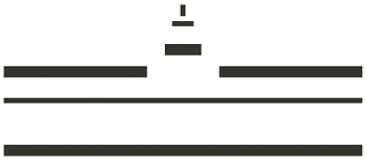


Fachbereich Mathematik und Informatik

Bachelorarbeit

Entwicklung eines Studientextes „Wie funktionieren Computer?“ für die Vorlesung „Informatik und Gesellschaft“ in den Allgemeinen Studien

Development of an essay „Wie funktionieren Computer?“ for the lecture „Informatik und Gesellschaft” in general studies

vorgelegt von

Eva Kubitz

Studiengang: 2-Fach-Bachelor Mathematik / Informatik

Betreuender Professor: Herr Prof. Dr. Marco Thomas

Zweitprüfer: Herr Prof. Dr. Achim Clausing

Ausgabedatum: 17.06.2011

Abgabedatum: 29.07.2011

Inhaltsverzeichnis

1. Zielsetzung

Im Rahmen dieser Bachelorarbeit wird ein Studientext entstehen, der Hintergrundwissen der Informatik zu der Vorlesung *Informatik und Gesellschaft*[1] von Herrn Prof. Dr. Marco Thomas an der Westfälischen Wilhelms Universität Münster zur Verfügung stellt. Die Vorlesung wird innerhalb der *Allgemeinen Studien* angeboten und daher nehmen auch Studierende teil, die fachfremd in der Informatik sind und somit wenig fachliches Vorwissen mitbringen. Der Studientext ist für diese Studierenden konzipiert.

In der Vorlesung IuG[2] erhalten die Studierenden Studienbriefe und multimediales Material, die in Heimarbeit bearbeitet, mittels Übungsaufgaben vertieft und in Präsenzveranstaltungen diskutiert werden. Die Übungsaufgaben können online bearbeitet und einreicht werden. In einer Blockveranstaltung präsentieren die Studierenden eigene Ausarbeitungen zu Themen der IuG.

Der Studientext ist dazu gedacht, Wissen zu vermitteln, welches das Verständnis der Vorlesung IuG erleichtert. Dafür ist der direkte Bezug zu den Studienbriefen aus der Vorlesung ratsam. So können gezielt Themen der Informatik behandelt werden, die den Zugang zu den Inhalten der jeweiligen Studienbriefe verbessern.

Besonders bei den Studienbriefen der ersten beiden Vorlesungseinheiten ist eine fachliche Unterstützung wichtig, da hier Sichtweisen auf das Fach IuG behandelt werden und somit das Fach an sich eingeordnet wird. Außerdem kann das erlangte Fachwissen auch bei der Bearbeitung der restlichen Vorlesungseinheiten von Vorteil sein. Daher bezieht sich der Studientext auf die Themen der Informatik, die das Verfolgen der ersten beiden Vorlesungen erleichtern. Das Material der ersten Vorlesungseinheit enthält drei kurze und das der zweiten Vorlesungseinheit ein en langen Studienbrief. Zwei der kurzen Studienbriefe beziehen sich weniger auf fachliches Wissen und sind auch für fachfremde[3] Studierende verständlich. Deswegen wird hier nicht näher auf sie eingegangen. Der dritte der kurzen Studienbriefe (Keil-Slawik, 2001)[4] und der längere und komplexere Studienbrief (Engbring, o.J.) aus der zweiten

---

[1] In dieser Bachelorarbeit werden Eigenbegriffe kursiv gedruckt, wenn sie hervorgehoben werden sollen. Dazu gehören Fachbegriffe aus der Informatik, Eigenbegriffe aus den Vorlesungsmaterialien der Vorlesung Informatik und Gesellschaft und Titel von Vorlesungen, Studienmodule und Texten.

[2] *IuG* wird im Folgenden als Abkürzung für *Informatik und Gesellschaft* verwendet.

[3] Im Folgenden steht *fachfremd* als Synonym für *fachfremd in der Informatik*.

[4] Literaturnachweise werden in dieser Bachelorarbeit in Form von Kurzbelegen aufgeführt.

Vorlesungseinheit sind ohne einige Grundlagenkenntnisse der Informatik schwer zu bearbeiten. Für ein Verständnis der Studienbriefe sind Kenntnisse aus dem Bereich des Rechneraufbaus und der Verarbeitung von Programmen durch einen Computer notwendig. Der Studientext beschäftigt sich daher mit diesen Themen. Der genauere Bezug des Studientextes zu den beiden oben genannten Studienbriefen wird in Abschnitt[5] 2.2 erläutert.

Zusätzlich zu dem gerade beschriebenen direkten Bezug zu den Inhalten der Vorlesung IuG hat das Thema des Studientextes noch einen weiteren Vorteil. Die Studierenden könnten zu Beginn der Vorlesung IuG die Erwartung haben, man würde sich hier mit den Auswirkungen von Informatik auf die Gesellschaft beschäftigen. Es ist möglich, dass bei ihnen ein Interesse daran besteht, mehr darüber zu wissen, wie Computer funktionieren, um die Wirkung der Informationstechnik auf die Gesellschaft besser einschätzen zu können. Daher kann die Beschäftigung mit der Funktionsweise von Computern einen persönlichen Nutzen für die Studierenden bringen, der über den Nutzen für das Verständnis der Vorlesung IuG hinausgeht.

Im nun folgenden Hauptteil dieser Arbeit wird der Studientext entwickelt. Darauf folgt das Ergebnis dieser Arbeit, der Studientext selbst. Am Ende stehen Evaluation und Fazit. In der Evaluation werden die Rückmeldungen einiger Studierenden, die den fertigen Studientext gelesen und einen Fragebogen dazu ausgefüllt haben, kurz erläutert.

2. Entwicklung des Studientextes

Der Studientext beschäftigt sich mit der Funktionsweise von Computern bzw. mit Rechneraufbau und Programmverarbeitung und trägt den Titel: *Wie funktionieren Computer?*

Die Basis für die Entwicklung des Studientextes bilden die fachlichen Inhalte der Studienbriefe, auf die er sich bezieht und didaktische Überlegungen zur Gestaltung. Außerdem ist die Zielgruppe, für die der Studientext verfasst wird, mit einzubeziehen. Die Entwicklung des Studientextes findet in mehreren Schritten statt.

Im Folgenden wird zunächst kurz auf die Zielgruppe eingegangen. Danach werden die fachlichen Inhalte des Studientextes anhand der Studienbriefe und in Bezug auf die Zielgruppe herausgearbeitet und in eine fachlich sinnvolle Reihenfolge gebracht.

[5] Die Kapitel der Bachelorarbeit werden in Abgrenzung zu den Kapiteln des Studientextes im Folgenden *Abschnitte* genannt.

Im nächsten Schritt werden allgemeine dida ktische Ideen zur Gestaltung dargelegt und auch konkrete Maßnahmen zur Umsetzung der didaktischen Ideen vorgestellt. Abschließend wird die Gestaltung der einz elnen Kapit el des Studientextes, die sich auf die herausgearbeiteten fac hlichen Themen und didaktis chen Überlegungen stützt, erläutert. Danach wird kurz die zu Hilfe genom mene Literatur vorgestellt und die Überlegungen zu Literaturempfehlungen für die LeserInnen des Studientextes werden erörtert.

2.1. Überlegungen bezüglich der Zielgruppe

Es kann bei den Studierenden an ein gewi sses Vorwissen angeknüpft werden. Im Studienalltag ist die Nutzung v on Comput ern gegeben, somit gehen alle Personen der Zielgr uppe regelmäßig mit Com putern um und haben dementsprechende Anwenderkenntnisse. Außerdem haben alle Studierenden ein Abitur o der eine gleichwertige Vorbildung, so dass auch auf dies em Vorwiss en aufgebaut wer den kann.

Die LeserInnen des Studientext es sind Informa tik-Fachfremde. Daher ist es nur bis zu einem gewiss en Grad not wendig, das s Erklärungen de tailgetreu sind und die Wirklichkeit genau abbilden. Stattdessen liegt der Fokus auf dem, was für ein Grundverständnis der Thematik und den Bezug zu den Studienbriefen wesentlic h ist. Da Fachfremde mit der Arbeitsweise in der Informa tik nicht vertraut s ind, ist es wichtig, theoretische Erklärungen mit Analogien aus der Alltagsw elt und mit grafischen Elementen zu veranschaulichen.

2.2. Ausarbeitung der Inhalte

Der erste der beiden Studienbrie fe, auf die der Studientext Bezug nimmt, heißt „Von Informatik und Gesellschaft zum Kontext der Informatik“ (Ke il-Slawik, 2001: S. 39). In der Textpassage „Die Sichtweise“ (Keil-Slawik, 2001: S.40) des Studienbriefs geht es darum, Besonderheit en der Informatik dar zulegen und so einz ugrenzen, was das Fach IuG eigentlich umfasst. S oftware wird hier als der Gegenstand der Informatik definiert und infolgedessen werden in dies em Teil des Studienbriefs die Besonderheiten von Software beleuchtet. Es wir d hingeführt zu dem Begriff vom *Kontext der Informatik*. Damit meint Keil-Slawik den „ Herstellungs- und Einsatzkontext “ (2001: S.42) von Software. (Vgl. Keil-Slawik, 2001)

Programmieren ist nach Keil-Slawik das Erstellen von Steuerungen, die auf Zeichen basieren. Sowohl die Verarbeitung von Programmen durch den Prozessor als auch die Unterschiede von Maschinensprache und höherer Programmiersprache werden in diesem Rahmen angesprochen. Es wird erklärt, dass die heutige Komplexität von Programmen erst durch höhere Programmiersprachen praktisch möglich ist, da sie lesbar und intellektuell beherrschbar sind. Keil-Slawik legt dar, dass Zeichen bzw. Programme bei der Verarbeitung durch den Computer in Signale übertragen werden, die die Steuerung der Abläufe im Computer bewirken. Bezüglich Maschinensprache und höherer Programmiersprache wird herausgestellt, dass diese insofern logisch äquivalent sind, als sie durch eine *formale* Übersetzung ineinander überführt werden können. Formal steht dafür, dass sich etwas „nur auf die Form und die Anordnung der Zeichen, nicht aber darauf, wofür sie stehen“ (Keil-Slawik, 2001: S.41) bezieht. Im Vordergrund der Ausführungen von Keil-Slawik steht, dass Programme aus formaler Schrift bestehen und daher eine formale Verarbeitung möglich ist, ohne die Zeichen zu verstehen. An dieser Stelle ist laut Keil-Slawik der Bogen von der Informatik zu dem Kontext der Informatik zu schlagen, da die Bedeutung der Zeichen erst bei ihrer Herstellung und Verwendung klar wird. So macht Keil-Slawik deutlich, dass es wichtig ist, den Kontext der Informatik mit einzubeziehen. (Vgl. Keil-Slawik, 2001)

Die Studierenden der Zielgruppe haben Begriffe wie *Maschinensprache*, *höhere Programmiersprache* und *Binärkodierung* noch nie gehört und auch noch nie Programmcode gesehen. Deswegen können sie nur schwer verstehen, wovon Keil-Slawik spricht. Vor allem ist für sie nicht leicht zu erkennen, wie so es Keil-Slawik für notwendig hält, den Kontext der Informatik zu betrachten. Um das nachzuvollziehen, ist es wichtig zu wissen, inwiefern Computer verarbeiten ohne zu verstehen, da dies den Kontext so bedeutend macht.

Der Studientext beinhaltet deswegen eine Erklärung, was Programmcode ist und inwiefern der Computer verarbeitet ohne zu verstehen. Ferner ist ein Einblick in Maschinensprache, höhere Programmiersprache und in deren Übersetzung zu geben, damit die oben genannten Besonderheiten dieser von Keil-Slawik angesprochenen Sprachebenen klar werden. Ein weiteres Ziel ist es darzustellen, inwiefern auch die Übersetzung formal erfolgt. Um herauszustellen, dass der Computer Zeichen nicht versteht wie Menschen sie verstehen, enthält der Studientext eine Beschreibung von Binärkodierung. Dabei ist wichtig, dass Befehle bei der Umsetzung in

Steuersignale abhängig von den Schaltkreisen im Prozessor interpretiert werden, nicht aber nach Wortbedeutung.

Keil-Slawik erwähnt den Begriff der *Assemblersprache* nicht. Daher ist es nicht notwendig, im Studientext zwischen Maschinensprache und Assemblersprache zu unterscheiden[6]. Es ist jedoch wichtig, dass es zum einen eine Programmiersprachebene gibt, die direkt den Prozessor anspricht und zum anderen eine höhere Ebene.

Der zweite Studienbrief, der für den Studientext relevant ist, trägt den Titel „‚kontextuelle Informatik' statt IuG" (Engbring, o.J.: S.1[7]). Es werden ähnliche Fragen angesprochen, der Studienbrief ist jedoch deutlich komplexer und behandelt das Thema noch tiefgehender. Ein schon bestehender, von Keil-Slawik entwickelter Ansatz[8], die *Differenzierung zwischen Produkt und Prozess*, wird von Engbring vorgestellt. Einem Produkt werden dabei folgende Eigenschaften zugeschrieben: Steuerbarkeit von außen, Determiniertheit, Verlässlichkeit, absolute Semantik und die Möglichkeit, einen früheren Zustand wiederherzustellen. Ein Prozess wird hingegen durch Unumkehrbarkeit, Selbstorganisation, eine auf die Umwelt bezogene Semantik und die Tatsache, dass vollständige Steuerung von außen nicht möglich ist, charakterisiert. So können die technischen Produkte der Informatik von dem Prozess ihrer Herstellung und Nutzung abgegrenzt werden und getrennt oder in gegenseitiger Ergänzung betrachtet werden. Des Weiteren stellt Engbring einige Begriffe aus der Informatik, die als Produkt eingeordnet werden, ähnlichen Begriffen im allgemeinen Sprachgebrauch gegenüber, welche einem Prozess zugeordnet sind. Es handelt sich um folgende Begriffe: *Daten* vs. *Information*, *Zeichen* vs. *Nachricht*, *Speicher* vs. *Gedächtnis*, *formale Typographie* vs. *Sprache*, *Formalismus* vs. *Verstehen*. (Vgl. Engbring, o.J.)

Die Erklärung zur Produkt-Prozess-Differenzierung im Studienbrief von Engbring bildet die Grundlage für die weiteren Inhalte des Studienbriefs. Daher ist es sehr wichtig, dass die Studierenden die Unterscheidung von Produkt und Prozess verstehen. Um die Differenzierung von Produkt und Prozess bis in die Tiefe nachvollziehen

---

[6] Im Folgenden steht der Begriff *Maschinensprache* für *Assembler-* und *Maschinensprache*.

[7] Der Studienbrief von Engbring hat im Original keine Seitenzahlen. Die hier aufgeführten Seitenzahlen entsprechen einer Nummerierung, die bei der ersten Seite des Studienbriefs mit 1 beginnt.

[8] In dieser Bachelorarbeit wird nicht auf die Originalquellen dieses Ansatzes eingegangen, da die Studierenden, die die Vorlesung IuG besuchen, direkt mit dem Studienbrief von Engbring arbeiten. Der Studientext ist dafür konzipiert, das Verständnis der Vorlesungsmaterialen und nicht der den Vorlesungsmaterialen zu grunde liegen den Literatur zu unterstützen. Die Literaturangaben sind im Studienbrief von Engbring enthalten und können dort nachvollzogen werden.

zu können, ist ein Verständnis von dem notwendig, was den Computer, seine Bestandteile und Funktionsweise zum Produkt macht. Das beinhaltet die Frage, inwiefern ein Computer die oben genannten Eigenschaften eines Produkts - Steuerbarkeit von außen, Determiniertheit, Verlässlichkeit, absolute Semantik und die Möglichkeit einen früheren Zustand wiederherzustellen - besitzt. Damit die Studierenden den Studienbrief von Engbring besser bearbeiten können, werden die zugrunde liegenden Inhalte aus der Informatik in den Studientext aufgenommen. Welche das sind bzw. inwiefern der Computer die Eigenschaften eines Produkts innehat, wird nun erläutert.

Ein früherer Zustand des Computers kann *wiederhergestellt* werden, indem ein Speicherabbild gemacht und zu einem späteren Zeitpunkt wieder aufgespielt wird. Daher sind der Speicher und der Computer an sich zurücksetzbar. Da die Zielgruppe des Studientextes aus Fachfremden besteht, ist es sinnvoll, im Studientext zunächst zu erläutern, wie Speicher aufgebaut ist. Dadurch wird die oben beschriebene Gegenüberstellung von *Speicher* und *Gedächtnis* noch deutlicher. Davon ausgehend kann erklärt werden, dass Speicher im Unterschied zum Gedächtnis systematisch und nicht assoziativ arbeitet.

Ein Computer ist *von außen steuerbar*, da er komplett durch von Menschen programmierte Programme gesteuert wird. Die weitere Steuerung wird vom Computernutzer durch Eingabebefehle vorgenommen. Im Studientext wird deswegen darauf eingegangen, inwiefern die Programme die Hardware des Computers steuern und was die Eingabe der Benutzer wiederum damit zu tun hat. Dabei ist es wichtig zu erklären, was Programmierung und Programmcode sind, da die Zielgruppe des Studientextes noch nie damit in Berührung gekommen ist. In diesem Rahmen kann im Studientext deutlich gemacht werden, was *absolute Semantik* ist und dass Programmcode eine absolute Semantik hat. Durch diese Erklärungen sollte es den Studierenden auch möglich sein, den Unterschied der oben genannten Begriffe *formale Typographie* und *Sprache* zu verstehen.

Die Eigenschaft der *Determiniertheit und Verlässlichkeit* hat ein Computer insofern inne, als dass alle Programme formalistisch verarbeitet werden. Von der Übersetzung von höherer Programmiersprache zu Maschinencode bis hin zur Verarbeitung des Maschinencodes durch den Prozessor arbeitet ein Computer nach vorgegebenen Regeln. Diese Vorgänge sind Inhalt des Studientextes, damit die LeserInnen verstehen, wieso ein Computer deterministisch und verlässlich arbeitet.

Um die Gegenüberstellung von *Daten* und *Zeichen* zu *Information* und *Nachricht* leichter verständlich zu machen, beschäftigt sich der Studientext mit der Binärkodierung von Daten und Programmcode und der en Interpretierung durch den Prozessor. Dabei ist wesentlich, dass der Computer Daten und Programmcode nicht als Information und Nachricht, sondern nur als binäre Kombination versteht und dies e durch *Formalismus* und nicht durch *Verstehen* interpretiert. So wird auch diese Unterscheidung greifbarer.

Laut Engbring ermöglicht es die Differenzierung von Produkt und Prozess zudem, die verschiedenen Potentiale von Mensch und Computer herauszustellen (Vgl. o.J.: S.8). Um diesen Unterschied auch im Studientext darzulegen, beinhaltet dies er eine Erläuterung, welche Art von Aufgaben durch Programme, also durch Computer, gelöst werden können und welche nicht.

Die beiden gerade erläuterten Studienbriefe behandeln ähnliche Fragen und die für die Bearbeitung hilfreichen Themen aus der Informatik sind dementsprechend auch sehr ähnlich.

Neben den bisher vorgestellten Themen, die den beiden Studienbriefen gemeinsam sind, wird von Keil-Slawik ein weiteres Thema der Informatik angesprochen, jedoch eher am Rande. In einem kurzen Textstück (Keil-Slawik, 2001: S.41) werden der Begriff „Künstliche Intelligenz“ und weitere Ausdrücke aus dem Bereich der künstlichen Intelligenz wie „neuronale Netze“ und „evolutionäre Algorithmen“ genannt. Außerdem erwähnt Keil-Slawik den Streit darum, ob Computer intelligent sein können (Vgl. 2001).

Dieses Thema wird von Keil-Slawik nur beiläufig erwähnt. Außerdem würde die Erörterung der verschiedenen Teilbereiche der künstlichen Intelligenz den Rahmen des Studientextes sprengen, wenn sie zusätzlich zu den bisher herausgearbeiteten Themen erfolgt. Trotzdem wäre es gut, wenn sich die Studierenden zumindest unter dem Oberbegriff *künstliche Intelligenz* etwas vorstellen könnten. Dies ist als kurze und allgemeine Erläuterung im Rahmen des Studientextes möglich. Da es sich dabei um einen ganz anderen Bereich der Informatik handelt, ist es sinnvoll, dies es Thema getrennt von den anderen Themen zu behandeln.

Der von Keil-Slawik erwähnte Streit über die Frage, ob Computer intelligent sein können (Vgl. 2001), hängt thematisch mit dem Studienbrief von Engbring zusammen, in dem die Unterschiede in der Verarbeitung von Informationen bzw. Daten zwischen Mensch und Computer hervorgehoben werden (Vgl. o.J.: S. 5-9). Daher ist es als

Vorbereitung auf die Bearbeitung des Textes von Engbring sinnvoll, die Frage nach der Intelligenz von Computern zu erörtern.

Zusammenfassend profitieren Studierende für ein Verständnis der beiden Studienbriefe davon, wenn der Studientext folgende Fragen und Aspekte behandelt:

- Welche Arten von Aufgaben werden durch Programme gelöst? Was ist ein Programm? Was ist eine Eingabe? Was ist Programmcode?
  Besonderer Aspekt: Programme steuern die Hardware und Programmcode hat eine absolute Semantik.

- Was sind Maschinensprache und höhere Programmiersprache und was sind die Besonderheiten?
  Besonderer Aspekt: Die Übersetzung zwischen den Ebenen läuft deterministisch ab.

- Wie ist Speicher aufgebaut?
  Besonderer Aspekt: Speicher arbeitet systematisch, das Gehirn assoziativ und man kann Speicher zurücksetzen.

- Wie werden Daten und Programme binär repräsentiert?
  Besonderer Aspekt: Die Binärcodes haben an sich keine Bedeutung, sie erhalten diese erst durch eine Interpretation.

