OPEN Compute Project®

OPEN SYSTEM FIRMWARE

SystemReady LS updates

Jeffrey Booher-Kaeding
System Architecture Engineer, Arm

Connect. Collaborate. Accelerate.

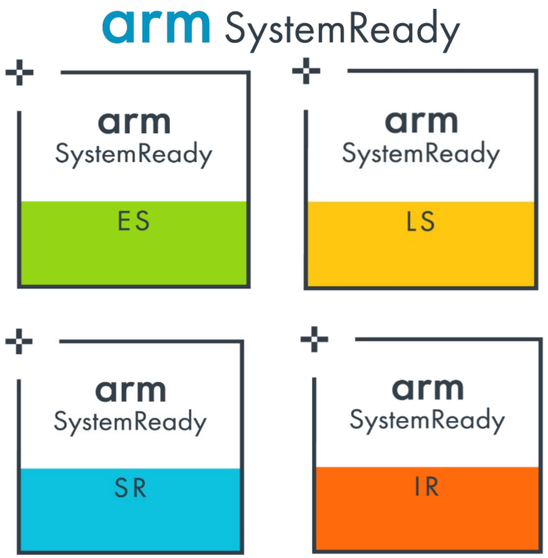
OPEN SYSTEM FIRMWARE

Arm SystemReady

- Review
  - SystemReady and SystemReady LS
  - LinuxBoot
- Standards for SystemReady LS
  - LBBR-V1 & LBBR-Vn
- SystemReady LS V0.9
  - Certification requirements and processes
- System Showcase
  - Showcase production systems
  - Showcase of proof of concepts (PoC) systems and resources

Arm SystemReady

- SystemReady is a foundational compliance certification program that brings a level of consistency across broad range of Arm devices, spanning Server, Infrastructure Edge and IoT Edge sectors.
- Our vision is for software to work seamlessly across a vibrant, diverse ecosystem of hardware
- Focusing on the common components of the software stack – the OS, hypervisor, and middleware components
- Establishing a more uniform hardware system architecture and consistency around key processes like boot, through our standards-based approach



SystemReady LS and SR

arm SystemReady LS

LS

"Just Works" for Linux OSes on Arm server SoCs

- Program tailored to meet needs of many hyperscalers
- Ensures standard firmware interfaces to deploy and maintain
- Targets hyperscalers’ Linux environment

arm SystemReady SR

SR

"Just Works" on Arm server or workstation SoCs

- Program tailored to meet needs of Windows, VMware, Linux, and BSD ecosystem
- Ensures standard firmware interfaces to deploy and maintain
- Supports old OSes to run on new hardware and vice versa
- Targets generic off-the-shelf OSes

LinuxBoot

What is LinuxBoot?

- Foundation for Arm SystemReady LS class of servers
- LinuxBoot is an alternative firmware stack (used by hyperscale datacenters) that relies on the Linux kernel and u-root as the Normal World firmware component.
- Re-uses existing Linux drivers code (without the need to duplicate work by writing DXE/UEFI drivers)

LinuxBoot

arm
SystemReady
LS
CERTIFIED
V0.9

https://linuxboot.org/

OPEN SYSTEM FIRMWARE

Standards for SystemReady LS

Connect. Collaborate. Accelerate.

OPEN Compute Project®

OPEN SYSTEM FIRMWARE

Arm SystemReady standards

Hardware requirements (BSA – Base System Architecture)

- **BSA v1.0b (May 2022)** – generic hardware target
- Documents a minimal set of CPU and system architecture necessary for an OS to boot and run. Includes aspects such as PCIe integration.

Server Hardware Supplement (SBSA)

- **SBSA**: Servers market segment specific hardware requirements
- Mostly follows the Arm architecture enhancements
- SBSA v7.0 (Jan 2021)

Firmware (BBR – Base Boot Requirements)

- **BBR v2.0 (May 2022)**
- SBBR, EBBR, LBBR Recipes targeting different Oses
- Expands to include common firmware interfaces, but recognizes that different software stacks will require different recipes

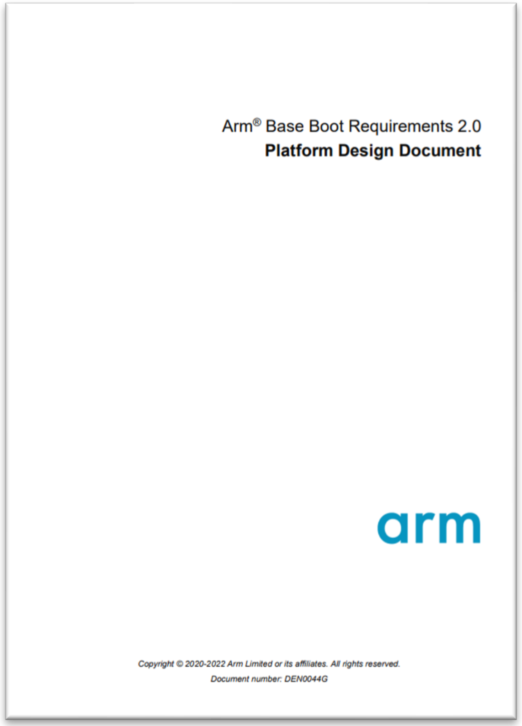
http://www.arm.com/systemready-certification-program

Standards for LS vs SR

arm SystemReady	arm SystemReady LS CERTIFIED V0.9	arm SystemReady SR CERTIFIED V2.2
Firmware Spec	UEFI (reduced) + ACPI + SMBIOS	UEFI + ACPI + SMBIOS
Platform Hardware	64bit Arm	64bit Arm
OS/Hypervisor	Linux (for LBBR-v1)	Generic, off-the-shelf
OS Distro *(examples)*	Linux distros (Ubuntu, CentOS, Fedora, Debian, openSUSE, etc...)	VMware ESXi, Windows Client/Server, RHEL, SLES, Ubuntu, CentOS, Fedora, openSUSE, Debian, CBL-Mariner, FreeBSD, NetBSD, OpenBSD, ...
Hardware Compliance Levels	**BSA+SBSA** Levels 3 through 6	**BSA+SBSA** Levels 3 through 6
BBR Recipe	**LBBR-v1** (LBBR-vn in devlopment)	**SBBR**
Certification	**Arm SystemReady LS** + System Compatibility List	**Arm SystemReady SR** + System Certification List

LBBR

- Define the “base boot requirements” for LinuxBoot based Arm servers, enabling SystemReady LS
- LBBR requirements are covered in the BBR specification
  - BBR v1.0 (Oct 2020) had preliminary LBBR requirements
  - BBR v2.0 **(May 2022)** defines **LBBR-v1** recipe requirements
- Defined in a phased approach
  - LBBR-v1: is a practical set of requirements that map to today’s Arm server implementations
- Goal is to continue evolving the LBBR recipes in the future
  - Reduce the dependency on underlying UEFI FW implementations
  - Improve the standard FW interfaces published by Linuxboot to the final Operating System.



Arm® Base Boot Requirements 2.0
Platform Design Document

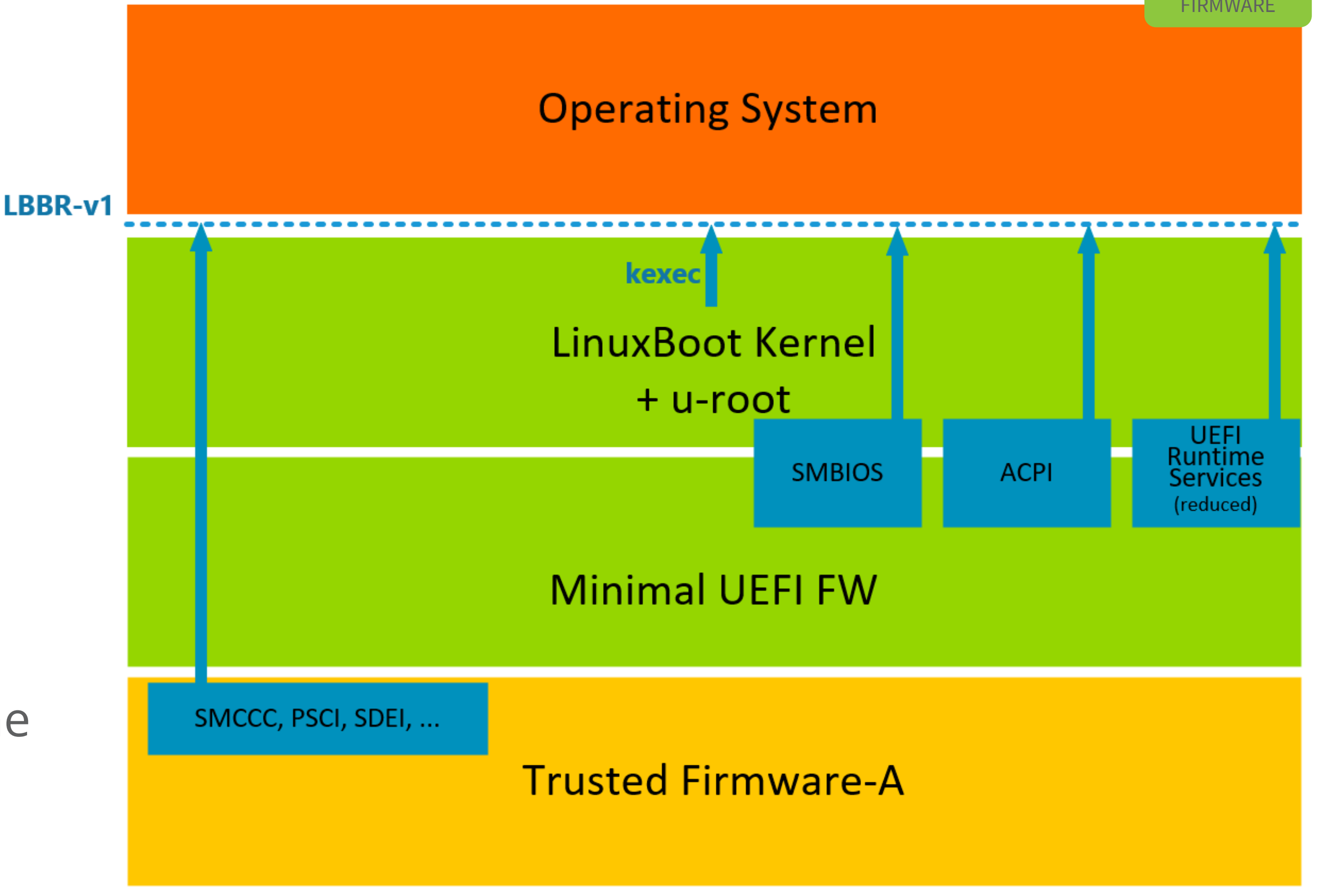
arm

Copyright © 2020-2022 Arm Limited or its affiliates. All rights reserved.
Document number: DEN0044G

