Embedded System Software

Project Report

과 목 명 : [CSE4116] 임베디드시스템소프트웨어

담당 교수 : 서강대학교 컴퓨터공학과 박성용

학번/이름 : 20151556 변홍수

개발 기간 : 2020. 06. 10. ~ 2020. 06. 28.

I. 개발 목표

JNI를 사용한 택시 미터기 프로그램과 지금까지 진행한 숙제(HW1 – HW3)를 외부와 연결을 통한 입력 없이 자체적으로 실행 할 수 있는 하나의 안드로이드 어플리케이션을 구현한다.

II. 개발 범위 및 내용

가. Kernel Image

- ✓ Interrupt Handler 사용을 위한 GPIO Button 수정
- ✓ Custom Module 등록

나. Android Application

1) C App, Module 실행/등록을 위한 Execute Linux Command
2) Interface

- ✓ JNI Program Run/Exit Button
- ✓ HW1 Run Button
- ✓ HW2
  - ■ Module Load/Unload Button
  - ■ App Run Button
  - ■ Parameter Input
- ✓ HW3
  - ■ Module Load/Unload Button
  - ■ App Run Button

다. JNI Program

- ✓ Android Thread
- ✓ Passing Parameter Java to C

1) FPGA FND

요금 증가를 알리는 값이 표시된다. 정해진 방법에 의해 값이 감소하며 0이 되었을 때 요금이 증가한다. 초기값은 2200, 0이 되어 다시 반복될 때부터는 1400으로 초기화한다.

2) FPGA DOT

택시 속력의 정도를 표시한다. 1층부터 10층까지 순서대로 On/Off를 반복하며 속력이 빠를수록 반복 주기가 짧아진다.

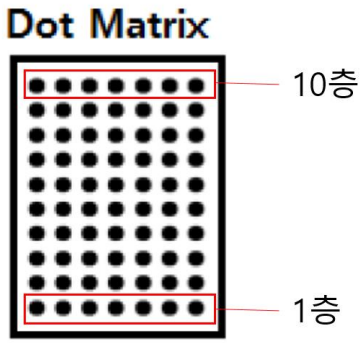

Figure 1 Dot Matrix

3) FPGA Text LCD

윗줄에는 택시 속력, 아랫줄에는 택시 요금이 표시된다.

4) FPGA Switch

버튼을 1개 눌렀을 시 가속, 2개 눌렀을 시 등속, 모두 떼었을 시 감속한다.

라. C Module / Application(changed)

1) HW1

✓ Volume Up : 모드 변경(Up)

✓ Volume Down : 프로그램 종료

✓ Back 버튼은 사용하지 않음

2) HW2

✓ Module 등록 시(insmod) 2번 LED On

3) HW3

- ✓ Module 등록 시(insmod) 3번 LED On
- ✓ Home Button : Start / Pause기능 동시에 수행(Toggle)
- ✓ Back Button : 짧게 눌렀을 시 타이머 초기화, 길게 눌렀을 시 종료

III. 추진 일정

2020. 06. 10. ~ 12.

- 프로젝트 기획
  - ✓ Kernel Image 수정
  - ✓ JNI를 사용한 프로그래밍

2020. 06. 13. ~ 16.

- Kernel Image 수정
  - ✓ GPIO Button
  - ✓ Custom Module 등록

2020. 06. 17. ~ 20.

- Execute Linux Command in Android Test & Experiment
  - ✓ Rooting

2020. 06. 21. ~ 22.

- JNI Program(택시 미터기)
  - ✓ FND, DOT, Text LCD, Switch

2020. 06. 23. ~ 24.

- Android Application
  - ✓ Graphic Interface
  - ✓ Load JNI & Execute Linux Command

2020. 06. 25. ~ 26.

- HW App/Module 수정

2020. 06. 27 ~ 28.

- Document 작성

IV. 개발 방법

가. Kernel Image

1) GPIO Button 수정

HW3 Module에서는 GPIO Button을 Interrupt Handler를 통해 사용하기 때문에 HW1 Application에서는 사용이 불가능하다. 따라서 HW1 Application에서는 Volume Up, Volume Down Button만을 사용하고, HW3 Module에서는 Home, Back Button만을 사용하도록 하고 이에 맞게 Kernel Image를 수정한다.

/work/achroimx_kernel/arch/arm/mach-mx6/board-achroimx.c

```
static struct gpio_keys_button ard_buttons[] = {
//      GPIO_BUTTON(SABREAUTO_ANDROID_HOME,
                    KEY_HOME,       1, "home",        0),
//      GPIO_BUTTON(SABREAUTO_ANDROID_BACK,
                    KEY_BACK,       1, "back",        0),
        GPIO_BUTTON(SABREAUTO_ANDROID_VOLUP,
                    KEY_VOLUMEUP,   1, "volume-up",   0),
        GPIO_BUTTON(SABREAUTO_ANDROID_VOLDOWN,
                    KEY_VOLUMEDOWN, 1, "volume-down", 0),
        GPIO_BUTTON(SABREAUTO_ANDROID_POWER,
                    KEY_POWER,      1, "power-key",   1),
};
```

Code 1 board-achrmimx.c

2) Custom Module 등록(Incomplete, Not used)

Android Application 내에서 Linux Command(*Runtime.getRuntime().exec()*)를 사용하여 HW C Application을 정상적으로 실행하기 위해서는, Application에서 사용하는 Module이 Load되어 있어야 한다. 따라서 Kernel Image를 수정하여 FPGA 부팅 시 자동으로 Load가 되게 하는 방법을 도모하였다.

achroimx_kernel/drivers/char/경로에 Module(fpga_led_driver.c) 을 추가한 뒤 Kconfig와 Makefile을 다음과 같이 수정한다.

