
7부. 임베디드 리눅스의 활용과 미니 프로젝트

23장. 임베디드 리눅스의 활용

24장. 미니 프로젝트

교육 목표

- 임베디드 리눅스의 다양한 활용 방법을 알아본다
- 임베디드 리눅스 기반의 미니 프로젝트를 실행한다

목 차

□ 23장. 임베디드 리눅스의 활용

01. Nano-X 윈도우 시스템

02. Qt/Embedded

03. Boa 웹서버

04. USB 장치 사용

24장. 미니 프로젝트

Nano-X 윈도우 시스템

□ Nano-X

- ❖ Microwindows로 잘 알려져 있다
- ❖ 임베디드 리눅스 플랫폼에서 윈도우 환경을 제공하기 위한 오픈 소스 프로젝트 (<http://www.microwindows.org>)

□ Nano-X는 계층화 설계 구조

- ❖ 저 수준(low-level) 계층
 - 스크린, 마우스/터치, 키보드 디바이스 드라이버를 이용한 실제적인 하드웨어 제어를 지원
- ❖ 중간 수준(mid-level) 계층
 - 라인 그리기, 영역 채우기, 폴리곤(polygons), 컬러 등 그래픽 엔진을 구현
- ❖ 상위(upper-level) 계층
 - 어플리케이션 프로그래머가 쉽게 사용할 수 있도록 다양한 API를 지원

Nano-X의 디바이스 접근

- 중간 수준(mid-level) 계층 루틴에서 하드웨어와 관련된 저 수준 계층의 디바이스 드라이버를 직접 호출
- 디바이스 접근 예
 - ❖ 임베디드 리눅스의 스크린 드라이버
 - 프레임버퍼(Framebuffer)를 직접 호출하여 사용
 - /dev/fb0를 열고 mmap() 시스템 호출을 사용하여 메모리에 있는 디스플레이 메모리를 매핑하여 사용
 - ❖ 마우스/터치 디바이스 드라이버
 - 시리얼 마우스 또는 터치 패널 디바이스를 직접 호출하여 사용하도록 구현

Nano-X 빌드와 실행 (페이지 702 참조)

1. Nano-X 소스 설치

- ❖ 최근 소스를 다운로드 받아서 설치 한다.

2. Nano-X 설정

- ❖ 플랫폼, 스크린 픽셀 타입, 폰트, 마우스/키보드 설정

3. Nano-X 컴파일

- ❖ 모든 라이브러리와 샘플 어플리케이션을 빌드 한다.

4. Nano-X 실행

- ❖ 샘플 실행 파일을 실행 한다.

실습 23-1 : Nano-X 사용

□ Nano-X 소스를 설치하고 빌드 한 후 실행한다.

목 차

□ 23장. 임베디드 리눅스의 활용

01. Nano-X 윈도우 시스템

02. Qt/Embedded

03. Boa 웹서버

04. USB 장치 사용

24장. 미니 프로젝트

Qt/Embedded

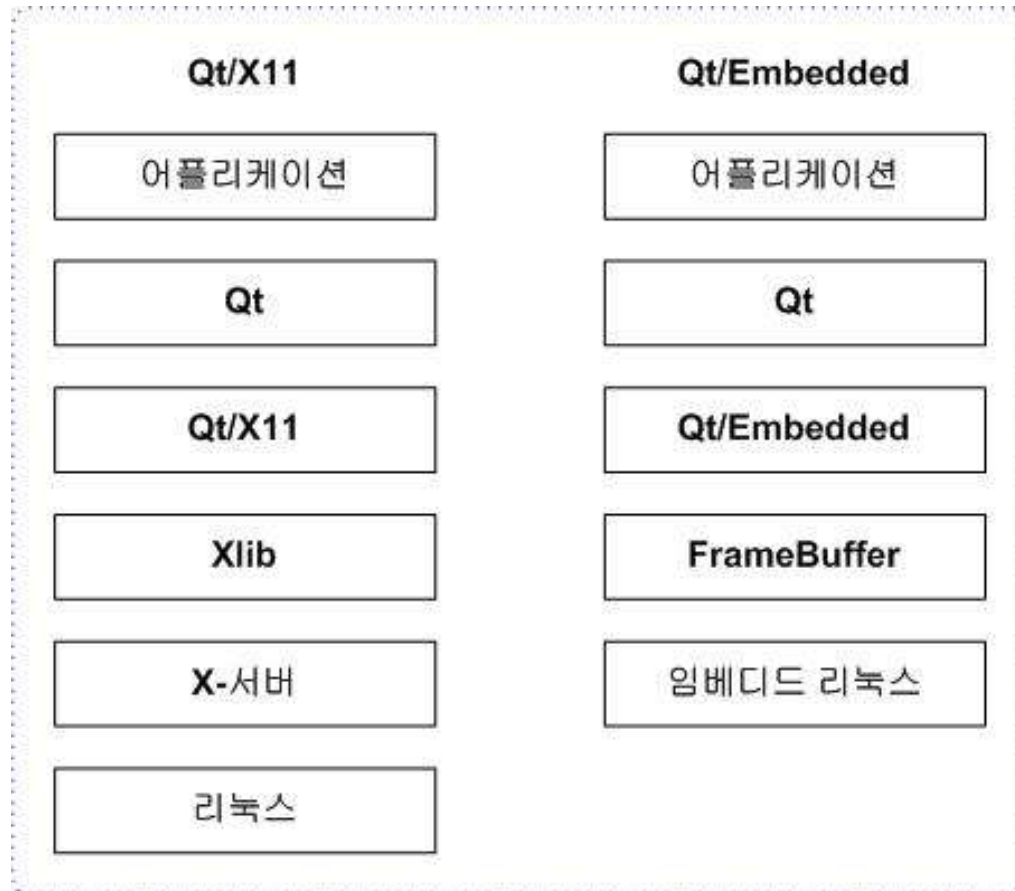
□ Qt

- ❖ Trolltech사에서 만든 교차 플랫폼(cross platform) GUI 툴 키트
- ❖ 공개 버전과 상용 버전 2개의 라이선스(QPL)로 관리
- ❖ PC 기반의 KDE에서 사용 되고 있는 것을 비롯하여 PDA 등 다양한 임베디드 플랫폼에서 가장 많이 사용

□ Qt가 지원되는 플랫폼

- ❖ Qt/X11
 - X11 기반의 PC에서 동작
- ❖ Qt/Embedded
 - 임베디드 시스템에서 동작 하도록 보다 가볍고 빠르게 재 구성하여 직접 FrameBffer를 제어하는 방법을 사용

Qt/X11 과 Qt/Embedded



Qt 소스 설치

□ 공중의 **FTP** 사이트 또는 부록 **CD**의 소스 사용

파일명	설명
qt-x11-free-3.3.3.tar.gz	압축된 Qt/X11 소스 파일
qt-embedded-free-3.3.3.tar.gz	압축된 Qt/Embedded 소스 파일

```
# tar zxvf qt-x11-free-3.3.3.tar.gz  
# tar zxvf qt-embedded-free-3.3.3.tar.gz
```

