Porting U-Boot Drivers to seL4

October 2022

INTRODUCTIONS

Mark Jenkinson and Stephen Williams

- Capgemini Engineering, Bath, UK
- Small team working with seL4 since late 2021

© Capgemini Engineering 2022. All rights reserved |

1

CONTEXT

WIDER PROJECT GOALS

- Make adopting seL4 easy, quick, and cheap

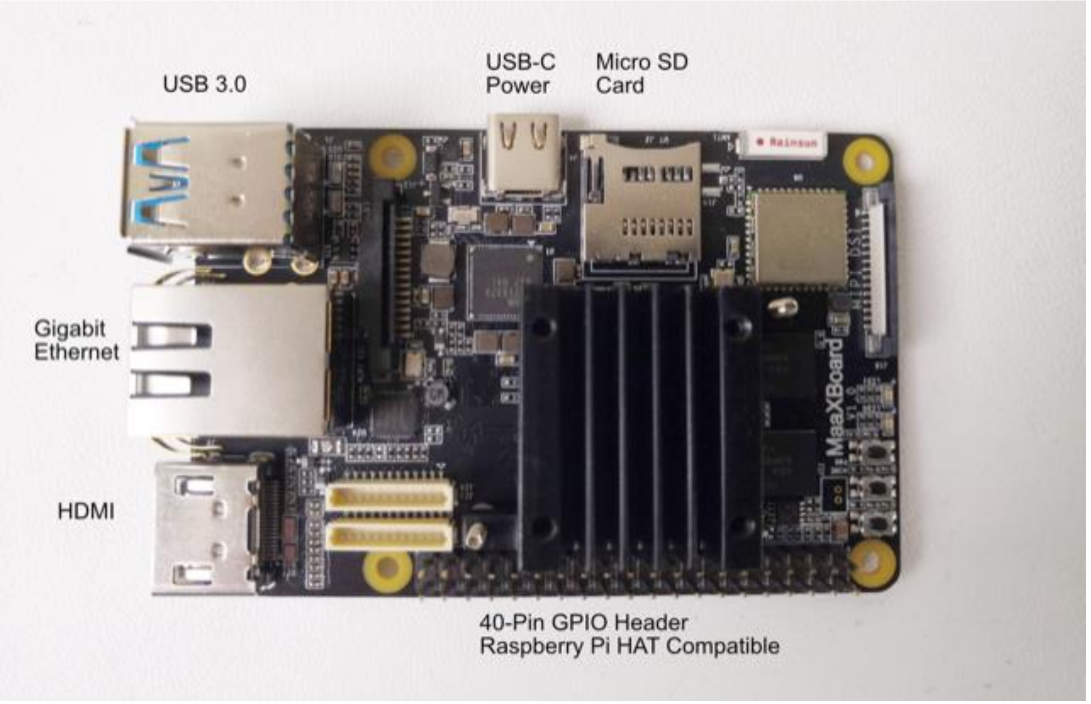
- Produced an open-source developer kit
  - Enables users to get started with seL4 very quickly
  - Provides a shopping list of required hardware
  - Provides pre-built binaries, tooling and a build environment
  - Provides clear and detailed setup instructions

- To be released imminently…

© Capgemini Engineering 2022. All rights reserved |

PLATFORM

- Selected a modern, low-cost board for development
  - Avnet MaaXBoard
  - i.MX 8M SoC (Quad ARM Cortex-A53)



© Capgemini Engineering 2022. All rights reserved |

DEVICE DRIVERS

- Focus today is on device drivers

- Availability of device drivers can be a significant barrier to use of seL4

- Developer kit provides extensive driver support for the chosen platform

- … but also provides a route to providing extensive driver support for a wide range of platforms