/work/achroimx_kernel/drivers/char/Kconfig

```
config MY_MODULE
       tristate "My Module"
       default y
```

Code 2 achroimx_kernel/drivers/char/Kconfig

```
666   config MXS_VIIM
667       tristate "MXS Virtual IIM device driver"
668       depends on (ARCH_MX50 || ARCH_MX6)
669       help
670         Support for access to MXS Virtual IIM device, most people should say N here.
671
672   config MY_MODULE
673       tristate "My Module"
674       default y
```

Figure 2 Example achroimx_kernel/drivers/char/Makefile

/work/achroimx_kernel/drivers/char/Makefile

```
obj-$(CONFIG_MY_MODULE)                 += fpga_led_driver.o
```

Code 3 achroimx_kernel/drivers/char/Makefile

```
69    obj-$(CONFIG_JS_RTC)     += js-rtc.o
70    obj-$(CONFIG_DEVIR)    += ir.o
71
72    obj-$(CONFIG_MY_MODULE)    += fpga_led_driver.o
```

Figure 3 Example achroimx_kernel/drivers/char/Makefile

achroimx_kernel/ 에서 make menuconfig를 실행하여 My Module 등록을 확인한다.

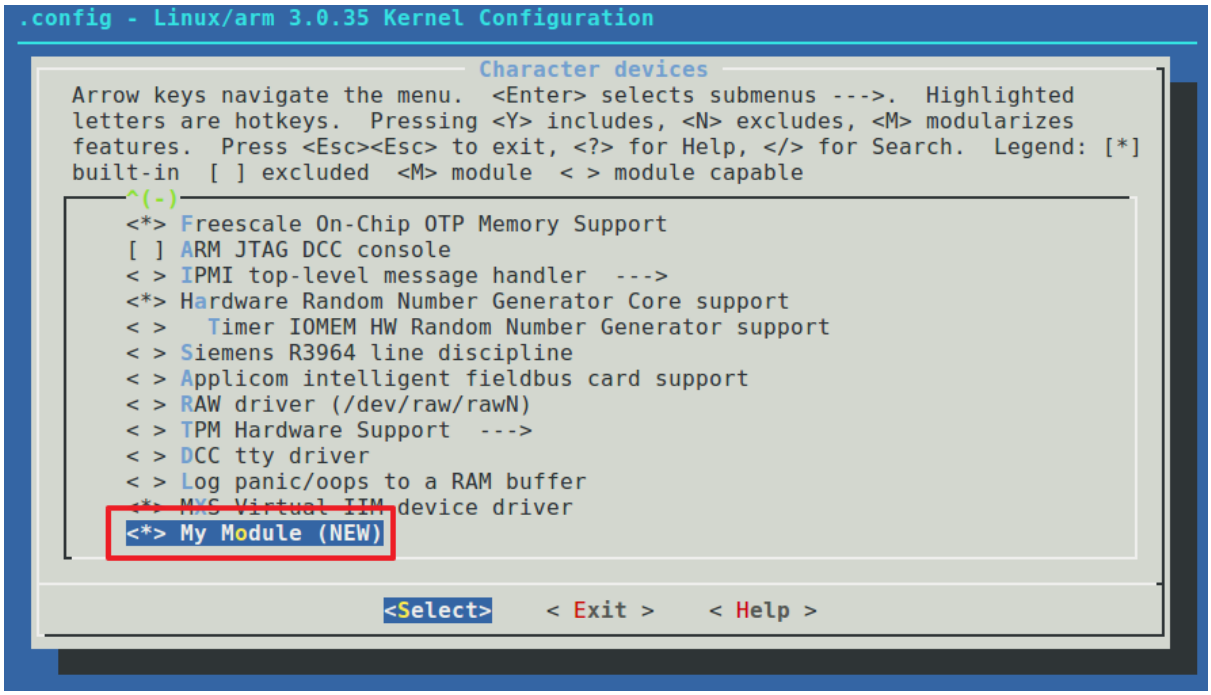

Figure 4 make menuconfig

Kernel Image를 make하면 위에서 등록한 Module이 Kernel에 추가됨을 확인할 수 있다.

```
root@embe-VirtualBox:/work/achroimx_kernel# make -j4
  CHK     include/linux/version.h
  CHK     include/generated/utsrelease.h
make[1]: 'include/generated/mach-types.h' is up to date.
  CALL    scripts/checksyscalls.sh
  CHK     include/generated/compile.h
  CC      drivers/char/fpga_led_driver.o
  LD      drivers/char/built-in.o
  LD      drivers/built-in.o
  LD      vmlinux.o
  MODPOST vmlinux.o
  GEN     .version
  CHK     include/generated/compile.h
  UPD     include/generated/compile.h
  CC      init/version.o
  LD      init/built-in.o
  LD      .tmp_vmlinux1
  KSYM    .tmp_kallsyms1.S
```

Figure 5 Kernel Image make

이 방법을 통하여 부팅 시 Custom Module 등록(insmod)은 구현하였으나 결정적으로 Device file까지 생성(mknod)하도록 Kernel Image를 완전하게 수정하지 못하여 이번 프로젝트 에서는 Module이 추가된 Kernel Image를 사용하지 않았다.

나. Android Application

1) Execute Linux Command

Android Application내에서 Module을 등록하고 C-Application을 실행하기 위해서는 Linux Command를 사용해야 했다.

