

GBI Crashkurs WS 2016/17

Miguel Santos Correa

24. Februar 2017

miguel-correa@web.de

<https://github.com/misaco93/GBI-Crashkurs-16-17>

Gliederung

- 1 Hoare-Kalkül
- 2 Prädikatenlogik
- 3 MIMA

Hoare-Kalkül

Hoare-Tripel

Ein Hoare-Tripel $\{P\}S\{Q\}$ besteht aus einer Vorbedingung P , einem Programmstück S und einer Nachbedingung Q .

Beispiel:

$$\{x = a\}$$

$$y \leftarrow x$$

$$\{y = a\}$$

Hoare-Tripel

- HT-A:

Das Hoare-Tripel $\{\sigma_{x/E}(Q)\} x \leftarrow E \{Q\}$ ist gültig.

- Beispiel:

$$\{?\}$$
$$x \leftarrow E$$
$$\{y = x\}$$

Hoare-Tripel

- HT-A:

Das Hoare-Tripel $\{\sigma_{x/E}(Q)\} x \leftarrow E \{Q\}$ ist gültig.

- Beispiel:

$$\{y = E\}$$

$$x \leftarrow E$$

$$\{y = x\}$$

Hoare-Tripel

- HT-S:

Sind die Hoare-Tripel $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ gültig, dann auch das Tripel $\{P\}S_1; S_2\{R\}$.

- Beispiel:

$$\{a \leq x\}$$

$$y \leftarrow x$$

$$\{a \leq y\}$$

$$\{a \leq y\}$$

$$z \leftarrow y$$

$$\{a \leq z\}$$

Hoare-Tripel

- HT-S:

Sind die Hoare-Tripel $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ gültig, dann auch das Tripel $\{P\}S_1; S_2\{R\}$.

- Beispiel:

$$\{a \leq x\}$$

$$y \leftarrow x$$

$$z \leftarrow y$$

$$\{a \leq z\}$$

Hoare-Tripel

- HT-E:

Ist ein Hoare-Tripel $\{P\}S\{Q\}$ gültig und sind die Aussagen $P' \Rightarrow P$ und $Q \Rightarrow Q'$ wahr, dann ist auch das Hoare-Tripel $\{P'\}S\{Q'\}$ gültig.

- Beispiel:

$$\{a \leq x\}$$

$$y \leftarrow x$$

$$\{a \leq y\}$$

Hoare-Tripel

- HT-E:

Ist ein Hoare-Tripel $\{P\}S\{Q\}$ gültig und sind die Aussagen $P' \Rightarrow P$ und $Q \Rightarrow Q'$ wahr, dann ist auch das Hoare-Tripel $\{P'\}S\{Q'\}$ gültig.

- Beispiel:

$$\{a = x\}$$

$$\{a \leq x\}$$

$$y \leftarrow x$$

$$\{a \leq y\}$$

$$\{a \leq y + 1\}$$

Hoare-Tripel

- HT-E:

Ist ein Hoare-Tripel $\{P\}S\{Q\}$ gültig und sind die Aussagen $P' \Rightarrow P$ und $Q \Rightarrow Q'$ wahr, dann ist auch das Hoare-Tripel $\{P'\}S\{Q'\}$ gültig.

- Beispiel:

$$\{a = x\}$$

$$y \leftarrow x$$

$$\{a \leq y + 1\}$$

Hoare-Tripel

HT-I:

Wenn die Hoare-Tripel $\{P \wedge B\} S_1 \{Q\}$ und $\{P \wedge \neg B\} S_2 \{Q\}$ gültig sind, so ist auch das Hoare-Tripel $\{P\} \text{if } B \text{ then } S_1 \text{ else } S_2 \text{ fi } \{Q\}$ gültig.