© Capgemini Engineering 2022. All rights reserved |

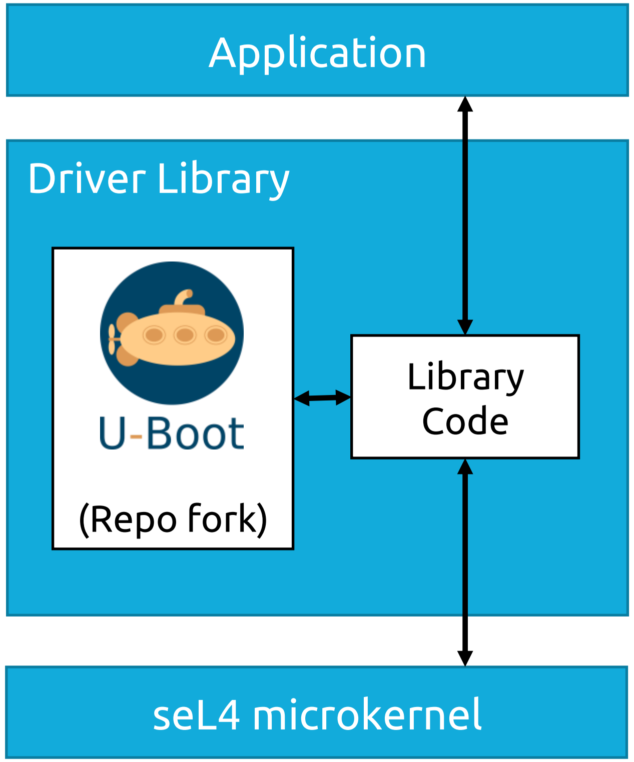
2

seL4 U-BOOT DRIVER LIBRARY

seL4 U-BOOT DRIVER LIBRARY

An extensible, native library allowing use of U-Boot device drivers

- Why build a driver library based upon U-Boot?
  - Individual drivers have previously been ported to seL4
    - Made to function independently of the U-Boot driver framework
    - Workable for simple devices, harder as device complexity increases (e.g. USB stack)
  - Wanted to make the U-Boot driver framework function within seL4
    - Provides a route for supporting any driver from U-Boot
- Primary design goal is ease of extension
  - Structured to allow new boards and devices to be easily supported
  - Allow device drivers to function with no, or minimal, code change



© Capgemini Engineering 2022. All rights reserved |

CURRENT LIBRARY CAPABILITIES

- Extensive support for Avnet MaaXBoard
  - USB, Ethernet, SPI, I^{2}C, MMC / SD, GPIO, Pin multiplexing (IOMUX), Clock and LED
- Limited support for Odroid-C2
  - GPIO, Pin multiplexing (IOMUX) and LED
  - Proof of concept for support of multiple boards
- xHCI USB stack
  - USB mass storage and USB keyboard drivers
- Disk support
  - Partition table formats: ISO, GPT, DOS and MAC
  - Filesystem formats: FAT(16/32) and EXT(2/3/4)
- Capable of utilising an external TCP stack
  - Demo application provided using PicoTCP stack

CURRENT LIBRARY LIMITATIONS

- Performance
  - Library is optimised for ease of extension, not performance
  - U-Boot drivers typically do not use interrupts, they must be polled
- No support for WiFi
  - Not supported by U-Boot
- Unlikely to be amenable to formal proofs

© Capgemini Engineering 2022. All rights reserved |

3

TECHNICAL CHALLENGES

REPLACING THE U-BOOT BUILD SYSTEM

- First obstacle is getting U-Boot code to build
- U-Boot uses a Kconfig / make build system whilst seL4 uses CMake / Ninja
- Created CMake script to replicate the U-Boot build system functionality
  - Defines which drivers to include based upon the platform
  - Control which source files to compile based upon required drivers
  - Set up the U-Boot pre-processor macros
  - Convert seL4 build options to equivalent U-Boot macros, e.g. logging level, architecture details

© Capgemini Engineering 2022. All rights reserved |

LIBRARY INITIALISATION

- Map device address ranges into the virtual address space

- Create a device tree for U-Boot to query
  - Derived from the seL4 device tree
  - Contains the minimal set of devices the library needs to access
  - Replace bus addresses with the mapped virtual memory address

- Perform bespoke initialisation of U-Boot
  - Initialise required sub-systems
  - Setup "global data" with required status and data; second-stage bootloader after relocation to RAM
  - Initialise "linker lists"; arrays of optional components normally held in linker sections

© Capgemini Engineering 2022. All rights reserved |

EXECUTING U-BOOT WITHIN VIRTUAL ADDRESS SPACE

- U-Boot is designed to execute within the physical address space
  - Drivers provide memory addresses to devices for DMA
  - Without modification, drivers will provide virtual memory addresses to devices
  - Devices cannot access a virtual memory address. Problem!

