

Prof. Dr. Florian Künzner

Technical University of Applied Sciences Rosenheim, Computer Science

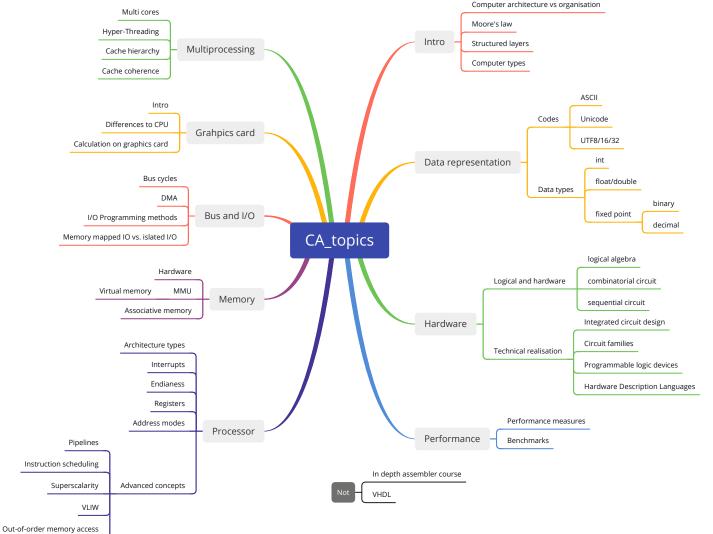
CA 5 - Processor 2

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

Computer Science



Goal



Computer Science





- Endianness
- Examples
- Usage
- Transfer
- Solutions



CAMPUS Rosenheim Computer Science



Endianness

Endianness: The definition of the **byte order** within a **multi-byte** word.

CAMPUS Rosenheim Computer Science



Endianness

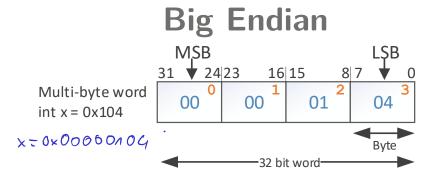
Endianness: The definition of the **byte order** within a **multi-byte** word.

Computer Science



Endianness

Endianness: The definition of the **byte order** within a **multi-byte** word.

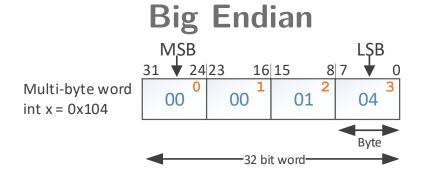


Computer Science



Endianness

Endianness: The definition of the **byte order** within a **multi-byte** word.



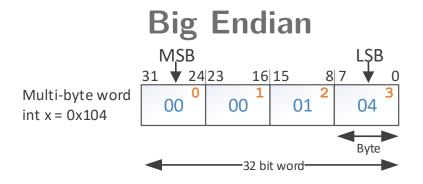
- MSB Most significant byte
- LSB Least significant byte

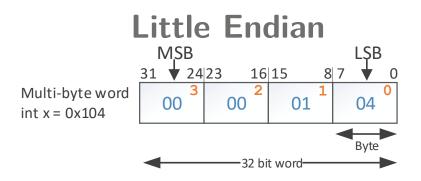
Computer Science



Endianness

Endianness: The definition of the **byte order** within a **multi-byte** word.



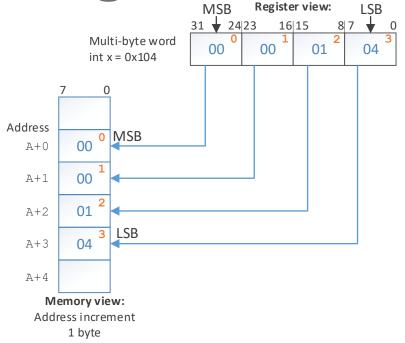


- MSB Most significant byte
- LSB Least significant byte

Computer Science



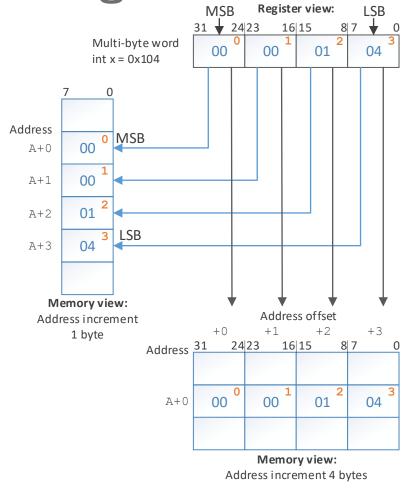
Endianness - Big endian (first)



