Design and Implementation of Embedded Linux System for Networking Devices

Hyun-Joon Cha

Distributed Processing and Network Management Laboratory
Division of Electrical and Computer Engineering
(Computer Science and Engineering)
tachyon@postech.ac.kr

Contents

- Introduction
- Current Embedded Operating Systems
- Requirements
- Design of Embedded Linux System
- Implementation
- Conclusions
- Future work

Introduction

- Networking Devices
  - Devices which has networking capability
  - Infrastructure of emerging information society
    - e.g.) Router, Switch, Gateway, Cache engine, Cellular phone, PDA, etc.
  - Network-capable devices will substitute current dummy and not-connected devices all around
  - Need more resources, processing power and OSs to coordinate it
  - Most networking devices use commercial Real-time OSs

Introduction – cont'd

- Embedded OSs for Networking Devices
  - Commercial: VxWorks, pSOS, QNX, Nucleus, LynxOS, VRTX, etc.
  - Free or Almost Free: Xinu, uC/OS, etc.

- Frequently Raised Problems from Industry and Academy
  - No OS approach or using educational OS is harmful
  - High purchase price and royalty -> affect development cost and device price
  - Limited target and development platform
  - OS specific architecture and interface
  - Technology dependency on specific OSs
  - Hard and take much time to get help
  - Hard to apply new networking technology
  - Really high entrance fee

Introduction – cont'd

- Embedded Linux System
  - Basically, stripped down Linux - to fit the constraints of devices - as an kernel for embedded systems
  - Additionally, uses it's own idea of kernel modification, library, file system, applications with respect to the application area
- Advantage
  - Linux is inherently modular and easily scaled down
  - No run-time royalties
  - Is proven to be sophisticated, efficient, robust and reliable
  - Superior network capability, excellent file system support
  - Supports a vast array of processors
  - Standard and widely supported APIs
  - Has enormous amount of open-source software, documents and developers
  - Technical support from Open-Source Community of all around world
  - It's truly a global standard

Introduction – cont'd

- Objectives
  - Design and Build Embedded Linux System
  - Optimize the Embedded Linux for Networking Devices
  - Apply leading-edge networking technologies
  - Implement to the Real-World Application
  - Evaluate it with Other Commercial Appilcations

Develop an Embedded Linux that has Full Features and Optimized for Networking Devices

Current Embedded Operating Systems

- Free or Almost Free Operating Systems

Name	CPU	Lang.	Min. ROM/RAM	Sched.	MMU	Modular	Multi Proc.	POSIX	Inst. Base	Price
Xinu	x86, SPARC, i960, 68k, PowerPC	Asm., C/C++	Unknown	Fixed Priority	No	No	No	No	Unknown	Free
uC/OS-I,II	Most	Asm., C/C++	4~8KB / 3KB	Fixed Prioriy	No	Yes	No	No	2,000	$50
eCOS	ARM, MIPS, SPARC, PowerPC	Asm., C/C++	Unknown	Unknown	No	Unknown	No	Yes	Unknown	Free

Current Embedded OSs – cont'd

- Commercial Operating Systems

Name	CPU	Lang.	Min. ROM/RAM	Sched.	MMU	Modular	Multi Proc.	POSIX	Inst. Base	Price
pSOS+	68k, ARM, M Core, ColdFire, i960, MIPS, PowerPC, SH, SPARC, x86	Asm., C/C++, Java	15KB / 5KB	RR, Time Slice, Cooperative, Rate Monotonic	Yes	Yes	Yes	Yes	7,000	$16,500
QNX	MIPS, PowerPC, x86	Asm., C/C++	32~64KB / 8KB	RR, Time Slice, Fixed Priority	Yes	No	Yes	Yes	One Million	$795
VRTX	68k, ARM, PowerPC, x86, M Core	Asm., C/C++, Java	Unknown	RR, Time Slice, Fixed Priority	Yes	Yes	No	Yes	50,000	$1,000

