Prof. Dr. Horst Völz

Geschichte, Grundlagen und Anwendungen von Kybernetik – Teil 2

Dies ist eine Fortsetzung und Ergänzung zum Teil 1 unter dem speziellen Gesichtspunkt von Hard- und Software.

Das Material beruht teilweise auf dem Buch

Völz, H.: Wissen - Erkennen - Information. Allgemeine Grundlagen für Naturwissenschaft, Technik und Medizin. Shaker Verlag, Aachen 2001

Umfangreiche weitere Literatur am Ende dieser Vorlesungsfolien.

Dieses Material wurde von meiner Homepage: horstvoelz.de heruntergeladen
Es ist für privaten Gebrauch frei nutzbar. Bei Publikationen, Vorträgen usw. ist die Angabe der Quelle notwendig.
Bei kommerzieller Nutzung ist eine Abstimmung mit mir erforderlich.
Die Bilder sind in höherer Qualität ca. 2000×3000 Pixel oder *.cdr Version X6 verfügbar.

Prof. Dr. Horst Völz, Koppenstr. 59, 10243 Berlin, Tel.: 030 288 617 08
Email: h.voelz (at) online.de

Zur Einführung

Die Begriffe Hard- und Software wurden ***ab 1950 für die Rechentechnik*** eingeführt [1].
Englisch bezeichnete Hardware (übersetzt harte Ware) viel früher und zunächst Metall- bzw. Eisenwaren.
Bei den frühen EDV-Anlagen war damit alles gemeint, was ***aus „Material“ hergestellt*** wurde.
Das war die gesamte für den Betrieb ***notwendige technische Ausrüstung***.
Doch bereits diese Anlagen benötigten unbedingt nichtmaterielle ***Programme***.
Sie müssen von Menschen geschrieben und den Anlagen für die gewünschte Berechnung „übergeben“ werden.
Dieser Teil wurde dann im Gegensatz dazu als ***Software*** (weiche Ware) bezeichnet.
Heute wäre etwa die folgende Aufteilung ein brauchbarer Ansatz:

- **Hardware** ist die technische Voraussetzung. Dazu gehören:
  - *elektronische Bauelemente* bis zu den komplexen Schaltkreisen wie CPU und Speicher,
  - *Rechnerarchitekturen* wie NEUMANN-Rechner, RISC, Parallelrechner und Neuronale Netze,
  - *Peripherietechnik* wie Monitor, Tastatur, Maus, Drucker, Plotter, Scanner, Sound- und Videokarten, Festplatten und Streamer.
- **Software** dient der Hardwarenutzung und ihrer Steuerung. Dazu gehören:
  - *Betriebssysteme*, wie DOS, UNIX, Windows und Linux,
  - *Entwicklungsumgebungen*, wie Programmiersprachen und Tools,
  - *Anwendungssoftware* vor allem für Textverarbeitung, Datenbanken, Tabellenkalkulation, DTP, Grafik- und Sound-Programme sowie OCR.

Kritikpunkte

Die obige Trennung wird nicht immer streng eingehalten.
Alle Software muss letztlich irgendwie materialisiert – z. B. in Speichern – verfügbar sein.
So wird zuweilen sogar propagiert: Es gibt überhaupt ***keine Software*** KITTLER [5] S. 225 - 242.
Das ist extremer und einfacher Materialismus. Daraus wäre auch abzuleiten, dass es nichts Geistiges gäbe.

Im Laufe der Zeit sind zusätzlich ***ähnliche Wortbildungen, Begriffe*** entstanden:

- **Firmware:** zur Hardware gehörende, vom Hersteller auf Festwertspeicher abgelegte und vom Benutzer nicht veränderbare Programme.
- **Brainware:** die geistige Vorarbeit bei der Programmerstellung.
- **Liveware:** das Personal, das in der Datenverarbeitung tätig ist.

Auch weit entfernte Anwendungen werden benutzt, z. B. in ***Biologie und Natur*** (s. Ende).

Erstaunlich ist, dass es bis heute so gut wie keine Arbeiten zur präzisen Fassung von Hard- und Software gibt.
Selbst eine ***Literaturrecherche*** geht ***unbefriedigend*** aus.
Die erst grundlegende Arbeit stammt von **KARL NICKEL** [1] S. 373 - 398.
Im Internet wurde gerade eine brauchbar gute Seite bei **Wikipedia** gefunden [7].
Von mir gibt es ältere ***eigene Arbeiten***: [2] S: 175 - 178, 187/8, 343 - 346; [3] 815 - 817 und [4] S. 186 - 190.

Es folgt nun der Versuch einer möglichst präzisen und umfassenden heutigen Sicht.

Information als eine Grundlage

Hard- und Software beruhen weitgehend auf Information.
Daher sei mit der klassischen Aussage NORBERT WIENERs (1894 – 1964) aus [8] begonnen werden:

„Information ist Information weder Stoff noch Energie“.

Stoff steht hier gemäß meiner Übersetzung vgl. [6].

Damit ist Information das dritte Beschreibungsmodell der Welt, neben Stoff und Energie
Daraus ergibt sich das folgende Bild.

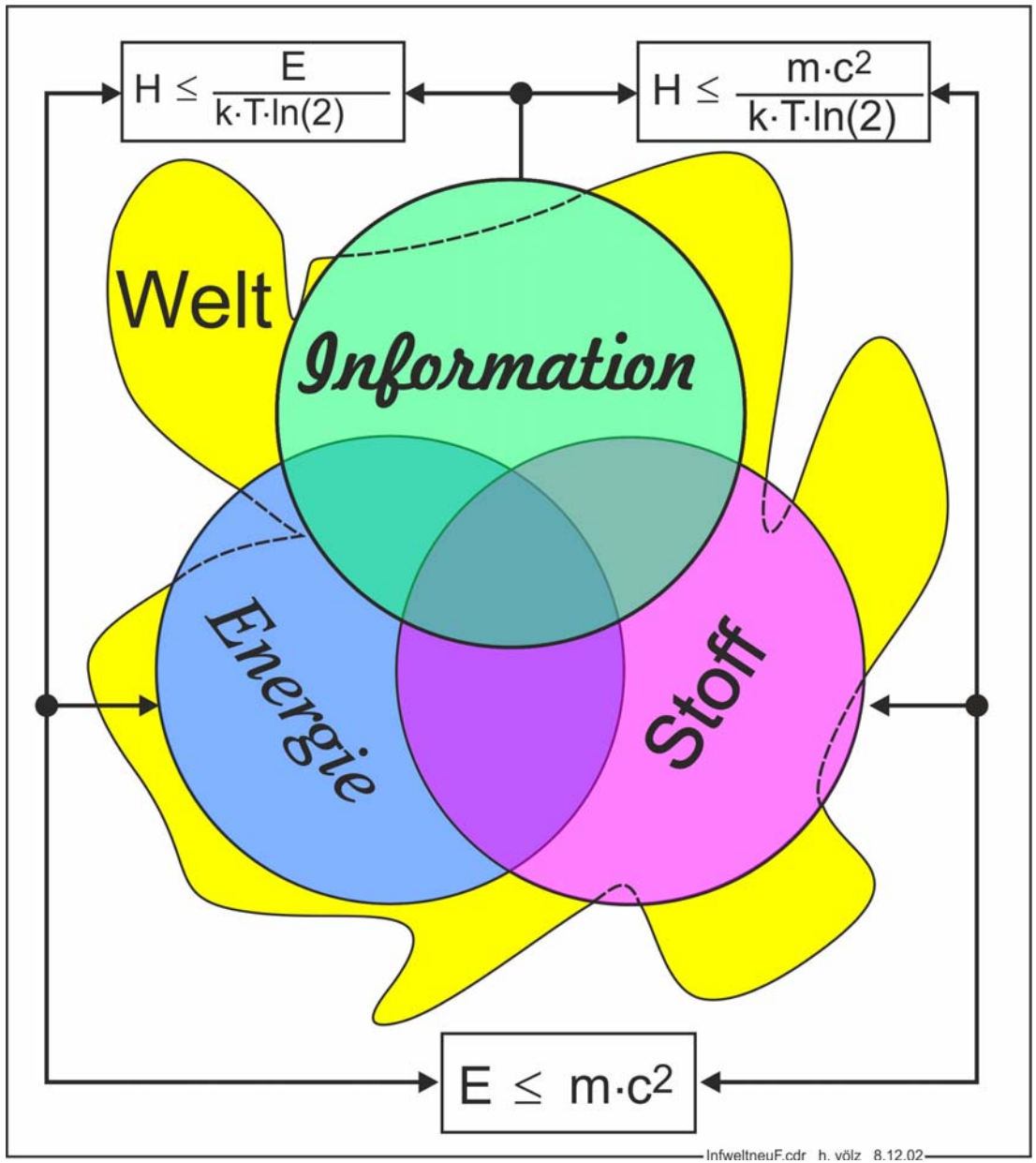
Vereinfacht gehört ***Hardware*** hauptsächlich zum ***Stoff*** und teilweise zur **Energie**.

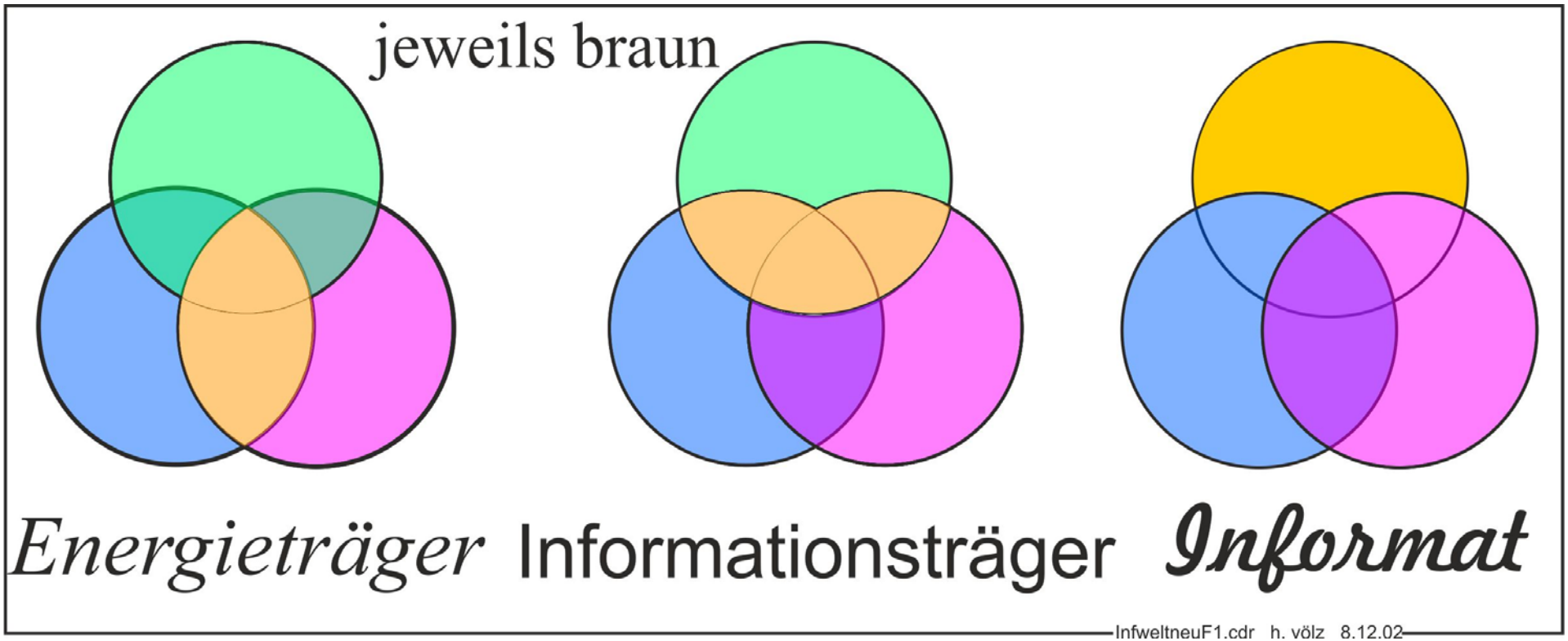
Dagegen gehört ***Software zur Information***.

Die Zusammenhänge sind im Detail jedoch komplizierter.

Information besteht nämlich aus dem stofflich-energetischen ***Informationsträger*** und dem ***Informat*** (nächstes Bild).

Hinweis: Genauere Details zur Information folgen im Teil 3.





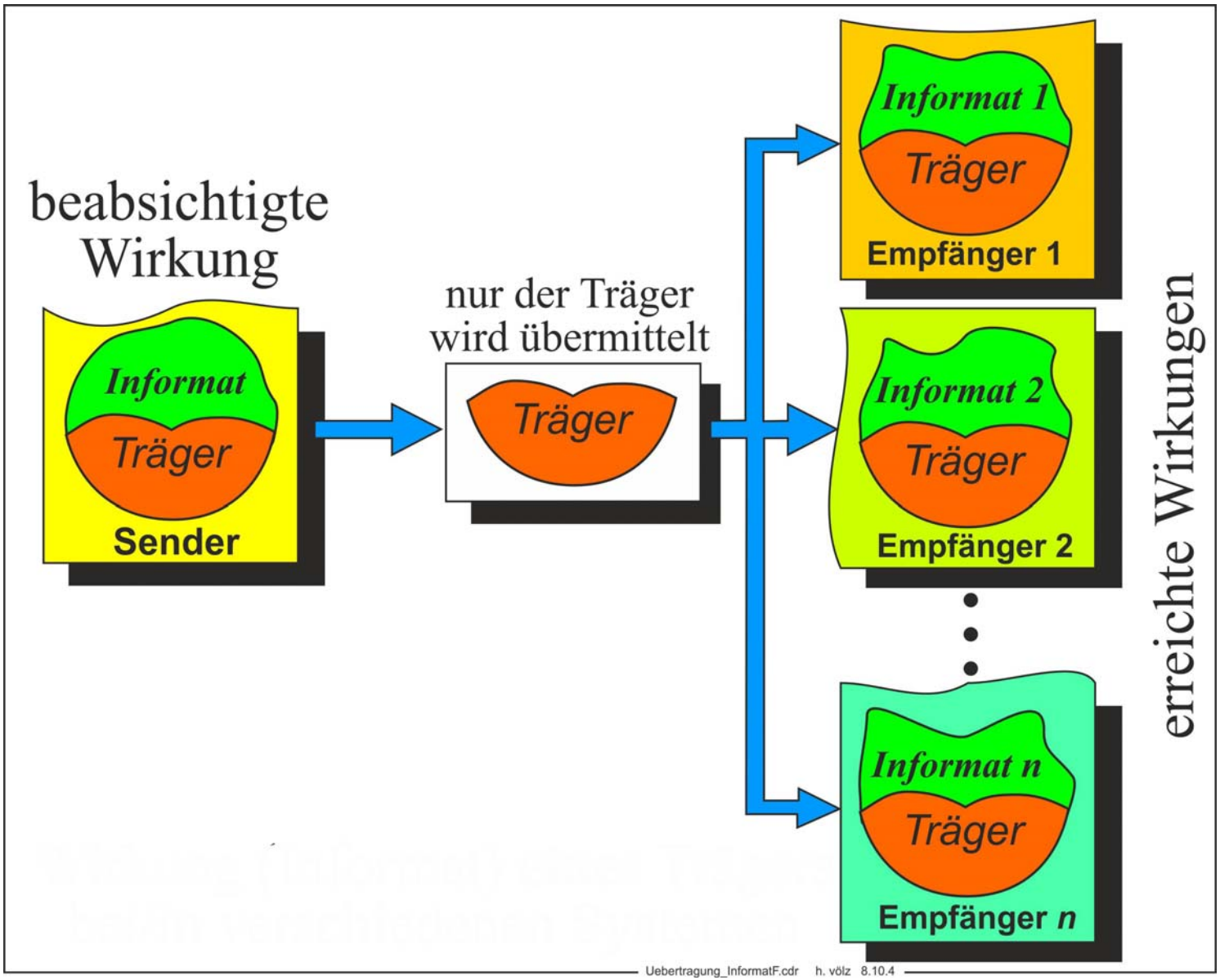
Es gibt drei Teilgebiete, jeweils braun gezeichnet: Energieträger, Informationsträger und Informat.

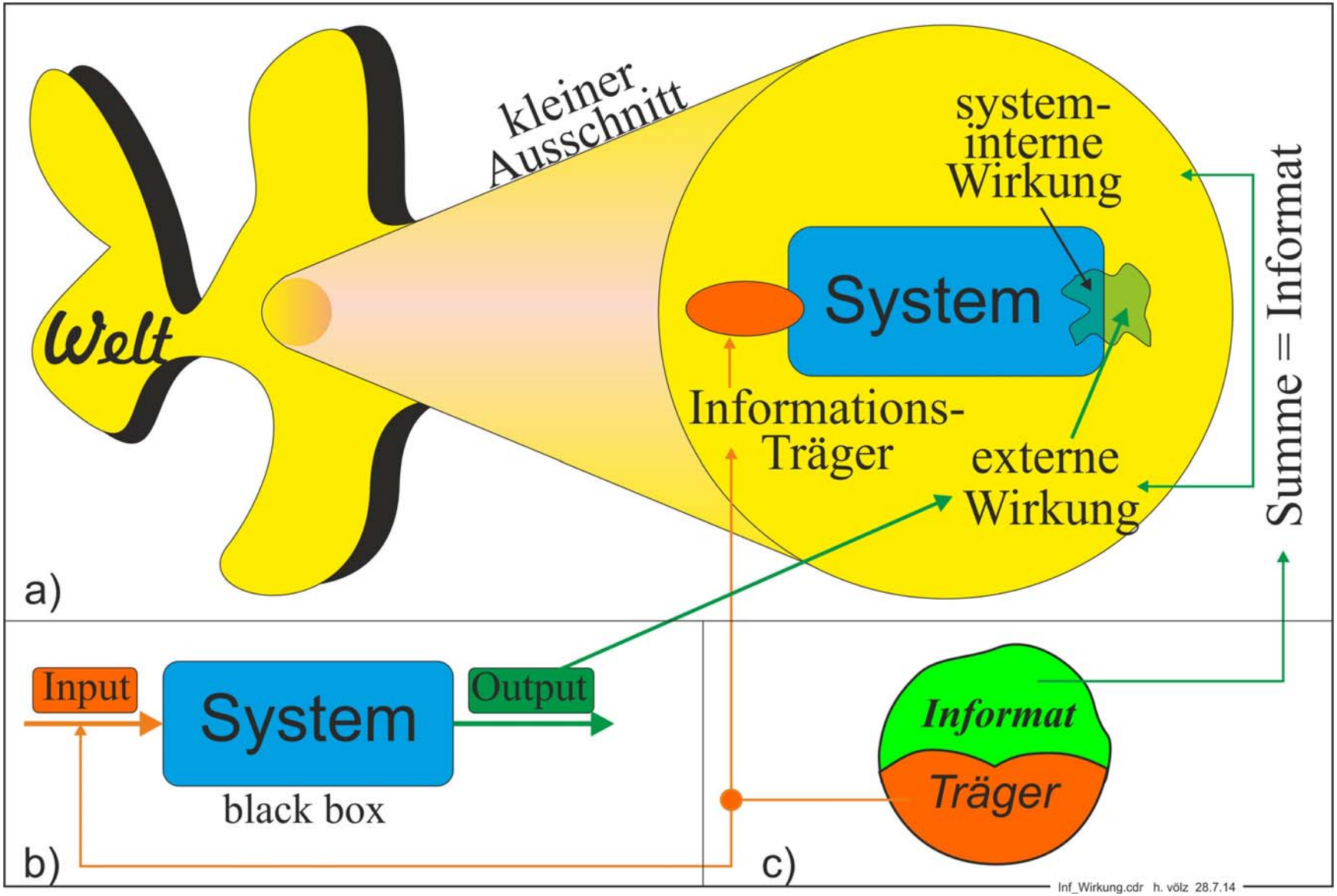
Das Informat ist dabei die Folge des Einwirkens vom Informationsträger auf ein System.
Deren Folge sind Veränderungen im System und in der Umgebung des Systems.