제공되는 *Runtime.getRuntime().exec() Method*를 사용하여 Linux Command를 실행할 수 있었지만 Android가 Root권한을 얻지 못하여 Application내에서 insmod와 mknod Command를 사용할 수 없었다.

위 문제 해결을 위해 부팅 시 스크립트 실행을 통한 Module 등록, Android User ID에 Root권한 부여, insmod, mknod 사용 권한을 일반 User에게도 부여 등 여러가지 방법을 시도해 보았지만 일반 PC에서 사용되는 Linux와는 구조의 차이가 있어 구현하지 못하였다.

따라서 다른 Android Rooting Program을 사용하여 Android 자체에 Root권한을 부여하는 방법을 선택하였다. *Runtime.getRuntime().exec() Method*를 사용한 코드 예시는 다음과 같다.

```
Runtime.getRuntime().exec("su -c insmod
                           /data/local/driver/fpga_fnd_driver.ko");
Runtime.getRuntime().exec("su -c insmod
                           /data/local/driver/fpga_dot_driver.ko");
Runtime.getRuntime().exec("su -c insmod
                           /data/local/driver/fpga_text_lcd_driver.ko");
Runtime.getRuntime().exec("su -c insmod
                           /data/local/driver/fpga_push_switch_driver.ko");

Runtime.getRuntime().exec("su -c mknod /dev/fpga_fnd c 261 0");
Runtime.getRuntime().exec("su -c mknod /dev/fpga_dot c 262 0");
Runtime.getRuntime().exec("su -c mknod /dev/fpga_text_lcd c 263 0");
Runtime.getRuntime().exec("su -c mknod /dev/fpga_push_switch c 265 0");

Runtime.getRuntime().exec("su -c chmod 777 /dev/fpga_fnd");
Runtime.getRuntime().exec("su -c chmod 777 /dev/fpga_dot");
Runtime.getRuntime().exec("su -c chmod 777 /dev/fpga_text_lcd");
Runtime.getRuntime().exec("su -c chmod 777 /dev/fpga_push_switch");
```

Code 4 Example Runtime.getRuntime().exec()

2) Interface

Relativelayout, Button, EditText를 사용하여 다음과 같이 Application UI를 만들었다. 각 버튼은 OnClickListener를 사용하여 지정된 함수를 호출한다. 함수 예시와 Graphic Interface는 다음과 같다.

```
Button projload = (Button) findViewById(R.id.btprojload) ;
projload.setOnClickListener(new Button.OnClickListener() {
        @Override
        public void onClick(View view) {
                if(running){
                        mytoast.show();
                        return ;
                }
                running = true;

                thread_switch = new Thread_switch();
                thread_dot = new Thread_dot();

                thread_switch.start();
                thread_dot.start();
        }
});
```

Code 5 Example OnlickListener

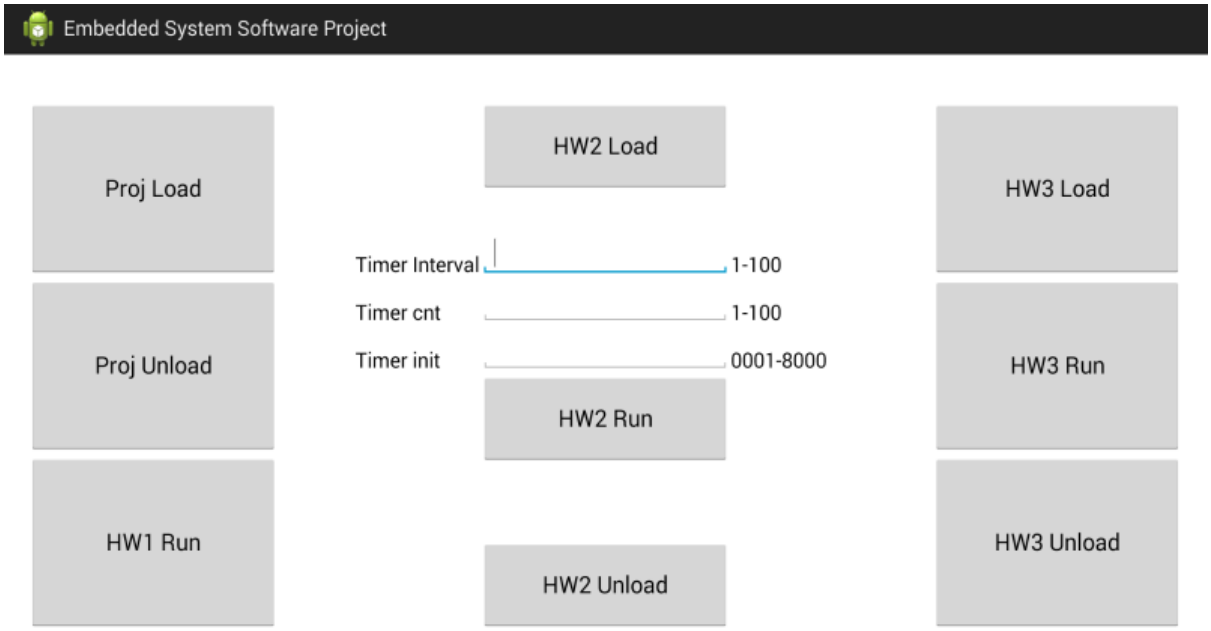

Figure 6 Android Application Interface

다. JNI Program

1) Thread

택시 미터기 구현 JNI JAVA에서 FPGA의 FND, DOT, Text LCD를 계속해서 업데이트 하기 위하여 새로운 Thread 2개를 생성하였다. Thread Interrupt로 인해 Thread가 종료될 때까지 해당 기능을 수행한다.

