ESPRESSObin Ultra Build Instruction

```
sudo apt-get update
sudo apt-get install make binutils build-essential gcc g++ \
      bash patch gzip bzip2 perl tar cpio python zlib1g-dev \
      gawk ccache gettext libssl-dev libncurses5 minicom git \
      bison flex device-tree-compiler gcc-arm-linux-gnueabi
```

Download the latest source code

You can get the latest ESPRESSObin Ultra source code from our [github](). We provided a batch download with repo command or you also can download the source code manually.

Batch download

1. install the repo utility

```
sudo apt-get install repo
```

2. run repo init <url> -m <mainifest name>

```
# for Linux v4.19.62
repo init -u https://github.com/globalscaletechnologies/manifest.git -b
espressobin-ultra -m espressobin_ultra-1.0.0.xml

# for Linux v4.19.113
repo init -u https://github.com/globalscaletechnologies/manifest.git -b
espressobin-ultra -m espressobin_ultra-1.0.1.xml
```

3. run repo sync to download the git repositories from github

```
repo sync
```

4. after run the 'repo sync' command, you should can find six folders under current directory.

Manually download

1. checkout a3700-utils

```
git clone https://github.com/globalscaletechnologies/A3700-utils-
```

```
marvell.git -b A3700_utils-armada-18.12.0-gti
```

2. checkout atf-marvell

```
git clone https://github.com/globalscaletechnologies/atf-marvell.git -b
atf-v1.5-armada-18.12
```

3. checkout mv-ddr-marvell

```
git clone https://github.com/globalscaletechnologies/mv-ddr-marvell.git
-b mv_ddr-armada-18.12
```

4. checkout u-boot-marvell

```
git clone https://github.com/globalscaletechnologies/u-boot-marvell.git
-b u-boot-2018.03-armada-18.12-gti
```

5. checkout linux-v4.19.62

```
git clone https://github.com/globalscaletechnologies/linux.git -b linux-
4.19.62-gti
```

Get the cross-compiler

For ESPRESSObin Ultra, we use linaro-7.3.1 (aarch64) toolchain, you can download the toolchain from linaro website.

```
wget https://releases.linaro.org/components/toolchain/binaries/7.3-
2018.05/aarch64-linux-gnu/gcc-linaro-7.3.1-2018.05-x86_64_aarch64-linux-
gnu.tar.xz
mkdir toolchain
tar -xvf gcc-linaro-7.3.1-2018.05-x86_64_aarch64-linux-gnu.tar.xz -C
toolchain/
rm gcc-linaro-7.3.1-2018.05-x86_64_aarch64-linux-gnu.tar.xz
```

Build the bootloader

Before start build the bootloader, you need to install "ARM 32-bit cross compiler" which is required by building WTMI image for CM3.

```
sudo apt-get install gcc-arm-linux-gnueabi
```

After install all related software packages, we can start to build the source code, you can follow below steps to build the bootloader image.

Setup environment variables

```
# Set the cross compiler
```

```
export CROSS_COMPILE=/path/to/toolchain/aarch64-linux-gnu-

# Set path for MV_DDR component
export MV_DDR_PATH=path/to/mv_ddr_marvell

# Set U-Boot image path
export BL33=path/to/u-boot.bin

# Set WTP Tools path
export WTP=path/to/A3700_utils_marvell
```

Build u-boot

The u-boot image need to build first before you start build the ATF image, so let us to build it first.

For ESPRESSObin Ultra platform, the setting file as below
    Device Tree : <u-boot>/arch/arm/dts/**armada-3720-ccpe.dts**
    Defconfig : <u-boot>/configs/**gti_ccpe-88f3720_defconfig**

```
make -C <u-boot path> gti_ccpe-88f3720_defconfig
make -C <u-boot path> DEVICE_TREE=armada-3720-ccpe
```

Build ATF

The ATF has many parameter need to configure, below are major parameters
- DEBUG : debug information enable or not.
- DDR_TOPOLOGY : DDR topology map index, you can see the <a3700_utils_marvell path>/tim/ddr/DDR_TOPOLOGY_<index>.txt file to make sure the DDR setting.
- CLOCKSPRESET : The CPU and DDR clock setting, current support CPU_600_DDR_600 / CPU_800_DDR_800 / CPU_1000_DDR_800 / CPU_1200_DDR_750
- BOOTDEV : The flash boot device, current support SPINOR / SPINAND / EMMCNORM / SATA, default is SPINOR.

