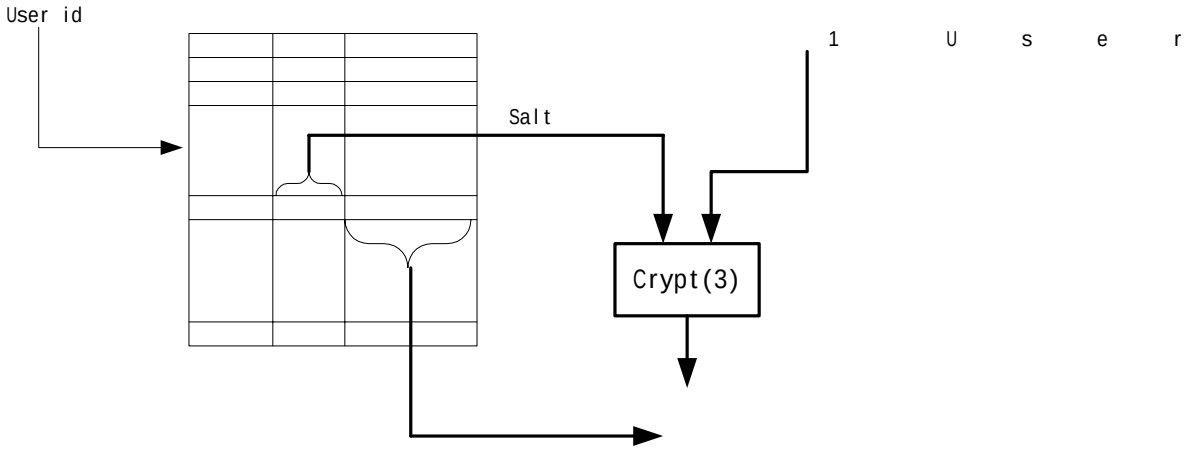
ID ID ID

- ID
ID

- ID
- ID ID



(a)



1 U s e r

UNIX ID ID
salt

DES 25
25

UNIX

(Purdue) 54
7000 3%

6 8

1	55	0.004
2	87	0.006
3	212	0.02
4	449	0.03
5	1260	0.09
6	3035	0.22
7	2917	0.21
8	5772	0.42
total	13787	1.0

2

Spafford,E.

- UNIX

-
-

3

8

4

-
-
-
-

8

“ ”

“ ”

7.4.3

-
-
-

Porras,P.Stat

-

1 2

-

1

2

Dorothy Denning

Smith

GAME <library>

COPY GAME.EXE TO <LIBRARY> GAME.EXE

Smith		<library>copy.exe	0	CPU=00002	11058721678
Smith		<smith>GAME.exe	0	RECORDS=0	11058721679
Smith		<library>copy.exe	write-viol	RECORDS=0	11058721680

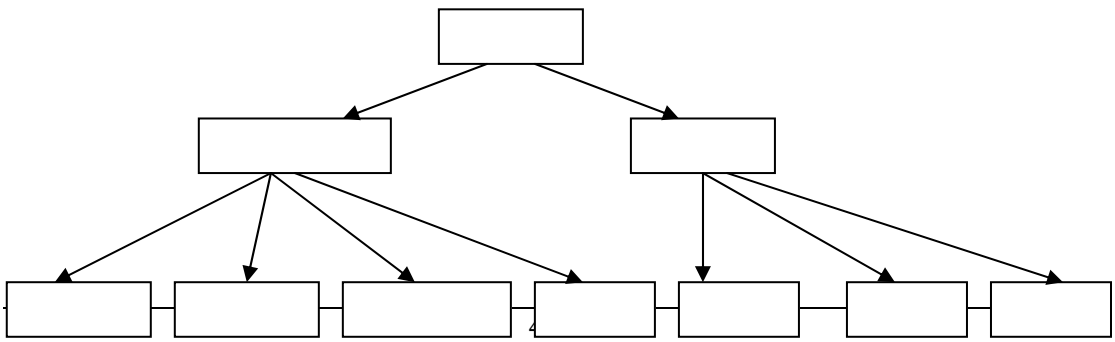
Smith <Library>

7.5

7.5.1

malware

malware



7-4

1 trap door

7.5.2

4

-
-
-
-

7.5.3

-
-
-
-
-
- word Microsoft Excel

7.5.4

-
-
-

1

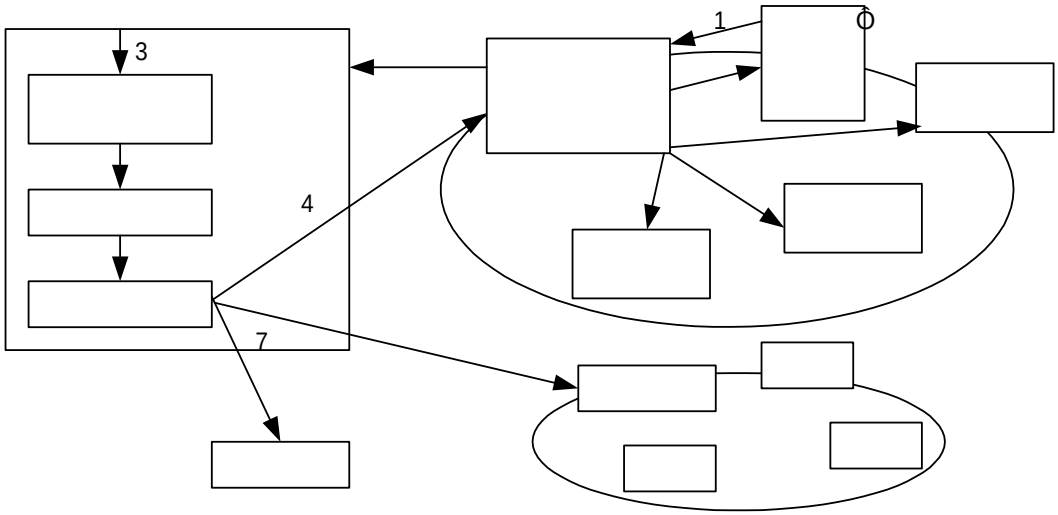
GD(General Decrypt)

GD GD

- CPU

-
-

GD


1
3
4
7

7.6

7.6.1

policy mechanism

?

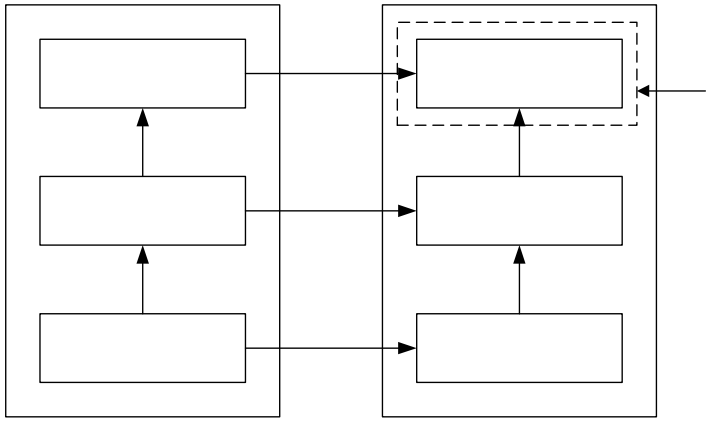
“ ”

——

-
-
-
-

1

7-6



7-6

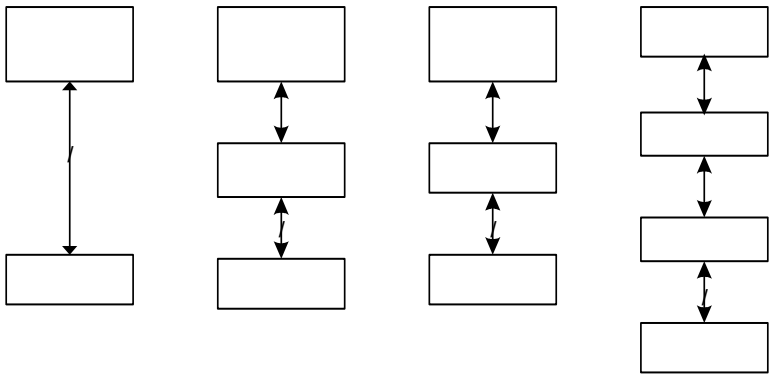
7-7

7-7 a b a

b

a

b



7-7

7-7 c d

c b c

b c

c b
d

2

Bell&Lapadula
3

( )

(information flow control)

D.E.Denning

Honeywell

Secure Ada Target

“ ”

4

(1)

(2)

(3)

(4)

(5)

(6)

( )

(multilevel security)

-
-

- **a**
- **b**

- a
- b

Φ

S O A

S
O
sclass(a)= s
oclass(o)= o
A(s,o)=
r = s o
w = s o
r,w = s o o
Φ = s o o
content o = o
subj=

a b contents(o)
subj
subj

state={S,O,sclass,oclass,A,contents,subj}

a b

a S o O
if r A(s,o),then sclass(s) ≥ oclass(o)
if w A(s,o),then oclass(s) ≥ sclass(o)

A

Creat_object(o,c)	o
Set_access(s,o,modes)	s o
Creat/Change_object(o,c)	o c o
Write_object(o,d)	contents(o)
Copy_object(from,to)	form to
Append_data(o,d)	d o

“ ’ ”

FUNCTION 1:Create_object(o,c)

If o O

Then ’O = O {o}

And

’oclass(o) = c

and

for all s S, ’A(s,o)= Φ

FUNCTION 2:Set_access(s,o,modes)

If o O and s S

And if {[r modes and sclass(s) ≥ oclass(o)] or r modes}

And

{[w modes and oclass(s) ≥ sclass(o)] or w modes}

then ’A(s,o) =modes

-
-
-
-

if conditions then ……

else conditions

Creat_object o

c

Set_access A

A(a,x) x ≠ o

→

Create_object

for all s S, ′A(s,o)= Φ

S_i mode r

mode sclass S_i ≥ oclass(o)

Create_object

sclass(s)

Create_object Set_access

Create_object Set_access

$S_0,O_0,Sclass_0,Oclass_0,Contents_0,Subj_0$

1 $S_0=\Phi,O_0=\Phi$

S ≠ Φ

2 For all s S_0,o O_0

$Sclass_0$ (s)=c_0

$Oclass_0$ (o)=c_0

A_0 (s,o)={r,w}

A_0

-
-
-

1

Create_object

FUNCTIION 3 Create/Change_object(o,c) ′ oclass(o)=c; and if o O then ′ o=O {o}; and for all s S,A(s,o)= Φ

- **c**

Create/change_object

1 For all o 0, ′ oclass(o) ≥ oclass(o)

2

“ ”

