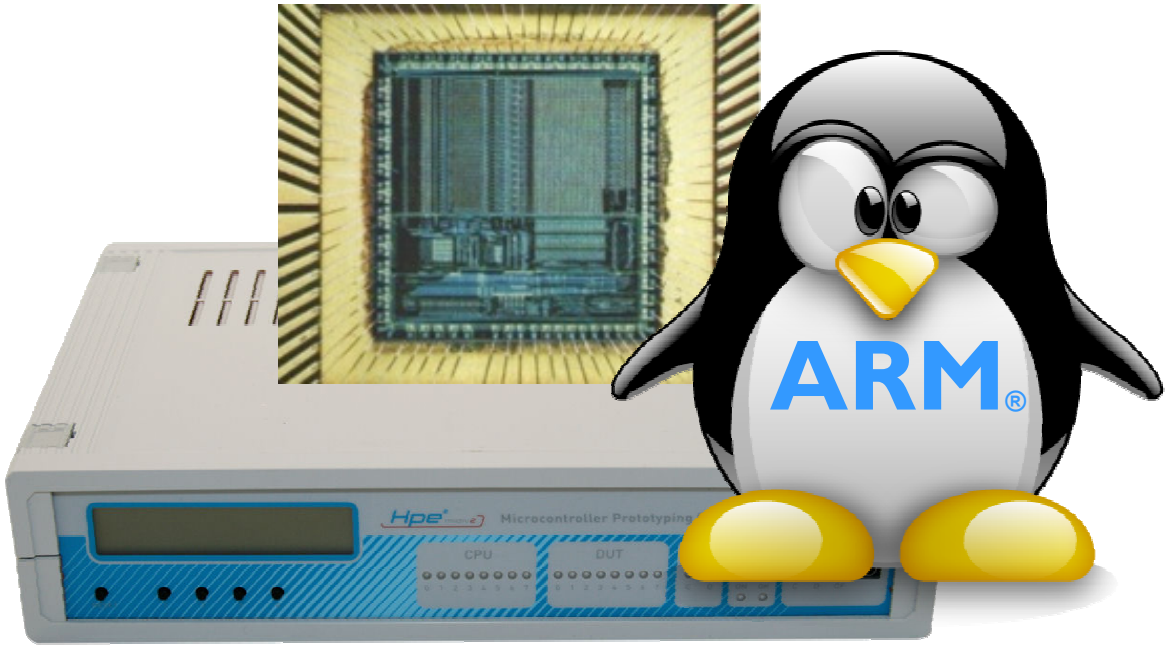
µClinux auf dem ARM Cortex-M4!?

(Eine Kosten- und Nutzenanalyse)



Masterthesis an der FH Emden/Leer und Doulos Ltd.

Student	Frank Mölendörp
Betreuer bei Doulos	Jens Stapelfeldt
Betreuer bei der FH Emden/Leer:	Prof. Dr. –Ing. Gerd von Cölln

Inhalt

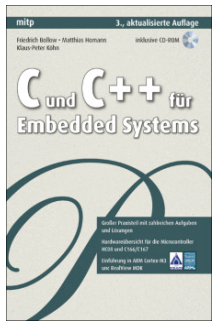
- Was ist das MPS?
- Was ist uClinux?
- Sinn und Zweck dieser Arbeit
- Was wurde erreicht?
- Wissenswertes
- Vorführung
- Kosten-Nutzen-Analyse
- Fazit

Doulos and Standards

Doulos is a Training and Know-How provider in electronics for over 18 years and has been involved with standards form day 1 !

