Embedded Linux on SoC: Comparison Intel - Xilinx

Embedded Computing Conference

5. September 2017

www.zhaw.ch/ines

Matthias Frei (frma@zhaw.ch)

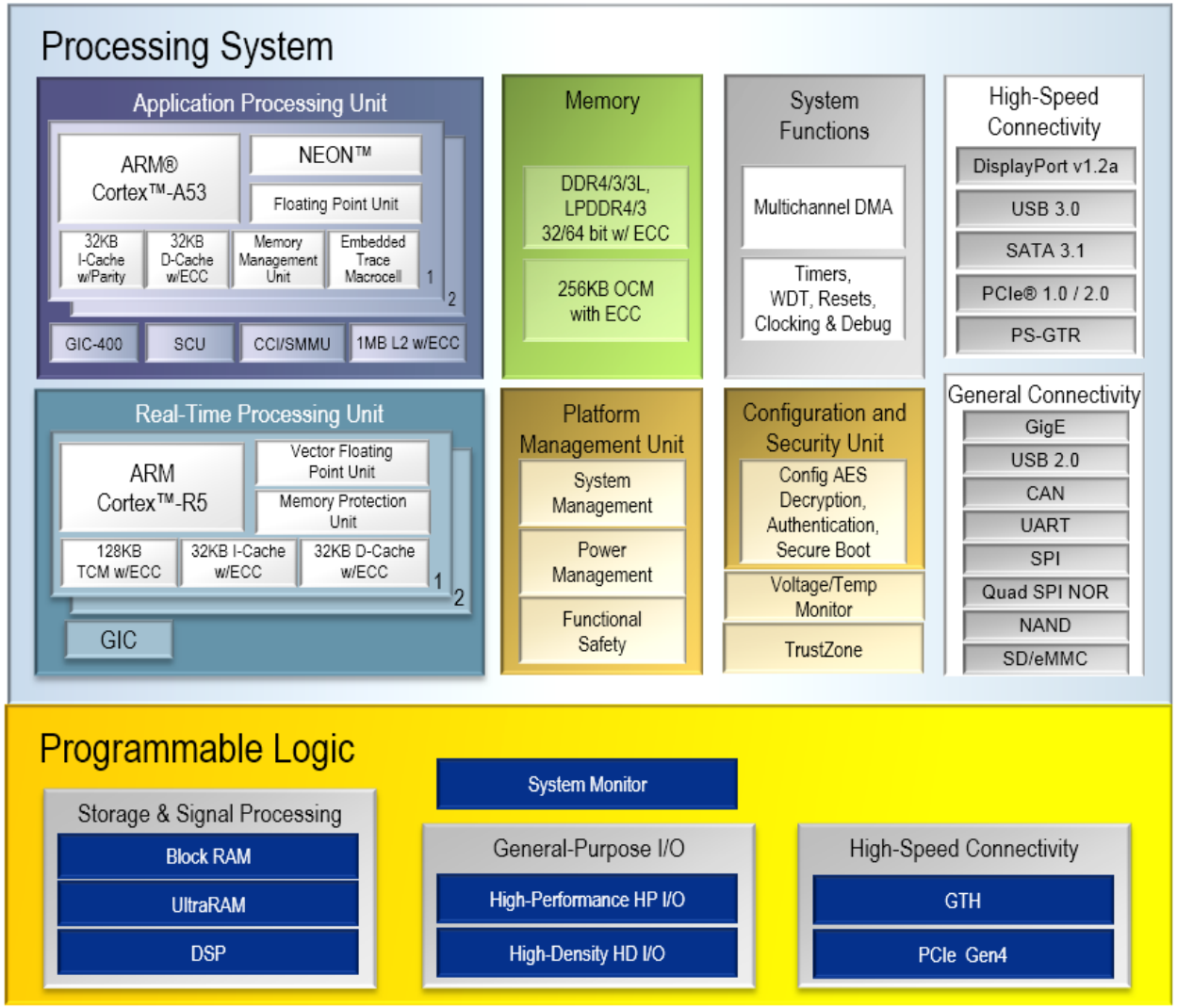
System on Chip Design Group

ZHAW Institute of Embedded Systems (InES)