- **d**

subj

2 For all o 0
if r A(subj,o)
then for all s S, ′ A(s,o)=A(s,o)
Set_access

Set_access

3

FUNCTION 4 Write_object(o,d)
If o 0 and w A(subj,o)
Then ′ contents(o)=d
Write_object

w A(subj,o)
b

/

- **e**

3 For all o 0,
if w A(subj,o)
then ′ contents(o)=contents(o)

/

5 Bell&LaPadul

Bell&LaPadul
David Bell Leonard LaPadula
Case Western Reserve University
BLP
BLP

BLP
“ ”

“ ≥ ” “ ≥ ”

“ ” “ ≥ ” “ ≤ ”

“ > ” > “ ”

“ ”

“ ≥ ”

- ≥
- ≥ ≥
- ≥ ≥ ≥

≥

-
-

SYSTEM HIGH SYSTEM LOW

SYSTEM HIGH

SYSTEM LOW

BLP

BLP 20

-
-
-

BLP

/

/

/

Bell LaPadula Multics BLP

Multics

6

“ ”

“ ”

1

-
-

—

- I/O
-
-
- CPU
-

Saltzer Schroeder

“

”

2

( ) ( )

-
- 8:00 5:00
-
-
-

3

4

5

7.6.2

UNIX

UNIX

UNIX

“ ”

UNIX

“
”

“ ”

1

2

—— “ Morris
”
11.c 99

finger UNIX UNIX “ finger name@host

finger
finger Finger .plan
.plan
finger

Morris finger
finger finger
finger
shell

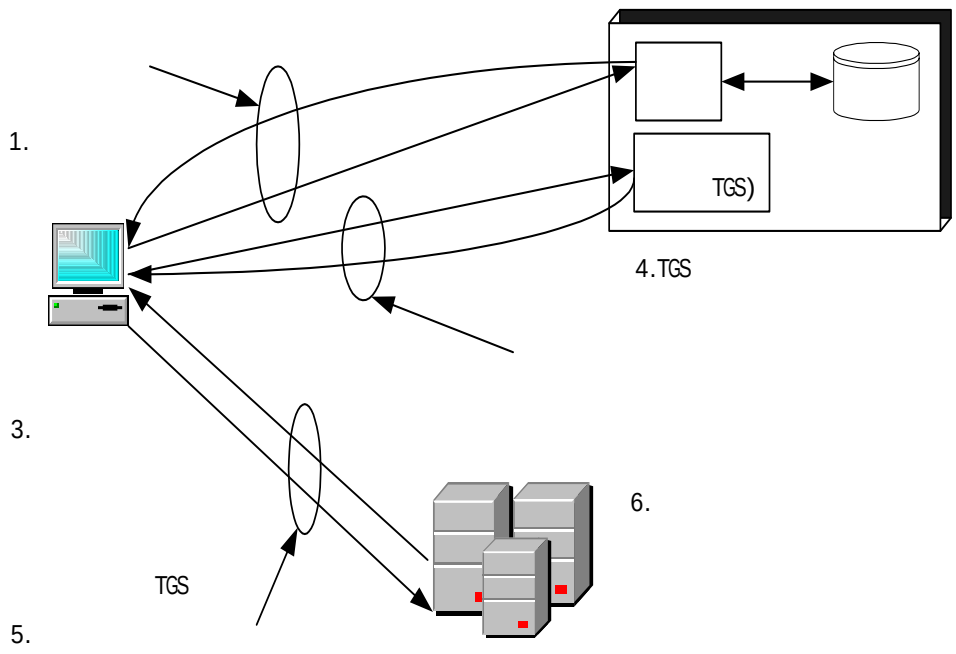
—

3 Kerberos

Kerberos MIT 1980
“ Athena ” “ — ”
Kerberos

Kerberos UNIX TCP/IP
Kerberos

- Kerberos
- Kerberos Kerberos
  Kerberos
-
-



7-8Kerberos

Kerberos

Kerberos

7-8

1

2

3

4

5

6

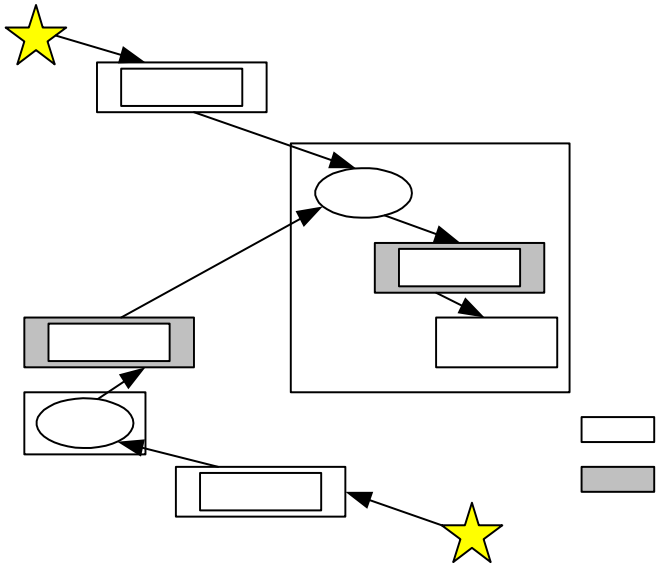
“ ” ___

1 6

7.6.3

1

7-9



7-9

shell

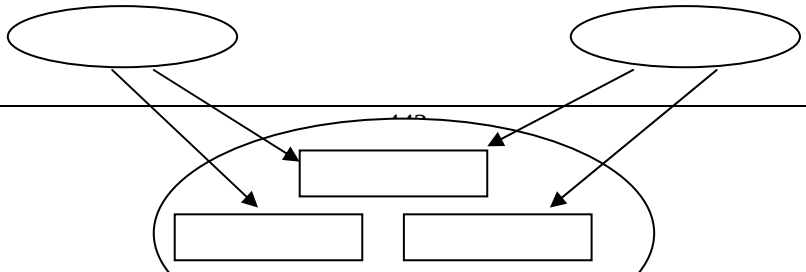
7-9

A W X Y Z 7-10

B W C W

B Y C Y C X B X A

Z



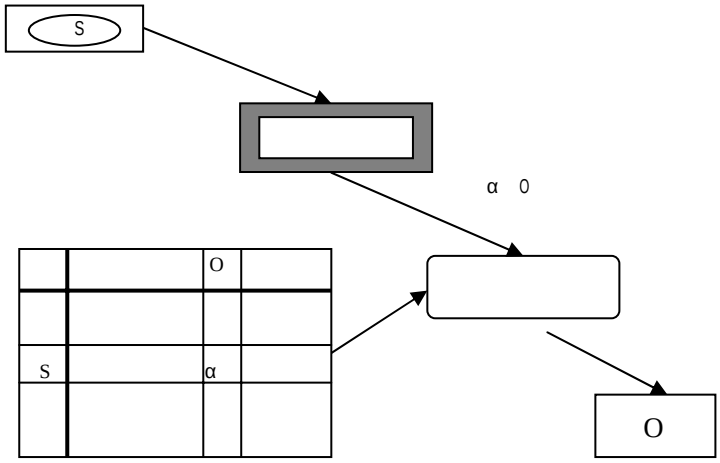
7-10

- ,

A (subject)
(object)
A[S O] S O A
i S_i j O_j
(S O A) (S O A)

7-12
1 0 α
2 S S S α 0

3 0 A[S 0] α A[S 0] α
A[S 0]



7-12

subjects = {S_1 S_2 S_3}
objects = subjects {F1 F2 D1 D2}
F1 F2 D1 D2 7-13
S_1 S_2 S_3
F1 S_1 * * S_2 F1 S_3 F1
S_2 F2
S_2 update F2 F2
A[S_2 F2] A[S_2 F2]
S_2 F2
S_2 executeF2 F2
A[S_2 F2] A[S2 F2]

	S_1	S_2	S_3	F1	F2	D1	D2
S_1				* *			
S_2					*		
S_3							

7-13

3

Graham Denning
7-14 7-13
S_0 A[S,O]
S_0 S_3 D2
A[S_0 O] A[S_3,D2]

	S0		
1 transfer{α|α*} to [S,O]	α* A[S_0,O]	A[S,O]=A[S,O]	{α|α*}
2 grant{α|α*} to [S,O]	owner A[S_0,O]	A[S,O]=A[S,O]	{α|α*}
3 deleteα from [S,O]	control A[S_0,S] or owner A[S_0,S]	A[S,O]=A[S,O]-{α}	

7.14
* S_0 O
α O α O S 7-14
S_1 F1 * S_2 S_3 S_1
2 S_0 O S_0

O

Graham Denning[1972] 7-14

- S
  X α X
- ---
- “ ”
  “ ”
  “ ” “ ”
- ---

4

-
-
-
-
-

(1)

——

N

[Organic 1972] Multics N

R_0 R_s R_{s+1} R_{n-1}

i<j R_i R_j R_0

R_1 R_{s+1}

R_0 Multics

—— R_i

R_j j>=i

R_0

Intel 80386

Intel 2 3

2

1 0

Intel 80486

0 ---

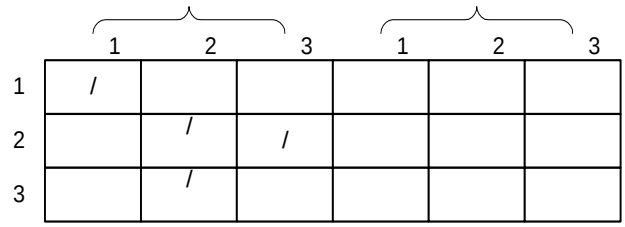
2

7-15VAX/VMS

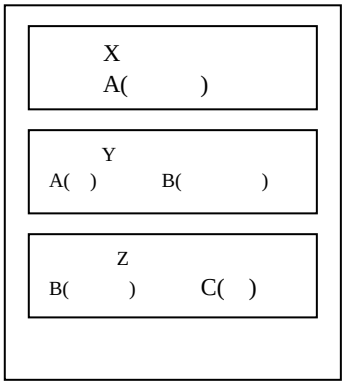
VAX/VMS

7-15

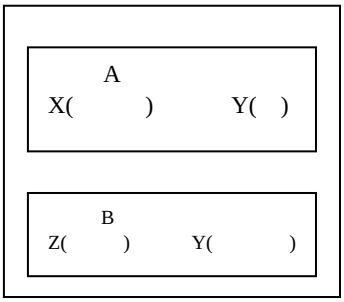
- kernel VMS
  I/O
- executive
- supervisor
- user



