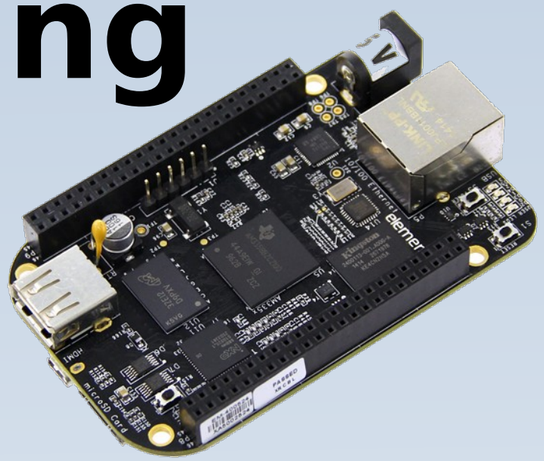
MindRoad

Presentationen finns på:
https://www.lysator.liu.se/~kjell-e/tekla/linux/dokument/linux-bootoptimering.pdf

Embedded-Linux, Boot-optimering



Kjell Enblom

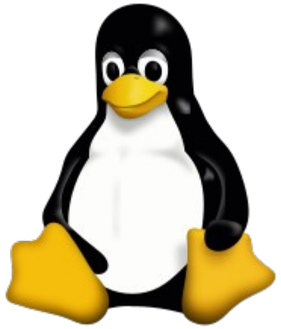
Senast uppdaterad:
2023-04-09

MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Sourcing

MindRoad

Vem är jag?

- Kjell Enblom
  - Kjell.Enblom@mindroad.se
- Har hållit på med unix sedan 1985.
- Har arbetat med Linux sedan 1995.
  - Servrar
  - Arbetsstationer
  - Inbyggda system med Linux sedan 2007.
- Förutom att konsulta undervisar jag i bland annat:
  - Grunderna i Linux
  - Linux i inbyggda system
  - Yocto Project.



På geocachingtur i Skottland

MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

MindRoad

Boot-optimering

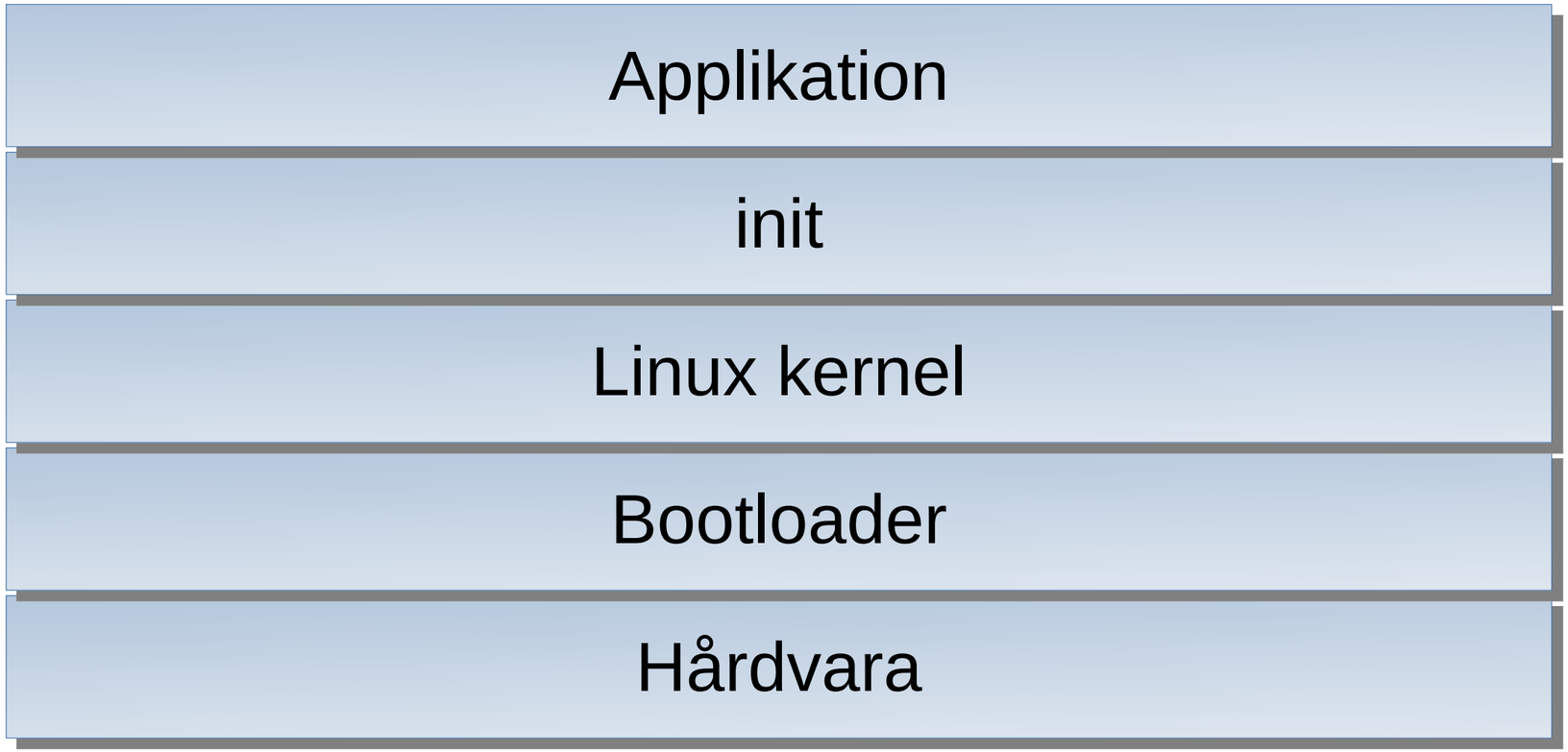
- Varför?
  - Kundkrav
  - Marknadskrav
  - En del av energisparande (power off och snabb boot istället för suspend).

