Viaggio nel kernel Linux

Concetti di base, struttura, note per debug e sviluppo

Angelo Dureghello
angelo@mime.email

Domenica 25 Ottobre 2020

Angelo Dureghello

- grande passione per l'opensource, elettronica e sistemi embedded
- attivo come programmatore su sistemi embedded dal 2001
- mainline Linux kernel contributor, 31 patch (author), 62 contributi totali (log msg), 3 driver completi, 1 driver-maintainer flag.
- m68k/ColdFire U-boot custodian
- progettista di alcune schede linux-embedded (amcore, stmark2, codice in kernel mainline)
- speaker a ELCE 2016, FOSDEM 2020, UNITS, linux-day cittadini
- lavora per Timesys Corporation, vive e lavora a Trieste, Italy

INTRODUZIONE

Cos'è il kernel Linux ?

Viaggio nel kernel Linux

- e' il cuore del sistema operativo GNU/Linux
- pensato e creato nel 1991 da Linus Torvalds
- e' un file binario
  - puro (XIP),
  - ELF (vmlinux, Image, linux.bin, utile per debug)
  - compressso, si autoscompatta (zImage, bzImage, vmlinmuz)
  - uImage (embedded, with u-boot header/wrapper)
- kernel "monolitico"
  - con moduli built-in [*] o caricabili runtime [M] (/lib/modules)
- 24 Oct 2020, github riporta 96.6% C, 1.3% C++, 1.1% ASM, 0.3% Obj-C, 0.2% Makefiles, 0.2% Other

Viaggio nel kernel Linux

- il kernel ufficiale si trova in `http://www.kernel.org`
- licenza GPLv2, branching e riutilizzo commerciale e' consentito
- largamente usato nei sistemi embedded
- migliaia di drivers
- non vincolato alla parte userspace
- ampiamente configurabile (menuconfig)
- supportate scheduling classes, policies (default class CFS)
- supportati gruppi di processi e risorse (cgroups, namespaces)

Viaggio nel kernel Linux

- supporta molte architetture, ma, 2 mondi,
  - PC, workstations (x86_64)
  - embedded, schedine varie, nas, router, cellulari (arm, e molte altre architetture)
- su pc, risiede generalmente in /boot/vmlinuz-xxx (con relativo initramfs)
- all'avvio il bootloader lo carica in ram e lo esegue (jmp)
- ci sono eccezioni, come esecuzione XIP da memorie flash
- i moduli compilati correttamente (kernel headers corretti, versione) sono caricabili runtime (modprobe)
  - per PC (x86_64), molti drivers bus pci, usb etc sono abilitati di default
  - in embedded, si abilita solo il necessario (devicetree, drivers)

Viaggio nel kernel Linux

- supporta l'esecuzione di diversi formati binari, ELF, flat, zflat
- la comunicazione da userspace avviene in diversi modi
  - accesso a driver e funzionalita' da /dev,
    con open/read/write/ioctl, echo, cat, dd, hexdump ...
  - pseudofs: sysfs, procfs, configfs, relayfs, debugfs
  - molte altre system calls, di solito effettuate da glibc wrapper
    man syscalls
  - netlink socket, mmap, ...

Viaggio nel kernel Linux

Con i suoi problemi, come in tutti gli OS, ma ....

il kernel Linux

e' il cuore di uno dei sistemi operativi piu "portati" e utilizzati al mondo,

e' tra i progetti opensource piu ampi al mondo,
su cui moltissime aziende hanno costruito il loro profitto.

STRUTTURA

Uno sguardo all'interno ...

Viaggio nel kernel Linux

recuperare i sorgenti ufficiali (mainline)
(per lo sviluppo, con tutta la storia)
git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git

recuperare i sorgenti ufficiali di una certa versione (tarball)
wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.4.68.tar.xz
oppure
git clone –depth 1 –single-branch –branch v5.4.68 \
git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git

