


## 퀘스트 수행 기록

 <b>QUEST 3</b>	<b>QUEST 3</b> A7/ M4 예제 데모 시현 (LED control, Button detect)	<b>도전하기</b> →
	<b>DATE</b> 2020년 5월 4일 (월 12:00) ~ 5월 10일 (일) 24:00	

**퀘스트 3/7 - A7/ M4 예제 데모 시현 (LED control, Button detect)** 작성자: 🤖 코난친구

1. SDK 추출, 생성 결과물 및 설치 결과물에 대한 디렉토리 정보와 함께 컴파일러(arch=armv7ve)의 정보(디렉토리 위치, CC 환경변수) 출력하여 제출 (\* Sample screenshot 참고)

a) 환경설정

```
DISTRO=openstlinux-weston MACHINE=stm32mp1 source layers/meta-st/scripts/envsetup.sh
```

b) SDK 추출

```
bitbake -c populate_sdk st-image-weston
```

c) SDK 설치 - 설치시 경로 지정 (./sdk로 입력함)

```
./build-openstlinuxweston-stm32mp1/tmp-glibc/deploy/sdk/st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.sh
```

d) 결과확인

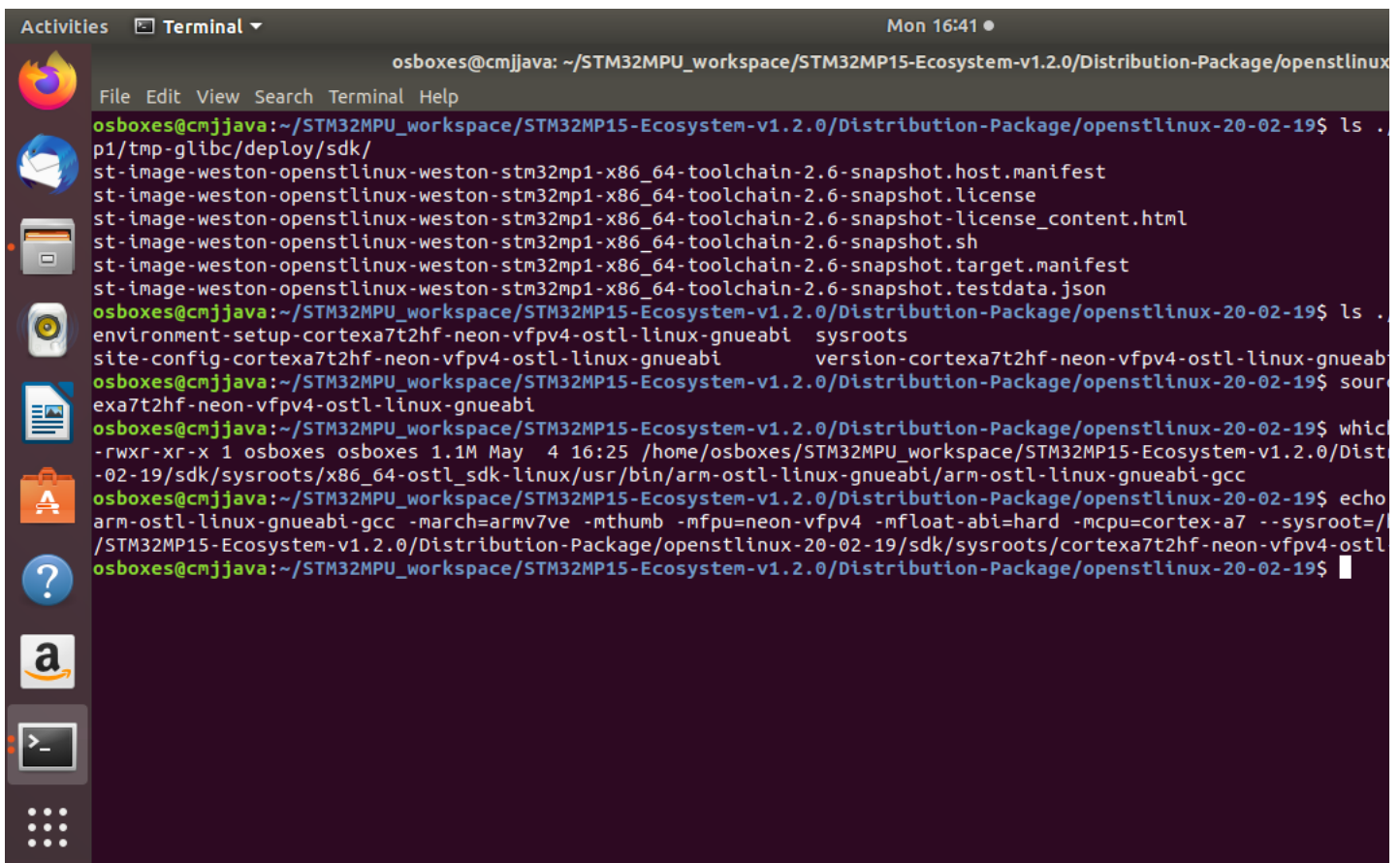
```
ls ./build-openstlinuxweston-stm32mp1/tmp-glibc/deploy/sdk/
```

```
ls ./sdk 설치시 지정한 경로
```

```
source ./sdk/environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
```

```
which $CC | xargs ls -alh
```

```
echo $CC
```



```

osboxes@cmjjava: ~/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19$ ls .
p1/tmp-glibc/deploy/sdk/
st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.host.manifest
st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.license
st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.license_content.html
st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.sh
st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.target.manifest
st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.testdata.json
osboxes@cmjjava: ~/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19$ ls .
environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi sysroots
site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
osboxes@cmjjava: ~/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19$ source
exa7t2hf-neon-vfpv4-ostl-linux-gnueabi
osboxes@cmjjava: ~/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19$ which
-rwxr-xr-x 1 osboxes osboxes 1.1M May  4 16:25 /home/osboxes/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Dist
-02-19/sdk/sysroots/x86_64-ostl_sdk-linux/usr/bin/arm-ostl-linux-gnueabi/arm-ostl-linux-gnueabi-gcc
osboxes@cmjjava: ~/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19$ echo
arm-ostl-linux-gnueabi-gcc -march=armv7ve -mthumb -mfpu=neon-vfpv4 -mfloat-abi=hard -mcpu=cortex-a7 --sysroot=/
/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19/sdk/sysroots/cortexa7t2hf-neon-vfpv4-ostl
osboxes@cmjjava: ~/STM32MPU_workspace/STM32MP15-Ecosystem-v1.2.0/Distribution-Package/openstlinux-20-02-19$
  
