Fast Startup-Linux

Wolfram Luithardt, Daniel Gachet und Guy Morand

Hochschule für Technik und Architektur, Fribourg, Schweiz

1.) Motivation

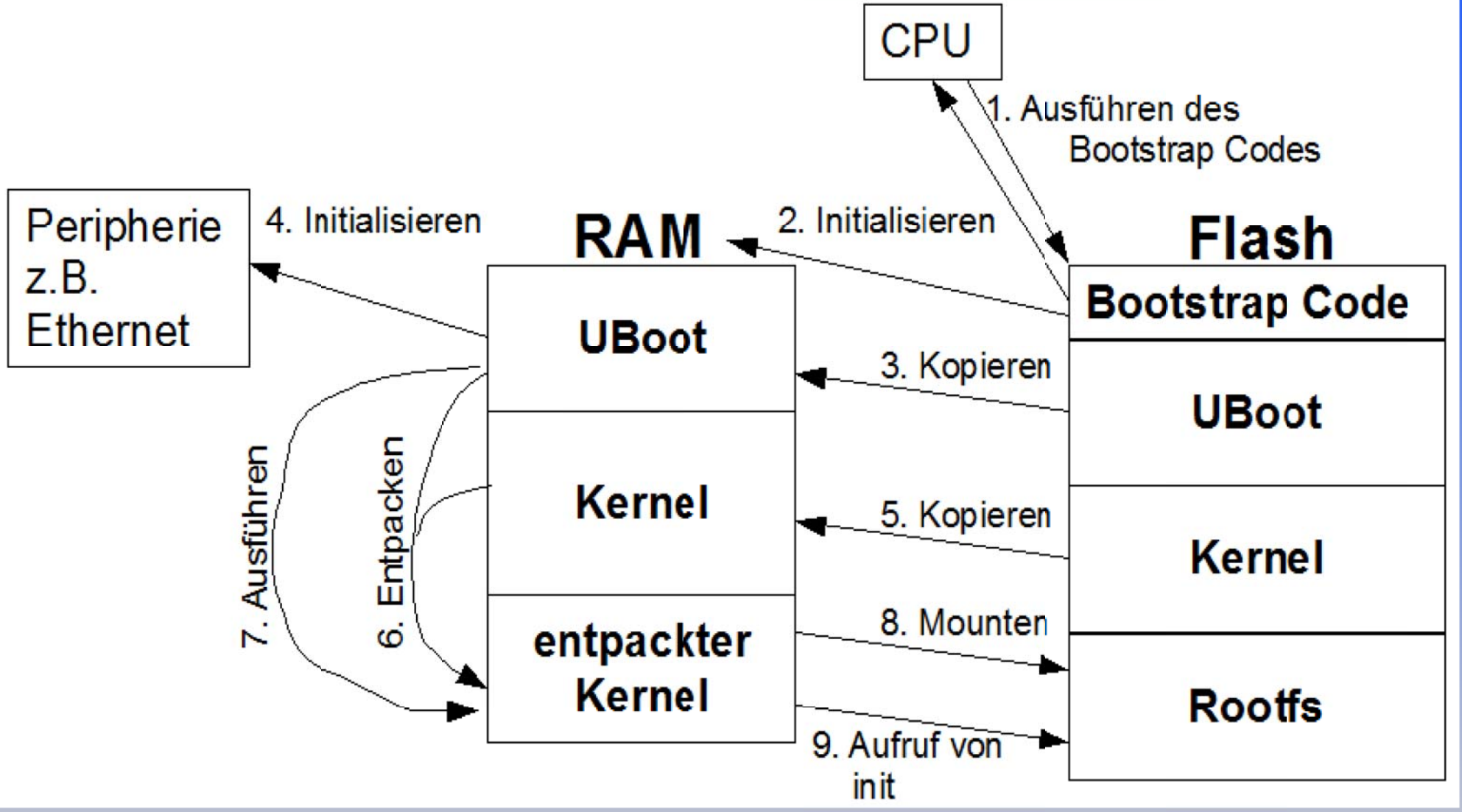
2.) Der Bootprozess

3.) Wie kann das Aufstarten verkürzt werden

4.) Zusammenfassung und Ausblick

Embedded Systems

Der Bootprozess



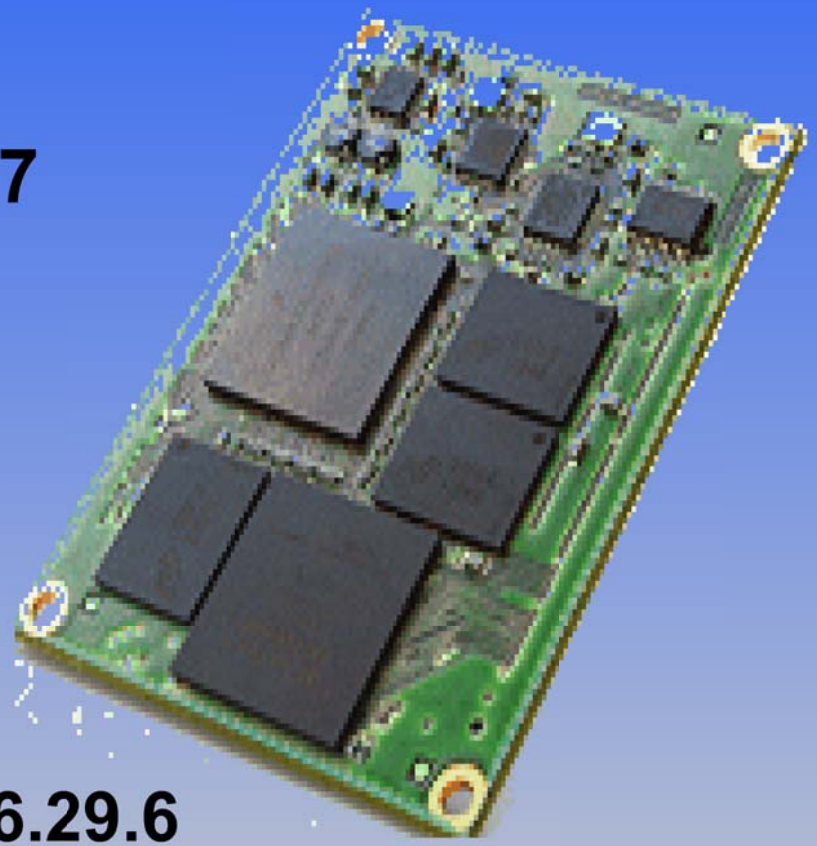
Das verwendete System

Plattform APF-27

www.armadeus.org

- Prozessor Freescale i.mx27
- Takt bis 400MHz
- 64 MB RAM
- 16 MB Flash
- Ethernet, USB etc...
- FPGA

....



Kernel 2.6.29.6

Messung der Bootzeit

LOGIC-Analyser

DO

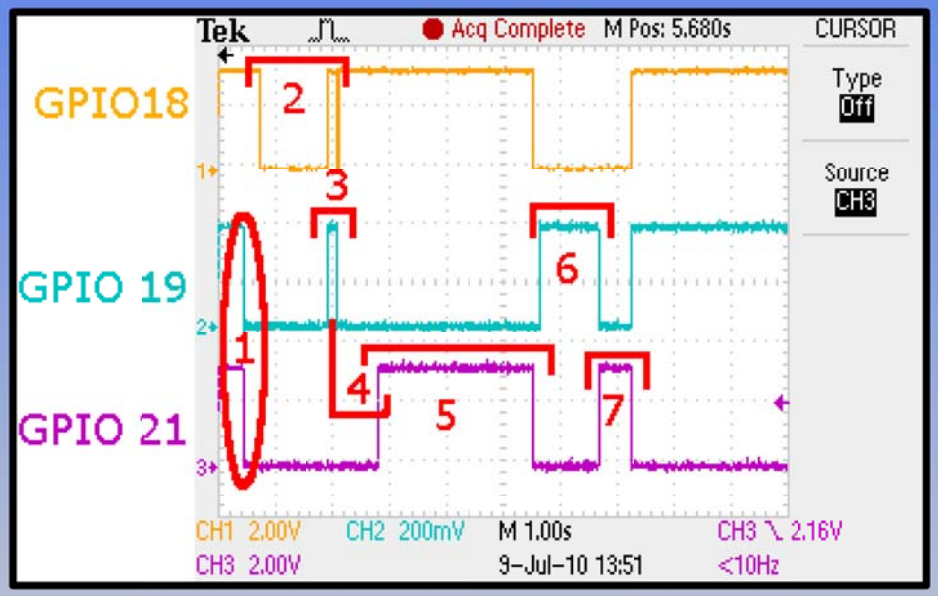
embedded System

RS-232

Terminalprogramm (z.B. minicom)

PC

1.) Patch und Ausgabe auf digitalem Ausgang: Untersuchung mit Logik-Analyser:



2.) Das Programm grabserial mit Parameter -t gibt einen Zeitstempel bei jeder Zeile am Eingang an:

```
[    0.000291] U-Boot 1.3.4 (Dec 29 2010 - 22:53:56) apf27 patch 2.1
[    0.004699]
[    0.004762] I2C:   ready
[    0.005083] DRAM:  256 MB
[    0.036006] NAND:  256 MiB
[    0.095948] In:    serial
[    0.097972] Out:   serial
[    0.098510] Err:   serial
[    0.215939] nand_unlock: start: 000a0000, length: 267780096!
[    0.231996] Hit any key to stop autoboot:  3
[    3.511954]
[    3.512196] Loading from NAND 256MiB 1,8V 16-bit, offset .....
[    4.291929] Automatic boot of image at addr 0xa0000000 ...
```