browse online
https://elixir.bootlin.com/linux/v5.9-rc8/source

Viaggio nel kernel Linux - Codice sorgente, root directory

-rw-r--r--	COPYING	
-rw-r--r--	CREDITS	
d---------	Documentation	documentazione ufficiale, sintetica ma
-rw-r--r--	Kbuild	piuttosto buona, in inglese
-rw-r--r--	Kconfig	
d---------	LICENSES	
-rw-r--r--	MAINTAINERS	maintainers responsabili di ogni ramo, mailing lists etc
-rw-r--r--	Makefile	
-rw-r--r--	README	
d---------	arch	codice di ogni specifica architettura
d---------	block	block layer support (CONFIG_BLOCK)
d---------	certs	certificati, chiavi, module-signing
d---------	crypto	algoritmi di crittografia (API)
d---------	drivers	drivers, e staging drivers
d---------	fs	filesystems
d---------	include	kernel headers
d---------	init	dove il kernel inzia, codice C, dopo init
d---------	ipc	inter-process communication
d---------	kernel	parte principale, scheduler, kernel/sched/fair.c
d---------	lib	librerie di funzioni
d---------	mm	gestione memoria
d---------	net	protocolli di rete, netlink, sockets, etc
d---------	samples	esempi codice kernelspace
d---------	scripts	scirpts di supporto allo sviluppo
d---------	security	selinux, securityfs (apparmor, etc)
d---------	sound	sound drivers alsa, codec, soc sound drivers
d---------	tools	strumenti di supporto allo sviluppo
d---------	usr	early-userspace, initramfs
d---------	virt	kvm

Viaggio nel kernel Linux - Architetture supportate, /arch

```
-rw-r--r--  .gitignore
-rw-r--r--  Kconfig
d---------  alpha
d---------  arc
d---------  arm
d---------  arm64
d---------  c6x
d---------  csky
d---------  h8300
d---------  hexagon
d---------  ia64
d---------  m68k
d---------  microblaze
d---------  mips
d---------  nds32
d---------  nios2
d---------  openrisc
d---------  parisc
d---------  powerpc
d---------  riscv
d---------  s390
d---------  sh
d---------  sparc
d---------  um
d---------  x86
d---------  xtensa
```

principalemnte desktop/ws/server

principalmente embedded

user-mode linux

Viaggio nel kernel Linux - Drivers, /drivers

Mode	Name	Size
-rw-r--r--	Kconfig	3913
-rw-r--r--	Makefile	5429
d---------	accessibility	139
d---------	acpi	3249
d---------	amba	143
d---------	android	367
d---------	ata	5072
d---------	atm	1527
d---------	auxdisplay	456
d---------	base	1434
d---------	bcma	989
d---------	block	1486
d---------	bluetooth	1647
d---------	bus	1223
d---------	cdrom	106
d---------	char	1566
d---------	clk	4327
d---------	clocksource	4117
d---------	connector	185
d---------	counter	369
d---------	cpufreq	4410
d---------	cpuidle	1419
d---------	crypto	2372
d---------	dax	348
d---------	dca	148
d---------	devfreq	713
d---------	dio	148
d---------	dma-buf	824
d---------	dma	3767
d---------	edac	3269

drivers, lista molto lunga,
mainline mostra 147 famiglie,
sono la forza del sistema operativo

altri sono inclusi in directory
"staging"

nel pc avviene il rilevamento dinamico
dei dispositivi tramite il concetto
di "bus" (/sys/bus)
(registrazione del driver sul bus)

funzione xxx_probe() e' eseguita se nel
bus specifico e' registrato il driver
corretto per quel dispositivo,

Su schede embedded invece si dichiarano
i dispositivi presenti tramite
platform bus (board.c, legacy) o
con il piu attuale "devicetree".

Viaggio nel kernel Linux - Startup, da /arch a /init

```
-rw-r--r--  COPYING
-rw-r--r--  CREDITS
d---------  Documentation
-rw-r--r--  Kbuild
-rw-r--r--  Kconfig
d---------  LICENSES
-rw-r--r--  MAINTAINERS
-rw-r--r--  Makefile
-rw-r--r--  README
d---------  arch /x86/kernel/head_64.S  __HEAD (entry point)
d---------  block          head64.c  x86_64_start_kernel()
d---------  certs
d---------  crypto
d---------  drivers
d---------  fs
d---------  include
d---------  init /main.c start_kernel()
d---------  ipc
d---------  kernel       tutti gli xxx_init()
d---------  lib          arch_call_rest_init() (funzione __weak)
d---------  mm           rest_init()
d---------  net          pid = kernel_thread(kernel_init, NULL CLONE_FS);
d---------  samples                          (pid 1)
d---------  scripts      ...
d---------  security     pid = kernel_thread(kthreadd,
d---------  sound                            NULL CLONE_FS | CLONE_FILES);
d---------  tools        ...
d---------  usr          cpu_startup_entry(CPUHP_ONLINE)
d---------  virt
```