Computer Science



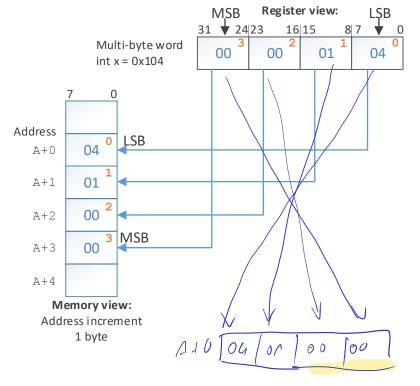
Endianness - Big endian



Computer Science



Endianness - Little endian (first)

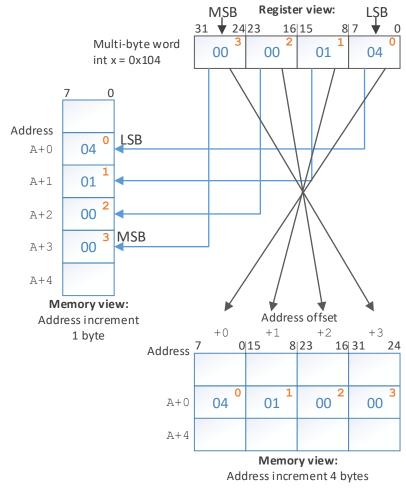


Computer Science



Slide 6 of 17

Endianness - Little endian

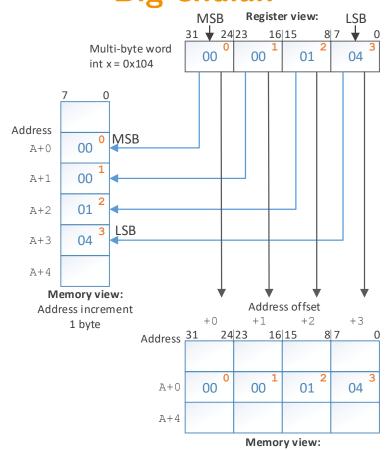


Computer Science



Endianness - BE/LE

Big endian



Address increment 4 bytes

Little endian LSB 31 \$\rightarrow\$ 24|23 16 15 Multi-byte word 00 04 00 01 int x = 0x104Address A+001 A+100 A+23 MSB 00 A+3A+4Memory view: Address offset Address increment 1 byte 8|23 16|31 Address 01 00 A+004 00 A+4Memory view:

Computer Science



Endianness - example BE

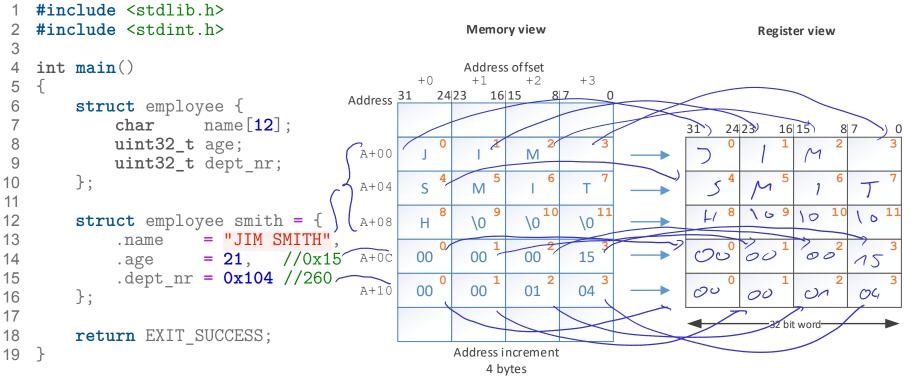
```
#include <stdlib.h>
   #include <stdint.h>
   int main()
       struct employee {
           char name [12];
           uint32_t age;
           uint32_t dept nr;
       };
12
       struct employee smith = {
                    = "JIM SMITH",
13
            .name
                    = 21, //0x15
14
           .age
15
           .dept nr = 0x104 //260
       };
16
17
18
       return EXIT SUCCESS;
19
  }
   [cmp: [1, p. 95-96]]
```

