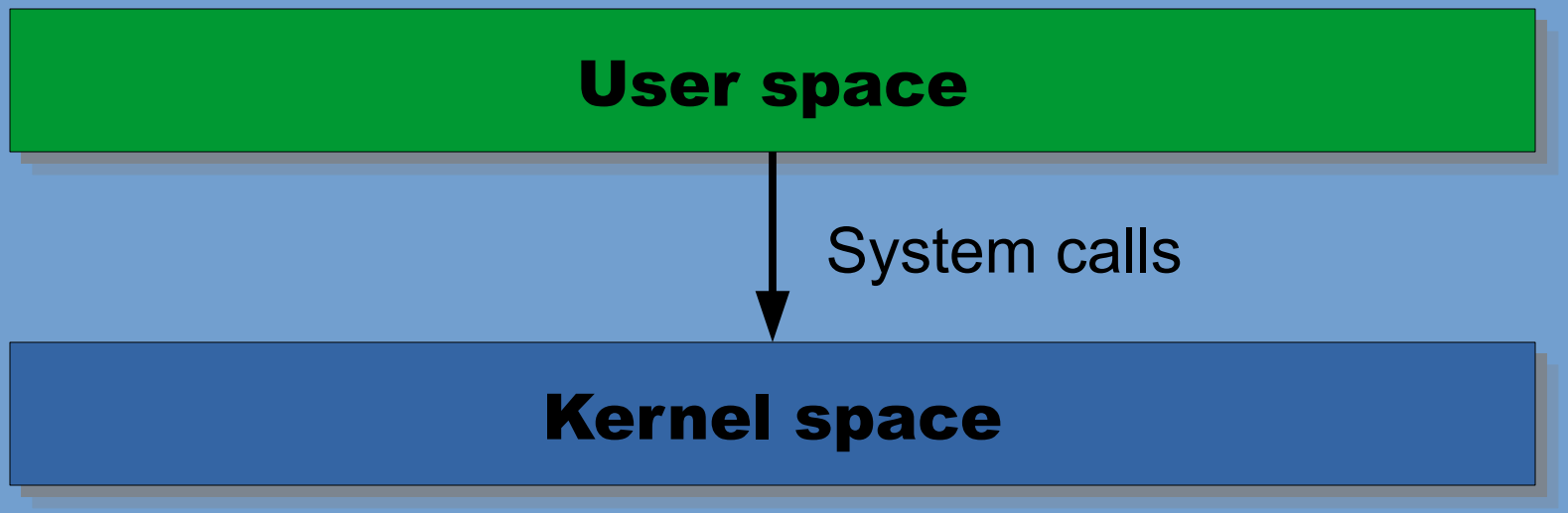
Inleiding Practicum Operating Systems

Mattias Holm & Kristian Rietveld

Universiteit Leiden
The Netherlands

Tot nu toe

- Shell: verkennen interface tussen user-space en kernel-space.



Doel van deze presentatie

- In de komende 3 practicumopdrachten zullen we gaan werken met een speciaal geschreven operating system.
- Kort introduceren van:
  - Hardware.
  - Kernel.
  - Tools.
  - De opdrachten.

Hardware

Hardware

- We hebben gekozen om te werken met de ARM architectuur.
- Waarom?
- Iets anders dan een “normale computer”.
- Er zijn meer ARM chips actief dan Intel chips.

Waar wordt ARM gebruikt?

- Feature phone / smart phone / tablet.
- Televisies / set-top boxes.
- Huis/tuin/keuken routers.
- Home/Small-Business NAS.
- Auto's.
- Embedded / microcontrollers.
- Etc.

ARM platforms

- Het bedrijft achter ARM verkoopt zelf geen chips, alleen licenties.
- Chips verschijnen in platforms:
  - NVidia Tegra
  - TI OMAP
  - Apple A4, A5, A5X, …
  - Samsung Exynos
  - Qualcomm Snapdragon
  - Etc.