https://developer.arm.com/documentation/den0044/latest

Arm LBBR-v1

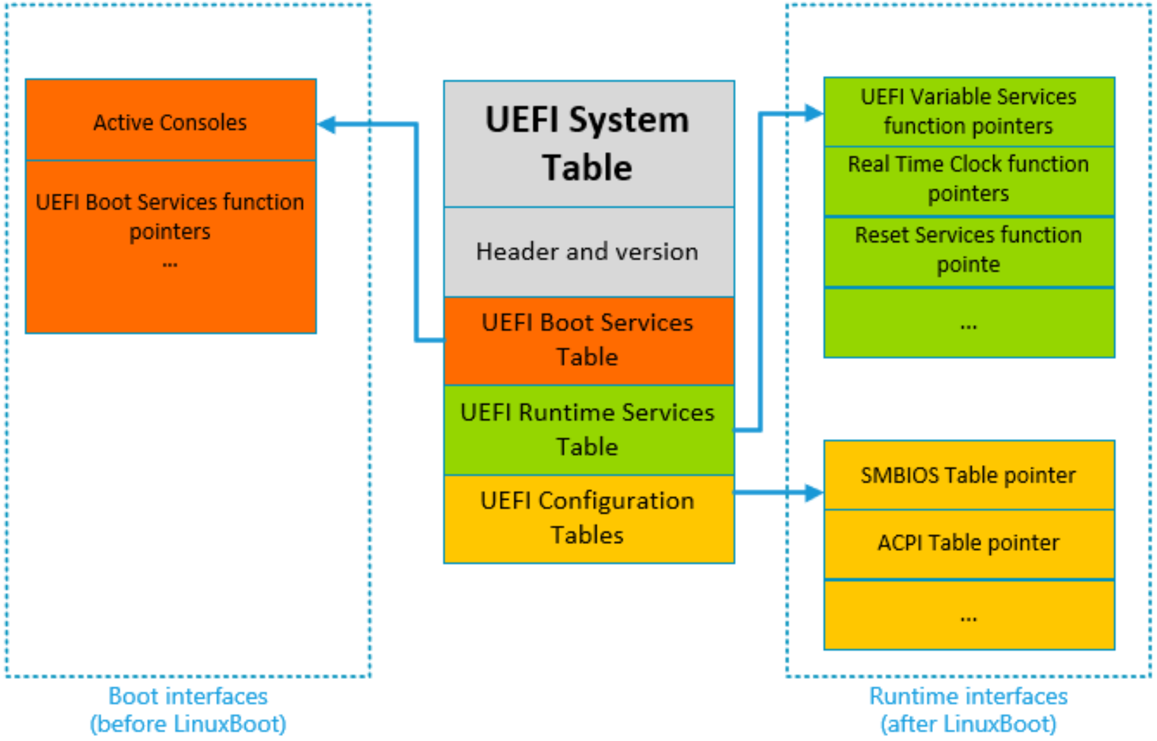
- Minimal UEFI FW (TianoCore or others) implements the OS runtime interfaces
  - SMBIOS + ACPI (needed for server use-cases)
  - UEFI Runtime services (reduced requirements)
- These runtime interfaces are also used internally by LinuxBoot itself
- LinuxBoot launches the final Linux OS using kexec (limited to running Linux OSes only)
- The interfaces are preserved and used by the final Linux OS



https://developer.arm.com/documentation/den0044/latest

Arm LBBR-v1

- On Arm64, the only way to convey the ACPI table to Linux kernel is using UEFI Configuration Table
  - https://www.kernel.org/doc/html/latest/arm64/arm-acpi.html#booting-using-acpi-tables
  - UEFI System Table is required (to host the ACPI and SMBIOS table pointers)
- This means the entire "runtime portion" of the UEFI System Table is needed
- All UEFI Runtime Services pointers **must** be implemented (cannot be NULL)
  - But can return UNSUPPORTED
- Boot Services pointers are not needed (not used by final OS)
- Tables of UEFI requirements and expectations can be found in BBR 2.0

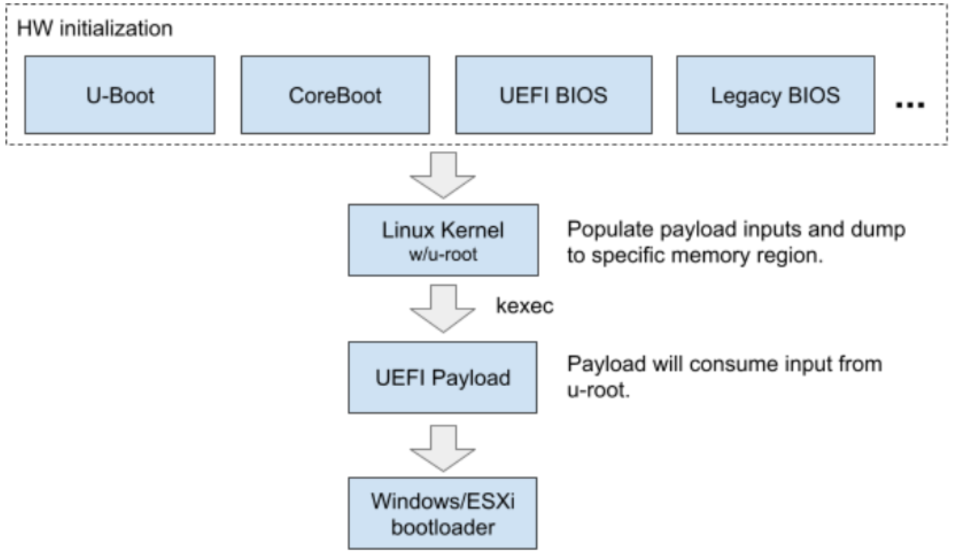


Active Consoles
UEFI Boot Services function pointers
...
UEFI System Table
Header and version
UEFI Boot Services Table
UEFI Runtime Services Table
UEFI Configuration Tables
UEFI Variable Services function pointers
Real Time Clock function pointers
Reset Services function pointe
...
SMBIOS Table pointer
ACPI Table pointer
...
Boot interfaces (before LinuxBoot)
Runtime interfaces (after LinuxBoot)

https://developer.arm.com/documentation/den0044/latest

Future work: UEFI on top of LinuxBoot

- Google leading ongoing work to implement UEFI ABI on top of LinuxBoot.
- Relies on UefiPayloadPkg from EDK2
  - Fork: https://github.com/linuxboot/edk2/tree/uefipayload
  - Upstream: https://github.com/tianocore/edk2/tree/master/UefiPayloadPkg
  - Approach: https://docs.google.com/document/d/11RRJSprAEp-whQYagtO9VTospLTCG5gzSOPoUEWlIJQ
  - Design details: https://docs.google.com/document/d/1mU6ICHTh0ot8U45uuRENKOGI8cVzizdyWHGYHpEguVg/
  - Join the discussions at OSFC slack server **#efi-boot-support** channel



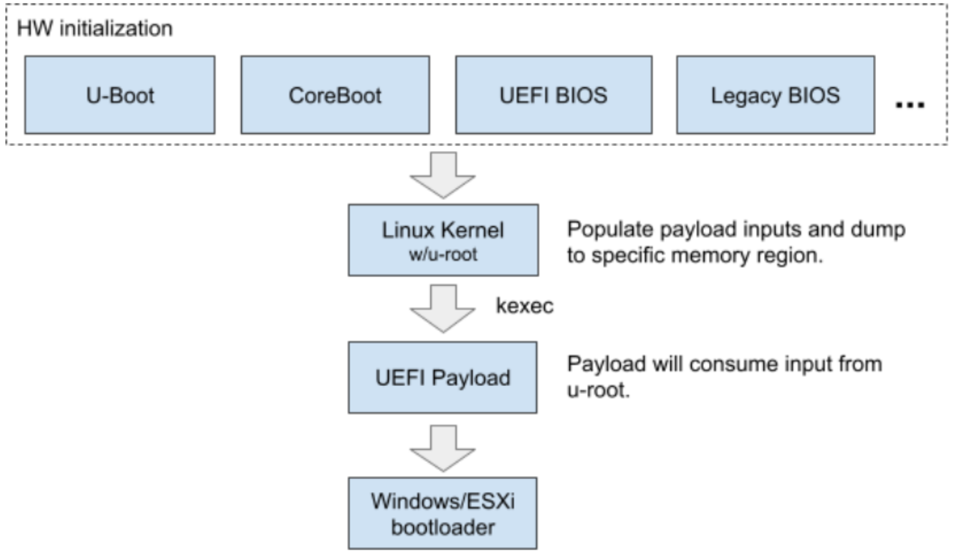
Populate payload inputs and dump to specific memory region.

Payload will consume input from u-root.

Future work: UEFI on top of LinuxBoot

- This approach can be used in future LBBR-vn
  - Allows removal of “minimal UEFI FW” below LinuxBoot, and replace with some other minimal platform FW (such as CoreBoot)
  - Allows presenting more complete UEFI interfaces to the final OS (including UEFI Boot Services), which enables booting non-Linux OSes
  - Can potentially lead to merging the Arm SystemReady SR and LS bands

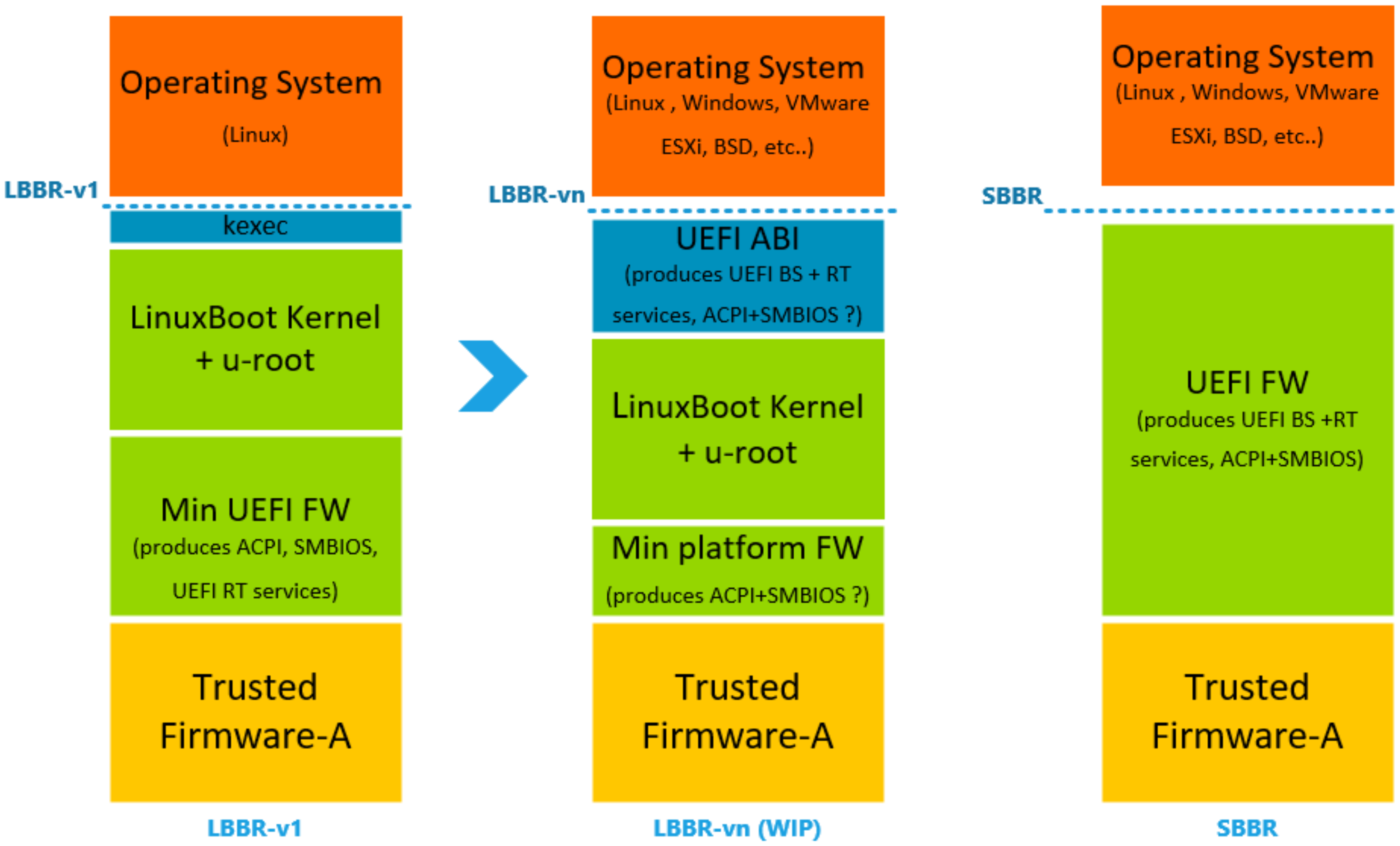
- Currently UefiPayloadPkg not supported on AARCH64
  - Development work needed



Populate payload inputs and dump to specific memory region.

Payload will consume input from u-root.

LBBR Evolution Roadmap



Connect. Collaborate. Accelerate.

OPEN Compute Project®

OPEN SYSTEM FIRMWARE

SystemReady LS V0.9

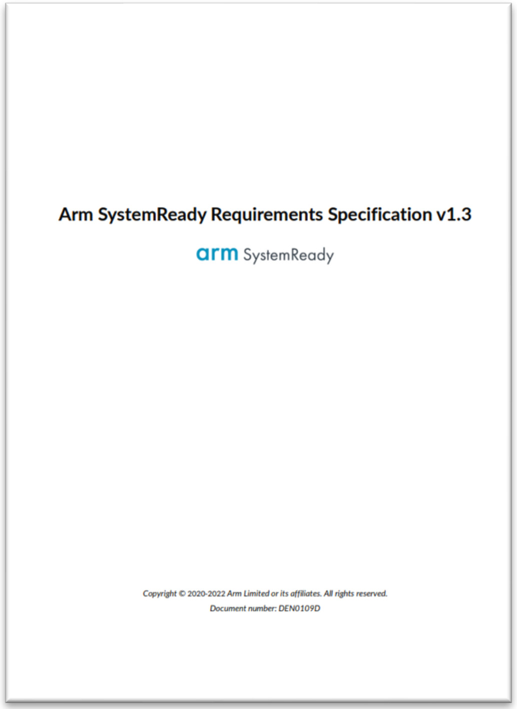
Certification requirements and processes

Connect. Collaborate. Accelerate.

