㈜세븐코아

ANDROID – 포팅

본 문서는 ㈜세븐코아의 SC-PXA270 보드에서 동작하는 ANDROID 개발 환경 및 소프트웨어 구성을 기술한다.

1. 개발 환경 설정 및 소스 코드 다운로드

안드로이드 플랫폼은 리눅스와 MacOS를 기본 개발 운영체제로 사용한다. 본 문서에서는 Ubuntu Linux를 사용하여 기본 개발 환경을 구축하고 안드로이드 소스코드를 다운 받는 방법을 설명한다. 다른 Linux를 사용하거나 MacOS를 사용하는 경우 공식 안드로이드 오픈 소스 프로젝트 사이트(http://source.android.com/)를 통해 개발환경 설정 방법과 소스 코드 다운로드 받는 방법을 확인할 수 있다.

안드로이드 응용 프로그램은 에뮬레이터를 통해서도 개발 할 수 있다. 안드로이드 응용 프로그램 공식 개발 사이트(http://code.google.com/intl/ko/android/intro/index.html)를 통해서 개발 환경을 설정하고 기본적인 개발 과정 및 방법들을 배울 수 있다.

1.1. 안드로이드 플랫폼 개발 환경 설정

1.1.1. Ubuntu Linux:i386

i386 계열의 CPU를 사용하고 Ubuntu를 동작시키는 호스트에서 개발하는 경우 안드로이드 개발을 위해서 다음 개발 툴들을 설치하거나 설치 여부를 확인한다.

- Git (버전 1.5.4 이상)와 GNU Privacy Guard

```
$ sudo apt-get install git-core gnupg
```

- JDK 5.0, update 12 또는 그 이상

```
$ sudo apt-get install sun-java6-j아
```

- Flex, bison, gperf, libs이-dev, libesd0-dev, libwxgtk2.6-dev(optional), build-essential, zip ,curl

```
$ sudo apt-get install flex bison gperf libs이-dev libesd0-dev libwxgtk2.6-dev build-essential zip curl libncurses5-dev zlib1g-dev
```

- Valgrind, memory leaks, stack corruption, array bounds overflows 등을 찾는데 사용한다.

```
$ sudo apt-get install valgrind
```

- Intrepid(8.10) 사용자는 새로운 버전의 libreadline이 필요할 수도 있다.

```
$ sudo apt-get install lib32readline5-dev
```

VMware 같은 virtual machne에서 리눅스를 사용하는 경우 안드로이드 소스를 빌드하기 위해서는 최소한 10GB 이상의 공간을 할당해야한다.

개발 툴들의 공식 웹사이트는 다음과 같다.

Python 2.4 : http://www.python.org/download/

JDK 5.0, update 12 : http://java.sun.com/javase/downloads/index.jsp

Git 1.5.4 : http://git-scm.com/

1.1.2. Repo 설치

안드로이드 소스는 Git 사용하여 관리되는데 repo는 안드로이드 소스 접근을 보다 쉽게 만들어 주는 툴이다. 다음과 같이 호스트 컴퓨터에 repo를 설치한다.

- Repo를 설치할 ~/bin 디렉토리를 생성하고 실행 경로 환경 변수에 추가한다.

$ cd ~

$ mkdir bin

$ export PATH=~/bin/:$PATH (리눅스 쉘 종류에 따라 export 대신 set이 사용될 수 있다.)

$ echo $PATH

- Repo 스크립트를 다운로드하고 실행 파일로 속성을 변경한다.

$ curl http://android.git.kernel.org/repo > ~/bin/repo

$ chmod a+x ~/bin/repo

Curl 명령을 실행할 때 실행 권한 문제가 발생하는 경우 root 계정으로 변경한 후 repo를 설치한다.

1.1.3. Repo 클라이언트 초기화

- 안드로이드 소스를 다운 받을 작업 디렉토리를 생성한다.

$ make ~/ANDROID_SOURCE

$ cd ~/ANDROID_SOURCE

- 최신 버전의 repo를 다운 받기 위해 repo init을 실행한다.

$ repo init –u git://android.git.kernel.org/platform/manifest.git

- repo init을 실행할 때 이름과 이메일 주소를 물어 보는데 자신이 수정한 안드로이드 소스를 배포할 계획이 없다면 적당히 입력해도 상관 없다.

1.1.4. 파일 다운로드 및 빌드

안드로이드 소스를 다운로드할 디렉토리로 이동하여 다음 명령을 실행한다.

$ repo sync

안드로이드 소스 트리의 최상위 디렉토리에서 make를 실행하므로 소스코드 전체(커널 제외)를 빌드할 수 있다. 커널을 포팅할 보드에 맞춰서 설정 및 수정한 수 따로 빌드한다.

$ cd ~/ANDROID_SOURCE

$ make

run-java-tool과 관련하여 에러가 발생해 소스코드 빌드에 실패한 경우 다음과 같이 환경 변수를 설정해 준다.

$ export ANDROID_JAVA_HOME = $JAVA_HOME

안드로이드 플랫폼 소스 전체를 빌드하려면 7GB 정도의 공간이 필요하고 호스트 컴퓨터의 성능에 따라 다르겠지만 많은 시간이 필요할 것이다

1.2. 안드로이드 응용 프로그램 개발 환경 설정

안드로이드는 현재 ARM 계열의 시스템에서 동작하도록 되어 있지만 에뮬레이터를 사용하여 일반 윈도우 환경에서도 안드로이드 응용 프로그램을 개발할 수 있다.

1.3. 안드로이드 SDK 설치

1.3.1. 개발 툴 준비

안드로이드 SDK를 사용해 응용프로그램을 개발하기 위해서는 다음과 같은 개발 환경이 필요하다.

- 운영 체제
  - WindowXP 또는 Vista
  - Mac OS X 10.4.8 또는 그 이상의 버전 (x86 only)
  - Linux (Ubuntu Dapper Drake)
- 개발 툴
  - Eclipse 3.3(Europa), 3.4(Ganymede)
  - JDK 5 또는 JDK 6
  - Android Development Tools plugin

1.3.2. 안드로이드 SDK 다운로드 및 설치

안드로이드 응용 프로그램 개발을 위한 SDK는 공식 사이트를 통해서 다운로드 받을 수 있다. (http://code.google.com/intl/ko/android/download.html)

안드로이드 SDK를 다운로드 받은 후 적절한 위치에 압축을 푼다. 기본적으로 안드로이드 SDK는 android_sdk_<platform>_<release>_<build> 형태의 이름을 갖고 디렉토리에 tools/ 와 samples/라는 서브디렉토리가 생성된다. Tools 디렉토리를 실행 경로에 추가한다.

