임베디드 시스템 프로젝트

GIThub용

학번 이름 x

목차

- AUDIO Player.

AUDIO - 구현 목표

- **DE1-SoC** 보드는 마이크에서

소리를 샘플링 하여 회로에 입력으로 제공할 수 있는 오디오 코덱이 장착 됨.

기본적인 코덱으로 가청음을 정확하게 표현하기 충분한 **48000**개 **(48kHZ)**의 샘플을 제공.

주의: 왼쪽 및 오른쪽 **FIFO**에 모두 써야 함. **FIFO**가 비어 있으면 오디오 코덱으로 전송되지 않음.

- 1. **DE1-SoC**에서 “음”(소리) 내는게 목표
- **2.** 각 스위치마다 **ON**버튼을 누르면

도(C) 레(D) 미(E) 파(F) 솔(G) 라(A) 시(B) 표현 및 음을 낼 수 있게 구현

- **3.**이를 바탕으로 도레미송을 연주

AUDIO – SW.C

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/init.h>
#include <linux/interrupt.h>
#include <asm/io.h>

#include <linux/fs.h>
#include <linux/uaccess.h>
#include <linux/types.h>
#include <linux/ioport.h>

MODULE_LICENSE("GPL");

#define base_lwFPGA 0xFF200000
#define len_lwFPGA 0x00200000

#define addr_SW 0x40

#define SW_DEVMAJOR 238
#define SW_DEVNAME  "sw"

static int sw_open(struct inode* minode, struct file* mfile);
static int sw_release(struct inode* minode, struct file* mfile);
"sw.c" 114L, 2496C                                    1,1           Top
```

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
static int sw_write(struct inode* minode, struct file* mfile);
static ssize_t sw_read(struct file* file, char __user* buf, size_t count, loff_t
* f_ops);
static int __init sw_init(void);
static void __exit sw_exit(void);

static struct file_operations sw_fops;
static void *mem_base;
static void *sw_addr;

// open operation
static int sw_open(struct inode *minode, struct file *mfile)
{
    // do nothing
    return 0;
}

// release operation
static int sw_release(struct inode *minode, struct file *mfile)
{
    // do nothing
    return 0;
}
                                                      45,1          25%
```

AUDIO - SW.C

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
// write operation
static int sw_write(struct inode *minode, struct file *mfile)
{
    // do nothing
    return 0;
}

// read operation
// read data from sw and store it in the buffer
static ssize_t sw_read(struct file *file, char __user *buf, size_t count, loff_t
 *f_ops)
{
    unsigned int sw_data = 0;
    int ret;

    sw_data = ioread32(sw_addr);

    ret = copy_to_user(buf, &sw_data, count);
    if (ret) {
        return -EFAULT;
    }

    return sizeof(sw_data);
                                                    68,1          50%
```

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
    ret = copy_to_user(buf, &sw_data, count);
    if (ret) {
        return -EFAULT;
    }

    return sizeof(sw_data);
}

// file operation initialize
static struct file_operations sw_fops = {
    .read = sw_read,
    .write = sw_write,
    .open = sw_open,
    .release = sw_release
};

// driver init
static int __init sw_init(void)
{
    int res;

    res = register_chrdev(SW_DEVMAJOR, SW_DEVNAME, &sw_fops);

                                                    85,0-1        68%
```

AUDIO - SW.C

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
    int res;

    res = register_chrdev(SW_DEVMAJOR, SW_DEVNAME, &sw_fops);

    if (res < 0) {
        printk(KERN_ERR "sw: failed to register device.\n");
        return res;
    }

    // mapping address
    mem_base = ioremap_nocache(base_lwFPGA, len_lwFPGA);

    if (!mem_base) {
        printk("Error mapping memory.\n");
        release_mem_region(base_lwFPGA, len_lwFPGA);
        return -EBUSY;
    }

    sw_addr = mem_base + addr_SW;

    printk("Device: %s Major: %d\n", SW_DEVNAME, SW_DEVMAJOR);
    return 0;
}
                                                    94,1          89%
```