$\{P\}$
if B
then
 $\{P \wedge B\}$
 S_1
 $\{Q\}$
else
 $\{P \wedge \neg B\}$
 S_2
 $\{Q\}$
fi
 $\{Q\}$

$\{|x| = 15\}$

if $x \geq 0$

then

$\{|x| = 15 \wedge x \geq 0\}$

$x \leftarrow x$

$\{x = 15\}$

else

$\{|x| = 15 \wedge \neg(x \geq 0)\}$

$x \leftarrow -x$

$\{x = 15\}$

fi

$\{x = 15\}$

Beispiel: Minimum

$\{x = a \wedge y = b\}$

if $x > y$

then

$\{\dots\}$

$z \leftarrow y$

$\{\dots\}$

else

$\{\dots\}$

$z \leftarrow x$

$\{\dots\}$

fi

$\{z = \min(a, b)\}$

$\{x = a \wedge y = b\}$

if $x > y$

then

$\{x = a \wedge y = b \wedge x > y\}$

$\{\dots\}$

$z \leftarrow y$

$\{\dots\}$

else

$\{\dots\}$

$z \leftarrow x$

$\{\dots\}$

fi

$\{z = \min(a, b)\}$

$\{x = a \wedge y = b\}$

if $x > y$

then

$\{x = a \wedge y = b \wedge x > y\}$

$\{\dots\}$

$z \leftarrow y$

$\{\dots\}$

else

$\{x = a \wedge y = b \wedge \neg(x > y)\}$

$\{\dots\}$

$z \leftarrow x$

$\{\dots\}$

fi

$\{z = \min(a, b)\}$

$\{x = a \wedge y = b\}$

if $x > y$

then

$\{x = a \wedge y = b \wedge x > y\}$

$\{\dots\}$

$z \leftarrow y$

$\{z = \min(a, b)\}$

else

$\{x = a \wedge y = b \wedge \neg(x > y)\}$

$\{\dots\}$

$z \leftarrow x$

$\{z = \min(a, b)\}$

fi

$\{z = \min(a, b)\}$

$\{x = a \wedge y = b\}$

if $x > y$

then

$\{x = a \wedge y = b \wedge x > y\}$

$\{y = \min(a, b)\}$

$z \leftarrow y$

$\{z = \min(a, b)\}$

else

$\{x = a \wedge y = b \wedge \neg(x > y)\}$

$\{x = \min(a, b)\}$

$z \leftarrow x$

$\{z = \min(a, b)\}$

fi

$\{z = \min(a, b)\}$

Hoare-Tripel

HT-W:

Wenn das Hoare-Tripel $\{I \wedge B\} S \{I\}$ gültig ist, so ist auch das Tripel $\{I\}$ **while** B **do** S **od** $\{I \wedge \neg B\}$ gültig.

Beispiel: While-Schleife

```
{I}  
while B  
do  
    {I ∧ B}  
    S  
    {I}  
od  
{I ∧ ¬B}
```

Beispiel: While-Schleife

```
{x ≥ 0}
while x ≤ 10
do
    {x ≥ 0 ∧ x ≤ 10}
    x ← x + 1
    {x ≥ 0}
od
{x ≥ 0 ∧ ¬(x ≤ 10)}
```

Beispiel: While-Schleife

$\{x = a \wedge y = b\}$

$\{\dots\}$

while $y \neq 0$

do

$\{\dots\}$

$y \leftarrow y - 1$

$\{\dots\}$

$x \leftarrow x + 1$

$\{\dots\}$

od

$\{\dots\}$

$\{x = a + b\}$

$\{x = a \wedge y = b\}$ $\{x + y = a + b\}$ **while** $y \neq 0$ **do** $\{\dots\}$ $y \leftarrow y - 1$ $\{\dots\}$ $x \leftarrow x + 1$ $\{\dots\}$ **od** $\{\dots\}$ $\{x = a + b\}$

$\{x = a \wedge y = b\}$ $\{x + y = a + b\}$ **while** $y \neq 0$ **do** $\{x + y = a + b \wedge y \neq 0\}$ $\{\dots\}$ $y \leftarrow y - 1$ $\{\dots\}$ $x \leftarrow x + 1$ $\{\dots\}$ **od** $\{\dots\}$ $\{x = a + b\}$