- 윈도우 환경
  - 내 컴퓨터에서 오른쪽 마우스 클릭
  - 속성 선택
  - 고급 탭 아래에 환경 변수 버튼 클릭
  - 다이얼 로그 화면이 나타나면 시스템 변수의 Path 값을 tools/ 디렉토리가 있는 경로를 추가하여 수정한다.
- 리눅스 환경
  - Export PATH=${PATH}:<your_sdk_dir>/tools
- Mac OS
  - .bash_profile 수정

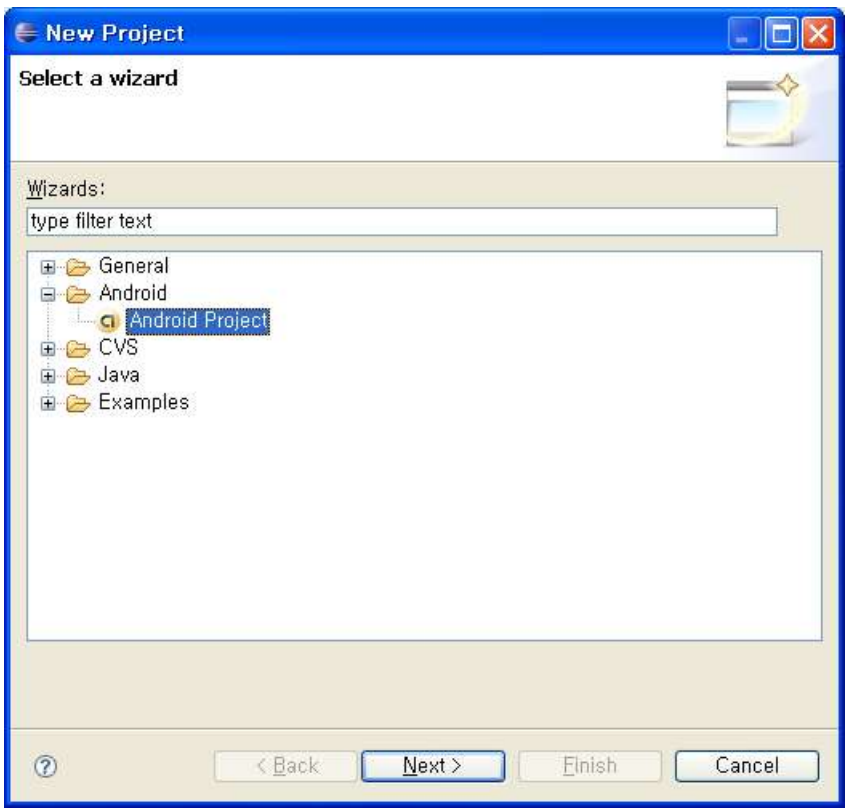
1.3.3. Eclipse 플러그인(ADT) 설치

Android Development Tools(ADT)는 안드로이드 응용 프로그램을 빠르고 쉽게 생성, 실행, 디버깅할 수 있도록 하는 이클립스용 플러그인이다. 본 문서에서는 Eclipse 3.4(Ganymede)를 사용한다고 가정한다. Eclipse 3.3(Europa)를 사용하는 경우는 공식 사이트를 통해 설치 방법을 참고 하길 바란다. (http://code.google.com/intl/ko/android/intro/installing.html)

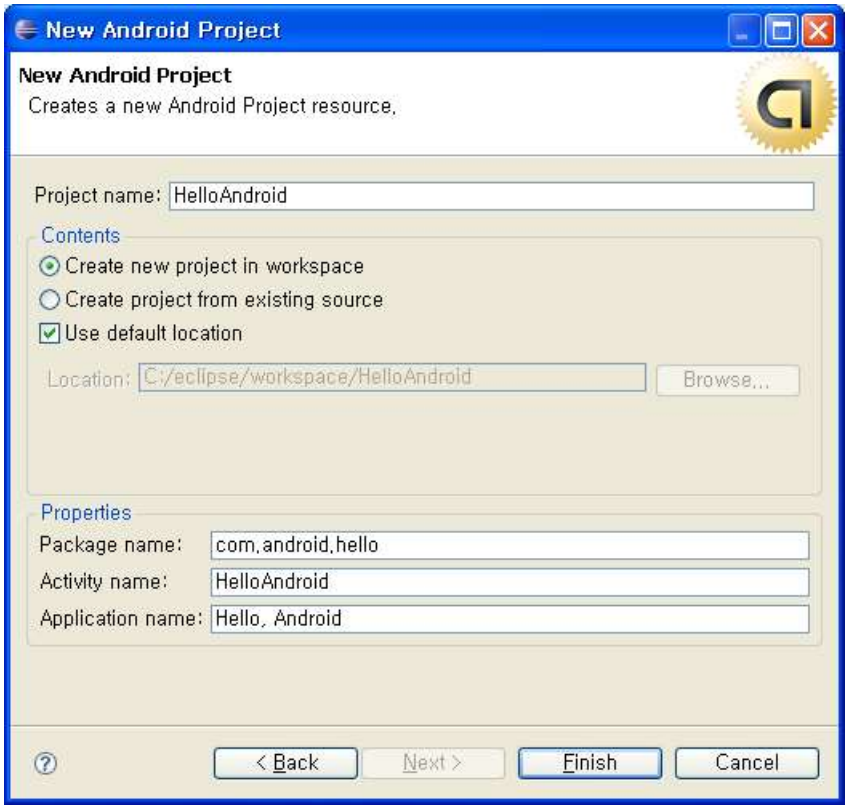
Eclipse 3.4 (Ganymede)
I) 이클립스 실행 Help -> Software Updates II) 다이얼로그가 나타나면 Available Software 탭 클릭 III) Add site 클릭 IV) https://dl-ssl.google.com/android/eclipse/ OK 클릭 V) 다시 Available Software view로 돌아갔을 때 플러그인이 보이지 않으면 위의 주소가 잘못 된 경우이므로 IV)에서 http://dl-ssl.google.com/android/eclipse/으로 다시 입력한다. 체크박스를 선택하고 install을 클릭한다. VI) 이어서 나타나는 윈도우에 Android Developer Tools와 Android Editors가 체크되어 있는지 확인한다. VII) 라이선스에 동의하고 Finish를 클릭한다. VIII) 이클립스를 다시 시작한다. IX) Window -> Preferences 실행 X) 왼쪽 패널에서 안드로이드를 선택하고 SDK의 위치를 지정한 후 Apply와 OK를 순서대로 클릭한다.

