

Projektseminar Rechnersysteme

Seminararbeit vorgelegt von (Robert Wiesner)

Betreuer: Dipl.-Ing. (Changgong Li)

Beginn: (Beginndatum) | Abgabe: (Abgabedatum)

Institut für Datentechnik | Fachgebiet Rechnersysteme | Prof. Dr.-Ing. Christian Hochberger



TECHNISCHE
UNIVERSITÄT
DARMSTADT



1 Aufgabenstellung

Entwicklung eines "Spill and Fill"-Mechanismus um den Framestack des AMIDAR Prozessors auf einen externen DDR3 Speicher auszulagern und so die maximal mögliche Stackgröße deutlich zu erhöhen. Dazu gehört das einrichten verschiebbarer "windows" die jeweils einen Ausschnitt des Stacks beinhalten. Bei einem Methodenaufruf, dessen Stackframe den aktuellen Ausschnitt überschreiten würde soll ein Teil der im Window enthaltenen Daten auf den externen Speicher ausgelagert werden um Platz für den neuen Stackframe zu schaffen. Bei einem Methoden Rücksprung sollen die Daten wieder zurück kopiert werden um den alten Stackframe wiederherzustellen. Es sollen 4 Windows angelegt werden um die Stacks von bis zu 4 threads vorzuhalten. Es soll die Möglichkeit einer Threadverdrängung realisiert werden, damit mehr als 4 Threads ausgeführt werden können.

2 Grundlagen

2.1 Spill and Fill

2.2 Verwendung eines Spill and Fill Mechanismus in anderen Prozessoren

2.2.1 Sun Sparc Prozessoren

Die Sun Sparc Prozessoren V8/V9 verfügen über ein sliding register Window mit jeweils 16 8 byte Registern in 7 Registersätzen. Sliding Window bezeichnet ein Verfahren, bei dem die Registersätze im Falle eines Funktionsaufruf nicht auf dem Stack gesichert werden müssen, stattdessen wird auf den nächsten Registersatz gewechselt. Dabei wird meistens auch die Parameterübergabe realisiert. Dabei sind die Input Register in dem Registerwindow des Callers identisch mit dem Output Registern, in dem Registerwindow des Callee. Mit speziellen Bytecode Instruktionen werden im Spill an Fill Verfahren Registersätze ausgetauscht, sobald die 7 Registersätze nicht mehr ausreichen, was allerdings wegen den großen Registersätzen recht viel Zeit in Anspruch nimmt.

2.3 AMIDAR

Bei der Klasse der AMIDAR Prozessoren handelt es sich um ein konfigurierbares System bestehend aus Function Units (FU) für die jeweiligen Aufgaben. Die FUs sind dabei untereinander mit einem Token- und einem Datennetzwerk verbunden.

2.3.1 Funktion

2.3.2 Framestack

3 Implementierung

3.1 DRAM Anbindung über AXI

AMIDAR ist über AXI mit 512 MB DDR Ram verbunden, der auf 800 MHz getaktet.

3.1.1 Addressbereich des Framestacks im Hauptspeicher

Eine Schwierigkeit bei der Auslagerung des Framestacks an den Hauptspeicher ist die Unterschiedliche Wortbreite. Für den Garbagecollector werden zu jedem 32bit Datenwort noch 2 Statusbit gespeichert in den festgehalten wird, ob es sich bei den Daten um Metadaten, Referenzen oder Werte handelt. Damit beim Auslagern der Daten auf dem externen Speicher diese Informationen nicht verloren gehen werden die Schreibvorgänge immer in Blöcken von 16 Wörtern durchgeführt, wobei die Statusbits der 16 Wörter als weiteres 32Bit Wort in den Speicher geschrieben werden. Dies ist kein Problem, da die Größe der beim Spill and Fill Vorgang übertragenen Bereiche einen Vielfachen von 16 Wörtern entspricht.

Der Ausgelagerter Framestack wird in das obere Ende des 512MB großen DDR3 Speicher gelegt und belegt $numbermaxthreads * wordspersperthread +$. Als Ausgangspunkt wird die höchste Speicheradresse des DDR Speichers verwendet und 2 mal nach links geschiftet, da der externe Speicher Byte adressiert, der Framestack jedoch wortadressiert ist. Davon wird die Maximale Größe des Framestacks abgezogen. Von da an aufwärts werden die Daten der einzelnen Threads gespeichert.