$\{x = a \wedge y = b\}$ $\{x + y = a + b\}$ **while** $y \neq 0$ **do** $\{x + y = a + b \wedge y \neq 0\}$ $\{\dots\}$ $y \leftarrow y - 1$ $\{\dots\}$ $x \leftarrow x + 1$ $\{x + y = a + b\}$ **od** $\{\dots\}$ $\{x = a + b\}$

$\{x = a \wedge y = b\}$ $\{x + y = a + b\}$ **while** $y \neq 0$ **do** $\{x + y = a + b \wedge y \neq 0\}$ $\{\dots\}$ $y \leftarrow y - 1$ $\{x + 1 + y = a + b\}$ $x \leftarrow x + 1$ $\{x + y = a + b\}$ **od** $\{\dots\}$ $\{x = a + b\}$

$\{x = a \wedge y = b\}$ $\{x + y = a + b\}$ **while** $y \neq 0$ **do** $\{x + y = a + b \wedge y \neq 0\}$ $\{x + 1 + y - 1 = a + b\}$ $y \leftarrow y - 1$ $\{x + 1 + y = a + b\}$ $x \leftarrow x + 1$ $\{x + y = a + b\}$ **od** $\{\dots\}$ $\{x = a + b\}$

$\{x = a \wedge y = b\}$ $\{x + y = a + b\}$ **while** $y \neq 0$ **do** $\{x + y = a + b \wedge y \neq 0\}$ $\{x + 1 + y - 1 = a + b\}$ $y \leftarrow y - 1$ $\{x + 1 + y = a + b\}$ $x \leftarrow x + 1$ $\{x + y = a + b\}$ **od** $\{x + y = a + b \wedge \neg(y \neq 0)\}$ $\{x = a + b\}$

Aufgabenblatt: WS 2015/16 A8.2

Beweisen Sie anhand des Hoare-Kalküls, dass das Hoare-Tripel

$$\{x = a \wedge y = b\}$$

if $x \geq y$

then

$$z \leftarrow x$$
$$x \leftarrow y$$
$$y \leftarrow z$$

else

$$x \leftarrow x$$

fi

$$\{x = \min(a, b) \wedge y = \max(a, b)\}$$

gültig ist.

$\{x = a \wedge y = b\}$

if $x \geq y$

then

$z \leftarrow x$

$x \leftarrow y$

$y \leftarrow z$

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

else

$x \leftarrow x$

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

fi

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

$\{x = a \wedge y = b\}$

if $x \geq y$

then

$\{y = \min(a, b) \wedge x = \max(a, b)\}$

$z \leftarrow x$

$x \leftarrow y$

$y \leftarrow z$

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

else

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

$x \leftarrow x$

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

fi

$\{x = \min(a, b) \wedge y = \max(a, b)\}$

$$\{x = a \wedge y = b \wedge x \geq y\}$$
$$\{y = \min(a, b) \wedge x = \max(a, b)\}$$

$$\{x = a \wedge y = b \wedge x < y\}$$
$$\{x = \min(a, b) \wedge y = \max(a, b)\}$$

Prädikatenlogik

Prädikatenlogik

Drei Schritte:

- Definiert Terme, die aus Konstanten, Variablen und Funktionssymbolen zusammengesetzt sind.
- Mit Hilfe von Relationssymbolen und Termen konstruiert man dann atomare Formeln.
- Mit Hilfe von zwei Quantoren werden allgemeine prädikatenlogische Formeln konstruiert.

Beispiel für Terme

- c
- y
- $g(x)$
- $f(x, g(z))$
- $f(c, g(g(z)))$

Atomare Formeln

- Relationssymbole mit Alphabet Rel_{PL}
- kurz R, S, \dots
- \doteq Relation immer dabei

Atomare Formeln

korrekte Formeln:

- $R(y, c, g(x))$
- $f(x, y) \doteq g(z)$
- $S(c)$

syntaktisch falsche Formeln:

- $S(x) \doteq S(x)$
- $f(R(x, c))$
- $R(S(x), c, y)$
- $x \doteq y \doteq z$

Quantoren

- Allquantor \forall
- Existenzquantor \exists
- Klammerregel: Quantoren binden am stärksten
- $(\exists x F)$ statt $(\neg(\forall x(\neg F)))$
- $A_{For} = A_{Rel} \cup \{\neg, \wedge, \vee, \rightarrow, \forall, \exists\}$
- z.B. $\forall x R(x, y) \wedge S(x)$

Interpretation

Es seien Alphabete $Const_{PL}$, Fun_{PL} und Rel_{PL} gegeben.