1.3.4. HelloWorld 실행

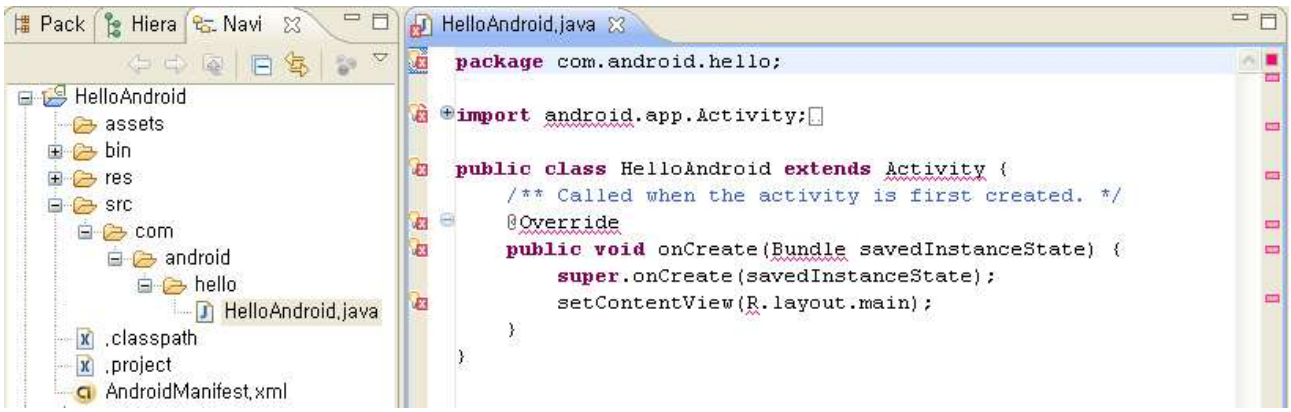
- 프로젝트 생성 (File -> New -> Project)



- 프로젝트의 세부 항목을 채운다.



- 프로젝트의 HelloAndroid>src>com>android>hello 아래의 HelloAndroid.java 의 내용을 확인한다.



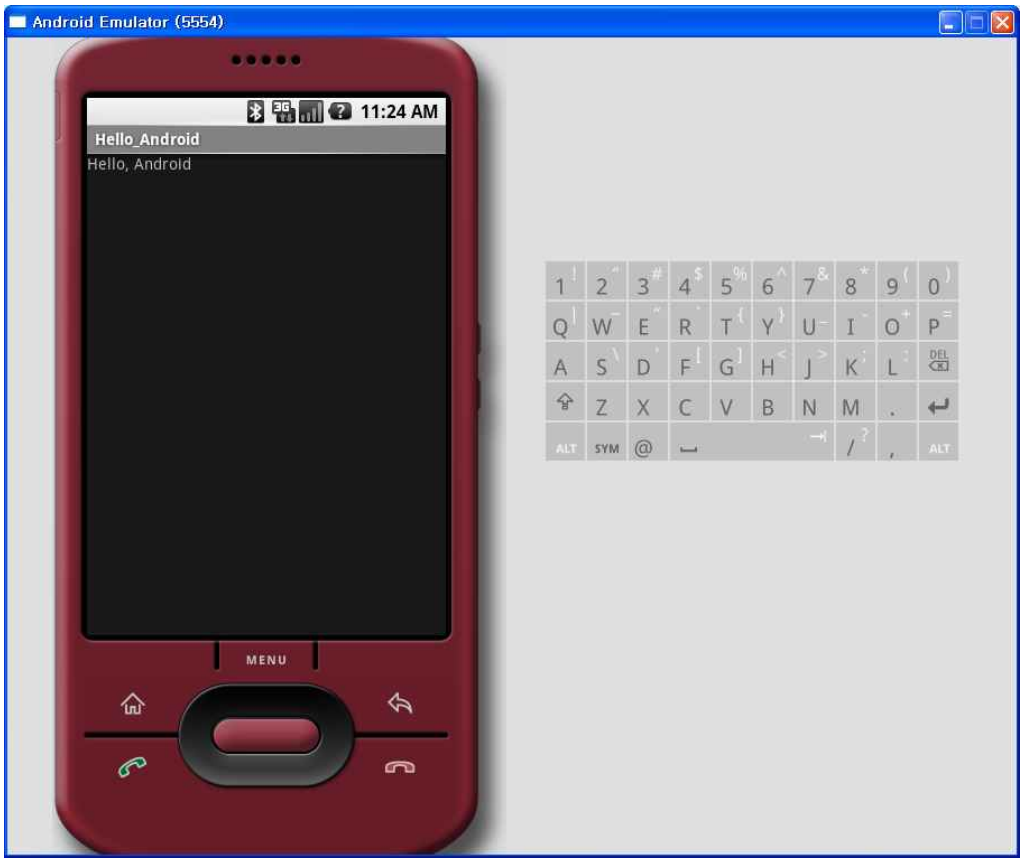
- HelloAndroid.java를 다음과 같이 수정한다.

```
package com.android.hello;

import android.app.Activity;
import android.os.Bundle;
```

```
import android.widget.TextView;

public class HelloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView tv = new TextView(this);
        tv.setText("Hello, Android");
        setContentView(tv);
    }
}
```

- Run(Ctrl + F11) 안드로이드 실행

- 안드로이드 에뮬레이터 실행화면


Android Emulator (5554)
11:24 AM
Hello_Android
Hello, Android
MENU


Android Emulator (5554)
11:24 AM
Dialer
Contacts
Browser
Maps
MENU

2. 안드로이드 리눅스 커널

안드로이드 리눅스 커널은 소스코드 패키지 중에서 kernel 디렉토리에 있다. Makefile을 통해서 커널 버전을 확인 할 수 있다.