Current Embedded OSs – cont'd

- Commercial Operating Systems – cont'd

Name	CPU	Lang.	Min. ROM/RAM	Sched.	MMU	Modular	Multi Proc.	POSIX	Inst. Base	Price
VxWorks	PowerPC, 68k, ColdFire, M Core, x86, i960, ARM, MIPS, SH, SPARC	Asm., C/C++, Java	Unknown	RR, Fixed Priority	Yes	Yes	Yes	Yes	Unknown – Very Large	$90,000 ~$200,000
Nucleus PLUS	68k, ARM, M Core, ColdFire, i960, MIPS, PowerPC, SH, x86	Asm., C/C++, Java	3~45KB / 2KB	Time Slice, Cooperative	Yes	Yes	Yes	No	2,300	$7,495 ~$12,495
OS-9	68k, ARM, PowerPC, SH, x86	Asm., C/C++, Java	128~6,000 KB / 128KB	RR, Time Slice, Rate Monotonic, Cooperative	Yes	Yes	No	No	Three Million	Unknown

Requirements

- Resource Requirements of Typical Networking Devices

	Tiny	Midrange	High-end	Embedded PC	Embedded Server
Flash	100KB ~ 4MB	2 ~ 8MB	8 ~ 32MB	16 ~ 64MB	64+ MB or disk
RAM	500KB ~ 2MB	2 ~ 4MB	4 ~ 16MB	16+ MB	64+ MB

- Embedded operating system and applications must meet tight computing resource requirements of embedded system

Requirements – cont'd

- Software Size or Quality
  - Networking applications need more memory resources
  - But, memory price does not hold large factor in current networking devices
  - Ensuring software quality is more valuable than reducing size

- Real-Time Consideration
  - Hard real-time and Soft real-time
  - System operation does not fail even if it can't process any networking operations
    - Hard real-time feature is not a must for networking devices

