BUILD GUIDE

Display Evaluation Kit

K_BeagleBone

For
4.0", 4.7", 4.9", 7.9", 10.7" and 11.5" Plastic Logic Displays

Part No.: 303003, 303005, 303007, 303011

Containing Part No. 301004 and 301054 or 301055

Revision 1
February 15th 2018

Revision Status	Date	Author	Reason of Modification
1	15-Feb-2018	RP	Initial Version

1 Contents

2 About this document

This document is a build guide for the **Display Evaluation Kit software components**. It is only to be used for kits containing a BeagleBone. It is intended to give sufficient information to:

1. Setup a build environment for the user space application epdc-app and compile the app.
2. Setup the build environment to build the Plasticlogic driver module and compile the Linux kernel (only for Falcon 2)
3. Setup an official BeagleBone SD-Card to use with the PLDEK

3 Setting up the Build environment for the user space application epdc-app

The Epdc-app source is intended to be compiled with eclipse CDT with a cross compiler Toolchain of your choice.

- Get the source zip or
  - Locate the epdc-app.zip on the SD-Card. **Hint: epdc/documentation/epdc.zip**
  - Extract the Epdc-app.zip to a folder of your choice and start eclipse CDT.
- Clone the GIT repository from Github

```
$ git clone hhtps://github.com/plasticlogic/pl-bb-epd.git
```

- Import the Project to your workspace
  - choose import existing Projects into Workspace
- Open the Project Properties
  - C/C++ Build Environment:
    - Add the PATH variable to the project. Check if your cross compiler is inside the path.
  - C/C++ Build Settings:
    - Set the Cross Settings to arm-Linux-gnueabihf and a path pointing to your cross compiler
    - Add the library shortcuts to Tool Settings / Cross GCC linker / Librarys (-l)
      - freetype
      - m (math)
      - Rt
  - C/C++ Tool Chain Editor:
    - Toolchain: Cross GCC
    - Builder: Gnu Make Builder
  - C/C++ General Paths and Symbols: Add the following folders to the Include Directories:
    - Epdc-app
    - Epdc-app/pl
    - Epdc-app/zlib-1.2.8
    - Epdc-app/freetype
    - Epdc-app/freetype/freetype2
  - C/C++ General Paths and Symbols: Add the following folders to the Include Directories:
    - /Epdc-app/freetype/libfreetype.so
  - Build the Project
    - If the project compiled successfully, a folder with the name “release” (or “debug” – depending on the build settings) appears in the build directory.

4 Compiling the kernel for BeagleBoneBlack

To compile the BBB kernel, a Linux environment is required. We use Ubuntu 1404 with ARM-GCC 4.6

4.1 Using the Plasticlogic kernel source

The source can be found on Github or on the SD-Card shipped with the kit.

```
git clone git://github.com/plasticlogic/linux.git Linux-bbb
cd Linux-bbb/
git checkout pl-omap-3.8.13-beaglebone
```

Or extract the bb-kernel-3.8.tar.gz to a folder of your choice.

Run the build-bbb.sh script.

```
#!/bin/bash -e
CC46="/path/to/gcc/Linux-x86/arm/arm-eabi-4.6/bin/arm-eabi-"
touch kernel/.scmversion && \
make -C kernel BeagleBone_defconfig ARCH=arm CROSS_COMPILE=$CC46 && \
make -C kernel zImage dtbs LOCALVERSION= ARCH=arm CROSS_COMPILE=$CC46 && \
make -C kernel modules LOCALVERSION= ARCH=arm CROSS_COMPILE=$CC46 && \
make -C kernel modules_install INSTALL_MOD_PATH=../modules_3.8
LOCALVERSION= ARCH=arm CROSS_COMPILE=$CC46 && \
cp -f kernel/arch/arm/boot/zImage modules_3.8/zImage && \

echo "Done!"
```

Finally, build the kernel

```
sh build-bbb.sh
```

4.2 Using the beagleboard.org kernel source

The development is based on the 3.8.13 kernel which is available through GIT from beagleboards Github site:

```
git clone git://github.com/beagleboard/kernel.git linux-bbb
cd linux-bbb/
git checkout 3.8
./patch.sh
cp configs/BeagleBone kernel/arch/arm/configs/BeagleBone_defconfig
```

Add the kernel Module dev_parallel to the kernel Source:

Extract the contents of misc.tar.gz to /path/to/kernel/drivers/misc. This will overwrite the Makefile and the Kconfig file. If you changed these two files, those changes might be lost.

The build-bbb.sh script builds the kernel, the device tree files and the kernel modules.

```
#!/bin/bash -e
CC46="/path/to/gcc/Linux-x86/arm/arm-eabi-4.6/bin/arm-eabi-"
```

```
touch kernel/.scmversion && \
make -C kernel beagleBone_pl_defconfig ARCH=arm CROSS_COMPILE=$CC46 && \
make -C kernel zImage dtbs LOCALVERSION= ARCH=arm CROSS_COMPILE=$CC46 && \
make -C kernel modules LOCALVERSION= ARCH=arm CROSS_COMPILE=$CC46 && \
make -C kernel modules_install INSTALL_MOD_PATH=../modules_3.8
LOCALVERSION= ARCH=arm CROSS_COMPILE=$CC46 && \
cp -f kernel/arch/arm/boot/zImage modules_3.8/zImage && \

echo "Done!"
```

Build the kernel

```
sh build-bbb.sh
```

5 Update the software on the BeagleBone

Note: The BaegleBone must be restarted after every update of the kernel or the modules.

5.1 Updating the kernel (modules)

After successful kernel build, the kernel image and the kernel modules can be found in the modules_3.8 folder.

- Copy the zImage (compressed kernel image) to the FAT-Partition of the SD-Card
- Copy the lib folder (modules and firmwares) to the ext4-rootfs-Partition of the SD-Card
- Reboot

If the BeagleBone is active while the kernel and the modules are updated, it is recommended to clear the cache after every copy process. This is especially the recommended if file transfer is performed through the USB-Mass-Storage-Gadget of the BeagleBone.

```
$ sync; echo 3 > /proc/sys/vm/drop_caches
```

5.2 Updating the epdc-app

For updating the epdc-app through the USB-Mass-Storage-Gadget it is also recommended to clear the caches before (and after) each copy process.

- Locate the epdc-app binary in your epdc-app build folder ([workspace]/epdc-app/Debug/epdc-app)
- Copy to /usr/bin on the rootfs partiton of the BeagleBone
- Drop the caches
- Restart the epdc-app