U-Boot from Scratch

Jagan Teki

Jagan Teki

- ➔ **Jagan**nadhaSutradharudu Teki
- ➔ Free software engineer
- ➔ Enthusiastic Linux kernel hacker
- ➔ U-Boot maintainer for SPI, SPI-FLASH, Allwinner sunXi SoC
- ➔ Buildroot, Yocto contributor
- ➔ Heading to ***Amarula Solutions India***

Agenda

- ➔ U-Boot In detail
- ➔ Image boot
- ➔ Features
- ➔ Port new hardware
- ➔ Future work

U-Boot In detail

- ➔ History
- ➔ Community
- ➔ Hardware support
- ➔ U-Boot
- ➔ Source
- ➔ Source tree
- ➔ Build process
- ➔ Boot Sequence
- ➔ Loading sources
- ➔ Debug
- ➔ Tools
- ➔ Testing

History

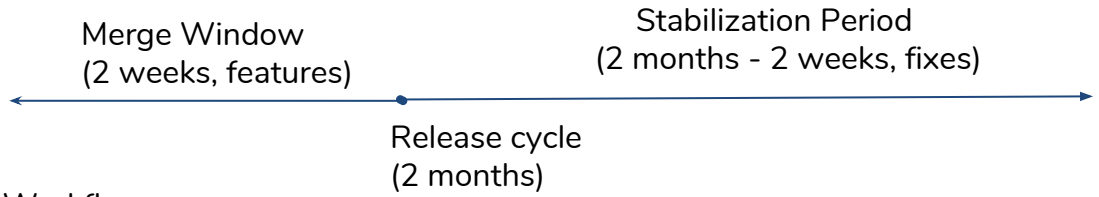
- ➔ 8xxROM written by Magnus Damm
- ➔ October 1999 Wolfgang Denk moved to sourceForge.net and as PPCBoot
- ➔ PPCBoot-0.4.1 released in July 19, 2000
- ➔ PPCBoot-2.0.0 became U-Boot-0.1.0 with x86
- ➔ Since then added many architectures, boards, features etc.
- ➔ Current name is termed as Das U-Boot
- ➔ Flexible and opensource bootloader
- ➔ Wolfgang Denk as head custodian for over 10 years
- ➔ Tom Rini as head custodian since September 2012
- ➔ Recent release v2018.09

Community

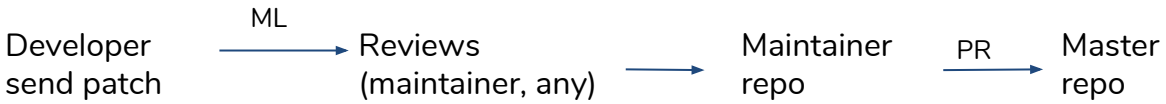
- ➔ On average over 25+ employers and 120+ developers contributing every release
- ➔ Number of talks on various conferences

Development process:

- ➔ 35+ Custodian/Maintainer for various subsystems
- ➔ Release cycle



- ➔ Workflow



Hardware support

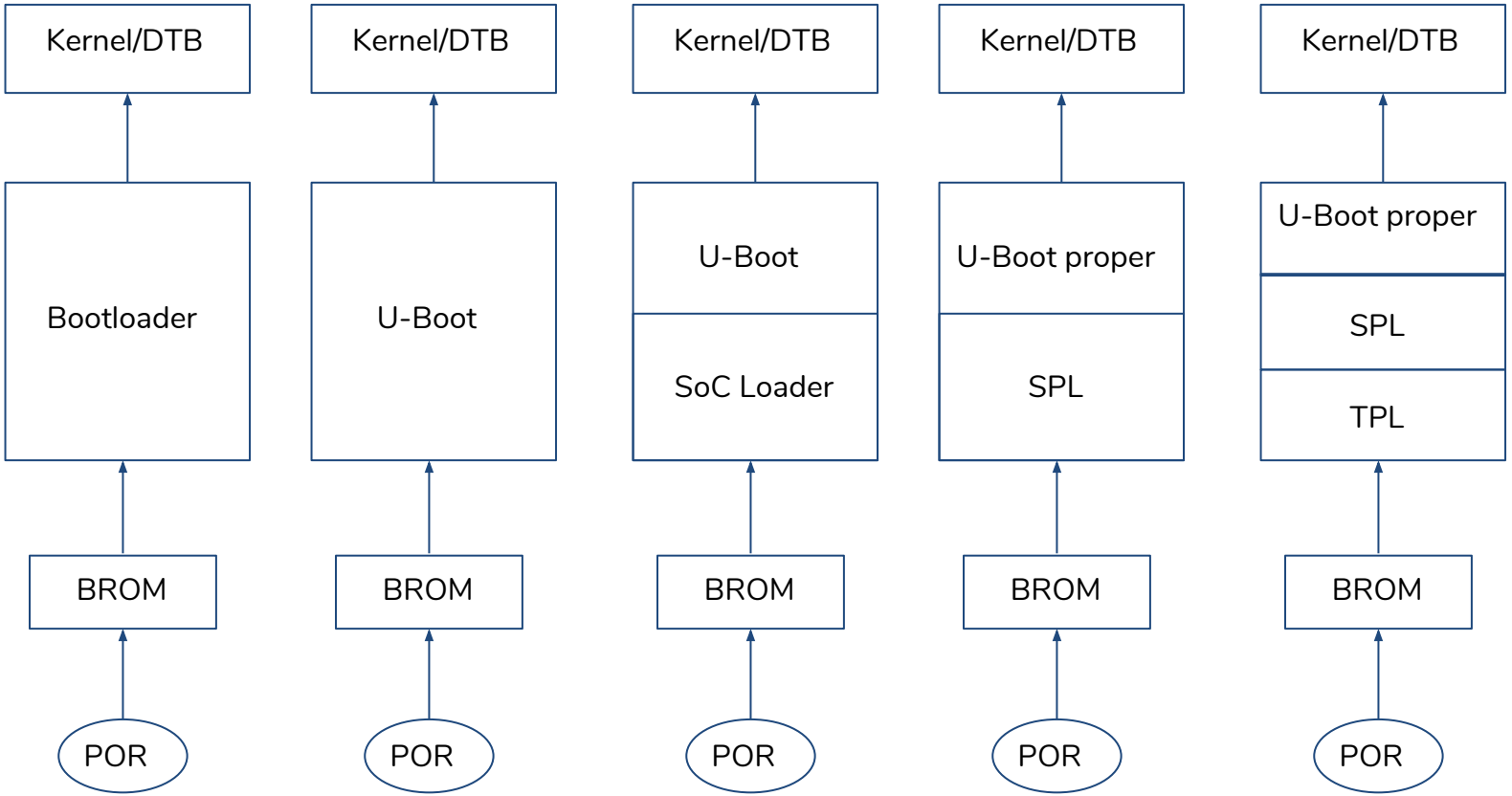
Architecture/SoC:

- ARM
  - 32-bit: Aspeed, Altera, Allwinner, Atmel, Broadcom, Qemu, Qualcomm, Marvell, NXP, Rockchip, STM32, Tegra, TI, UniPhier, Xilinx
  - 64-bit: Allwinner, Marvell, NXP, Rockchip, Tegra, UniPhier, Xilinx
- X86 (Baytrail, Broadwell, Quark, etc)
- ARC, M68K, MicroBlaze, MIPS, NDS32, NIOS2, PowerPC, RISCV, Sandbox, SuperH, Xtensa

Boards:

- 186+ different board vendors
- 1083+ different boards

U-Boot


Kernel/DTB
Bootloader
BROM
POR
Kernel/DTB
U-Boot
BROM
POR
Kernel/DTB
U-Boot
SoC Loader
BROM
POR
Kernel/DTB
U-Boot proper
SPL
BROM
POR
Kernel/DTB
U-Boot proper
SPL
TPL
BROM
POR

U-Boot build

- ➔ Git master or dev tree at
  http://git.denx.de/?p=u-boot.git;a=summary

- ➔ Custodian's or Maintainers tree at
  http://git.denx.de/?p=u-boot.git;a=forks

Example of building vyasa RK3288 board, which is ARM platform with arm-linux-gnueabi

1. $ git clone git://git.denx.de/u-boot.git
2. $ u-boot
3. $ export ARCH=arm
4. $ export CROSS_COMPILE=arm-linux-gnueabi-
5. $ make vyasa-rk3288_defconfig
6. $ make

U-Boot tree

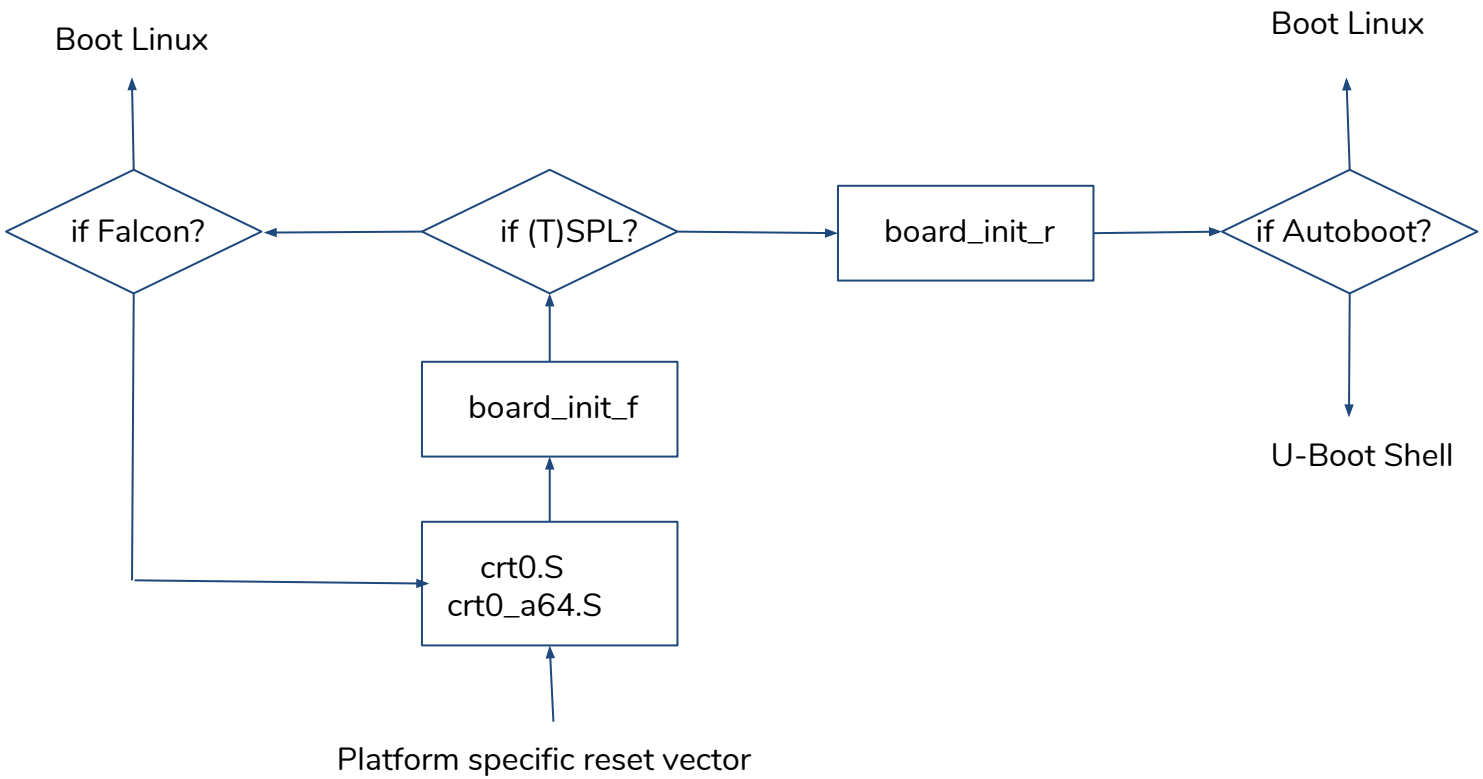
- Arch - Architecture specific code
  - arch/foo - foo CPU specific code
  - arch/foo/mach-joo - joo MACHINE specific code
  - arch/foo/dts - foo devicetree code
- board - board specific code
- cmd - commands
- common - common code
  - common/spl - SPL code
- configs - per board configuration
- disk - disk or partition
- doc - documentation
- drivers - drivers code
- env - environment
- *fs* - filesystems
- *net* - Networking
- Kbuild - Kbuild scripts
- Kconfig - Global kconfig file
- Makefile - Global makefile
- scripts, tools - build script, etc