3.1.2 Schreiben von Framestackinhalten in den Hauptspeicher

Für das Zwischenspeichern und packen der Blöcke wird ein eigenes Modul verwendet. Vom Framestack aus kann auf dieses Modul geschrieben werden. Dabei wird die Startadresse des Blocks, die Länge des AXI Bursts und das erste Datenwort angelegt. Zum Zwischenspeichern dieser Daten werden jeweils Fifos verwendet. Wobei die Fifos für Burstlänge und Startadresse einen Sechzehntel der Größe des Datenfifos haben. Für jedes übertragene Datenwort werden 2 Statusbits in einen 32 Bit Register geschrieben. Nach dem 16 Datenworte in den Datenfifo geschrieben wurden wird der Inhalt dieses Registers in den Fifo geschrieben. Das Modul, indem die eigentliche AXI Übertragung verarbeitet wird wartet darauf das der Adress- und Burstlengthfifo nicht leer sind. Wenn die der Fall ist wird die Adresse und Burstlänge ausgelesen und wie folgt umgerechnet:

Anschließend werden die Daten aus dem Fifo übertragen bis die neu berechnete Burstlänge erreicht wurde. Solange der noch Einträge in den Fifos für Burstlänge und Adresse liegen wird eine neue Übertragung gestartet.

3.1.3 Lesen von Framestackinhalten aus den Hauptspeicher

Wenn die Daten ausgelesen werden sollen, wird das selbe Modul verwendet. Es wird die Startadresse des auszulesenden Blocks und die Burstlänge übergeben. Diese wird für den DDR Speicher umgerechnet und der Lesevorgang über AXI gestartet. Für das Zwischenspeichern der ausgelesenen Daten werden

Das erfolgreiche Einlesen wird dem Framestack signalisiert, woraufhin die Daten des Blocks ausgelesen werden. Zum Zwischenspeichern der ausgelesenen Daten werden 2 Fifos verwendet. Ein Fifo wird für die gelesenen Daten speichert 32 Wörter mit 32 Bit Wortbreite. In einen weiteren Fifo ebenfalls 32Bit werden die kumulierten Statusbits gespeichert. Nachdem die ersten 16bit Datenworte in den Fifo gespeichert wurden, enthält das nächste Wort die Statusbits, dieses wird in den entsprechenden Fifo gespeichert. Sobald das Fifo für die Statusbits nicht leer ist, kann auf Seiten des Framestacks mit den Auslesen begonnen werden. Bei jeden takt in dem eine 1 an readable angelegt ist wird ein Datenwort aus dem Fifo und 2 Statusbits übertragen. Nach dem 16 Wörter aus den Datenfifos gelesen wurden wird wenn vorhanden das nächste Statuswort aus den Statusfifos gelesen.

3.2 Spill and Fill Windows

Bei den Spill and Fill Prozess werden Teile des aktuellen Threads und evtl die anderer Threads in sogenanntes Windows vorgehalten. Die Windows werden als Blockrams realisiert. Die Anzahl und Größe dieser Windows kann über Parameter eingestellt werden. Für jedes Window werden die Framestack Adressen der obersten und untersten Grenze, des Adressbereichs gespeichert, der in dem Window vorgehalten wird. Dazu wird jeweils die Basis Adresse im Window gespeichert, die angibt an welcher Window Adresse die unterste Framestackadresse liegt. Das Window selber wird als Ringspeicher organisiert. Für jedes Window wird in einen Register gespeichert welcher Thread in diesen momentan vorgehalten wird. Die Adressen werden für das Window wie folgt umgerechnet:

3.3 Framestackteile des aktiven Threads auslagern (Spill)

Stackteile des aktiven Threads auf den externen Speicher auslagern ist eines der zwei Hauptbestandteile des Spill and Fill Mechanismus. Nämlich Spill. Bei dieser Implementierung wird der Spill Mechanismus ausgelöst, wenn beim Funktionsaufruf festgestellt wird, dass der Stack das aktuelle Window überschreiten würde.

3.3.1 Überprüfen der Größe

Bei jeden Funktionsaufruf wird der Platzbedarf im Stack abgeschätzt und geprüft ob noch Platz dafür im aktuellen Window ist.

Ein Stackframe in AMIDAR beginnt mit den Argumenten und lokalen Variablen, darüber kommt der Callercontext bestehend aus: Localspointer, Callercontextpointer und Stackpointer. Zum bestimmen ob der im Window vorhandene Platz noch ausreicht wird vom aktuellen Stackpointer die Anzahl der Parameter der aufgerufenen Funktion abgezogen und Anzahl lokaler Variablen die Größe des Callercontexts drauf addiert und eine Konstante für die Anzahl möglicher Parameter drauf addiert. Wenn das Ergebnis noch im aktuellen Adressbereich des Windows liegt, wird ein normaler Invoke durchgeführt, andernfalls wird ein Spill durchgeführt.

3.3.2 Spill

Zu Beginn des Spill Vorgangs wird die Burstlänge für die AXI Übertragung auf 128 gesetzt und die Bottom Adresse des unteren Windows wird in ein Register zwischengespeichert, das als Laufvariable für den folgenden Prozess verwendet wird. Anschließend wird darauf gewartet, dass das AXI Modul geschrieben werden kann, ist dies der Fall wird der Speicher im Window mit den Wert im Address Register adressiert.