- Grundläggande principer
  - Mät
  - Utvärdera
  - Ändra

MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

MindRoad

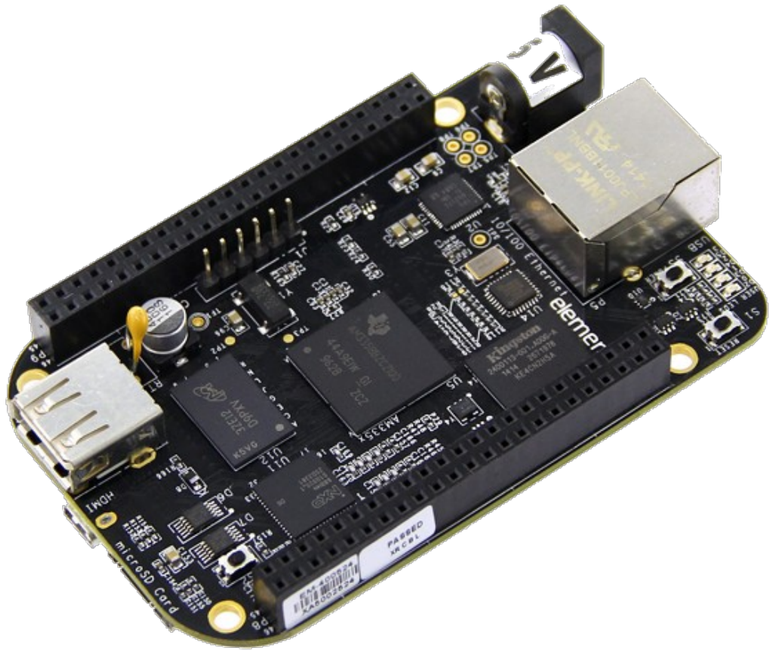
De olika delarna



MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

Labbuppställning

- I de exempel som tas upp här används en BeagleBone Black (single core Cortex A-8, 1GHz)
- U-Boot 2019.01
- Linuxkärna 5.15.12
- Busybox init
- Några enkla program



MindRoad

Verktyg

- Grabserial: Läser från seriekonsolen och lägger till tidsstämplar.
  - https://elinux.org/Grabserial (skriven av Tim Bird)
- Bootchart2: Visar tider för user space processer.
  - https://github.com/xrmx/bootchart
- Bootgraph: Visar tider för olika funktioner i linuxkärnan.
  - Följer med källkoden till Linuxkärnan.
- Arduino med en 7-segment display eller annan dator där den mäter tiden mellan reset och att en GPIO-pinne höjs.
- Oscilloskop (ändra på någon eller några GPIO-pinnar och mät med oscilloskop)
- Logikanalysator

MindRoad

Mätningar

- Notera att om vi mäter samma sak flera gånger kan tiderna variera.
- Mätningar kan också påverka det som mäts.

MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

grabserial

- Vi börjar med att mäta med grabserial.
  - grabserial -d /dev/ttyUSB0 -t -m "U-Boot SPL" -e 45 -o boot.log
    - -d device att läsa från
    - -t lägg till tidsstämplar
    - -m “regexp” Startar mätningarna vid detta reguljära uttryck.
    - -e fånga data så här många sekunder.
    - -o spara resultatet i denna fil, här boot.log.

grabserial

```
[0.000002 0.000002]
[0.000277 0.000276] U-Boot SPL 2019.01 (Jan 06 2022 - 16:45:57 +0100)
[0.063803 0.063803] Trying to boot from MMC1
…
[0.352536 0.000036] U-Boot 2019.01 (Jan 06 2022 - 16:45:57 +0100)
[0.374262 0.021726]
[0.374307 0.000045] CPU  : AM335X-GP rev 2.1
[0.374948 0.000641] I2C:   ready
[0.421813 0.046865] DRAM:  512 MiB
...
...
[3.561370 0.000033] Starting kernel ...
[3.561888 0.000518]
[6.250698 2.688810] [    0.000000] Booting Linux on physical CPU 0x0
[6.294788 0.044091] [    0.000000] Linux version 5.15.12 (kjell-e@gromit) (arm-cortex_a8-linux-gnueabi-gcc (crosstool-NG 1.24.0) 8.3.0,
GNU ld (crosstool-NG 1.24.0) 2.32) #1 SMP Thu Jan 6 01:20:39 CET 2022
...
...
[7.832279 0.023493] [    3.347123] devtmpfs: mounted
[7.833105 0.000826] [    3.352599] Freeing unused kernel image (initmem) memory: 2048K
[7.834754 0.001649] [    3.359313] Run /sbin/init as init process
[7.927807 0.093053] Running /etc/inittab
[7.943956 0.016149] mounting /proc
[7.944519 0.000563] mounting /tmp
[7.959922 0.015404] mounting /sys
[7.960408 0.000486]  Mounting /dev  Mounting /dev/pts  Populating /dev Starting system logger
[8.328653 0.368245] Starting a large application
[23.413501 15.084847] Starting the important application
[23.461618 0.048117] This is the important application!
```

MindRoad

Mätning i user space

- I detta exempel kan vi se att det tar ca 23 sekunder innan det viktiga programmet startar.
- Genom att aktivera bootchartd I busybox kan vi få tider på program i userspace.
- Sätt init till **bootchartd**. Den kommer i sin tur att starta init.
- Exempel på kärnparametrar:
  - console=ttyO0,115200n8 root=/dev/mmcblk0p2 rw rootfstype=ext4 rootwait **init=/sbin/bootchartd**
- Lyft över filen /var/log/bootlog.tgz till host-maskinen och kör:
  pybootchartgui ./bootlog.tgz
  - Som resultat får man filen bootchart.png
- För systemd kan vi använda programmet **systemd-analyze** för att få tider på program I user space.

MindRoad

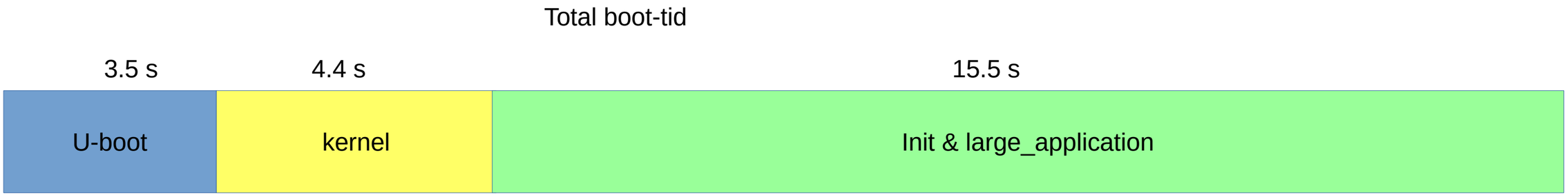
Bootchart2 – exempel (bootchart.png)

Ungefär här startar
det viktiga programmet

MindRoad

Urspungliga tider

- Tiden fram tills den viktiga applikationen startar är ca 23 s, varav 3.5s för bootloadern, 4.4 s för linuxkärnan och 15.5 s i user space.



MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

MindRoad

Optimering i user space

- Genom att ändra i startordningen kan vi lägga den viktiga applikationen före andra program.
- Det går även att boota till ett script som först startar den viktiga applikationen och därefter startar init som i sin tur startar resten.
  - Notera dock att alla filsystem antagligen inte är monterade när den viktiga applikationen startar.

init=/usr/sbin/mystartscript.sh

```
#! /bin/sh

# starting the important application
theapplication &

exec /sbin/init
```

MindRoad

Optimering i user space

- Tiden tills den viktiga applikationen startar är nu 8.3 sekunder istället för 23.5 som vi hade tidigare. Detta fick vi genom att vi bara ändrade om i startordningen.

```
[0.000001 0.000001]
[0.000253 0.000252] U-Boot SPL 2019.01 (Jan 06 2022 - 16:45:57 +0100)
[0.063834 0.063834] Trying to boot from MMC1
…
...
[7.859599 0.001026] [    3.376210] devtmpfs: mounted
[7.860112 0.000513] [    3.381620] Freeing unused kernel image (initmem) memory: 2048K
[7.861152 0.001040] [    3.388609] Run /sbin/bootchartd as init process
[7.951860 0.090708] Running /etc/inittab
[7.968248 0.016387] mounting /proc
[7.998946 0.030698] mounting /tmp
[7.999419 0.000473] mounting /sys
[7.999807 0.000387]  Mounting /dev  Populating /dev
[8.335649 0.335842] Starting the important application
[8.352516 0.016867] Starting system logger
[8.367764 0.015248] Starting a large application
```

MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

MindRoad

Optimering i user space

- Genom att skapa alla nödvändiga devicefiler på förhand (mknod) istället för att använda mdev/udev för dynamisk hantering av devicefiler kan man spara ytterligare lite tid.
- Här här minskade tiden med ca 0.3s.

```
[7.996976 0.104738] Running /etc/inittab
[7.997608 0.000632] mounting /proc
[8.027033 0.029425] mounting /tmp
[8.027499 0.000466] mounting /sys
[8.027885 0.000386] Starting the important application
```

- Fler saker som kan vara värda att titta på och mäta är tiden för att montera olika filsystem.
- Byte av filsystemstyp för root-filsystemet kan ge en tidsvinst. Ubifs är t.ex. snabbare än jffs2.
  - Ett alternativ är att lägga hela root-filsystemet i en initramfs och tillsammans med device tree lägga till dem till linuxkärnfilen. Det ger en en fil och endast en filläsning istället för tre; kärna, devicetree och rootfilsystem.

MindRoad

Optimering i user space

- Genom att köra strip på programmet kan tiden minska ytterligare. Hur mycket beror bland annat på programmets storlek.
- Länka programmet statiskt kan minska tiden ytterligare.
  - I detta exempel gick tiden ner till ca 7.98 sekunder.
- Genom att ta bort bootchartd igen och minska storleken på busybox minskade tiden till 7.97s. Att ta bort bootchartd gjorde en del av minskningen.
- Bygg program med bara de features som behövs och välj bort de som inte behövs.
- Optimera features som behövs.
- Bygg busybox med bara de funktioner som behövs.
- Om möjligt montera /proc och /sys efter att systemet har kommit igång.

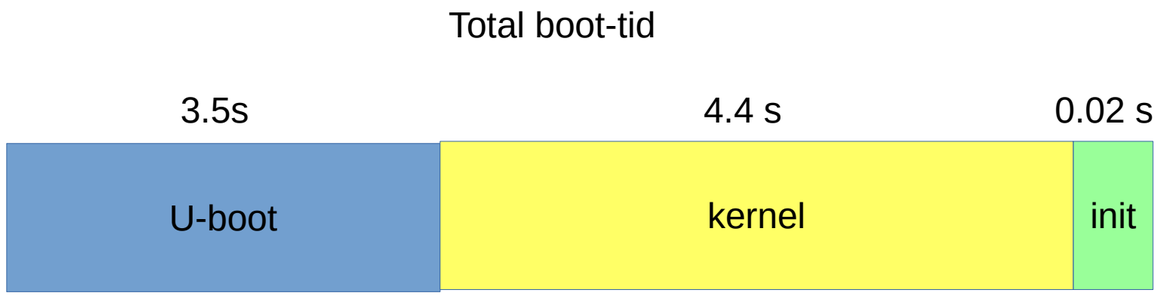
Optimering i user space

- Att länka de viktiga programmen statiskt kan ge tidsvinst. Linux läser bara in den delen av programmet som ska exekveras närmast.
- Med dynamiskt länkade program måste även de dynamiska biblioteken hittas och därefter i sin helhet läsas in i RAM.
- Tänk på att det tar ganska lång tid att detektera USB-enheter.
  - Om det går att undvika att ha viktiga saker på USB är det en fördel.
- Filsystem som inte behövs omedelbart kan med fördel automounteras (automount).
  - Om en process behöver accessa en fil i ett sådant filsystem kommer den att vänta tills filsystemet är monterat.

