Rosenheim
Technical University
of Applied Sciences

Prof. Dr. Florian Künzner
Technical University of Applied Sciences Rosenheim, Computer Science

CA 12 – Bus and I/O 1

The lecture is based on the work and the documents of Prof. Dr. Theodor Tempelmeier

CAMPUS Rosenheim
Computer Science

Rosenheim Technical University of Applied Sciences

Goal

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Goal

CA::Bus and I/O

- Bus systems
- F-Bus
- Basic bus cycles
- Program sequence and bus cycles
- Access I/O devices

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Intro

Which bus systems do you know?

CAMPUS Rosenheim
Computer Science

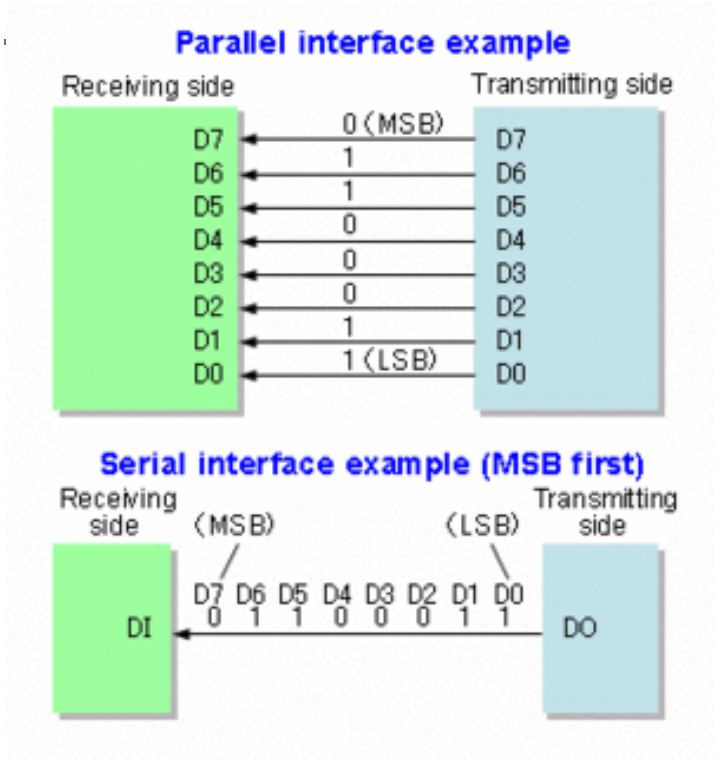
Rosenheim
Technical University
of Applied Sciences

Intro

A bus is a **communication system** inside the computer that **transfers data between components**.

It contains:

- **Wires:**
  - Parallel connection: N parallel wires
  - Bit serial connection: 2 (or more) wires
- **Protocol:**
  - Rules about the meaning of signals
  - Definition of the chronological order of the signals



[source: wikipedia.org]

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Intro

The bus considered here is a fictional bus:

- Fictional bus or
- Fantasy bus

It is a realistic mixture of the VME bus and the PCI bus.

VME bus

The VME bus (Versa Module Europe) is a powerful 32- or 64-bit bus and open. The VME bus is widely used in technical applications because it is also superior to PC buses in terms of its design (vibration, resistance, etc.).

