Linux kernel

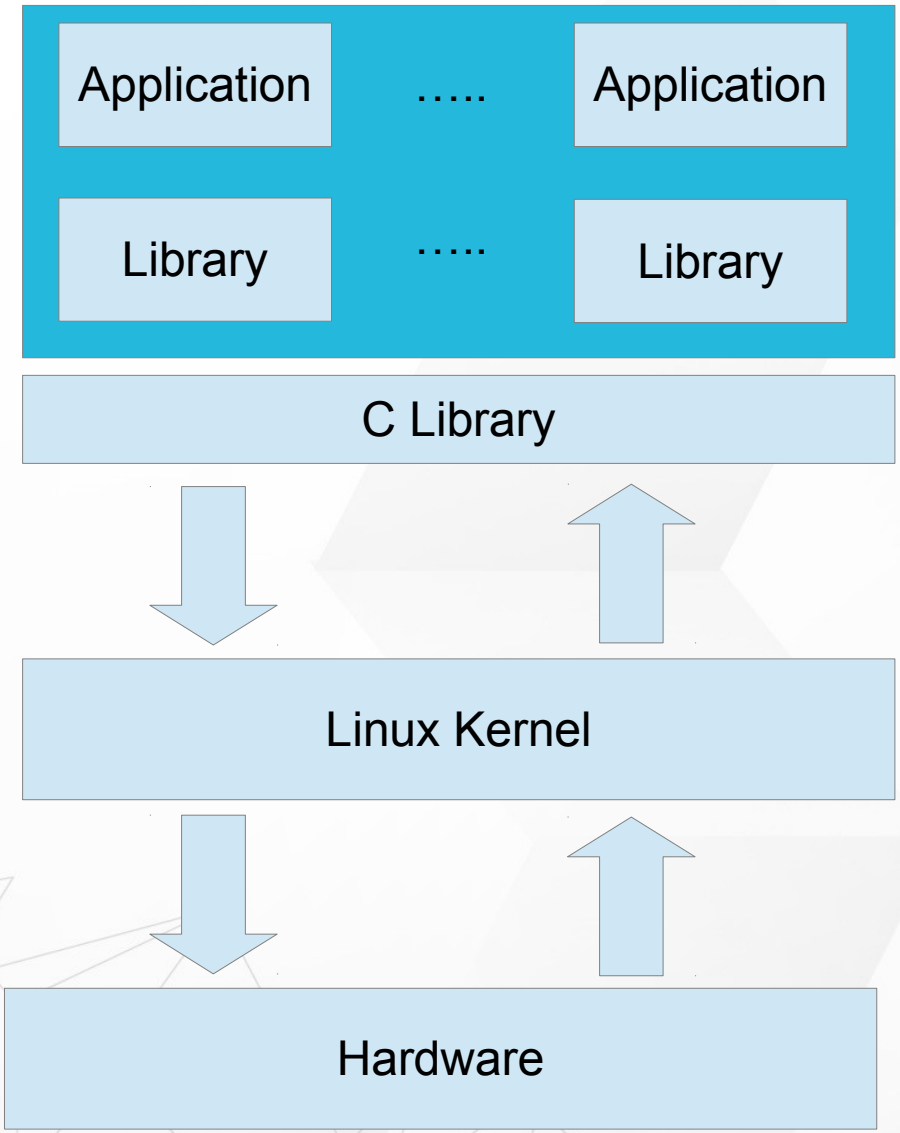
中華行動數位科技
Chinese Action Digital Technology

Linux kernel key features

- Portability and hardware support
- Scalability
- Compliance to standards and interoperability
- Exhaustive networking support
- Stability and reliability
- Modularity
- Easy to program.

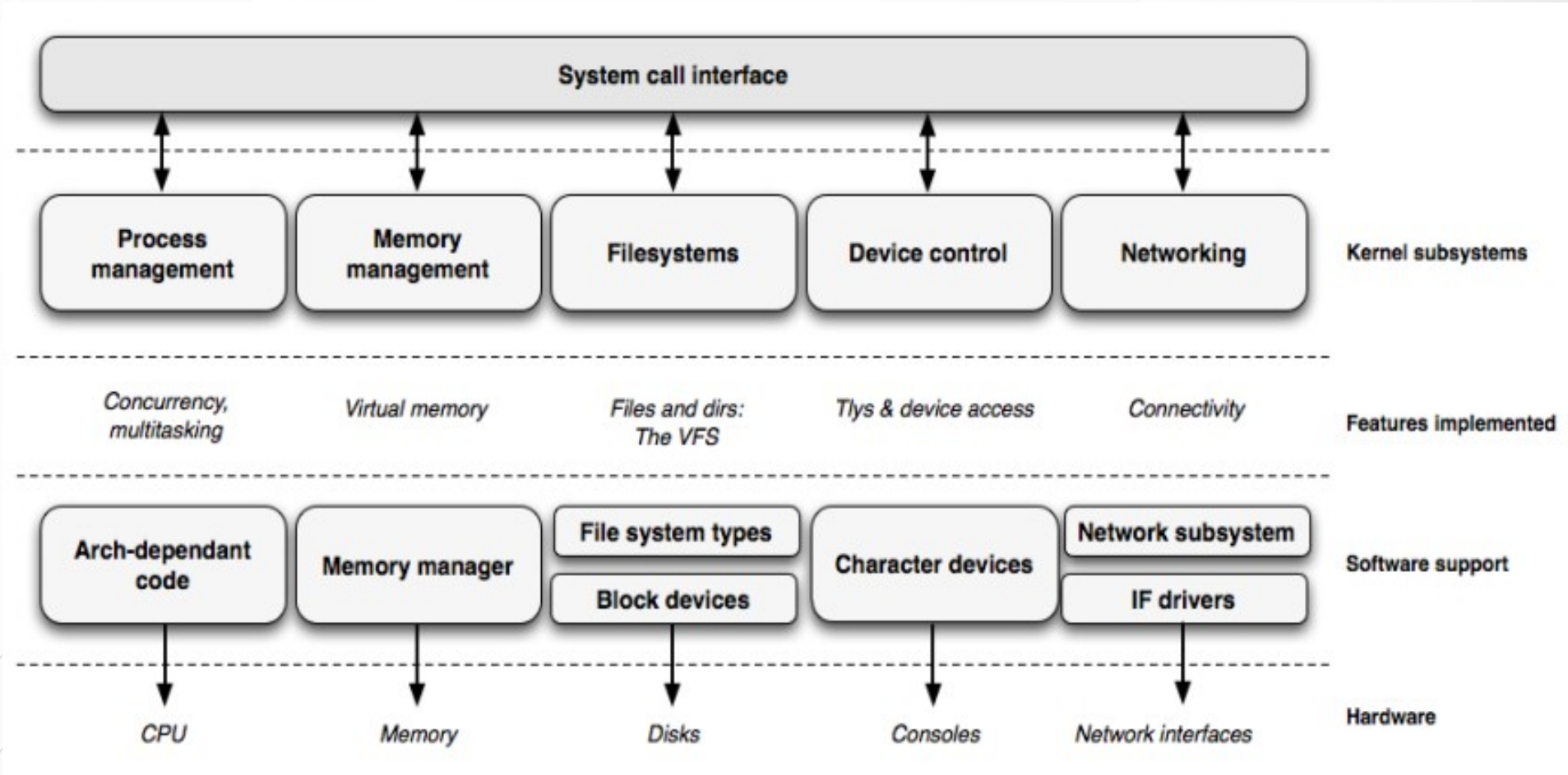
中華行動數位科技
Chinese Action Digital Technology