```
static void __exit sw_exit(void)
{
    unregister_chrdev(SW_DEVMAJOR, SW_DEVNAME);
    printk("%s unregistered.\n", SW_DEVNAME);
    iounmap(mem_base); // virtual mem release
}

module_init(sw_init);
module_exit(sw_exit);
```

AUDIO- SW.C

SW.C를 만든 이유

- sw.c를 통해 스위치의 입력을 읽어오도록 구현

-> 사용자공간에서 스위치를 읽을 수 있게 함

- INIT 함수

Register_chrdev함수로 디바이스파일 등록

Ioremap_nocache함수로 메모리를 매핑

- EXIT함수

Unreigster_chrdev함수로 디바이스파일 해체

Iounmap 함수로 매핑된 메모리를 해체

AUDIO – HEX.C

COM8 - Tera Term VT

File Edit Setup Control Window Help

```c
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/init.h>
#include <linux/interrupt.h>
#include <asm/io.h>

#include <linux/fs.h>
#include <linux/uaccess.h>
#include <linux/types.h>
#include <linux/ioport.h>

MODULE_LICENSE("GPL");

#define base_lwFPGA 0xFF200000
#define len_lwFPGA 0x00200000

#define addr_HEX3HEX0   0x20
#define addr_HEX5HEX4   0x30

#define HEX_DEVMAJOR    240
#define HEX_DEVNAME     "hex"

static void *mem_base;
```

"hex.c" 110L, 2589C 1,1 Top

COM8 - Tera Term VT

File Edit Setup Control Window Help

```c
static void *hex3hex0_addr;
static void *hex5hex4_addr;

int hex_conversions[8] = {
    0x39, 0x5E, 0x79, 0x71, 0x7D, 0x77, 0x7C, 0x00  //C D E F G A B dummy
};

// Open operation
static int hex_open(struct inode *minode, struct file *mfile) {
    // Do nothing
    return 0;
}

// Release operation
static int hex_release(struct inode *minode, struct file *mfile) {
    // Do nothing
    return 0;
}

// Write operation
static ssize_t hex_write(struct file *file, const char __user *buf, size_t count
, loff_t *f_pos) {
    unsigned int hex_data = 0;  // 32-bit data
```

45,1 26%

AUDIO – HEX.C

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
    unsigned int hex_data = 0;  // 32-bit data
    int ret;
    unsigned int hex3hex0_data;

    // Get data from user buffer
    ret = copy_from_user(&hex_data, buf, sizeof(hex_data));
    if (ret) {
        return -EFAULT;  // Return an appropriate error code on copy error
    }

    hex3hex0_data = hex_conversions[hex_data];

    // Write the converted data to hex3hex0
    iowrite32(hex3hex0_data, hex3hex0_addr);

    return count;
}

// Read operation
static ssize_t hex_read(struct file *file, char __user *buf, size_t count, loff_
t *f_pos) {
    // Do nothing
    return 4;
                                                    65,1          50%
```

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
// File operations initialize
static struct file_operations hex_fops = {
    .read = hex_read,
    .write = hex_write,
    .open = hex_open,
    .release = hex_release
};

// Driver initialization
static int __init hex_init(void) {
    int res;

    res = register_chrdev(HEX_DEVMAJOR, HEX_DEVNAME, &hex_fops);
    if (res < 0) {
        printk(KERN_ERR "hex: failed to register device.\n");
        return res;
    }

    // Map the address
    mem_base = ioremap_nocache(base_lwFPGA, len_lwFPGA);

    if (!mem_base) {
        printk("Error mapping memory!\n");
                                                    91,1          78%
```

AUDIO - HEX.C

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
    mem_base = ioremap_nocache(base_lwFPGA, len_lwFPGA);

    if (!mem_base) {
        printk("Error mapping memory!\n");
        release_mem_region(base_lwFPGA, len_lwFPGA);
        return -EBUSY;
    }

    hex3hex0_addr = mem_base + addr_HEX3HEX0;
    hex5hex4_addr = mem_base + addr_HEX5HEX4;

    printk("Device: %s, MAJOR: %d\n", HEX_DEVNAME, HEX_DEVMAJOR);
    return 0;
}

static void __exit hex_exit(void) {
    unregister_chrdev(HEX_DEVMAJOR, HEX_DEVNAME);
    printk("%s unregistered.\n", HEX_DEVNAME);
    iounmap(mem_base);
}

module_init(hex_init);
module_exit(hex_exit);
                                                    110,1         Bot
```