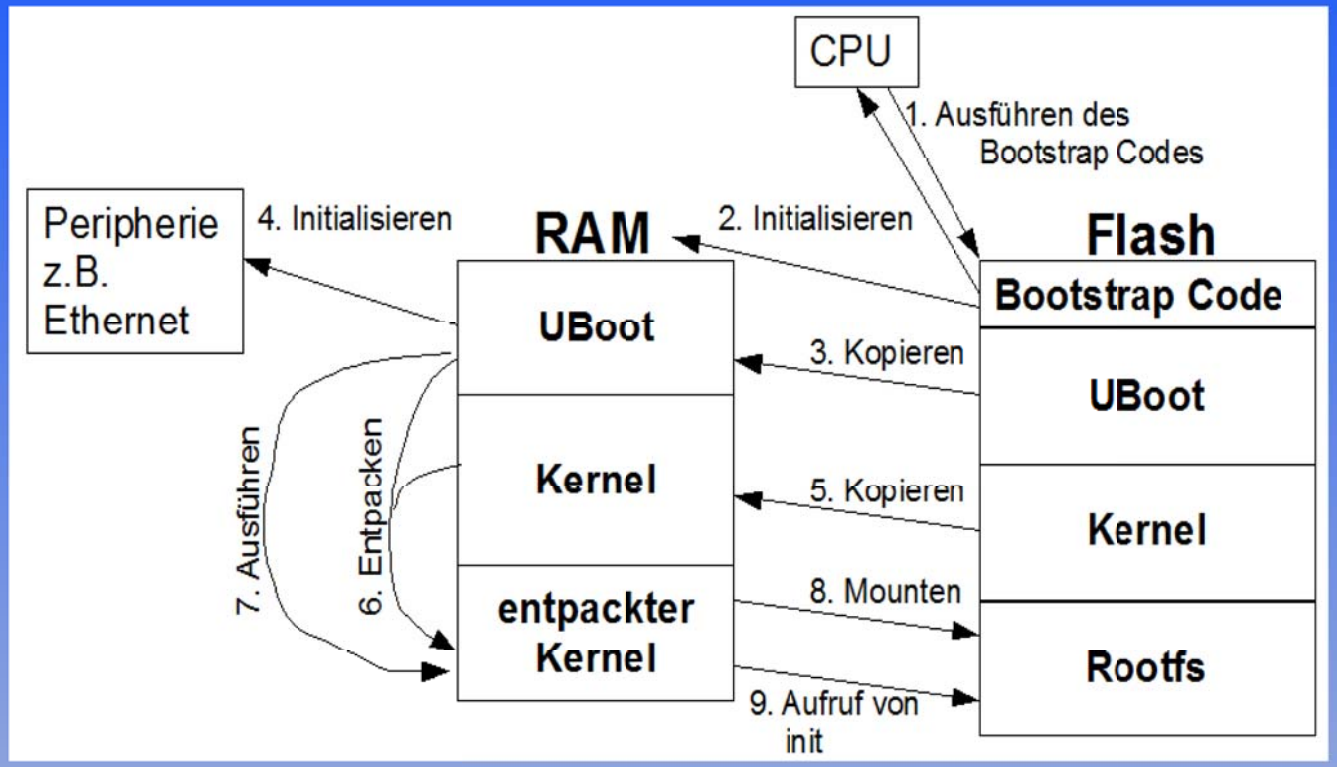
Booten der Distribution

```
[    0.000291] U-Boot 1.3.4 (Dec 29 2010 - 22:53:56) apf27 patch 2.1
[    0.004699]
[    0.004762] I2C:    ready
[    0.005083] DRAM:   256 MB
[    0.036006] NAND:   256 MiB
[    0.095948] In:     serial
[    0.097972] Out:    serial
[    0.098510] Err:    serial
[    0.215939] nand_unlock: start: 000a0000, length: 267780096!
[    0.231996] Hit any key to stop autoboot:  3

....

[   15.544356] Mounting all filesystems in /etc/fstab
[   15.600292] starting pid 396, tty '/dev/ttySMX0': '/sbin/getty -L
ttySMX0 11
[   16.635896]
[   16.636024]
[   16.640449]
[   16.640546] Welcome to the Armadeus development environment.
[   16.641858] armadeus login:
```

Aufteilung



1: Starten des Bootcodes: 52 ms

2+3:Init des Ram + Ausführen des Bootcodes: 392 ms

4: Initialisieren der Schnittstellen: 332 ms

4a: Bootdelay: 3.0 s

5: Laden des Kernels: 1.86 s

6: Entpacken des Kernels: 1.73 s

7: Aufstarten des Kernels: 3.17s

8: Mounten des Filesystems: 4.70s

9: Starten und ausführen von Init: 1.41s

$\Sigma \approx 16.6$ Sekunden

Bootdelay

Parameter **bootdelay** in U-Boot!