VHDL, SystemC, SystemVerilog, C++, OVM……, ARM, CMSIS

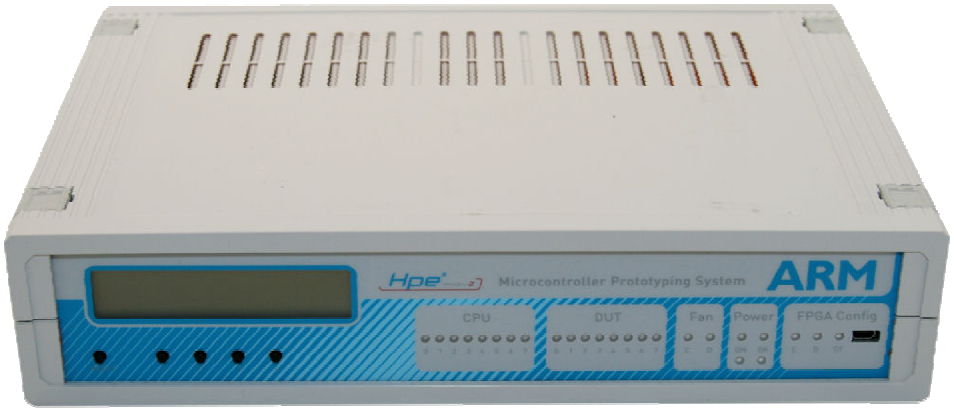
- Doulos is has delivered Training to over 800 companies and in 36 countries world-wide.
- Public classes are scheduled in regularly Europe and the USA.
- Doulos has been an ATC (Approved Training Centre) for over 10 years
  - Free resources in our Know-How section on the web
  - See www.Doulos.com/KnowHow
    - Tutorial: Getting started with CMSIS
    - German book with nice Cortex-M3 Intro and Examples !



Was ist das MPS?

Das MPS …

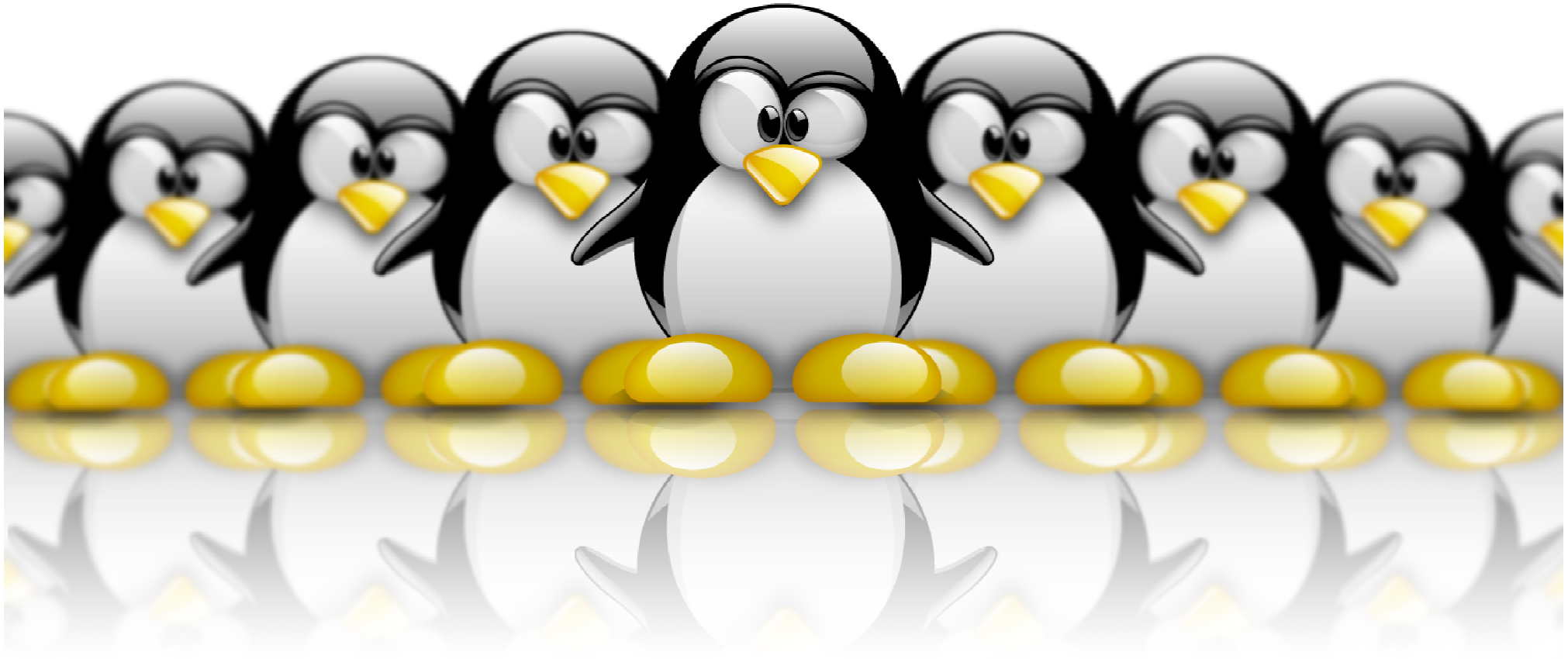
- … steht für Microcontroller Prototyping System.
- … ist ein FPGA basiertes System, welches in der Lage ist, Cortex-M basierte Microcontrollern zu emulieren.