Build process

```
scripts/kconfig/conf  --syncconfig Kconfig
  CHK     include/config.h
  UPD     include/config.h
  CFG     u-boot.cfg
  GEN     include/autoconf.mk
  GEN     include/autoconf.mk.dep
  CFG     spl/u-boot.cfg
  GEN     spl/include/autoconf.mk
  CFG     tpl/u-boot.cfg
  GEN     tpl/include/autoconf.mk
  CHK     include/config/uboot.release
  UPD     include/config/uboot.release
  CHK     include/generated/version_autogenerated.h
  UPD     include/generated/version_autogenerated.h
  CHK     include/generated/timestamp_autogenerated.h

  LD      arch/arm/cpu/built-in.o
  CC      arch/arm/cpu/armv7/cache_v7.o
  AS      arch/arm/cpu/armv7/cache_v7_asm.o
  CC      arch/arm/cpu/armv7/cpu.o
  CC      arch/arm/cpu/armv7/cp15.o
  CC      arch/arm/cpu/armv7/syslib.o
  AS      arch/arm/cpu/armv7/sctlr.o
  AS      arch/arm/cpu/armv7/lowlevel_init.o
  LD      arch/arm/cpu/armv7/built-in.o
  AS      arch/arm/cpu/armv7/start.o
  CC      arch/arm/lib/eabi_compat.o
  AS      arch/arm/lib/crt0_arm_efi.o
  CC      arch/arm/lib/reloc_arm_efi.o
  CC      arch/arm/mach-rockchip/boot_mode.o
  CC      arch/arm/mach-rockchip/rk3288-board.o
  CC      arch/arm/mach-rockchip/sdram_common.o
  CC      arch/arm/mach-rockchip/rk_timer.o
  CC      arch/arm/mach-rockchip/rk3288/clk_rk3288.o
  CC      arch/arm/mach-rockchip/rk3288/rk3288.o
  CC      arch/arm/mach-rockchip/rk3288/syscon_rk3288.o
  CC      board/amarula/vyasa-rk3288/vyasa-rk3288.o
```

```
  CC      spl/arch/arm/mach-rockchip/sdram_common.o
  CC      spl/arch/arm/mach-rockchip/rk_timer.o
  CC      spl/arch/arm/mach-rockchip/rk3288/clk_rk3288.o
  CC      spl/arch/arm/mach-rockchip/rk3288/rk3288.o
  CC      spl/arch/arm/mach-rockchip/rk3288/syscon_rk3288.o
  CC      spl/arch/arm/mach-rockchip/bootrom.o
  CC      spl/arch/arm/mach-rockchip/rk3288-board-spl.o
  CC      spl/arch/arm/cpu/armv7/cache_v7.o
  CC      spl/arch/arm/cpu/armv7/cpu.o
  AS      spl/arch/arm/cpu/armv7/lowlevel_init.o
  AS      spl/arch/arm/cpu/armv7/start.o
  LD      spl/u-boot-spl
OBJCOPY spl/u-boot-spl-nodtb.bin
  COPY    spl/u-boot-spl.dtb
  CAT     spl/u-boot-spl-dtb.bin
  COPY    spl/u-boot-spl.bin

  CC      tpl/arch/arm/mach-rockchip/sdram_common.o
  CC      tpl/arch/arm/mach-rockchip/rk_timer.o
  CC      tpl/arch/arm/mach-rockchip/rk3288/clk_rk3288.o
  CC      tpl/arch/arm/mach-rockchip/rk3288/rk3288.o
  CC      tpl/arch/arm/mach-rockchip/rk3288/syscon_rk3288.o
  CC      tpl/arch/arm/mach-rockchip/bootrom.o
  CC      tpl/arch/arm/mach-rockchip/rk3288-board-tpl.o
  CC      tpl/arch/arm/cpu/armv7/cache_v7.o
  AS      tpl/arch/arm/cpu/armv7/cache_v7_asm.o
  CC      tpl/arch/arm/cpu/armv7/cpu.o
  CC      tpl/arch/arm/cpu/armv7/cp15.o
  CC      tpl/arch/arm/cpu/armv7/syslib.o
  AS      tpl/arch/arm/cpu/armv7/lowlevel_init.o
  AS      tpl/arch/arm/cpu/armv7/start.o
  LDS     tpl/u-boot-spl.lds
  LD      tpl/u-boot-tpl
OBJCOPY tpl/u-boot-tpl-nodtb.bin
  COPY    tpl/u-boot-tpl.bin
  COPY    u-boot.dtb
MKIMAGE u-boot-dtb.img
```

Boot Sequence



Loading sources

- ➔ U-Boot can support variety of loading sources
- ➔ RAM
- ➔ Network: tftp, DHCP
- ➔ Serial: Kermit
- ➔ SPI Flash
- ➔ MMC
- ➔ Parallel NOR
- ➔ NAND, UBI
- ➔ USB
- ➔ SATA
- ➔ AHCI
- ➔ NVME

Debug

- ➔ printf
- ➔ CONFIG_DEBUG
- ➔ GDB
- ➔ Early UART (CONFIG_DEBUG_UART)

Tools

- Patman
  - Manual patch creation, cover-letter, adding maintainers etc can be error-prone.
  - Patman make above automated
  - Create patch, insert cover-letter, add maintainer (via ~/.git-mailrc), run checkpatch.pl etc
  - How to use? tools/patman/README
- Buildman
  - U-Boot builder for multiple commits, branches etc
  - Replaced by legacy MAKEALL
  - Understandable output summary
  - Checking image sizes
  - tools/buildman/README
- Binman
  - Packaging multiple image components

```
/ {
    binman {
      filename = "u-boot-sunxi-with-spl.bin";
      pad-byte = <0xff>;
      blob {
            filename = "spl/sunxi-spl.bin";
      };
      u-boot-img {
            offset = <CONFIG_SPL_PAD_TO>;
      };
    };
};
```

Testing

- travis-ci.org
  - Automated build environment, with limited run-time, free to use
  - May take longer duration, if more jobs are initiated
  - .travis.yml, u-boot travis build plugin
- test.py
  - Pytest framework
  - Works for sandbox, qemu, some real hardware
  - Sanity tests for dm code
  - doc/README.trace
- trace
  - Kind of Linux ftrace
  - Collect execution and sent to host for analysis
- tbot
  - Execute test cases on boards
  - Heiko Schocher demo, https://www.youtube.com/watch?v=zfjpj3DLsx4

Image Boot

- ➔ Legacy image
- ➔ FIT
- ➔ Verified image
- ➔ Distro boot
- ➔ Secure boot
- ➔ Falcon mode

Legacy Image

- ➔ Fixed offset images - standalone, zImage binaries
- ➔ go addr [arg ...]
- ➔ u-boot Image format
- ➔ Single component uImage
- ➔ Monolithic, combination of images
- ➔ bootm [addr [arg ...]]

? Not flexible, indexing

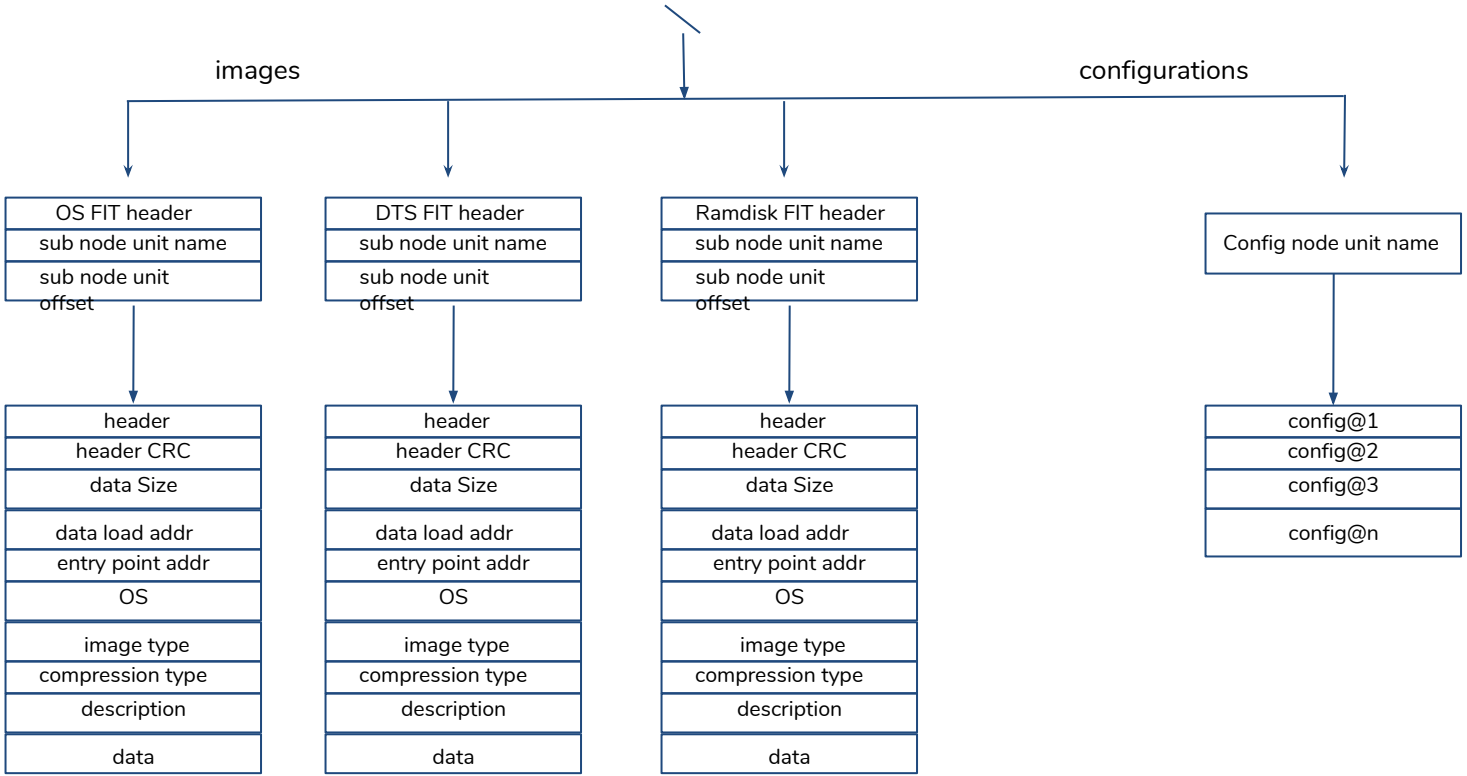
? No hash integrity

? No scope of security addition

```
$ mkimage \
  > -A arm \
  > -O linux \
  > -T kernel \
  > -a 0x10008000 \
  > -e 0x10008000 \
  > -n "Linux-4.19.0-rc2" \
  > -d zImage uImage
Image Name:   Linux-4.19.0-rc2-next-20180905-0
Created:      Tue Sep 11 23:07:55 2018
Image Type:   ARM Linux Kernel Image (uncompressed)
Data Size:    8372992 Bytes = 8176.75 kB = 7.99 MB
Load Address: 10008000
Entry Point:  10008000


$ mkimage
> -A arm \
> -O linux \
> -T multi \
> -a 0x10008000 \
> -e 0x10008000 \
> -n 'Multi image' \
> -d vmlinux.bin.gz:ramdisk.image.gz:imx6q-icore-rqs.dtb uMulti
Image Name:   Multi image
Created:      Tue Sep 11 23:07:55 2018
Image Type:   ARM Linux Multi-File Image (gzip compressed)
Data Size:    82092755 Bytes = 80168.71 KiB = 78.29 MiB
Load Address: 10008000
Entry Point:  10008000
Contents:
   Image 0: 7609333 Bytes = 7430.99 KiB = 7.26 MiB
   Image 1: 74445331 Bytes = 72700.52 KiB = 71.00 MiB
   Image 2: 38071 Bytes = 37.18 KiB = 0.04 MiB

```

FIT (Flattened Image Tree)

➔ Like FDT, tree topology for various images uboot, spl, atf, dtb, kernel, ramdisk etc



