

Xilinx Zynq FPGA, TI DSP, MCU 기반의 프로그래밍 및 회로 설계 전문가 과정

강사 – Innova Lee(이상훈)

gcccompil3r@gmail.com

학생 – hoseong Lee(이호성)

hslee00001@naver.com

-code

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/mman.h>
#include <fcntl.h>
```

```
#define IN    0
#define OUT   1
```

```
#define GPIO_MAP_SIZE    0x10000
```

```
#define GPIO_DATA_OFFSET  0x00
#define GPIO_TRI_OFFSET   0x04
#define GPIO2_DATA_OFFSET 0x00
#define GPIO2_TRI_OFFSET  0x04
```

```
void usage(void){

    printf("*argv[0] -d <UIO_DEV_FILE> -i | -o <VALUE>\n");
    printf(" -d UIO device file - ex) /dev/uio0");
    printf(" -i Input from GPIO\n");
    printf(" -o <VALUE> Output to GPIO\n");
```

```
}
```

```
int main(int argc, char *argv[])
{
```

```
    int c, fd, value, direction = IN;
    char *uiod;
    void *ptr;
    int i=0;
```

```
printf("GPIO UIO Test\n");
printf("GITO!!!\n");
```

```
while( (c = getopt(argc, argv, "d:io:h")) != -1){
```

```
    switch(c){
```

```
        case 'd':
```

```
            uiod = optarg; // -d 뒤에 있는 /dev/uiio0 이라는 문자열을 가져온다.
```

```
            break;
```

```
        case 'i':
```

```
            direction = IN;
```

```
        case 'o':
```

```
            direction = OUT;
```

```
            value = atoi(optarg); //optarg : -o 뒤에 있는 인수 즉, "-0" "-1" 과 같은 문자열을 가져온다
```

```
            break;
```

```
        default :
```

```
            printf("Invalid Option: %c\n", (char)c);
```

```
            usage();
```

```
            return -1;
```

```
    }
```

```
}
```

```
fd = open(uiod, O_RDWR);
```

```
if(fd < 1){
```

```
    perror(argv[0]);
```

```
    printf("Onvalid UIO Device File: %s\n", uiod);
```

```
    usage();
```

```
    return -1;
```

```
}
```

```
//ptr = mmap(NULL, GPIO_MAP_SIZE, PROT_READ|PROT_WRITE,MAP_SHARED, fd, 0);
```

```

if(direction == IN){
    *((unsigned *)(ptr + GPIO_TRI_OFFSET)) == 255;
    printf("%s:Input: %08x\n", argv[0], value);
}else{
    ptr = mmap(NULL, GPIO_MAP_SIZE, PROT_READ|PROT_WRITE,MAP_SHARED, fd, 0);
    *((unsigned *)(ptr + GPIO_TRI_OFFSET)) = 0;
    munmap(ptr, GPIO_MAP_SIZE);

    while(1){
        ptr = mmap(NULL, GPIO_MAP_SIZE, PROT_READ|PROT_WRITE,MAP_SHARED, fd, 0);
        *((unsigned *)(ptr + GPIO_DATA_OFFSET)) = 1;
        munmap(ptr, GPIO_MAP_SIZE);

        for(i=0; i<10000; i++){

        }

        ptr = mmap(NULL, GPIO_MAP_SIZE, PROT_READ|PROT_WRITE,MAP_SHARED, fd, 0);
        *((unsigned *)(ptr + GPIO_DATA_OFFSET)) = 0;
        munmap(ptr, GPIO_MAP_SIZE);

        for(i=0; i<10000; i++){

        }

    }

}

//munmap(ptr, GPIO_MAP_SIZE);

return 0;
}

/*void* mmap(void* start, size_t length, int prot, int flags, int fd, off_t offset);
->ptr = mmap(NULL, GPIO_MAP_SIZE, PROT_READ|PROT_WRITE,MAP_SHARED, fd, 0);

```

파일이나 디바이스를 응용 프로그램의 주소 공간 메모리에 대응시킨다.

1인자 => 시작 포인터 주소 (아래의 예제 참조)

2인자 => 파일이나 주소공간의 메모리 크기

3인자 => PROT 설정 (읽기, 쓰기, 접근 권한, 실행)

4인자 => flags는 다른 프로세스와 공유할지 안할지를 결정한다.

5인자 => fd는 쓰거나 읽기용으로 열린 fd값을 넣어준다.

6인자 => offset은 0으로 하든지 알아서 조절한다.