Linux kernel in the system



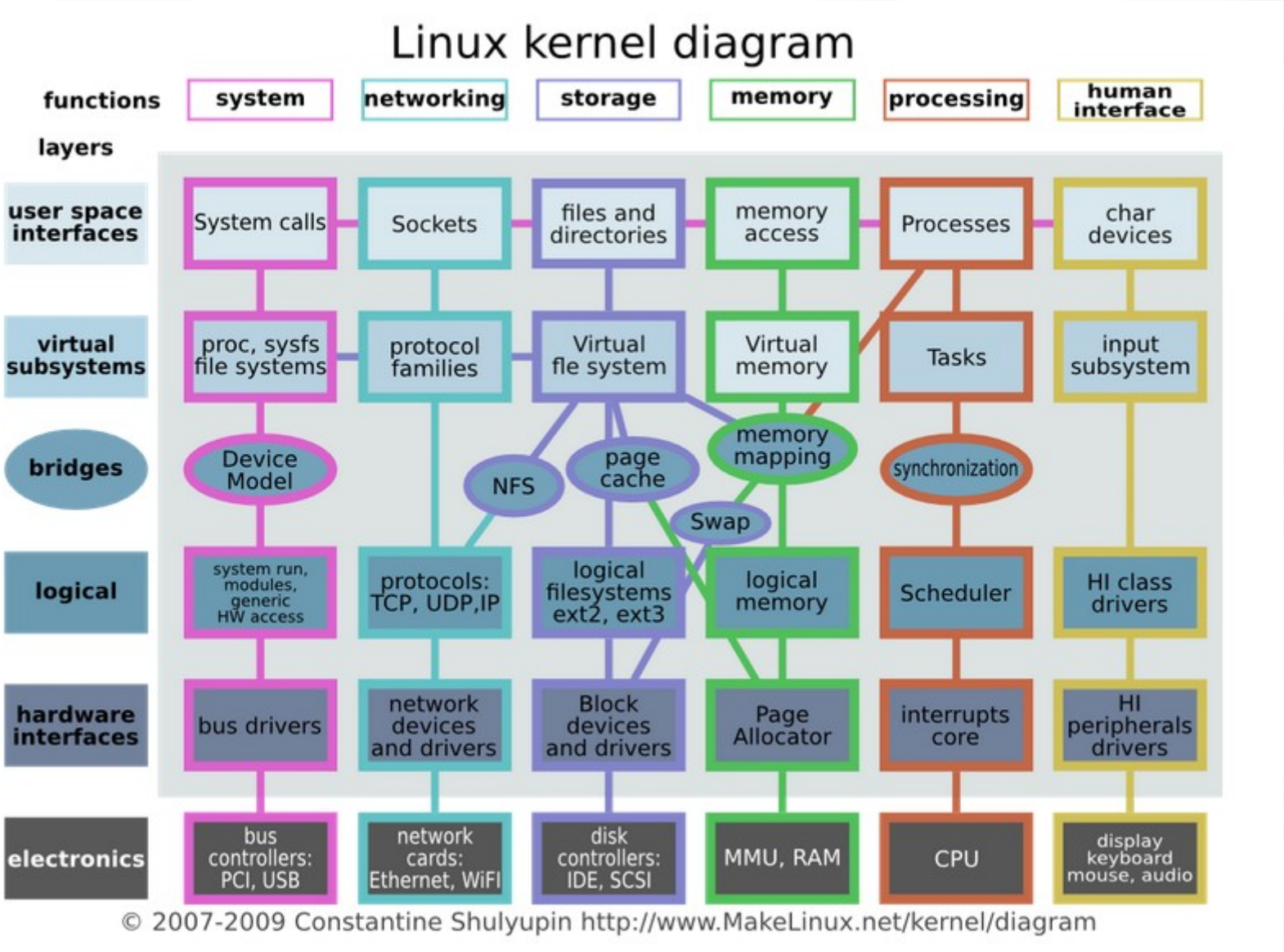
中華行動數位科技
Chinese Action Digital Technology

Linux kernel



中華行動數位科技
Chinese Action Digital Technology

Linux kernel



中華行動數位科技
Chinese Action Digital Technology

Kernel Source

- https://www.kernel.org/
- Many chip vendors
- kernel sub-communities
  - Architecture communities
    - ARM, MIPS, PowerPC ...
  - device drivers communities
    - I2C, SPI, USB, PCI, network ...

中華行動數位科技
Chinese Action Digital Technology

Programming language

- Implemented in C like all Unix systems
- A little Assembly is used too
- **No C++ used**
- **No C library**
- **No floating point** computation
- Kernel code has to supply its own library implementations
  - X : printf(), memset(), malloc(),...
  - O: printk(), memset(), kmalloc()

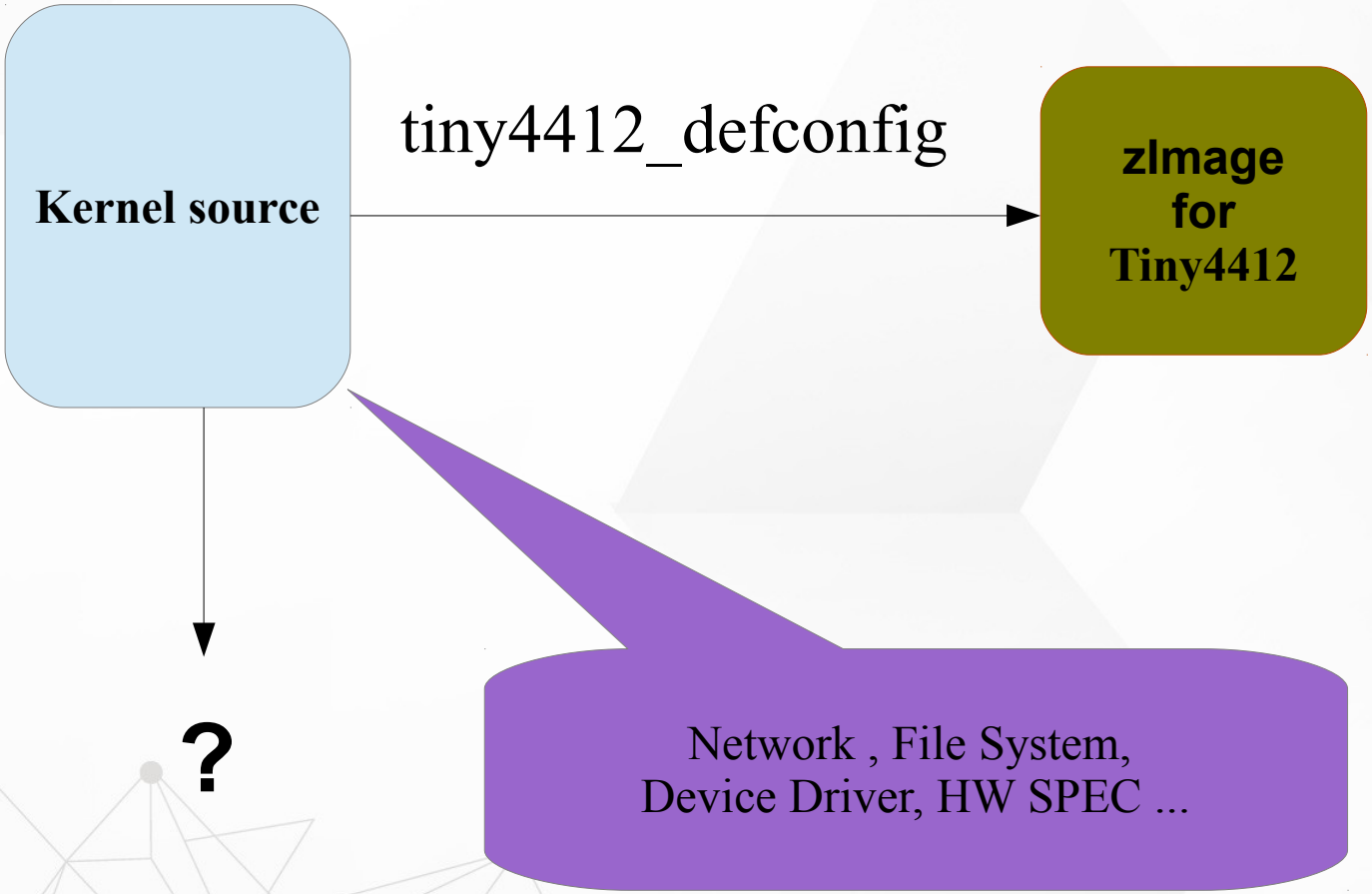
中華行動數位科技
Chinese Action Digital Technology