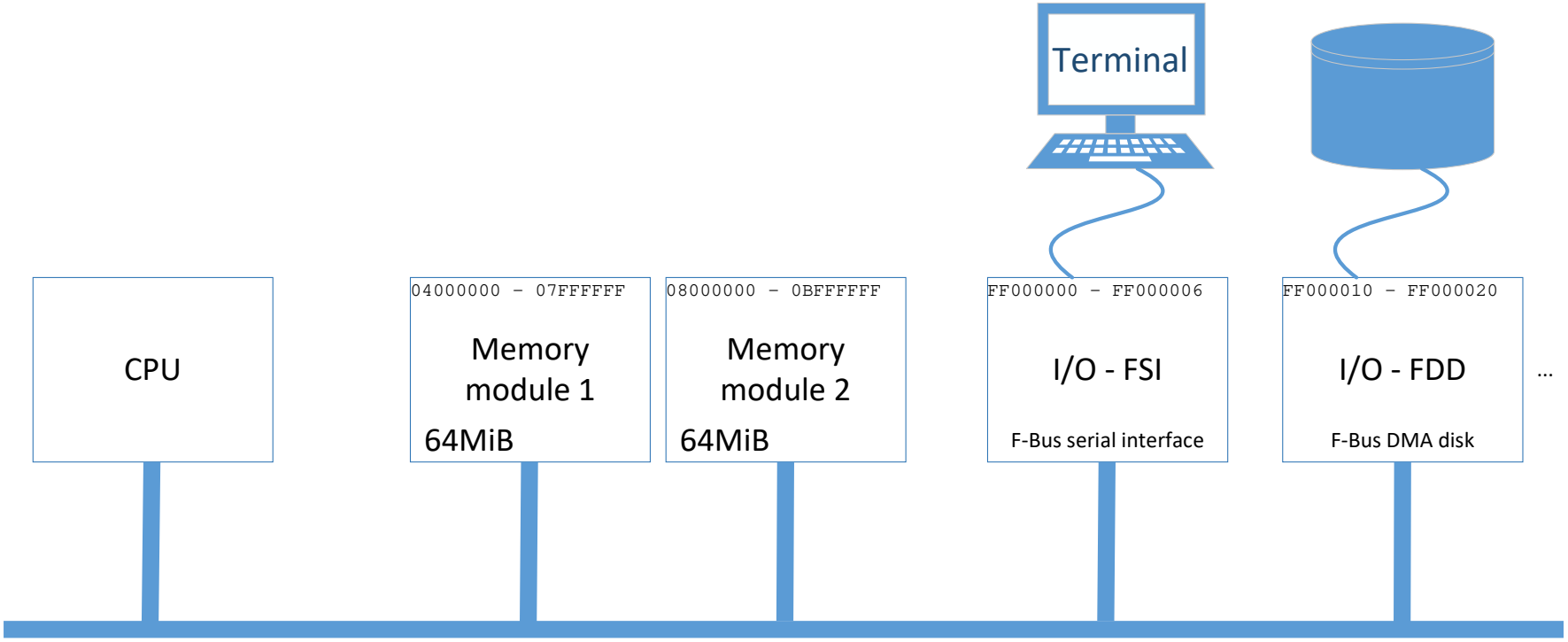
PCI bus

The PCI bus (Peripheral Component Interconnect) from Intel is an important bus for PCs. It is increasingly being replaced by PCI Express (PCIe).

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Universal bus overview



Details

- Universal bus: all components are connected via the same bus system
- F-Bus: all components are mapped to fixed addresses

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

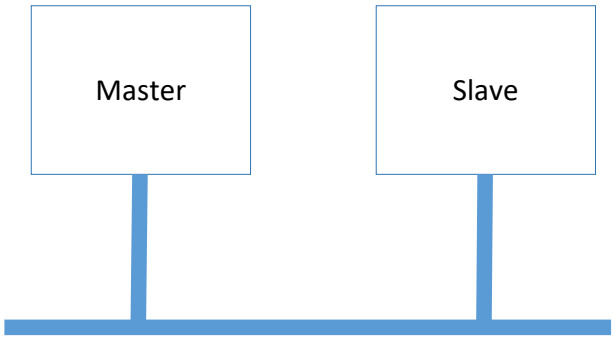
F-Bus: Properties

Properties

- 32 bit bus (32 bit addresses, 32 bit words)
- Asynchronous (unclocked: requires handshake for data flow control)
- Universal bus (allows different types of HW)
- Multiplex lines (sends multiple signals over the same line)
- Memory mapped I/O (access HW via memory addresses)
- Multimaster capability (with decentralised DMA control)
- Central arbitration and interrupt prioritisation (+ daisy chaining)

Alternatives to these special characteristics will be discussed later.

F-Bus: Participants



A bus component is the master:
(we start with the CPU as the master)

- Starts the bus cycle
- Is the chief on the bus

The communication partner is the slave:

- Responses if the master wants something
- Only sends signals after the master initiated a bus cycle
- (Can send interrupts–but we consider this later...)

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Signal lines

Signal line	Description
various	Supply voltage(s), ground, ...
DCOK	DC power ok
POK	Power ok
INIT	Initialize (reset)
ADL00 ADL.. ADL31	Address data lines (ADL), are used for the transmission of addresses and data in time-division multiplex mode; 32-bit bus
FRAME	Start of a bus frame
MSTRRDY	Master is ready
REPLAY	Slave replay
WRITE	Write bus cycle
LOCK	Bus lock
BURST	Burst cycle
...	

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Interrupt and system vectors

Number	Address (hex)	Address (dec)	Description
0	0x00000000	0	Reset
2	0x00000008	8	Bus error
3	0x0000000C	12	Address error
4	0x00000010	16	Illegal instruction
5	0x00000014	20	Division by zero
24	0x00000060	96	Power fail
32	0x00000080	128	TRAP #0
33	0x00000084	132	TRAP #1
...	...	...	...
47	0x000000BC	188	TRAP #15
48	0x000000C0	192	Parallel interface (FPI)
50	0x000000C8	200	Serial interface (FSI)
66	0x00000108	264	DMA disk
72	0x00000120	288	Realtime clock