SystemReady Requirements Specification (SRS) v1.3

New SystemReady LS requirements and version!

- SystemReady LS v0.9 requires the certified devices to be compliant to the following specifications:
  - BSA v1.0b and Level 3-6 as defined in SBSA Supplement v7.0.
  - LBBR-v1 recipe in BBR v2.0.
- To certify a device for SystemReady LS v0.9, the following need to be submitted:
  - Results from running the SystemReady SR ACS v1.0 UEFI SBSA tests on the device with SBBR-compliant firmware
  - Results from running the SystemReady SR ACS v1.0 FirmwareTestSuite (FWTS) and Linux SBSA tests on the same device with LBBRv1-compliant firmware
  - Boot logs from two of the Linux distros are required. The recommended distros are CentOS, Debian, Ubuntu, openSUSE and Fedora



Arm SystemReady Requirements Specification v1.3

arm SystemReady

Copyright © 2020-2022 Arm Limited or its affiliates. All rights reserved.

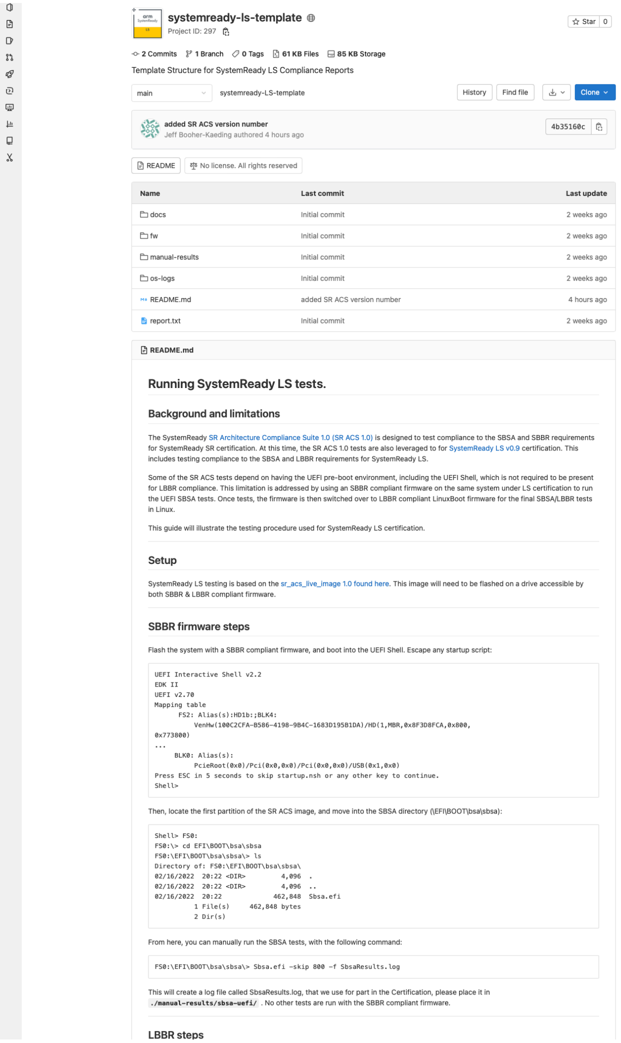
Document number: DEN0109D

https://developer.arm.com/documentation/den0109/latest

systemready-ls-template repo

- New repo to with all details of how to Certify LS V0.9
  - Detailed README
    - Step-by-step guide of collecting ACS-SR 1.0 logs
    - Recommendations to avoid common issues when first working with LinuxBoot & U-root
    - Checklist of all results, logs, and what firmware they need to be collected with.
  - Including folder structure to help keep track of results and logs
    - ACS results folder both SBBR & LBBR firmware
    - Linux OS logs, with link to download OS'es
    - Document and firmware folder for notes.
  - Platform to open issues, and contiguously improve documentation and processes.

https://gitlab.arm.com/systemready/systemready-LS-template



systemready-ls-template

Project ID: 297

2 Commits 1 Branch 0 Tags 61 KB Files 85 KB Storage

Template Structure for SystemReady LS Compliance Reports

systemready-LS-template

added SR ACS version number
Jeff Booher-Kaeding authored 4 hours ago
4b35160c

Name	Last commit	Last update
docs	Initial commit	2 weeks ago
fw	Initial commit	2 weeks ago
manual-results	Initial commit	2 weeks ago
os-logs	Initial commit	2 weeks ago
README.md	added SR ACS version number	4 hours ago
report.txt	Initial commit	2 weeks ago

README.md

Running SystemReady LS tests.

Background and limitations

The SystemReady SR Architecture Compliance Suite 1.0 (SR ACS 1.0) is designed to test compliance to the SBSA and SBBR requirements for SystemReady SR certification. At this time, the SR ACS 1.0 tests are also leveraged to for SystemReady LS v0.9 certification. This includes testing compliance to the SBSA and LBBR requirements for SystemReady LS.

Some of the SR ACS tests depend on having the UEFI pre-boot environment, including the UEFI Shell, which is not required to be present for LBBR compliance. This limitation is addressed by using an SBBR compliant firmware on the same system under LS certification to run the UEFI SBSA tests. Once tests, the firmware is then switched over to LBBR compliant LinuxBoot firmware for the final SBSA/LBBR tests in Linux.

This guide will illustrate the testing procedure used for SystemReady LS certification.

Setup

SystemReady LS testing is based on the sr_acs_live_image 1.0 found here. This image will need to be flashed on a drive accessible by both SBBR & LBBR compliant firmware.

SBBR firmware steps

Flash the system with a SBBR compliant firmware, and boot into the UEFI Shell. Escape any startup script:

```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70
Mapping table
      FS2: Alias(s):HD1b:;BLK4:
          VenHw(100C2CFA-B586-4198-9B4C-1683D195B1DA)/HD(1,MBR,0x8F3D8FCA,0x800,
0x773800)
...
     BLK0: Alias(s):
          PcieRoot(0x0)/Pci(0x0,0x0)/Pci(0x0,0x0)/USB(0x1,0x0)
Press ESC in 5 seconds to skip startup.nsh or any other key to continue.
Shell>
```

Then, locate the first partition of the SR ACS image, and move into the SBSA directory (\EFI\BOOT\bsa\sbsa):

```
Shell> FS0:
FS0:\> cd EFI\BOOT\bsa\sbsa
FS0:\EFI\BOOT\bsa\sbsa\> ls
Directory of: FS0:\EFI\BOOT\bsa\sbsa\
02/16/2022  20:22 <DIR>         4,096  .
02/16/2022  20:22 <DIR>         4,096  ..
02/16/2022  20:22             462,848  Sbsa.efi
          1 File(s)     462,848 bytes
          2 Dir(s)
```

From here, you can manually run the SBSA tests, with the following command:

```
FS0:\EFI\BOOT\bsa\sbsa\> Sbsa.efi -skip 800 -f SbsaResults.log
```

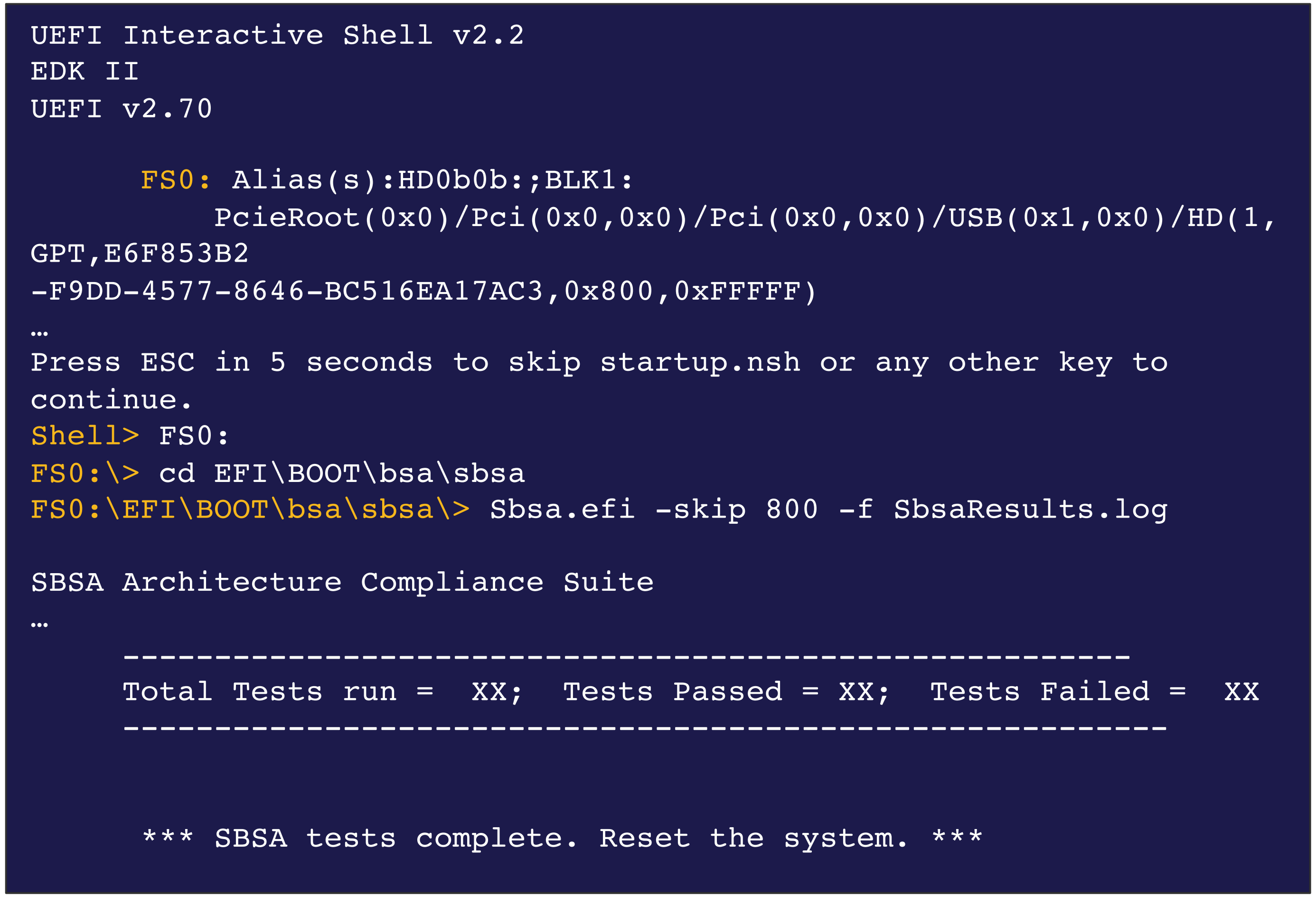
This will create a log file called SbsaResults.log, that we use for part in the Certification, please place it in `./manual-results/sbsa-uefi/` . No other tests are run with the SBBR compliant firmware.