Linux sources structure

- arch/<ARCH>
- Architecture specific code
  - arch/<ARCH>/mach-<machine>
  - arch/<ARCH>/include/asm
  - arch/<ARCH>/boot/
- driver/
- Documentation/

中華行動數位科技
Chinese Action Digital Technology

Kernel configuration



中華行動數位科技
Chinese Action Digital Technology

Kernel configuration

- The kernel configuration and build system is based on multiple Makefiles
- The configuration is stored in the .config file at the root of kernel sources
- As options have dependencies, typically never edited by hand, but through graphical or text interfaces
  - make menuconfig → Text
  - make xconfig → graphical

中華行動數位科技
Chinese Action Digital Technology

Kernel configuration

```
slash@slash-huang:linux_3.5.0_tiny4412$ ls -la
total 574796
drwxr-xr-x  24 slash slash      4096 Sep  9 14:14 .
drwxrwxr-x   4 slash slash      4096 Sep  8 11:44 ..
drwxr-xr-x  29 slash slash      4096 Apr 22  2014 arch
drwxr-xr-x   3 slash slash      4096 Sep  9 14:06 block
-rwxrwxr-x   1 slash slash        37 Mar  4  2015 change_lib.sh
-rw-rw-r--   1 slash slash     83180 Sep  9 14:04 .config
-rw-r--r--   1 slash slash     18693 Apr 22  2014 COPYING
-rw-r--r--   1 slash slash     94956 Apr 22  2014 CREDITS
drwxr-xr-x   3 slash slash      4096 Sep  9 14:12 crypto
```

.config

$KERNEL_PATH/arch/arm/configs/

tiny4412_defconfig

```
slash@slash-huang:linux_3.5.0_tiny4412$ ls arch/arm/configs/
acs5k_defconfig             cm_x2xx_defconfig          g3evm_defconfig          lpd270_defconfig        pcm027_defconfig          spear13xx_defconfig
acs5k_tiny_defconfig        cm_x300_defconfig          g4evm_defconfig          lubbock_defconfig       pleb_defconfig            spear3xx_defconfig
afeb9260_defconfig          cns3420vb_defconfig        h3600_defconfig          mackerel_defconfig      pnx4008_defconfig         spear6xx_defconfig
ag5evm_defconfig            colibri_pxa270_defconfig   h5000_defconfig          magician_defconfig      prima2_defconfig          spitz_defconfig
am200epdkit_defconfig       colibri_pxa300_defconfig   h7201_defconfig          mainstone_defconfig     pxa168_defconfig          stamp9g20_defconfig
ap4evb_defconfig            collie_defconfig           h7202_defconfig          marzen_defconfig        pxa255-idp_defconfig      tct_hammer_defconfig
armadillo800eva_defconfig   corgi_defconfig            hackkit_defconfig        mini2440_defconfig      pxa3xx_defconfig          tegra_defconfig
assabet_defconfig           cpu9260_defconfig          imote2_defconfig         mmp2_defconfig          pxa910_defconfig          tiny4412_android_defconfig
at91_dt_defconfig           cpu9g20_defconfig          imx_v4_v5_defconfig      msm_defconfig           qil-a9260_defconfig       tiny4412_linux_defconfig
at91rm9200_defconfig        da8xx_omapl_defconfig      imx_v6_v7_defconfig      mv78xx0_defconfig       raumfeld_defconfig        tiny4412_ubuntu_defconfig
at91sam9260_defconfig       davinci_all_defconfig      integrator_defconfig     mxs_defconfig           realview_defconfig        trizeps4_defconfig
at91sam9261_defconfig       dove_defconfig             iop13xx_defconfig        neponset_defconfig      realview-smp_defconfig    u300_defconfig
at91sam9263_defconfig       ebsa110_defconfig          iop32x_defconfig         netwinder_defconfig     rpc_defconfig             u8500_defconfig
at91sam9g20_defconfig       edb7211_defconfig          iop33x_defconfig         netx_defconfig          s3c2410_defconfig         usb-a9260_defconfig
at91sam9g45_defconfig       em_x270_defconfig          ixp4xx_defconfig         nhk8815_defconfig       s3c6400_defconfig         versatile_defconfig
at91sam9rl_defconfig        ep93xx_defconfig           jornada720_defconfig     nuc910_defconfig        s5p64x0_defconfig         vexpress_defconfig
at91x40_defconfig           eseries_pxa_defconfig      kirkwood_defconfig       nuc950_defconfig        s5pc100_defconfig         viper_defconfig
badge4_defconfig            exynos4_android_defconfig  kota2_defconfig          nuc960_defconfig        s5pv210_defconfig         xcep_defconfig
bcmring_defconfig           exynos4_defconfig          ks8695_defconfig         omap1_defconfig         sam9_l9260_defconfig      zeus_defconfig
bonito_defconfig            ezx_defconfig              kzm9g_defconfig          omap2plus_defconfig     shannon_defconfig
cam60_defconfig             footbridge_defconfig       lart_defconfig           orion5x_defconfig       shark_defconfig
cerfcube_defconfig          fortunet_defconfig         lpc32xx_defconfig        palmz72_defconfig       simpad_defconfig
slash@slash-huang:linux_3.5.0_tiny4412$
```