SC-PXA270 보드에는 안드로이드 소스 코드 프로젝트 공식 사이트에서 배포한 2.6.27 안드로이드 리눅스 커널을 디바이스 드라이버와 함께 포팅하였다. 특별히 안드로이드를 포팅을 위해 일반 리눅스 디바이스 드라이버와 다른 부분은 프레임 버퍼와 입력 장치, 커널 설정이 있다.

2.1. 커널 설정

안드로이드를 위해서 반드시 필요한 커널 설정은 다음과 같다.

```
CONFIG_AEABI=y
CONFIG_OABI_COMPAT=y
CONFIG_ANDROID_POWER=y
CONFIG_ANDROID_PMEM=y
CONFIG_BINDER_IPC=y
CONFIG_LOW_MEMORY_KILLER=y
```

2.2. 커널 컴파일

make 또는 make zImage 수행

2.3. 프레임 버퍼

안드로이드는 더블 버퍼링 부분과 프레임 버퍼 ioctl 함수인 file operation인 .fb_pan_display 코드를 작성해야 한다.

다음과 같이 pxafb.c 파일 수정

```
--- pxafb_old.c         2009-01-28 10:46:00.000000000 +0900
+++ pxafb_new.c         2009-02-24 15:25:10.000000000 +0900
@@ -50,7 +50,7 @@
 #include <asm/irq.h>
 #include <asm/div64.h>
 #include <mach/pxa-regs.h>
-#include <mach/pxa2xx-gpio.h>
+#include <mach/mfp-pxa27x.h>
 #include <mach/bitfield.h>
 #include <mach/pxafb.h>


@@ -69,6 +69,8 @@
 #define LCCR3_INVALID_CONFIG_MASK  (LCCR3_HSP | LCCR3_VSP |\
```

```
                                        LCCR3_PCD | LCCR3_BPP)

+#define C_CHANGE_DMA_BASE               (10)
+
 static void (*pxafb_backlight_power)(int);
 static void (*pxafb_lcd_power)(int, struct fb_var_screeninfo *);

@@ -309,7 +311,7 @@
                var->sync                       = mode->sync;
                var->grayscale                          = mode->cmap_greyscale;
                var->xres_virtual       = var->xres;
-               var->yres_virtual       = var->yres;
+               var->yres_virtual       = var->yres * 2;
 }

 /*
@@ -350,7 +352,7 @@
                var->xres_virtual =
                  max(var->xres_virtual, var->xres);
                var->yres_virtual =
-                 max(var->yres_virtual, var->yres);
+                 max(var->yres_virtual, var->yres * 2);

                /*
                 * Setup the RGB parameters for this display.
@@ -516,6 +518,34 @@
                return -EINVAL;
 }

+#ifdef CONFIG_ANDROID_POWER
+static void pxafb_early_suspend(android_early_suspend_t *h)
+{
+#if 0
+               struct pxafb_info *fb = container_of(h, struct pxafb_info, early_suspend);
+#endif
+}
+
```

```
+static void pxafb_late_resume(android_early_suspend_t *h)
+{
+#if 0
+		struct pxafb_info *fb = container_of(h, struct pxafb_info, early_suspend);
+#endif
+}
+#endif
+
+
+
+
+static int pxafb_pan_display(struct fb_var_screeninfo *var, struct fb_info *info)
+{
+		struct pxafb_info *fbi = (struct pxafb_info *)info;
+
+		fbi->fb.var.yoffset = var->yoffset;
+		pxafb_schedule_work(fbi, C_CHANGE_DMA_BASE);
+		return 0;
+}
+
 static struct fb_ops pxafb_ops = {
		.owner			= THIS_MODULE,
		.fb_check_var		= pxafb_check_var,
@@ -526,6 +556,7 @@
		.fb_imageblit		= cfb_imageblit,
		.fb_blank	= pxafb_blank,
		.fb_mmap	= pxafb_mmap,
+		.fb_pan_display = pxafb_pan_display,
 };

 /*
@@ -814,17 +845,6 @@
		}
		return 0;
 }
-#elif CONFIG_FB_NX1009
-
```

```
-int pxafb_smart_queue(struct fb_info *info, uint16_t *cmds, int n_cmds)
-{
-       return 0;
-}
-
-int pxafb_smart_flush(struct fb_info *info)
-{
-       return 0;
-}



 #else
@@ -1221,6 +1241,8 @@
                        __pxafb_backlight_power(fbi, 1);
                 }
                 break;
+        case C_CHANGE_DMA_BASE:
+                fbi->fb.fix.smem_start = fbi->screen_dma +(fbi->fb.var.xres*fbi->fb.var.yoffset*fbi->fb.var.bits_per_pixel/8);


         }
         mutex_unlock(&fbi->ctrlr_lock);
 }
@@ -1373,6 +1395,7 @@


         for (i = 0; i < num_modes; i++) {
                 smemlen = modes[i].xres * modes[i].yres * modes[i].bpp / 8;
+                smemlen *= 2;
                 if (smemlen > fbi->fb.fix.smem_len)
                         fbi->fb.fix.smem_len = smemlen;
         }
@@ -1451,7 +1474,7 @@
         fbi->fb.fix.type        = FB_TYPE_PACKED_PIXELS;
         fbi->fb.fix.type_aux    = 0;
         fbi->fb.fix.xpanstep    = 0;
-        fbi->fb.fix.ypanstep    = 0;
+        fbi->fb.fix.ypanstep    = 1;
         fbi->fb.fix.ywrapstep   = 0;
```

```
            fbi->fb.fix.accel       = FB_ACCEL_NONE;


@@ -1853,6 +1876,12 @@
                                    CPUFREQ_POLICY_NOTIFIER);
 #endif

+#ifdef CONFIG_ANDROID_POWER
+           fbi->early_suspend.suspend = pxafb_early_suspend;
+//         fbi->early_suspend.resume = pxafb_late_resume;
+           android_register_early_suspend(&fbi->early_suspend);
+#endif
+
            /*
             * Ok, now enable the LCD controller
             */
@@ -1945,3 +1974,4 @@

 MODULE_DESCRIPTION("loadable framebuffer driver for PXA");
 MODULE_LICENSE("GPL");
+
```