```
int munmap(void* start, size_t length);  
->munmap(ptr, GPIO_MAP_SIZE);
```

할당된 메모리 영역을 해제한다.

1인자 => 위에 mmap으로 지정된 포인터값 넣어주고

2인자 => 위에서 사용했던 `length`와 동일하게 넣어준다.

(왜냐면.. 할당했던거 동일하게 해제해야 하니깐..)

1. Petalinux 와 Vivado 2017.1 연동하기
2. 하드웨어(HW)와 소프트웨어(SW) Co-Design
3. Vivado 로는 HW 를 설계함
4. Linux 로 SW 를 설계함
5. Device Driver 를 통해서 SW 와 HW 를 서로 묶어줌
6. Device Driver 를 모르면서 OS 를 할 줄 안다는 것은 어불 성설임
7. RTOS 는 Linux 와 같은 GP OS 와 개념부터가 완전히 다름
8. RTOS 를 Linux 로 구현하기 위해 추가된 개념이 RT Linux 임
____(역시 RTOS 와는 내부 구현이 너무나도 상이함)
9. PetaLinux 환경 설정하기(Vivado HW 설계 및 구현은 공개 안함)

(0) 리눅스 부팅

- (1) Xilinx Vivado, Vivado HLS 및 Vivado SDK 를 설치한다.
- (2) 우선 LiveUSB 를 다운받는다.
- (3) Xilinx 사이트에서 PetaLinux 를 다운 받는다.
- (4) 리눅스가 올라가 Device Driver 를 연동할 것이므로
____ HW 설계 및 구현 파트와 SW 영역을 분리하는 것이 좋다.
____ 앞으로 작업할 workspace 를 별도로 만든다.
____ 예로 이름을 fpga_dev_driver 라고 지정한다.
- (5) cd ~
- (6) mkdir fpga_dev_driver
- (7) cd ~/Downloads
- (8) 실행 권한도 안주고 실행 안된다는 소리 하지 말자

chmod 755 petalinux-~~~~-dec.run

(9) unzip LiveUSB~~~.zip

(10) 작업을 수행하기 위해 필요한 SW 패키지들을 미리 사전에 준비한다.

(11) sudo apt-get update

(12) sudo apt-get install tofrodos iproute tftpd-hpa gawk gcc git-core make net-tools libncurses5-dev zlib1g-dev libssl-dev flex bison lib32z1 lib32ncurses5 lib32stdc++6 libselinux1

(13) sudo dpkg --add-architecture i386

(14) sudo apt-get update

(15) sudo apt-get install libbz2-1.0:i386

(16) sudo apt-get install tofrodos iproute tftpd-hpa gawk gcc git-core make net-tools libncurses5-dev zlib1g-dev libssl-dev flex bison lib32z1 lib32ncurses5 libselinux1

(17) sudo apt-get install xinetd tftpd-hpa

(18) sudo apt-get install qemu-user-static qemu-system

(19) sudo apt-get install linaro-image-tools

(20) sudo apt-get install gcc-arm-linux-gnueabi

(21) ./petalinux-~~~~-dec.run ~/petalinux_zynq/

(22) 라이선스 동의하고 gogogo

(23) cd ~/petalinux_zynq

(24) cd petalinux-v~~~-final

(25) cp ~/Downloads/LiveUSB~~~/ZYBO_petalinux_v~~~.bsp ./

(26) cd ..

(27) chmod -R 755 petalinux-v~~~-final

(28) cd petalinux-v~~~-final

(29) cd components/linux-kernel/xlnx-4.0/drivers/uio

(30) vi uio_pdrv_genirq.c

아래와 같이 내용 추가하고 저장

(리눅스 에디터에서 CONFIG_OF 로 검색해서 찾도록 한다)

```
#ifdef CONFIG_OF
static struct of_device_id uio_of_genirq_match[] = {
    { .compatible = "generic-uio", },
};
MODULE_DEVICE_TABLE(of, uio_of_genirq_match);
module_param_string(of_id, uio_of_genirq_match[0].compatible, 128, 0);
MODULE_PARAM_DESC(of_id, "Openfirmware id of the device to be handled by uio");
#endif
```

(31) vi ~/.bashrc

맨 아래쪽에 내용을 추가하고 저장한다.