LBBR steps

Certifying For LS with ACS-SR

SBSA tests under SBBR firmware

- Some of the SR ACS tests depend on having the UEFI pre-boot environment, which is not required to be present for LBBR compliance.
- This limitation is addressed by using an SBBR compliant firmware on the same system under LS certification to run the UEFI SBSA tests.
- After escaping the shell, running the SBSA tests are simple



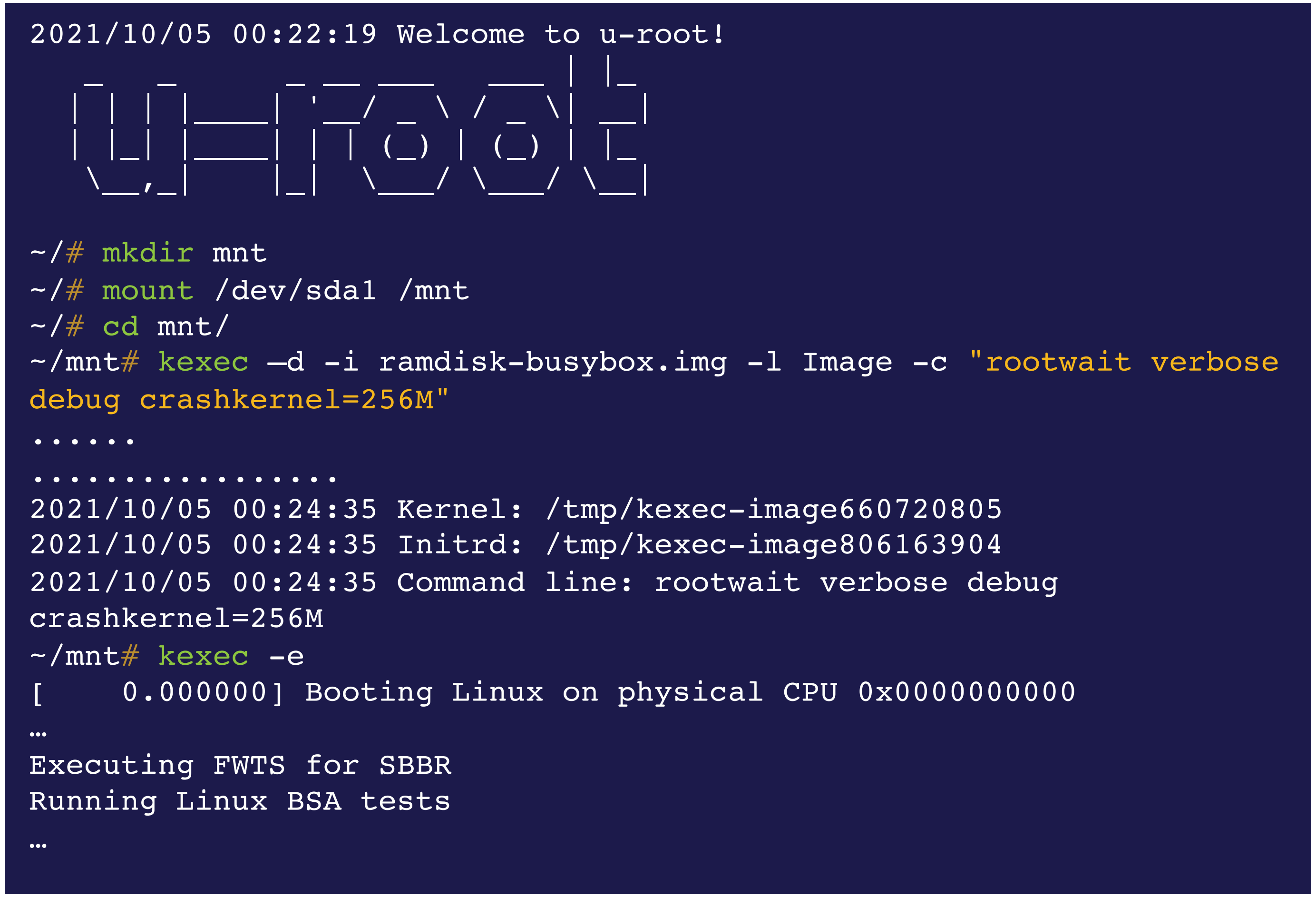
```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.70

      FS0: Alias(s):HD0b0b:;BLK1:
          PcieRoot(0x0)/Pci(0x0,0x0)/Pci(0x0,0x0)/USB(0x1,0x0)/HD(1,
GPT,E6F853B2
-F9DD-4577-8646-BC516EA17AC3,0x800,0xFFFFF)
…
Press ESC in 5 seconds to skip startup.nsh or any other key to
continue.
Shell> FS0:
FS0:\> cd EFI\BOOT\bsa\sbsa
FS0:\EFI\BOOT\bsa\sbsa\> Sbsa.efi -skip 800 -f SbsaResults.log

SBSA Architecture Compliance Suite
…
     -------------------------------------------------------
     Total Tests run =  XX;  Tests Passed = XX;  Tests Failed =  XX
     ---------------------------------------------------------


      *** SBSA tests complete. Reset the system. ***
```

https://gitlab.arm.com/systemready/systemready-LS-template

Certifying For LS with ACS-SR

ACS Linux from LBBR firmware

- The FWTS and BSA tests of the ACS test suite can utilize the LBBR firmware
- We can flash the system to LBBR firmware to complete the test suites.
- Run a kexec into the acs image and it will automatically run the remaining tests
- (FWTS tests for SBBR, but only test cases related to LBBR are analyzed)



```
2021/10/05 00:22:19 Welcome to u-root!
   _   _      _ __ ___   ___ | |_
  | | | |____| '__/ _ \ / _ \| __|
  | |_| |____| | | (_) | (_) | |_
   \__,_|    |_|  \___/ \___/ \__|

~/# mkdir mnt
~/# mount /dev/sda1 /mnt
~/# cd mnt/
~/mnt# kexec –d -i ramdisk-busybox.img -l Image -c "rootwait verbose
debug crashkernel=256M"
......
.................
2021/10/05 00:24:35 Kernel: /tmp/kexec-image660720805
2021/10/05 00:24:35 Initrd: /tmp/kexec-image806163904
2021/10/05 00:24:35 Command line: rootwait verbose debug
crashkernel=256M
~/mnt# kexec -e
[    0.000000] Booting Linux on physical CPU 0x0000000000
…
Executing FWTS for SBBR
Running Linux BSA tests
…
```

https://gitlab.arm.com/systemready/systemready-LS-template

OPEN SYSTEM FIRMWARE

System Showcase

Production and Proof of Concept systems

Connect. Collaborate. Accelerate.

OPEN Compute Project®

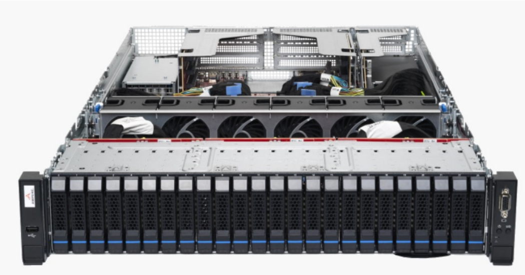
OPEN SYSTEM FIRMWARE

Ampere Mt. Jade: The First SystemReady LS Platform!

AMPERE

OCP ACCEPTED

- Arm SystemReady SR v2.0 certified And LS V0.9
- First Arm server as an OCP Accepted design in OpenCompute Project
- Firmware options both open-source (TianoCore EDK2, OpenBMC, LinuxBoot) and commercial (AMI Aptio, AMI MegaRAC)
- TianoCore : https://github.com/AmpereComputing/edk2-platforms/tree/ampere/Platform/Ampere/AmperePlatformPkg
- LinuxBoot : https://github.com/linuxboot/mainboards/tree/master/ampere/jade
- Ampere EDK2 + LinuxBoot integration include a build option to replace the EDK2 BDS phase with LinuxBoot & u-root UI. Ampere contributed LinuxBootBootManagerLib common library that can be used by other Arm systems for seamless Linuxboot transition.