MindRoad

Optimering i user space

- Tiden i user space har minskat till ca 0.02 sekunder.



MindRoad

Kernel space optimering

- För att mäta tider för olika funktioner I kärnan behöver vi konfigurera kärnan och aktivera CONFIG_PRINTK_TIME och CONFIG_KALLSYMS.
- Sedan behöver vi skicka med initcall_debug bland parametrarna till kärnan.
  - console=ttyO0,115200n8 root=/dev/mmcblk0p2 rw rootfstype=ext4 rootwait **initcall_debug**

MindRoad

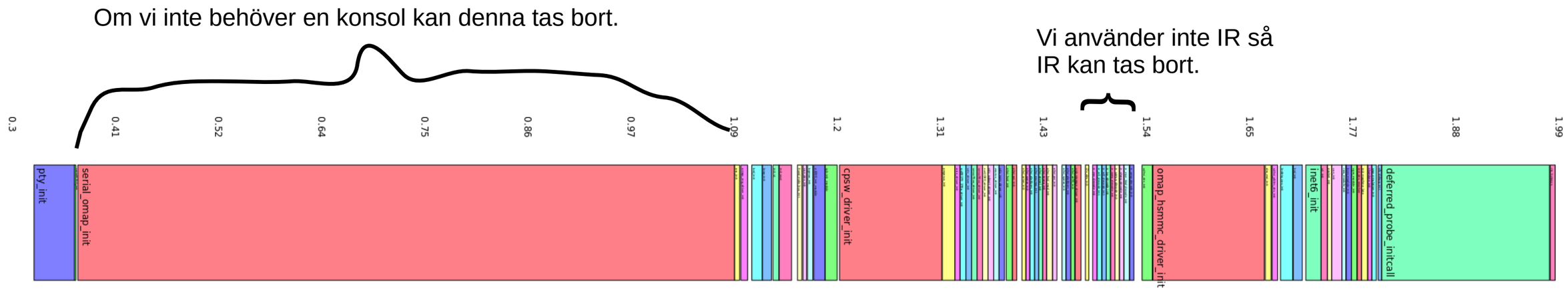
Kernel space optimering

- Efter att systemet har bootat behöver loggarna sparas vilket görs med:
  - **dmesg > /var/log/kernel.log**
- Kopiera filen till host-maskinen och kör följande:
  - **linuxkernelsource/scripts/bootgraph.pl kernel.log > kernelboot.svg**
- Använd något lämpligt program för att titta på svg-filen, t.ex. inkscape.

MindRoad

Kernel space optimering

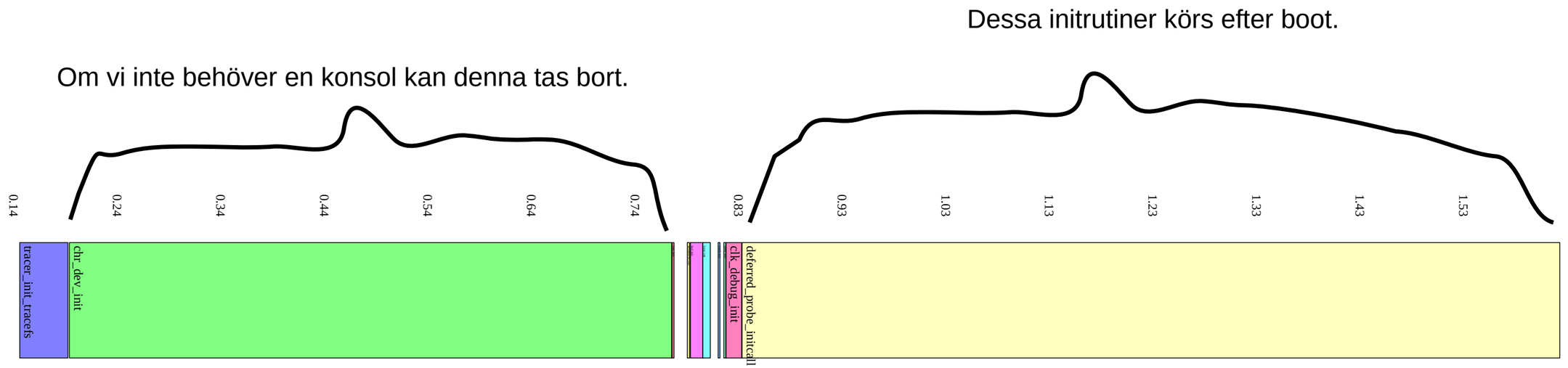
- Exempel, här med en 4.x-kärna:



MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

Kernel space optimering

- Exempel, här med vår 5.15.12-kärna:



MindRoad

Kernel space optimering

- Titta i loggarna från när kärnan bootar och titta efter calibrating delay loop.
  - **Calibrating delay loop... 996.14 BogoMIPS (lpj=4980736)**
- Lpj och det värde som står inom parentes använder vi och skickar som parameter till kärnan. Kan spara ca 100-200 ms.
- Avaktivera och ta bort sådant som inte används.
- Ta bort CONFIG_PRINTK_TIME, CONFIG_PRINTK, CONFIG_KALLSYMS och CONFIG_BUG
- Ta bort det som inte används från device-tree.
- Stäng av utskrifter från kärnan under boot genom att lägga till **quiet** till parametrarna till kärnan, alternativt sätt loglevel=5 (endast visa varningar och allvarliga fel).
  - console=ttyO0,115200n8 root=/dev/mmcblk0p2 rw rootfstype=ext4 rootwait **lpj=4980736 quiet**

MindRoad

Kernel space optimering

- Byt komprimeringsmetod för kärnan.
- Storleken på kärnan och tiden det tar att kopiera den respektive tiden det tar att dekomprimera den kan behöva mätas.
- GZIP och LZO är vettigt att välja som komprimering för inbyggda system då de ger bäst hastighet och vettig storlek.
- Om det finns möjlighet att lägga bootloader och linuxkärna på ett media som det går att exekvera från (execute in place) som NOR flash, ROM, EPROM, etc kan spara en del tid istället för att kopiera från t.ex. NAND flash till RAM och exekvera därifrån.
- Ta bort sysfs om den inte behövs.

MindRoad

Kernel space optimering

- I detta exampel har flera saker som inte behövs tagits bort.
- Tiden minskade från ca 5.9 till ca 5.7 sekunder från start tills den viktiga applikationen kom igång.
- Att stänga av möjligheten att ladda moduler under drift kan minska boot-tiden med några 10-tals millisekunder. Observera att du först behöver deaktivera de delar som skulle ha blivit moduler så att de inte läggs in fast in i kärnan.
- Att sätta lpj=<värde> kan spara ca 100 ms.
- Med quiet minskar boot-tiden ytterligare med ca 0.8 sekunder.
- Ta bort systemanrop som inte kommer att användas.
- Om det bara finns en CPU-kärna, deaktivera SMP-stöd.

MindRoad

Kernel space optimering

- Med LZO valt, lpj satt, allt stöd för grafik borttaget, inget SMP-stöd, stöd för laddbara moduler borttaget, etc får vi ner tiden för kärnan till 0.795 sekunder.
- Total tid fram till start av den viktiga applikationen ligger nu på 3.95 s.

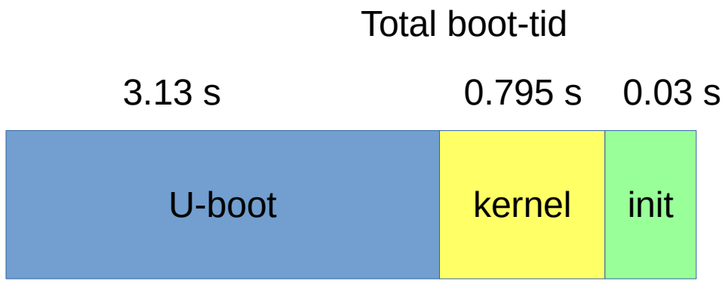
```
[0.000002 0.000002]
[0.001438 0.001436] U-Boot SPL 2018.01 (Apr 17 2020 - 08:47:05)
[0.061983 0.061983] Trying to boot from MMC1
[0.205915 0.143932] *** Warning - bad CRC, using default environment
[0.213948 0.008033]
[0.221732 0.007784] reading u-boot.img
...
[3.132047 0.000077] Starting kernel ...
[3.133363 0.001316]
[3.927178 0.793815] Running /etc/inittab
…
[3.931947 0.004769] mounting /tmp
[3.946582 0.014635] Starting the important application
```

MindRoad Academy MindRoad Embedded MindRoad Application MindRoad Syd MindRoad Sourcing

MindRoad

Kernel space optimering

- Tiden för Linuxkärnan har därmed minskat från ca 2.96 sekunder till ca 0.795 sekunder.
- Med lite mer arbete, t.ex. ta bort seriekonsolen, kan det gå att få ner tiden ytterligare något.



