Gestor de arranque - U-Boot

Objetivos: Configurar una comuniación serie, compilar e instalar el gestor de arranque U-Boot. Uso básico de los comandos de U-Boot.

Dado que el gestor de arranque es la primer pieza de Software ejecutada por una plataforma Hardware, el procedimiento de instalación del gestor de arranque es específico para dicha plataforma.

En general existen dos casos:

- El procesador no ofrece ninguna facilidad para la instalación del gestor de arranque, en cuyo caso se debe utilizar JTAG para inicializar el almacenamiento flash y escribir el código del gestor de arranque en la misma. Es necesario un conocimiento detallado del Hardware para poder realizar estas operaciones.
- El procesador ofrece un monitor, implementado en ROM, a través del cual es más simple acceder a la memoria.

La placa Beaglebone Black, utiliza el SoC AM3359 y entra en la segunda categoría. Por defecto, el monitor integrado en la ROM del AM3359 lee la tarjeta MMC1 (la memoria integrada eMMC) seguido de la MMC0 (MicroSD), la UART0 y finalmente el USB0. Si el pulsador S2 es presionado durante el encendido de la placa, la ROM iniciara desde la interface SPI0, luego la MMC0 (MicroSD), la USB0 y finalmente la UART0.

Configuración de la tarjeta MMC/SD

En primer lugar conecte el lector de tarjetas a la estación de desarrollo con la tarjeta MMC/SD dentro. Ejecute `dmesg` para ver que dispositivo es utilizado por la estación de trabajo. En caso de que el dispositivo sea `/dev/sdX`, vera algo como los siguiente:

```
sd 7:0:0:0: [sdc] 30537728 512-byte logical blocks: (15.6 GB/14.5 GiB)
```

Si su PC posee un lector de tarjeta interno MMC/SD, el dispositivo puede ser detectado como `/dev/mmcblkX` y la primera partición como `/dev/mmcblkXp1`.

En las próximas instrucciones, vamos a asumir que su tarjeta MMC/SD está asociada al `/dev/sdX` por la estación de trabajo (donde X es la unidad).

Ejecute el comando `mount` para verificar cuales son las particiones actualmente montadas. Si alguna de las particiones de la MMC/SD están montadas, desmóntelas:

```
sudo umount /dev/sdX1
sudo umount /dev/sdX2
...
```

Ahora, borre cualquier contenido posible de sesiones anteriores de la MMC/SD:

```
sudo dd if=/dev/zero of=/dev/sdX bs=1M count=256
```

El monitor en la ROM necesita una geometría especial en la partición para poder leer su contenido. La tarjeta MMC/SD debe tener 255 cabezas y 63 sectores.

Vamos a utilizar el comando `cfdisk` para crear la primer partición con la siguiente configuración:

```
sudo cfdisk -h 255 -s 63 /dev/sdX
```

Nota: si el comando anterior muestra la ayuda, es porque posee una versión diferente de dicho comando. En ese caso vamos a utilizar el comando `fdisk` (ver más adelante).

En la interfaz de `cfdisk` cree una primer partición comenzando al principio con un tamaño de 64Mb, **iniciable** y de tipo `0c` (W95 FAT32 (LBA)). Presione `Escribir` y listo.

Luego asigne formato a la nueva partición en FAT32, con la etiqueta `BOOT`:

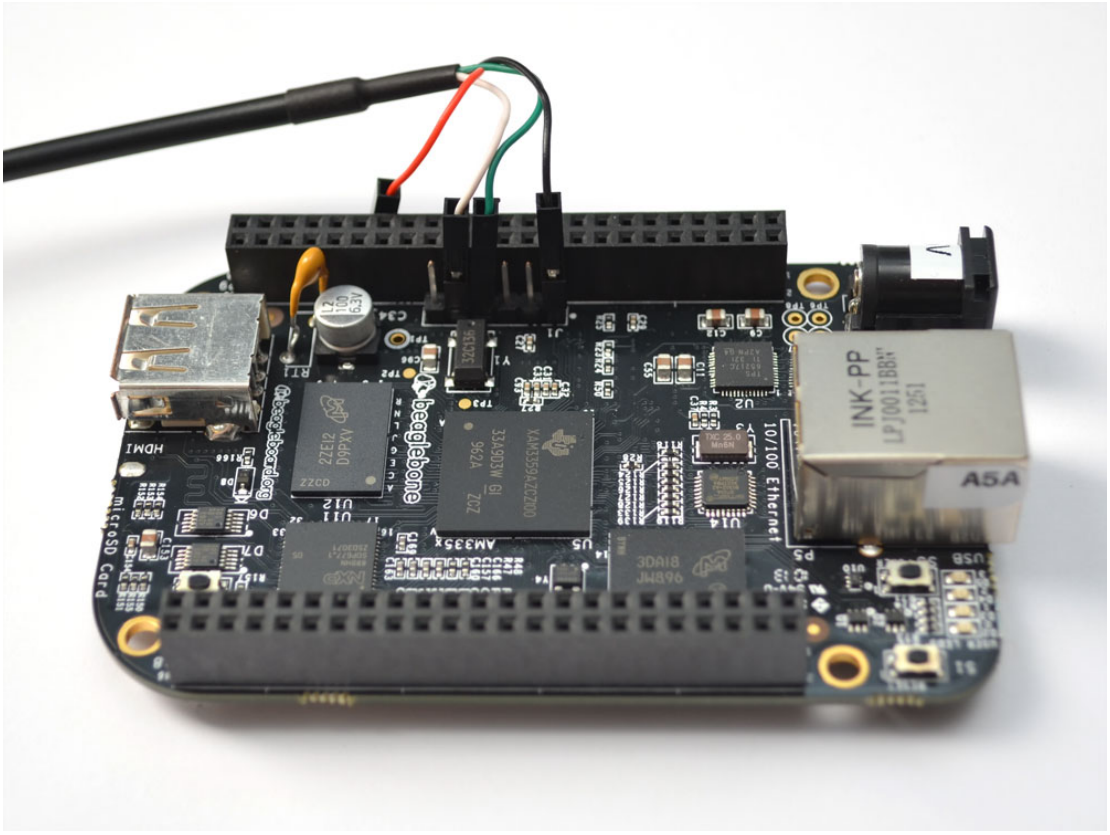
```
sudo mkfs.vfat -n BOOT -F 32 /dev/sdX1
```

Al finalizar, remueva la tarjeta y vuelva a ingresarla.

La tarjeta está lista para utilizarce.

Configuración de la comunicación serie con la placa

Conecte cable del adaptador USB-a-TTL a la placa Beaglebone Black. El cable negro se conecta al pin 1 `GND` del conector J1, el cable `TXD` al pin 4 del conector J1 y el cable `RXD` al pin 5 del conector J1. Cuando se conecte el adaptador a la estación de desarrollo, deberá aparecer un nuevo puerto serie `/dev/ttyUSB0`.



Es posible visualizar si el dispositivo fue reconocido por la estación de desarrollo consultando la salida del comando `dmesg`.

Para comunicarse con la placa a través del puerto serie, instale el programa de comunicaciones `picocom`:

```
sudo apt-get install picocom
```

Es necesario que pertenezca al grupo `dialout` para que tenga permisos para escribir sobre la consola serie:

```
sudo adduser $USER dialout
```

Puede consultar el nombre de usuario con el comando:

```
whoami
```

Nota: Debe salir del sistema y volver a entrar para que los cambios de grupo se hagan efectivos.

Ejecute `picocom -b 115200 /dev/ttyUSB0`, para iniciar una comunicación serie en `/dev/ttyUSB0`, con un baudrate de 115200. Si desea salir de `picocom`, presione `[Ctrl][a]` seguido de `[Ctrl][x]`.

Configuración

Configuración de U-Boot

Vaya al directorio de los laboratorios `${PROJECT_ROOT}/bootloader`.

Descargue U-Boot:

```
git clone git://git.denx.de/u-boot.git
cd u-boot
git checkout v2019.04
```

Vamos a aplicar un parche para la Beaglebone Black:

```
cd ..
wget -c https://raw.githubusercontent.com/eewiki/u-boot-patches/master\
        /v2019.04/0001-am335x_evm-uEnv.txt-bootz-n-fixes.patch

cd u-boot
patch -p1 < ../0001-am335x_evm-uEnv.txt-bootz-n-fixes.patch
```

Recomendamos entienda los pasos de configuración y compilación, leyendo el archivo `README`, y especialmente la sección *Building the Software*.