AUDIO- HEX.C

hex.c만든 이유

- 7-세그먼트 디스플레이에 값을 출력하려는 기능을 구현하려고 만듬

Hex_conversions 배열을 0~7까지 설정하여 각 인덱스에 해당하는 값을 출력

- INIT함수

Regster_chrdev함수로 미리 정의한 문자 디바이스 등록

Ioremao_nocache함수를 사용하여 base_lwFPGA, len_lwFPGA의 값으로 메모리 맵 생성

-> 주소 공간을 커널의 가상주소 공간으로 매핑 할 수 있음

Mem_base로 addr_hex3hex0, addr_hex5hex5에 정의된 값을 더하여 hex3hex0레지스터와 Hex5hex4레지스터에 접근할 수 있는 가상 주소를 얻을수 있음

- EXIT함수

Unregister_chrdev로 등록된 문자 디바이스 해체

Iounmap함수로 mem_base에 매핑된 가상주소를 해체

AUDIO - audioplay.c

COM8 - Tera Term VT

File Edit Setup Control Window Help

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

#include <sys/types.h>
#include <sys/ioctl.h>
#include <sys/stat.h>
#include <sys/mman.h>

#include <math.h>

#define HW_REGS_BASE 0xFF200000
#define HW_REGS_SPAN 0x00200000
#define HW_REGS_MASK HW_REGS_SPAN-1

#define AUDIO_PIO_BASE 0x3040

#define PI  3.141592
#define SAMPLE_RATE 48000

void write_to_audio (double freq, volatile unsigned int* h2p_lw_audio_addr);
void musicplay (volatile unsigned int* h2p_lw_audio_addr);
```

"audioplayer.c" 341L, 10461C 1,1 Top

COM8 - Tera Term VT

File Edit Setup Control Window Help

```c
double tone[8] = {
    261.626, //도 0 C
    293.655, //레 1 D
    329.628, //미 2 E
    349.228, //파 3 F
    391.992, //솔 4 G
    440.000, //라 5 A
    493.883, //시 6 B
    523.251   //도 7 C
};

int hex_display;
int dev_hex;

int main(){
    int dev_sw, data, rdata;

    //sw file open
    dev_sw = open("/dev/sw", O_RDWR);
    if(dev_sw<0) { //if open failed
        fprintf(stderr, "Cannot open SW device.\n");
        return 1;
    }
```

47,1 7%

AUDIO – audioplay.c

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
//hex file open
dev_hex = open("/dev/hex", O_RDWR);
if(dev_hex<0){ //if open failed
    fprintf(stderr, "cannot open HEX device.\n");
    return 2;
}

//audio part initialize
volatile unsigned int* h2p_lw_audio_addr = NULL;

void * virtual_base;
int fd;

//open /dev/mem
if((fd=open("/dev/mem", (O_RDWR|O_SYNC)))==-1){
    printf(" ERROR : could not open /dev/mem.. \n");
    return 1;
}

//get virtual addr that maps to physical
virtual_base = mmap(
    NULL,
                                                    70,1          14%
```

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
        HW_REGS_SPAN,
        (PROT_READ | PROT_WRITE),
        MAP_SHARED,
        fd,
        HW_REGS_BASE
    );

    if(virtual_base == MAP_FAILED){
        printf(" ERROR : mmap() failed... \n");
        close(fd);
        return 1;
    }

    //get virtual addr of AUDIO
    //h2p_lw_audio_addr = (unsigned int*)(virtual_base+(AUDIO_PIO_BASE) & (unsig
ned int*)(HW_REGS_MASK));
    h2p_lw_audio_addr = (unsigned int*)((unsigned int)virtual_base + (AUDIO_PIO_
BASE & HW_REGS_MASK));
    //first, clear the WRITE FIFO by using the CW bit in Control Reg
    *(h2p_lw_audio_addr) = 0xC;     //CW=1, CR=1, WE=0, RE=0
    //and need to reset to 0 to use
    *(h2p_lw_audio_addr) = 0x0;
                                                    91,0-1        21%
```

AUDIO - audioplay.c

COM8 - Tera Term VT

File Edit Setup Control Window Help

```c
double freq;

//execute part
while(1){
    read(dev_sw, &rdata, 4);

    switch(rdata){
        case 0x01:
            freq = tone[0];
            hex_display = 0;
            break;
        case 0x02:
            freq = tone[1];
            hex_display = 1;
            break;
        case 0x04:
            freq = tone[2];
            hex_display = 2;
            break;
        case 0x08:
            freq = tone[3];
            hex_display = 3;
            break;
```

114,1 28%

COM8 - Tera Term VT

File Edit Setup Control Window Help