Qt/X11 컴파일

□ Qt/X11

- ❖ X 윈도우 기반 PC에서 Qt 환경을 사용하기 위해서 설치
- ❖ 임베디드 시스템 용으로 사용하는 경우 설치하지 않아도 되지만 향후 Qt 기반의 어플리케이션을 작성할 때 가상 플랫폼 QVFB(Qt Virtual Fram Buffer)를 사용하거나 임베디드 시스템에 Qt를 탑재하기에 앞서 PC 기반에서 비교적 쉽게 환경 시험 가능

□ Qt/X11 컴파일 환경 설정

- ❖ 계정의 .bash_profile 파일 사용

```
# vi ~/.bash_profile
```

```
# source ~/.bash_profile
```

.bash_profile 내용 수정

```
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs
PATH=$PATH:$HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME="root"

# For Qt/X11
QTDIR=/root/qt-x11-free-3.3.3
PATH=$QTDIR/bin:$PATH
LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH
export QTDIR LD_LIBRARY_PATH
export USERNAME BASH_ENV PATH
```

Qt/X11 빌드

```
# cd qt-x11-free-3.3.3
```

```
# ./configure
```

```
This is the Qt/X11 Free Edition.
```

```
----- 이 하 생 략 -----
```

```
# gmake
```

```
----- 이 하 생 략 -----
```

```
The Qt library is now built in ./lib
```

```
The Qt examples are built in the directories in ./examples
```

```
The Qt tutorials are built in the directories in ./tutorial
```

```
Enjoy! - the Trolltech team
```

Qt/Embedded 컴파일 및 실행

□ 컴파일 환경 설정

- ❖ .bash_profile 파일 수정
- ❖ 새로운 환경변수 적용
- ❖ Qt/X11의 사용자 인터페이스 컴파일러(uic) 복사

```
# vi ~/.bash_profile

# source ~/.bash_profile

# cp /root/qt-x11-free-3.3.3/bin/uic /root/qt-embedded-free-3.3.3/bin
```

.bash_profile 내용 수정

```
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs
PATH=$PATH:$HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME="root"

# For Qt/Embedded for ARM
QTDIR=/root/qt-x11-free-3.3.3
QMAKESPEC=/root/qt-embedded-free-3.3.3/mkspecs/qws/linux-arm-g++
PATH=$QTDIR/bin:$PATH
LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH
export QTDIR LD_LIBRARY_PATH
```


Qt/Embedded 빌드

```
# cd qt-embedded-free-3.3.3
# ./configure --prefix=/usr -platform qws/linux-x86-g++ -xplatform qws/linux-
arm-g++ -release -thread -depths 32 -no-cups -no-ipv6 -no-nis -qt-gif -qt-
libjpeg -qt-imgfmt-jpeg -qt-mouse-linuxtp
This is the Qt/Embedded Free Edition.
```

----- 이 하 생 략 -----

```
# gmake
```

----- 이 하 생 략 -----

The Qt library is now built in ./lib
The Qt examples are built in the directories in ./examples
The Qt tutorials are built in the directories in ./tutorial
Enjoy! - the Trolltech team

Qt/Embedded 탑재

□ 공유 라이브러리와 폰트 그리고 애플리케이션

- ❖ 라이브러리와 애플리케이션은 불필요한 심볼을 제거하기 위해서 `arm-linux-strip`을 실행 한 후에 복사
- ❖ 공유 라이브러리는 생성된 실제 라이브러리만 복사하고 링크를 설정하여 사용 (타겟에서 링크)

```
# mkdir /tftpboot/qt
# mkdir /tftpboot/qt/lib
# cd qt-embedded-free-3.3.3/lib
# arm-linux-strip libqte-mt-so.3.3.3
# cp libqte-mt-so.3.3.3 /tftpboot/qt/lib
# cp -rf fonts /tftpboot/qt/lib/
# cd ../examples/hello
# arm-linux-strip hello
# cp hello /tftpboot/qt
```

Qt/Embedded 탑재 - 타겟

```
[root]$ mount -o nolock 192.168.1.10:/tftpboot /mnt
[root]$ cd /usr
[root]$ mkdir lib
[root]$ cd lib
[root]$ cp /mnt/qt/lib/fonts .
[root]$ cp /mnt/qt/lib/libqte-mt.so.3.3.3 .
[root]$ ln -s libqte-mt.so.3.3.3 libqte-mt.so.3.3
[root]$ ln -s libqte-mt.so.3.3.3 libqte-mt.so.3
[root]$ ln -s libqte-mt.so.3.3.3 libqte-mt.so
[root]$ cd /usr/bin
[root]$ cp /mnt/qt/hello .
```

Qt/Embedded 실행

□ 복사된 애플리케이션을 명령어 라인에서 실행

- ❖ 애플리케이션을 실행할 때 서버로 동작 시키기 위해서 `-qws` 옵션을 사용

```
[root]$ hello -qws
```

실습 23-2 : Qt/Embedded 실습

- Qt/Embedded를 설치하고 컴파일 하여 실행한다.

목 차

□ 23장. 임베디드 리눅스의 활용

01. Nano-X 윈도우 시스템

02. Qt/Embedded

03. Boa 웹서버

04. USB 장치 사용

24장. 미니 프로젝트

임베디드 시스템의 **WEB** 서버

□ 비교적 가볍고 빠른 **WEB** 서버를 채택

❖ PC 기반 **WEB** 서버와 다른 점

- 제한된 메모리나 저장 공간
- PC에 비해 느린 성능

□ 사용 가능한 임베디드 **WEB** 서버

WEB 서버	설명	관련 사이트
thttpd	Tiny/turbo/throttling HTTP 서버	http://www.acme.com/software/thttpd
boa	Single tasking HTTP 서버	http://www.boa.org
goahead	GoAhead 임베디드 WEB 서버	http://www.goahead.com

Boa 웹 서버 빌드

1. Boa 소스 얻기

- ❖ 홈페이지 <http://www.boa.org>

2. Boa 소스 설치

- ❖ TAR 명령 사용

```
# tar zxvf boa-0.94.13.tar.gz
```

3. Boa 컴파일 환경 설정과 컴파일

- ❖ 컴파일 환경 설정

```
# cd boa-0.94.13/src  
# ./configure
```

- ❖ 컴파일러 지정

- ❖ 컴파일

```
# make
```

```
-- 생략 --
```

```
CC = arm-linux-gcc
```

```
CPP = arm-linux-gcc -E
```

```
CFLAGS += -march=armv4 -mtune=arm9tdmi
```

```
-- 이하 생략 --
```


Boa 동작 환경 설정

□ Boa 동작에 필요한 필수 항목

❖ 'boa' 실행 파일, 동작환경 설정 파일(boa.conf) 및 HTML문서

□ 동작 환경 설정 파일 (NFS 마운트 사용)

설정 변수	지정 값	내용
ErrorLog	/var/log/boa/error_log	에러 로그(log)를 저장하는 파일 /var/log/boa 폴더는 생성해 준다
AccessLog	/var/log/boa/access_log	엑세스 로그를 저장하는 파일
ServerName	192.168.1.11	서버의 이름, 타겟의 IP 지정
DocumentRoot	/mnt/boa	Boa 웹서버의 HTML 루트 지정
DirectoryMaker	/mnt/boa/lib/boa_indexer	디렉토리 리스트 생성 프로그램
MimeTypes	/mnt/boa/conf/mime.types	사용할 마임(mime) 타입 지정
AddType	application/x-httpd-cgi cgi	CGI 확장자의 기본 마임타입 지정
Group	nobody	Group를 nobody로 지정

