Mango-IMX6Q Yocto Project Source 다운로드 및 소스 추출 방법

http://www.mangoboard.com/
http://cafe.naver.com/embeddedcrazyboys
Crazy Embedded Laboratory

Document History

Revision	Date	Change note
Init	2016-01-27	전종인

1. 소스 다운로드 및 컴파일 방법 5
2. 툴 체인 설치 방법 12
3. Kernel, uboot 소스 추출 및 컴파일 방법 14
3.1. U-boot 14
3.2. Kernel 16
4. Image Write 방법 19

1. 소스 다운로드 및 컴파일 방법

1) Git 설정을 해줍니다.

```
$ mkdir fsl-release-bsp
$ cd fsl-release-bsp
$ git config --global user.name "이름"
$ git config --global user.email "이메일"
```

```
$ repo init -u git://git.freescale.com/imx/fsl-arm-yocto-bsp.git -b imx-3.10.17-1.0.0_ga
Get https://gerrit.googlesource.com/git-repo
remote: Counting objects: 117, done
remote: Finding sources: 100% (117/117)
remote: Total 2920 (delta 1538), reused 2920 (delta 1538)
Receiving objects: 100% (2920/2920), 2.47 MiB | 3.48 MiB/s, done.
Resolving deltas: 100% (1538/1538), done.
From https://gerrit.googlesource.com/git-repo
 * [new branch]      maint      -> origin/maint
 * [new branch]      master     -> origin/master
 * [new branch]      stable     -> origin/stable
 * [new tag]         v1.0       -> v1.0
 * [new tag]         v1.0.1     -> v1.0.1
 * [new tag]         v1.0.2     -> v1.0.2
 * [new tag]         v1.0.3     -> v1.0.3
 * [new tag]         v1.0.4     -> v1.0.4
 * [new tag]         v1.0.5     -> v1.0.5
 * [new tag]         v1.0.6     -> v1.0.6
 * [new tag]         v1.0.7     -> v1.0.7
 * [new tag]         v1.0.8     -> v1.0.8
 * [new tag]         v1.0.9     -> v1.0.9
 * [new tag]         v1.1       -> v1.1
                ... 생 략 ....
 * [new tag]         v1.8.2     -> v1.8.2
 * [new tag]         v1.9.0     -> v1.9.0
 * [new tag]         v1.9.1     -> v1.9.1
 * [new tag]         v1.9.2     -> v1.9.2
 * [new tag]         v1.9.3     -> v1.9.3
 * [new tag]         v1.9.4     -> v1.9.4
```

```
 * [new tag]             v1.9.5       -> v1.9.5
 * [new tag]             v1.9.6       -> v1.9.6
Get git://git.freescale.com/imx/fsl-arm-yocto-bsp.git
remote: Counting objects: 104, done.
remote: Compressing objects: 100% (103/103), done.
remote: Total 104 (delta 33), reused 0 (delta 0)
Receiving objects: 100% (104/104), 14.83 KiB, done.
Resolving deltas: 100% (33/33), done.
From git://git.freescale.com/imx/fsl-arm-yocto-bsp
 * [new branch]          imx-3.10.17-1.0.0_ga -> origin/imx-3.10.17-1.0.0_ga
 * [new branch]          imx-3.10.17-1.0.1_ga -> origin/imx-3.10.17-1.0.1_ga
 * [new branch]          imx-3.10.31-1.1.0_alpha -> origin/imx-3.10.31-1.1.0_alpha
 * [new branch]          imx-3.10.31-1.1.0_beta -> origin/imx-3.10.31-1.1.0_beta

Your identity is: treego <treego@crz-tech.com>
If you want to change this, please re-run 'repo init' with --config-name

Testing colorized output (for 'repo diff', 'repo status'):
  black     red       green     yellow    blue      magenta   cyan      white
  bold      dim       ul        reverse
Enable color display in this user account (y/N)?
```

