Inleiding Practicum Operating Systems

Mattias Holm & Kristian Rietveld

Universiteit Leiden
The Netherlands

Doel

- In komende 3 practica zullen we gaan werken met een custom OS.
- 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.
- 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
  - Etc.

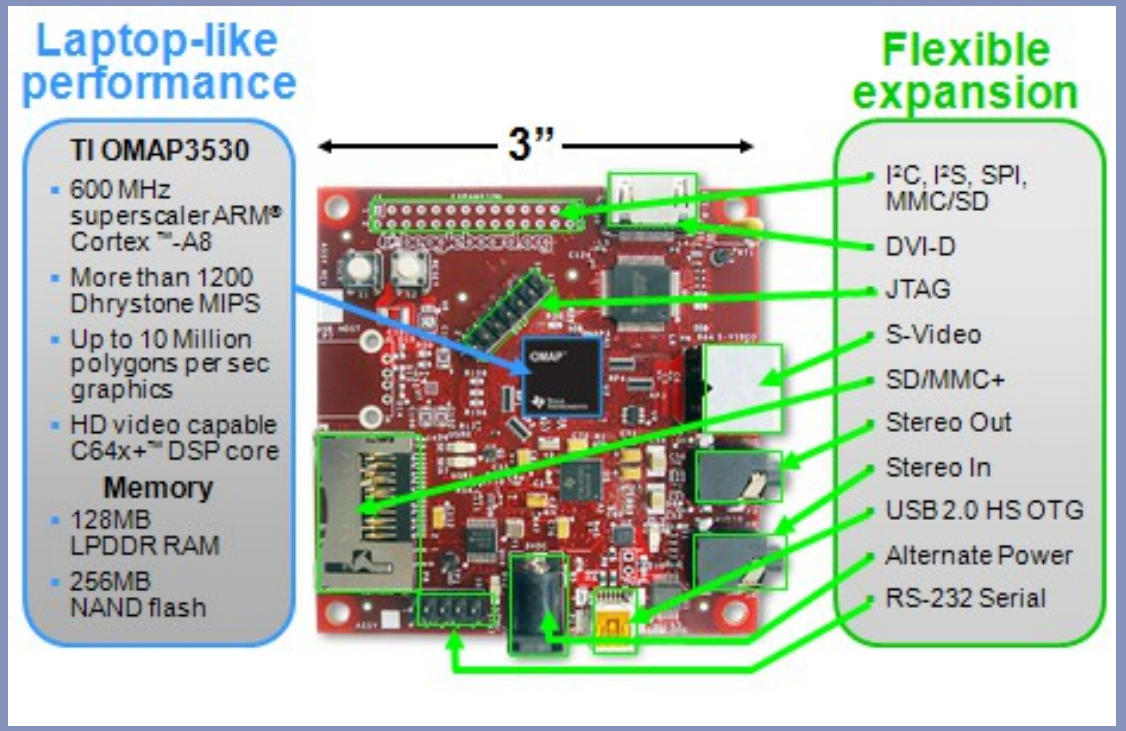
TI OMAP$_3$

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

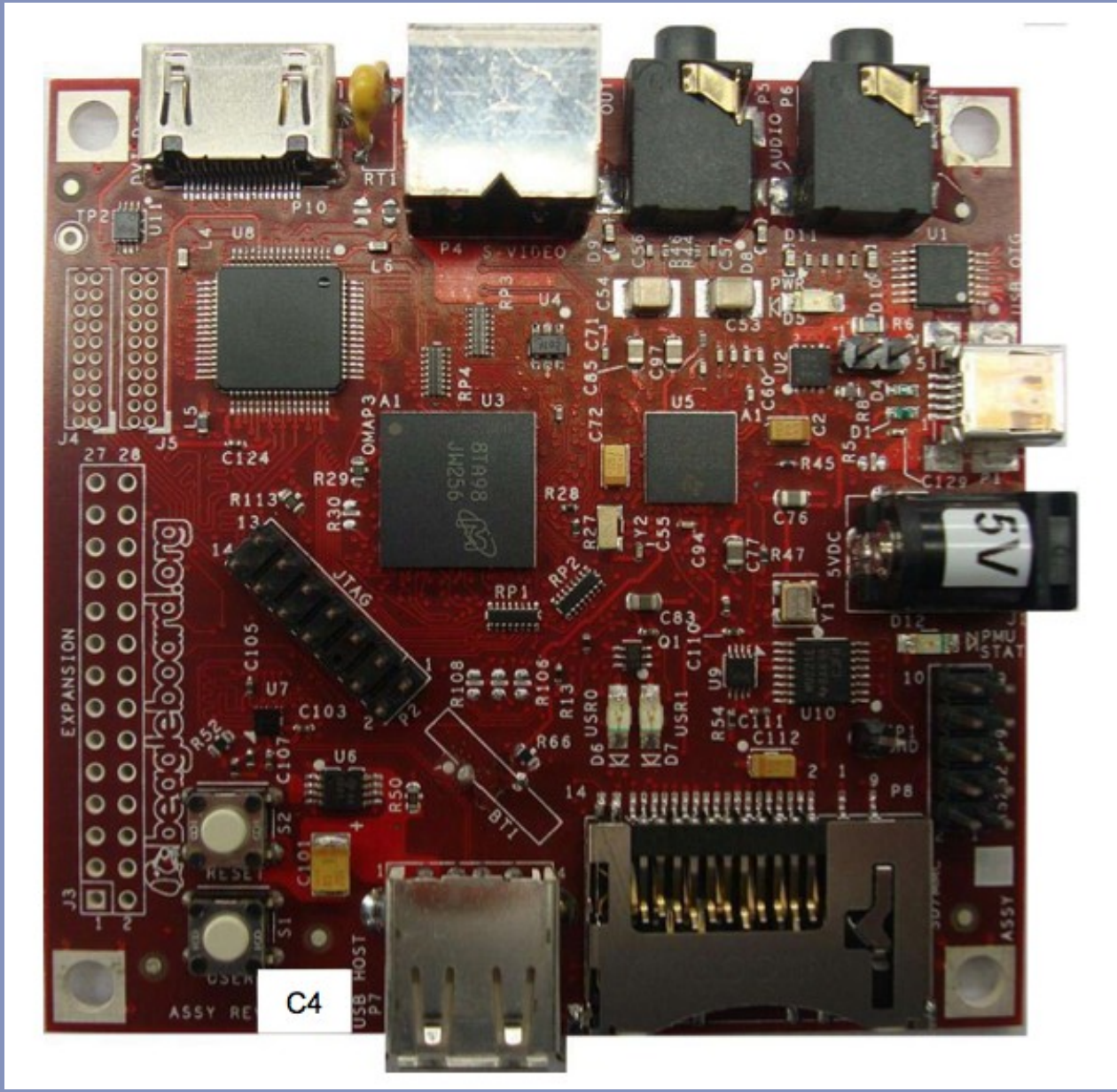
Bron: http://www.n9nokia.nl/

BeagleBoard C4

- OMAP3530
- ARM Cortex-A8 (600MHz)
- 128MB RAM
- 256MB Flash
- PowerVR



Bron: http://beagleboard.org/hardware


C4
beagleboard.org
EXPANSION
JTAG
RESET
USB HOST
S-VIDEO
AUDIO
5VDC
OMAP3
JM256

De kernel

Hoe boot het systeem?

- Twee bootloaders op de 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:
  - U-boot laadt kernel.bin van SD kaart in het geheugen.
  - U-boot laadt ramdisk.img van SD kaart in het geheugen.
  - U-boot springt naar begin kernel.

SMACK kernel

- Features:
  - 32-bit, memory protection.
  - Pre-emptive scheduling.
  - VFS (FAT16, WFS, DevFS).
  - User-space; ELF binaries.
- Nog geen disk I/O.

Tools

Kernel compileren

- We doen development op een Intel machine, maar willen compileren naar ARM code.
- Dit wordt gedaan dmv een “cross-compiler”.
- Op Linux gebruiken we de CodeSourcery compiler.

Kernel compileren (vervolg)

- Om te compileren maar drie commando's nodig.

```
mkdir build
cmake \