Nachdem die Daten aus dem Blockram gelesen wurden wird ein in das oben beschriebene AXI Modul geschrieben. Dafür wird die Threadadresse zur Framestackadresse umgerechnet. Gleichzeitig wird der Wert im Register mit der Laufvariable um eins erhöht.

3.3.3 Fill

3.4 Threadwechsel mit Verdrängung

3.5 Garbage Collector Interface

4 Evaluation

4.1 Dauer

4.2 Dauer von Spill and Fill Vorgängen

4.3 Dauer von Threadwechseln

5 Fazit

6 Installation von Latex

Um LaTeX verwenden zu können braucht man das “Satzprogramm” an sich und einen Editor um den Quellcode zu erstellen. Das Satzprogramm nennt sich unter Windows MiKTeX, unter Linux heißt die Tex-Distribution TeX Live (Nachfolger von teTeX). Als Editor genügt an sich das Notepad, mehr Komfort wie Syntaxhighlighting bieten Editoren wie z.B. xEMACS, TeXnicCenter und Winedt.

6.1 Wie beschaffe ich mir LaTeX?

Da die meisten Latex-Distributionen Freeware sind kann die Software ohne Probleme aus dem Internet geladen werden.

6.1.1 tudfonts und tudesign

Im Hauptordner dieser Vorlage befinden sich zwei Zip-Dateien. Diese Dateien beinhalten die “Fonts” und das “Design” der TU-Darmstadt und sind sehr wichtig für die Formatierung dieser Arbeit. Um die Fonts und das Design der TU richtig zu setzten, müssen die Schritte unter dem nächsten Unterkapitel 6.2 genau befolgt werden. Es ist darauf zu achten dass eine falsche Installation zu Fehlermeldungen in Latex bzw. zu einer falschen Formatierung führen kann.

Nach einer erfolgreichen Installation von Latex und der TU-Fonts sollte die Dokumentation ohne Probleme in Latex geöffnet werden können. Da die Beispielbilder in dieser Vorlage in .pdf und .jpg vorliegen, muss im Menü von Latex unter “Setzen” , die Option “**Pdftex**” gewählt werden.

6.2 Installation des TUD-Design

Im folgenden werden die Installationsanleitungen für die verschiedenen Betriebssysteme gründlich erläutert. Befolgen Sie bitte die Schritte so genau wie möglich.

*Dieses Kapitel wurde komplett aus der TU-Homepage übernommen

6.2.1 WinXP / MiKTeX

Die vorliegende Anleitung wurde mit MikTeX 2.7 unter Windows XP sowie Vista getestet. Über ältere MikTeX-Versionen kann ich derzeit keine Aussage treffen, ich hatte allerdings unter der Version 2.5 Fehlermeldungen. Es ist daher zu empfehlen, die aktuelle Version zu installieren.

Installation

1. In einen beliebigen Ordner die Zip-Dateien mit den Schriftarten (tudfonts-tex_current.zip) und dem TUD-Design (latex-tudesign_current.zip) herunterladen und entpacken.
2. Den Ordner texmf\fonts\map\dvipdfm inklusive seines Inhalts löschen.

3. Kopieren der Unterverzeichnisse von texmf in den MiKTeX-Ordner. (i. d. R. C: / Programme/MiKTeX 2.7 oder C:/ Program Files/ MiKTeX 2.7).
Das Verzeichnis updmap.d wird nicht benötigt.

4. Eine Konsole öffnen (z. B. mit der Eingabe von cmd unter Ausführen im Startmenü)

5. Auf der Konsole erst den Befehl

```
cd %PROGRAMFILES% /MiKTeX 2.7
```

und dann den Befehl

```
texhash
```

ausführen.

6. Dannach den Befehl

```
initexmf -edit-config-file updmap
```

ausführen.

7. Folgende Zeilen in die sich öffnende Datei einfügen und speichern:

```
Map 5ch.map
```

```
Map 5fp.map
```

```
Map 5sf.map
```

(Entspricht dem Inhalt der Datei updmap.d/20tex-tudfonts.cfg.)

8. Zuletzt den Befehl

```
initexmf -mkmaps
```

ausführen.