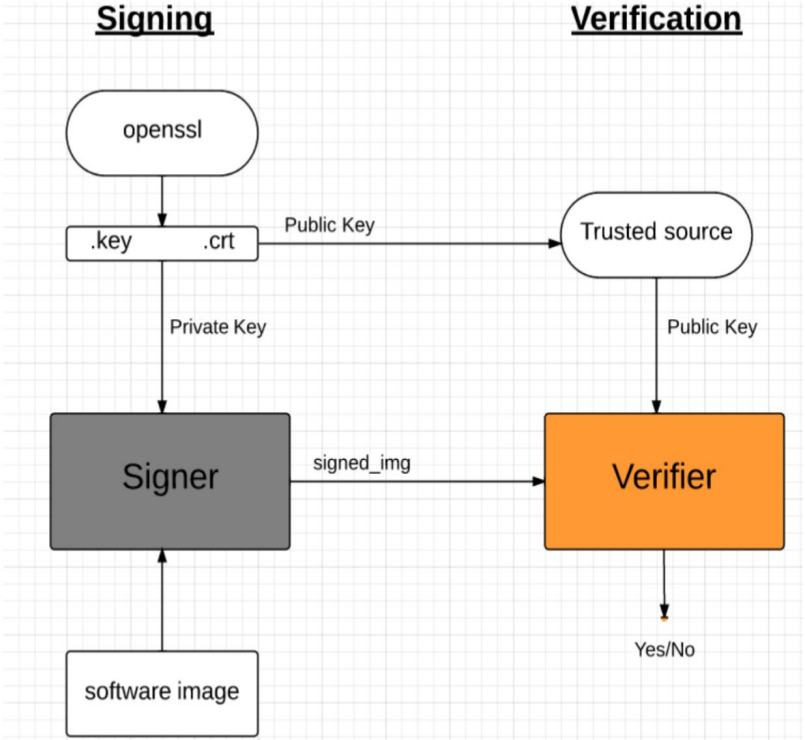
FIT, contd

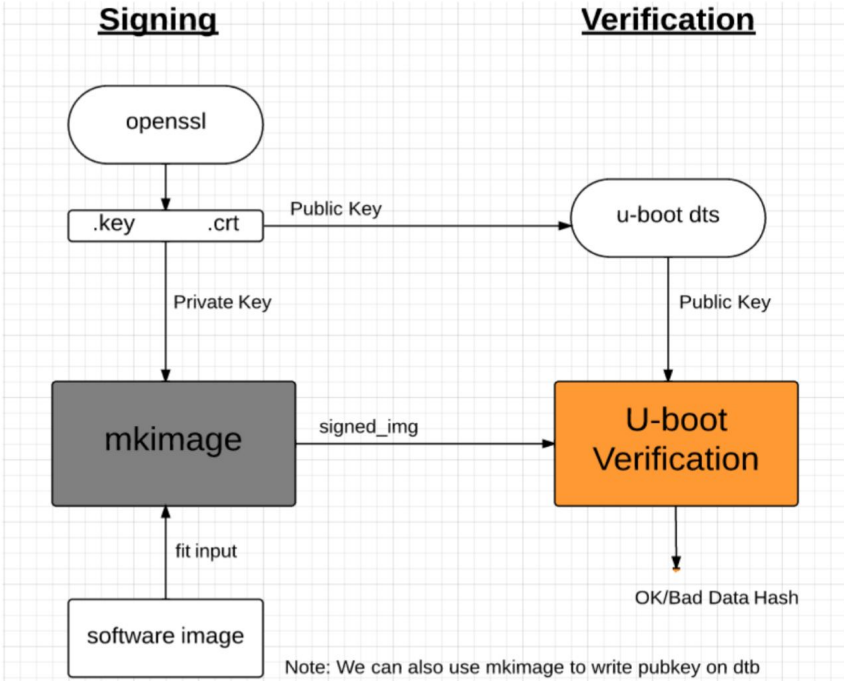
```
/dts-v1/;
/ {
    description = "FIT with single Linux kernel and FDT blob";
    #address-cells = <1>;
    images {
        kernel@1 {
            description = "i.MX6 Linux kernel";
            data = /incbin/("./vmlinux.bin.gz");
            type = "kernel";
            arch = "arm";
            os = "linux";
            compression = "gzip";
            load = <0x10008000>;
            entry = <0x10008000>;
            hash@1 {
                algo = "md5";
            };
            hash@2 {
                algo = "sha1";
            };
        };

        fdt@1 {
            description = "Engicam i.CoreM6 Quad/Dual RQS Starter Kit Devicetree blob";
            data = /incbin/("./imx6q-icore-rqs.dtb");
            type = "flat_dt";
            arch = "arm";
            compression = "none";
            hash@1 {
                algo = "md5";
            };
            hash@2 {
                algo = "sha1";
            };
        };
    };

    configurations {
        default = "conf@1";
        conf@1 {
            description = "Boot Linux kernel with FDT blob";
            kernel = "kernel@1";
            fdt = "fdt@1";
        };
    };
};
```

```
icorem6qdl-rqs> fatload mmc 0:1 $loadaddr fit.itb
reading fit.itb
6167494 bytes read in 335 ms (17.6 MiB/s)
icorem6qdl-rqs> bootm $loadaddr
Booting FIT image from mmc ...
## Loading kernel from FIT Image at 12000000 ...
   Using 'conf@1' configuration
   Verifying Hash Integrity ... OK
   Trying 'kernel@1' kernel subimage
     Description:  i.MX6 Linux kernel
     Type:         Kernel Image
     Compression:  gzip compressed
     Data Start:   0x120000f0
     Data Size:    6130148 Bytes = 5.8 MiB
     Architecture: ARM
     OS:           Linux
     Load Address: 0x10008000
     Entry Point:  0x10008000
     Hash algo:    md5
     Hash value:   b975a202ea2963c53c53f527329930cd
     Hash algo:    sha1
     Hash value:   78b93fe404b795de8c837af27d67f4df9b96083a
   Verifying Hash Integrity ... md5+ sha1+ OK
## Loading fdt from FIT Image at 12000000 ...
   Using 'conf@1' configuration
   Trying 'fdt@1' fdt subimage
     Description:  Engicam i.CoreM6 Quad/Dual RQS Starter Kit Devicetree blob
     Type:         Flat Device Tree
     Compression:  uncompressed
     Data Start:   0x125d8dbc
     Data Size:    35298 Bytes = 34.5 KiB
     Architecture: ARM
     Hash algo:    md5
     Hash value:   4371a4dfe55127c2fda8a9feb4d3b313
     Hash algo:    sha1
     Hash value:   e34a9326b5e7fd43557753ef980fe67326f82ea1
   Verifying Hash Integrity ... md5+ sha1+ OK
   Booting using the fdt blob at 0x125d8dbc
   Uncompressing Kernel Image ... OK
   Using Device Tree in place at 125d8dbc, end 125e479d

Starting kernel ...
```

Verified boot

➔ Verify the loaded software to ensure that it is authorized during boot





Verified boot, cont

- Build FIT-enabled U-Boot
  - CONFIG_FIT=y
  - CONFIG_FIT_SIGNATURE=y
  - CONFIG_RSA=y
- Create vmlinux.bin
  - $ arm-linux-gnueabi-objcopy -O binary vmlinux vmlinux.bin
  - $ gzip vmlinux.bin
  - $ cp imx6q-icore-rqs.dtb imx6q-icore-rqs-pubkey.dtb
- FIT input, with added signature
- Generate RSA keys
  - $ mkdir mykeys
  - $ openssl genrsa -F4 -out mykeys/eng.key 2048
  - $ openssl req -batch -new -x509 -key mykeys/eng.key -out mykeys/eng.crt
- FIT output
  - mkimage -f kernel_fdt.its -K imx6q-icore-rqs-pubkey.dtb -k mykeys/ -r fit.itb
- Build U-Boot proper with public key
  - make DEV_TREE_BIN=../imx6q-icore-rqs-pubkey.dtb
- Boot verified boot

Verified boot, cont

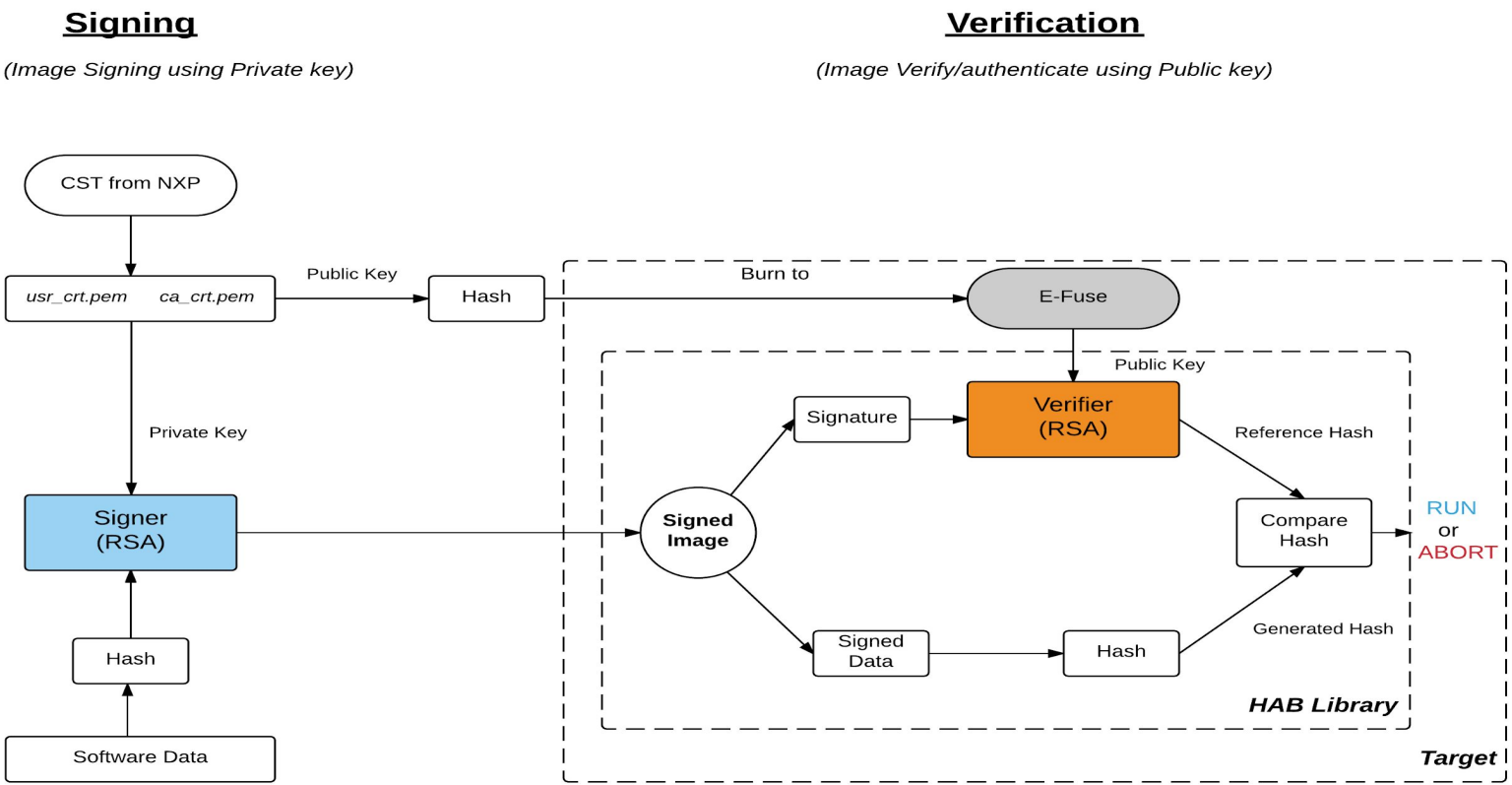
```
/* Simple U-Boot uImage source file containing a single kernel and FDT blob */
/dts-v1/;
/ {
    description = "Verified RSA image with single Linux kernel and FDT blob";
    #address-cells = <1>;
    images {
        kernel@1 {
            description = "i.MX6 Linux kernel";
            data = /incbin/("./vmlinux.bin.gz");
            type = "kernel";
            arch = "arm";
            os = "linux";
            compression = "gzip";
            load = <0x10008000>;
            entry = <0x10008000>;
            hash@1 {
                algo = "md5";
            };
            hash@2 {
                algo = "sha1";
            };
            signature@1 {
                algo = "sha1,rsa2048";
                key-name-hint = "eng";
            };
        };

        fdt@1 {
            description = "Engicam i.CoreM6 Quad/Dual RQS Starter Kit Devicetree blob";
            data = /incbin/("./imx6q-icore-rqs.dtb");
            type = "flat_dt";
            arch = "arm";
            compression = "none";
            hash@1 {
                algo = "md5";
            };
            hash@2 {
                algo = "sha1";
            };
            signature@1 {
                algo = "sha1,rsa2048";
                key-name-hint = "eng";
            };
        };
    };

    configurations {
        default = "conf@1";
        conf@1 {
            description = "Boot Linux kernel with FDT blob";
            kernel = "kernel@1";
            fdt = "fdt@1";
        };
    };
};
```

```
icorem6qdl-rqs> fatload mmc 0:1 $loadaddr fit.itb
reading fit.itb
6167494 bytes read in 335 ms (17.6 MiB/s)
icorem6qdl-rqs> bootm $loadaddr
Booting FIT image from mmc ...
## Loading kernel from FIT Image at 12000000 ...
   Using 'conf@1' configuration
   Verifying Hash Integrity ... OK
   Trying 'kernel@1' kernel subimage
     Description:  i.MX6 Linux kernel
     Type:         Kernel Image
     Compression:  gzip compressed
     Data Start:   0x120000f0
     Data Size:    6130148 Bytes = 5.8 MiB
     Architecture: ARM
     OS:           Linux
     Load Address: 0x10008000
     Entry Point:  0x10008000
     Hash algo:    md5
     Hash value:   b975a202ea2963c53c53f527329930cd
     Hash algo:    sha1
     Hash value:   78b93fe404b795de8c837af27d67f4df9b96083a
     Sign algo:    sha1,rsa2048:eng
     Sign value:   4288ce2c7380a90b7b7b9c000760f086fe67560d16fb5ea85bc792ff3ed70e381956bbff99c514213e00e3d21838650ada0eb68439e253ef493e3e0098e0d47109d3e
   Verifying Hash Integrity ... md5+ sha1+ sha1,rsa2048:eng- OK
## Loading fdt from FIT Image at 12000000 ...
   Using 'conf@1' configuration
   Trying 'fdt@1' fdt subimage
     Description:  Engicam i.CoreM6 Quad/Dual RQS Starter Kit Devicetree blob
     Type:         Flat Device Tree
     Compression:  uncompressed
     Data Start:   0x125d8dbc
     Data Size:    35298 Bytes = 34.5 KiB
     Architecture: ARM
     Hash algo:    md5
     Hash value:   4371a4dfe55127c2fda8a9feb4d3b313
     Hash algo:    sha1
     Hash value:   e34a9326b5e7fd43557753ef980fe67326f82ea1
     Sign algo:    sha1,rsa2048:eng
     Sign value:   94cebd60a6ff2e123ed763760b88c026b74b12eb9c37a97d73eec1a25e01d6e29284f393c5ca20951a605378bf8b547bdc0ce0aae16e069e6db0c5af7f00d4cfc6c94
   Verifying Hash Integrity ... md5+ sha1+ sha1,rsa2048:eng- OK
   Booting using the fdt blob at 0x125d8dbc
   Uncompressing Kernel Image ... OK
   Using Device Tree in place at 125d8dbc, end 125e479d

Starting kernel ...
```