Boa 실행 준비

□ NFS 마운트 되는 호스트의 tftpboot 에 파일 및 디렉토리 준비

```
# mkdir /tftpboot/boa
# mkdir /tftpboot/boa/lib
# mkdir /tftpboot/boa/conf
# cd boa-0.94.13
# cp boa.conf /tftpboot/boa/conf
# cp /etc/mime.types /tftpboot/boa/conf
# cd src
# arm-linux-strip boa
# arm-linux-strip boa_indexer
# cp boa /tftpboot/boa
# cp boa_indexer /tftpboot/boa/lib
```

□ HTML 문서 준비

```
# cd /mnt/cdrom/Software/ARM_Linux/Resource/Boa/html
# cp -rf * /tftpboot/boa/.
```

Boa WEB 서버 실행

□ 폴더 속성 변경

```
# cd /tftpboot  
# chmod -R 777 boa  
# chown -R nobody boa  
# chgrp -R nobody boa
```

□ WEB 서버 실행 (타깃에서 실행)

```
[root]$ mkdir /var/log/boa  
[root]$ mount -o nolock 192.168.1.10:/tftpboot /mnt  
[root]$ cd /mnt/boa  
[root]$ ./boa -c /mnt/boa/conf
```

□ WEB 서버 접속 (호스트의 WEB 브라우저에서 실행)

```
http://192.168.1.11
```

WEB 접속 화면



CGI 프로그래밍

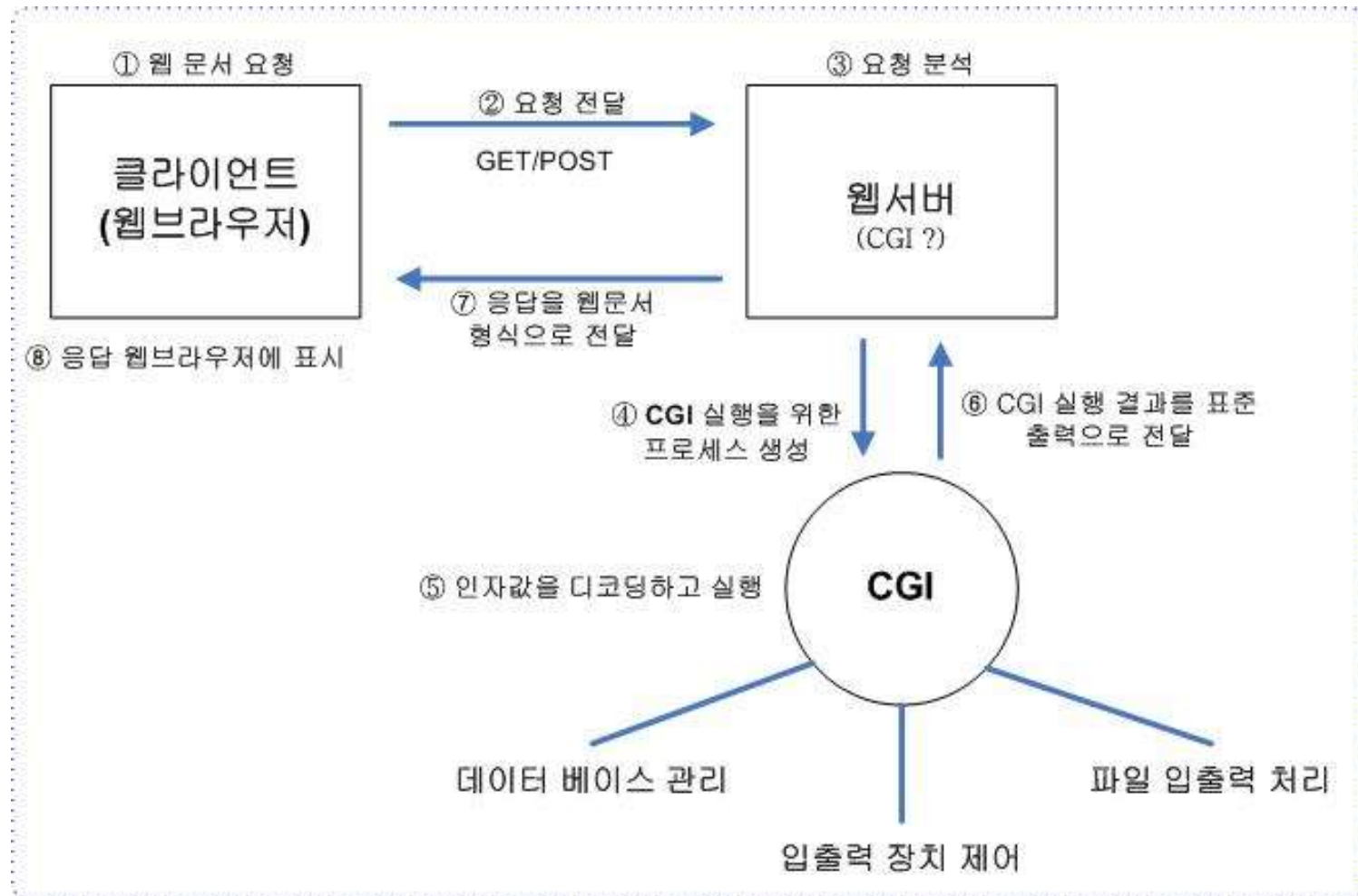
□ CGI(Common Gateway Interface)

- ❖ 웹서버의 기능을 확장하기 위해서 웹서버와 다른 프로그램 간에 정해진 인터페이스 규격
- ❖ 방명록, 파일 업로드와 같은 사용자의 자료 입력을 비롯한 클라이언트의 다양한 요구를 처리하기 위해서 사용
- ❖ 특별히 정해진 프로그램 방식이 없으며 표준 입출력 기능을 제공하여 웹서버와 인터페이스가 가능하면 무엇이든 사용 가능
 - 쉘스크립트, C/C++, Perl, Tcl, 자바 등

□ C로 작성된 간단한 CGI 프로그램의 예

```
#include <stdio.h>
int main(void)
{
    printf("Content-type: text/html\n\n");
    printf("<html>\n");
    printf("<head><title>Hello CGI Title !!!</title></head>\n");
    printf("<body>\n<center>Hello CGI Body !!!</center>\n</body> \n");
    printf("</html>");
}
```