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

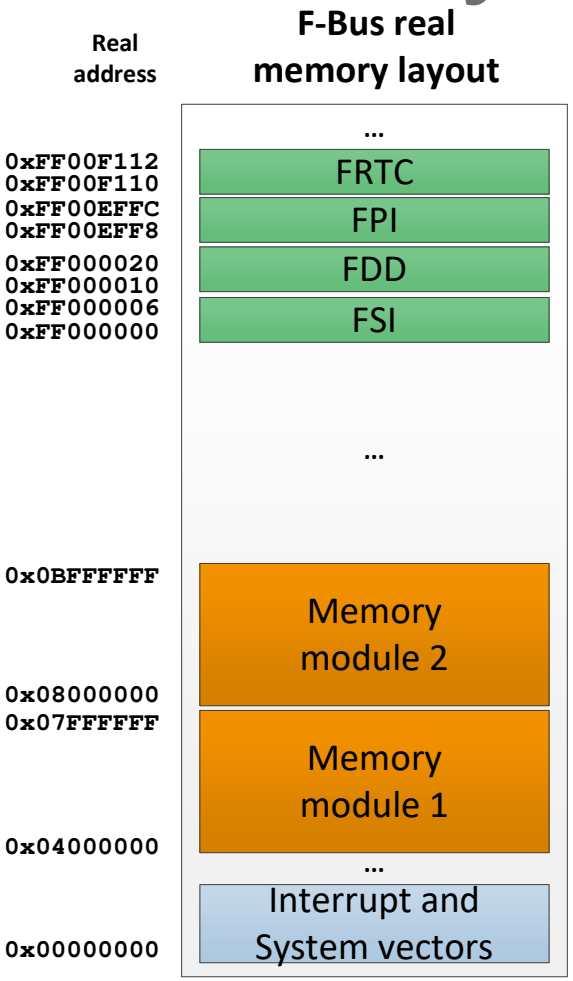
F-Bus: I/O memory layout

Address (hex)	Symbol	Description	Component
0xFF000000	FSI.CSR	Control and status register	
0xFF000002	FSI.TBUF	Transmit buffer	**FSI**
0xFF000004	FSI.RBUF	Receive buffer	**F-Bus serial interface**
0xFF000006	FSI.CFR	Configuration register	
0xFF000010	FDD.CSR	Control and status register	
0xFF000014	FDD.DARH	Disk address register high	**FDD**
0xFF000018	FDD.DARL	Disk address register low	**F-Bus DMA disk**
0xFF00001C	FDD.BAR	Bus address register	
0xFF000020	FDD.BCR	Byte count register	
0xFF00EFF8	FPI.DRCSR	Control and status register	
0xFF00EFFA	FPI.DROUT	Data out	**FPI**
0xFF00EFFC	FPI.DRIN	Data in	**F-Bus parallel interface**
0xFF00F110	FRTC.CSR	Control and status register	**FRTC**
0xFF00F112	FRTC.BPR	Buffer/preset register	**F-Bus realtime clock**

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Memory layout



Properties

- Components are mapped to fixed addresses
- Everything is in the linear address space

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Basic bus cycles

Supported basic bus cycles

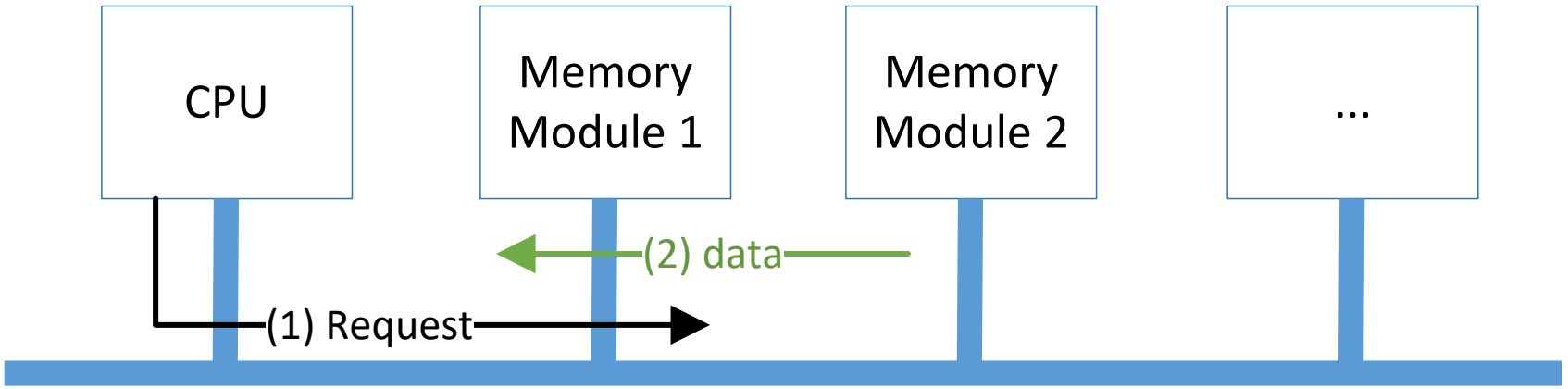
- **Read**: read one word
- **Write**: write one word
- **Atomic read/write**: atomically read and write one word
- **Burst read** (or write): read multiple (b_{max}) words

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Basic bus cycles

Read:

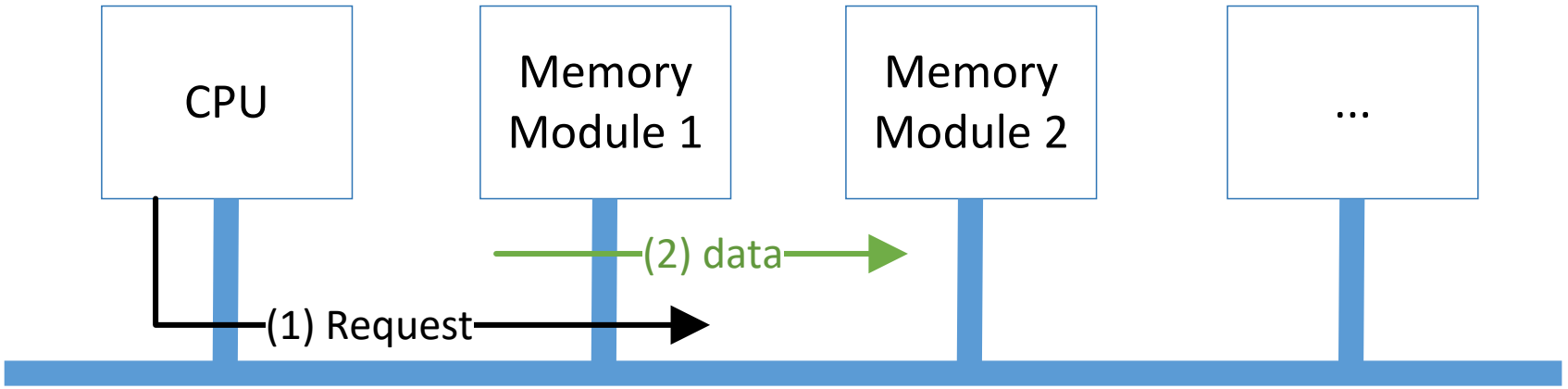


CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Basic bus cycles

Write:

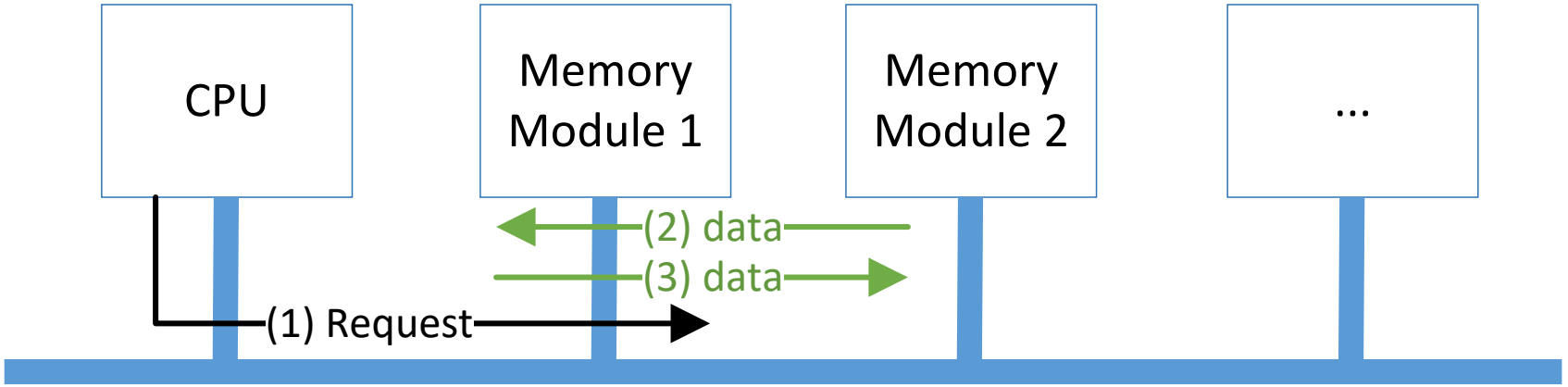


CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Basic bus cycles

Atomic read/write:

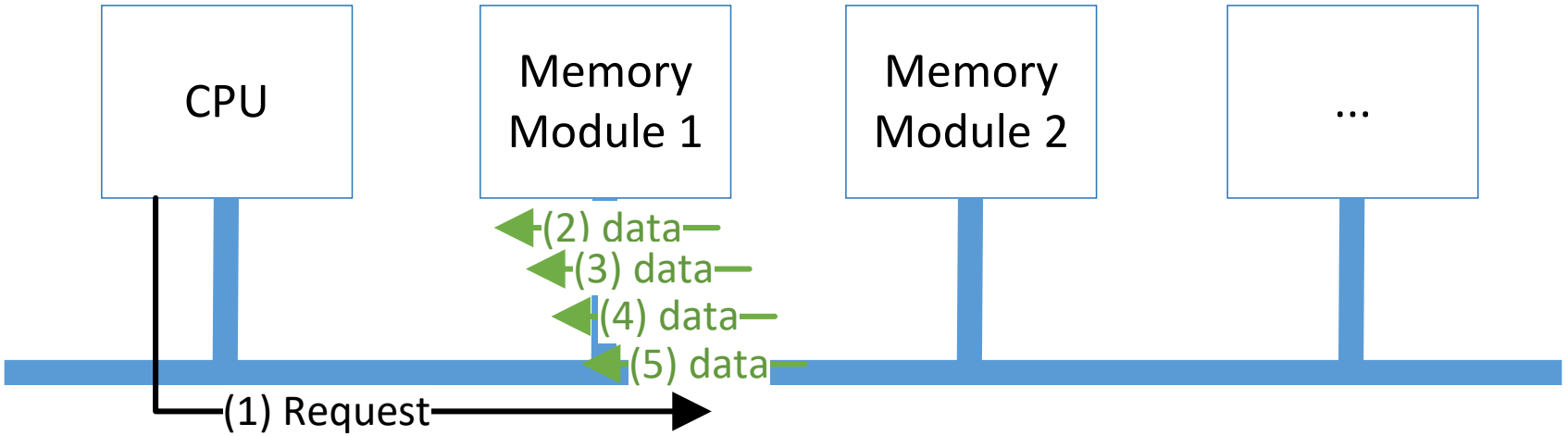


CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

F-Bus: Basic bus cycles

Burst read:



Reads b_{max} words within one burst read cycle. In this lecture and in the exam we define $b_{max} = 4$.