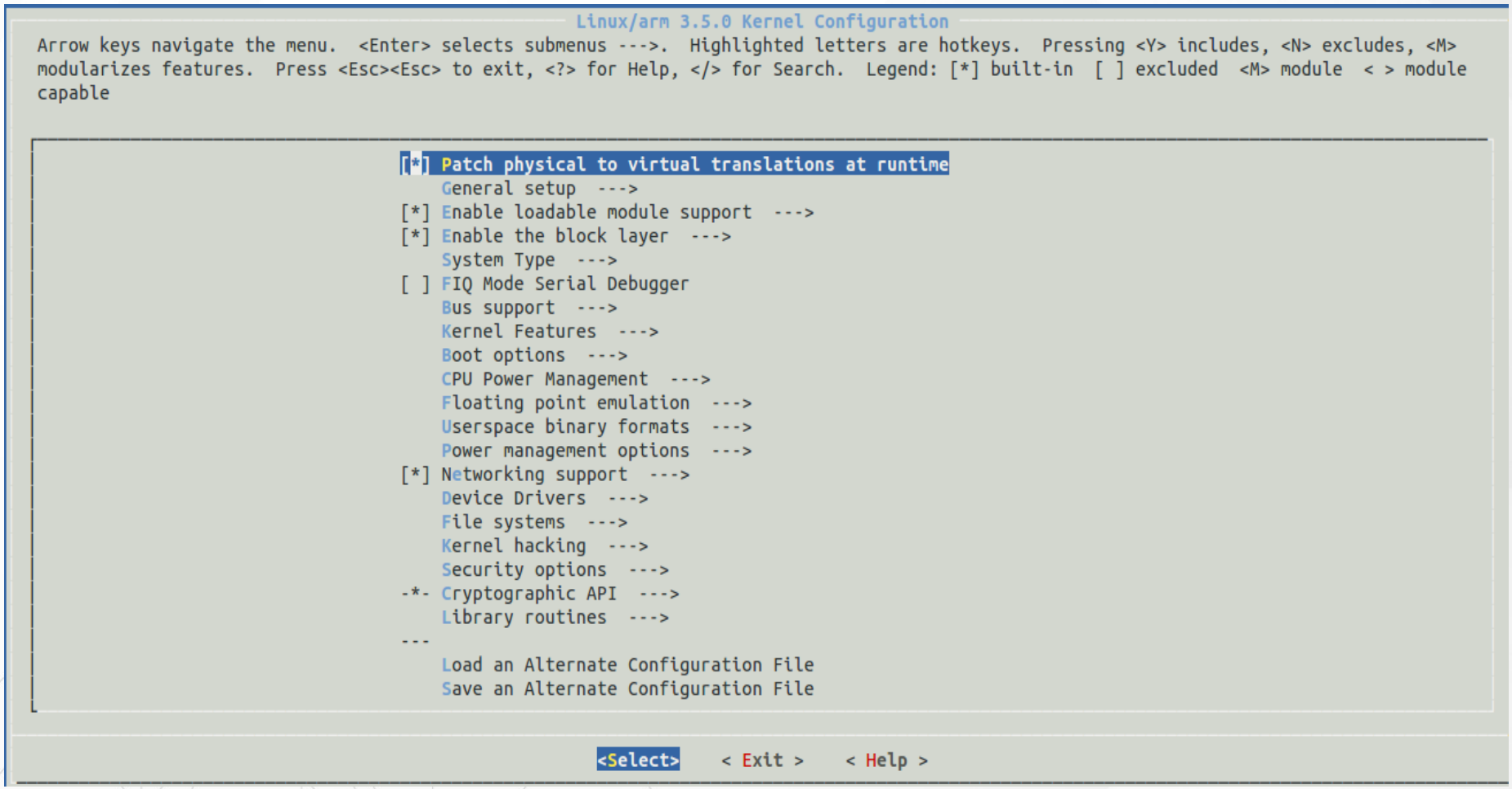

中華行動數位科技
Chinese Action Digital Technology

Kernel or module?

- The kernel image is a single file, resulting from the linking of all object files that correspond to features enabled in the configuration
- Some features (device drivers, filesystems, etc.) can however be compiled as modules

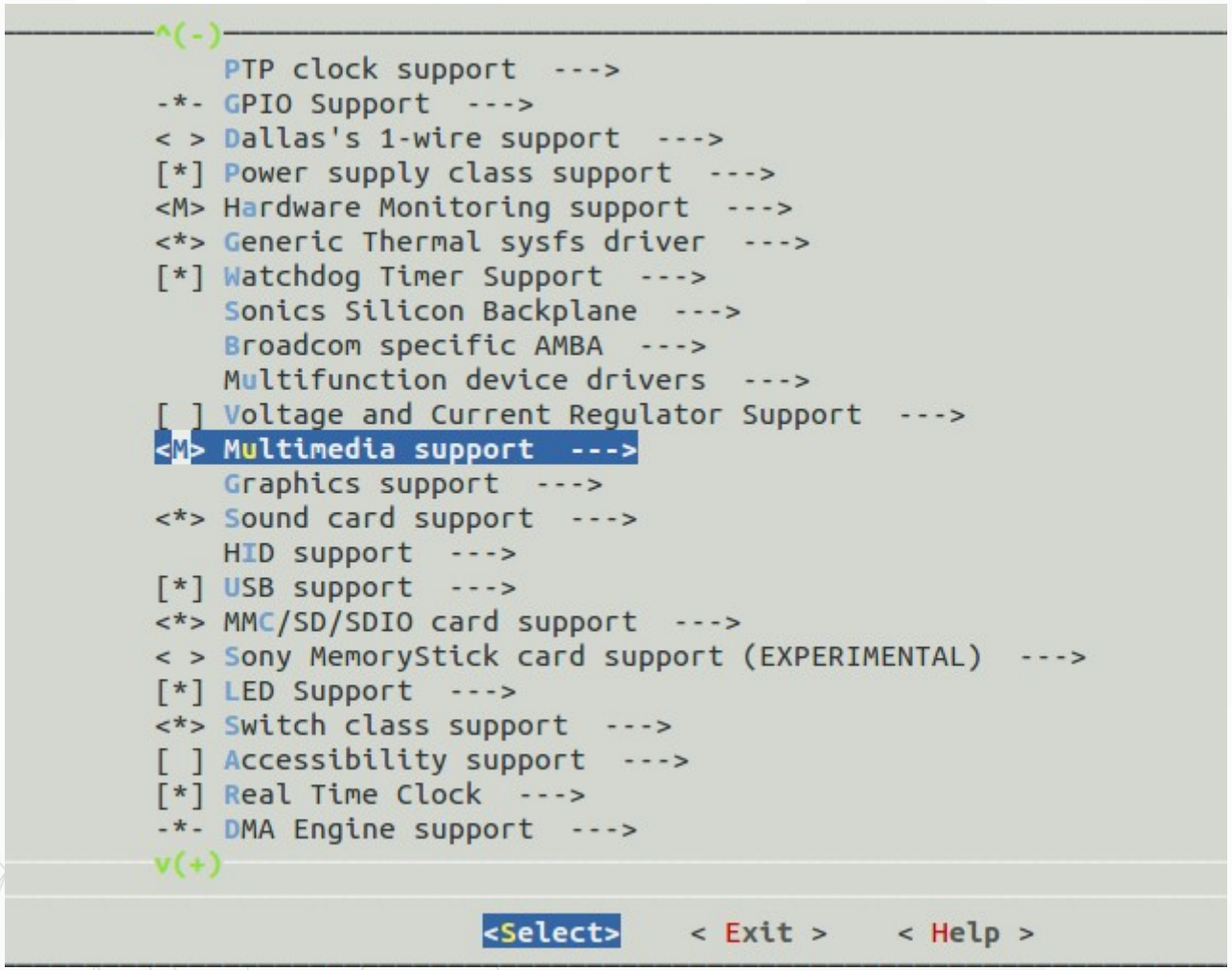
中華行動數位科技
Chinese Action Digital Technology