Danach sollte alles funktionieren. Happy Working! :-)

Kontrollmöglichkeit für die Schriftarten

Sollte es zu Problemen mit den Schriftarten kommen, empfiehlt sich ein Blick in die Datei psfonts.map, die man unter C: /Dokumente und Einstellungen/All Users/Anwendungsdaten/MiKTeX/2.7/dvips/config findet. In ihr sollten folgende Einträge, i. d. R. in den ersten Zeilen zu finden sein:

```
5chb8r Charter-Bold "TeXBase1Encoding ReEncodeFont<8r.enc <5chb8a.pfb
```

```
5chbo8r Charter-Bold "TeXBase1Encoding ReEncodeFont 0.167 SlantFont<8r.enc <5chb8a.pfb
```

```
5chr8r Charter "TeXBase1Encoding ReEncodeFont<8r.enc <5chr8a.pfb
```

```
5chri8r Charter-Italic "TeXBase1Encoding ReEncodeFont<8r.enc <5chri8a.pfb
```

```
5chro8r Charter "TeXBase1Encoding ReEncodeFont 0.167 SlantFont<8r.enc <5chr8a.pfb
```

```
5fpm8r FrontPageMedium "TeXBase1Encoding ReEncodeFont«8r.enc <5fpm8a.pfb
5fpmo8r FrontPageMedium "TeXBase1Encoding ReEncodeFont 0.167 SlantFont«8r.enc <5fpm8a.pfb
5fpr8r FrontPage "TeXBase1Encoding ReEncodeFont«8r.enc <5fpr8a.pfb
5fpro8r FrontPage "TeXBase1Encoding ReEncodeFont 0.167 SlantFont«8r.enc <5fpr8a.pfb
5sfr8r Stafford "TeXBase1Encoding ReEncodeFont«8r.enc <5sfr8a.pfb
5sfro8r Stafford "TeXBase1Encoding ReEncodeFont 0.167 SlantFont«8r.enc <5sfr8a.pfb
```

Ist dies nicht der Fall oder stehen dort zwar die TU Schriftarten, aber mit wesentlich kürzeren Einträgen, so ist zu überprüfen ob das Verzeichniss `texm/fonts/map/dvipdfm/tex-tudfonts` existiert. Es sollte nicht vorhanden bzw. leer sein.

6.2.2 Apple OS-X

Um LaTeX-Dokumente im TUD Design auf einem Apple-Rechner zu erstellen ist die LaTeX-Distribution TeXLive zu empfehlen. Diese lässt sich entweder bei der TUG (<http://www.tug.org/mactex/>) herunterladen und installieren oder per MacPorts (<http://www.macports.org/>) hinzufügen. Hier wird die zweite Möglichkeit beschrieben, erstere sollte aber analog funktionieren. An dieser Stelle empfiehlt sich die LaTeX-Distribution manuell zu laden und installieren da die Datei über die MacPorts von der Grösse her kleiner ist als in die Datei unter www.tug.org/mactex/. Nach erfolgreicher Installation von MacPorts öffne man Terminal.app (zu finden unter /Programme/Dienstprogramme bzw. /Applications/Utilities). Hier sind nun die folgenden Befehle einzugeben:

```
$ sudo port selfupdate
$ sudo port -d install texlive_texmf-full
```

Nachdem die Installation von TeXLive und dessen Abhängigkeiten abgeschlossen ist lege man die gepackten Schriften (`tudfonts-tex_current.zip`) sowie das eigentliche TUDDesign (`latex-tuddesign_current.zip`) direkt im Stammverzeichnis des eigenen Benutzers ab. In Terminal.app sind anschließend folgende Befehle abzusetzen:

```
$ unzip latex-tuddesign_current.zip -d /Library/
$ unzip tudfonts-tex_current.zip -d /Library/
```

Dies resultiert in einem neuen `/Library/texmf` Verzeichnis, in dem sich alle notwendigen Daten befinden. Indiziert und für LaTeX bereitgestellt werden diese durch die Eingabe von:

```
$ sudo mktexlsr
```

Der letzte Schritt ist das Erstellen der Fontmaps für die TUD-Schriften. Dafür führe man folgendes aus:

```
$ sudo updmap -enable Map 5ch.map
$ sudo updmap -enable Map 5fp.map
$ sudo updmap -enable Map 5sf.map
```

6.2.3 Debian / Ubuntu (Global)

Für Ubuntu können die lenny Pakete verwendet werden. Tragen Sie für etch

```
deb http://exp1.fkp.physik.tu-darmstadt.de/tuddesign/ etch tud-design
deb-src http://exp1.fkp.physik.tu-darmstadt.de/tuddesign/ etch tud-design
```

bzw. für lenny

```
deb http://exp1.fkp.physik.tu-darmstadt.de/tuddesign/ lenny tud-design
deb-src http://exp1.fkp.physik.tu-darmstadt.de/tuddesign/ lenny tud-design
```

in die `/etc/apt/sources.list` ein und führen Sie

```
apt-get update ; /
apt-get install debian-tuddesign-keyring ; /
apt-get update
```

aus. Dannach können Sie die gewünschten Pakete installieren.