```c
        case 0x10:
            freq = tone[4];
            hex_display = 4;
            break;
        case 0x20:
            freq = tone[5];
            hex_display = 5;
            break;
        case 0x40:
            freq = tone[6];
            hex_display = 6;
            break;
        case 0x80:
            freq = tone[7];
            hex_display = 7;
            break;

        case 0x200:         //music func output
            musicplay(h2p_lw_audio_addr);
            hex_display = 7;
            sleep(1);
            break;
```

136,1 35%

AUDIO - audioplay.c

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
            default:
                hex_display = 7;
                write(dev_hex, &hex_display, 4);
        }
        if(hex_display != 7){
            write_to_audio(freq, h2p_lw_audio_addr);
            sleep(1);
        }
    }//while(1)

    //close
    close(dev_sw);
    return 0;
}

//주파수, audio port의 주소를 받아 출력하는 함수
void write_to_audio(double freq,volatile unsigned int* h2p_lw_audio_addr){
    usleep(10000);
    //write data in LeftData & RightData
    int nth_sample;

    write(dev_hex, &hex_display, 4);
```

160,0-1 43%

COM8 - Tera Term VT

File Edit Setup Control Window Help

```
    //Max volume when multiplied by sin() which ranges from -1 to 1
    int vol = 0x3FFFFFFF;

    for(nth_sample = 0; nth_sample<SAMPLE_RATE*5; nth_sample++){
        *(h2p_lw_audio_addr+2) = vol * sin(nth_sample*freq*2*PI / (SAMPLE_RATE*5
));
        *(h2p_lw_audio_addr+3) = vol * sin(nth_sample*freq*2*PI/(SAMPLE_RATE*5))
;
    }
}
//Sound of Music 도 레 미 송
void musicplay(volatile unsigned int* h2p_lw_audio_addr) {
    write_to_audio(tone[0], h2p_lw_audio_addr); // 도 C
    usleep(150000);
    write_to_audio(tone[1], h2p_lw_audio_addr); // 레 D
    write_to_audio(tone[2], h2p_lw_audio_addr); // 미
    usleep(150000);
    write_to_audio(tone[1], h2p_lw_audio_addr);
    write_to_audio(tone[2], h2p_lw_audio_addr);
    write_to_audio(tone[1], h2p_lw_audio_addr);
    write_to_audio(tone[2], h2p_lw_audio_addr);
    usleep(200000);
```

181,0-1 50%

AUDIO - audioplay.c

audioplay.c를 만든 이유

- 스위치 입력을 감지하여 해당 음의 주파수를 생성하고 그 사운드를 출력함
- 그리고 hex디스플레이에 스위치를 ON한 음을 표시해준다

- 스위치 입력에 따라 다양한 음계를 사운드를 통해

출력 하면서 hex디스플레이에 해당 음계를 표시하는게 목표

Write_to_audio함수에서 주파수 출력 및 hex 디스플레이 갱신

Sin함수를 사용하여 해당 주파수의 사인파형을 생성한다.

While를 통해 무한반복을하며 sw스위치를 읽고 동작을 수행한다.

리눅스 환경에서 동작해보기

- 1. 최신 날짜로 업데이트 해준다 date 월,일,시간,년도
- 2.Makefile로 만든 .ko 파일을 커널에 적재
- 3.insmod 사용하여 sw.ko, hex.ko 적재
- 4.mknod 사용하여 sw의 마이너 번호 238, hex의 메이저 번호 240 설정
- 5. gcc 을 사용하여 응용 프로그램인 audioplay.c를 실행
  (주의: 컴파일시 수학함수를 사용함으로 뒤에 -lm도 해야함)
- 6.즐겁게 음악 감상

참고문헌

- https://github.com/jiyeoon/DE1-SOC-audio
  - //Git-hub, audio code 참조
- http://www-ug.eecg.utoronto.ca/desl/nios_devices_SoC/dev_audio.html
  - //Audio core 참조
- https://blog.naver.com/PostView.naver?blogId=liebe_straum&logNo=221052015023
  - //도레미송 악보 참고

감사합니다