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Basic bus cycle protocols

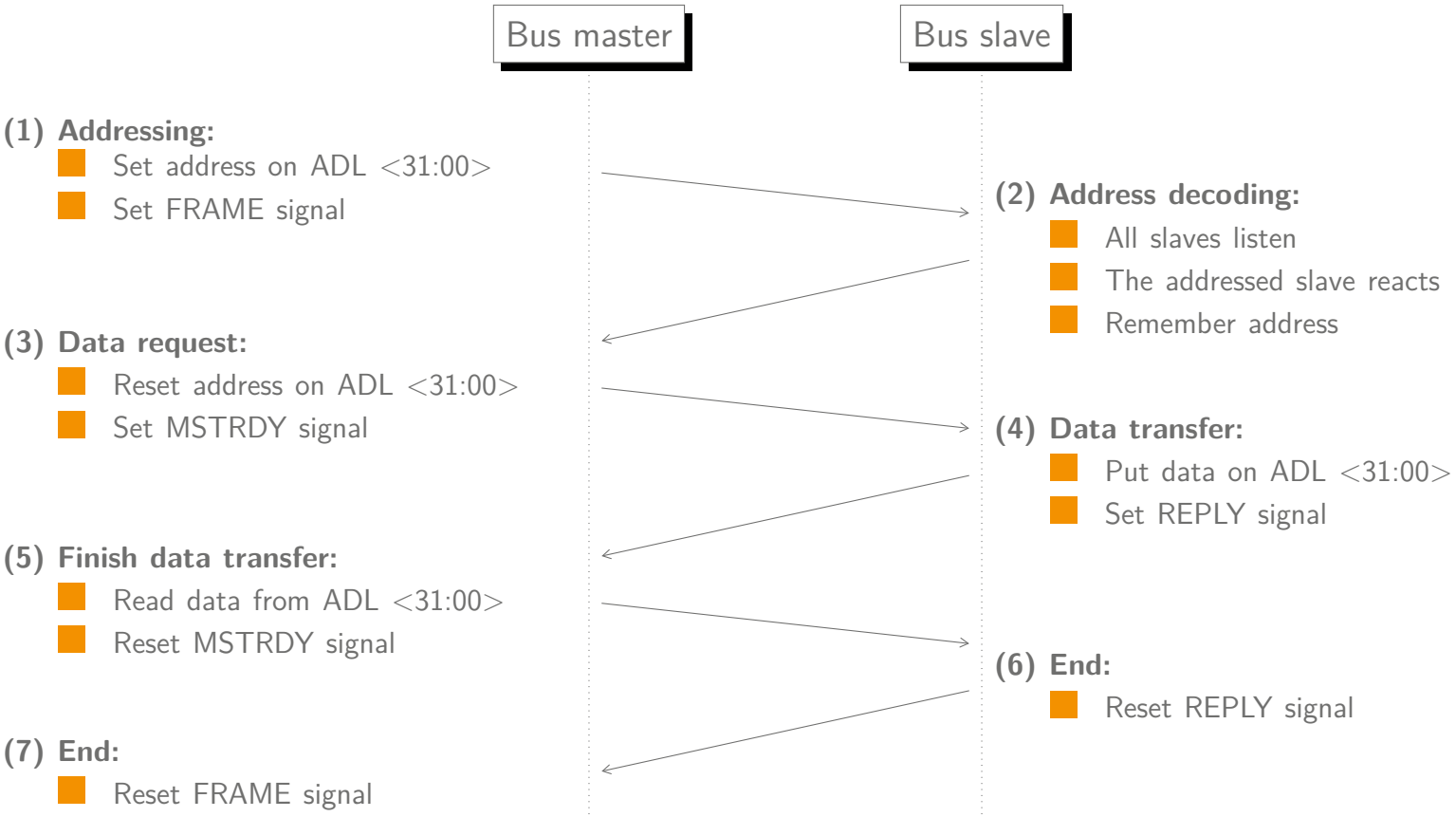
Bus cycle protocols

- **Read**: read one word
- **Write**: write one word
- **Atomic read/write**: atomically read and write one word
- **Burst read** (or write): read multiple words

