Embedded Systems

Ch 3A
Linux Development Environment

Byung Kook Kim
Dept of EECS
Korea Advanced Institute of Science and Technology

Overview

- 1. Embedded Linux
- 2. Cross-Development System
- 3. Setting Linux Development Environment
- 4. 개발 시스템 환경 구축
- 5. Linux Basics

1. Embedded Linux

- Software for embedded target
  - Simple systems (Ex. Automatic electric power meter)
    - Application software, modem communication software
  - More complex systems (Ex. PDA)
    - Application/service software
      - **개인정보관리**, game, **전자상거래**, **원격가전제어**, **무선전화**, Web surfing, chatting, etc.
    - Hardware dependent platform driving software
      - Software driving internal and external hardware
      - Device driver for LCD, keypad, touch panel, network, etc.
    - Network software
      - xDSL, cable modem, Ethernet, Bluetooth, Wifi, CDMA, etc.
    - Fundamental software
      - Operating system, DBMS, GUI, MMI, Web server

Embedded Linux (II)

- Operating system for embedded systems
  - Why OS?
    - Complex programs and diverse services (network and devices)
    - Fast development time and expandability
    - OS: **컴퓨터에 연결된** hardware alc software **자원을 효율적으로 관리하는 프로그램**.
    - ~60% of embedded systems utilize OS.
  - Constraints
    - Should fit within system's memory
      - RAM: data
      - ROM or flash: program
  - Operating systems
    - Wind River Systems: VxWorks, Tornado
    - Palm computing: Palm OS
    - Microsoft: Windows CE
    - **-: Embedded Linux - Next OS to be applied (49%)**

Embedded Linux (III)

- Linux originated by Linus Torvalds
  - Vast application software
  - Stability of kernels
  - Success in servers and workstations using PC

- Advantages of Linux
  - Compatible with Unix
  - Open source, free
  - Stable (than Windows)
  - Improved hardware utilization
  - Powerful networking and Internet support
  - Vast application programs
  - Multi-user, multi-tasking
  - Supports POSIX (Portable Operating System Interface for Computer Environment).

Embedded Linux (IV)

- Disadvantages of Linux
  - Standardization
  - Software reliability
  - *Much development effort*

- **Embedded Linux**
  - ***Scaled-down Linux for embedded processors to fit into ROM or flash.***
  - **낮은 성능의 프로세서와 적은 크기의 메모리를 가진 내장형 시스템용으로 개발된 리눅스.**

  - With or without memory management software (or virtual memory)
  - Ported processors
    - 32bit: Intel x86, Motorola Power PC, ARM9, MIPS, etc.
    - 64-bit: IA-64
    - W/O MMU: ARM7, Motorola 68K, Intel i960, AXIS, etc.

2. Cross-Development System

- Embedded system software
  - Embedded system**을 개발하는데 필요한 모든** software
    - User: Software running on the embedded system
    - Developer: Software for cross development environment

- Software development tools
  - Editor: Edit source files
  - Compiler: Translates into object files
  - Linker: Links object files and libraries
  - Debugger: Step-by-step execution and status check

Cross-Development System (II)

- Stand-alone system
  - PC and Workstation
  - Self-contained for general purposes
    - Hardware: CPU. Memory, general-purpose user interface, Disk,
    - Software: Operating system, application program, editor, compiler, linker
  - Native compiler
    - **PC에서 동작 되는 프로그램은 PC상에서 동작되는 컴파일러를 이용하여 개발하고, 동일한 시스템내에서 프로그램의 수행이 가능하다**
- Embedded system
  - Self-contained for a specific purpose
    - Limited hardware: CPU, memory, specific I/O, Flash or ROM
    - Software: Embedded OS, application program
  - Cross compiler
    - **개발 호스트를 구축하여 개발 호스트에서 컴파일하여 작성된 프로그램을 장비에 다운로드라는 작업을 통하여 실행 프로그램을 써 넣고 이를 수행하게 된다.**

Cross-Development System (III)

- Development stages for embedded system
  - *In the host*
    - Hardware**에 독립적인** software **설계 및 개발**
      - Host**에서** compile, run, and debug
    - Embedded processor**에서 실행 가능한** code**로** cross compile
    - Download the executable code to the embedded target
  - *In the target (with the host)*
    - Run and debug using debugging tools
    - Transfer the verified program to ROM or flash in the embedded target.