CGI 프로그램의 동작 절차



실습 23-3 : Boa 웹서버와 CGI

- Boa 웹서버를 설치하고 실행한다
- CGI를 이용하여 문자 LCD를 제어한다

목 차

□ 23장. 임베디드 리눅스의 활용

01. Nano-X 윈도우 시스템

02. Qt/Embedded

03. Boa 웹서버

04. USB 장치 사용

24장. 미니 프로젝트

USB 장치

□ 대표적인 USB 장치 들

- ❖ USB 저장 장치, USB 카메라, USB 스캐너, USB 프린터 등

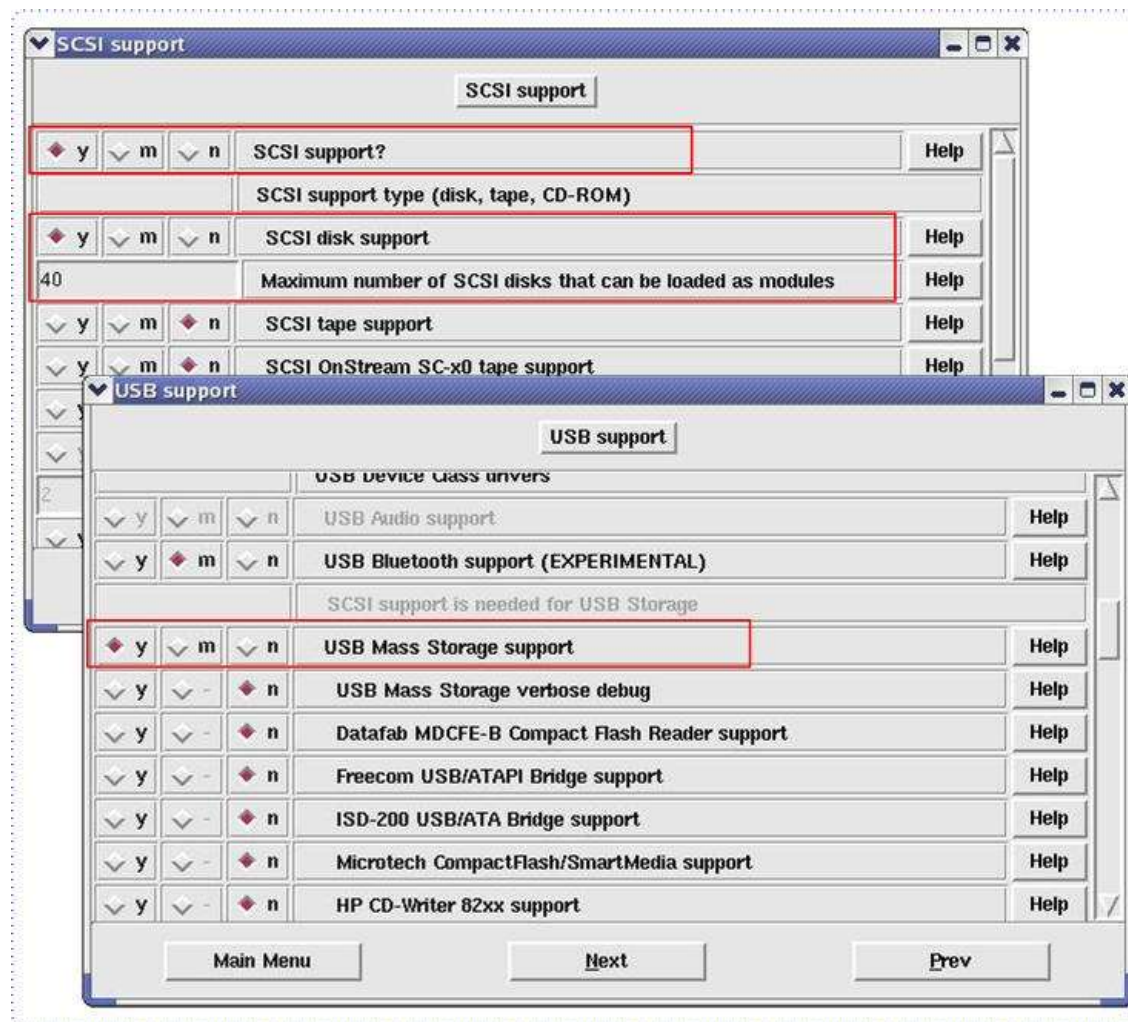
□ USB 저장 장치 사용

- ❖ USB 메모리 스틱 또는 USB 하드 디스크
- ❖ USB 저장 장치 사용을 위한 리눅스 커널의 설정 변경
 - USB 저장 장치를 지원하도록 설정
 - SCSI(Small Computer System Interface) 장치를 사용하도록 설정

