



Technische Informatik II
Leitung: Prof. Dr. Mirosław Malek

SwARM

Swarming Autonomous Robotic Mobiles

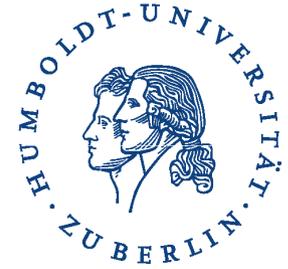
Jörg Pohle, Peter Hartig
Projektbetreuung: Dr. Sommer

Übersicht



- Problemanalyse
- Schaubilder
- Beschränkungen
- Zustandsübergangsgraphen
- Befehlsübersicht
- Beispiele aus dem Code
- Schlussbetrachtungen

Problemanalyse



Beispielsituationen:

- WLAN auf Großveranstaltungen
- Überwachung des Mikroklimas auf landwirtschaftlichen Anbauflächen
- Kontrolle eines Gebietes auf Bewegung von gepanzerten Fahrzeugen

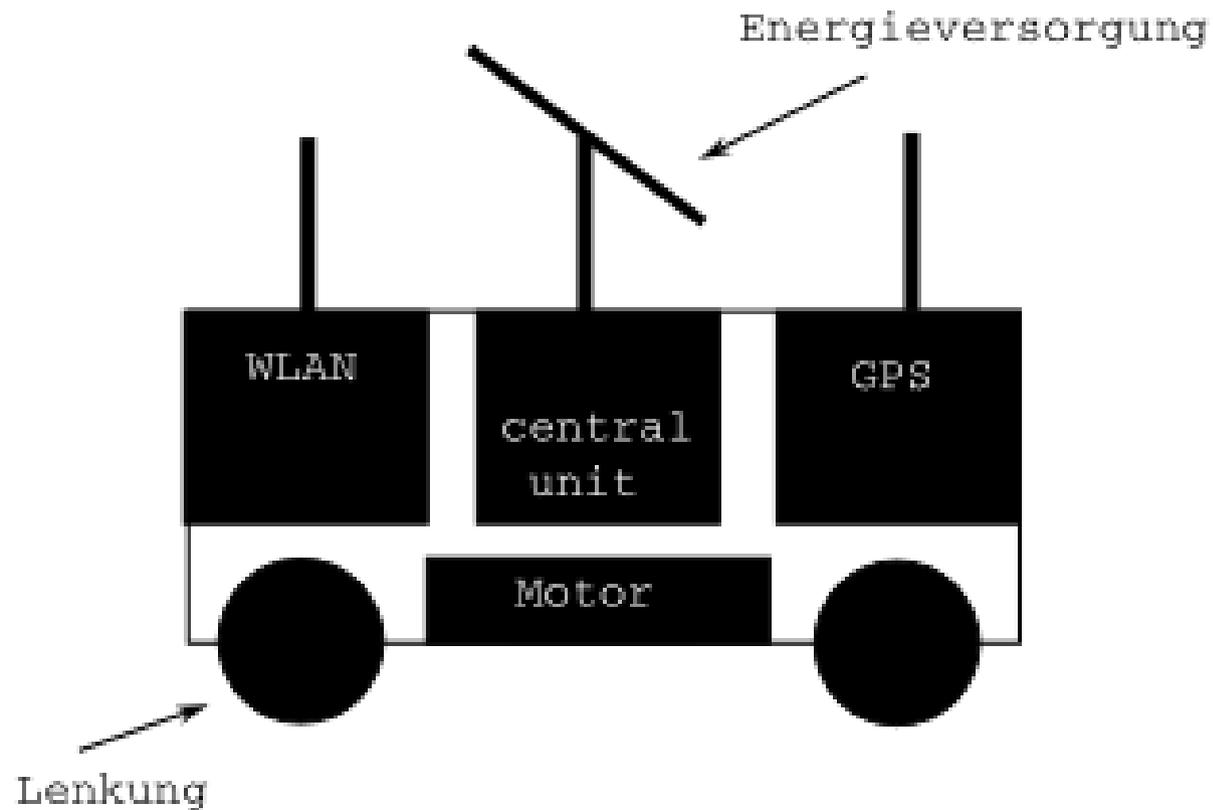
Problem:

- Positionierung der Geräte immer von Hand notwendig

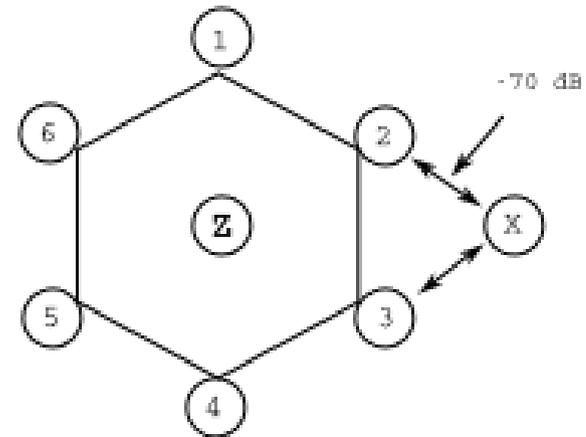
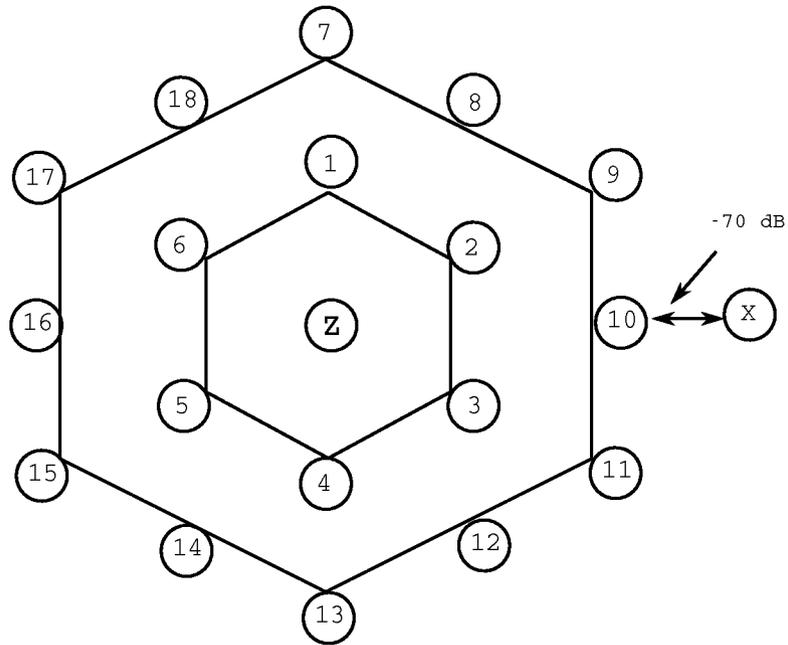
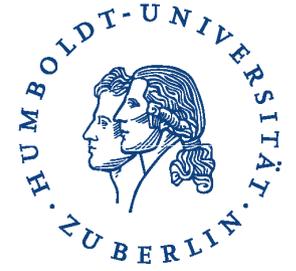
Lösung:

- autonom agierende Geräte, die um einen gegebenen Zielpunkt ein Netzwerk aufbauen