- Wie werden (binäre) Programme durch den Prozessor verarbeitet?
  Besonderer Aspekt: Die Befehle werden in Steuersignale umgesetzt und nicht nach Wortbedeutung verarbeitet. Die Verarbeitung ist deterministisch.

- (kurz) Was versteht man unter *künstlicher Intelligenz*? Sind Computer intelligent?
  Besonderer Aspekt: Die Arbeitsweise von Gehirn und Computer ist unterschiedlich.

Um die Themen in eine sinnvolle Reihenfolge zu bringen, ist zu beachten, wie sie aufeinander aufbauen. Im nächsten Abschnitt werden daher die herausgearbeiteten Themen in eine inhaltlich sinnvolle Reihenfolge gebracht.

2.3. Reihenfolge der Inhalte

Einige der oben genannten Themen bauen aufeinander auf. Die Zusammenhänge werden hier zunächst in einem Diagramm dargestellt und danach näher erläutert. Im

Diagramm zeigen Pfeile an, welche T hemen für das Verständnis eines anderen Themas notwendig sind. Eine gestrichelte Linie bedeut et, dass die so verbundenen Themen einen inhaltlichen Zusammenhang haben. Teilweise ist angemerkt, worin die Notwendigkeit eines Themas für ein anderes bzw. der Zusammenhang zwischen zwei Themen besteht.

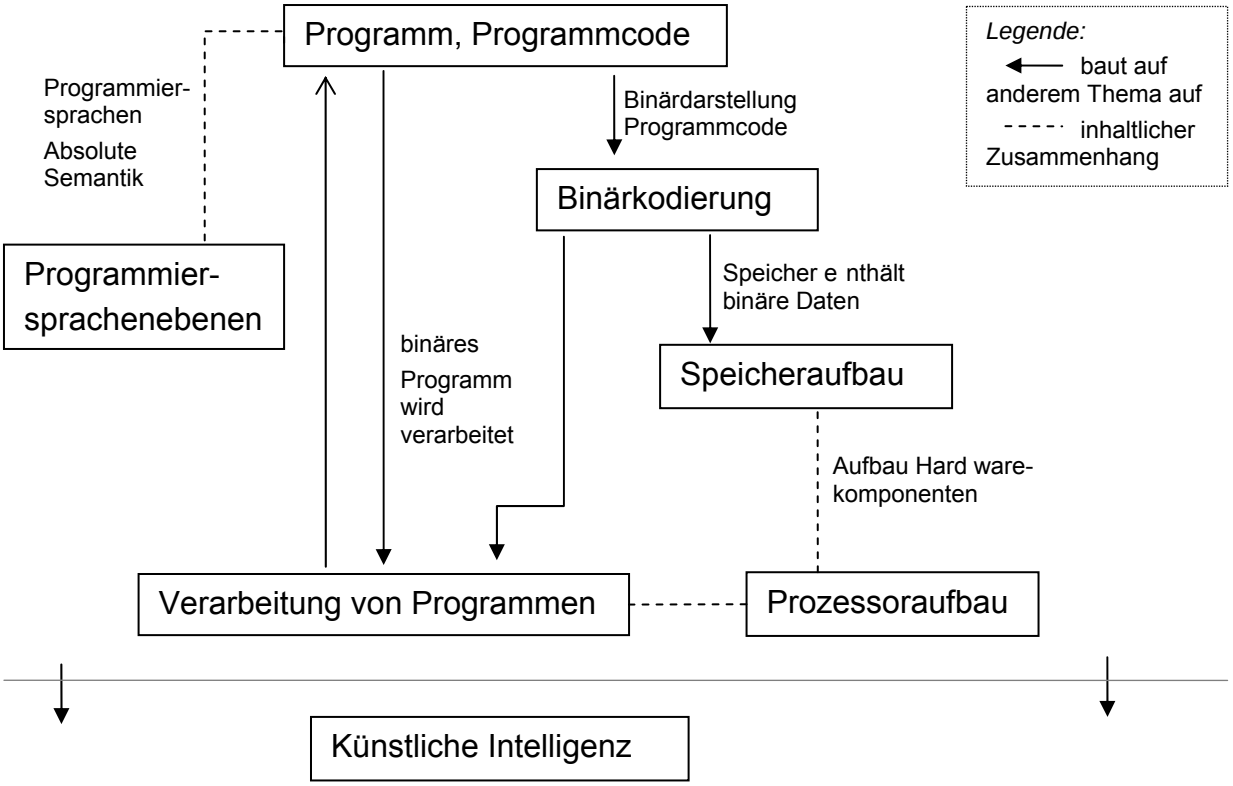


Abb. 0 Zusammenhang der herausgearbeiteten Themen

Die verschiedenen Pr ogrammiersprachebenen hängen eng mit der Frage zusammen, was ein Programm ist, da man zur Darstellung eines Programms auf jeden Fall eine Progr ammiersprache wählen muss. Außer dem trifft die absolute Sem antik von Programmiersprachen im Allgemeinen natür lich auch auf die verschiedenen Arten von Programmiersprachen zu.

Für die Erläuterung der Funktionsweise eine s Speichers ist es vorteilhaft, sc hon zu wissen, wie Daten binär repr äsentiert werden, da de r Speicher diese binär en Daten enthält. Der Aufbau v om Speicher hängt außer dem thematisch mit der Verarbeitung von Programmen zusammen, da dafür die F unktionsweise des Prozessors erklärt werden muss. Es geht also in beiden Fäll en um die F unktionsweise von Hardware-komponenten. Daher werden diese beiden Themen in einem Themenkomplex, der den Aufbau eines Computers insgesamt behandelt, zusammengefasst.

Das Thema, das das meiste Vorwissen erfordert, ist die Verarbeitung von Programmen durch den Prozessor. Denn dafür muss man wissen, was ein Programm ist, wie es in seiner binären Form aussieht und das s die Programme im Speicher enthalten sind. Um die binäre Repräsentierung von Daten und Programmen zu verstehen, ist es empfehlenswert, vorher zu wissen, wie Maschinencode aussieht. Diese Zusammenhänge legen es nahe, zuerst den Themenkomplex der Programme zu erläutern, danach die binäre Repräsentierung von Daten und Programmen und am Ende den Aufbau eines Computers und die Verarbeitung von Programmen durch den Prozessor.

Den Bereich der künstlichen Intelligenz und die Frage nach der Intelligenz von Computern zu skizzieren ist einfacher, wenn die restlichen Inhalte schon bekannt sind. Daher ist dieses Thema an das Ende des Studientextes gestellt.

Daraus ergibt sich insgesamt diese Themenreihenfolge:

1. Programme, Programmcode

   Verschiedene Arten von Programmiersprachen
2. Binärkodierung
3. Aufbau des Computers

   Speicher

   Prozessor
4. Programmverarbeitung
5. Künstliche Intelligenz (kurz)

Es gibt aber noch eine Schwierigkeit bei dieser Reihenfolge: Um Fachfremden zu erklären, wieso Maschinenprogramme auf die Art und Weise geschrieben werden, wie man sie schreibt, ist es wichtig zu wissen, welche Funktionen der Prozessor bereitstellt. Dies ist deshalb essentiell, weil die Maschinensprachbefehle vom Aufbau des Prozessors abhängig sind.

Um dieses Problem zu lösen, wird vor dem Kapitel über Programme kurz erläutert, welche Funktionen der Prozessor bereitstellt. Diese Erklärung ist in das Einführungskapitel, das im nächsten Abschnitt zur Sprache kommt, eingebettet.

2.4. Ideen zur didaktischen Gestaltung

In den letzten beiden Abschnitten wurden die fachlichen Inhalte des Studientextes anhand der Studienbriefe herausgearbeitet und in eine Reihenfolge gebracht. In die-

sem Abschnitt werden didaktische Überlegungen zur Vermittlung dieser Themen angestellt.

Der Studientext hat das Ziel, die oben beschriebenen Inhalte fachfremden Studierenden zu vermitteln. Dafür ist zum einen eine verständliche Erklärung wichtig, die das Aufnehmen der Inhalte ermöglicht und zum anderen die Festigung des so erlernten Wissens. Außerdem wird das Lernen erleichtert, wenn Motivation dazu vorhanden ist. Daher enthält der Studientext auch motivierende Elemente. Nun wird beschrieben, welche Möglichkeiten bei der Gestaltung des Studientextes wahrgenommen werden, um verständliche Erklärungen, Festigung des Wissens und Motivation zu erreichen.

Um den Studientext möglichst verständlich zu gestalten, ist es notwendig, die Erklärungen an den Wissensstand der Zielgruppe anzupassen. Da es sich bei den LeserInnen um Fachfremde handelt, bedeutet dies, die Inhalte auf sehr niedrigem Schwierigkeitsniveau darzustellen. Dabei ist zu beachten, dass die Studierenden sich bei der Bearbeitung des Studientextes vor allem zu Beginn in die für sie neuen Themen und Begriffe einfinden müssen. Daher ist ein Einstieg in den Studientext angebracht, der nicht zu komplex ist und einen Bezug zum Vorwissen der Studierenden herstellt. Es bietet sich an, den Studientext mit einem Kapitel zu beginnen, das in das Thema Computer einführt. So ist es möglich, von dem ausgehend, was aus dem alltäglichen Umgang mit Computern bekannt ist, zur tiefergehenden Sicht, die im Studientext vermittelt werden soll, überzugehen. Dies sorgt gleichzeitig dafür, dass ein bestimmtes Grundwissen aufgebaut wird, auf das in den anderen Kapiteln des Studientextes zurückgegriffen werden kann. Wie im letzten Abschnitt 2.3 erwähnt, erläutert das Einführungskapitel unter anderem, welche Funktionen der Prozessor innehat, damit im Kapitel über Programme darauf aufgebaut werden kann.

Für die Verständlichkeit ist außerdem wichtig, Begriffe aus der Informatik nicht ohne eine Erklärung ihrer Bedeutung zu verwenden. Dies wird im gesamten Studientext so gehandhabt.

Des Weiteren ist ein roter Faden im Studientext erforderlich, so dass den LeserInnen klar ist, an welchem Punkt der Erklärungen sie sich gerade befinden. Dies geschieht durch Überleitungen zwischen den verschiedenen Kapiteln.

Für das Verständnis und die Festigung der Inhalte ist ein rundes Gesamtbild der Funktionsweise von Computern nützlich. Natürlich ist den LeserInnen klar, dass die Funktionsweise auf einer anderen Ebene genauer erklärt werden kann. Um auf der

einfachen Ebene des Studientextes Lücken in der Erklärung zu vermeiden, wird die Erklärung des Aufbaus von Speicher und Prozessor im vierten Kapitel in den Gesamtaufbau des Computers eingebunden.

Außerdem ist für die Vermittlung der Inhalte relevant, dass Studierende nicht nur Vorwissen mitbringen, das ihnen dabei hilft, die Inhalte zu verstehen. Ihr Zugang zu Computern war bisher der von AnwenderInnen. Die Vorstellung, die sie von Computern haben, ist aus dem Umgang mit Computern entstanden. Es könnte zu Irritationen kommen, wenn Erklärungen im Studientext den Vorstellungen der Studierenden widersprechen. Auch bestimmte Begriffe sind bei AnwenderInnen anders konnotiert als bei InformatikerInnen, z.B. der Begriff *Programm*. Daher wird bei der Gestaltung aller Kapitel darauf geachtet, die schon bestehenden Vorstellungen aufzugreifen und mit dem Inhalt des Kapitels in Zusammenhang zu bringen.

Das Einbinden des Vorwissens und der alltagsnahen Vorstellungen über Computer ist auch für die Festigung des Wissens sehr wichtig. Wenn das Erlernte auch mit dem Vorwissen und den Alltagserfahrungen im Umgang mit Computern in Einklang gebracht wird, werden die Studierenden es sich besser merken können. Außerdem werden sie möglicherweise im alltäglichen Umgang mit Computern an die Erklärungen denken und sie sich so wieder ins Gedächtnis rufen. Es ist demnach vorteilhaft, die neu erlernten Inhalte mit den alltäglichen Vorgängen am Computer in Verbindung zu bringen. Dies erfolgt zum einen innerhalb der einzelnen Kapitel, um einen Bezug zwischen dem jeweiligen Thema und dem Umgang mit Computern im Alltag herzustellen. Zum anderen ist es zielführend, wenn die Themen in ihrer Gesamtheit auch in Beziehung zu alltäglichen Vorgängen gesetzt werden. Zu diesem Zweck hat der Studientext analog zum Einführungskapitel auch ein abschließendes Kapitel, in welchem die neu erlernten Inhalte in einen alltäglichen Vorgang am Computer eingebunden werden.

Um die Motivation der LeserInnen hochzuhalten, ist der Studientext möglichst lesbar gestaltet. Dies beinhaltet, eine verständliche Sprache zu verwenden und Quellenangaben nicht innerhalb des Studientextes, sondern an dessen Ende aufzuführen. Bei den Quellenangaben wird außerdem Literatur zum Weiterlesen empfohlen, damit die Studierenden sich bei Interesse noch weiterbilden können.

Um die Motivation während des Lesens zu erhöhen, werden Erklärungen hin und wieder durch Informationen aus anderen Wissensbereichen aufgelockert, z.B. durch geschichtliche Hintergründe. Dabei wird natürlich immer eine Verbindung zum ei-

gentlichen Thema hergestellt. Besonders zu Beginn des Studientextes ist es wichtig, Interesse zu wecken, damit überhaupt Motivation besteht, den Studientext zu bearbeiten. Dies wird erreicht, indem in einer Einleitung deutlich gemacht wird, wieso der Inhalt des Studientextes ein für die LeserInnen relevantes Thema ist. Um in der Einleitung einen Bezug zum restlichen Studientext herzustellen, wird eine der Quintessenzen des Gesamttextes mit eingebunden.

Auch die Kapitelüberschriften sind so gestaltet, dass sie zum Weiterlesen animieren. Daher trägt zum Beispiel das erste Kapitel statt *Einführung* den *Namen Einführung in die Computerwelt* und das dritte Kapitel heißt nicht *Binärkodierung*, sondern *Binärkodierung - Von Nullen und Einsen*.

Um schon vor der Lektüre des Studientextes die Studierenden für die Inhalte zu interessieren, ist der Titel so gewählt, dass Fachfremde direkt verstehen können, worum es geht. Würde der Titel *Rechneraufbau und Programmverarbeitung* heißen, wäre dies etwas, womit Fachfremde zunächst nicht viel anfangen könnten.

Zur Festigung des Wissens, zum Verständnis der Inhalte und zur Motivierung sind Analogien zum Alltag und Grafiken gut geeignet. In Analogien zum Alltag können Vorgänge in der Informatik durch Vorgänge, die aus dem Alltag bekannt sind, veranschaulicht werden. Dadurch wird das Wissen besser aufgenommen werden und es wirkt auch als Auflockerung für den Studientext. Auch Grafiken bieten eine Abwechslung im Textfluss und machen die Inhalte leichter verständlich, da dies e in einer anderen Struktur dargestellt werden. Außerdem kann sich Wissen besser festigen, wenn man dazu eine Analogie aus dem Alltag oder eine Grafik in Erinnerung hat. Daher enthält der Studientext mehrere Grafiken und Analogien.

Des Weiteren sind im Studientext nach jedem Kapitel Aufgaben enthalten, die die Studierenden optional bearbeiten können. Dadurch kann das Verständnis der Inhalte vertieft und gefestigt werden. Außerdem können die Studierenden so nach jedem Kapitel überprüfen, ob sie die Themen verstanden haben und bei Bedarf genauer nachlesen. Die Aufgabenlösungen sind dieser Arbeit im Anhang I beigefügt. Die Form der Aufgaben I, II und XII entstammt *Informatik bis zum Abitur* (Engelmann, 2002): Dabei sind für Aufgabe II die Form der Aufgabe und Teile der Formulierung der Aufgabenstellung aus der Aufgabe 1.3.1 des Buches übernommen (Vgl. S.91) und bei den Aufgaben I und XII stammt die Form der Aufgabe aus der Aufgabe 4.1.5 des Buches (Vgl. S.392).

2.5. Gestaltung der einzelnen Kapitel

Im Folgenden wird die Gestaltung der einzelnen Kapitel des Studientextes erläutert. Dies erfolgt anhand der in der bisherigen Entwicklung des Studientextes herausgearbeiteten fachlichen Inhalte und Zielsetzungen des jeweiligen Kapitels und anhand der zuvor beschriebenen Gesichtspunkte zur didaktischen Gestaltung.

In der folgenden Tabelle sind die Inhalte bzw. die Zielsetzung der einzelnen Kapitel aufgelistet. Die Inhalte der Kapitel 2-4 und 6 sind in Abschnitt 2.2 schon anhand der Vorlesung IuG herausgearbeitet worden, während das Einführungskapitel, die Einleitung und das Kapitel 5 sich erst im Rahmen der didaktischen Gestaltungsideen in Abschnitt 2.4 entwickelt haben.

Kapitel und Thema	**Inhalte / Zielsetzung**	**Besonderer Gesichtspunkt**
Einleitung Die	Relevanz der Inhalte des Studientextes hervorheben.	Eine Quintessenz des Studientextes einbinden.
Kapitel 1: **Einführung**	Einführung ist Thema des Studientextes. Vom Vorwissen aus dem Alltag ausgehen.	Welche Funktionen stellt der Prozessor bereit?
Kapitel 2: **Programme**	Welche Arten von Aufgaben werden durch Programme gelöst? Was sind Programm, Eingabe und Programmcode?	Programmcode hat eine absolute Semantik.
	Was sind Maschinensprache und höhere Programmiersprache und ihre Besonderheiten?	Die Übersetzung zwischen den Ebenen läuft deterministisch ab.
Kapitel 3: **Binärkodierung**	Wie werden Daten und Programme binär repräsentiert?	Binärcodes haben an sich keine Bedeutung, sie erhalten diese erst durch eine Interpretation.
Kapitel 4: **Aufbau eines Computers** - **Speicher** - **Prozessor**	Wie ist Speicher aufgebaut? Wie werden (binäre) Programme durch den Prozessor verarbeitet?	Speicher arbeitet systematisch, das Gehirn assoziativ. Die Programmbefehle werden in Steuersignale umgesetzt und nicht nach Wortbedeutung ver-

und Programm-verarbeitung	Gesamtaufbau des Computers	arbeitet. Die Verarbeitung ist deterministisch.
Kapitel 5: **Verbindung zum Alltag**	Zusammenhang zwischen den Inhalten des Studientextes und einem alltäglichen Vorgang am Computer herstellen	Das neu Erlernte in das vorher Bekannte integrieren.
Kapitel 6: **Künstliche Intelligenz**	Was versteht man unter *künstlicher Intelligenz*? Sind Computer intelligent?	Die Arbeitsweise von Gehirn und Computer ist unterschiedlich.

Tab. 0 Inhalte bzw. Zielsetzung der einzelnen Kapitel des Studientextes

2.5.1. Einleitung

Die Aufgabe der Einleitung ist zum einen, die Studierenden zur Lektüre des Studientextes zu motivieren und zum anderen, eine der Quintessenzen des Studientextes zur Sprache zu bringen.

Zur Motivation wird die Relevanz der Informationstechnologie in unserer Gesellschaft durch den Hinweis auf aktuelle Anwendungsbereiche und durch den Vergleich mit der industriellen Revolution aufgezeigt. Innerhalb des Vergleichs wird einer der Unterschiede zwischen Menschen und Computer herausgestellt, nämlich, dass Computer nur Aufgaben lösen können, die durch einen Algorithmus darstellbar sind, andererseits aber sehr schnell arbeiten. (Vgl. Goldschlager, et al., 1990: S.1)

Da bei den LeserInnen ein Verständnis von Begriffen wie *Algorithmus* oder *Computerchip* nicht vorausgesetzt werden kann, werden die dahinter liegenden Gedanken in einfachen Worten angerissen. Durch die Übertragung der Eigenschaften der industriellen Revolution auf die Informationstechnologie soll ein Eindruck davon entstehen, dass Computer im Gegensatz zu Menschen andere Möglichkeiten der Aufgabenlösung haben.

2.5.2. Kapitel 1: Einführung in die Computerwelt

In diesem Kapitel wird in das Thema des Studientextes eingeführt. Da im Studientext der Aufbau und die Funktionsweise von Computern und die Verarbeitung von Programmen zentrale Themen sind, ist es sinnvoll, kurz den Aufbau des Computers und den Zusammenhang zwischen Hard- und Software zu erläutern. Damit im nächsten

Kapitel darauf aufgebaut werden kann, ist eine Erklärung der Grundfunktionen des Prozessors enthalten.

Für den Einstieg in das Thema *Wie funktionieren Computer?* ist das Vorwissen der Studierenden relevant. Aus dem Alltag kennen die Studierenden PCs und Laptops. Sie schalten den Computer an, benutzen die Anwendungsprogramme und die Ein- und Ausgabegeräte und speichern und öffnen Dateien.

Am Anfang des Kapitels wird daher alltagsnah anhand der Ein- und Ausgabegeräte das Konzept der Datenverarbeitung Eingabe- Verarbeitung-Ausgabe erklärt. Davon ausgehend könnten nun entweder die Komponenten oder weitere Funktionsprinzipien des Computers erklärt werden. Da Fachfremde einen größeren Bezug zu den Vorgängen der Speicherung und Dateneingabe haben als zu den zugehörigen Geräten, ist es sinnvoll, zunächst die Funktionsprinzipien in den Vordergrund zu stellen. (Vgl. Gookin, 2010: S.32)

Anhand dieser Prinzipien kann die nun folgende Erklärung des Aufbaus des Computers im Studientext viel einfacher verstanden werden. Zur Anschaulichkeit ist der Aufbau des Computers als Diagramm abgebildet. Im Verlauf des Studientextes wird das Diagramm wiederverwendet und erweitert. So wird das neu Erlernte auch bildlich in das schon zuvor im Studientext erlernte Wissen integriert.