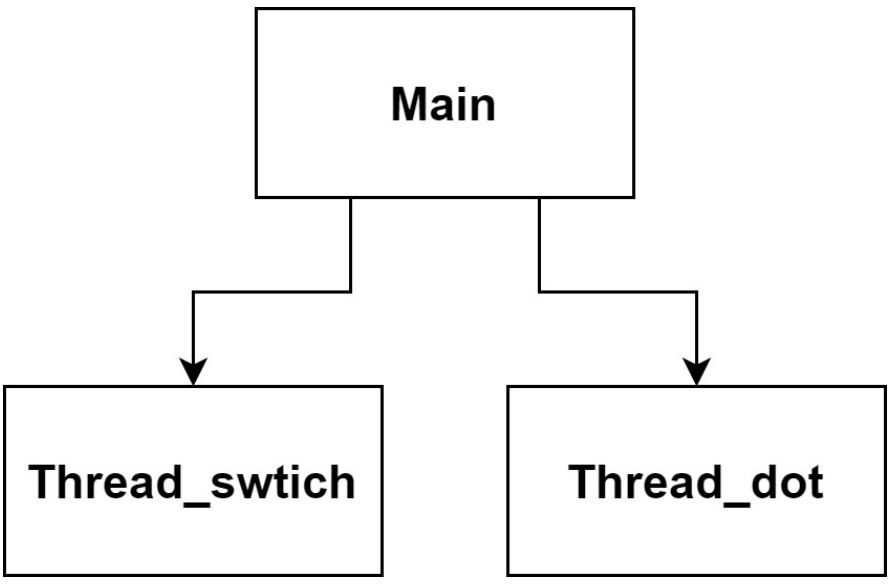

Figure 7 Thread

Thread_switch

controlswitch(JNI C Function)를 계속해서 호출하여 정수형 변수를 넘겨받아 count에 저장한다. count에 따라 속력(acc)를 증감하여 0일시 감속, 1일시 가속, 다른 경우 등속을 유지한다.

```
public class Thread_switch extends Thread {
	public void run(){
		int count;
		while(!Thread.currentThread().isInterrupted()){
			SystemClock.sleep(50);
			count = controlswitch();
			if(count == 0 && acc > 0) acc--;
			else if(count == 1 && acc < 200) acc++;
		}
	}
}
```

Code 6 Thread_switch

Thread_dot

controlfnd, controltextlcd(JNI C Function)를 계속해서 호출한다. Thread_switch에 의해 결정되는 속력(acc)를 사용하여 FPGA FND, Text LCD를 동작한다.

```
public class Thread_dot extends Thread {
	public void run(){
		int count = 0;
		int sign = 1;
		int fnd_data = 2200;
		int amount = 3800;

		while(!Thread.currentThread().isInterrupted()){
			SystemClock.sleep(205-acc);
			count += sign;
			controldotfnd(count, fnd_data);
			if(count == 0 || count == 10) sign *= -1;
			fnd_data -= 17;
			if(fnd_data < 0) {
				fnd_data = 1400;
				amount += 100;
			}
			controltextlcd(String.valueOf(acc),
					String.valueOf(amount));
		}
	}
}
```

Code 7 Thread_dot

2) JNI C Function

각 Thread에서 기능에 맞게 Native Function을 호출한다.