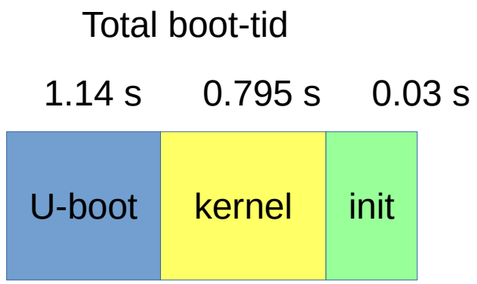
MindRoad

Kernel space optimering – dynamiska moduler

- Om vi vill ha stöd för laddbara moduler kan vi se till att moduler som inte behövs för att kunna köra den viktiga applikationen att de laddas in senare.

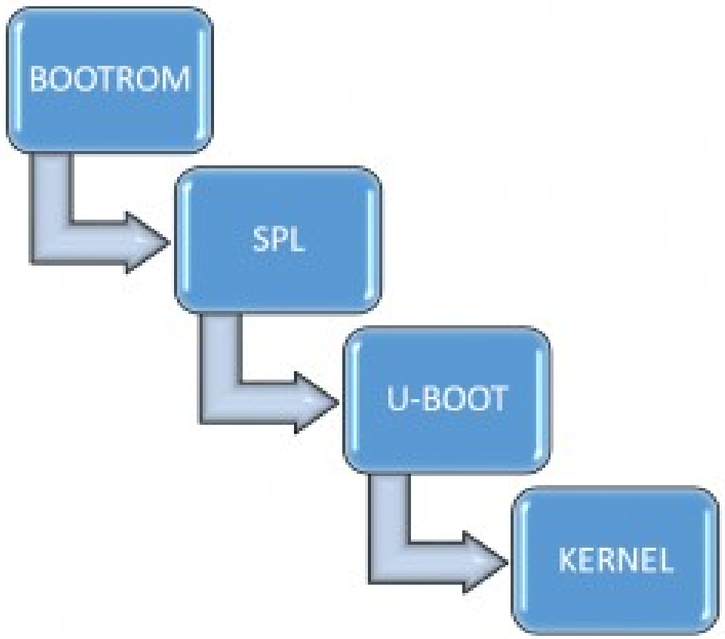
Bootloader-optimering

- Här används bootloadern U-Boot. Det finns andra bootloaders som t.ex. Barebox.
- Börja med att stänga av bootdelay, det sparar I detta exempel 2 sekunder.
- Tiden för U-Boot är nu ca 1.14 sekunder istället för 3.13.
- Total tid fram till start av den viktiga applikationen är nu nere i strax under 2 sekunder (ca 1.97).



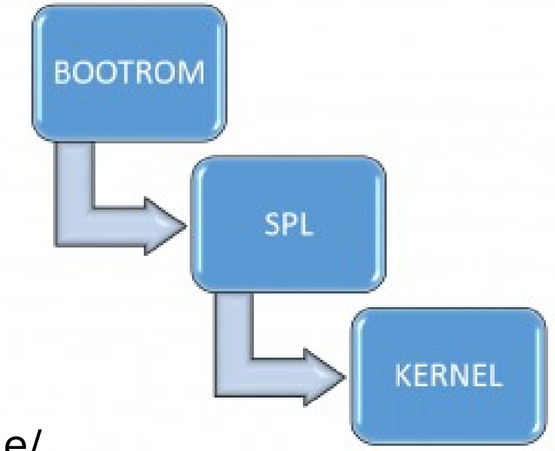
Bootloader-optimering

- Standardboot för att starta embedded Linux är:
  - Bootrom I hårdvaran.
  - Secondary Program Loader (SP
  - U-Boot
  - Linuxkärnan



Bootloader-optimering

- Förenkla konfiguration och bootscript till U-Boot.
- Ta bort funktionalitet som inte behövs.
- Använd U-Boot Falcon mode.
  - Med Falcon mode används bara SPL (Secondary Program Loader) som får ladda Linuxkärnan.
  - Falcon mode kräver en hel del arbete för att få till men ger tidsvinst.
  - Se u-boot-source/doc/README.falcon för detaljer.
  - För Falcon mode med Beaglebone se:
    - U-boot-source/board/ti/am335x/README
    - guiden som finns på https://embexus.com/2017/05/07/fast-boot-linux-with-u-boot-falcon-mode/