Bei dieser Betrachtung ist zunächst das System, auf das der Informationsträger einwirkt, unbeachtet geblieben.
Doch von ihm hängt ganz wesentlich ab, was er bewirkt, sowohl im System als auch in der Umgebung.
Beides zusammen ist das Informat. Diese Auswirkungen demonstriert das nächst Bild.

Bei gleichen Informationsträger entsteht je nach dem Empfangs-System (gemäß rechts) ein eigenes typisches Informat.

Wird auch die Umwelt einzogen, so ergibt sich das nächste Bild. Es vergleicht die Zusammenhänge mit dem kybernetischen System aus black box, Input - und Output einerseits und dem Anteil von Informationsträger und Informat andererseits.




Welt
kleiner Ausschnitt
system-interne Wirkung
System
Informations-Träger
externe Wirkung
Summe = Informat
a)
Input
System
Output
black box
b)
Informat
Träger
c)
Inf_Wirkung.cdr h. völz 28.7.14

Bezug zur Rechentechnik

Die Rechentechnik kann als Spezialisierung (Einschränkung) betrachtet werden.
Dabei entspricht näherungsweise der Input bzw. Informationsträger den ***Ausgangsdaten*** für eine Berechnung.
Hinzu kommt zumindest noch teilweise der dazu benötigten ***Software***.
Der Rechner entspricht dem ***System***.

Er kann bei der Rechnung ***interne Änderungen*** (Speicherzustände usw.) „erleiden“.
Sie interessieren nur mittelbar, müssen aber bei der Programmierung berücksichtigt werden.
Für das gewünschte Ergebnis der Berechnung sind ausschließlich die externen Wirkungen wichtig.
Das eigentliche Informat der Information ist also deutlich umfassender.

Nach dieser Betrachtung wäre folglich nur das System die Hardware
Der Input (= Informationsträger) ist die Software.
Dabei ist dann nicht erfasst, dass im System auch Software enthalten sein kann.
Diese kann sogar zuvor dem System zu seiner Arbeitsfähigkeit übergeben worden sein.
Folglich können Hard- und Software in einem mehrstufigen Prozess recht unterschiedlich sein.
Dieser Fakt wird im Folgenden weiter untersucht.

Die kybernetische Sicht

Grundlegend für die Kybernetik ist die Regelungs- und Steuerungstechnik.
Dann sind neben dem Rechner viele ***andere Geräte*** gesteuerte, besser steuerbare Technik.
Der Input (= Informationsträger) steuert das System und erzeugt dabei den Output als externe Wirkung.
Absichtlich stark vereinfacht unterschied ich bereits in [2] ab S. 343 dazu ***vier Geräteklassen*** (s. Tabelle).
So wird das ***Soft- und Hardware-Verhältnis allgemeiner*** und weit über die Informatik hinaus wichtig.
Ähnlich sind auch teilweise Soft- (Steuer-Information) und Hardware (Gerätetechnik) ***austauschbar***.
Daher ist auch hier eine möglichst gute Definition und gegenseitige Abgrenzung beider Begriffe gewünscht.

Gerätezweck bzw. -art	**In- und Output der Geräte**	**Beispiele für typische „Systeme“**
Fertigung von Produktion.	Stoff (Material) wird zur Produktion von Geräten usw. verarbeitet.	Werkzeugmaschinen, Fabriken.
Erzeugung, Wandlung und Speicherung von Energie.	Energie (Energieträger), u. a. Elektrizität, Wärme, Gas, Wasser.	Kraftwerke,. Generatoren, Motoren, Speicher-Seen, Transport-Trassen.
Informationsverarbeitung.	Daten, Algorithmen, Programme (Software)	Rechner, Messgeräte, Wandler, Sensoren.
Bildung, Wissen und Unterhaltung.	Physisches und psychisches Interesse des Menschen an Kultur, Kunst und Spiel.	Bücher, Zeitschriften, Fotografie, Film, Rundfunk, Fernsehen, Telefon und „Spiele“.

Elektronik als Ausgangspunkt

Ursprünglich entstanden Rechner auf Basis von ***Relais*** und teilweise ***Pneumatik***.

Doch schnell wurde die Elektronik zum Fundament der Rechentechnik.

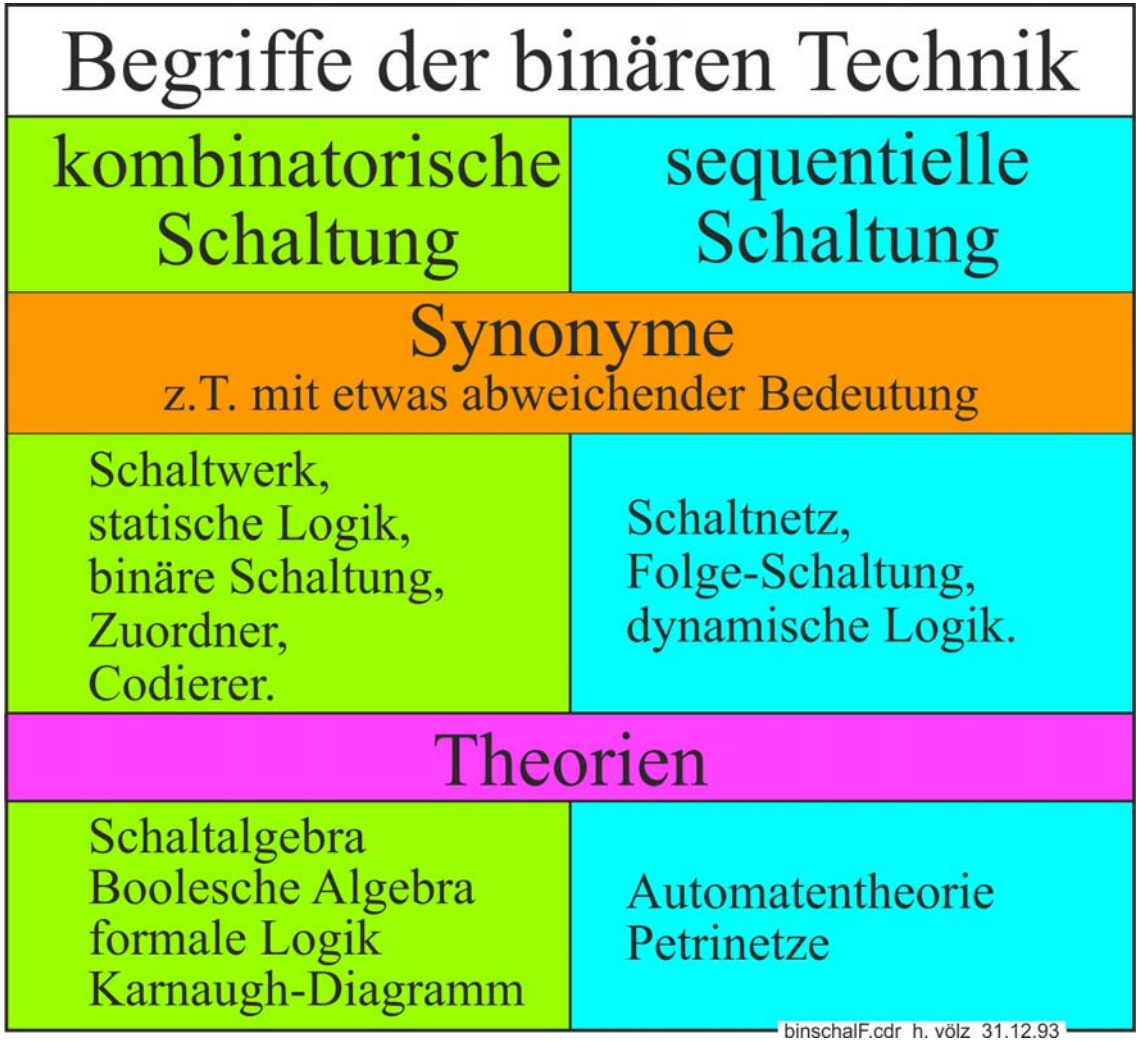
Jedoch auch die ***Röhren-*** und ***Analog-Rechner*** fanden nur relativ kurze Zeit eine gewisse Verbreitung.

Die folgenden Betrachtungen beschränken sich zusätzlich auf die digitale Elektronik.

Fast ausschließlich werden dabei als Grundlage die ***kombinatorische*** und ***sequentielle Logik*** (Schaltungen) eingeführt.

Je nach Lehrmeinung und Institution werden aber ***unterschiedliche Begriffe*** benutzt (s. Bild).

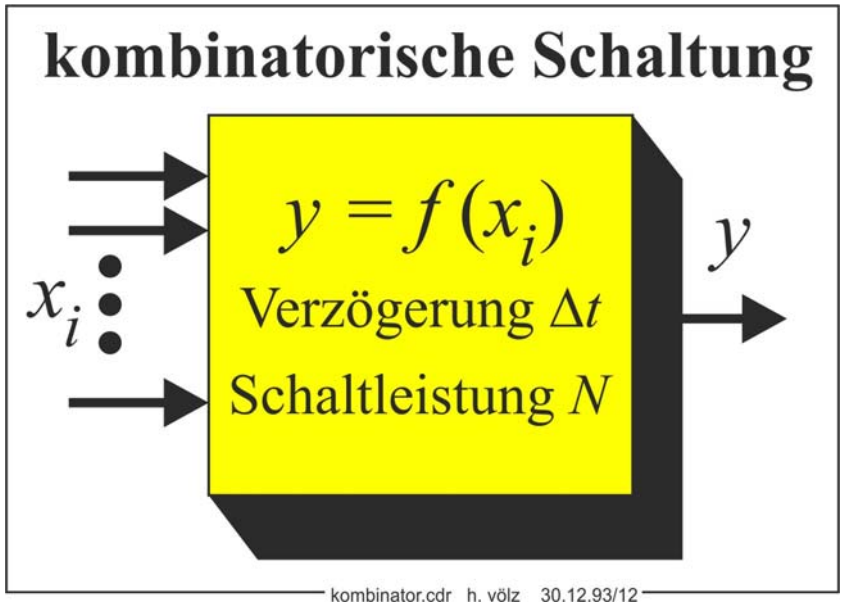
In jeden Fall kommt dabei die ***Speichertechnik*** zu kurz.



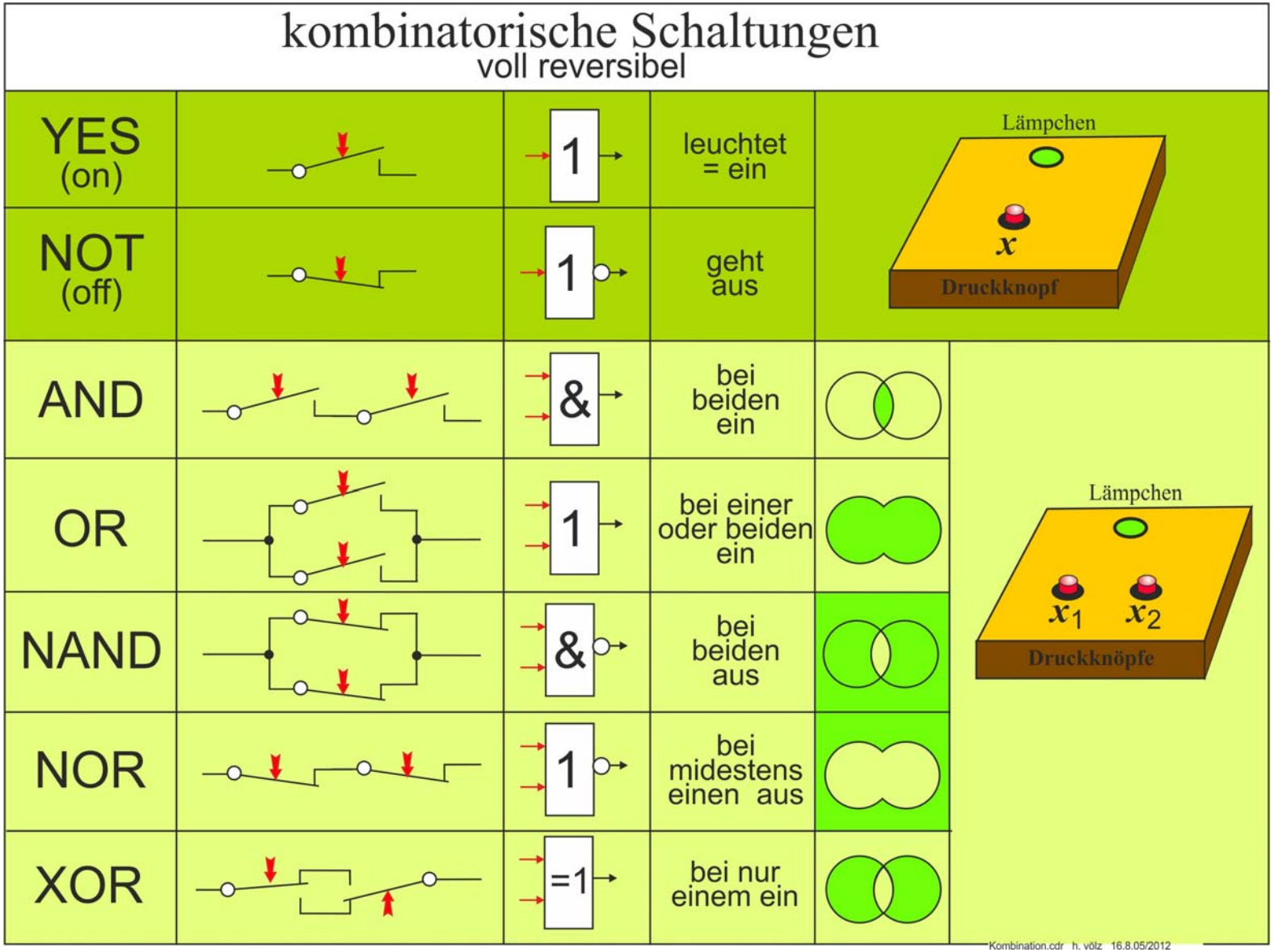
Die kombinatorische Schaltung

Eine typische kombinatorische Schaltung ist gemäß dem Bild durch folgende Eigenschaften gekennzeichnet.

- Sie besitzt ***mehrere Eingänge*** x_1, x_2 bis x_i.
- Aus ihren digitalen 0/1-Werten wird über eine ***digitale Funktion f*** ein digitaler Wert $y = f(x_1, x_2, \ldots x_i)$ abgeleitet.
- Die dabei auftretende ***Verzögerung Δt*** soll möglichst klein sein, theoretisch $\Delta t \rightarrow 0$.
- Für den Betrieb der Schaltung ist eine gewisse ***Grundleistung*** N_0, und während des Schaltens eine zusätzliche ***Schalt-Leistung*** N_s erforderlich.



Vielfach werden nur 2 – selten sogar nur 1 – Eingang benutzt.
Für diese Fälle zeigt das folgende Bild alle möglichen Funktionen für *f*.
Für die Praxis ist es vorteilhaft, dass sich alle zweiwertigen Funktionen aus 3 oder weniger ***„Basis"-Funktionen*** durch Kombination erzeugen lassen.
Das sind hauptsächlich AND, OR und NOT, NAND bzw. NOR.


kombinatorische Schaltungen
voll reversibel
YES
(on)
1
leuchtet
= ein
NOT
(off)
1
geht
aus
Lämpchen
x
Druckknopf
AND
&
bei
beiden
ein
OR
1
bei einer
oder beiden
ein
NAND
&
bei
beiden
aus
NOR
1
bei
midestens
einen aus
XOR
=1
bei nur
einem ein
Lämpchen
x1
x2
Druckknöpfe
Kombination.cdr h. völz 16.8.05/2012

Aufgabe einer kombinatorischen Schaltung

Typischer Weise hat sie Daten gemäß festgelegter logischen Funktion zu erzeugen.
Daher ist sie das ***typische Hardware-Element*** der Rechentechnik.
Sie besitzt nur bestenfalls einen recht mittelbaren Bezug zur Software.
Die zweite Spalte im letzten Bild zeigt entsprechende ***Schalteranordnungen***.
Die dritte Spalte die üblichen ***Schaltbilder***, die vierte eine verbale Beschreibung der Funktionen.
In der letzten Spalte ist eine ***black box*** gewählt.
Ihre Eingänge sind die ***roten Taster***
Ihr Ausgang ist das ***Lämpchen***, das je nach Ausgangs-Zustand leuchtet oder nicht leuchtet.
Bei den Schaltungen mit zwei Eingängen zeit die vorletzte Spalte die mengentheoretischen VENN-Diagramme.
JOHANN VENN (1834 – 1923).
Dabei ist zu beachten, dass bei NAND und NOR die „Umgebung“ (Ruhezustand) leuchtet.

Im Allgemeinen ist – wie oben angedeutet – ein gewisser Austausch zwischen Hard- und Software möglich.
In dieser Hinsicht stellen die kombinatorischen Schaltungen eine Besonderheit dar.
Sie sind zwar „geistig“ nachvollziehbar und bilden dann die Grundlage der mathematischen Logik.
In der Technik sind jedoch keine Software-Lösungen bekannt. So etwas ist auch technisch nicht realisierbar.