setenv bootdelay 0
saveenv

Aber Achtung: Bei eine Bootdelay von 0 kann oft nicht mehr unterbrochen werden!!!
In diesem Fall:
- JTAG
- flash_erase /dev/mtd3

Einsparung 3.0s

verify?

```
[    4.324247]    Load Address: a0008000
[    4.327998]    Entry Point:  a0008000
[    4.328945]    Verifying Checksum ... OK
[    4.892174]    Loading Kernel Image ... OK
[    5.460185] OK
[
```

Verifying Checksum dauert 0.56 Sekunden

In U-Boot kann dass Überprüfen der Checksumme mit der Umgebungsvariable
verify = n
ausgeschaltet werden.

Einsparung 0.56s

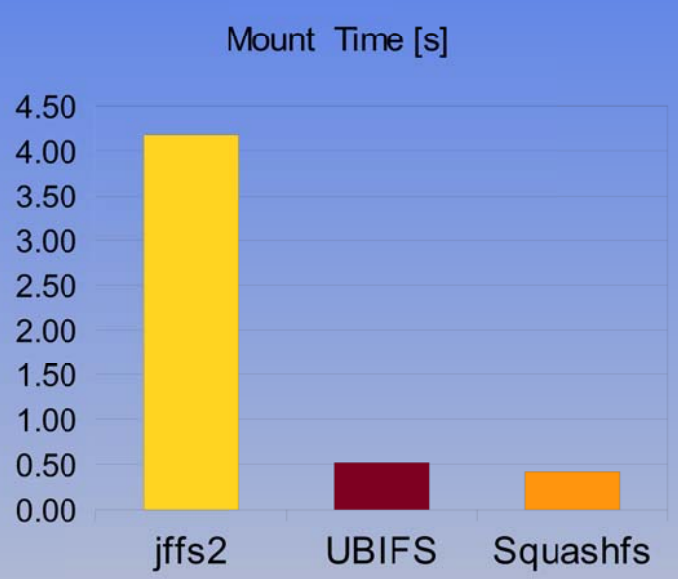
Mounten des Filesystems

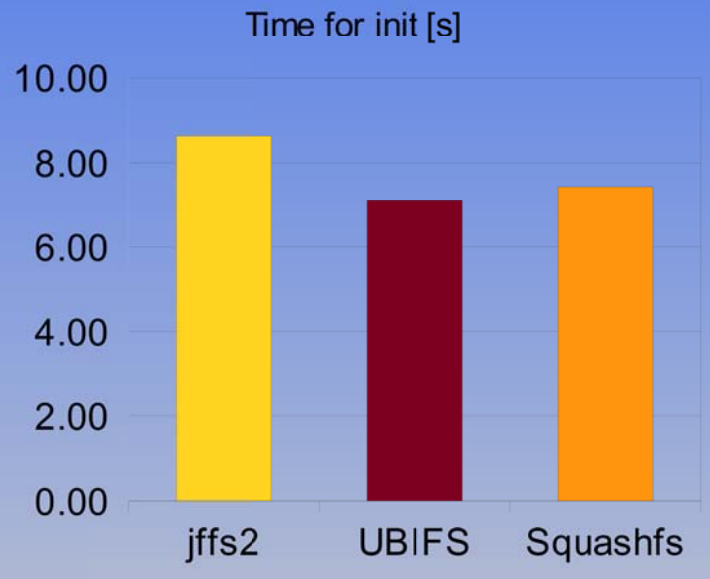
```
....
[    9.432202]      host=apf27, domain=, nis-domain=(none),
[    9.437040]      bootserver=192.168.0.2, rootserver=192.168.0.2, rootpath=
[   14.183978] VFS: Mounted root (jffs2 filesystem) on device 31:4.
[   14.189792] Freeing init memory: 124K
[   14.668542] init started: BusyBox v1.12.1 (2011-02-27 12:45:56 CET)
....
```

Mounten des jffs2 (journaling flash filessystem 2): Dauer 4.7 Sekunden!

