Technische Hochschule Rosenheim

Modul - IT Systems (IT)

Bachelor Programme AAI

03 - Lecture: Processor Architecture

Prof. Dr. Marcel Tilly

Faculty of Computer Science, Cloud Computing

Agenda

Technische Hochschule Rosenheim

We will look at the following in this lecture:

- Switching networks
- Latches and flip-flops
- How are memories realized?
- What is behind RISC and CISC? And what does it mean at all?
- What is behind ARM?
- Which processor architectures exist at all?

Learning objectives

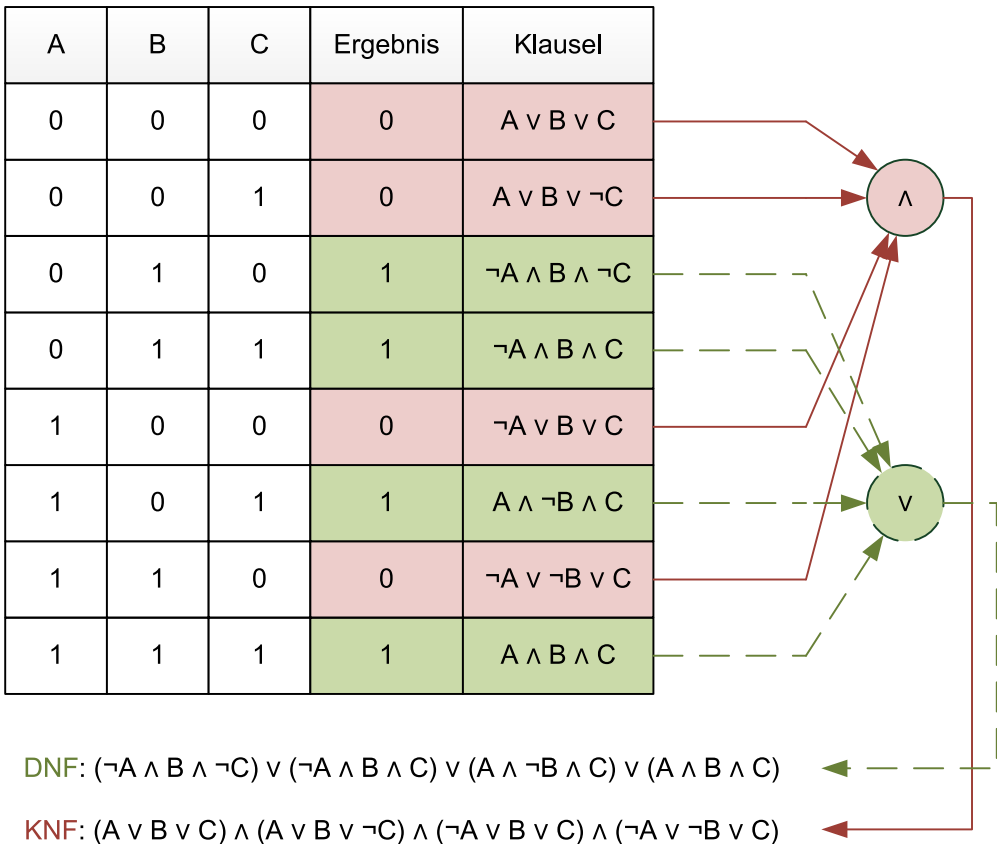
Technische Hochschule Rosenheim

Students ...

- ... can set up the value table for switching networks and calculate the Boolean function.
- ... can explain flip-flops
- ... can explain different types of memory and their intended use
- ... can explain the difference between RISC and CISC architectures
- ... can describe the advantages of ARM architectures