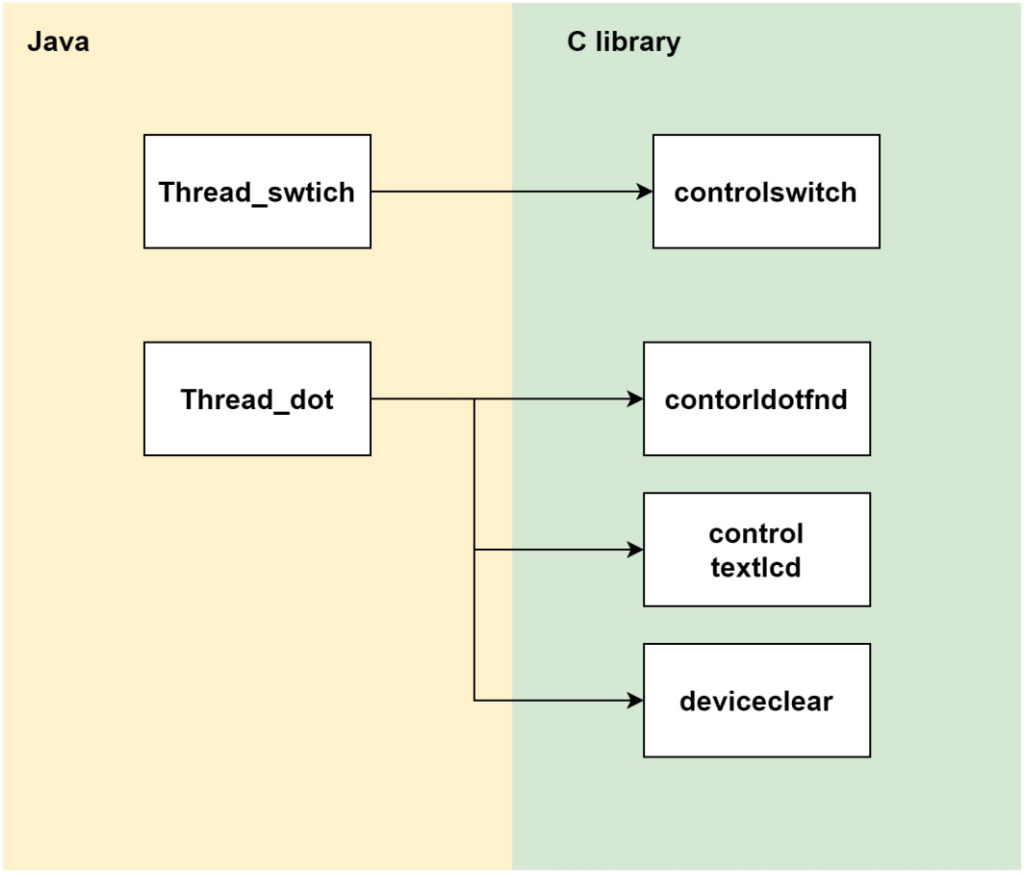

Figure 8 JNI

controlswitch

FPGA Switch를 입력 받는다. Switch 번호의 상관 없이 눌린 개수를 카운트 하여 리턴한다.

```
jint JNICALL
Java_com_embe2020_project_MainActivity_controlswitch
(JNIEnv *env, jobject this){
	int i;
	int dev_switch;
	int buff_size;
	int count;
	int acc;
	unsigned char push_sw_buff[MAX_BUTTON];

	dev_switch = open("/dev/fpga_push_switch", O_RDWR);
	buff_size=sizeof(push_sw_buff);

	read(dev_switch, &push_sw_buff, buff_size);
	count = 0;
```

```
        for(i=0;i<MAX_BUTTON;i++) {
                if(push_sw_buff[i] == 1) count++;
        }

        close(dev_switch);
        return count;
};
```

Code 8 controlswitch

controldotfnd

FPGA DOT, FND에 출력할 값을 넘겨받아 출력한다.

```
void JNICALL
Java_com_embe2020_project_MainActivity_controldotfnd
(JNIEnv *env, jobject this, jint set_num, jint fnd_data){
        int str_size;
        int dev_dot;
        int dev_fnd;
        unsigned char data[4];

        dev_dot = open("/dev/fpga_dot", O_WRONLY);
        dev_fnd = open("/dev/fpga_fnd", O_WRONLY);

        data[3] = fnd_data%10; fnd_data /= 10;
        data[2] = fnd_data%10; fnd_data /= 10;
        data[1] = fnd_data%10; fnd_data /= 10;
        data[0] = fnd_data%10;
        write(dev_fnd, &data, 4);

        str_size = sizeof(fpga_number[set_num]);
        write(dev_dot, fpga_number[set_num], str_size);

        close(dev_dot);
        close(dev_fnd);
};
```

Code 9 controldotfnd

controltextlcd

FPGA Text LCD에 출력할 값을 넘겨받아 출력한다. JNI의 Passing String을 사용한다.

```
void JNICALL
JNIEXPORT void JNICALL
Java_com_embe2020_project_MainActivity_controltextlcd
(JNIEnv *env, jobject this, jstring speed_path, jstring amount_path){
        int dev_textlcd;
        int str_size;
        const char *speed;
        const char *amount;
        unsigned char string[32];
```

```
        dev_textlcd = open("/dev/fpga_text_lcd", O_WRONLY);
        memset(string, 0, sizeof(string));

        jboolean isCopy_s, isCopy_a;
        speed = (*env)->GetStringUTFChars(env, speed_path, &isCopy_s);
        amount = (*env)->GetStringUTFChars(env, amount_path, &isCopy_a);

        strncat(string, "Speed  : ", 9);
        str_size=strlen(speed);
        strncat(string, speed, str_size);
        memset(string+9+str_size, ' ', 16-9-str_size);

        strncat(string, "Amount : ", 9);
        str_size=strlen(amount);
        strncat(string, amount, str_size);
        memset(string+16+9+str_size, ' ', 16-9-str_size);

        write(dev_textlcd, string, 32);

        if (isCopy_s == JNI_TRUE) {
                (*env)->ReleaseStringUTFChars(env, speed_path, speed);
        }
        if (isCopy_a == JNI_TRUE) {
                (*env)->ReleaseStringUTFChars(env, amount_path, amount);
        }

        close(dev_textlcd);
};
```

Code 10 controltextlcd

deviceclear

사용 중이었던 FPGA의 모든 Device를 초기화한다.

```
void JNICALL
Java_com_embe2020_project_MainActivity_deviceclear(
JNIEnv *env, jobject this){
        int dev_dot;
        int dev_fnd;
        int dev_textlcd;
        int str_size;

        unsigned char string[32];
        unsigned char data[4];

        dev_dot = open("/dev/fpga_dot", O_WRONLY);
        dev_fnd = open("/dev/fpga_fnd", O_WRONLY);
        dev_textlcd = open("/dev/fpga_text_lcd", O_WRONLY);

        data[0] = data[1] = data[2] = data[3] = 0;
        write(dev_fnd,&data,4);

        str_size = sizeof(fpga_number[0]);
        write(dev_dot, fpga_number[0], str_size);
        memset(string, ' ', sizeof(string));
        write(dev_textlcd, string, 32);
```

```
        close(dev_dot);
        close(dev_fnd);
        close(dev_textlcd);
};
```

Code 11 deviceclear

3) C Application / Module(HW)