Requirements – cont'd

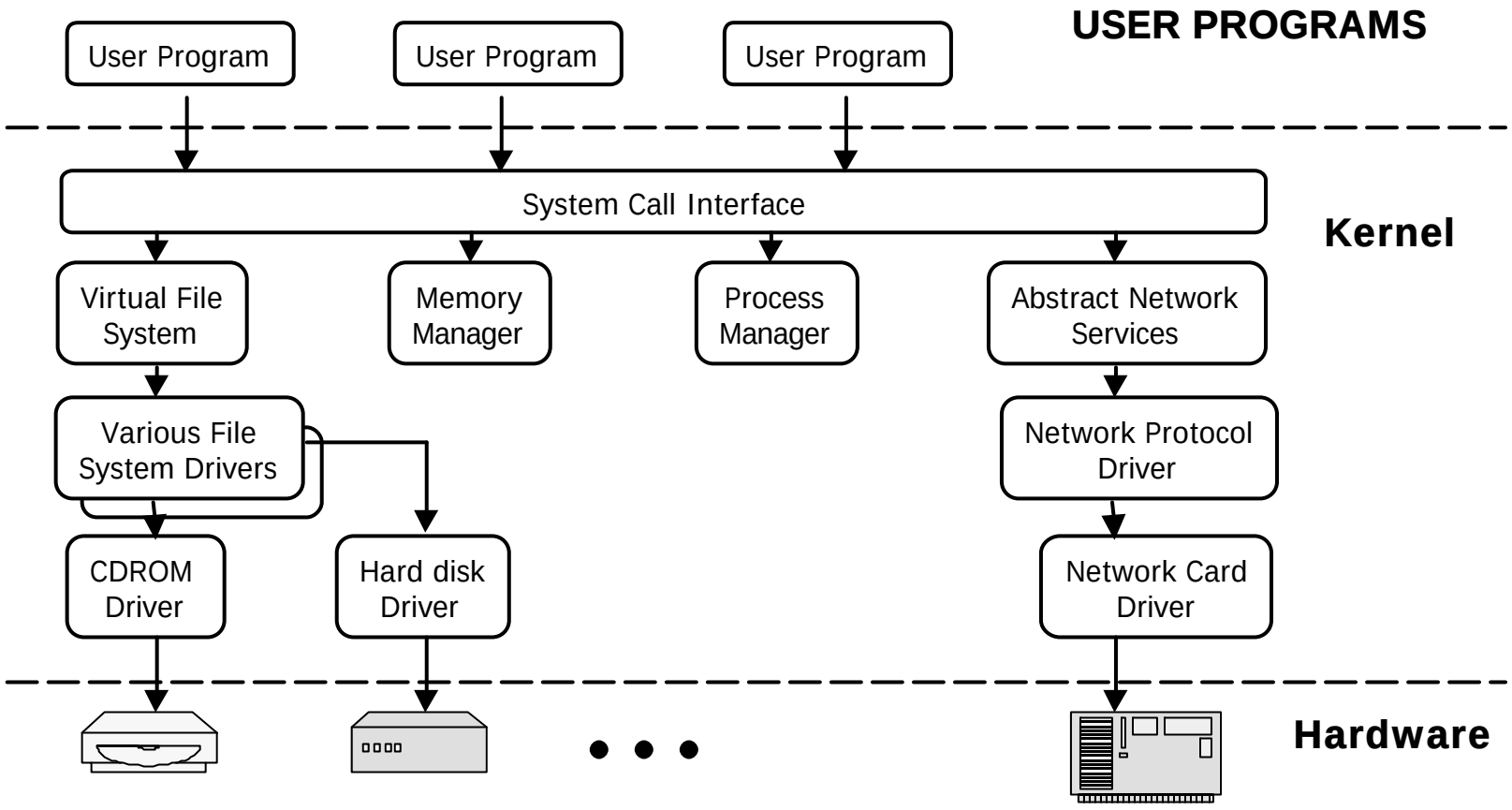
- The Requirements
  - Portability to new processors
  - Scalability to match varied applications
  - Multi-processor support
  - Extended services support
  - Existence of several application programs
  - Standard/POSIX compliance
  - Language support
  - Well-suited development environment
  - Flexible licensing arrangements and price

Design

- Satisfying Requirements
  - Linux kernel have been ported to many processors and most codes are made up of portable language
  - Is highly scalable
  - Supports SMP up to 16 processors
  - Have many kernel extensions which are suitable for networking applications
  - Have large number of open source software
  - Is POSIX compliant
  - Supports almost all programming languages
  - Have well-suited development environments
  - Is licensed with GPL which is an approved license of Open Source Initiative