7-16



7-17a



7-17b

1

“ ” ACL Access Control List

1 A 2 B 3 C 1 X 2 Y 3 Z 7-17a

ACL

ACL

UNIX ACL UNIX UID GID UID GID

UNIX

“ d ” “ - ” 9

“ ” “ r ”

“ - ”

“ w ” “ x ”

setUID setUID

UID

Windows ACL [Solomon 1998]

ACL

2

“ ” CL(Capability List)

Kerberos

“ k i ” “ j

”

7-17b

(unforgeable)

- “ ”
-
-
-
-
-

```
struct capability{
     type tag
     long addr
}
```

c tag —c.tag address

—c.addr c.addr

c.tag

tag

tag

Burroghs tag

capability *other* tag

Mach Rochester Intelligent Gateway RIG Accent Mach

IPC

[Accetta et al. 1986]

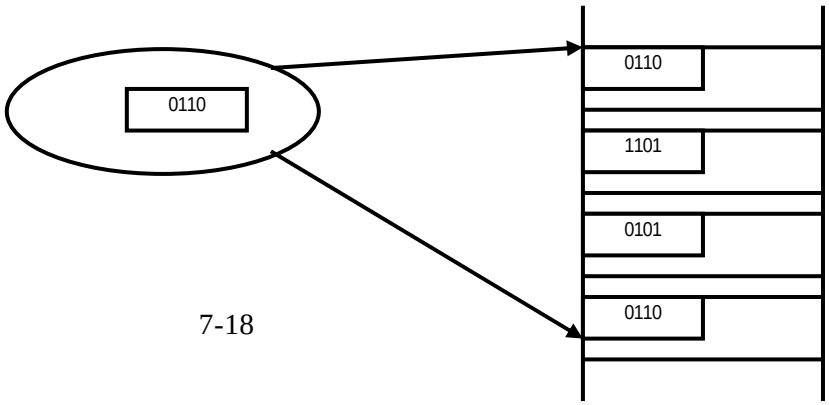
Mach IPC

(4)

ACL

20 70

ACL



7-18

20 60 h

k k 7-18

CPU

20 60 h=16 k=4
16

“ ” “ ”

IBM System/370
2KB
7 2
4 “ ” 1
“ ”

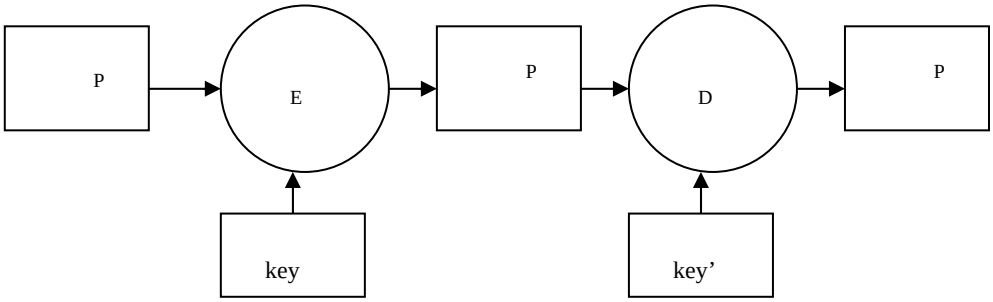
PSW
DMA I/O

1 15 0
“ ” “
” “ ”
1 “ ” 0

7.6.4

1

encrypt decrypt
decrypt key’ encrypt key plain text =plain text
encrypt key decrypt
key’ plain text cipher text
7-19



7-19

-
-
-
-

Kerckhoff

/

2

(1)

M	**E**	**G**	**A**	**B**	**U**	**C**	**K**
7	**4**	**5**	**1**	**2**	**8**	**3**	**6**
P	**l**	**e**	**a**	**s**	**e**	**t**	**r**
a	**n**	**s**	**f**	**e**	**r**	**o**	**n**
e	**m**	**i**	**l**	**l**	**i**	**o**	**n**
d	**o**	**l**	**l**	**a**	**r**	**s**	**t**
o	**m**	**y**	**S**	**w**	**i**	**s**	**s**
B	**a**	**n**	**k**	**a**	**c**	**c**	**o**
u	**n**	**t**	**s**	**i**	**x**	**t**	**w**
o	**t**	**w**	**o**	**a**	**b**	**c**	**d**

Please transfer one million dollars to my Swiss Bank account six two two …

AFLLSKSOSELAWAIATOOSSCTCLNMAN
TESILYNTWRNNTSOWDPAEDOBNO …

7-20

7-20

A 1 U 8

(2)

Julius Caeser

house krxvh

k

26 26 7-21

house qifsb

a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
↓	↓	↓							.....														↓	↓	↓
z	**x**	**c**	**v**	**b**	**n**	**m**	**q**	**w**	**e**	**r**	**t**	**y**	**u**	**i**	**o**	**p**	**a**	**s**	**d**	**f**	**g**	**h**	**j**	**k**	**l**

7-21 26

3

(1)

DES(Data Encryption Standard) 70

IBM

DES 20

DES DES

DES 64 DES
64 / 64
DES
64 8 64
56 DES
DES 1) 64
2) 56 16 3) 64 32
32 4)
DES 56 2^{56}
DES DES

(2)
DES

1976 Diffie
Hallman

-
-
- A B B
- B B

4

- A
B B A
A A
A

-
A
A B 1) A
1 2)A B
1 2 B 3)B
2 1 4)B A
1

5

?

- ( )

/ ?

/

-

-

-

?

KDC(Key Distribution Center)

ACC(Access Control Center)
A B (1) A
B (2) A B (3) A B
(4) A B C
C A B

7.6.5

(auditing)

( ) ( ) (
) ( )

7.7 Windows 2000/XP

7.7.1. Windows 2000/XP

Windows2000/XP
C2 1995 Windows NT Server Windows NT Workstation 3.5 NCSC United States National Computer Security Center C2 http://www.radium.ncsc.mil 1996 Windows NT Server Windows NT Workstation 3.51 ITSEC UK Information Technology Security Evaluation and Certification F-C2/E3 C2 http://www.itsec.gov.uk Windows NT 4.0 Windows2000 NCSC ITSEC

-
-
-
-
  Windows2000/XP

Windows2000/XP
ID
Windows NT4
Windows2000/XP

-
- Kerberos 5
  Internet
- Secure Sockets Layer 3.0
- CryptoAPI 2.0

7.7.2. Windows2000/XP

Windows2000/XP

- SRM NTOSKRNL.EXE
- LSA LSASS EXE

“ ”

- LSA

HKEY-LOCAL-MACHINE\security

- SAM
  LSASS
- SAM
  HKEY-LOCAL-MACHINE\SAM
- MSV1_0 DLL DLL
  Windows LSASS DLL
  SAM
- WINLOGON.EXE
  LSA
- SERVICES.EXE
  LSASS

7.7.3. Windows2000/XP

Windows2000/XP

Windows2000/XP

Windows2000/XP

7.7.4.

Windows 2000/XP Windows 2000 /

ID SID

-
-

SID

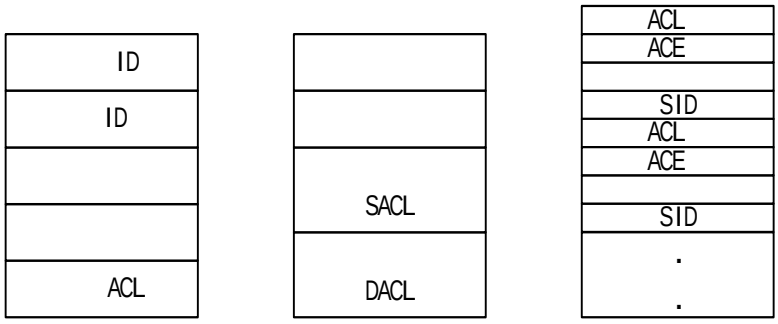
Windows 2000/XP
Windows 2000/XP

Windows 2000/XP /
RPC

7.7.5.

7.22 a

- ID
- SID ID
  SID
  SID SID
-
-
  SID
- ACL
  ACL



7.22 WINDOWS 2000/XP

7.7.6.

7.22 b

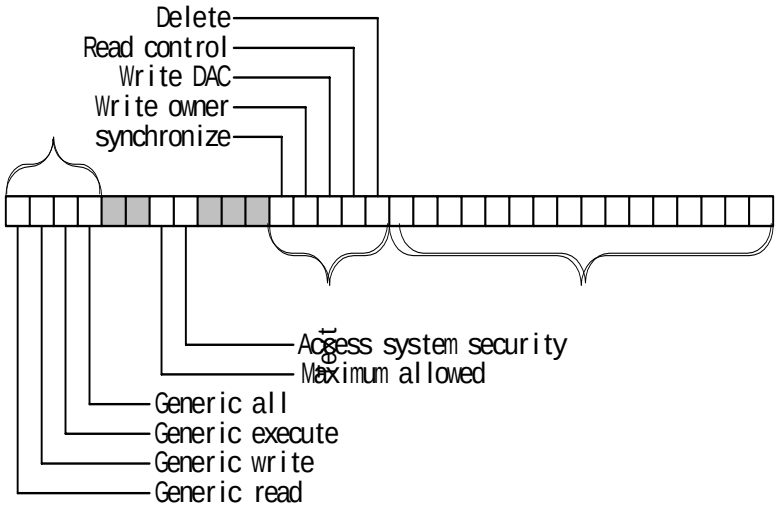
- SACL

DACL

RPC

- SID SID DACL
- SACL
SACL
SACL
- DACL
ACE
SID
SID SID

Windows 2000/XP
SID SID
SID Windows 2000/XP
SID SID DACL
ACE SID SID
ACE
7.23 16
0 File_Read_Data 0 Event_Query_Status



7.23

16 5

- Synchronize
- Write_Owner
DAC

- Write_DAC DACL
- Read_Control DACL
- Delete

4

ACE ACE
ACE ACE

- Generic_all
- Generic_execute
- Generic_write
- Generic_read

Generic_Read
Read_control Synchronize
File_Read_Data File_Read_Attribute File_Read_EA ACE
SID SID Generic_Read SID
5
Access_System_Security
SID ACE SID

Maximum_Allowed SID
DACL Windows 2000/XP Windows 2000/XP DACL
ACE ACE
DACL DACL
Maximum_Allowed

SID

Windows 2000/XP
Windows 2000/XP
/

/

7.8

I/O

DES

ISO RSA

( )

1

2

3
4 ?
5 ?
6

7 ? ?
8
9 ? ?
10 ?
11
12 ? ?
13 ?
14 ? ?
15 ? ?
16 ? ?
17 ?
18 ? ?
19 ? ?
20 ?
21 ?
22
23 ? ?
24 Bell&LaPadul
25 “ ” “ ”
26 ?
27 Kerberos
28
29
30
31
32 IBM ?
33
34
35
36
37 DES
38 ?
39 Windows 2000/XP ?
40 Windows2000/XP ?
41 Windows2000/XP
42 Windows2000/XP
43 Windows2000/XP

1 UNIX/Linux rwx
?(1)Rick Jennifer (2)Helen Anna
(3)Cathy UNIX/Linux

2

3 26 4
?