Was ist das MPS?

- bietet diverse Schnittstellen und Ein-/Ausgabegeräte
- verfügt über 8 MB RAM und 64 MB Flash

Was ist uClinux



Linux ist auf dem Vormarsch!

Was ist uClinux?

- eine Distribution
- Linux für Microcontroller
- verbraucht wenig Speicher
- benötigt keine MMU[1]
- verwendet leicht veränderten Standard-Kernel
- spezielle Möglichkeiten für embedded systems[2] vorhanden (bspw. XIP[3])

1) Memory Management Unit: Speicherverwaltung
2) Embedded Systems: eingebettete Systeme
3) eXecution In Place: im aktuellen Speicherbereich ausführen

Sinn und Zweck dieser Arbeit

- akademischer Ansatz zur Prüfung, ob die Ausführung von Linux auf dem Cortex M4 sinnvoll ist
- Linux in der Microcontroller-Ebene nutzen
- vorhandene Portierung für Cortex M3 nutzen und Features des Cortex M4 integrieren (FPU, Vektorbefehle)

Sinn und Zweck dieser Arbeit

- Verwendung der Distribution anstelle der Kernel-Standalone-Lösung von ARM mit precompiled executables
- Beispielanwendung

Was wurde erreicht?

- lauffähiger Kernel mit Initramfs
- Anwendungen der Distribution und die Bibliothek uClibc lauffähig
- minimale Beispielanwendung in die Distribution integriert

uClinux vs. FreeRTOS

uClinux

- leicht erweiterbar
- Treiberunterstützung
- Konsole vorhanden
- Dateisystem vorhanden
- hoher Speicherbedarf

- große Community
- Standardkernel wird verwendet

andere (Echtzeit-) Betriebssysteme

- leicht konfigurierbar
- keine Treiberunterstützung
- ohne Konsole
- ohne Dateisystem
- geringer bis mittlerer Speicherbedarf
- mittlere Community
- Eigenständiges OS

Kosten vs. Nutzen

Aufwand

- Kernel portieren (400 – 800 Mannstunden)
- Distribution portieren (300 – 500 Mannstunden)
- eigene Software implementieren

Ertrag

- Verwendbarkeit von Dateisystem, Netzwerk, USB
- Verwendbarkeit von diversen Tools
- eigene Funktionalität nutzen

Kosten vs. Nutzen

- uClinux bietet diverse Schnittstellen zur Kommunikation
- Speicher- und Flashverbrauch erfordern externen RAM und FLASH

  → zusätzliche Peripherie
- Grundeinsatz für eine Architektur erforderlich, wenn diese noch nicht portiert wurde

Wissenswertes

- Busybox Applet Mechanismus

  erstes Argument ist Name des ausgeführten Befehls
- nur UART 2 verfügt über RTS-Controlflow
- uClinux verwendet bFLT anstelle von ELF
- Keil uVision 4 kann maximal 32 kB in der Demo-Version übertragen

Systemanforderungen

- Mindestens 8 MB RAM
- 4 MB Flash
- Systemtakt von 50 MHz

- Da intern meist weniger zur Verfügung steht, muss auf externen Speicher zu gegriffen werden.