Computer Science



Endianness - example BE

Big endian memory -> Register



[cmp: [1, p. 95-96]]

Computer Science



Endianness - example BE

Big endian memory -> Register

```
#include <stdlib.h>
    #include <stdint.h>
                                                          Memory view
                                                                                               Register view
    int main()
                                                          Address offset
                                                    +0
                                                           +1
                                                                         +3
                                           Address 31
                                                       24|23
                                                             16 15
                                                                     8 7
         struct employee {
                         name [12];
              char
                                                                                           24|23
                                                                                                  16 15
              uint32_t age;
                                                                                                                3
                                             A + 0.0
                                                                  M
                                                                                                       M
              uint32_t dept nr;
         };
                                             A+04
                                                     S
                                                           M
                                                                         Т
                                                                                         S
                                                                                                M
                                                                                                              Т
                                                                    10
                                                                           11
12
         struct employee smith = {
                                                                                                             \0
                                                                         \0
                                                                                                \0
                                                                                                      \0
                                             A+08
                                                           \0
                                                                  \0
                                                                                         Н
                         = "JIM SMITH",
13
              .name
                                                                         15
                                                                                                      00
                                                                                                             15
                         = 21, //0x15
                                                    00
                                                           00
                                                                  00
                                                                                         00
                                                                                               00
14
                                             A+0C
              .age
15
              .dept nr = 0x104 //260
                                                                         04
                                                    00
                                                           00
                                             A+10
                                                                  01
                                                                                         00
                                                                                               00
                                                                                                      01
                                                                                                             04
16
         };
17
                                                                                                 -32 bit word-
18
         return EXIT SUCCESS;
   }
                                                         Address increment
19
                                                             4 bytes
```

[cmp: [1, p. 95-96]]

Computer Science



Endianness - example LE

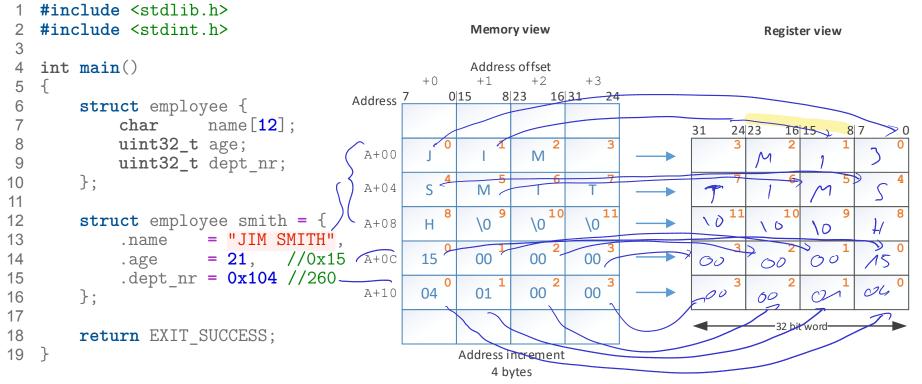
```
#include <stdlib.h>
   #include <stdint.h>
   int main()
       struct employee {
           char name [12];
           uint32_t age;
           uint32_t dept nr;
       };
12
       struct employee smith = {
            .name = "JIM SMITH",
13
                    = 21, //0x15
14
           .age
15
           .dept nr = 0x104 //260
       };
16
17
18
       return EXIT SUCCESS;
19
  }
   [cmp: [1, p. 95-96]]
```

Computer Science



Endianness - example LE

Little endian memory -> Register



[cmp: [1, p. 95-96]]