4 5000 4990
(1) (2)

5 A1 A2 A3 B1 B2
S

6
WJ

CH8

8.1

8.1.1

1

1 ——

2

——

3 ——

2

-
-
-

‘ ô ˈ ˌ č ő z ɯ ʮ ˇ

IBM SNA System Network Architecture Dec DNA Digital Network Architecture Univac DCA Data Communication Architecture Burroughs BNA Burroughs Network Architecture

1977 ISO “ TC97 ” “ SC16 ” “ OSI/RM

80 Internet OSI/RM

90 BISDN

1993 9 NII National Information Infrastructure “ ”

4

1.

1

2

2.

1

2

3.

1 WAN Wide Area Network

2 LAN Local Area Network

3 MAN Metropolitan Area Network WAN LAN

4 Intranet WAN LAN Internet Enterprise Network

4 (1) (2) (3) (4) (5)ATM

5

1 2 3

4 5

6

1

2

3

8.1.2

1

2

3

-
-
-

4

-
-
-

5

-
-

6

-
-

7

-
-
-
-

8

-
-
-

packet

- X.25
- ATM

9.

10

“ / b/s ” “ / Kb/s ” “ / Mb/s ”

8.1.3

1

N
N-1 N+1 N
N+1 N N+1
N N

Network Architecture

ARPA

X.25 ARPA
IBM DEC

1980 12
“
OSI/RM ” 1983 ISO7492
2

Network Protocol

-
-
-

3 **OSI/RM**
OSI

7

8-1 OSI/RM

1

bit

RS-232

2

-
-
-
-
-
-

X · 25

IP IP

IP IP X · 25

4

-
-
-
-
-

X · 25 IP

5

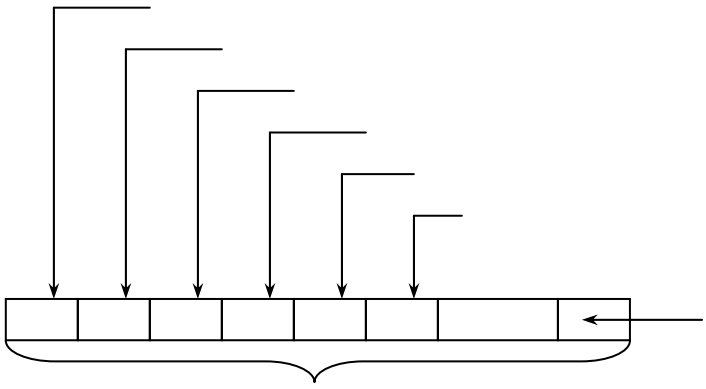
6

OSI

-
-
-

●
●
7
OSI OSI

OSI
FTP
FTP VIP MHS
JTM TP



8-2

OSI/RM
8-2

4 TCP/IP

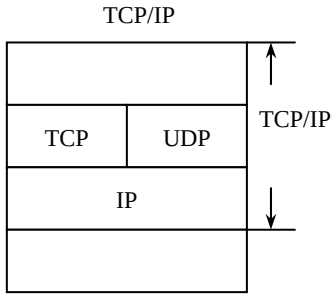
OSI/RM IBM SNA CCITT
X · 25 Novell IPX/ SPX Internet work packet Exchange/
Sequenced packet Exchange Microsoft windows NT
Internet TCP/IP 1983 ARPA
UNIX BSD TCP/IP OSI

TCP Transmission Centrol Protocol **I**P Internet Protocol
Internet TCP/IP TCP/IP
TCP IP TCP/IP
Internet

——

TCP/IP TCP
IP TCP

IP
IP IP
TCP/IP 8-3



8-3 TCP/IP

TCP/IP OSI
OSI OSI TCP UDP OSI
IP TCP/IP

TCP/IP
1 OSI
- SMTP Simple Mail Transfer Protocol ASC
- MIME Multipurpose Internet Mail Extensions
  ASC
- DNS Domain Name System IP
- Telnet Telecommunication Network
- FTP File Transfer Protocol
- NetBIOS Network Basic Input Output System

2 OSI
- TCP Transmission Control Protocol
- UDP User Data gram protocol
- NVP Network Voice Protocol
  OSI
- IP Internet protocol
- ICMP Internet Control Message Protocol
  IP
- ARP Address Reslovtion Protocol IP
- RARP Reverse ARP IP

8.2

8.2.1

windows NT MS-DOS OS/2 Windows 98 Windows for wrokgroup UNIX

WWW

1

UNIX

2 /

Netware Windows NT

C/S C/S

3

Netware Lite Windows for workgroup

8.2.2

UNIX
Windows NT Netware Netware Windwos NT

1 VINES
VINES Banyan System
PC VINES
UNIX system
802.X X.25 HDLC IP ARP ICMP VINES
RTP ICP TCP UDP VINES IPC SPP
LAN WAN street talk
SMP

2 Windows NT
Windows NT
Windows NT
32
SMP
Windows NT Microsoft SQL Server /
SQL Server NT

3 Netware
Novell Netware /
Netware Netware shell
Netware 5.0
Internet (1)
(2) GB-32TB
DOS OS/2 MAC UNIX (3)
(4)
(5) Internet

4 OS/2 Warp Server
IBM 1995 OS/2
Warp connect3.0 1996 Warp Server4.0 SMP
TCP/IP Internet DHCP DNS
Java IBM HTTP

8.2.3

Windows NT
Microsoft
Netware TCP/IP UNIX Windows NT

- I/O
-
-

Novell Banyan Sun NFS VINES NT

-

Windows NT

1

domain

/

Windows NT

NT

NT

workgroup NT

/

NT

NT

-
-
-
-
-
-
-

2 **NT**

NT

NT NTFS NTFS

3 **NT**

NT

NT

4 **NT**

NT Internet/Intranet

- DNS IP
  DNS Windows Internet
- Internet Internet IIS NT
  IIS NT Internet
- NT
  IP IPX LAN WAN
- IP
  TCP/IP
- NT NT

- (PPTP)
  Internet Windows NT RAS WAN
  NT RAS Windows NT

- DCOM NT COM
  Internet

5 NT

8.3

8.3.1

“

”Single Computer System Image

-
-
-
-

“ ”

“ ”

-
-
-
-
-

Distributed Operating System

-
-
-
-

8.3.2

RPC Remote Procedure **socket**

1

/

send receive

1

0

i 1 2 BSD UNIX

“ ” “ ”

16 32

“ ”

64

“ ”

ASC

2

- Send(P, Message)
- Receive(Q, Buffer)

P Q Message Buffer

send

receive

3

receive(addr,&m)

addr

m

receive send receive

send

receive

? ?