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

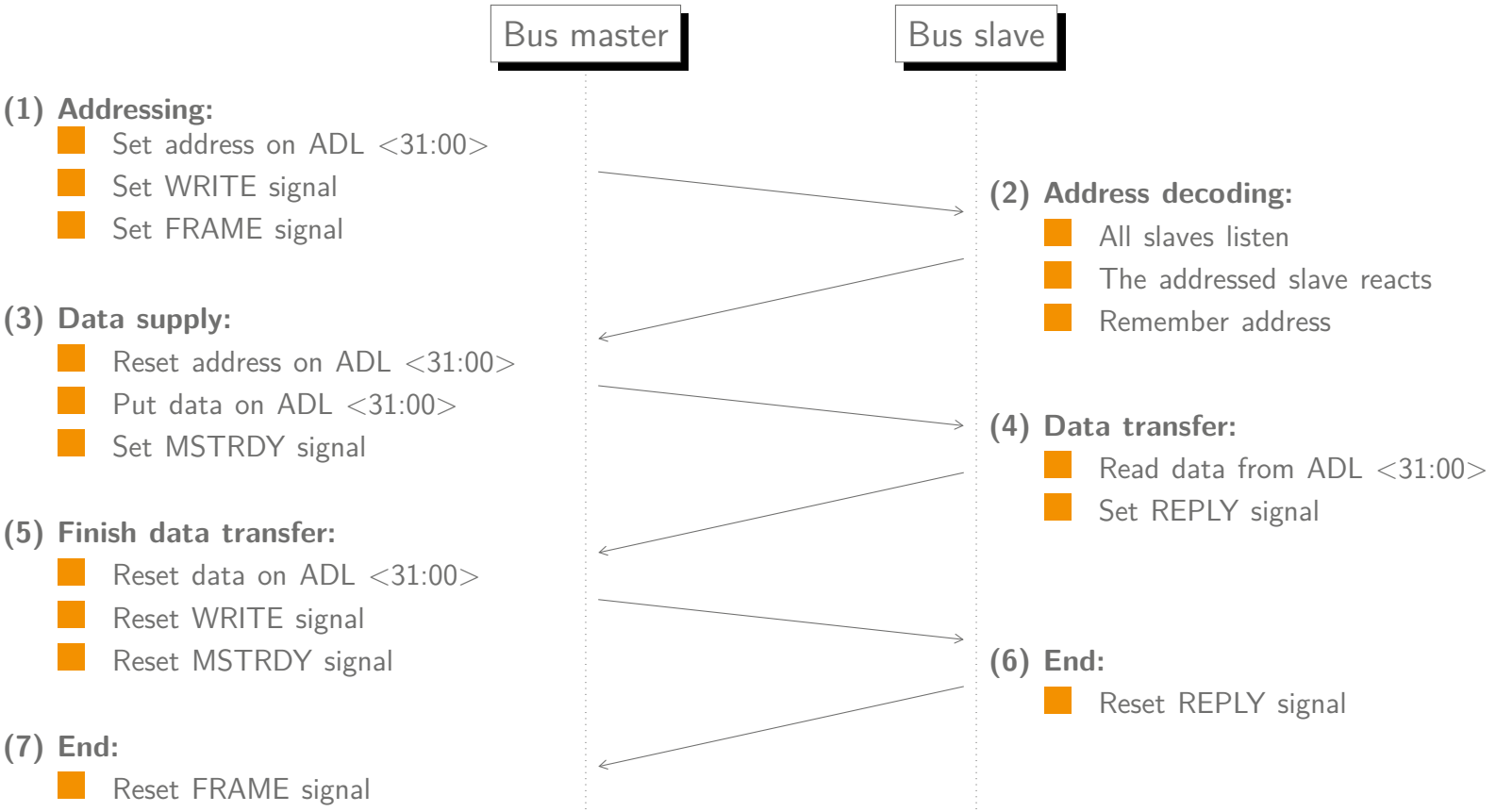
Bus cycle protocol: read



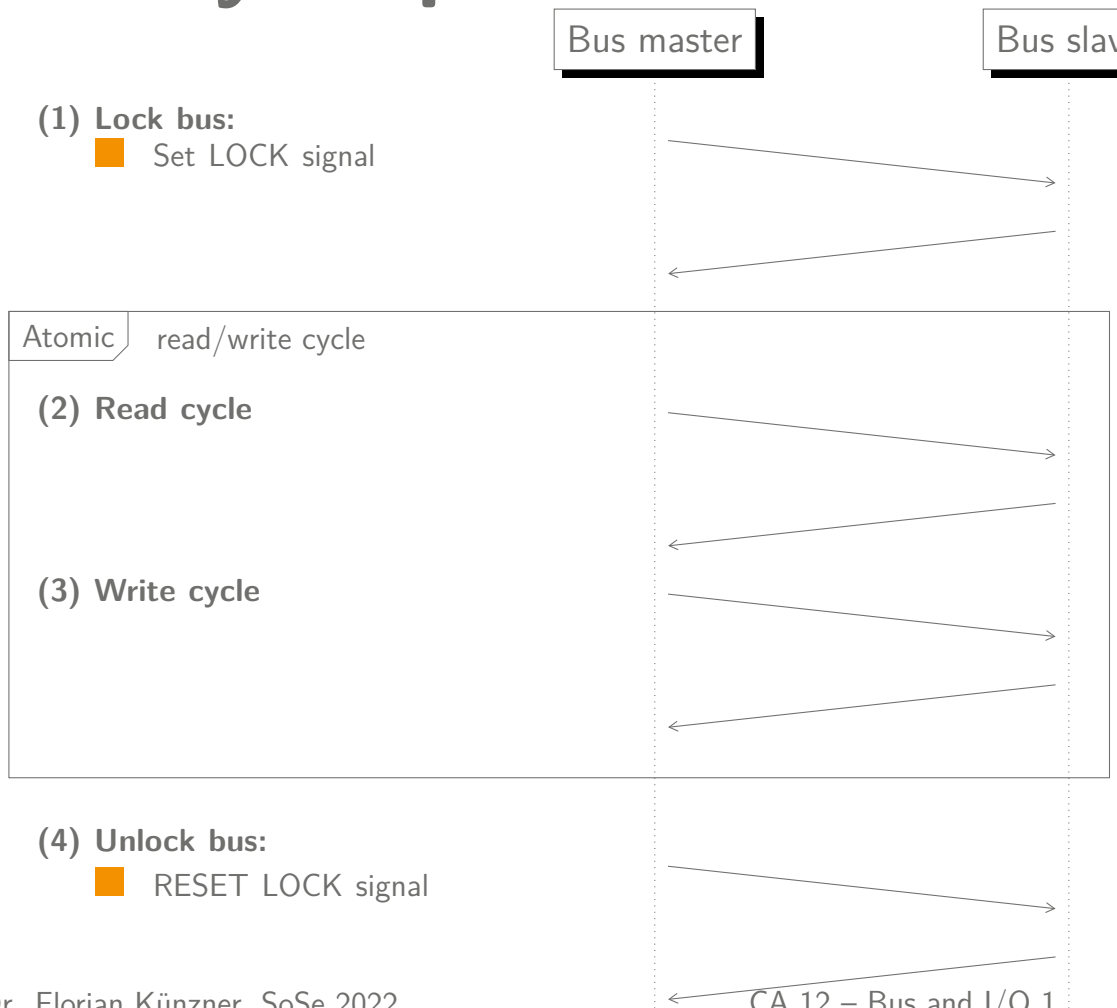
CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Bus cycle protocol: write