Sind eine Interpretation (D, I) und eine Variablenbelegung β festgelegt, so kann man

- jedem Term einen Wert aus D und
- jeder Formel einen Wahrheitswert zuordnen.

Interpretation von Termen

$$val_{D,I,\beta}(t) = \begin{cases} \beta(x_i), & \text{falls } t = x_i \in Var_{PL} \\ I(c_i), & \text{falls } t = c_i \in Const_{PL} \\ I(f_i)(val_{D,I,\beta}(t_1), \dots, val_{D,I,\beta}(t_k)), & \text{falls } t = f_i(t_1, \dots, t_k) \end{cases}$$

Interpretation von atomaren Formeln

$$val_{D,I,\beta}(R_i(t_1, \dots, t_k)) = \begin{cases} w, & \text{falls } (val_{D,I,\beta}(t_1), \dots, val_{D,I,\beta}(t_k)) \in I(R_i) \\ f, & \text{falls } (val_{D,I,\beta}(t_1), \dots, val_{D,I,\beta}(t_k)) \notin I(R_i) \end{cases}$$

$$val_{D,I,\beta}(t_1 \doteq t_2) = \begin{cases} w, & \text{falls } val_{D,I,\beta}(t_1) = val_{D,I,\beta}(t_2) \\ f, & \text{falls } val_{D,I,\beta}(t_1) \neq val_{D,I,\beta}(t_2) \end{cases}$$

Quantoren

$$\beta_{x_i}^d : Var_{PL} \rightarrow D : x_j \rightarrow \begin{cases} \beta(x_j), & \text{falls } i \neq j \\ d, & \text{falls } i = j \end{cases}$$

$$val_{D,I,\beta}(\forall x_i F) = \begin{cases} w, & \text{falls f\"ur jedes } d \in D \text{ und } \beta' = \beta_{x_i}^d : val_{D,I,\beta'}(F) = w \\ f, & \text{sonst} \end{cases}$$

$$val_{D,I,\beta}(\exists x_i F) = \begin{cases} w, & \text{falls f\"ur mind. } d \in D \text{ und } \beta' = \beta_{x_i}^d : val_{D,I,\beta'}(F) = w \\ f, & \text{sonst} \end{cases}$$

Beispiele

- $D = \mathbb{N}_0$, $I(c) = 0$, $I(f)$ Addition, $I(R)$ kleiner oder gleich
- $\beta(x) = 3$, $\beta(y) = 42$
- $val_{D,I,\beta}(R(y, c)) = ?$

Beispiele

- $D = \mathbb{N}_0$, $I(c) = 0$, $I(f)$ Addition, $I(R)$ kleiner oder gleich
- $\beta(x) = 3$, $\beta(y) = 42$
- $val_{D,I,\beta}(R(y, c)) = f$
- weil $\beta(y) > I(c)$

Beispiele

- $D = \{a, b\}^+$, $I(c) = bb$, $I(f)$ Konkatination
- $I(R)$ hat gleich viele a's
- $\beta(x) = a$, $\beta(y) = abba$
- $val_{D,I,\beta}(f(f(x, c), y)) = ?$
- $val_{D,I,\beta}(R(f(y, x), c)) = ?$

Beispiele

- $D = \{a, b\}^+$, $I(c) = bb$, $I(f)$ Konkatenation
- $I(R)$ hat gleich viele a's
- $\beta(x) = a$, $\beta(y) = abba$
- $val_{D,I,\beta}(f(f(x, c), y)) = abbabba$
- $val_{D,I,\beta}(R(f(y, x), c)) = f$

Allgemeingültigkeit

Eine prädikatenlogische Formel heißt allgemeingültig, wenn (D, I) und jede passende Variablenbelegung β gilt: $val_{D,I,\beta}(F) = w$.