Computer Science



Endianness - example LE

Little endian memory -> Register

```
#include <stdlib.h>
    #include <stdint.h>
                                                          Memory view
                                                                                              Register view
    int main()
                                                          Address offset
                                                    + 0
                                                           +1
                                                                        +3
                                                              8 23
                                                                   16 31
                                           Address 7
         struct employee {
              char
                         name [12];
                                                                                          2423
                                                                                                 16 15
              uint32_t age;
                                            A + 00
                                                                 M
                                                                                               M
              uint32_t dept nr;
         };
                                                    S
                                            A + 0.4
                                                                         Т
                                                          M
                                                                                        Т
                                                                                                     M
                                                                       \0 11
                                                                                              \0 10
                                                                                       11
                                                                   10
12
         struct employee smith = {
                                                          \0
                                            A+08
                                                    Н
                                                                 \0
                                                                                                     \0
                                                                                                            Н
                         = "JIM SMITH",
13
              .name
                         = 21, //0x15
                                                                        00
14
                                                   15
                                                          00
                                                                 00
                                                                                        00
                                                                                               00
                                                                                                     00
                                                                                                            15
              .age
                                            A+0C
15
              .dept nr = 0x104 //260
                                                                 00
                                                                        00
                                                                                        00
                                                                                               00
                                                                                                     01
                                                          01
                                                                                                            04
                                            A+10
                                                    04
16
         };
17
                                                                                                -32 bit word-
18
         return EXIT SUCCESS;
19
   }
                                                        Address increment
                                                            4 bytes
```

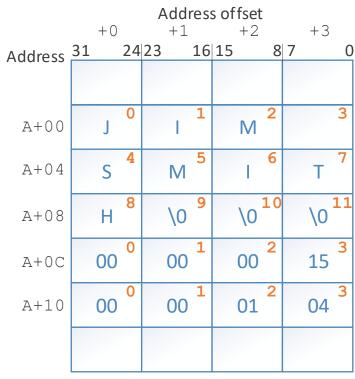
[cmp: [1, p. 95-96]]

Computer Science



Endianness - example BE/LE

Big endian memory



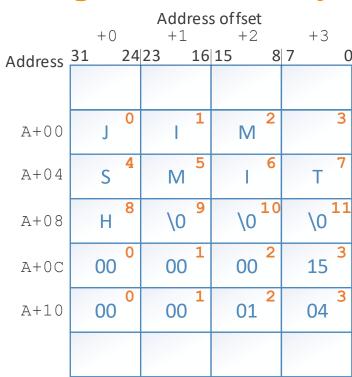
Address increment 4 bytes

Computer Science



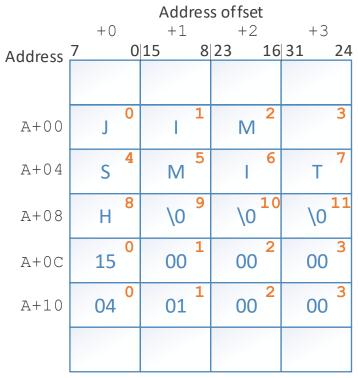
Endianness - example BE/LE

Big endian memory



Address increment 4 bytes

Little endian memory



Address increment 4 bytes

[cmp: [1, p. 95-96]]

Prof. Dr. Florian Künzner, SoSe 2022

Computer Science



Endianness - usage

Big endian

- IBM Mainframe
- Freescale ColdFire
- Atmel AVR/AVR32
- ARM Thumb and ARM64 (also Apple M1)

Little endian

- Intel x86
- x86-64 (AMD64, Intel 64)
- RISC-V
- Qualcomm Hexagon

Computer Science



Endianness - usage

Big endian

- IBM Mainframe
- Freescale ColdFire
- Atmel AVR/AVR32
- ARM Thumb and ARM64 (also Apple M1)

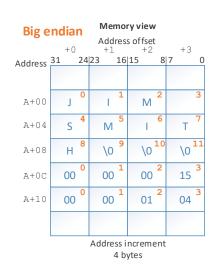
Little endian

- Intel x86
- x86-64 (AMD64, Intel 64)
- RISC-V
- Qualcomm Hexagon

Computer Science



Endianness - transfer: BE to LE



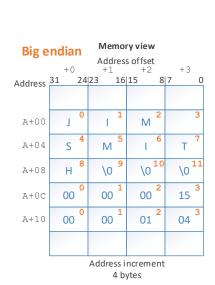
[cmp: [1, p. 95-96]]

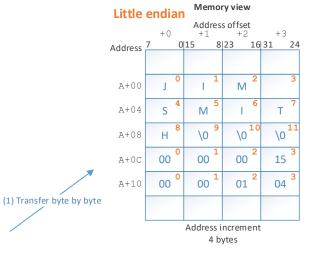
Prof. Dr. Florian Künzner, SoSe 2022

Computer Science



Endianness - transfer: BE to LE





[cmp: [1, p. 95-96]]