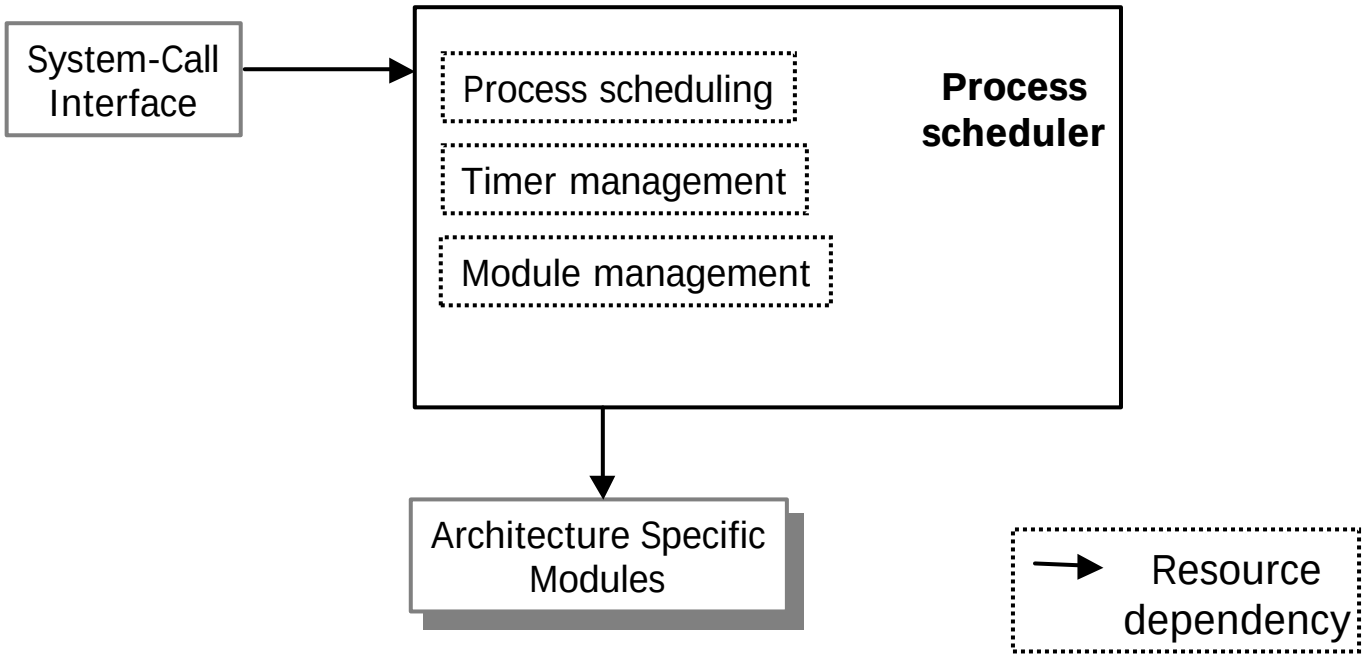
Design – cont'd

- Conceptual Architecture of Linux Kernel



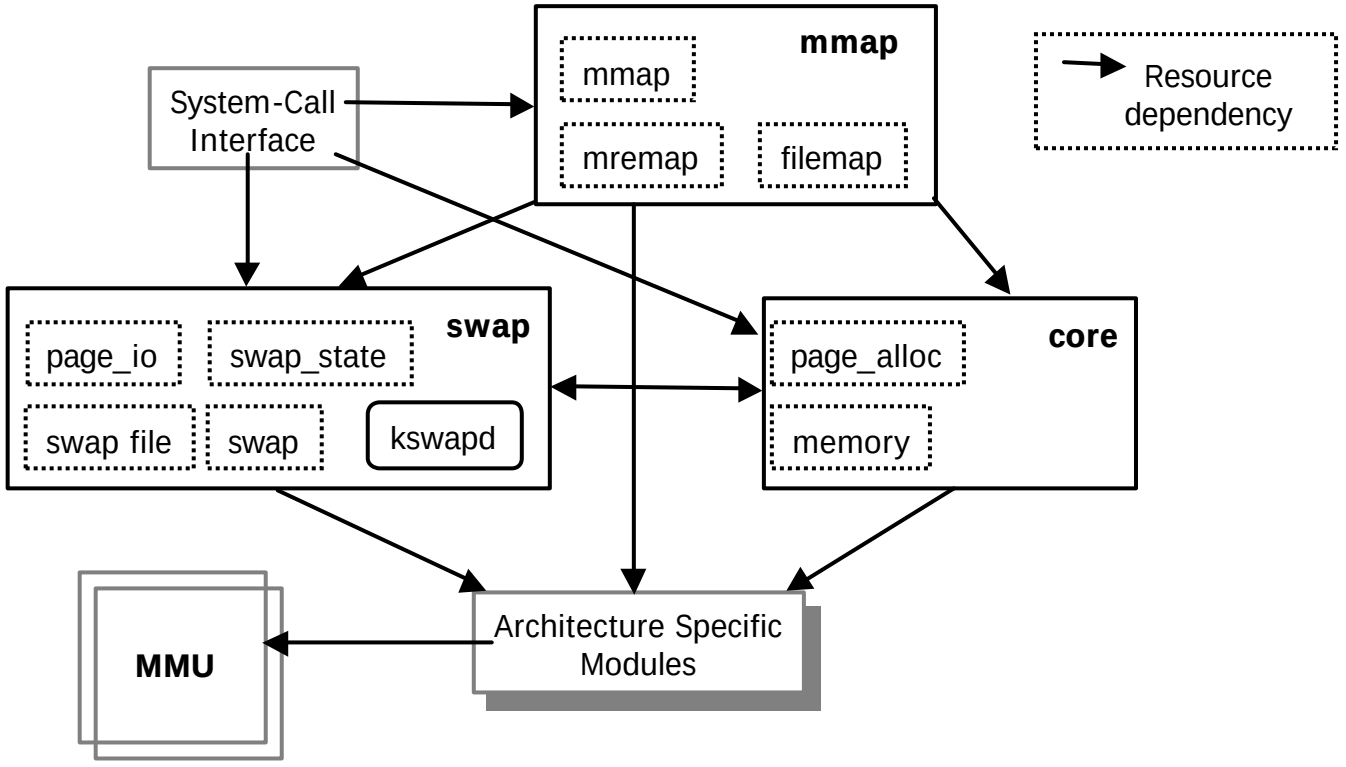
Design – cont'd