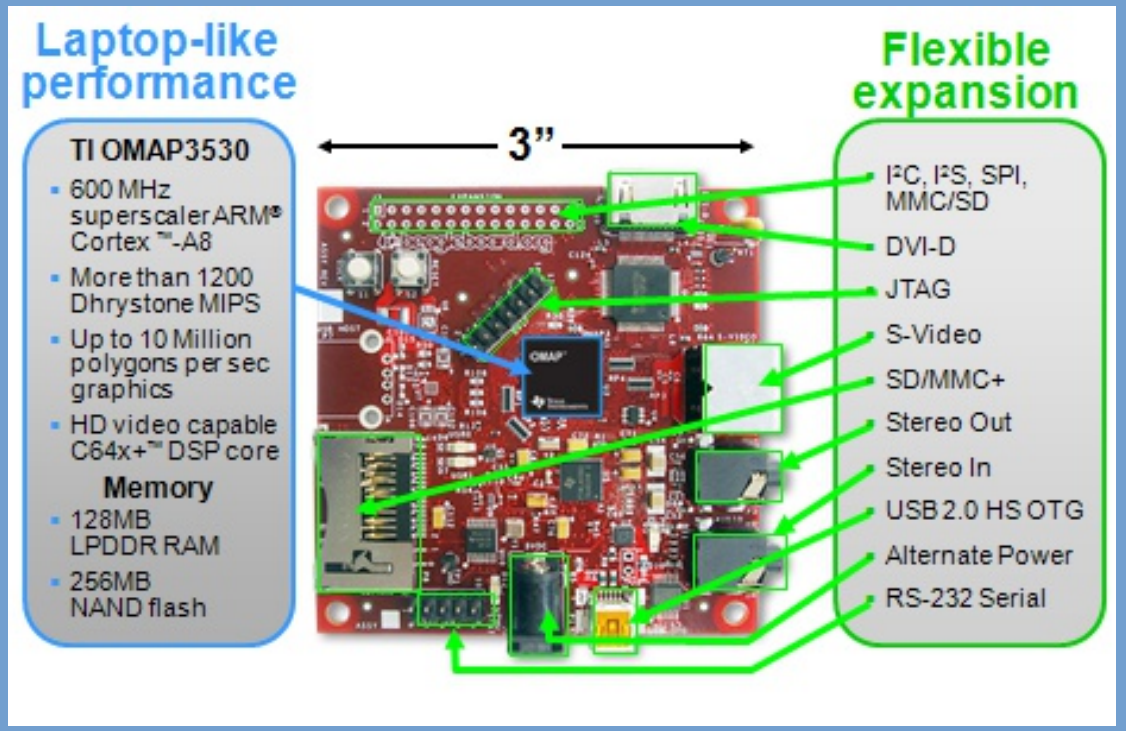
TI OMAP3

- Onder andere gebruikt in:
  - Nokia N900, Nokia N9
  - Palm Pre
  - Motorola Droid
  - Nook color
- (OMAP2: N800, N95).