Básicamente, necesitaremos:

- Asignar la variable de entorno `CROSS_COMPILE`
- Ejecutar `make <nombre>_defconfig`, donde `<nombre>` es el nombre de la placa que vamos a utilizar. En nuestro caso es `am335x_boneblack_defconfig`
- Finalmente, ejecutamos `make`, para construir U-Boot

Una vez finalizado el proceso de compilación, insertamos tarjeta SD/MMC en la estación de desarrollo y copiamos los archivos `MLO` y `u-boot.img` generados.

Al archivo `MLO` se lo conoce como **First Stage Bootloader**. Al archivo `u-boot.img` se lo conoce como **Second Stage Bootloader**.

Probando U-Boot

Levante la comunicación serie con `picocom`. Reinicie la placa y verifique que inicia el nuevo cargador de inicio (recuerde mantener presionado el botón S2). Es posible verificar esto consultando las fechas:

```
U-Boot 2017.07-dirty (Aug 06 2017 - 20:04:38 -0300)

CPU  : AM335X-GP rev 2.1
I2C:   ready
DRAM:  512 MiB
No match for driver 'omap_hsmmc'
No match for driver 'omap_hsmmc'
Some drivers were not found
Reset Source: Power-on reset has occurred.
MMC:   OMAP SD/MMC: 0, OMAP SD/MMC: 1
Using default environment

Board: BeagleBone Black
<ethaddr> not set. Validating first E-fuse MAC
Net:   eth0: MII MODE
cpsw
Press SPACE to abort autoboot in 2 seconds
```

Interrumpa la cuenta regresiva para entrar a la línea de comandos de U-Boot:

```
=>
```

En U-Boot, escriba el comando `help`, y explore alguno de los comandos disponibles.

También es posible consultar la versión de U-Boot con el comando `version`

Listado comandos U-Boot

```
?       - alias for 'help'
askenv  - get environment variables from stdin
base    - print or set address offset
bdinfo  - print Board Info structure
boot    - boot default, i.e., run 'bootcmd'
bootd   - boot default, i.e., run 'bootcmd'
bootm   - boot application image from memory
bootp   - boot image via network using BOOTP/TFTP protocol
bootz   - boot Linux zImage image from memory
chpart  - change active partition
cmp     - memory compare
coninfo - print console devices and information
cp      - memory copy
crc32   - checksum calculation
dfu     - Device Firmware Upgrade
dhcp    - boot image via network using DHCP/TFTP protocol
echo    - echo args to console
editenv - edit environment variable
eeprom  - EEPROM sub-system
env     - environment handling commands
```



```
exit    - exit script
ext2load- load binary file from a Ext2 filesystem
ext2ls  - list files in a directory (default /)
ext4load- load binary file from a Ext4 filesystem
ext4ls  - list files in a directory (default /)
false   - do nothing, unsuccessfully
fatinfo - print information about filesystem
fatload - load binary file from a dos filesystem
fatls   - list files in a directory (default /)
fatwrite- write file into a dos filesystem
fdt     - flattened device tree utility commands
go      - start application at address 'addr'
gpio    - query and control gpio pins
gpt     - GUID Partition Table
help    - print command description/usage
i2c     - I2C sub-system
iminfo  - print header information for application image
imxtract- extract a part of a multi-image
itest   - return true/false on integer compare
load    - load binary file from a filesystem
loadb   - load binary file over serial line (kermit mode)
loads   - load S-Record file over serial line
loadx   - load binary file over serial line (xmodem mode)
loady   - load binary file over serial line (ymodem mode)
loop    - infinite loop on address range
ls      - list files in a directory (default /)
md      - memory display
mdio    - MDIO utility commands
mii     - MII utility commands
mm      - memory modify (auto-incrementing address)
mmc     - MMC sub system
mmcinfo - display MMC info
mtdparts- define flash/nand partitions
mw      - memory write (fill)
nand    - NAND sub-system
nboot   - boot from NAND device
nfs     - boot image via network using NFS protocol
nm      - memory modify (constant address)
part    - disk partition related commands
ping    - send ICMP ECHO_REQUEST to network host
printenv- print environment variables
reset   - Perform RESET of the CPU
run     - run commands in an environment variable
saveenv - save environment variables to persistent storage
setenv  - set environment variables
sf      - SPI flash sub-system
showvar - print local hushshell variables
sleep   - delay execution for some time
source  - run script from memory
spl     - SPL configuration
sspi    - SPI utility command
test    - minimal test like /bin/sh
```



```
tftpboot- boot image via network using TFTP protocol
true    - do nothing, successfully
usb     - USB sub-system
usbboot - boot from USB device
version - print monitor, compiler and linker version
```