pxafb.h 파일을 다음과 같이 수정

```
--- pxafb_old.h         2009-01-09 07:48:59.000000000 +0900
+++ pxafb_new.h         2009-02-24 15:26:45.000000000 +0900
@@ -21,6 +21,12 @@
   * for more details.
   */

+#ifdef CONFIG_ANDROID_POWER
+#include <linux/android_power.h> /* android_early_suspend_t */
+#endif
+
+
+
 /* PXA LCD DMA descriptor */
 struct pxafb_dma_descriptor {
            unsigned int fdadr;
```

```
@@ -124,6 +130,11 @@
                struct notifier_block       freq_transition;
                struct notifier_block       freq_policy;
 #endif
+#ifdef CONFIG_ANDROID_POWER
+               android_early_suspend_t early_suspend;
+#endif
+
+
 };

 #define TO_INF(ptr,member) container_of(ptr,struct pxafb_info,member)
@@ -149,3 +160,4 @@
 #define MIN_YRES   64

 #endif /* __PXAFB_H__ */
+
```

2.4. 입력 장치

2.4.1. 싱글 터치 스크린

안드로이드는 일반 리눅스의 tslib 가 지원하지 않는다. 그러므로 터치 칩에서 받은 데이터를 디바이스 드라이버에서 가공해 실제 LCD 해상도에 해당하는 값으로 응용프로그램에 전달한다.
다음 코드는 TSC2007 터치칩 과 800 x 480의 해상도를 가진 LCD를 사용할 경우이다.

input device 초기화

```
 input_dev->evbit[0] = BIT_MASK(EV_KEY) | BIT_MASK(EV_ABS) | BIT_MASK(EV_SYN);
 input_dev->keybit[BIT_WORD(BTN_TOUCH)] = BIT_MASK(BTN_TOUCH);
 input_dev->absbit[0] = BIT(ABS_X) | BIT(ABS_Y);

 input_set_abs_params(input_dev, ABS_X, 0, 800, 0, 0);
 input_set_abs_params(input_dev, ABS_Y, 0, 480, 0, 0);
 input_set_abs_params(input_dev, ABS_PRESSURE, 0, 0, 0, 0);
```

터치 칩에서 받은 좌료를 input device에게 전달 함수

```
static void tsc2007_send_event(void *tsc)
```

```
{
	struct tsc2007 *ts = tsc;
	u32 Rt;
	u16 x, y, z1, z2;
	u16 report_x = 0;
	u16 report_y = 0;
	static u16 prev_x = 0;
	static u16 prev_y = 0;

	x = ts->tc.x;
	y = ts->tc.y;
	z1 = ts->tc.z1;
	z2 = ts->tc.z2;

	/* range filtering */
	if (x == MAX_12BIT)
	    x = 0;

	if (likely(x && z1)) {
	    /* compute touch pressure resistance using equation #1 */
	    Rt = z2;
	    Rt -= z1;
	    Rt *= x;
	    Rt *= ts->x_plate_ohms;
	    Rt /= z1;
	    Rt = (Rt + 2047) >> 12;
	} else
	    Rt = 0;

	if (Rt > MAX_12BIT) {
	    dev_dbg(&ts->client->dev, "ignored pressure %d\n", Rt);
	    complete(&ts->penirq_completion);
	    return;
	}

	if (Rt) {
	    struct input_dev *input = ts->input;
```

```
            if (!ts->pendown) {
                        input_report_key(input, BTN_TOUCH, 1);
                        ts->pendown = 1;
            }

            report_x = x * 800 / MAX_12BIT;
            report_x = 800 - report_x;
            report_y = y * 480 / MAX_12BIT;

            if (report_x > 800 || report_x <= 0 )
                        report_x = prev_x;
            if (report_y > 480 ||report_y <= 0 )
                        report_y = prev_y;

            input_report_abs(input, ABS_X, report_x);
            input_report_abs(input, ABS_Y, report_y);
            input_report_abs(input, ABS_PRESSURE, z1);
            input_report_key(input, BTN_TOUCH, 1);
            input_sync(input);

            prev_x = report_x;
            prev_y = report_y;
        }

}
```

3. 안드로이드 루트파일 시스템

이번 장에서는 빌드된 안드로이드 플랫폼 소스에서 안드로이드 루트 파일 시스템을 구성하고 실제 보드에서 실행하는 방법을 설명한다.

3.1. 안드로이드 루트 파일 시스템 구성

위에서 설명한 것처럼 안드로이드 플랫폼 소스를 다운로드 받고 빌드를 실행하면 out/ 디렉토리가 새로 생성되면서 빌드 결과물이 복사된다. 안드로이드 루트 파일 시스템은 out/target/product/generic/ 아래에 생성된다. out/target/product/generic 아래의 root 디렉토리가 안드로이드 루트파일 시스템 디렉토리이다.

- 안드로이드 루트 파일 시스템 디렉토리로 이동

$ cd [안드로이드 작업 디렉토리]/out/product/generic/

$ ls root

```
data  default.prop  dev  init  init.goldfish.rc  init.rc  proc  sbin  sys  system
```

root/ 디렉토리의 대부분의 디렉토리는 비어 있는데 실제로 안드로이드 프로그램이 들어있는 system/ 디렉토리는 root/ 디렉토리 밖에 따로 구성되어 있다. 그러므로 완전한 안드로이드 루트 파일 시스템을 만들기 위해서는 root/system/ 아래에 system/ 디렉토리의 내용이 모두 복사되어야 한다.

- System/ 디렉토리 내용 확인

$ ls root/system

$ ls system

```
app  bin  build.prop  etc  fonts  framework  lib  media  sounds  usr  xbin
```

- System/ 디렉토리 내용 복사

$ cp –a system/* root/system/

- 디바이스 노드 생성 (이전 파일 시스템에서 복사하거나 새로 생성한다.)