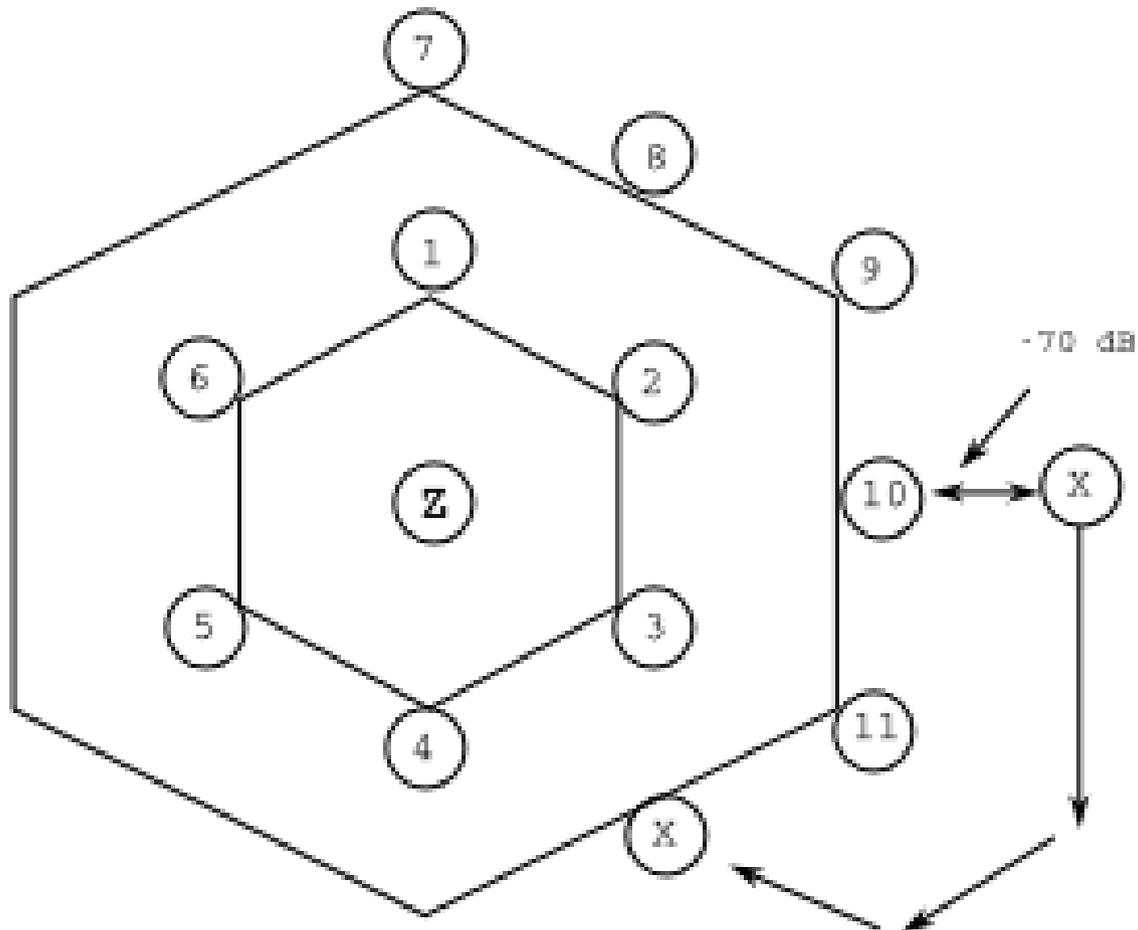
Schaubilder



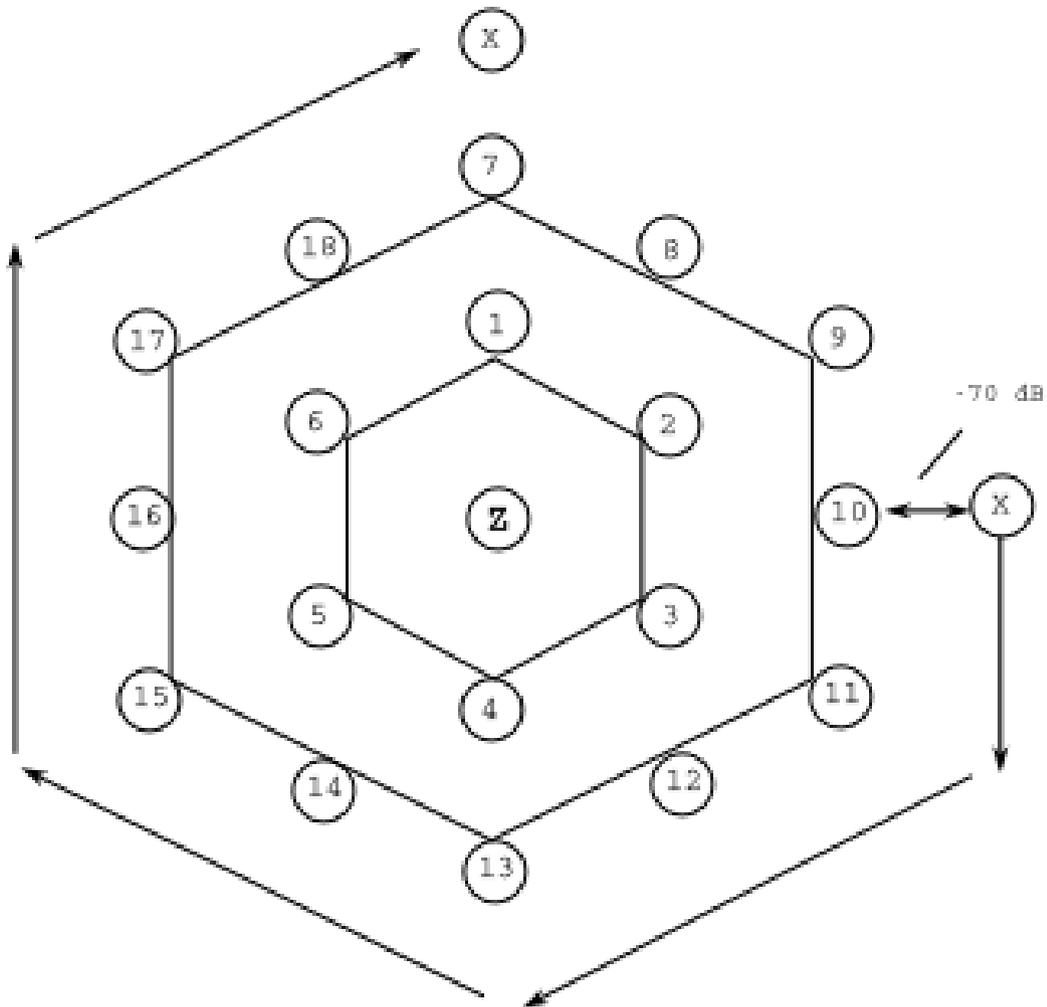
Schaubilder



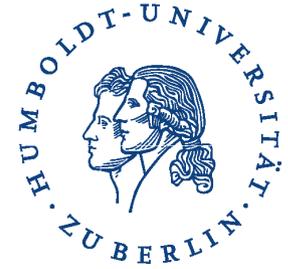
Schaubilder



Schaubilder