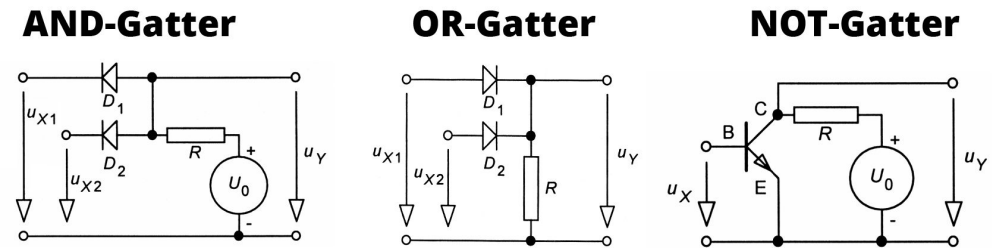
CNF vs. DNF

A	B	C	Ergebnis	Klausel
0	0	0	0	A ∨ B ∨ C
0	0	1	0	A ∨ B ∨ ¬C
0	1	0	1	¬A ∧ B ∧ ¬C
0	1	1	1	¬A ∧ B ∧ C
1	0	0	0	¬A ∨ B ∨ C
1	0	1	1	A ∧ ¬B ∧ C
1	1	0	0	¬A ∨ ¬B ∨ C
1	1	1	1	A ∧ B ∧ C



https://de.wikipedia.org/wiki/Konjunktive_Normalform#/media/Datei:Knf+dnf.svg

Gates (electronic)



- **AND gate**: The two diodes are aligned so that the higher potential of the gate pullup resistor (high resistance, 10k) is pulled to ground until both inputs are also at the high potential, allowing the pullup resistor to pull the voltage to the high level.
  - Both inputs must be HIGH to switch output HIGH
- **OR gate**: This time the two diodes are the other way around and the resistor (high resistance, 10k) pulls the potential to ground. But if at least one of the two inputs is HIGH, the diodes let this voltage through unhindered and pulls the potential of the output up with it.
  - One input is HIGH to switch putput to HIGH
- **NOT-Gate**:The standard blocking transistor closes at a LOW gate voltage, so the pullup resistor pulls the potential up. However, if HIGH is now applied to the gate, then the transistor switches through, causing the output to be directly connected to ground.

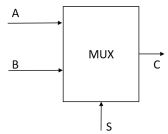
Multiplexer

Technische Hochschule Rosenheim

Digital computers have to process and transmit a large amount of data. To handle all these transmissions separately would be a utopian effort. Imagine you would need an extra cable for each signal!

Example:

- Multiplexers (or selectors) select (input) signals.
- A multiplexer has several inputs and one output.
- One input is used for control (selection)



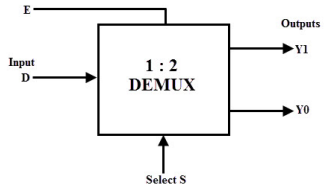
A	B	S	C
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Function: $C=(A \wedge \neg S) \vee (B \wedge S)$

Exercise 1: How does a DEMUX work?

Technische Hochschule Rosenheim

Break out and discuss. Can you create the truth table?



When the select input S is LOW, then the input will be passed to Y0 and if the select input is HIGH, then the input will be passed to Y1.

