Organisatorisches

- Meine Website: http://www.cs.ubbcluj.ro/~dianat/
- Meine E-mail Adresse: dianat [at] cs.ubbcluj.ro
- Für Fragen: Tutorium und Sprechstunde auf Teams (Code 666wwoj)
  - Montag, 14.00 Uhr
  - Dienstag, 16.00 Uhr

- Gruppenvertreter?

Anwesenheit

- Seminar: **75% Anwesenheit** (maximal 2 unmotivierte Abwesenheiten)
- **Aktive** Anwesenheit erforderlich!
- Moodle (https://moodle.cs.ubbcluj.ro/) – **Rechnerarchitektur**
- **Aufpassen!** Ab nächste Woche wird die **Anwesenheit auf Moodle** überprüft. Ihr müsst bis dann alle in dem Moodle Kurs angemeldet sein!

Seminar 1

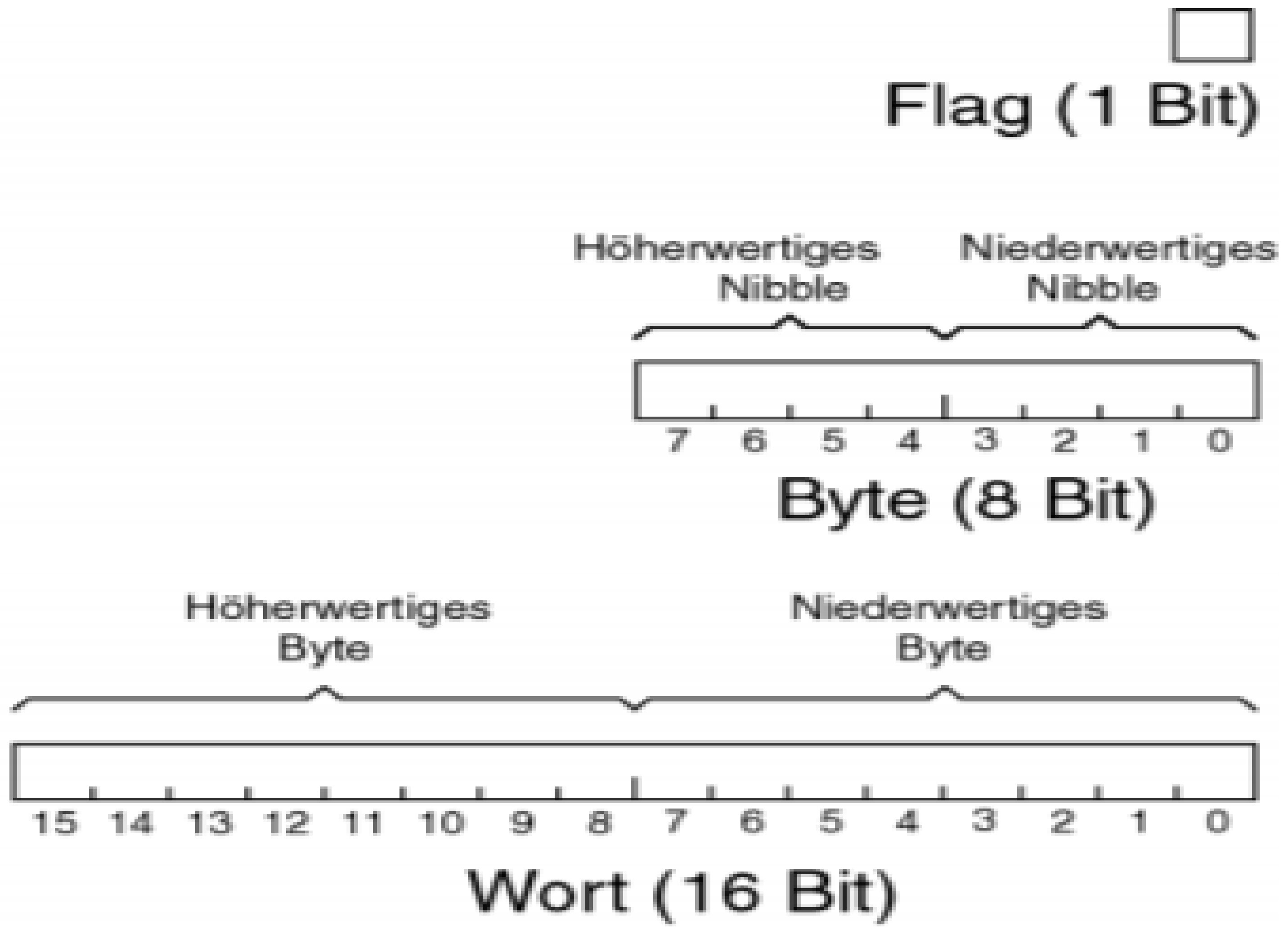
Rechnerarchitektur

Darstellung von ganzen Zahlen

- Ein Mikroprozessorsystem verarbeitet immer Bitmuster in Einheiten zu 8, 16, 32 oder mehr Bit
- Die **kleinste Informationseinheit**, das **Bit**, hat den Wert 1 oder 0.
- Erst durch die Art der Verarbeitung wird diesem Bitmuster eine bestimmte Bedeutung zugewiesen.
- Wenn einen arithmetischen Maschinenbefehl auf ein Bitmuster angewendet wird, dann wird es als Zahl interpretiert

Bit. Byte. Wort. Doppelwort. Quadwort

- Eine Einheit aus **4 Bit** heißt **Tetrade** oder **Nibble**.
- Eine **8 Bit-Einheit** heißt **Byte** .
- Ein **Wort (word) ist eine 16 Bit-Einheit.**
- Ein **Doppelwort ist eine 32 Bit-Einheit.**
- Ein **Quadwort ist eine 64 Bit-Einheit.**
- Innerhalb einer Einheit sind die Bits von rechts nach links nummeriert
- Das niederwertigste Bit, das **Least significant Bit**, abgekürzt das LSB , ist immer Bit 0.
- Das höchstwertige Bit, das **Most significant Bit** , abgekürzt das MSB , ist bei einem Byte Bit 7, bei einem 16 Bit-Wort Bit 15, usw.


Flag (1 Bit)
Höherwertiges Nibble
Niederwertiges Nibble
7 6 5 4 3 2 1 0
Byte (8 Bit)
Höherwertiges Byte