3. Setting Linux Development Environment

- Partitioning Disk
  - Tools
    - Windows 98, ME: fips, fdisk
    - Windows 2000, XP: Partition Magic (Commercial software)
  - Partition
    - Windows requires at least one disk partition (C:)
      - Add one more partition for user space (D:)
    - Linux requires at least two disk partitions (/ and swap)
      - Add one more partition for user space (/home)
    - Solution
      - Primary partition: C: for Windows
      - Secondary partition: Up to 4 logical partitions
        - 1st partition: D: for windows
        - 2nd partition: / for Linux (1 GB or more)
        - 3rd partition: Swap for Linux (2x memory size)
        - 4th partition: /home for Linux (2 GB or more).

Setting Linux Development Environment (II)

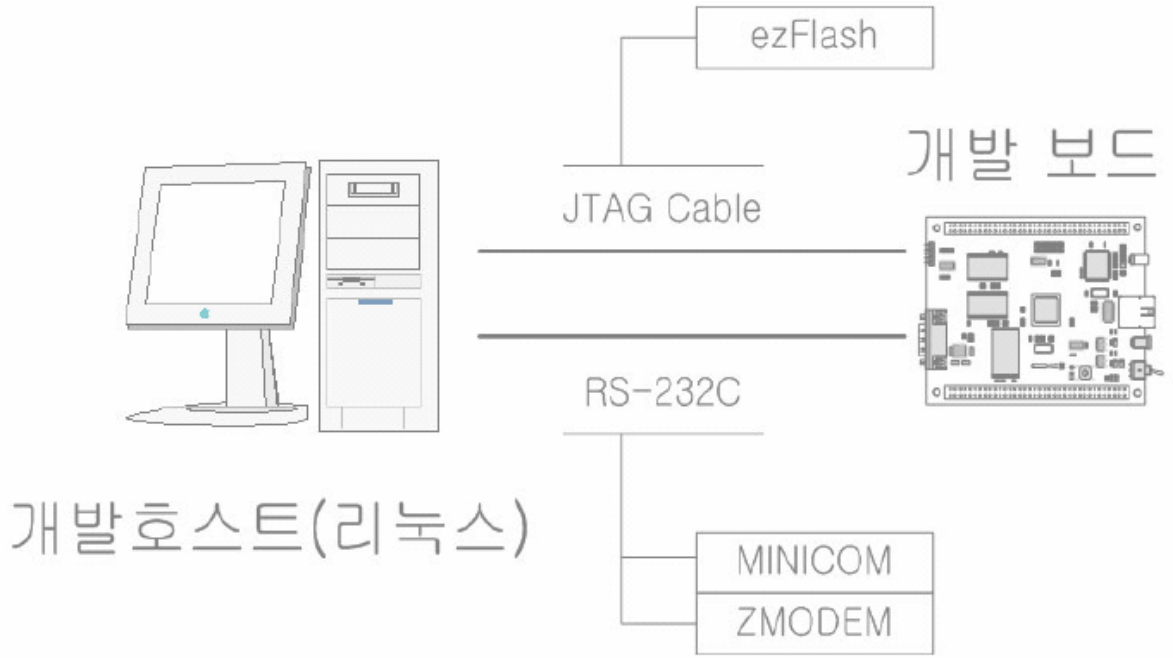
- Installing Linux
  - Suggested Linux: Redhat Linux 9.0
    - http://www.redhat.com
  - Installation menu
    - Install mode: Test, graphic, expert, rescue
    - Installation type: Workstation, server, Laptop, Custom
    - Drive partition: Disk druid, fdisk
    - Boot loader: LILO boot loader (Linux Loader)
      - Selectable boot for Windows or Linux
    - Network addresses
      - IP address, Netmask, network, broadcast, hostname, gateway, primary DNS, secondary DNS, tertiary DNS
    - Time zone: Seoul
    - Root and user ids and passwords
    - Selection of packages
    - Selection of video cards
    - Boot diskette
    - X windows setting

Setting Linux Development Environment (III)

- Install new kernel (if necessary)
  - Select modules for I/O devices
    - # make menuconfig
    - # make xconfig
- Install cross development software
  - Assembler
  - Compiler
  - Linker
- Set network environments
  - Nfs (Network File System)
  - Tftp (Tiny File Transfer Protocol)