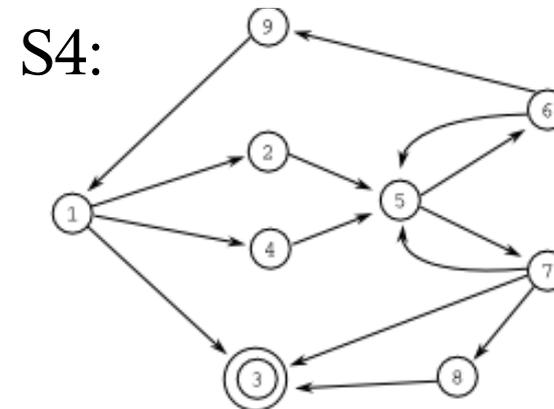
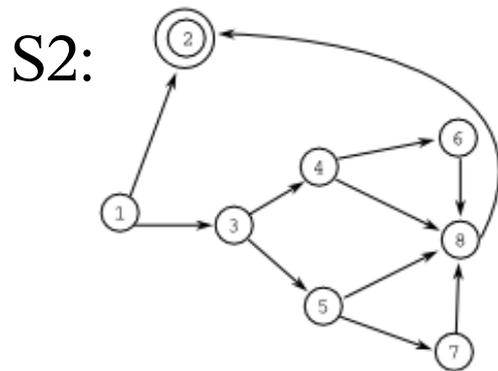
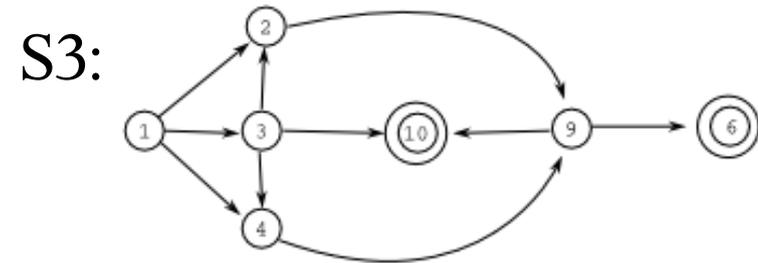
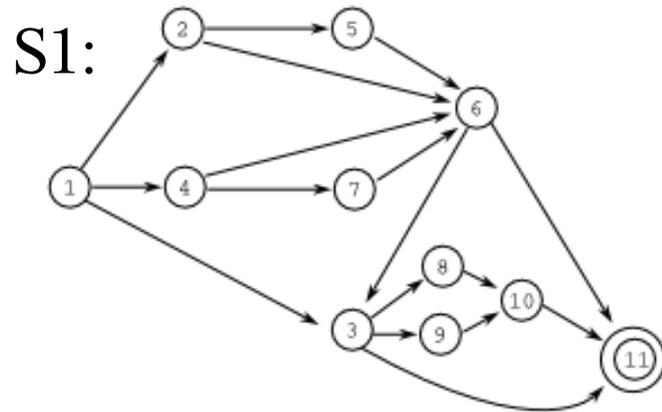