- Driver modification is required in limited cases
  - Only devices that utilise DMA are affected
  - Library provides seL4 specific DMA routines for allocation, address translation and cache flush / invalidate
  - Replace local memory allocation / deallocation with DMA equivalents
  - Drivers using the Linux “DMA Mapping” API work without modification; library provides a mapping onto seL4 routines

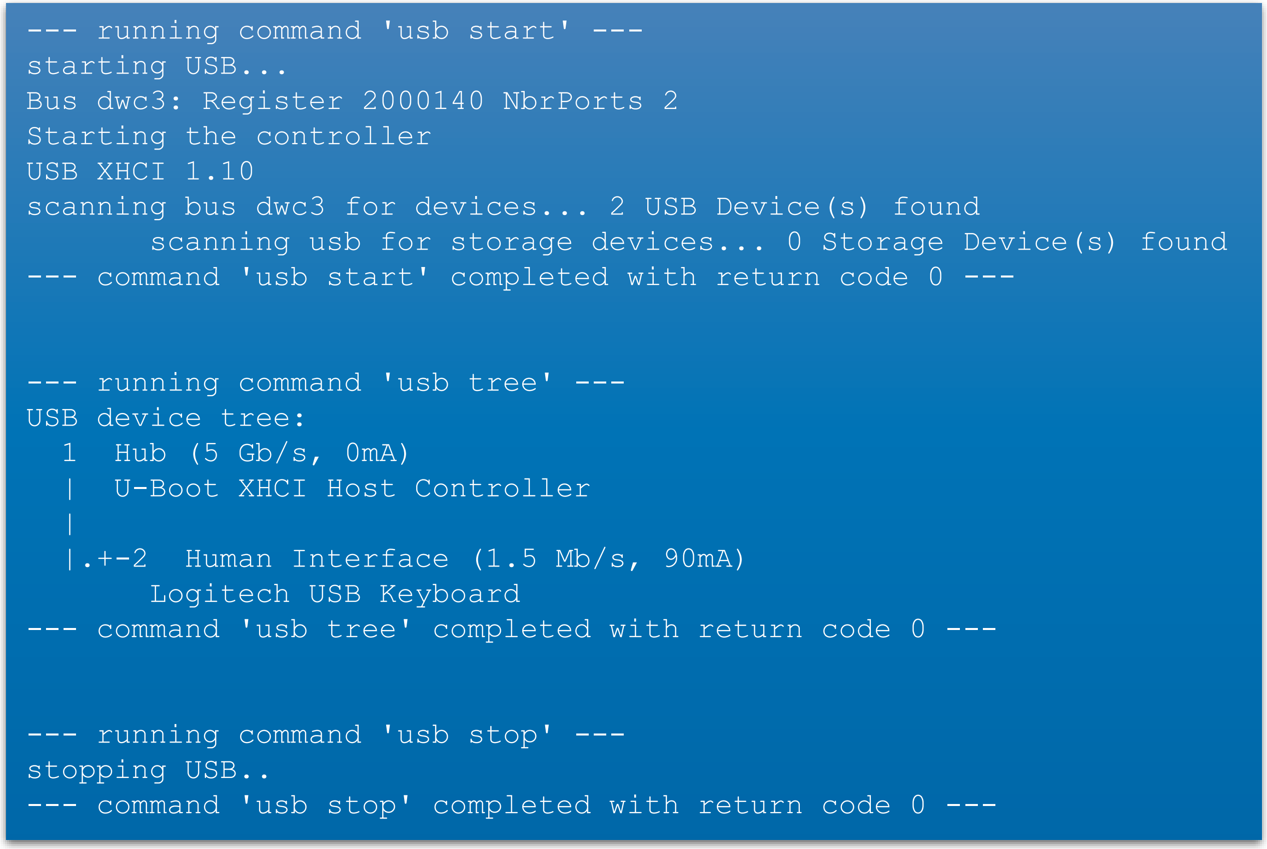
© Capgemini Engineering 2022. All rights reserved |

4

LIBRARY USAGE EXAMPLES

PUBLIC INTERFACE

- Primary interface is through textual commands
  - Same commands as input to the U-Boot CLI
  - Exposes full capability of U-Boot
- Easily extendable to provide programmatic interface for any command
- Interface to send / receive raw Ethernet frames
  - Allows library to be linked to an external TCP stack
- Interface to receive input from character devices
  - Receive input from USB keyboard

