universität ulm

Betriebssysteme SS2024

Tutorium 01

Architektur und Grundprinzipien

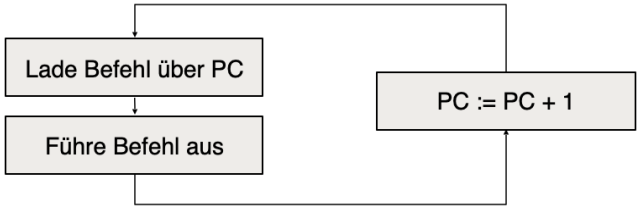
Inhaltsverzeichnis

Der Prozessor

- Der Prozessor ist eine aktive Einheit im Rechensystem und besitzt mehrere Register mit schnellem Zugriff.
- Diese Register sind die Speicherbereiche für Daten.
- Der Programmzähler PC enthält die Adresse einer Speicherzelle welche den Beginn des nächsten kodierten Maschinenbefehls markiert.
- Die Maschinenbefehle sind im Instruktionsspeicher gespeichert, wobei ein Befehl typischerweise mit mehreren Bytes kodiert ist.

Befehlsabarbeitung

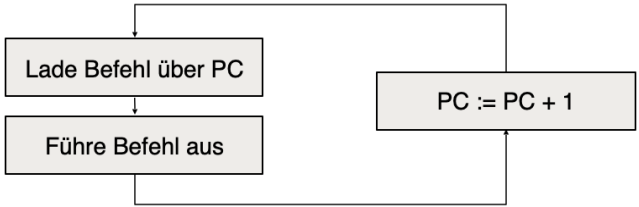
Der PC arbeitet Maschinenbefehle wie folgt ab:



- Laden des Inhalts der Speicherzelle mittels des PC

Befehlsabarbeitung

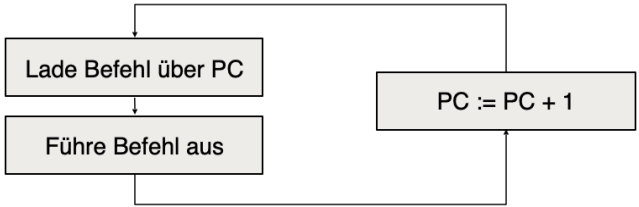
Der PC arbeitet Maschinenbefehle wie folgt ab:



- Laden des Inhalts der Speicherzelle mittels des PC
- Dekodieren der Instruktion

Befehlsabarbeitung

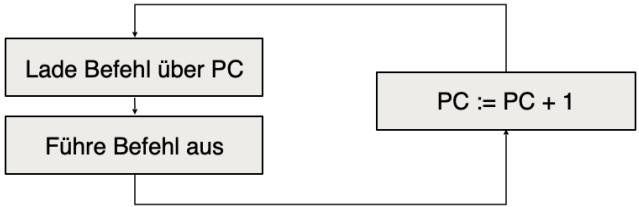
Der PC arbeitet Maschinenbefehle wie folgt ab:



- Laden des Inhalts der Speicherzelle mittels des PC
- Dekodieren der Instruktion
- Ausführen des Befehls

Befehlsabarbeitung

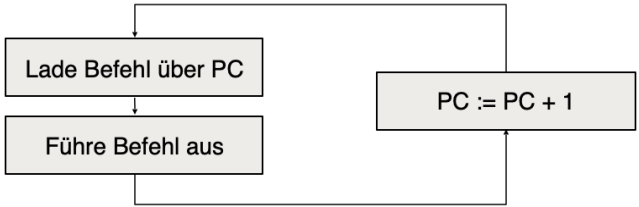
Der PC arbeitet Maschinenbefehle wie folgt ab:



- Laden des Inhalts der Speicherzelle mittels des PC
- Dekodieren der Instruktion
- Ausführen des Befehls
- Inkrementieren des PC (PC=PC+1)
  $\rightarrow$ PC zeigt nun auf nächsten Befehl im Speicher

Befehlsabarbeitung

Der PC arbeitet Maschinenbefehle wie folgt ab:



- Laden des Inhalts der Speicherzelle mittels des PC
- Dekodieren der Instruktion
- Ausführen des Befehls
- Inkrementieren des PC (PC=PC+1)
  $\rightarrow$ PC zeigt nun auf nächsten Befehl im Speicher
- Wiederholen der Prozedur

Beispiel: Befehlsabarbeitung (1)

Gegeben sei ein Rechnermodell mit folgenden Registern und initialen Werten:
R0 = 00_{16}, R1 = 02_{16}, PC = $1C_{16}$

Zudem hat der Rechner folgenden Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Aufgabe: Bestimmen Sie die Registerbelegung nach jedem ausgeführten Befehl!