MindRoad

Bootloader-optimering – Falcon mode

- Genom att gå in på SPL/TPL I U-Bootkonfigurationen och aktivera falcon mode och deaktivera “Support an environment” minskade tiden ytterligare och den viktiga applikationen startar nu efter ca 1.68 sekunder.

```
[0.000002 0.000002]
[0.000004 0.000012] U-Boot SPL 2018.01 (Jul 22 2020 - 14:03:15)
[0.062749 0.062749] Trying to boot from MMC1
[0.142680 0.079931] reading args
[0.158289 0.015609] reading uImage
[0.160540 0.002251] reading uImage
[1.661616 1.501076] Running /etc/inittab
[1.677279 0.015663] Starting the important application
[1.709246 0.031967] This is the important application!
[3.708604 1.999358] mounting /proc
[3.713983 0.005379] mounting /sys
[3.718576 0.004593] Starting system logger
[3.725697 0.007121] Starting a large application
```

MindRoad

Mer att göra

- Nu har tiden för bootloadern minskat drastiskt. Kvar är att få ner tiden för Linuxkärnan ännu mer.
- Tidigare kunde vi se att bland annat initieringen av konsolen tar lång tid.
- Så om vi kan ändra den viktiga applikationen till att höja en GPIO-pinne och mäta på den istället behöver vi inte konsolen.

MindRoad

Fler saker att titta på

- Bygg toolchainet med stöd för hårdvaruflyttal om hårdvaran har stöd för det.
- Bygg toolchainet så att den genererar Thumb2 för ARM. Detta brukar ge lite mindre binärer.
- Titta på andra standardbibliotek. Uclibc-ng är t.ex. mindre än glibc. Den är dessutom mer konfigurerbar än glibc.
- Se om det går att få ner storleken på root-filsystemet ännu mer.
  - Ta bort filer som inte behövs.
  - Köra strip på så många binärer som möjligt.
- Se om det går att optimera ner storleken på Linuxkärnan ännu mer.
- Går hårdvaran att uppgradera till snabbare hårdvara, t.ex. snabbare lagringsmedia.

MindRoad

Länkar

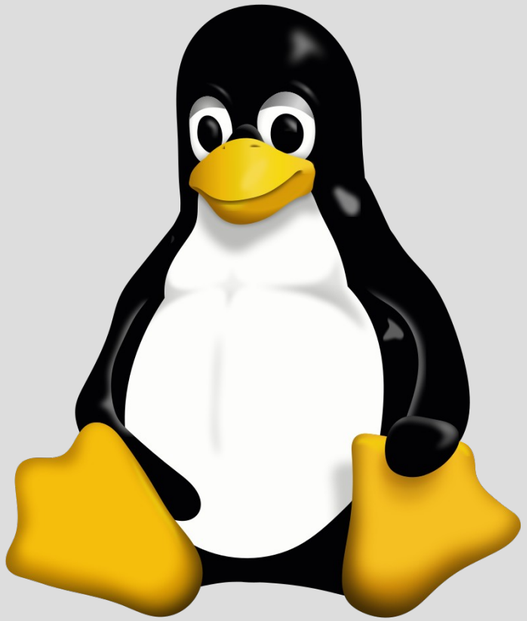
- https://www.elinux.org/Boot_Time
- https://bootlin.com/doc/training/boot-time/
- https://www.youtube.com/watch?v=6sUiDJlJiYw
  Timing Boot Time Reduction Techniques - Michael Opdenacker, Bootlin
- https://www.youtube.com/watch?v=gTzkskAkd8w
  Booting faster with U-Boot Falcon mode - Fabien Lahoudère
- https://www.youtube.com/watch?v=gIK1he6Ocpg
  A Pragmatic Guide to Boot-Time Optimization - Chris Simmonds, Consultant
- https://www.youtube.com/watch?v=Um7jlqNzjL8
  Boot-Time Optimization for the Real World (Embedded Linux Conference Europe 2020)

MindRoad

Länkar - presentationer

- https://elinux.org/images/d/d1/Alexandre_Belloni_boottime_optimizations.pdf
- https://elinux.org/images/9/97/Boot_one_second_altenberg.pdf
- https://elinux.org/images/6/64/Chris-simmonds-boot-time-elce-2017_0.pdf
- https://elinux.org/images/5/5d/Systemd-csimmonds-elce-2019.pdf

- Presentationen finns på:
  https://www.lysator.liu.se/~kjell-e/tekla/linux/dokument/linux-bootoptimering.pdf



Frågor?