

Digitaltechnik & Rechnersysteme

Einführung, Information & Kodierung

Martin Kumm



WiSe 2022/2023

Wer bin ich?



- Martin Kumm
- Prof. für *Embedded Systems*
- Sprechstunde: Nach Absprache per Mail
- Raum 124, Gebäude 46 (E)
- Mail: martin.kumm@ai.hs-fulda.de
- WebEx: <https://hs-fulda.webex.com/meet/martin.kumm> (**nach Absprache**)



Veranstaltungsübersicht

- 2 SWS Vorlesung
 - 2 SWS Übung
 - Angebot: Vorlesung jedes Wintersemester, Klausur jedes Semester
 - Schriftliche Klausur am Ende des Semesters, 90 Minuten
 - Hilfsmittel:
 - Nicht programmierbarer Taschenrechner
 - Handgeschriebene Formelsammlung:
ein beidseitig beschriebenes DIN-A4 Blatt

Verwendete Tools

Zur Organisation der Veranstaltung wird Moodle und Discord verwendet:

- **Moodle:** Zentraler Anlaufpunkt für
 - Unterlagen (Vorlesungsfolien, Übungsblätter)
 - Vorlesungsaufzeichnungen
 - Abgabe von Hausübungen

»Digitaltechnik und Rechnersysteme - AI1002 (WiSe22/23)«

⇒ <https://elearning.hs-fulda.de/ai>

 - **Discord:** Für alle asynchronen Fragen zu Übungen

Discord



<https://discord.gg/9x9UFn65>

Übungen I



Es vier Übungsgruppen:

- ① Mo 15:30 – 17:00, Raum 31.005
Übungsleiter: Jonas Kühle, Stud. Tutor: Raffael Krakau
 - ② Mo 13:45 – 15:15, Raum 31.005
Übungsleiter: Jonas Kühle, Stud. Tutor: Raffael Krakau
 - ③ Fr 09:50 – 11:20, Raum 46.105
Übungsleiter: Tobias Habermann, Stud. Tutor: David Lemming
 - ④ Fr 11:40 – 13:10, Raum 46.105
Übungsleiter: Tobias Habermann, Stud. Tutor: Anton Pieper

Jedes Übungsblatt ist aufgeteilt in

- Gruppenübung: Erlernen eines neuen Themengebietes
 - Hausübung: Weitere Vertiefung mit Lernerfolgskontrolle

Übungsablauf



Die Gruppenübungen werden von Ihnen asynchron bearbeitet

Gruppenübungen werden besprochen und es gibt Musterlösungen

Hausübungen werden online abgegeben und bewertet

Ablauf:

- Woche n , **Vorlesung, Bearbeitung Gruppenübung Thema n**
- Woche $n + 1$, **Besprechung Gruppenübung Thema n ,**
Vorbesprechung Hausübung Thema n
- Woche $n + 1$: **Bearbeitung Hausübung Thema n**
- Woche $n + 1$: **So 23:59, Abgabe Hausübung Thema n**
- Woche $n + 2$, Fr: Korrektur zu Hausübung Thema n
abgeschlossen

blau: an der Hochschule

grün: zu Hause/in Lerngruppe

Hausübungen



- Je Hausübung 10 Punkte erreichbar, 130 Punkte insgesamt
- Alle Übungsaufgaben und deren Lösung werden **von Ihnen** während der Gruppenübung vorgestellt
- 10 Zusatzpunkte bei Vorstellung einer Aufgabe
- **Bonussystem:** bei 100 Punkten oder mehr,
Notenverbesserung um eine Stufe bei bestandener Klausur
(0,3 oder 0,4 besser)
- Gesamtpunktzahl: Punkte in Hausübung + Punkte durch Vorstellung
- Wer die Aufgabe vorstellt wird am Anfang der Übung festgelegt, jeder kann sich freiwillig melden