□ USB 카메라 사용

- ❖ ‘Logitech QuickCam 4000 Pro’ 제품 사용

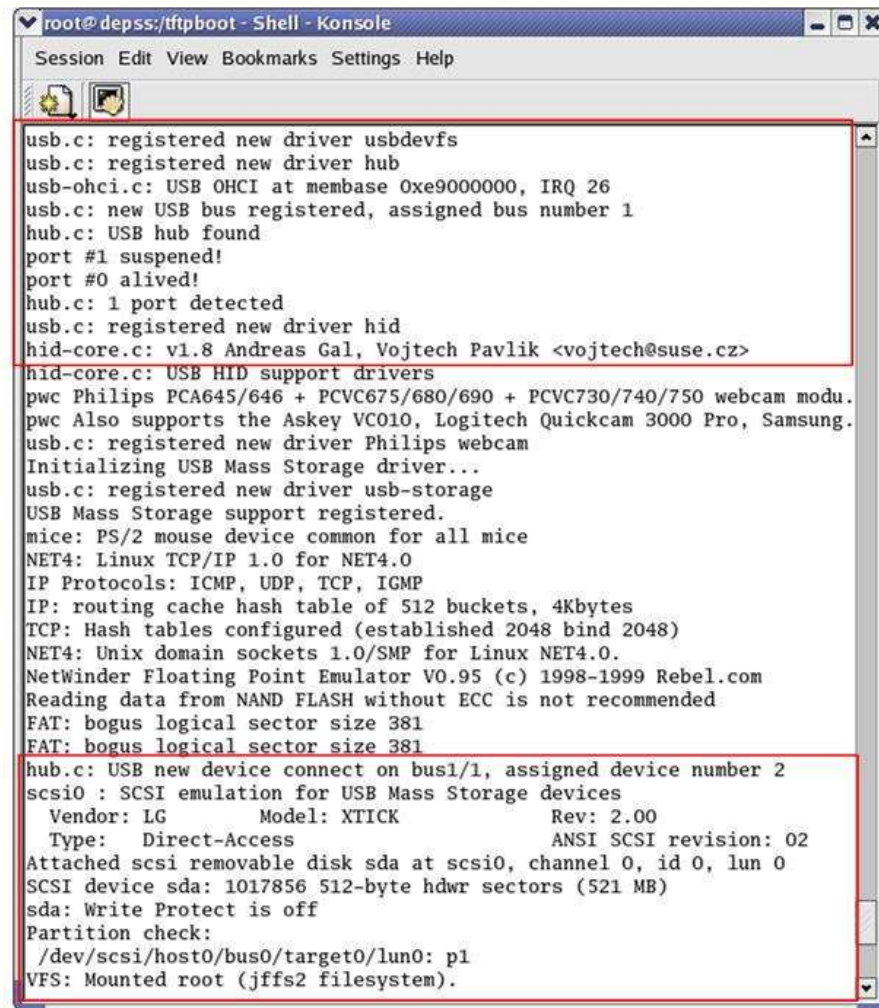
USB 저장 장치를 위한 커널 설정



USB 장치 인식

□ 커널 실행 화면USB

- ❖ 저장 장치는 커널 부팅 전에 코넥터에 삽입



```
root@depss:/tftpboot - Shell - Konsole
Session Edit View Bookmarks Settings Help

usb.c: registered new driver usbdevfs
usb.c: registered new driver hub
usb-ohci.c: USB OHCI at membase 0xe9000000, IRQ 26
usb.c: new USB bus registered, assigned bus number 1
hub.c: USB hub found
port #1 suspended!
port #0 alive!
hub.c: 1 port detected
usb.c: registered new driver hid
hid-core.c: v1.8 Andreas Gal, Vojtech Pavlik <vojtech@suse.cz>
hid-core.c: USB HID support drivers
pwc Philips PCA645/646 + PCVC675/680/690 + PCVC730/740/750 webcam modu.
pwc Also supports the Askey VC010, Logitech Quickcam 3000 Pro, Samsung.
usb.c: registered new driver Philips webcam
Initializing USB Mass Storage driver...
usb.c: registered new driver usb-storage
USB Mass Storage support registered.
mice: PS/2 mouse device common for all mice
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP, IGMP
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 2048 bind 2048)
NET4: Unix domain sockets 1.0/SMP for Linux NET4.0.
NetWinder Floating Point Emulator V0.95 (c) 1998-1999 Rebel.com
Reading data from NAND FLASH without ECC is not recommended
FAT: bogus logical sector size 381
FAT: bogus logical sector size 381
hub.c: USB new device connect on bus1/1, assigned device number 2
scsi0 : SCSI emulation for USB Mass Storage devices
Vendor: LG          Model: XTICK          Rev: 2.00
Type: Direct-Access          ANSI SCSI revision: 02
Attached scsi removable disk sda at scsi0, channel 0, id 0, lun 0
SCSI device sda: 1017856 512-byte hdwr sectors (521 MB)
sda: Write Protect is off
Partition check:
/dev/scsi/host0/bus0/target0/lun0: p1
VFS: Mounted root (jffs2 filesystem).
```

USB 저장 장치 마운트

- 마운트 명령을 실행하여 **USB** 저장 장치를 마운트
/dev/scsi/host0/bus0/target0/lun0/part1



```
root@depss:/tftpboot - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@S3C2410 root]$
[root@S3C2410 root]$ls /mnt/
[root@S3C2410 root]$
[root@S3C2410 root]$mount -t vfat /dev/scsi/host0/bus0/target0/lun0/part1 /mnt
[root@S3C2410 root]$ls /mnt/
The_Magic_of_Flight_720_WMA_audio.wmv play_osd
[root@S3C2410 root]$
```

USB 카메라 사용

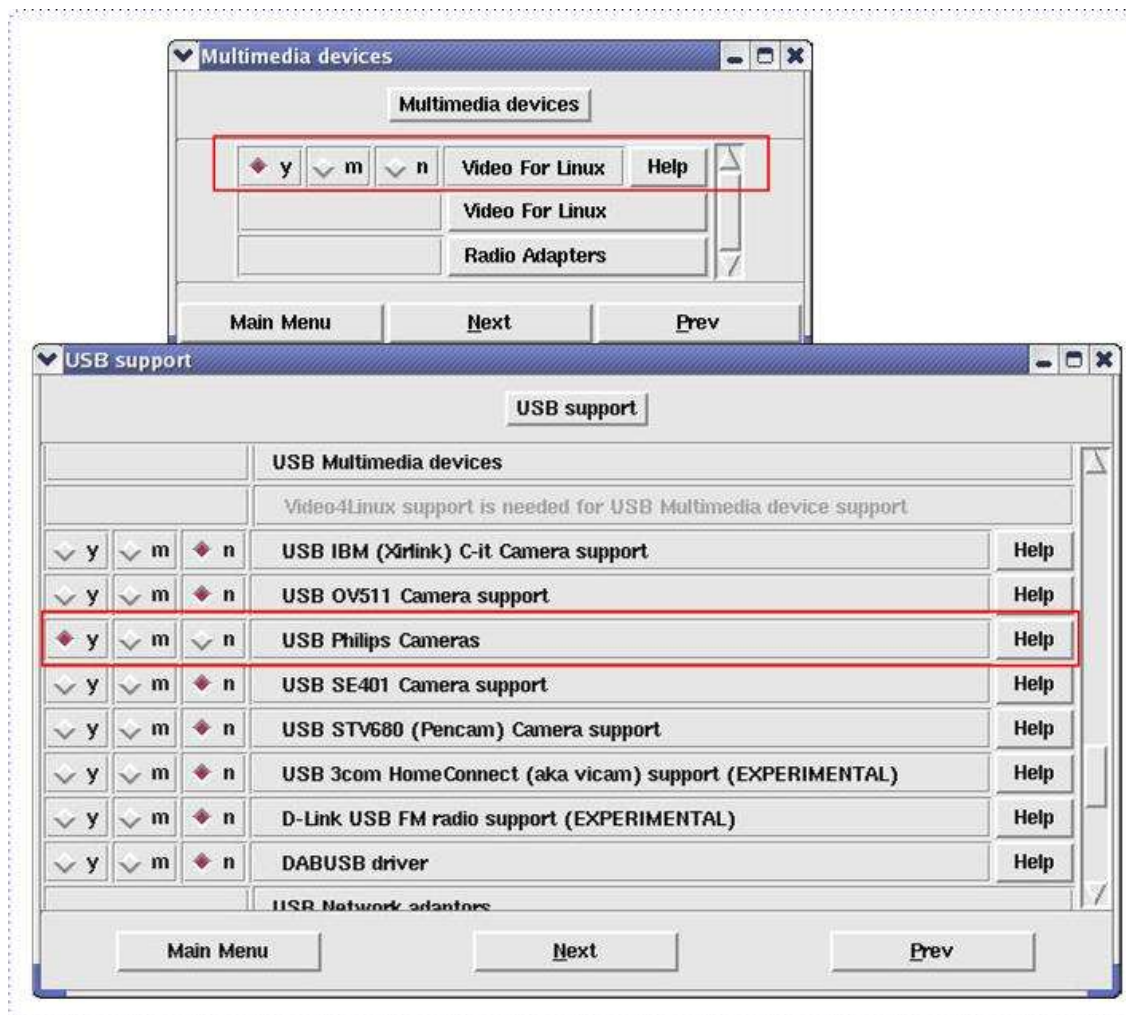
□ 리눅스 디바이스 드라이버가 제공되는 **USB** 카메라 선택

- ❖ ‘Logitech QuickCam 4000 Pro’ 제품 사용
- ❖ 필립스(Philips) 드라이버와 호환
- ❖ 필립스 웹카메라 드라이버(/drivers/usb/pwc-if.c)를 사용