Viaggio nel kernel Linux - Altre directory importanti

kernel

- sched/fair.c: CFS (Con Colivas, Ingo Molnar), from 2.6.23, default schedule class
- cgroups, implementazione gruppi e controllo risorse (cgroups, namespaces)
- bpf (Berkeley Packet Filter)
- rcu sincronizzazioni (consente contemporaneita' di un updater e reader multipli)

fs

- pseudo (sysfs, procfs, debugfs, etc)
- frequenti in embedded / memorie flash (cramfs, jffs2, ubi, squashfs, romfs, ext4, ...)
- desktop/server (ext2, ext4, btrfs, xfs, reiserfs, ntfs, network fs, ...)

mm

- directory storica, gestione memoria, pagine, zone, etc

Viaggio nel kernel Linux - Kconfig e Makefile

- quasi ogni directory contiene un file Kconfig e un file "Makefile"
- Kconfig contiene le opzioni di configurazione disponibili
- Makefile include nel processo di compilazione solo gli object file relativi a una certa opzione Kconfig
- la selezione tramite menuconfig si traducde in "m" o "y" (modulo caricabile o built-in)

Kconfig

```
config MMC_SDHCI_ESDHC_MCF
        tristate "SDHCI support for the Freescale eSDHC ColdFire controller"
        depends on M5441x
        depends on MMC_SDHCI_PLTFM
```

Makefile

```
obj-$(CONFIG_MMC_SDHCI_ESDHC_MCF)       += sdhci-esdhc-mcf.o
```

Viaggio nel kernel Linux - Coding style

```
1614  static void lpuart32_configure(struct lpuart_port *sport)
1615  {
1616          unsigned long temp;
1617
1618          if (sport->lpuart_dma_rx_use) {
1619                  /* RXWATER must be 0 */
1620                  temp = lpuart32_read(&sport->port, UARTWATER);
1621                  temp &= ~(UARTWATER_WATER_MASK << UARTWATER_RXWATER_OFF);
1622                  lpuart32_write(&sport->port, temp, UARTWATER);
1623          }
1624          temp = lpuart32_read(&sport->port, UARTCTRL);
1625          if (!sport->lpuart_dma_rx_use)
1626                  temp |= UARTCTRL_RIE;
1627          if (!sport->lpuart_dma_tx_use)
1628                  temp |= UARTCTRL_TIE;
1629          lpuart32_write(&sport->port, temp, UARTCTRL);
1630  }
1631
1632  static int lpuart32_startup(struct uart_port *port)
1633  {
1634          struct lpuart_port *sport = container_of(port, struct lpuart_port, port);
1635          unsigned long flags;
1636          unsigned long temp;
1637
1638          /* determine FIFO size */
1639          temp = lpuart32_read(&sport->port, UARTFIFO);
1640
1641          sport->txfifo_size = UARTFIFO_DEPTH((temp >> UARTFIFO_TXSIZE_OFF) &
1642                                              UARTFIFO_FIFOSIZE_MASK);
1643          sport->port.fifosize = sport->txfifo_size;
1644
1645          sport->rxfifo_size = UARTFIFO_DEPTH((temp >> UARTFIFO_RXSIZE_OFF) &
1646                                              UARTFIFO_FIFOSIZE_MASK);
1647
1648          /*
```