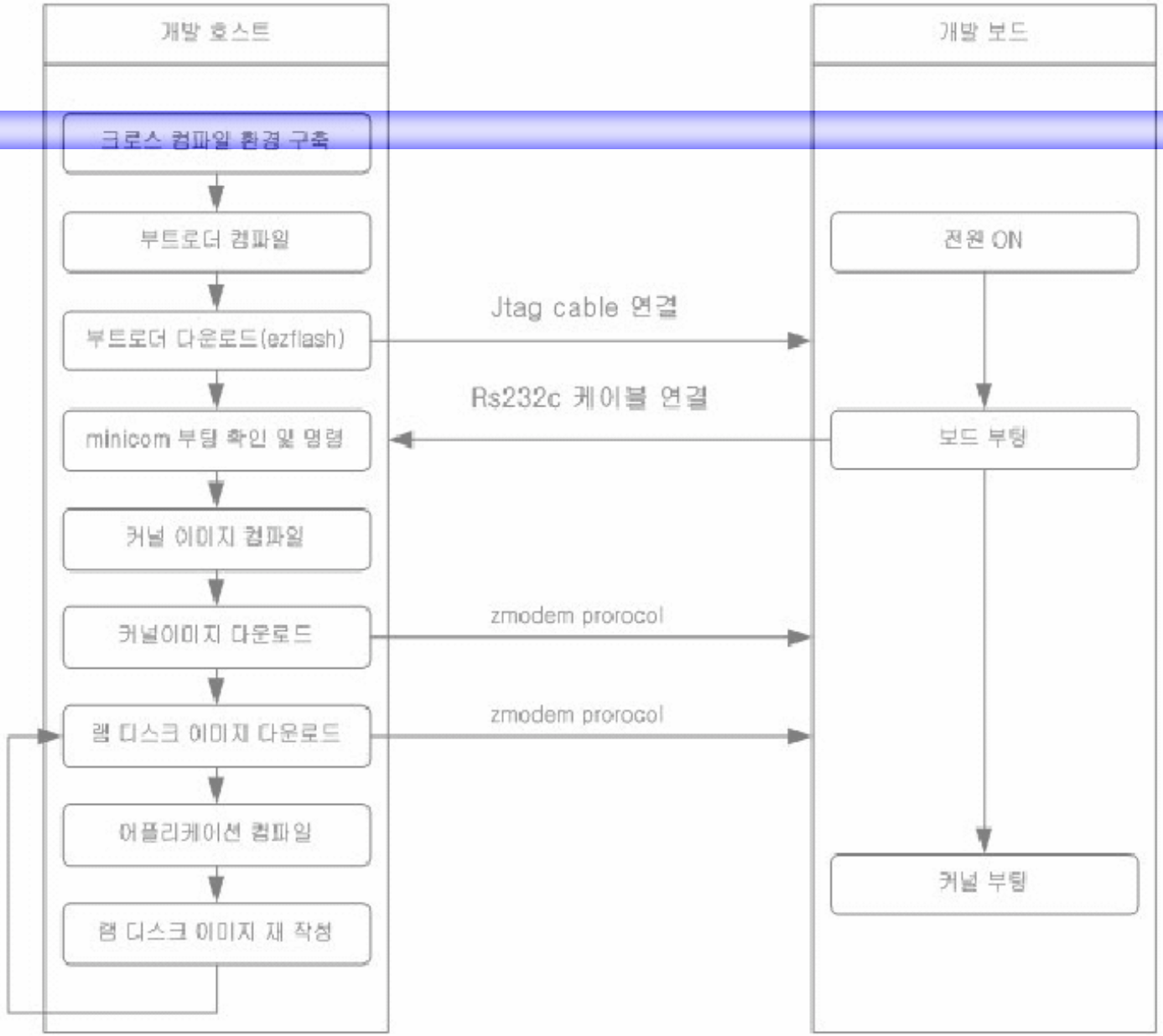
4. 개발시스템 환경 구축

- 1. 최소 개발 시스템
  - BOOT image load: JTAG with EzFlash
    - Very slow
  - Kernel and Ramdisk load: Serial with zmodem
    - Moderate