□ 리눅스 멀티미디어 디바이스 지원

- ❖ USB 카메라를 지원하기 위해서 필요
- ❖ USB의 멀티미디어 디바이스에서 사용하고자 하는 장치
드라이버 선택

USB 카메라를 위한 커널 설정



USB 카메라 인식과 시험

```
root@depss:/tftpboot - Shell - Konsole
Session Edit View Bookmarks Settings Help

For further information check:
    http://www.dignsys.com/

S3C2410 login: root
Password:
[root@S3C2410 root]$
[root@S3C2410 root]$
[root@S3C2410 root]$
[root@S3C2410 root]$hub.c: USB new device connect on bus1/1, assigned device number 2
pwc Logitech QuickCam 4000 Pro detected.
pwc Registered as /dev/video0.

[root@S3C2410 root]$
[root@S3C2410 root]$ls -al /dev/video0
lr-xr-xr-x  1 root  root          10 Nov  3 02:05 /dev/video0 -> v4l/video0
[root@S3C2410 root]$
[root@S3C2410 root]$mount -o nolock 192.168.1.10:/tftpboot /mnt
[root@S3C2410 root]$
[root@S3C2410 root]$cd /mnt
[root@S3C2410 mnt]$ls cam2fb
can2fb
[root@S3C2410 mnt]$./cam2fb
pwc Video mode SIF@10 fps is only supported with the decompressor module (pwcx).
```

실습 23-4 : USB 카메라 사용 시험

□ USB 카메라를 사용하여 LCD에 화상을 출력 한다

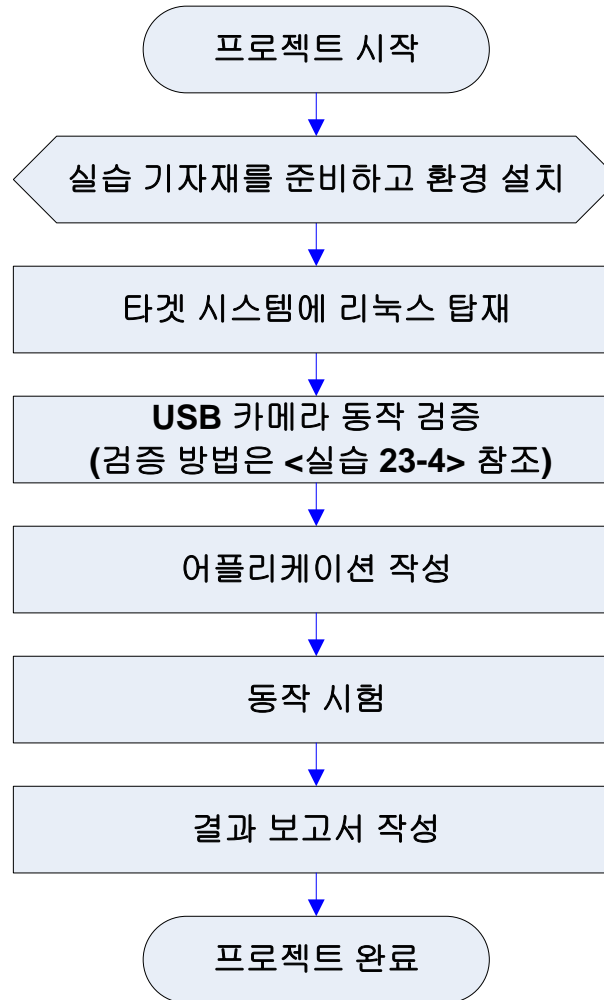
목 차

23장. 임베디드 리눅스의 활용

□ 24장. 미니 프로젝트

- 01. 카메라에 입력된 영상 LCD로 출력
- 02. 웹서버를 활용한 입출력 장치 제어
- 03. 접프-업 프로젝트

프로젝트 진행 과정



실습 기자재

하드웨어	DTK2410	DEP2410 CPU 보드
		3.5인치 TFT LCD
	USB 카메라	리눅스 드라이버가 지원 되는 것이 용이 없으면 드라이버도 같이 개발
소프트웨어	임베디드 리눅스	리눅스 커널
		루트 파일시스템
개발용 PC	리눅스 PC	PC의 리눅스는 어떤 버전의 어떤 리눅스가 되더라도 임베디드 시스템에서 사용되는 리눅스와는 무관 이 책에서는 RedHat 9.0을 기준으로 설명되었다.

프로젝트에서 사용되는 영상 형식

□ USB로 입력되는 영상 YUV 형식

- ❖ YUV 형식은 휘도 신호(Y), 휘도 신호와 적색 성분의 차(U), 휘도 신호와 청색 성분의 차(V)의 3가지 정보로 색을 표현하는 방식
- ❖ 텔레비전에서 많이 사용
- ❖ 일반적으로 Y:U:V의 코드 비율은 4:2:2를 사용

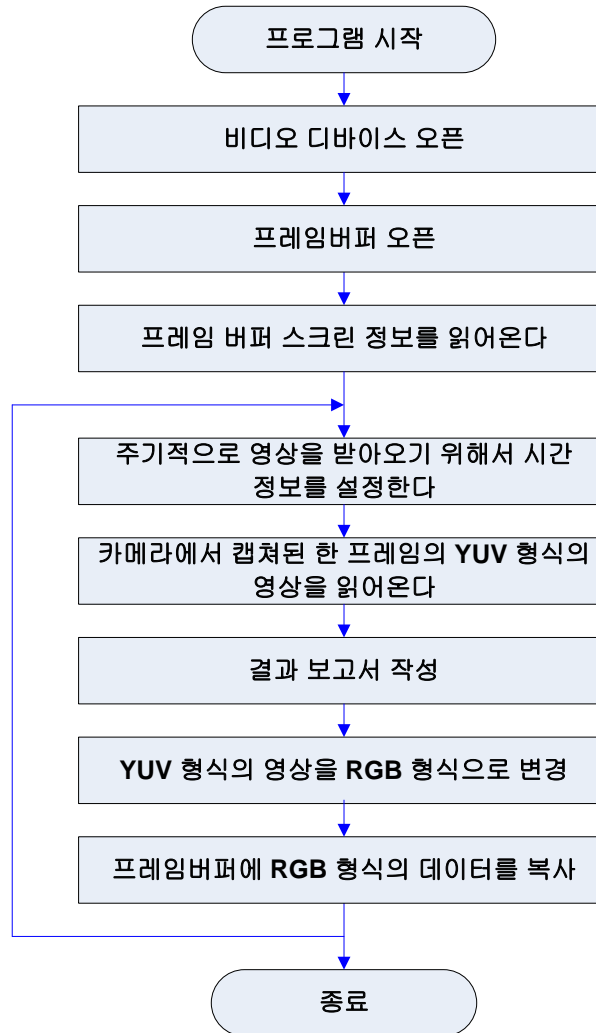
□ LCD에 출력되는 신호 GRB 신호

- ❖ DTK2410에서 사용되는 3.5인치 LCD는 폭 320, 높이 240 화면 크기의 32bpp(bit per pixel) 깊이의 RGB 형식을 사용
- ❖ 32bpp는 24bpp 깊이의 RGB를 32비트로 표현한 것
 - 24비트에 나머지는 알파 블렌드(Alpha Blend) 채널로 사용되어 백그라운드 영상을 표현
- ❖ 실습에서는 알파 채널은 사용하지 않고 0값으로 한다