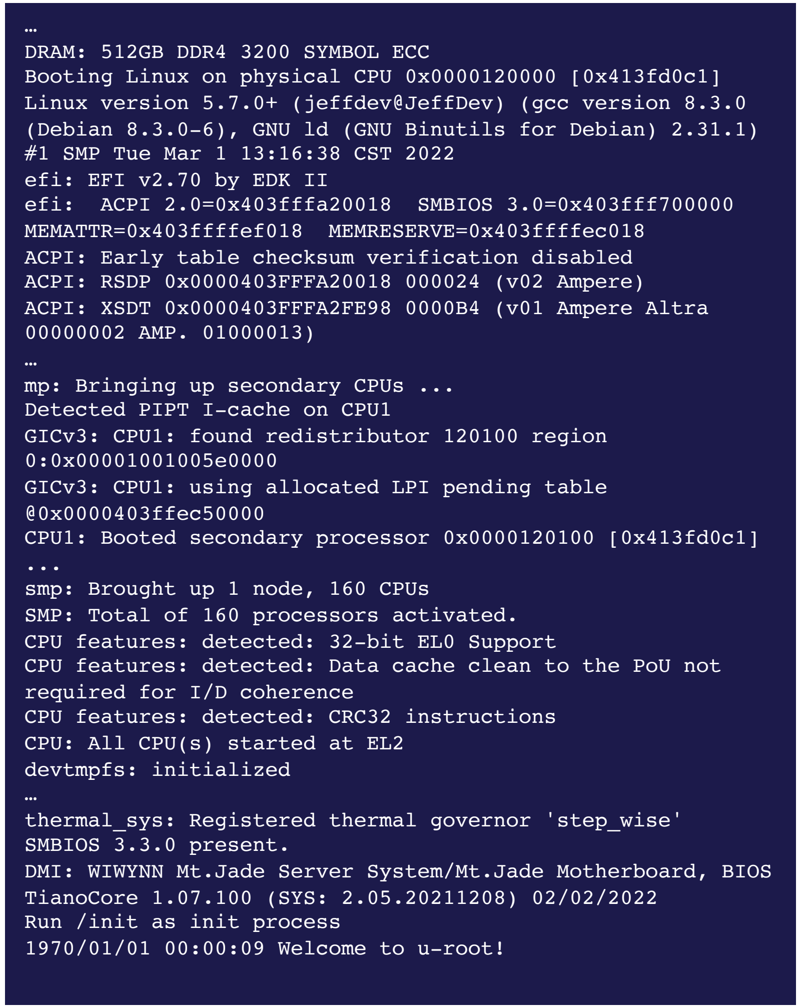


arm SystemReady SR CERTIFIED V2.0

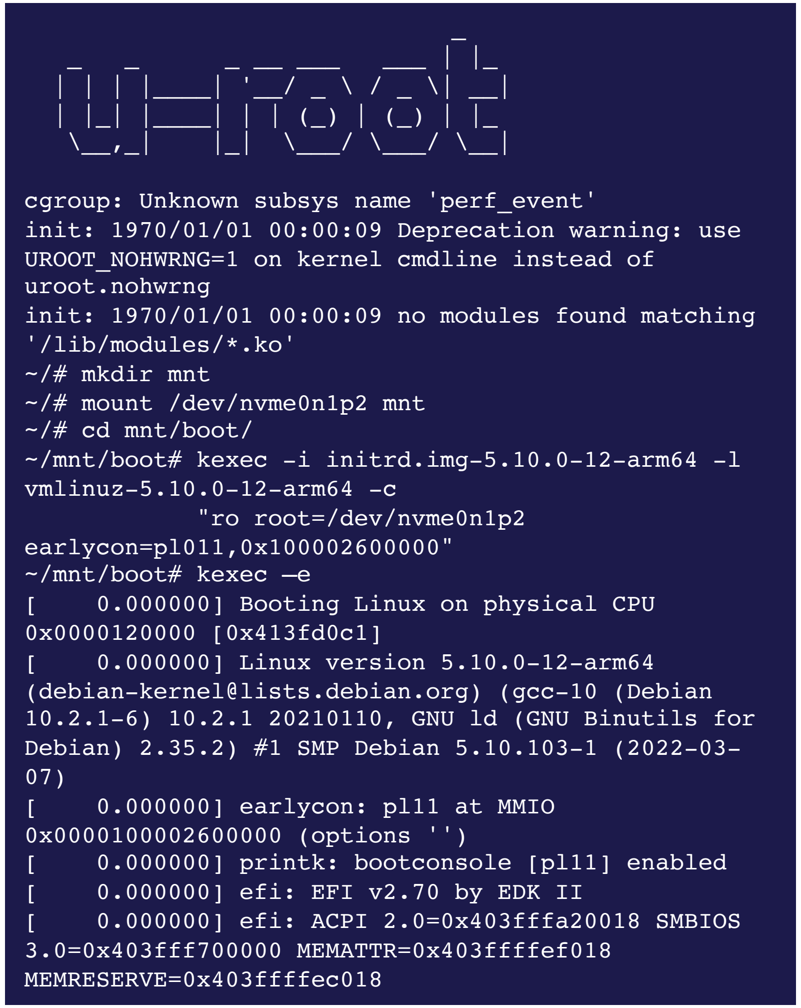
arm SystemReady LS CERTIFIED V0.9

Ampere Mt. Jade Seamless Boot into U-root

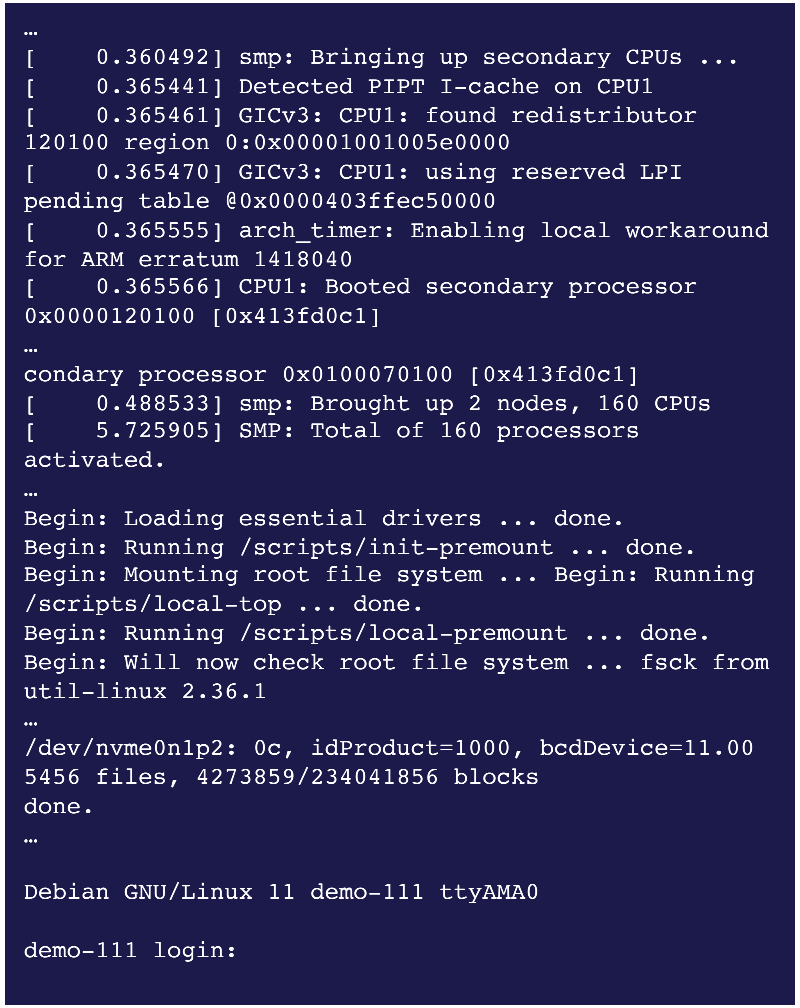
Truncated boot log, Seamless into u-root, manual into final OS



```
…
DRAM: 512GB DDR4 3200 SYMBOL ECC
Booting Linux on physical CPU 0x0000120000 [0x413fd0c1]
Linux version 5.7.0+ (jeffdev@JeffDev) (gcc version 8.3.0
(Debian 8.3.0-6), GNU ld (GNU Binutils for Debian) 2.31.1)
#1 SMP Tue Mar 1 13:16:38 CST 2022
efi: EFI v2.70 by EDK II
efi:  ACPI 2.0=0x403fffa20018  SMBIOS 3.0=0x403fff700000
MEMATTR=0x403ffffef018  MEMRESERVE=0x403ffffec018
ACPI: Early table checksum verification disabled
ACPI: RSDP 0x0000403FFFA20018 000024 (v02 Ampere)
ACPI: XSDT 0x0000403FFFA2FE98 0000B4 (v01 Ampere Altra
00000002 AMP. 01000013)
…
mp: Bringing up secondary CPUs ...
Detected PIPT I-cache on CPU1
GICv3: CPU1: found redistributor 120100 region
0:0x00001001005e0000
GICv3: CPU1: using allocated LPI pending table
@0x0000403ffec50000
CPU1: Booted secondary processor 0x0000120100 [0x413fd0c1]
...
smp: Brought up 1 node, 160 CPUs
SMP: Total of 160 processors activated.
CPU features: detected: 32-bit EL0 Support
CPU features: detected: Data cache clean to the PoU not
required for I/D coherence
CPU features: detected: CRC32 instructions
CPU: All CPU(s) started at EL2
devtmpfs: initialized
…
thermal_sys: Registered thermal governor 'step_wise'
SMBIOS 3.3.0 present.
DMI: WIWYNN Mt.Jade Server System/Mt.Jade Motherboard, BIOS
TianoCore 1.07.100 (SYS: 2.05.20211208) 02/02/2022
Run /init as init process
1970/01/01 00:00:09 Welcome to u-root!
```



```
                              _
  _   _      _ __ ___   ___ | |_
 | | | |____| '__/ _ \ / _ \| __|
 | |_| |____| | | (_) | (_) | |_
  \__,_|    |_|  \___/ \___/ \__|

cgroup: Unknown subsys name 'perf_event'
init: 1970/01/01 00:00:09 Deprecation warning: use
UROOT_NOHWRNG=1 on kernel cmdline instead of
uroot.nohwrng
init: 1970/01/01 00:00:09 no modules found matching
'/lib/modules/*.ko'
~/# mkdir mnt
~/# mount /dev/nvme0n1p2 mnt
~/# cd mnt/boot/
~/mnt/boot# kexec -i initrd.img-5.10.0-12-arm64 -l
vmlinuz-5.10.0-12-arm64 -c
              "ro root=/dev/nvme0n1p2
earlycon=pl011,0x100002600000"
~/mnt/boot# kexec –e
[    0.000000] Booting Linux on physical CPU
0x0000120000 [0x413fd0c1]
[    0.000000] Linux version 5.10.0-12-arm64
(debian-kernel@lists.debian.org) (gcc-10 (Debian
10.2.1-6) 10.2.1 20210110, GNU ld (GNU Binutils for
Debian) 2.35.2) #1 SMP Debian 5.10.103-1 (2022-03-
07)
[    0.000000] earlycon: pl11 at MMIO
0x0000100002600000 (options '')
[    0.000000] printk: bootconsole [pl11] enabled
[    0.000000] efi: EFI v2.70 by EDK II
[    0.000000] efi: ACPI 2.0=0x403fffa20018 SMBIOS
3.0=0x403fff700000 MEMATTR=0x403ffffef018
MEMRESERVE=0x403ffffec018
```

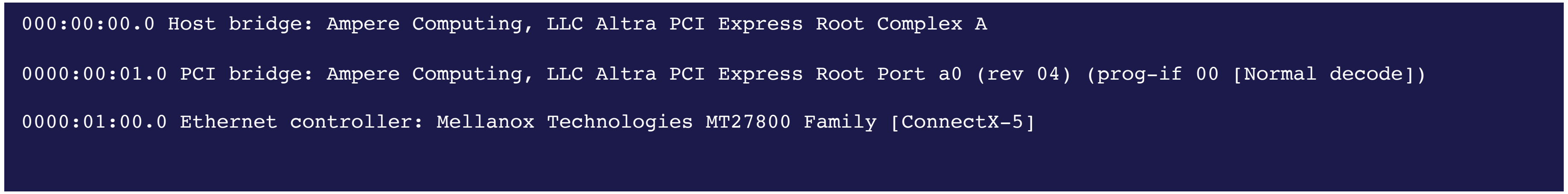


```
…
[    0.360492] smp: Bringing up secondary CPUs ...
[    0.365441] Detected PIPT I-cache on CPU1
[    0.365461] GICv3: CPU1: found redistributor
120100 region 0:0x00001001005e0000
[    0.365470] GICv3: CPU1: using reserved LPI
pending table @0x0000403ffec50000
[    0.365555] arch_timer: Enabling local workaround
for ARM erratum 1418040
[    0.365566] CPU1: Booted secondary processor
0x0000120100 [0x413fd0c1]
…
condary processor 0x0100070100 [0x413fd0c1]
[    0.488533] smp: Brought up 2 nodes, 160 CPUs
[    5.725905] SMP: Total of 160 processors
activated.
…
Begin: Loading essential drivers ... done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... Begin: Running
/scripts/local-top ... done.
Begin: Running /scripts/local-premount ... done.
Begin: Will now check root file system ... fsck from
util-linux 2.36.1
…
/dev/nvme0n1p2: 0c, idProduct=1000, bcdDevice=11.00
5456 files, 4273859/234041856 blocks
done.
…

Debian GNU/Linux 11 demo-111 ttyAMA0

demo-111 login:
```

OS logs

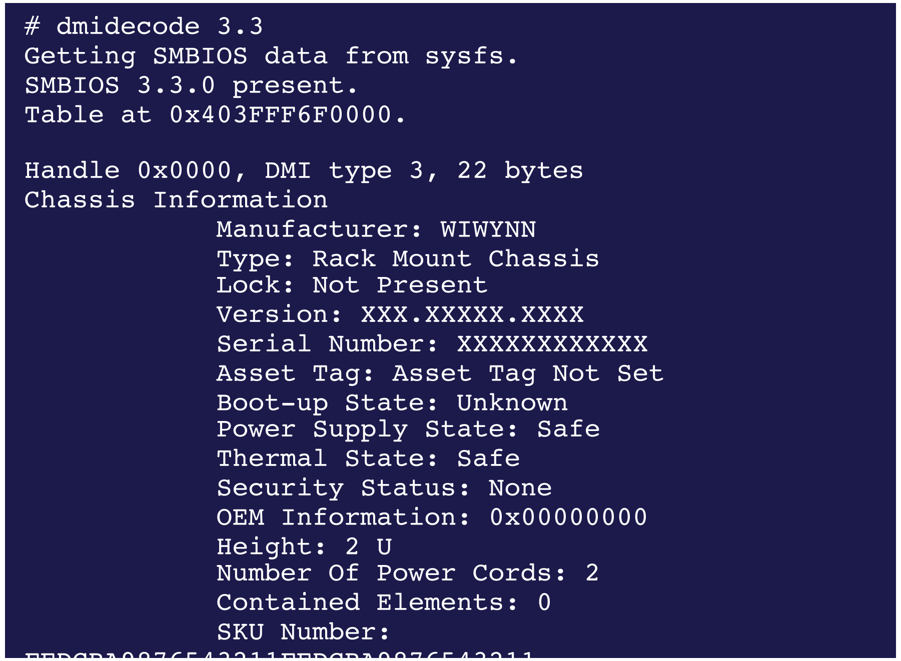
(Truncated for space)

- lspci



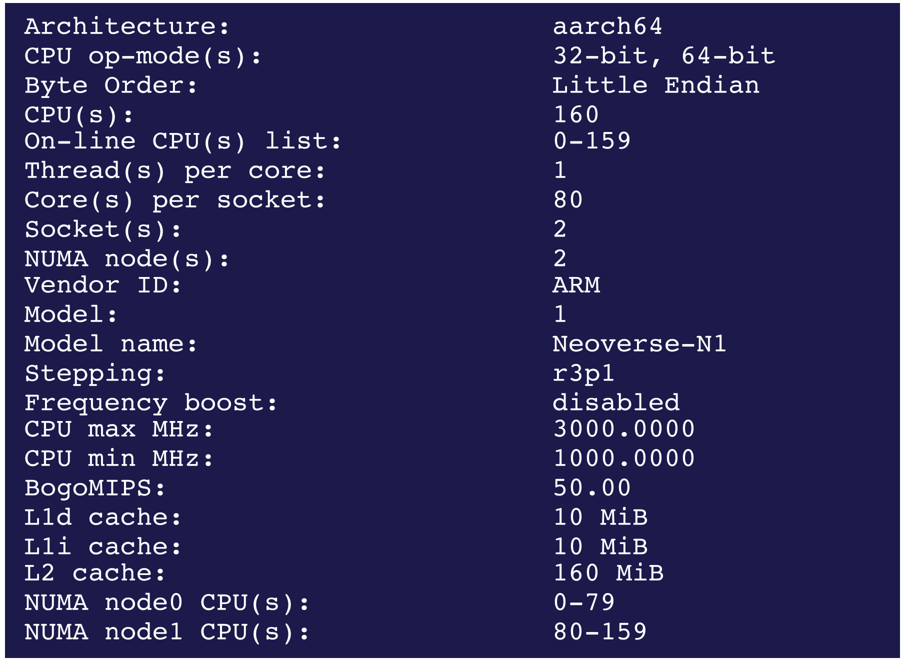
```
000:00:00.0 Host bridge: Ampere Computing, LLC Altra PCI Express Root Complex A

0000:00:01.0 PCI bridge: Ampere Computing, LLC Altra PCI Express Root Port a0 (rev 04) (prog-if 00 [Normal decode])

0000:01:00.0 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
```