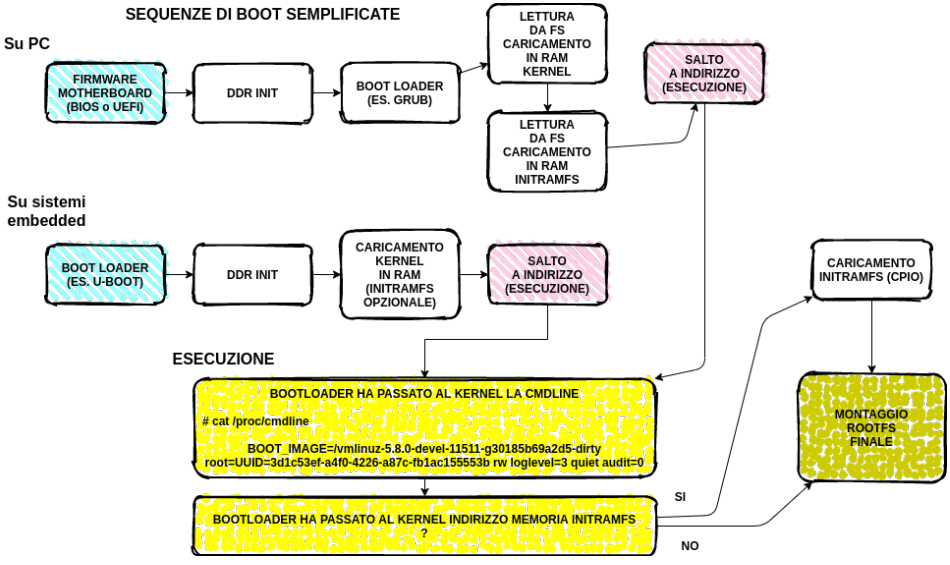
- tab a 8, tab puro
- parentesi graffe in linea
- spaziature eccetto chiamate e nomi funzione
- ... ed altro in coding-style.rst

Viaggio nel kernel Linux

- codice segue il Linux kernel coding-style, motivazioni di ogni scelta:
  `https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/process/coding-style.rst`
- qualsiasi modifica ai sorgenti (patch) deve superare:
  scripts/checkpatch.pl

Viaggio nel kernel Linux

Viaggio nel kernel Linux



COMPILARE IL KERNEL

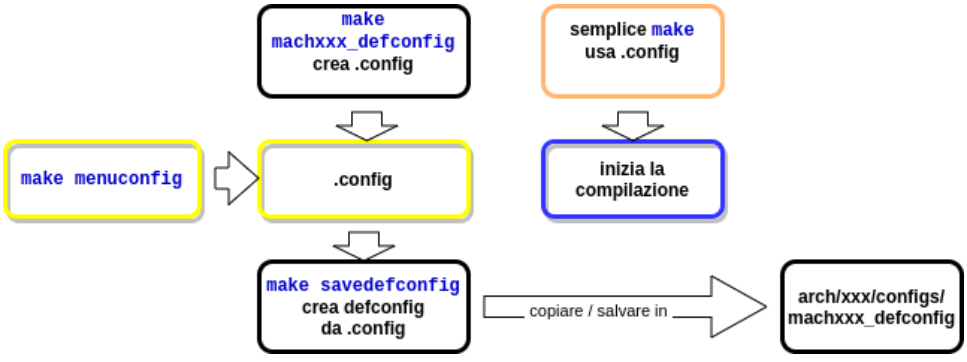
Modo classico

Viaggio nel kernel Linux - Preparazione

- strumenti necessari: o singoli pacchetti, gcc, gnu-make, ...
  (o, direttamente build-essential libncurses-dev bison flex libssl-dev libelf-dev)

- sorgenti in /usr/src/linux-xxx con link simbolico "linux"
  (ln -s /usr/src/linux-xxx /usr/src/linux)

Viaggio nel kernel Linux - Configurazione



Viaggio nel kernel Linux - Configurazione

Il file .config contiene tutte le configurazioni scelte
ed e' usato da gnu-make in fase di compilazione

- .config contiene tutte le configurazioni
- xxx_defconfig solo i settaggi diversi dai default in Kconfig
- se non esiste .config, e non si parte da un xxx_defconfig, .config viene generato dai default in Kconfig

```
# partire dalla configurazione corrente (in genere piu utilizzato)
make olddefconfig
# oppure, partendo da un defconfig salvato
make x86_64_defconfig # x86_64 defaults, oppure
make xxx_defconfig # config specifica salvata
# per ulteriori modifiche partendo dall attuale .config
make menuconfig
# salva configurazione in ./defconfig, che puo essere copiata in arch/xxx/config/xxx_defconfig
make savedefconfig
```