```
source ~/petalinux_zynq/petalinux-v~~~~-final/settings.sh
```

(32) source ~/.bashrc

(33) cd fpga_dev_driver

(34) cp ~/petalinux_zynq/petalinux-v~~~~-final/ZYBO_petalinux_v~~~~.bsp ./

(31) petalinux-create -t project -n test --template zynq

이 명령어가 잘 먹으면 PetaLinux 환경 설정이 잘 되었음을 의미한다.

(32) rm -rf test

(33) petalinux-create -t project -s ZYBO_petalinux_v~~~~.bsp

(34) petalinux-build

(35) 만약 오류가 난다면 아래 명령을 입력해보도록 한다.

____ `sudo dpkg-config dash`

____ 화면이 나오면 No 를 선택한다.

(36) `petalinux-build`

(37) `petalinux-boot --qemu --kernel`

____ 여기까지 진행이 되면 가상화 기법으로

____ ARM 커널을 x86 에서 구동시킬 수 있게 된다.

____ 이제 진짜 FPGA HW(보드)에 리눅스를 올려야 한다.

____ 이로서 PetaLinux 환경 구성은 완벽하게 완료되었다.

******* 디바이스 드라이버를 아직 만들지 못했다면 *******

******* 일단 그냥 57 번으로 넘어가서 작업하면 *******

******* FPGA 보드 상에 리눅스는 올라간다 *******

(38) 이제 HW 설계와 SW 연동을 수행해보도록 한다.

____ 그 이전에 각종 설계를 수행할 것인데

____ 관리를 위해 디렉토리를 한 단계 더 분할하도록 한다.

____ `mkdir hw_sw_co_design`

(39) `cd hw_sw_co_design`

(40) FPGA 로 설계한 HW 를 보관할 디렉토리를 만든다.

____ `mkdir hardware`

(41) Xilinx Vivado 툴을 실행한다.

(42) 수업에서 진행하였듯이 아래와 같은 각종 HW 를 설계한다.

____ (GPIO, ADC, I2C, SPI, PWM, 기타 전용 HW 등등)

프로젝트 저장장치를 방금 만든 hardware 디렉토리에 저장하도록 한다.

hardware 의 위치는 아래와 같다.

fpga_dev_driver/hw_sw_co_design/hardware

(43) petalinux-create -t project -n kernel --template zynq

(44) cd kernel

(45) petalinux-config --get-hw-description=~/.fpga_dev_driver/hw_sw_co_design/hardware/~~~.sdk

여기서 sdk 는 Vivado 에서 HW 설계한 내용에 해당한다.

(46) cd components/bootloader/zynq_fsbl

(47) ls

FPGA 베이스의 Cortex-A9 부트 코드를 볼 수 있다

(48) cd ../../..

(49) petalinux-config -c u-boot

(50) petalinux-build

(51) petalinux-create -t apps -n device_driver --enable

(52) cd components/apps/device_driver

(53) vi device_driver.c

HW 를 제어하기 위한 SW 코드인 Device Driver 코드를 작성한다.

(54) cd ~/.fpga_dev_driver/hw_sw_co_design/kernel/images/linux

(55) ls

여기에 부트 로더와 리눅스 이미지가 있는 것을 볼 수 있을 것이다.

(56) petalinux-build

(57) petalinux-package --boot --fsbl zynq_fsbl.elf --fpga ./비트스트림 --u-boot --force

Vivado 에서 설계한 HW 와 관련한 비트스트림 정보가 여기에 있다.

예로 비트스트림 파일명이 test_wrapper.bit 라면

_____./비트스트림은 ./test_wrapper.bit 로 변경되어야 한다.

(58) 수업중 제공한 문서를 기반으로 SD 카드에

____ 부트 로더와 리눅스 이미지를 옮긴다.

(59) FPGA 보드의 점퍼를 SD 카드 부팅으로 변경한다.

(60) 컴퓨터와 FPGA 보드를 USB 로 연결한다.

(61) 전원을 인가한다.

(62) dmesg 를 통해 USB Device Driver 가 잘 잡히는지 확인한다.

(63) sudo apt-get install putty

(64) 푸티의 폰트 등 각종 설정을 수행한다.

(65) sudo chmod 666 /dev/ttyUSB1

(66) Baud Rate 를 115200 으로 지정하고 연결한다.

(67) 리눅스 부팅되는 모습을 볼 수 있다.

10. 학생들 멘탈 찢김 ?!

_____(FPGA 은상 수상자 랩실에서 이거 1 년 동안 했다고함 ㅋㅋㅋ)

_____근대 우리 수업에선 하루 컷 했음 ;;;

_____이거 도대체 어떻게 설명해야함 ?

PetaLinux SDK contains software from a number of sources. Please review the following licenses and indicate your acceptance of each to continue.

You do not have to accept the licenses, however if you do not then you may not use PetaLinux SDK.

Use PgUp/PgDn to navigate the license viewer, and press 'q' to close

Press Enter to display the license agreements

Do you accept Xilinx End User License Agreement? [y/N] > y

Do you accept Webtalk Terms and Conditions? [y/N] > y

Do you accept Third Party End User License Agreement? [y/N] > y

INFO: Checking installation environment requirements...

INFO: Checking free disk space

INFO: Checking installed tools

INFO: Checking installed development libraries

INFO: Checking network and other services

INFO: Installing PetaLinux SDK to "/home/jhb/petalinux_zynq/petalinux-v2015.4-final"

INFO: PetaLinux SDK has been installed to /home/jhb/petalinux_zynq/petalinux-v2015.4-final

jhb@onestar:~\$