- dmidecode



```
# dmidecode 3.3
Getting SMBIOS data from sysfs.
SMBIOS 3.3.0 present.
Table at 0x403FFF6F0000.

Handle 0x0000, DMI type 3, 22 bytes
Chassis Information
            Manufacturer: WIWYNN
            Type: Rack Mount Chassis
            Lock: Not Present
            Version: XXX.XXXXX.XXXX
            Serial Number: XXXXXXXXXXXX
            Asset Tag: Asset Tag Not Set
            Boot-up State: Unknown
            Power Supply State: Safe
            Thermal State: Safe
            Security Status: None
            OEM Information: 0x00000000
            Height: 2 U
            Number Of Power Cords: 2
            Contained Elements: 0
            SKU Number:
FEDCBA9876543211FEDCBA9876543211
```

- lscpu



```
Architecture:                    aarch64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
CPU(s):                          160
On-line CPU(s) list:             0-159
Thread(s) per core:              1
Core(s) per socket:              80
Socket(s):                       2
NUMA node(s):                    2
Vendor ID:                       ARM
Model:                           1
Model name:                      Neoverse-N1
Stepping:                        r3p1
Frequency boost:                 disabled
CPU max MHz:                     3000.0000
CPU min MHz:                     1000.0000
BogoMIPS:                        50.00
L1d cache:                       10 MiB
L1i cache:                       10 MiB
L2 cache:                        160 MiB
NUMA node0 CPU(s):               0-79
NUMA node1 CPU(s):               80-159
```

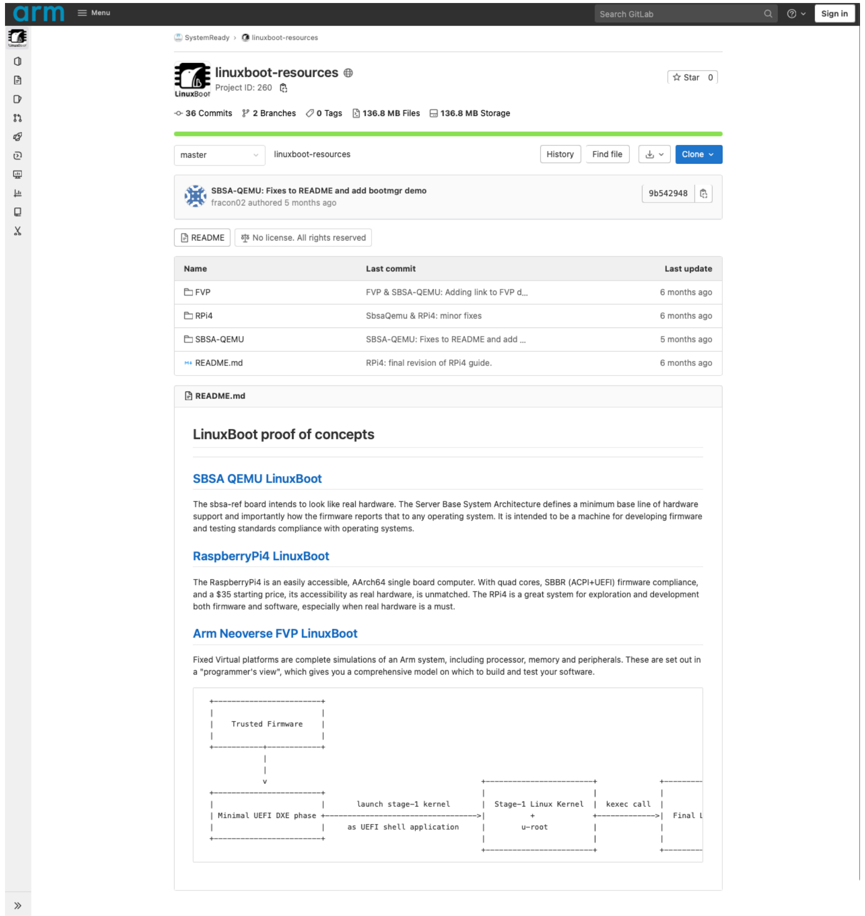
Arm LinuxBoot PoC repo

- Arm Gitlab repo with arm several proof of concepts systems
- Several PoC's in development
- sbsa-ref QEMU
- Virtual platform supporting SBSA Specs
- Guide for building firmware for LinuxBoot and booting OS.
- target of future upstreaming

- RaspberryPi
- Easily accessible, real hardware
- Guide for building firmware for LinuxBoot and booting OS.

- Arm Neoverse FVP
- complete simulation of an Arm system, including processor, memory and peripherals
- Guide for setting up FVP, building firmware image and booting OS.

- Will be updated as LBBR and SystemReady progresses



linuxboot-resources

Project ID: 260

36 Commits 2 Branches 0 Tags 136.8 MB Files 136.8 MB Storage

linuxboot-resources

SBSA-QEMU: Fixes to README and add bootmgr demo

fracon02 authored 5 months ago

9b542948

No license. All rights reserved

Name	Last commit	Last update
FVP	FVP & SBSA-QEMU: Adding link to FVP d...	6 months ago
RPi4	SbsaQemu & RPi4: minor fixes	6 months ago
SBSA-QEMU	SBSA-QEMU: Fixes to README and add ...	5 months ago
README.md	RPi4: final revision of RPi4 guide.	6 months ago

README.md

LinuxBoot proof of concepts

SBSA QEMU LinuxBoot

The sbsa-ref board intends to look like real hardware. The Server Base System Architecture defines a minimum base line of hardware support and importantly how the firmware reports that to any operating system. It is intended to be a machine for developing firmware and testing standards compliance with operating systems.

RaspberryPi4 LinuxBoot

The RaspberryPi4 is an easily accessible, AArch64 single board computer. With quad cores, SBBR (ACPI+UEFI) firmware compliance, and a $35 starting price, its accessibility as real hardware, is unmatched. The RPi4 is a great system for exploration and development both firmware and software, especially when real hardware is a must.

Arm Neoverse FVP LinuxBoot

Fixed Virtual platforms are complete simulations of an Arm system, including processor, memory and peripherals. These are set out in a "programmer's view", which gives you a comprehensive model on which to build and test your software.

```
+--------------------+
|                    |
|  Trusted Firmware  |
|                    |
+---------+----------+
          |
          |
          v                                              +--------------------+            +--------
+--------------------+                                   |                    |            |
|                    |      launch stage-1 kernel        | Stage-1 Linux Kernel | kexec call |
| Minimal UEFI DXE phase +------------------------------->|         +          +----------->|  Final L
|                    |     as UEFI shell application     |       u-root       |            |
+--------------------+                                   |                    |            |
                                                         +--------------------+            +--------
```

https://gitlab.arm.com/systemready/linuxboot-resources

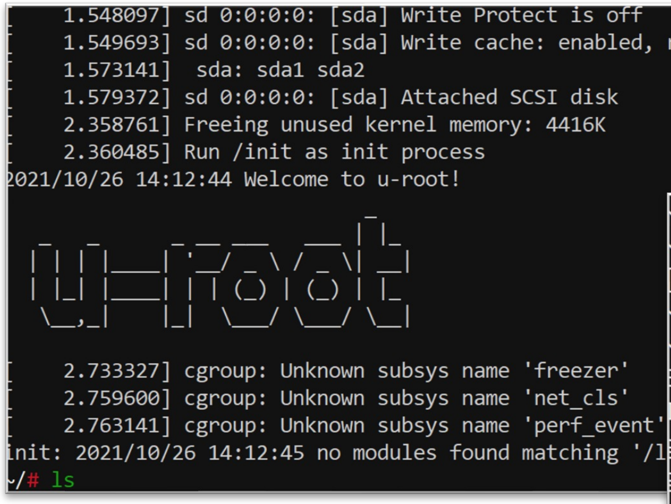
sbsa-ref QEMU

- Virtual platform for Armv8-A, with support for Arm SBSA specifications
  - Available as “sbsa-ref” machine
  - Supports SBSA HW such as GICv3, generic timer, watchdog, etc..
  - Choice as an environment for developing firmware and testing operating systems and compliance testing
- Linaro working on completing SBSA and SBBR support Upstream and testing compliance with the ACS test suite
  - https://github.com/tianocore/edk2-platforms/tree/master/Platform/Qemu/SbsaQemu
- Fujitsu presentation on LinuxBoot running on top of UEFI in QEMU
  - https://sysadmin.miniconf.org/2021/lca2021-Naohiro_Tamra-LinuxBoot_AArch64.pdf
    - Building EDK2 with extended firmware volume, Build image with kernel & initramfs, Use UTK to replace UEFI shell with this Linux Image
- Arm presentation on LinuxBoot running on top of UEFI in QEMU
  - https://talks.osfc.io/media/osfc2021/submissions/NVDFNC/resources/LBBR_OSFC_2021_Bg6kGLT.pdf
    - Building Linux boot into an edk2 image, work on removing DXEs

PoC demo: sbsa-ref QEMU & ACS image

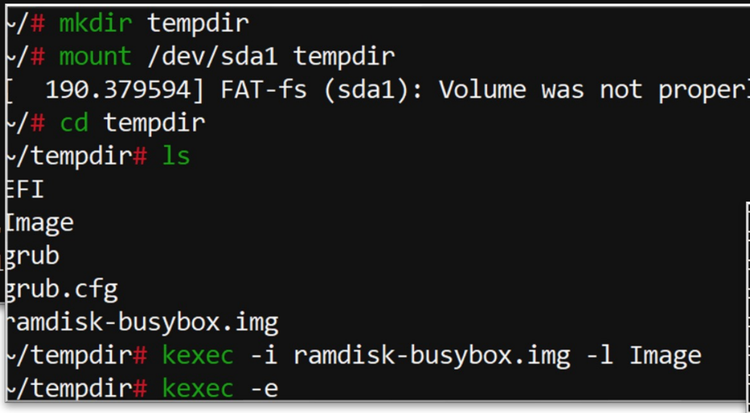
OPEN SYSTEM FIRMWARE

Booting Kernel + BusyBox, and running Arm SystemReady ACS (Architectural Compliance test Suite)



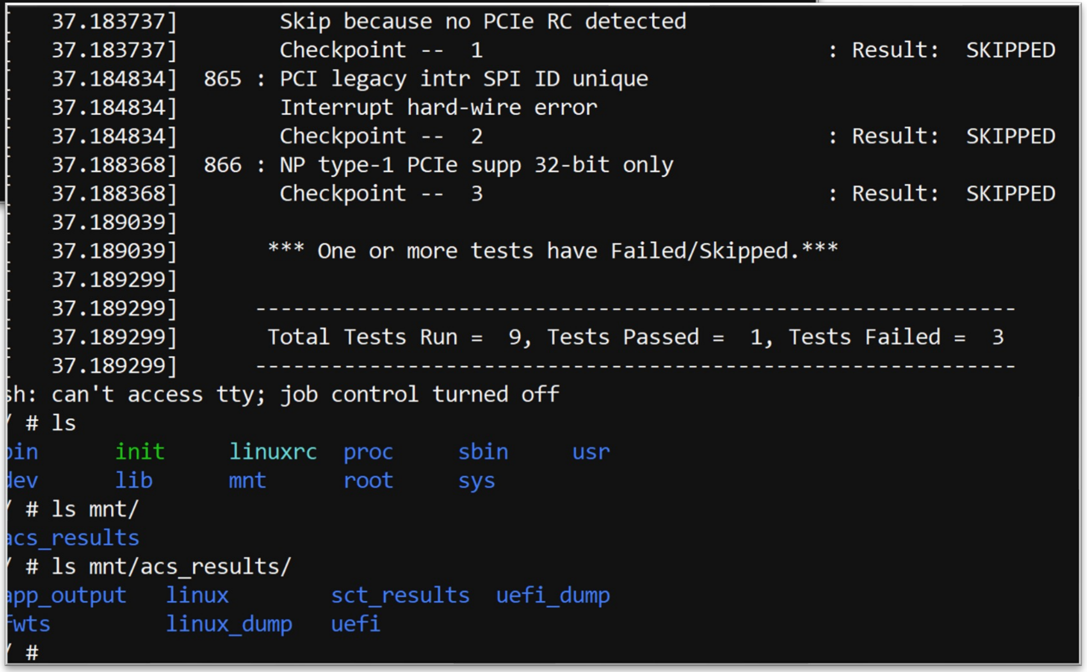
```
[    1.548097] sd 0:0:0:0: [sda] Write Protect is off
[    1.549693] sd 0:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
[    1.573141]  sda: sda1 sda2
[    1.579372] sd 0:0:0:0: [sda] Attached SCSI disk
[    2.358761] Freeing unused kernel memory: 4416K
[    2.360485] Run /init as init process
2021/10/26 14:12:44 Welcome to u-root!
                              _
  _   _      _ __ ___   ___ | |_
 | | | |____| '__/ _ \ / _ \| __|
 | |_| |____| | | (_) | (_) | |_
  \__,_|    |_|  \___/ \___/ \__|

[    2.733327] cgroup: Unknown subsys name 'freezer'
[    2.759600] cgroup: Unknown subsys name 'net_cls'
[    2.763141] cgroup: Unknown subsys name 'perf_event'
init: 2021/10/26 14:12:45 no modules found matching '/l
~/# ls
```