Geplanter Ablauf



Termin*	Datum	Inhalt Vorlesung	Bearb. GÜ	Besp. GÜ	Abg. HÜ	Besp. HÜ	Korr. HÜ
1	25.10.22	Orga + Motivation + Information & Kodierung	1				
2	01.11.22	Zahlenkodierung	2	1	06.11.22		
3	08.11.22	Boolesche Algebra, Schaltfunktionen	3	2	13.11.22	1	1
4	15.11.22	Schaltfunktionen, Darstellungsalternativen, Entwicklungssatz, Basissysteme	4	3	20.11.22	2	2
5	22.11.22	KV-Diagramme, Minimierung	5	4	27.11.22	3	3
6	29.11.22	don't care, Arithmetik	6	5	04.12.22	4	4
7	06.12.22	Schaltwerke, Moore/Mealy	7	6	11.12.22	5	5
8	13.12.22	Latches, FFs, Synchrone Automaten	8	7	18.12.22	6	6
9	20.12.22	Automatenentwurf	9	8	08.01.23	7	7
10	10.01.23	Zeitverhalten Gatter, Speicher	10	9	15.01.23	8	8
11	17.01.23	Speicher / Minimal-Prozessor I	11	10	22.01.23	9	9
12	24.01.23	Minimal-Prozessor II, Rechner	12	11	29.01.23	10	10
13	31.01.23	MIPS-Prozessor, MIPS-Assembler	13	12	05.02.23	11	11
14	07.02.23	Ausblick / wrap-up		13	12.02.23	12	12

* Ablauf vorläufig, Inhalte können sich verschieben

Literatur zu Teil 1 - Digitaltechnik



- Lipp, H. M., Becker J.: *Grundlagen der Digitaltechnik*; Oldenbourg Verlag; 7. verb. Aufl.; 2011; ISBN 978-3-486-70693-2 ⇒ [Online aus Compusnetz verfügbar](#)
- Mano, M. Morris and Ciletti, Michael D.: *Digital Design*; Pearson International Edition; 4. Aufl.; 2007; ISBN 0132340437

Literatur zu Teil 1 - Rechnerarchitektur

- Hennessy, J. L. und Patterson, D. A.: *Computer architecture: a quantitative approach*, Morgan Kaufmann, 2012
⇒ [Online aus Compusnetz verfügbar](#)

bzw. die deutsche Übersetzung:

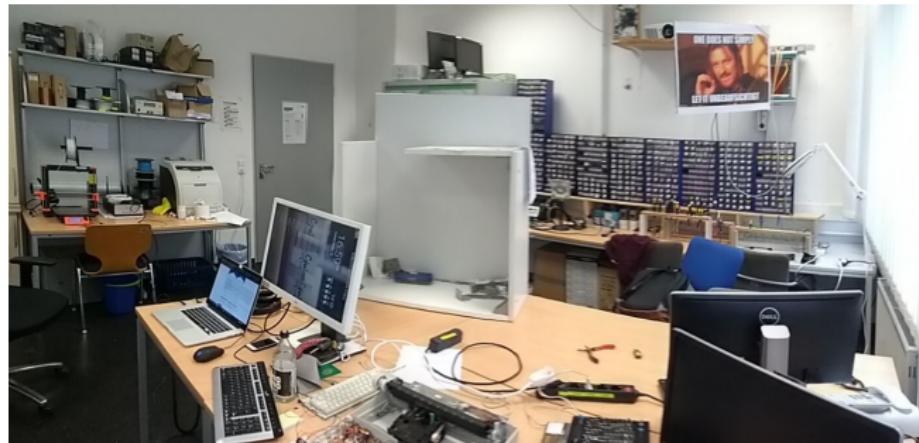
Hennessy, J. L. und Patterson, D. A.: *Rechnerorganisation und Rechnerentwurf*, Oldenbourg Verlag München, 2011

- Tanenbaum, Andrew S.: *Rechnerarchitektur: von der digitalen Logik zum Parallelrechner*, 6., aktualisierte Aufl., Pearson, 2014

Makerspace

Makerspace am Fachbereich AI!

- Für alle die DIY-Projekte umsetzen wollen
- Bietet Zugang zu Werkzeugen (z.B. 3D-Drucker, Laser-Cutter), Entwicklungsboards (Arduino, Raspberry Pi, etc.) und Messgeräten (Oszilloskop, Multimeter, etc.)