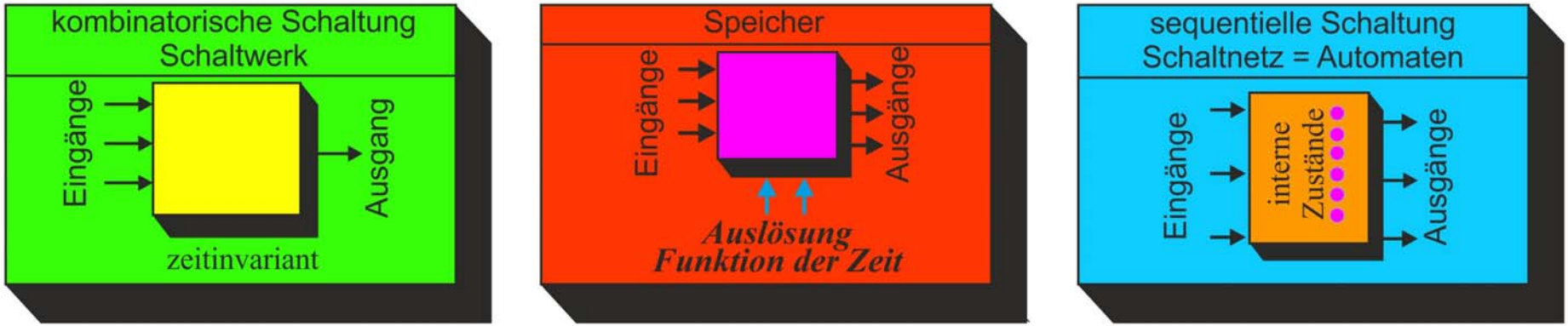
Daher bilden kombinatorische Schaltungen grundsätzlich den Kern jeder digitalen rechentechnischen* Hardware-*Realisierung.

Die drei fundamentalen elektronischen Systeme

Bevor die sequentielle Schaltung behandelt werden kann, ist der Speicher zu beschreiben.
Denn jede sequentielle Schaltung besteht aus ***Speichern und kombinatorischen Schaltungen.***
Offensichtlich ist er aber den Technikern so ***selbstverständlich***, dass er nicht gesondert eingeführt wird.
Er wird „stillschweigend“ ***schlicht vorausgesetzt***.
Deshalb gibt es kaum gesonderte ***Lehrveranstaltungen oder Bücher*** zu ihm, Ausnahme [9] bis [11].
Das ist umso erstaunlicher, da er die ***Weiterentwicklung der mikroelektronischen Technologie*** ständig und zumindest wesentlich bestimmt hat.
Es sei noch betont, dass der Speicher auch für die Abgrenzung von ***Hard- und Software*** äußerst wichtig ist.

Mit ihm ergeben sich daher die ***drei Grundschaltungen*** der Elektronik des folgenden Bildes.

Die drei fundamentalen diskreten Systeme



$x_a = f(x_1, x2, \ldots x_n)$
ohne Zeiteinfluß

$x_a = x_e$
speichert Eingänge
durch Auslösung
zu anderen Zeiten

$x_a = f(Z_i, x_e)$
$Z_i = g(Z_i, x_e)$
komplexe und zeitabhängige
Auswirkung der Eingänge

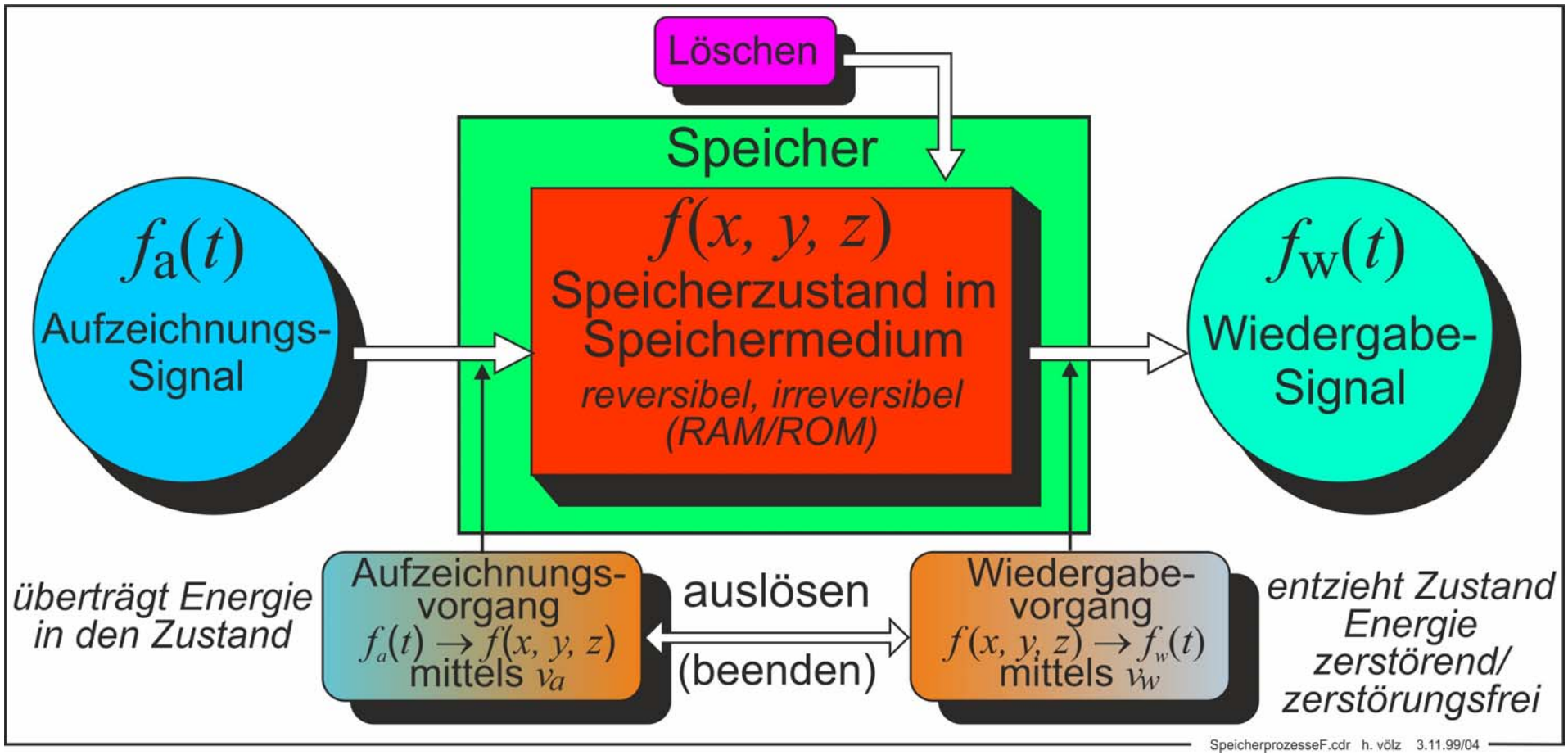
diskrete_SystemeF.cdr h. völz 24.3.00

Der typische technische Speicher

Gegenüber den kombinatorischen Schaltungen unterscheiden sich Speicher durch folgende Eigenschaften:

- Sie besitzen ***gleichviel Ein- und Ausgänge***.
- Zwischen dem Ein- und Ausgang besteht ***kein direkter Übergang***. Vielmehr tritt ein beachtlicher und meist völlig unbestimmter ***Zeitverzug*** Δt auf.
- Die ***Eingangswerte*** werden zu einer extern bestimmten ***Zeit t_a übernommen*** und dann intern langfristig festgehalten (gespeichert).
- Die ***Ausgangwerte*** entsprechen möglichst genau den früheren Eingangswerten, gelangen aber nur zu einer extern bestimmten ***Zeit t_w*** zum Ausgang.

Hieraus ergibt sich das folgende allgemeine Schema
Neue Bezeichnung: ***Eingangssignal*** = ***Aufzeichnungssignal*** $f_a(t)$.
Es wir zu einer extern bestimmten (gewünschten) Zeit t_a mittels ***Aufzeichnungsprozess*** gespeichert.
Und zwar im Inneren des Speichers fortlaufend an ***Orte f (x, y, z)*** abgelegt und dann festgehalten.
Der Aufzeichnungsprozess – also die ***Dauer*** von t_a – kann in einigen Fällen sehr kurz sein.
Dann wird vielleicht nur ein ***einzelner*** (digitaler) ***Wert*** in den Speicher übernommen.
Er kann aber auch einige Zeit Δt_a dauern. Dann wird eine ***Signalfolge*** gespeichert.



Ohne Aufzeichnungssignal geschieht im Speicher und auch an seinem Ausgang nichts.
Erst mit einem Wiedergabesignal t_w gelangt der Speicherinhalt zum Ausgang.
Im Speicher gibt also eine Aufzeichnungs- und Wiedergabegeschwindigkeit v_a bzw. v_w.
Im Normalfall sind beide gleich $v_a = v_w$. Nur dann ist $f_a(t) = f_w(t+\Delta t)$ erfüllt.
Mit dem zeitlichen Abstand Δt gilt dabei zwischen der Auszeichnung und Wiedergabe $t_a = t_w + \Delta t$.

Aussage zum Speicher

Rein technisch gesehen ist der Speicher somit eine ***Hardware***.

Sein ***Zweck*** besteht aber nur aber darin, ***Daten bzw. Software*** zeitweilig ***zwischenzuspeichern***.

Daher ist er eindeutig ein ***Zwitter*** aus Hard- und Software.

Im Gegensatz zur kombinatorischen Schaltung hat er ***keine funktionale Aufgabe zur Verknüpfung*** von Daten.

Dennoch bleibt zunächst weiter unklar, was Software eigentlich ist.

Kombinatorische Schaltungen als Schalter

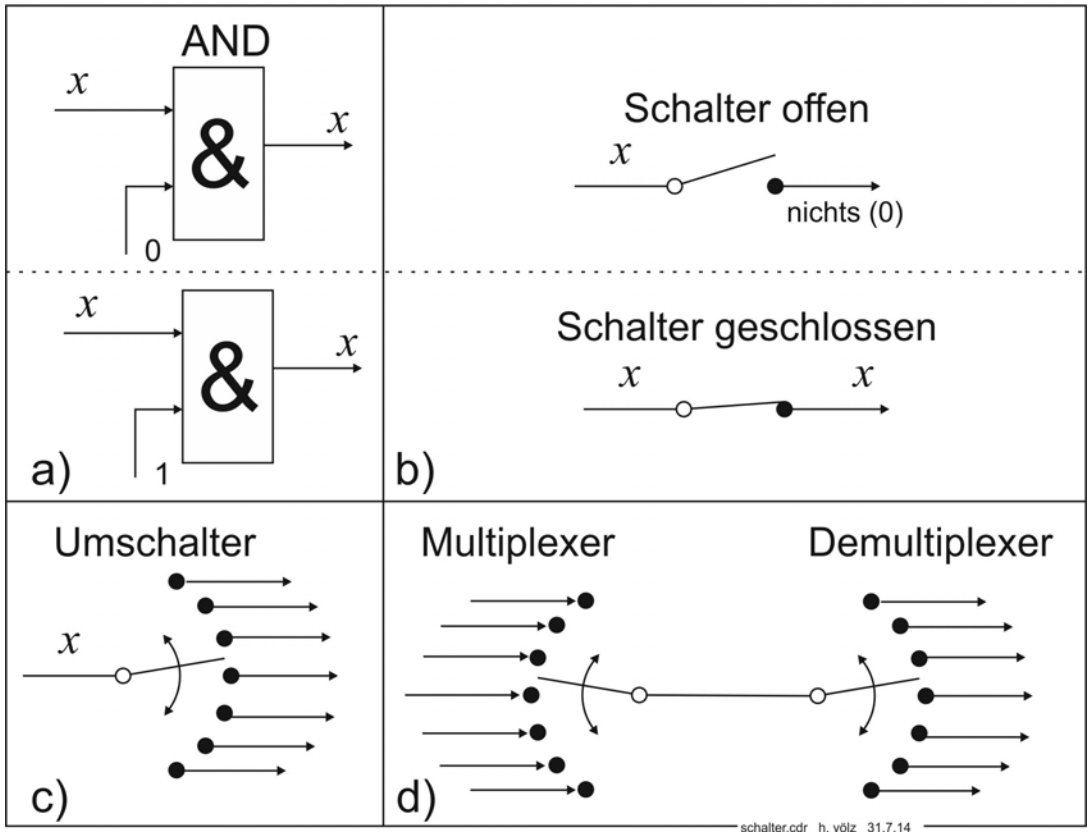
Kombinatorische Schaltungen werden primär zur Verknüpfung von 0/1-Signalen.
Sie können aber auch zu anderen Zwecken herangezogen werden.

Die Teilbilder a) und b) demonstrieren wie eine AND-Schaltung als offener (oben) bzw. geschlossener (unten) betrieben werden kann. Das ***Signal x*** wird dabei entweder ***hindurch geleitet oder nicht.***

Die Steuersignale 0/1 werden bevorzugt aus einem Speicher abgeleitet.

Dieses Verhalten lässt sich ganz ähnlich mit einer XOR-Schaltung verwirklichen. Dann sind für das Steuersignal $0 \leftrightarrow 1$ zu tauschen.

Aus einer Kombination mehrerer derartiger Grundschaltungen ist auch ein Umschalter (c) zu verwirklichen.



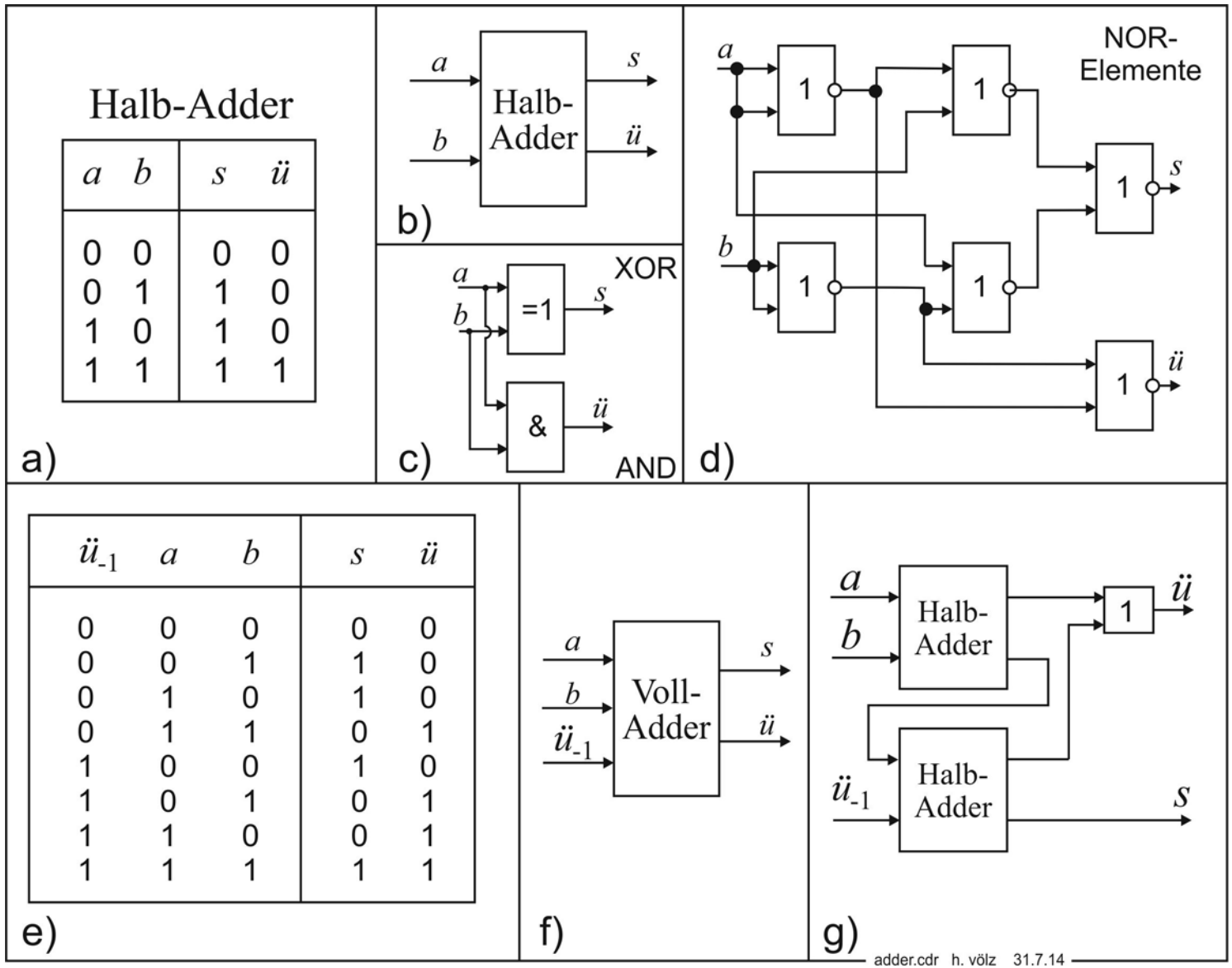
Mit zwei ähnlichen Schaltungen ist sogar eine Multiplexer-Demultiplexer-Schaltung (d) möglich.
Sie ermöglicht Signale mehreren Quellen über nur eine Leitung an verschiedene Empfänger weiterzuleiten.

Adder-Schaltungen

Kombinatorische Schaltungen werden häufig recht komplex miteinander verschaltet.
Ein Musterbeispiel ist die Addition von mehrstelligen Binär-Zahlen.
Besonders einfach ist dabei der Halb-Adder. An seinen beiden Eingängen kann 0 oder 1 liegen.
Am Ausgang können dann die Binär-Werte 00, 01, 10 und 11 entstehen.
Für einzelne Bit sind also zwei Ausgänge erforderlich:
Einer für die Summe *s* (letzte Binär-Ziffer) und ein zweiter für den Übertrag *ü* (erste Binär-Ziffer).
Die dazugehörende Werte-Tabelle zeigt **Bild** a) bzw. die Prinzipschaltung von b).
Eine einfache Schaltung aus einem ***XOR- und AND***-Bauelement zeigt c).

Meist wird jedoch nur ein Typ der kombinatorischen Schaltungen benutzt.
Für 6 NOR-Bausteine folgt die Schaltung nach d).
Prinzipiell ist die komplexe Hardware-Verschaltung ist auch per Soft-Ware möglich.