Niederwertiges Byte
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Wort (16 Bit)

Dezimalsystem (Basis 10)

- Im Dezimalsystem werden 10 verschiedene Ziffern mit Potenzen der Zahl 10 gewichtet
- Eine Dezimalzahl mit **n Ziffern** hat den Wert:

$$\boldsymbol{Z = \sum_{i=0}^{n-1} a_i * 10^i}$$

- z.B. 123 = 1 * 10^2 + 2 * 10^1 + 3 * 10^0

Binärsystem (Basis 2)

- In der Mikroprozessortechnik haben die kleinsten Speichereinheiten, die Bits, nur zwei Zustände: 0 oder 1
- Der Wert einer vorzeichenlosen **Binärzahl** (**unsigned binary numbers**) ist:

$$Z = \sum_{i=0}^{n-1} a_i * 2^i$$

- z.B. 11100101b = $1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 0 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0$
- Um Binärzahlen von Dezimal- u.a. Zahlen zu unterscheiden, wird an die Ziffernfolge der Buchstabe **’b’ angehängt** (1101b) oder **die Zahlenbasis als tiefgestellter Index** (1101_2).

Hexadezimalsystem (Basis 16)

- Da nur 10 Zahlenzeichen zur Verfügung stehen, verwendet man die ersten sechs Buchstaben des Alphabets für die Zahlen 10 bis 15:
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Standardeinheit der Informationsgröße ist ein Byte = 8 Bit
- In einer Hexadezimalzahl **entsprechen** immer **genau zwei Ziffern einem Byte** ($256 = 16^2$)

Umwandeln von Dezimalzahlen in eine andere Zahlenbasis

1. Dividieren Sie die Dezimalzahl durch die Ziel-Zahlenbasis und merken Sie sich das Ergebnis und den Restwert.
2. Dividieren Sie weiter das vorige Ergebnis durch die Ziel-Zahlenbasis bis Sie ein Ergebnis von 0 erhalten.
3. Der Wert in der Ziel-Zahlenbasis ergibt sich aus den Restwerten in umgekehrte Reihenfolge.

Übung (ohne Rechner):

Wandle die Zahl 347 in die Basis 16 um.

Wandle die Zahl 57 in die Basis 2 um.