Agenda

1. Was ist Yocto
2. Creating a Linux for Intel and Xilinx
3. Scheduling Example

Beispiel Zynq Ultrascale+



- Bis zu 4xA53
  - Taktrate bis 1,5GHz
- 2xR5
  - Taktrate bis 600MHz

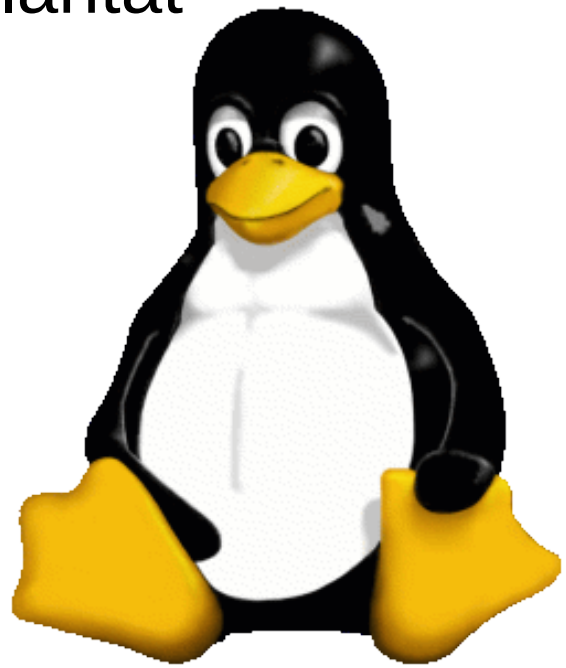
Source: www.xilinx.com

Ausgangslage

- Moderne SoCs haben nebst Logik auch meist einen oder mehrere Prozessoren integriert
- Operating system of choice:
  - Embedded-Linux
  - RTOS
  - Bare Metal (No Operating System)

Embedded-Linux

- Linux ist ein weit verbreitetes Betriebssystem
- Dank leistungsfähiger SoCs mit wachsender Popularität
- Zwei Ansätze für die Implementierung
  - Open Source Lösung
  - Kommerzielle Lösung



Hürden

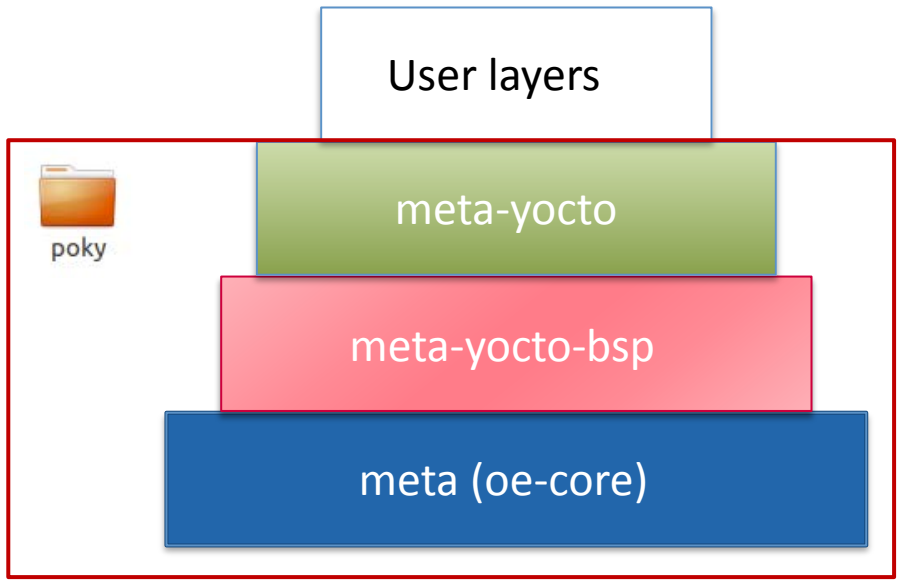
- Grosse Auswahl von ARM basierter Hardware am Markt
  - Xilinx SoCs (FPGA based SoC)
  - Intel-FPGA SoC (FPGA based SoC)
  - Raspberry-Py
  - Beagle Bone
  - Nvidia-Tegra Serie
- Anpassungen je nach Peripherie/Hardware nötig