Um auch einen vorhergehenden Übertrag $ü_{-1}$ einer Stufe zu erfassen, ist ein Voll-Adder notwendig.
Die entsprechende Wertetabelle zeigt e) und die dazugehörende Schaltung f).
Sie kann problemlos aus zwei Halb-Addern aufgebaut werden.


Halb-Adder
a b s ü
0 0 0 0
0 1 1 0
1 0 1 0
1 1 1 1
a)
a
b
Halb-
Adder
s
ü
b)
a
b
=1
&
s
ü
XOR
AND
c)
NOR-
Elemente
1
s
ü
d)
ü-1 a b s ü
0 0 0 0 0
0 0 1 1 0
0 1 0 1 0
0 1 1 0 1
1 0 0 1 0
1 0 1 0 1
1 1 0 0 1
1 1 1 1 1
e)
Voll-
Adder
f)
Halb-
Adder
g)
adder.cdr h. völz 31.7.14

Endliche Automaten

Das Zusammenschalten kombinatorischer Schaltungen mit Speichern benutzt unterschiedliche Verbindungen.
Bei den Addern werden dagegen spezielle Rechenschaltungen erzeugt.

Wichtiger sind jedoch „endliche“ Automaten.
Dabei werden die Eingangssignale auch zum „Umschalten“ der Speicher benutzt.
Entsprechend ihrem jeweils aktuellen Zustand ergeben sich dann unterschiedliche Ausgaben.
Dabei sind MEALY- oder MOORE-Automat möglich (G. H. MEALY bzw. EDWARD F. MOORE; 1925 – 2003).

Beim MEALY-Automaten erfolgt die Ausgabe beim Wechsel eines Speicherzustandes
Beim MOORE-Automaten ist sie dagegen dauerhaft durch die jeweiligen Speicherzustände bestimmt.

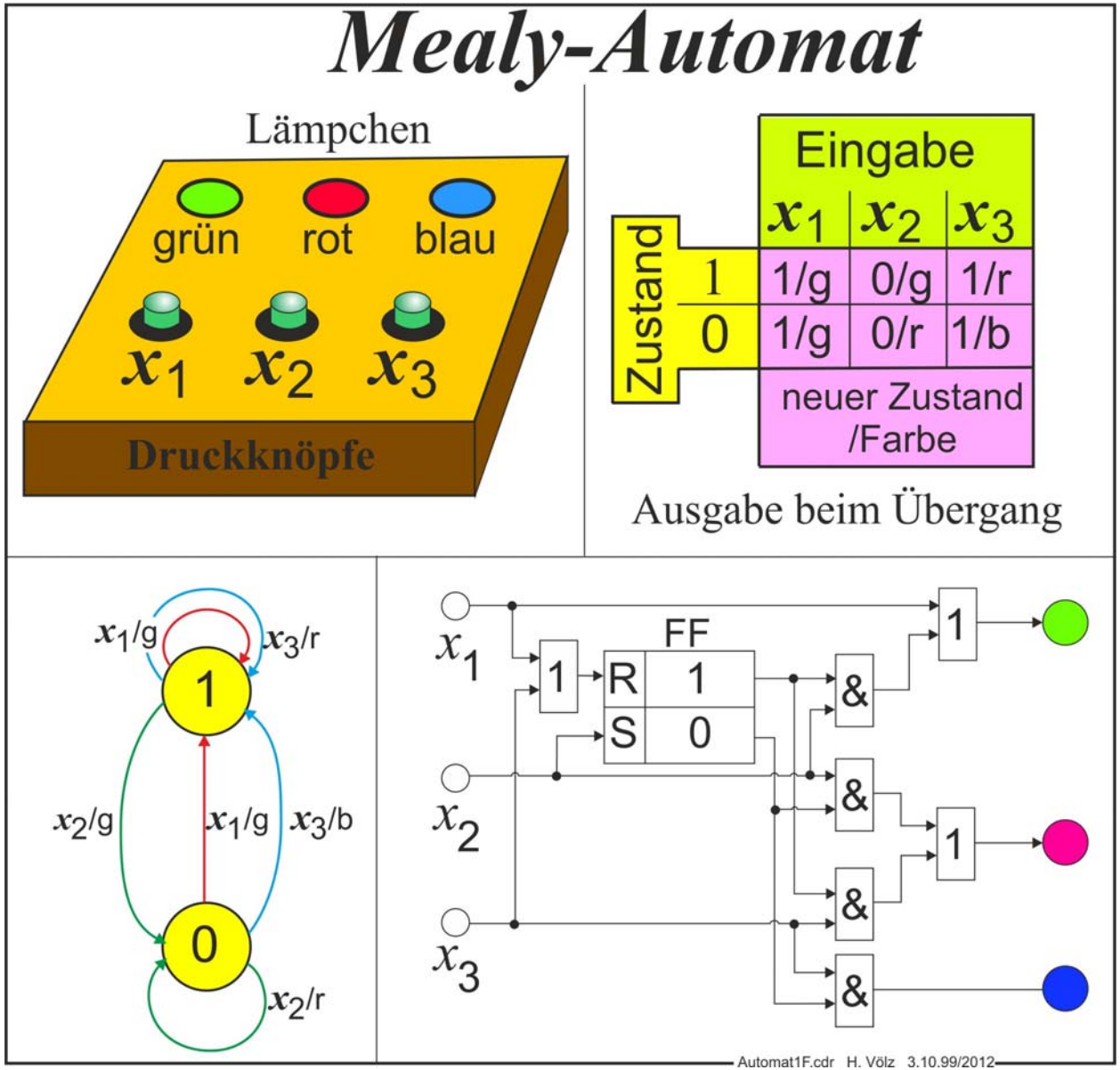
Anschaulich sind Schaltungen durch ein ***Terminal*** mit Tasten (Eingabe) und farbigen Lampen (Ausgaben).
Ihre Beschreibung kann auch durch ***Übergangs-Tabellen*** und ***-Diagramme*** erfolgen.

Die beiden fogenden Beispiele sind absichtlich sehr einfach gewählt.
Praktisch genutzte Automaten sind meist viel komplexer, typisch sind z.B. Fahrkarten- und Bankautomaten.

Erst ihre Weiterentwicklung mit programmierbaren Verknüpfungen und Eigenschaften führt über den TURING-Automaten zum Universal-Rechner (s. u.).

Die Lampen des MEALY-Automat leuchten dann, wenn die Speicherzustände wechseln bzw. eine Taste gedrückt wird.

Der Kasten FF ist ein binärer Flip-Flop-Speicher

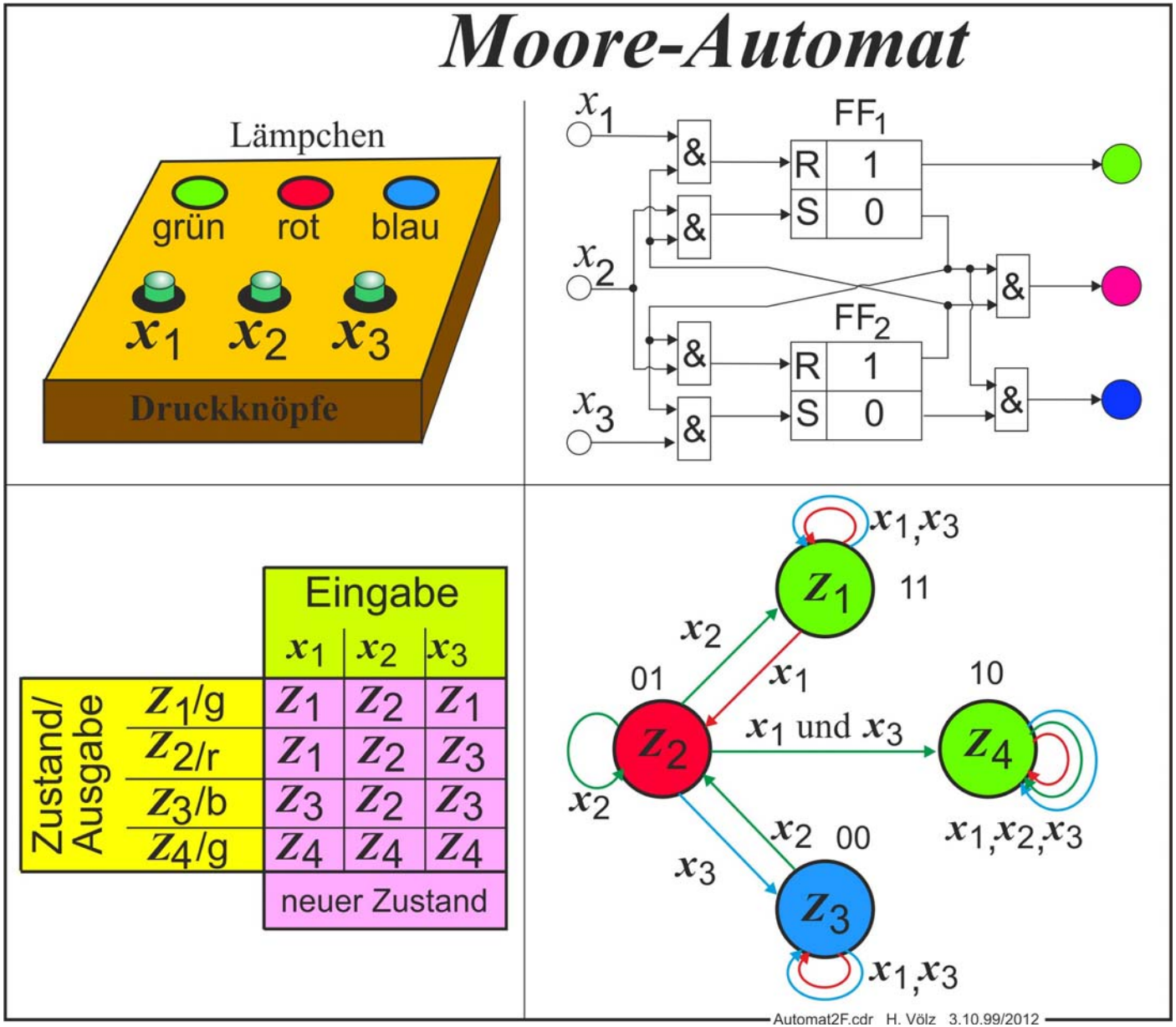


Beim Moore-Automaten leuchtet eine Lampen in Abhängigkeit von den Speicherzuständen.

Sobald eine Taste gedrückt wird, kann der Speicherzustand wechseln.

Dann leuchtet danach eine andere Lampe.

Bei der gewählten Schaltung kann der Speicherzustand Z_4 durch gleichzeitiges Drücken der Tasten x_1 und x_2 erreicht werden. Dann er nicht mehr zu verlassen.

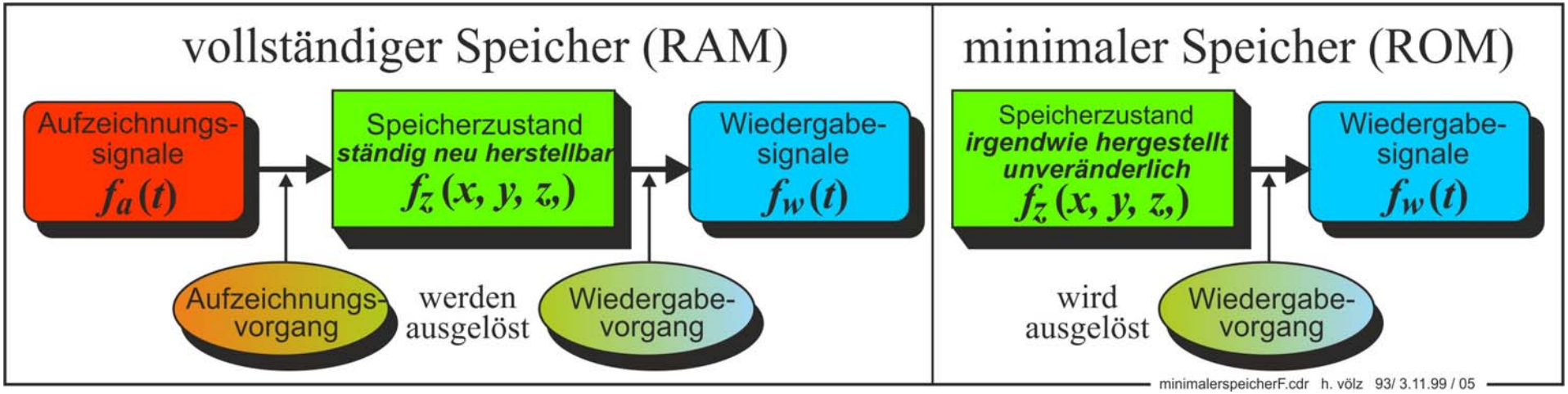


Speichervarianten

Es gibt eine Vielzahl von Speicher-Ausführungen mit unterschiedlichen Eigenschaften und Anwendungen.
Sie ist so groß, dass bisher keine allgemeingültige Systematik gelungen ist.
Die wichtigsten Varianten können auf drei Inhalte bezogen werden:

- Die ***Hardware***-Eigenschaften, wie Elektronik, Festplatte, CD usw.
- Die zu speichernde bzw. gespeicherte ***Information***, vorwiegende Bit-Werte, z. T. aber auch zeitliche Signale und kontinuierliche Werte.
- **RAM**-Varianten mit durch Aufzeichnung stets zu verändernde Wert
- **ROM**-Varianten, die einmal eingebrachte Werte unveränderlich festhalten

Zwischen den ***beiden letzten Extremen*** gibt es viele Zwischeneigenschaften. (s. u.).



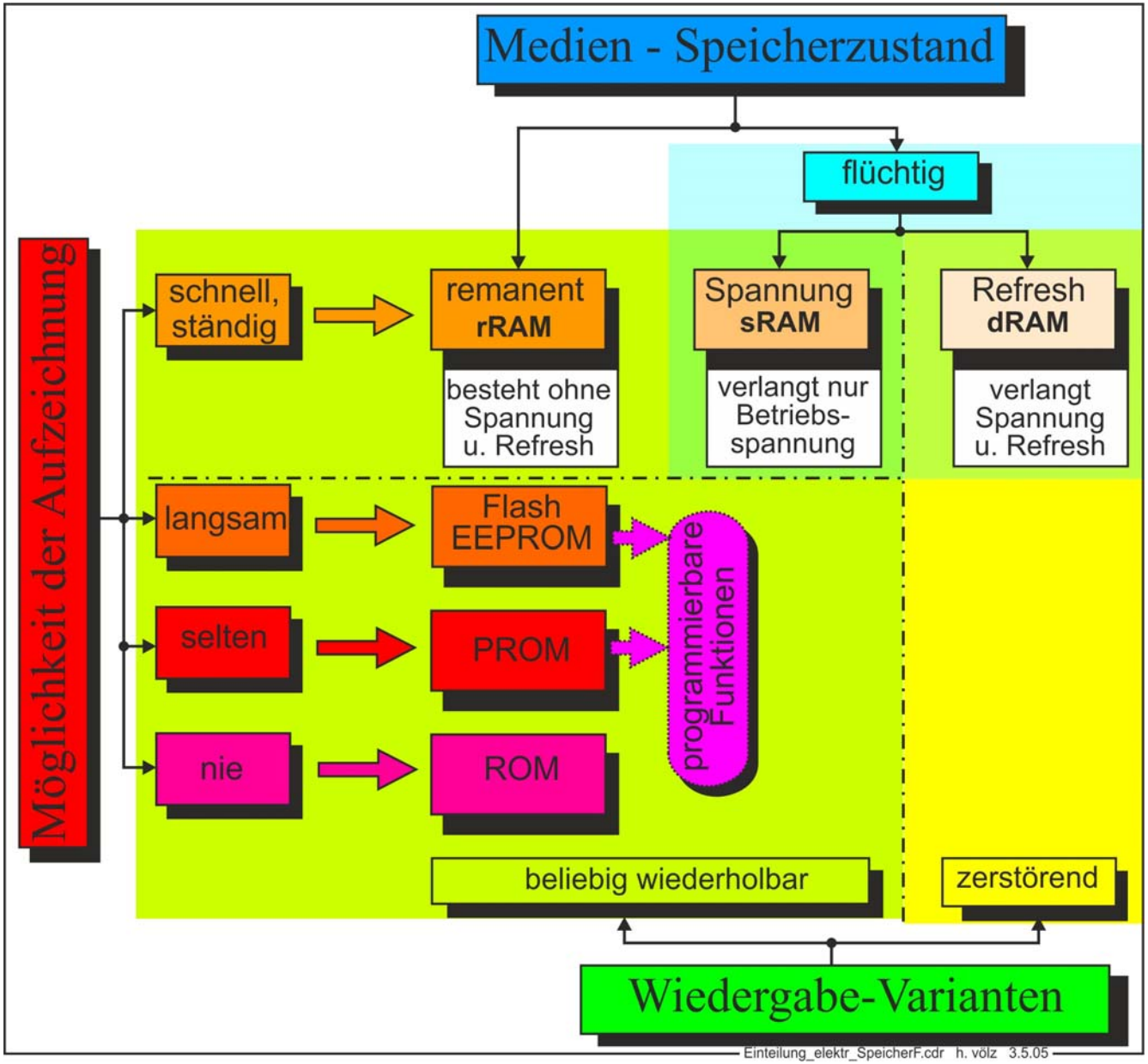
Beim RAM (random access memory) können die Daten zu jeder Zeit eingeschrieben und ausgelesen werden.

Beim ROM (read only memory) sind die Daten vorwiegend bei der Produktion unveränderlich eingeprägt.
Sie können aber dennoch jeder Zeit ausgelesen werden.
Der erste „Abschnitt“ des RAM ist hier also für die Nutzer nicht verfügbar.
Infolge dieser Vereinfachung ist er deutlich preiswerter.

Die drei Haupteigenschaften

1. Die verschiednen Möglichkeiten (im Bild rot und links) bei der ***Geschwindigkeit*** für die ***Aufzeichnung „neuer“ Daten***.
2. Die ***Beständigkeit der Daten*** im Speicher (oben und blau), mit drei Untervarianten:

   ° **rRAM** (remanent): Die Daten bleiben auch bestehen, wenn die Betriebsspannung nicht vorhanden sind.
   hierzu gehören die neueren MRAM, FeRAM, Ovonics usw., aber auch ROM, PROM, Flash usw.
   ° **sRAM** (static): Die Daten bleiben so lange bestehen, wie die Betriebsspannung anliegt.
   Dabei kein Refesh erforderlich)
   ° **dRAM** (dynamic): Damit die Daten bestehen bleiben, ist etwa alle 10 - 100 ms ein Refresh erforderlich.
   Er benötigt Energie. Dies sind heute die häufigsten Speicher.
3. Ob die ***Wiedergabe*** der gespeicherten Daten mit ihrem Verlust (***zerstörend***) einhergeht oder nicht.