2) 해당 소스를 다운 받습니다. 시간이 오래 걸립니다.

```
$ repo sync
Fetching project meta-browser
Fetching project meta-openembedded
remote: Counting objects: 1324, done.
remote: Counting objects: 41552, done.
remote: Compressing objects: 100% (15835/15835), done.
remote: Total 1324 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (1324/1324), 358.69 KiB | 142 KiB/s, done.
Resolving deltas: 100% (790/790), done.
From git://github.com/OSSystems/meta-browseriB | 159 KiB/s
                ...    생    략 ....
 * [new tag]           yocto-1.5   -> yocto-1.5
 * [new tag]           yocto-1.5.1 -> yocto-1.5.1
 * [new tag]           yocto-1.5.1.final -> yocto-1.5.1.final
```

```
 * [new tag]         yocto-1.5.2 -> yocto-1.5.2
 * [new tag]         yocto-1.5.3 -> yocto-1.5.3
 * [new tag]         yocto-1.5.final -> yocto-1.5.final
 * [new tag]         yocto-1.5_M5.rc2 -> yocto-1.5_M5.rc2
 * [new tag]         yocto-1.6   -> yocto-1.6
 * [new tag]         yocto-1.6.1 -> yocto-1.6.1
 * [new tag]         yocto_1.5_M5.rc8 -> yocto_1.5_M5.rc8
Fetching projects: 100% (8/8), done.
Checking out files: 100% (4486/4486), done. files:   10% (466/4486)
Syncing work tree: 100% (8/8), done.
```

3) 완료 되면, 아래와 같이 소스가 다운 받아 집니다.

```
$ ls
README  downloads  fsl-setup-release.sh  imx6q-yocto  setup-environment  sources
```

4) "fsl-setup-release.sh" 스크립트 파일을 이용해서, 컴파일 설정을 해줍니다.

```
$ MACHINE=imx6qsabresd source fsl-setup-release.sh -b imx6q-yocto -e fb
Build directory is   imx6q-yocto

 Using FB backend with FB DIST_FEATURES to override poky X11 DIST FEATURES
Configuring for imx6qsabresd

Some SoC depends on libraries and packages that are covered by
Freescale EULA. To have the right to use those binaries in your images
you need to read and accept the EULA that will be displayed.

LA_OPT27 v4 June 2013

FREESCALE SEMICONDUCTOR SOFTWARE LICENSE AGREEMENT

IMPORTANT. Read the following Freescale Semiconductor Software
License Agreement ("Agreement") completely. By selecting the
```

"I Accept" button at the end of this page, you indicate that
you accept the terms of this Agreement. You may then
download the file.

This is a legal agreement between you, as an authorized representative
of your employer (together "you"), and Freescale Semiconductor, Inc.
("Freescale") and its Affiliates. It concerns your rights to use this
software and any accompanying written documentation (the "Licensed
Software"). In consideration for Freescale allowing you to access
the Licensed Software, you are agreeing to be bound by the terms of
this Agreement. If you do not agree to all of the terms of this
Agreement, do not download the Licensed Software. If at any point
you no longer agree to all the terms of this Agreement, stop using
the Licensed Software immediately and delete all copies of the
Licensed Software in your possession or control. Any copies of the
Licensed Software that you have already distributed, where permitted,
and that have not been destroyed, will continue to be governed by
this Agreement. Your prior use of the Licensed Software will also
continue to be governed by this Agreement.

... 생 략 ....

(a) to prepare derivative works, only as part of, or integrated
within, Authorized Systems and not on a stand alone basis, of the
Licensed Software;

Do you accept the EULA you just read? (y/n) **y**
EULA has been accepted.

Welcome to Freescale Community BSP

The Yocto Project has extensive documentation about OE including a

```
reference manual which can be found at:
    http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
    http://www.openembedded.org/

You can now run 'bitbake <target>'

Common targets are:
    core-image-minimal
    meta-toolchain
    meta-toolchain-sdk
    adt-installer
    meta-ide-support

Your build environemnt has been configured with:

    MACHINE=imx6qsabresd
    SDKMACHINE=i686
    DISTRO=poky
    EULA=1
```