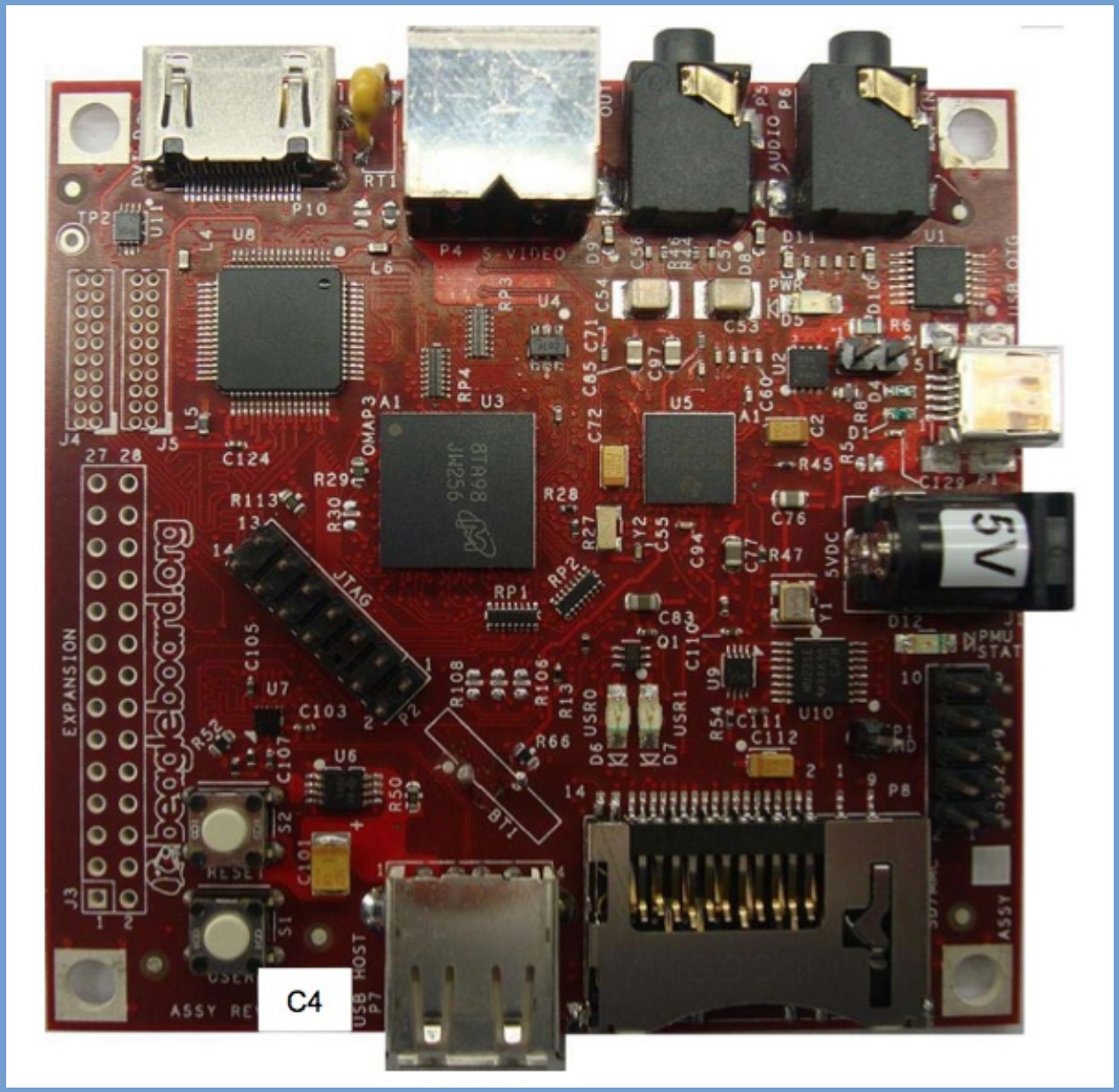
Bron: http://www.n9nokia.nl/

BeagleBoard C4

- OMAP3530
- ARM Cortex-A8 (600MHz)
- 256MB RAM
- 256MB Flash
- PowerVR
- SD card slot



Bron: http://beagleboard.org/hardware


C4

De kernel

Hoe boot het systeem?

- Twee bootloaders op de interne flash chip:
  - 1st stage: X-loader
  - 2nd stage: u-boot
  - U-boot kan kernel in het geheugen laden vanaf flash of SD card en jumpt naar de kernel.

Hoe boot het systeem?

- In ons geval:
  - De SD kaart heeft twee partities: 1 FAT, 1 WFS.
  - U-boot laadt kernel.bin van de FAT partitie op de SD kaart in het geheugen.
  - U-boot springt naar begin kernel.
  - Kernel mount root file system (WFS partitie) en start init.

Communicatie met systeem

- Geen USB stack; dus geen USB toetsenbord of muis.
- Framebuffer output (HDMI) werkt wel maar zeer gelimiteerd (geen fonts).
- I/O geschiedt via serial port (UART).

SMACK kernel

- Features:
  - 32-bit, memory protection.
  - (Pre-emptive scheduling).
  - VFS (FAT16, WFS, DevFS).
  - User-space; ELF binaries.
  - SD kaart driver.