menuconfig



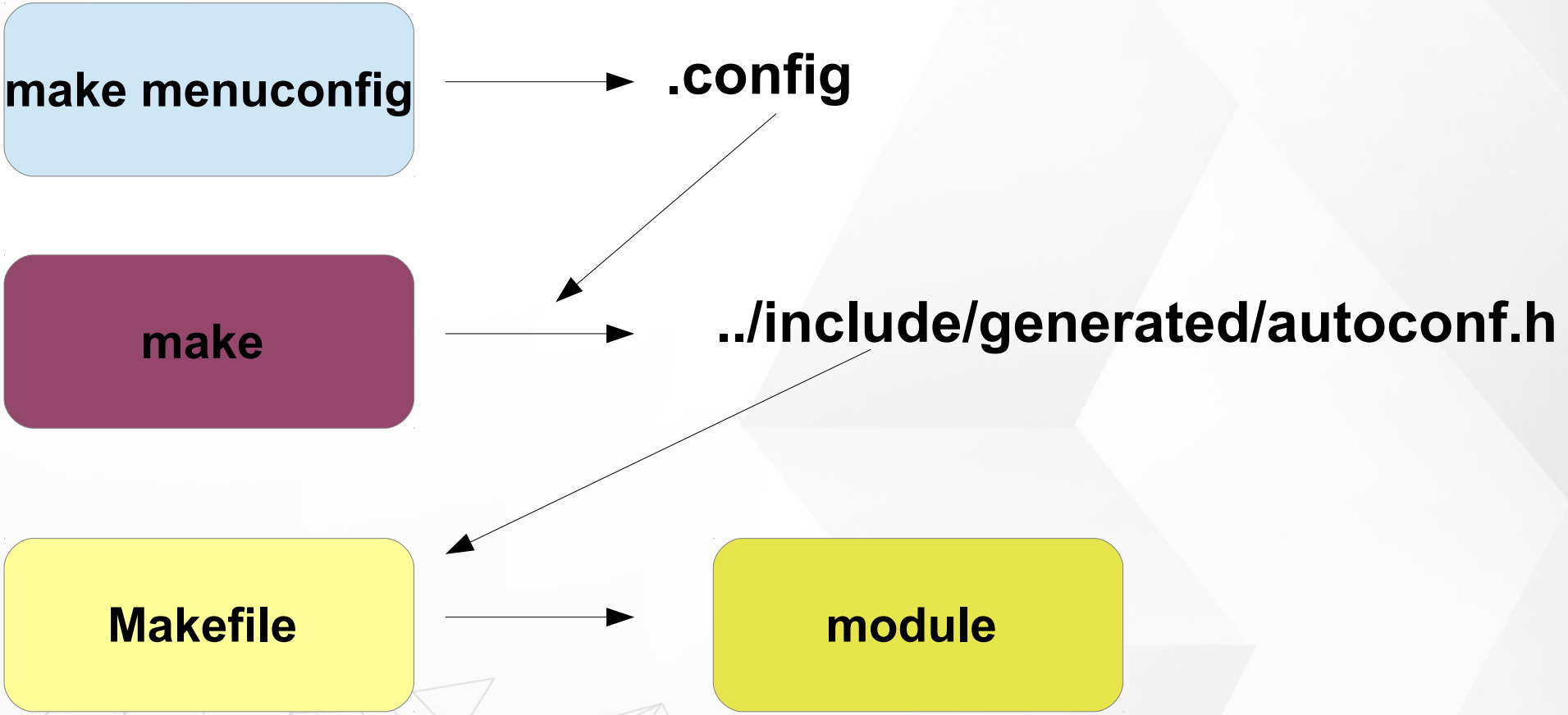
中華行動數位科技
Chinese Action Digital Technology

Kernel configuration options



中華行動數位科技
Chinese Action Digital Technology

Configuration



中華行動數位科技
Chinese Action Digital Technology

Corresponding .config file excerpt

```
ONFIG_BLK_DEV_SD=y
 CONFIG_CHR_DEV_ST is not set
 CONFIG_CHR_DEV_OSST is not set
 CONFIG_BLK_DEV_SR is not set
ONFIG_CHR_DEV_SG=y
 CONFIG_CHR_DEV_SCH is not set
ONFIG_SCSI_MULTI_LUN=y
 CONFIG_SCSI_CONSTANTS is not set
 CONFIG_SCSI_LOGGING is not set
 CONFIG_SCSI_SCAN_ASYNC is not set
ONFIG_SCSI_WAIT_SCAN=m

 SCSI Transports

 CONFIG_SCSI_SPI_ATTRS is not set
 CONFIG_SCSI_FC_ATTRS is not set
 CONFIG_SCSI_ISCSI_ATTRS is not set
 CONFIG_SCSI_SAS_ATTRS is not set
 CONFIG_SCSI_SAS_LIBSAS is not set
 CONFIG_SCSI_SRP_ATTRS is not set
ONFIG_SCSI_LOWLEVEL=y
 CONFIG_ISCSI_TCP is not set
 CONFIG_ISCSI_BOOT_SYSFS is not set
```

```
# --- NOTE ORDERING HERE ---
# For kernel non-modular link, transport attributes need to
# be initialised before drivers
# ---------------------------
obj-$(CONFIG_SCSI_SPI_ATTRS)    += scsi_transport_spi.o
obj-$(CONFIG_SCSI_FC_ATTRS)     += scsi_transport_fc.o
obj-$(CONFIG_SCSI_ISCSI_ATTRS)  += scsi_transport_iscsi.o
obj-$(CONFIG_SCSI_SAS_ATTRS)    += scsi_transport_sas.o
obj-$(CONFIG_SCSI_SAS_LIBSAS)   += libsas/
obj-$(CONFIG_SCSI_SRP_ATTRS)    += scsi_transport_srp.o
obj-$(CONFIG_SCSI_DH)       += device_handler/
```

$(KERNEL_PATH)/drivers/scsi/Makefile

.config

中華行動數位科技
Chinese Action Digital Technology

Kernel cleanup targets

- **make clean**
  - Kernel cleanup targets
- **make mrproper**
  - Remove all generated files (.config)
- **make distclean**
  - Remove editor backup and patch reject files

中華行動數位科技
Chinese Action Digital Technology

Specifying cross-compilation

- make **ARCH=arm CROSS_COMPILE=arm-linux-** …
- export ARCH=arm

  export CROSS_COMPILE=arm-linux
- Add above setting to script
  - source set_arm_toolchain.sh

中華行動數位科技
Chinese Action Digital Technology

Predefined configuration files

- Default configuration files available, per board or per-CPU family
  - They are stored in arch/<arch>/configs/, and are just minimal .config files
- **make tiny4412_linux_defconfig**
- To create your own default configuration file
  - make savedefconfig, to create a minimal configuration file
  - mv defconfig arch/<arch>/configs/myown_defconfig

中華行動數位科技
Chinese Action Digital Technology

Kernel compilation

- make
  - To run multiple jobs in parallel if you have multiple CPU cores
    - **make -j4**
- Generates
  - arch/<arch>/boot/*Image
    - zImage for ARM,