Beispiel: Befehlsabarbeitung (2)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	**Setze R0 auf 05_{16}**
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 00_{16}, R1 = 02_{16}, PC = $1C_{16}$

Beispiel: Befehlsabarbeitung (3)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	**Addiere R1 zu R0 und speichere das Ergebnis in R0**
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 05_{16}, R1 = 02_{16}, PC = 20_{16}

Beispiel: Befehlsabarbeitung (4)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	**Addiere R1 zu R1 und speichere in R1**
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 07_{16}, R1 = 02_{16}, PC = 24_{16}

Beispiel: Befehlsabarbeitung (5)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
$\mathbf{28}_{16}$	**Falls** $R0 \geq 07_{16}$ **springe zu Adresse** $\mathbf{C8}_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 07_{16}, R1 = 04_{16}, PC = 28_{16}

Beispiel: Befehlsabarbeitung (6)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	**Stoppe**

Registerbelegung: R0 = 07_{16}, R1 = 04_{16}, PC = $C8_{16}$

Beispiel: Befehlsabarbeitung (7)

Die finale Tabelle sieht folgendermaßen aus:

Ausgeführter Befehl	R0	R1	PC
Keiner	00_{16}	02_{16}	$1C_{16}$
$1C_{16}$	05_{16}	02_{16}	20_{16}
20_{16}	07_{16}	02_{16}	24_{16}
24_{16}	07_{16}	04_{16}	28_{16}
28_{16}	07_{16}	04_{16}	$C8_{16}$
$C8_{16}$	07_{16}	04_{16}	–

Übung: Befehlsabarbeitung (1)

Gegeben sei das gleiche Rechnermodell aber nun mit folgenden initialen Werten:
R0 = 00_{16}, R1 = 01_{16}, PC = $1C_{16}$

Der Rechner hat wieder folgenden Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Aufgabe: Bestimmen Sie die Registerbelegung nach jedem ausgeführten Befehl!

Übung: Befehlsabarbeitung (2)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	**Setze R0 auf 05_{16}**
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 00_{16}, R1 = 01_{16}, PC = $1C_{16}$

Übung: Befehlsabarbeitung (3)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	**Addiere R1 zu R0 und speichere das Ergebnis in R0**
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 05_{16}, R1 = 01_{16}, PC = 20_{16}

Übung: Befehlsabarbeitung (4)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	**Addiere R1 zu R1 und speichere in R1**
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 06_{16}, R1 = 01_{16}, PC = 24_{16}

Übung: Befehlsabarbeitung (5)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
$\mathbf{28}_{16}$	**Falls** $R0 \geq 07_{16}$ **springe zu Adresse** $\mathbf{C8}_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 06_{16}, R1 = 02_{16}, PC = 28_{16}

Übung: Befehlsabarbeitung (6)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	**Springe zu Adresse 20_{16}**
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 06_{16}, R1 = 02_{16}, PC = $2C_{16}$

Übung: Befehlsabarbeitung (7)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	**Addiere R1 zu R0 und speichere das Ergebnis in R0**
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 06_{16}, R1 = 02_{16}, PC = 20_{16}

Übung: Befehlsabarbeitung (8)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	**Addiere R1 zu R1 und speichere in R1**
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 08_{16}, R1 = 02_{16}, PC = 24_{16}

Übung: Befehlsabarbeitung (9)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
$\mathbf{28}_{16}$	**Falls** $R0 \geq 07_{16}$ **springe zu Adresse** $\mathbf{C8}_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	Stoppe

Registerbelegung: R0 = 08_{16}, R1 = 04_{16}, PC = 28_{16}

Übung: Befehlsabarbeitung (10)

Instruktionsspeicher:

Adresse	Befehl
$1C_{16}$	Setze R0 auf 05_{16}
20_{16}	Addiere R1 zu R0 und speichere das Ergebnis in R0
24_{16}	Addiere R1 zu R1 und speichere in R1
28_{16}	Falls $R0 \geq 07_{16}$ springe zu Adresse $C8_{16}$
$2C_{16}$	Springe zu Adresse 20_{16}
...	...
$C8_{16}$	**Stoppe**