Medien - Speicherzustand
flüchtig
Möglichkeit der Aufzeichnung
schnell, ständig
remanent
rRAM
besteht ohne Spannung u. Refresh
Spannung
sRAM
verlangt nur Betriebs-spannung
Refresh
dRAM
verlangt Spannung u. Refresh
langsam
Flash EEPROM
selten
PROM
nie
ROM
programmierbare Funktionen
beliebig wiederholbar
zerstörend
Wiedergabe-Varianten
Einteilung_elektr_SpeicherF.cdr h. völz 3.5.05

Verschiedene Zugriffsarten

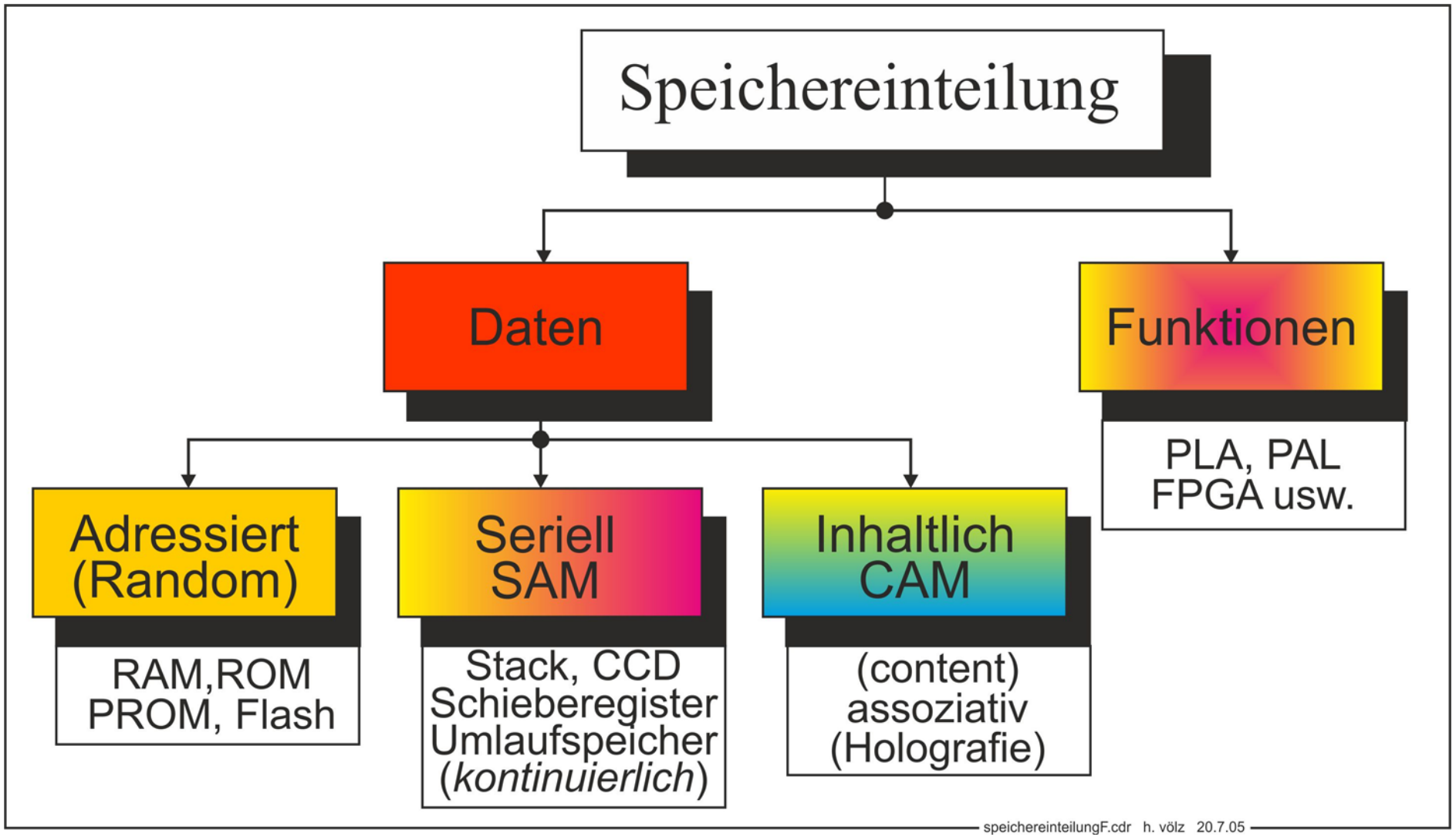
Dabei sind zu unterscheiden

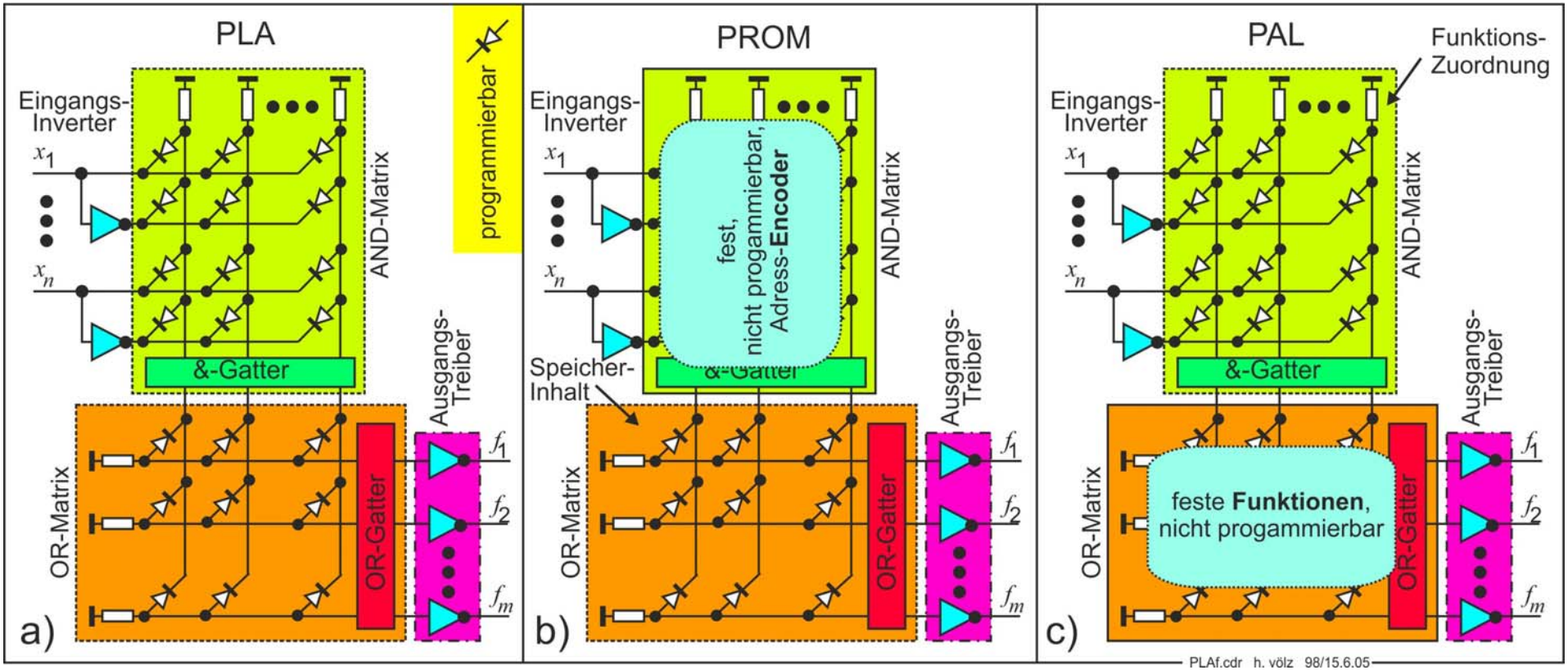
Beim ***Random-Zugriff*** werden die Daten per Adresse gespeichert und wiedergegeben.
Englisch Random (deutsch Zufall) kommt hier unglücklicherweise nur noch aus historischen Gründen vor.

Beim ***seriellen Zugriff*** laufen die Daten ständig um, und es muss daher gewartet werden,
bis der gewünschte Speicherort erreicht ist.

Der CAM (content addressable) auf ***Inhalt bezogne Zugriff*** wird trotz vieler Vorteile selten benutzt
Vorteile gibt es u. A. beim Suchen, es ist voll parallel möglich.

Schließlich gibt es neben der Datenspeicherung auch noch eine ***Funktionen-Speicherung***.
Durch die Speicherung werden hier vor allem verschiedene kombinatorische Schaltungen erzeugt.
Die drei typischen Beispiele schematisiert dann das folgende Bild.


Speichereinteilung
Daten
Funktionen
PLA, PAL
FPGA usw.
Adressiert
(Random)
RAM,ROM
PROM, Flash
Seriell
SAM
Stack, CCD
Schieberegister
Umlaufspeicher
(kontinuierlich)
Inhaltlich
CAM
(content)
assoziativ
(Holografie)
speichereinteilungF.cdr h. völz 20.7.05



Drei Varianten bei denen mehrere kombinatorische Schaltungen gespeichert, verändert und abgerufen werden.
Sie bestehen aus je einer Matrix für AND- und OR-Schaltungen.
Im Sinne von RAM können dabei auch Transistoren statt der ROM Dioden eingefügt (eingeschaltet) werden.
Je nachdem ob beide Matrizen oder nur eine programmierbar werden unterschieden:
PLA (programmable logic array), PROM (programmable read only memory), PAL (programmable array logic)

Zusammenhänge zwischen den drei Grundschaltungen

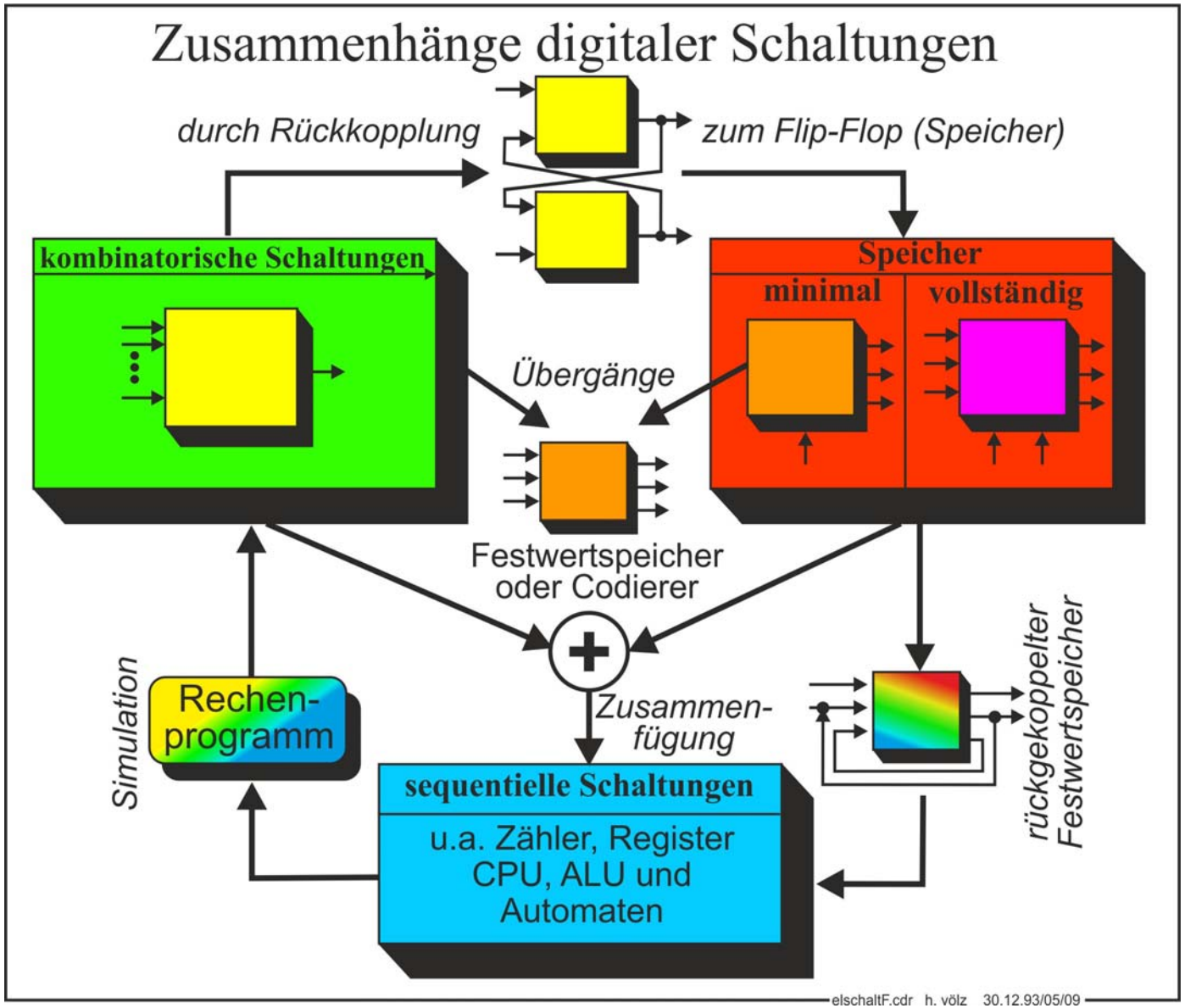
Es gibt alle Übergängen gibt zwischen den drei Grundschaltungen.

Insbesondere sind die ***kombinatorischen Schaltungen*** lediglich mittels Rechner zu simulieren.
Auch dies macht sie zur fundamentalen Hardware.

Nicht ganz so extrem ist es mit den ***Speichern***.
Binäre Flip-Flop lassen sich zwei kombinatorischen Schaltungen erzeugen,
sie sind dann aber deutlich komplexer als Schaltungen, die dafür z. B. die magnetische Hysterese nutzen.
Im Gegensatz zu den kombinatorischen Schaltungen sind sie nicht „Selbstzweck“
sondern dienen „nur“ zur Speicherung von Daten.
Daher stehen sie der Software deutlich näher als die kombinatorischen Schaltungen.
Außerdem gibt es zwischen logischen Schaltungen und Speichern ROM und Codierer.

Die ***sequentiellen Schaltungen*** enthalten immer kombinatorische Schaltungen und Speicher.
Nur wenige Sonderfälle lassen sich durch rückgekoppelte Festwertspeicher erzeugen.

Allgemein folgt hieraus erneut: ***Software*** dient primär zur Steuerung von Hardware
Allerdings kann sie dazu auch in die Hardware, z. B. von Speichern
oder bei der Zusammenschaltung von kombinatorischen Schaltungen realisiert sein.
Deshalb ist noch weiterhin auf Hard- und Software einzugehen.


Zusammenhänge digitaler Schaltungen
durch Rückkopplung
zum Flip-Flop (Speicher)
kombinatorische Schaltungen
Speicher
minimal
vollständig
Übergänge
Festwertspeicher
oder Codierer
Simulation
Rechen-
programm
Zusammen-
fügung
sequentielle Schaltungen
u.a. Zähler, Register
CPU, ALU und
Automaten
rückgekoppelter
Festwertspeicher
elschaltF.cdr h. völz 30.12.93/05/09

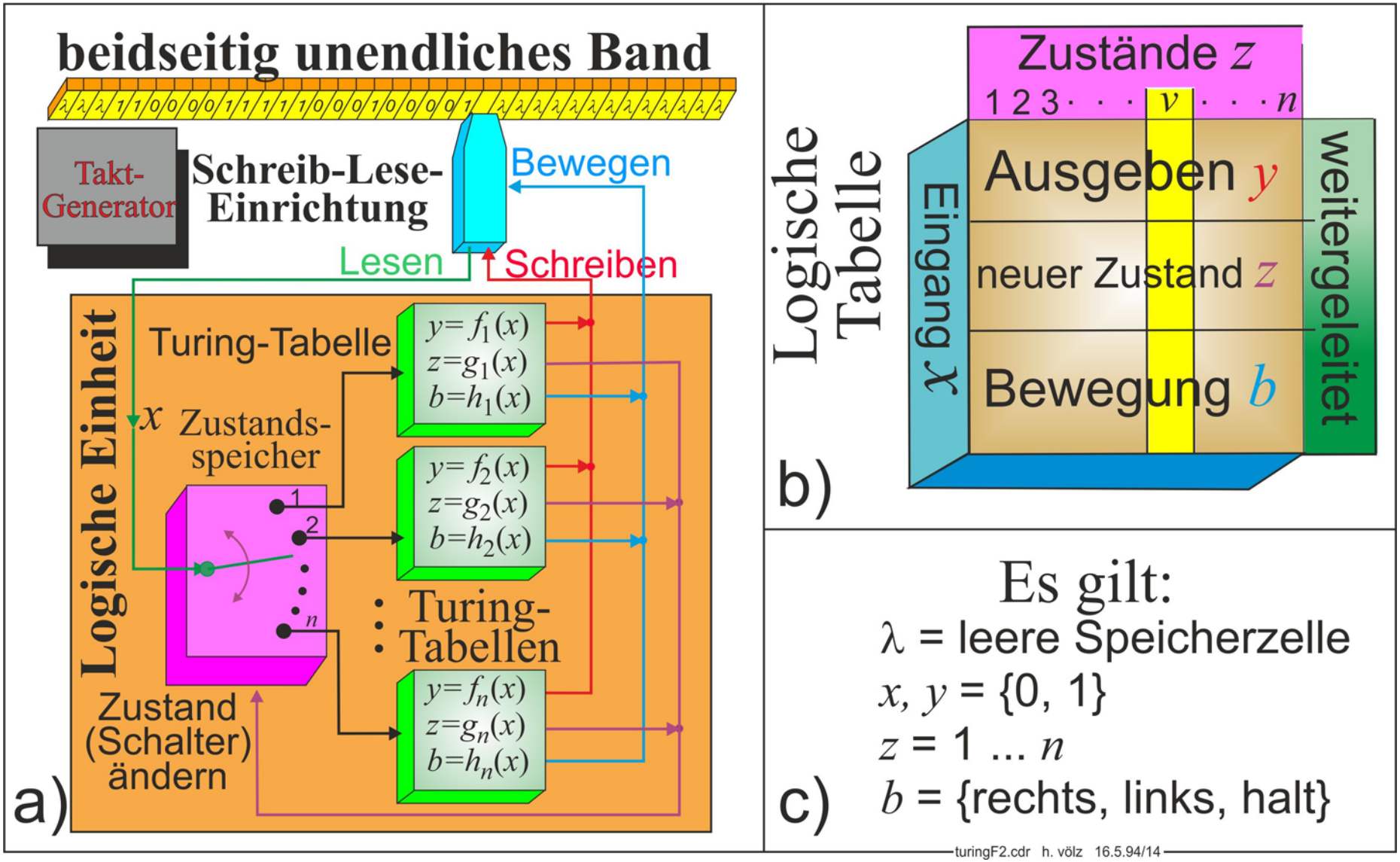
Die Idee des TURING-Automaten

Ein typischer endlicher Automat ist für ***spezielle Anwendungen*** gebaut.
Interessant ist aber, was Automaten prinzipiell leisten können.
Hierfür entwickelte 1936 ALAN MATHISON TURING (1912 – 1954) seinen TURING-Automaten.
Er ist kein technisches Gebilde sondern ***ein gedankliches Modell***.
Es benutzt ebenfalls kombinatorische Schaltungen und Speicher.
Jedoch wurde er ***nie gebaut*** – nur sehr viel später für die Ausbildung.
Der ***erste technisch nutzbare*** (Universal-) Rechner entstand unabhängig hiervon Ende 1930er Jahre.
Er wurde von KONRAD ZUSE (1910 – 1995) für wirkliche Berechnungen gebaut.
Die anderen technischen Computer entstanden ***erst ab 1945***