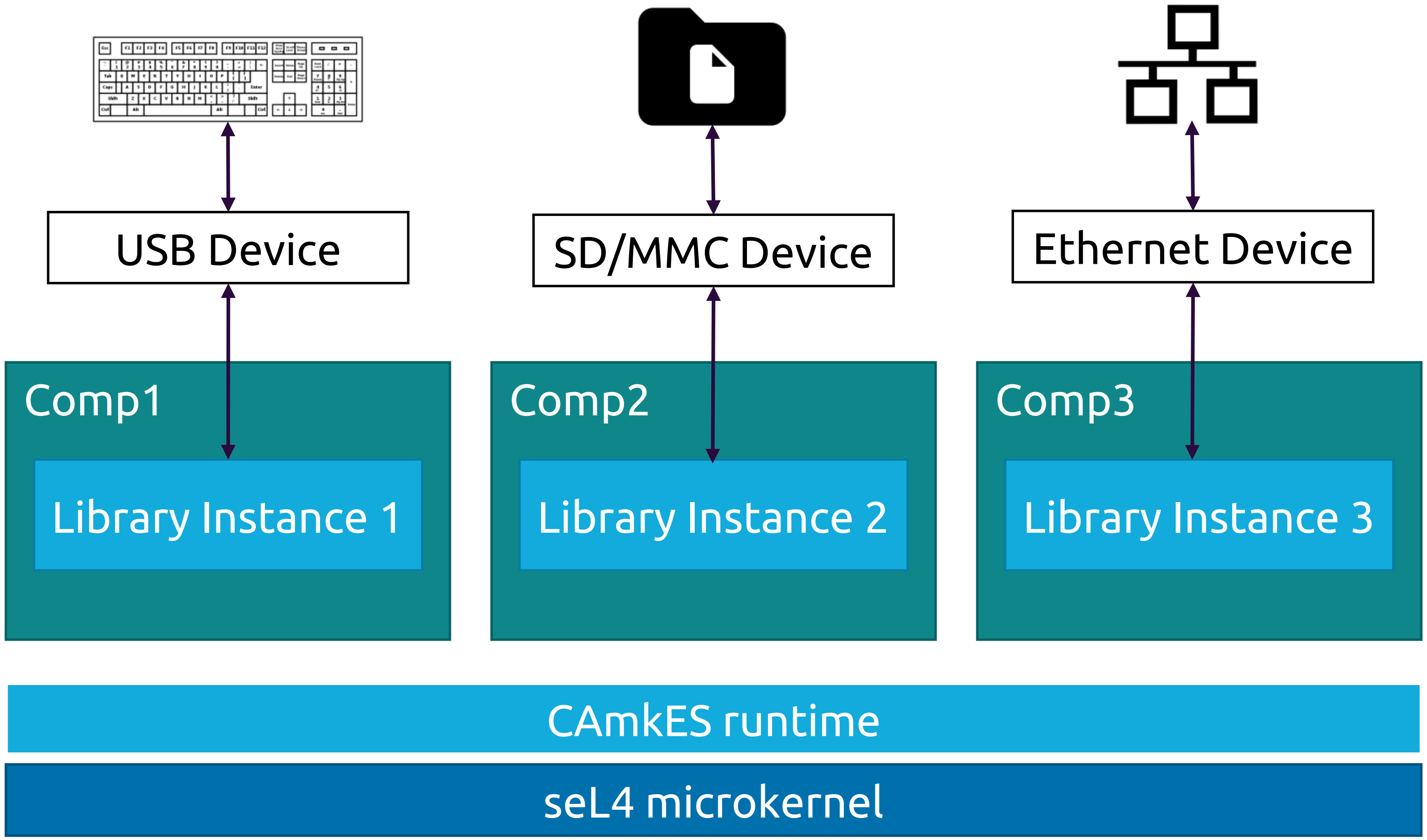


```
--- running command 'usb start' ---
starting USB...
Bus dwc3: Register 2000140 NbrPorts 2
Starting the controller
USB XHCI 1.10
scanning bus dwc3 for devices... 2 USB Device(s) found
       scanning usb for storage devices... 0 Storage Device(s) found
--- command 'usb start' completed with return code 0 ---


--- running command 'usb tree' ---
USB device tree:
  1  Hub (5 Gb/s, 0mA)
  |  U-Boot XHCI Host Controller
  |
  |.+-2  Human Interface (1.5 Mb/s, 90mA)
       Logitech USB Keyboard
--- command 'usb tree' completed with return code 0 ---


--- running command 'usb stop' ---
stopping USB..
--- command 'usb stop' completed with return code 0 ---
```