5) "bitbake" 명령으로 컴파일을 진행하게 되는데, 아래와 같이 에러가 발생 됩니다.

```
$ bitbake fsl-image-fb
ERROR: Fetcher failure for URL: 'http://cgit.freedesktop.org/~whot/evtest/snapshot/evtest-1.25.tar.bz2;name=archive'. Checksum mismatch!
File: '/home/treego/PROJECT/temp2/wisol/freescale-imx6/work/Yocto/fsl-release-bsp/downloads/evtest-1.25.tar.bz2' has md5 checksum 0ef3fe5e20fa2dee8994827d48482902 when 770d6af03affe976bdbe3ad1a922c973 was expected
File: '/home/treego/PROJECT/temp2/wisol/freescale-imx6/work/Yocto/fsl-release-bsp/downloads/evtest-1.25.tar.bz2' has sha256 checksum 6e93ef54f0aa7d263f5486ce4a14cac53cf50036bfd20cf045fef2b27ee6664b when 3d34123c68014dae6f7c19144ef79ea2915fa7a2f89ea35ca375a9cf9e191473 was expected
```

```
If this change is expected (e.g. you have upgraded to a new version without updating the
checksums) then you can use these lines within the recipe:
SRC_URI[archive.md5sum] = "0ef3fe5e20fa2dee8994827d48482902"
SRC_URI[archive.sha256sum] =
"6e93ef54f0aa7d263f5486ce4a14cac53cf50036bfd20cf045fef2b27ee6664b"
Otherwise you should retry the download and/or check with upstream to determine if the file has
become corrupted or otherwise unexpectedly modified.

ERROR: Function failed: Fetcher failure for URL:
'http://cgit.freedesktop.org/~whot/evtest/snapshot/evtest-1.25.tar.bz2;name=archive'. Unable to fetch
URL from any source.
ERROR: Logfile of failure stored in: /home/treego/PROJECT/temp2/wisol/freescale-
imx6/work/Yocto/fsl-release-bsp/imx6q-yocto/tmp/work/cortexa9hf-vfp-neon-poky-linux-
gnueabi/evtest/1.25-r0/temp/log.do_fetch.13688
ERROR: Task 1785 (/home/treego/PROJECT/temp2/wisol/freescale-imx6/work/Yocto/fsl-release-
bsp/sources/meta-openembedded/meta-oe/recipes-support/evtest/evtest_1.25.bb, do_fetch) failed
with exit code '1'
NOTE: Tasks Summary: Attempted 2363 tasks of which 642 didn't need to be rerun and 1 failed.
Waiting for 0 running tasks to finish:
```

6) evtest 패키지가 해당 URL에서 받아오질 못해서 생기는 에러입니다. 아래 명령으로 다른 위치에서 다운받아서 처리합니다.

```
$ wget http://dev.gateworks.com/sources/evtest-1.25.tar.bz2
$ cp evtest-1.25.tar.bz2 /home/treego/PROJECT/temp2/wisol/freescale-imx6/work/Yocto/fsl-release-
bsp/downloads/evtest-1.25.tar.bz2
$ touch /home/treego/PROJECT/temp2/wisol/freescale-imx6/work/Yocto/fsl-release-
bsp/downloads/evtest-1.25.tar.bz2.done
```

7) 다시 컴파일을 진행 합니다.

```
$ bitbake fsl-image-fb
Loading cache: 100%
|#################################################################################
##| ETA:  00:00:00
Loaded 2005 entries from dependency cache.
Parsing recipes: 100%
|#################################################################################|
Time: 00:00:00
```

```
Parsing of 1605 .bb files complete (1604 cached, 1 parsed). 2004 targets, 160 skipped, 0 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies
NOTE: multiple providers are available for runtime libgl-mesa-dev (mesa, mesa-gl)
NOTE: consider defining a PREFERRED_PROVIDER entry to match libgl-mesa-dev
NOTE: multiple providers are available for jpeg (jpeg, libjpeg-turbo)
NOTE: consider defining a PREFERRED_PROVIDER entry to match jpeg

Build Configuration:
BB_VERSION           = "1.20.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "Ubuntu-12.04"
TARGET_SYS           = "arm-poky-linux-gnueabi"
MACHINE              = "imx6qsabresd"
DISTRO               = "poky"
DISTRO_VERSION       = "1.5.1"
TUNE_FEATURES        = "armv7a vfp neon callconvention-hard cortexa9"
TARGET_FPU           = "vfp-neon"
meta
meta-yocto           = "(nobranch):bee7e3756adf70edaeabe9d43166707aab84f581"
meta-oe              = "(nobranch):eb4563b83be0a57ede4269ab19688af6baa62cd2"
meta-fsl-arm         = "(nobranch):af392c22bf6b563525ede4a81b6755ff1dd2c1c6"
meta-fsl-arm-extra = "(nobranch):07ad83db0fb67c5023bd627a61efb7f474c52622"
meta-fsl-demos       = "(nobranch):5a12677ad000a926d23c444266722a778ea228a7"
meta-fsl-arm
meta-fsl-demos       = "(nobranch):16c911d80ade96702e3c42ce97f1d69069576bdc"
meta-browser         = "(nobranch):fc3969f63bda343c38c40a23f746c560c4735f3e"
meta-gnome
meta-networking      = "(nobranch):eb4563b83be0a57ede4269ab19688af6baa62cd2"

NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
WARNING: lttng-modules: no modules were created; this may be due to CONFIG_TRACEPOINTS not being enabled in your kernel.
WARNING: Failed to fetch URL
http://ftp.de.debian.org/debian/pool/main/m/mklibs/mklibs_0.1.38.tar.gz, attempting MIRRORS if available
```

```
WARNING: nbench-byte: No generic license file exists for: freely in any provider
WARNING: nbench-byte: No generic license file exists for: distributable in any provider
NOTE: Tasks Summary: Attempted 4040 tasks of which 2452 didn't need to be rerun and all
succeeded.

Summary: There were 4 WARNING messages shown.
```