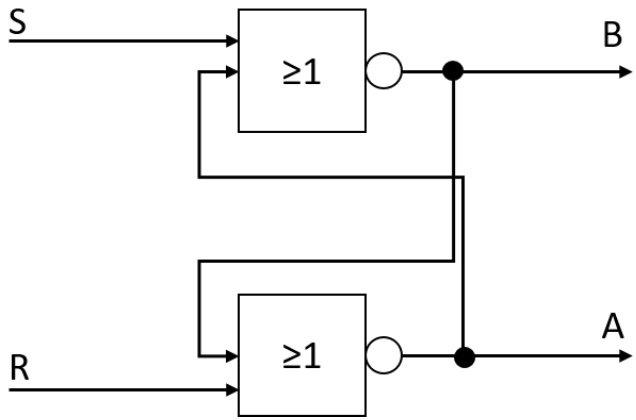
S	D	Y1	Y0
0	0		
0	1		
1	0		
1	1		

Realization of 1-bit memories

A memory is used to store instructions and data to be executed

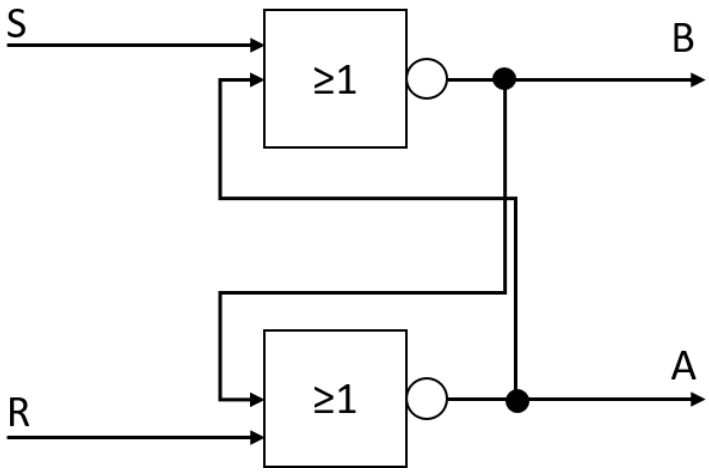
How can such a gate-level memory work?

To create a 1-bit memory, we need a circuit that remembers input values. Such a circuit can be built from two NOR gates.



Why does this work?

- We are looking at an **SR latch**.
- This circuit has two inputs: S (set) to set and R (reset) to clear, and two complementary outputs A and B.
- B' and A' are the results at A and B after the pass!



S	R	A	B'	A'
0	0	0	1	0
1	0	0	0	1
1	0	1	0	1
0	1	1	0	0
0	1	0	1	0
0	0	1	0	1
1	1	-	-	-
1	1	-	-	-

Why does this work?

Technische Hochschule Rosenheim

- We consider a **SR latch**.
- This circuit has two inputs: S (set) to set and R (reset) to clear and two complementary outputs A and B.

S	R	A	B'	A'
0	0	0	1	0
1	0	0	0	1
1	0	1	0	1
0	1	1	0	0
0	1	0	1	0
0	0	1	0	1
1	1	-	-	-
1	1	-	-	-

- **1st case**: S=R=0,

  if A=0 => B=1 => A'=0 if A=1 => B=0 => A'=1 => consistent, i.e. A unchanged
- **2nd case**: S=1 and R=0

  => A=0 => B=0 => A=1 => A=1 => B=0 => A=1 => if you set S, you set A
- **3rd case**: R=1 and S=0

  => A=1 => B=0 => A=0 => A=0 => B=1 => A=0 => if you set R, you delete A
- **4th case**: R=S=1 is not allowed!

SR-Latch

Technische Hochschule Rosenheim

We can summarize that if ...

- S=R=0 the current state is preserved
- When setting S or R, the latch element ends in a certain state (regardless of the previous state)
  - S=1 at R=0 sets A=1
  - R=1 at S=0 sets A=0

Thus, the circuit remembers whether S or R were last set.

Computer memories can be developed on the basis of this property.

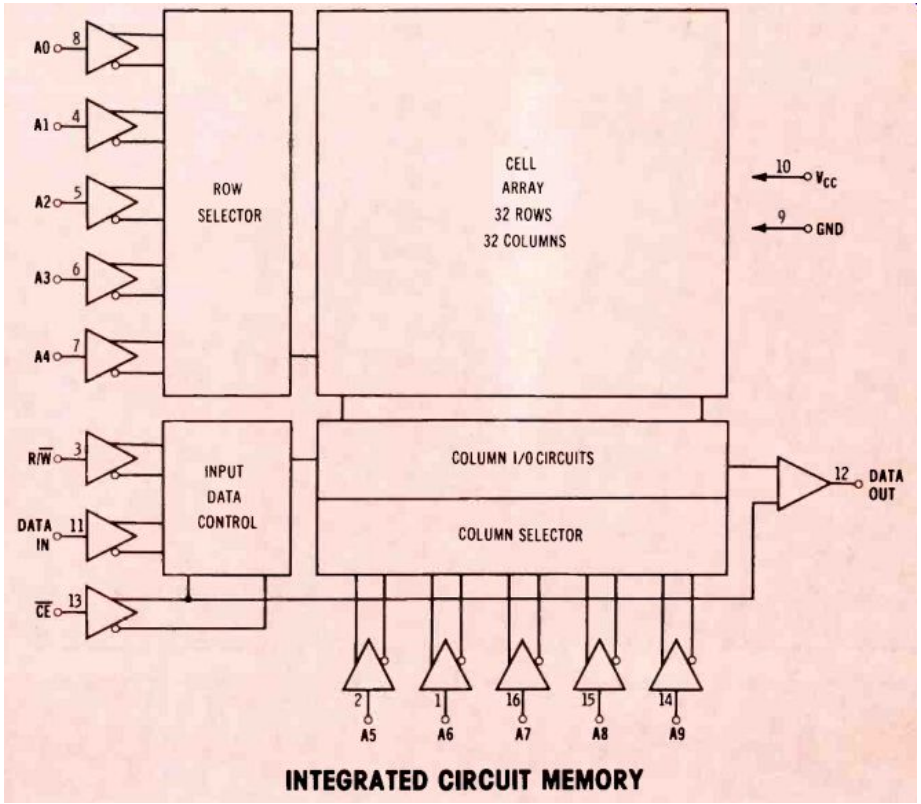
> Note: An equivalent behavior of a 1-bit memory can also be realized by a latch of NAND gates.