Als nächstes wird der Unterschied zwischen Hardware und Software dargelegt. Dabei ist zu beachten, dass die Studierenden unter einem Programm ein Anwendungsprogramm verstehen. Für ein Verständnis der Funktionsweise von Computern ist wichtig, dass der komplette Computer durch Programme gesteuert wird. Daher wird an dieser Stelle zwischen Anwendungs- und Systemprogrammen differenziert. Für die noch unbekannten Systemprogramme werden Beispiele angeführt, die Fachfremde sich gut vorstellen können. (Vgl. Goldschlager et al., 1990: S.16f)

2.5.3. Kapitel 2: Programmierbarkeit – Dem Computer befehlen

Die Inhalte dieses Kapitels sind unter anderem, welche Aufgaben Computer lösen können und was Programme sind. Ein Computer kann alle Aufgaben erledigen, die als Algorithmus formuliert werden können. Das Thema der Programme baut auf diesem Thema auf, weil hinter jedem Programm ein Algorithmus steht. Daher steht die Erklärung über Algorithmen am Anfang des Kapitels.

Fachfremden ist nicht klar, wieso Aufgabenlösungen für den Computer in Form eines Algorithmus formuliert werden müssen. Das wird daran deutlich, dass die Aufgaben-

lösung in kleinen Schritten, die der Prozessor ausführen kann, formuliert werden muss. (Vgl. Goldschlager et al., 1990: S.11)

Da das Konzept von Algorithmen den LeserInnen fremd vorkommen mag, wird es anhand eines Beispiels einer Anleitung im Alltag anschaulicher gemacht - dem Backen eines Kuchens nach Rezept. Die Analogie wird auch im weiteren Verlauf des Kapitels verwendet. (Vgl. Harel, 2002: S.1ff)

Um Algorithmen von Anleitungen aus dem Alltag abzugrenzen, wird dargelegt, dass nur diejenigen Algorithmen durch Computer ausgeführt werden können, die mit Daten zu tun haben.

An dieser Stelle wird die Besonderheit der Universalität von Computern durch einen Vergleich mit anderen Maschinen, die eine wesentlich geringere Fülle von Funktionen bereitstellen, hervorgehoben. Als Beispiel werden sowohl die Maschinen vor dem Computerzeitalter als auch Maschinen erwähnt, die aktuell verwendet werden. (Vgl. Rechenberg, 2000: S.13f)

Vom Aufbau des Kapitels her ist nun ein Übergang vom Algorithmus zum Programm notwendig. Dieser besteht inhaltlich darin, dass das Programm den Algorithmus in eine Form bringt, die der Computer bearbeiten kann (Vgl. Goldschlager et al., 1990: S.16). Diese Verbindung wird erkennbar, indem sie in die Erklärung des Entstehungsprozesses eines Programms eingebettet wird. Für eine bessere Anschaulichkeit wird der Entstehungsprozess in einer Grafik abgebildet (Vgl. Harel, 2010: S.65). Diese wird im Studientext später erweitert, wenn dem Entstehungsprozess inhaltlich etwas hinzugefügt wird.

Bei der Erläuterung von Programmen ist zu beachten, dass die LeserInnen noch nie mit Programmcode zu tun hatten. Es ist sinnvoll, entweder Maschinencode oder höhere Programmiersprache zu verwenden, da ständiges Wechseln zwischen den Programmiersprachebenen verwirren könnte. Da es im vierten Kapitel des Studientextes bei der Verarbeitung der Programme durch den Prozessor um Maschinencode geht, wird auch hier Maschinensprache verwendet. Die Besonderheiten von höheren Programmiersprachen werden erklärt, ohne einen größeren Schwerpunkt auf Programmcode in höherer Programmiersprache zu legen.

Die Vermittlung tiefergehender Programmierkenntnisse ist im Studientext weder notwendig noch möglich. Es ist jedoch wichtig, einen Eindruck davon zu geben, was Programmcode ist. Der Verständlichkeit halber wird dafür ein einfaches Beispiel

ausführlich erklärt. Für diesen Zweck bietet sich eine kleine Berechnung an. Hier ist als Beispiel die Berechnung des Mittelwerts zweier Zahlen gewählt. Fachfremde könnte es irritieren, dass der Maschinencode so klein schrittig programmiert werden muss, da sie die genaue Funktionsweise eines Prozessors noch nicht kennen. Daher wird diese Tatsache kurz erklärt. Außerdem wird an dieser Stelle zwar Maschinencode abgebildet, aber es wird mit Variablen statt mit Speicheradressen gearbeitet, da die LeserInnen die Organisation des Speichers noch nicht kennen gelernt haben.

In Abschnitt 2.5.5 wird erläutert, wieso die hier verwendete Form der Maschinensprache bzw. die dahinterliegende Rechnerarchitektur gewählt wird.

Ein Ziel des Studientextes ist es zu vermitteln, dass der Computer den Programmcode deterministisch verarbeitet. Das damit zusammenhängende Risiko, fehlerhaften Code zu befolgen, ist für Fachfremde nicht offensichtlich. Diese Inhalte können anhand der in diesem Kapitel verwendeten Analogie zum Kuchenbacken sehr gut verdeutlicht werden und werden daher schon an dieser Stelle erläutert und nicht erst im vierten Kapitel.

Die Abbildung von Programmcode ist im Studientext auf ein sehr einfaches Programmbeispiel beschränkt. Da die Studierenden im Alltag wesentlich komplexere Programme benutzen, wird im weiteren Verlauf erklärt, worin der Zusammenhang besteht. Hinter heutigen Programmen steckt natürlich viel mehr als hinter einem kurzen Stück Programmcode. Zum Beispiel werden Programme objektorientiert mit komplexen Datenstrukturen und Datenbanken in höheren Programmiersprachen programmiert und es gibt ein Betriebssystem, das anderen Programmen Dienste zur Verfügung stellt. All das hat mit den Schwerpunkten des Studientextes nicht viel zu tun und würde auch dessen Rahmen sprengen. Doch eine Grundidee davon zu vermitteln ist wichtig, damit die LeserInnen die Verbindung zwischen dem Gelernten und dem, was sie aus dem Alltag kennen, schlagen können. Viele der heutigen Programmiertechniken verbindet die Möglichkeit, bestehende Programme in anderen Programmen zu verwenden. Dies steckt hinter dem Konzept von Prozeduren, Übersetzungsprogrammen und dem Betriebssystem.

Daher wird an dieser Stelle kurz erklärt, dass komplexe Programme aus vielen kleinen Programmen bestehen, Programme andere Programme verwenden können und so die Fülle von Programmen, auf die man aufbaut, immer größer wird. Auch hier ist für ein besseres Verständnis ein Beispiel angeführt.

Thematisch ist dies eine gute St elle, um höhere Programmiersp rachen einzuführen. Denn auc h diese tragen dazu bei, dass di e heutige Entwicklung von Programmen möglich ist. Die LeserInnen haben im Studientext schon Programmcode kennen gelernt, der aus Maschinensprache besteht, aber nicht unter diesem Begriff. Außerdem ist in den Studientext noch nicht eingegang en, dass es Programmiersprachen gibt, die nicht direkt vom P rozessor verarbeitet werden können. Auf Grund dieser Überlegungen wird ausgehend von den Besonderheiten der Maschinens prache die Notwendigkeit der Nutzung höherer Programmi ersprachen erklärt. Dabei wird direkt klargestellt, dass der Prozessor Programme in höher er Programmiersprache nicht verarbeiten kann. Da es inhaltlic h und l ogisch an dies e Stelle passt, wird dann der Vorgang der Übersetzung von höherer Programmiersprache zur Maschinensprache erläutert. Zum besseren Verständnis wir d das schon bekannt e Programm zur Berechnung des Mit telwerts in höherer Programmiersprache abgebildet. (Vgl. Goldschlager et al., 1990: S.16f)

Diesem Kapitel fehlt noch die Erklärung des Begriffs der *absoluten Semantik*. Da sowohl die Semantik der Maschinensprac he als auch die der höheren Programmiersprache mit dem Übersetzungsvorgang zu tun haben, wird die absolute Semantik anhand des Übersetzungsvorgangs erklärt. Mith ilfe dessen, was die LeserInnen bis her über Programme und Algorithmen gelernt haben, kann man ihnen verständlich machen, dass ein Pr ogramm, das übersetzen soll, klare Regeln für die Übersetzung benötigt. Davon ausgehend führt die Tatsache, dass die Übersetzung nur bei Befehlen erfolgen kann, die vordefiniert sind, zum Begriff der *absoluten Semantik*. Durch den Kontrast zur Übersetzung von menschlic her Sprache wird die absolute Semantik von der auf die Umwelt bezogenen Semant ik abgegr enzt. (Vgl. Harel, 2002: S.58 - 62)

Zur Auflockerung wir d an dieser Stelle erläutert, dass die Wörter der höheren Programmiersprache frei gewählt werden können, solange es ein Übersetzungsprogramm gibt. Als Beispiel wird die Programmiersprache Shakespeare genannt.

2.5.4. Kapitel 3: Binärkodierung - Von Nullen und Einsen

Binärkodierung ist für die LeserInnen der Zielgruppe ein unbekanntes Konzept. Daher wird in der Erklärung darauf verzicht et, reale Mec hanismen der Binärk odierung wie z.B. die Gleitkommadarstellung zu erl äutern. Stattdessen wird das Prinzip der

Binärkodierung, welches darin besteht eine Zuordnung zwischen einer Binärkombination und einem anderen Wert vorzunehmen, erklärt. (Vgl. Rechenberg, 2000: S.25)

Da auch Binärkombinationen den LeserInnen unbekannt sind, ist hier zunächst eine Erklärung notwendig, wie solche Kombinationen aufgebaut sind. Um vom Einfachen zum Komplexen hinzuführen, werden zunächst Binärkombinationen mit einer Stelle, dann mit drei und zuletzt mit acht Stellen betrachtet. Damit die Studierenden Bekanntes wiederfinden, werden in diesem Rahmen die Begriffe *Byte* und *Bit* eingeführt. Danach folgt die eigentliche Erklärung des Prinzips der Binärkodierung.

Die deterministische Verarbeitung im Computer wird besonders auf der elektronischen Ebene deutlich. Dies wird in Abschnitt 2.5.5 genauer begründet. In diesem dritten Kapitel des Studientextes wird daher die elektronische Repräsentation binärer Daten im Computer beschrieben. Es wird in der Beschreibung mit der Vorstellung von Schaltern und Strom gearbeitet, die den Studierenden aus dem Physikunterricht in der Schule noch präsent sein dürfte. (Vgl. Lange, 2004: S.27)

Im späteren Verlauf der Arbeit wird beschrieben, dass der Prozessor rechnet und Maschinenbefehle interpretiert. Wichtig ist es dafür, an dieser Stelle zum einen zu verdeutlichen, wie es möglich ist, mit binären Zahlen zu rechnen und zum anderen, dass Maschinenbefehle binär kodiert werden. Für die Fachfremden ist es nicht wichtig, über Details wie Opcodes und Operanden Bescheid zu wissen; das Verständnis des Grundkonzeptes steht im Vordergrund. Dies wird auf vereinfachte Weise dargestellt, indem jedem Maschinenbefehl eine binäre Zahl zugeordnet wird (Vgl. Rechenberg, 2000: S.36). Aus der Schule sollte den Studierenden das Konzept verschiedener Zahlensysteme bekannt sein. Zur Auffrischung wird die Addition von Binärzahlen auf einer einfachen Ebene erläutert, der Addition der einzelnen Ziffern (Vgl. Hattenhauer, 2010: S.41).

Für die Gesamtdarstellung der Binärkodierung muss noch die binäre Kodierung von Daten wie Bildern, Videos, Musik, etc. erwähnt werden. Als Beispiel wird die Binärkodierung eines Bildes erklärt, weil dies besonders anschaulich darstellbar ist. Dies wird anhand einer Grafik aus dem Buch *Was ist Informatik?* (Vgl. Rechenberg, 2000: S.28) dargestellt. Auch zu den anderen zuvor beschriebenen Erklärungen werden im Studientext Beispiele angeführt.

2.5.5. Kapitel 4: Wie die Hardware arbeitet

Die Zielsetzung dieses Kapitels ist es, die Arbeitsweise von Prozessor und Speicher zu erklären und in den Gesamtaufbau des Computers einzuordnen. Dabei ist hervorzuheben, dass der Speicher systematisch arbeitet und Programme deterministisch verarbeitet werden.

Für die Einordnung in den Gesamtzusammenhang wird zu Beginn des Kapitels die Erklärung des Computeraufbaus aus dem ersten Kapitel in Erinnerung gerufen.

Insgesamt ist in diesem Kapitel die Frage besonders wichtig, auf welcher Ebene der Aufbau der Hardware erklärt wird. Die Informatik betrachtet den Rechneraufbau auf verschiedenen Ebenen – z.B. auf der physikalischen, der digitalen oder der Mikroebene – und so entsteht am Ende ein Gesamtbild. Es ist abzuwägen, welche der Inhalte, die in der Informatik behandelt werden, für Fachfremde im Zusammenhang mit der Aufgabenstellung dieses Kapitels und somit für das Verständnis der Vorlesung IuG relevant sind.

Des Weiteren ist zu beachten, dass der Rechneraufbau und die Programmierung des Computers eng zusammenhängen. Denn die Befehle der Maschinensprache hängen von dem Aufbau des Computers ab. Die Maschinensprache, die im zweiten Kapitel verwendet wird, ergibt sich also daraus, wie der Computeraufbau in diesem Kapitel erläutert wird. Auch dies ist bei der Gestaltung miteinzubeziehen.

Die Determiniertheit der Verarbeitung ist auf der digitalen bzw. elektronischen Ebene des Computers besonders gut erkennbar, da es einleuchtend ist, dass elektronische Schaltkreise so auf Signale reagieren, wie sie gebaut wurden. Aus dem Schulunterricht bringen die Studierenden schon eine Vorstellung davon mit, was elektronische Schaltkreise sind und dass diese aufgrund physikalischer Gesetzmäßigkeiten nach festen Regeln arbeiten. Darauf aufbauend wird die Determiniertheit herausgestellt, indem ein Bezug zur elektronischen Ebene hergestellt und darauf hingewiesen wird, dass die Verarbeitung elektronisch erfolgt. Es ist für die Zielsetzung des Kapitels nicht notwendig, detailliert zu erklären, wie die Schaltkreise aufgebaut sind.

Insgesamt wird also in diesem Kapitel eine einfache Beschreibungsebene gewählt und der Bezug zum zweiten Kapitel berücksichtigt. Außerdem ist die elektronische Ebene auf einem Niveau enthalten, das gerade ausreicht, um hervorzuheben, dass die Verarbeitung elektronisch erfolgt.

Als Basis für die Erklärung der Elektronik in Speicher und Prozessor wird zu Beginn dieses vierten Kapitels kurz beschrieben, dass die Komponenten des Computers aus Schaltkreisen bestehen. Da die Begriffe der Elektronik wie *Schalter* und *Leitung* den LeserInnen durch den Schulunterricht geläufig sind, werden sie hier ohne detaillierte Erklärung verwendet. Auch der Begriff des *Chips* wird hier verwendet, da die Studierenden ihn wahrscheinlich schon gehört haben und es sie motivieren kann, nun zu erfahren, was ein Chip ist. Der Anschaulichkeit halber werden diese Inhalte in eine Erläuterung ihrer geschichtlichen Entwicklung eingebettet. (Vgl. Precht et al., 2004: S.58)

Als nächstes folgt die Erklärung der Arbeitsweise des Speichers. Dabei ist zu beachten, dass die Studierenden das Wort *speichern* aus dem Zusammenhang kennen, dass sie selbst Dokumente speichern. Der hier relevante Speicher, der auch bisher im Studientext unter dem Begriff *Speicher* gemeint war, ist der, mit dem die CPU arbeitet. Den Begriff *Arbeitsspeicher* haben die Studierenden möglicherweise schon einmal gehört, bringen ihn aber nicht unbedingt mit dem hier verwendeten Begriff von Speicher in Verbindung. Dieses zwar vorhandene, aber unsortierte Vorwissen kann verwendet werden, um deutlich zu machen, dass der Begriff des *Speichers* ein anderer ist als der Alltagsbegriff *speichern*. Aus diesen Gründen ist an dieser Stelle eine kurze Textpassage eingefügt, die den Unterschied zwischen dem Arbeitsspeicher und dem permanenten Speicher erläutert (Vgl. Patterson et al., 2005: S.21). Für die Anschaulichkeit ist eine Analogie zur Alltagswelt gewählt, die zu der Analogie zur Erklärung des Prozessors passt, auf die später noch eingegangen wird.

Ein im Zusammenhang mit der Funktionsweise des Speichers fachlich wichtiges Konzept ist das Von-Neumann- Prinzip, nach dem Daten und Programme im Speicher stehen. Dies ist auch für die später folgende Erklärung der Verarbeitung von Programmen relevant. Da das Von-Neumann-Prinzip auch bei der Abgrenzung von Arbeitsspeicher und permanentem Speicher hilfreich ist, wird es direkt zu Beginn der Erklärung von Speicher vorgestellt. Für die Anschaulichkeit werden als Kontrast zum Programmspeicher Lochkarten vorgestellt (Vgl. Precht et al., 2004: S.56).

Es wird also, um die Vorstellungen der Studierenden mit einzubeziehen, zunächst erläutert, dass auch Programme im Speicher stehen. Dann wird der Arbeitsspeicher vom permanenten Speicher abgegrenzt. Nun folgt der Hauptpunkt der Erläuterungen über Speicher, nämlich wie ein Speicher organisiert ist und dass diese Organisation im Gegensatz zum Gehirn sehr systematisch ist. Der Speicher ist insofern systema-

tisch organisiert, als er in Speicherzell en unterteilt ist und diese Speicher zellen nur über die Adresse angespro chen werden können. Auf De tails, wie die Technik verschiedener Speicherart en, wird verzichtet, da sie für ein Grundverständnis des Speichers nicht notwendig sind. Der Aufbau des Speichers wird mit einer Grafik illustriert. Dass die Inhalte nur über die Ad resse erreichbar sind, wird anhand einer Analogie zu Briefkästen verdeutlicht. (Vgl. Rechenberg, 2000: S.35)

Um die Verbindung z ur elektronischen Ebene her zustellen, wir d direkt zu Begin n darauf eingegangen, dass der S peicher aus Schaltern besteht. Zudem werden in der Grafik zur Erläuterung des Speicheraufbaus die Speicherinhalte als Nullen und Einsen dargestellt.

Danach wird das Konzept von Daten- und Adressleitungen anhand der Adressierung des Speichers erklärt. Dies verdeutlicht zum einen, inwiefern der Speicher anders als das Gedächtnis arbeitet. Zum anderen zeigt es, wie Pr ogrammbefehle elektronisch verarbeitet werden k önnen. Denn die Verar beitung ist durch die Über tragung der Signale und deren Umwandlung in andere Signale mö glich. Auch hier ist eine erklärende Grafik eingefügt.

Um sicherzustellen, dass die Studierenden die Funk tionsweise des Speic hers der des Gedächtnisses gegenüberstellen, wird dieser Unterschied am Ende noch einmal explizit hervorgehoben.

Als nächstes folgt die Erläuterung der Arbe itsweise des Prozess ors. Wie zuvor erläutert, ist dabei die Verbindung zwischen Programmierung und Hardware wichtig. Es soll deutlich werden, wie ein Programm befehl die jeweiligen Hardwarekomponenten steuert. Dazu gehören fachlich die Trennung in Steuerwerk und Rechenwerk, die Fetch-Decode-Execute-Schleife[9] und die Tatsache, dass der Prozessor im Execute-Teil je nach Befehl Daten speichert, lädt oder Berechnungen durchführt.

Zu Beginn wird erläutert, welche Aufgaben Steuerwerk und Rechenwerk im Prozessor wahrnehmen. Da dies für die Leserl nnen ein neues Konzept ist, wird es mit den Vorgängen einer Berechnung an einem Schr eibtisch verglichen (Vgl. Rechenberg, 2000: S.42). Darauf folgt die Erklärung der Arbeitsweis e von Steuerwerk und Rechenwerk. Es wird dabei auf die Behandlung von Registern, wie Befehlsregister und Befehlszähler, Daten- und Ad ressregister verzichtet. Die s wäre ein weiteres neues Konzept, dass die Studierenden sich aneig nen müssten und es ist für ein Verständ-

[9] Fetch-Decode-Execute wird im Folgenden mit FDE abgekürzt.

nis der Arbeitsweise des Proz essors nic ht unbedingt notwend ig. Die FDE-Schleife wird also ohne die dafür notwendigen Register beschrieben.

Sprungbefehle und bedingte Spr ungbefehle werden sowohl hier als auch im restlichen Studientext auß en vor gelassen. Das hat den Vorteil, dass das Konzept der Verzweigung und die daraus folgenden Mög lichkeiten in der Programmierung nicht vorgestellt werden. So kann der Fokus auf die für die Aufgabenstellung des Studientextes wichtige Verarbeitung von Pr ogrammen und deren Determiniertheit gelegt werden. Dies ist auf der anderen Seite auc h ein Nachteil, da di e LeserInnen die genaueren Möglic hkeiten der Steuerung durch Programmierung nicht kennenlernen. Dieser Nac hteil wird aber zuguns ten der Schwerpunktlegung auf die Programmverarbeitung in Kauf genommen.