```

2) SDK 를 사용하여 빌드 된 executable file 의 정보를 target(DK2 board) 시스템 정보를 함께 출력하여 제출 (\* Sample screenshot 참고)

a) source : [https://wiki.st.com/stm32mpu/wiki/How\\_to\\_control\\_a\\_GPIO\\_in\\_userspace](https://wiki.st.com/stm32mpu/wiki/How_to_control_a_GPIO_in_userspace) 참고

```
mkdir gpio
```

```
cd gpio
```

```
mkdir -p install_artifact install_artifact/usr install_artifact/usr/local install_artifact/usr/local/bin
```

```
gpio.c 소스 작성
```

```
#include <errno.h>
```

```
#include <fcntl.h>
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```

#include <string.h>
#include <sys/ioctl.h>
#include <unistd.h>

#include <linux/gpio.h>

int main(int argc, char **argv)
{
    struct gpiohandle_request req;
    struct gpiohandle_data data;
    char chrdev_name[20];
    int fd, ret;

    strcpy(chrdev_name, "/dev/gpiochip0");

    /* Open device: gpiochip0 for GPIO bank A */
    fd = open(chrdev_name, 0);
    if (fd == -1) {
        ret = -errno;
        fprintf(stderr, "Failed to open %s\n", chrdev_name);

        return ret;
    }

    /* request GPIO line: GPIO_A_14 */
    req.lineoffsets[0] = 14;
    req.flags = GPIOHANDLE_REQUEST_OUTPUT;
    memcpy(req.default_values, &data, sizeof(req.default_values));
    strcpy(req.consumer_label, "led_gpio_a_14");
    req.lines = 1;

    ret = ioctl(fd, GPIO_GET_LINEHANDLE_IOCTL, &req);
    if (ret == -1) {
        ret = -errno;
        fprintf(stderr, "Failed to issue GET LINEHANDLE IOCTL (%d)\n",
            ret);
    }
    if (close(fd) == -1)
        perror("Failed to close GPIO character device file");

    /* Start led blinking */
    while(1) {

        data.values[0] = !data.values[0];
        ret = ioctl(req.fd, GPIOHANDLE_SET_LINE_VALUES_IOCTL, &data);
        if (ret == -1) {
            ret = -errno;
            fprintf(stderr, "Failed to issue %s (%d)\n",
                "GPIOHANDLE_SET_LINE_VALUES_IOCTL", ret);
        }
        sleep(1);
    }

    /* release line */
    ret = close(req.fd);
    if (ret == -1) {
        perror("Failed to close GPIO LINEHANDLE device file");
        ret = -errno;
    }
    return ret;
}

```

compile : [https://wiki.st.com/stm32mpu/wiki/How\\_to\\_cross-compile\\_with\\_the\\_Developer\\_Package#Adding\\_a\\_.22hello\\_world.22\\_user\\_space\\_example](https://wiki.st.com/stm32mpu/wiki/How_to_cross-compile_with_the_Developer_Package#Adding_a_.22hello_world.22_user_space_example) 참고 하여 진행  
컴파일