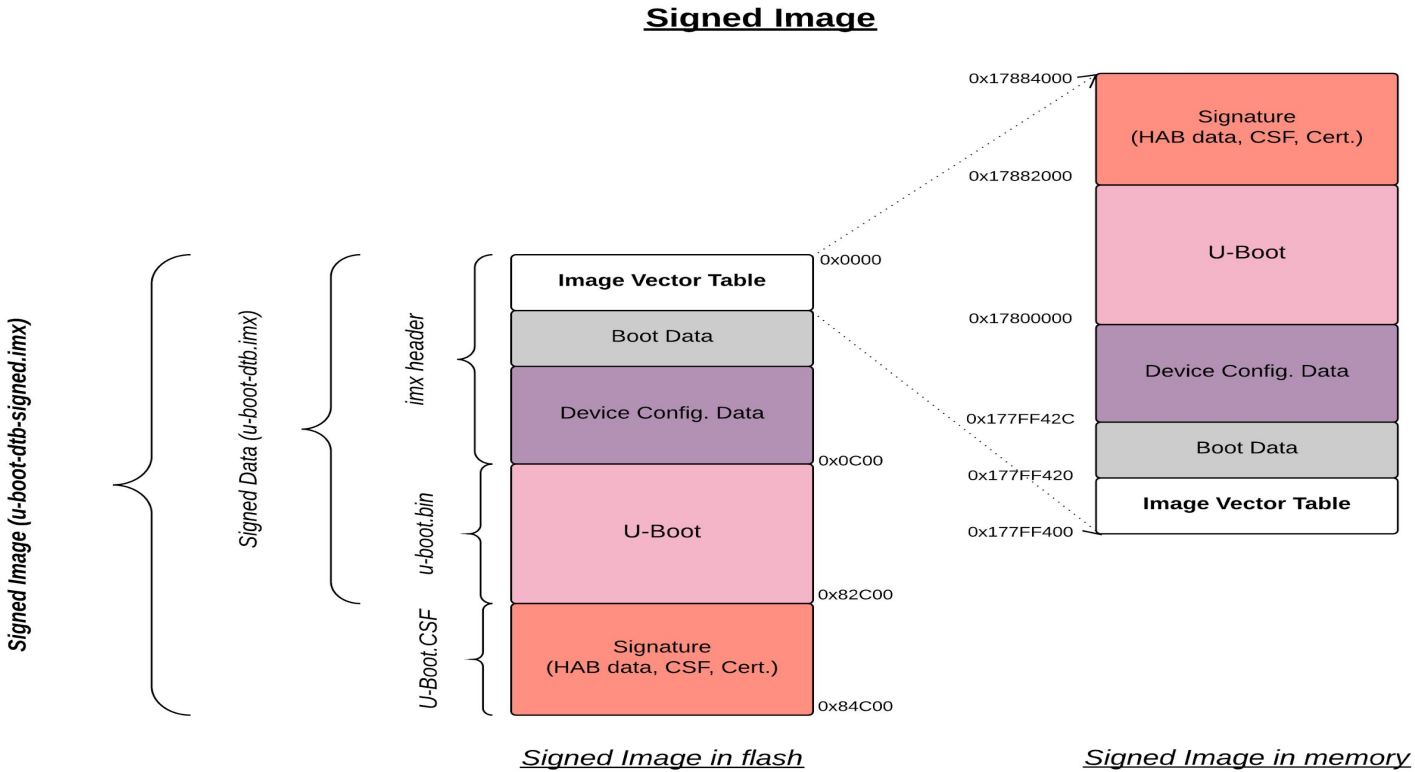
Secure boot

- Verify the loaded software to ensure that it is authorized during runtime is usually termed as Verified or Secure or Trusted system mechanism.
- Trusted execution, example via HAB
- Two possible ways of secure boot
  - Signed boot
  - Encrypted boot
- More information on https://openedev.amarulasolutions.com/display/ODWIKI/i.MX6+HABv4

Secure boot, Signed



Secure boot, Signed format



Note: Copy signed image at offset 0x400 in flash

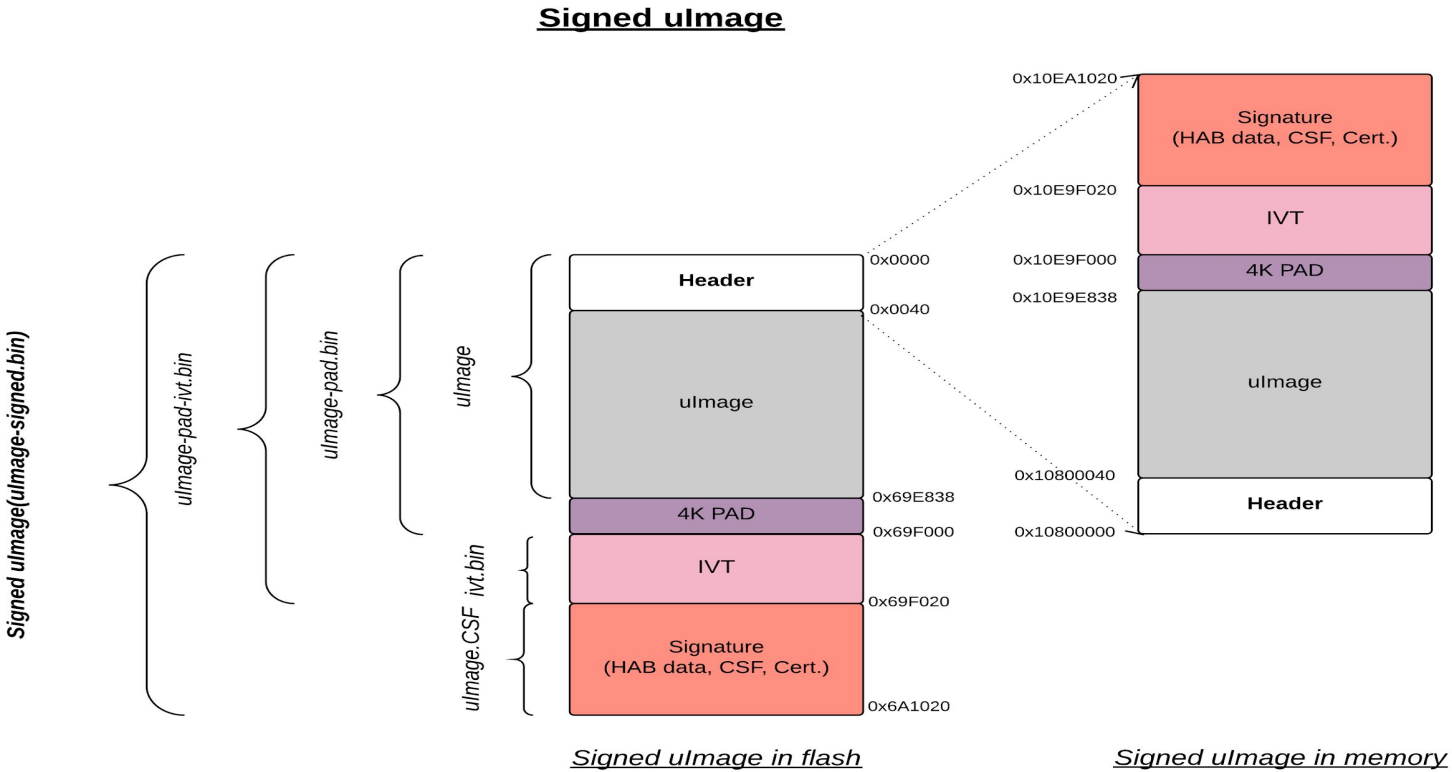
Secure boot, e-fuse burn

```
icorem6qdl> fuse prog -y 3 0 0x67C7506F
Programming bank 3 word 0x00000000 to 0x67c7506f...
icorem6qdl> fuse prog -y 3 1 0x7D51EFD0
Programming bank 3 word 0x00000001 to 0x7d51efd0...
icorem6qdl> fuse prog -y 3 2 0x9E450811
Programming bank 3 word 0x00000002 to 0x9e450811...
icorem6qdl> fuse prog -y 3 3 0x74ED8483
Programming bank 3 word 0x00000003 to 0x74ed8483...
icorem6qdl> fuse prog -y 3 4 0xB774A2A
Programming bank 3 word 0x00000004 to 0x0b774a2a...
icorem6qdl> fuse prog -y 3 5 0xD33FF045
Programming bank 3 word 0x00000005 to 0xd33ff045...
icorem6qdl> fuse prog -y 3 6 0x3343F187
Programming bank 3 word 0x00000006 to 0x3343f187...
icorem6qdl> fuse prog -y 3 7 0xC86DDA92
Programming bank 3 word 0x00000007 to 0xc86dda92...
```

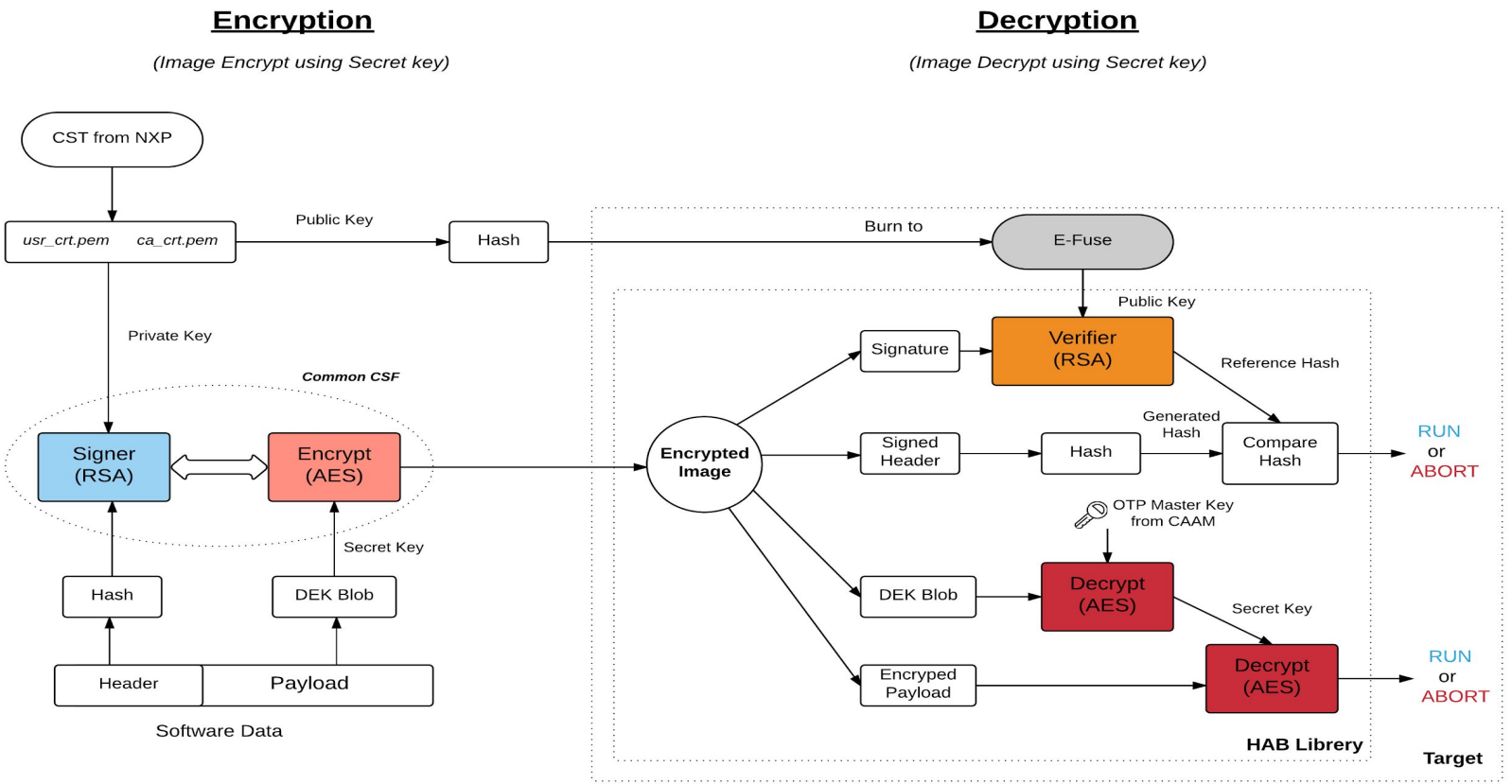
Secure boot, Enable (are you sure??)

```
icorem6qdl> hab_status

Secure boot disabled

HAB Configuration: 0xf0, HAB State: 0x66
No HAB Events Found!

icorem6qdl> fuse prog 0 6 0x2
Programming bank 0 word 0x00000006 to 0x00000002...
Warning: Programming fuses is an irreversible operation!
      This may brick your system.
      Use this command only if you are sure of what you are doing!

Really perform this fuse programming? <y/N>
y
icorem6qdl> reset
resetting ..
```