Aufbau des TURING-Automaten

Er besteht aus vier Grundeinheiten

1. Ein ***unendliches Speicherband*** mit einzelnen Kästchen als Bit-Zellen.
   Damals gab es noch kein Magnetband.
   Deshalb hat TURING an einen Papierstreifen gedacht.
   Sie können nach Bedarf durch Ankleben von Teilstreifen beliebig verlängert werden kann.
   Auf die einzelnen Speicherplätze können dann Daten-Bits 0 oder 1 geschrieben werden.
   Zusätzlich gibt es noch Start- und Ende-Zeichen α und Ω.
   Leere oder gelöschte Plätze werden mit λ gekennzeichnet.
2. Eine ***Lese-Schreib-Einrichtung***, die Daten vom Band lesen oder darauf schreiben bzw. löschen kann.
   Mit jedem Takt kann sie (bei Bedarf) um eine Speicherzelle nach rechts (*r*) oder links (*l*) bewegt werden.
3. Eine ***Logische Einheit*** (kombinatorische Schaltung), übernimmt die gelesenen Daten
   Sie werden über den Schalter (Zustandsspeicher) zur ausgewählten TURING-Tabelle weiterleitet.
   So wie kombinatorische Schalten bewirken sie logische Verknüpfungen.
   Die sich ergebenden Daten dienen zum Schreiben *y* und Bewegen *b*.
   Ein weiteres Ergebnis *z* bestimmt eine neue Schalterstellung.
   Im nächsten Takt gelangen dadurch die Lese-Daten zur so ausgewählten TURING-Tabelle usw.
   Die *n* Tabellen sind meist zu einer dreidimensionalen Tabelle über die n Zustände zusammengefasst.
4. Ein ***Taktgenerator*** schaltet periodisch die gedanklichen Einheiten 1 bis 3 immer um einen Schritt weiter.


beidseitig unendliches Band
λ λ λ 1 1 0 0 0 0 1 1 1 1 1 0 0 0 1 0 0 0 0 0 1 λ λ λ λ λ λ λ λ λ λ λ
Takt-Generator
Schreib-Lese-Einrichtung
Bewegen
Lesen
Schreiben
Logische Einheit
Turing-Tabelle
x
Zustands-speicher
1
2
n
y= f1(x)
z=g1(x)
b=h1(x)
y= f2(x)
z=g2(x)
b=h2(x)
Turing-Tabellen
y= fn(x)
z=gn(x)
b=hn(x)
Zustand (Schalter) ändern
a)
Logische Tabelle
Zustände z
1 2 3 · · · v · · · n
Eingang x
Ausgeben y
neuer Zustand z
Bewegung b
weitergeleitet
b)
Es gilt:
λ = leere Speicherzelle
x, y = {0, 1}
z = 1 ... n
b = {rechts, links, halt}
c)
turingF2.cdr h. völz 16.5.94/14

Die Arbeitsweise

Für jeden TURING-Automaten legen die TURING-Tabellen die interne Arbeitweise (Eigenschaften) fest.
Durch ihren Aufbau und Austausch können so sehr vielfältige Aufgaben gelöst werden.

Für jede spezielle ***Berechnung*** ist ein ***passendes Programm*** (Software) mit den Start-/Ende-Daten nötig.
Es muss programmiert und dann auf das Band geschrieben werden.

Dann werden die Daten schrittweise – ***bei α beginnend*** – vom Band gelesen und ausgeführt.
Daraus berechnete Daten müssen dann auf das Band (evtl. an anderer Stelle) geschrieben werden.
Mit dem Takt wiederholt sich dieser Prozess solange ***bis Ω erreicht*** ist und damit alles berechnet ist.
Dann steht auf dem Band dann das Ergebnis.

Eine detaillierte Beschreibung des Ablaufes für eine einfache Addition enthält [2] ab S. 68, [4] ab S 251.
Die Leistung des recht komplexen Vorganges entspricht der in Hardware realisierten Adder-Schaltung.

Der universelle Turing-Automat

Für jede neue Berechnung muss für den TURING-Automaten ***ein passendes Programm*** geschrieben werden.
Da es ***unterschiedliche aufgebaute*** TURING-Automaten gibt, muss es immer speziell erzeugt werden.
Das ist sehr ungünstig.
Deshalb war der Nachweis, dass es einen ***universellen*** TURING-Automaten gibt, ein beachtlicher Fortschritt.
Für ihn ist für jede Aufgabe nur ein einziges Programm erforderlich.
Mit einem Hilfsprogramm (Simulator) wird dann ***jeder sequentielle Rechner*** universeller TURING-Automat.
Das gilt sogar für Kleinstrechner und bedeutet:

Alle sequentiellen Rechner lösen die gleichen Probleme.

Neben dem Hilfsprogramm benötigen sie nur ausreichende ***Speicherkapazität***.
Außerdem und rechnen sie ***unterschiedlich lange***
Da heutige Rechner generell mit großer Speicherkapazität auszustatten sind, gibt es kaum Einschränkungen.

Die CHURCH-These

Nur wenig nach TURINGs Betrachtungen entstanden ähnliche Methoden für das „maschinelle“ Berechnen:

- Gleichungskalkül nach GÖDEL-HERBRAND,
- μ-rekursive Funktionen nach KLEENE,
- MARKOW-Algorithmen,
- Minimal-Logik nach FITCH,
- Kanonischer Kalkül von POST,
- Graphen-Schemata von KALUZIN,
- Rekursions-Schemata von MCCARTHY und
- λ-Funktionen von CHURCH.

Schließlich wurde bewiesen, dass alle derartigen Methoden (Verfahren) mathematisch völlig gleichwertig sind. Sie behandeln das Problem des Berechnens nur auf unterschiedlicher Weise und Kompliziertheit.

Daher stellte 1940 ALONZO CHURCH (1903 –1995) eine ***nicht beweisbare Hypothese*** auf, die aber dennoch von den meisten Mathematikern als gültig akzeptiert wird:

Alle heute vorhandenen und auch künftig gewonnenen Methoden für die Berechenbarkeit sind gleich leistungsfähig.

Insgesamt ersetzen sie die ***intuitive Berechenbarkeit*** durch den exakten Begriff des ***Algorithmus***. Daraus folgt:

Alles Berechenbare ist rekursiv.

Möglichkeiten und Grenzen

Insbesondere gilt: alle elementaren mathematischen Funktionen sind rekursiv berechenbar.
Es lässt sich aber zeigen, dass es auch ***Nicht-Berechenbares*** gibt.
Die erste nichtberechenbare Funktion stellte 1962 TIBOR RADO (1895 – 1965) vor [4] S. 261.
Von ihm stammt auch das Problem „Fleißiger Biber“.
Ein Programm soll mit dem TURING-Automaten möglichst viele aufeinander folgende 1 aufs Band schreiben.
Für 1111 wurde die Lösung erst 1972 gefunden.
Für 11111 oder länger gibt es bisher nur Abschätzungen für den notwendigen Aufwand.

Ergänzt sei, dass mit dem zweiten CANTOR-Diagonal-Verfahren folgendes beweisen lässt.
Es gibt sogar ***überabzählbar viele nichtberechenbare Funktionen*** gibt [4] S. 261.

Darüber hinaus gibt noch eine ***Vielzahl „negativer“ Aussagen*** der theoretischen Informatik, z. B.:

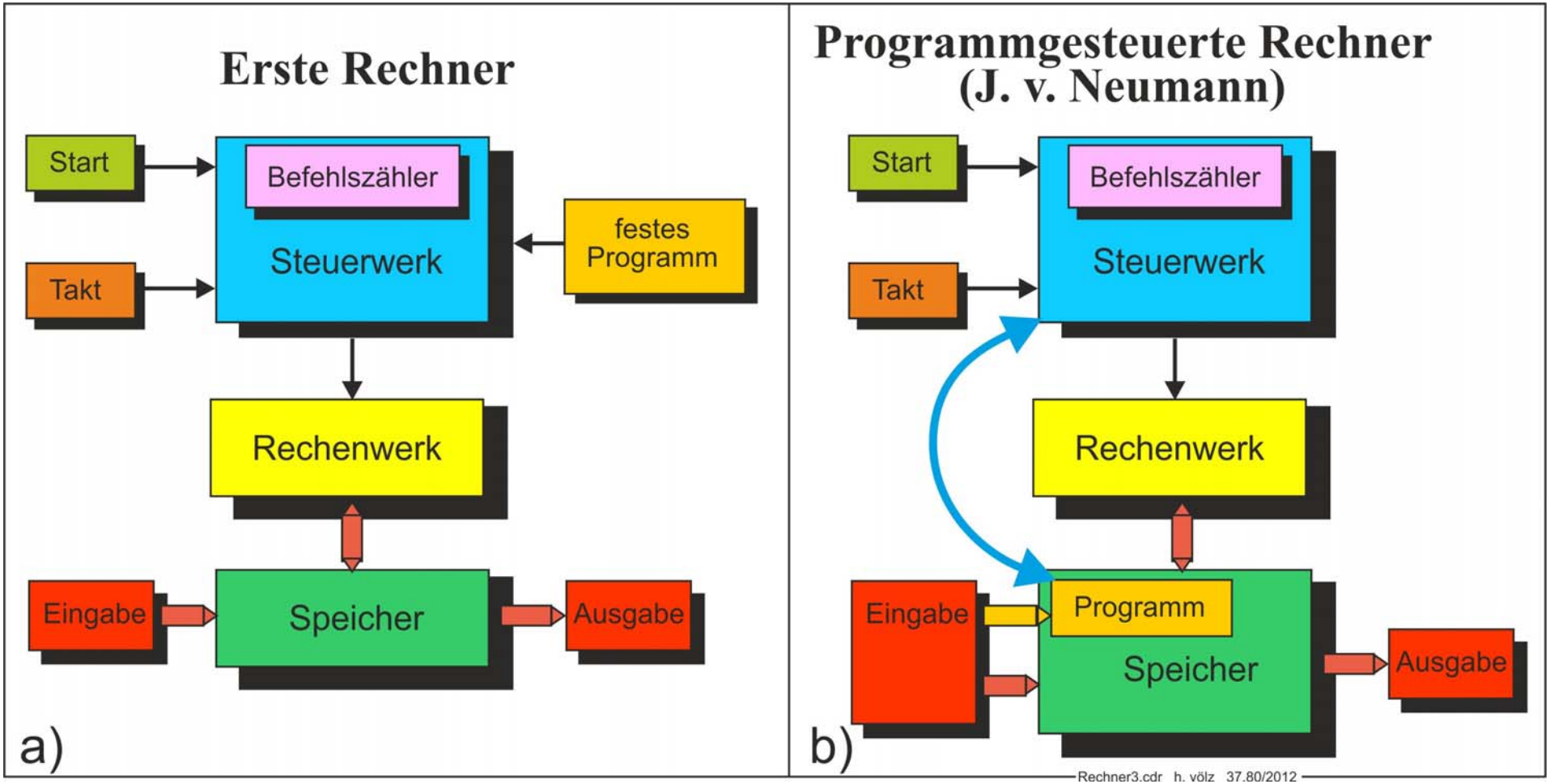
Es ist kein Beweis dafür möglich, ob ein Rechner bei einem Programm das Ω erreicht.

Dieses ***Halteproblem*** hat TURING schon 1940 bewiesen. Genauso wenig lässt sich zeigen, ob ein Algorithmus überhaupt das gewünschte Problem löst. Auch ob den nichtberechenbaren Funktionen etwas in der Realität entspricht, ist unsicher.

Zum programmierbaren Universalrechner

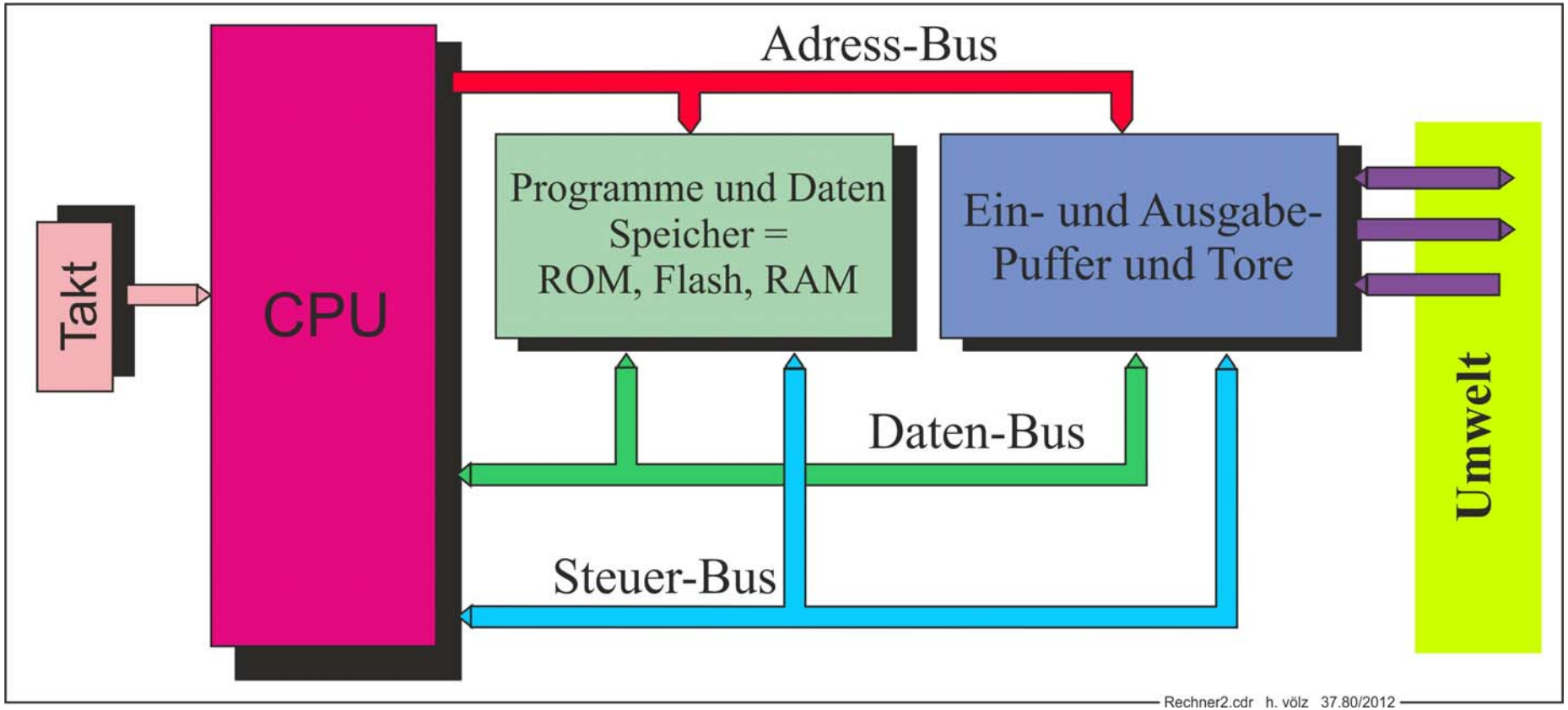
Zu Beginn wurde das Programm als ***Hardware-Bauteil*** in den Rechner eingefügt.
Bei Programmänderungen musste es ausgetauscht werden.
Um 1945 legte JOHN VON NEUMAN (1903 – 1957) das Programm ***in den Arbeitsspeicher*** des Rechners.
Dadurch wurde es genauso leicht wie die Eingangs-Daten veränderbar.
Dadurch entstand aber auch der ***„Flaschenhals“*** zwischen Rechenwerk und Speicher.
Denn Daten und Programm mussten nun diesen einen Weg nun gemeinsam passieren.
Dennoch hat sich die NEUMAN-Rechnerstruktur ohne wesentliche Änderungen schnell durchgesetzt.
Sie ermögliche die leichtere und schnellere Programmierung.

Ferner ist zu beachten, dass ein Universalrechner auch gut zur ***Steuerung von Prozessen*** geeignet ist.
Dann erzeugt er eigentlich keine mathematischen Rechenergebnisse sondern Ausgaben (Wirkungen).
Sie beeinflussen den zu steuernden Prozess.