- Resource Dependency between Components of Process Scheduler



Design – cont'd

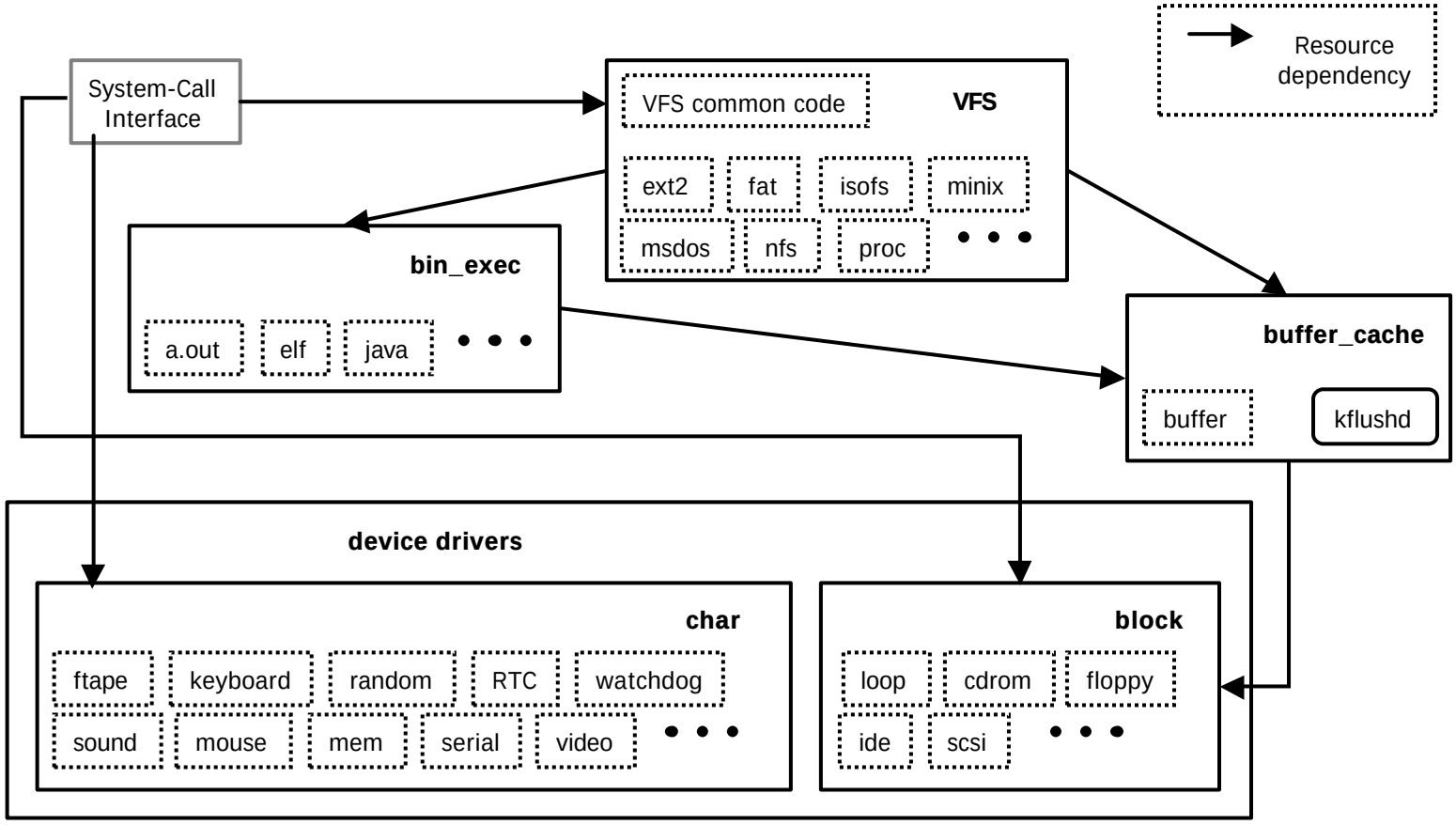
- Resource Dependency between Components of Memory Manager