Architekturen: alpha amd64 arm hppa i386 ia64 mips mipsel powerpc sparc s390 source

TUD Schriftarten für Debian (etch / lenny)

Die TUD Schriftarten sind auf drei Pakete aufgeteilt:

- Das Paket `t1-tudfonts` beinhaltet die Type1 Schriftarten und macht sie unter X11 zugänglich, so dass sie z.B. unter scribus verwendet werden können.
- Das Paket `tex-tudfonts` macht die TUD Schriftarten unter LaTeX und Programmen wie z.B. dvips nutzbar.
- Das Paket `ttf-tudfonts` beinhaltet die TrueType Schriftarten und macht sie unter X11 nutzbar, so dass sie z.B. unter OpenOffice oder StarOffice (kostenfrei für Uni-Angehörige und Studenten nutzbar) zur Verfügung stehen.

Installieren können Sie die Pakete mit

```
apt-get install t1-tudfonts tex-tudfonts ttf-tudfonts
```

TUD-Design für Debian (etch / lenny)

Zur Zeit gibt es nur ein Paket. Eventuell wird diese Paket später in mehrere aufgeteilt.

Das Paket `latex-tuddesign` enthält die folgenden LaTeX-Klasse:

- `tudreport` für Texte, wie z.B. Diplomarbeiten
- `tudbeamer` für Präsentationen z.B. mit einem Beamer
- `tudposter` für Aushänge und Poster, z.B. für Konferenzbeiträge

- tudletter für Briefe
- tudexercise für Übungsblätter

Die Dokumentation befindet sich nach der Installation im Verzeichnis `/usr/share/doc/latex-tuddesign`. Installieren können Sie die Pakete mit

```
apt-get install latex-tuddesign
```

Hinweis für Ubuntu: Es muß das Paket `texlive-fonts-recommended` oder `tetex-extra` installiert sein

6.2.4 Linux (Global)

Um LaTeX-Dokumente im TUD Design auf einem Linux-Rechner zu erstellen ist die LaTeX-Distribution TeXLive oder TeTeX zu empfehlen. Diese sind in fast allen Distributionen vorhanden.

Nachdem die Installation von TeXLive/TeX und deren Abhängigkeiten abgeschlossen ist entpackt man die Schriften (`tudfonts-tex_current.zip`) sowie das eigentliche TUDDesign (`latex-tuddesign_current.zip`) in das Verzeichnis `/usr/share/texmf`. Dabei ist darauf zu achten, dass die Leserechte richtig gesetzt sind. Das geht am einfachsten, wenn man auf einer Konsole als root folgende Befehle ausführt:

```
> umask 022
> unzip latex-tuddesign_current.zip -d /usr/share/
> unzip tudfonts-tex_current.zip -d /usr/share/
```

Anschließend muss ebenfalls als root der Befehl

```
> texhash
```

ausgeführt werden. Sollte das Verzeichnis `/usr/share/texmf` nicht in der Liste der aktualisierten Verzeichnisse aufgeführt werden hilft:

```
> texhash /usr/share/texmf
```

Der letzte Schritt ist das Einbinden der Fontmaps für die TUD-Schriften. Dafür führe man folgendes als root aus:

```
> updmap-sys -enable Map 5ch.map
> updmap-sys -enable Map 5fp.map
> updmap-sys -enable Map 5sf.map
```

Ob die Schriften installiert sind, kann man mit

```
> updmap-sys -listmaps | egrep "^Map[[:blank:]]*5"
```

überprüfen. Der Befehl sollte folgendes zurückliefern:

```
updmap: This is updmap, version 1122009795-debian
updmap: no permissions for writing '/var/lib/texmf/web2c/updmap.log', so no transcript
```

```
Map 5ch.map
Map 5fp.map
Map 5sf.map
```

Ist dies der Fall, sollte alles funktionieren.

6.2.5 Linux (Lokal)

Um LaTeX-Dokumente im TUD Design auf einem Linux-Rechner zu erstellen ist die LaTeX-Distribution TeXLive oder TeTeX zu empfehlen. Diese sind in fast allen Distributionen vorhanden. Wenden Sie Sich an Ihren Administrator, damit er sie installiert oder installieren Sie sie lokal nach Anleitung der LaTeX-Distribution.

Nachdem die Installation von TeXLive/TeX und deren Abhängigkeiten abgeschlossen ist entpackt man die Schriften (tudfonts-tex_current.zip) sowie das eigentliche TUDDesign (latex-tuddesign_current.zip) in das Verzeichnis \$HOME. Das geht am einfachsten, wenn man auf einer Konsole folgende Befehle ausführt:

```
> unzip latex-tuddesign_current.zip -d $HOME/
> unzip tudfonts-tex_current.zip -d $HOME/
```

Unter Debian muss noch das Verzeichnis \$HOME/updmap.d nach \$HOME/.texmf-config/ verschoben werden und aus der Datei \$HOME/.texmf-config/updmap.d/20tex-tudfonts.cfg die Zeile, falls vorhanden, # -- DebPkgProvidedMaps -- entfernt werden.