Verschiedene Filesysteme

File System	Mount time [s]
JFFS2	4.18
UBIFS	0.52
SQUASHFS	0.40



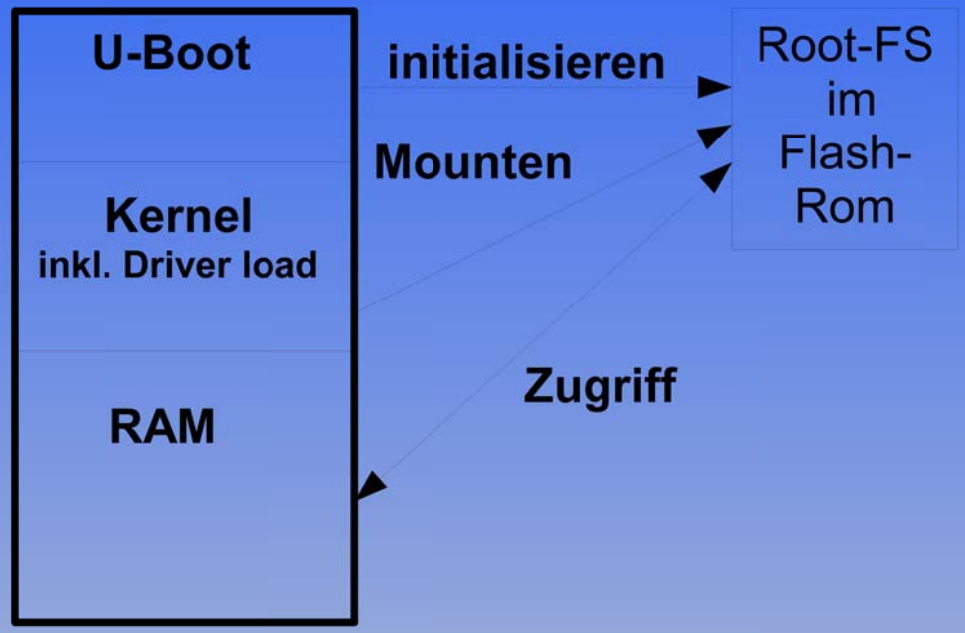


Einsparung 5.0 s

???

Ersparnis 5.0 s von 4.7 s ???

File System	Size [MB]
UBIFS	21.0
JFFS2	17.3
SQUASHFS	13.9



→ **Im Root-FS nur die Daten halten, die auch wirklich benötigt werden !**

Vereinfachung des Kernels

Per default beinhaltet der Kernel häufig viele Dienste die nicht unbedingt notwendig sind. Eine Optimierung des Kernels kann viele Vorteile bringen

Kernel optimieren

System sicherer

System transparenter

System schneller

System startet schneller

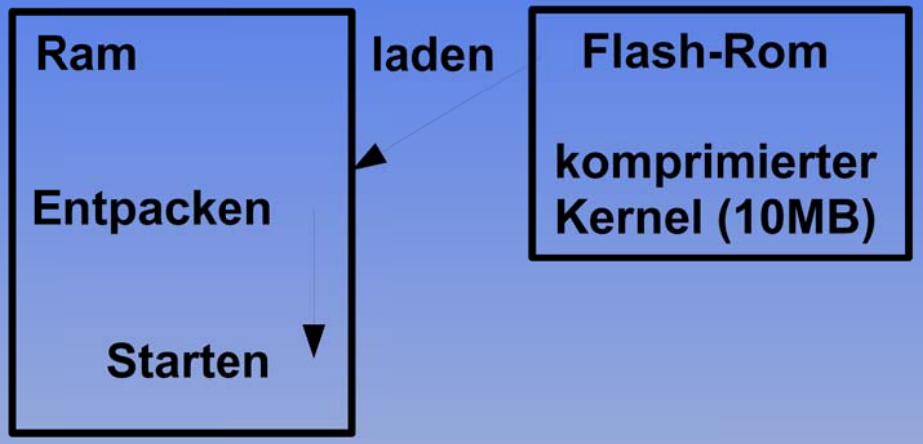
1.) Kopieren vom Flash in's Ram

2.) Abarbeiten des Kernelstarts

```
$
$ make linux26-config
.....
$ make linux26
```

Komprimieren oder nicht ?

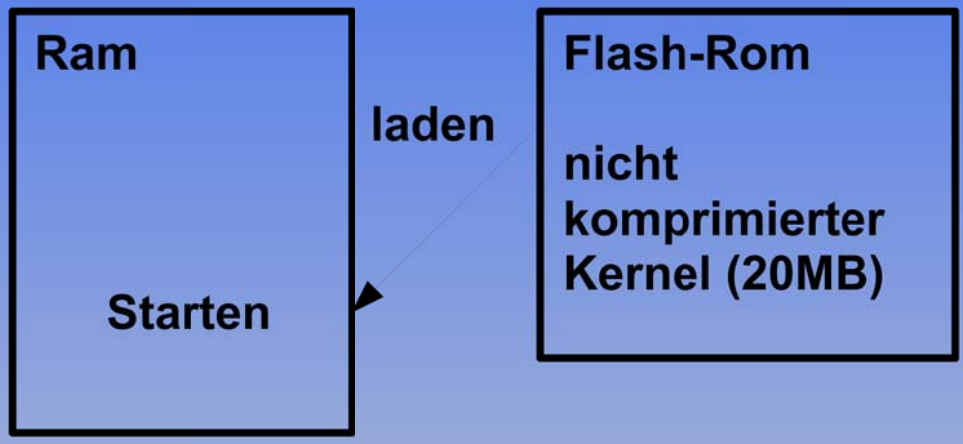
```
[    5.471741] Starting kernel ...
[    5.475888]
[    5.475963] Uncompressing Linux.....................................................
..... done, booting the kernel.
[    7.127679] Linux version 2.6.29.6 (root@luidl-laptop) (gcc version 4.2.1)
```



$T = T_{load} + T_{decompress}$

(Bus) (CPU)

UNC90: Zeitgewinn 20ms



$T = T_{load}$

(Bus)

APF27: Zeitverlust: 162 ms!

