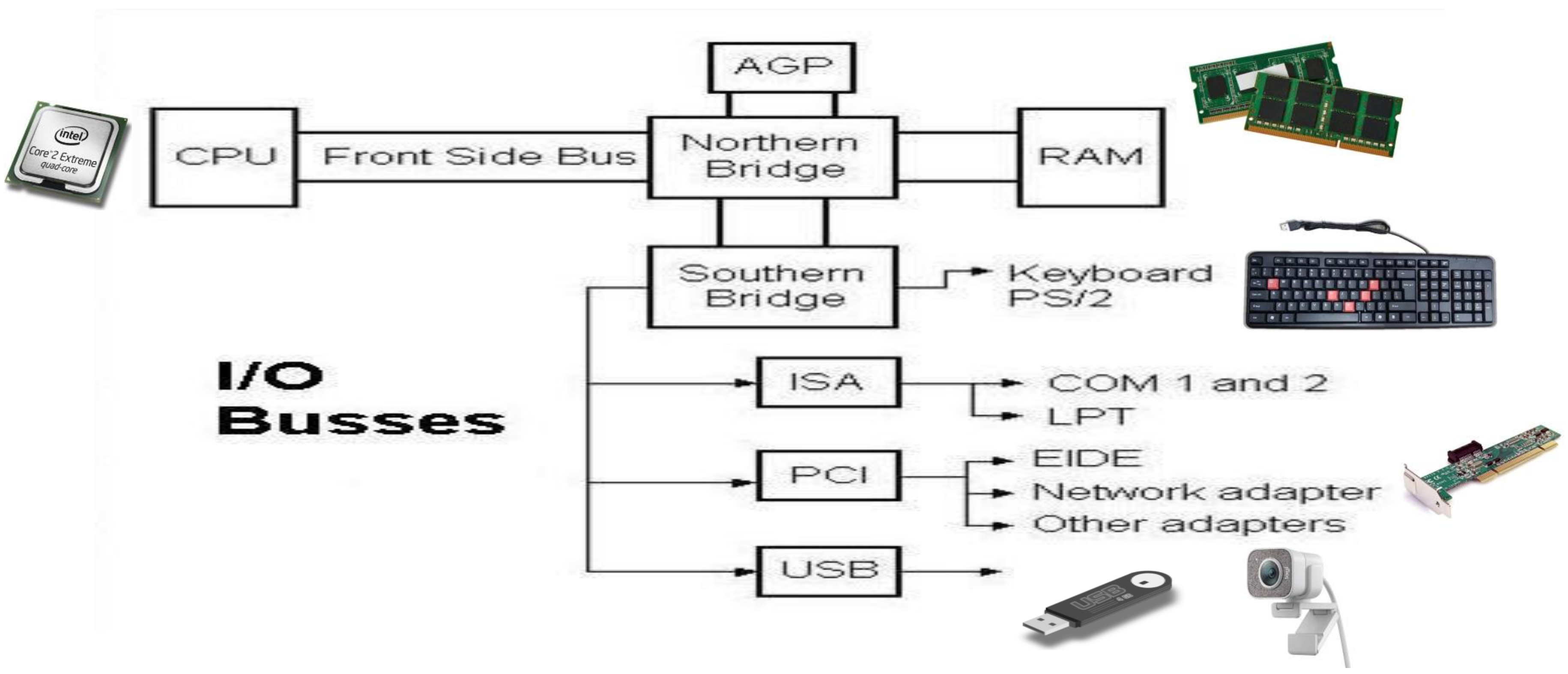
EMBEDDED OPERATING SYSTEMS

Embedded Linux on Beaglebone Black

Computer Bus Architecture



Booting and bootloaders

- Microprocessors / Microcontrollers
  - Can execute code residing in memory (RAM / ROM)
- Operating Systems (OSes)
  - Reside on large-capacity devices
    - Hard-disks, SD cards,…

- On boot, memory does not contain the OS
- Special software needed
  - Bring OS from media to memory

- Role of bootloader
  - Initialize hardware, specially the memory controller
  - Provide boot parameters to the OS
  - Start the OS