Bsp.

$$(\forall x_i (G \rightarrow H)) \rightarrow ((\forall x_i G) \rightarrow (\forall x_i H))$$

freie und gebundene Vorkommen

Wenn in einer prädikatenlogischen Formel G in einem Term eine Variable x steht, dann spricht man auch von einem Vorkommen der Variablen x in G . (Die Anwesenheit einer Variablen unmittelbar hinter einem Quantor zählt nicht als Vorkommen.)

freie und gebundene Vorkommen

- Für jede Formel G , die atomar ist, sind alle Vorkommen von Variablen frei
- Für jede Formel G der Form $(\forall x_i H)$ oder $(\exists x_i H)$ ist jedes Vorkommen von x in H gebunden
- Beispiel: $\forall x(R(x, y) \wedge \exists yR(x, y))$

Substitutionen

Es ist möglich Variablen einer prädikatenlogischen Formel durch Terme zu ersetzen. Eine Substitution ist eine Abbildung $\sigma : Var_{PL} \rightarrow L_{Ter}$

$\sigma_{\{x/c, y/f(x)\}}$:

$$\sigma(x) = c$$

$$\sigma(y) = f(x)$$

$$\sigma(z) = z, z \notin \{x, y\}$$

Kollisionsfrei

Eine Substitution σ heie kollisionsfrei fr eine Formel G , wenn fr jede Variable x_i , die durch σ verndert wird (also $\sigma(x_i) \neq x_i$) und jede Stelle eines freien Vorkommens von x_i in G gilt: Diese Stelle liegt nicht im Wirkungsbereich eines Quantors $\forall x_j$ oder $\exists x_j$, wenn x_j eine Variable ist, die in $\sigma(x_i)$ vorkommt.

Beispiel

Kollisionsfrei:

- $G = S(x) \wedge \forall x(R(x, y))$
- $\sigma_{\{x/f(y), y/c\}}(G) = S(f(y)) \wedge \forall x(R(x, c))$

nicht Kollisionsfrei:

- $G = \exists y(R(y, c) \wedge R(x, c))$
- $\sigma_{\{x/f(y), y/c\}}(G) = \exists y(R(y, c) \wedge R(f(y), c))$

Klausuraufgabe: WS 2015/16 A2

Beantworten Sie für jede der folgenden prädikatenlogischen Formeln die Frage: „Ist die Formel allgemeingültig?“

- (i) $(\neg \exists x : P(x)) \leftrightarrow (\forall x : \neg P(x))$
- (ii) $(\forall x \exists y \exists z : Q(x, y, z)) \rightarrow (\exists y \forall x \exists z : Q(x, y, z))$
- (iii) $(\exists z \exists y \forall x : Q(x, y, z)) \rightarrow (\forall x \exists y \exists z : Q(x, y, z))$

Dabei ist P ein einstelliges Relationssymbol und Q ein dreistelliges Relationssymbol.

Klausuraufgabe: WS 2015/16 A2

Beantworten Sie für jede der folgenden prädikatenlogischen Formeln die Frage: „Ist die Formel allgemeingültig?“

- (i) $(\neg \exists x : P(x)) \leftrightarrow (\forall x : \neg P(x))$
- (ii) $(\forall x \exists y \exists z : Q(x, y, z)) \rightarrow (\exists y \forall x \exists z : Q(x, y, z))$
- (iii) $(\exists z \exists y \forall x : Q(x, y, z)) \rightarrow (\forall x \exists y \exists z : Q(x, y, z))$

Dabei ist P ein einstelliges Relationssymbol und Q ein dreistelliges Relationssymbol.