Analyse Start des Ethernets

```
[    7.951905] NET: Registered protocol family 17
[    7.955911] RPC: Registered udp transport module.
[    7.961008] RPC: Registered tcp transport module.
[    7.972021] rtc-ds1374 0-0068: setting system clock to 2008-12-11 02:03:11 UTC
[    8.484002] eth0: config: auto-negotiation on, 100FDX, 100HDX, 10FDX, 10HDX.
[    9.491916] IP-Config: Complete:
[    9.494260]      device=eth0, addr=192.168.1.20, mask=255.255.255.0, gw=192.168.1.3,
[    9.499886]      host=apf27, domain=, nis-domain=(none),
[    9.503615]      bootserver=192.168.1.3, rootserver=192.168.1.3, rootpath=
[    9.516055] VFS: Mounted root (squashfs filesystem) readonly on device 31:4.
[    9.519095] Freeing init memory: 120K
```

Warum dauert die Konfiguration des Ethernets mehr als 1.5 Sekunden ?

Ethernet: Delays optimieren

Häufig sind aus Gründen der Generalisierung bei der Initialisierung von Geräten delays eingebaut, die optimiert werden können. Beispiel: net/ipv4/ipconfig.c

```
/* Define the friendly delay before and after opening net devices */
#define CONF_PRE_OPEN       500 /* Before opening: 1/2 second */
#define CONF_POST_OPEN      1   /* After opening: 1 second */
/* Define the timeout for waiting for a DHCP/BOOTP/RARP reply */
#define CONF_OPEN_RETRIES   2   /* (Re)open devices twice */
#define CONF_SEND_RETRIES   6   /* Send six requests per open */
# ....
```

Zeit vor dem Öffnen (500 ms) und nach dem Öffnen (1s)

Patch

```
# ...
/* Define no delay before and after opening net devices */
#define CONF_PRE_OPEN      0   /* Before opening: no delay */
#define CONF_POST_OPEN     0   /* After opening: no delay */
/* Define the timeout for waiting for a DHCP/BOOTP/RARP reply */
#define CONF_OPEN_RETRIES  2   /* (Re)open devices twice */
#define CONF_SEND_RETRIES  6   /* Send six requests per open */
# ....
```

```
[   7.951905] NET: Registered protocol family 17
[   7.955911] RPC: Registered udp transport module.
[   7.961008] RPC: Registered tcp transport module.
[   7.972021] rtc-ds1374 0-0068: setting system clock to 2008-12-11 02:03:11 UTC
[   7.986331] eth0: config: auto-negotiation on, 100FDX, 100HDX, 10FDX, 10HDX.
[   7.990664] IP-Config: Complete:
[   7.991474]      device=eth0, addr=192.168.1.20, mask=255.255.255.0, gw=192.168.1.3,
[   7.998452]      host=apf27, domain=, nis-domain=(none),
[   8.002951]      bootserver=192.168.1.3, rootserver=192.168.1.3, rootpath=
[   8.020477] VFS: Mounted root (squashfs filesystem) readonly on device 31:4.
[   8.024020] Freeing init memory: 120K
```

Einsparung 1.5 s

Init et al.

```
[    9.764298] init started: BusyBox v1.12.1 (2011-02-18 22:05:04 CET)
[    9.900017] starting pid 351, tty '': '/sbin/mdev -s'
[   10.568027] starting pid 362, tty '': '/etc/init.d/rcS'
[   10.624003] /etc/init.d/S00machine_name: line 8: can't create /etc/machine: Read-only file system
[   10.636021] /etc/init.d/S01fb: line 11: can't create /etc/fb.modes: Read-only file system
[   10.652019] Starting portmap: done
[   10.780021] Initializing random number generator... read-only file system detected...done
[   10.820017] Starting network...
[   10.876021] ip: RTNETLINK answers: File exists
[   10.916043] Mounting all filesystems in /etc/fstab
[   10.984044] starting pid 392, tty '/dev/ttySMX0': '/sbin/getty -L ttySMX0 115200,57600,38400 vt100'
[   12.015956]
[   12.016795]
[   12.016976]
[   12.017044] Welcome to the Armadeus development environment.
[   12.020807] armadeus login:
```

Wieviele Prozesse wurden bereits gestartet?