In Bezug auf die elektronische Ebene wir d in der Beschreibung des Steuerwerks erläutert, d ass die Befehle als elektronische Signale da s Steuerwerk erreichen und auch als solche verarbeitet werden. Bei der Erklärung wird die Determiniertheit in der Umsetzung der Befehle hervorgehobe n, jedoch nic ht das W ort *Determiniertheit* verwendet. Der Ansc haulichkeit halber wir d der Dek odierungsvorgang dur ch eine Grafik illustriert.

Es könnte die LeserInnen irritieren, wenn der Vorgang der Dekodi erung sehr detailliert und die FDE-Schleife sehr ungenau er klärt wird, obwohl be ides Aufgaben des Steuerwerks sind. Daher wird in der Besch reibung des Steuerwerks nur die Dekodierung und Ausführung eines ei nzelnen Bef ehls thematisie rt und die Befehlsschleife wird in einem eigenen Absatz behandelt. Diese r ist an den Absatz über die Arbeitsweise des Prozessors angesch lossen. Dies macht es möglich, die Dekodier ung und Ausführung eines einzelnen Bef ehls etwas detaillierter zu behandeln und v on der FDE-Schleife nur eine ungefähre Idee zu vermitteln.

In der Bes chreibung des Rechenwerks wir d, wie schon erwähnt, auf Daten- und Adressregister und verschiedene Rechenregister verzichtet. Stattdessen wird hier mit der Vorstellung gearbeitet, dass die Daten di rekt aus dem Speicher in das Rechenwerk kommen (Vgl. Rechenberg, 2000: S.36) und es wird nur mit einem Register, dem Akkumulator, gearbeitet. Dies ist nu r eine modellhafte Beschreibung der Vor gänge, aber für die Zwecke dieses Kapitels völlig ausreichend. Die elektronischen Vorgänge im Rechen werk werd en anhan d des Beispiels der Ad dition un d mit Hilfe einer Grafik dargestellt.

Am Ende folgt die Beschreibung der Verarbeitung eines Programms durch die FDE-Schleife, die, wie oben erklärt, von der Beschreibung des Steuerwerks getrennt ist. Da natürlich inhaltlich ein enger Zusammenhang besteht, werden die zuvor vermittelten Informationen über den Prozessor eingebunden. Zur Festigung des Wissens und um die Inhalte noch anschaulicher zu machen, wird ein Programm abgebildet, wie es im Speicher steht (Vgl. Rechenberg, 2000: S.59) und der Programmablauf durch eine Grafik illustriert.

Als Maßnahme zur Festigung des gesamten Wissens aus diesem vierten Kapitel wird an dessen Ende der Aufbau des Computers in einer erweiterten Grafik dargestellt.

2.5.6. Kapitel 5: Was passiert, wenn ich etwas eintippe?

In diesem fünften Kapitel wird das neu Erlernte in das vorher aus dem Alltag Bekannte integriert. Dabei werden die Inhalte aus allen bisherigen Kapiteln noch einmal angesprochen und in einen Vorgang am Computer, der alltäglich ist, eingebunden. Dafür macht es Sinn, einen ganz kleinen Vorgang zu nehmen, den man auch in einer kurzen Textpassage beschreiben kann.

Die in dem Studientext vorgestellte Funktionsweise des Computers wird im Rahmen der Erläuterung dieses kleinen Vorgangs zusammengefasst. Aus dem Kapitel 2 wird die Tatsache rekapituliert, dass hinter jedem Vorgang Programmcode steht, der in höherer Programmiersprache verfasst und dann in Maschinensprache übersetzt wird. Die Inhalte des dritten Kapitels werden über die Binärkodierung des Programmcodes eingebunden. Die Verbindung zum vierten Kapitel wird hergestellt, indem die Verarbeitung des Programmcodes durch Steuerwerk und Rechenwerk und die Speicherung des Programmcodes und der Programmdaten im Speicher in Erinnerung gerufen werden.

Als beispielhafter Vorgang im Computer, der die bisher beschriebenen Elemente der vorigen Kapitel erklärt und im Alltag regelmäßig ausgeführt wird, könnte ein Mausklick oder ein Tastendruck benutzt werden. Das Beispiel des Tastendrucks macht die Übertragung von Programmdaten in den Speicher besonders anschaulich. Daher werden die Inhalte aus den vorherigen Kapiteln anhand des Vorgangs beschrieben, dass ein Buchstabe eingetippt wird.

Die verschiedenen Inhalte des Studientextes werden zusammenhängend in einer Grafik dargestellt, die den Entstehungsprozess von Programmcode aus dem zweiten Kapitel und die FDE-Schleife enthält.

2.5.7. Kapitel 6: Exkurs: Sind Computer intelligent?

In diesem Kapitel wird kurz beschrieben, was unter dem Forschungsgebiet der künstlichen Intelligenz zu verstehen ist. Außerdem wird die Frage nach der Intelligenz von Computern erläutert. Dies ist besonders interessant, da die Studierenden nun die im Studientext erlernten Inhalte auf diese Frage anwenden können. Außerdem werden einige Punkte aus dem Studientext so noch einmal hervorgehoben, z.B. dass Computer verarbeiten ohne zu verstehen.

Zu Beginn des Kapitels wird erklärt, was unter dem Begriff *künstliche Intelligenz* zu verstehen ist. Davon ausgehend wird dann die Frage nach der Intelligenz von Computern diskutiert. Als Bezug zu den im Studientext erlernten Inhalten wird erklärt, dass auch die Programme aus der künstlichen Intelligenz dem Computer sein Verhalten vorgeben und er sie verarbeitet ohne zu verstehen.

Danach wird dann die philosophische Seite der Frage nach der Intelligenz des Computers betrachtet. Dabei wird dargelegt, dass die Antwort auf diese Frage davon abhängt, von welchem Intelligenzbegriff die Rede ist. Es wird dafür zwischen der Definition über das Verhalten und der über die Existenz von Bewusstsein unterschieden. (Vgl. Goldschlager et al., 1990: S.292f)

Nun werden die im Studientext erlernten Inhalte auf die Gegenüberstellung der Arbeitsweise von Gehirn und Computer bezogen. Im Studientext haben die Studierenden gelernt, dass im Computer Programmbefehle in Schaltkreisen deterministisch verarbeitet werden. Dies wird hier als ein Gegenargument zur Existenz von Bewusstsein bei Computern eingebracht. Als weiteres Gegenargument wird die unterschiedliche Arbeitsweise von Gedächtnis und Speicher angeführt, die die Studierenden nun aus dem Studientext kennen. Danach wird auch die Gegenhypothese dargelegt, nach der die Möglichkeit besteht, dass Computer Bewusstsein entwickeln (Vgl. Harel, 2002: S.178). Die Antwort auf die Frage, ob Computer jemals Bewusstsein entwickeln werden, wird offen gelassen, da sie durch fachliches Wissen der Informatik nicht beantwortet werden kann.

2.6. Verwendete Literatur

In die Gestaltung der einzelnen Kapitel sind Ideen aus verschiedenen Büchern eingeflossen. Dabei waren vor allem Ideen zur anschauliche n und einfachen Darstellungsweise von Sachverhalten der Informatik nützlich.

Besonders hilfreich waren dabei die Bücher des Autors David Harel *Das Affenpuzzle* (2002) und *Algorithmik* (2010) und die in die Informatik einführenden Werke *Informatik* (Goldschlager et al., 1990) und *Was ist Informatik?* (Rechenberg, 2000). All diese Werke sind verständlich geschrieben und enthalten Ideen, wie Themen der Informatik anschaulich vermittelt werden können. Die beiden Harel-Werke enthalten zum Teil identische Textstücke. Im Studientext ist dabei jeweils nur eine der beiden Quellen von Harel aufgeführt.

Neben diesen vier Hauptquellen sind einige Anregungen für eine anschauliche Gestaltung auch aus anderen Büchern eingeflossen. Dazu gehören Werke, die hauptsächlich die Bedienung von Computern thematisieren, aber auch einen kleinen Einblick in deren Funktionsweise geben: *EDV-Grundwissen* (Precht et al., 2004), *PCs für Dummies* (Gookin, 2010) und *Einführung in die PC-unterstützte Datenverarbeitung* (Lange, 2004).

Des Weiteren werden die Schulbücher *Informatik bis zum Abitur* (Engelmann, 2002) und *Informatik für Schule und Ausbildung* (Hattenhauer, 2010) verwendet. Außerdem wird auf das Einführungskapitel des Buches *Rechnerorganisation und -entwurf* (Patterson et al., 2005) zurückgegriffen.

Am Ende des Studientextes wird den Studierenden weiterführende Literatur empfohlen. Hier wird kurz begründet, welche Literatur für eine weitere Beschäftigung mit dem Thema für die fachfremden Studierenden gut geeignet ist. *Das Affenpuzzle* behandelt inhaltlich, abgesehen von dem Einführungskapitel, nur die Berechenbarkeitstheorie und ist daher für eine Vertiefung des Studientextthemas nicht geeignet. In *Algorithmik* werden hingegen verschiedene Bereiche von Algorithmen und Programmen für Laien verständlich erklärt, z. B. Kontrollstrukturen von Programmiersprachen, die verschiedenen Programmiersprachebenen und objektorientierte Programmierung. Dieses Buch ist insofern einzigartig, als es inhaltlich und sprachlich speziell für Laien verfasst wurde. Daher wird es den Studierenden empfohlen, wenn sie sich über den Studientext hinaus mit Programmen und Programmiersprachebenen beschäftigen wollen. Außerdem gehört es zu den Werken, die sich mit zentralen Konzepten und nicht mit konkreter Programmierung befassen. Die Ausrichtung auf die Vermittlung von wesentlichen Ideen haben *Informatik* und *Was ist Informatik?* mit Harels Buch gemeinsam. Des Weiteren sind auch diese Werke verständlich und anschaulich verfasst und bieten einen Einblick in alle Teilbereiche der Informatik. Daher werden auch diese Bücher zur weiterführenden Lektüre empfohlen.

3. Der Studientext „Wie funktionieren Computer?“

Heute bee influssen Computer unseren Allt ag maßge blich: Mit ihrer Hilfe werden Banken gesteuert und Flüge gebuc ht, sie stehen uns als PCs in je dem Büro und fast jedem Haushalt zur Verfügung und ein großer Teil der Kommunikation findet inzwischen über das Internet statt. Die Veränder ung von Arbeitsprozessen, die s eit der Erfindung der Computer stat tgefunden hat, ist mit der industriellen Revolution vergleichbar. Durch die industrie lle Revolution wurde es möglich, körperliche Arbeit durch Maschinen verrichten zu lassen, sofe rn es sich um automatisierbare Aufgaben handelt. Außerdem sind Maschinen in der Lage ein Vielfaches der menschlichen Muskelkraft und Schnelligkeit aufzubringen. So können sie z.B. sehr schwere Gegenstände heben und tausende Bücher in einer Stunde drucken. Auf ähnlich e Weise ermöglichen Computer eine Steigerung unserer geistigen Kräfte. Sie könne n automatisierbare geistige Aufgaben sehr schnell und z uverlässig erledigen und riesige Mengen an Daten s peichern. Die Geschwin digkeit und der Speicherplatz nehmen außerdem ständig zu, was sich in der ra schen Entwicklung der Einsatzmöglichkeiten von Computern in Alltag und Technik zeigt.

Doch die wenigsten Menschen wissen, wie ei n Computer funktioniert. Die Konzepte, die dem Ganzen zu Grunde liegen, sind gar ni cht so schwer zu begreifen. Tatsächlich haben sie s ich in den letzten 50 J ahren kaum v erändert. Schwieriger und unübersichtlicher wird es erst, wenn man a lle technisc hen Details eines Computers begreifen möchte. Dieser Studientext bes chreibt die Grundkonzepte der Funktionsweise v on Computern. Die Bearbeitung der enthaltenen Aufgaben unters tützt das Verständnis der Inhalte.

1. Einführung in die Computerwelt

In diesem Kapitel wird eine Einführung in z entrale Begriffe und Ideen der Informatik gegeben, die in den nächsten Kapiteln dann genauer erläutert werden.

Computer betreiben Datenverar beitung: der Nutzer gibt dem Computer Daten, dies e werden vom Computer verarbeitet und dann werden andere Dat en wieder ausgegeben.

Eingabe -> Computer -> Ausgabe

Die **Eingabe** (englisch: **Input**) erfolgt über **Eingabegeräte** wie M aus, Tastatur, CD–Laufwerk, Mikrophon oder Scanner. So erhält der Computer bestimmte Daten, z.B.

ein Wort, das eingetippt wurde. Und dann erfolgt eine **Ausgabe** der Daten (englisch: **Output**) über **Ausgabegeräte** wie Drucker, Lautsprecher, CD-Brenner und Bildschirm. Entsprechend kann die Ausgabe z.B. die Form einer gedruckten Seite oder eines Bildes auf dem Bildschirm haben. Die Eingabe und Ausgabegeräte zusammen nennt man abkürzend auch **E/A-Geräte**.

Dass auf die Eingabe (z.B. Mausklick) eine andere Ausgabe erfolgt (z.B. Ton aus dem Lautsprecher), liegt daran, dass die Daten im Computer verarbeitet werden. Neben der Verarbeitung und der Ein- und Ausgabe werden die Daten auch im Computer gespeichert. Außerdem müssen die Daten übertragen werden, z.B. zwischen den Eingabegeräten und den Geräten, die die Datenverarbeitung vornehmen. Dies sind auch schon die vier grundlegenden Prinzipien eines Computers:

- Eingabe / Ausgabe
- Datenspeicherung
- Datenverarbeitung
- Datenübertragung

Alle Teile des Computers, die man sehen und anfassen kann, auch wenn der Computer aus ist, werden **Hardware** genannt. Dazu gehören die E/A-Geräte wie Maus, Bildschirm, etc., das Gehäuse und die darin enthaltenen Bauteile. Die wichtigsten dieser inneren Bauteile sind der Prozessor und der Speicher:

Der **Prozessor**, auch CPU (Central Processing Unit) genannt, ist das Herzstück des Computers, denn hier findet die Datenverarbeitung statt. Er kann nur sehr einfache Aktionen durchführen: die Grundrechenarten (Addition, Subtraktion, Division, Multiplikation) und Daten aus dem **Speicher** holen oder dort ablegen. Weil der Prozessor diese Aktionen jedoch in sehr hoher Geschwindigkeit ausführen kann, sind Computer so leistungsfähig. Zwischen den verschiedenen Komponenten eines Computers, z.B. zwischen Prozessor und Speicher oder zwischen Prozessor und einem Eingabegerät, werden die Daten durch Datenleitungen übertragen.

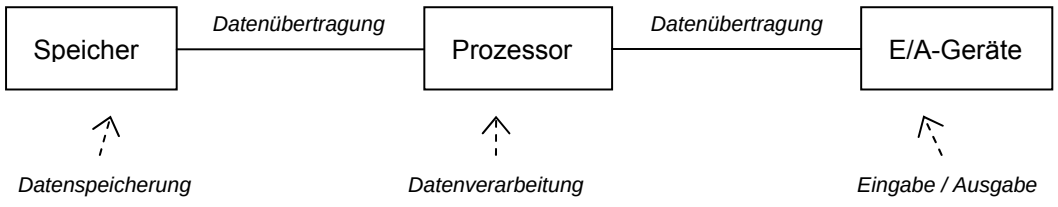


Abb. 1 Aufbau eines Computers

Sobald ein Computer angeschaltet ist, laufen verschiedene **Programme** an. Ohne Programme könnte ein Computer nicht arbeiten, denn sie steuern die Hardware. Die Hardware stellt die Möglichkeiten für Datenverarbeitung, -speicherung, -übertragung und Ein- und Ausgabe bereit, aber erst die Programme sorgen dafür, dass diese Prozesse auch stattfinden. Wenn man von mehreren Programmen spricht, sagt man dazu auch **Software**. Allseits bekannt ist **Anwendungssoftware**, das sind die Programme, die man ständig im Alltag benutzt: z.B. Textverarbeitungsprogramme, Webbrowser, Musikplayer und Zeichenprogramme. Die Anwendungsprogramme bauen auf anderen, unsichtbaren Programmen auf, der so genannten **Systemsoftware**. Diese koordiniert unter anderem den Start-Vorgang des Computers beim Einschalten und regelt, welches Programm wann verarbeitet wird. Viele der Systemprogramme sind im **Betriebssystem** zusammengefasst.

Aufgabe I Ordnen Sie die folgenden Begriffe entweder der Hardware oder der Anwendungssoftware oder der Systemsoftware zu (Mehrfachnennungen möglich): USB-Stick, CPU, Programm, DVD-Brenner, Word, Betriebssystem, Programm, Datenleitung, Drucker

Aufgabe II Ordnen Sie die folgenden Begriffe entweder der Eingabe, Ausgabe, Datenverarbeitung, Datenspeicherung oder Datenübertragung zu (Mehrfachnennungen möglich): Prozessor, DVD-Brenner, Subtraktion, Mikrofon, Rechnen, Programmausführung, Touchscreen, Datenleitung, Drucker, Speicher

Was hinter Programmen steckt und wie sie erstellt werden, wird im nächsten Kapitel erläutert. Auch wie die Hardware die Programme verarbeitet und wie die Bestandteile der Hardware funktionieren, wird im Verlauf des Studientextes erklärt. Zudem wird darauf eingegangen, welche Form Programme und Daten haben müssen, damit ein Computer sie verarbeiten kann. Im fünften Kapitel werden all diese Informationen nochmal explizit mit dem, was wir aus dem Alltag von Computern kennen, in Verbindung gebracht. Alle Erklärungen sind bewusst modellhaft verfasst und spiegeln nicht die exakte Wirklichkeit wieder, da so die Grundprinzipien leichter zu verstehen sind. Insgesamt wird ein Bild davon entstehen, nach welchen Prinzipien Computer funktionieren.

2. Programmierbarkeit – Dem Computer befehlen

Der Prozessor (und somit der Computer als Ganzes) kann nur Aufgaben erledigen, die durch die oben genannten einfachen Aktionen (z.B. Speicherzugriff, Addition) auszudrücken sind. Für eine Bearbeitung durch den Computer muss eine Aufgabe also in viele kleine Schritte zerlegt werden, die der Prozessor dann bearbeiten kann. Die Formulierung einer Aufgabe in vielen kleinen Schritten heißt **Algorithmus.** Der

Begriff des Algorithmus ist dabei nicht auf den Bereich der Informatik beschränkt, sondern kann jede Anleitung meinen, die Schritt für Schritt erledigt werden muss. Zum Beispiel braucht ein Bäcker, der einen Kuchen backen möchte, ein Rezept für den Kuchen. In diesem Rezept oder Algorithmus sind die verschiedenen Schritte beschrieben, die ausgeführt werden müssen, um den Kuchen zu backen. Der Computer kann jede Aufgabe lösen, die

1. durch einen Algorithmus ausgedrückt werden kann *und*
2. deren Inhalt Zahlen, Buchstaben, Bilder oder andere Daten sind, die im Speicher gespeichert werden können. Mehr dazu in Kapitel 3.

Dies macht den Computer zu etwas Besonderem, zu einer vielseitigen oder **universellen Maschine**. Vor dem Computerzeitalter gab es schon Maschinen, die durch Mechanik betrieben wurden und bei denen war das ganz anders: Eine Maschine wurde ihrer Aufgabe entsprechend gebaut. Um eine neue Aufgabe zu erfüllen, musste die Maschine umgebaut werden. Dies ist bei Maschinen, die für ein bestimmtes Produkt gebaut werden, auch heute noch so, z.B. kann eine autobauende Maschine keine Fahrräder bauen. Heute sind selbst solche Maschinen teilweise bis zu einem gewissen Grad programmierbar. Es gibt aber keine mechanische Maschine, die alle körperlichen Tätigkeiten verrichten kann, so wie ein Computer alle geistigen Tätigkeiten, die die oben genannten Bedingungen erfüllen, ausführen kann.

Im nächsten Absatz wird anhand eines ganz einfachen Beispielprogramms erläutert, wie ein Computerprogramm entsteht. Darauf folgend wird erklärt, wie es möglich ist, komplexe Programme, wie wir sie im Alltag kennen, zu entwickeln.

Aufgabe III Welche der folgenden Tätigkeiten lassen sich durch einen Algorithmus beschreiben?

a. Lösen einer quadratischen Gleichung
b. Pullover stricken
c. Benoten eines Aufsatzes
d. Sortieren von 200 verschiedenen Zahlen
e. Zimmer aufräumen
f. Nachschlagen einer Telefonnummer
g. Aufbauen eines Schranks
h. Leiten einer Gesprächsrunde

Aufgabe IV Welche davon kann der Computer ausführen?

2.1. Von der Idee zum Programm

Wenn ein Programm entstehen soll, liegt dem Ganzen zunächst eine Aufgabenstellung zu Grunde, z.B. ein kleines Programm zu entwickeln, das den Mittelwert zweier

Zahlen berechnet. Als nächstes muss eine Lösung für diese Aufgabe gefunden werden, die als Algorithmus ausgedrückt werden kann.