In andere Distributionen kann diese Verzeichnis inklusive Inhalt gelöst werden.

Wenn das Verzeichnis \$HOME/.texmf-var nicht existiert, muss es mit

```
> mkdir $HOME/.texmf-var
```

angelegt werden. Anschließend ist der Befehl

```
> texhash
```

auszuführen. Sollte das Verzeichnis \$HOME/texmf nicht aufgeführt werden hilft:

```
> texhash $HOME/texmf --
```

Der letzte Schritt ist das Einbinden der Fontmaps für die TUD-Schriften. Unter Debian wird das mit

```
> update-updmap
> updmap
```

erreicht. Unter andere Distributionen mit

```
> updmap -enable Map 5ch.map
> updmap -enable Map 5fp.map
> updmap -enable Map 5sf.map
```

Ob die Schriften installiert sind, kann man mit

```
> updmap -listmaps | egrep "^Map[[:blank:]]*5"
```

überprüfen. Der Befehl sollte folgendes zurückliefern:

```
updmap: This is updmap, version 1122009795-debian
updmap: using transcript file
`$HOME/.texmf-var/web2c/updmap.log'
Map 5ch.map
Map 5fp.map
Map 5sf.map
```

Ist dies der Fall, sollte alles funktionieren.

7 Überschriften

Für solch eine Arbeit wird der Text in diverse Abschnitte, Unterabschnitte und evtl. auch noch Unterunterabschnitte unterteilt. Diese müssen dann mit der entsprechenden Überschrift gekennzeichnet werden. Die Dokumentklasse parkskip kennt hier die Unterscheidung in chapter, section, subsection, subsubsection und paragraph.

Der Befehl `\chapter{Überschrift1}` beginnt ein neues Kapitel, erzeugt eine Kapitelüberschrift und trägt diese ins Inhaltsverzeichnis ein. So ist der oben genannte Kapitel mit dem Befehl `\chapter{Überschriften}` als chapter definiert.

Optional kann man mit Hilfe des Befehls `\chapter[Kurzform]{Überschrift}` eine Kurzform für den Kapitelname angeben. Die Kurzform wird dann anstelle der Überschrift ins Inhaltsverzeichnis eingetragen.

Beispiel

`\chapter[Anfänge (1920)]{Anfänge der modernen Science-Fiction-Literatur (1920)}`. Im Inhaltsverzeichnis erscheint nur "Anfänge (1920)".

7.1 Überschrift2

Mit dem Befehl `\section{Überschrift2}` wird einen neuen Abschnitt des Dokuments auf der section-Ebene erzeugt. Die zugehörige Überschrift wird definiert und ins Inhaltsverzeichnis eingetragen. Wenn eine Kurzform erwünscht ist, kann sie auch mit dem entsprechenden Befehl erzeugt werden (Siehe Beispiel unter Kapitel7).

7.1.1 Überschrift3

In der subsection-Ebene, wird ebenfalls die zugehörige Überschrift erzeugt und ins Inhaltsverzeichnis eingetragen. Hier ist auch ebenfalls möglich eine Kurzform angegeben. Der Befehl lautet folgendermaßen `\subsection{Überschrift3}`.

Überschrift4 und Paragraph

Der Befehle dazu lauten `\subsubsection{Überschrift4}` bzw. `\paragraph{"Paragraph"}`. Die Überschriften für diese Ebenen erfolgen ohne Nummerierung und werden nicht im Inhaltsverzeichnis aufgenommen. Das o.g. Beispiel wurde z. B. als Paragraph definiert.

8 Text in Latex

In Latex gibt es für einige Symbole bestimmte Befehle, die man eingeben muss, um sie als Text richtig darstellen zu können. Im folgenden werden einige Befehle bzw. Kodierungen zur Erstellung einiger Symbole erläutert.

8.1 Umlaute und Ähnliches

Es gibt einige Methoden Umlaute in Latex zu setzten. Die einfachste erfolgt mit Hilfe eines Anführungszeichens, das vor dem gewünschten Vokal positioniert werden muss, d.h. um das Wort “Kühe” richtig darzustellen muss man im Editor “K”uhe” schreiben. Genauso werden auch die Umlaute ä und ö erzeugt. Für scharfes s “ß” braucht man auch vor einem “s” wieder ein Anführungszeichen zu setzten. Die Codierung bzw. entsprechenden Befehle zu weiteren Sonderzeichen kann man unter www.wikibooks.org finden.