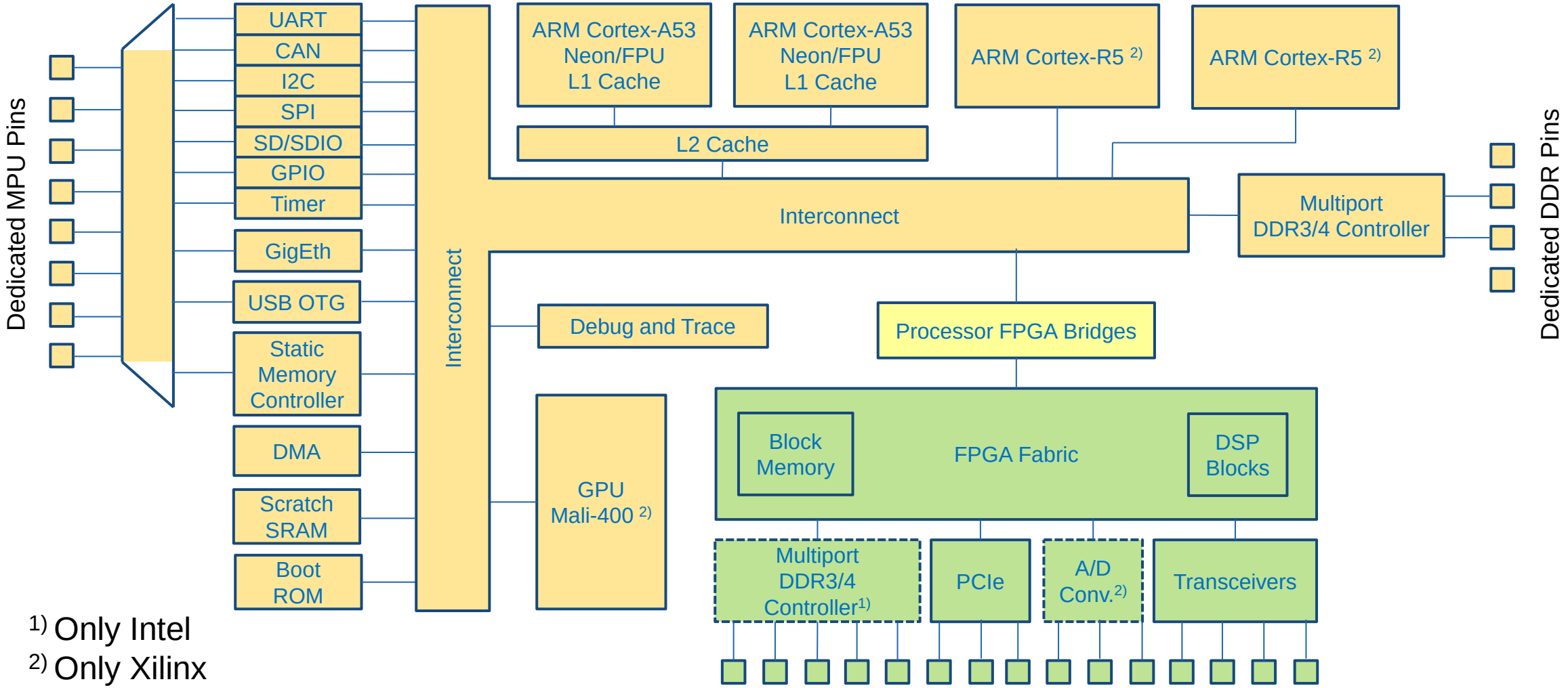
Was ist Yocto

- Yocto bildet ihr «eigenes» anwendungsspezifisches Linux
- Treiber und Software werden gebildet mittels «Kochrezepten» in Form von «Bitbake Files» (.bb)
- Schichtweiser Aufbau der Kochrezepte in Form von Layern

Konzept

- Poky ist die Kernkomponente von Yocto
- Poky ist sowohl reference als auch build system





1) Only Intel
2) Only Xilinx

Boot Optionen

SD-Karte (in diesem Beispiel)

Source: www.enclustra.com

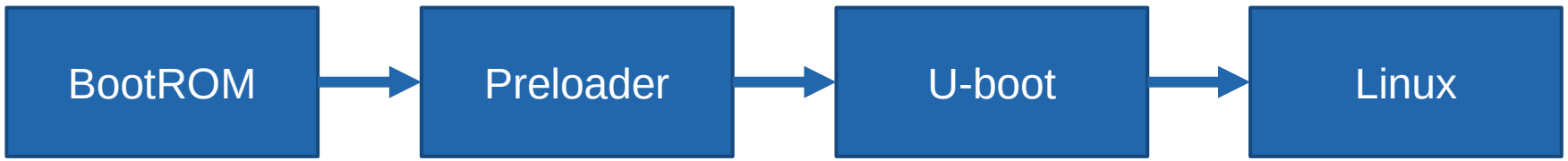
QSPI(Flash)