- (i) Ja

Klausuraufgabe: WS 2015/16 A2

Beantworten Sie für jede der folgenden prädikatenlogischen Formeln die Frage: „Ist die Formel allgemeingültig?“

- (i) $(\neg \exists x : P(x)) \leftrightarrow (\forall x : \neg P(x))$
- (ii) $(\forall x \exists y \exists z : Q(x, y, z)) \rightarrow (\exists y \forall x \exists z : Q(x, y, z))$
- (iii) $(\exists z \exists y \forall x : Q(x, y, z)) \rightarrow (\forall x \exists y \exists z : Q(x, y, z))$

Dabei ist P ein einstelliges Relationssymbol und Q ein dreistelliges Relationssymbol.

- (i) Ja
- (ii) Nein

Klausuraufgabe: WS 2015/16 A2

Beantworten Sie für jede der folgenden prädikatenlogischen Formeln die Frage: „Ist die Formel allgemeingültig?“

- (i) $(\neg \exists x : P(x)) \leftrightarrow (\forall x : \neg P(x))$
- (ii) $(\forall x \exists y \exists z : Q(x, y, z)) \rightarrow (\exists y \forall x \exists z : Q(x, y, z))$
- (iii) $(\exists z \exists y \forall x : Q(x, y, z)) \rightarrow (\forall x \exists y \exists z : Q(x, y, z))$

Dabei ist P ein einstelliges Relationssymbol und Q ein dreistelliges Relationssymbol.

- (i) Ja
- (ii) Nein
- (iii) Ja

Klausuraufgabe: WS 2015/16 A2

Formulieren Sie die folgenden Aussagen als prädikatenlogische Formeln über dem Universum aller Menschen:

- (i) Jeder Student außer Tom lächelt.
- (ii) Jeder mag jeden, der sich nicht selbst mag.

Klausuraufgabe: WS 2015/16 A2

Formulieren Sie die folgenden Aussagen als prädikatenlogische Formeln über dem Universum aller Menschen:

(i) Jeder Student außer Tom lächelt.

(ii) Jeder mag jeden, der sich nicht selbst mag.

(i) $\forall x(student(x) \rightarrow (\neg Tom(x) \leftrightarrow lächelt(x)))$

Klausuraufgabe: WS 2015/16 A2

Formulieren Sie die folgenden Aussagen als prädikatenlogische Formeln über dem Universum aller Menschen:

(i) Jeder Student außer Tom lächelt.

(ii) Jeder mag jeden, der sich nicht selbst mag.

(i) $\forall x (student(x) \rightarrow (\neg Tom(x) \leftrightarrow lächelt(x)))$

(ii) $\forall x \forall y (\neg mag(y, y) \rightarrow mag(x, y))$

Klausuraufgabe: WS 2015/16 A2

Gegeben sei die prädikatenlogische Formel

$$\forall x \forall y (R(x, y) \rightarrow R(f(x), f(y)))$$

und eine Interpretation (D, I) dafür, wobei das Universum D die Menge $\{a, b\}$ sei und die Interpretationsabbildung I gegeben sei durch $I(f)(a) = b$, $I(f)(b) = a$ und $I(R) = \{(a, a), (a, b)\}$.

- (i) Geben Sie den Wahrheitswert der Formel in der Interpretation an.
- (ii) Erläutern sie kurz Ihre Antwort aus Teil (i):

Klausuraufgabe: WS 2015/16 A2

Gegeben sei die prädikatenlogische Formel

$$\forall x \forall y (R(x, y) \rightarrow R(f(x), f(y)))$$

und eine Interpretation (D, I) dafür, wobei das Universum D die Menge $\{a, b\}$ sei und die Interpretationsabbildung I gegeben sei durch $I(f)(a) = b$, $I(f)(b) = a$ und $I(R) = \{(a, a), (a, b)\}$.

- (i) Geben Sie den Wahrheitswert der Formel in der Interpretation an.
- (ii) Erläutern sie kurz Ihre Antwort aus Teil (i):

Wahrheitswert: Falsch.

MIMA

MIMA

- Adressen: 20 bit
- Speicherwerte: 24 bit
- (die meisten) Befehle: 4 bit + 20 bit
- z.B. LDC 10

Beispiel: Multiplikation

```
LDC 0
STV prod
start : LDC 0
        NOT
        ADD b
        STV b
        JMN end
        LDV prod
        ADD a
        STV prod
        JMP start
end : HALT
```

Befehle

- **LDC const** Lädt eine 20 bit Zahl in den Akkumulator.

