EDM Yocto 1.7 Pre-Built Image User's Guide

Rev 1.2 20160303

Contents

1. Environment Requirement

1.1 Supported hardware

These are the systems covered in this guide:

System-on-Modules:

- EDM1-CF-IMX6
- EDM1-CF-IMX6SX
- EDM2-CF-IMX6
- PICO-IMX6

Carrier Boards:

- EDM1-FAIRY
- EDM1-GOBLIN
- EDM2-ELF
- Toucan-0700
- PICO-DWARF
- PICO-HOBBIT

Box industrial PC:

- TEK3-IMX6

1.2 Software version

name	version
u-boot	2015.04
linux kernel	3.14.52
Yocto	1.7 (dizzy)

1.3 Host setup

The build process is tested under Ubuntu-12.04 64bit. So we recommend to set up ubuntu-12.04 environment for building yocto. In building yocto image process, it may take about 75GB hard disk space.

Install Yocto Project host packages:

```
sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib \
 build-essential chrpath  socat \
 libsdl1.2-dev xterm  sed cvs subversion coreutils texi2html \
 docbook-utils python-pysqlite2 help2man make gcc g++ desktop-file-utils \
 libgl1-mesa-dev libglu1-mesa-dev mercurial autoconf automake groff curl lzop asciidoc
```

EDM layers host packages for Ubuntu 12.04 host setup only:

```
sudo apt-get install uboot-mkimage
```

EDM layers host packages for Ubuntu 14.04 host setup only:

```
sudo apt-get install u-boot-tools
```

2. Get EDM Yocto BSP Source Code

There are two ways that you can get EDM Yocto BSP source code.

1. From Technexion website:
http://www.technexion.com/support/download-center/edm/edm1-cf-imx6

In Yocto section, download the source tarball. There are already pre-downloaded source packages in the "downloads" folder inside the source tarball.

2. From Technexion github:
https://github.com/TechNexion/edm-yocto-bsp

To get the BSP you need to have "repo" installed. Install the "repo" utility:

```
mkdir ~/bin
curl http://commondatastorage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
chmod a+x ~/bin/repo
```

Download the BSP source:

```
PATH=${PATH}:~/bin
mkdir edm_yocto
cd edm_yocto
repo init -u https://github.com/TechNexion/edm-yocto-bsp.git -b dizzy_3.14.52-1.1.0_GA
repo sync
```

To speed up the download process, you can add "-j8" after "repo sync", e.g. "repo sync -j8".

3. Build EDM Yocto Image

3.1 Start an image build:

QT5 with X11 image for **edm-fairy-imx6 with HDMI output**:

```
DISPLAY=hdmi720p MACHINE=edm-fairy-imx6 source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

QT5 with X11 image for **edm-fairy-imx6 with 7 inch LVDS panel**:

```
DISPLAY=lvds7 MACHINE=edm-fairy-imx6 source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

QT5 with X11 image for **edm-toucan-imx6 with 7 inch LVDS panel**:

```
DISPLAY=lvds7 MACHINE=edm-toucan-imx6 source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

QT5 with X11 image for **edm-goblin-imx6sx with 7 inch LVDS panel**:

```
MACHINE=edm-goblin-imx6sx source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

Note:
1. Because i.mx6sx lacks VPU, the freescale proprietory video decoder can't be used in video playback.
2. edm-goblin-imx6sx doesn't support HDMI output, and it only supports to output to LVDS 7-inch panel now.

QT5 with X11 image for **picosom-dwarf-imx6 with HDMI output**:

```
MACHINE=picosom-dwarf-imx6 source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

QT5 with X11 image for **picosom-dwarf-imx6 with 7 inch LVDS panel:**:

```
DISPLAY=lvds7 MACHINE=picosom-dwarf-imx6 source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

QT5 with X11 image for **tek3-imx6 with HDMI output**:

```
MACHINE=tek3-imx6 source edm-setup-release.sh -b build-x11 -e x11

bitbake fsl-image-qt5
```

3.2 Choosing a EDM Yocto project image

The following bitbake target images are available:

Image name	Target
core-image-minimal	A small image that only allows a device to boot.

core-image-base	A console-only image that fully supports the target device hardware.
core-image-sato	An image with Sato, a mobile environment and visual style for mobile devices. The image supports X11 with a Sato theme, Pimlico applications. It contains a terminal, an editor and a file manager.
fsl-image-machine-test	An FSL Community i.MX core image with console environment - no GUI interface
fsl-image-gui	Builds a Freescale image with a GUI without any QT content. This image recipe works on all backends for X11, DirectFB, Frame Buffer and Wayland
fsl-image-qt5	Builds a QT5 image for X11, Frame Buffer and Wayland backends

3.3 Parameters for setup script

"MACHINE" is the target Technexion hardware platform.

"DISPLAY" is the disply type.

"-b" specify the build directory.