receive

receive

receive

—

receive

4

2

RPC

/

/

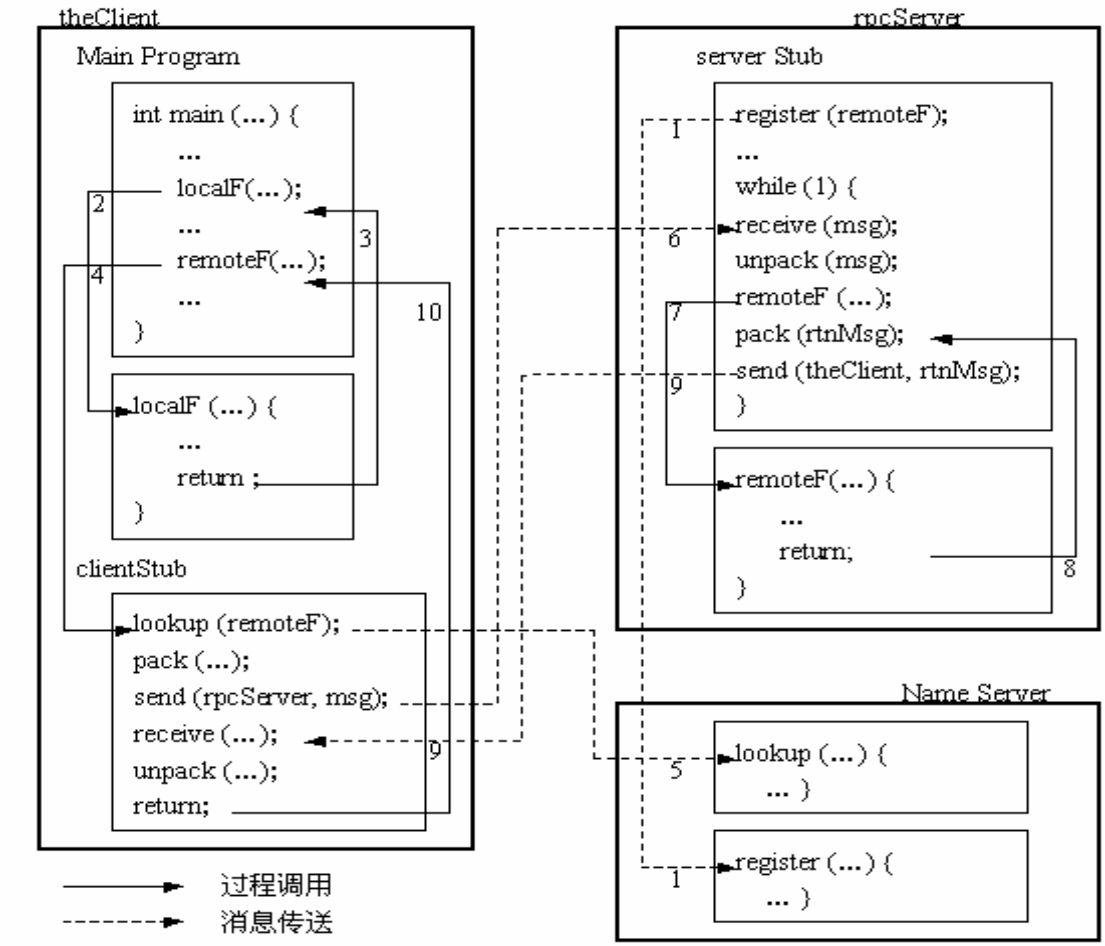
stub

RPC

RPC

Q¸╪AÁËDWYâ╣qCQ4]Q¥ãbŒ

server stub



8-5

RPC

RPC

ASCII 0 1

3

UNIX BSD socket

socket

socket

socket

socket socket

socket

Socket

socket C/S / socket

socket socket socket

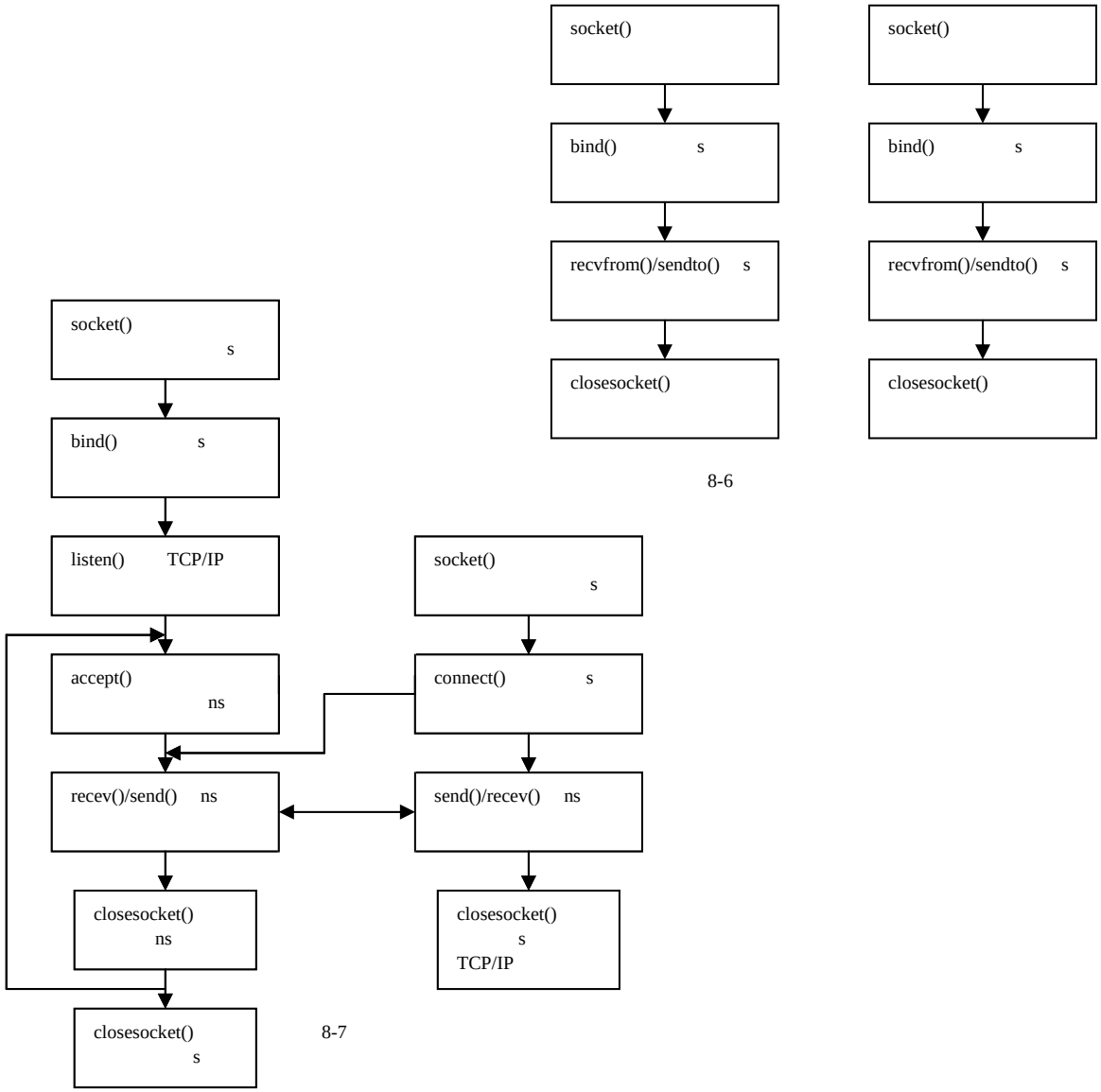
socket C/S

socket

socket TCP/IP
UDP
Socket socket socket()
bind() socket connect() listen() accept()
send() sendto() sendmsg() write() writev() read() readv() recvfrom()
recvmsg()
socket TCP/IP TCP/IP
C/S
8-6 8-7

fork

Windows2000/XP



8-6

8-7

8.3.3

č ώ ̓ ο

- p
- p

  p

  p
- p

- **3**
-
-

  0
-

  S( )

- S=0
- S=w*a+1 a
- w
-

  S S
- S=0 Sr=0 0
- S=min(Sr1+1,…,Sre+1,Sr)
- ,Sr Sr1,…,Sre

  S Sre+1 Sre

  S Sr

- S

-
-
-

8.3.4

1

1978 Lamoprt
Lamport

“ ” “ $a \rightarrow b$ ” “ a b ”
a b “ ”

(1) a b a b $a \rightarrow b$
(2) a b a $\rightarrow$ b
(3) c $a \rightarrow c$ $c \rightarrow b$ $a \rightarrow b$
(1) “ ” (2)

(3) “ ”
“ ”

P_1 P_2 P_3

a f x y

$x \rightarrow y$ $y \rightarrow x$

timestamping

C a

b C(a) C(b) C(a) C(b) a b

a b a

b

C

(1) P e_j e_j P 1

$C(e_j)=1$

e_j P j j-1 e_{j-1}

$C(e_j)=C(e_{j-1})+1$

(2) P e_r e_r P 1

$C(e_r)=1+C(e_s')$

- e_s' P' e_r P r r-1

e_{r-1}

- $C(e_r)=1+\max[c(e_{r-1})\ C(e_s')]$

n

i C_i

1 (m,T_i,i) m

T_i i

j (2)

1

1

i x j

y x y

(1)$T_i < T_j$

(2) $T_i = T_j$ $i < j$

Lamport

8-8

P_0

6 P_1 8 P_2

10

6 P_0 A P_1

P_1 16

A 6 P_1 10

B P_1 P_2 16

C P_2 P_1 60

56 D P_1 P_0

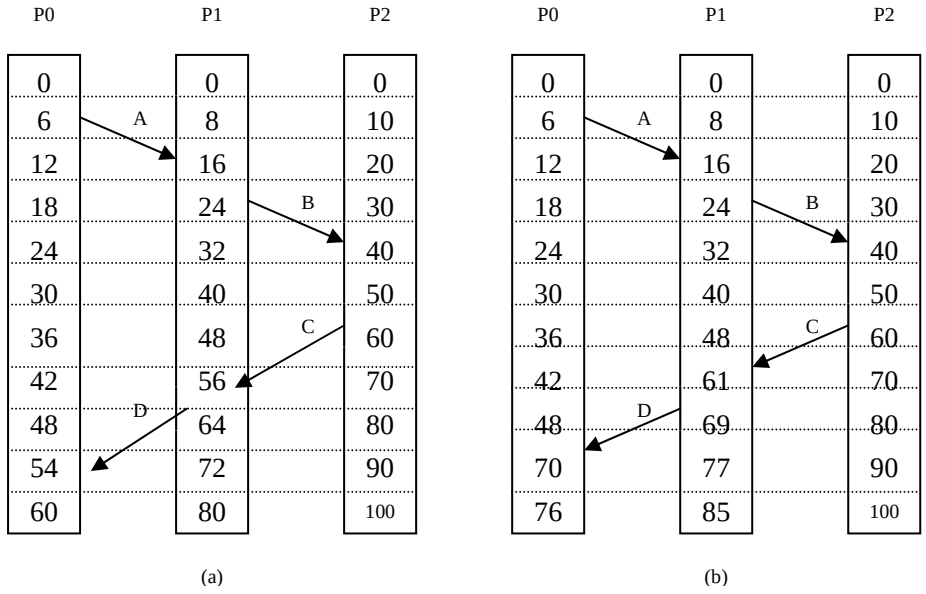
64 54

Lamport “ ” C

60 61 61

1

8-8(b) C 61 D 70



8-8 (a)

(b)Lamport

8-9

P_1 0 a 1 (a , 1, 1)

2 3 0

2=1+max[0,1] P_2 3

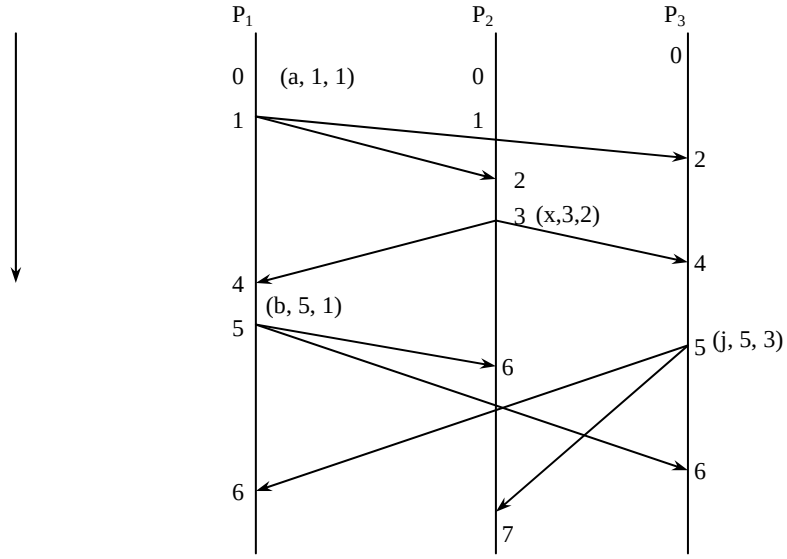
P_1 P_3 4

P_1 b P_3 j

a x b j

8-10 P_1 P_4 P_1 2 P_4 3 P_1 P_4

a q 3 a q ? P_1 P_4 (T_1 =1 T_4=1) i < j P_1 P_4 3 a q



8-9

2

(1) Lamport

Lamport
FCFS

Lamport N
Lamport

1~N

applicationstack =(release 0 i) i=1,…,N

- (request T_i i) P_i
- (reply T_j j) P_j
- (release T_k k) P_k

Lamport

(1) P_i
(reguest,T_i,i) applicationstack[i]

(2) P_j (reguest,T_i,i)
applicationstack[i] P_j (reguest,T_i,i)
P_j

(3) P_i
1) P_i applicationstack request

(T_i,i)

(4) P_i applicationstack [i] (reguest,T_i,i)
(release, T_i,i)

(5) P_j P_i release applicationstack[i]
P_i (reguest,T_i,i)