Zentrale Frage



Wie funktioniert ein Computer?

Warum?



Warum interessiert mich das?

Als Informatiker benötigen Sie ein Grundverständnis über die Funktionsweise von Rechnern um ...

- Selbst Rechner zu konstruieren (ok, eher unwahrscheinlich)
 - Effiziente Programme schreiben zu können
 - Hardware-nahe Programmieren (Embedded Systems)
- ⇒ **Um zu verstehen wie diese »Black Box« tickt**

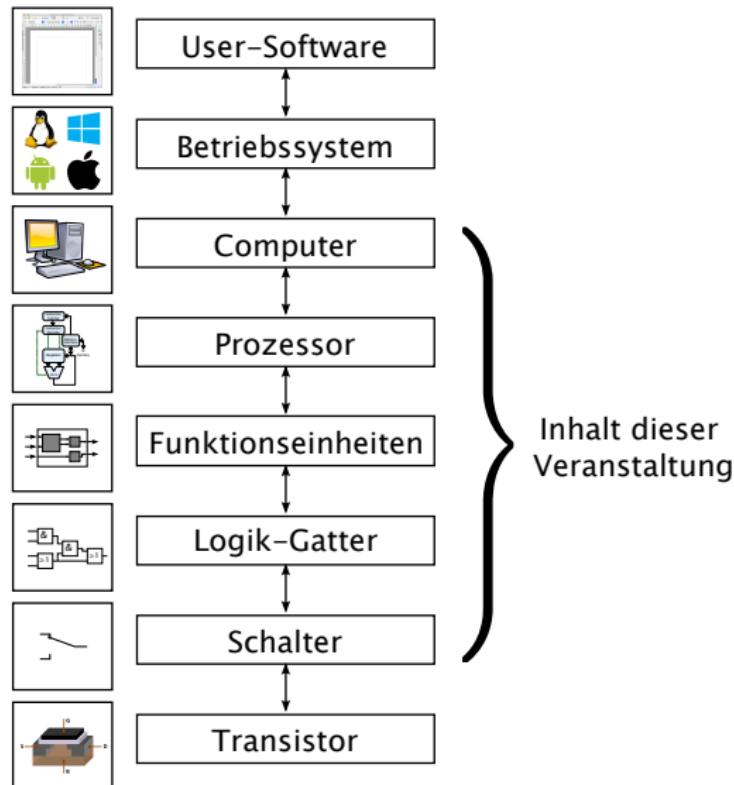
Wie?



Ok, aber ist das nicht sau-komplex?

Klar, aber zu bewältigen dank der **Macht der Abstraktion!**

Die Macht der Abstraktion



Ansatz der Veranstaltung: Bottom-Up!

Information



Bevor wir uns der *Informationsverarbeitung* widmen, sollten wir klären was Information überhaupt bedeutet...

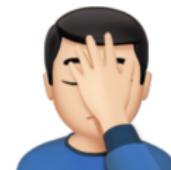
Was ist Information?



Information: Kommunizierte oder empfangene Daten, die Ungewissheit über ein bestimmtes Fakt oder einen Sachverhalt auflöst.

Informationsgehalt wird in **Bit** gemessen.

Ein Bit?



Das Bit



- Bit ist die Kurzform für *binary digit* (deutsch: binäre Stelle)
- Lateinisch *bina* bedeutet doppelt oder zwei
- Ein Bit kann zwei Werte annehmen: 0 und 1
- Das Bit ist die kleinste Informationseinheit
- Technisch realisiert als
 - Spannung vorhanden, z.B. 5V (1) oder nicht (0)
 - Schalter geschlossen (1) oder offen (0)
 - Material magnetisiert (1) oder nicht (0)
 - ...

