Procedure for Loading System Software onto Low-Power EA3141 based ESP CPU boards

Revised: 10/10/17 brent@mbari.org

Background:

The Embedded Artists LPC3141 DIMM socket modules contain the following non-volatile storage:

2Mbyte NOR flash chip storing the U-Boot system bootloader and the Linux kernel
256Mbyte NAND flash chip storing the Linux root filesystem
A removable MicroSD card typically storing deployment data and a backup of system software

A factory programmed ROM in the LPC3141 CPU supports loading the U-Boot 2nd stage bootloader from many different sources including USB and even reading code from the console serial port! However, the ESP only supports booting from NOR flash or the micro SD card.

Advantages to booting from the SD card are:

1) Functions on CPU cards with corrupt or as yet uninitialized internal flash memory
   → *New boards must be configured with boot jumpers set to load U-Boot from the SD card.*
2) Faster boot time (Linux kernel loads from SD in 1 second verses 15 seconds for SPI)
3) Much larger rootfs size (up to 128GB depending on SD card vs 250MB internal flash)
4) Higher read and (especially) write speeds

Advantages of booting from the internal flash are:

1) Higher reliability – *which is why most ESPs are configured to boot from their internal flash*.
2) The ESP can boot when the SD card is missing or damaged.
3) SD cards may be easily removed to recover data from a damaged CPU board
   Most SD cards are waterproof. The circuit boards are not!

The "clone" script:

A removable "template" micro SD card facilitates the initial loading of the internal (soldered on) NOR and NAND flash chips via a script called "clone." Once the system software is loaded in the internal flash, it may be further "cloned" onto a different SD card. The resulting SD card may then be cloned back onto the internal flash of (another?) CPU module. Thus, the clone script also provides convenient means for backing up an ESP system after it has been customized. Note that clone never backs up deployment (scientific or engineering) data.

Cloning to internal flash or the SD card completely overwrites the respective destination device, provided the clone destination device is not in use. For example, it is not possible clone to the SD card if there are files currently opened on it.

When booting form internal flash, the system will automatically use any properly formatted SD installed card for storing data and system swap space. To allow cloning to an SD card, one must therefore boot the ESP without any card installed, installing the destination card only after the ESP has booted and issued its initial login prompt.

Formatting SD cards for the ESP:

SD cards must be specially formatted for use in the ESP. This one time formatting may be done on another Linux computer. This describes the card formatting process only using the ESP itself.

The internal flash is never accessed when booting from an SD card. However, when booting from internal flash, any SD card installed will be used to store system logs and deployment data. This is why one must boot the ESP without any SD card installed before attempting to format a new one. Never remove the SD card from an ESP that booted with it installed, as this will likely "hang" the ESP and may cause data corruption on the card. Power the ESP down first, or, reboot, and remove the SD card during the 15 seconds or so it takes the Linux kernel to load from internal flash. Once booted from flash with no SD card installed, proceed as follows:

Install the SD card

(note that the cfdisk utility described below displays its menus better when run from a network login rather than from the serial console)

```
# cfdisk  -z  /dev/mmcblk0
```
#the -z option ignores any previously stored partitions

Create three primary partitions of the indicated types and sizes:

```
   Name      Flags    Part TypeFS Type        [Label]     Size (MB)
 ------------------------------------------------------------------
   mmcblk0p1          Primary W95 FAT32 (LBA)               100.00
   mmcblk0p3          Primary BootIt                          0.37
   mmcblk0p2          Primary Linux                  (rest of disk)
```

The sizes are not critical. The first partition stores the Linux kernel and (optionally) the compressed factory image of the root filesystem. It should be >30MB. I usually allocate 100MB for it. But, if the factory image is omitted, it may be made as small as 4MB.
The p3 partition (of type 'BootIt') holds the U-Boot boot loader.
It must be physically before the big Linux partition on the logical disk.

To accomplish this with the cfdisk utility, create the p1 partition first, then, put the large Linux p2 partition at the end of the remaining storage, taking care to explicitly specify a size for it that is about 0.5 MB less than the maximum remaining available space. When prompted whether to allocate the big p2 partition at the begin or end of the free space, indicate you want it at the end. This will leave a ~0.5MB free hole between the 1st and 2nd partitions. Fill this tiny hole with the BootIt partition. P3 holds the U-Boot 2nd stage bootloader which requires only about 0.3MB.