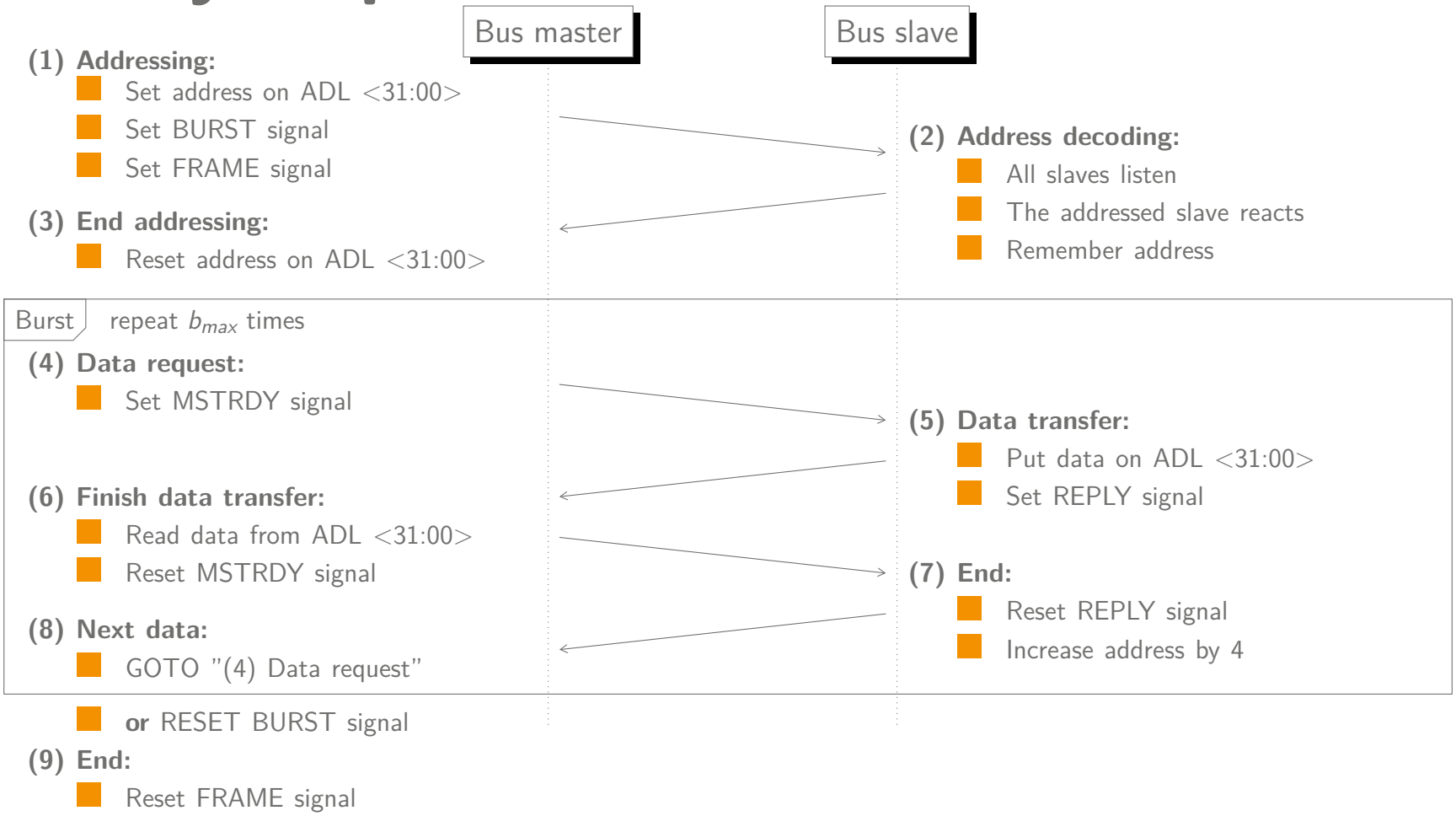
Bus cycle protocol: atomic read/write



CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Bus cycle protocol: burst read



CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Relationship between program sequence and bus cycles

Instructions can lead to

- **No** bus cycle
- **One** bus cycle
- **Many** bus cycles

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Example 1

Assumption: No cache + no data/instruction is in the register

C code

```
1 y = x + y;
```

Assembler code

```
1 ADD x, y ;y = x + y
```

Compiled progam (image, *.elf)

```
1 ...
2 0x..0: 1. Word: Code for ADD
3 0x..4: 2. Word: Address of x
4 0x..8: 3. Word: Address of y
5 ...
```

Resulting bus cycles

Nr.	Cycle	Comment
1	Read	1. Word: Code for ADD
2	Read	2. Word: Address of x
3	Read	Operand x
4	Read	3. Word: Address of y
5	Read	Operand y
6	Write	Result to y

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Example 2

Assumption: Cache + data/instruction is in the register

C code

```
1 y = x + y;
```

Assembler code

```
1 ADD R1, R2 ;R2 = R1 + R2
```

Resulting bus cycles

Nr.	Cycle	Comment
none		

Compiled progam (image, *.elf)

```
1 ...
2 0x..0: 1. Word: Code for ADD
3 ...
```