Darstellung von 0 und 1

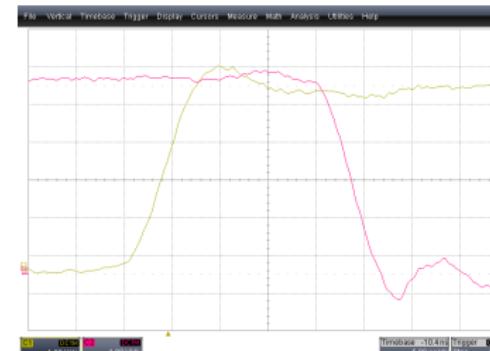


Häufigste Darstellung in digitalen Schaltungen (Computer):
0 und 1 werden mit Spannungspegeln signalisiert

Beispiel, 5V CMOS Logik:
0: 0 V ... 1,5 V
1: 3,5 V ... 5 V

Bei modernen CPUs sind Pegel deutlich kleiner, z.B. 1,2 V
(Core i7)

Beispielmessung CMOS-Inverter (gelb: Eingabe, pink: Ausgabe):



Und was bringt uns das?



Ok, mit einem Bit kann ich zwei verschiedene *Dinge* darstellen. Aber wie kann ich damit komplexere Informationen darstellen, z.B. Text?

Mit mehr Bits!

Die verschiedenen *Dinge* werden hierbei als **Symbole** bezeichnet.

Die mehreren Bits werden **Codewort** bezeichnet

Codes



- Ein Code ist eine Abbildungsvorschrift für eindeutige Zuordnung (Codierung) von
 - Symbolen einer Urmenge zu
 - Symbolen einer Bildmenge.
- Die Zuordnung muss nicht (eindeutig) umkehrbar sein

In der Digitaltechnik:

- Bildmenge ist i.d.R. Vektor aus 0 und 1, d.h. $X \in \{0, 1\}^N$
- Vektor X wird als Codewort bezeichnet
- Mit N Bit lassen sich $K = 2^N$ unterschiedliche Symbole darstellen
- Umgekehrt werden für K Symbole $N = \log_2(K)$ bits benötigt

Binäre Codewörter



Codewortlänge	Mögliche Codewörter	Anzahl Codewörter
1 Bit	0, 1	2
2 Bit	00, 01, 10, 11	4
3 Bit	000, 001, 010, 011, 100, 101, 110, 111	8
4 Bit	0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111	16
:		
N Bit	$\underbrace{00\dots 00}_{=N}, \dots, \underbrace{11\dots 11}_{=N}$	2^N

Informationsdarstellung



Codierung von Information: gibt der Information günstige Eigenschaften bzgl. Merkmalen wie

- Verarbeitbarkeit
- Lesbarkeit (Mensch / Maschine)
- Übertragbarkeit
- Fehlersicherheit
- Speicherbarkeit

Je nach Ziel werden unterschiedliche **Codes** verwendet

Beispiel: Textcodierung



Jedem Zeichen wird ein 7-Bit Codewort zugeordnet (ASCII-Code):

Codewort	Symbol	Codewort	Symbol	Codewort	Symbol	Codewort	Symbol	Codewort	Symbol
0000000	NUL	0100000	~ (space)	1000000	@	1100000	'		
0000001	SOH	0100001	!	1000001	A	1100001	a		
0000010	STX	0100010	"	1000010	B	1100010	b		
0000011	ETX	0100011	#	1000011	C	1100011	c		
0000100	EOT	0100100	\$	1000100	D	1100100	d		
0000101	ENQ	0100101	%	1000101	E	1100101	e		
0000110	ACK	0100110	&	1000110	F	1100110	f		
0000111	BEL	0100111	,	1000111	G	1100111	g		
0001000	BS	0101000	(1001000	H	1101000	h		
0001001	TAB	0101001)	1001001	I	1101001	i		
0001010	LF	0101010	*	1001010	J	1101010	j		
0001011	VT	0101011	+	1001011	K	1101011	k		
0001100	FF	0101100	,	1001100	L	1101100	l		
0001101	CR	0101101	-	1001101	M	1101101	m		
0001110	SO	0101110	.	1001110	N	1101110	n		
0001111	SI	0101111	/	1001111	O	1101111	o		
0010000	DLE	0110000	0	1010000	P	1110000	p		
0010001	DC1	0110001	1	1010001	Q	1110001	q		
0010010	DC2	0110010	2	1010010	R	1110010	r		
0010011	DC3	0110011	3	1010011	S	1110011	s		
0010100	DC4	0110100	4	1010100	T	1110100	t		
0010101	NAK	0110101	5	1010101	U	1110101	u		
0010110	SYN	0110110	6	1010110	V	1110110	v		
0010111	ETB	0110111	7	1010111	W	1110111	w		
0011000	CAN	0111000	8	1011000	X	1111000	x		
0011001	EM	0111001	9	1011001	Y	1111001	y		
0011010	SUB	0111010	:	1011010	Z	1111010	z		
0011011	ESC	0111011	;	1011011	[1111011	{		
0011100	FS	0111100	<<	1011100	\	1111100	---		
0011101	GS	0111101	=	1011101	}	1111101	-}		
0011110	RS	0111110	>>	1011110	-	1111110	-		
0011111	US	0111111	?	1011111	-	1111111	DEL		