flip-flops

Technische Hochschule Rosenheim

- The first electronic flip-flop was invented in 1918 by the British physicists William Eccles and F. W. Jordan. It was initially called the Eccles–Jordan trigger circuit and consisted of two active elements (vacuum tubes)
- Flip-flops can be either simple (transparent or asynchronous) or clocked (synchronous).
  - In the context of hardware description languages, the simple ones are commonly described as latches, while the clocked ones are described as flip-flops.
  - **flip-flop** is called edge-triggered, while a **latch** is called level-triggered
- Simple flip-flops can be built around a single pair of cross-coupled inverting elements: vacuum tubes, bipolar transistors, field effect transistors, inverters, and inverting logic gates have all been used in practical circuits.
  - Triggers on transition from 0 to 1 (rising edge) or 1 to 0(falling edge)
- classic flip-flop circuits allow the storage of one bit over the duration of one clock.
- To store multiple bits, flip-flop devices are combined (2-bit registers, 8-bit registers…)

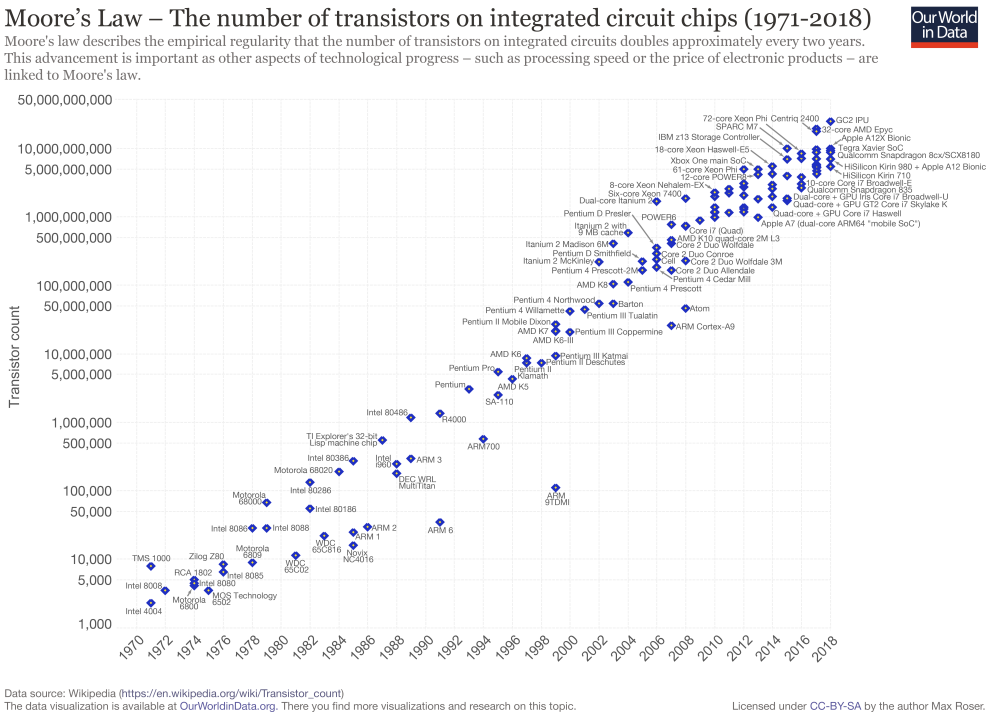
Memory



INTEGRATED CIRCUIT MEMORY

Morree's law

In the course of improvement, the number of bits that can be packed onto a chip increases by a factor of 2 every 18 months!