Befehle

- **LDC const** Lädt eine 20 bit Zahl in den Akkumulator.
- **LDV adr** Lädt den Speicherwert von adr in den Akkumulator.

Befehle

- **LDC const** Lädt eine 20 bit Zahl in den Akkumulator.
- **LDV adr** Lädt den Speicherwert von adr in den Akkumulator.
- **STV adr** Speichert den Wert vom Akkumulator in adr.

Befehle

- **LDC const** Lädt eine 20 bit Zahl in den Akkumulator.
- **LDV adr** Lädt den Speicherwert von adr in den Akkumulator.
- **STV adr** Speichert den Wert vom Akkumulator in adr.
- **LDIV adr** Lädt den Speicherwert vom Speicherwert von adr $M(M(adr))$ in den Akkumulator.

Befehle

- **LDC const** Lädt eine 20 bit Zahl in den Akkumulator.
- **LDV adr** Lädt den Speicherwert von adr in den Akkumulator.
- **STV adr** Speichert den Wert vom Akkumulator in adr.
- **LDIV adr** Lädt den Speicherwert vom Speicherwert von adr $M(M(adr))$ in den Akkumulator.
- **STIV adr** Speichert den Wert vom Akkumulator in $M(M(adr))$.

Befehle

- **LDC const** Lädt eine 20 bit Zahl in den Akkumulator.
- **LDV adr** Lädt den Speicherwert von adr in den Akkumulator.
- **STV adr** Speichert den Wert vom Akkumulator in adr.
- **LDIV adr** Lädt den Speicherwert vom Speicherwert von adr $M(M(adr))$ in den Akkumulator.
- **STIV adr** Speichert den Wert vom Akkumulator in $M(M(adr))$.
- **ADD adr** Addiert den Speicherwert von adr auf den Akkumulator und speichert das Ergebnis im Akkumulator.

Befehle

- **AND adr** Bitweise AND vom Speicherwert von adr mit dem Akkumulator. Ergebnis wird im Akkumulator gespeichert.

Befehle

- **AND adr** Bitweise AND vom Speicherwert von adr mit dem Akkumulator. Ergebnis wird im Akkumulator gespeichert.
- **OR adr** Bitweise OR.

Befehle

- **AND adr** Bitweise AND vom Speicherwert von adr mit dem Akkumulator. Ergebnis wird im Akkumulator gespeichert.
- **OR adr** Bitweise OR.
- **XOR adr** Bitweise XOR.

Befehle

- **AND adr** Bitweise AND vom Speicherwert von adr mit dem Akkumulator. Ergebnis wird im Akkumulator gespeichert.
- **OR adr** Bitweise OR.
- **XOR adr** Bitweise XOR.
- **NOT** Invertiert die Bits des Akkumulators.

Befehle

- **RAR** Rotation der Akku-Bits nach rechts. Beispiel:
101100 → 010110

Befehle

- **RAR** Rotation der Akku-Bits nach rechts. Beispiel:
101100 → 010110
- **EQL adr** Vergleicht den Speicherwert von adr mit dem Akkumulator. Wenn die Zahlen gleich sind wird die Zweierkomplementdarstellung von -1 im Akkumulator gespeichert, wenn nicht dann wird die Zahl 0 im Akkumulator gespeichert.

Befehle

- **RAR** Rotation der Akku-Bits nach rechts. Beispiel:
101100 → 010110
- **EQL adr** Vergleicht den Speicherwert von adr mit dem Akkumulator. Wenn die Zahlen gleich sind wird die Zweierkomplementdarstellung von -1 im Akkumulator gespeichert, wenn nicht dann wird die Zahl 0 im Akkumulator gespeichert.
- **JMP adr** Programm springt an die Adresse adr und setzt mit dem Befehl in adr fort.