컴파일한 이미지는 **"./imx6q-yocto/tmp/deploy/images/imx6qsabresd/"** 위치에 존재 합니다.

2. 툴 체인 설치 방법

1) 이전에 받은 Yocto 소스 위치에서 " bitbake meta-toolchain" 으로 Toolchain을 다운 받습니다.

```
$ bitbake meta-toolchain
Loading cache: 100%
|#################################################################################
##| ETA:  00:00:00
Loaded 2005 entries from dependency cache.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION           = "1.20.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "Ubuntu-12.04"
TARGET_SYS           = "arm-poky-linux-gnueabi"
MACHINE              = "imx6qsabresd"
DISTRO               = "poky"
DISTRO_VERSION       = "1.5.1"
TUNE_FEATURES        = "armv7a vfp neon callconvention-hard cortexa9"
TARGET_FPU           = "vfp-neon"
meta
meta-yocto           = "(nobranch):bee7e3756adf70edaeabe9d43166707aab84f581"
meta-oe              = "(nobranch):eb4563b83be0a57ede4269ab19688af6baa62cd2"
meta-fsl-arm         = "(nobranch):af392c22bf6b563525ede4a81b6755ff1dd2c1c6"
```

```
meta-fsl-arm-extra = "(nobranch):07ad83db0fb67c5023bd627a61efb7f474c52622"
meta-fsl-demos     = "(nobranch):5a12677ad000a926d23c444266722a778ea228a7"
meta-fsl-arm
meta-fsl-demos     = "(nobranch):16c911d80ade96702e3c42ce97f1d69069576bdc"
meta-browser       = "(nobranch):fc3969f63bda343c38c40a23f746c560c4735f3e"
meta-gnome
meta-networking    = "(nobranch):eb4563b83be0a57ede4269ab19688af6baa62cd2"

NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
WARNING: QA Issue: gcc-cross-canadian-arm: found library in wrong location:
/opt/poky/1.5.1/sysroots/x86_64-pokysdk-linux/usr/libexec/arm-poky-linux-gnueabi/gcc/arm-poky-
linux-gnueabi/4.8.1/liblto_plugin.so.0.0.0
gcc-cross-canadian-arm: found library in wrong location: /opt/poky/1.5.1/sysroots/x86_64-pokysdk-
linux/usr/libexec/arm-poky-linux-gnueabi/gcc/arm-poky-linux-gnueabi/4.8.1/liblto_plugin.so.0
gcc-cross-canadian-arm: found library in wrong location: /opt/poky/1.5.1/sysroots/x86_64-pokysdk-
linux/usr/libexec/arm-poky-linux-gnueabi/gcc/arm-poky-linux-gnueabi/4.8.1/liblto_plugin.so
NOTE: Tasks Summary: Attempted 1827 tasks of which 1059 didn't need to be rerun and all
succeeded.

Summary: There was 1 WARNING message shown.
```

2) 완료가 되면, " tmp/deploy/sdk/" 폴더가 생성이 됩고, "tmp/deploy/sdk/" 폴더위치에 하나의 스크립트 파일이 존재합니다.