✓ 프로젝트 수행에 맞게 수정, 기능상의 변화는 없음

V. 연구 결과

- JNI를 사용한 프로그램과 디바이스 모듈을 사용한 프로그램을 외부와 연결을 통한 입력 없이 자체적으로 실행 할 수 있는 하나의 안드로이드 어플리케이션을 구현

- 프로그램 흐름도

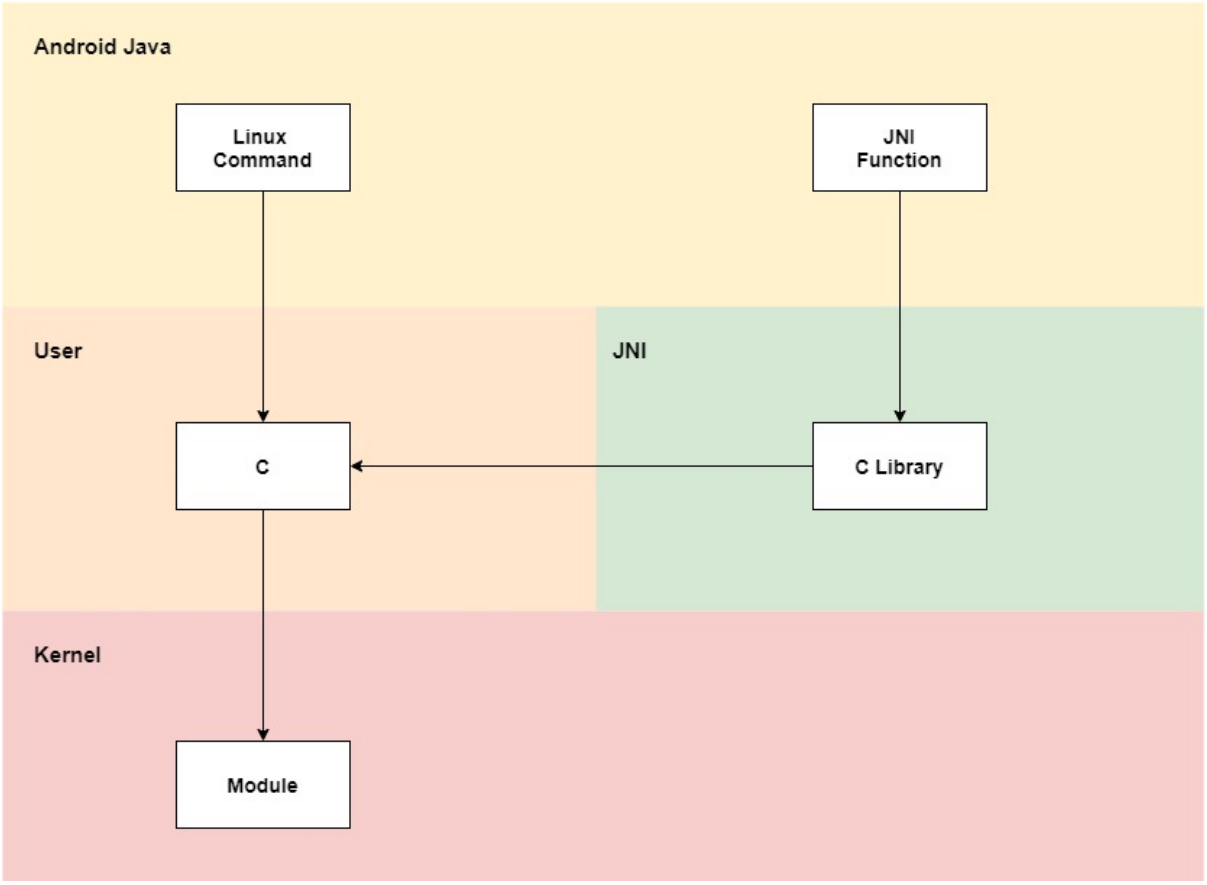

Figure 9 FlowChart

VI. 기타

처음 프로젝트의 기획대로 Kernel Image를 완전하게 수정하지 못해 아쉬움이 남는다. 또한 초기에는 택시 미터기를 만들며 Server, Database, Get Request등을 사용하여 손님의 택시 호출 요청/응답 기능을 구현하고 싶었지만 Achro iMX 보드에서 Wifi 기능이 작동하지 않아 Android Application에서 인터넷을 사용하지 못한 아쉬움과 보드 Spec상에는 Wifi기능이 탑재되어 있다고 나와있는데 Android 내에서 작동하지 않는 의문이 들었다.

JNI 프로그래밍을 하며 동일한 기능을 수행하는 코드에서 Java, JNI, C에서 수행 시간의 차이를 비교해보고 싶은 생각도 들었다.