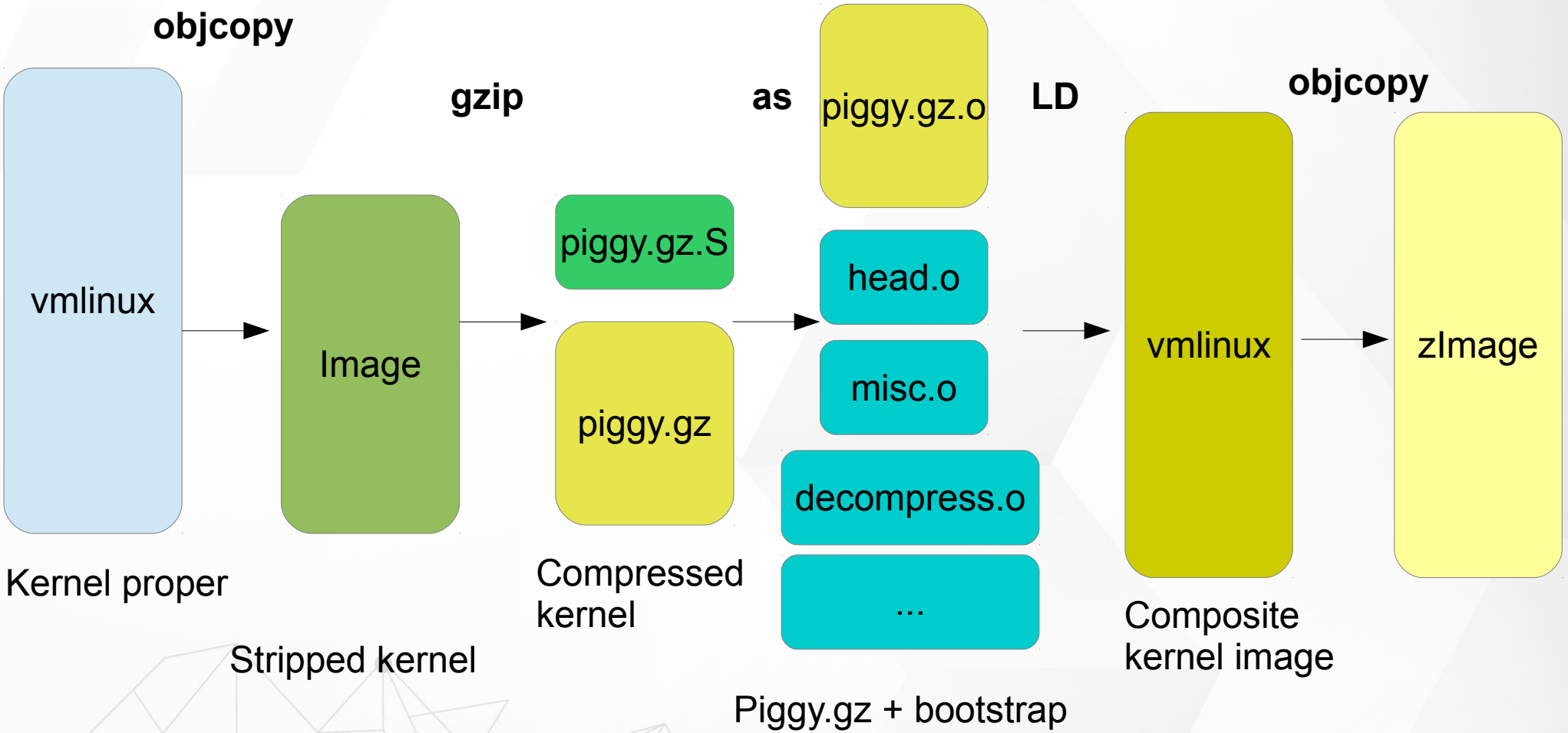
中華行動數位科技
Chinese Action Digital Technology

Make help

```
acs5k_defconfig            - Build for acs5k
acs5k_tiny_defconfig       - Build for acs5k_tiny
afeb9260_defconfig         - Build for afeb9260
ag5evm_defconfig           - Build for ag5evm
am200epdkit_defconfig      - Build for am200epdkit
ap4evb_defconfig           - Build for ap4evb
armadillo800eva_defconfig - Build for armadillo800eva
assabet_defconfig          - Build for assabet
at91_dt_defconfig          - Build for at91_dt
at91rm9200_defconfig       - Build for at91rm9200
at91sam9260_defconfig      - Build for at91sam9260
at91sam9261_defconfig      -
at91sam9263_defconfig      -
at91sam9g20_defconfig      -
at91sam9g45_defconfig      -
at91sam9rl_defconfig       -
at91x40_defconfig          -
badge4_defconfig           -
bcmring_defconfig          -
bonito_defconfig           -
cam60_defconfig            -
cerfcube_defconfig         -
cm_x2xx_defconfig          -
cm_x300_defconfig          -
cns3420vb_defconfig        -
colibri_pxa270_defconfig   -
colibri_pxa300_defconfig   -
collie_defconfig           -
corgi_defconfig            -
cpu9260_defconfig          - Build for cpu9260
cpu9g20_defconfig          - Build for cpu9g20
```

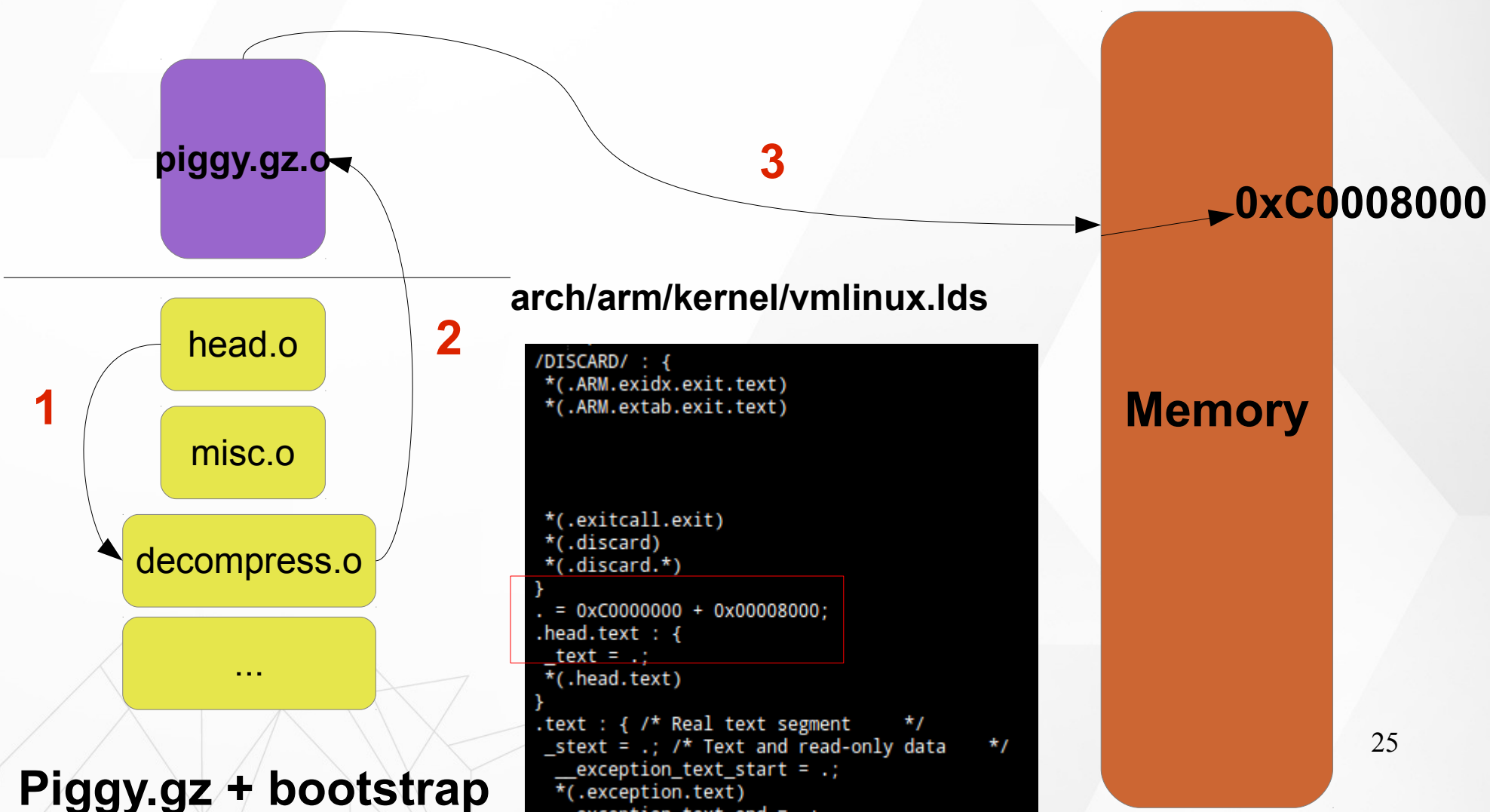
```
Other generic targets:
  all              - Build all targets marked with [*]
* vmlinux          - Build the bare kernel
* modules          - Build all modules
  modules_install  - Install all modules to INSTALL_MOD_PATH (default: /)
  firmware_install- Install all firmware to INSTALL_FW_PATH
                    (default: $(INSTALL_MOD_PATH)/lib/firmware)
  dir/             - Build all files in dir and below
  dir/file.[oisS]  - Build specified target only
  dir/file.lst     - Build specified mixed source/assembly target only
                    (requires a recent binutils and recent build (System.map))
  dir/file.ko      - Build module including final link
  modules_prepare  - Set up for building external modules
  tags/TAGS        - Generate tags file for editors
  cscope           - Generate cscope index
  gtags            - Generate GNU GLOBAL index
  kernelrelease    - Output the release version string
  kernelversion    - Output the version stored in Makefile
  headers_install  - Install sanitised kernel headers to INSTALL_HDR_PATH
                    (default: /home/xlloss/work/tiny-4412/build/linux_3.5.0_tiny4412/usr)
```

