UEFI+Linux on ARM Making it "Just Work"

UEFI Summerfest – July 15-19, 2013

Presented by Grant Likely

Linaro, Ltd,

What is Linaro?

Open source software for ARM SoC

“Linaro is a not-for-profit engineering organization consolidating and optimizing open source Linux software and tools for the ARM architecture.” – Linaro Website

In other words, we make Linux run well on ARM

Agenda

- Introduction
- Enterprise
- Embedded & Mobile
- Future Work?
- Questions

UEFI+Linux on ARM

Enterprise

Enterprise: Rationale

- ARM servers must be familiar and predictable
- Linux on ARM should look much like Linux on a PC
  - Without breaking existing users

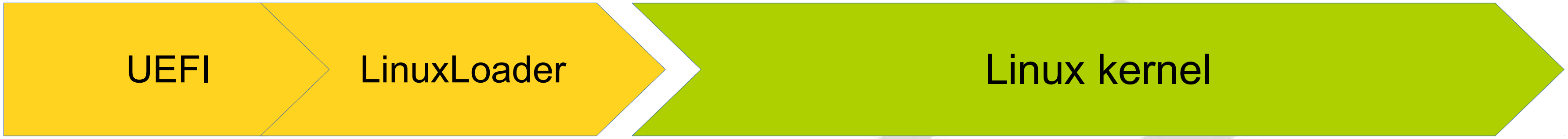
Enterprise: LEG

- ARM servers need to run Linux
- ARM servers will use UEFI firmware
- What does both UEFI and Linux need for Linux to run seamlessly on ARM?
- Linaro Enterprise Group (LEG) has a UEFI team making sure Linux runs well

Requirements

- Linux Boot Manager and OS Loader
- Local storage Boot
- Network Boot
- Serial Console
- Userspace UEFI tools
  - dmidecode
  - efibootmgr
- Hypervisor
- kexec

LinuxLoader

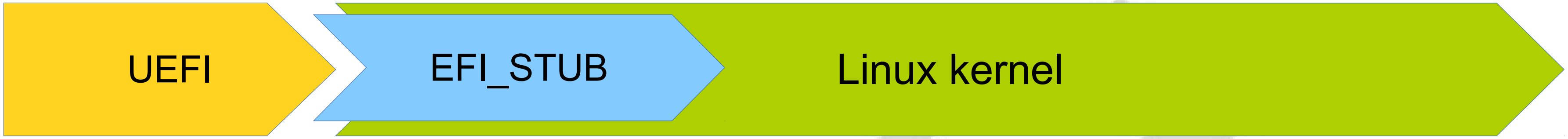

UEFI

LinuxLoader

Linux kernel

- LinuxLoader was first solution for ARM Linux
  - Built into tianocore
  - loads a initrd and device tree images
- Problems
  - Built into UEFI, cannot be updated
  - No guarantee LinuxLoader will be available
    - not part of spec
  - Linux unaware of UEFI – no runtime services

Enterprise: EFI STUB



An EFI OS loader embedded into the Linux kernel

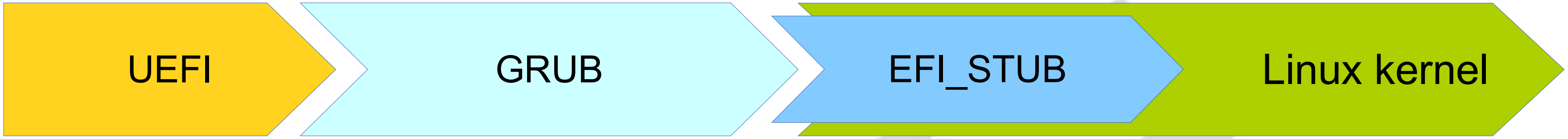
- Linux kernel becomes a native efi application
- Derived from x86 Linux EFI_STUB
- Can easily be changed in lockstep with kernel
- 100% Compatible with existing ARM firmware

