


퀘스트 수행 기록

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

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

역시나 쉽지 않은 과정입니다.
(한번에 쉽게 넘어가면 별거 아닐 듯 한데요.)

먼저, 다른 분들이 먼저 글을 작성해서 공개해 주신 내용이 있어서, 그나마 많은 도움이 됩니다.

하드디스크 공간을 더 늘렸습니다. 60GB -> 100GB
(작업을 위한 리소스가 장난 아니네요).

1. SDK 추출 하여 제출하기

먼저 SDK 추출에 대해서는 링크가 없어서, 많이 힘들었습니다.

zmffito** 님이 남겨주신 댓글을 통해서 정보 획득!

https://wiki.st.com/stm32mpu/wiki/How_to_create_an_SDK_for_OpenSTLinux_distribution

작업 위치 : /work/STM32MP1/Distribution-Package/openstlinux-20-02-19\$

에서, bitbake에 대한 환경 변수 반영 스크립트 실행

PC \$> DISTRO=openstlinux-weston MACHINE=stm32mp1 source layers/meta-st/scripts/envsetup.sh

PC \$> bitbake -c populate_sdk st-image-weston

이 작업 또한 bitbake를 통해서 SDK를 생성하는 과정으로 대략 3~5시간 소요된다고하여서,
VMware상에서 Processor 할당을 1 -> 4로 변경하였습니다.
이를 통해서 처음에 진행 속도가 좀 많이 개선이 되었는데, 중간에 에러에 의한 종료가 1회 발생
이후 83% or 97%에서 무수히 많은 시간을 소비.
(이 부분에서 12시간 이상 소요~)

```

jobaek@ubuntu: ~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19
File Edit View Search Terminal Help
meta-qt5
meta-st-openstlinux
meta
workspace = "<unknown>:<unknown>"

Initialising tasks: 100% |#####| Time: 0:00:05
Sstate summary: Wanted 862 Found 366 Missed 496 Current 1573 (42% match, 79% complete)
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks

Keyboard Interrupt, closing down...

NOTE: Tasks Summary: Attempted 307 tasks of which 293 didn't need to be rerun and all succeeded.
NOTE: Writing buildhistory
ERROR: Command execution failed: Stopped build

Summary: There was 1 ERROR message shown, returning a non-zero exit code.
Execution was interrupted, returning a non-zero exit code.
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/build-op
enstlinuxweston-stm32mp1$ ^C
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/build-op
enstlinuxweston-stm32mp1$ ls

```

그래서 Ctrl + C를 통해서 종료하고, 다시 진행했더니,
83%에서 잠시 후 바로 84% -> 96%로 빠른 진행을 보였습니다.

```

jobaek@ubuntu: ~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/build-openstlinux...
meta-oe
meta-gnome
meta-xfce
meta-initramfs
meta-multimedia
meta-networking
meta-webserver
meta-fileystems
meta-perl
meta-python
meta-st-stm32mp
meta-qt5
meta-st-openstlinux
meta
workspace = "<unknown>:<unknown>"

Initialising tasks: 100% [#####] Time: 0:00:03
Sstate summary: Wanted 384 Found 365 Missed 19 Current 2051 (95% match, 99% complete)
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 6675 tasks of which 6650 didn't need to be rerun and all succeeded.
NOTE: Writing buildhistory
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/build-openstlinuxweston-stm32mp1$

```

추출된 SDK에 대한 설치 스크립트 실행

PC \$> ./build-openstlinuxweston-stm32mp1/tmp-glibc/deploy/sdk/st-image-weston-openstlinux-weston-stm32mp1-x86_64-toolchain-2.6-snapshot.sh
설치 위치를 물어보는 프롬프트가 나오면, ./sdk 지정 (default는 다른 값으로 되어 있음)

설치 결과 확인 (과제 1)

```

jobaek@ubuntu: ~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19
=====
Enter target directory for SDK (default: /opt/st/stm32mp1/2.6-snapshot): ./sdk
You are about to install the SDK to "/home/jobae/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/sdk". Proceed[Y/n]?
Extracting SDK.....done
Setting it up...done
SDK has been successfully set up and is ready to be used.
Each time you wish to use the SDK in a new shell session, you need to source the environment setup script e.g.
$ . /home/jobae/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/sdk/environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ ls
build-openstlinuxweston-stm32mp1 layers layers.tar.gz sdk
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ ls ./build-openstlinuxweston-stm32mp1/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
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ ls ./sdk
environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
sysroots
version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$

```

설치 스크립트를 수행하면, 위와 같이 .json 파일이 하나더 추가되어 있다.