Tools

Kernel compileren

- We doen development op een Intel machine, maar willen compileren naar ARM code.
- Dit wordt gedaan dmv een “cross-compiler”.
  - Vergelijk de Alpha en RISC-V compilers gebruikt bij Computer Architectuur.
- We maken gebruik van gcc, target “arm-none-eabi”, dit geeft ons de “arm-none-eabi-gcc”.

Kernel compileren (vervolg)

- Om te compileren maar drie commando's nodig.

```
mkdir build
cmake \
-DCMAKE_TOOLCHAIN_FILE=../config/arm-none-eabi-gcc.cmake \
-DBSP=bsp/beagle.cmake ..
make
```

Bestanden kopiëren

- `build/kernel.bin` moet worden geplaatst op de FAT partitie van SD kaart.
  - Of fysiek, of op sdcard.img.
- User-space utilities moeten op de WFS partitie worden geplaatst.
- FAT partitie SD kaart bevat uiteindelijk:
  - `kernel.bin`
  - `boot.scr`

Bestanden kopiëren (verv.)

- Doe dit een paar keer “met de hand”.
- Schrijf er daarna een shell script voor.

```
#!/bin/sh
mcopy -o -i sdcard.img@@1048576 kernel/build/kernel.bin ::

wfstool sdcard.img@@5242880 put kernel/build/apps/ls ls
```

Starten emulator

```
qemu-system-arm -M beagle \
  -drive file=/data/software/beagle-nand.bin,if=mtd,format=raw \
  -drive file=sdcard.img,if=sd,format=raw -s -nographic
```

Starten hardware

- Sluit serial line aan.
- Plaats de SD kaart.
- Ga na dat minicom draait.
- Power on.

Debuggen

- Het makkelijkst is om “printf-debugging” toe te passen.
- Mocht je al ervaren zijn met gdb, je kan gdb aan qemu koppelen:

```
target remote localhost:1234
file /path/to/kernel.elf
```

- **Opgelet**: gebruik de ARM gdb! “arm-none-eabi-gdb”.

Installeren software

- In zaal 411 is alle software geïnstalleerd en hebben we volledige flexibiliteit (ook voor aansluiten hardware).
- In zalen 302/304/306/308/huisuil (NUWD) is ook alle software geïnstalleerd. Hardware aansluiten is hier echter niet mogelijk.
- Zelf installeren op Linux (of Mac) laptop met behulp van Installation Guide.
- Virtual Machine images met installatie zoals die in zaal 411 worden nog beschikbaar gemaakt in VirtualBox formaat.

Overzicht opdrachten

Overzicht opdrachten

- Opdracht 2: Processes
  - Deadline: 1 april
- Opdracht 3: Virtual Memory Management
  - Deadline: 29 april
- Opdracht 4: File Systems
  - Deadline: 20 mei

Processes

- Startpunt: kernel met First Come First Serve scheduler.
- Te implementeren:
  - Round Robin scheduler.
  - Multi-Level Feedback Queue scheduler, bestaande uit FCFS en RR.
  - Bijhouden “wait time” per proces.
- Vervolgens voer je een aantal experimenten uit om de performance van de verschillende schedulers te bestuderen.

Virtual Memory

- (Onder voorbehoud)
- Startpunt: er wordt 2 MB aan aaneengesloten geheugen gealloceerd voor de page table voor **elk** proces.
  - Dit komt omdat alle Level2 page tables vooraf worden gealloceerd.
  - Dit beperkt het aantal processen dat we gelijktijdig in het geheugen kunnen hebben.
- Doel: pre-allocatie vervangen met dynamische allocatie. We krijgen dan echte “paged” page tables.

File Systems

- Startpunt: incomplete implementatie WFS.
- Te implementeren:
  - Write support.
  - Full subdirectory support (read, create, remove).

Algemeen

- Begin op tijd aan de opgaven!
- Als er problemen zijn, laat ons dat zo snel mogelijk weten.
- Ga er absoluut niet vanuit dat je alles in de werkcolleges afkrijgt.

Practicum

- Korte koffiepauze :)
- Daarna start practicum zalen 306/308, zaal 411.