3) 스크립트 파일을 실행합니다.

```
$ ./poky-eglibc-x86_64-meta-toolchain-cortexa9hf-vfp-neon-toolchain-1.5.1.sh
Enter target directory for SDK (default: /opt/poky/1.5.1):
You are about to install the SDK to "/opt/poky/1.5.1". Proceed[Y/n]?
[sudo] password for treego:
Extracting SDK...done
Setting it up...done
SDK has been successfully set up and is ready to be used.
```

4) 완료가 되면, "/opt/poky/1.5.1/" 에 SDK 및 toolchain이 설치가 됩니다.

" /opt/poky/1.5.1/sysroots/x86_64-pokysdk-linux/usr/bin/arm-poky-linux-gnueabi" 위치에 arm용 툴체인이 설치가 됩니다.

```
treego@treego-C2SBA:/opt/poky/1.5.1/sysroots/x86_64-pokysdk-linux/usr/bin/arm-poky-linux-gnueabi$ ls
arm-poky-linux-gnueabi-addr2line  arm-poky-linux-gnueabi-gcc-ar      arm-poky-linux-gnueabi-nm
arm-poky-linux-gnueabi-ar         arm-poky-linux-gnueabi-gcc-nm      arm-poky-linux-gnueabi-objcopy
arm-poky-linux-gnueabi-as         arm-poky-linux-gnueabi-gcc-ranlib  arm-poky-linux-gnueabi-objdump
arm-poky-linux-gnueabi-c++filt    arm-poky-linux-gnueabi-gcov        arm-poky-linux-gnueabi-ranlib
arm-poky-linux-gnueabi-cpp        arm-poky-linux-gnueabi-gdb         arm-poky-linux-gnueabi-readelf
arm-poky-linux-gnueabi-elfedit    arm-poky-linux-gnueabi-gprof       arm-poky-linux-gnueabi-size
arm-poky-linux-gnueabi-g++        arm-poky-linux-gnueabi-ld          arm-poky-linux-gnueabi-strings
arm-poky-linux-gnueabi-gcc        arm-poky-linux-gnueabi-ld.bfd      arm-poky-linux-gnueabi-strip
```

3. Kernel, uboot 소스 추출 및 컴파일 방법

3.1. U-boot

1) uboot소스 위치는 아래에 있습니다. 소스 파일을 원하는 위치에 복사를 합니다.

Uboot 소스 위치 **"imx6q-yocto/tmp/work/imx6qsabresd-poky-linux-gnueabi/u-boot-imx/2013.04-r0/git "**

2 복사한 u-boot 소스 위치에서, "build_uboot" 스크립트 파일을 생성 합니다.

```
$ vi build_uboot
#!/bin/sh

export ARCH=arm
export CROSS_COMPILE=arm-poky-linux-gnueabi-
export PATH=/opt/poky/1.5.1/sysroots/x86_64-pokysdk-linux/usr/bin/arm-poky-linux-gnueabi/:$PATH

KERNEL_IMAGE=uImage

# Default kernel configurations
KERNEL_CONFIG=$2
```

```
INSTALL_BINDIR=../image
export LOADADDR=0x10008000
DTB_FILENAME=imx6q-sabresd.dtb

#{
CPU_JOB_NUM=$(grep processor /proc/cpuinfo | awk '{field=$NF};END{print field+2}')
START_TIME=`date +%s`
#sudo apt-get install libncurses5-dev
case "$1" in
clean)
	echo make -j$CPU_JOB_NUM mrproper
	cp .config arch/arm/configs/$KERNEL_CONFIG
	cp .config config.sav
	make -j$CPU_JOB_NUM mrproper
	;;
config)
	sudo apt-get install libncurses5-dev
	echo make -j$CPU_JOB_NUM menuconfig
	make -j$CPU_JOB_NUM menuconfig
	;;
defconfig)
	if [ "$2" ] ; then
		KERNEL_CONFIG=$2
	fi
	echo make -j$CPU_JOB_NUM $KERNEL_CONFIG
	make -j$CPU_JOB_NUM $KERNEL_CONFIG
	;;
all|*)
	echo make -j$CPU_JOB_NUM uImage
	make -j$CPU_JOB_NUM uImage
	make $DTB_FILENAME
	if [ $? != 0 ] ; then
		exit 1
	fi
	if [ "$2" ] ; then
		INSTALL_BINDIR=$2
	fi
	if [ $INSTALL_BINDIR ] ; then
```

```
        echo cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR/$KERNEL_IMAGE
        cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR/$KERNEL_IMAGE
        echo cp -a arch/arm/boot/dts/$DTB_FILENAME $INSTALL_BINDIR/$KERNEL_IMAGE
        cp -a arch/arm/boot/dts/$DTB_FILENAME $INSTALL_BINDIR/$KERNEL_IMAGE
    fi
    if [ $INSTALL_BINDIR2 ] ; then
        echo cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR2/$KERNEL_IMAGE
        cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR2/$KERNEL_IMAGE
    fi
     ;;
 esac

 END_TIME=`date +%s`
 echo "Total compile time is $(($END_TIME-$START_TIME)/60)) minutes $(($END_TIME-
$START_TIME)%60)) seconds"
 #} 2>&1 |tee b.out
```