Erste Rechner
Start
Befehlszähler
Steuerwerk
festes Programm
Takt
Rechenwerk
Eingabe
Speicher
Ausgabe
a)
Programmgesteuerte Rechner
(J. v. Neumann)
Start
Befehlszähler
Steuerwerk
Takt
Rechenwerk
Eingabe
Programm
Speicher
Ausgabe
b)
Rechner3.cdr h. völz 37.80/2012

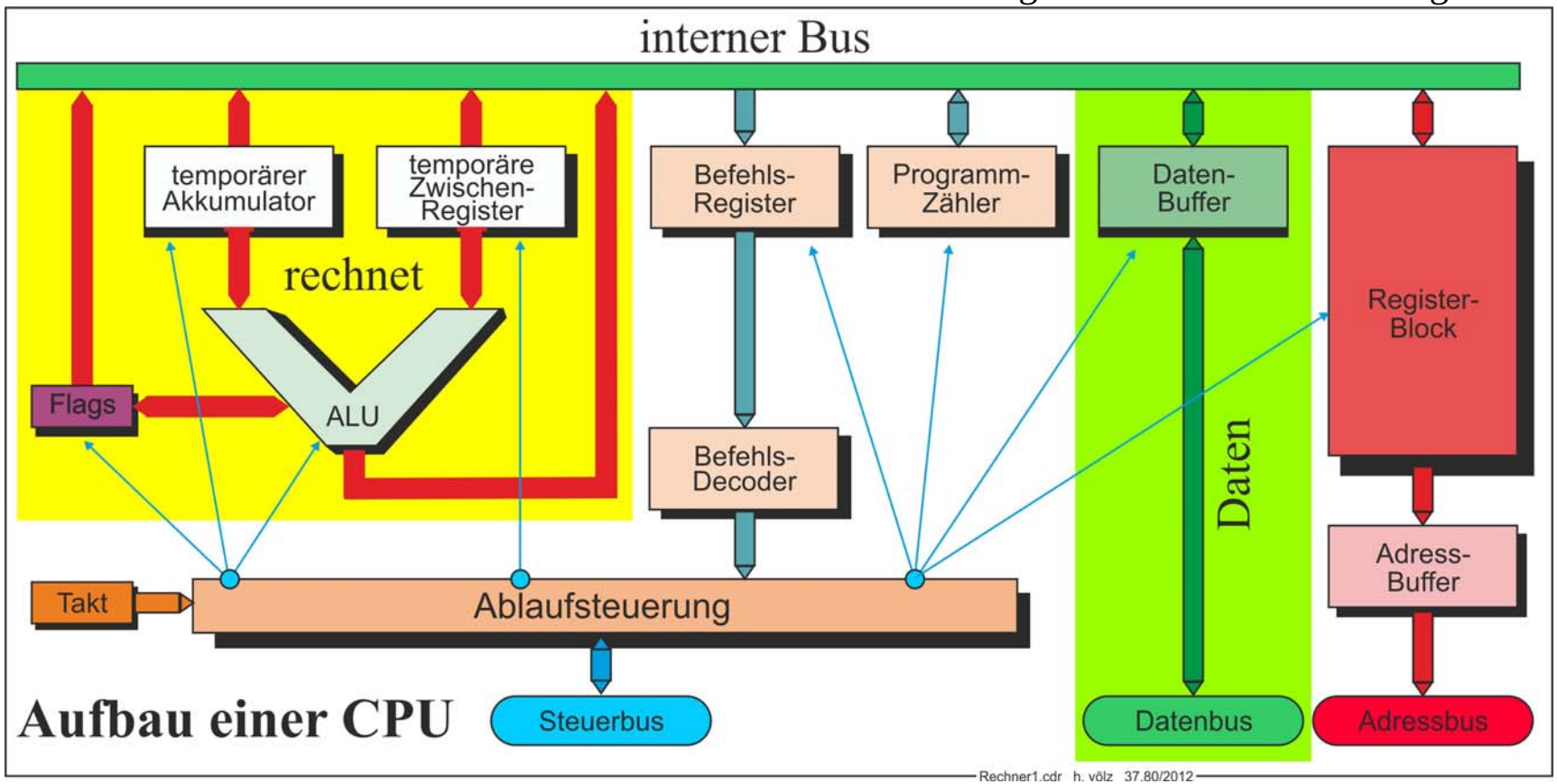
Die CPU für neue Rechnerstrukturen

1970 schuf TED HOFF die erste CPU 4004 (central processing unit).
Eigentlich war sie für einen Taschenrechner gedacht, war dafür aber zu langsam.
Aus ihr gingen viele Weiterentwicklungen für die CPU hervor.
Sie führten zu einem neuen Rechneraufbau mit den drei typischen Bussen.



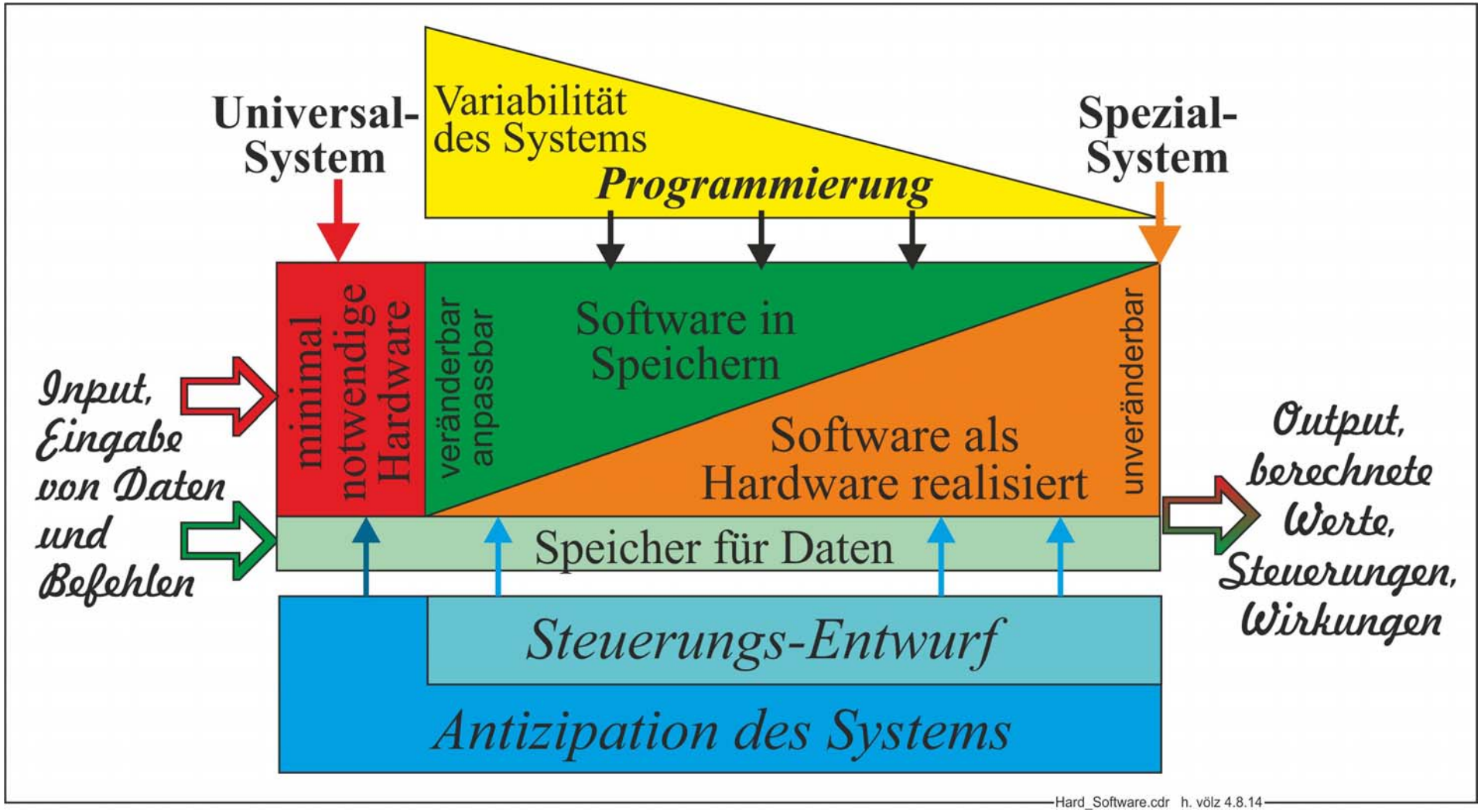
Aufbau der typischen CPU

Die minimale Hardware ist hier die ALU (Arithmetic and Logic Unit).
Sie kann als eine erhebliche Erweiterung des (hardware-) Adders.
Die anderen Teile sind in Hardware realisierte Software sowie Steuerungs- und Kontrolleinrichtungen.



Verhältnis Hard- und Software

Hard- und Software setzen eine Einrichtung, ein durch Information steuerbares System voraus.
Es wird ein kybernetischer Ursache-Wirkungs-Zusammenhang erzeugt, der überwiegend deterministisch ist.
Dieses System kann u. a. ein Spezial- oder Universalrechner sein.
Mit ihm werden aus Eingangsdaten die gewünschten Ausgangsdaten erzeugt.
Dabei können die Zusammenhänge von recht einfach bis zu höchst komplex reichen.
Die Eingangsdaten sind überwiegend Zahlen- bzw. Messwerte von Sensoren.
Die Ergebnisse können dann z. B. sein:
Rechenergebnisse, aber auch vielfältige Informationen für weitere Systeme und selbst Software.
Hard- und Software betreffen den technischen Aufbau und die Funktion des Systems.
Für die Zusammenhänge, einschließlich Ersatz von Soft- durch Hardware ergibt sich so das folgende Schema.


Universal-
System
Variabilität
des Systems
Programmierung
Spezial-
System
minimal notwendige Hardware
veränderbar anpassbar
Software in Speichern
Software als Hardware realisiert
unveränderbar
Input, Eingabe von Daten und Befehlen
Speicher für Daten
Output, berechnete Werte, Steuerungen, Wirkungen
Steuerungs-Entwurf
Antizipation des Systems
Hard_Software.cdr h. völz 4.8.14

Soft-Hardware-Ersatz

Beim ***Universal-System*** strebt die Hardware gegen ein unbedingt ***notwendiges Minimum***.
Das können u. a. ***kombinatorische Schaltungen*** oder eine **CPU** sein.
Für den Betrieb des Systems ist dann relativ ***viel Software notwendig***.
Ihre leichte Austauschbarkeit ermöglicht den universellen Einsatz.
Für ***jede neue Anwendung*** ist dann aber ein ***neues Programm*** erforderlich.

Wird die Software in Hardware übertragen so entsteht das ***Spezial-System***.
Es arbeitet meist ***deutlich schneller***, aber eben nur für den ***einen Zweck***.

Zwischen diesen beiden Extremen gibt es eine Vielzahl von Varianten.
In der ***Rechentechnik*** sind die ***drei Varianten*** mit den Beispielen üblich:
Universalrechner, programmierbarer Taschenrechner und üblicher, ganz einfacher Taschenrechner Beispiele.

Wechselwirkung von Hard- und Software

Bei der Hardware oder Programmierung gewonnene neue Erkenntnisse gelangten selten ins andere Gebiet. Die Wechselwirkung zwischen Theorie, Hardware und Software ist insgesamt sehr gering geblieben. Hierauf weist bereits NICKEL in [1] auf S. 392: hin:

"Kaum jemals haben die Beschränkungen oder Vorteile einer speziellen Hardwaretechnik dazu angeregt, spezielle Softwaretechniken zu entwickeln. Ein Beispiel für einen Ausnahmefall - der allerdings nie praktisch realisiert worden ist - ist die Feld-Rechenmaschine von Konrad Zuse. Sie will die Tatsache ausnutzen, daß im Trommelspeicher (oder Plattenspeicher oder Kernspeicher, dagegen nicht im Bandspeicher) die gespeicherte Information flächenmäßig, d. h. zweidimensional vorliegt. Zuse wollte diese hardwaremäßige Eigenschaft nutzbar machen für die softwaremäßige Behandlung partieller Differentialgleichungen, die sich sonst in den (eindimensionalen) modernen Computern nicht "sachgemäß" behandeln lassen.
Der umgekehrte Fall, daß bewährte Softwaretechniken in die Hardware übernommen wurden, trifft zwar immer mal wieder ein (siehe ... Verdrahtung der Gleitkomma-Arithmetik oder der Duodezimal-Konversation). Seltsamerweise ist es jedoch auch jetzt, mehr als 10 Jahre nach der Entwicklung des Kellerungsprinzips, immer noch nicht selbstverständlich, eine moderne Rechenanlage hardwaremäßig mit einem oder mehreren Kellern auszurüsten. Dieses Hinterherhinken gegenüber den bewährten Softwaretechniken findet man auch in vielen anderen Fällen."

Beispiel Stack

Wahrscheinlich hat WILHELM KÄMMERER (1905 – 1994) die Idee erstmalig in seiner Dissertation behandelt.
Bereits bei Vorarbeiten zur Rechenmaschine OPREMA – Betrieb 1955 – hat er dieses Prinzip als LIFO (last in first out) angewandt.

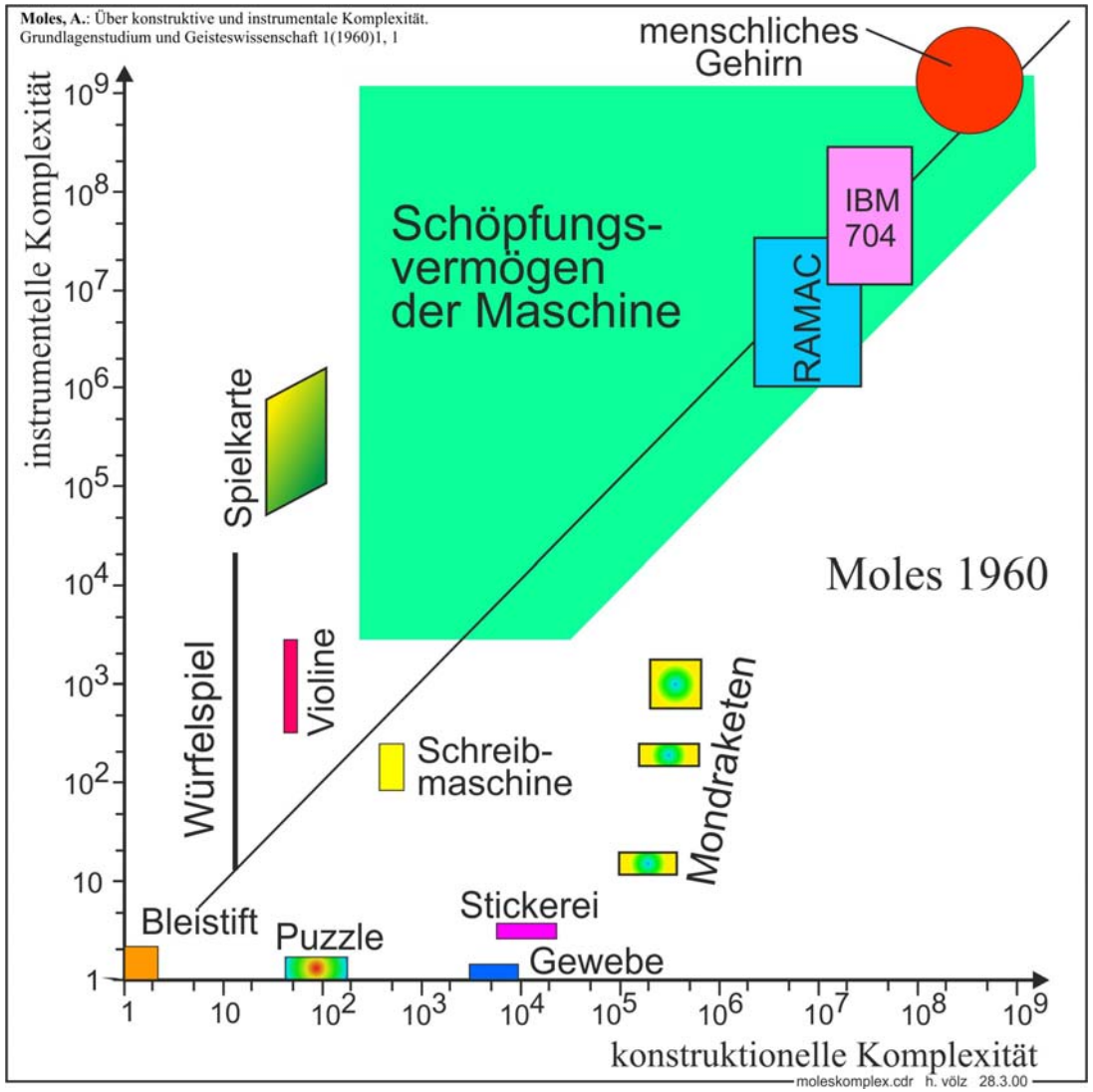
Zuweilen werden auch FRIEDRICH LUDWIG BAUER (*1924) und KLAUS SAMUELSON genannt.
Sie dieses Prinzip um 1957 bei Programmiersprachen eingeführt haben.
Bis heute ist hierfür jedoch keine echte Hardware-Lösung bekannt.

Zur Komplexität

Bereits 1960 unterschied ANDRÉ ABRAHAM MOLES (1920 – 1992) zwischen ***instrumenteller und konstruktiver Komplexität***.

Dazu gehört das nur leicht von mir geänderte Bild rechts.

Er stellte untersuchte, wie komplex ein System aufgebaut ist (entspricht der Hardware)
bzw. wie groß die Vielfalt der Anwendungs-Möglichkeiten für das jeweilige System ist (entspricht der Software).