3(N-1) (N-1) request (N-1) reply (N-1) release

```
/*                                          */
type   message=record
       class: (application,reply,release);  /*                                  */
       source: 1..n;                        /*              */
       timestamp:integer;                   /*                  */
       clock:integer;                       /*                */
       valid:Boolean;                       /*                */
end;
/*                                         */
var
     T:integer;                                  /*              1*/
     applicationstack:array [1..n] of message;/*                                  */
     replycount:0..n-1;                          /*            */
procedure   Apply;                               /*                    */
    var M:message;
        i:integer;
    begin with M do
        begin   class:=application;          /*              */
                source:=me;                  /*me               */
                timestamp:=T                 /*          */
                valid:=true;
        end;
        T:=T+1;
        applicationstack[me]:=M;     /*                          */
        replycount:=n-1;             /*        n-1                  */
        For   i := 1   to   n   do
        begin   M.clock:=T;T:=T+1;
                If i ≠ me then send(M,i)/*              i          */
        end;
       waitfor(Replycount=0);              /*                  n-1   */
       for i:= 1   to   n   do             /*                        */
       waitfor (not Applicationstack[i]→Applicationstack[me]);
    /*      " →"
```

```
        get  resource;                    /*                */
        end;
    procedure Receive(var M:message;);
      /*                    waitfor                          */
     var R:message;
     begin T:=1+max(M.clock,T);
          with M do
          case  class  of
     application:
          begin applicationstack[source]:=M;
                with R do
           begin
                class:=reply;
                source:=me;
                clock:=T;
           end;
           T:=T+1;
           send(R,source);              /*                */
       end;
     reply:
         replycount:=replycount-1;
     release:
         applicationstack[source].valid:=false;
     end;
     procedure Release;                 /*                */
       var R:message;
           i:integer;
      begin  with R do
                begin class:=release;
                source:=me;
                clock:=T;
      end;
        T:=T+1;
      applicationstack[me].valid:=false;
      for i:= 1  to  n  do
          if i ≠ me then send(M,i);     /*                */
      end;
```

(2) G.Ricart

Ricart release

2(N-1)

(1) P_i (reguest,T_i,i)

(2) P_j (reguest,T_i,i)

1) P_j

(reply,T_j ,j) P_i

2) P_j

$T_i < T_j$ $T_i = T_j$ i < j

(reply,T_j ,j) P_i reply

(3) P_i reply

(4) P_i

reply

```
type message=record
        class: (application,reply );        /*                    */
        source: 1..n;                       /*          */
        timestamp:integer;                  /*                */
end;
var
        T:integer;                          /*              1*/
        Applicationtime:integer;            /*          */
        Replydeffered:array [1..n] of  Boolean;
    /*      i                                   Replydeffered[i]          */
        Replycount:0..n-1;                  /*          */
        Requesting:Boolean;                 /*          */

procedure  Apply;                           /*                                */
Var M:message;
begin  with M do
      begin class:=application;
            Timestap:=T;
            Source:=me;
      end;
      applicationtime:=T;
      T:=T+1;
      requesting:=true;
      replycount:=n-1;
      broadcast(M);                         /*          */
      waitfor(replycount=0);                /*       replycount=0*/
end;
procedure Receive(var M:message;);
    /*                  waitfor                                    */
begin with M do
      case class of
```

```
application:
      replydeffered[source]:=
      requesting and ((timestap>applicationtime) or
                        (timestap=applicationtime and source>me));
      /*       source                                          */
      T:=1+max(T,timestap);
      If not replydeffered [source] then
      sendreply(source);                  /*                  source*/
reply:
       replycount:=replycount-1;
end;
procedure Release;                        /*                          */
var   i:integer;
  begin for i:= 1 to n do
             if replydeffered[i]   then
             begin replydeffered[i]:=false;
                  sendreply(i);          /*          */
            end;
            requesting:=false;
 end;
```

8-11

Ricart

request reply

request reply

reply

request request

reply reply 2(N-1) (N-1) request

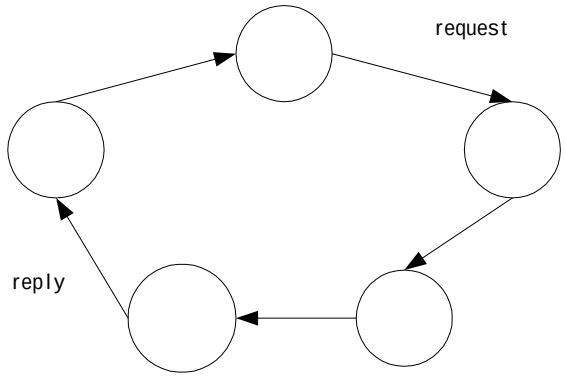
Pi (N-1) reply

request

reply request reply

request reply

request request request



8-11 Ricart

(3)

Suzuki 1982

Logical Ring

Token

K K+1

8.3.5

1

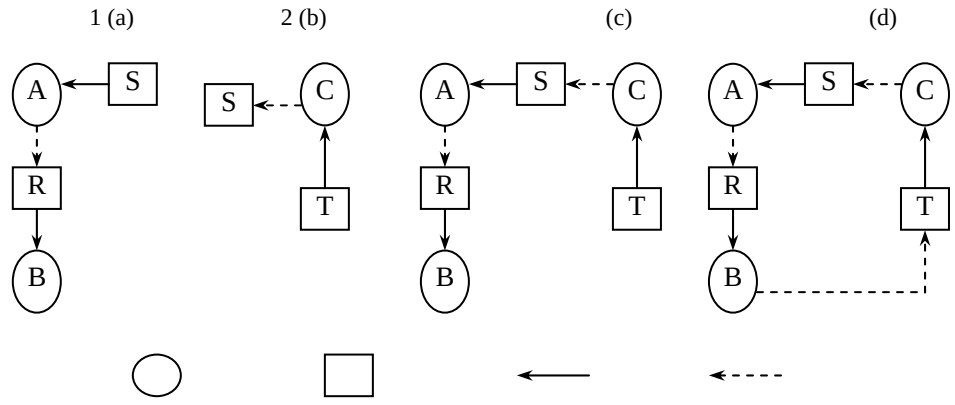
A B B C C A

2

1

A B 1 C 2 R S T
8-12(a)(b)

- A S R R B
- B R
- C T



8-12

8-12 c B
A R C S
B R T 1
B T 2 1
8-12(d

Lamport 2
1 2
1 2
“ 2
T
” R
B

2

1
2
3

4

“ ”

8.3.6

1

-
-

File Service **File server**

2

vfs vnode
Sun NFS

-

-
- **XDR**
-
-

RPC

-
-
-
- **I/O** I/O
-

3

/

+

/

—— NFS

1 NFS

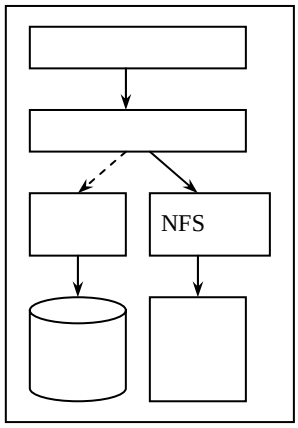
NFS

NFS

NFS

/etc/exports

8-13 NFS Sun



NFS

A B

NFS

2 NFS

NFS

NFS /

NFS

i

/etc/rc Shell

Shell

Sun UNIX

/etc/rc /etc/rc

NFS

NFS

NFS

NFS UNIX open close
open close

open lookup
read

NFS UNIX UNIX

NFS
NFS
NFS UNIX
rwx NFS

NIS NIS
yellow page

3 NFS
NFS SUN NFS
open read close
VFS

VFS
UNIX i UNIX i i
VFS v i

mount open read v
mount

(1) Mount

mount mount

v NFS
r i v r VFS
v NFS r
i v

(2) Open

v r
NFS NFS

r VFS v
r
Open VFS v

(3) / Read/Write read
VFS v i r

8192
8K

30

NFS

UNIX

NFS

NFS

8.3.7

1

1 Data Migration

A B

B A

A

B

A

B A

B A

B Sun

NFS Microsoft NETBU

2 Computation Migration

RPC

3 process Migration

(1)

(2)

(3)

(4)

(1) (2) (3)

IBM AIX UNIX

-
-
- “ ”

“ ”

-

8.4 Windows2000

8.4.1 Windows 2000

1. Windows 2000

Windows 2000 API
Windows 2000 I/0
TDI
Windows 2000 API Windows 2000 Windows 2000

(1) API APl

API

(2) TDI Transport Driver Interface
API TDI
I/O IRP IRP Windows 2000

(3) TDI TDI
IRP IRP TDI TCP
UDP IPX IRP
NDIS
TDI

(4) NDIS Ndis.sys Windows 2000
NDIS TDI

(5) NDIS NDIS miniport driver
TDI NDIS Windows
2000 NDIS Consumer Windows NDIS
IRP NDIS NDIS
TDI NDIS
NDIS HAL
Windows 2000 OSI/RM
TDI 4 3

2. API

Windows 2000 APl
API API

-
- Windows Winsock
- RPC
- CIFS
- NetBIOS

(1)
named pipe mailslot 0S 2
API Windows NT
Windows
2000 APl Windows 2000

Windows 2000 UNC
UNC Windows
1)

\\ServerXPipe\PipeName
Server DNS
mspress microsoft com NetBIOS mspress IP 255.0.0.0
Pipe “Pipe” PipeName
\\MyComputer\Pipe\MyServerApp\ConnectionPipe
CreateNamedPipeWin32
\\.\Pipe\PipeName “\\.\” Win32

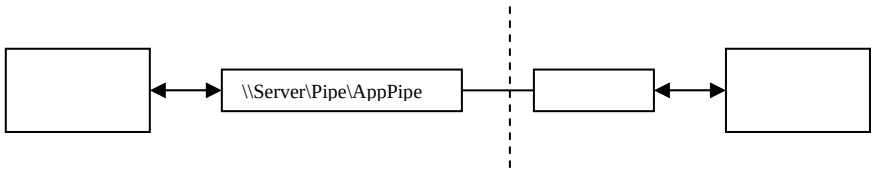
APl

CreateNamedpipe
CreateNamedPipe

ConnectNamedPipe Win32
CreateNamedPipe

Win32CreateFile CallNamedPipe
ConnectNamedPipe

ConnectNamedPipe
ReadFile WriteFile Win32
8-14



8-14

API
ImpersonateNamedPipeClient
2)

Win32 API CreateMailslot

CreateMailslot “\\.\Mailslot\MailslotName”