- There are a lot of microcontrollers which doesn't have MMU

Design – cont'd

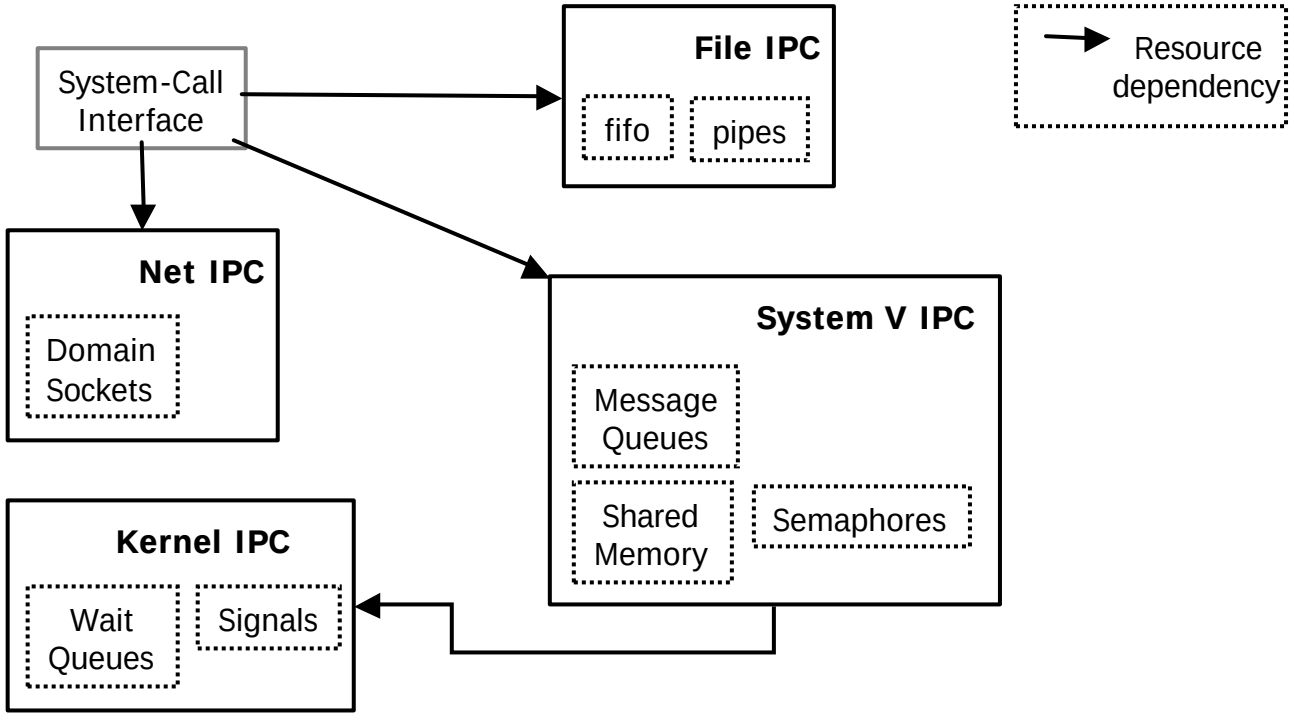
- Resource Dependency between Components of Virtual File System