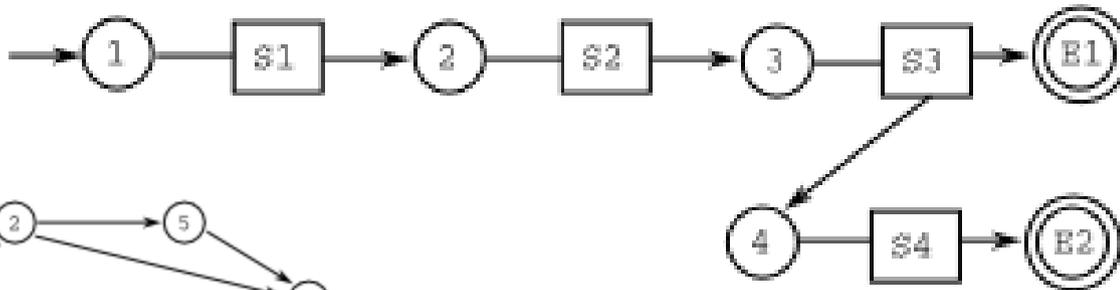
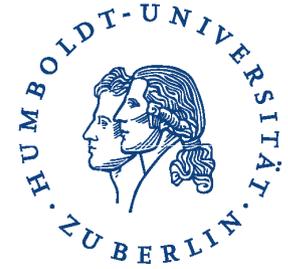


Beschränkungen

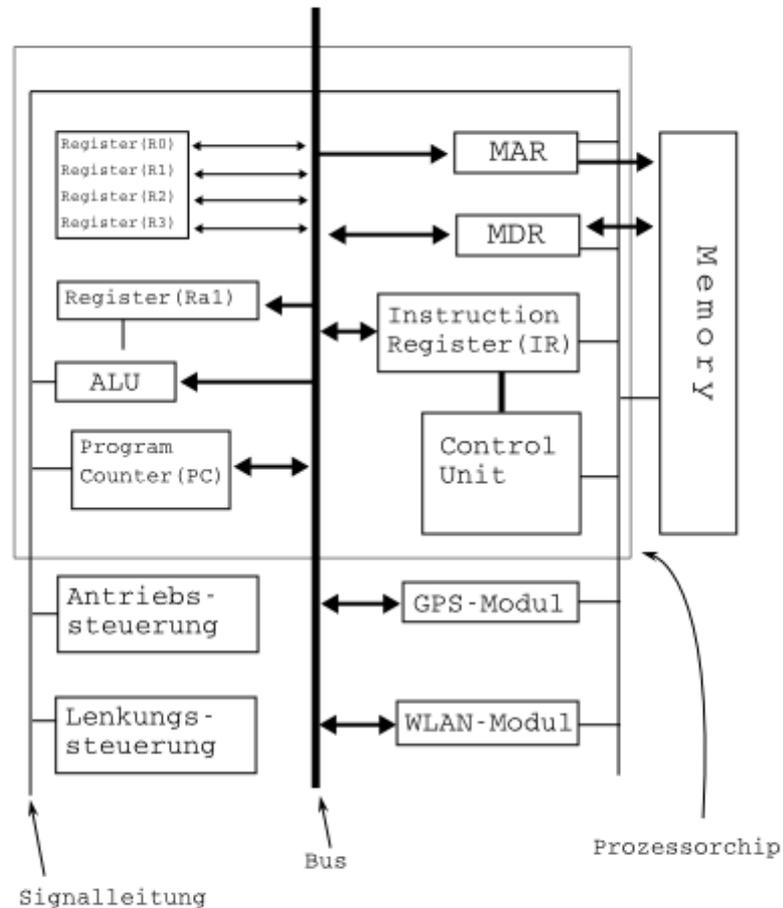
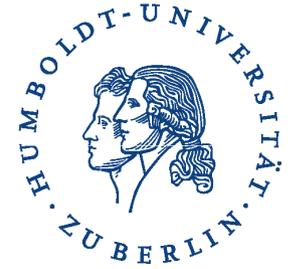