How to zImage



中華行動數位科技
Chinese Action Digital Technology

Kernel boot



中華行動數位科技
Chinese Action Digital Technology

Linux boot

$(linux)/arch/arm/boot/compressed/head.S

entering

```
/*
 * sort out different calling conventions
 */
        .align
        .arm                    @ Always enter in ARM state
start:
        .type   start,#function
        .rept   7
        mov r0, r0
        .endr
   ARM(     mov r0, r0      )
   ARM(     b   1f      )
 THUMB(     adr r12, BSYM(1f)   )
 THUMB(     bx  r12     )

        .word   0x016f2818      @ Magic numbers to help the loader
        .word   start           @ absolute load/run zImage address
        .word   _edata          @ zImage end address
 THUMB(     .thumb          )
1:      mov r7, r1          @ save architecture ID
        mov r8, r2          @ save atags pointer

#ifndef __ARM_ARCH_2__
        /*
         * Booting from Angel - need to enter SVC mode and disable
         * FIQs/IRQs (numeric definitions from angel arm.h source).
         * We only do this if we were in user mode on entry.
         */
```


中華行動數位科技
Chinese Action Digital Technology

Check overwrite

```
/*
 * Check to see if we will overwrite ourselves.
 *   r4  = final kernel address
 *   r9  = size of decompressed image
 *   r10 = end of this image, including  bss/stack/malloc space if non XIP
 * We basically want:
 *   r4 - 16k page directory >= r10 -> OK
 *   r4 + image length <= address of wont_overwrite -> OK
 */
        add r10, r10, #16384
        cmp r4, r10
        bhs wont_overwrite
        add r10, r4, r9
        adr r9, wont_overwrite
        cmp r10, r9
        bls wont_overwrite

/*
```


中華行動數位科技
Chinese Action Digital Technology

Relocate

```
/*
 * Relocate ourselves past the end of the decompressed kernel.
 *   r6  = _edata
 *   r10 = end of the decompressed kernel
 * Because we always copy ahead, we need to do it from the end and go
 * backward in case the source and destination overlap.
 */
        /*
         * Bump to the next 256-byte boundary with the size of
         * the relocation code added. This avoids overwriting
         * ourself when the offset is small.
         */
        add r10, r10, #((reloc_code_end - restart + 256) & ~255)
        bic r10, r10, #255

        /* Get start of code we want to copy and align it down. */
        adr r5, restart
        bic r5, r5, #31

        sub r9, r6, r5      @ size to copy
        add r9, r9, #31     @ rounded up to a multiple
        bic r9, r9, #31     @ ... of 32 bytes
        add r6, r9, r5
        add r9, r9, r10

1:      ldmdb   r6!, {r0 - r3, r10 - r12, lr}
        cmp r6, r5
```


中華行動數位科技
Chinese Action Digital Technology

Decompress kernel

```
/*
 * The C runtime environment should now be setup sufficiently.
 * Set up some pointers, and start decompressing.
 *   r4  = kernel execution address
 *   r7  = architecture ID
 *   r8  = atags pointer
 */
        mov r0, r4
        mov r1, sp              @ malloc space above stack
        add r2, sp, #0x10000    @ 64k max
        mov r3, r7
        bl  decompress_kernel
        bl  cache_clean_flush
        bl  cache_off
        mov r0, #0              @ must be zero
        mov r1, r7              @ restore architecture number
        mov r2, r8              @ restore atags pointer
 ARM(       mov pc, r4  )       @ call kernel
 THUMB(     bx  r4  )       @ entry point is always ARM

        .align  2
        .type   LC0, #object
LC0:        .word   LC0         @ r1
        .word   __bss_start     @ r2
        .word   _end            @ r3
        .word   _edata          @ r6
        .word   input_data_end - 4  @ r10 (inflated size location)
        .word    got_start      @ r11
```


中華行動數位科技
Chinese Action Digital Technology

Kernel startup entry point

- MMU = off, D-cache = off, I-cache = dont care
- R0 = 0, R1 = machine ID, R2 = atags or dtb pointer
- Machine ID
  - linux/arch/arm/tools/mach-types
- kernel at 0xc0008000

中華行動數位科技
Chinese Action Digital Technology

Kernel Startup Entry Point

```
/*
 * Kernel startup entry point.
 * ---------------------------
 *
 * This is normally called from the decompressor code.  The requirements
 * are: MMU = off, D-cache = off, I-cache = dont care, r0 = 0,
 * r1 = machine nr, r2 = atags or dtb pointer.
 *
 * This code is mostly position independent, so if you link the kernel at
 * 0xc0008000, you call this at __pa(0xc0008000).
 *
 * See linux/arch/arm/tools/mach-types for the complete list of machine
 * numbers for r1.
 *
 * We're trying to keep crap to a minimum; DO NOT add any machine specific
 * crap here - that's what the boot loader (or in extreme, well justified
 * circumstances, zImage) is for.
 */
   .arm

   __HEAD
ENTRY(stext)

 THUMB( adr r9, BSYM(1f)    )   @ Kernel is always entered in ARM.
 THUMB( bx  r9      )   @ If this is a Thumb-2 kernel,
 THUMB( .thumb          )   @ switch to Thumb now.
 THUMB(1:           )

   setmode PSR_F_BIT | PSR_I_BIT | SVC_MODE, r9 @ ensure svc mode
                        @ and irqs disabled
   mrc p15, 0, r9, c0, c0      @ get processor id
   bl  __lookup_processor_type     @ r5=procinfo r9=cpuid
   movs    r10, r5             @ invalid processor (r5=0)?
 THUMB( it  eq )        @ force fixup-able long branch encoding
```


中華行動數位科技
Chinese Action Digital Technology

Machine Init

$linux/arch/arm/mach-exynos/mach-tiny4412.c

```
        else
            initialize_non_prime_clocks();
    }
#ifdef CONFIG_BUSFREQ_OPP
    dev_add(&busfreq, &exynos4_busfreq.dev);
    ppmu_init(&exynos_ppmu[PPMU_DMC0], &exynos4_busfreq.dev);
    ppmu_init(&exynos_ppmu[PPMU_DMC1], &exynos4_busfreq.dev);
    ppmu_init(&exynos_ppmu[PPMU_CPU], &exynos4_busfreq.dev);
#endif
    set_tmu_platdata();
}

MACHINE_START(TINY4412, "TINY4412")
    /* Maintainer: FriendlyARM (www.arm9.net) */
    /* Maintainer: Kukjin Kim <kgene.kim@samsung.com> */
    /* Maintainer: Changhwan Youn <chaos.youn@samsung.com> */
    .atag_offset    = 0x100,
    .init_irq   = exynos4_init_irq,
    .map_io     = smdk4x12_map_io,
    .handle_irq = gic_handle_irq,
    .init_machine   = smdk4x12_machine_init,
    .init_late  = exynos_init_late,
    .timer      = &exynos4_timer,
    .restart    = exynos4_restart,
    .reserve    = &smdk4x12_reserve,
MACHINE_END
```