USAGE EXAMPLE



© Capgemini Engineering 2022. All rights reserved |

5

EXTENDING THE LIBRARY

HOW TO EXTEND THE LIBRARY

- Adding a new platform or driver, what is required?
- Update the CMake build script
  - List which drivers are associated with the platform
  - Define the source files and macros for the driver
- Provide “linker lists” initialisation routine
- Provide a monotonic real-time clock
  - Required for drivers with timing requirements
- Perform DMA driver updates

```
if("${KernelPlatform}" STREQUAL "odroidc2")
   set(iomux_driver "meson-gxbb-pinctrl")
   set(gpio_driver "meson_gx_gpio_driver")
   set(led_driver "gpio_led")

...

if(iomux_driver MATCHES  "meson-gxbb-pinctrl")
   list(APPEND uboot_deps uboot/drivers/pinctrl/meson/pinctrl-meson-gxbb.c)
   list(APPEND uboot_deps uboot/drivers/pinctrl/meson/pinctrl-meson.c)
   list(APPEND uboot_deps uboot/drivers/pinctrl/meson/pinctrl-meson-gx-pmx.c)
```

```
void initialise_driver_data (void) {
   driver_data.uclass_driver_array[0] = _u_boot_uclass_driver__nop;
   driver_data.uclass_driver_array[1] = _u_boot_uclass_driver__root;
   driver_data.uclass_driver_array[2] = _u_boot_uclass_driver__simple_bus;
   driver_data.uclass_driver_array[3] = _u_boot_uclass_driver__phy;
   driver_data.uclass_driver_array[4] = _u_boot_uclass_driver__blk;
   driver_data.uclass_driver_array[5] = _u_boot_uclass_driver__pinconfig;
   driver_data.uclass_driver_array[6] = _u_boot_uclass_driver__pinctrl;
   driver_data.uclass_driver_array[7] = _u_boot_uclass_driver__gpio;
   driver_data.uclass_driver_array[8] = _u_boot_uclass_driver__led;

   driver_data.driver_array[0] = _u_boot_driver__root_driver;
   driver_data.driver_array[1] = _u_boot_driver__simple_bus;
   driver_data.driver_array[2] = _u_boot_driver__pinconfig_generic;
   driver_data.driver_array[3] = _u_boot_driver__meson_gxbb_pinctrl;
   driver_data.driver_array[4] = _u_boot_driver__meson_gx_gpio_driver;
   driver_data.driver_array[5] = _u_boot_driver__led_gpio_wrap;
   driver_data.driver_array[6] = _u_boot_driver__led_gpio;

   driver_data.cmd_array[0] = _u_boot_cmd__dm;
   driver_data.cmd_array[1] = _u_boot_cmd__env;
   driver_data.cmd_array[2] = _u_boot_cmd__setenv;
   driver_data.cmd_array[3] = _u_boot_cmd__pinmux;
   driver_data.cmd_array[4] = _u_boot_cmd__gpio;
   driver_data.cmd_array[5] = _u_boot_cmd__led;
}
```

© Capgemini Engineering 2022. All rights reserved |

6

ANY QUESTIONS?

About Capgemini Engineering

Capgemini Engineering combines, under one brand, a unique set of strengths from across the Capgemini Group: the world leading engineering and R&D services of Altran – acquired by Capgemini in 2020 - and Capgemini's digital manufacturing expertise. With broad industry knowledge and cutting-edge technologies in digital and software, Capgemini Engineering supports the convergence of the physical and digital worlds. Combined with the capabilities of the rest of the Group, it helps clients to accelerate their journey towards Intelligent Industry. Capgemini Engineering has more than 52,000 engineer and scientist team members in over 30 countries across sectors including aeronautics, automotive, railways, communications, energy, life sciences, semiconductors, software & internet, space & defence, and consumer products.

Capgemini Engineering is an integral part of the Capgemini Group, a global leader in partnering with companies to transform and manage their business by harnessing the power of technology. The Group is guided every day by its purpose of unleashing human energy through technology for an inclusive and sustainable future. It is a responsible and diverse organization of 270,000 team members in nearly 50 countries. With its strong 50-year heritage and deep industry expertise, Capgemini is trusted by its clients to address the entire breadth of their business needs, from strategy and design to operations, fueled by the fast evolving and innovative world of cloud, data, AI, connectivity, software, digital engineering and platforms. The Group reported in 2020 global revenues of €16 billion.

Get the Future You Want | www.capgemini.com/capgemini-engineering

This presentation contains information that may be privileged or confidential and is the property of the Capgemini Group.

Copyright © 2022 Capgemini. All rights reserved.