- There is a complicated relationship between device driver interface and file system

Design – cont'd

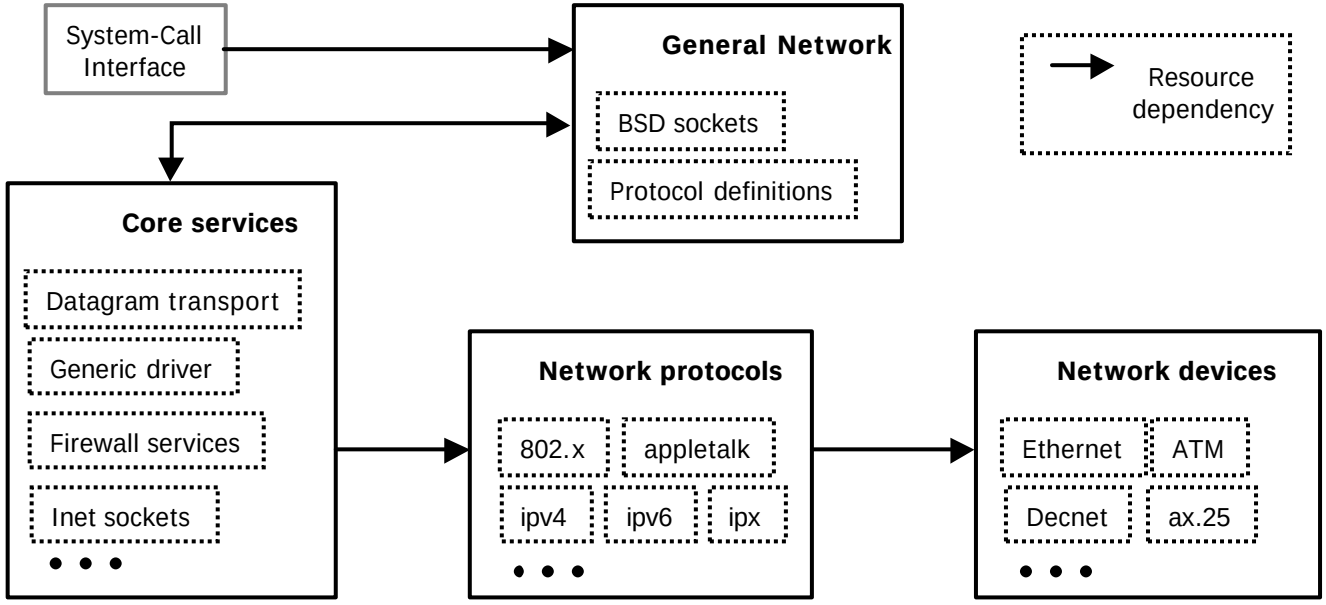
- Resource Dependency between Components of Inter Process Communication Subsystem



•To port Open-Source applications, IPC is important even if logical dependency doesn't exist