CreateMailslot
CreateMailslot

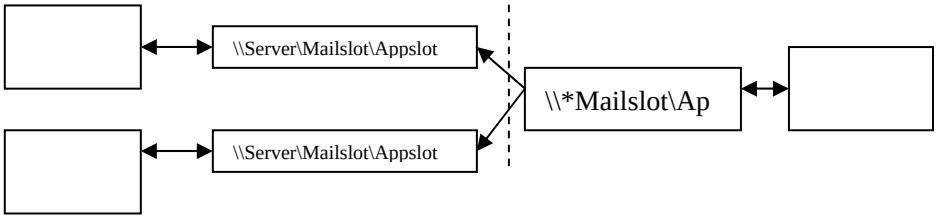
CreateMailslot CreateNamedPipc
ReadFile

CreateFile
“ Server Mailslot MailslotName ”

“ * Mailslot MailslotName”
“ Domain VlailslotLMailslotName”

WriteFite
425 426

425 426 424 Windows 2000 425
426 8-15



8- 15

3)
Win32
Kernel32.dll Win32 DLL ReadFile WriteFile
Win32 I/O
CreateFile Win32I O
\Winnt\System32\Npfs.sys
\Winnt\System32\Drivers\Msfs.sys
\Device\NamedPipe
\??\Pipe Device Mailslot
\??\Mailslot \\.\Pipe\... \.Lmailslot\...
CreateFile \\.\ \??
CreateNamedPipe CreateMailslot
NtCreateNamedPipeFile NtCreateMailslotFile

FSD FSD

- FSD Windows 2000
- CreateFile FSD
- Win32 ReadFile WriteFile
- FSD
- FSD

FSD
CIFS CIFS IPX TCP/IP NetBEUI

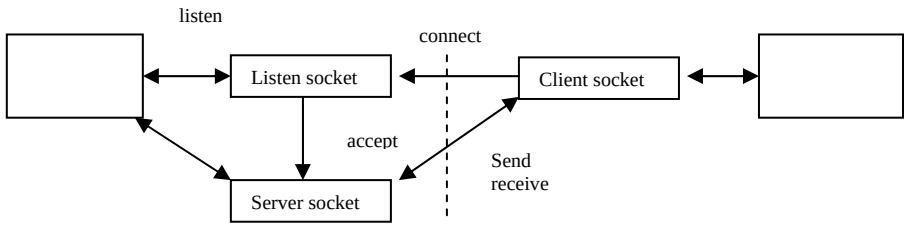
(2) Windows
Windows Winsock BSD
API 20 80 UNIX Windows
2000/XP UNIX Windows 2000/XP
Winsock BSD
Winsock
Windows 2000/XP Winsock 2.2
ConsumerWindows
Winsock I/O
QoS QoS
Windows 2000 Winsock
Winsock

1)Winsock
Winsock APl Winsock

Winsock — API
Winsock

Winsock listen
accept accept

accept — receive
send
Winsock connect
8-16 Winsock



2)Winsock
Windows Winsock API Windows
Winsock — Windows
Windows
— I/O
BSD
Winsock WSA WSAAccept
BSD Winsock
AcceptEx TransmitFile Windows 2000 Web
AcceptEx accept
AcceptEx Web
Winsock accept
Web
TransmitFile Windows 2000

TransmitFile
Web

3) Winsock
Windows 2000 Winsock APl
Winsock Winsock
Winsock SPl KWinsock
Winsock Winsock
connect accept

Winsock /
TCP/IP
Web
Web IP 80 80 HTTP
Web
—

Winsock gethostbyaddr getservbyname getservbyport

4)Winsock
Winsock 8-17 API DLL
(Ws2_32 dll) Winscok Ws2_32.dll
Msafd.dll Winsock
WinsockHelper
Wshtcpip.dll TCP IP Helper Wshnetbs.d11 NetBEUI Helper Winsock
Mswsock.dll TransmitFile AcceptEx WSARecvEx Windows
2000 TCP IP NetBEUI AppleTalk IPX SPX ATM IrDA
Help DLL DNS TCP IP IPX SPX

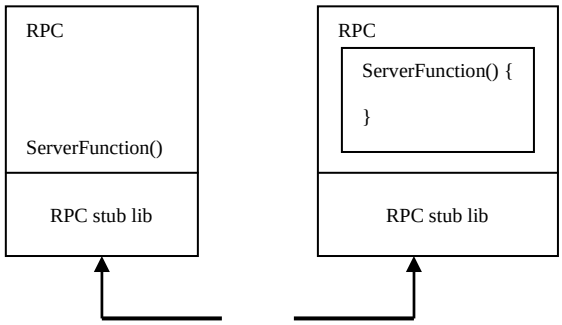
API Winsock Win32 I O

Msafd.dll Afd sys AFD TDI
TDI IRP
AFD Msafd.d11 AFD
AFD

(3)
RPC 20 80
OSF RPC DCE
unRPC RPC RPC OSF DCE RPC
API

1)RPC RPC

I O Windows 2000
Ë O ↰ Z O



8-18 RPC

RPC

I O

RPC Windows 2000 RPC

RPC

Cray

RPC

RPC

DLL

DLL DCOM

RPC

RPC

Windows 2000 RPC

RPC RPC RPC

RPC

Win32 WaitForSingleObject

RPC RPC

MIDL MIDL RPC

C C++

MIDL MIDL

RPC

RPC

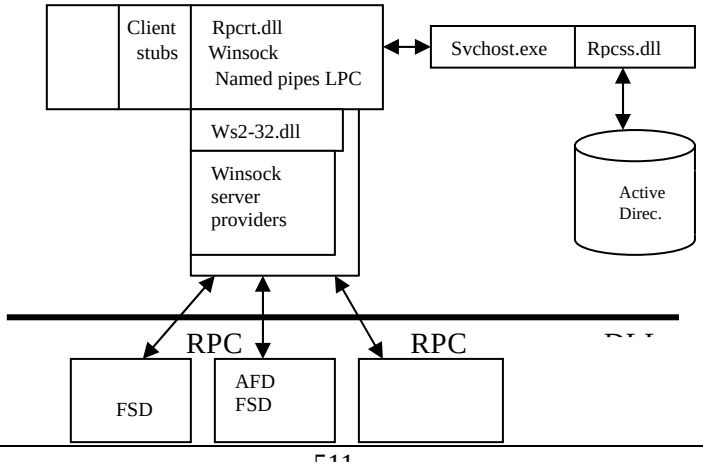
RPC RPC RPC
RPC Windows 2000
RPC NetBIOS TCP IP DLL
DLL
Windows 2000 RPC

RPC RPC
NetBIOS WindowsNT4 RPC

2)RPC
Windows 2000 RPC SSP
RPC SSP SSP

RPC RPC

SSP RPC Winsock
Windows 2000 SSP Kerberos SSP Kerberos 5
Secure Channel SChannel SecureSocketsLayer SSL
TransportLayerSecurity TLS PCT SSP
RPC
RPC RpclmpersonateClient
RpcRevertToSelf
RpcRevertToSelfEx
3)RPC
8-19 RPC RPC RPC DLL
\Winnt\System32\Rpcrt4.d11
RPC DLL RPC RPC
DLL RPC RPC
RPC RPC DLL
LPC API RPC
RPC DLL Winsock API



RPC RPC Winnt

8-19 RPC

System32\Rpcns4.dll DLL RPC RPCSS- Winnt System32\Rpcss.d11
RPC Win32 RPCSS RPC
RPCSS

(4)

CIFS SMB
Windows 2000 Win32 I/O
CIFS CIFS
I/O CIFS Windows 2000 CIFS
CIFS API
CIFS Win32API
CIFS Windows 2000
Windows 2000 Samba UNIX
Windows 2000 UNIX
Windows 2000 CIFS DECVMS AppleMacintosh
Windows 2000 FSD FSD
FSD FSD
Win32 I/O CIFS

FSD FSD Windows 2000 I/O
TDI
FSD
Win32 I O CreateFile ReadFile
WriteFile
Windows 2000 FSD FSD

FSD UNC

1)CIFS
Windows 2000 FSD /
Winnt Sysytem32 Drivers Rdbss.sys FSD
FSD
FSD FSD TDI
CIFS Winnt Sysytem32 Drivers
Mrxsmb.sys
Mrxsmb.sys CIFS
TDI APl CIFS
TDI API NetBEUI NetBT TCP IP NetBIOS TCP IP
FSD \Winnt\System32\Drivers\Srv sys
CIFS FSD
Windows 2000FSD FSD
FSD MDL
FSD MDL TDI TDI
TDI Ã ¹ ^

Windows 2000 oplock

- I
-
-

(5)NetBIOS

20 90 NetBIOS API PC API NetBIOS Windows 2000 NetBIOS NetBIOS NetBIOS API Winsock Windows 2000 TCP IP NetBEUI IPX SPX NetBIOS

1)NetBlOS

NetBIOS 16 NetBIOS NetBIOS UNIQUE GROUP UNIQUE NetBIOS GROUP NetBIOS GROUP

Windows NetBlOS 16

WindowsNT4 Windows 2000 NetBIOS DNS 15 mspress.Microsoft.com NetBIOS mspress Windows 2000 NetBIOS

NetBIOS LANA NetBIOS LANA TCP/lP NetBEUI LANA LANA NetBIOS LANA

LANA NetBlOS

Windows Windows InternetName Service WINS NetBIOS IP WINS NetBl0S Windows NetBIOS NetBIOS DNS DNS DNS Windows 2000 NetBIOS

2)NetBIOS

NetBIOS NetBIOS API LANA NetBIOS LANA NetBlOS

NetBIOS

NetBIOS
NetBIOS session
NetBIOS
NetBIOS NetBI0S
MS-DOS NetBI0S MS-DOS
NetBIOS MS-DOS MS-DOS NetBIOS
NetBIOS
Windows 2000 Netbios
NetBIOS
3)NetBIOS API
NetBIOS \Winnt\System32\Netapi32.dll Netapi32.dll
NetBIOS NetBIOS emulation driver
Winnt\System32\Drivers\Netbios sys Win32 DeviceloControl
NetBIOS NetBlOS TDI

TCP/IP NetBIOS NetBIOS
NetBT \Winnt\kSystem32\Drivers\Netbt.sys NetBT TCP/IP
NetBIOS TCP IP NetBIOS NetBIOS
NetBEUI NetBIOS NetBEUI NetBIOS
NetBT TCP/IP NwLinkNB
IPX/SPX NetBIOS

3.