Einige Schlussfolgerungen

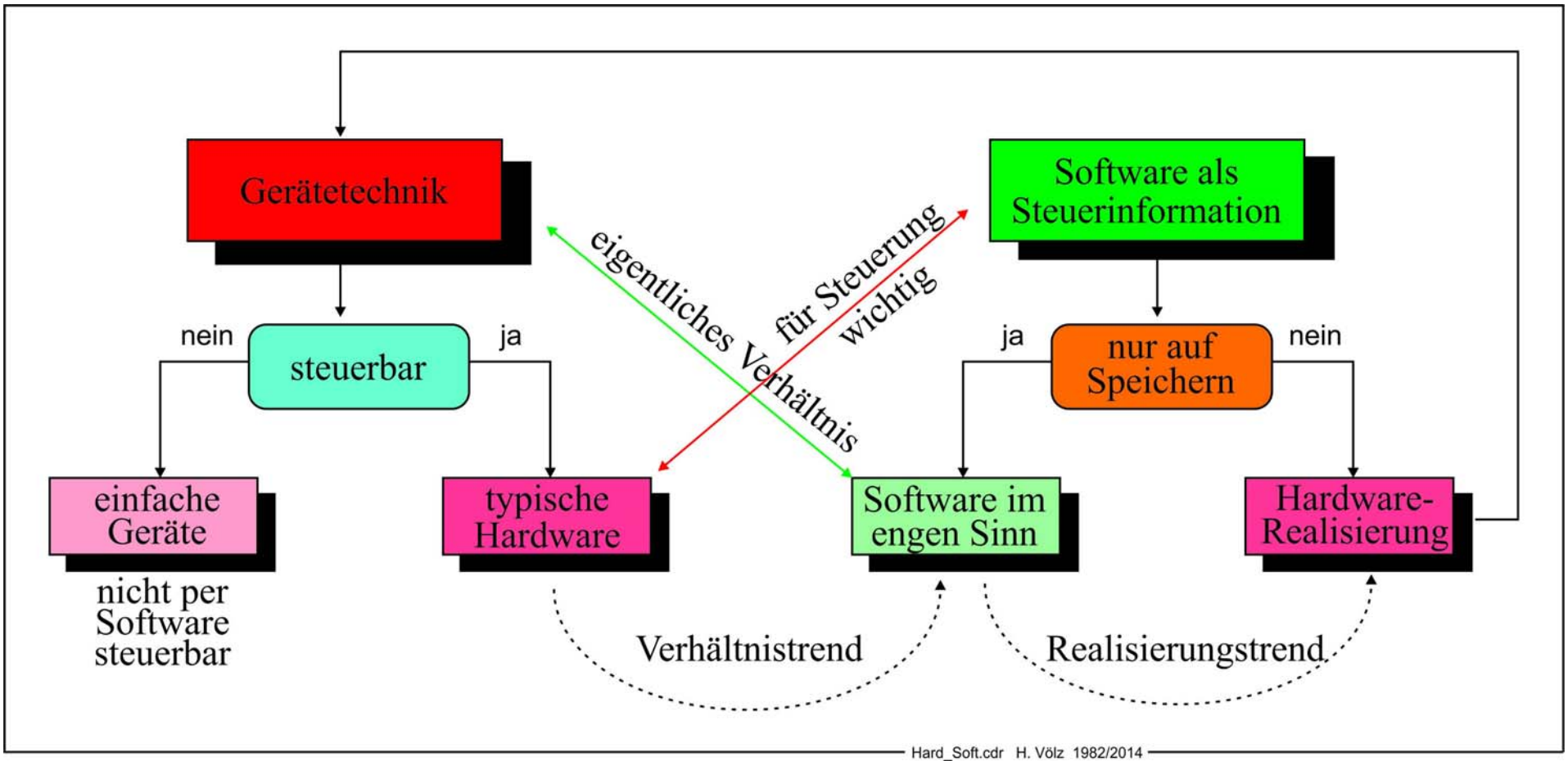
FRIEDRICH A. KITTLER (1943 – 2013) vertritt nun die Auffassung, dass es keine Software gibt [5].
Nach seiner Auffassung muss sie ***immer an Hardware gebunden***, also fixiert sein.
Hierbei bleibt aber unbeachtet oder gar ungeklärt, ***woher die Daten*** in den Speicher ***kommen***.
Negiert wird dabei ***Unterschied*** zu beachten:

- Ob das Ausgangsprodukt nur in ***Speichern abgelegt*** wird oder
- zur Strukturierung von ***spezieller Hardware*** genutzt wird.

Weiter ist zu beachten, dass jedes technische System zunächst ***antizipiert*** werden muss.
Erst dann kann es ***entwickelt*** und dann ***konstruiert*** werden.
Dabei ist auch der ***Steuerungsentwurf*** für die „Software“ im engeren Sinn zu erzeugen.
Sie kann dann für das jeweilige System programmiert oder in entsprechende ***Hardware umgesetzt*** werden.
Zumindest verbleibt so die Software bis zum Eingeben in den Speicher ein ***geistiges Produkt***.

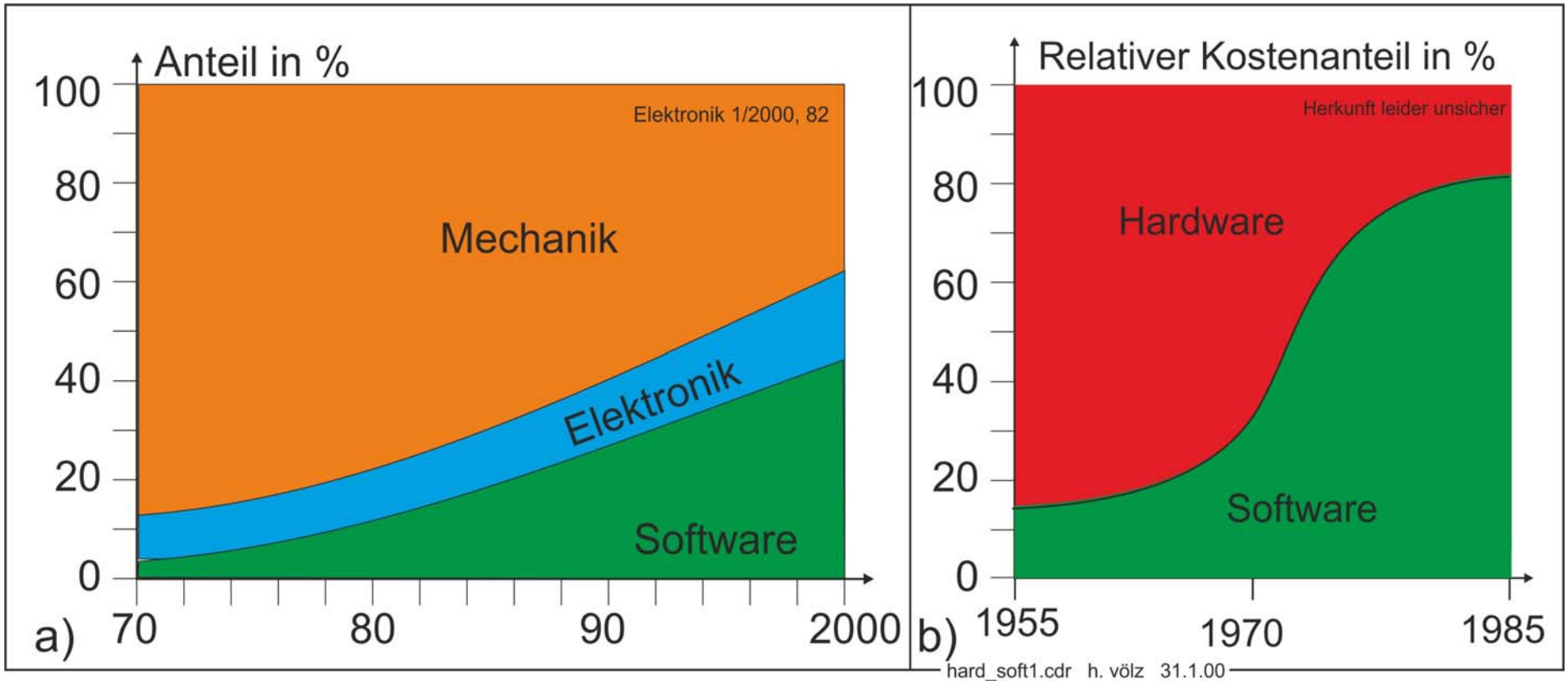
Aus ähnlichen Gründen habe ich bereits 1982 das ***folgende Bild*** gezeichnet [2] S. 177.
Es macht das dialektische Verhältnis von Hard- und Software deutlich.
Außerdem weist es deutlich die vielfältigen Zusammenhänge aus.

Hierbei gibt es gewisse **Trends**.
So wird seit langem immer mehr Software in Hardware-Lösungen übersetzt.
Erste Beispiele waren Mikroprogramme (eingeführt 1951 durch MAURICE VINCENT WILKES; *1913).
Dann der Arithmetik-Prozessoren (um 1980) und weiter Video-, Audio-Prozessoren, Simulatoren usw.


Gerätetechnik
Software als Steuerinformation
eigentliches Verhältnis
für Steuerung wichtig
nein
steuerbar
ja
ja
nur auf Speichern
nein
einfache Geräte
typische Hardware
Software im engen Sinn
Hardware-Realisierung
nicht per Software steuerbar
Verhältnistrend
Realisierungstrend
Hard_Soft.cdr H. Völz 1982/2014

Software nimmt überproportional zu

Im Laufe der Geschichte wird der Software-Anteil bei vielen Rechnern immer größer.
Leider sind hierzu kaum genauere Angaben zu finden.
Zwei gefundene Beispiele zeigt das Bild. a) betrifft die Produktionsanteile und b) die relativen Kosten.



Gegenüberstellung von Hard- und Software

Bezug	Software	Hardware
Gesetze, Komplexität	Gehorcht logischen Gesetzen, dabei ist sehr hohe Komplexität möglich, bringt aber Probleme zu deren Beherrschung, kann so teilweise Grenzen der Hardware unterlaufen.	Folgt physikalischen Gesetzen, meist erheblich begrenzte Komplexität.
Zweck und Einsatz	Ist nur für eine gegebene Hardware (als Steuerungs-Information) nützlich, ermöglicht dessen Spezialisierung (in Struktur und Funktion) und damit Anpassung an verschiedene Aufgaben. Tendenz zum Universalrechner. Ist ähnlich notwendig wie die Energieversorgung.	Spezialisierte Hardware ist direkt (autonom) nutzbar, aber meist nur im eingeengten Anwendungsbereich. Tendenz zum Spezial-Rechner.
Arbeitsweise	Überwiegend sequentiell	Gute Parallelität ist möglich.
Herstellung, Kosten	Durch Programmieren (Simulation, Testphasen), oft schneller herstellbar als entsprechende Hardware, z. T. auch weniger Risiko. Fast nur Lohnkosten sowie Benutzergebühren	Musterbau; kann auch im Voraus (preiswert!) als Software getestet (simuliert) werden. Löhne, Materialkosten, Vorrichtungen, Maschinenpark, Lizenzen
Vervielfältigung	Sehr einfach, keine Spezialkräfte, wesentlich ohne Material, so gut wie keine Ressource-Grenzen,. umweltfreundlich, kein schädlicher Abfall, kein Patentschutz möglich, mit der Folge, dass viel illegale Software benutzt wird.	Verlangt stets Material, umfangreiche technische Einrichtungen mit hohem technisch-technologischem Niveau (nur einmalig) und spezialisierte Arbeitskräfte. Nicht immer umweltfreundlich.
Speicherung, Archivierung	Muss immer in irgendeiner Form gespeichert werden (z. B. Papier oder Datei).	Muster und Konstruktionsunterlagen. Hardware ist größtenteils materialisierte Information.
Datenschutz, Aktualität	Softwareschutz (Aktivierung usw.) und Pflege (updates). Ist leichter als Hardware auszutauschen.	Patentschutz, (schneller) moralischer Verschleiß, erhebliche Probleme bzgl. der Kompatibilität.

Ergänzungen

Zu diesen Fakten ist noch einiges hinzuzufügen.
Die ***Bezeichnung der Computergenerationen*** erfolgte auf Grundlage ihrer Hardware:
1. Generation: Röhrenmaschinen;
2. Generation: Transistorgeräte;
3. Generation: integrierte Bausteine
4. Generation (nur z. T.): mikrominiaturisierte und sehr stark integrierte (large scale integrated) Bausteine.

Hardware im allgemeinen Sinne ist ***materiell (stofflich)*** festgelegt und betrifft dann ***nicht nur die Elektronik***.
Schon lange wurde dazu auch die ***steuerbare Pneumatik*** und ***Hydraulik*** dazugerechnet.
Künftig dürfte das wohl auch für ***optische Techniken*** gelten.

Auch die Formeln für ***Konstanten*** wie π und e können als Software für deren Werte angesehen werden.
Technisch lassen sich entweder die Werte als Zahlen speichern oder im Rechner mit den Formeln generieren.

Außerhalb der Technik

Auch hier kann das Verhältnis Soft-Hardware angewendet werden.

So werden zuweilen ***Noten*** als Software für Musik (menschliche Stimme) angesehen.
Entsprechend sind die Streifen (***Lochbänder***) Software für den WELTE-STEINWAY-Flügel.
Sie enthalten eigentlich nur (exakte) Information dafür, wie der jeweilige Pianist gespielt hat.
Dafür sind sie dann „nur“ ein Informationsspeicher.

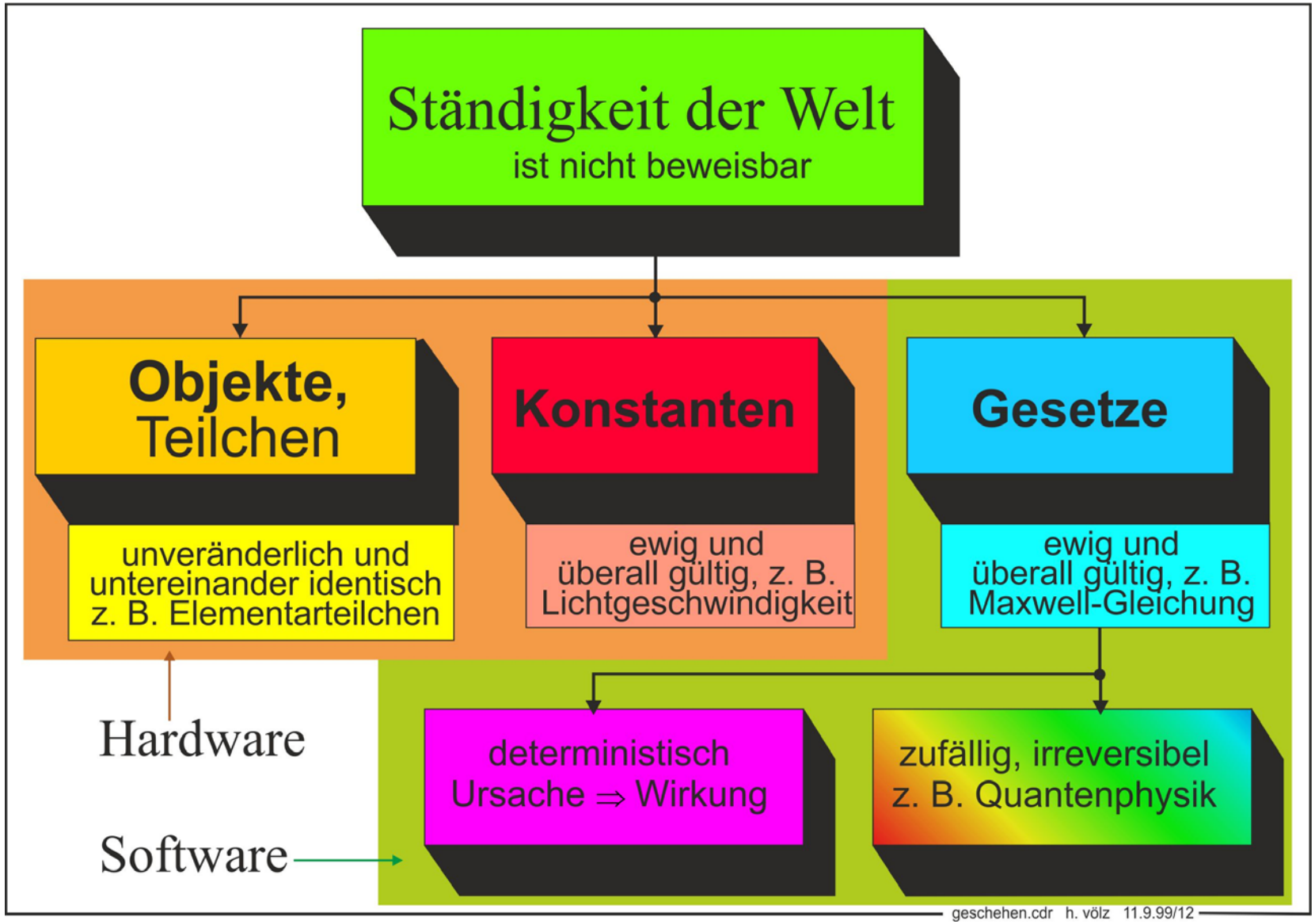
Folglich ist es auch falsch, Schall- oder Bildplatten und CD-DA als hardwarerealisierte Software zu betrachten.

Natürlich gibt es dabei jedoch ***Grenzprobleme***:
Eine ***Messschallplatte*** kann z. B. Steuerinformation für einen Messkomplex enthalten.
Musik kann unsere Sinne und ***Empfindungen „steuern“***.
Extrem ist es, die ***Schiene*** als „Steuerinformation" für ein Räderfahrzeug anzusehen.

Schließlich sei noch ein möglicher Übergang in die ***Biologie und Wirklichkeit*** angeführt.
So enthält eine **DNS** sowohl Daten als auch Steuerungs-Information für das jeweilige Lebewesen.
Für die ***Natur (Wirklichkeit)*** entsprächen

- die ***Konstanten*** und Teilchen den Daten der Hardware und
- die ***Naturgesetze*** eine Steuerungs-Information, also der Software.

Hierfür gilt das folgende Bild


Ständigkeit der Welt
ist nicht beweisbar
Objekte,
Teilchen
Konstanten
Gesetze
unveränderlich und
untereinander identisch
z. B. Elementarteilchen
ewig und
überall gültig, z. B.
Lichtgeschwindigkeit
ewig und
überall gültig, z. B.
Maxwell-Gleichung
Hardware
Software
deterministisch
Ursache ⇒ Wirkung
zufällig, irreversibel
z. B. Quantenphysik
geschehen.cdr h. völz 11.9.99/12

Literatur

[1] Nickel, K.: Die Dualität Hardware – Software. in Nova Acta Leopoldina Band 37.1; Nr. 206; Informatik, Johannes Ambrosius Barth, Leipzig 1972.

[2] Völz, H.: Information I - Studie zur Vielfalt und Einheit der Information. Akademie Verlag, Berlin 1982.

[3] Völz, H.: Elektronik - Grundlagen - Prinzipien - Zusammenhänge. 5. Aufl. Akademie Verlag, Berlin 1989

[4] Völz, H.: Grundlagen der Information. Akademie - Verlag, Berlin 1991,

[5] Kittler, F.: Draculas Vermächtnis - Technische Schriften. Reclam-Verlag, Leipzig 1993

[6] Völz, H.: Grundlagen und Inhalte der vier Varianten von Information. Springer Verlag. Wiesbaden 2014.

[7] http://de.wikipedia.org/wiki/Hardware Download 27.7.14

[8] Wiener, N.: "Cybernetics or control and communication in the animal and the machine" Hermann, Paris 1948; in "Regelung und Nachrichtenübertragung in Lebewesen und in der Maschine", Econ - Verlag, Düsseldorf - Wien 1963. Econ - Verlag, Düsseldorf - Wien - New York - Moskau, 1992

[9] Völz, H.: Handbuch der Speicherung von Information Bd. 1 Grundlagen und Anwendung in Natur, Leben und Gesellschaft. Shaker Verlag Aachen 2003

[10] Völz, H.: Handbuch der Speicherung von Information Bd. 2 Technik und Geschichte vorelektronischer Medien. Shaker Verlag Aachen 2005

[11] Völz, H.: Handbuch der Speicherung von Information Bd. 3 Geschichte und Zukunft elektronischer Medien. Shaker Verlag Aachen 2007

[12] Moles, A.: Über konstruktive und instrumentale Komplexität. Grundlagenstudium und Geisteswissenschaft 1(1960)1, 1