The hexadecimal type of the FAT32(LBA) partition is 0C (12 decimal)
The hexadecimal type of the Linux partition is 83
The hexadecimal type of the BootIt partition is DF

Once you have the partition table displayed correctly, write it to out to the SD card with the W command, then quit cfdisk.

The formatted SD card is now ready for the clone command to use as a destination device.

How to program the ESP's internal flash:

1) Power off the ESP CPU module
2) Ensure that the middle boot jumper is selecting “SD” (not “SPI”)

3) Install a micro SD card previously prepared for the ESP into board's micro SD socket
   OEMs will likely use an MBARI supplied “template” SD card

4) Connect a terminal emulator to the ESP CPU's console serial port
   115200 baud, 8 data bits, 1 stop bit, no parity, no flow control

5) Power up the ESP CPU
   As soon as U-Boot starts, press <Control-C> to get to the U-Boot command prompt.
   Note: Plugging into the board's USB console port will also power up the board. Press the reset button after starting your terminal emulator in this case

6) At the ESP's U-Boot prompt type:
   `EA3141-MBARI:` **`run resetEnvironment`**
   The system should reboot, loading U-Boot and Linux from the SD card

7) Log into Linux as the root user
8) Type either of the commands:
   `# clone`

or

   `# clone --factory`
   Answer yes to confirm you wish to overwrite the internal flash storage
   The –factory option writes the “known good” factory supplied image rather than making a complete copy of the running system.

9) Follow the instructions displayed by the clone script to finish configuration. In U-Boot:
   `EA3141-MBARI:` **`run flashSDuboot`**
   `EA3141-MBARI:` **`edit ethaddr`**
   `edit:` **`00:1a:f1:xx:yy:zz`**
   #where xx, yy, and zz are replaced by the end of the MAC address printed on the CPU board
   `EA3141-MBARI:` **`edit bootcmd`**
   `edit:` **`run flashBoot`**
   #configure U-Boot to load the Linux kernel from internal flash by default
   `EA3141-MBARI:` **`run flashSDkernel`**
   #stores kernel in internal NOR flash and saves updates the U-Boot environment

9) Power off the CPU board
10) Move the middle (usually RED) boot jumper from SD to 'SPI'
11) Remove the template SD card
12) Plug in the ethernet network (optional)

When the board is powered, it should boot from the internal flash without need for the SD card.

How to backup Internal Flash to an SD card, making it "bootable" in the process:

1) Power the ESP down
2) Remove any SD card that had been installed in the ESP
3) Connect a terminal emulator to the ESP CPU's console serial port
   115200 baud, 8 data bits, 1 stop bit, no parity, no flow control

4) Power up the ESP CPU, booting from Internal Flash
   The middle of the three boot jumpers should be already be on 'SPI'

5) Install the SD card to be overwritten in the ESP's micro SD socket
   Wait until after the ESP has booted before installing the SD card

6) Log in as the root user

7) Format the SD card with the cfdisk utility
   see **Formatting SD cards for the ESP** instructions above

8) Type either of the commands:

```
# clone
```

or

```
# clone --factory
```

Answer yes to confirm you wish to overwrite the internal flash memory
The –factory option writes the "known good" factory image rather than making a complete copy of the running system.

The next step (#9) is optional!
Proceed only if you wish to have this ESP boot from any installed SD card rather than from its internal flash memory. If you stop here, the ESP will continue to boot from its internal flash. This is OK! Booting from internal flash is the "normal configuration" of most deployed ESPs.

Optional:
9) To make this ESP boot from any installed SD card, reboot into U-Boot:
   (by pressing <Control-C> on the serial console when prompted as system reboots)

```
EA3141-MBARI:   edit bootcmd
            edit:   run sdBoot
```

#configures U-Boot to load the Linux kernel from internal flash by default

```
EA3141-MBARI:   save
```

#updates the U-Boot environment in non-volatile flash memory

10) Power off the CPU board
11) Ensure the middle (usually RED) boot jumper is positioned at 'SD'

If step #9 is completed, when the board is next powered up,
it should boot from the installed SD card without accessing internal Flash.