Umwandeln von Zahlen aus einer beliebigen Zahlenbasis in die Dezimalbasis

Für die Zahl:

$$\mathbf{Nr(b) = C_n\, C_{n-1}\, C_{n-2} \ldots C_2 C_1\, C_0}$$

ist der Wert in der Dezimalbasis:

$$\mathbf{Nr(10) = C_n * b^n + C_{n-1} * b^{n-1} + \ldots + C_2 * b^2 + C_1 * b^1 + C_0}$$

Übungen:

- Wandle die Zahl $3A8_{(h)}$ in die Dezimalbasis um
- Wandle die Zahl $86C_{(h)}$ in die Dezimalbasis um
- Wandle die Zahl $1101101_{(2)}$ in die Dezimalbasis um

Umwandeln von Binär- in Hexadezimalzahlen und umgekehrt

Hexadezimal	**Binär**	**Hexadezimal**	**Binär**
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Umwandeln von Binär- in Hexadezimalzahlen und umgekehrt

- **Vom Binär- ins Hexadezimalsystem:**
  - Unterteile die Binärzahl **von rechts nach links** in 4er-Päckchen, und wandle jedes Päckchen nach nebenstehender Tabelle in die entsprechende Hexadezimalziffer um.
- **Vom Hexadezimal- ins Binärsystem:**
  - Wandle die Hexadezimalziffern der Reihe nach in die entsprechenden vierstelligen Binärzahlen um.

Übungen:

- Wandle die Zahl $1B5_{(h)}$ in die Basis 2 um
- Wandle die Zahl $101010_{(2)}$ in die Basis 16 um

Repräsentierung der negativen Zahlen

- Bei der alltäglichen Benutzung können wir einfach ein Minus vor die Dezimalzahl setzen. Können wir dies auch mit Binärzahlen machen?
- Eine Lösung wäre ein Bit für die Darstellung des Vorzeichens zu benutzen: **Vorzeichenbit (sign bit)**
- Das Vorzeichenbit hat den Wert **1 für negative Zahlen** und den Wert **0 für positive Zahlen**.
- Das Vorzeichenbit ist das **Most Significant Bit (das erste Bit von links).**

Direkter Kod

- Man könnte den positiven Wert darstellen und das Most significant Bit auf 0 für positive oder auf 1 für negative Zahlen setzen
- z.B. repräsentiere 5 und -5 auf einem Byte:

$$5 = 00000101_{(2)}$$

$$-5 = 10000101_{(2)}$$

- **Problem:**

$$5 + (-5) = 10001010_{(2)} \neq 0$$

Einerkomplement

- Alle positiven Zahlen werden normal binär repräsentiert
- Alle negative Zahlen werden invertiert, d.h. alle Ziffern der Darstellung des positiven Wertes werden negiert

$$7 = 0111_{(2)}$$

$$-7 = 1000_{(2)}$$

- **Problem**:

$$7 - 7 = 1111_{(2)} \neq 0$$

Zweierkomplement

- Äquivalente Komplementierungsregeln:
  I. Die Vorzeichenumkehr einer Zahl im Zweierkomplement wird bewirkt durch Invertieren aller Bits und anschließendes Inkrementieren.
  II. Von rechts angefangen, alle Nullen und die erste Eins abschreiben und alle nachfolgenden Stellen invertieren.
  III. Subtraktion von der Wertebereichsgrenz: um das Komplement für eine binäre Zahl mit n Ziffern zu berechnen, subtrahiert man binär die Zahl aus $10\ldots0_{(b)}$, wobei diese Zahl n 0-Ziffern hat
  IV. Subtraktion von der Wertebereichsgrenz: um das Komplement für eine hexadezimale Zahl mit n Ziffern zu berechnen, subtrahiert man hexadezimal die Zahl aus $10\ldots0_{(h)}$, wobei diese Zahl n 0-Ziffern hat
- Wie werden die Zahlen 3 und -3 in dem Zweierkomplement repräsentiert? Führt -3 + 3 zu dem Ergebnis 0?

Zweierkomplement

- Vorteile der Zweierkomplement:
  - Die **Additionsoperation wird gleich ausgeführt**, egal ob es um **negative oder positive Zahlen** geht. Die Addition ist eine **normale bitweise Addition**, wo der Stellenüberlauf ignoriert wird. Das Ergebnis wird mit gleicher Stellenanzahl wie Minuend und Subtrahend interpretiert.