Viaggio nel kernel Linux - Compilazione

Modo classico, ci si puo creare uno script:

```
(opzionale, per cross-compilazione) export ARCH=m68k
(opzionale, per cross-compilazione) export CROSS_COMPILE=/path/toolchain
make clean
make -j8 KALLSYMS_EXTRA_PASS=1 LOCALVERSION=-custom
sudo make -j8 KALLSYMS_EXTRA_PASS=1 modules_install
sudo cp arch/x86_64/boot/bzImage /boot/vmlinuz-5.6.0-rc3-custom
.... attendi :) ...
# initramfs (comando specifico della distribuzione)
sudo mkinitcpio -g /boot/initramf-5.6.0-rc3-custom.img -k 5.6.0-rc3-custom
sync
# aggiornare grub (comando specifico della distribuzione)
sudo grub-mkconfig -o /boot/grub/grub.cfg
sync
```

Altrimenti, sempre possibile utilizzare guide specifiche propria distribuzione (distro-way)

OPERAZIONI UTILI

Debug di problemi ed altre cose utili

Viaggio nel kernel Linux

grep e' sempre il migliore amico

E' supportato un certo dispositivo, galcore GC2000 ? Esiste il driver ?

```
[linux.git] cd drivers                                                    wip/stmark2-jobs *
[drivers] grep -iR gc2000                                                 wip/stmark2-jobs *
gpu/drm/etnaviv/etnaviv_gpu.c:          if (etnaviv_is_model_rev(gpu, GC2000, 0x5108) ||
gpu/drm/etnaviv/etnaviv_gpu.c:      etnaviv_is_model_rev(gpu, GC2000, 0x5108) ||
gpu/drm/etnaviv/etnaviv_gpu.c:           * NXP likes to call the GPU on the i.MX6QP GC2000+, but in
gpu/drm/etnaviv/etnaviv_gpu.c:          if (etnaviv_is_model_rev(gpu, GC2000, 0xffff5450)) {
gpu/drm/etnaviv/etnaviv_gpu.c:      etnaviv_is_model_rev(gpu, GC2000, 0x5108))
gpu/drm/etnaviv/etnaviv_gpu.c:  /* GC2000 rev 5108 needs a special bus config */
gpu/drm/etnaviv/etnaviv_gpu.c:  if (etnaviv_is_model_rev(gpu, GC2000, 0x5108)) {
gpu/drm/etnaviv/common.xml.h:#define chipModel_GC2000                                  0x00002000
gpu/drm/etnaviv/etnaviv_perfmon.c:              gpu->identity.model == chipModel_GC2000 ||
gpu/drm/etnaviv/etnaviv_perfmon.c:              gpu->identity.model == chipModel_GC2000 ||
[drivers]                                                                 wip/stmark2-jobs *
```

Viaggio nel kernel Linux

Ci sono novita' nel filesystem btrfs ? "git log" o "gitk *"

```
[linux.git] cd fs                                                    wip/stmark2-jobs  ★ ●
[fs] cd btrfs                                                        wip/stmark2-jobs  ★ ●
[btrfs] git log *                                                    wip/stmark2-jobs  ★ ●
commit edf6b0e1e4ddb12e022ce0c17829bad6d4161ea7
Merge: 5a3c558a9f05 2d892ccdc163
Author: Linus Torvalds <torvalds@linux-foundation.org>
Date:   Sat Sep 12 12:28:39 2020 -0700

    Merge tag 'for-5.9-rc4-tag' of git://git.kernel.org/pub/scm/linux/kernel/git/kdave/linux

    Pull btrfs fixes from David Sterba:
     "A few more fixes:

        - regression fix for a crash after failed snapshot creation

        - one more lockep fix: use nofs allocation when allocating missing
          device

        - fix reloc tree leak on degraded mount

        - make some extent buffer alignment checks less strict to mount
          filesystems created by btrfs-convert"

    * tag 'for-5.9-rc4-tag' of git://git.kernel.org/pub/scm/linux/kernel/git/kdave/linux:
      btrfs: fix NULL pointer dereference after failure to create snapshot
      btrfs: free data reloc tree on failed mount
      btrfs: require only sector size alignment for parent eb bytenr
      btrfs: fix lockdep splat in add_missing_dev
```

Viaggio nel kernel Linux

E' stato caricato un determinato firmware ? dmesg

```
root@colibri-imx7-emmc:~# dmesg | grep firmware
[    0.362625] imx-sdma 20ec000.sdma: loaded firmware 3.3
```