JTAG

Source: www.fpgadeveloper.com

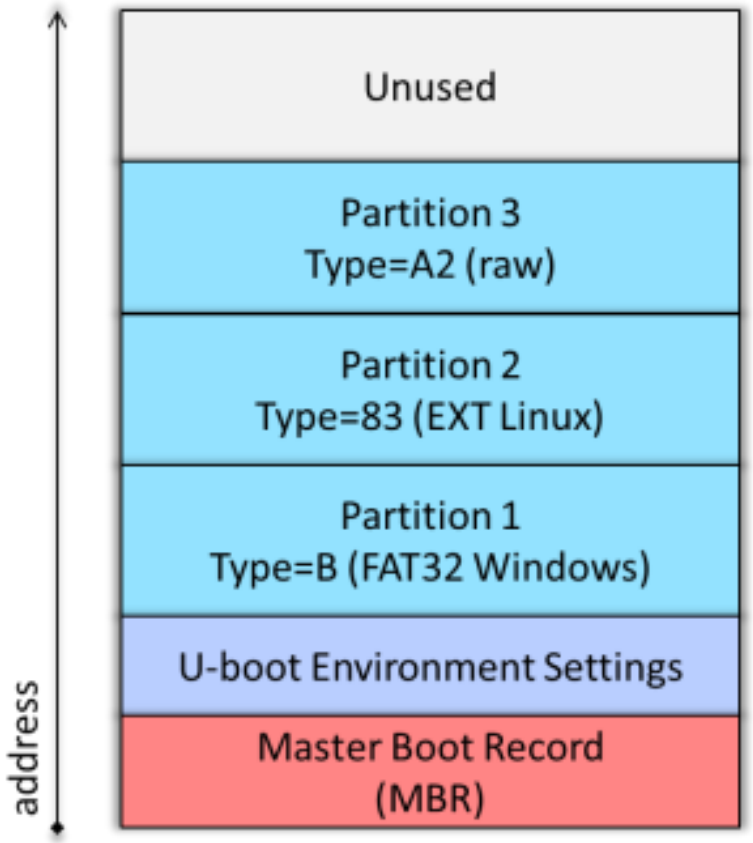
Beispiel Bootsequence auf SoC-FPGA

- BootROM
- First Stage Bootloader - Preloader
- Second Stage Bootloader - U-boot
- Linux



SD-Karten Image Intel-FPGA (Altera)

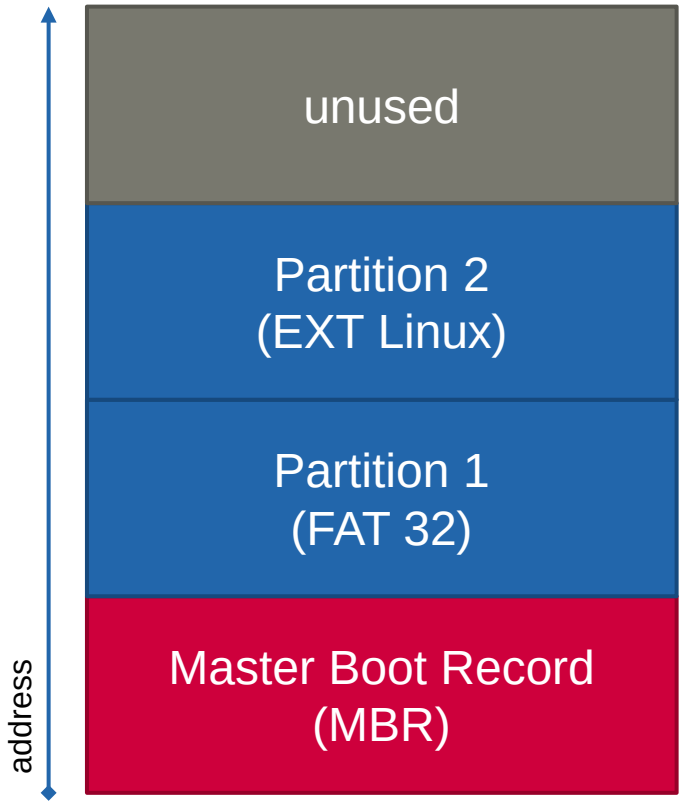
- Partition 1
  - Device Tree
  - FPGA Image
  - U-boot scipt für FPGA konfiguration
  - Kernel Image
- Partition 2
  - Root filesystem
- Partition 3
  - Preloader image
  - U-Boot image



Source: www.rocketboards.org/foswiki

SD-Karten Image Xilinx

- Partition 1
  - Boot.bin
    - Preloader
    - U-Boot
    - FPGA Image
  - Device Tree
  - Kernel Image
- Partition 2
  - Root filesystem