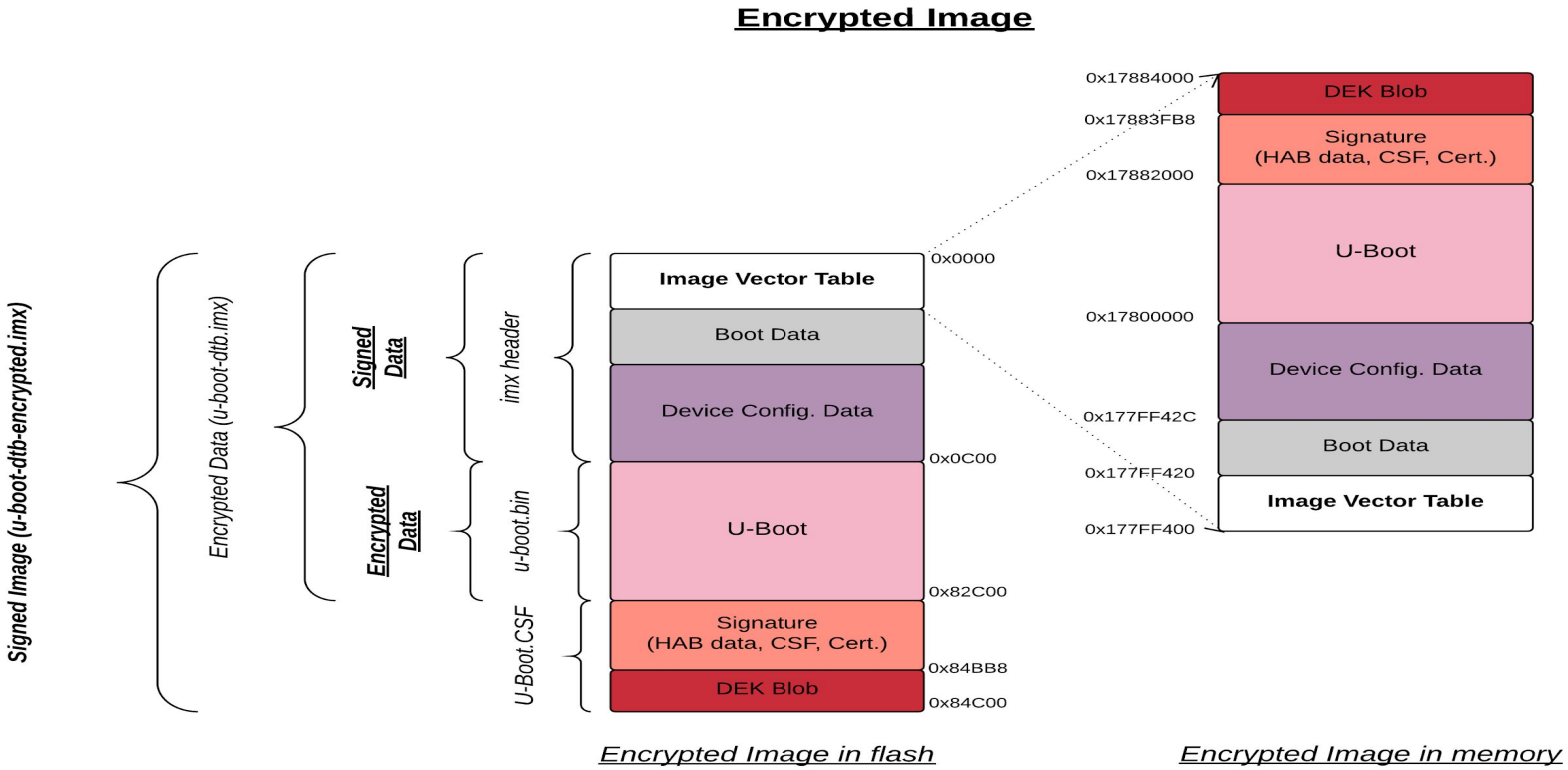
Secure boot, Signed-uImage



Secure boot, Encrypted

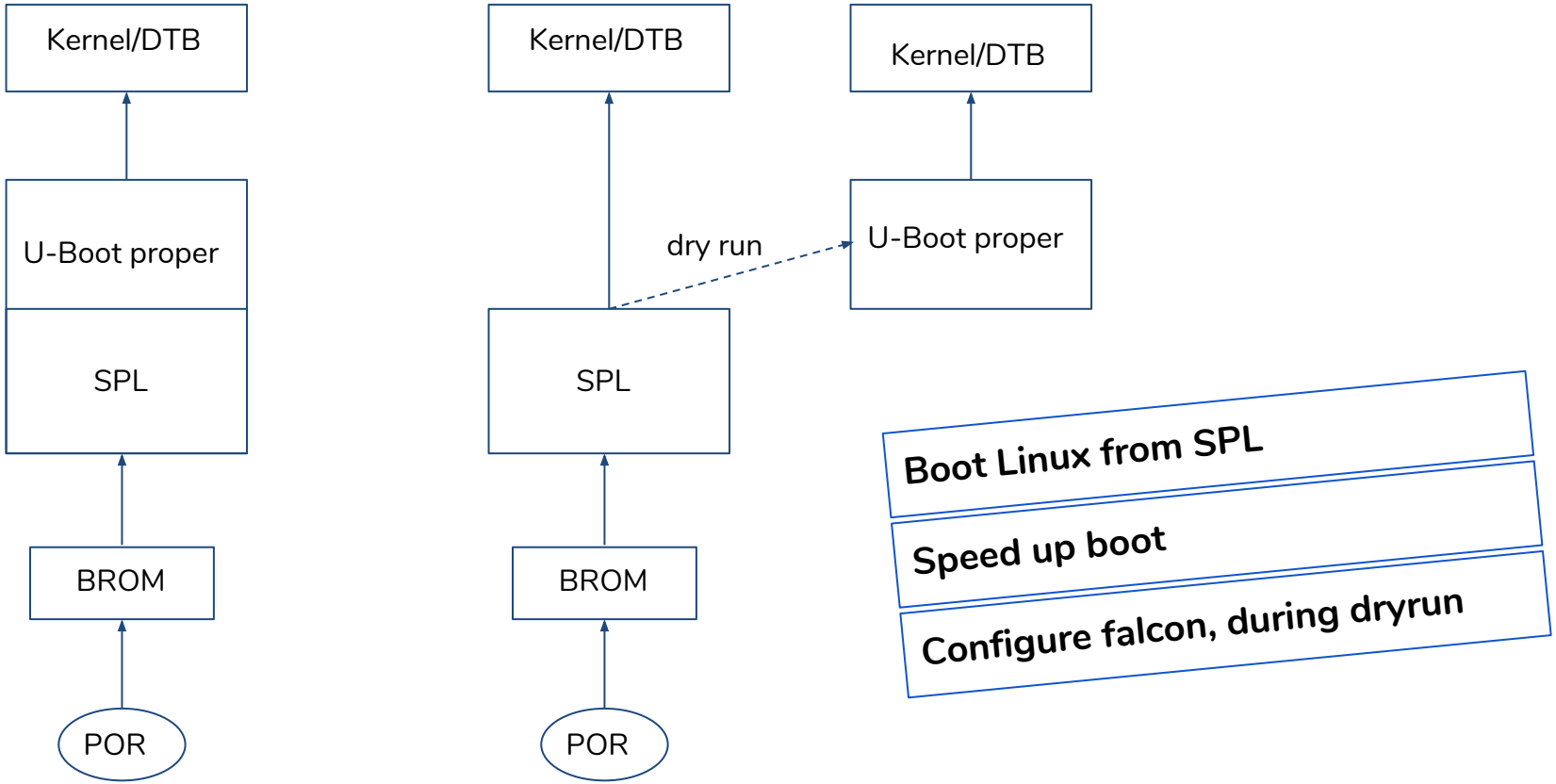


Secure boot, Encrypted format



Note: Copy signed image at offset 0x400 in flash

Falcon mode


Kernel/DTB
U-Boot proper
SPL
BROM
POR
Kernel/DTB
dry run
SPL
BROM
POR
Kernel/DTB
U-Boot proper
Boot Linux from SPL
Speed up boot
Configure falcon, during dryrun

Falcon, CONFIG_

```
/* Falcon Mode */
#define CONFIG_SPL_FS_LOAD_ARGS_NAME    "args"
#define CONFIG_SPL_FS_LOAD_KERNEL_NAME  "uImage"
#define CONFIG_CMD_SPL
#define CONFIG_SYS_SPL_ARGS_ADDR        0x0ffe5000
#define CONFIG_CMD_SPL_WRITE_SIZE      (128 * SZ_1K)

/* Falcon Mode - MMC support: args@16MB kernel@17MB */
#define CONFIG_SYS_MMCSD_RAW_MODE_ARGS_SECTOR           0x8000  /* 16MB */
#define CONFIG_SYS_MMCSD_RAW_MODE_ARGS_SECTORS          (CONFIG_CMD_SPL_WRITE_SIZE / 512)
#define CONFIG_SYS_MMCSD_RAW_MODE_KERNEL_SECTOR         0x8800  /* 17MB */
#endif

/* Enable Falcon */
#define CONFIG_SPL_OS_BOOT

int spl_start_uboot(void)
{
        /* break into full u-boot on 'c' */
        if (serial_tstc() && serial_getc() == 'c')
                return 1;

        return 0;
}
```

Falcon, configure

➔ Load device tree from rootfs
=> ext4load mmc 1:1 $fdt_addr_r /boot/rk3288-vyasa.dtb

➔ Load kernel from rootfs
=> ext4load mmc 1:1 $kernel_addr_r /boot/uImage

➔ Write kernel at 17 MB offset
=> mmc write $kernel_addr_r 0x8800 0x4000

➔ Setup kernel bootargs
=> setenv bootargs 'console=ttyS2,115200n8 root=/dev/mmcblk0p1 rootwait'

➔ Prepare args
=> *spl export fdt $kernel_addr_r - $fdt_addr_r*

➔ Write args at 16 MB offset
=> *mmc write 0x0ffe3000 0x8000 0x800*

Falcon, measure boot

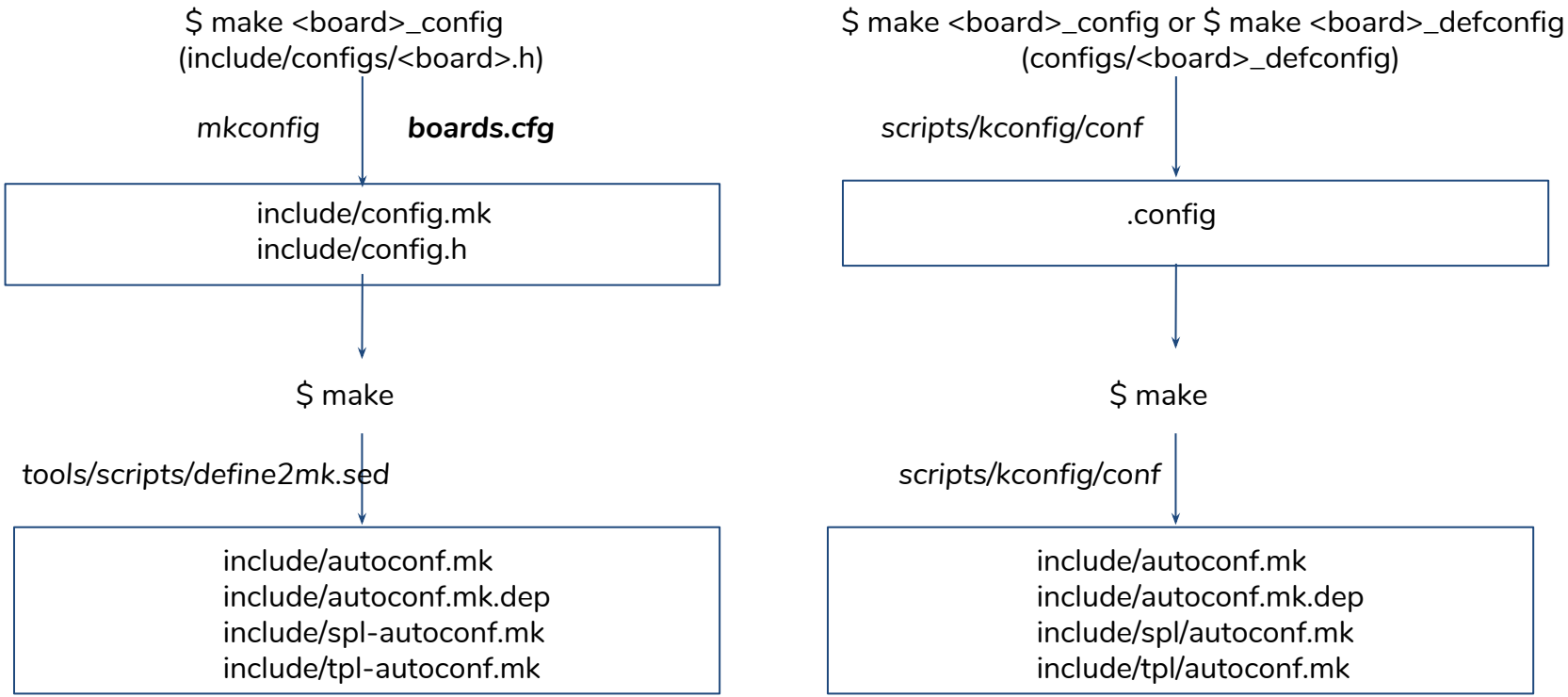
```
[0.001344 0.001342] U-Boot TPL 2017.09-rc2-13373-g2cffd0d-dirty (Aug 31 2017 - 20:41:14)
[0.005975 0.004631] Trying to boot from BOOTROM
[0.008394 0.002419] Returning to boot ROM...
[0.216735 0.208341]
[0.217195 0.000460] U-Boot SPL 2017.09-rc2-13373-g2cffd0d-dirty (Aug 31 2017 - 20:41:14)
[0.223097 0.005902] Trying to boot from MMC1[0.262093 0.038996] Expected Linux image is not found. Trying to start U-boot
[0.436129 0.174036]
[0.436416 0.000287]
[0.436696 0.000280] U-Boot 2017.09-rc2-13373-g2cffd0d-dirty (Aug 31 2017 - 20:41:14 +0530)
[0.442273 0.005577][0.442369 0.000096] Model: Amarula Vyasa-RK3288
[0.444854 0.002485] DRAM:  2 GiB
[0.479422 0.034568] MMC:   dwmmc@ff0c0000: 1
[0.627295 0.147873] *** Warning - bad CRC, using default environment
[0.631527 0.004232]
[0.635980 0.004453] In:    serial@ff690000
[0.637982 0.002002] Out:   serial@ff690000
[0.640004 0.002022] Err:   serial@ff690000
[0.642244 0.002240] Model: Amarula Vyasa-RK3288
[0.644783 0.002539] Net:   Net Initialization Skipped
[0.647824 0.003041] No ethernet found.
[0.651954 0.004130] Hit any key to stop autoboot:  0
[0.772503 0.120549] switch to partitions #0, OK
[0.774802 0.002299] mmc1 is current device
[1.069762 0.294960] Scanning mmc 1:1...
[1.312386 0.242624] Found /boot/extlinux/extlinux.conf
[1.315209 0.002823] Retrieving file: /boot/extlinux/extlinux.conf
[1.353460 0.038251] 145 bytes read in 28 ms (4.9 KiB/s)
[1.356412 0.002952] 1:    Vyasa Linux-4.13
[1.358237 0.001825] Retrieving file: /boot/uImage
[1.740753 0.382516] 7836344 bytes read in 375 ms (19.9 MiB/s)
[1.744302 0.003549] append: console=ttyS2,115200n8 root=/dev/mmcblk0p1 rootwait quiet
[1.750086 0.005784] Retrieving file: /boot/rk3288-vyasa.dtb
[1.786952 0.036866] 36291 bytes read in 28 ms (1.2 MiB/s)
[1.789950 0.002998] ## Booting kernel from Legacy Image at 02000000 ...
[1.794524 0.004574]    Image Name:   Linux-4.13.0-rc4-next-20170810-0
[1.799040 0.004516]    Image Type:   ARM Linux Kernel Image (uncompressed)
[1.803818 0.004778]    Data Size:    7836280 Bytes = 7.5 MiB
[1.807426 0.003608]    Load Address: 02000000
[1.809776 0.002350]    Entry Point:  02000000
[1.812097 0.002321]    Verifying Checksum ... OK
[1.945372 0.133275] ## Flattened Device Tree blob at 01f00000
[1.948106 0.002734]    Booting using the fdt blob at 0x1f00000
[1.951715 0.003609]    Loading Kernel Image ... OK
[1.968363 0.016648]    Loading Device Tree to 0fff4000, end 0ffffdc2 ... OK
[1.974709 0.006346]
[1.974911 0.000202] Starting kernel ...
[0.002243 0.002243]
[1.273396 1.271153] [    0.090111] dmi: Firmware registration failed.
[1.617881 0.344485] [    0.581926] EXT4-fs (mmcblk0p1): couldn't mount as ext3 due to feature incompatibilities
[1.627592 0.009711] [    0.592177] EXT4-fs (mmcblk0p1): couldn't mount as ext2 due to feature incompatibilities
[4.596400 2.968808] Starting logging: OK
[4.610216 0.013816] Initializing random number generator... done.
[4.622379 0.012163] Starting network: OK
[4.746329 0.123950]
[4.748163 0.001834] Welcome to VYASA RK3288!
[4.750725 0.002562] vyasa-rk3288 login:
```

```
[0.001135 0.001134] U-Boot TPL 2017.09-rc2-13373-g2cffd0d-dirty (Aug 31 2017 - 20:41:14)
[0.005690 0.004555] Trying to boot from BOOTROM
[0.008221 0.002531] Returning to boot ROM...
[0.196488 0.188267]
[0.196704 0.000216] U-Boot SPL 2017.09-rc2-13373-g2cffd0d-dirty (Aug 31 2017 - 20:41:14)
[0.202759 0.006055] Trying to boot from MMC1
[1.879613 1.676854] [    0.090151] dmi: Firmware registration failed.
[2.287880 0.408267] [    0.645755] EXT4-fs (mmcblk0p1): couldn't mount as ext3 due to feature incompatibilities
[2.301935 0.014055] [    0.660209] EXT4-fs (mmcblk0p1): couldn't mount as ext2 due to feature incompatibilities
[2.425052 0.123117] Starting logging: OK
[2.440868 0.015816] Initializing random number generator... done.
[2.452302 0.011434] Starting network: OK
[2.580451 0.128149]
[2.584354 0.003903] Welcome to VYASA RK3288!
[2.586697 0.002343] vyasa-rk3288 login:
```