□ 형식 변환 프로그램

- ❖ RGB 형식을 사용하는 LCD에 YUV 형식으로 입력되는 영상을 출력하기 위한 프로그램

애플리케이션의 구조



프로젝트 1

□ **USB로 입력되는 영상을 LCD 화면에 출력**

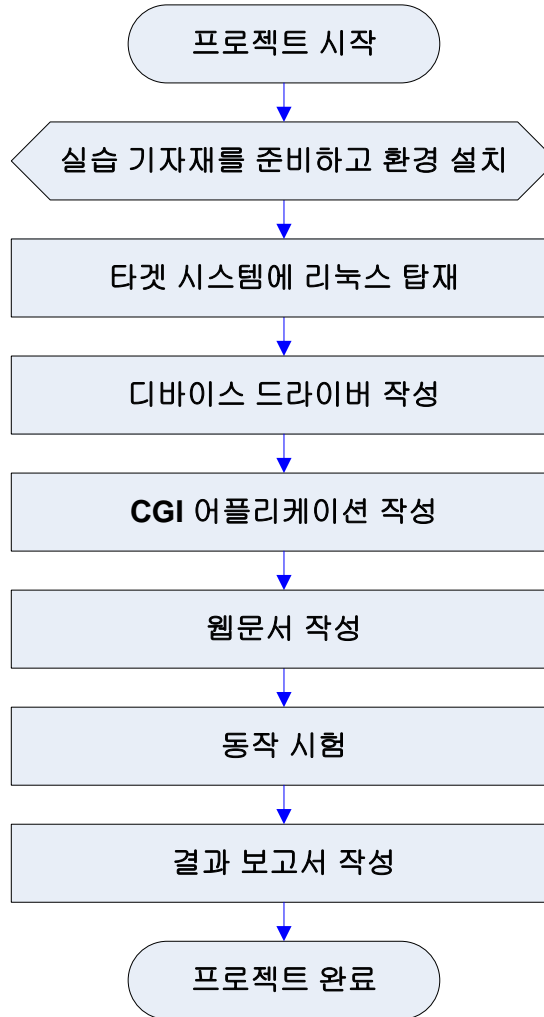
목 차

23장. 임베디드 리눅스의 활용

□ 24장. 미니 프로젝트

- 01. 카메라에 입력된 영상 LCD로 출력
- 02. 웹서버를 활용한 입출력 장치 제어
- 03. 접프-업 프로젝트

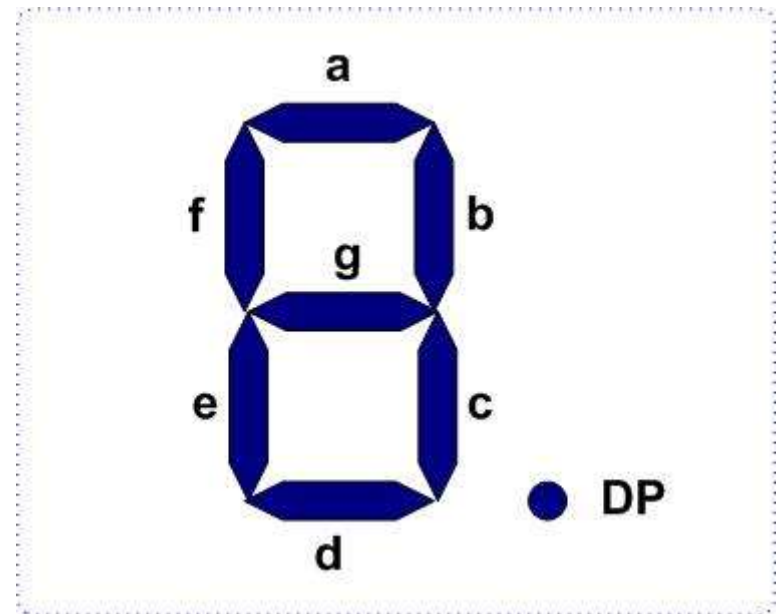
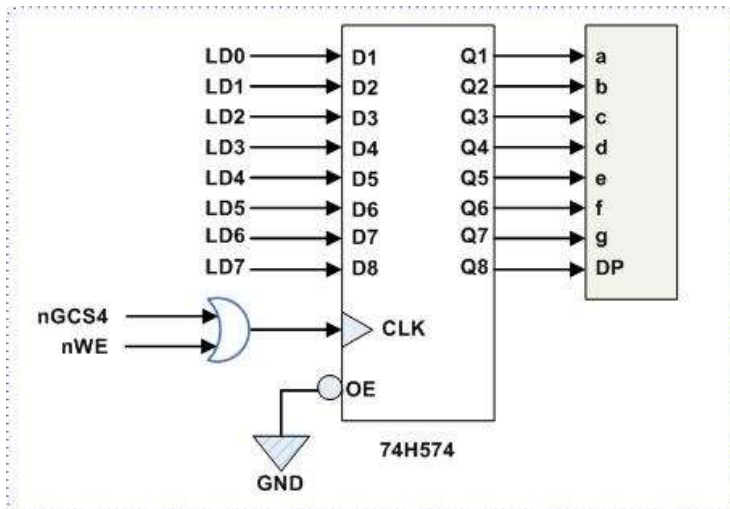
프로젝트 진행 과정



7-세그먼트 LED 이해

□ 7-세그먼트 LED

- ❖ LED를 숫자 모양으로 배치해 놓은 것
- ❖ a,b,c,d,e,f,g,DP LED 제어
 - 0 : LED 켜
 - 1 : LED 끄



7-세그먼트 LED 제어 예

표시 값	16진 값	비트 별 제어 값							
		DP	g	f	e	d	c	b	a
0	0x40	0	1	0	0	0	0	0	0
1	0x79	0	1	1	1	1	0	0	1
2	0x24	0	0	1	0	0	1	0	0
3	0x30	0	0	1	1	0	0	0	0
4	0x19	0	0	0	1	1	0	0	1
5	0x12	0	0	0	1	0	0	1	0
6	0x02	0	0	0	0	0	0	1	0
7	0x78	0	1	1	1	1	0	0	0