taken from https://upload.wikimedia.org

Random Access Memory (RAM)

Technische Hochschule Rosenheim

- Memory devices that can read and write are called RAMs (Random Access Memory):
  - Static RAMs (SRAMS) are internally implemented with circuits that structurally correspond to *flip-flop memory devices*. They have the ability to retain their contents as long as current is flowing. PRO: SRAM access time in nanosecond range.
  - Dyanmic RAMS (DRAMS) consist of a *transistor* and *capacitor* which can then be charged and discharged. So zeros and ones can be stored. However, here each bit must be *refreshed* every few millliseconds. PRO: Have a large storage density.

DRAM are slower than SRAM.

Which one would be a good pick for a cache?

Other memories

Technische Hochschule Rosenheim

- ROMs (*Read-only Memory*): Since RAMs lose their memory contents without power, there are ROMs whose contents *are* burned in as a memory mask.
- PROMs (*Programmable ROM*): This chip works like a ROM. Here, the chip can be programmed **once** by *burning* fuses.
- EPROM (*Erasable PROM*): Can be programmed as well as erased. If a quartz window of an EPROM is exposed to a strong ultraviolet trick for about 15 minutes, all bits are set to 1. (Extension is an EEPROM = *Extended EPROM*).
- The **Flash memory** can be erased and reprogrammed block by block.

Overview memory

Technische Hochschule Rosenheim

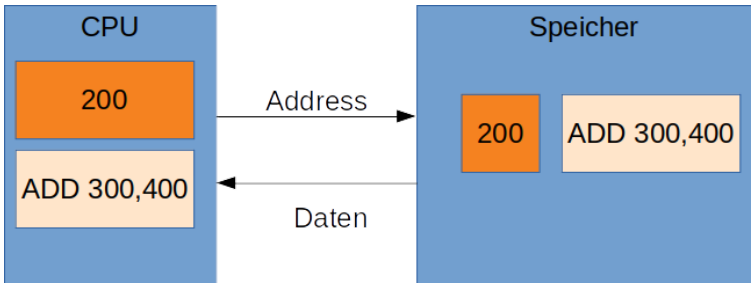
Memory type	Usage type	Deletion	Modifiable	Volatile	Usage
SRAM	Read/Write	Electrical	yes	yes	L2 Cache
DRAM	Read/Write	Electrical	yes	main memory	
ROM	Read only	not possible	no	large volume devices	
PROM	Read only	not possible	once	no	small volume devices
EPROM	mainly read	by means of UV light	reusable	no	device prototypes
EEPROm	mainly read	electrically, bit by bit	with special device	no	device prototypes
Flash	read/write	Electrical blockwise	blockwise	no	SD card

processor *architecture* (revisited)!

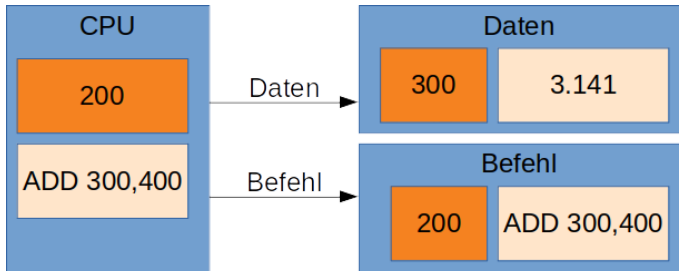
von-Neumann vs. Harvard

- The separation of memory and CPU defines the computer
- CPU fetches instructions from memory
- Program Counter (PC) and Instruction Register (IR) help with processing