U-Boot Features

- ➔ Kconfig
- ➔ FDT
- ➔ OF livetree
- ➔ FDT Overlay
- ➔ Driver model
- ➔ OF platdata
- ➔ DFU

Kconfig


$ make <board>_config
(include/configs/<board>.h)
mkconfig
boards.cfg
include/config.mk
include/config.h
$ make
tools/scripts/define2mk.sed
include/autoconf.mk
include/autoconf.mk.dep
include/spl-autoconf.mk
include/tpl-autoconf.mk
$ make <board>_config or $ make <board>_defconfig
(configs/<board>_defconfig)
scripts/kconfig/conf
.config
$ make
scripts/kconfig/conf
include/autoconf.mk
include/autoconf.mk.dep
include/spl/autoconf.mk
include/tpl/autoconf.mk

Kconfig, contd

- ➔ Still maintain old configuration system
- ➔ autoconf.mk for U-Boot proper, SPL, TPL
- ➔ *./scripts/config_whitelist.tx*, update recursively once the specific CONFIG_ moved
- ➔ new CONFIG_ options are Kconfig

- ➔ Sample, ***drivers/phy/allwinner/Makefile***

```
1 obj-$(CONFIG_PHY_SUN4I_USB)                  += phy-sun4i-usb.o
```

- ➔ Sample, ***drivers/phy/allwinner/Kconfig***

```
 1 config PHY_SUN4I_USB
 2         bool "Allwinner Sun4I USB PHY driver"
 3         depends on ARCH_SUNXI
 4         select PHY
 5         help
 6           Enable this to support the transceiver that is part of Allwinner
 7           sunxi SoCs.
 8
 9           This driver controls the entire USB PHY block, both the USB OTG
10           parts, as well as the 2 regular USB 2 host PHYs.
```

FDT (Flat Device Tree)

- ➔ Run-time hardware configuration
- ➔ Single U-Boot binary for multiple boards (with board controlled dts)
- ➔ Handle via libfdt
- ➔ Enabled via CONFIG_OF_CONTROL

- ➔ FDT supported U-Boot can build
  - ◆ with default dts, CONFIG_DEFAULT_DEVICE_TREE=<dts-file-name> in defconfig
    - $ make
  - ◆ with user-specified dts
    - $ make DEVICE_TREE=<dts-file-name>

- ➔ Sample, UniPhier Pro4 reference, Pro4 Ace, Pro4 Sanji boards
  - ◆ $ make uniphier_v7_defconfig (single configuration)
  - ◆ $ make DEVICE_TREE=uniphier-pro4-ref
  - ◆ $ make DEVICE_TREE=uniphier-pro4-ace
  - ◆ $ make DEVICE_TREE=uniphier-pro4-sanji

- ➔ DTB packing during build
  - ◆ CONFIG_OF_EMBED
  - ◆ CONFIG_OF_SEPARATE

FDT, contd

```
/* arch/arm/dts/imx6q.dtsi */
/ {
        cpus {
                #address-cells = <1>;
                #size-cells = <0>;

                cpu0: cpu@0 {
                        compatible = "arm,cortex-a9";
                        device_type = "cpu";
                        reg = <0>;
                        [...]
                };
        [...]
};
```

```
/* arch/arm/dts/imx6qdl.dtsi */
/ {
        soc {
                #address-cells = <1>;
                #size-cells = <1>;
                compatible = "simple-bus";
                [...]

                aips-bus@02100000 { /* AIPS2 */
                        compatible = "fsl,aips-bus", "simple-bus";
                        #address-cells = <1>;
                        #size-cells = <1>;
                        reg = <0x02100000 0x100000>;
                        [...]

                        usdhc3: usdhc@02198000 {
                                compatible = "fsl,imx6q-usdhc";
                                reg = <0x02198000 0x4000>;
                                interrupts = <0 24 IRQ_TYPE_LEVEL_HIGH>;
                                clocks = <&clks IMX6QDL_CLK_USDHC3>,
                                         <&clks IMX6QDL_CLK_USDHC3>,
                                         <&clks IMX6QDL_CLK_USDHC3>;
                                clock-names = "ipg", "ahb", "per";
                                bus-width = <4>;
                                status = "disabled";
                        };

                        [...]
        };
};
```

```
/* arch/arm/dts/imx6q-icore-rqs.dts */
/ {
        model = "Engicam i.CoreM6 Quad/Dual Starter Kit";
        compatible = "engicam,imx6-icore", "fsl,imx6q";
};

&usdhc3 {
        pinctrl-names = "default";
        pinctrl-0 = <&pinctrl_usdhc3>;
        cd-gpios = <&gpio1 1 GPIO_ACTIVE_LOW>;
        no-1-8-v;
        status = "okay";
};
```

FDT, u-boot

- ➔ Maintain U-Boot specific node definitions in separate file
- ➔ Useful for DT allocation in SPL
- ➔ u-boot,dm-pre-alloc, u-boot,dm-spl

```
/* arch/arm/dts/imx6qdl-u-boot.dtsi */
/ {
      soc {
                  u-boot,dm-spl;
                  aips-bus@02000000 {
                        u-boot,dm-spl;
                  };
       };
};

&gpio1 {
      u-boot,dm-spl;
};
```

FDT, contd

```c
int fdt_delprop(void *fdt, int nodeoffset, const char *name)
{
        struct fdt_property *prop;
        int len, proplen;

        FDT_RW_CHECK_HEADER(fdt);

        prop = fdt_get_property_w(fdt, nodeoffset, name, &len);
        if (!prop)
                return len;

        proplen = sizeof(*prop) + FDT_TAGALIGN(len);
        return fdt_splice_struct_(fdt, prop, proplen, 0);
}

int ft_board_setup(void *blob, bd_t *bd)
{
        int nodeoffset;

        nodeoffset = fdt_path_offset(blob, "/soc/aips-bus@02100000/usdhc@02198000");

        return fdt_delprop(blob, nodeoffset, "no-1-8-v");
}
```

? add/update, copy large amount

? tree need to rebuilt

? tree traversing is slow

Livetree (Live Device Tree)

- ➔ Pointer-based *hierarchical structures*
- ➔ Support after relocation
- ➔ ofnode , point to either flat tree or livetree
- ➔ Enabled via CONFIG_OF_LIVE

```
static int zynq_spi_ofdata_to_platdata(struct udevice *bus)
{
     struct zynq_spi_platdata *plat = bus->platdata;

     /* old code */
     plat->regs = (struct zynq_spi_regs *)devfdt_get_addr(bus);
     plat->frequency = fdtdec_get_int(blob, node, "spi-max-frequency",
250000000);

     /* new code */
     plat->regs = (struct zynq_spi_regs *)dev_read_addr(bus);
     plat->frequency = dev_read_u32_default(bus, "spi-max-frequency", 250000000);

    return 0;
}
```

FDT Overlay

- ➔ DTO, enable centralize DTB to be overlaid on the device tree.
- ➔ Single image of multitude of similar boards and their expansion options
- ➔ DTO can load U-Boot via
  - ◆ FIT image
  - ◆ Manual load

```
/dts-v1/;
/ {
    images {
        kernel {
                data = /incbin/("./zImage");
                type = "kernel";
                load = <0x10080000>;
                entry = <0x10080000>;
        };
        fdt-1 {
                data = /incbin/("./imx6q-icore.dtb");
                type = "flat_dt";
        };
        fdt-2 {
                data = /incbin/("./imx6q-icore-mipi.dtb");
                type = "flat_dt";
        };
        configurations {
                default = "imx6q-icore.dtb";
                imx6q-icore.dtb {
                        kernel = "kernel";
                        fdt = "fdt-1";
                };
                imx6q-icore-mipi.dtb {
                        kernel = "kernel";
                        fdt = "fdt-2";
                };
        };
    };
};
```

FDT Overlay, loading

```
/* via FIT */
=> bootm $loadaddr#imx6q-icore.dtb#imx6q-icore-mipi

/* Manual load */
=> setenv fdt_addr 0x18000000
=> setenv fdt_ovaddr 0x180c0000

=> load mmc 0:1 ${fdt_addr} ${bootdir}/base.dtb
=> load mmc 0:1 ${fdt_ovaddr} ${bootdir}/overlay.dtb

=> fdt resize 8192

=> fdt apply $fdt_ovaddr

=> bootm ${loadaddr} - ${fdt_addr}
```

U-Boot Driver model

- ➔ Driver model
- ➔ Clock framework
- ➔ Reset framework
- ➔ PHY framework
- ➔ Pinctrl framework
- ➔ Power domain framework
- ➔ PMIC framework
- ➔ Regulator framework
- ➔ RTC framework
- ➔ Thermal framework
- ➔ Timer framework
- ➔ Serial framework
- ➔ SPI framework
- ➔ SPI Flash framework
- ➔ I2C framework
- ➔ GPIO framework
- ➔ Ethernet framework
- ➔ Block framework
- ➔ USB framework

Driver model

? ad-hoc model, direct functions call

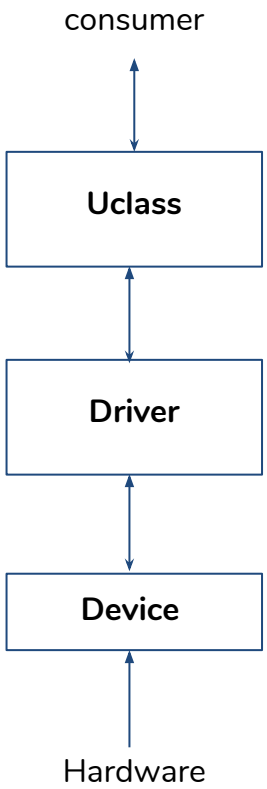
? multiple controllers can't fit same driver

? not scalable, difficult to maintain

Simple, scalable, modular, homogeneous

Lazy initialization, but bounded

Small overhead



```
UCLASS_DRIVER(spi) = {
	.id		= UCLASS_SPI,
	.name		= "spi",
	.flags		= DM_UC_FLAG_SEQ_ALIAS,
	.post_bind	= dm_scan_fdt_dev,
	.post_probe	= spi_post_probe,
	.child_pre_probe = spi_child_pre_probe,
	.per_device_auto_alloc_size = sizeof(struct dm_spi_bus),
	.per_child_auto_alloc_size = sizeof(struct spi_slave),
	.per_child_platdata_auto_alloc_size = sizeof(struct dm_spi_slave_platdata),
	.child_post_bind = spi_child_post_bind,
};
U_BOOT_DRIVER(zynq_qspi) = {
	.name	= "zynq_qspi",
	.id	= UCLASS_SPI,
	.of_match = zynq_qspi_ids,
	.ops	= &zynq_qspi_ops,
	.ofdata_to_platdata = zynq_qspi_ofdata_to_platdata,
	.platdata_auto_alloc_size = sizeof(struct zynq_qspi_platdata),
	.priv_auto_alloc_size = sizeof(struct zynq_qspi_priv),
	.probe	= zynq_qspi_probe,
};
```

```
=> dm tree
 Class     index  Probed  Driver      Name
-----------------------------------------
 root          0  [ + ]   root_drive  root_driver
 clk           0  [   ]   fixed_rate  |-- oscillator
 mmc           0  [ + ]   rockchip_r  |-- dwmmc@ff0c0000
 blk           0  [ + ]   mmc_blk     |   `-- dwmmc@ff0c0000.blk
 mmc           1  [ + ]   rockchip_r  |-- dwmmc@ff0f0000
 blk           1  [   ]   mmc_blk     |   `-- dwmmc@ff0f0000.blk
 serial        0  [ + ]   ns16550_se  |-- serial@ff690000
 eth           0  [   ]   gmac_rockc  |-- ethernet@ff290000
 usb           0  [   ]   dwc2_usb    |-- usb@ff540000
 usb           1  [   ]   dwc2_usb    |-- usb@ff580000
 ram           0  [   ]   rockchip_r  |-- dmc@ff610000
 i2c           0  [ + ]   i2c_rockch  |-- i2c@ff650000
 pmic          0  [ + ]   rk8xx pmic  |   `-- pmic@1b
 regulator     0  [   ]   rk8xx_buck  |       |-- DCDC_REG1
 regulator     1  [   ]   rk8xx_buck  |       |-- DCDC_REG2
 regulator     2  [   ]   rk8xx_buck  |       |-- DCDC_REG3
 regulator     3  [   ]   rk8xx_buck  |       |-- DCDC_REG4
 regulator     4  [   ]   rk8xx_ldo   |       |-- LDO_REG1
```