Design – cont'd

- Resource Dependency between Components of Network Subsystem



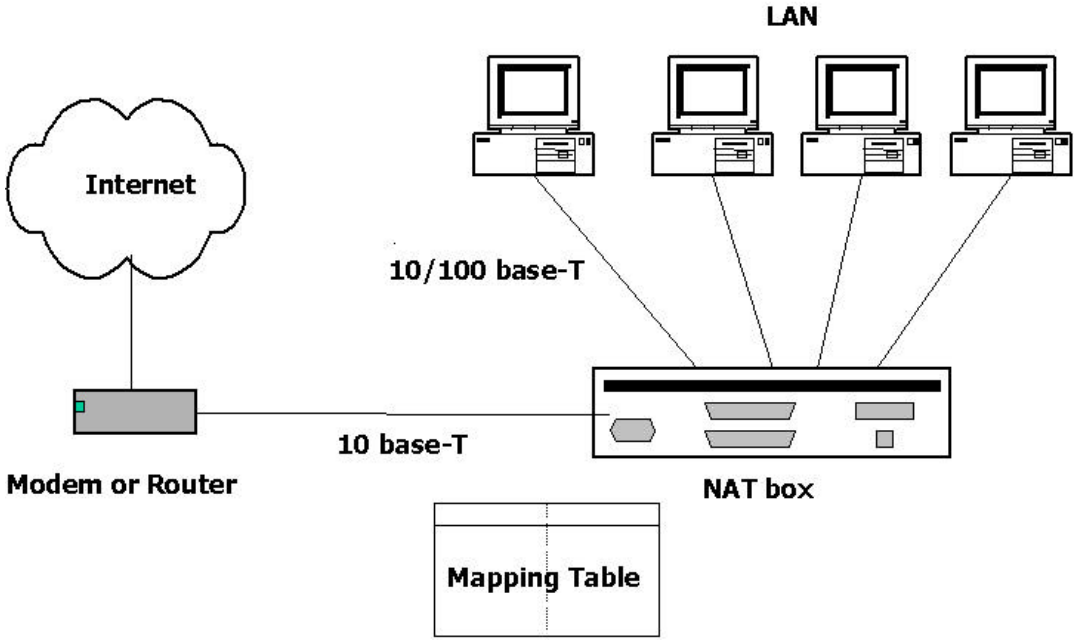
- Need to optimize each components to make specified networking applications

Design – cont'd

- Linux System Components : Run-Time Library and File System
- Run-Time Library
  - No library
    - Use kernel thread and functions only
  - glibc, sglibc, libc5, dietlibc, newlib
    - glibc is too big but provides highest application portability
- File System
  - No file system
  - NFS
  - Ramdisk as a physical storage
    - ext2, RAMFS, CRAMFS
  - Flash as a physical storage
    - Compressed Flash File System, Journaling Flash File System
- Library and File System are important at a view of size

Implementation

- Target - Internet Sharing Device
  - Uses Network Address Translation (NAT) technology



Implementation – cont'd

- Hardware
  - MPC850DE with dual ethernet controller
- Software
  - Kernel: Ethernet, PPPoE, TCP/IP, NAT, Flash driver
  - Applications: DHCP client and server, Web, telnet, and ftp servers, System utilities
- Library
  - Uses slim-sized glibc for highest application portability
- File System
  - Uses ext2 on ramdisk for speed

Conclusions

- We developed embedded Linux system which is suitable for embedded networking devices
  - Extracted common design information from several embedded OSs and used it to construct our requirements
  - Showed that embedded Linux system satisfied the requirements
  - Designed Linux kernel and system components to suit embedded purpose
  - Implemented an custom-built Internet sharing device with designed system
- Resulting system was fast and reliable which satisfied our expectation of commercially usable system
- We concluded that embedded Linux system is quite suitable and attractive operating system for networking devices

Future Work

- Future Work
  - Performance and reliability comparison with other embedded OSs with same hardware and software functions
  - Evaluate it with other embedded Linux Distributions
  - Balance resource requirements with application speed for system optimization
  - Investigate several real-time facilities on Linux
  - Port to other processors and improve the design