Ein Algorithmus für die Berechnung des Mittelwerts zweier Zahlen ist der folgende:

1. Nimm zwei Zahlen
2. Addiere die beiden Zahlen
3. Dividiere das Ergebnis durch zwei

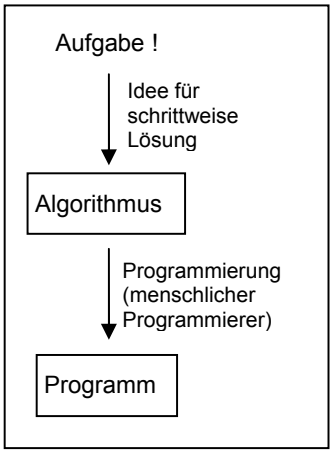


Abb. 2 Von der Aufgabe zum Programm

Für die Verarbeitung eines Algorithmus durch den Computer muss dieser in einer Form aufgeschrieben werden, die ein Computer verstehen kann, durch ein **Programm**. So ein Programm wird in einer **Programmiersprache** erstellt und besteht aus **Programmcode.** Jeder Schritt bzw. jede Zeile des Programmcodes heißt **Befehl** oder **Anweisung**. Das Erstellen des Programms bzw. Programmcodes heißt **programmieren**. Das Programm kann im Fall der Mittelwertberechnung so aussehen:

```
BerechneMittelwert
    1. Input a
    2. Input b
    3. Lade a
    4. Addiere b
    5. Dividiere 2
    6. Speichere c
    7. Output c
```

Abb. 3 Beispielprogramm

Der Name des Programms steht am Anfang: „BerechneMittelwert“, so wie beim Kuchenrezept der Name, z.B. „Apfelkuchen“. In den ersten beiden Befehlen werden zwei Zahlen a und b (über die Tastatur) eingegeben und dann in den Speicher übertragen. Die beiden Zahlen sind die **Zutaten**, die das Programm braucht, um ihren Mittelwert berechnen zu können. So wie man die Zutaten vom Kuchen braucht, um ihn backen zu können. Diese Programmzutaten nennt man **Input oder Eingabe.** In den Anweisungen 3-6 wird dann die eigentliche Rechnung durchgeführt, die den Mittelwert, also c=(a+b)/2 berechnet. Dies entspricht beim Kuchenbacken dem Backvorgang. Bei Computern ist es so, dass nur ein Wert aus dem Speicher geholt oder dorthin gebracht werden kann und dass der Prozessor auch nur einen Wert bearbeiten kann. Dies wird in Kapitel 4 noch genauer erläutert. Auf jeden Fall sind deswegen für die Rechnung drei Programmbefehle nötig. In der dritten Anweisung wird der Wert, der in a gespeichert ist, in den Prozessor geholt. In der vierten Anweisung wird der Wert, der in b gespeichert ist, aus dem Speicher geholt und dazu addiert. In der fünften Anweisung wird das Ergebnis durch 2 geteilt. Im sechsten Be-

fehl wird das Ergebnis der gesamten Rechnung, also der Mittelwert, in c gespeichert. Am Ende erfolgt die **Ausgabe** oder der **Output** – das Ergebnis, die Zahl in c, wird ausgegeben. Das ist beim Kuchenbacken der **fertige Kuchen.**

Nun ist der Programmcode nicht weiter als ein Stück Text. Die Wirkung entfaltet der Programmcode erst bei der Ausführung der Schritte bzw. Befehle, so wie der Kuchen nicht durch das Rezept einfach da ist, sondern erst entsteht, während der Bäcker den Kuchen backt. Bei der Mittelwertberechnung beginnt die Ausführung des Programms, wenn es z.B. durch einen Klick auf ein Symbol auf dem Bildschirm gestartet wird. Wie die Ausführung von Programmen im Computer genau abläuft, wird in Kapitel 4 beschrieben.

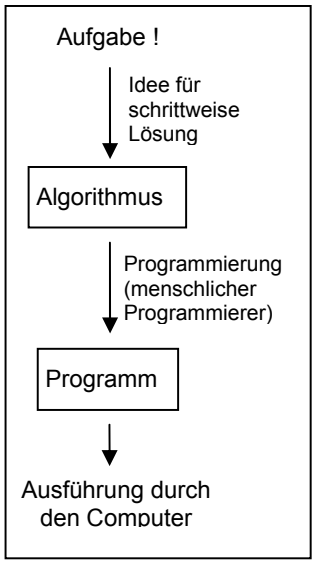


Abb. 4 Von der Aufgabe zur Programmausführung

Bei der Ausführung des Programms befolgt der Computer genau die Programmbefehle. Dies könnte ein Bäcker beim Kuchen backen nicht garantieren. Dies ist ein Vorteil in der Bearbeitung von Programmen durch einen Computer. Andererseits führt der Computer das Programm blind durch und dadurch werden auch Fehler, die der Programmierer bei der Erstellung des Programmcodes gemacht hat, mit ausgeführt. Ein Bäcker würde den Kuchen aus dem Ofen nehmen, wenn er anfängt zu qualmen, auch wenn im Rezept durch einen Tippfehler „10 Stunden backen“ statt „1 Stunde backen“ steht. Ein Computer hingegen würde den Kuchen verbrennen lassen, da er sich an die Zahl 10 halten würde. Denn Computer haben keinen Alltagsverstand, sie verstehen überhaupt nichts von dem Programm, welches sie ausführen. Das muss man im Hinterkopf behalten, wenn Aufgaben an Computer übertragen werden.

Aufgabe V Schreiben Sie ein Programm, das von zwei Zahlen jeweils das Quadrat berechnet und dann die zwei Quadratzahlen addiert! Verwenden Sie dafür die Befehle Input, Output, Lade, Speichere, Addiere und Multipliziere.

2.2. Entstehung komplexer Programme

Nun ist die Frage, wie so ein einfaches Programm zur Mittelwertberechnung mit den komplexen Programmen, die wir im Alltag verwenden, zusammenhängt. Die Computerprogramme, die wir im Alltag verwenden, bestehen aus vielen Tausend oder Millionen Programmbefehlen. Bei so einer umfangreichen Programmierung den Überblick zu behalten und keine Fehler einzubauen, ist mit den bisher vorgestellten Methoden schwierig, wenn nicht unmöglich. Es gibt aber andere Methoden, die die

Programmierung erleichtern. Zum einen können Programme innerhalb von anderen Programmen verwendet werden und zum anderen gibt es Programmiersprachen, die anders arbeiten als die bisher vorgestellten und die die Programmierung einfacher machen.

2.2.1. Programme in Programmen verwenden

Es gibt die Möglichkeit, dass ein Programm schon bestehende Programme verwendet und auf ihre Funktionen zurückgreift. So könnte zum Beispiel ein Tabellenkalkulationsprogramm, das die Möglichkeit bereitstellen soll, in einem Feld automatisch den Mittelwert zweier anderer Felder anzuzeigen, das zuvor erstellte Programm zur Mittelwertberechnung verwenden. Ein anderes Beispiel ist, dass Anwendungsprogramme wie Textverarbeitung oder Musikplayer nicht selbst für die Darstellung ihrer Grafik auf dem Bildschirm sorgen. Stattdessen gibt es ein Grafikprogramm, das für alle Programme des Computers die Darstellung auf dem Bildschirm übernimmt. In solchen Fällen werden Input und Output nicht über E/A-Geräte getätigt, sondern die Daten werden von Programm zu Programm übergeben. Bei der Mittelwertberechnung im Rahmen eines Tabellenkalkulationsprogramms würden dann die Zahlen intern aus den Feldern der Tabelle und nicht über die Tastatur übergeben werden.

Bei der Erstellung neuer komplexer Programme kann also auf die Funktionen bestehender Programme zurückgegriffen werden, wie zum Beispiel die des Grafikprogramms. Außerdem wird bei der Erstellung komplexer Programme direkt zu Beginn das Programm in einzelne Teilprogramme unterteilt, die bis zu einem gewissen Grad unabhängig erstellt und verändert werden können. Das vereinfacht die Programmierung erheblich und man kann sie unter mehreren Menschen aufteilen. Es ist also so, dass komplexe Programme aus vielen kleinen Programmen bestehen und Programme schon bestehende Programme verwenden können. Dies ist ein Grund dafür, dass im Verlauf der Zeit die Möglichkeiten, die Computer bieten, immer weiter anwachsen und sich weiterentwickeln.

2.2.2. Andere Programmiersprachen

Das zuvor vorgestellte Programm zur Mittelwertberechnung ist in einer sogenannten **Maschinensprache** geschrieben. Der Begriff kommt daher, dass die Programmiersprache an dem Prozessor orientiert ist: Ein Befehl der Maschinensprache entspricht genau einer Aktion des Prozessors. Dies macht die Programmierung allerdings aufwändig und schwer zu handhaben, da das Programm sehr kleinschrittig erstellt

werden muss. Im Mittelwertprogramm konnte man sehen, dass für eine einfache Rechnung wie c=(a+b)/2 vier Maschinenbefehle benötigt werden.

Früher wurde in Maschinensprache programmiert. Um die Nachteile zu überwinden, wurden dann sogenannte **höhere Programmiersprachen** entwickelt, bei denen ein Befehl mehrere Aktionen des Prozessors bewirkt. Es werden also mehrere kleine Befehle der Maschinensprache zu einem anderen mächtigeren Befehl zusammengefasst. Zum Beispiel kann man „Lade a, Addiere b, Dividiere 2, Speichere c" zu „c=(a+b)/2" zusammenfassen. Allerdings kann der Prozessor mit solchen mächtigen Befehlen nicht umgehen, d.h. er kann ein Programm in höherer Programmiersprache nicht verarbeiten. Daher gibt es **Übersetzungsprogramme**, die ein Programm aus der höheren Programmiersprache in ein Programm in Maschinensprache übersetzen. Das Übersetzungsprogramm geht das Programm in höherer Programmiersprache Befehl für Befehl durch und übersetzt jeden Befehl in mehrere Maschinenbefehle.

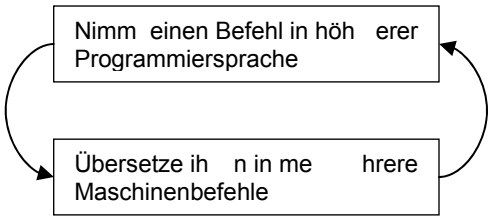


Abb. 5 Übersetzungsvorgang

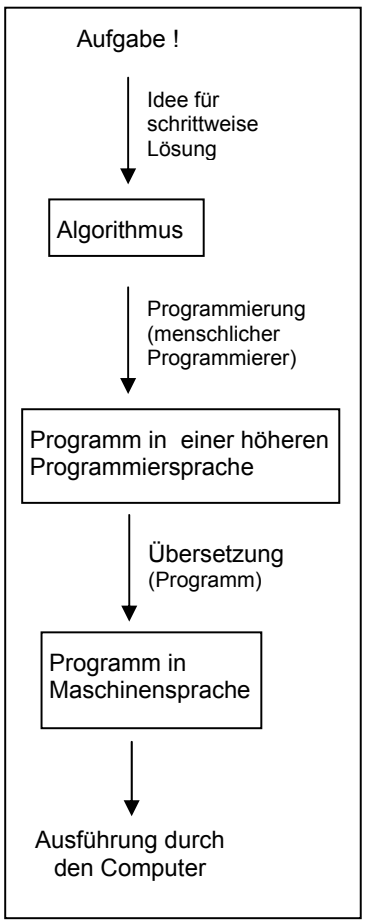


Abb. 6 Von der Aufgabe zur Programmausführung (erweitert)

Das oben dargestellte Programm zur Mittelwertberechnung würde in einer höheren Programmiersprache wie folgt lauten:

```
        Input a, b
c=      (a+b)/2
Output  c
```

An diesem Beispiel kann man sehen, wie höhere Programmiersprachen das Programmieren erleichtern. Eine höhere Programmiersprache eröffnet noch viele andere Möglichkeiten, auf die hier nicht näher eingegangen werden kann. Das Besondere an höheren Programmiersprachen ist unter anderem, dass die Programmbefehle weniger kleinschrittig und somit viel näher an der menschlichen Denkweise sind, als bei der Maschinensprache. Heute werden fast nur noch höhere Programmiersprachen für die Programmierung verwendet und durch die Übersetzungsprogramme in Maschinensprache übersetzt. Die heutige Komplexität und Vielfalt von

Programmen ist erst durch höhere Programmiersprachen möglich geworden.

Bei der Übersetzung eines Befehls aus der höheren Programmiersprache in mehrere Maschinenbefehle muss in dem Übersetzungsprogramm ganz genau festgelegt sein, welcher Befehl aus der höheren Programmiersprache zu welchen Befehlen in der Maschinensprache wird. Das heißt auch, dass es nur eine begrenzte Anzahl an Befehlen in jeder Programmiersprache geben kann, so dass eine eindeutige Definition der Übersetzung möglich ist. Eine Programmiersprache besteht also nur aus ganz bestimmten „Wörtern", den Befehlen, und diese müssen in der immer gleichen Form verfasst werden. Man kann nicht einfach statt „Input x" „x Input" schreiben, da das Übersetzungsprogramm für diesen Befehl keine Übersetzung parat hätte. Dieser Übersetzungsvorgang hat, wie man sieht, wenig gemeinsam mit dem Übersetzen eines Textes von Deutsch in Englisch. In der menschlichen Sprache können Wörter in ganz verschiedener Art und Weise kombiniert werden oder sogar neue Wörter erfunden werden, die sich dann aus dem Kontext erschließen lassen. Es gibt auch einen Begriff für diesen Unterschied zwischen der Sprache von Computern und Menschen: Programmiersprachen haben eine absolute Semantik und menschliche Sprachen haben eine auf die Umwelt bezogene Semantik. Es könnte eine Programmiersprache geben, in der „grumli" für „Input" steht und „grmps" für „+", wenn jemand sich den Spaß macht, ein Übersetzungsprogramm für so eine Programmiersprache zu erstellen. Tatsächlich gibt es eine Programmiersprache „Shakespeare", die den Programmcode in der Form von Shakespeare-Dramen darstellt.

In der Maschinensprache entspricht jeder Befehl einer Aktion des Prozessors. Wie kann der Prozessor Befehle der Maschinensprache wie „Addiere" und „Speichere" verstehen? Das sind auch Wörter, und der Prozessor spricht keine menschliche Sprache. Die Antwort ist, dass der Prozessor die Maschinenbefehle verarbeiten kann, weil sie **binär kodiert** sind. Was das bedeutet wird, im nächsten Kapitel erklärt.

Aufgabe VI Schreiben Sie das Programm aus Aufgabe V nun in höherer Programmiersprache.

Aufgabe VII Inwiefern ist schon vor der Übersetzung eines Programms aus höherer Programmiersprache in Maschinensprache vorherbestimmt gewesen, wie der Maschinencode nach der Übersetzung aussieht?

3. Binärkodierung - Von Nullen und Einsen

Computer enthalten viele elektrische Schalter, die, wie alle Schalter, zwei Zustände annehmen können: „Schalter auf" und „Schalter zu". Dies bedeutet dann: „Strom fließt" oder „Strom fließt nicht". Der Aufbau des Computers wird im nächsten Kapitel

erörtert. An dieser Stelle ist wichtig, dass alle Daten, die ein Computer verarbeiten soll, durch Binärkodierung dargestellt werden müssen. Das bedeutet, dass sie durch Kombinationen von 0 und 1 ausgedrückt werden. Dabei steht 0 für „Strom aus“ und 1 für „Strom an“.

Man sagt bei einem Schalter, er speichere ein **Bit** an Daten, eine 0 oder eine 1, einmal „an“ oder „aus“. Um mehr Daten darstellen zu können, werden mehrere Nullen und Einsen kombiniert. Aus drei Schaltern, die aus oder an sein können, also Null oder Eins repräsentieren, lassen sich zum Beispiel acht verschiedene Kombinationen bilden:

000 001 010 011 100 101 110 111

In der Regel werden 8 Schalter verbunden, man sagt, sie speichern ein **Byte** an Daten. Aus acht Schaltern können 256 Kombinationen gebildet werden, die in Form von Nullen und Einsen so aussehen:

00000000 00000001 …. 11111100 11111110 11111111

Jeder Kombination kann eine bestimmte Bedeutung zugewiesen werden. Man legt z.B. fest, dass 01000001 ein „A“ ist und 01 000010 ein „B“, dass 00100001e in „!“ ist und dass 00001111 die Zahl 5 darstellt. Es gäbe viele verschiedene Möglichkeiten, auf diese Art Bedeutungen zuzuweisen, daher muss dies einheitlich erfolgen. Für **Buchstaben und andere Zeichen** wie =, ? oder § wird dies international einheitlich, aber willkürlich festgelegt.

Zeichen	binäre Darstellung
A	1000001
B	1000010
C	1000011
D	1000100
E	1001001
F	1001010
……..	
!	0100001
“	0100001
#	0100010
$	0100011
%	0100100
……..	

Tab. 1 Binärkodierung von Buchstaben und Zeichen

Zahlen werden nicht willkürlich kodiert wie Buchstaben, sondern sie ergeben tatsächlich ein Zahlensystem, in dem Zahlen addiert, multipliziert, subtrahiert und dividiert werden können. Allerdings rechnet der Computer dabei nicht mit dem Zahlensystem, das wird aus dem Alltag kennen. Im Alltag benutzen wird das Dezimalsystem, das aus zehn Ziffern besteht. Computer hingegen rechnen im Binärsystem, das nur zwei Ziffern enthält, die 0 und die 1. Die Addition von zwei Ziffern im normalen Zahlensystem funktioniert wie in den folgenden Beispielen:

4 + 5 = 9
7 + 4 = 1 mit Übertrag also 7 + 4 = 11
9 + 8 = 7 mit Übertrag also 9 + 8 = 17

Im Binärsystem funktioniert die Rechnung genauso:

0 + 0 = 0
0 + 1 = 1
1 + 1 = 0 mit Übertrag 1 also 1 + 1 = 10

Zahl	binäre Darstellung
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100

Tab. 2 Binärkodierung von Zahlen

Die **Maschinenbefehle** werden nicht einfach durch die Buchstaben, aus denen sie bestehen, kodiert. Für den Prozessor sind die Buchstaben, aus denen ein Befehl besteht, völlig unwichtig, da ein Prozessor keine menschliche Sprache spricht. Wichtig ist, welche Aktion der Maschinenbefehl im Prozessor bewirken soll. Daher werden die Befehle ihrer Funktion nach durchnummeriert.

Programmbefehl	binäre Darstellung
……..	
Lade	0000
Speichere	0001
Addiere	0010
Subtrahiere	0011
……..	

Tab. 3 Binärkodierung von Maschinenbefehlen

Der Befehl „Lade“ wird also als 0000 kodiert und nicht als „L“ „a“ „d“ „e“ = 1001100 1100001 1100100 1100101

Auch Töne, Bilder, Videos etc. lassen sich binär darstellen. Beispielhaft wird nun erklärt, wie ein Bild binär kodiert wird. Das Bild besteht aus vielen Quadraten, sogenannten Pixeln und jedes kann den Wert 0 oder 1 annehmen. Ist es die Null, ist das Pixel schwarz, ist es die 1, so ist es weiß.

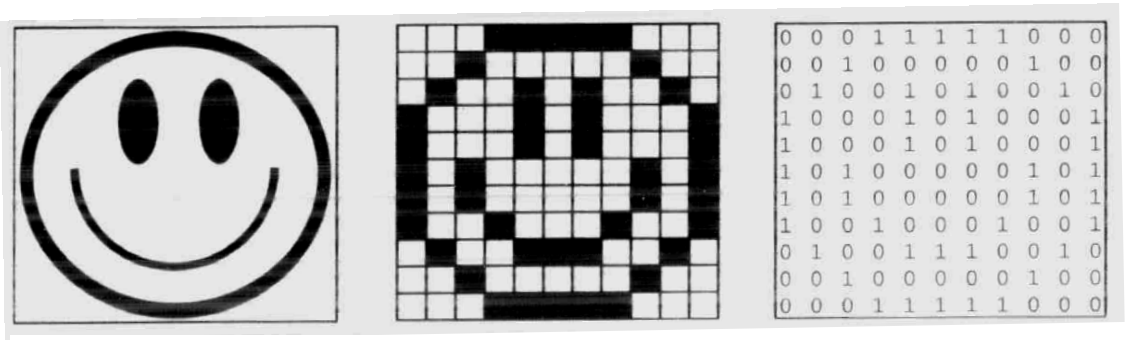


Abb. 7 Binärkodierung von Bildern

Dies lässt sich auf mehr Farben als schwarz und weiß ausweiten, indem jeder Farbe wie bei Zahlen und Buchstaben eine Acht-Bit- bzw. Ein-Byte-Kombination aus Nullen und Einsen zugeordnet wird, wie oben beschrieben. So kann man 256 Farben darstellen. Dann kann jeder Pixel einen Binärcode zwischen 00000000 und 11111111 annehmen und so weiß man, welche Farbe welcher Pixel haben soll.

Wir Menschen sehen zwar Zahlen, Buchstaben, Programmbefehle und Farben als verschiedene Dinge an, aber durch die Binärkodierung bekommen sie für den Computer alle die gleiche Form. Binärcodes erhalten erst dann eine Bedeutung, wenn man sie interpretiert. Wie der Prozessor binäre Programmbefehle interpretiert, wird im nächsten Kapitel erläutert.

Aufgabe VIII Was für eine Zahl unseres Zahlensystems ist der Binärcode 0011? Welcher Maschinenbefehl verbirgt sich hinter dem Binärcode 0011? Und welcher Buchstabe verbirgt sich hinter 0100001? Was ist „DA!“ in Binärschreibweise? Was ist die Zahl „7“ in Binärschreibweise? Wie wird der Maschinenbefehl „Speichere“ in Binärschreibweise ausgedrückt? Benutzen Sie für diese Aufgabe die obigen Tabellen.

Aufgabe IX Was ist 0100+0011 binär addiert? Was für Zahlen unseres Zahlensystems verbergen sich hinter 0100, 0011 und dem Ergebnis?

4. Wie die Hardware arbeitet

Wie in der Einführung schon erwähnt, sind die Hauptbestandteile des Computers Speicher, E/A-Geräte und Prozessor.