Speicherverbrauch

z. Z. wird Kernel vom Flash ins RAM kopiert, hierdurch wird ...

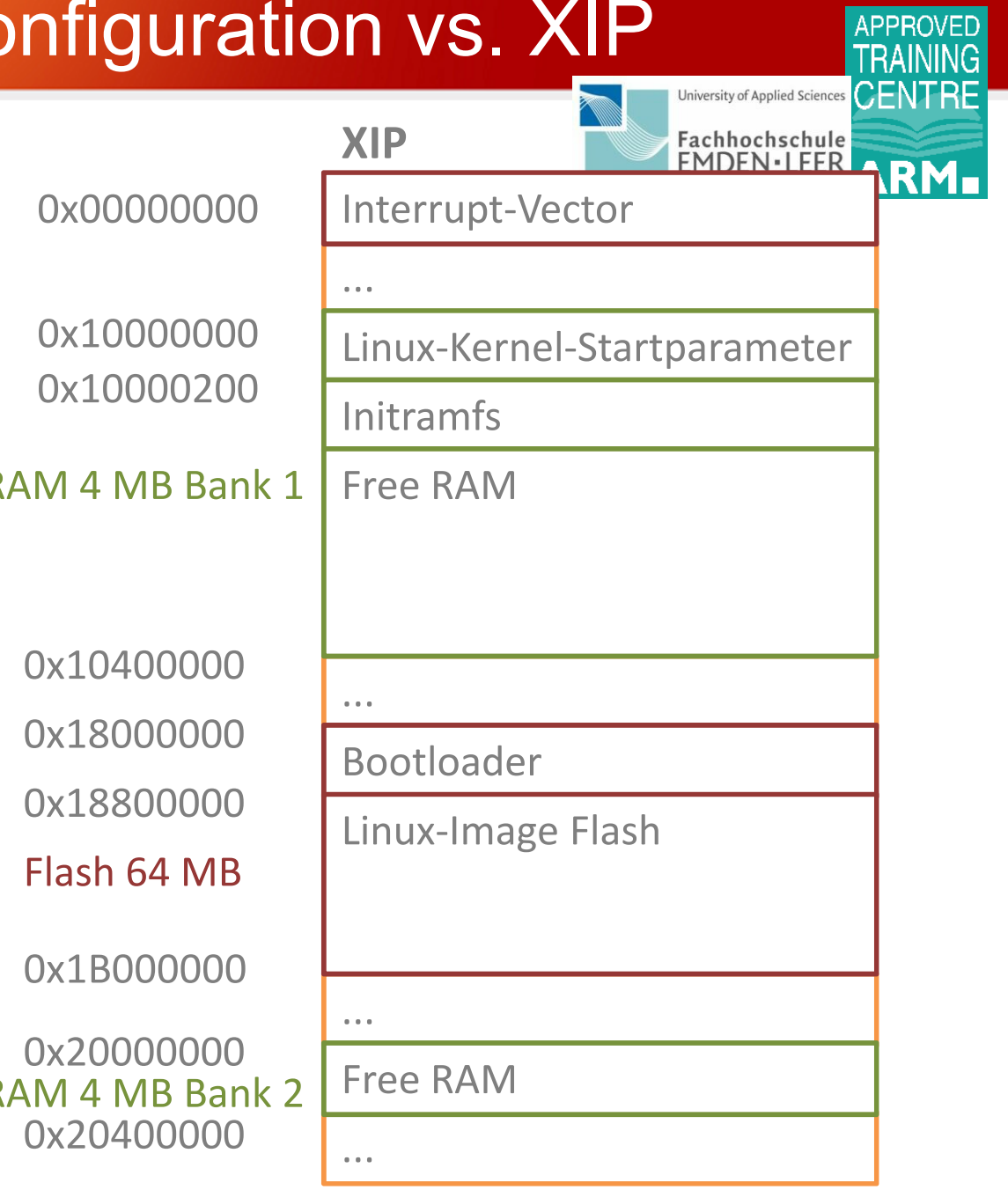
- ... mehr RAM benötigt
- ... der Startvorgang verzögert
- ... die Ausführgeschwindigkeit erhöht

Speicherverbrauch

Es ist möglich den Kernel via XIP (eXecuting In Place) im Flash auszuführen, dies ...