```
mhn-Z20NH-AS51B5U:~$ source ~/.bashrc
sh: /home/mhn/petalinux_zynq/petalinux-v2015.4-final/setting.sh: No such file or directory
mhn-Z20NH-AS51B5U:~$ cd ../petalinux_zynq/petalinux-v2015.4-final/
sh: cd: ../petalinux_zynq/petalinux-v2015.4-final/: No such file or directory
mhn-Z20NH-AS51B5U:~$ cd fpga_test/
mhn-Z20NH-AS51B5U:~/fpga_test$ petalinux-create -t project -n test --template zynq
petalinux-create: command not found
mhn-Z20NH-AS51B5U:~/fpga_test$ source ~/.bashrc
sh: /home/mhn/petalinux_zynq/petalinux-v2015.4-final/setting.sh: No such file or directory
mhn-Z20NH-AS51B5U:~/fpga_test$ ls ~/petalinux_zynq/petalinux-v2015.4-final/
components/  etc/  settings.csh  settings.sh  .sourcelog  tools/
mhn-Z20NH-AS51B5U:~/fpga_test$ vi ~/.bashrc
mhn-Z20NH-AS51B5U:~/fpga_test$ source ~/.bashrc
PetaLinux environment set to '/home/mhn/petalinux_zynq/petalinux-v2015.4-final'
INFO: Checking free disk space
INFO: Checking installed tools
INFO: Checking installed development libraries
INFO: Checking network and other services
mhn-Z20NH-AS51B5U:~/fpga_test$ ls
mhn-Z20NH-AS51B5U:~/fpga_test$ petalinux-create -t project -n test --template zynq
INFO: Create project: test
INFO: New project successfully created in /home/mhn/fpga_test/test
mhn-Z20NH-AS51B5U:~/fpga_test$
```

Connect to Server

Download

random seed: random read with 0 bits of entropy available
Starting internet superserver: inetd.
update-rc.d: /etc/init.d/run-postinsts exists during rc.d purge (
Removing any system startup links for run-postinsts ...
/etc/rcS.d/S99run-postinsts
INIT: Entering runlevel: 5
Configuring network interfaces... done.
starting Busybox HTTP Daemon: httpd... done.
NET: Registered protocol family 10
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
Built with PetaLinux v2015.4 (Yocto 1.8) ZYBO_petalinux_v2015_4 /dev
ZYBO_petalinux_v2015_4 login: root
Password: macb e000b000.ethernet eth0: link up (1000/Full)
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
root
Login incorrect
ZYBO_petalinux_v2015_4 login: root
Password:
login[828]: root login on 'ttyPS0'
root@ZYBO_petalinux_v2015_4:~#

Re: Petalinux settings.sh problem [New]

03-25-2016 10:18 AM

This is what worked for me. First make sure your SHELL is /bin/bash b

```
chsh -s /bin/bash
```

You will have to log out and log back in, then you can then check it by ty

```
sudo rm /bin/sh
```

```
sudo ln -s /bin/bash /bin/sh
```

1 Views)

Add Tag...

gung

【출처】 [64 회차 교육 로그](#) | 작성자 [silenc3502](#)