**\$CC gpio.c -o ./install\_artifact/usr/local/bin/gpio**

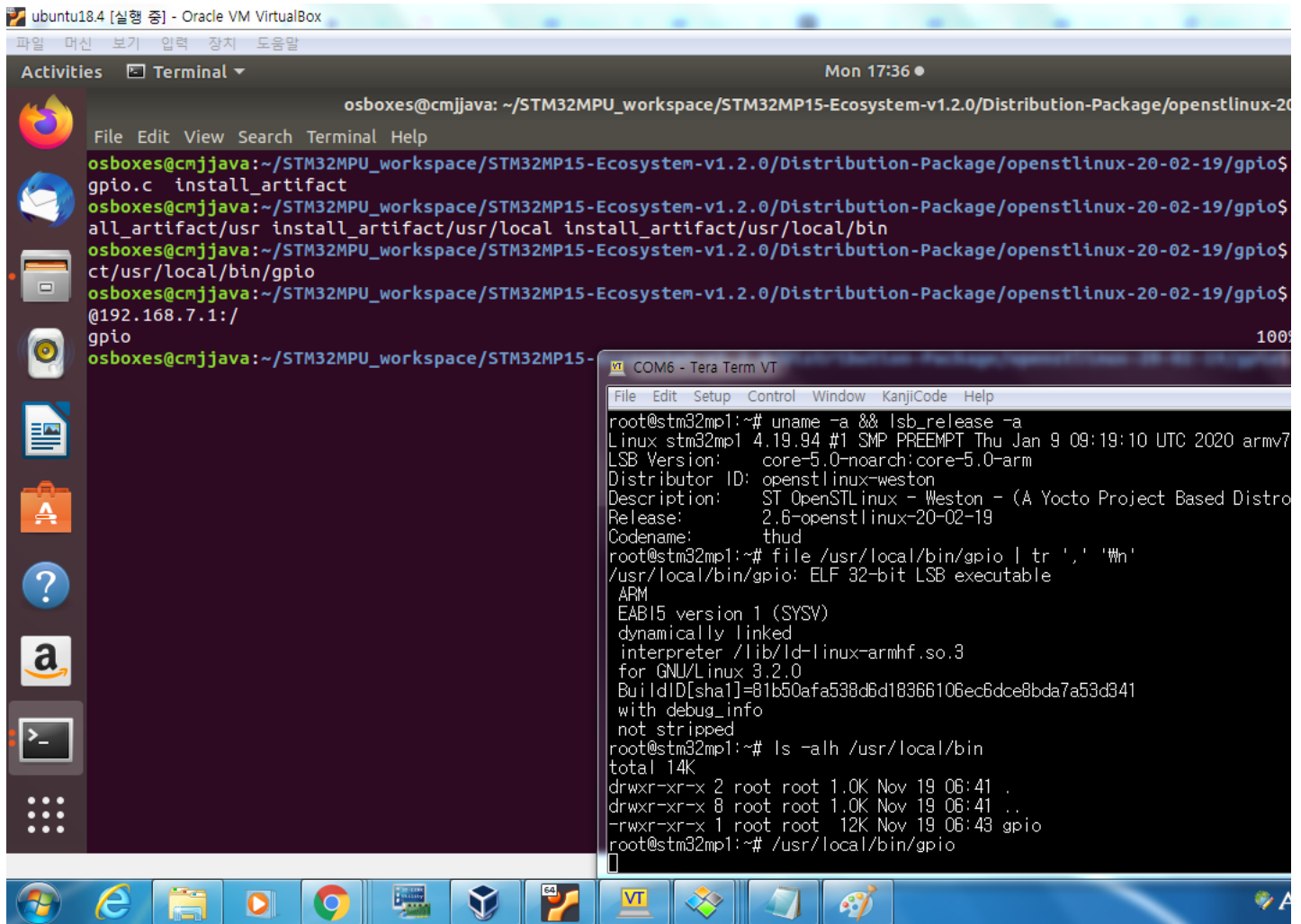
다운로드

**scp -r install\_artifact/\* root@192.168.7.1:/**

b) 터미널에서 실행

**uname -a && lsb\_release -a**

```
file /usr/local/bin/gpio | tr ', ' '\n'
ls -alh
```



3) MP1 터미널(tty serial, ssh, target terminal등)에서 LED blinking 어플리케이션 실행하면 LED가 깜빡이는 장면 영상 제출  
a) `/usr/local/bin/gpio`

초록색이 켜진 상태에서 녹화 시작, 5초후 프로그램 실행으로 초록색이 깜빡임

Virtualbox에서 진행 하고 있는데.. 디스크 용량이 90GB입니다.

128G SSD에 100G할당 했는데.. Quest 3 이후에도 용량을 많이 차지하는 미션이 있으면 큰일이네요..