8.2 Querverweis

Innerhalb eines Textes kann man einen Querverweis einfügen. Dies funktioniert mit dem Befehl `\ref{Name}`. Der Befehl erzeugt einen Querverweis auf eine Textstelle, die zuvor durch einen `\label`-Befehl mit dem angegebenen Namen versehen wurde. Der Querverweis gibt die Gliederungsnummer der betreffenden Textstelle an. Aus diesem Grund es ist sinnvoll Kapiteln bzw. Unterkapiteln auf die man oft verweisen möchte mit dem Befehl `\label{Name}` zu vermerken.

Beispiel

Siehe Kapitel 8 um ausführliche Information zum Text in Latex.

In diesem Beispiel wurde auf den Kapitel Text in Latex verwiesen mit `\ref{text}` da das Kapitel Text in Latex mit dem Befehl `\label{cha:text}` versehen wurde. Der Befehl `\label` muss direkt nach dem Befehl für den Gliederungsabschnitt bzw. Überschrift erfolgen.

9 Formeln in Latex

Im Prinzip ist die Syntax relativ einfach aufgebaut. Wenn man eine Formel in LaTeX eingeben möchte, muss eine entsprechende Umgebung gewählt werden. Die bekanntesten sind `math` und `displaymath`.

- `displaymath` setzt die Formel ab und zentriert sie
- `math` baut die Formel mehr in den Text ein.

Eingeleitet wird eine solche Umgebung durch `\begin` also z.B. `\begin{displaymath}` und durch `\end` beendet. Anhand folgenden Beispiels lässt sich zeigen wie ein Bruch aufbaut und dargestellt wird.

Mit den Befehlen,

```
\begin{math}
\frac{1}{2}
\end{math}
```

wird der Bruch $\frac{1}{2}$ mitten im Text erscheinen. Gibt man die Befehle,

```
\begin{displaymath}
\frac{1}{2}
\end{displaymath}
```

erscheint der Bruch zentriert als separate Formel, nämlich,

$$\frac{1}{2}$$

Andere Konstruktionselemente lassen sich besser an konkreten Formeln darstellen. Daher hier einige bekannte Formel aus Naturwissenschaft und Statistik.

Formel für das arithmetische Mittel einer Stichprobe:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

Formel für die Varianz einer Stichprobe:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

Formel für die Standardabweichung:

$$s = \sqrt{s^2} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$$

oder:

$$= \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}}$$

Geometrisches Mittel:

$$G = \sqrt[n]{\prod_{i=1}^n x_i}$$

Binomialkoeffizient:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Zeitunabhängige dreidimensionale Schrödingergleichung:

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} + \frac{\partial^2 \psi}{\partial z^2} = -\frac{2m}{\hbar^2} (E - U) \psi.$$

Faradaysches Induktionsgesetz:

10 Einbindung von Graphen

Um Bilder in Latex einzubinden es ist nur wichtig dass sie im richtigen Format und im richtigen Verzeichnis vorliegen. Unter dem Ordner “Dokumentation” für diese Vorlage ist der Unterordner “images” angelegt. In diesem Ordner liegen alle Bilder die in diesem Dokument enthalten sind. Da können auch alle Bilder angelegt werden die in der Arbeit aufgerufen werden sollen. Die Bilder sollen für diese Vorlage in .pdf oder .jpg vorliegen. Die Bilder können auch direkt im Hauptverzeichnis angelegt werden. Anhand folgenden Beispiels werden die Befehle erläutert die man braucht um ein Bild in diesem Kapitel hinzuzufügen.

Beispiel

Das Bild mit dem Namen “blockschaltbild.pdf”, das im Unterordner “images” liegt, wird mit folgenden Befehlen aufgerufen,

```
\begin{figure}[ht]
\centering
\includegraphics[width=0.66\textwidth]{images/blockschaltbild.pdf}
\caption{Beispiel: Blockschaltbild [1]}
\label{fig:blockschaltbild}
\end{figure}
```

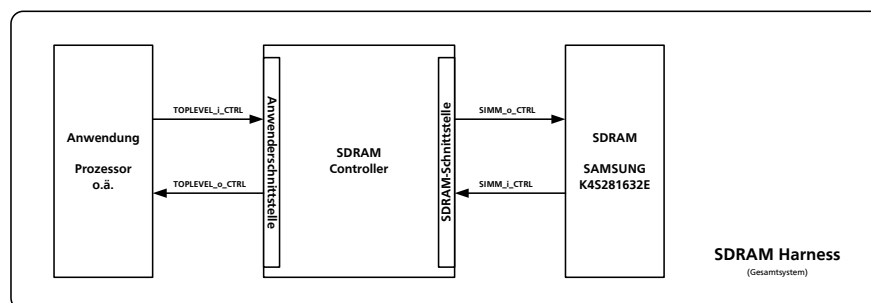


Abbildung 10.1: Beispiel: Blockschaltbild [1]