Prof. Dr. Florian Künzner, SoSe 2022

Goal Endianness Examples Usage Transfer Solutions Summary

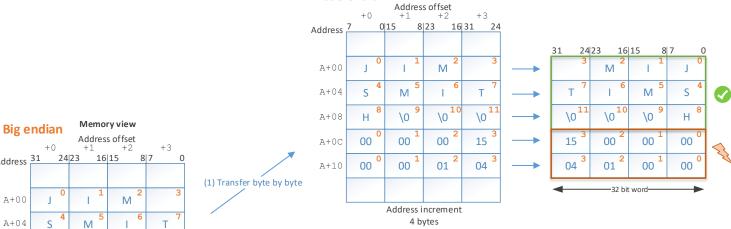
CAMPUS Rosenheim

Computer Science

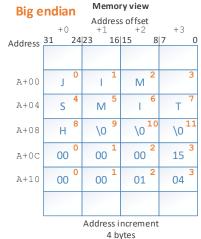


Register view

Endianness - transfer: BE to LE



Little endian



Little endian

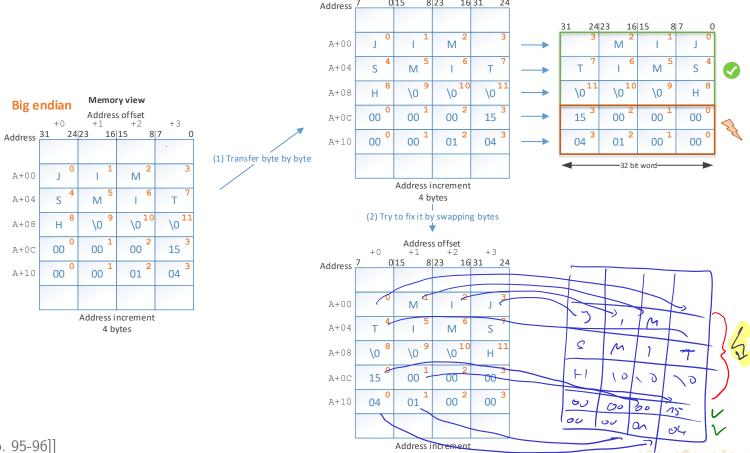
Address offset +1 +2 +3 0|15 8|23 16|31 24

CAMPUS Rosenheim

Computer Science



Endianness - transfer: BE to LE



[cmp: [1, p. 95-96]] Prof. Dr. Florian Künzner, SoSe 2022

CA 5 – Processor 2 Slide 12 of 17

Little endian

Address offset

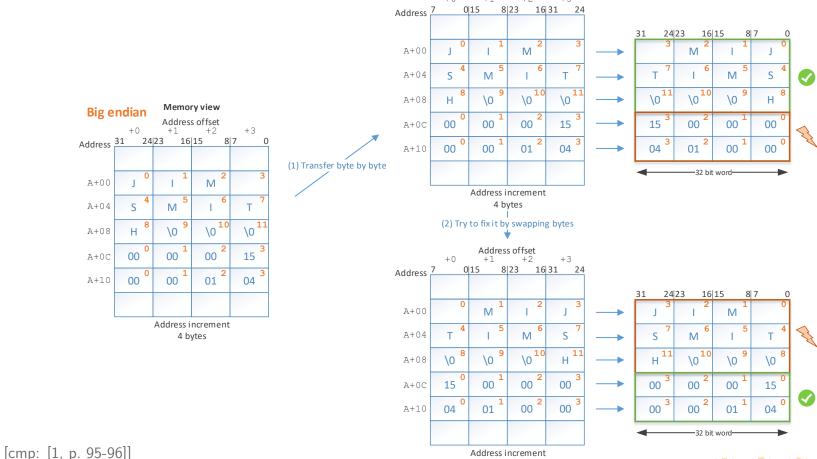
CAMPUS Rosenheim

Computer Science



Register view

Endianness - transfer: BE to LE



Prof. Dr. Florian Künzner, SoSe 2022

CA 5 – Processor 2

Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types
- lacktriangle Data type consists of more than one byte (multi-byte word, \geq 2)
- Data are transferred between BE/LE systems

Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types

CAMPUS Rosenheim Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types
- Data type consists of more than one byte (multi-byte word, ≥ 2)
- Data are transferred between BE/LE systems

Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types
- Data type consists of more than one byte (multi-byte word, ≥ 2)
- Data are transferred between BE/LE systems

Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types
- Data type consists of more than one byte (multi-byte word, ≥ 2)
- Data are transferred between BE/LE systems

- Single-byte data is transferred byte by byte (e.g. ASCII)
- lacksquare Data is transferred within same endianness (LE -> LE, BE -> BE)

Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types
- Data type consists of more than one byte (multi-byte word, ≥ 2)
- Data are transferred between BE/LE systems

- Single-byte data is transferred byte by byte (e.g. ASCII)
- lacksquare Data is transferred within same endianness (LE -> LE, BE -> BE)

Computer Science



Endianness - problem

Problem can occur if

- Different data types are mixed: numbers, strings, or other data types
- Data type consists of more than one byte (multi-byte word, ≥ 2)
- Data are transferred between BE/LE systems

- Single-byte data is transferred byte by byte (e.g. ASCII)
- Data is transferred within same endianness (LE -> LE, BE -> BE)

CAMPUS Rosenheim Computer Science



Endianness - conclusion

Without the knowledge about the data types and the endianness, a transfer between BE/LE systems is not feasible.

Tanenbaum: "There is no easy solution to this" [1, p. 96]



Endianness - conclusion

Without the knowledge about the data types and the endianness, a transfer between BE/LE systems is not feasible.

Tanenbaum: "There is no easy solution to this" [1, p. 96]





Endianness - possible solutions

- **Know** the endianness (e.g. **meta data!**)
- Transfer byte by byte (no problem for single-byte data)
- If endianness is different and a multi-byte word is transferred: additionally swap the bytes





Endianness - possible solutions

- Know the endianness (e.g. meta data!)
- Transfer byte by byte (no problem for single-byte data)
- If endianness is different and a multi-byte word is transferred: additionally swap the bytes

Computer Science



Endianness - possible solutions

- Know the endianness (e.g. meta data!)
- **Transfer** byte by byte (no problem for single-byte data)



Endianness - possible solutions

- Know the endianness (e.g. meta data!)
- Transfer byte by byte (no problem for single-byte data)
- If endianness is different and a multi-byte word is transferred: additionally swap the bytes

Computer Science



Endianness - solutions

- Network order: always BE
- Java: always BE; for transfer with others, ByteOrder can be set
- Unicode UTF-16/32: uses a BOM (byte order mark)
- TIF files: BE/LE identifier in header
- RPC (remote procedure call): marshalling (data as byte stream) solves the problem by using meta data

Computer Science



Endianness - solutions

- Network order: always BE
- Java: always BE; for transfer with others, ByteOrder can be set
- Unicode UTF-16/32: uses a BOM (byte order mark)
- TIF files: BE/LE identifier in header
- RPC (remote procedure call): marshalling (data as byte stream) solves the problem by using meta data

Computer Science



Endianness - solutions

- Network order: always BE
- Java: always BE; for transfer with others, ByteOrder can be set
- Unicode UTF-16/32: uses a BOM (byte order mark)
- TIF files: BE/LE identifier in header
- RPC (remote procedure call): marshalling (data as byte stream) solves the problem by using meta data

Computer Science



Endianness - solutions

- Network order: always BE
- Java: always BE; for transfer with others, ByteOrder can be set
- Unicode UTF-16/32: uses a BOM (byte order mark)

Computer Science



Endianness - solutions

- Network order: always BE
- Java: always BE; for transfer with others, ByteOrder can be set
- Unicode UTF-16/32: uses a BOM (byte order mark)
- TIF files: BE/LE identifier in header



Endianness - solutions

- Network order: always BE
- Java: always BE; for transfer with others, ByteOrder can be set
- Unicode UTF-16/32: uses a BOM (byte order mark)
- TIF files: BE/LE identifier in header
- RPC (remote procedure call): marshalling (data as byte stream) solves the problem by using meta data

Computer Science



Summary and outlook

Summary

- Endianness
- Examples
- Usage
- Transfer
- Solutions

Outlook

- Processor registers
- Processor examples
- Addressing modes

Computer Science



Summary and outlook

Summary

- Endianness
- Examples
- Usage
- Transfer
- Solutions

Outlook

- Processor registers
- Processor examples
- Addressing modes