Changing the ESP's local timezone:

The command timezone will change the ESP's default local timezone.
Type:

```
# timezone --help
```

To see usage examples. One need not reboot immediately after changing the system's timezone.

Changing the ESP's hostname:

After cloning, the newly bootable device will be an exact copy of the original. But, all hostnames on the a LAN should be unique. So, the recommended practice after cloning (especially from a repeatedly sourced “template” SD) is to change the ESP's hostname via the console port.

Use the “setHostname” command to change the ESP's hostname from the root account. Reboot soon after changing the hostname. Plug in the Ethernet port as the ESP is booting back up with its unique hostname. Remember that ESP hostnames all should start with the “ESP” prefix.

```
# setHostname ESPunique #the shell prompt should reflect the new hostname
# reboot
```

Recommended Procedure for Loading System Software onto a new ESP:

1) Use template SD card to program ESP's internal flash using the "**`clone -factory`**" command
2) Power off the ESP
3) Remove the template SD card
4) Change the Boot jumper from SD to SPI
5) Completely boot the ESP to from its newly programmed internal flash
6) Install the SD card that will be used for data storage after ESP.
7) Change the ESP's hostname to something unique that starts with ESP.
8) Change the ESP's local timezone to the customer's.
9) Use the "**`clone`**" command to prepare the SD card to be shipped with the ESP.
10) Reboot the esp to verify that the newly prepared SD card mounts properly.

Recovering from a corrupted internal flash rootfs:

1) Reboot into U-Boot by pressing <Control-C> on the serial console when prompted after reset.
2) Instruct U-Boot to boot from the SD card (just one time):
   `EA3141-MBARI:` **`run sdBoot`**
3) Log in as the Linux root user, then type:
   `# clone`

When ESP is rebooted, it should use the recovered flash rootfs.
The above assumes that only the rootfs was damaged – not U-Boot or the Linux kernel.
If the ESP will not start U-Boot, switch the boot jumper to SD and repeat the entire procedure for programming the internal flash.

Recovering a corrupted SD card:

If the SD card has no critical deployment data on it, use the same procedure as:

→ **How to backup Internal Flash to an SD card, making it "bootable" in the process**

However, if the SD card contains important data that is unreadable, you may be able to recover it with the following procedure:

1) shut down the ESP
2) Remove the corrupted SD card
3) Replace the corrupted SD card only after the ESP has completely rebooted from flash
4) As the root Linux user:

```
# fsck.ext4 -a /dev/mmcblk0p2  #fix filesystem errors
# reboot
```

5) Recover what data you can from the card, then replace (or at least reformat) it ASAP.

clone script –help:

```
Clone ESP linux to create a bootable images -- 10/8/17 brent@mbari.org
Usage:  [with optional elements in {}]
  clone {--factory} {--to destination} {source}
Flags:
  --factory   #read from the factory recovery image (archive)
      (this is equivalent specifying source archive /boot/rootfs.tgz)
  --to destination  #specifies clone output destination where:
      SD|card outputs to the SD card
      flash   outputs to internal flash memory
      /*      outputs tar archive to file at specified absolute path
      -       outputs tar archive to stdout
  --noimage   #do NOT include a copy of the recovery image in destination
  --help  #display this
Source argument:
  Read the root filesystem image from the specified directory or
  (compressed) tar archive
  If source is -, read archive from stdin
  If omitted, source defaults to the current rootfs  (i.e. / )
Environment Variables:
  TARFLAGS[=-z]
      overrides the default arguments to tar in reading or writing archives
Notes:
  If --to is not specified, the destination is assumed to be the storage medium
  that is currently NOT in use.  In practice, this means...
  If rootfs is the SD card, destination is internal flash and visaversa.
  The --factory flag and source argument are mutually exclusive
Examples:
  clone
    #Without args: clones current root filesystem to unused storage medium
    #If booted from flash, creates a bootable SD card.
    #If booted from SD, writes a bootable image to the internal flash.
  clone --factory
    #As above, but writes the "pristine" factory image instead
  clone /template
    #Copy from the root filesystem at /template, making it bootable
  clone --to /boot/rootfs.tgz
    #Make the current rootfs the new factory image
  clone --to /boot/rootfs.tgz  /template
    #Make the rootfs at /template the new factory image
  clone --to flash -
    #Overwrite internal flash with rootfs image archive read from stdin
```