이제 sdk 아래에 생성된 크로스컴파일러에 대한 환경 등록

PC \$> source ./sdk/environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi

그리고 적용된 결과 확인

PC \$> which \$CC | xargs ls -alh

PC \$> echo \$CC

```
jobaek@ubuntu: ~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19
-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ ls
build-openstlinuxweston-stm32mp1 layers layers.tar.gz sdk
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ ls ./build-openstlinux
weston-stm32mp1/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
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ ls ./sdk
environment-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
site-config-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
sysroots
version-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ source ./sdk/environme
nt-setup-cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ which $CC | xargs ls -
alh
-rwxr-xr-x 1 jobaek jobaek 1.1M May  5 06:14 /home/jobaek/work/STM32MP1/Distribution-Package/op
enstlinux-20-02-19/sdk/sysroots/x86_64-ostl_sdk-linux/usr/bin/arm-ostl-linux-gnueabi/arm-ostl-l
inux-gnueabi-gcc
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$ echo $CC
arm-ostl-linux-gnueabi-gcc -march=armv7ve -mthumb -mfpv=neon-vfpv4 -mfloat-abi=hard -mcpu=corte
x-a7 --sysroot=/home/jobaek/work/STM32MP1/Distribution-Package/openstlinux-20-02-19/sdk/sysroot
s/cortexa7t2hf-neon-vfpv4-ostl-linux-gnueabi
jobaek@ubuntu:~/work/STM32MP1/Distribution-Package/openstlinux-20-02-19$
```

2. SDK를 이용해서 사용자 예제 작성하기

https://wiki.st.com/stm32mpu/wiki/How_to_control_a_GPIO_in_userspace

작업 폴더 아래에 사용자 폴더 생성 (user)

PC \$> mkdir user

PC \$> cd user

사용자 폴더 아래에 gpio 예제를 작성

PC \$> mkdir gpio

PC \$> cd gpio

이제, GPIO 관련 예제 코드 생성

PC \$> vi gpio.c

위에 wiki에서 3.2 항목의 소스 코드 참조 : GREEN LED On/Off 예제

(cmjij** 님 수행 글 참조!)

사용자 영역에 대한 코드 추가 방법에 대해서는 3.2 코드 아래에 링크가 있음!

그곳에서 한 번 더, "Adding a "hello world" user space example 링크를 참조

https://wiki.st.com/stm32mpu/wiki/How_to_cross-compile_with_the_Developer_Package#Adding_a_22hello_world.22_user_space_example

컴파일 후 생성될 binary 파일이 저장될 디렉토리 생성

PC \$> mkdir -p install_artifact install_artifact/usr install_artifact/usr/local install_artifact/usr/local/bin

컴파일 수행

PC \$> \$CC gpio.c -o ./install_artifact/usr/local/bin/gpio

생성된 실행 파일을 이제 보드에 다운로드 진행

보드에 랜선을 PC와 연결하고, 보드의 IP 주소 설정 : 192.168.1.10

PC \$> scp -r install_artifact/* root@192.168.1.10:/

이제 보드에 정상적으로 다운로드 되어 있는지 확인 수행을 위해서 PC와 보드간에 추가로 USB 케이블 연결

(보드의 ST-LINK를 이용해서 PC에 연결 : VCP 추가 됨)

PC에서 Terminal 프로그램을 실행하여 보드와 연결

설치 결과 확인 (과제 2)

BOARD \$> uname -a && ls -l release -a

BOARD \$> file /usr/local/bin/gpio | tr ',' '\n'

BOARD \$> ls -alh /usr/local/bin/


```

COM5 - Tera Term VT
메뉴(F) 수정(E) 설정(S) 제어(O) 창(W) 도움말(H)

Try 'tr --help' for more information.
root@stn32mp1:~# file /usr/local/bin/gpio | tr ', ' '\n'
/usr/local/bin/gpio: ELF 32-bit LSB executable
ARM
EHABI version 1 (SVSV)
dynamically linked
interpreter /lib/ld-linux-armhf.so.3
for GNU/Linux 3.2.0
BuildID[sha1]=81b50afa538d6d18366106ec6dce8bda7a53d341
with debug_info
not stripped
root@stn32mp1:~# ls -alh
total 4.0K
drwxr-xr-x 2 root root 1.0K Nov 18 2019 .
drwxr-xr-x 3 root root 1.0K Mar 9 2018 ..
-rw-r--r-- 1 root root 777 Nov 18 22:59 .ash_history
-rw-r--r-- 1 root root 238 Mar 9 2018 README-CHECK-GPU
root@stn32mp1:~#
root@stn32mp1:~#
root@stn32mp1:~#
root@stn32mp1:~#
root@stn32mp1:~# uname -a && lsb_release -a
Linux stn32mp1 4.19.94 #1 SMP PREEMPT Thu Jan 9 09:19:10 UTC 2020 armv7l armv7l armv7l GNU/Linux
LSB Version: core-5.0-noarch:core-5.0-arm
Distributor ID: openstlinux-weston
Description: ST OpenSTLinux - Weston - (A Yocto Project Based Distro) 2.6-openstlinux-20-02-19
Release: 2.6-openstlinux-20-02-19
Codename: thud
root@stn32mp1:~# file /usr/local/bin/gpio | tr ', ' '\n'
/usr/local/bin/gpio: ELF 32-bit LSB executable
ARM
EHABI version 1 (SVSV)
dynamically linked
interpreter /lib/ld-linux-armhf.so.3
for GNU/Linux 3.2.0
BuildID[sha1]=81b50afa538d6d18366106ec6dce8bda7a53d341
with debug_info
not stripped
root@stn32mp1:~# ls -alh /usr/local/bin/
total 14K
drwxr-xr-x 2 root root 1.0K Nov 18 22:57 .
drwxr-xr-x 8 root root 1.0K Nov 18 22:57 ..
-rwxr-xr-x 1 root root 12K Nov 18 22:57 gpio
root@stn32mp1:~#

```

3. 사용자 예제(gpio) 동작 영상 업로드



뒤로