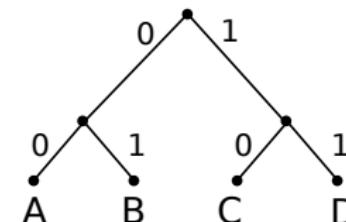
Codierung mit fester Länge



Wenn alle Möglichkeiten **gleich wahrscheinlich** sind (oder es keinen Grund zu einer anderen Annahme gibt), dann wird oft eine Codierung mit **fester Länge** gewählt. Ein solcher Code wird wenigstens genug Bit haben, um den Informationsinhalt zu repräsentieren.

Darstellung eines Code als Binärbaum:

Codewort	Symbol
00	A
01	B
10	C
11	D



⇒ Ein Beispiel ist der 7-Bit ASCII Code

Codierung mit variabler Länge



Wir hätten gerne, dass unsere Codierung die Bits effizient nutzt:

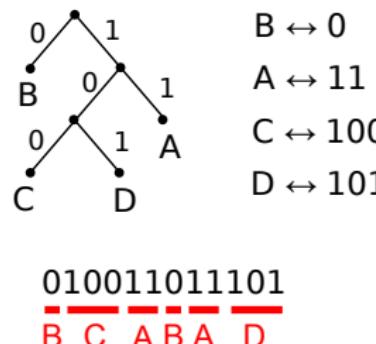
Ziel: Beim Codieren von Daten würden wir gerne die Codelänge an den Informationsgehalt der Daten anpassen.

Im praktischen Gebrauch heißt das:

- Höhere Wahrscheinlichkeit → kürzere Codierung
- Niedrigere Wahrscheinlichkeit → längere Codierung

Beispiel mit variabler Länge

Wahl _i	p _i	Kodierung
"A"	1/3	11
"B"	1/2	0
"C"	1/12	100
"D"	1/12	101



Hohe Wahrscheinlichkeit
weniger Information

Geringe Wahrscheinlichkeit,
mehr Information

Erwartete Länge dieser Codierung für ein Symbol:

$$(2)(1/3) + (1)(1/2) + (3)(1/12)(2) = 1,667 \text{ Bit}$$

Erwartete Länge für 1000 Symbole:

- mit fester Länge, 2 Bit/Symbol = 2000 Bit
- mit variabler Länge = 1667 Bit

Vorlesungsaufgabe



Sie Empfangen die folgende Nachricht:

0110100101011

Wie lautet deren Inhalt bei folgender Codierung?

Symbol	Kodierung
A	11
B	0
C	100
D	101

Huffman-Codierung

🤔 Wie erhält man die günstigste Codierung?