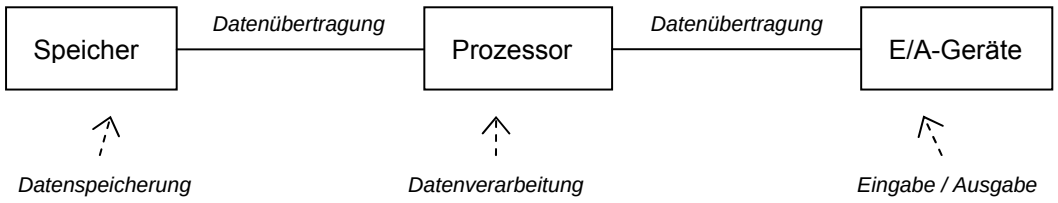


Abb. 8 Aufbau eines Computers (Wiederabbildung von Abb. 1)

Alle Teile des Computers besitzen viele Schalter, die einen Kontakt öffnen oder schließen können. Diese Schalter sind durch Leitungen miteinander verbunden und ergeben so verschiedene Schaltkreise. Aus solchen Schaltkreisen bestehen alle Bestandteile des Computers, auch der Prozessor und der Speicher.

Früher waren Computer viel größer als heute. Sie bestanden aus elektrischen Schaltern, die aus heutiger Sicht sehr groß waren und von Hand auf Chips gelötet wurden. Ende der 1940er Jahre wurde der Transistor erfunden, der zwar auch ein elektrischer Schalter ist, der aber viel kleiner, zuverlässiger und stromsparender ist als die bis dahin verwendeten Schalter. Seit Ende der 1960er Jahre gibt es die Möglichkeit, die Transistoren maschinell auf Chips zu drucken. Seitdem wurden Transistoren immer kleiner und der Druck auf Chips besser, bis schließlich in den 1970er Jahren ein kompletter Prozessor auf einen Chip passte. Heutige Computer bestehen also aus Prozessorchip, Speicherchip und anderen Bauteilen, die hergestellt werden, indem Schaltkreise mit winzigen Transistoren auf die Chips gedruckt werden. Im Fol-

genden wird genauer erläutert, wie Speicher und Prozessor aufgebaut sind und wie Programme durch die Hardware verarbeitet werden.

4.1. Speicher

Im Speicher sind Daten und Programme in Binärkodierung enthalten: Zahlen, Buchstaben und Zeichen, Bildschirmfarben, Töne oder Maschinenbefehle. Es stehen also nicht nur die Daten, die zum Programm gehören, sondern auch das Programm selbst im Speicher. Als der Mathematiker John von Neumann in den 1940er Jahren die Idee aufkam, nicht nur Daten, sondern auch Programme im Speicher aufzubewahren, war dies ein ganz neues Konzept. Zuvor wurde ein Programm dem Computer zugeführt, indem eine Lochkarte eingelegt wurde. Das kann man sich vorstellen, als ob man um ein Programm zu starten immer eine CD einlegen muss. Das Besondere daran, wenn Programme im Speicher stehen, ist, dass immer darauf zugegriffen werden kann und dass die Programme durch andere Programme verändert werden können. John von Neumann verband diese Idee der Programmspeicherung mit anderen, schon bestehenden Konzepten und stellte ein Modell eines Computers auf. Der bisher vorgestellte Aufbau eines Computers durch Speicher, Prozessor und E/A-Geräte entspricht genau dem Von-Neumann-Modell, da Computer bis heute konzeptuell nach dem Von-Neumann-Modell gebaut werden.

Nun wird erläutert, welche Arten von Speicher es gibt, wie der Speicher aufgebaut ist und was der Unterschied zwischen Speicher und Gedächtnis ist.

4.1.1. Speicherarten

Der Computer hat nicht nur einen Speicher. Es gibt den **Arbeitsspeicher** und den **permanenten Speicher**, wie z.B. die Festplatte, CDs und USB-Sticks. Im Arbeitsspeicher werden die Programme ausgeführt, Zwischenergebnisse gespeichert, er wird während der Arbeitsvorgänge benutzt. Wenn der Computer ausgeschaltet wird, wird der Arbeitsspeicher gelöscht, alle Daten gehen verloren. Der permanente Speicher, z.B. die Festplatte, ist dafür da, Daten dauerhaft zu speichern. So wird zum Beispiel ein Wort, das jemand in einem Textverarbeitungsprogramm schreibt, zunächst nur im Arbeitsspeicher gespeichert. Klickt man auf „Speichern“, wird es zusammen mit dem gesamten Dokument auf die Festplatte übertragen und somit permanent gespeichert.

Als Analogie kann man sich einen Schreibtisch vorstellen. Der Arbeitsspeicher ist ein Notizzettel, auf dem zwischendurch etwas notiert wird, und die Festplatte ist ein Ak-

tenschrank, in den Zettel eingeordnet werden können. Auf den Notizzettel kann man schnell schreiben oder ihn lesen. Etwas in den Aktenschrank einzusortieren dauert recht lange. Der Notizzettel wird weggeschmissen, wenn man mit der Arbeit fertig ist, der Aktenschrank nicht.

Die Programme bzw. der Programmcode sind auf der Festplatte gespeichert, damit sie nicht verloren gehen. In dem Moment, in dem ein Programm ausgeführt werden soll, wird der Programmcode in den Arbeitsspeicher kopiert und vom Prozessor ausgeführt. Denn der Prozessor kann nur mit Daten bzw. Programmcode direkt arbeiten, wenn sie sich im Arbeitsspeicher befinden. Daher ist der Arbeitsspeicher der Speicher, der für den Prozessor relevant ist. Die Festplatte ist aus Sicht des Prozessors nur ein weiteres Eingabe-/ Ausgabe-Gerät.

4.1.2. Speicheraufbau

Der Speicher besteht aus einer Aneinanderreihung von Schaltern, die, als Nullen und Einsen aufgefasst, die zu speichernden Inhalte in Binärkodierung darstellen. Er ist in kleine Abschnitte eingeteilt, von denen jeder acht Schalter umfasst. In jedem Abschnitt können also acht Nullen oder Einsen, ein Byte, gespeichert werden und somit eine Zahl oder ein Buchstabe oder ein Maschinenbefehl. Um die Ein-Byte-Abschnitte getrennt ansprechen zu können, werden sie durchnummeriert.

1 01100101	
2 10110010	
3 01011100	
4 01011000	
5 11100011	
….	
2756 10101010	
2757 01100101	
2758 00111000	
……	

Tab. 4 Speicheraufbau

Man nennt einen Abschnitt eine **Speicherzelle**. Die Nummern heißen **Speicheradressen** und der Inhalt der Speicherzelle heißt **Speicherinhalt.**

Die Speicherzellen kann man sich wie Briefkästen vorstellen. Die Nummern auf den Briefkästen sind die Speicheradressen und die Speicherinhalte sind die Briefe im Briefkasten. Wenn jemand einen Brief bringt, weiß er nur die Nummer des Briefkastens und kennt keine anderen Merkmale des Briefkastens. Wenn jemand den Briefkasten leert, kann er vorher nicht wissen, was drin ist. Genauso ist es mit Speicherzellen, man kann etwas nur unter Verwendung der Speicheradresse speichern (Brief hinbringen) oder laden (Brief holen): „Hole den Inhalt der Speicherzelle 32“ oder „Lege den Buchstaben ‚i’ in die Speicherzelle 54“. Der Unterschied dabei ist, dass ein Brief nicht mehr im Kasten ist, wenn man ihn holt. Im Speicher bleibt der Inhalt erhalten, wenn er von dort geholt wird. Man kann belie-

big oft die Daten abrufen und sie verschwin den nicht, bis jemand einen neuen Inhalt dort speichert.

Die Durchnummerierung der Speicherzellen funktioniert, indem ac ht Stromleitungen vom Prozessor ausgehen und am Speicher vorbeiführen, sogenannte Adressleitungen. Diese Adressleitungen k önnen an oder aus sein, also Nullen und Eins en übertragen. So übertragen sie die Sp eicheradressen. Die Spei cherzellen sind wiederum so gebaut, dass s ie nur dann reagieren, wenn die An-Aus-Kombination der Stromleitungen ihrer Speicheradresse entspricht. Die Speicherzelle reagiert, indem sie ihren Inhalt an die Datenleitung weitergibt, die dann den Speic herinhalt zum Prozes sor tragen.

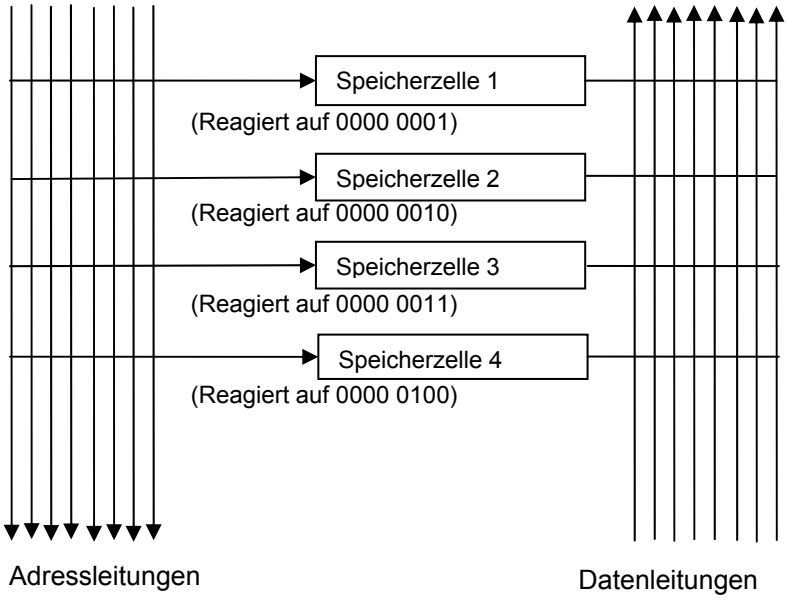


Abb. 9 Speicheradressierung

4.1.3. Speicher und Gedächtnis

Der Speicher ist insgesamt sehr systemat isch organisiert. Durch die Adress e kommt man zum Inhalt, auf keine andere Weis e. Das menschlic he Gedächtnis arbeitet hingegen assoziativ. Denkt je mand an die Sonne, fällt ihm auc h der Mond ein und vielleicht denkt er auch direkt daran, wie er das letzte Mal in der Sonne saß. Im Speicher gibt es solche Verknüpfungen nach Wortbedeutung oder Erinnerung nicht, er arbeitet rein systematisch. Es gibt Vers uche, auch im Speicher Verknüpfungen herzustellen, indem man die Verk nüpfung selbst auch im Speich er hinterlegt. Dies ist in kleinem Rahmen möglich. Im menschlichen Gehirn gibt es aber so viele und so komplexe Verknüpfungen, dass es unmöglich ist, sie alle darzustellen.

Aufgabe X Worauf kann der Pro zessor schn eller zugreifen, auf Arbeitsspeicher oder permanenten Speicher?

Aufgabe Xl Beschreiben Sie, wie der Speicherinhalt aus einer Speicherzelle geholt werden kann.

Aufgabe XII Wie könnte man den Speicher zurücksetzen, d.h. wie könnte man es bewerkstelligen, in einem b estimmten Mo ment alle s, was im Arb eitsspeicher steht, irgend wie festzuhalte n und an ei nem anderen Tag an dieser Stelle weiterzumachen?

4.2. Prozessor

Wie zuvor erwähnt, ist es die Aufgabe des Prozessors, die Programme auszuführen. Er ist in zwei Teile auf geteilt, in **Steuerwerk** und **Rechenwerk,** wobei jeder Teil verschiedene Funktionen übernimmt. Im Rechenwerk können di e Grundrechenarten ausgeführt werden. Das Steuerwerk ist dafür zuständig, die M aschinenbefehle au s dem Arbeitsspeicher zu holen u nd auszufü hren. Dabe i kontrollier t es das Rechenwerk und kann Daten zwischen dem Rechen werk und dem Arbeit sspeicher hin und her schicken.

Die Organisation des Prozessors veransch aulicht folgendes Beispiel: Ein Mensch sitzt am S chreibtisch und muss Rechenauf gaben lös en, die auf einem Notizblock notiert sind. Auf dem Tisch liegt außer dem ein Tasc henrechner. Die Reche naufgaben auf dem Notizblock (Arbeitsspeicher) sind in kleinen Schr itten aufgeschrieben (Programm). Nun muss der Mensch (Steuerwerk) nur noch die Schritte nacheinander lösen, indem er die Rechnung in den Tasch enrechner eingibt (Rechenwerk) und die Zwischenergebnisse auf dem Notizblock (Ar beitsspeicher) notiert und dann wieder für seine Rechnungen verwendet (Speicherzugriff).

Nun wird die Funktionsweise von Steuerwer k und Rec henwerk näher erläut ert und dann wird gezeigt, wie Programme verarbeitet werden.

4.2.1. Steuerwerk

Für die Verarbeitung der Pr ogrammbefehle ist das St euerwerk z uständig. Die Besonderheit liegt darin, dass der Computer die Worte der Be fehle nicht versteht, sondern mit Nullen und Einsen, also mit der Binärdarstellung der Befehle arbeitet. Wie er diese auf elektronische Art verarbeitet, wird nun erläutert:

Im Speicher besteht jeder Befehl aus ac ht Nullen und Einsen bz w. aus acht Schaltern, und jeder ist an oder aus. Der Befehl wird an das Steuerwerk durch Datenleitungen übertragen (Strom fließt = 1, Strom fließ t nicht = 0). Im Steuerwerk sind Schaltkreise, die daraufhin

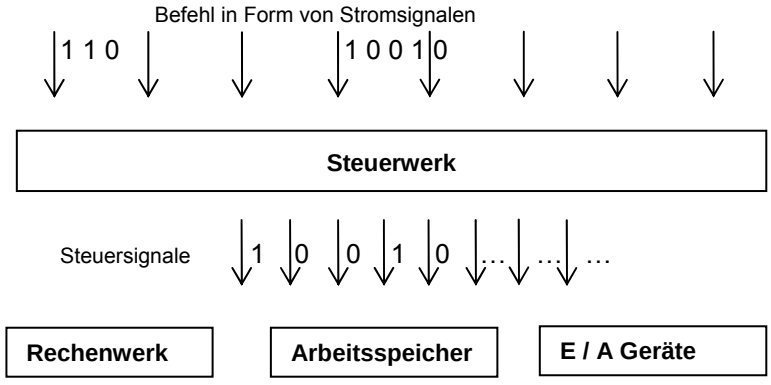


Abb. 10 Befehlsverarbeitung durch das Steuerwerk

andere Str omsignale, sogenann te Steuersignale auslösen. Dies e fließ en z um Rechenwerk und lösen dort die entsprechenden Vorgänge, wie z.B. Addieren oder Subtrahieren aus. Oder die Steuersignale werden an den Sp eicher gesendet, damit Daten geladen und gespeicher t werden können oder an die E / A – Geräte, um von dort oder dahin Daten zu übertragen.

Bestimmte eingehende Signale laufen also im Steuerwerk durch einen Schaltkreis und dann kommen andere Signale, die St euersignale, am Ende heraus. Zum Beispiel löst der Befehl 0 000 Steuersignale aus, die da s Laden bewirken. Das s dabei die richtigen Steuersignale er zeugt werden, wird beim Bau der Schaltkreise sichergestellt. Wie die Befehle verarbeitet werden, ist also schon beim Bau des Prozessors bestimmt worden.

4.2.2. Rechenwerk

Das Rec henwerk erhält also Steuersigna le vom Steuerwerk. Und je nachdem, welche das sind, führt es eine best immte Rechnung aus. Angenom men, die Steuersignale befehlen dem Rechenwerk zu addieren: Addier e 723. Dann addiert es die Binärzahl, die gerade im Rechenwerk ist, mit einer Binärzahl aus der Speicherzelle 723.

Das Rec henwerk besitzt daher für jede Grundrechenart einen Schaltk reis, der dies e Rechnung umsetzt. Außerdem besitzt

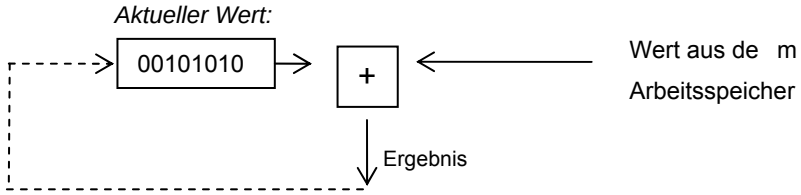


Abb. 11 Addition im Rechenwerk

es einen k leinen internen Speic herbaustein, der den Wert speichert, der gerade im Rechenwerk ist. Dies ist notwendig, da ein Prozessor immer nur einen Speicherzugriff durchführen kann. Für die A ddition von zwei Zahlen wird als o erst die eine Zahl in das Rechenwerk geholt (lade). Im nächst en Befehl wird dann die andere Zahl geholt und die eigentliche Rech nung durchgeführt (addiere, subtrahiere). Das Ergebnis landet erst einmal wieder im Rechenwer k, bis ein Bef ehl „speichere“ dafür sorgt, dass der aktuelle Rechenwerk-Wert in den Arbeitsspeicher übertragen wird.

Aufgabe XIII Ordnen Sie die folgenden Begriffe dem Steuerwerk oder Rechenwerk oder beidem zu:
Steuersignal, Subtra ktion, elekt rischer Schalter, Schaltkreis, binärer Pro grammcode, re chnen, bi näre Zahlen, Programmausführung, kontrollieren

Wie nun ein komplettes Programm durch die gerade vorgestellten Bestandteile de s Prozessors verarbeitet wird, wird im Folgenden erläutert.

4.3. Ausführung von Programmen

Wird ein Programm gestartet, wird es zunächst von der Festplatte in den Arbeitsspeicher kopiert. Dort stehen dann alle Befehle in nacheinander liegenden Speicherzellen. Die Verarbeitung des Programms funktioniert so, dass zuerst der erste Befehl aus dem Arbeitsspeicher in das Steuerwerk geholt und ausgeführt wird. Dann wird der nächste Befehl geholt und ausgeführt und so weiter.

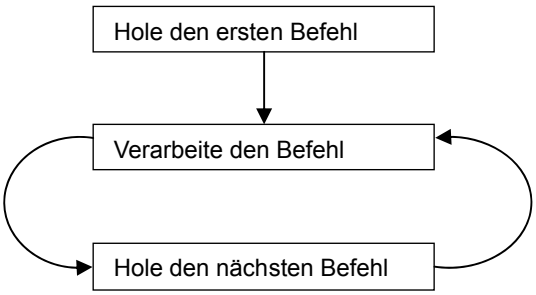


Abb. 12 Kreislauf der Programmverarbeitung

Wenn ein Befehl in das Steuerwerk kommt, schickt es die entsprechenden Steuersignale an Rechenwerk und Arbeitsspeicher. Ist dies „lade“ oder „speichere“, werden Daten zwischen dem Rechenwerk und dem Arbeitsspeicher ausgetauscht. Ist es „addiere“, werden Daten aus dem Arbeitsspeicher ins Rechenwerk geholt und zum aktuellen Wert addiert.

Als Beispiel für die Ausführung eines Programms durch den Prozessor wird nun gezeigt, in welcher Abfolge das Programm, das den Mittelwert berechnet, verarbeitet wird. Die Programmbefehle waren in Kapitel 2.1 vereinfacht dargestellt, indem Variablen a, b und c benutzt wurden. Hier sind sie nun genauer abgebildet: der Prozessor kann nicht mit Variablen arbeiten, seine einzige Möglichkeit, auf den Arbeitsspeicher zuzugreifen, ist über Adressen. Nehmen wir an, die Variablen a, b und c hätten im Speicher die Adressen 501, 502 und 503. Das Programm sähe dann im Arbeitsspeicher so aus:

Speicheradresse	Speicherzelle und -inhalt			Kommentar
501		S		peicherplatz für Variable a
502		S		peicherplatz für Variable b
503		S		peicherplatz für Rückgabewert c
504	Input	501	Eingabe	der ersten Zahl, speichern in a bzw. 501
505	Input 502			Eingabe der zweiten Zahl, speichern in b bzw. 502
506	Lade 501			Lade a in das Rechenwerk
507	Addiere	502	Addiere	b
508	Dividiere#	2	Dividiere	durch 2
509	Speichere 503			Speichere den Wert aus dem Rechenwerk in c
510	Output	503	Ergeb	nis c wird ausgegeben

Tab. 5 Programm im Speicher

Beim Aufruf des Progr amms werden der er ste Befehl und dann die nächsten geholt und ausgeführt. Angenommen, bei der Eingabe werden die Zahle n 22 und 16 eingegeben, dann sieht der Ablauf wie folgt aus:

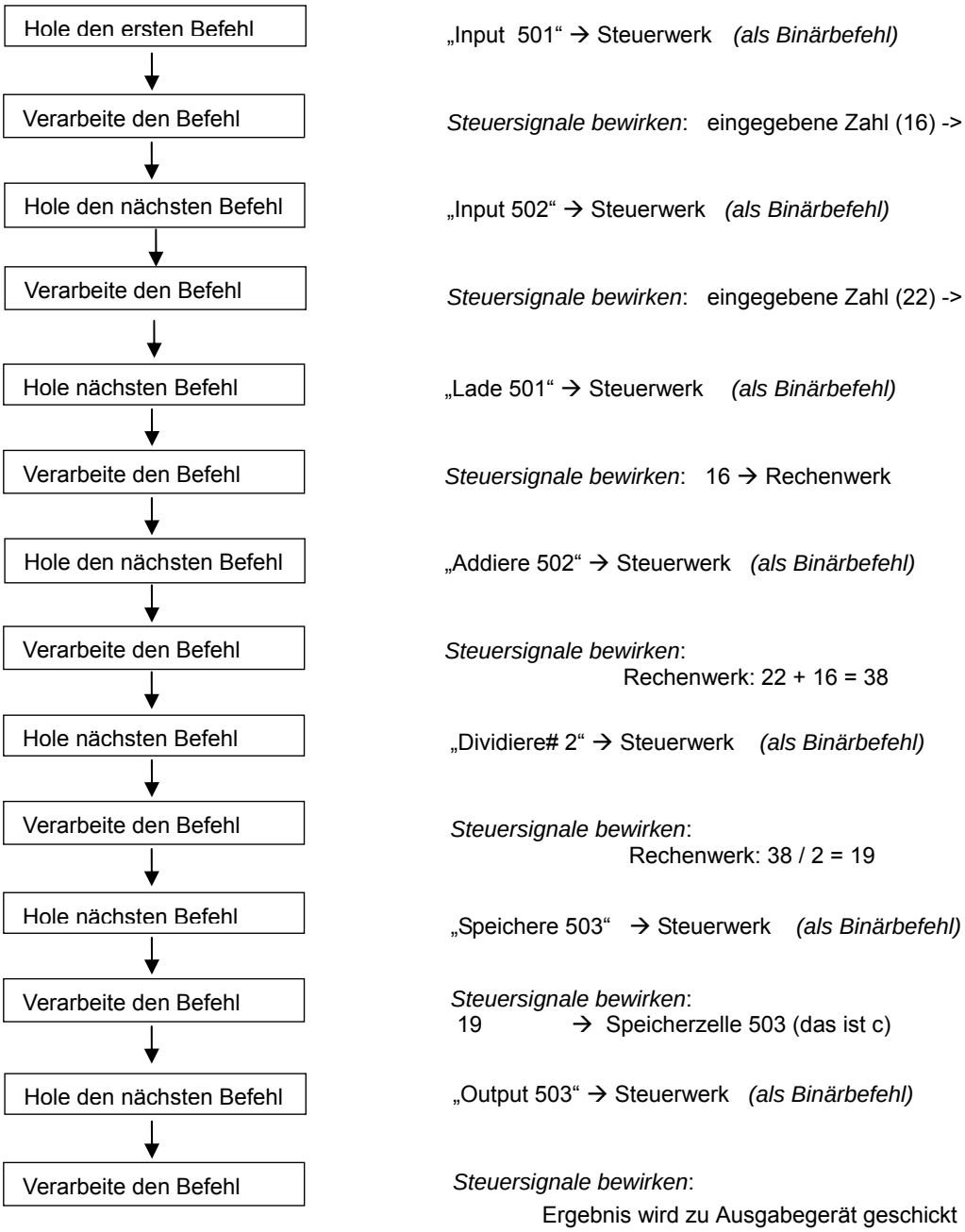


Abb. 13 Ausführung eines Programms

Aufgabe XIV Stellen Sie das Programm aus Aufgabe V dar, wie es im Speicher steht und zeichnen Sie die Verarbeitung durch den Prozessor wie in der vorigen Abbildung auf!