中華行動數位科技
Chinese Action Digital Technology

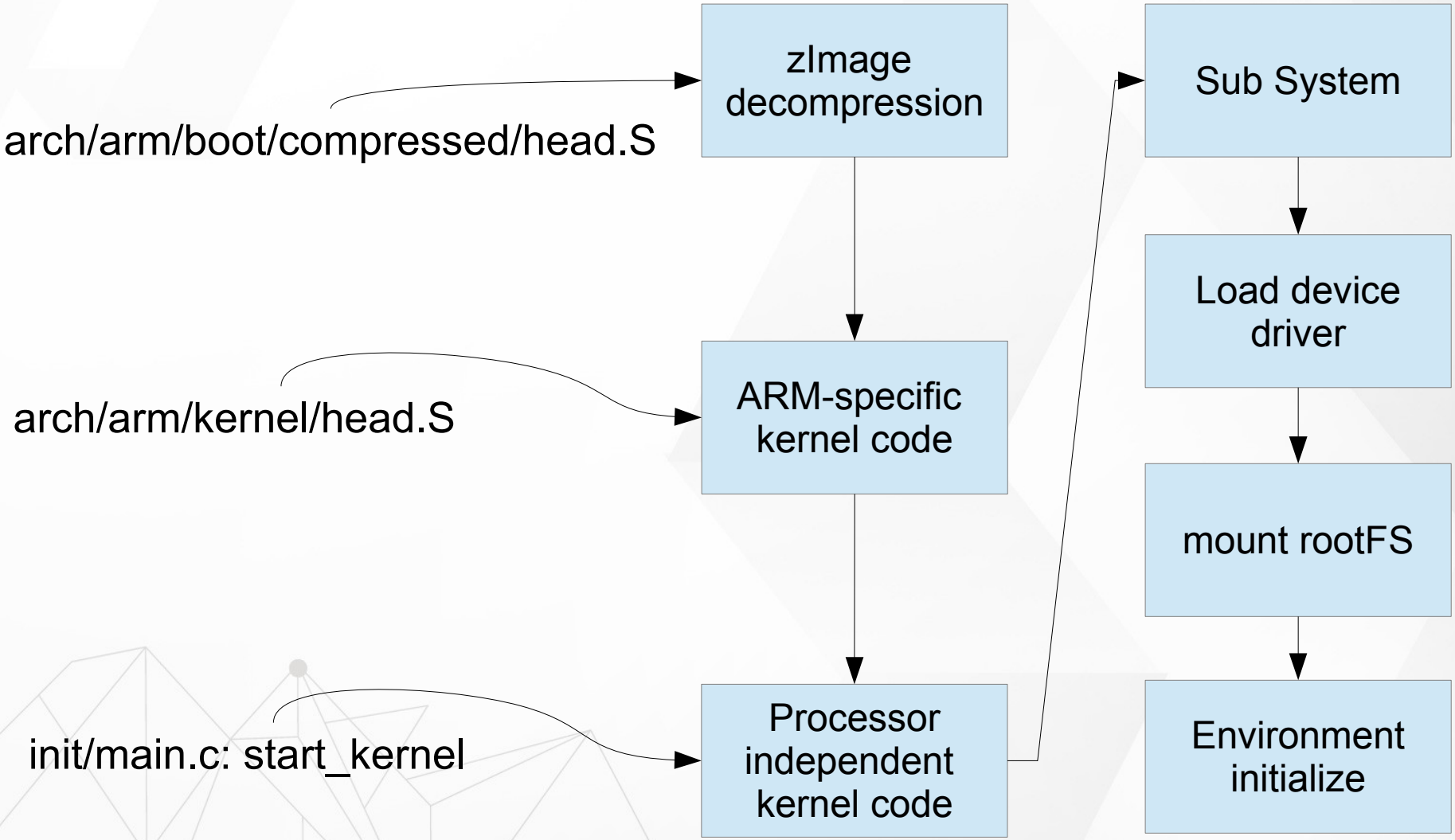
Mach Type

$linux/arch/arm/tools/mach-types

```
vybrid_vf4xx        MACH_VYBRID_VF4XX   VYBRID_VF4XX        4148
aria_g25        MACH_ARIA_G25       ARIA_G25        4149
bcm21553        MACH_BCM21553       BCM21553        4150
smdk5410        MACH_SMDK5410       SMDK5410        4151
lpc18xx         MACH_LPC18XX        LPC18XX         4152
oratisparty     MACH_ORATISPARTY    ORATISPARTY     4153
qseven          MACH_QSEVEN     QSEVEN          4154
gmv_generic     MACH_GMV_GENERIC    GMV_GENERIC     4155
th_link_eth     MACH_TH_LINK_ETH    TH_LINK_ETH     4156
tn_muninn       MACH_TN_MUNINN      TN_MUNINN       4157
rampage         MACH_RAMPAGE        RAMPAGE         4158
visstrim_mv10       MACH_VISSTRIM_MV10  VISSTRIM_MV10       4159
mx28_wilma      MACH_MX28_WILMA     MX28_WILMA      4164
msm8625_ffa     MACH_MSM8625_FFA    MSM8625_FFA     4166
vpu101          MACH_VPU101     VPU101          4167
baileys         MACH_BAILEYS        BAILEYS         4169
familybox       MACH_FAMILYBOX      FAMILYBOX       4170
ensemble_mx35       MACH_ENSEMBLE_MX35  ENSEMBLE_MX35       4171
sc_sps_1        MACH_SC_SPS_1       SC_SPS_1        4172
tiny4412        MACH_TINY4412       TINY4412        4608
```


中華行動數位科技
Chinese Action Digital Technology

Linux Kernel Booting



中華行動數位科技
Chinese Action Digital Technology

Write rootFS to eMMC

Step 1 – Creation of the actual EXT4.img

dd if=/dev/zero of=rootfs_ext4.img bs=512k count=60

Translation of the terms,

bs =blocksize,

count=60, the number of block`s, in our case will result an image of 30 Mb.

To get the exact size of the image that you create use simple maths.

60 * 512K = 31457280 byte = 30M bytes

Step 2 Formating the rootfs_ext4.img with EXT4

mkfs.ext4 rootfs_ext4.img

It will be a question where you will select yes (Y)

Step 3 mount the directories that we previous created.

mkdir rootfs_ext4 && mount -o loop rootfs_ext4.img rootfs_ext4/

中華行動數位科技
Chinese Action Digital Technology

Write rootFS to eMMC

Step 4 copy the content from the old system.img in the system_new.img

cp -v -r -p /home/cadtc/tiny4412/experiment/root_mkfs/* ./rootfs_ext4

Step 5 sync the files

sync

Step 6 Unmounting the partitons.

umount rootfs_ext4/

Step 7 reboot EVB and into Linux console with NFS

Step 8 dd rootfs_ext4.img to /dev/mmcblk0p2

dd if=/tmp/rootfs_ext4.img of=/dev/mmcblk0p2

Step 9 Modify u-boot args then reboot

setenv bootargs 'noinitrd init=/linuxrc root=/dev/mmcblk0p2 rw noinitrd rootfstype=ext4 console=ttySAC0 lcd=S70'

中華行動數位科技
Chinese Action Digital Technology