CLK framework

Clock provider

- ➔ via UCLASS_CLK
- ➔ uclass core: drivers/clk/clk-uclass.c
- ➔ platform specific uboot driver: drivers/clk/sunxi/
- ➔ CLK enable, disable, set_rate, get_rate via clk_ops
- ➔ include/clk-uclass.h

Clock consumer

- ➔ Get the clk
- ➔ Consume CLK framework via clk_enable, get_rate, set_rate, disable

Reset framework

Reset provider

- ➔ via UCLASS_RESET
- ➔ uclass core: drivers/reset/reset-uclass.c
- ➔ platform specific uboot driver: drivers/reset/reset-sunxi.c
- ➔ RST deassert, assert via reset_ops
- ➔ include/reset-uclass.h

Reset consumer

- ➔ Get the reset
- ➔ Consume RESET framework via reset_deassert, assert

PHY framework

PHY provider

- ➔ via UCLASS_PHY
- ➔ uclass core: drivers/reset/reset-uclass.c
- ➔ platform specific uboot driver: drivers/phy/allwinner/phy-sun4i-usb.c
- ➔ PHY init, exit, power_on, power_off via phy_ops
- ➔ include/generic-phy.h

PHY consumer

- ➔ Get the phy
- ➔ Consume PHY framework via generic_phy_init, power_on, power_off, exit

```
static int sun4i_usb_phy_exit(struct phy *phy)
{
    clk_disable(&usb_phy->clocks);
    reset_assert(&usb_phy->resets);
}
static int sun4i_usb_phy_init(struct phy *phy)
{
    clk_enable(&usb_phy->clocks);
    reset_deassert(&usb_phy->resets);
}
static struct phy_ops sun4i_usb_phy_ops = {
    .init = sun4i_usb_phy_init,
    .exit = sun4i_usb_phy_exit,
};
static int sun4i_usb_phy_probe(struct udevice *dev)
{
    clk_get_by_name(dev, "usb0_phy", &phy->clocks);
    reset_get_by_name(dev, "usb0_reset", &phy->resets);
}
U_BOOT_DRIVER(sun4i_usb_phy) = {
    .id    = UCLASS_PHY,
    .ops = &sun4i_usb_phy_ops,
    .probe = sun4i_usb_phy_probe,
};
```

Pinctrl framework

Pinctrl provider

- ➔ via UCLASS_PINCTRL
- ➔ uclass core: drivers/pinctrl/pinctrl-uclass.c
- ➔ platform specific uboot driver: drivers/pinctrl/rockchip/pinctrl_rk3288.c
- ➔ Pinctrl pinmux_set, group_set, pinmux_set, group_set via pinctrl_ops
- ➔ include/dm/pinctrl.h

Pinctrl consumer

- ➔ Get the reset
- ➔ Consume Pinctrl framework via pinctrl_decode_pin_config, pinctrl_get_gpio_mux

Power Domain framework

- ➔ Control power on portion of SoC

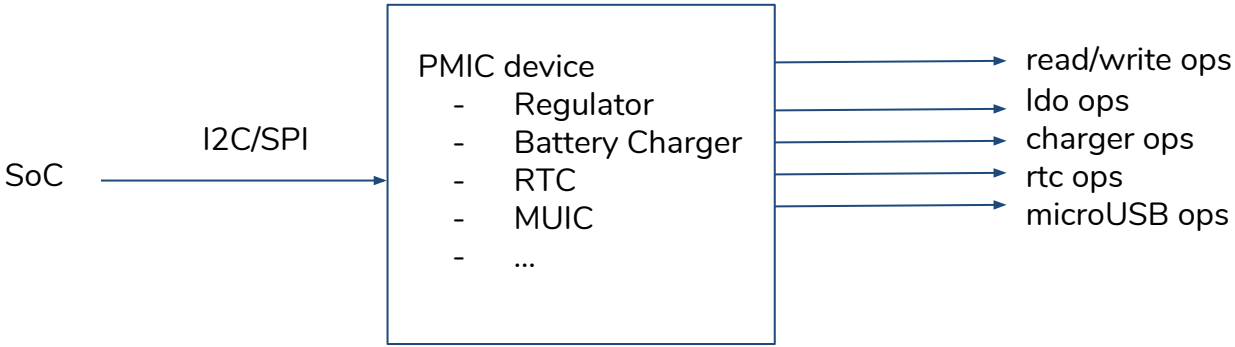
Power domain provider

- ➔ via UCLASS_POWER_DOMAIN
- ➔ uclass core: drivers/power/domain/power-domain-uclass.c
- ➔ platform specific uboot driver: drivers/power/domain/bcm6328-power-domain.c
- ➔ power domain on, off via power_domain_ops
- ➔ include/power-domain-uclass.h

Power domain consumer

- ➔ Get the power domain
- ➔ Consume power domain via power_domain_on/off

PMIC IC



➔ Basic PMIC read/write operations via UCLASS_PMIC
➔ PMIC features like voltage, current regulations via UCLASS_REGULATOR

PMIC framework

PMIC provider

- ➔ via UCLASS_PMIC
- ➔ uclass core: drivers/power/pmic/pmic-uclass.c
- ➔ platform specific uboot driver: drivers/power/pmic/pfuze100.c
- ➔ PMIC reg_count, read, write via dm_pmic_ops
- ➔ include/power/pmic.h

PMIC consumer

- ➔ Get the pmic
- ➔ Consume PMIC framework via pmic_read, write, reg_read, reg_write etc

Regulator framework

Regulator provider

- ➔ via UCLASS_REGULATOR
- ➔ uclass core: drivers/power/regulator/regulator-uclass.c
- ➔ platform specific uboot driver: drivers/power/regulator/pfuze100.c
- ➔ Regulator get/set value, set/get current, set/get mode, set/get enable via dm_regulator_ops
- ➔ include/power/regulator.h

PMIC consumer

- ➔ Get the supply regulator
- ➔ Consume Regulator framework via regulator_set/get value, current, mode etc

RTC framework

RTC provider

- ➔ via UCLASS_RTC
- ➔ uclass core: drivers/rtc/rtc-uclass.c
- ➔ platform specific uboot driver: drivers/rtc/mvrtc.c
- ➔ RTC get, set, read, read8, write8 via rtc_ops
- ➔ include/rtc.h

RTC consumer

- ➔ Get the rtc
- ➔ Consume RTC framework via dm_rtc_get, set, reset, read8, write8 etc

Thermal framework

Thermal provider

- ➔ via UCLASS_THERMAL
- ➔ uclass core: drivers/thermal/thermal-uclass.c
- ➔ platform specific uboot driver: drivers/thermal/imx_thermal.c
- ➔ Thermal get_temp via dm_thermal_ops
- ➔ include/thermal.h

Thermal consumer

- ➔ Get the thermal
- ➔ Consume Thermal framework via thermal_get_temp

Timer framework

Timer provider

- ➔ via UCLASS_TIMER
- ➔ uclass core: drivers/timer/timer-uclass.c
- ➔ platform specific uboot driver: drivers/timer/rockchip_timer.c
- ➔ Timer get_count via timer_ops
- ➔ include/timer.h

Timer consumer

- ➔ Get the timer
- ➔ Consume timer framework via timer_get_count

Serial framework

- ➔ via UCLASS_SERIAL
- ➔ uclass core: drivers/serial/serial-uclass.c
- ➔ platform specific uboot driver: drivers/serial/serial_mxc.c
- ➔ Serial setbrg, getc, putc, via dm_serial_ops
- ➔ include/serial.h
- ➔ board_f => serial_init()
- ➔ board_r => initr_serial()
- ➔ Early debug access via CONFIG_DEBUG_UART

```
#include <debug_uart.h>

static inline void _debug_uart_init(void)
{
    _mxc_serial_init(base);
 }

static inline void _debug_uart_putc(int ch)
{
    while (!(readl(&base->ts) & UTS_TXEMPTY))
      WATCHDOG_RESET();

    writel(ch, &base->txd);
}
DEBUG_UART_FUNCS

debug_uart_init();
printch('T');
printch('P');
printch('L');
```

SPI framework

- ➔ via UCLASS_SPI
- ➔ uclass core: drivers/spi/spi-uclass.c
- ➔ platform specific uboot driver: drivers/spi/zynq_spi.c
- ➔ SPI claim_bus, release_bus, set_speed, set_mode, xfer via dm_spi_ops
- ➔ include/spi.h
- ➔ cmd/spi.c

```c
static int zynq_spi_probe(struct udevice *bus)
{
      zynq_spi_init_hw(priv);
}
static int zynq_spi_ofdata_to_platdata(struct udevice *bus)
{
       plat->regs = (struct zynq_spi_regs *)dev_read_addr(bus);
}
static const struct udevice_id zynq_spi_ids[] = {
    { .compatible = "xlnx,zynq-spi-r1p6" },
    { }
};
U_BOOT_DRIVER(zynq_spi) = {
    .name    = "zynq_spi",
    .id    = UCLASS_SPI,
    .of_match = zynq_spi_ids,
    .ofdata_to_platdata = zynq_spi_ofdata_to_platdata,
    .platdata_auto_alloc_size = sizeof(struct zynq_spi_platdata),
    .priv_auto_alloc_size = sizeof(struct zynq_spi_priv),
    .probe    = zynq_spi_probe,
};
```

```
static int zynq_spi_xfer(struct udevice *dev, unsigned int bitlen,
                const void *dout, void *din, unsigned long flags)
{
     if (flags & SPI_XFER_BEGIN)
          spi_cs_activate(dev);
     /* process TX and RX FIFO */
     if (flags & SPI_XFER_END)
          spi_cs_deactivate(dev);
}

static const struct dm_spi_ops zynq_spi_ops = {
    .claim_bus    = zynq_spi_claim_bus,
    .release_bus    = zynq_spi_release_bus,
    .xfer         = zynq_spi_xfer,
    .set_speed    = zynq_spi_set_speed,
    .set_mode    = zynq_spi_set_mode,
};

U_BOOT_DRIVER(zynq_spi) = {
     .ops    = &zynq_spi_ops,
};
```

SPI FLASH framework

- ➔ via UCLASS_SPI_FLASH
- ➔ uclass core: drivers/mtd/spi/sf-uclass.c
- ➔ SPI Flash core driver: drivers/mtd/spi/spi_flash.c
- ➔ SPI Flash erase, write and read via dm_spi_flash_ops
- ➔ include/spi_flash.h
- ➔ cmd/sf.c

I2C framework

- ➔ via UCLASS_I2C
- ➔ uclass core: drivers/spi/spi-uclass.c
- ➔ platform specific uboot driver: drivers/i2c/mxc_i2c.c
- ➔ I2C probe_chip, set_bus_speed, xfer via dm_i2c_ops
- ➔ include/i2c.h
- ➔ cmd/i2c.c

GPIO framework

- ➔ via UCLASS_GPIO
- ➔ uclass core: drivers/gpio/gpio-uclass.c
- ➔ platform specific uboot driver: drivers/gpio/mxc_gpio.c
- ➔ GPIO direction_input, output, get/set value, get_function via dm_gpio_ops
- ➔ include/asm-generic/gpio.h
- ➔ cmd/gpio.c

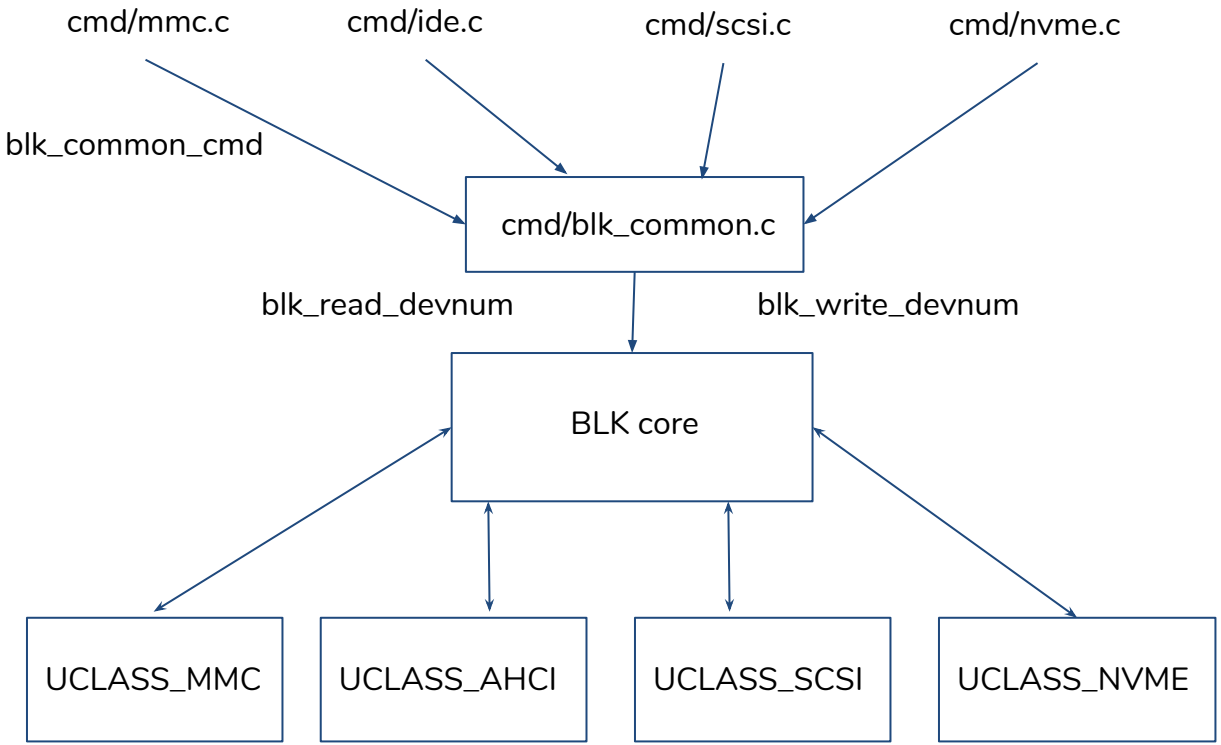
Ethernet framework

- ➔ via UCLASS_ETH
- ➔ uclass core: net/eth-uclass.c
- ➔ platform specific uboot driver: drivers/net/fec_mxc.c
- ➔ ETH start, send, recv, stop via eth_ops
- ➔ include/net.h
- ➔ cmd/ping.c