von-Neumann



Harvard Architecture



von-Neumann vs. Harvard

Technische Hochschule Rosenheim

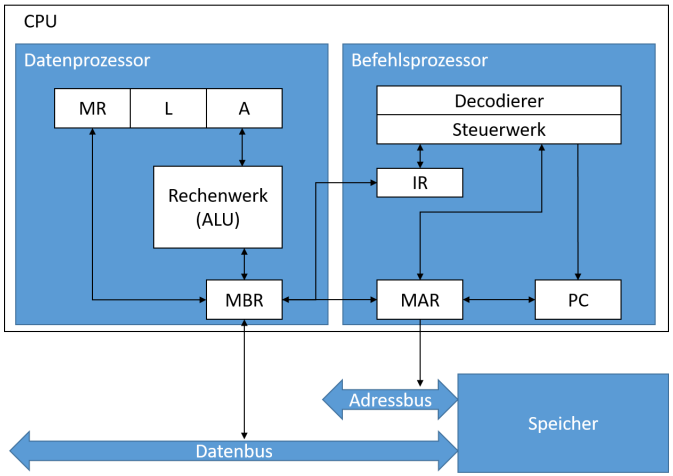
von-Neumann

- in memory are data and code
- An address/data bus between memory and CPU
- Sequential access to data and code

Harvard

- Two separate memories for code and data
- Two types of address/data bus
- In the Harvard architecture, memory and code can be accessed in a *parallel* fashion

Processor (revisited!)



- Data is loaded into registers so that it can be processed

```
02 LD R1 M1 ; M1 -> R1
...
04 ADD R3 R1 R2 ; R1 + R2 -> R3
```

CISC

Technische Hochschule Rosenheim

CISC = Complex Instruction Set

- In a CISC model the CPU usually has few registers (memory locations in the CPU)
- Complex instruction set (e.g. process several registers at once in one loop).
- The instructions usually have unequal instruction lengths. This allows the code to be kept compact.
- Usually these instructions are realized by microcode. This is a decomposition of the instructions into smaller units, a kind of "interpreter" on the CPU.
- In modern CPUs, this microcode is even modifiable, allowing the manufacturer to introduce bug fixes.
- More instructions make the CPU design more complex, however programming more easily complex what CISC makes.

RISC

Technische Hochschule Rosenheim

RISC = Reduced Instruction Set

- RISC goes a different way.
- One limits oneself to the really necessary instructions.
- As compensation you have much more registers (up to 256) on the chip, so you have much more fast register-to-register operations than slow memory-to-register operations.
- The few instructions make the design simpler and the processor cheaper to produce.
- Instead of microcode, one uses direct wiring of the instructions in the decoder, which makes execution faster than with CISC.
- At the same time, the data format is standardized.

RISC vs CISC

Technische Hochschule Rosenheim

CISC

```
01 ;CISC addition
02 ADD M3 M1 M2 ; M1 + M2 -> M3
```

RISC

```
01 ;RISC addition
02 LD R1 M1 ; M1 -> R1
03 LD R2 M2 ; M2 -> R2
04 ADD R3 R1 R2 ; R1 + R2 -> R3
05 ST M3 R3 ; R3 -> M3
```

RISC vs CISC

Technische Hochschule Rosenheim

Some major differences between CISC and RISC architectures are listed

CISC	RISC
The original microprocessor ISA	Redesigned ISA that emerged in the early 1980s
Instructions can take several clock cycles	Single-cycle instructions
Hardware-centric design – the ISA does as much as possible using hardware circuitry	Software-centric design – High-level compilers take on most of the burden of coding many software steps from the programmer
More efficient use of RAM than RISC	Heavy use of RAM (can cause bottlenecks if RAM is limited)
Complex and variable length instructions	Simple, standardized instructions
May support microcode (micro-programming where instructions are treated like small programs)	Only one layer of instructions
Large number of instructions	Small number of fixed-length instructions
Compound addressing modes	Limited addressing modes

RISC vs CISC

Technische Hochschule Rosenheim

Typical CISC representatives

- Intel 8080 and 8086. On the 8080 successor Z80 many computers were developed - The Schneider CPC, Tandy TRS-80, the Sinclair ZX-81 or Spektrum.
- Today: x86 systems (PCs)

Typical RISC representatives

- 1975, West Coast Computer Fair (WSCOM): On the booth of the small company "MOS Industries" the microprocessor 6502 is presented.
- The 6502 later became the basis for a number of computers that made history - the Apple II, the C64 or the Atari VCS 2600 game console.
- Today: ARM

Sidenote: ARM

Technische Hochschule Rosenheim

A sad role played the ARM processor at first!