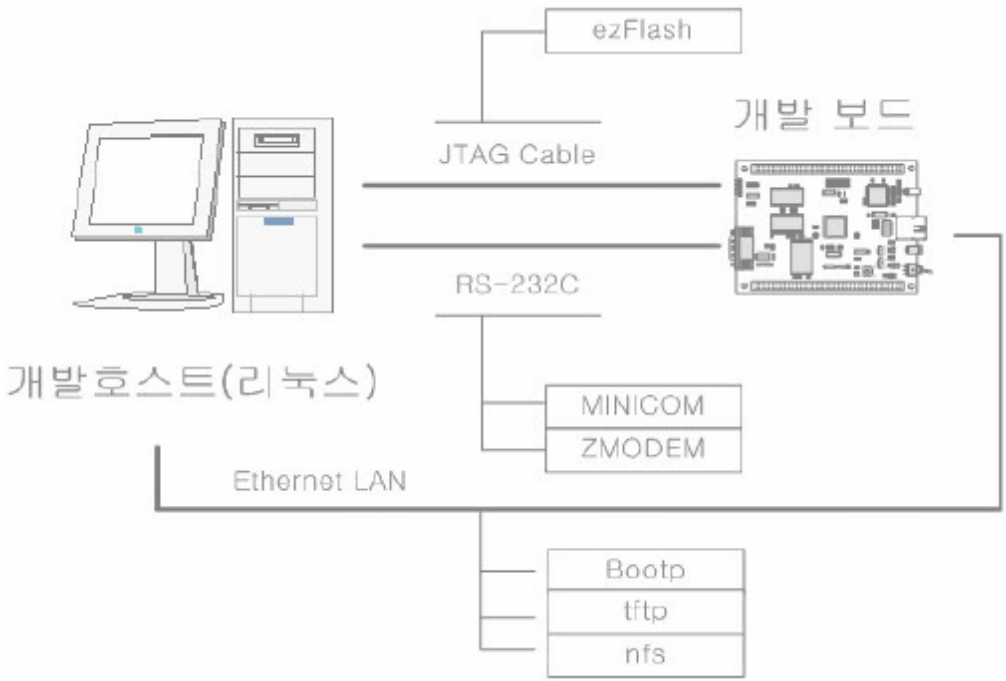
개발시스템 환경 구축 (II)

- Development flow



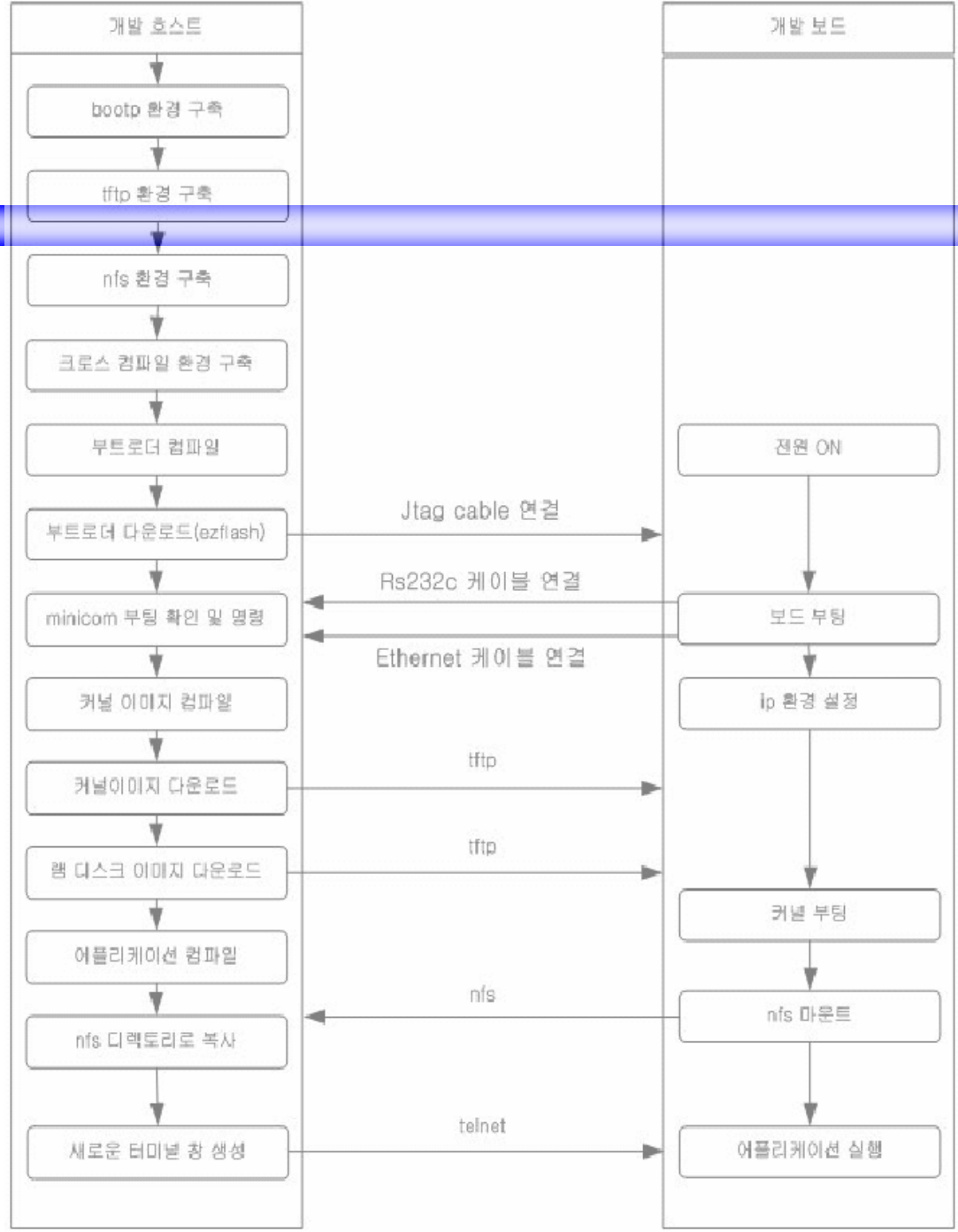
개발시스템 환경 구축 (III)