Enterprise: EFI STUB

Embedding into the ARM Linux kernel

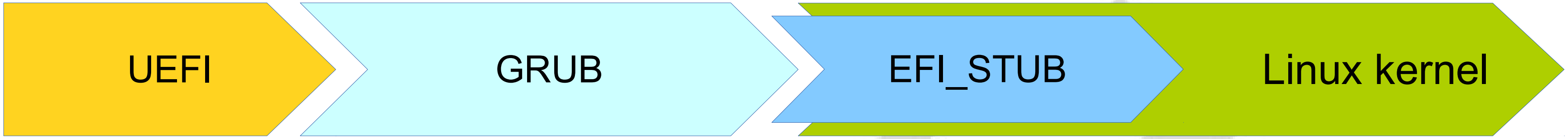
```
 start:
                .type   start,#function
-               .rept   7
+#ifdef CONFIG_EFI_STUB
+               .word   0x62805a4d              @ Magic MSDOS signature for PE/COFF + ADD opcode
+#else
+               mov     r0, r0
+#endif
+               .rept   5
                mov     r0, r0
                .endr
    ARM(                mov     r0, r0          )
    ARM(                b       1f              )
  THUMB(                adr     r12, BSYM(1f)   )
  THUMB(                bx      r12             )
+ THUMB(                .thumb                  )
+1:
+               b       zimage_continue

                .word   0x016f2818              @ Magic numbers to help the loader
                .word   start                   @ absolute load/run zImage address
                .word   _edata                  @ zImage end address
+
+       .org    0x3c
+       .long   pe_header                       @ Offset to PE-COFF header
```

Enterprise: GRUB



- Combination of boot menu and OS Loader
- Linux distributions already use GRUB
  - Users are familiar with it
- Boot non-Linux too – Xen, *BSD
- Works for both local and network boot
- Ported to ARM

Runtime Services

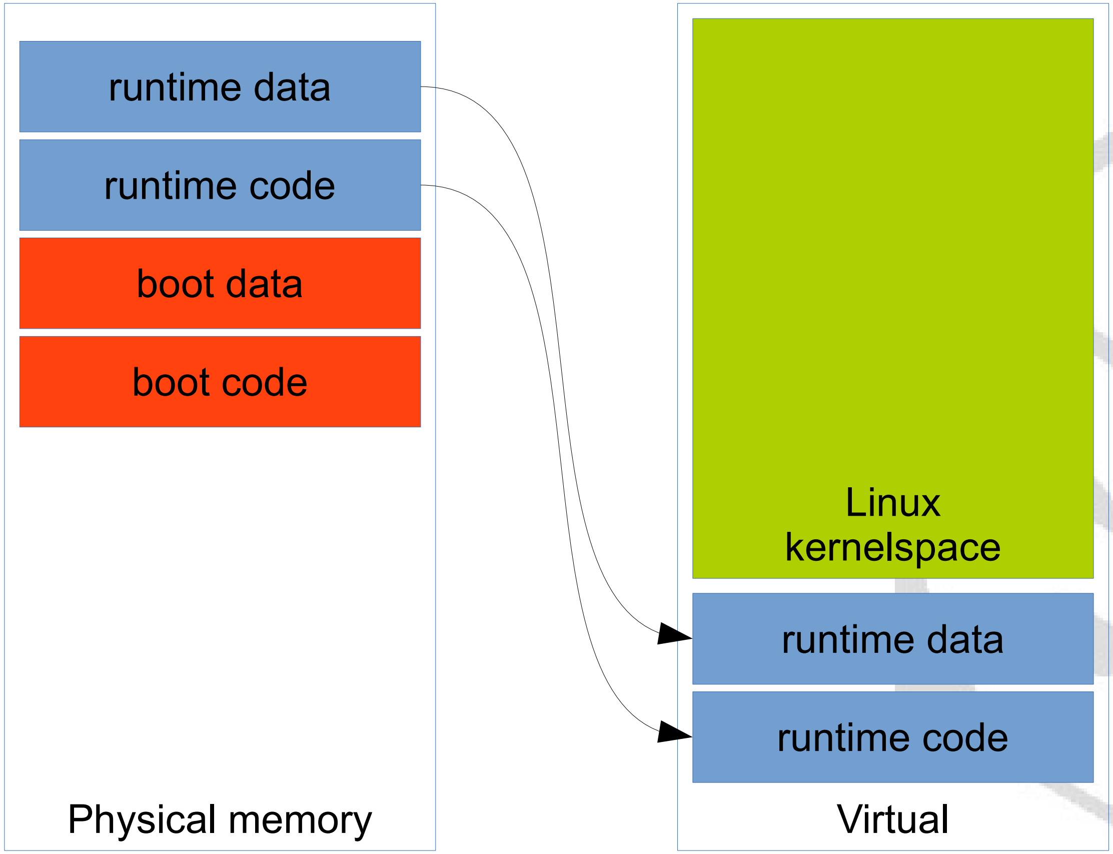


- Required by grub_install and efibootmgr
- Linux needs to become UEFI aware
  - Retrieve and use UEFI memory map
  - Parse system table