Intel-FPGA (Altera) vs Xilinx

Intel

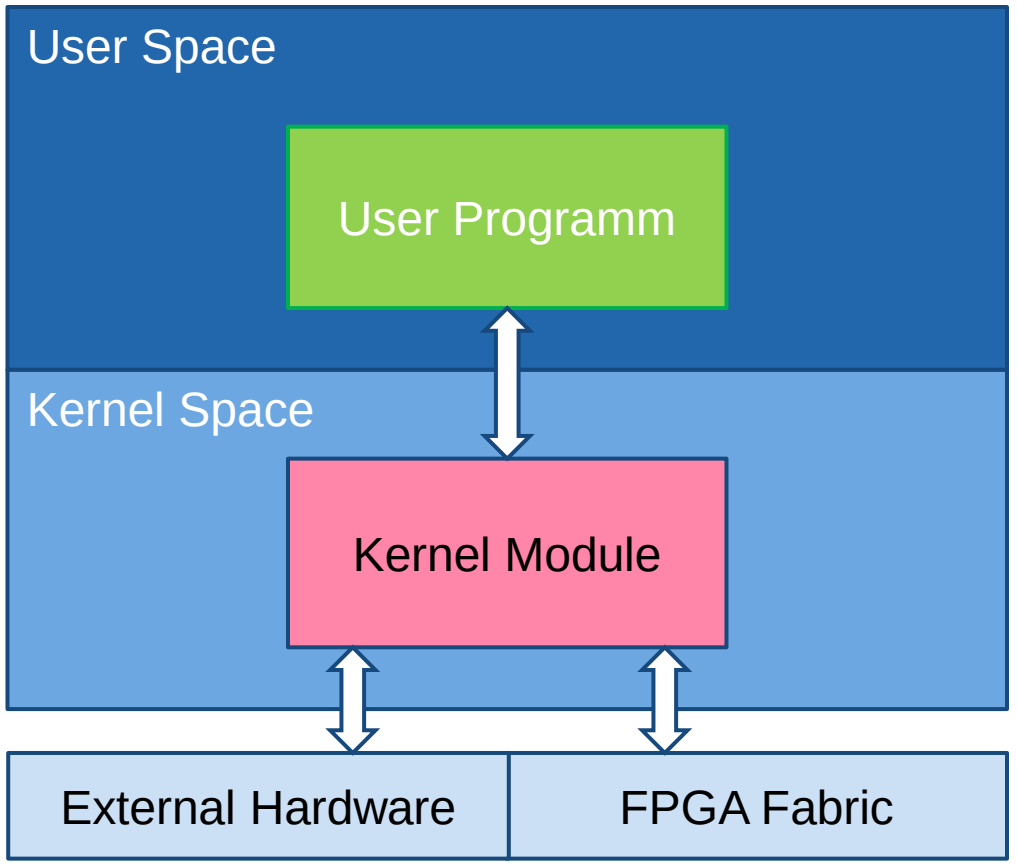
- Yocto
- Befehle:
  - bitbake virtual/bootloader
  - bitbake virtual/kernel
  - bitbake core-image-minimal

Xilinx

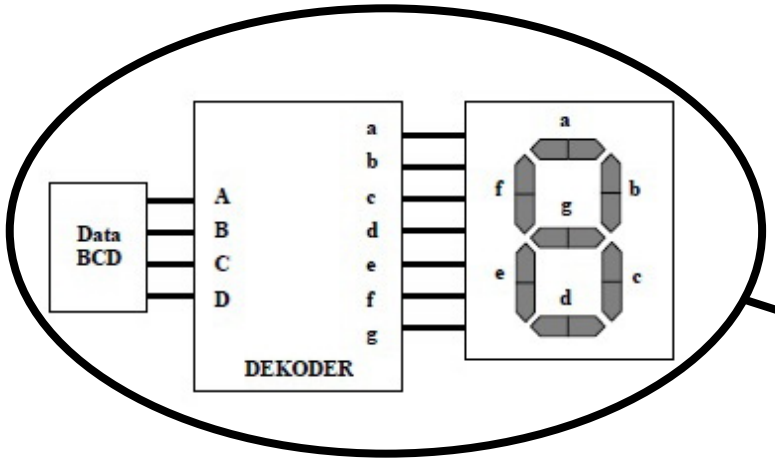
- Peta-Linux (basierend auf Yocto)
- Befehle:
  - petalinux-create
  - petalinux-config
  - petalinux-build
- Boot.bin File erstellen
  - petalinux-package --boot --fpga fpga.bit --u-boot --kernel