Enter u-root & first stage kernel, greeted with a command line



```
~/# mkdir tempdir
~/# mount /dev/sda1 tempdir
[  190.379594] FAT-fs (sda1): Volume was not properly unmounted. Some data may be corrupt. Please run fsck.
~/# cd tempdir
~/tempdir# ls
EFI
Image
grub
grub.cfg
ramdisk-busybox.img
~/tempdir# kexec -i ramdisk-busybox.img -l Image
~/tempdir# kexec -e
```

Setting up & kexec, this process can be automated via u-root's SystemBoot and Uinit script



```
[   37.183737]        Skip because no PCIe RC detected
[   37.183737]        Checkpoint --  1                            : Result:  SKIPPED
[   37.184834]  865 : PCI legacy intr SPI ID unique
[   37.184834]        Interrupt hard-wire error
[   37.184834]        Checkpoint --  2                            : Result:  SKIPPED
[   37.188368]  866 : NP type-1 PCIe supp 32-bit only
[   37.188368]        Checkpoint --  3                            : Result:  SKIPPED
[   37.189039]
[   37.189039]       *** One or more tests have Failed/Skipped.***
[   37.189299]
[   37.189299]      ------------------------------------------------------------
[   37.189299]       Total Tests Run =  9, Tests Passed =  1, Tests Failed =  3
[   37.189299]      ------------------------------------------------------------
sh: can't access tty; job control turned off
/ # ls
bin       init      linuxrc  proc     sbin     usr
dev       lib       mnt      root     sys
/ # ls mnt/
acs_results
/ # ls mnt/acs_results/
app_output  linux        sct_results  uefi_dump
fwts        linux_dump   uefi
/ #
```

Booting into ACS and running tests

LinuxBoot + Arm: Raspberry Pi 4 Model B

- Arm SystemReady ES and IR certified
- Fully open-source firmware community project
- TianoCore: https://github.com/tianocore/edk2-platforms/tree/master/Platform/RaspberryPi
- Discord community: https://discord.gg/VfYbkfp
- Arm working on LinuxBoot support PoC. Not a target of SystemReady LS certification.
  - IoT/Embedded market is targeted by SystemReady ES and IR
  - RPi4 HW popularity and availability (And open-source community FW) makes it attractive for PoC
  - Repo containing PoC code: https://gitlab.arm.com/systemready/linuxboot-resources/-/tree/master/RPi4

BROADCOM

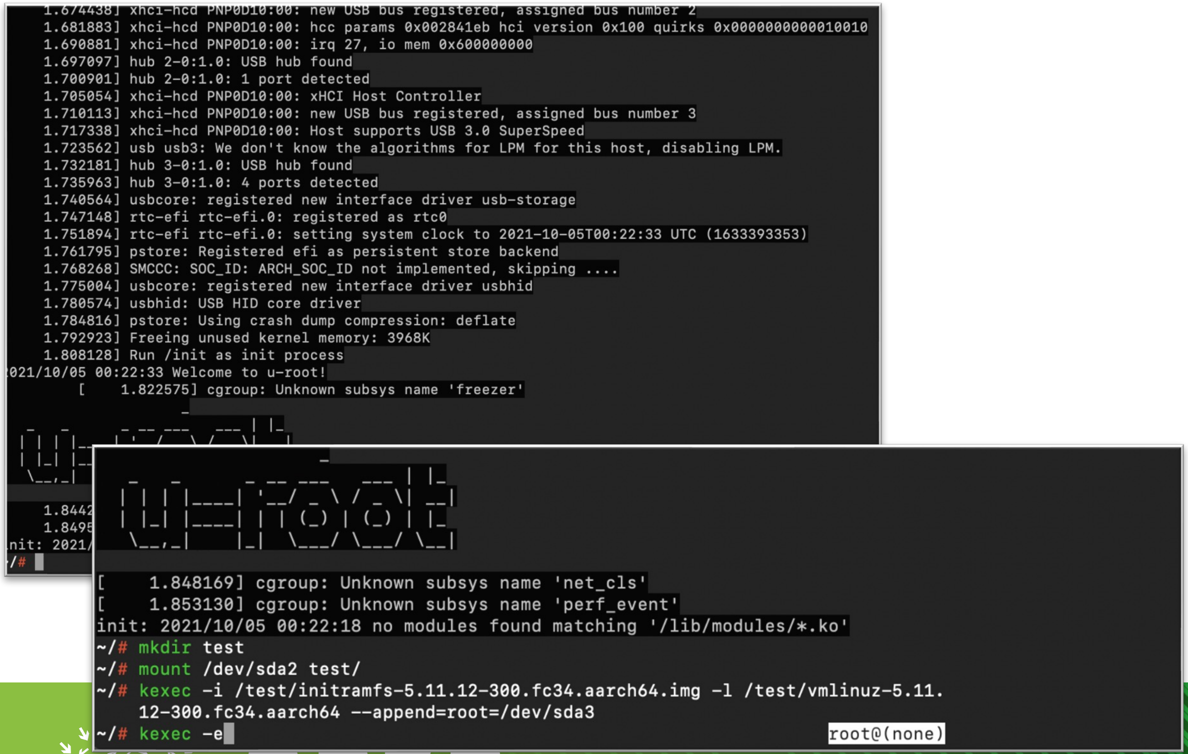
arm SystemReady ES CERTIFIED V1.0

arm SystemReady IR CERTIFIED V1.0

PoC demo: RPi4 & Fedora 34

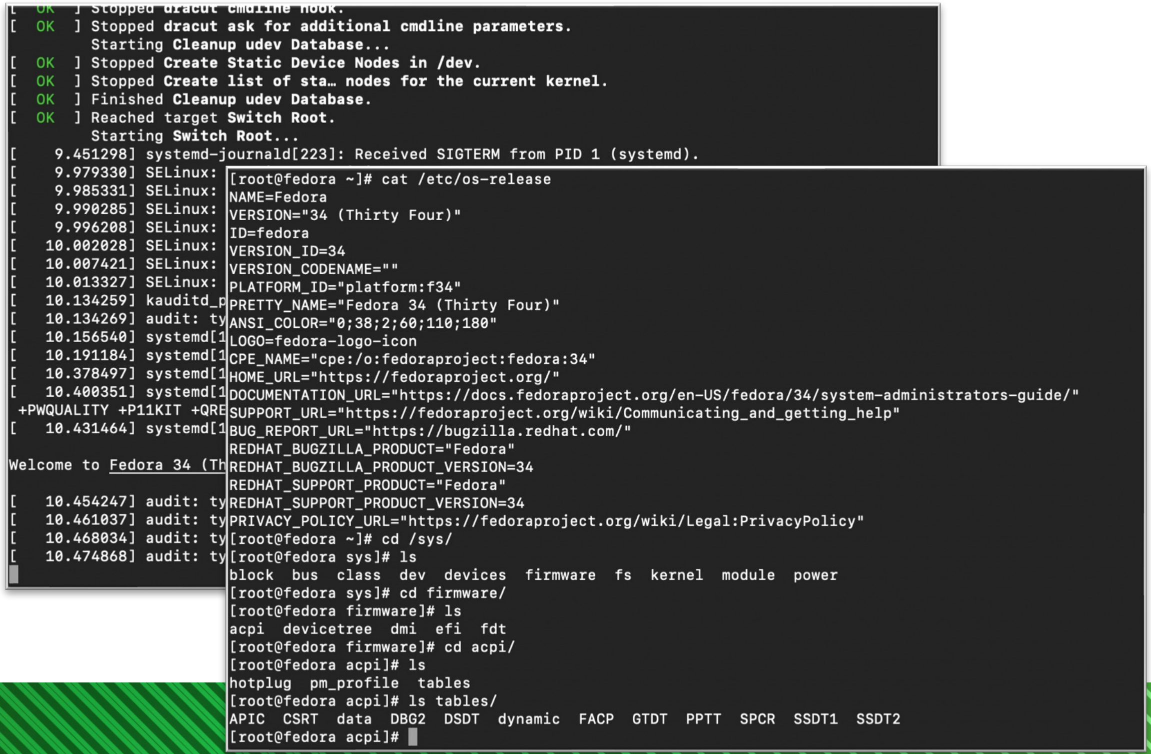
Setup and execute kexec.
(can automated with systemboot or Uinit)

Boot Fedora 34, with ACPI



```
    1.674438] xhci-hcd PNP0D10:00: new USB bus registered, assigned bus number 2
    1.681883] xhci-hcd PNP0D10:00: hcc params 0x002841eb hci version 0x100 quirks 0x0000000000010010
    1.690881] xhci-hcd PNP0D10:00: irq 27, io mem 0x600000000
    1.697097] hub 2-0:1.0: USB hub found
    1.700901] hub 2-0:1.0: 1 port detected
    1.705054] xhci-hcd PNP0D10:00: xHCI Host Controller
    1.710113] xhci-hcd PNP0D10:00: new USB bus registered, assigned bus number 3
    1.717338] xhci-hcd PNP0D10:00: Host supports USB 3.0 SuperSpeed
    1.723562] usb usb3: We don't know the algorithms for LPM for this host, disabling LPM.
    1.732181] hub 3-0:1.0: USB hub found
    1.735963] hub 3-0:1.0: 4 ports detected
    1.740564] usbcore: registered new interface driver usb-storage
    1.747148] rtc-efi rtc-efi.0: registered as rtc0
    1.751894] rtc-efi rtc-efi.0: setting system clock to 2021-10-05T00:22:33 UTC (1633393353)
    1.761795] pstore: Registered efi as persistent store backend
    1.768268] SMCCC: SOC_ID: ARCH_SOC_ID not implemented, skipping ....
    1.775004] usbcore: registered new interface driver usbhid
    1.780574] usbhid: USB HID core driver
    1.784816] pstore: Using crash dump compression: deflate
    1.792923] Freeing unused kernel memory: 3968K
    1.808128] Run /init as init process
021/10/05 00:22:33 Welcome to u-root!
       [    1.822575] cgroup: Unknown subsys name 'freezer'
    1.8442
    1.8495
nit: 2021/
/#
```

```
                _
  _   _      _ __ ___   ___  | |_
 | | | |____| '__/ _ \ / _ \| __|
 | |_| |____| | | (_) | (_) | |_
  \__,_|    |_|  \___/ \___/ \__|

[    1.848169] cgroup: Unknown subsys name 'net_cls'
[    1.853130] cgroup: Unknown subsys name 'perf_event'
init: 2021/10/05 00:22:18 no modules found matching '/lib/modules/*.ko'
~/# mkdir test
~/# mount /dev/sda2 test/
~/# kexec -i /test/initramfs-5.11.12-300.fc34.aarch64.img -l /test/vmlinuz-5.11.
    12-300.fc34.aarch64 --append=root=/dev/sda3
~/# kexec -e                                                  root@(none)
```