Ganz einfach: ps ax

Häufig wurden bereits während des Aufstartmechanismusses einige Hundert Prozesse aufgerufen:

APF27: pid=396

Start von getty

```
[   10.876021] ip: RTNETLINK answers: File exists
[   10.916043] Mounting all filesystems in /etc/fstab
[   10.984044] starting pid 392, tty '/dev/ttySMX0': '/sbin/getty -L ttySMX0 115200,57600,38400 vt100'
[   12.015956]
[   12.016795]
```

1,032 Sekunden ?

armadeus-3.3/buidroot/project_build_armv5te/apf27/busybox-1.12.1/loginutils/getty.c

```
/*
 * Wait for a while, then read everything the modem has said so
 * far and try to extract the speed of the dial-in call.
 */
sleep(1);
nread = safe_read(STDIN_FILENO, buf, size_buf - 1);
if (nread > 0) {
        buf[nread] = '\0';
        .....
```

busybox

getty ist ein programm aus Busybox

Sourcecode ändern

```
$ make busybox-clean
$ make busybox
$ make
```

```
[   10.876021] ip: RTNETLINK answers: File exists
[   10.916043] Mounting all filesystems in /etc/fstab
[   10.984044] starting pid 392, tty '/dev/ttySMX0': '/sbin/getty -L ttySMX0 115200,57600,38400 vt100'
[   11.060065]
[   11.060870]
```

Einsparung: 1.0s

Init Scripts

zum Beispiel: S00machine_name

```
[   10.624003] /etc/init.d/S00machine_name: line 8: can't create /etc/machine: Read-only file system
```

```
#!/bin/sh
FILE=/etc/machine
if [ ! -f "$FILE" ]; then
        MACHINE=`cat /proc/cpuinfo | grep Hardware`
        echo ${MACHINE:20} > $FILE
fi
exit 0
```

Einsparung: einige ms

mdev

```
[    9.764298] init started: BusyBox v1.12.1 (2011-02-18 22:05:04 CET)
[    9.900017] starting pid 351, tty '': '/sbin/mdev -s'
[   10.568027] starting pid 362, tty '': '/etc/init.d/rcS'
```

mdev (mini udev) ist ein Usermode-tool zur Initialisierung des /dev-Filesysytems. Mit dem Parameter -s 'scanned' es das sysFS und erstellt daraus die Devices mit mkdev.

```
# ls /dev/tty*
/dev/tty      /dev/tty20    /dev/tty33    /dev/tty46    /dev/tty59
/dev/tty0     /dev/tty21    /dev/tty34    /dev/tty47    /dev/tty6
/dev/tty1     /dev/tty22    /dev/tty35    /dev/tty48    /dev/tty60
/dev/tty10    /dev/tty23    /dev/tty36    /dev/tty49    /dev/tty61
/dev/tty11    /dev/tty24    /dev/tty37    /dev/tty5     /dev/tty62
/dev/tty12    /dev/tty25    /dev/tty38    /dev/tty50    /dev/tty63
/dev/tty13    /dev/tty26    /dev/tty39    /dev/tty51    /dev/tty7
/dev/tty14    /dev/tty27    /dev/tty4     /dev/tty52    /dev/tty8
/dev/tty15    /dev/tty28    /dev/tty40    /dev/tty53    /dev/tty9
/dev/tty16    /dev/tty29    /dev/tty41    /dev/tty54    /dev/ttySMX0
/dev/tty17    /dev/tty3     /dev/tty42    /dev/tty55    /dev/ttySMX2
/dev/tty18    /dev/tty30    /dev/tty43    /dev/tty56
/dev/tty19    /dev/tty31    /dev/tty44    /dev/tty57
/dev/tty2     /dev/tty32    /dev/tty45    /dev/tty58
#
```

Reduktion der Devices

./project_build_armv5te/apf27/linux-2.6.29.6/include/linux/vt.h

```
/*
 * These constants are also useful for user-level apps (e.g., VC
 * resizing).
 */
#define MIN_NR_CONSOLES 1       /* must be at least 1 */
#define MAX_NR_CONSOLES 63      /* serial lines start at 64 */
#define MAX_NR_USER_CONSOLES 63 /* must be root to allocate above this */
            /* Note: the ioctl VT_GETSTATE does not work for
               consoles 16 and higher (since it returns a short) */
```

z.B. 15

Einsparung: 198 ms

Ausgabemeldungen

Mit dem Bootparameter 'quiet' können die Ausgabemeldungen des Kernels abgeschaltet werden!!!