- ebener Untergrund ohne Hindernisse
- keine Kollisionen zwischen den Geräten
- kein Ausfall der Geräte
- GPS-Daten immer vorhanden
- WLAN immer mit gleicher Sendestärke und gleicher Dämpfung
- keine Berührung mit Äquator, Nullmeridian, Nordpol und Südpol
- zu einem Zeitpunkt nur ein Gerät unterwegs

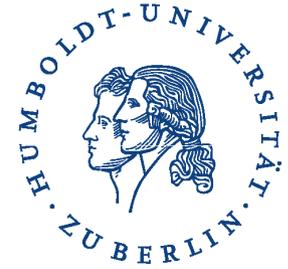
Zustandsübergangsgraphen



Schaubilder



Befehlsübersicht



- insgesamt 24 Befehle
- bei linearer Kodierung: Länge des OP-Codes = 5 Bit

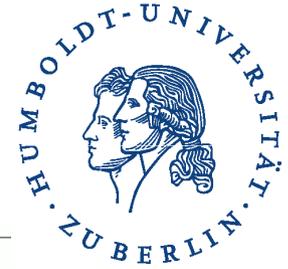
Problem:

- 3 Bit für Registeradresse
- 12 Bit für eine ganze Zahl
- Busbreite: 16 Bit

Lösung:

- Befehlskodierung in Anlehnung an Huffman-Code

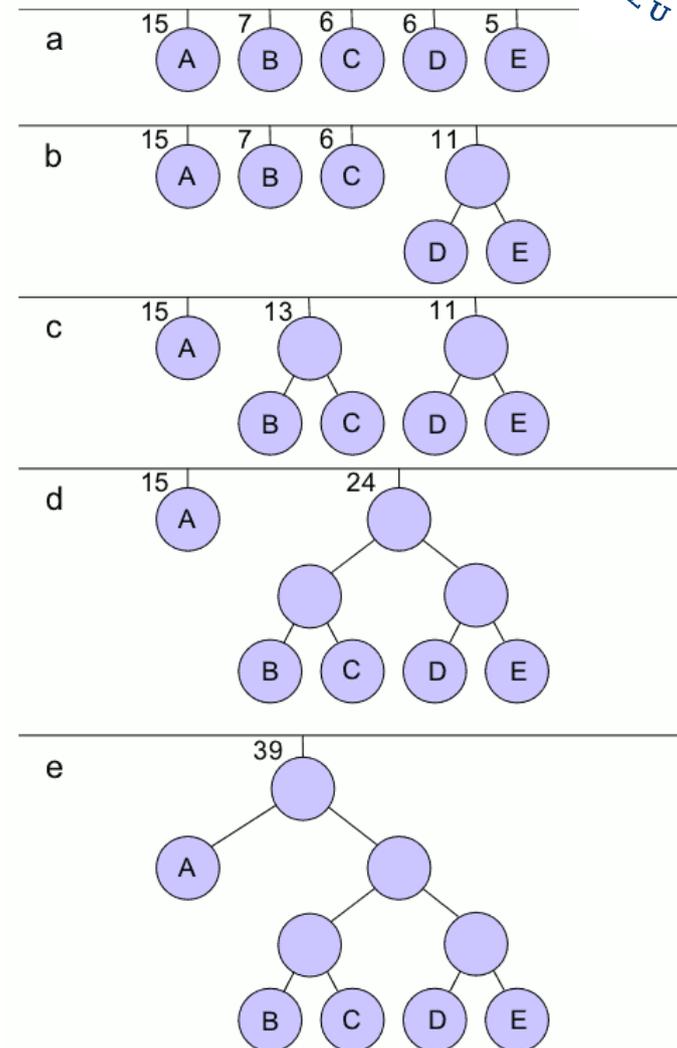
Huffman-Code



- David A. Huffman, 1952
- Entropie-Kodierung (Methode zur verlustfreien Datenkompression)
- verschieden lange Codes
- präfixfreier Code (keine Bitfolge, die für ein einzelnes Zeichen steht, darf am Anfang eines anderen Zeichens stehen)
- Kodierung wahrscheinlichkeitsabhängig

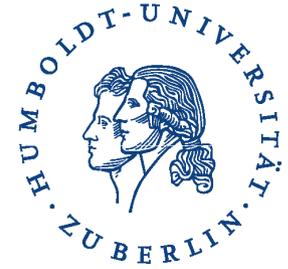
Beispiel:

- 5 Zeichen => 3 Bit pro Zeichen
- nach Huffman-Kodierung: ~ 2,28 Bit



Quelle: de.wikipedia.org/wiki/Huffman-Kodierung

Befehlsübersicht



Move-Befehl (für ganze Zahlen)

movD #n, Rx (Rx <- n)

(OP-Code: 1 => 12-Bit-Zahlen darstellbar)

Subtraktion

sub #n, Rx (Rx <- Rx - n)

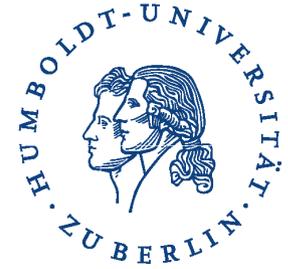
(OP-Code: 01 => 11-Bit-Zahlen darstellbar)

Sprünge

OP-Code: 001 XXX

- höherwertigen Bits immer gleich
- niedrigerwertigen Bits variabel

Beispiele aus dem Code

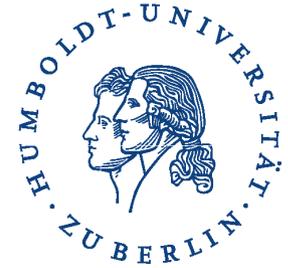


```
# Aufruf der Subroutine w_go(R1, R2)
movR PC, R3      # speichere den PC in R3
add #2, R3       # erhoehere R3 um 2
jmp w_go         # springe zur Subroutine
```

ADDR	Inhalt
n - 1	# vorheriger Befehl
n	movR PC, R3
n + 1	add #2, R3
n + 2	jmp w_go
n + 3	# nächster Befehl

```
# Ruecksprung aus der Subroutine
movR R3, PC      # verlasse die Subroutine und springe zurueck
```

Beispiele aus dem Code



Subroutine: w_go(R1, R2)

```
:w_go    AS_start    # das Geraet faehrt los
:w_g     movPR R2, WLAN # WLAN <- R2 (Befehl aus R2
                        # an wlan-modul)
                        movG WLAN, Ra1 # Ra1 <- WLAN (aktueller Wert)
                        cmp Ra1, R1    # vergleiche den aktuellen mit dem
                        # Zielwert
                        ifne w_g       # wenn der Zielwert noch nicht
                        # erreicht wurde
                        AS_stop       # Gerät anhalten
                        movR R3, PC   # verlasse die Subroutine und
                        # springe zurueck
```

Schlussbetrachtung



- Bewertung der Einsatzmöglichkeiten
- Bewertung der Einschränkungen
- Sicherheits- bzw. Fehlerbetrachtungen
- Zeitkritische Abläufe
- Testmöglichkeiten
- Mögliche Erweiterungen