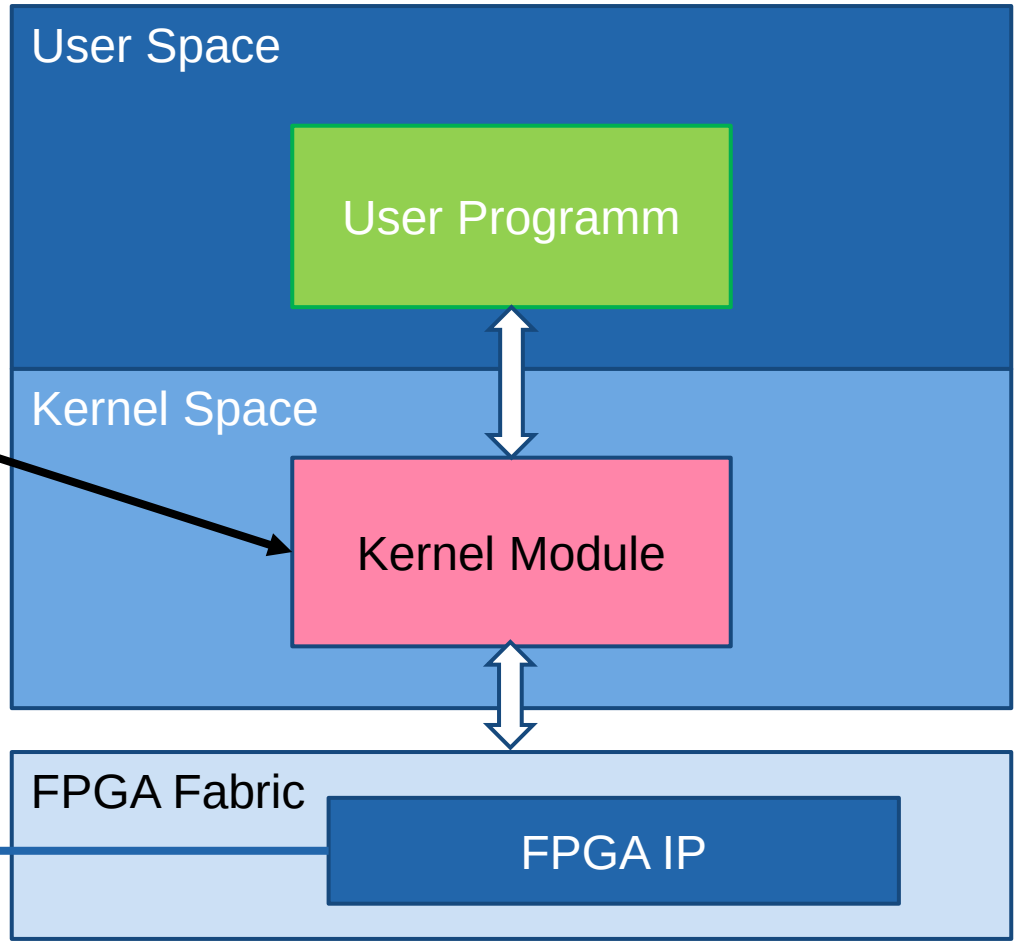
Design Spezifische Anpassungen

- Schnittstelle zur Hardware
- Kernelmodul: Treiber



Beispiel eines einfachen Kernel Moduls

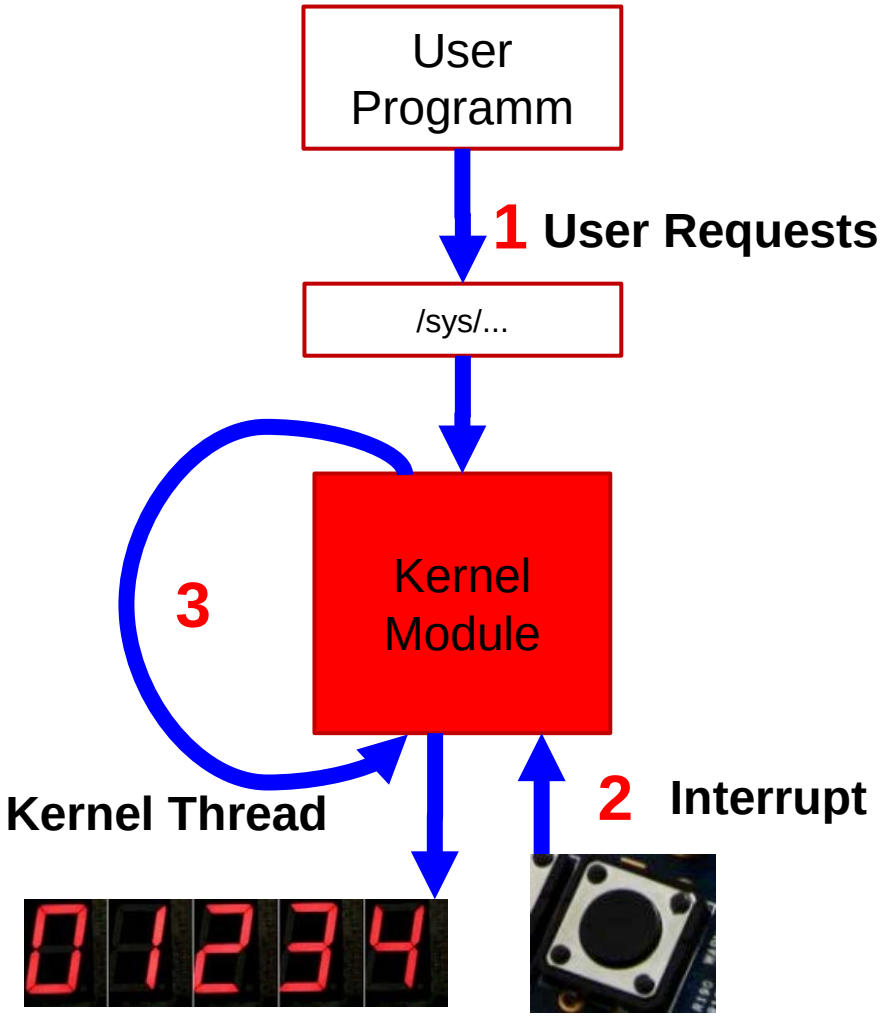




7-Segment Display

Beispiel eines Kernel Moduls

- 1: User Requests Handler
- 2: Hardware Request Handler
  - React on Hardware Requests
    → **Interrupt Handler**
- 3: Iterative Tasks
  - Kernel can handle periodic tasks
    → **Kernel Thread**