Einsparung: 540 ms

LPJ-Kalibrierung

Der Kernel 'misst' am Anfang die Geschwindigkeit des Systems. Dieser Prozess benötigt Zeit, es gibt aber immer die gleichen Werte. Der Kernel erlaubt die manuelle Eingabe dieses Wertes: z.B. hier lpj =995328

Dieser Wert kann in der Datei: /var/log/messages entnommen werden:

```
Dec 23 21:05:04 armadeus user.info kernel: Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
Dec 23 21:05:04 armadeus user.info kernel: Memory: 128MB 128MB = 256MB total
Dec 23 21:05:04 armadeus user.info kernel: Calibrating delay loop... 199.06 BogoMIPS (lpj=995328)
Dec 23 21:05:04 armadeus user.warn kernel: Mount-cache hash table entries: 512
Dec 23 21:05:04 armadeus user.info kernel: CPU: Testing write buffer coherency: ok
Dec 23 21:05:04 armadeus user.debug kernel: PM: Adding info for No Bus:platform
Dec 23 21:05:04 armadeus user.info kernel: net_namespace: 716 bytes
Dec 23 21:05:04 armadeus user.info kernel: NET: Registered protocol family 16
```

Einsparung: 134 ms

Alles zusammen....

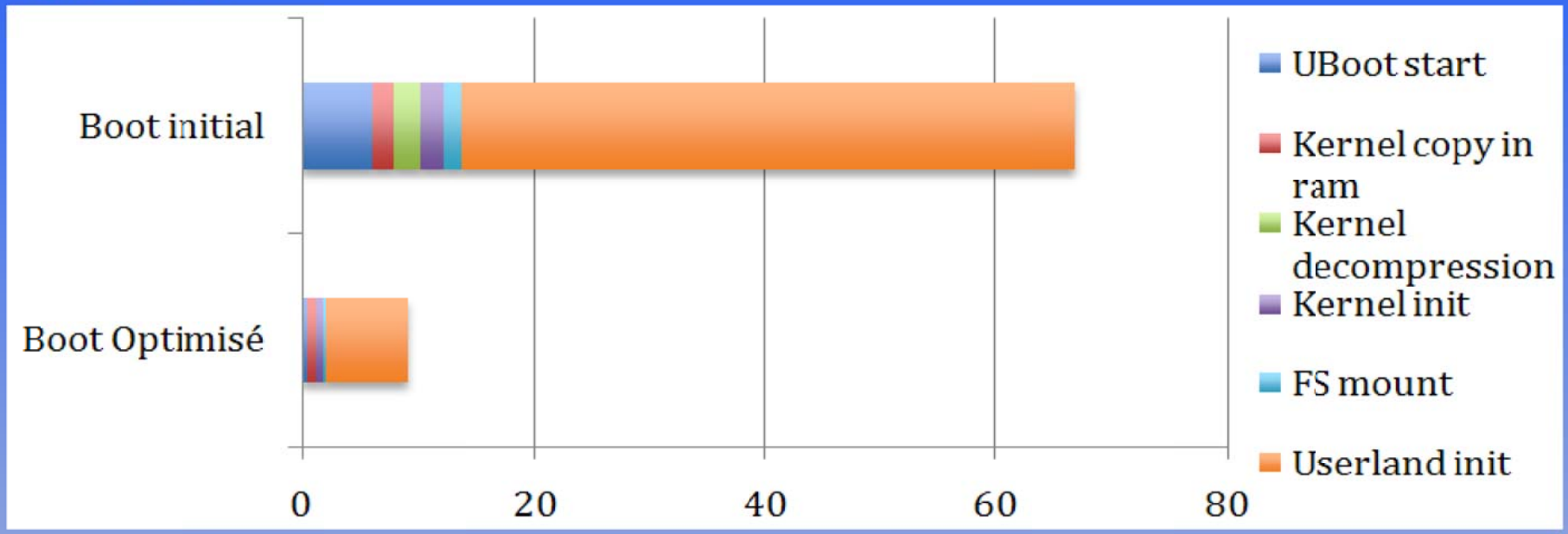
```
[    4.891167]
[    4.891214]
[    4.891260] Welcome to the Armadeus development environment.
[    4.892067] armadeus login:
```

Gesamte Zeitersparnis 16.7 s - 4.9 s = 11.8 Sekunden ≙ 70%

Mit Vereinfachung des Kernels: Startzeit = 3.3 s

Anderes System

Karo TX-25



66 Sek ► 8 Sek

Quelle: D. Doninelli: TX25-Fast Boot, EIA-FR 2011

Und wie weiter ?

Es gibt einige Distributionen, die auf eine schnelle Bootzeit optimiert sind:
Moblin, Jolicloud, xPud. PuppyLinux

initng ersetzt das sys-V init und parallelisiert viele Dienste beim Aufstarten.

NOR-Flash....

Dank an die beteiligten Personen

Vielen Dank an:

Guy Morand und **Davide Doninelli** für Ihre tollen Arbeiten zum Fastboot an verschiedenen Systemen.

Silvain Décastel, Xavier Menoud, Andéol Demierre und Igor Zanetti für die interessanten Analysen des Linux-Systems.

Sowie an **Daniel Gachet** und **Nicolas Schroeter** für viele interessante Diskussionen zu diesem Thema....

Vielen Dank auch an Euch für die Aufmerksamkeit !

wolfram.luithardt@hefr.ch