"-e" sets the graphical back end for frame buffer and direct fb images. X11 is default if no backend is set.

paramerter	**Available options**
MACHINE	edm-fairy-imx6
	edm-toucan-imx6
	edm-goblin-imx6sx
	picosom-dwarf-imx6
	tek3-imx6
DISPLAY	lvds7
	hdmi720p
	hdmi1080p
	lcd
	lvds7_hdmi720p
	custom
-b	<build dir>
-e	fb

	dfb
	wayland
	x11

Every time after you change the display settings:

You need to clean the target build first:

```
bitbake -c clean fsl-image-qt5
cd ../
DISPLAY=lvds7 MACHINE=edm-fairy-imx6 source edm-setup-release.sh -b build-x11 -e x11
```

When you issue the “bitbake” command, you need to make sure the present directory is “build” directory.

If the build process hangs on fetching some packages, please terminate the existing build process then restart it.

4. Image Deployment

After build succeeds, the generated release image is under **“build-x11/tmp/deploy/images/<MACHINE>”:**

```
fsl-image-qt5-edm-toucan-imx6.ext3
fsl-image-qt5-edm-toucan-imx6.manifest
fsl-image-qt5-edm-toucan-imx6.sdcard
fsl-image-qt5-edm-toucan-imx6.tar.bz2
```

4.1 Flash image into SD card

An SD card image provides the full system to boot with U-Boot and kernel. To flash an SD card image, run the following command:

```
sudo dd if=<image name>.sdcard  of=/dev/sd<partition> bs=1M && sync
```

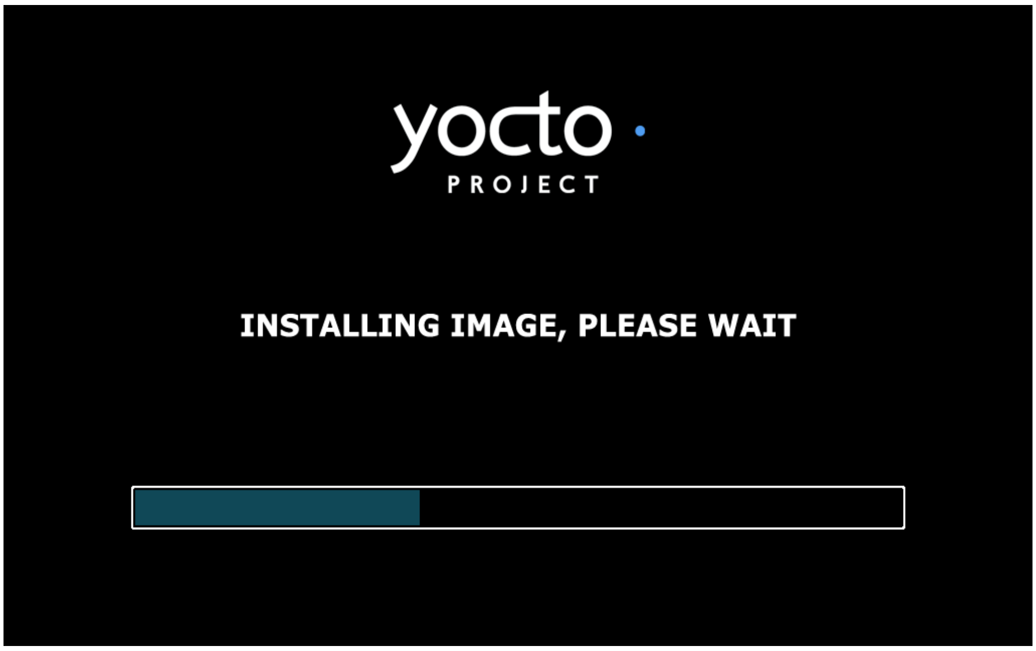
4.2 Flash image into eMMC

There are two ways to flash image into eMMC.

1. Use installer card to automatically install image into eMMC:

The behavior is like the pre-built image. Set up the hardware boot mode, then insert the SD card.