Befehle

- **RAR** Rotation der Akku-Bits nach rechts. Beispiel:
101100 → 010110
- **EQL adr** Vergleicht den Speicherwert von adr mit dem Akkumulator. Wenn die Zahlen gleich sind wird die Zweierkomplementdarstellung von -1 im Akkumulator gespeichert, wenn nicht dann wird die Zahl 0 im Akkumulator gespeichert.
- **JMP adr** Programm springt an die Adresse adr und setzt mit dem Befehl in adr fort.
- **JMN adr** Programm springt an die Adresse adr, falls der Akkumulator negativ ist.

Wichtige Blöcke

Lade -1:

LDC 0

NOT

Wichtige Blöcke

Addiere Konstante c:

LDC c

ADD a

STV a

Wichtige Blöcke

Addiere negative Konstante $-c$:

LDC $c - 1$

NOT

ADD a

STV a

Wichtige Blöcke

Addiere negative Konstante -5 :

LDC 4

NOT

ADD a

STV a

Wichtige Blöcke

Verwandle $a \rightarrow -a$:

LDV a

NOT

STV a

LDC 1

ADD a

STV a

Wichtige Blöcke

n Schleifendurchläufe:

```
start :LDC 0  
      NOT  
      ADD n  
      STV n  
      JMN end  
      Block  
      JMP start
```

Beispiel: Division

LDC 0
STV *div*
LDV *b*
NOT
STV *b*
LDC 1
ADD *b*
STV *b*

start : LDV *a*
 ADD *b*
 STV *a*
 JMN *end*
 LDC 1
 ADD *div*
 STV *div*
 JMP *start*
end : HALT

Aufgabenblatt: WS 2015/16 A5.2

Es seien a_1 und a_2 zwei verschiedene 20bit Adressen. Im Speicher stehe in Adresse a_1 die Zweierkomplementdarstellung einer nicht-negativen ganzen Zahl x , für die 2^x mit 24bit in Zweierkomplementdarstellung darstellbar ist. Ergänzen Sie die fehlenden Konstanten und Adressen im unvollständigen Minimalmaschinenprogramm derart, dass nach dessen Ausführung 2^x in Zweierkomplementdarstellung im Speicher bei Adresse a_2 steht. Beachten Sie, dass alle arithmetischen Ausdrücke, in denen x vorkommt, keine Konstanten sind, und, dass $2^0 = 1$ gilt.

Aufgabenblatt: WS 2015/16 A5.2

```
LDC
STV
while :LDC
  NOT
  ADD
  STV
  JMN end
  LDV
  ADD
  STV
  JMP while
end :HALT
```


Aufgabenblatt: WS 2015/16 A5.2

```
LDC 1
STV a2
while :LDC
  NOT
  ADD
  STV
  JMN end
  LDV
  ADD
  STV
  JMP while
end :HALT
```

Aufgabenblatt: WS 2015/16 A5.2

```
LDC 1
STV a2
while :LDC 0
    NOT
    ADD a1
    STV a1
    JMN end
LDV
ADD
STV
JMP while
end :HALT
```

Aufgabenblatt: WS 2015/16 A5.2

```
LDC 1
STV a2
while :LDC 0
    NOT
    ADD a1
    STV a1
    JMN end
    LDV a2
    ADD a2
    STV a2
    JMP while
end :HALT
```

Aufgabenblatt: WS 2014/15 A5.2

Es seien a_1 und a_2 zwei verschiedene Adressen. Welche Zahlen in Zweierkomplementdarstellung stehen nach Ausführung des Programms in den Adressen a_1 und a_2 im Speicher?

LDV a_1

XOR a_2

STV a_1

LDV a_2

XOR a_1

STV a_2

LDV a_1

XOR a_2

STV a_1

Aufgabenblatt: WS 2014/15 A5.2

Beispiel mit zwei 8 bit Zahlen:

a_1	a_2
11101001	00100111
11001110	00100111
11001110	11101001
00100111	11101001

Aufgabenblatt: WS 2014/15 A5.2

Beispiel mit zwei 8 bit Zahlen:

a_1	a_2
11101001	00100111
11001110	00100111
11001110	11101001
00100111	11101001

Die Speicherwerte der Adressen a_1 und a_2 werden vertauscht.