- ... reduziert den Speicherverbauch um bis zu 4 MB
- ... reduziert die Ausführgeschwindigkeit bei höherer Zugriffszeit auf den Flash
- ... reduziert die benötigte Startzeit

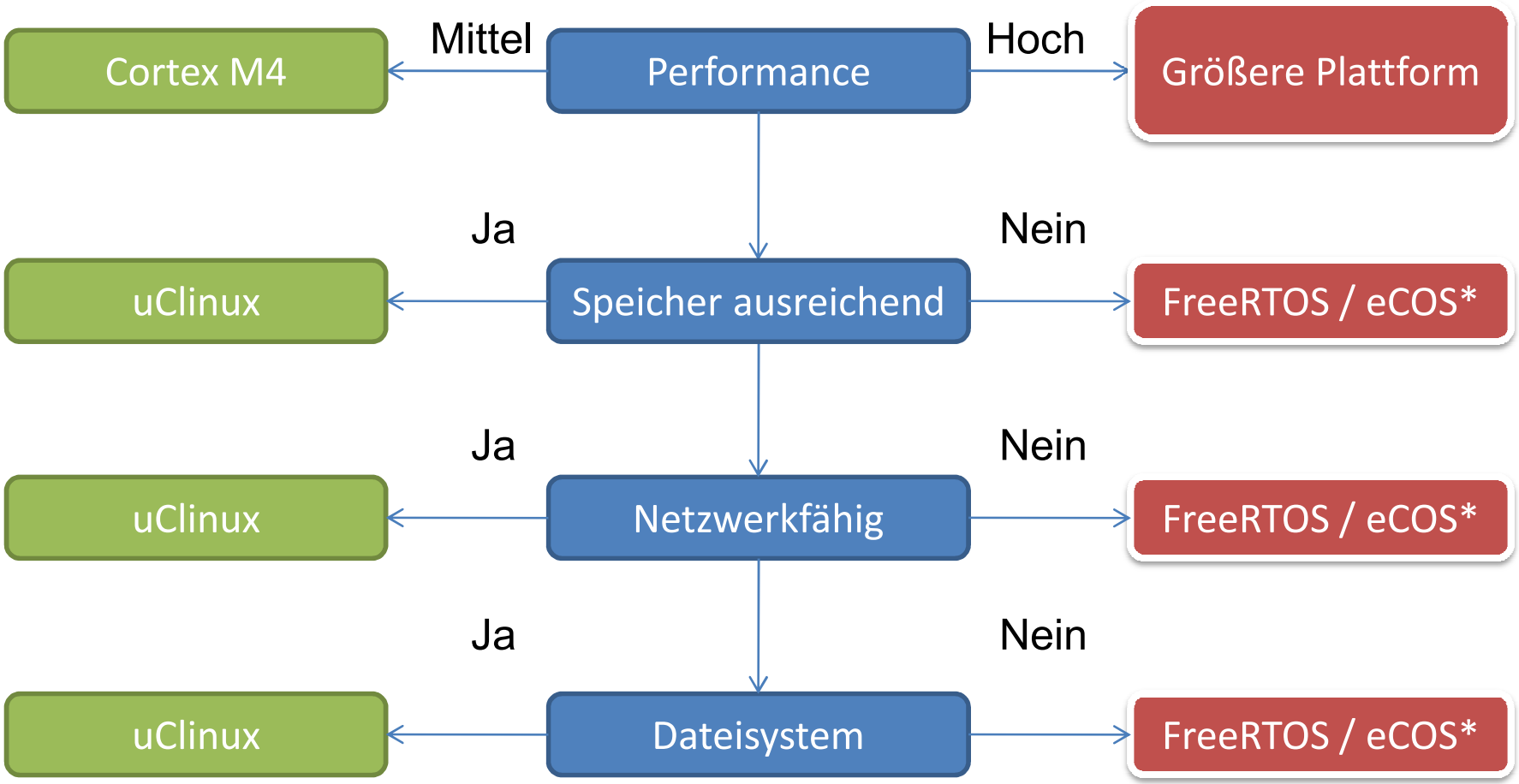
Normale Konfiguration vs. XIP



Wann ist der Einsatz sinnvoll

- Bei Low-cost Produkten mit Multitasking-Voraussetzungen
- Bei Systemen, welche bei geringen Strombedarf auf Dateisysteme und/oder Netzwerke zugreifen müssen
- Bei Systemen, welche nachrüstbar sein müssen

Wann ist der Einsatz sinnvoll



* Beispiele für andere (Echtzeit-)Betriebssysteme

Aufwand

- Der größte Aufwand entsteht bei der Einarbeitung in die Linux-Kernel Architektur.

- Es ist mit einer Zeit von ungefähr 800 Mann-Stunden zu rechnen.

Vorführung

- PuTTY als Konsolensoftware
- zwei serielle Schnittstellen
  - erste für Kommunikation mit dem Bootloader
  - zweite für Kommunikation mit der Linux-Konsole
- Übersetzung erfolgt in Linux-Umgebung
  - Verwendung von Symlinks während des Make-Prozesses macht dies erforderlich

Weiterführende Arbeiten

- vollständige Integration des ROMFS
- Aktivierung der Schnittstellen wie:
  - Ethernet
  - USB
  - MMC
  - Sound
- Änderungen an der Bootloader-Methode

Fazit