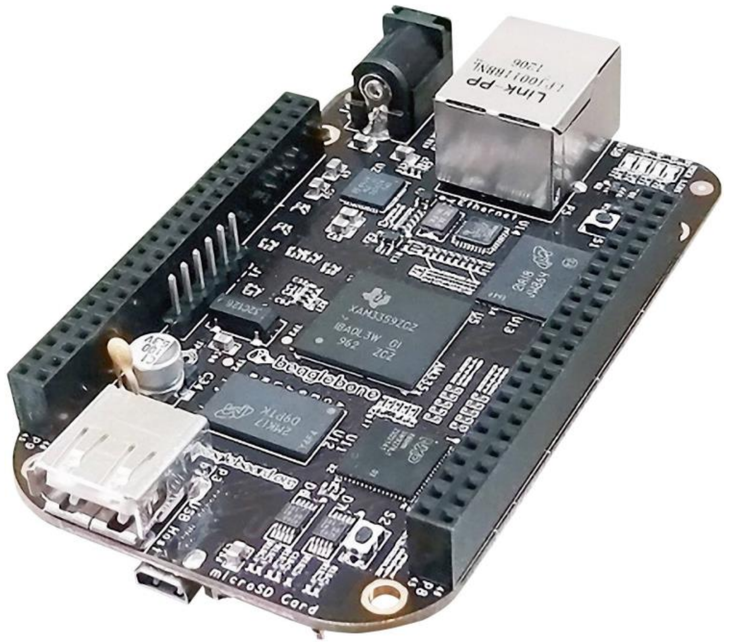
U-Boot bootloader

- Open-source, cross-platform bootloader
- Out-of-box support for boards and CPUs
  - PowerPC, x86, ARM, MIPS, …
- Features:
  - Customizable footprint
  - Command shell / monitor
    - With inbuilt commands
  - Variables and scripts
  - Support for Ethernet and USB

Beaglebone Black

- Low-cost, high-expansion development platform
    - Community-supported

- Hardware
    - Chipset
        - TI Sitara AM3358
        - 1GHz ARM CORTEX A8
    - Memory
        - 512MB DDR3L SDRAM
        - On-board Flash 4GB (eMMC)
        - SD/MMC connector for microSD card
    - Debug
        - 20-pin CTI JTAG, Serial header
    - Connectivity
        - USB2.0 Host, Client
        - Serial (UART0) 3.3V TTL, 10/100 Ethernet (RJ45)

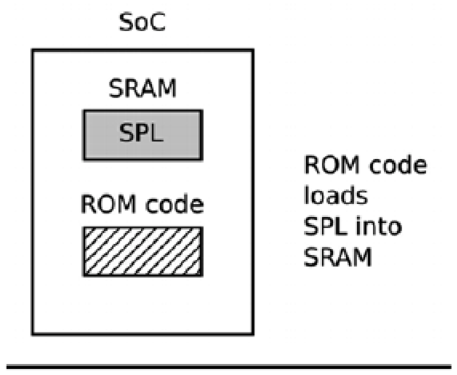


Beaglebone boot (1/5)

- Problem
  - Linux to be booted from eMMC / SD card
- Issue
  - On boot, ARM A-8 CPU
    - Does not have access to:
      - On-board SDRAM
      - Complex devices such as eMMC /SD cards
    - But does have on-chip SRAM (64KB) and ROM (256KB)
  - This leads to a 3-phase boot sequence

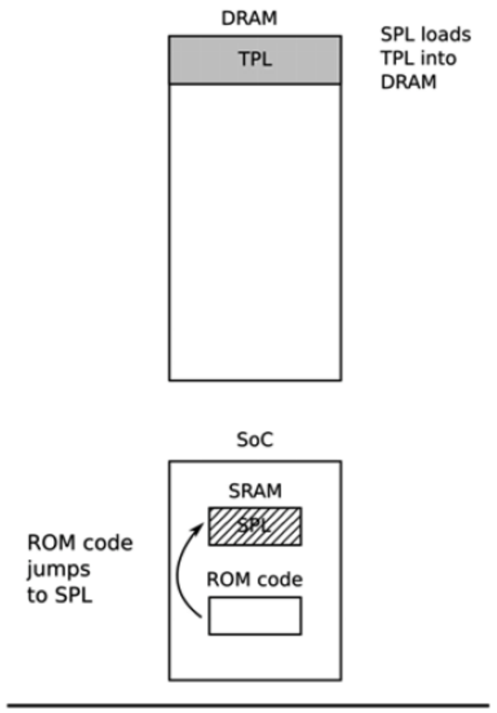
Beaglebone boot (2/5)

- Phase-1
  - On-chip SRAM not large enough
    - Cannot hold/run full bootloader
    - Runs Secondary Program Loader (SPL)
  - Code resident in ROM (ROM code)
    - Loads SPL from pre-programmed locations into SRAM
      - eMMC / SD card / Ethernet
    - Then it jumps to beginning of that code