$ cp –a [기존의 루트파일시스템]/dev/* root/dev/

또는

$ cd root/dev/

$ mknod [device node name] [c/b] [major number] [minor number]

SCPXA270R5 보드에는 다음과 같이 노드가 생성되어 있다.

```
crw-r--r-- 1 root root   10, 134 Nov 22   2006 apm_bios
crw------- 1 root root   14,    4 Dec  5   2007 audio
crw------- 1 root root   14,   20 Dec  5   2007 audio1
crw-rw-rw- 1 root root   14,   36 Dec  5   2007 audio2
crw-rw-rw- 1 root root   14,   52 Dec  5   2007 audio3
crw------- 1 root root   14,    7 Dec  5   2007 audioctl
crw--w--w- 1 root root    5,    1 Dec  5   2007 console
```

```
crw------- 1 root root   14,    3 Dec   5   2007 dsp
crw------- 1 root root   14,   19 Dec   5   2007 dsp1
crw-rw-rw- 1 root root   14,   35 Dec   5   2007 dsp2
crw-rw-rw- 1 root root   14,   51 Dec   5   2007 dsp3
lrwxrwxrwx 1 root root            3 Feb 17 20:24 fb -> fb0
crw------- 1 root root   29,    0 Dec   5   2007 fb0
crw------- 1 root root   29,    1 Dec   5   2007 fb1
crw------- 1 root root   29,    2 Dec   5   2007 fb2
crw------- 1 root root   29,    3 Dec   5   2007 fb3
crw------- 1 root root   29,    4 Dec   5   2007 fb4
crw------- 1 root root   29,    5 Dec   5   2007 fb5
crw------- 1 root root   29,    6 Dec   5   2007 fb6
crw------- 1 root root   29,    7 Dec   5   2007 fb7
prw------- 1 root root            0 Dec   5   2007 initctl
drwxr-xr-x 2 root root        4096 Dec   5   2007 input
crw-r--r-- 1 root 1016   10,   90 Jan   7   2008 ipm
crw-r--r-- 1 root 1016 233,    0 Jan   7   2008 ipmc
crw-r----- 1 root root    1,    2 Dec   5   2007 kmem
brw-rw---- 1 root root    7,    0 Dec   5   2007 loop0
brw-rw---- 1 root root    7,    1 Dec   5   2007 loop1
brw-rw---- 1 root root    7,    2 Dec   5   2007 loop2
crw-r----- 1 root root    1,    1 Dec   5   2007 mem
crw------- 1 root root   14,    2 Dec   5   2007 midi00
crw------- 1 root root   14,   18 Dec   5   2007 midi01
crw------- 1 root root   14,   34 Dec   5   2007 midi02
crw------- 1 root root   14,   50 Dec   5   2007 midi03
crw------- 1 root root   14,    0 Dec   5   2007 mixer
crw------- 1 root root   14,   16 Dec   5   2007 mixer1
crw-rw-rw- 1 root root   14,   32 Dec   5   2007 mixer2
crw-rw-rw- 1 root root   14,   48 Dec   5   2007 mixer3
brw-r--r-- 1 root root 254,    0 Nov 22   2006 mmcblk
brw-r--r-- 1 root root 254,    1 Nov 22   2006 mmcblk0
lrwxrwxrwx 1 root root            5 Feb 17 20:24 mouse -> psaux
crw-rw-r-- 1 root root   90,    0 Dec   5   2007 mtd0
crw-rw-r-- 1 root root   90,    2 Dec   5   2007 mtd1
crw-rw-r-- 1 root root   90,    4 Dec   5   2007 mtd2
crw-rw-r-- 1 root root   90,    6 Dec   5   2007 mtd3
```

```
crw-rw-r-- 1 root root  90,   8 Dec  5  2007 mtd4
crw-rw-r-- 1 root root  90,  10 Dec  5  2007 mtd5
brw-rw-r-- 1 root root  31,   0 Dec  5  2007 mtdblock0
brw-rw-r-- 1 root root  31,   1 Dec  5  2007 mtdblock1
brw-rw-r-- 1 root root  31,   2 Dec  5  2007 mtdblock2
brw-rw-r-- 1 root root  31,   3 Dec  5  2007 mtdblock3
brw-rw-r-- 1 root root  31,   4 Dec  5  2007 mtdblock4
brw-rw-r-- 1 root root  31,   5 Dec  5  2007 mtdblock5
crw-rw-rw- 1 root root   1,   3 Dec  5  2007 null
crw------- 1 root root 108,   0 Dec  5  2007 ppp
crw------- 1 root root  10,   1 Dec  5  2007 psaux
crw-rw-rw- 1 root root   5,   2 Dec  5  2007 ptmx
drwxr-xr-x 2 root root        4096 Dec  5  2007 pts
crw-r--r-- 1 root root 254,   0 Nov 22  2006 pvrsrv
brw-rw---- 1 root root   1,   0 Dec  5  2007 ram0
crw-r--r-- 1 root root   1,   8 Dec  5  2007 random
brw-rw---- 1 root root   8,   0 Dec  5  2007 sda
brw-rw---- 1 root root   8,   1 Dec  5  2007 sda1
crw------- 1 root root  14,   1 Dec  5  2007 sequencer
drwxr-xr-x 2 root root        4096 Aug  1  2006 snd
crw------- 1 root root  14,   6 Dec  5  2007 sndstat
crw-r--r-- 1 root root  11,   0 Dec  5  2007 ts
crw-rw-rw- 1 root root   5,   0 Dec  5  2007 tty
crw--w---- 1 root root   4,   0 Dec  5  2007 tty0
crw--w---- 1 root root   4,   1 Dec  5  2007 tty1
crw------- 1 root root   4,   2 Dec  5  2007 tty2
crw------- 1 root root   4,   3 Dec  5  2007 tty3
crw------- 1 root root   4,   4 Dec  5  2007 tty4
crw------- 1 root root   4,   5 Dec  5  2007 tty5
crw------- 1 root root   4,   6 Dec  5  2007 tty6
crw--w---- 1 root root   4,   7 Dec  5  2007 tty7
crw--w---- 1 root root   4,   8 Dec  5  2007 tty8
crw--w---- 1 root root   4,   9 Dec  5  2007 tty9
crw-r--r-- 1 root root 166,   0 Dec 12  2006 ttyACM0
crw-rw---- 1 root root   4,  64 Dec  5  2007 ttyS0
crw-rw---- 1 root root   4,  65 Dec  5  2007 ttyS1
crw-r--r-- 1 root root   1,   9 Dec  5  2007 urandom
```

```
crw------- 1 root root   81,    0 Dec   5   2007 video0
crw------- 1 root root   81,    1 Dec   5   2007 video1
crw-rw-rw- 1 root root    1,    5 Dec   5   2007 zero
```

- 루트 파일 시스템 압축

```
$ tar cfz android_rootfs.tar.gz root/
```

3.2. 안드로이드 루트 파일 시스템 사용

안드로이드 루트 파일 시스템은 EXT3나 YAFFS2 파일 시스템에서만 정상적으로 동작한다. 압축된 안드로이드 루트 파일 시스템은 약 32MB이므로 SCPXA270R5 보드의 NAND flash memory에 안드로이드 루트 파일 시스템을 복사한다.