- Most of the new RISC processors were meant for workstations.
- For Acorn - a well known company in England - the 6502 was not powerful enough.
- Acorn developed the Acorn RISC Architecture (ARM).
- The ARM could compete with a 386 at a fraction of the cost.
- But the company lacked the notoriety to establish this device, the Archimedes.
- ARM recognized the signs of the times and made the ARM processor successful as a microcontroller.
- The ARM was then used in the Newton from Apple (1992) - then under Advanced RISC Machines (=ARM)
- ARM processors have a 70% market share in the 32 bit microcontroller space and are manufactured under license by numerous companies - even Intel.
- Most PDAs, tablets, smartphones use this processor because it is fast and power efficient.

Task 1

Technische Hochschule Rosenheim

Find out which type of processor is used in your...

- ... PC/Notebook (https://pingo.coactum.de/) **No: 115001** ...

- ... In their smartphone (https://pingo.coactum.de/) **No: 066052** ...

is used!

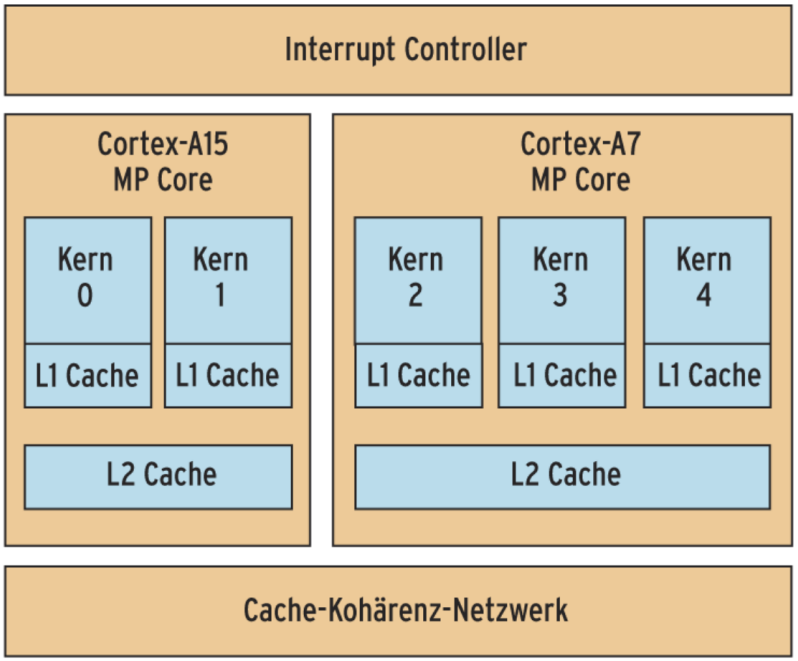
ARM today

Technische Hochschule Rosenheim

Today ARM offers several processor families:

- The CPUs of the somewhat aging ARM7 series are now only installed in devices of the lower price-performance class. The chips are based on an architecture from 1994, which was very successful in the following years. More than ten billion processors of this type perform their services in simple electronic devices.
- The ARM9 series consists of three single-core processors, for example for digital signal processing or Java applications. A total of five billion CPUs of this family have been produced to date. This series also has the most licensees worldwide.
- CPUs in the ARM11 series deliver significantly more performance. These chips have one to four cores, each clocked at up to 1 GHz. They are used in smartphones in particular.
- The fastest ARM processors are currently found in the Cortex family. These processors power upper-class smartphones, notebooks and simple servers.

The ARM Cortex



ARM license

Technische Hochschule Rosenheim

ARM is not a finished chip, but a license and CPU blueprint!

Licensee:

- Apple - Ax and Mx Chip
- Qualcomm: Snapdragon
- Samsung, Motorola, Freescale ...

With Windows 8 he gave for the first time a Windows on ARM!

Why?

Summary

Technische Hochschule Rosenheim

We looked at the following in detail:

- Switching functions
- Latches and flip-flops
- memory
- CISC vs RISC
- ARM