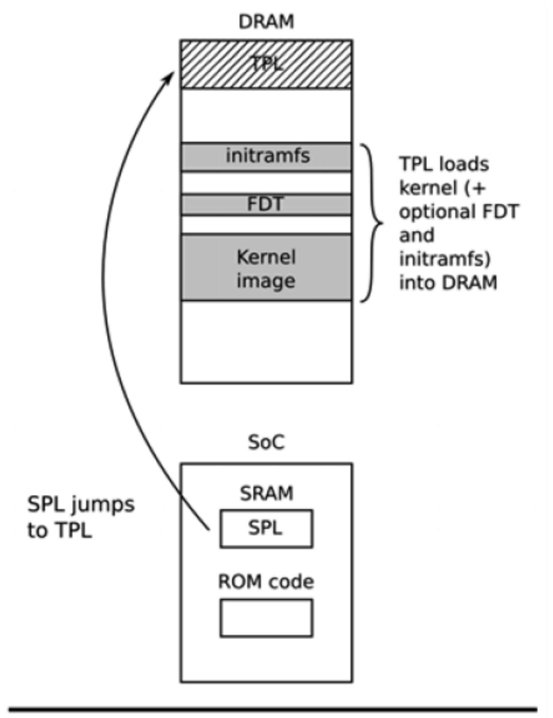
Beaglebone boot (3/5)

- Phase-2
  - SPL sets up memory controller
    - Gets access to on-board DRAM
  - Reads bootloader (*uboot.img*)
    - Called Tertiary Program Loader (TPL)
      - From partition into SDRAM



Beaglebone boot (4/5)

- Phase-3
  - TPL is a full bootloader
  - Loads kernel, devicetree and initrd
    - Into SDRAM
  - Calls the kernel's entry point
    - With boot parameters
  - Kernel takes over and boots
    - Reclaims memory used by bootloader

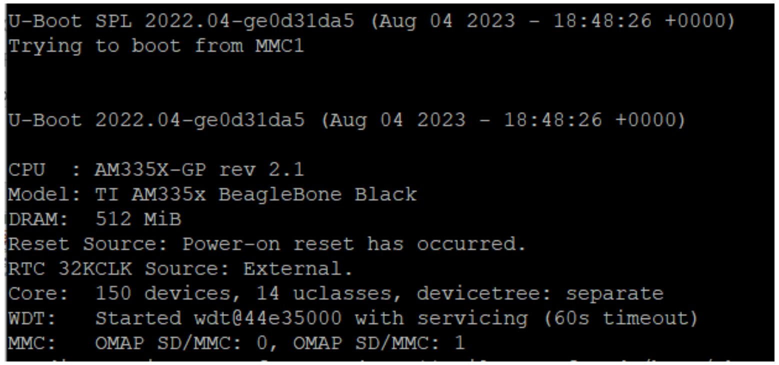


Beaglebone boot (5/5)

- U-Boot bootloader passes following info to kernel
  - Machine number of CPU
  - Basic hardware details
    - Size of DRAM detected, CPU clock speed
  - Kernel command line
    - ASCII
    - Contains the “root” filesystem
      - Path to main filesystem
    - Other behavioral parameters
  - Optionally,
    - device tree
    - initramfs (initial RAM filesystem)

Beaglebone boot screenshots

- U-Boot SPL and TPL

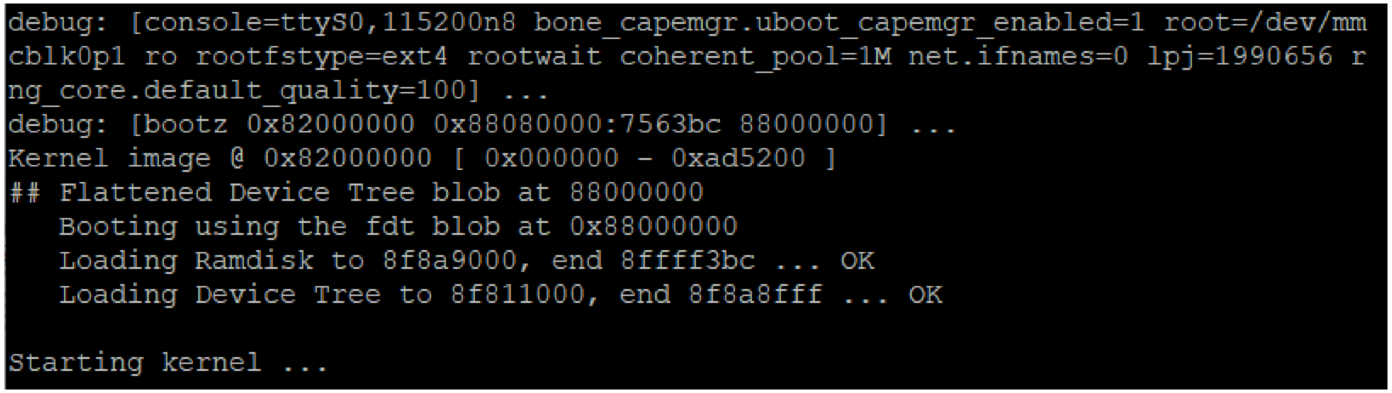


```
U-Boot SPL 2022.04-ge0d31da5 (Aug 04 2023 - 18:48:26 +0000)
Trying to boot from MMC1


U-Boot 2022.04-ge0d31da5 (Aug 04 2023 - 18:48:26 +0000)

CPU  : AM335X-GP rev 2.1
Model: TI AM335x BeagleBone Black
DRAM:  512 MiB
Reset Source: Power-on reset has occurred.
RTC 32KCLK Source: External.
Core:  150 devices, 14 uclasses, devicetree: separate
WDT:   Started wdt@44e35000 with servicing (60s timeout)
MMC:   OMAP SD/MMC: 0, OMAP SD/MMC: 1
```