웹문서 작성

- 숫자 버튼을 만들어 숫자를 클릭하면 LED에 클릭된 숫자를 표시



프로젝트 2

- 웹으로 인터넷에서 **7-세그먼트 LED**를 제어하는 디바이스 드라이버, **CGI** 애플리케이션, 웹문서를 개발 한다

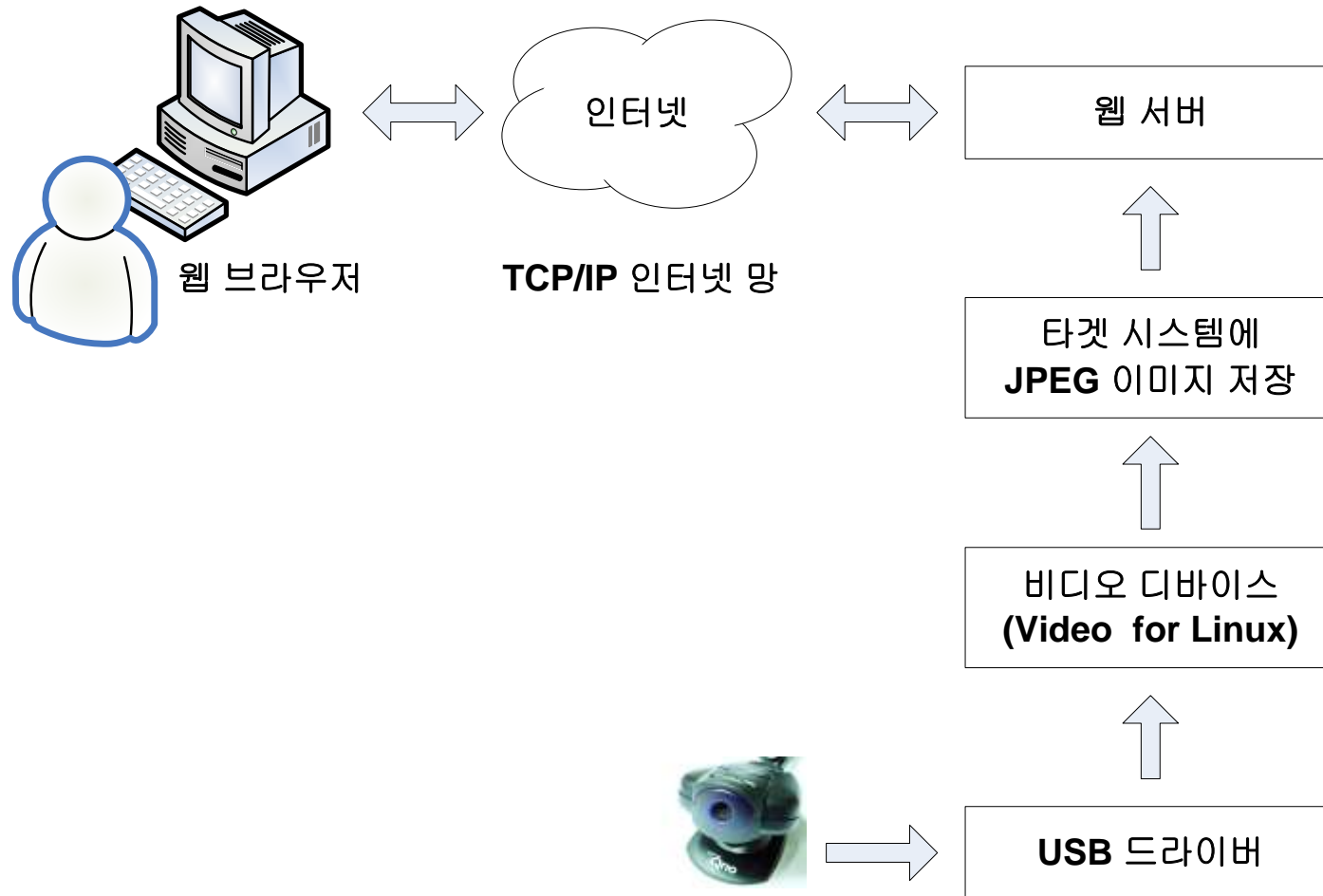
목 차

23장. 임베디드 리눅스의 활용

□ 24장. 미니 프로젝트

- 01. 카메라에 입력된 영상 LCD로 출력
- 02. 웹서버를 활용한 입출력 장치 제어
- 03. 접프-업 프로젝트

웹 카메라 동작 과정



프로젝트 진행

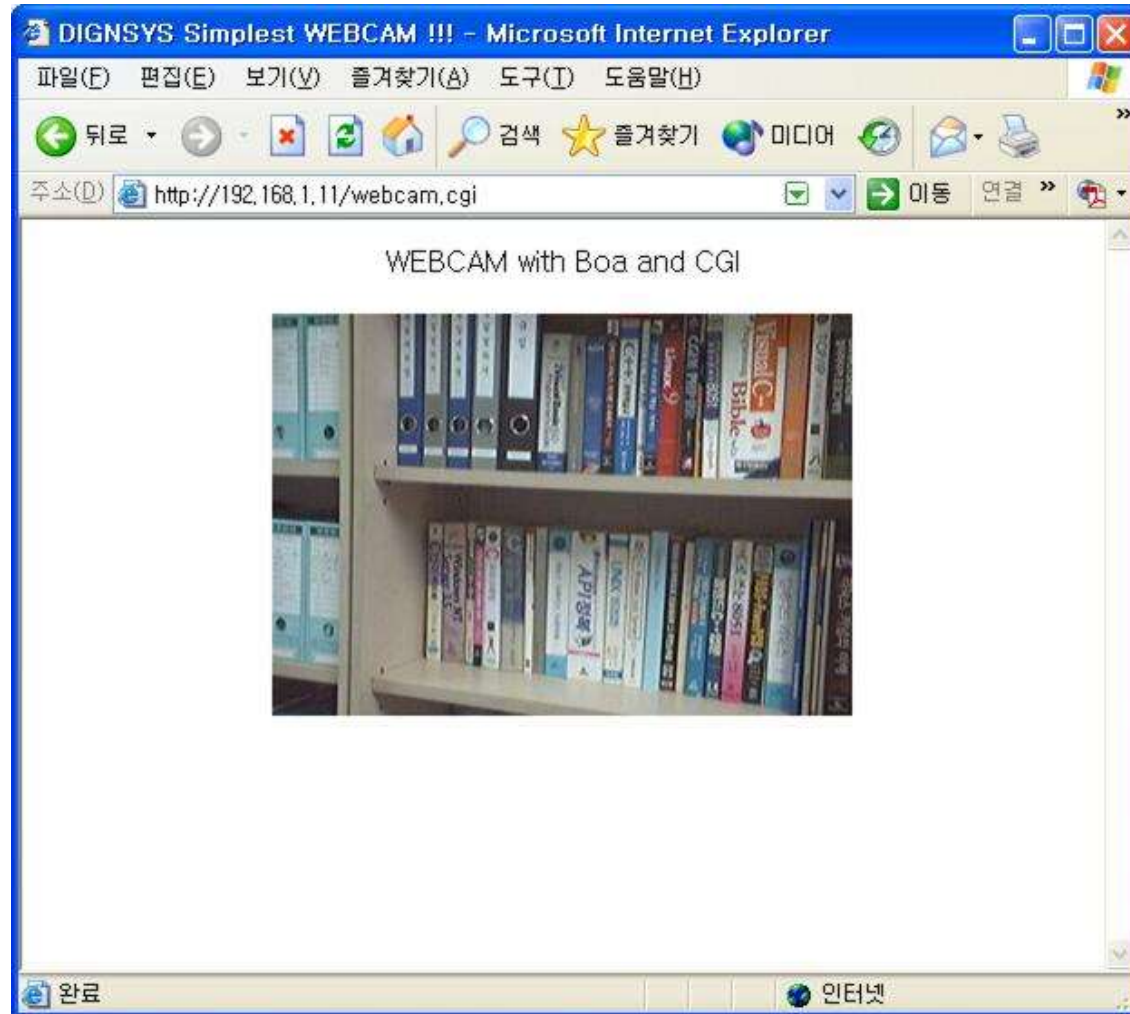
□ JPEG 데이터 변환 프로그램 작성

- ❖ USB 카메라에서 입력된 YUV를 웹에서 사용할 수 있는 JPEG 형식으로 변환

□ META 태그 사용

- ❖ 실시간으로 영상 업데이트
- ❖ 주기적으로 리프레쉬(refresh)
 - HTML 문서의 META 태그를 사용하여 처리

웹 카메라 동작 화면



질의 응답