3 만든 스크립트를 이용해서, 컴파일을 합니다.

```
$ ./build_uboot config
$ ./build_uboot
```

3.2. Kernel

1) kernel소스 위치는 아래에 있습니다. 소스 파일을 원하는 위치에 복사를 합니다.

Kernel 소스 위치 **"imx6q-yocto/tmp/work/imx6qsabresd-poky-linux-gnueabi/linux-imx/3.10.17-r0/git/"**

2) 복사한 Kernel 위치에서, "build_kernel" 스크립트 파일을 생성 합니다.

```
$ vi build_kernel
#!/bin/sh

export ARCH=arm
export CROSS_COMPILE=arm-poky-linux-gnueabi-
export PATH=/opt/poky/1.5.1/sysroots/x86_64-pokysdk-linux/usr/bin/arm-poky-linux-gnueabi/:$PATH

KERNEL_IMAGE=uImage
```

```
# Default kernel configurations
KERNEL_CONFIG=$2

INSTALL_BINDIR=../image

#{
CPU_JOB_NUM=$(grep processor /proc/cpuinfo | awk '{field=$NF};END{print field+2}')
START_TIME=`date +%s`
#sudo apt-get install libncurses5-dev
case "$1" in
clean)
	echo make -j$CPU_JOB_NUM mrproper
	cp .config arch/arm/configs/$KERNEL_CONFIG
	cp .config config.sav
	make -j$CPU_JOB_NUM mrproper
	;;
config)
	sudo apt-get install libncurses5-dev
	echo make -j$CPU_JOB_NUM menuconfig
	make -j$CPU_JOB_NUM menuconfig
	;;
defconfig)
	if [ "$2" ] ; then
		KERNEL_CONFIG=$2
	fi
	echo make -j$CPU_JOB_NUM $KERNEL_CONFIG
	make -j$CPU_JOB_NUM $KERNEL_CONFIG
	;;
all|*)
	echo make -j$CPU_JOB_NUM uImage
	make -j$CPU_JOB_NUM uImage
	if [ $? != 0 ] ; then
		exit 1
	fi
	if [ "$2" ] ; then
		INSTALL_BINDIR=$2
	fi
```

```
    if [ $INSTALL_BINDIR ] ; then
        echo cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR/$KERNEL_IMAGE
        cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR/$KERNEL_IMAGE
    fi
    if [ $INSTALL_BINDIR2 ] ; then
        echo cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR2/$KERNEL_IMAGE
        cp -a arch/arm/boot/$KERNEL_IMAGE $INSTALL_BINDIR2/$KERNEL_IMAGE
    fi

    ;;
esac

END_TIME=`date +%s`
echo "Total compile time is $(($END_TIME-$START_TIME)/60)) minutes $(($END_TIME-$START_TIME)%60)) seconds"
```

3) 컴파일을 하면, 에러가 발생이 됩니다.

```
$ ./build_kernel defconfig imx_v7_defconfig
$ ./build_kernel
  CHK     include/generated/uapi/linux/version.h
  CHK     include/generated/utsrelease.h
make[1]: `include/generated/mach-types.h'는 이미 갱신되었습니다.
  CC      scripts/mod/devicetable-offsets.s
  CALL    scripts/checksyscalls.sh
  GEN     scripts/mod/devicetable-offsets.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTLD  scripts/mod/modpost
  CHK     include/generated/compile.h
  CHK     kernel/config_data.h
  Kernel: arch/arm/boot/Image is ready
  LZO     arch/arm/boot/compressed/piggy.lzo
  AS      arch/arm/boot/compressed/piggy.lzo.o
  LD      arch/arm/boot/compressed/vmlinux
  OBJCOPY arch/arm/boot/zImage
  Kernel: arch/arm/boot/zImage is ready
multiple (or no) load addresses:
This is incompatible with uImages
```

```
Specify LOADADDR on the commandline to build an uImage
make[1]: *** [arch/arm/boot/uImage] 오류 1
make: *** [uImage] 오류 2
```