-DCMAKE_TOOLCHAIN_FILE=../config/arm-elf-sourcery.cmake \
-DBSP=bsp/beagle.cmake ..
make
```

Bestanden kopiëren

- `build/kernel.bin` moet worden geplaatst op de SD kaart.
  - Of fysiek, of op sdcard.img.
- User-space utilities moeten op `ramdisk.img` worden geplaatst; deze moet weer op de SD kaart terecht komen.

Starten emulator

```
../qemu/arm-softmmu/qemu-system-arm -M beagle -mtdblock
beagle-nand.bin -sd sdcard.img -serial stdio -s
```

Starten hardware

- 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
```

Overzicht opdrachten

Overzicht opdrachten

- Opdracht 1: Process Scheduling
  - Deadline: 2 april
- Opdracht 2: Virtual Memory Management
  - Deadline: 30 april
- Opdracht 3: File Systems
  - Deadline: 21 mei

Process Scheduling

- Startpunt: kernel met First Come First Serve scheduler.
- Te implementeren:
  - Round Robin scheduler.
  - Multi-Level Feedback Queue scheduler.
  - Earliest Deadline First scheduler.

Process Scheduling

- Bonusopdracht voor wie zich voelt uitgedaagd.
- Implementeren context switching voor FPU codes.

Virtual Memory

- Startpunt: suboptimaal VM systeem:
  - 1 page descriptor per page.
  - Linked list of free/used pages.
- Te implementeren:
  - Beter page allocatie systeem met “regions”.
  - Vergelijkbaar wat gebeurt in Linux.

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 niet vanuit dat je alles in de werkcolleges afkrijgt.

Practicum

- Practicum vanmiddag in zaal 411.