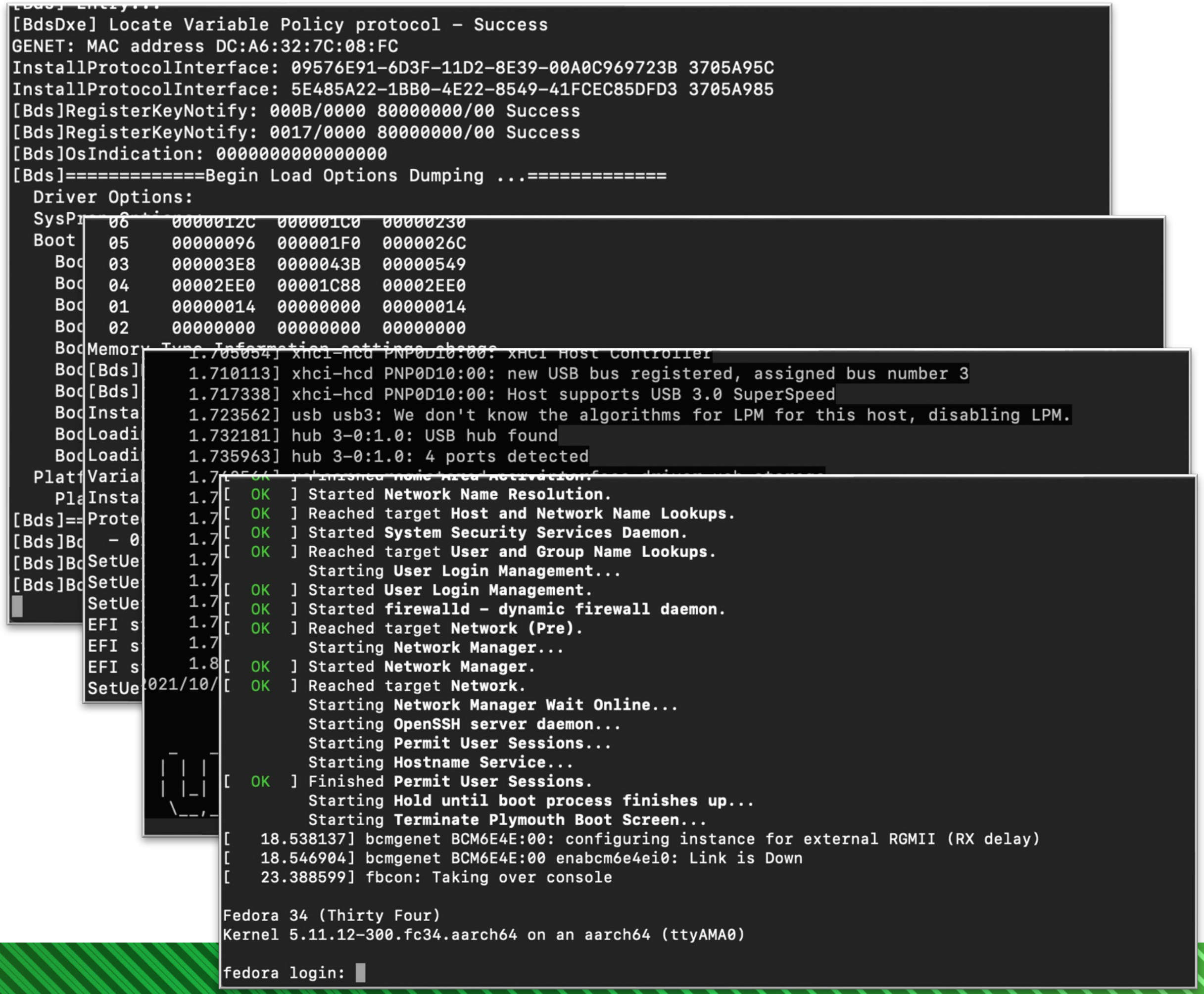


```
[  OK  ] Stopped dracut cmdline hook.
[  OK  ] Stopped dracut ask for additional cmdline parameters.
         Starting Cleanup udev Database...
[  OK  ] Stopped Create Static Device Nodes in /dev.
[  OK  ] Stopped Create list of sta… nodes for the current kernel.
[  OK  ] Finished Cleanup udev Database.
[  OK  ] Reached target Switch Root.
         Starting Switch Root...
[    9.451298] systemd-journald[223]: Received SIGTERM from PID 1 (systemd).
[    9.979330] SELinux:
[    9.985331] SELinux:
[    9.990285] SELinux:
[    9.996208] SELinux:
[   10.002028] SELinux:
[   10.007421] SELinux:
[   10.013327] SELinux:
[   10.134259] kauditd_p
[   10.134269] audit: ty
[   10.156540] systemd[1
[   10.191184] systemd[1
[   10.378497] systemd[1
[   10.400351] systemd[1
 +PWQUALITY +P11KIT +QRE
[   10.431464] systemd[1

Welcome to Fedora 34 (Th

[   10.454247] audit: ty
[   10.461037] audit: ty
[   10.468034] audit: ty
[   10.474868] audit: ty
```

```
[root@fedora ~]# cat /etc/os-release
NAME=Fedora
VERSION="34 (Thirty Four)"
ID=fedora
VERSION_ID=34
VERSION_CODENAME=""
PLATFORM_ID="platform:f34"
PRETTY_NAME="Fedora 34 (Thirty Four)"
ANSI_COLOR="0;38;2;60;110;180"
LOGO=fedora-logo-icon
CPE_NAME="cpe:/o:fedoraproject:fedora:34"
HOME_URL="https://fedoraproject.org/"
DOCUMENTATION_URL="https://docs.fedoraproject.org/en-US/fedora/34/system-administrators-guide/"
SUPPORT_URL="https://fedoraproject.org/wiki/Communicating_and_getting_help"
BUG_REPORT_URL="https://bugzilla.redhat.com/"
REDHAT_BUGZILLA_PRODUCT="Fedora"
REDHAT_BUGZILLA_PRODUCT_VERSION=34
REDHAT_SUPPORT_PRODUCT="Fedora"
REDHAT_SUPPORT_PRODUCT_VERSION=34
PRIVACY_POLICY_URL="https://fedoraproject.org/wiki/Legal:PrivacyPolicy"
[root@fedora ~]# cd /sys/
[root@fedora sys]# ls
block  bus  class  dev  devices  firmware  fs  kernel  module  power
[root@fedora sys]# cd firmware/
[root@fedora firmware]# ls
acpi  devicetree  dmi  efi  fdt
[root@fedora firmware]# cd acpi/
[root@fedora acpi]# ls
hotplug  pm_profile  tables
[root@fedora acpi]# ls tables/
APIC  CSRT  data  DBG2  DSDT  dynamic  FACP  GTDT  PPTT  SPCR  SSDT1  SSDT2
[root@fedora acpi]#
```

Future work: PoCs

- Seamless LinuxBoot transitions
  - LinuxBootBootManagerLib integration into Arm PoCs & Integration of U-root SytemBoot bootloaders (represented to the right)
- Firmware size Reduction
  - Removal of DXEs that are no longer needed
  - Reduction of Kernel & initramfs size
- Upstream
  - Contribute refence code and platforms to EDK2
  - Update LinuxBoot Resources Repo as work progresses
- Investigate LBBR-Vn features
  - Payload Package?
  - Alterative hardware initialization firmware?



```
[BdsDxe] Locate Variable Policy protocol - Success
GENET: MAC address DC:A6:32:7C:08:FC
InstallProtocolInterface: 09576E91-6D3F-11D2-8E39-00A0C969723B 3705A95C
InstallProtocolInterface: 5E485A22-1BB0-4E22-8549-41FCEC85DFD3 3705A985
[Bds]RegisterKeyNotify: 000B/0000 80000000/00 Success
[Bds]RegisterKeyNotify: 0017/0000 80000000/00 Success
[Bds]OsIndication: 0000000000000000
[Bds]============Begin Load Options Dumping ...=============
  Driver Options:
  SysPr
  Boot
    Boo
    Boo
    Boo
    Boo
    Boo
    Boo
    Boo
    Boo
    Boo
    Boo
  Platf
    Pla
[Bds]==
[Bds]Bo
[Bds]Bo
[Bds]Bo
```

```
  05     00000096  000001F0  0000026C
  03     000003E8  0000043B  00000549
  04     00002EE0  00001C88  00002EE0
  01     00000014  00000000  00000014
  02     00000000  00000000  00000000
Memory
[Bds]
[Bds]
Insta
Loadi
Loadi
Varia
Insta
Prote
  - 0
SetUe
SetUe
SetUe
EFI s
EFI s
EFI s
SetUe
```

```
    1.710113] xhci-hcd PNP0D10:00: new USB bus registered, assigned bus number 3
    1.717338] xhci-hcd PNP0D10:00: Host supports USB 3.0 SuperSpeed
    1.723562] usb usb3: We don't know the algorithms for LPM for this host, disabling LPM.
    1.732181] hub 3-0:1.0: USB hub found
    1.735963] hub 3-0:1.0: 4 ports detected
2021/10/
```

```
[  OK  ] Started Network Name Resolution.
[  OK  ] Reached target Host and Network Name Lookups.
[  OK  ] Started System Security Services Daemon.
[  OK  ] Reached target User and Group Name Lookups.
         Starting User Login Management...
[  OK  ] Started User Login Management.
[  OK  ] Started firewalld - dynamic firewall daemon.
[  OK  ] Reached target Network (Pre).
         Starting Network Manager...
[  OK  ] Started Network Manager.
[  OK  ] Reached target Network.
         Starting Network Manager Wait Online...
         Starting OpenSSH server daemon...
         Starting Permit User Sessions...
         Starting Hostname Service...
[  OK  ] Finished Permit User Sessions.
         Starting Hold until boot process finishes up...
         Starting Terminate Plymouth Boot Screen...
[   18.538137] bcmgenet BCM6E4E:00: configuring instance for external RGMII (RX delay)
[   18.546904] bcmgenet BCM6E4E:00 enabcm6e4ei0: Link is Down
[   23.388599] fbcon: Taking over console

Fedora 34 (Thirty Four)
Kernel 5.11.12-300.fc34.aarch64 on an aarch64 (ttyAMA0)

fedora login:
```

Recap

Review

- SystemReady and SystemReady LS
- LinuxBoot

Standards for SystemReady LS

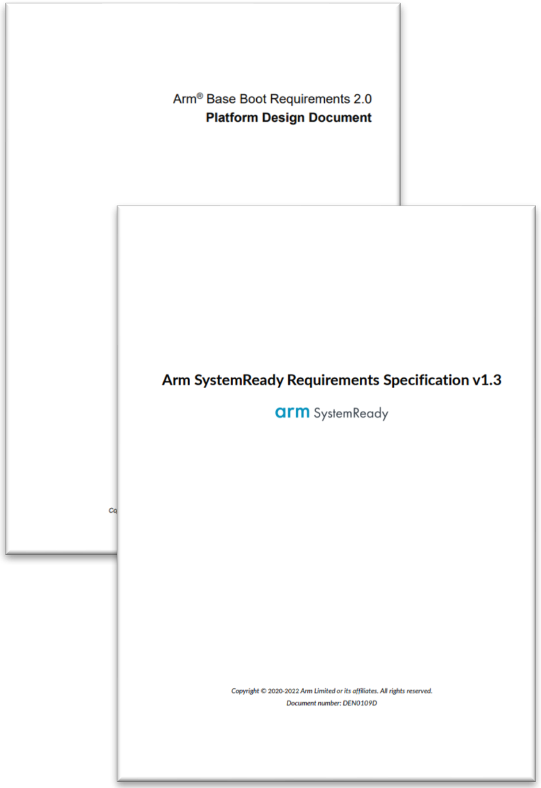
- LBBR-V1 & LBBR-Vn

SystemReady LS V0.9

- Certification requirements and processes

System Showcase

- Showcase production systems
- Showcase of proof of concepts (PoC) systems and resources

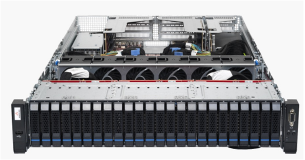


Arm® Base Boot Requirements 2.0
Platform Design Document

Arm SystemReady Requirements Specification v1.3

arm SystemReady

Copyright © 2020-2022 Arm Limited or its affiliates. All rights reserved.
Document number: DEN0109D



Reach out, get involved!

Arm SystemReady and LinuxBoot resources:

- SystemReady Certification Program Website

https://www.arm.com/systemready-certification-program

- Arm LinuxBoot PoCs, Code changes, video demos

https://gitlab.arm.com/systemready/linuxboot-resources

- SystemReady-LS-template Repo for SystemReady certation procedure and guidance

https://gitlab.arm.com/systemready/systemready-LS-template

- Contact systemready@arm.com
- Speaker Jeff.Booher-Kaeding@arm.com