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

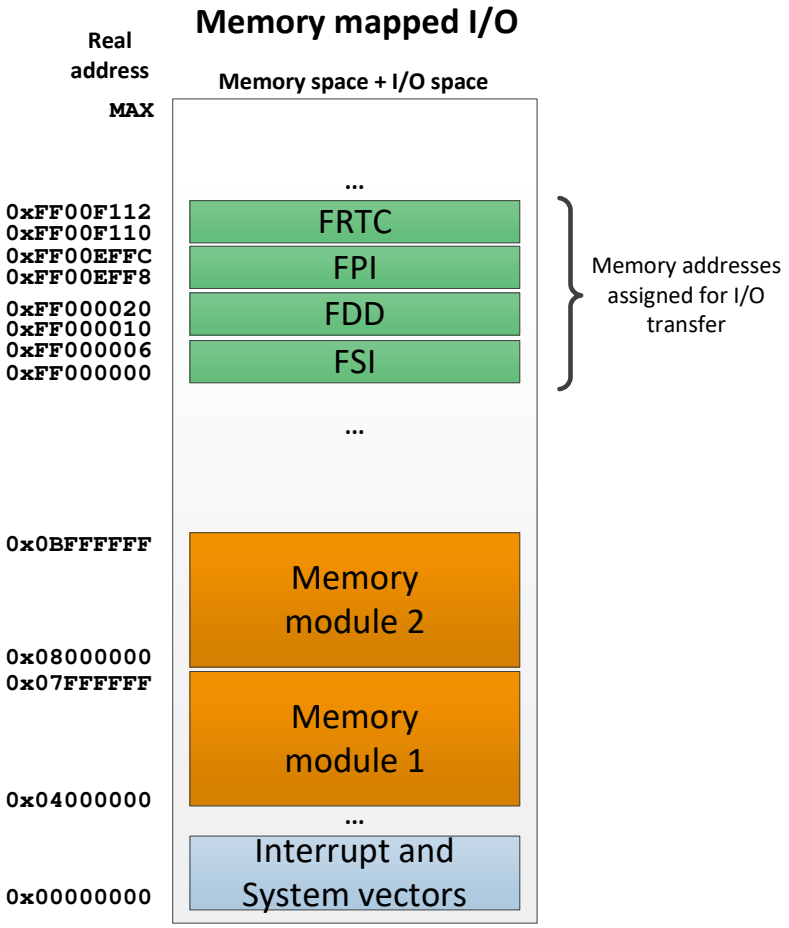
Access I/O devices

Memory mapped vs isolated I/O

CAMPUS Rosenheim
Computer Science

Rosenheim Technical University of Applied Sciences

Memory mapped I/O (MMIO)



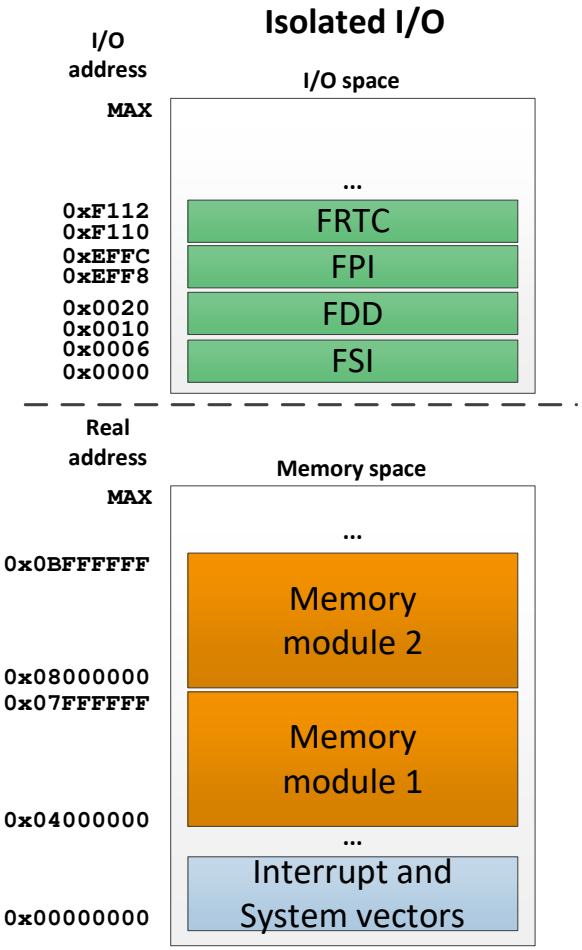
Properties:

- Everything is mapped into **one address space**
- **Addresses** an **I/O device** just **like ordinary memory**
- **Only** a **MOVE instruction** is **required** for data transfer
- Almost **all instructions** used to manipulate the memory **can be used** for I/O devices
- Use of **fixed, absolute addresses** for the device registers

CAMPUS Rosenheim
Computer Science

Rosenheim Technical University of Applied Sciences

Isolated I/O



Properties:

- Two **separate address spaces**: for memory and I/O
- The **I/O** address space has its **own addresses**
- **Requires special instructions** for data transfer: e.g. IN and OUT
- The I/O addresses **can't be used in** the **instructions** used to manipulate the memory
- Also called port-mapped I/O (PMIO)

Memory mapped vs isolated I/O

	Memory mapped I/O		Isolated I/O	
Address spaces	**I/O** registers in the memory address space are **addressed like normal memory** cells.	–	**Separate address spaces** for I/O and memory.	+
Addresses	Memory mapped **I/O** devices are **treated as memory** locations.		The addresses for the isolated **I/O** devices are called **ports**.	
Memory	**Part** of the address space is **reserved for I/O** registers.	–	**Full memory** address space **usable** for memory.	+
Instructions	Almost **any instruction** for memory access can be used.	+	**Special instructions** such as `IN` or `OUT` has to be used.	–
Protection	Access protection integrated into memory protection	+	Separate access protection required "I/O Permission Bit Map", IOPL in EFLAGS.	–
Caching	Additional **cache prevention necessary**	–	No inadvertently caching possible	+

CAMPUS Rosenheim
Computer Science

Rosenheim
Technical University
of Applied Sciences

Summary and outlook

Summary

- Bus systems
- F-Bus
- Bus cycles
- Program sequence and bus cycles
- Access I/O devices

Outlook

- I/O programming modes
- Interrupts
- DMA bus cycle
- FSI and FDD (DMA) programming example