3.2.1. 안드로이드 루트파일 시스템을 YAFFS2로 사용

- NAND flash를 YAFFS2로 마운트

```
$ mount –t yaffs2 /dev/mtdblock4 /mnt/mtd1
```

- MMC 카드를 이용해 안드로이드 루트 파일 시스템 복사

```
$ mount –t vfat /dev/mmcblk0 /mnt/card
$ cd /mnt/mtd1
$ tar zxf /mnt/card/android_rootfs.tar.gz
$ ls /mnt/mtd1
```

```
root lost+found
```

```
$ mv root android
```

안드로이드를 실행하기 위해서는 안드로이드 루트 디렉토리로 루트 파일 시스템을 변경하고 init 프로세스를 실행한다.

```
$ chroot /mnt/mtd1/android /init
```

“ANDROID_”가 콘솔 창에 출력되고 안드로이드 로봇이 보인 후 안드로이드 초기 화면을 볼 수 있다.

안드로이드가 자동으로 부팅되게 하려면 기존 루트파일 시스템의 실행 스크립트를 수정해 위에서 설명한 디바이스 마운트, 안드로이드 루트 파일 시스템 변경, init 프로세스 실행이 자동으로 실행되도록 한다.

3.2.2. 안드로이드 루트파일 시스템을 EXT3로 사용

MMC 카드를 EXT3로 포맷한 후 MMC 카드를 사용해 안드로이드를 부팅할 수 있다. 개발 환경에서 루트 파일 시스템의 변경이 잦은 경우 MMC 카드를 이용해 부팅하면 루트파일 시스템을 보드로 복사할 필요가 없으므로 개발 시간을 줄일 수 있다.

- MMC 카드를 EXT3로 포맷 (MMC 카드를 usb 젠더를 사용해서 호스트에 연결)