- U-Boot handover to Linux kernel



```
debug: [console=ttyS0,115200n8 bone_capemgr.uboot_capemgr_enabled=1 root=/dev/mm
cblk0p1 ro rootfstype=ext4 rootwait coherent_pool=1M net.ifnames=0 lpj=1990656 r
ng_core.default_quality=100] ...
debug: [bootz 0x82000000 0x88080000:7563bc 88000000] ...
Kernel image @ 0x82000000 [ 0x000000 - 0xad5200 ]
## Flattened Device Tree blob at 88000000
   Booting using the fdt blob at 0x88000000
   Loading Ramdisk to 8f8a9000, end 8ffff3bc ... OK
   Loading Device Tree to 8f811000, end 8f8a8fff ... OK

Starting kernel ...
```

Course environment

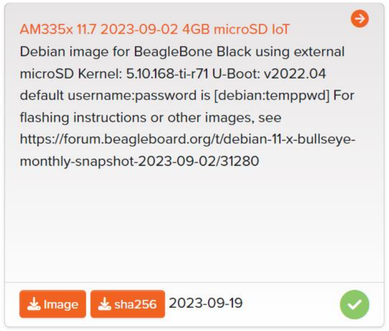
- Host (VM)
  - OS: Ubuntu Linux 22.0.4.3LTS
  - Machine: Intel x86_64 based Laptop/Notebook

- Target
  - Beaglebone Black Rev C

- *Prerequisites*
  - *C programming*
  - *Make utility*
  - *Familiarity with Linux shell, VM/dual-boot usage*

Beaglebone setup

- Download latest Beaglebone distro
  - We use 4GB IOT for microSD
  - https://www.beagleboard.org/distros
- Use Balena Etcher
  - For flashing microSD card on Windows
  - https://etcher.balena.io/#download-etcher
- Use microSD boot (press SW S2)
  - *S2 is near the microSD card*
- Use USB to 3.3V TTL 4-pin cable for Serial debug
  - Follow this link to get it working on Windows
  - Use PuTTY for observing serial logs

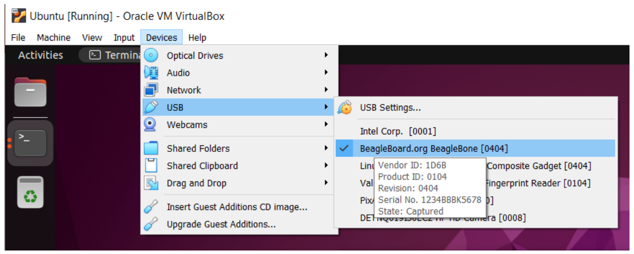


AM335x 11.7 2023-09-02 4GB microSD IoT

Debian image for BeagleBone Black using external microSD Kernel: 5.10.168-ti-r71 U-Boot: v2022.04 default username:password is [debian:temppwd] For flashing instructions or other images, see https://forum.beagleboard.org/t/debian-11-x-bullseye-monthly-snapshot-2023-09-02/31280

Beaglebone ssh login

- Enable Beaglebone in Ubuntu VM
    - Beaglebone sets up Ethernet network on USB
        - Check using *$ ip addr*



Vendor ID: 1D6B
Product ID: 0104
Revision: 0404
Serial No. 1234BBBK5678
State: Captured

- Connecting via SSH
    - IP addresses
        - VM gets 192.168.7.1
        - Board gets 192.168.7.2

```
26: enxe415f6f399c2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 1000
    link/ether e4:15:f6:f3:99:c2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.7.1/24 brd 192.168.7.255 scope global dynamic noprefixroute enxe415f6f399c2
       valid_lft 1036sec preferred_lft 1036sec
    inet6 fe80::ae6:1698:df7e:b7c2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

THANK YOU!