  - Die Subtraktion lässt sich auf eine Negation mit anschließender Addition zurückführen.

Zweierkomplement

1. Berechne das Zweierkomplement für die Zahl $(18)_{10}$ repräsentiert auf einem Byte mit allen vier Methoden.
2. Überprüfe mit mehreren Regeln, dass:

a) die Zahlen $(9A7D)_{16}$ und $(7583)_{16}$ repräsentiert auf 2 Bytes komplementär sind.

b) die Zahlen $(000F095D)_{16}$ und $(FFF0F6A3)_{16}$ repräsentiert auf 4 Bytes komplementär sind.

c) die Zahlen $(7F)_{16}$ und $(81)_{16}$ repräsentiert auf 1 Byte komplementär sind.

Darstellung der ganzen Zahlen

- Eine Zahl zwischen **-2^{n-1} und $2^{n-1} - 1$** wird auf **n Bits** folgendermaßen repräsentiert:
  - falls die Zahl **positiv** ist, dann wird sie **durch den binären Wert dargestellt**
  - falls die Zahl **negativ** ist, dann wird **sie durch den Zweierkomplement** der binären Wert **dargestellt**

Bemerkung. Die Zahl **-2^{n-1}** kann nicht auf n-1 Bits dargestellt werden!

- **-2^{n-1}** wird **auf n Bits als 10…0** dargestellt, wobei die Most Significant Bit 1 ist, da die Zahl negativ ist.
- Die Zahl 10…0 hat sich selbst als Komplement, also der Wert ist genau **-2^{n-1}**
- Dieselbe Zahl 10…0 interpretiert als positive Zahl ist die Zahl **2^{n-1}**.

Darstellungsbereiche

- Da in der Mikroprozessortechnik immer die Bitstellenzahl begrenzt ist, ist auch der Zahlenbereich begrenzt
- Zahlen außerhalb dieses Bereichs sind nicht darstellbar und eine Operation, deren **Ergebnis über eine der Grenzen hinausführt, ergibt ein falsches Ergebnis**
- Diese **Bereichsüberschreitung** wird vom Mikroprozessor mit dem **Übertragsflag** (**Carryflag**) angezeigt.

Darstellungsbereiche

Anzahl Bytes	Nicht signierte Interpretation (**vorzeichenlosen**, unsigned representation)	Signierte Interpretation (**vorzeichenbehafteten**, signed representation)
1	$[0, 2^8-1]$ = [0 , 255]	$[-2^7 , 2^7-1]$ = [-128 , 127]
2	$[0, 2^{16}-1]$ = [0 , 65535]	$[-2^{15} , 2^{15}-1]$ = [-32 768 , 32 767]
4	$[0, 2^{32}-1]$ = [0 , 4 294 967 295]	$[-2^{31} , 2^{31}-1]$ = [-2 147 483 648 , 2 147 483 647]
8	$[0, 2^{64}-1]$ = [0 , 18 446 824 753 389 551 615]	$[-2^{63} , 2^{63}-1]$ = [-9 223 412 376 694 775 808 , 9 223 412 376 694 775 807]

Darstellungsbereiche

- Wie wird eine Zahl auf 7 Bits auf 8 Bits dargestellt?
  - **es hängt von der Interpretation ab!**

- In der **vorzeichenlosen** Interpretation werden die restlichen höherwertige Bits (high bits) **mit Nullwerte ausgefüllt**.
- In der **vorzeichenbehafteten** Interpretation werden die restlichen höherwertige Bits (high bits) **mit dem Vorzeichenbit** (sign bit) **ausgefüllt**.

Beispiel. Wie wird $(10011)_2$ auf einem Byte dargestellt?

Übungen

1. Repräsentiere folgende Zahlen auf einem Byte:
   - 10, -10, -1

Überlege wie dieselben Zahlen auf einem Word repräsentiert werden.

2. Wandle die Zahl 347 ins Binär- und Hexadezimalsystem um.
3. Wandle die gegebene Zahlen aus Binär- ins Hexadezimalsystem oder umgekehrt:
   - $10\ 0101\ 1010\ 1011_{(2)}$
   - $49A02_{(16)}$
   - $9A7D_{(16)}$
   - $1\ 0010\ 1110_{(2)}$

😊

Es gibt 10 Gruppen von Menschen: diejenigen, die das Binärsystem verstehen, und die anderen.