mkfs.ext3 /dev/sda

- 안드로이드 루트 파일 시스템을 MMC로 복사

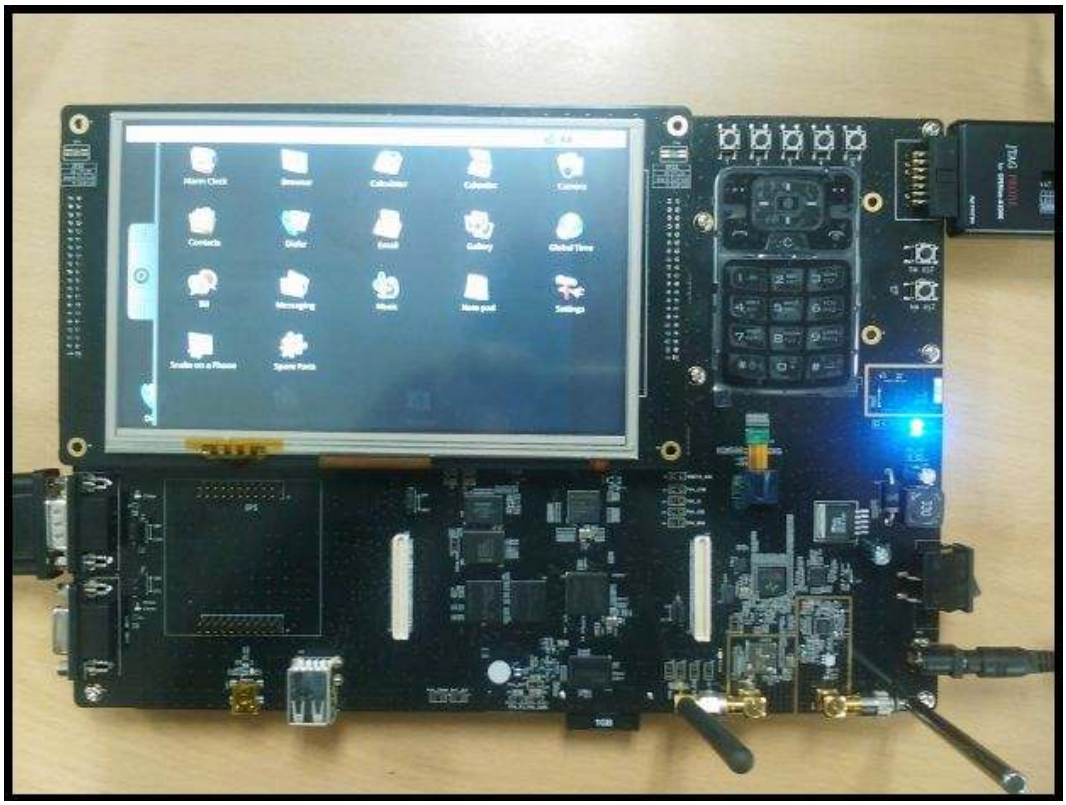
$ cp root/ /mnt/card/android

- MMC 카드를 보드에 EXT3로 마운트

mount –t ext3 /dev/mmcblk0 /mnt/card

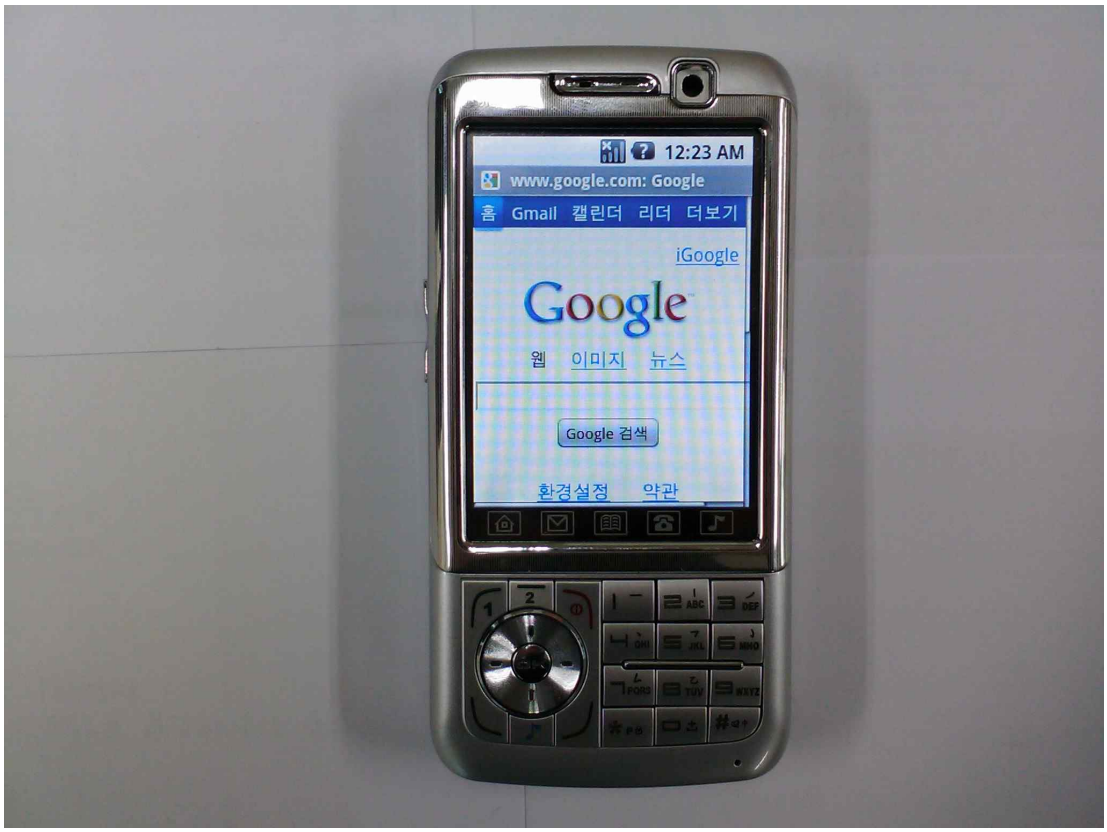
- 안드로이드 루트 파일 시스템으로 루트 파일 시스템을 변경하고 init 프로세스 실행

chroot /mnt/card/android /init



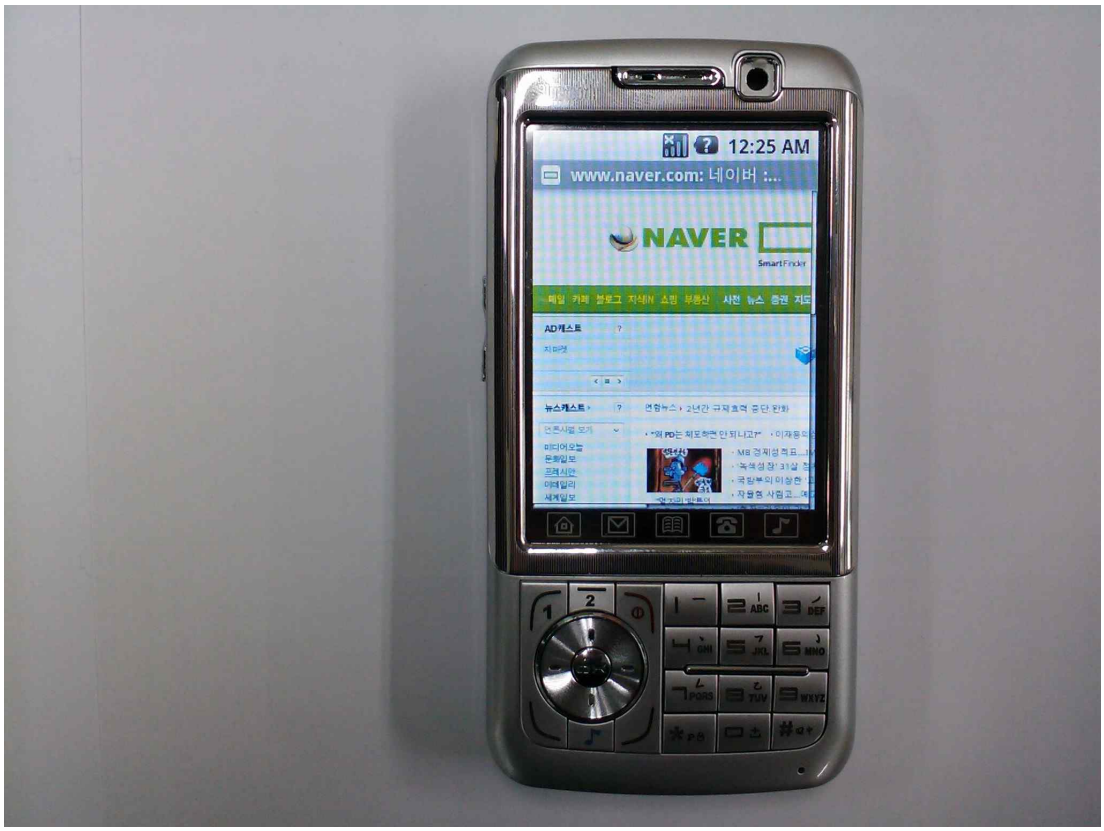

12:02 AM
Alarm Clock
Browser
Calculator
Calendar
Camera
Contacts
Dialer
Email


12:21 AM
Dialer
Contacts
Browser


12:23 AM
www.google.com: Google
홈 Gmail 캘린더 리더 더보기
iGoogle
Google
웹 이미지 뉴스
Google 검색
환경설정 약관


12:23 AM
www.google.com: Google
홈 Gmail 캘린더 리더 더보기
iGoogle
Go to URL
Search
Bookmarks
New window
Refresh
More


12:23 AM
Bookmarks
New bookmark
Current page: Google
네이버 :: 세상의 모든
http://www.naver.com/
24 HOUR NEWS
http://www.ytn.co.kr/_cn/0107..
Google
http://www.google.com/
Yahoo!
http://www.yahoo.com/


12:25 AM
www.naver.com: 네이버 :...
NAVER


12:02 AM
Alarm Clock
Browser
Calculator
Calendar
Camera
Contacts
Dialer
Email


12:26 AM
www.naver.com: 네이버 :...
NAVER
SmartFinder
AD캐스트
뉴스캐스트