Linux ist durchaus auf einem Microcontroller verwendbar, allerdings wird externer Speicher sowie Flash benötigt. Dies somit auch der einzige Grund einen größeren Prozessor zu wählen.

Die Arbeit zeigt das Linux auch auf kleinen Architekturen lauffähig ist und absehbaren Overhead erzeugt.

Training & Support options

- **ARM Training's options include:**
  - ARM Cortex-M(3) Embedded Software Workshop (4 days)
    - Covering (M0, M1, M3, M4)
  - ARM Cortex-R4 Embedded Software Workshop (4 days)
  - RG Advanced Debug for ARM Cortex-M
    - One day events with partners !
- **Embedded Softrware classes:**
  - Embedded C for real time applications
  - Embedded Linux for ..
  - Fundaments of RTOS
- ..for more Infos see **www.Doulos.com/ARM**

Quellen und Markenrechte

Quellen:

- Masterarbeit

Markenrechte:

- Cortex™ ist ein eingetragenes Warenzeichen von ARM®
- uVision 4™ ist ein eingetragenes Warenzeichen von Keil®
- Linux wie auch uClinux stehen unter der GNU-License

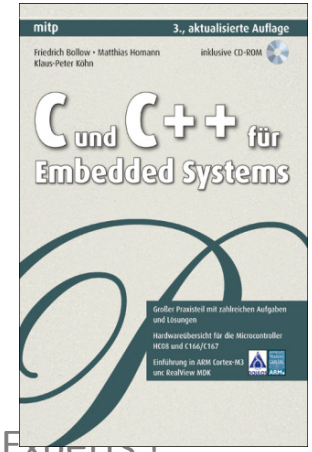
  Weiteres zum Thema GNU-License:

  http://www.gnu.org/licenses/old-licenses/gpl-2.0.html

References

- Training from Doulos - www.doulos.com/ARM
- www.keil.com / www.ARM.com

- Books
  - The Definitive Guide to the ARM Cortex-M3
    - ISBN: 978-0-7506-8534-4
  - C und C++ für Embedded Systems
    - First German ARM Cortex-M Into from Doulos ARM Experts !