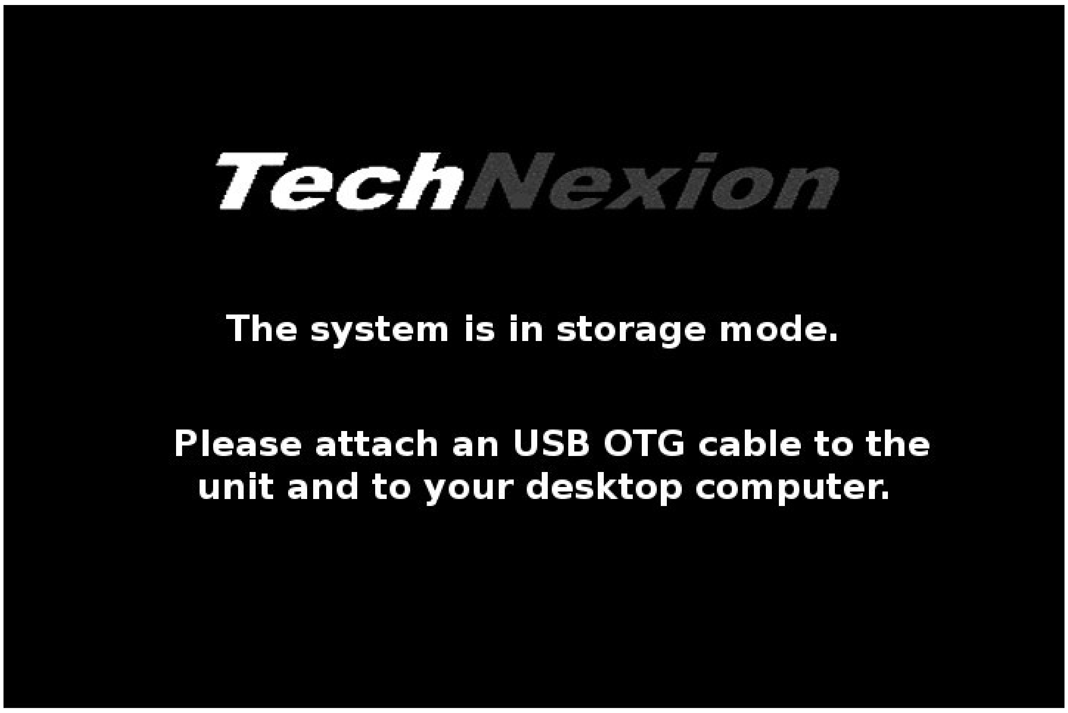
The installing process will automatically start. Please follow the document "**General_Installer_User_Guide.pdf**" in section "**5.3 Automatic mode**". This method is useful when you need to deploy for mass production.



INSTALLING IMAGE, PLEASE WAIT

2. Use generic installer card to boot into USB OTG storge mode:

Please follow the document "**General_Installer_User_Guide.pdf**" in section "**5.2 Storage mode**".



The system is in storage mode.

Please attach an USB OTG cable to the unit and to your desktop computer.

This method is convenient when you are in developing stage. This mode can let you manipulate eMMC as USB storage.

5. Customize the image release

5.1 Change the default audio output

The default audio output for target image is SGTL5000. You can change it to HDMI audio or SPDIF.

vim sources/meta-edm-bsp-release/recipes-multimedia/pulseaudio/pulseaudio/default.pa

```
#set-default-sink alsa_output.platform-sound-hdmi.25.analog-stereo
set-default-sink alsa_output.platform-sound.23.analog-stereo
#set-default-sink alsa_output.platform-sound-spdif.24.analog-stereo
```

5.2 Change the image size

$ vim conf/local.conf

```
MACHINE ??= 'edm-fairy-imx6'
DISTRO ?= 'poky'
PACKAGE_CLASSES ?= "package_rpm"
EXTRA_IMAGE_FEATURES = "debug-tweaks"
USER_CLASSES ?= "buildstats image-mklibs image-prelink"
PATCHRESOLVE = "noop"
BB_DISKMON_DIRS = "\
    STOPTASKS,${TMPDIR},1G,100K \
    STOPTASKS,${DL_DIR},1G,100K \
    STOPTASKS,${SSTATE_DIR},1G,100K \
    ABORT,${TMPDIR},100M,1K \
    ABORT,${DL_DIR},100M,1K \
    ABORT,${SSTATE_DIR},100M,1K"
PACKAGECONFIG_append_pn-qemu-native = " sdl"
PACKAGECONFIG_append_pn-nativesdk-qemu = " sdl"
ASSUME_PROVIDED += "libsdl-native"
CONF_VERSION = "1"

BB_NUMBER_THREADS = '8'
PARALLEL_MAKE = '-j 8'

DL_DIR ?= "${BSPDIR}/downloads/"
ACCEPT_FSL_EULA = "1"

DISPLAY_TYPE = "hdmi720p"

IMAGE_ROOTFS_SIZE = "3000000"
```

5.3 Change the kernel configuration

```
bitbake -c menuconfig virtual/kernel

cp tmp/work/edm_fairy_imx6-poky-linux-gnueabi/linux-edm-fairy/3.10.53-r0/git/.config ../sources/meta-edm-bsp-release/recipes-kernel/linux/linux-edm-fairy-3.10.53/defconfig

bitbake -c cleansstate virtual/kernel

bitbake fsl-image-qt5
```

6. Create the toolchain for cross-compiling

Bitbake a poky toolchain:

```
bitbake meta-toolchain
```

Install the toolchain in host PC:

Run the installation script located in “build-x11/tmp/deploy/sdk”

```
sh poky-glibc-x86_64-meta-toolchain-cortexa9hf-vfp-neon-toolchain-1.7.sh
```

Compile the C file:

```
source /opt/poky/1.7/environment-setup-cortexa9hf-vfp-neon-poky-linux-gnueabi

$CC hello_arm_world.c
```