UNC
Win32 WNet API

CIFS CIFS
CIFS FSD Workstaion
Novell NetWare Windows 2000

- Multipleproviderrouter MPR DLL
  Win32 WNetAPI
- UNC MultipleUNCProvider MUP
  Win32I O API
  Windows 2000 —— DNS

1)
Win32 WNet Windows

WNETAPI

provider Windows 2000
WNet
WNet DLL Workstanon
DLL

WNet MPR.DLL MPR
MPR DLL
provider interface MPR
WNet FSD
\Winnt\System32\Ntlanman.dllk
HKLM\SYSTEM\CurrentControlSet\Services\lanmanworkstation\NetworkProvJder
ProviderPath
WnetAddConnection API MPR
HKLM\ SYSTEM\CurrentControlSet\Control\Network Provider\Order\ProviderOrder
MPR

ProviderOrder

WNetAddConnection

2) UNC
UNC MUP MPR UNC
I/O \\
MPR MUP MPR
MUP
UNC MUP
Win32 DLL Kernel32.dlt I/O API DLL
UNC “\??\UNC” NtCreateFilc
“\??\UNC” \Device\Mup MUP

MUP
IRP

\\WIN2KSERVER\PUBLIC\insidew2k\chapl3.doc
\\WIN2KSERVER\PUBLIC MUP

MUP
MUP
ProviderOrder I/O
3)
DNS IP
DNS IP TCP IP DNS DNS
DNS ‘ IP ’ IP
zone DNS
DNS Windows 2000 Windows 2000
Windows 2000DNS Win32 Winnt System32 Dns.exe
Windows 2000 DNS
Windows 2000 DNS

4.

API API API
API
API
Windows 2000 DLC NetBEUI TCP IP NWLink
Windows2000
ServicesForMacintosh AppleTalk

- DLC IBM HP
  “ ” API DLC
  DLC
- IBM 1985 NetBEUI NetBEUI LAN
  NetBIOS API NetBEUI
  WAN NetBIOS NetBEUI
  NetBIOS Extended UserInterface NetBIOS
  NetBEUI NetBEUI NetBIOS
  NBF Windows 2000 NetBEUI Windows

- è

accept

- TDI connect IRP IRP
  TDI connectIRP

Winsock connect
TDI UDP TDI TDI
TDI TDI TDI
TDI TDI

TDI

5. NDIS

1989 3COM
NDIS
NDIS NDIS NDIS
gsã# &r ÀÃ
Windows 2000 NDIS NDIS 5
Windows 2000 NDIS

Windows 2000 ATM Windows 2000 NDIS NDIS API NDIS

NDIS NDIS 5

- NDIS

ATM 3.1(UNl

- MCM

MCM NDIS MCM NDIS LAN LANE ATM

8.4.2 WindowS 2000

Windows 2000 APl FRS DFS Windows 2000 TCP IP NAT 1PSec QoS

1

Windows 2000

Windows 2000

-

- VPN VPN IP Windows2000

2

LDAP Windows 2000 Windows 2000 Windows 2000

Windows

APl

- LDAPC API C API LDAP C C++ API API
- ADSI COM LDAP

ADSI MicrosoftVisual Basic C MicrosoftVisual C++ ADSI Windows WSH

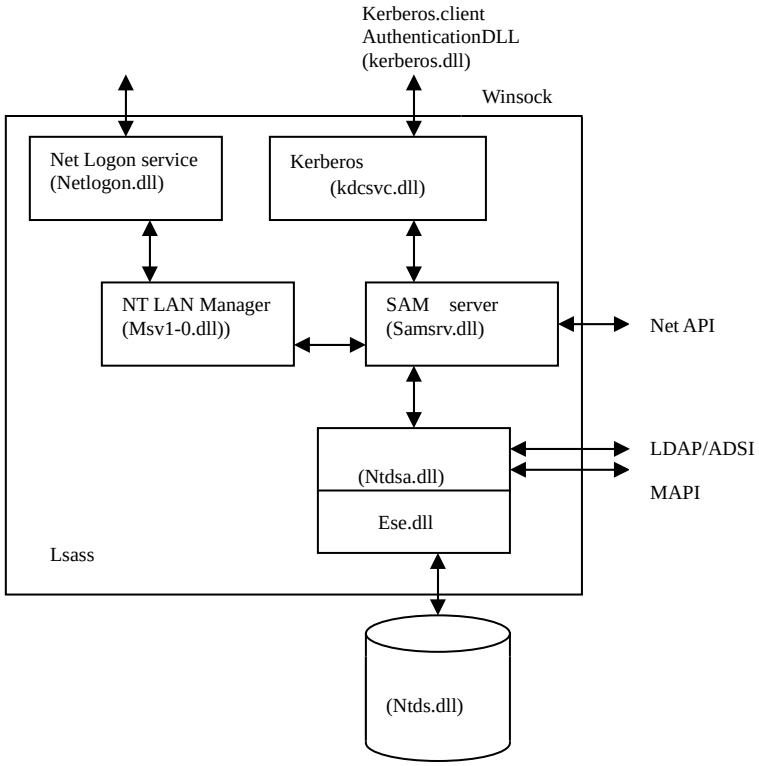
- API MAPl MicrosoftExchange OutlookAddressBook
- SAM API MSVl_0 \Winnt\System32\Msvl_0.dll NT LAN Kerberos \Winnt\System32\Kdcsc d11
- WindowsNT4 API NetAPl NT4 SAM

WinntkNtds Ntds dit

Lsass Win32 on-disk DLL

ESE

MicrosoftExchangeServer5.5 8-20



8-20

3

Windows 2000 AdvancedServer NDIS 32

IP

NDIS

TCP IP

TCP IP

——

Windows

4

Windows 2000 Server FRS \SYSVOL
\SYSVOL Windows 2000
FRS
DFS FRS

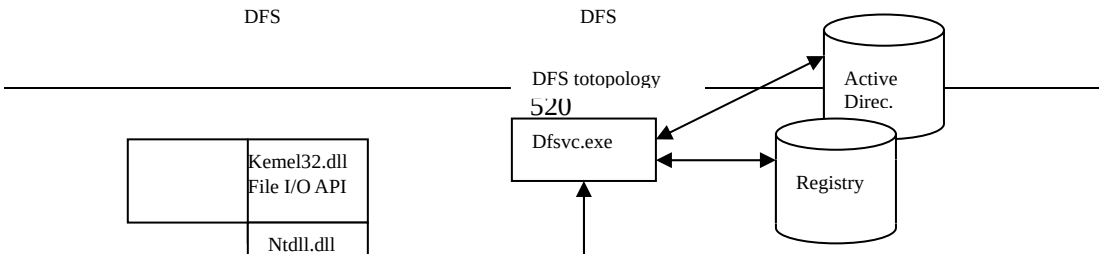
FRS
NTFS
FRS NTFS
FRS

FRS Win32 \Winnt\System32\Ntffs.exe
RPC FRS API
FRS

5

DFS
DFS
DFS
Windows2000Server
DFS DFS
FRS DFS
FRS
DFS
DFS
DFS 8-21 DFS Win32
\Winnt\System32\ Dfssvc.exe \Winnt\System32\Drivers\Dfs.sys
DFS DFS
DFS DFS

DFS MUP NetWare CIFS
DFS I O
MUP DFS



6 TCP IP

Windows 2000 TCP IP
TCP IP
NAT 1PSec QoS

1)
NAT IP IP
NAT IP
NAT IP
NAT IP

Windows 2000 NAT TCP IP NAT
NAT
MMC
NAT Windows 2000

2)
1PSec Windows 2000TCP IP IP
IP
IPSec

- IP
- IP
-
-

Windows 2000 IPSec Kerberos 5 IPSec IPSec Windows 2000 CryptoAPI IPSec
IPSec IPSec Winnt System32kDriversklpsec.sys
TCP IP
IPSec IPSec IPSec
IP IPSec IKE
IKE IPSec
IPSec

3) QoS

IP

QoS

ERP QoS

QoS

IEEE 802.lp QoS OSI

Windows 2000 QoS Winsock API

Winsock QoS WSCInstallQOSTemplate

QoS APl TC API

Windows 2000 QoS RSVP Win32 \Winnt\System32\Rsvp.exe RSVPWinsock \Winnt\System32\RsvpMil RPC RSVP RSVP TC API TC API \Winnt\System32\Traffic.dll I O GPC \Winnt\System32\Drivers\Msgpc.sys GPC QoS NDIS \Winnt\System32\Drivers\Psched.sys QoS

QoS QoS

8.5

OSI/RM

TCP/IP

Windows NT

C/S

Lamport

Lamport Ricart

Sun NFS

1 2 3

1. ? ?
2.
3.
4.
5. ?
6. ? ?
7.
8. / / /
9. ?
10. OSI/RM
11.
12. TCP/IP
13.

14.
15.
16. ?
17. ” ?

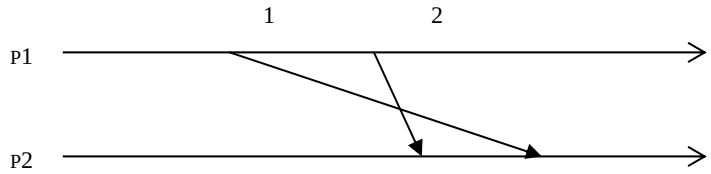
4 0 Tc ( )<41a913d7115e1d00349817e4/TT2 1 Tf 19 0 TD 7

3 8-8(b) A A

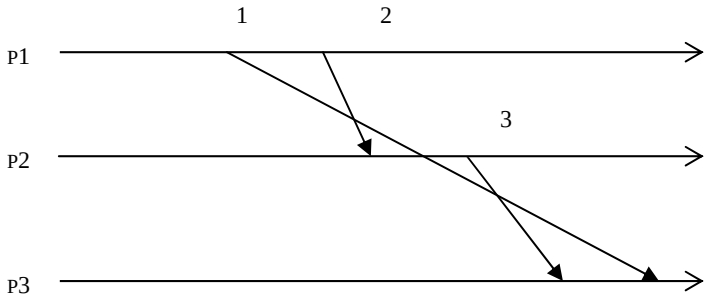
A

4

( )



5



1. Operating Systems Internals and Design Principles,William Stalling (Fourth edition) Prentice-Hall International , Inc. ,1998
2. Operating System Concepts ,Abraham Silberschatz (Bell Labs) , Peter Baer Galvin (Cor. Tech. Inc.) (Fifth edition) , John Wiley & Sons, Inc. ,1997
3. Distributed Systems Principles and Paradigms Andrew S.Tanenbaum,Maarten van Steen,Prentics Hall 2002
4. Applied operating System Concepts ,Abraham Silberschatz , Peter Baer Galvin, John Wiley & Sons, Inc. ,1998
5. Computer Network (Third Edition) Andrew S.Tanenbaum Prentics-Hall