Ci sono errori nel boot?

```
root@colibri-imx7-emmc:~# dmesg | grep err
[    0.000000] L2C-310 errata 752271 769419 enabled
[    0.214562] pwm-backlight: probe of backlight_lcd failed with error -22
[    0.214755] pwm-backlight: probe of backlight_lvds failed with error -22
[    0.358069] mxc_sdc_fb: probe of fb@1 failed with error -1
[    1.334296] sdhci: Copyright(c) Pierre Ossman
[    2.624227] imx-sgtl5000: probe of sound failed with error -22
```

Tipo di errore : include/uapi/asm-generic/errno-base.h

Con un po' di conoscenza del linguaggio C, grep errore, nano sound/soc/fsl/imx-sgtl5000.c

```
        cpu_np = of_parse_phandle(pdev->dev.of_node, "cpu-dai", 0);
        codec_np = of_parse_phandle(pdev->dev.of_node, "audio-codec", 0);
        if (!cpu_np || !codec_np) {
                dev_err(&pdev->dev, "phandle missing or invalid\n");
                ret = -EINVAL;
                goto fail;
        }
```

Viaggio nel kernel Linux - kernel oops

```
[    2.520000] Unable to handle kernel NULL pointer dereference at virtual address (ptrval)
[    2.520000] Oops: 00000000                    tipo d'errore
[    2.520000] PC: [<401ef2f4>] sdhci_esdhc_mcf_probe+0x20/0x1b2   <-- punto d'origine del problema
[    2.520000] SR: 2000  SP: (ptrval)  a2: 4700c500
[    2.520000] d0: 470a9a60    d1: 00000000    d2: 4038222a    d3: 4038222a
[    2.520000] d4: 00000000    d5: 4001a210    a0: 470a9a60    a1: 470a9ae4
[    2.520000] Process swapper (pid: 1, task=(ptrval))
[    2.520000] Frame format=4 eff addr=4038222a pc=402fabaa
[    2.520000] Stack from 4701le54:
[    2.520000]         00000014 00000000 00000000 40382234 4039abdc 401abc56 4038222a 40382234
[    2.520000]         4039abdc 401aa538 40382234 00000001 47011ed8 0000004a 40382234 4039abdc
[    2.520000]         401aacb2 40382234 4039abdc 00000001 40382234 401aadf4 4039abdc 40382234
[    2.520000]         00000000 401a8840 47495ef0 401a8abc 40382234 4039abdc 00000000 00000000
[    2.520000]         4039abdc 47022e9c 47042040 401aae0e 40398f04 00000000 4039abdc 401aacce
[    2.520000]         401a9258 4039abdc 00000007 4039abdc 403ba7e4 403b82b4 401ab396 4039abdc
[    2.520000] Call Trace: [<401abc56>] platform_drv_probe+0x1a/0x46
[    2.520000]  [<401aa538>] really_probe+0x246/0x3c4
[    2.520000]  [<401aacb2>] device_driver_attach+0x30/0x4c
[    2.520000]  [<401aadf4>] __driver_attach+0x126/0x12a
[    2.520000]  [<401a8840>] next_device+0x0/0x18
[    2.520000]  [<401a8abc>] bus_for_each
[    2.520000]  [<401aae0e>] driver_attac
[    2.520000]  [<401aacce>] __driver_atta
[    2.520000]  [<401a9258>] bus_add_drive
[    2.520000]  [<401ab396>] driver_regist
[    2.520000]  [<403b23a0>] sdhci_esdhc_m
[    2.520000]  [<403b23ae>] sdhci_esdhc_m
[    2.520000]  [<403a6a32>] do_one_initca
[    2.520000]  [<4001a210>] parse_args+0x
[    2.520000]  [<403a69d8>] do_one_initca
[    2.520000]  [<402cddc0>] strcpy+0x0/0x
[    2.520000]  [<403a6c74>] kernel_init_f
[    2.520000]  [<402cddc0>] strcpy+0x0/0x
[    2.520000]  [<403a6cbe>] kernel_init_f
[    2.520000]  [<403b23a0>] sdhci_esdhc_m
[    2.520000]  [<402d40be>] kernel_init+0
[    2.520000]  [<402d40c6>] kernel_init+0
[    2.520000]  [<402d40be>] kernel_init+0
```

```
static int sdhci_esdhc_mcf_probe(struct platform_device *pdev)
{
        struct sdhci_host *host;
        struct sdhci_pltfm_host *pltfm_host;
        struct pltfm_mcf_data *mcf_data;
        int err;

        host = sdhci_pltfm_init(pdev, &sdhci_esdhc_mcf_pdata,
                                sizeof(*mcf_data));

        host = 0;   <-- errore creato artificialmente

        if (IS_ERR(host))
                return PTR_ERR(host);

        pltfm_host = sdhci_priv(host);
        mcf_data = sdhci_pltfm_priv(pltfm_host);
```