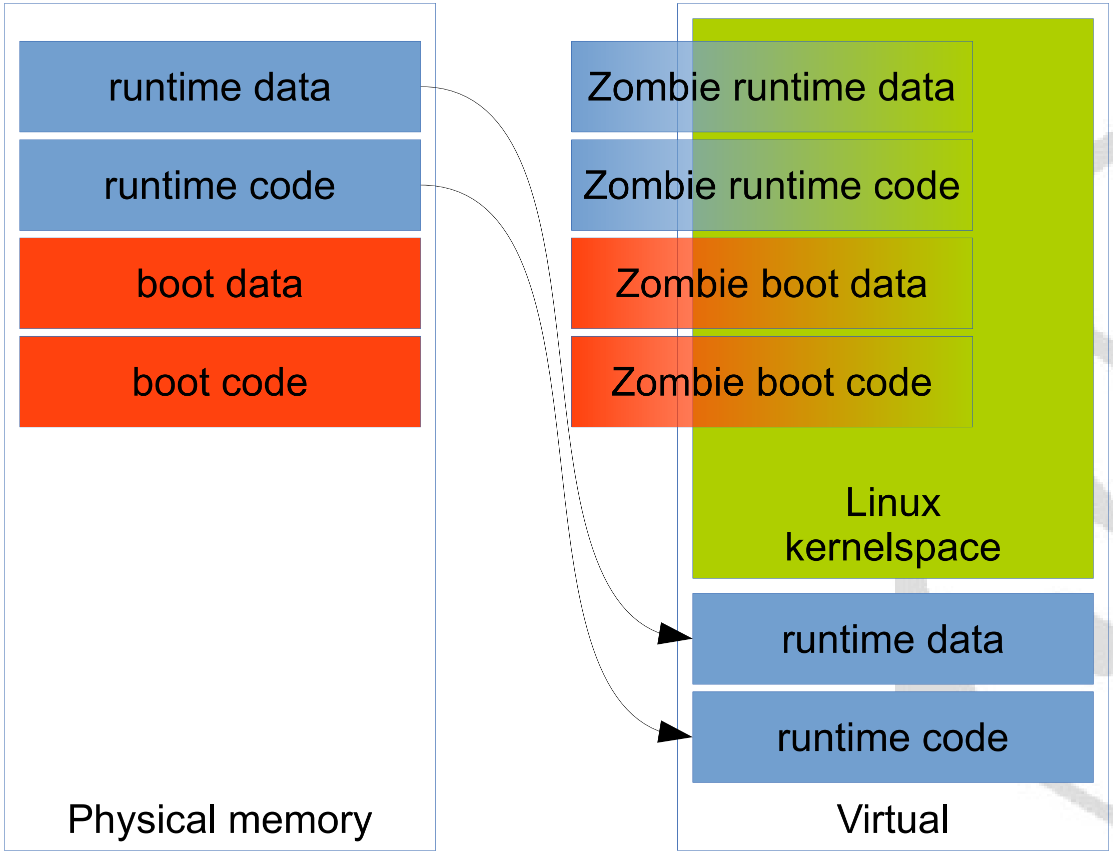
Virtual Address Pain

- SetVirtualAddressMap() requires **all** uefi code to correctly update internal pointer references
  - Easy to make mistakes
  - Stray pointer references can corrupt Linux data structures if not protected
  - Sleepy vendors won't patch firmware bugs
- Doesn't play well with kexec
  - Can only be called once
  - New kernel forced to use same map as old one
- What do we do?
  - Use SetVirtualAddressMap() anyway and live with bugs?
  - Use separate page tables when executing runtime services?
  - Don't support runtime services in Linux?