11 Tabellen

Um eine einfache Tabelle in Latex zu erzeugen, kann man sich zunächst mit der `tabular` Umgebung begnügen. Die Syntax dafür ist die folgende:

```
\begin{tabular}{cols}
Tabelleneinträge
\end{tabular}
```

Im Argument “cols” gibt man für jede Spalte, die in der Tabelle stehen soll, die Ausrichtung an. Man hat dabei folgende Optionen:

l	Linksbündig
r	Rechtsbündig
c	Zentriert
p{x cm}	Parbox der Breite x cm

Tabelle 11.1: Tabellenausrichtung

Will man die Spalten durch einen oder mehrere Striche trennen, so fügt man mittels der Tastenkombination AltGr + “<” einen senkrechten Strich (|) zwischen (bzw. vor oder nach) den Ausrichtungsangaben ein (siehe Beispiele). Die Breite der Spalten bestimmt LATEX selbst, es sei denn man gibt eine Parbox an. Die Parbox hat den weiteren Vorteil, dass in einer Zelle der Tabelle mehrere Zeilen möglich sind. Weiters kann man mittels Parboxen auch Tabulatoren wie im Word erzeugen.

Die Tabelleneinträge werden zeilenweise eingegeben. Dabei erfolgt die Trennung innerhalb der Zeile mit “&”. Um in die nächste Zeile zu springen hat man den Befehl “\\”. Will man auch die Zeilen mittels Strichen trennen, so sind diese mit dem Befehl “\hline” an die entsprechende Stelle zu setzen.

Das folgenden Beispiele erläutern die `tabular`-Umgebung: Mit diesen Kommandos

```
\begin{tabular}{1p{10cm}}
\textbf{tabular} & Die Umgebung tabular wird in Latex gewählt um einfache Tabellen zu erzeugen\\
\textbf{table} & table wird gebraucht um abgesetzte Tabellen zu erzeugen. Die Tabelle kann damit als
Gleitobjekt eingefügt werden\\
\textbf{hline} & Mit diesem Befehl kann man mit einem Strich Zeilen trennen\\
\textbf{\textbackslash\textbackslash} & Um auf die nächste Zeile zu springen\\
\end{tabular}
```

wird folgende Tabelle erzeugt:

tabular	Die Umgebung tabular wird in Latex gewählt um einfache Tabellen zu erzeugen
table	table wird gebraucht um abgesetzte Tabellen zu erzeugen. Die Tabelle kann damit als Gleitobjekt eingefügt werden
hline	Mit diesem Befehl kann man mit einem Strich Zeilen trennen
\\	Um auf die nächste Zeile zu springen

Dies ist ein Beispiel, wie man Parboxen zur “Erzeugung” von Tabulatoren benutzt werden kann.

Tabellen, welche mittels der tabular-Umgebung geschrieben werden, werden einfach, wie ein großer Buchstabe in den Text eingefügt. Um abgesetzte Tabellen zu erzeugen bedient man sich zusätzlich der table-Umgebung, die die Tabelle als sogenanntes Gleitobjekt einfügt. Sie hat folgende Syntax:

```
\begin{table}[Ausrichtung]
tabular-Umgebung
\end{table}
```

In die eckigen Klammern steht die Ausrichtung. Dafür hat man folgende Optionen:

h	“here”- Tabelle soll an der selben Stelle wie im Quelltext eingerichtet werden
t	“top”- Tabelle wird an den unteren Rand der Seite gestellt
b	“bottom”- Tabelle wird an den unteren Rand der Seite gestellt
p	“page”- Tabelle wird auf einer Gleitobjektseite eingerichtet

Tabelle 11.2: table-Ausrichtung

Weiteres Beispiel

Die folgenden Befehle erzeugen eine Tabelle mit Linien:

```
\begin{table}[!h]
\begin{center}
\begin{tabular}{|l|r|r|r|r|r|}
\hline&$x_{\min}$&$x_{0,25}$&$x_{0,5}$&$x_{0,75}$&$x_{\max}$\\
\hline Alter &14&19&20,5&23&70\\
\hline Semester &0&2&2&4&14\\
\hline \end{tabular}
\end{center}
\end{table}
```

Mehr Informationen über Tabellen finden sie [hier](#).

	x_{min}	$x_{0,25}$	$x_{0,5}$	$x_{0,75}$	x_{max}
Alter	14	19	20,5	23	70
Semester	0	2	2	4	14

Tabelle 11.3: Verteilung von Alter und Semesteranzahl

12 Literaturverzeichnis

- [1] Tao Guo. Ansteuerung eines SDRAM auf einem Xilinx FPGA-Board. Master's thesis, TU Darmstadt, Fachgebiet Rechnersysteme, 2008.