4) uImage는 zImage와 다르게, 커널 로딩 주소를 갖는데, 정의가 되지 않아서 생기는 에러입니다. 아래 파일 하나를 생성 하면 됩니다.

```
$ vi arch/arm/mach-imx/Makefile.boot
zreladdr-$(CONFIG_SOC_IMX6Q) := 0x10008000
params_phys-$(CONFIG_SOC_IMX6Q) := 0x10000100
initrd_phys-$(CONFIG_SOC_IMX6Q) := 0x10800000
```

4. Image Write 방법

1) Micro SD를 삽입하고, 장치 이름을 확인 합니다.

```
[ 7456.857578] sd 4:0:0:0: [sdc] 15644672 512-byte logical blocks: (8.01 GB/7.45 GiB)
[ 7456.859072] sd 4:0:0:0: [sdc] No Caching mode page present
[ 7456.859076] sd 4:0:0:0: [sdc] Assuming drive cache: write through
[ 7456.861822] sd 4:0:0:0: [sdc] No Caching mode page present
[ 7456.861825] sd 4:0:0:0: [sdc] Assuming drive cache: write through
[ 7456.863091]  sdc:
```

2) 아래 명령으로 SD에 이미지 파일을 write할 수 있습니다.

```
sudo dd if=u-boot.imx of=/dev/sdc bs=512 seek=2; sync
sudo dd if=uImage of=/dev/sdc bs=512 seek=2048 conv=fsync
sudo dd if=imx6q-sabresd.dtb of=/dev/sdc bs=512 seek=20480 conv=fsync
```

Rootfs 만드는 소스를 찾지 못해서, Yocto 프로젝트 컴파일시에 나온 rootfs 이미지 파일을 복사해서, write를 진행 했습니다.

3) sd에 파티션을 설정합니다.

```
Command (m for help): p

Disk /dev/sdc: 7948 MB, 7948206080 bytes
```

```
245 heads, 62 sectors/track, 1021 cylinders, total 15523840 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xd29dc410

   Device Boot      Start         End      Blocks   Id  System

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-15523839, default 2048): 40000
Last sector, +sectors or +size{K,M,G} (40000-15523839, default 15523839):
Using default value 15523839

Command (m for help): p

Disk /dev/sdc: 7948 MB, 7948206080 bytes
245 heads, 62 sectors/track, 1021 cylinders, total 15523840 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xd29dc410

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1           40000    15523839     7741920   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

4) 설정한 파티션을 포멧 합니다.

```
$ sudo mkfs.ext4 /dev/sdc1
mke2fs 1.42 (29-Nov-2011)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
484800 inodes, 1935480 blocks
96774 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1983905792
60 block groups
32768 blocks per group, 32768 fragments per group
8080 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

5) 복사한 rootfs 이미지 파일을 write를 합니다.

```
$ mkdir temp
$ sudo mount /dev/sdc1 temp
$ sudo tar xf fsl-image-fb-imx6qsabresd-20140925044842.rootfs.tar.bz2 -C temp/
$ sync
$ sudo umount /dev/sdc1
```

6) SD로 부팅 후, uboot prompt 상에서 환경변수를 수정 합니다.

```
U-Boot > setenv loadaddr 0x12000000
U-Boot > setenv fdt_addr 0x18000000
U-Boot > setenv fdt_high 0xffffffff
U-Boot > setenv bootargs_base 'setenv bootargs console=ttymxc0,115200'
U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/mmcblk0p1 init=/sbin/init
rootwait rw video=mxcfb1:dev=ldb,LDB-XGA,if=RGB666
```

```
video=mxcfb0:dev=hdmi,1920x1080M@60,if=RGB24'
U-Boot > setenv bootcmd_mmc 'run bootargs_base bootargs_mmc;mmc dev 1;mmc read ${loadaddr}
0x800 0x8000;mmc read ${fdt_addr} 0x5000 0x800;bootm ${loadaddr} - ${fdt_addr}'
U-Boot > setenv bootcmd 'run bootcmd_mmc'
U-Boot > saveenv
U-Boot > run bootcmd
```