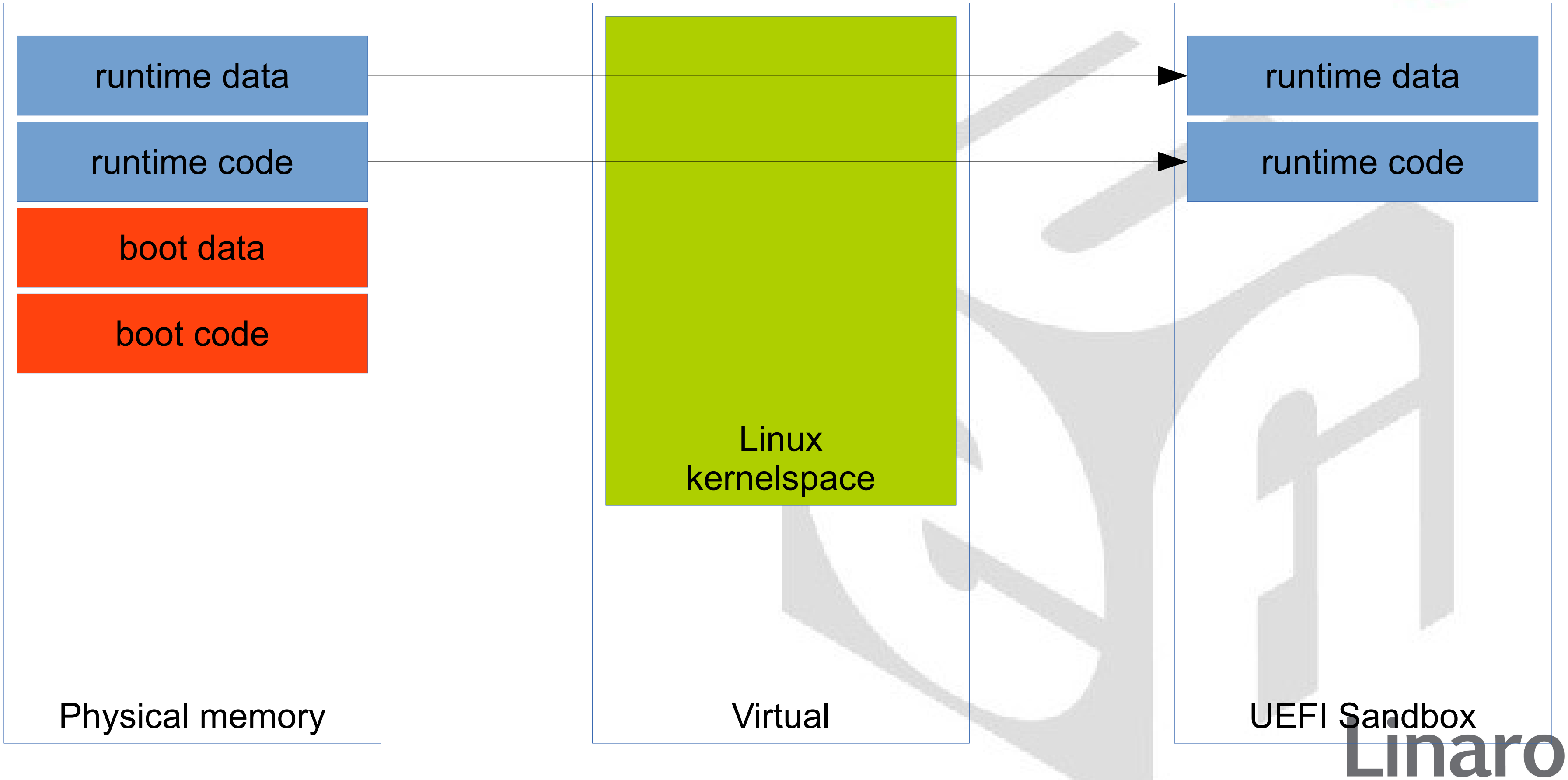
Virtual Address Pain



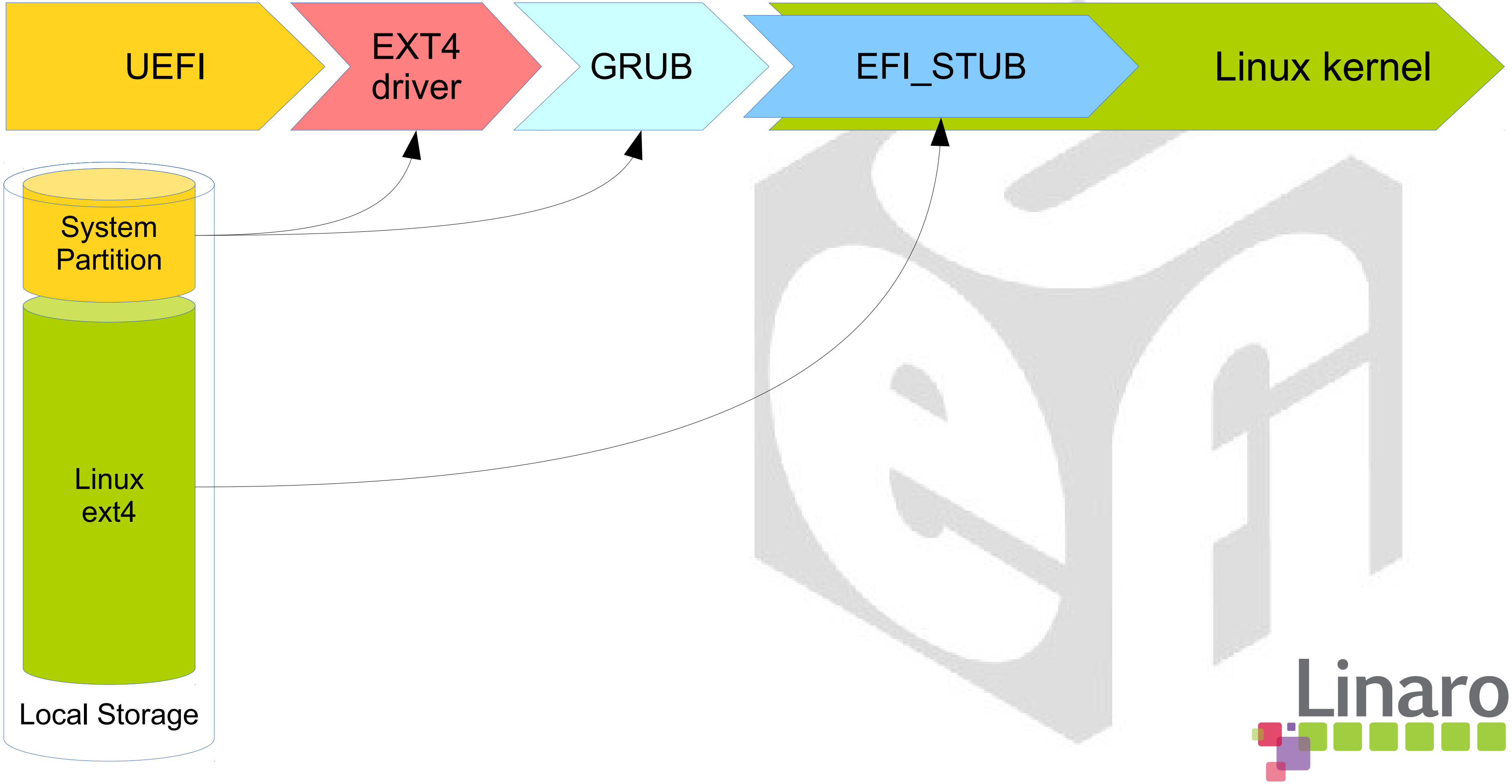
Virtual Address Pain



Virtual Address Pain



Enterprise: EXT2/3/4



Miscellaneous

- SMBIOS
  - Dmidecode
  - in-kernel smbios driver
  - Just needs to be enabled in UEFI config
- Hypervisor support
  - Covered by spec on AArch64
  - AArch32 in progress
  - Runtime Services calling still an open issue

UEFI+Linux on ARM

Embedded & Mobile

Embedded

- Vendor controls entire stack
  - Firmware
  - Kernel
  - Userspace & Applications
- Access to Open Source board support
  - Often small development teams
  - Ease of development more important than standardization
  - U-Boot is healthy incumbent

Mobile

- Android
  - Google responsible for most of userspace
  - Vendor has control over firmware and kernel
  - Fastboot
- Security
  - Secure Boot
  - Unlockable phones

Future Work

- Fastboot
- Platform support in mainline
- Device Tree support for embedded
- Lots and lots of testing!

UEFI+Linux on ARM

Thank You

UEFI+Linux on ARM

Questions?

Thanks for attending the UEFI Summerfest 2013

For more information on the Unified EFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by