⇒ Die günstigste Codierung bezügl. min. Informationsgehalt ist die Huffman-Codierung

Den Huffman-Code erhält man durch die Konstruktion des Kodierungsbaums von den Blättern bis zur Wurzel:

- ① Jedes Symbol wird als Blatt-Knoten mit seiner Wahrscheinlichkeit dargestellt
- ② Die zwei günstigsten noch nicht verbundenen Knoten werden zusammengefasst zu einem neuen Knoten
- ③ Die Wahrscheinlichkeiten werden addiert und ergeben die Wahrscheinlichkeit des neuen Knoten
- ④ Wiederhole ab Schritt 2 bis Wurzelknoten erreicht ist

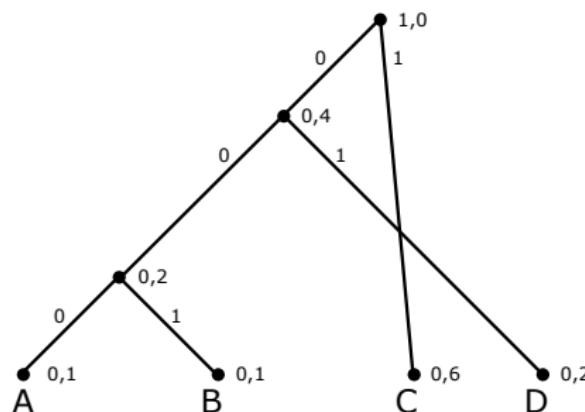
Beispiel: Huffman-Codierung



Wir suchen die minimale Huffman-Codierung für folgende Symbole:

Symbol	Wahrscheinlichkeit	Kodierung
A	0,1	?
B	0,1	?
C	0,6	?
D	0,2	?

Lösung des Beispiels



Symbol	Wahrscheinlichkeit	Kodierung
A	0,1	000
B	0,1	001
C	0,6	1
D	0,2	01

Information quantifizieren



Vorlesungsaufgabe:

Sie erhalten einige Daten zu einer Karte, die zufällig aus einem Spiel mit 52 Karten (Poker) gezogen wurde. Welche der folgenden Daten enthält die meiste Information?

Sortieren Sie beginnend mit der größten Informationsgehalt!

- A Die Karte ist ein Herz
- B Die Karte ist nicht das Pik-Ass
- C Die Karte ist der Herz-König
- D Die Karte ist Bildkarte (Bube, Dame oder König)

Und nun systematisch...



Gegeben ist diskrete Zufallsvariable x

- N mögliche Werte: x_1, x_2, \dots, x_N
- Zugehörige Wahrscheinlichkeiten: p_1, p_2, \dots, p_N

$1/p_i$ ist proportional zur Unsicherheit der Wahl x_i

Erhaltene Information wenn bekannt wird, dass Wahl x_i war:

$$I(x_i) = \log_2 \left(\frac{1}{p_i} \right)$$

Information wird in Bit gemessen, also der notwendigen Anzahl von 0/1-Werten zur Kodierung der Möglichkeiten.

Credits: Claude Shannon, 1948, Begründer der Informationstheorie

Beispiele: Informationsgehalt



- Gezogene Karte aus einem 52'er Spiel ist ein Herz:
 $N = 52, M = 13 \quad I = \log_2(1/(13/52)) = \log_2(52/13) = 2 \text{ bit}$
- Information in einem Münzwurf:
 $N = 2, M = 1 \quad I = \log_2(1/(1/2)) = \log_2(2) = 1 \text{ bit}$
- Werfen von 2 Würfeln:
 $N = 36, M = 1 \quad I = \log_2(1/(1/36)) = \log_2(36/1) = 5,17 \text{ bit}$

Vorlesungsaufgabe: Bestimmen Sie die Information der noch fehlenden Aussagen:

- B Die Karte ist nicht das Pik-Ass
- C Die Karte ist der Herz-König
- D Die Karte ist Bildkarte (Bube, Dame oder König)

Wahrscheinlichkeit & Informationsgehalt

Fall	Daten	p_{data}	Informationsgehalt
A	Herz	13/52	2 Bit
B	Nicht Pik-Ass	51/52	
C	Herz-König	1/52	
D	Bildkarte (Bu, Da, Kö)	12/52	

Korrekte Reihenfolge: CDAB

Shannons Definition für Informationsgehalt passt gut zur Intuition: Wir erhalten mehr Information wenn die Daten mehr Unsicherheit auflösen bezüglich der zufällig ausgesuchten Karte.