errore di programmazione o problema hardware

Viaggio nel kernel Linux

- messaggi d'errore possono essere risolti con semplice ricerca google
- per chi ha una buona conoscenza del linguaggio C, grep del messaggio e visualizzazione dei sorgenti puo aiutare
- casi di freeze completo del kernel:
  - generalmente si tratta di un problema software
  - ma il problema puo derivare da un malfunzionamento dell'hardware, psu vecchio, mb, memorie difettose (provare memtest86+)
  - cercare di trovare un modo per riprodurre il problema
  - provare una diversa versione di kernel, anche piu vecchia, come 4.14, 4.19
  - succede solo da Xorg ? O anche da console pura ?
  - divide and conquer, ovvero cercare di escludere possibili sorgenti del problema, escludendo gli hardware e poi i software
  - provare un mprime fft test (almeno mezz'ora, con psensor tenere d'occhio la temp cpu)
  - quando si blocca, tentare comandi magic SysRq
  - se si verifica il problema, provare ad entrare in un altra virtual console (CTRL+ALT+F2)
  - possibile abilitare varie configurazioni di debug (menuconfig, kernel hacking)

Viaggio nel kernel Linux - Operazioni utili

```
dmesg

PROCFS
cat /proc/cpuinfo
cat /proc/version
cat /proc/cmdline
cat /proc/meminfo
cat /proc/interrupts
cat /proc/devices
cat /proc/iomem
cat /proc/mounts
cat /proc/partitions
cat /proc/sys/kernel/ostype
cat /proc/sys/kernel/osrelease
```

```
cat /proc/sysrq-trigger
cat /proc/sys/kernel/sysrq
cat /proc/sys/kernel/tainted
zcat /proc/config.gz | grep IDLE

SYSFS
cat /sys/class/rtc/rtc0/time
cat /sys/class/net/enp4s0/mtu
cat /sys/class/hwmon/hwmon0/name
cat /sys/class/hwmon/hwmon0/temp1_label
cat /sys/class/hwmon/hwmon0/temp1_input
cat /sys/class/input/input0/name
ls /sys/module
```

```
READ/WRITE OPERATIONS
echo 1123123123 > /dev/ttyS0
hexdump -C -n 32 /dev/sdb
hexdump -C -n 32 /sys/bus/.../eeprom
hexdump -C -n 32 /dev/urandom
echo 1234 > /sys/bus/.../eeprom

MODULES
modprobe modulo
modprobe -r modulo
lsmod | grep modulo

DEBUGFS
mount -t debugfs debugfs /sys/kernel/debug
```

```
GPIO
echo 200 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio200/direction
echo 1 > /sys/class/gpio/gpio200/value
echo 0 > /sys/class/gpio/gpio200/value
echo 0 > /sys/class/pwm/pwmchip0/export
echo in > /sys/class/gpio/gpio200/direction
cat /sys/class/gpio/gpio200/value
```

```
LEDS
echo 1 > /sys/class/leds/led1/brightness

PWM
echo 1000000 > /sys/class/pwm/pwmchip0/pwm0/period
echo 500000 > /sys/class/pwm/pwmchip0/pwm0/duty_cycle
echo 1 > /sys/class/pwm/pwmchip0/pwm0/enable

DRIVER-DEVICE BINDING
echo driver-name > /sys/bus/platform/devices/.../driver_override
echo device-name > /sys/bus/platform/drivers/.../bind
```

DOMANDE ?

GRAZIE A TUTTI

Sei interessato allo sviluppo nel kernel Linux ?
Sei interessato ad ulteriori dettagli ?

Visita:
http://kernelspace.baselinux.net
http://solidground.baselinux.net

Community:
https://matrix.to/#/@kernel-linux:matrix.org
https://matrix.to/#/@linux-embedded:matrix.org