- 2. **일반적 개발 시스템**
  - Bootp **및** tftp**를 이용한** application program download
    - Bootp: **개발** board**의** IP **획득**
    - Tftp: Tiny file transfer protocol
  - Nfs: Network File System
    - Embedded system**을 위한** file system



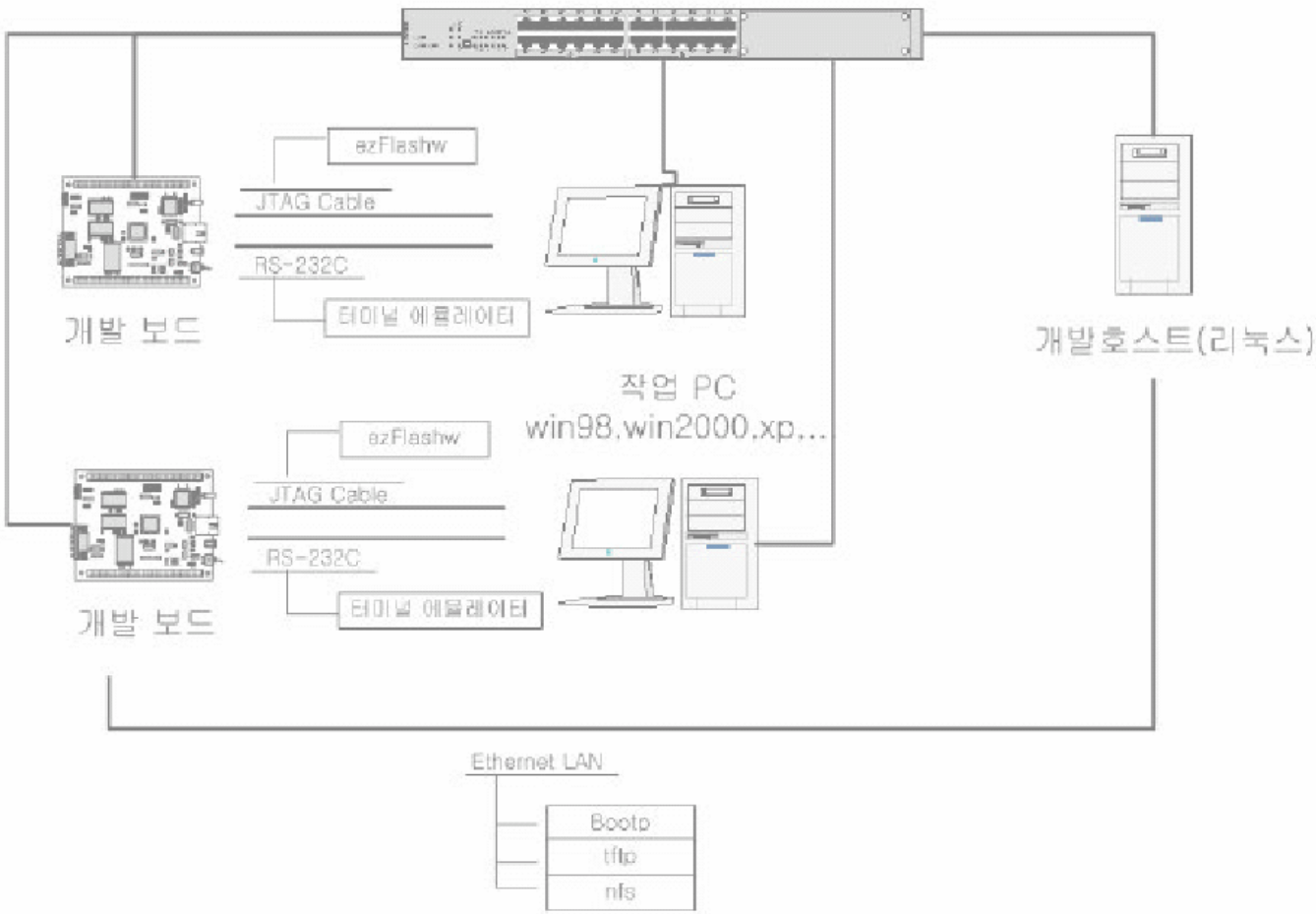
개발시스템 환경 구축 (IV)