Hinzufügen von Modul bei Intel-FPGA (Altera)

Intel

- Modul in build/config/local.conf hinzufügen
- IMAGE_INSTALL_append = <MODULENAME>
- bitbake <MODULENAME>

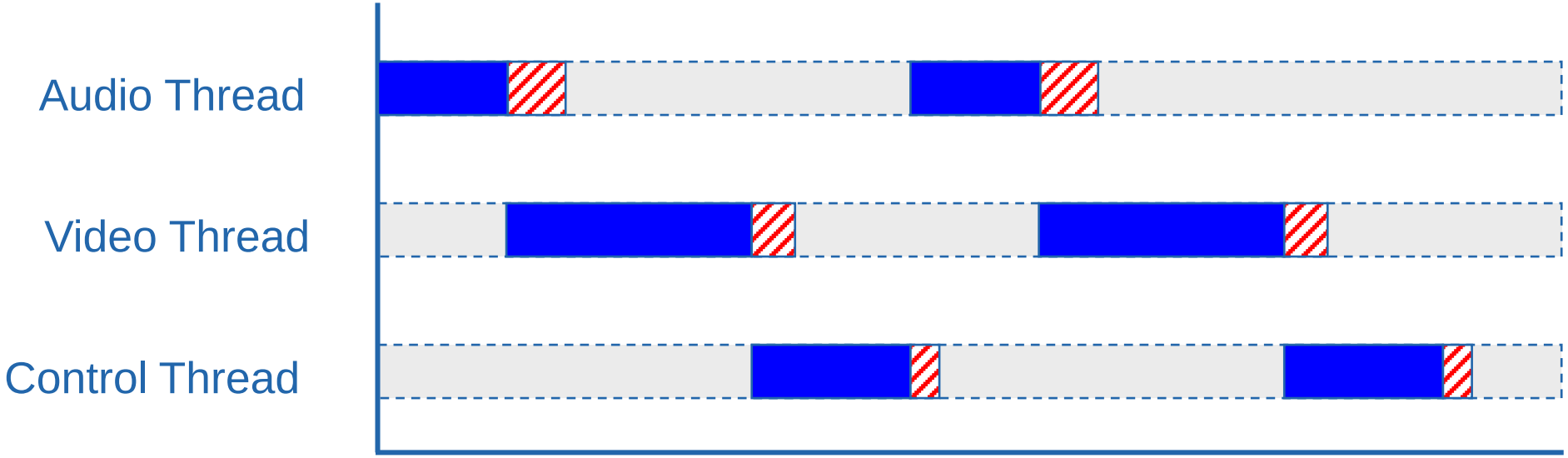
Xilinx

- petalinux-create --type modules --name <MODULENAME> --enable
- petalinux-build -c <MODULENAME>

Vergleich Xilinx und Intel

Task	Intel	Xilinx
FPGA Bitstream erstellen	Quartus	Vivado
BSP generieren	BSP-Editor	PetaLinux / SDK
Devicetree blob	Sopc2dts	PetaLinux / SDK
Kernel image	Yocto	PetaLinux
U-boot image	Yocto	PetaLinux
Pre loader	Yocto	PetaLinux / SDK
Root filesystem	Yocto	PetaLinux
Boot image	Yocto	PetaLinux / SDK

Time-Sliced A/V Application



- Problem:
  - 3 Tasks (Audio, Video, Control) use >100% Processing
- SCHED_OTHERS
  - No Task finishes in time
  - Changing Nice Levels is unpredictable!

Time-Sliced A/V Application

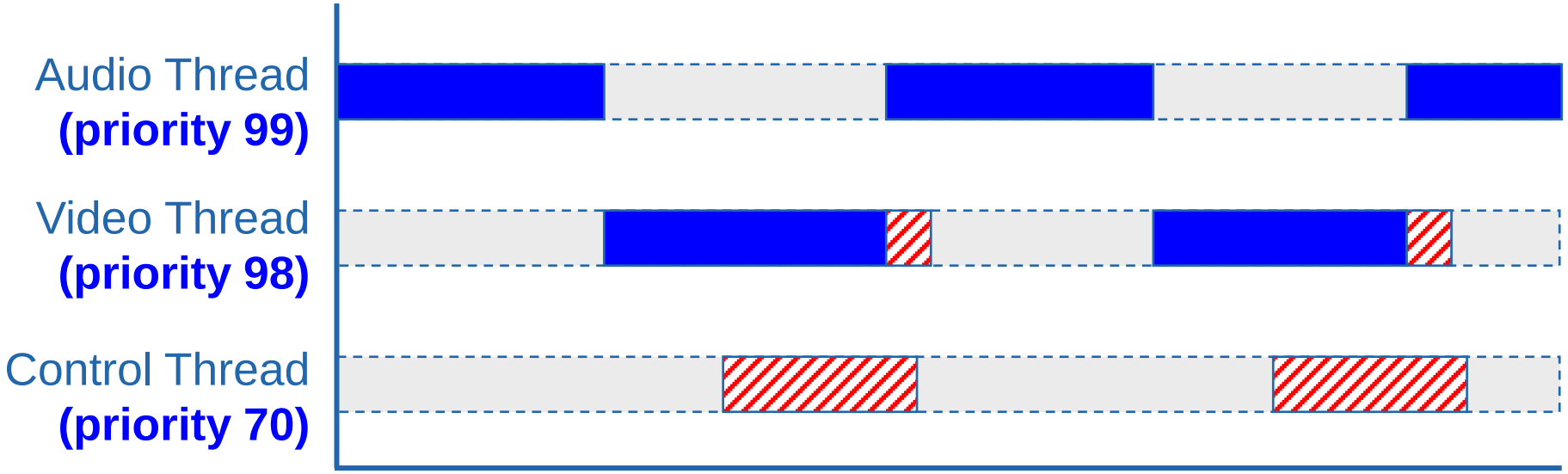
Audio Thread  Audio thread completes 80% of samples