Aufgabe XV Inwiefern ist die Vera rbeitung der binären Maschinenbefehle schon vor dem e igentlichen Prozess der Verarbeitung vorherbestimmt worden? Begründen Sie!

Hier wird noch einmal der um die Informati onen aus diesem vierten Kapitel ergänzte Aufbau des Computers abgebildet. Danach werden die Informationen aus den bisherigen Kapiteln in einen alltäglichen Vorgang am Computer eingebettet.

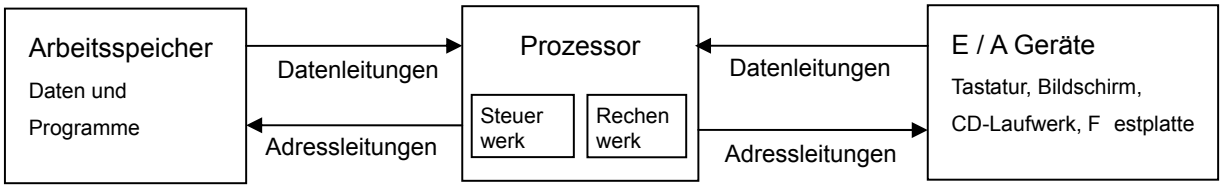


Abb. 14 Aufbau eines Computers (erweitert)

5. Was passiert, wenn ich etwas eintippe?

Wie hängen nun Programme, der Aufbau des Computers und die Binärkodier ung mit dem zusammen, was wir im Allt ag kennen? Betrachten wir dafür die folgende Situati on: Der Computer ist an und es ist ein Dok ument mit einem Textverarbeitungsprogramm geöffnet worden.

Was passiert, wenn ich nun etwas eintippe?

Wenn jemand in einem Textverarbeitungsprogramm einen Buchstaben eintippt, wird ein Teilprogramm gestartet, das den eingetippten Buchstaben im Arbeitsspeicher speichern und die Anzeige auf dem Bilds chirm ändern soll. Damit das Teilprogramm verarbeitet werden kann, wird dessen Maschinenc ode von der Festplatte in den Arbeitsspeicher kopiert. Irgendwann hat sich jemand den Algorithm us fü r dieses Teilprogramm überlegt und den Progra mmcode in einer höheren Programmiersprache geschrieben. Das wur de von einem Übersetzungsprogramm in Maschinensprache übersetzt und binär kodiert auf der Festplatte gespeichert.

Wenn das Programm dann startet, wird der Maschinencode in seiner binären Form Befehl für

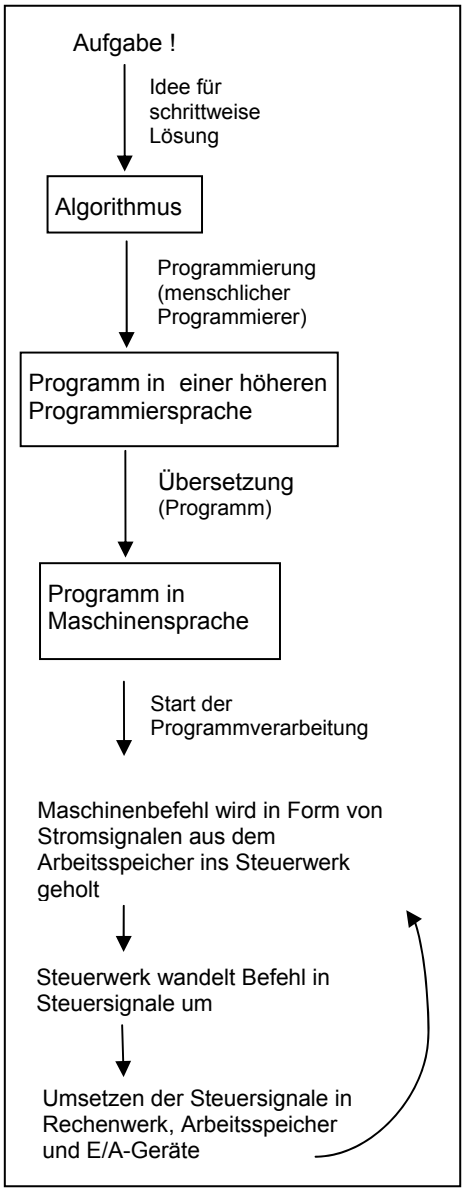


Abb. 15 Von der Aufgabe zur befehlsweisen Programmausführung

Befehl durch Stromsignale aus dem Arbeitsspeicher an das Steuerwerk übertragen und durch Steuerwerk und Rechenwerk verarbeitet. Das Textverarbeitungs-Teilprogramm enthält einen Input-Befehl, der den Buchstaben von der Tastatur als Binärkombination in den Prozessor lädt und einen Speichere-Befehl, der den Buchstaben als Binärkombination in den Arbeitsspeicher überträgt. Außerdem enthält das Teilprogramm einen Output-Befehl, der dafür sorgt, dass der Buchstabe auf dem Bildschirm angezeigt wird. Das Programm wird im Bruchteil einer Sekunde vom Prozessor verarbeitet. So schnell, dass es uns scheint, als ob der Buchstabe, den wir eintippen, sofort auf dem Bildschirm erscheint.

Aufgabe XVI Erläutern Sie anhand des Programms aus Aufgabe III die Stationen eines Programms von der Aufgabenstellung bis einschließlich zur Ausführung des Programms durch den Prozessor.

6. Exkurs: Sind Computer intelligent?

Es gibt das Forschungsgebiet der Künstlichen Intelligenz in der Informatik. Dort wird daran gearbeitet, Computern intelligentes Verhalten wie das Verstehen von menschlicher Sprache, das Erkennen von Gesichtern oder logisches Schlussfolgern einzuprogrammieren. Das wirkt auf uns intelligent. Doch wenn der Computer intelligentes Verhalten zeigt, liegt das daran, dass jemand den Computer so programmiert hat und nicht daran, dass er ein Bewusstsein dafür hat, was er tut. Computer verstehen auch die Programmbefehle, die wir geben, nicht in ihrer Wortbedeutung, wie wir sie verstehen. Dies scheint intuitiv dem zu widersprechen, was wir für intelligent halten.

Die Antwort auf die Frage, ob Computer intelligent sind, hängt von der Definition von Intelligenz ab. Philosophen streiten sich schon seit Jahrhunderten um die Frage, was Intelligenz ausmacht. Sie scheint schwer zu beantworten zu sein. Wenn man es als intelligent bezeichnet, intelligentes Verhalten zu zeigen, z.B. eine Aufgabe zu lösen oder auf andere den Eindruck zu erwecken, einen eigenen Willen zu haben, haben Computer dies bis zu einem gewissen Grad bereits erreicht.

Ein anderer Intelligenzbegriff besagt, dass zu Intelligenz auch Bewusstsein gehört. Das ist auch der Punkt, an dem wir dem Computer die Intelligenz absprechen, da er alles nur tut, weil es programmiert worden ist. Wir wissen allerdings wenig darüber, was Bewusstsein ist und wie es entsteht. Der Vorgang, dass das Steuerwerk auf bestimmte Signale von Nullen und Einsen hin Steuersignale sendet und somit andere Schalter im Computer verändert, ist aus unserer Sicht ein physikalischer Vorgang. Es könnte aus philosophischer Sicht möglich sein, dass irgendwo in diesen elektrischen Verbindungen ein Bewusstsein entsteht, das „weiß“, dass nun z.B. der

Buchstabe „n“ auf dem Bildschirm erscheinen sollte, so wie wir ja auch die Vorgänge in unserem Gehirn nicht kennen und trotzdem wissen, dass wir nun „Hallo“ sagen wollen.

Allerdings funktionieren Gehirn und Prozessor und Speicher sehr unterschiedlich, wie im Studientext deutlich geworden ist. Es ist unwahrscheinlich, dass im Computer Bewusstsein vorhanden ist. Man kann weder mit Sicherheit sagen, dass Computer irgendwann ein Bewusstsein besitzen werden, noch kann diese Möglichkeit ausgeschlossen werden. Im Moment scheint es aber so, dass Computer kein Bewusstsein besitzen.

Aufgabe XVII Versteht ein Computer Maschinensprache?

a. Nein, aber er versteht höhere Programmiersprache.
b. Nein, er kann gar nichts verstehen, nur Schalter schalten und Strom fließen lassen.
c. Ja, das Steuerwerk kann Maschinenbefehle in Form von Stromsignalen umsetzen.

Literatur:

Die Bücher, die für eine weitere Beschäftigung mit den im Studientext behandelten Themen gut geeignet und für Laien verständlich sind, sind mit einem * markiert.
Für eine weitere Beschäftigung mit dem Thema der Programmierung ist besonders „Algorithmik“ von Harel sehr zu empfehlen. Um sich genauer mit den Schaltkreisen im Prozessor zu befassen, ist das vierte Kapitel aus dem Buch „Informatik“ (ab S.150) hervorragend geeignet.

Engelmann, Lutz, [Hrsg.]. 2002. *Informatik bis zum Abitur.* Berlin : paetec Gesellschaft für Bildung und Technik mbH, 2002.

*** Goldschlager, Les und Lister, Andrew. 1990.** *Informatik.* München [u.a.] : Carl Hanser Verlag; Prentice-Hall International, 1990.

Gookin, Dan. 2010. *PCs für Dummies.* Weinheim : WILEY-VCH Verlag CmbH & Co. KGaA, 2010.

*** Harel, David. 2010.** *Algorithmik.* Heidelberg [u.a.] : Springer-Verlag, 2010.

Harel, David. 2002. *Das Affenpuzzle.* Berlin [u.a.] : Springer-Verlag, 2002.

Hattenhauer, Rainer. 2010. *Informatik für Schule und Ausbildung.* München [u.a.] : Pearson Schule, 2010.

Lange, Christian. 2004. *Einführung in die PC-unterstützte Datenverarbeitung.* Ludwigshafen (Rhein) : Friedrich Kiehl Verlag GmbH, 2004.

Patterson, David A. und Hennessy, John L. 2005. *Rechnerorganisation und -entwurf.* Heidelberg : Elsevier GmbH, Spektrum akademischer Verlag, 2005.

Precht, Manfred, Meier, Nikolaus und Tremel, Dieter. 2004. *EDV-Grundwissen.* München [u.a.] : Addison-Wesley Verlag, 2004.

*** Rechenberg, Peter. 2000.** *Was ist Informatik.* München [u.a.] : Carl Hanser Verlag, 2000.

4. Evaluation des Studientextes durch fachfremde TestleserInnen

Zu dem im Hauptteil entwickelten Studientext wurden Rückmeldungen von fünf fachfremden Studierenden eingeholt. Diese TestleserInnen haben zunächst den Studientext gelesen und dann einen Fragebogen zur Evaluation des Studientextes beantwortet. Der Fragebogen und die Antworten sind in Anhang II zu finden. Die Antworten der verschiedenen TestleserInnen zu einer Frage sind zusammen aufgelistet. Dabei ist jedem der TestleserInnen ein Buchstabe zugeordnet, alle Antworten mit dem gleichen Buchstaben stammen also von der gleichen Person.

Bei der Evaluation ist zu beachten, dass die TestleserInnen nicht an der Vorlesung IuG teilgenommen haben. Daher können aus der Evaluation nur Rückschlüsse über den Studientext an sich gezogen werden, nicht aber über dessen Hilfestellung bei der Bearbeitung der Studienbriefe der Vorlesung IuG.

Im Folgenden werden die Fragen und Antworten anhand von drei Gesichtspunkten beleuchtet: Inhalte des Studientextes, Umfang und Verständlichkeit.

Der erste Aspekt behandelt die Fragen, ob die Erwartung der Studierenden an die Inhalte des Studientextes erfüllt wurde, was sie sich noch an ergänzenden Inhalten gewünscht hätten und was aus ihrer Sicht aus dem Studientext entfernt werden könnte. Auf diese Fragestellung waren die erste, fünfte und neunte Frage ausgerichtet. Aus den Antworten ging hervor, dass die Erwartungen der TestleserInnen bezüglich der Inhalte insgesamt erfüllt worden sind.

Die Wünsche zur Erweiterung der Inhalte waren bei den Studierenden unterschiedlich: TestleserIn a hätte sich mehr Erklärung und Einschätzung der aktuellen Computeranwendungen gewünscht und eine bessere Erklärung davon, wieso Computer auch manchmal nicht funktionieren. TestleserIn d hätte es interessiert, noch komplexere Inhalte über die Funktionsweise von Computern zu lernen, wie die genaue Funktionsweise des Rechenwerks oder die Darstellung von bewegten Bildern. TestleserIn e hätte gerne noch etwas zu den weiteren Komponenten der Hardware erfahren. Da die Erweiterungswünsche der Studierenden sehr unterschiedlich sind, ist es kaum möglich, sie in den Studientext einzuarbeiten und seinen Umfang beizubehalten. Sie können aber als Anregung für weitere ähnliche Arbeiten dienen.

Auch die Einschätzung dessen, was nicht unbedingt behandelt werden müsste oder weniger ausführlich erklärt werden könnte, war bei den Studierenden verschieden. TestleserIn a nannte die Möglichkeit, die Darstellung des Prozessors zu vereinfa-

chen. Aus Sicht von TestleserIn b könnte das Kapitel zur künstlichen Intellig enz entfernt werden ohne dass die Funktionsweise des Computers schl echter verstanden wird. Die Inhalte des Einf ührungskapitels waren Testle serIn e schon beka nnt. Diese Anregungen können hilfreich sein, wenn ei n Kürzen des Studien textes in Frage kommt, sind aber nic ht zwingend anzuwen den, besonders da die Einschät zung der Studierenden bezüglich der zu kürzenden Inhalte variiert.

Ein zweiter Gesichtspunkt ist die Frage nach dem Umfang der Arbeit. Dies konnte aus den Antworten zu den Fragen 5 und 6 entnommen werden. Insgesamt fanden die TestleserInnen den Umfang des Textes in Ordnung. TestleserIn b merkte jedoc h an, die Konzentration hätte zum Ende hin nachgelass en und Te stleserIn c bearbeitete die Aufgaben aus Zeitgründen nicht. Das deutet darauf hin, dass es sinnv oll sein kann, den Studientext es in Et appen zu bearbeiten. Inhaltl ich k ann vor dem dritten oder vierten Kapitel eine Pause gemacht werden.

Ein dritter Gesichtspunkt in den Antwort en und Fragen war, wie verständlich der Studientext ist und was für das Verständnis hi lfreich oder hindernd ist. Dies ging vor allem aus den Antworten zu den Fragen 2, 3, 4, 6 und 7 hervor. Die meisten der TestleserInnen fanden den Studi entext ins gesamt verständlich. Es kann dav on ausgegangen werden, dass den Studi erenden Inhalte bes onders im Gedächtnis bleiben, wenn sie diese verstanden haben. Daher ist das Ergebnis der vierten Frage für eine Einschätzung der Verständlichk eit der einz elnen Inhalte sehr interessant. Daraus ging hervor, dass die meisten der Studier enden sich die Erklärungen der Funktionsweise der einzelnen Bestandteile des Computers besonders gut merken konnten. Die Verarbeitung der Befehle und der Arbeitsspei cher wurden jeweils von zwei TestleserInnenn erwähnt. Auch die meisten der anderen I nhalte wurden von einzelnen TestleserInnen erwähnt, wie die Adress- und Datenleitungen, die Binärkodierung, die Determiniertheit des Computers und die Au sführungen zu KI. Da ss die Ha uptinhalte des Studientextes dabei erwähnt werden, deutet darauf hin, dass die Inhalte insgesamt verständlich waren. Die Inhalte des zweiten K apitels, Programme und Programmiersprachebenen, wur den in den Antworten ni cht genannt. Das kann jedoch daran liegen, dass dieses Thema in der Nennung der anderen Themen impliziert wurde oder dass es zu Beginn des Studientextes behandelt wurde.

Auch einige andere Inhalte wur den weniger gut verstanden. An di eser Stelle berichteten zwei der TestleserInnen, die Notw endigkeit der Umwand lung der Maschinenbefehle zu Steuersignalen aus Kapitel 4.2.2 nicht gut verstanden zu haben. Jeweils

eine Person benannte die Kapitel 4.3 und 3 als schwierig verständlich. Die Kapitel 3, 4.2.2 und 4.3 können also noch verbessert werden.

Einige Aufgaben wurden von den Studierenden als zu einfach und daher nicht hilfreich für die Bearbeitung des Studientextes eingeschätzt und zwei Aufgaben waren für jeweils eine Person schwer verständlich. Die beiden Aufgaben, die als schwer verständlich angemerkt wurden, sind Aufgaben, die schon eine Verbindung zur Vorlesung IuG aufbauen. Daher sollten sie im Studientext enthalten bleiben, könnten aber noch mal auf Verständlichkeit hin überarbeitet werden. Es ist nicht unbedingt notwendig, die Aufgaben, die als zu einfach eingestuft wurden zu überarbeiten, da die Bearbeitung der Aufgaben optional ist. Somit können die LeserInnen die Aufgaben, die ihnen zu einfach erscheinen, überspringen.

Als hilfreich für die Verständlichkeit des Studientextes erwähnten die meisten TestleserInnen Analogien zum Alltag und bildliche Beispiele. Zwei der Studierenden bemerkten zudem, dass die Sprache des Studientextes für Laien gut zu verstehen war. Die TestleserInnen, die die Aufgaben bearbeitet haben, fanden viele Aufgaben gut um die Arbeitsweise des Computers zu verstehen bzw. das Wissen anzuwenden. Dies bestätigt insgesamt die didaktischen Überlegungen, die zu diesen Punkten in Abschnitt 2.4 vorgenommen wurden.

5. Fazit

In dieser Arbeit wurde ein Studientext entwickelt, der das Verständnis der Vorlesung „Informatik und Gesellschaft“ für Fachfremde erleichtert. Der Studientext bezieht sich dabei auf zwei Studienbriefe der ersten beiden Vorlesungseinheiten, die ohne Informatikkenntnisse schwer zu verstehen sind. Es werden in diesen Studienbrief Überlegungen angestellt, die für fachfremde Studierende nicht bis in die Tiefe verstanden werden können.

In der Entwicklung des Studientextes wurde deutlich, dass beim Verfassen von Texten für Fachfremde auf einem sehr einfachen Niveau erklärt werden muss. Da es um sehr umfangreiche Themen geht müssen viele Details weggelassen werden, die für InformatikerInnen wichtig sind. Die Ebene des Studientextes ist also eine ganz andere als die, die in Fachkreisen diskutiert wird. Dies stellte vor allem bei der Erklärung der Verarbeitung von Programmen und des Aufbaus der Hardware eine Herausforderung dar. Dies wurde in diesem Fall durch eine einfache Erklärung der Themen auf zwei Ebenen, der Modellebene und der elektronischen Ebene, gelöst.

Eine weitere wichtige Erkenntnis ist die, dass die Studierenden eigene Vorstellungen aus dem Alltag mitbringen, die mit dem was im Studientext erklärt wird in Konflikt treten könnten und daher miteinbezogen werden müssen. Dazu gehört im Studientext vor allem, den Bezug zwischen „Programm“ und „speichern“ aus dem Alltag und den entsprechenden Begriffen aus der Informatik herzustellen. Außerdem scheinen Bezüge zum Alltag das Verständnis der Inhalte stark zu fördern, was auch durch die Evaluation in Abschnitt 4 bestätigt wurde.