- Development flow



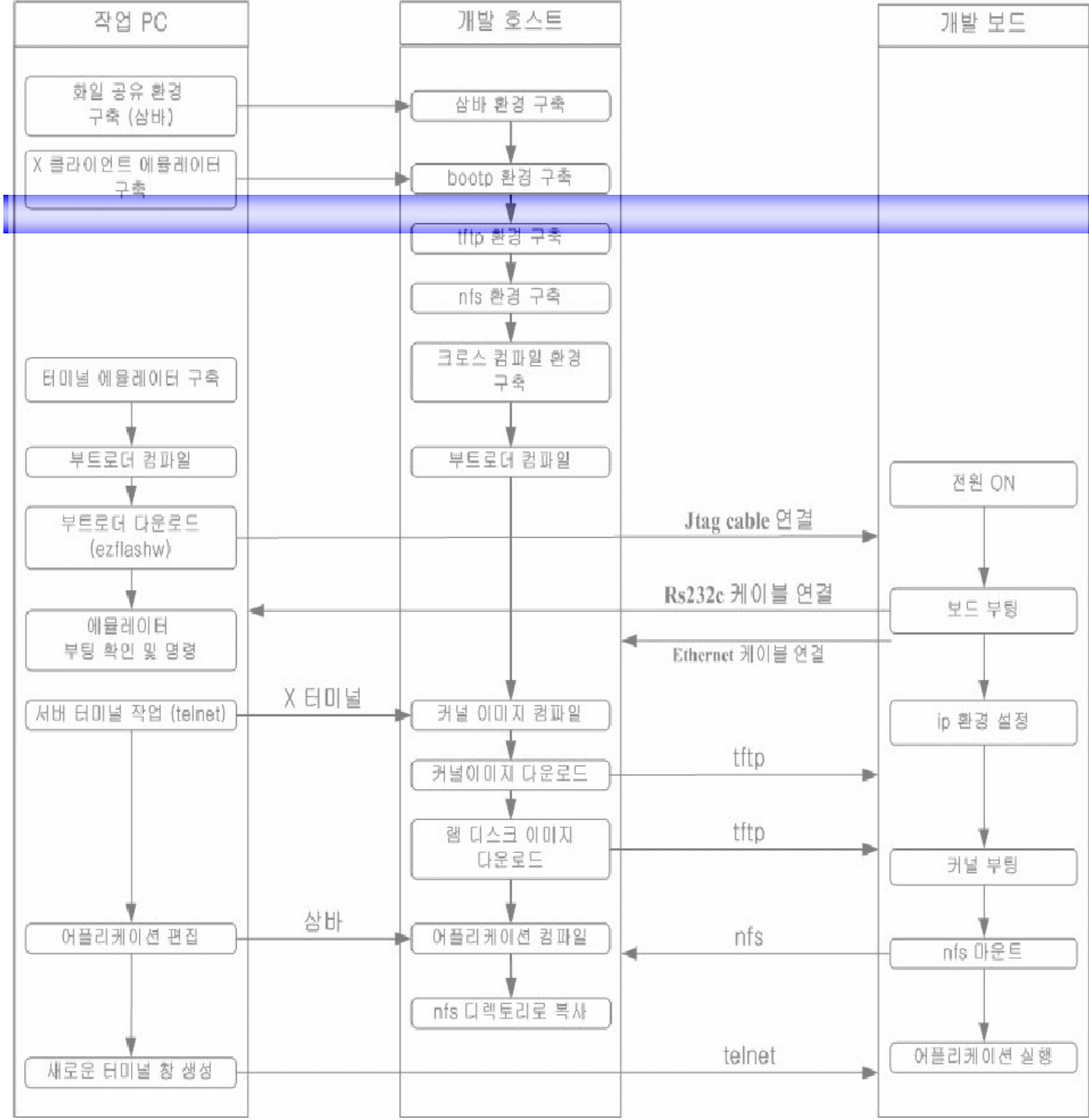
개발시스템 환경 구축 (V)