Registerbelegung: R0 = 08_{16}, R1 = 04_{16}, PC = $C8_{16}$

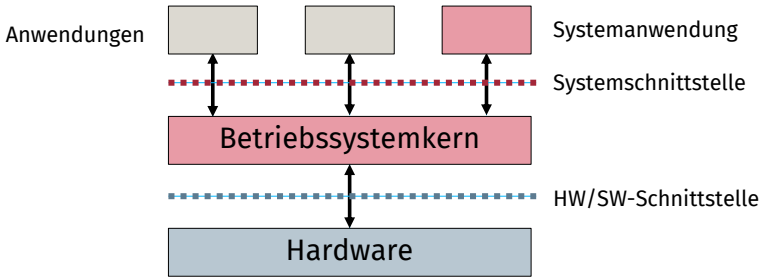
Übung: Befehlsabarbeitung (11)

Die finale Tabelle sieht folgendermaßen aus:

Ausgeführter Befehl	R0	R1	PC
Keiner	00_{16}	01_{16}	$1C_{16}$
$1C_{16}$	05_{16}	01_{16}	20_{16}
20_{16}	06_{16}	01_{16}	24_{16}
24_{16}	06_{16}	02_{16}	28_{16}
28_{16}	06_{16}	02_{16}	$2C_{16}$
$2C_{16}$	06_{16}	02_{16}	20_{16}
20_{16}	08_{16}	02_{16}	24_{16}
24_{16}	08_{16}	04_{16}	28_{16}
28_{16}	08_{16}	04_{16}	$C8_{16}$
$C8_{16}$	08_{16}	04_{16}	–

Betriebssystem

- Software zur Ressourcenverwaltung
- Stellt Grundkonzepte zur Strukturierung von Programmen bereit
- Schematischer Aufbau:



Betriebsmodi

User Mode (Benutzermodus):

- Für Anwendungen
- Eingeschränkter Befehlssatz
  $\rightarrow$ Nicht alle Befehle werden bearbeitet

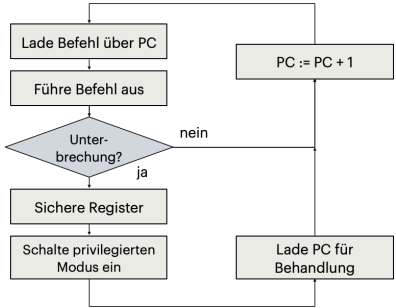
Supervisor Mode (Privilegierter Modus):

- Für das Betriebssystem
- Alle Befehle können ausgeführt werden
  $\rightarrow$ z.B Wechsel des Modus, Konfigurationsänderung

Interrupts

- Vorübergehende Unterbrechung eines laufenden Programms
- Eine spezielle Prozedur welche typischerweise kürzer aber zeitkritisch ist wird abgearbeitet
- Benötigt eine Unterbrechungsanforderung
  $\rightarrow$ Interrupt Request (IRQ)
- Behandlung durch eine Unterbrechungsroutine
  $\rightarrow$ Interrupt Service Routine (ISR)
- Anschließend wird das Programm an der Stelle vor der Unterbrechung fortgesetzt

External Interrupt



1. Prozessor unterbricht die laufende Bearbeitung und sichert alle Registerinhalte an einer bestimmten Speicherstelle
2. Prozessor führt Behandlung im priviligierten Modus aus
3. Wiederherstellung des ursprünglichen Zustands durch Laden aller gesicherten Registerinhalte

Internal Interrupt

Unterbrechung durch die Befehlsausführung:

- Fehlersituationen wie z.B Division durch Null oder eine versuchte Ausführung eines privilegierten Befehls im User Mode
- Unterbrechung des Befehlsflusses (Sprung in eine Unterbrechungsbehandlung)

System Call

Übergang ins Betriebssystem, auch User Interrupt genannt.

- Spezielle Befehle zum Eintritt in den Supervisor Mode
- Prozessor schaltet in den privilegierten Modus, sichert Register und führt definierte Befehlsfolge aus (vom privilegierten Modus aus konfigurierbar)
- Implementiert die Schnittstelle zum Betriebssystem

Beispiele für Interrupts

Interrupt	Ereignis
External	Tastenanschlag
Internal	Speichern in ein bereits belegtes Register
System Call	Anweisung in Register (gib die Datei BS-Blatt03.pdf aus)