ARM Cortex-M(3) embedded software

DOULOS

University of Applied Sciences Fachhochschule EMDEN·LEER

APPROVED TRAINING CENTRE ARM

	Day 1	Day 2	Day 3	Day 4
9am	The ARM Architecture	Thumb-2 Instruction Sets	Embedded SW Development	Cortex-M3 MPU *
	Coretx-M3 Introduction & Processor core	Migrating Legacy ARM/Thumb code to Coretx-M3	Compiler Hints and Tips	CoreSight Debug Architecture Overview
Lunch				
	RealView Overview	Coretx-M3 Interrupts	Cortex-M3 Memory Types	Invasive & Non-invasive Debug
		Coretx-M3 Exception Handling		
5pm	**RealView Introductory Workbook**	**Cortex-M3 Embedded SW-Workbook**	**Embedded MCU SW Workbook**	**Embedded MCU SW Workbook**

Next Doulos Trainings

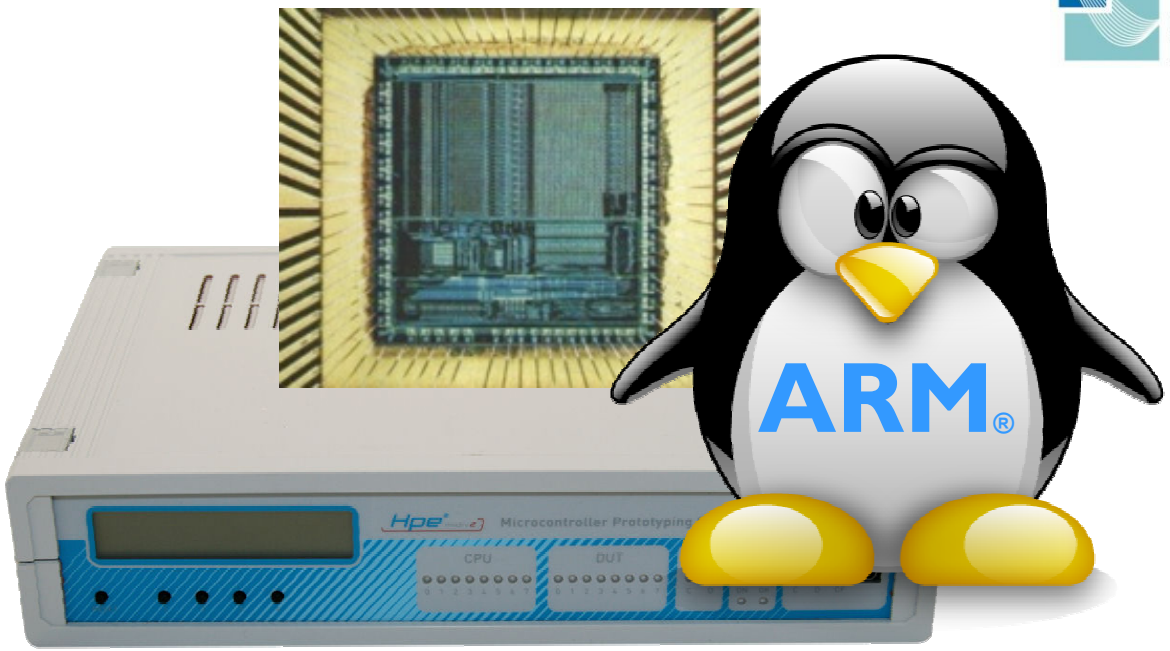
- ARM Cortex-M(3) Software Design (4 days)
  - November 22nd, 2010 Ringwood, UK
  - October 18th, 2010 Paris, FR
  - December 12th, 2010 Hannover, DE
- ARM Cortex-R4 Software Design (4 days)
  - October 25th, 2010 Hannover, DE

• on-site any time anywhere !