I. Appendix

MainActiviry.java

```
import java.io.IOException;
import android.support.v7.app.ActionBarActivity;
import android.annotation.SuppressLint;
import android.os.Bundle;
import android.os.SystemClock;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

@SuppressLint("ShowToast")
public class MainActivity extends ActionBarActivity {

	public native void ledwrite(int dev, int value);
	public native int controlswitch();
	public native void controldotfnd(int set_num, int fnd_data);
	public native void deviceclear();
	public native void controltextlcd(String speed, String amount);

	int acc = 0;
	Thread_switch thread_switch;
	Thread_dot thread_dot;
	boolean running=false;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		Log.d("MainActiviry", "Start");

		try {
			Runtime.getRuntime().exec("su -c insmod
```

```
                 /data/local/driver/fpga_fnd_driver.ko");
                Runtime.getRuntime().exec("su -c insmod
                 /data/local/driver/fpga_dot_driver.ko");
                Runtime.getRuntime().exec("su -c insmod
                 /data/local/driver/fpga_text_lcd_driver.ko");
                Runtime.getRuntime().exec("su -c insmod
                 /data/local/driver/fpga_push_switch_driver.ko");

                Runtime.getRuntime().exec("su -c mknod
                /dev/fpga_fnd c 261 0");
                Runtime.getRuntime().exec("su -c mknod
                /dev/fpga_dot c 262 0");
                Runtime.getRuntime().exec("su -c mknod
                /dev/fpga_text_lcd c 263 0");
                Runtime.getRuntime().exec("su -c mknod
                 /dev/fpga_push_switch c 265 0");

                Runtime.getRuntime().exec("su -c chmod 777 /dev/fpga_fnd");
                Runtime.getRuntime().exec("su -c chmod 777 /dev/fpga_dot");
                Runtime.getRuntime().exec("su -c chmod 777
                 /dev/fpga_text_lcd");
                Runtime.getRuntime().exec("su -c chmod 777
                 /dev/fpga_push_switch");
            } catch (IOException e) {
                   e.printStackTrace();
            }

            setContentView(R.layout.activity_main);
            System.loadLibrary("project");

            findViewById(R.id.layout).requestFocus();
            findViewById(R.id.et1).clearFocus();
            findViewById(R.id.et2).clearFocus();
            findViewById(R.id.et3).clearFocus();

            final Toast mytoast = Toast.makeText(this.getApplicationContext(),
                    "Access Deny", Toast.LENGTH_SHORT);

            Button projload = (Button) findViewById(R.id.btprojload) ;
            projload.setOnClickListener(new Button.OnClickListener() {
            @Override
            public void onClick(View view) {
                   if(running){
                          mytoast.show();
                          return ;
                   }
                   running = true;

                   thread_switch = new Thread_switch();
                   thread_dot = new Thread_dot();

                   thread_switch.start();
                   thread_dot.start();
            }
        });

            Button projunload = (Button) findViewById(R.id.btprojunload) ;
```

```
    projunload.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
           if(!running){
                  mytoast.show();
                  return ;
           }
           running = false;

           thread_switch.interrupt();
           thread_dot.interrupt();

           thread_switch = null;
           thread_dot = null;

           SystemClock.sleep(250);
           deviceclear();
    }
});

    Button bthw1run = (Button) findViewById(R.id.bthw1run) ;
    bthw1run.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
           try {
                  Runtime.getRuntime().exec("su -c
                         ./data/local/proj/hw1/app");

           } catch (IOException e) {

                  e.printStackTrace();
           }
    }
});

Button bthw2load = (Button) findViewById(R.id.bthw2load) ;
bthw2load.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
           try {
                  Runtime.getRuntime().exec("su -c insmod
                   /data/local/proj/hw2/dev_module.ko");
                  Runtime.getRuntime().exec("su -c mknod
                   /dev/dev_driver c 242 0");
                  Runtime.getRuntime().exec("su -c chmod 777
                   /dev/dev_driver");

           } catch (IOException e) {
                  e.printStackTrace();
           }
    }
});

Button bthw2run = (Button) findViewById(R.id.bthw2run) ;
bthw2run.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
```

```
                EditText et1 = (EditText)findViewById(R.id.et1);
                EditText et2 = (EditText)findViewById(R.id.et2);
                EditText et3 = (EditText)findViewById(R.id.et3);
                String p1 = et1.getText().toString();
                String p2 = et2.getText().toString();
                String p3 = et3.getText().toString();
                try {
                        Runtime.getRuntime().exec("su -c
                ./data/local/proj/hw2/app " + p1 + " " + p2 + " " + p3);
                        } catch (IOException e) {
                        e.printStackTrace();
                }
        }
});

Button hw2unload = (Button) findViewById(R.id.bthw2unload) ;
hw2unload.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
            try {
                    Runtime.getRuntime().exec("su -c rmmod
                     /dev_module");
                    Runtime.getRuntime().exec("su -c rm
                     /dev/dev_driver");
            } catch (IOException e) {
                    e.printStackTrace();
            }
    }
});

Button bthw3load = (Button) findViewById(R.id.bthw3load) ;
bthw3load.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
            try {
                    Runtime.getRuntime().exec("su -c insmod
                     /data/local/proj/hw3/dev_module.ko");
                    Runtime.getRuntime().exec("su -c mknod
                     /dev/stopwatch c 242 0");
                    Runtime.getRuntime().exec("su -c chmod 777
                     /dev/stopwatch");
            } catch (IOException e) {
                    e.printStackTrace();
            }
    }
});

Button bthw3run = (Button) findViewById(R.id.bthw3run) ;
bthw3run.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View view) {
            try {
                    Runtime.getRuntime().exec("su -c
                    ./data/local/proj/hw3/app");
            } catch (IOException e) {
                    e.printStackTrace();
            }
```

```
        }
    });
    Button hw3unload = (Button) findViewById(R.id.bthw3unload) ;
    hw3unload.setOnClickListener(new Button.OnClickListener() {
        @Override
        public void onClick(View view) {
               try {
                      Runtime.getRuntime().exec("su -c rmmod
                       /dev_module");
                      Runtime.getRuntime().exec("su -c rm
                       /dev/stopwatch");

               } catch (IOException e) {
                      e.printStackTrace();
               }
        }
    });
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
       getMenuInflater().inflate(R.menu.main, menu);
       return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
       int id = item.getItemId();
       if (id == R.id.action_settings) {
              return true;
       }
       return super.onOptionsItemSelected(item);
}

@Override
public void onBackPressed(){
       super.onBackPressed();
       this.finish();
}

@Override
public void onDestroy(){
       super.onDestroy();
       System.out.println("Termination");
}

@Override
public void onStop(){
       super.onStop();
       System.out.println("Stop");
}

public class Thread_switch extends Thread {
       public void run(){
              int count;
              while(!Thread.currentThread().isInterrupted()){
                     SystemClock.sleep(50);
```

```
                        count = controlswitch();
                        if(count == 0 && acc > 0) acc--;
                        else if(count == 1 && acc < 200) acc++;
                }
        }
}

public class Thread_dot extends Thread {
        public void run(){
                int count = 0;
                int sign = 1;
                int fnd_data = 2200;
                int amount = 3800;

                while(!Thread.currentThread().isInterrupted()){
                        SystemClock.sleep(205-acc);
                        count += sign;
                        controldotfnd(count, fnd_data);
                        if(count == 0 || count == 10) sign *= -1;
                        fnd_data -= 17;
                        if(fnd_data < 0) {
                                fnd_data = 1400;
                                amount += 100;
                        }
                        controltextlcd(String.valueOf(acc),
                                String.valueOf(amount));
                }
        }
}
}
```