Block framework

- ➔ Generic Block layer Like Linux Block Subsystem
- ➔ Interfaces like IDE, SCSI, USB, MMC, SD, NVME etc
- ➔ via UCLASS_BLK
- ➔ uclass core: drivers/block/blk-uclass.c
- ➔ BLK read, write, erase via blk_ops

Generic Block Layer



USB framework

- ➔ via UCLASS_USB
- ➔ uclass core: drivers/usb/host/usb-uclass.c
- ➔ platform specific uboot driver: drivers/usb/host/ehci-generic.c
- ➔ USB control, bulk, interrupt, create_int_queue etc via dm_usb_ops
- ➔ include/usb.h
- ➔ cmd/usb.c
- ➔ USB Gadgets can probe via Gadget UCLASS
- ➔ drivers/usb/gadget/ether.c, USB ETH Gadget
- ➔ MUSB can operate Host and Peripheral
- ➔ MUSB Host access via UCLASS_USB
- ➔ MUSB Peripheral access via UCLASS_USB_DEV_GENERIC (may be fixed, not sure)
- ➔ drivers/usb/musb-new/sunxi.c, SunXi MUSB driver

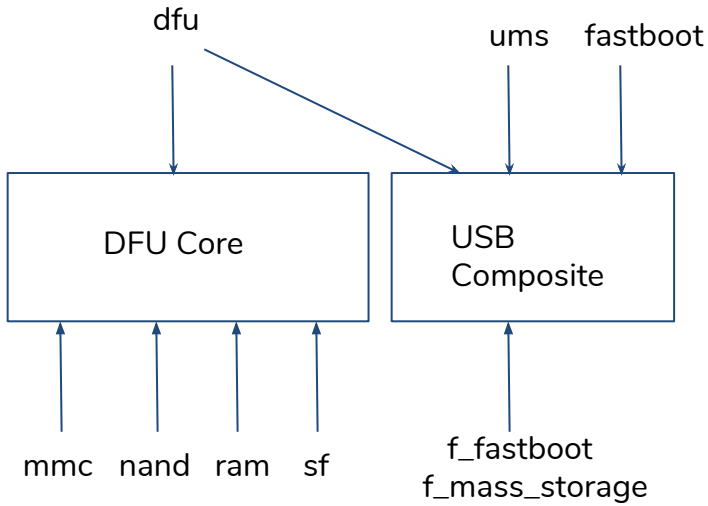
OF Platdata

- ➔ SPL size increases with FDT
- ➔ Enabled via CONFIG_SPL_OF_PLATDATA
- ➔ Explicitly define the device details Like legacy platform_device in Linux

```
#include <dm/platform_data/spi_davinci.h>

static const struct davinci_spi_platdata davinci_spi_data = {
.regs = (struct davinci_spi_regs *)0x01f0e000,
.num_cs = 4,
};

U_BOOT_DEVICE(davinci_spi) = {
.name = "davinci_spi",
.platdata = &davinci_spi_data,
};
```

Firmware Upgrade

- ➔ Upgrade firmware images on running U-Boot
- ➔ DFU
- ➔ DFU via tftp
- ➔ UMS
- ➔ Fastboot



```
=> setenv dfu_alt_info "spl raw 0x2 0x400"
=> dfu 0 mmc 2
$ dfu-util -D SPL -a spl
=> dfu tftp 0 mmc 0

=> mmc dev 1
=> mmc part

Partition Map for MMC device 0  --   Partition Type: EFI

Part  Start LBA       End LBA         Name
      Attributes
      Type GUID
      Partition GUID
  1   0x00000010      0x0000004f      "loader1"
      attrs:  0x0000000000000000
      type:   ebd0a0a2-b9e5-4433-87c0-68b6b72699c7
      guid:   d117f98e-6f2c-d04b-a5b2-331a19f91cb2
=> fastboot 0
$ fastboot -i 0x1f3a flash loader1 spl/sunxi-spl.bin
```

Port new hardware

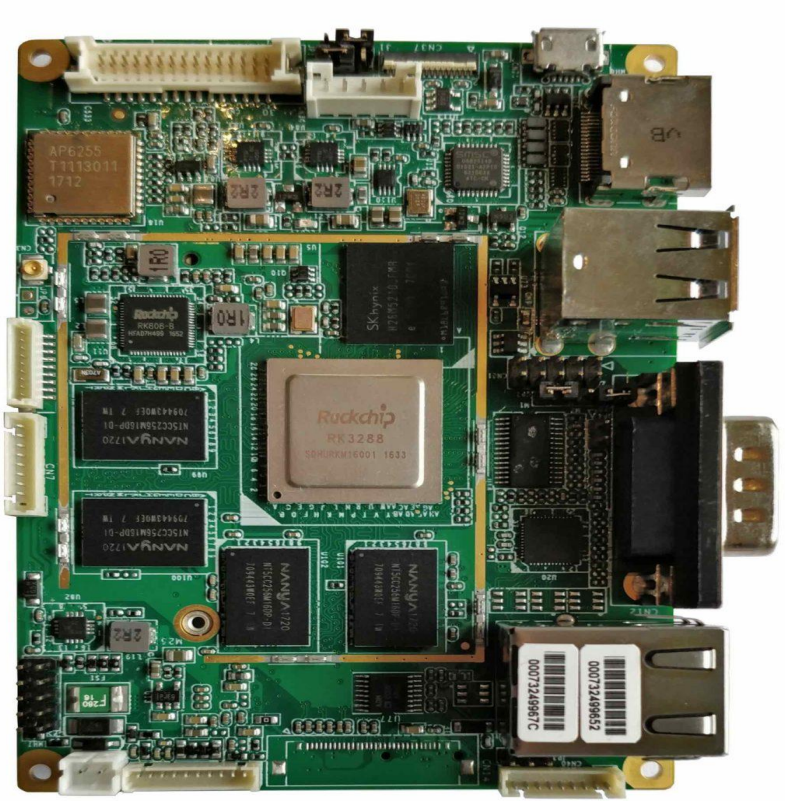
- ➔ Prerequisite
- ➔ Vyasa RK3288 port
- ➔ Demo run, BPI-M64

Prerequisite

- ➔ SRAM size restrictions
- ➔ DDR configuration and timings
- ➔ Start with Serial port, can be debuggable further

Vyasa RK3288 port

- 3 stage bootloader
- TPL
  - CPU, dram init, clocks, debug uart
  - SPL BOOTROM
- SPL
  - SPL_OF_CONTROL, UART, falcon
  - SPL MMC
- U-Boot proper
  - OF_CONTROL, UART, MMC, I2C,
  - CLK, Reset, commands etc

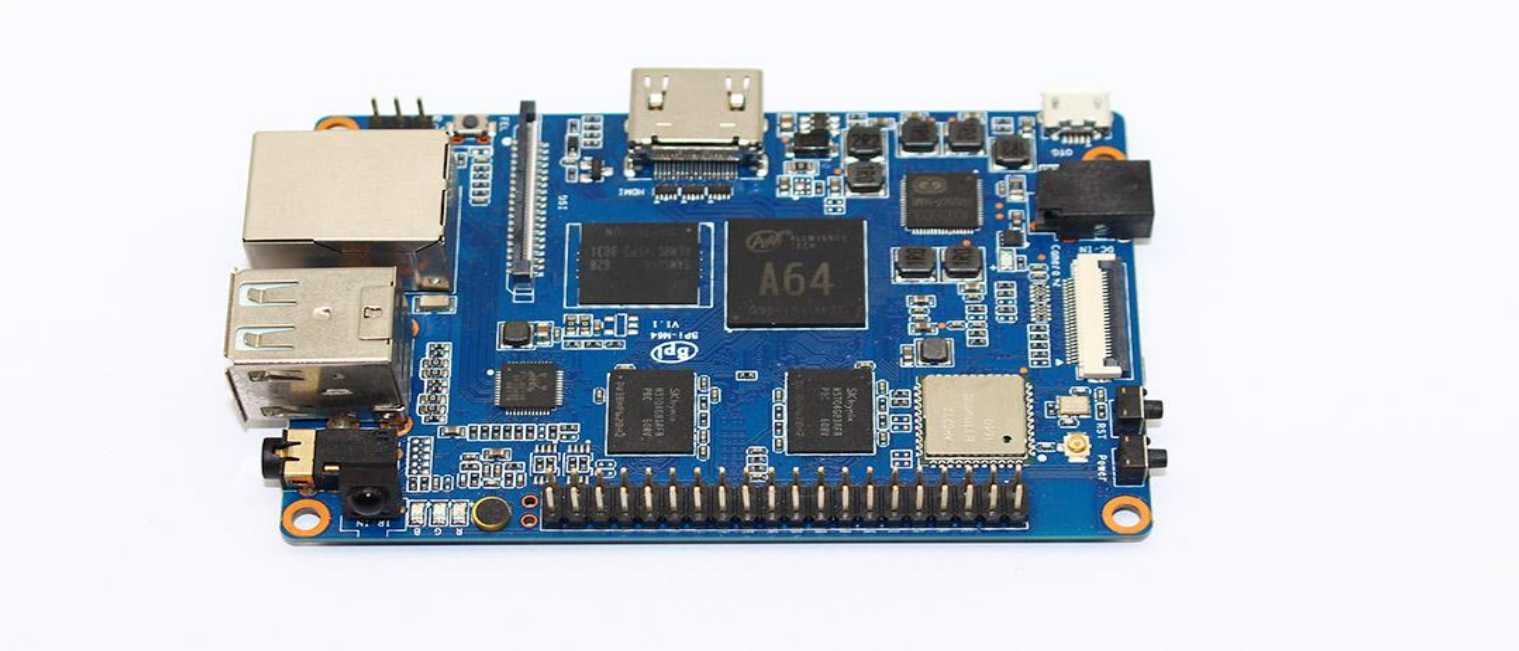


- Add SoC support
  - arch/arm/mach-rockchip
  - arch/arm/mach-rockchip/rk3288
  - CPU clock, syscon, linker script etc
  - Add TARGET_VYASA_RK3288 in arch/arm/mach-rockchip/rk3288/Kconfig
- Add DTS support
  - arch/arm/dts/rk3288-vyasa.dts
  - arch/arm/dts/rk3288-vyasa-u-boot.dtsi
- Add Board support
  - board/amarula/vyasa-rk3288/
  - Board specific code
  - Board specific SPL code, if require
  - board/amarula/vyasa-rk3288/MAINTAINERS
- Add header file
  - include/configs/vyasa-rk3288.h
  - Include common useful headers
  - Distro CONFIG_ definitions
  - CONFIG_ items which doesn’t support Kconfig yet
- Add defconfig file
  - CONFIG_ items which support Kconfig
- Finally run buildman or travis to make sure all build fine
- And use patman for sending patches to Mainline

```
U-Boot TPL 2018.09-00097-gd1e15041abf3 (Sep 13 2018 - 15:37:34 +0530)
Trying to boot from BOOTROM
Returning to boot ROM...

U-Boot SPL 2018.09-00097-gd1e15041abf3 (Sep 13 2018 - 15:37:34 +0530)
Trying to boot from MMC1
Expected Linux image is not found. Trying to start U-boot

U-Boot 2018.09-00097-gd1e15041abf3 (Sep 13 2018 - 15:37:34 +0530)

Model: Amarula Vyasa-RK3288
DRAM:  2 GiB
MMC:   dwmmc@ff0c0000: 1, dwmmc@ff0f0000: 0
Loading Environment from MMC... *** Warning - bad CRC, using default environment

In:   serial
Out:   serial
Err:   serial
Model: Amarula Vyasa-RK3288
Net:  eth0: ethernet@ff290000
Hit any key to stop autoboot:  0
switch to partitions #0, OK
=>
```

Demo run, BPI-M64

- ➔ Build ATF, U-Boot, Linux for BPI-M64
- ➔ Create verified image and run
- ➔ https://openedev.amarulasolutions.com/display/ODWIKI/Verified+Boot+on+SUNXI64



Future plan

- ➔ Kconfig migration
- ➔ Driver model migrations
  - ◆ Challenges for SPL, if DM enabled
- ➔ MTD driver model
- ➔ SPI NAND support, likely to merge v2018.11
- ➔ Allwinner CLK, RESET, Pinctrl Subsystems
- ➔ UCLASS_OPTEE

Conclusion

- ➔ Use DT and DM for new ports
- ➔ Hands on with DM conversion
- ➔ ML: u-boot@lists.denx.de
- ➔ IRC: #u-boot
- ➔ Lin: https://www.linkedin.com/in/jaganteki/
- ➔ Amarula Wiki: https://openedev.amarulasolutions.com

Questions??

Thank you

Jagan Teki <jagan@amarulasolutions.com>