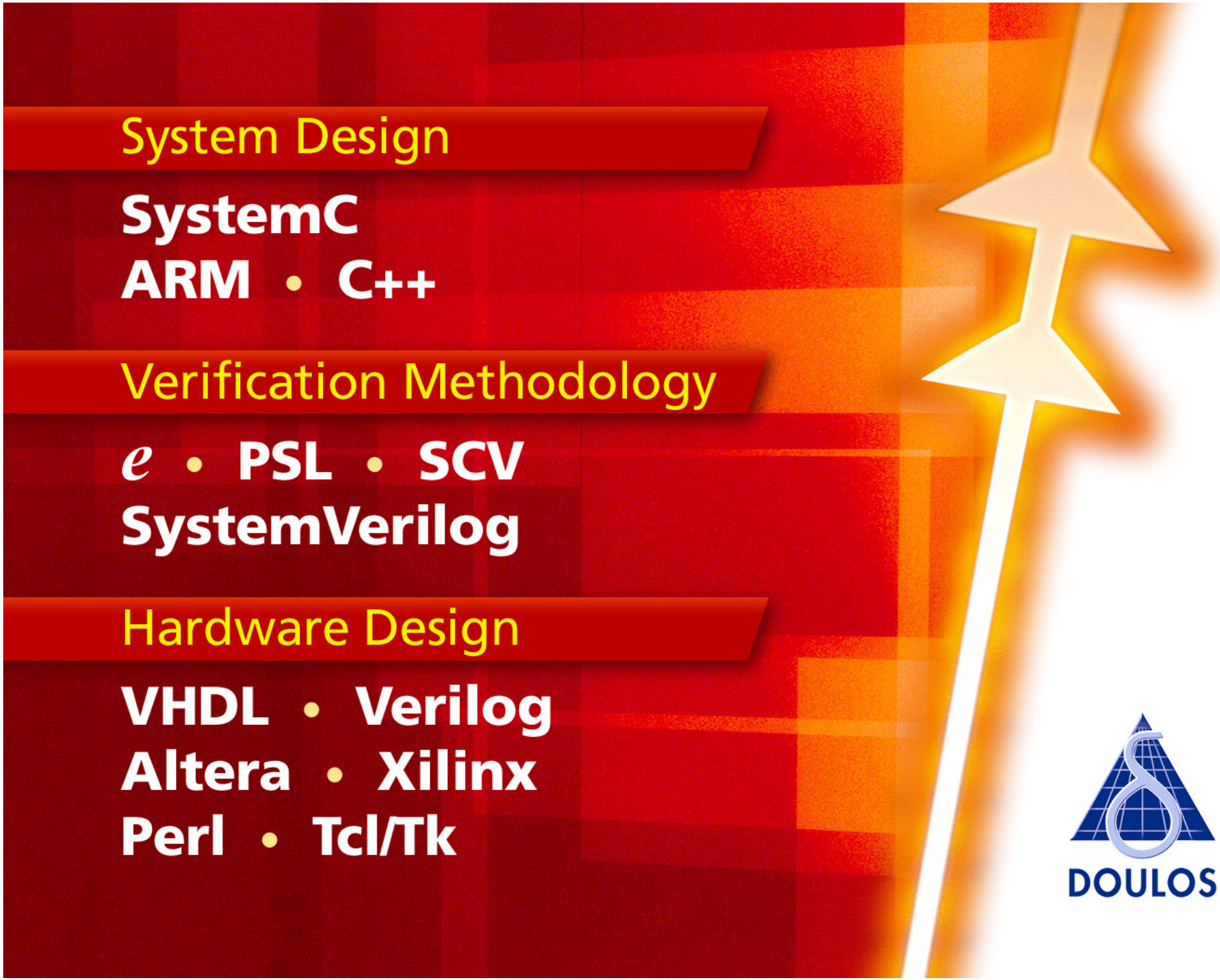
DOULOS

University of Applied Sciences
Fachhochschule EMDEN·LEER

APPROVED TRAINING CENTRE ARM



Vielen Dank für Ihre Aufmerksamkeit


System Design
SystemC
ARM • C++
Verification Methodology
e • PSL • SCV
SystemVerilog
Hardware Design
VHDL • Verilog
Altera • Xilinx
Perl • Tcl/Tk
DOULOS

Concluding slide

CONTACT

Frank Mölendörp

Junior Consultant

+49 (0)511 277 1340

Frank.Moelendoerp@doulos.com

Doulos Central European Office
Garbsener Landstrasse 10
30149 Hannover
Deutschland

DOULOS