Code 12 MainActiviry.java

Proj.c(JNI)

```
#include <jni.h>
#include <fcntl.h>
#include "android/log.h"
#include "fpga_dot_font.h"

#define LOG_TAG "MyTag"
#define LOGV(...)   __android_log_print(ANDROID_LOG_VERBOSE,
                                        LOG_TAG, __VA_ARGS__)
#define MAX_BUTTON 9

jint JNICALL
Java_com_embe2020_project_MainActivity_controlswitch
(JNIEnv *env, jobject this){
        int i;
        int dev_switch;
        int buff_size;
        int count;
        int acc;
        unsigned char push_sw_buff[MAX_BUTTON];
```

```
        dev_switch = open("/dev/fpga_push_switch", O_RDWR);
        buff_size=sizeof(push_sw_buff);

        read(dev_switch, &push_sw_buff, buff_size);
        count = 0;
        for(i=0;i<MAX_BUTTON;i++) {
                if(push_sw_buff[i] == 1) count++;
        }

        close(dev_switch);
        return count;
};

void JNICALL
Java_com_embe2020_project_MainActivity_deviceclear
(JNIEnv *env, jobject this){
        int dev_dot;
        int dev_fnd;
        int dev_textlcd;
        int str_size;

        unsigned char string[32];
        unsigned char data[4];

        dev_dot = open("/dev/fpga_dot", O_WRONLY);
        dev_fnd = open("/dev/fpga_fnd", O_WRONLY);
        dev_textlcd = open("/dev/fpga_text_lcd", O_WRONLY);

        data[0] = data[1] = data[2] = data[3] = 0;
        write(dev_fnd,&data,4);

        str_size = sizeof(fpga_number[0]);
        write(dev_dot, fpga_number[0], str_size);

        memset(string, ' ', sizeof(string));
        write(dev_textlcd, string, 32);

        close(dev_dot);
        close(dev_fnd);
        close(dev_textlcd);
};

void JNICALL
Java_com_embe2020_project_MainActivity_controldotfnd
(JNIEnv *env, jobject this, jint set_num, jint fnd_data){
        int str_size;
        int dev_dot;
        int dev_fnd;
        unsigned char data[4];

        dev_dot = open("/dev/fpga_dot", O_WRONLY);
        dev_fnd = open("/dev/fpga_fnd", O_WRONLY);

        data[3] = fnd_data%10; fnd_data /= 10;
        data[2] = fnd_data%10; fnd_data /= 10;
        data[1] = fnd_data%10; fnd_data /= 10;
        data[0] = fnd_data%10;
```

```
        write(dev_fnd, &data, 4);

        str_size = sizeof(fpga_number[set_num]);
        write(dev_dot, fpga_number[set_num], str_size);

        close(dev_dot);
        close(dev_fnd);
};

JNIEXPORT void JNICALL
 Java_com_embe2020_project_MainActivity_controltextlcd
(JNIEnv *env, jobject this, jstring speed_path, jstring amount_path){
        int dev_textlcd;
        int str_size;
        const char *speed;
        const char *amount;
        unsigned char string[32];

        dev_textlcd = open("/dev/fpga_text_lcd", O_WRONLY);
        memset(string, 0, sizeof(string));

        jboolean isCopy_s, isCopy_a;
        speed = (*env)->GetStringUTFChars(env, speed_path, &isCopy_s);
        amount = (*env)->GetStringUTFChars(env, amount_path, &isCopy_a);

        strncat(string, "Speed  : ", 9);
        str_size=strlen(speed);
        strncat(string, speed, str_size);
        memset(string+9+str_size, ' ', 16-9-str_size);

        strncat(string, "Amount : ", 9);
        str_size=strlen(amount);
        strncat(string, amount, str_size);
        memset(string+16+9+str_size, ' ', 16-9-str_size);

        write(dev_textlcd, string, 32);

        if (isCopy_s == JNI_TRUE) {
                (*env)->ReleaseStringUTFChars(env, speed_path, speed);
        }
        if (isCopy_a == JNI_TRUE) {
                (*env)->ReleaseStringUTFChars(env, amount_path, amount);
        }

        close(dev_textlcd);
};
```

Code 13 Proj.c(JNI)