Im Studientext wurde wegen der Prioritätensetzung auf Inhalte, die für die Vorlesung „Informatik und Gesellschaft“ relevant sind, auf eine genauere Erläuterung der Steuerung des Computers durch Programme verzichtet. In einer weiteren Arbeit könnte auf die Methoden der Algorithmik oder Softwareentwicklung eingegangen werden.

Aus der Evaluation des Studientextes ist hervorgegangen, dass die fachfremden Studierenden die Inhalte des Studientextes insgesamt gut verstanden haben. Die Erklärung der Umwandlung von Befehlen zu Steuersignalen in Kapitel 4.22 könnte noch überarbeitet werden, da zwei der TestleserInnen angaben, die Notwendigkeit der Umwandlung nicht verstanden zu haben. Einige der TestleserInnen gaben Themen an, über die sie gerne noch mehr erfahren hätten. Dazu gehören die Frage, woran es liegt, wenn Computer nicht funktionieren und eine noch genauere Erklärung der Funktionsweise von Computern. Diese Themen könnten in weiteren Arbeiten erklärt werden.

Da die Evaluation lediglich eine allgemeine Rückmeldung zum Studientext gibt, wäre es möglich, in einer weiteren Evaluation zu ermitteln, ob der Studientext für die Bearbeitung der Studienbriefe der Vorlesung IuG hilfreich ist.

Bei der Literaturrecherche für die Bachelorarbeit war auffällig, dass es wenig Literatur gibt, die darauf abzielt, Laien die Funktionsweise von Computern näher zu bringen. Dies ist erstaunlich, wenn man bedenkt, dass fast jeder Mensch in Deutschland Computer benutzt. Das Verständnis der Funktionsweise von Computern zu fördern, könnte Inhalt weiterer Studienarbeiten oder ganzer Bücher sein.

Literaturverzeichnis

Engbring, Dieter. o.J. Studienbrief zur Kontextuellen Informatik. In *: Informatik und Gesellschaft.* Skript und Materialien zur gleichnamigen Vorlesung im Wintersemester 2009/2010, Universität Münster.

Engelmann, Lutz, [Hrsg.]. 2002. *Informatik bis zum Abitur.* Berlin : paetec Gesellschaft für Bildung und Technik mbH, 2002.

Goldschlager, Les und Lister, Andrew. 1990. *Informatik.* München [u. a.] : Carl Hanser Verlag; Prentice-Hall International, 1990.

Gookin, Dan. 2010. *PCs für Dummies.* Weinheim : WILEY-VCH Verlag CmbH & Co. KGaA, 2010.

Harel, David. 2010. *Algorithmik.* Heidelberg [u.a.] : Springer-Verlag, 2010.

Harel, David. 2002. *Das Affenpuzzle.* Berlin [u.a.] : Springer-Verlag, 2002.

Hattenhauer, Rainer. 2010. *Informatik für Schule und Ausbildung.* München [u.a.] : Pearson Schule, 2010.

Keil-Slawik, Reinhard. 2001. Von Informatik und Gesellschaft zum Kontext der Informatik. In: *FifF-Kommunikation Nr. 4/2001 Informatik und Gesellschaft als akademische Disziplin.* 2001.

Lange, Christian. 2004. *Einführung in die PC-unterstützte Datenverarbeitung.* Ludwigshafen (Rhein) : Friedrich Kiehl Verlag GmbH, 2004.

Patterson, David A. und Hennessy, John L. 2005. *Rechnerorganisation und -entwurf.* Heidelberg : Elsevier GmbH, Spektrum akademischer Verlag, 2005.

Precht, Manfred, Meier, Nikolaus und Tremel, Dieter. 2004. *EDV-Grundwissen.* München [u.a.] : Addison-Wesley Verlag, 2004.

Rechenberg, Peter. 2000. *Was ist Informatik.* München [u.a.] : Carl Hanser Verlag, 2000.

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang I: Aufgabenlösungen

Aufgabe I

Hardware	USB-Stick, CPU, DVD-Brenner, Datenleitung, Drucker
Anwendungssoftware	Programm, Word
Systemsoftware	Programm, Betriebssystem, Rechnen

Aufgabe II

Eingabe	Mikrofon, Touchscreen
Ausgabe	DVD-Brenner, Touchscreen, Drucker
Datenverarbeitung	CPU, Subtraktion, Rechnen, Programmausführung
Datenspeicherung	Speicher
Datenübertragung	Datenleitung

Aufgabe III Durch einen Algorithmus lassen sich die Tätigkeiten a, b, d, f, und g beschreiben, da für diese Vorgänge Anleitungen aufgestellt werden können.
Die anderen Punkte sind nicht durch einen Al gorithmus zu beschreiben, da sie nicht nach festen Regeln ablaufen und man keine Anleitung der Form 1. 2. 3. formulieren kann.

Aufgabe IV Der Computer k ann a, d und f ausführen, da es in diesen Vor gängen um Zahlen und Buchstaben geht.

Aufgabe V Es gibt verschiedene Lösungswege. Zwei davon sind hier Aufgeschrieben. Wenn Ihre Lösung in etwa einem dieser beiden Wege entspricht, haben Sie gute Arbeit geleistet.

Eine mögliche Lösung:

Input x
Input y
Lade x
Multipliziere x
Speichere a
Lade y
Multipliziere y
Speichere b
Lade a
Addiere b
Speichere c
Output c

Eine andere mögliche Lösung:

Input x
Input y
Lade x
Multipliziere x
Speichere a
Lade y
Multipliziere y
Addiere a
Speichere c
Output c

Aufgabe VI Input x, y
$c = x^2 + y^2$
Output c
Auch hier ist es ausreichend, wenn Sie eine ähnliche Lösung gefunden haben.

Aufgabe VII
Die Übersetzung ist in feste Regeln gefass t: Zu jedem Befehl der höheren Programmiersprache wurde bei der Erstellung des Übersetzungsprogramms schon festgelegt, in welche Maschinenbefehle er übersetzt wird. Somit ist schon vorherbestimmt, welcher Maschinencode durch die Übersetzung entsteht.

Aufgabe VIII
0011 steht für die Zahl 3 und den Maschinenbefehl „Subtrahiere“.
0100001 ist das Zeichen „!“.
„DA!“ ist in Binärschreibweise 1000100 1000001 0100001.
Die Zahl „11“ ist in Binärschreibweise 1011.
Der Maschinenbefehl „Speichere“ ist in Binärschreibweise 0001.

Aufgabe IX Binär addiert ist 0100+0011 = 0111 und es gilt 0100 = 4, 0011 = 3 und 0111=7.

Aufgabe X Der Prozessor kann schneller auf den Arbeitsspeicher zugreifen.

Aufgabe XI Um den Speicherinhalt aus ei ner Speicherzelle abzurufen, muss dem Speicher die entsprec hende Adresse vermi ttelt werden. Er gibt dann den zugehörigen Speicherinhalt zur ück. Das geschieht, i ndem die Adresse in binärer Form über die Adressleitungen z um Speicher geschic kt wird. Die Speic herzelle, die die richtige Adresse hat, reagiert und sendet ihren Inhalt (in binärer Form) über die Datenleitungen zurück.

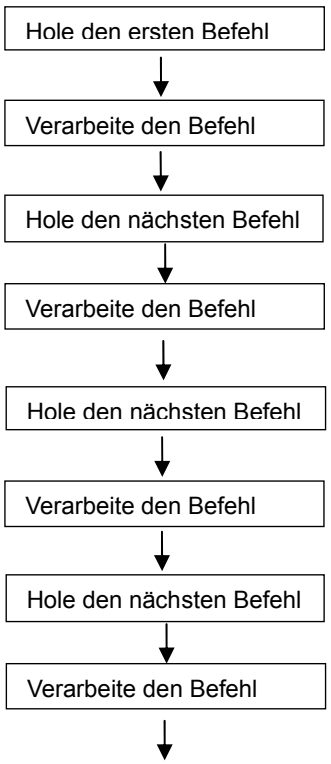
Aufgabe XII Man kann eine Kopie des Inhalts des Arbeitsspeichers erstellen, indem man den Inhalt aller Speicherzellen auf der Festplatte speichert. Um den Arbeitsspeicher dann zurückzusetzen muss dieses Speicherabbild wieder in den Arbeitsspeicher zurück kopiert werden.
Das passiert zum Beispiel, wenn der Co mputer in den Standby-Modus oder Ruhezustand versetzt wird.

Aufgabe XIII

Steuerwerk	Steuersignal, binärer Pr ogrammcode, Programmausführung, kontrollieren
Rechenwerk	Subtraktion, rechnen, binäre Zahlen
Beides	elektrischer Schalter, Schaltkreis

Aufgabe XIV Für diese Lösung wir d das Maschinenprogramm aus der zweiten Lösungsmöglichkeit von Aufgabe V verwendet . Wenn Sie ein anderes Maschinenprogramm gewählt haben, kann auch Ihre Lösung anders aussehen.

Speicherzelle	Inhalt in der Speicherzelle			Kommentar
….				
01		S		peicherplatz für Variable x
02		S		peicherplatz für Variable y
03		S		peicherplatz für Zwischenergebnis a
04		S		peicherplatz für Rückgabewert c
05	Input	01	x	wird eingegeben
06	Input	02	y	wird eingegeben
07	Lade	01	Lade	x
08		Multipliziere 01		Berechne x*x = x²
09	S	peichere 03	S	peichere in 403
10	Lade	02	Lade	y
11	Multipliziere	02	Berechne	y*y=y²
12		Addiere 03		Addiere a dazu
13		Speichere 04		Speichere das Ergebnis in c
14	Output	04	Gebe	c zurück
….				



„Input 01“ → Steuerwerk *(als Binärbefehl)*

Steuersignale bewirken: eingegebene Zahl (4) -> 01

„Input 02“ → Steuerwerk *(als Binärbefehl)*

Steuersignale bewirken: eingegebene Zahl (3) -> 02

„Lade 01“ → Steuerwerk *(als Binärbefehl)*

Steuersignale bewirken: 4 -> Rechenwerk

„Multipliziere 01“ → Steuerwerk *(als Binärbefehl)*

Steuersignale bewirken:
Rechenwerk: 4 x 4 = 16

Aufgabe XV Jede Binärkombination (jeder Maschinenbefehl) wird auf eine eindeutige Weise durch das Steuerwerk in Steuersignale umgewandelt. Wie das geschieht, ist von den Schaltkreisen des Steuerwerks abhängig und wurde beim Bau des Prozessors festgelegt. Also ist schon vor dem Prozess der Verarbeitung eines konkreten Programms vorherbestimmt, wie es verarbeitet wird.

Aufgabe XVI Zuerst gibt es die Aufgabe, zwei Zahlen jeweils zu quadrieren und zu addieren. Der Algorithmus dafür sieht so aus, dass erst die eine Zahl quadriert wird, dann die andere und zuletzt die Quadratzahlen addiert werden.
Der Programmcode in höherer Programmiersprache wird von einem Programmierer geschrieben (Vgl. Aufgabe VI). Das Übersetzungsprogramm verwandelt ihn dann in Maschinencode (Vgl. Aufgabe V). Wenn das Programm gestartet wird, wird es in den Arbeitsspeicher übertragen (Vgl. Aufgabe XIV). Dann werden die Befehle nacheinander an das Steuerwerk gesendet. Das setzt sie in Steuersignale um, die das Rechenwerk und den Arbeitsspeicher dazu veranlassen die entsprechenden Befehle durchzuführen (Vgl. Aufgabe XIV).

Aufgabe XVII

Es ist falsch, a anzukreuzen, denn der Prozessor kann die höhere Programmiersprache nicht umsetzen.

Die anderen Positionen kann man alle vertreten. Das Steuerwerk erhält die Maschinenbefehle in binärer Form durch Stromsignale und setzt sie in Steuersignale um, die Rechenwerk und Arbeitsspeicher kontrollieren. Wie jeder Maschinenbefehl verarbeitet wird, ist in den Schaltkreisen festgelegt. Der Computer „versteht“ sie nicht als Worte (so kann man b begründen), aber er versteht sie insofern, als er die Stromsignale zu anderen Stromsignalen, den Steuersignalen, verarbeitet (so kann man c begründen).

Anhang II: Evaluationsfragebogen und Antworten

Der Fragebogen:

1. Hat der Studientext die Inhalte vermittelt, die du dir erhofft hast?	
Wenn nicht: Was hat gefehlt? Was war zu viel?	
2. Welche Erklärungen waren für dich gut zu verstehen?	
3. Welche Erklärungen waren weniger gut zu verstehen?	
4. Welche Inhalte sind dir besonders in Erinnerung geblieben?	
5. Waren die Bearbeitungszeit und der Umfang des Textes in Ordnung?	
Wenn nicht: Hast du eine Idee dazu, welche Inhalte man weggelassen könnte?	
6. Wenn du die Aufgaben bearbeitet hast, was hast du dazu für Anmerkungen?	
7. Was hat dir insgesamt gut gefallen?	
8. Was hat dir insgesamt nicht so gut gefallen?	
9. Was hättest du dir noch gewünscht?	
Hier ist Platz für weitere Anmerkungen:	

Die Antworten der TestleserInnen

Frage 1: Hat der Studientext die Inhalte vermittelt, die du dir erhofft hast?

a) Im wesentlichen Ja.

b) Ja.

c) Ja, vor allem die physikalischen Strukturen hatten mich interessiert.

d) Ja, aber (weiter bei der nächsten Frage)

e) Computer mit ihren B estandteilen (Hardw are) sowie mit ihrer Sof tware sind sehr komplex. Die Einführungen am Anfang kannte ich schon, so dass ich hier nic hts Neues gelernt habe. Die weiteren Themen haben mir viel Neues gezeigt.

Wenn nicht: Was hat gefehlt? Was war zu viel?

a) Gefehlt hat mir ein bisschen der Bez ug zur praktischen Arbeit mit Computern, also, wo s ind s ie demnach besonders gut einsetzbar, wo eher weniger. Ist technisches oder menschlic hes Versagen w ahrscheinlicher bei heiklen Aufgaben (Flugzeug, AKW,...). U nd vor allem, waru m stürzen die Dinger ständig ab, wenn sie nur nach festen Regeln einfache Rechenoperationen durchführen...

b) –

c) –

d) Es hat meiner Meinung nach gef ehlt, wie das Rechenzentrum das Rechnen „gelernt“ hat. Außerdem würde mich noc h die Fra ge interes sieren, wie ein Computer zum Beispiel Bewegungen ver arbeitet (aber vielleicht ist das zu kompliziert).

e) Eben nur der Anfang.

Frage 2: Welche Erklärungen waren für dich gut zu verstehen?

a) Das allermeiste.

b) Es war eigentlich alles se hr gut erklärt. Besonders einfach fiel es mir z u verstehen, wenn bildlich oder verg leichend bes chrieben wurde (z.B. mit Briefkästen, Kuchen etc.)

c) Vor allem die, welche mit Beispielen aus dem Alltagsleben unterfüttert waren.

d) Vor allem die Erklärungen, wo der Comput er mit Alltagssituatio nen (wie z.B. dem Bäcker, dem Briefkasten oder dem Schreibtisch) beschrieben wurden.

e) Ich konnte alles gut verstehen.

Frage 3: Welche Erklärungen waren weniger gut zu verstehen?

a) Nicht so richtig ve rstanden habe ich das Steuerwe rk. Worin liegt denn der Unterschied zwischen Befehl und Steuersignal?

b) Das Kapitel 4.3. war für mich sc hwerer zu verstehen. Dies mag aber auch daran liegen, dass es am E nde des Studient extes steht und die Konzentration nachlässt.

c) Erklärungen ohne praktischen Bezug zum Alltagsleben.

d) Die Erklärung des Steuerwerks. Ic h habe nicht ganz verstanden, warum das Steuerwerk die ankommenden Befehle umwandeln muss.

e) Was ich nicht verstanden habe war die Binärkodierung (Kapitel 3).

Frage 4: Welche Inhalte sind dir besonders in Erinnerung geblieben?

a) Der Computer kennt nur Nullen und Einsen, die er nach festen R egeln verarbeitet. Die wichtigsten Bestandte ile sind E/A Geräte, D atenleitungen, Prozessor und Speicher. Der Comp uter ist dumm und es ist genau vorherbestimmt, was bei welcher Eingabe passiert.

b) Die Sache mit dem Arbeitsspeic her und die Erklärung der einz elnen Bestandteile (Prozessor etc.).

c) Adressleitungen und Datenleitungen.

d) Wie ein Computer speichert, wie ei n Computer Befehl e entgegen nimmt und wie der Ablauf innerhalb des PC vom perma nenten Speicher zu m Arbeitsspeicher zum Rechenzentrum abläuft.

e) Besonders der Aufbau des Prozessors (S teuer- und Rechenwerk). Interessant waren die Gedanken zu KI.

Frage 5: Waren die Bearbeitungszeit und der Umfang des Textes in Ordnung?

a) Die Bearbeitungszeit war okay, ich habe knappe 2 Stunden gebraucht.

b) Ich habe gemerkt, dass die Konzentra tion zum Ende nachgelass en hat. Daher war der Text evtl. Etwas lang.

c) Im Rahmen meines selbst ges etzten Zeitbudgets habe ich den Text durchgearbeitet und konnte mein Erkenntnisinter esse in diesem Zeitraum befriedigen.

d) Die Bearbeitungszeit war ungefähr 90-100 Min. Das war in Ordnung.

e) Grundsätzlich ja, wobei es auch drauf ankommt wie viel Wissen man in dieser Thematik hat.

Wenn nicht: Hast du eine Idee dazu, welche Inhalte man weggelassen könnte?

a) Was man weglassen könnte: Vielleic ht k ann man den Prozes sor vereinfachen und den Unterschied Steuerwerk / Rechenwerk weglassen.

b) Das Kapitel zur künstlichen Intelligenz ist int eressant aber für den Aufbau eines Computers und dessen Vorgehensweis e ni cht notwendig. Daher könnte man das weg lassen.

c) –

d) Aufgabe VI II war nicht unbedingt notwendi g, da man sich die Binärcodes nicht merken kann und einfach nur oben abschre iben musste. Ich denke es is t nur wichtig z u wissen, wie die Codes auf gebaut sind , aber man muss nicht die einzelnen Codes kennen.

e) –

Frage 6: Wenn du die Aufgaben bearbeitet hast, was hast du dazu für Anmerkungen?

a) Die meisten habe ich bearbeitet, einige war en mir zu aufwändig. Gut fand ich V, VI, VIII, X IV. Hier konnte ich das Wi ssen an konkreten Beispielen anwenden. Nicht so gut fand ich I,II,X und XIII. Das waren mult iple Choice Aufgaben, die jeder lösen kann, der ein gewis ses Text verständnis hat und kein Lerneffekt da war.

b) Ich habe die Aufgaben nicht bearbeitet.
c) Habe sie nur geistig überflogen, da ich ein begrenztes Zeitbudget hatte.
d) Frage VII habe ich nicht ve rstanden. Frage VIII fand ich nicht sehr sinnvoll. Die Aufgaben, bei denen man se lbst die Programme schreiben s ollte waren sehr interessant, da man sich so sehr gut in die Arbeitsweise eines PC eindenken konnte.
e) Gut waren die Aufgaben in denen man selber „programmi eren“ durfte z.B.: V und VI, obwohl ich immer noch viel nach schauen musste. Aufgabe XV fand ic h zu schwer und konnte sie nicht lösen.

Frage 7: Was hat dir insgesamt gut gefallen?

a) klare Struktur des Textes, anschauliche Beispiele
b) Dies Sprac he und der Schreibstil. S ehr sc hön geschrieben und ganz einfach. Außerdem fand ich die bildliche Darstellung, zu den einzelnen Kapiteln sehr gut
c) Erklärungen mit praktischem Bezug zum Alltagsleben
d) Die Arbeitsweise eines PC wurde durch anschauliche Be ispiele und Sprache, die auch Laien verstehen, erläutert.
e) Sehr gut waren die Beispiele z. B. am Anfang der Bäc ker der den Kuchen backt und der Vergleich mit den Algorithmen.

Frage 8: Was hat dir insgesamt nicht so gut gefallen?

a) Siehe 3.
b) Ich habe nichts zu bemängeln.
c) –
d) Es wurde nur auf die einf achsten Dinge, die ein PC ausführen muss eingegangen. Dies hängt aber wahrscheinlich damit zusammen, dass es sonst zu kompliziert geworden wäre.
e) Aufgabe XV.

Frage 9: Was hättest du dir noch gewünscht?

a) siehe 2. Etwas mehr Pr axisbezug, Kapitel 6 war als Praxisbezug ganz interessant, aber eben nur ein Thema bei der Arbeit mit Computern.
b) –
c) –
d) Siehe Punkt 8 und 1
e) Maximal noch kurze Erläuterungen zu anderer Hardware wie z.B. die Grafikkarte

Zusatz: Hier ist Platz für weitere Anmerkungen

a) –
b) Hat Spaß gemacht es zu lesen und ich habe etwas gelernt.
c) –
d) –
e) –

Erklärung der Studierenden

Hiermit versichere ich, dass ich die vorliegende Arbeit mit dem Titel

Entwicklung eines Studientextes „Wie funktionieren Computer?“ für die Vorlesung „Informatik und Gesellschaft“ in den Allgemeinen Studien

bzw.

Development of an essay „Wie funktionieren Computer?“ for the lecture „Informatik und Gesellschaft” in general studies

selbständig verfasst habe, und dass ich keine anderen Quellen und Hilfsmittel als die angegebenen benutzt habe und dass di e Stellen der Arbeit, die anderen Werken – auch elektronischen Medien – dem Wortla ut oder Sinn nach entnommen wurden, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht worden sind.

Ort, Datum	Unterschrift