- 3. **대규모 개발 시스템**



개발시스템 환경 구축 (VI)

- Development flow



5. Linux Basics

- Getting Started
  - # login: *username* or *root*
  - # password: *user_password* or *root_password*
  - # logout
  - # shutdown –h now ; Shutdown the computer
- Basic commands
  - # date ; Display date and time
    - Wed Sep 1 12:12:29 EDT 2004
  - # who ; List users currently logged in
  - # man command ; Display manual of the command
  - # pwd ; Print the complete pathname of the current directory
  - # cd /usr/src/linux ; Change directory to /usr/src/linux

Linux Basics (II)

- File manipulation
  - # ls [-la] ; List files in the current directory
  - # cat filename ; Prints the file with filename
  - # cp source_file dest_file ; Copy source_file to dest_file
    - # cp file /dev/ttyS0 ; Copy file to COM1
  - # rm junk_file ; Remove junk_file
  - # mv old_file to new_file ; Rename the old_file to new_file
- Manipulating directories
  - # mkdir new_dir ; Make a new_dir directory
  - # rmdir old_dir ; Delete the old_dir directory
  - # mv old_dir new_dir ; Rename old_dir directory to new_dir
  - # cd new_dir ; Change directory to new_dir
    - # cd .. ; Change to upper directory
    - # cd / ; Chenage to root directory

Linux Basics (III)

- System inquiries
  - # ps ; List active processes with process_id
  - # kill -9 process_id ; Kill the process with process_id
  - # du ; Disk usage of the current directory
  - # df ; Display file system usage
  - # su ; Become the superuser (root)
    - # password: *root_password*
  - # exit ; Become a normal user

Linux Basics (IV)

- Editing files with vi
  - # vi file.c ; Visual edit file.c
  - # Ctrl-F, Ctrl-B : Move forward/backward a full screen
  - # space, backspace, return ; Move cursor right/left/next_line
  - # i... esc ; Insert characters before cursor (until escape)
  - # a... esc ; Insert characters after cursor (until escape)
  - # o... esc ; Insert line by line after the current line
  - # O... esc ; Insert line by line before the current line
  - # x ; Delete the current character
  - # dw ; Delete the current word
  - # dd ; Delete the current line
  - # r file ; Read the file
  - # s/old/new/g ; Substitute old to new globally
  - # :q ; Quit without saving
  - # :wq ; Quit after saving

Linux Basics (V)

```
#include <stdio.h>
void main()
{
        printf("Hello, Embedded system!₩n");
}
```

- Compile and run
  - # mkdir /embedded/test
  - # cd /embedded/test
  - # vi hello.c
  - # gcc –o hello hello.c
    ; Cross-compile and link the program to produce hello.
  - # ./hello
    ; Run hello
  - Hello, Embedded board!
    ; Output: print a string on the console

Linux Basics (VI)

- Make command
  - # vi Makefile

```
main.o average.o: defs.h
average: main.o average.o
        gcc –o average main.o average.o –lm
```

  - # vi defs.h
  - # vi main.c
  - # vi average.c
  - # make average.o ; Compile average.c to average.o
  - # make average ; Compile main.c to main.o
    Link main.o, average.o, and lib into average
  - # ./average ; Run average