Video Thread  Video thread drops 6 of 30 frames

Control Thread  User response delayed 1mS

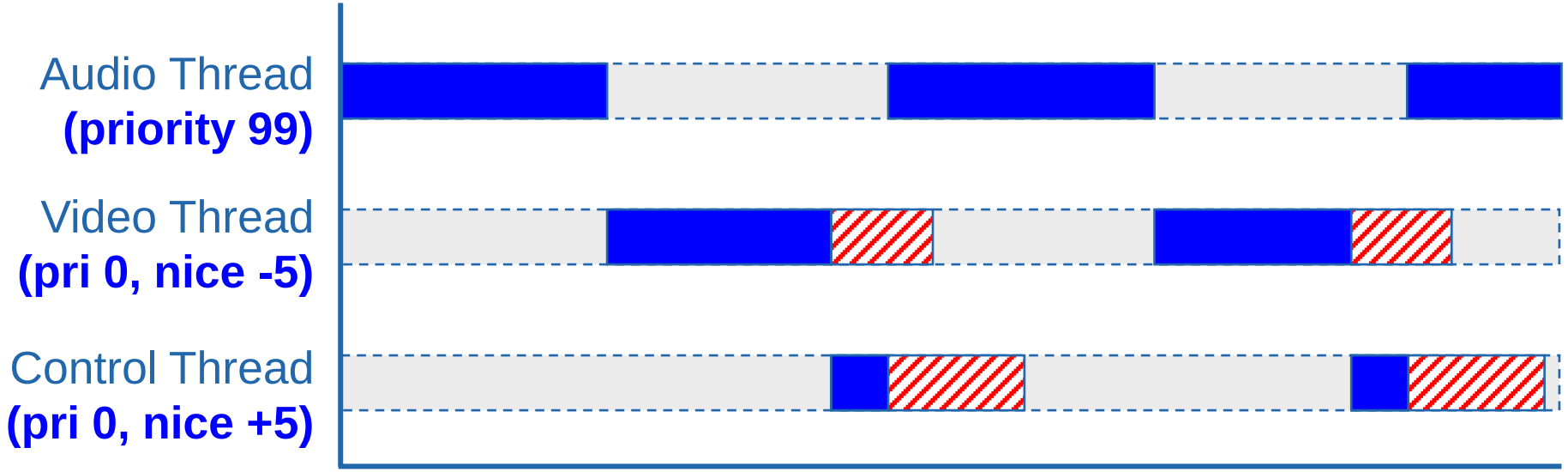
- All threads suffer, but not equally:
  - Audio samples are lost: Highly perceptible
  - Video frames are lost: Slightly perceptible
  - Control thread not finished: System a bit slower remotely perceptible
  - Note: Time-slicing may also cause real-time failure in systems that are <100% loaded due to increased thread latency

Time-Sliced A/V Application



- SCHED_FIFO
  - Audio thread is guaranteed the bandwidth it needs (P99)
  - Video thread takes the rest (P98)
  - Control thread never runs!

Time-Sliced A/V Application



- SCHED_FIFO and SCHED_OTHER
  - Audio thread is guaranteed the bandwidth it needs (P99)
  - Video thread takes most of remaining bandwidth (Nice -5)
  - Control thread gets a small portion of remaining bandwidth (Nice +5)

Time-Sliced A/V Application

Audio Thread		Audio thread completes, no distortion
Video Thread		Video thread drops 2 of 30 frames
Control Thread		User response delayed 100ms

- Compromise:
  - Audio thread completes as desired
  - Video thread failure is barely perceptible
  - Control thread delayed response is acceptible
  - → **Graceful degradation**

Zusammenfassung

Kontakt

https://blog.zhaw.ch/high-performance

frma@zhaw.ch