For ESPRESSObin Ultra platform, the default build command as below

```
# build for CPU-1GHz DDR-1GB (single chip-select)
make -C <atf path> DEBUG=0 USE_COHERENT_MEM=0 LOG_LEVEL=20
CLOCKSPRESET=CPU_1000_DDR_800 PLAT=a3700 DDR_TOPOLOGY=5 all fip

# build for CPU-1.2GHz DDR-1GB (single chip-select)
```

```
make -C <atf path> DEBUG=0 USE_COHERENT_MEM=0 LOG_LEVEL=20
CLOCKSPRESET=CPU_1200_DDR_750 PLAT=a3700 DDR_TOPOLOGY=5 all fip
```

The final image is output to

- For non-debug build
  <atf path>/build/a3700/release/**flash-image.bin**
- For debug enabled build
  <atf path>/build/a3700/debug/**flash-image.bin**

PS. when you build the atf image, the build script also generate an uart-boot image on the same folder, this image can be recovery the device via UART when you bricked your device. The recovery method you can reference the ESPRESSObin wiki page ([Bootloader recovery via UART](#)).

Build the kernel

The current ESPRESSObin Ultra kernel is based on **Linux longterm support** version v4.19.y.

For ESPRESSObin Ultra platform, the setting file as below
  Device Tree : <linux path>/arch/arm64/boot/dts/marvell/armada-3720-ccpe.dts
  Defconfig : <linux path>/arch/arm64/configs/gti_ccpe-88f3720_defconfig

The default build command as below

```
# build kernel and modules image
make -C <linux path> gti_ccpe-88f3720_defconfig
make -C <linux path> -j4

# install modules to target path
make -C <linux path> modules_install INSTALL_MOD_PATH=<module output
path>
```

Build image with gti build script file

We also provided a build script to build ESPRESSObin Ultra image. When you use 'repo sync' to batch download source code, it also checkout a build script "env-espressobin_ultra.sh".

If you download source manually, not using 'repo sync', you also can flow below steps to download buld script.

```
git clone https://github.com/globalscaletechnologies/build.git -b
espressobin_ultra

ln -s build/espressobin_ultra/env-setup.sh env-espressobin_ultra.sh
```

Now you can execute below commands to build the bootloader and kernel image.

```
source env-espressobin_ultra.sh
# build bootloader image
gtibuild bootloader

# build linux kernel and modules image
gtibuild kernel

# build bootloader & kernel
gtibuild all
```

The final output image is output to ./out/build-<yyyymmdd-dbg>/

Update bootloader image

You can follow below steps to update the ESPRESSObin Ultra bootloader image.
1. Copy <atf path>/build/a3700/release/**flash-image.bin** to usb disk and insert usb disk to ESPRESSObin Ultra. If you use gti build script to build bootloader image, you can find the bootloader image under ./out/build-<yyyymmdd-dbg>/espressobin_ultra-bootloader-*-dbg.bin.
2. Power on the device and press any key on uart debug console during booting
3. execute "bubt flash-image.bin spi usb" to update u-boot image
4. reset device to load new bootloader image

Update Ubuntu rootfs image

We provided a prebuilt Ubuntu 18.04 rootfs for ESPRESSObin Ultra, you can download the prebuilt image from our official website.

You can follow below steps to update your device.

1. download our prebuilt image from our official website
2. prepare a usbdisk and format it to FAT/FAT32 filesystem.
3. copy all files under the recovery folder to usbdisk.
4. copy all files under the bootloader and rootfs to usbdisk.
5. insert usbdisk to ESPRESSObin Ultra board and then power on device.
6. press any key on uart debug console during booting to enter u-boot console.
7. run below command on u-boot console to update bootloader, the image must fit your device hardware model and memory size. If you don't want to update bootloader, you can ignore this step.

```
bubt espressobin_ultra-bootloader-cpu-1000-ddr4-<xxx>-rel.bin spi usb
```

8. run below command to load ramdisk image and boot it

```
usb reset; fatload usb 0 $loadaddr recovery.bin; source $loadaddr
```

9. when device boot into system, run below commands to install ESPRESSObin Ultra rootfs image to eMMC.

```
mount /dev/sda1 /media
/root/mkemmc.sh /media/espressobin_ultra-rootfs-ubuntu1804-<xxx>.tar.gz
```

10. after update completed, reboot the system and enter u-boot console again.
11. run below command to setup the device boot from eMMC.

```
env default -a
setenv image_name boot/Image
setenv fdt_name boot/armada-3720-ccpe.dtb
setenv bootcmd 'mmc dev 0; ext4load mmc 0:1 $kernel_addr_r
$image_name;ext4load mmc 0:1 $fdt_addr_r $fdt_name;setenv bootargs
$console root=PARTUUID=89708921-01 rw rootwait net.ifnames=0
biosdevname=0; booti $kernel_addr_r - $fdt_addr_r'
saveenv
boot
```