**Aufgabe 5**

1. Inhalt von PC: P1

Inhalt von SZ P1: Init S2

Inhalt von SZ S2: leer

**Fetch:**

Hole den Inhalt von SZ P1 über RM nach IR

Setze Befehlszähler PC auf den Nächsten Adresswert

**Decode:**

Erkenne, dass S1 initialisiert werden soll.

**Execute:**

Setzt S2 = 0;

**Write-Back:**

Schreibe 0 von AR über WM nach SZ S2

1. Inhalt von PC: P3

Inhalt von SZ P3: Add S2, S1

Inhalt von SZ S2: 0

**Fetch:**

Hole den Inhalt von SZ P3 über RM nach IR

Setze Befehlszähler PC auf den nächsten Adresswert

**Decode:**

Erkenne, dass es sich um eine Addition aus S2 und S1 handelt, die als neuer Wert in SZ S2 gespeichert werden soll.

Lade Inhalt (0) von SZ S2 über RM in ersten AR

Lade Inhalt [unbekannt] von SZ S1 über RM in zweiten AR

**Execute:**

Führe Addition aus (0 + [unbekannt] = [unbekannt])

**Write-Back:**

Schreibe Ergebnis [unbekannt] von AR über WM nach SZ S2

1. Inhalt von PC: P2

Inhalt von SZ P2: SPRUNG0 P6, S1

Inhalt von SZ S6: [unbekannt]

**Fetch:**

Hole den Inhalt von SZ P2 über RM nach IR

**Decode:**

Erkenne, dass es sich um eine Sprung-Anweisung handelt, wenn S1 = 0 ist.

Lade Inhalt [unbekannt] von SZ S1 über RM nach AR

Wenn der Wert von S1 = 0, dann

**Execute:**

Führe Sprung-Anweisung aus (Springe zu SZ P6)

Setze Befehlszähler PC auf den Adresswert P6

**Write-Back:**

[leer]

**Aufgabe 6:**

**a)**

**Decode:**

Erkenne, dass es sich um eine Ausgabe-Anweisung [P6] von S2 handelt.

Lade Inhalt [unbekannt] von SZ S2 über RM nach AR

Wenn der Wert von S1 = 0, dann

Steuerwerk ‘schreibt‘ S2 nach AM

Steuerwerk setzt D auf Lesen (0)

Steuerwerk sendet A

Speicherwerk ‘liest‘ Adresse von AM (also S2)

Speicherwerk sendet Inhalt von S2 an WM

Speicherwerk sendet T

**b)**

**Decode:**  
 Erkenne, dass es sich um einen Lese-Befehl handelt.  
 P1: INIT S2

Steuerwerk ’schreibt’ S2 nach AM

Steuerwerk ’schreibt’ Wert 0 nach WM

Steuerwerk setzt D auf Schreiben (1)

Steuerwerk sendet A

Speicherwerk ’liest’ Adresse von AM (also S2)

Speicherwerk ’liest’ Wert von WM

Speicherwerk überschreibt Inhalt der SZ S2 mit diesem Wert

Speicherwerk sendet T

**c)**

Schreiben des Ergebnisses im Write-Back der Abarbeitung von P3.  
Schreibe Ergebnis [unbekannt] von AR über WM nach SZ S2

Steuerwerk ’schreibt’ S2 nach AM

Steuerwerk ’schreibt’ Ergebnis der Addition von S2 und S1 von AR nach WM

Steuerwerk setzt D auf Schreiben (1)

Steuerwerk sendet A

Speicherwerk ’liest’ Adresse von AM (also S2)

Speicherwerk ’liest’ Wert von WM

Speicherwerk überschreibt Inhalt der SZ S2 mit diesem Wert

Speicherwerk sendet T

**Aufgabe 7:**

a)

Der Wert welcher S1 zugewiesen ist wird mit S2 addiert und dann um 1 verringert. Dies geschieht bis S1 den wert 0 erreicht hat und S2 also sie Summe von n=0 bis n.

b)

a \* 5 + 1 - b == 0

P1: ADD S1, S1

Eingabeparameter: S1 = a

S2 = b

Das Maschinenprogramm beginn damit S1 (a) fünfmal auf zu addieren, bzw. mit 5 zu multiplizieren. Das Ergebnis wird nun um 1 erhöht und mit S2 (b) subtrahiert.

Falls S1 gleich null ist wird S1 noch um 1 erhöht bevor S1 ausgegeben wird (WAHR), falls S1 jedoch nicht gleich Null ist, wird S1 mit S1 subtrahiert (also gleich 0 gesetzt) und so direkt ausgegeben (FALSCH)

P2: ADD S1, S1

P3: ADD S1, S1

P4: ADD S1, S1

P5: INCREMENT S1

P6: SUB S1, S2

P7: SPRUNG0 P10, S1

P8: SUB S1, S1

P9: SPRUNG0 P11, S1

P10: INCREMENT S1

P11: RÜCKGABE S1

c)

Das Maschinenprogramm initialisiert die beiden Variablen S2 und S3.

S2 wird der wert von S1 zugewiesen um diesen mit den beiden im Datenspeicher vermerkten Werten D1 und D2 zu vergleichen ohne den, S1 zugeordneten Wert zu verlieren.

Nun wird verglichen ob S1 größer oder kleiner als die Zahlen 48 bis 58 ist (die ASCII – Zahlen von arabischen Ziffern).

Dementsprechend wird S3 entweder gleich als 0 ausgegeben oder um 1 erhöht und dann ausgegeben.

P1: INIT S2

P2: INIT S3

P3: S2 ADD S1

P4: SUB S2, D1

P5: SPRUNGNEG S2, P11

P6: INIT S2

P7: S2 ADD S1

P8: SUB D2, S2

P9: SPRUNGNEG S2, P11

P10: INCREMENT S3

P11: RÜCKGABE S3

D1: 47

D2: 58

**Aufgabe 8**

a)

Variable a entspricht der Adresse S1, Variable b entspricht der Adresse S2

Die Funktion vergleicht die Eingabe a mit 0, wenn a < 0 wird -1 ausgegeben, wenn a > 0 wird 1 ausgegeben, wenn a == 0 wird 0 ausgegeben.

#include <stdio.h>

#include <stdlib.h>

int compare\_to\_zero(int a);

int main(void)

{

int a;

scanf("%i", &a);

printf("%i\n", compare\_to\_zero(a));

return 0;

}

int compare\_to\_zero(int a)

{

int b = 0;

if(a == 0){

return b;

}else if(a <= 0){

return --b;

}else{

return ++b;

}

}

b)

Die Funktion berechnet a mod b.

Variable a entspricht der Adresse S1, Variable b entpricht S2

#include <stdio.h>

int mod(int a, int b);

int main(void)

{

int a, b;

printf("Eingabe für a: ");

scanf("%i", &a);

printf("\nEingabe für b: ");

scanf("%i", &b);

printf("Ergebnis: %i\n", sum(a, b));

return 0;

}

int mod(int a, int b)

{

int c = 0;

c = a - b;

while(c >= 0){

a -= b;

c = 0;

c = a - b;

}

return a;

}

c)

Die Funktion berechnet die Summe

Die Variable a entspricht der Adresse S1, die Variable b entspricht der Adresse S2, Variable c entspricht der Adresse S3 und Variable d entspricht der Adresse D1

#include <stdio.h>

int sum2a(int a);

int main(void)

{

int a;

printf("Eingabe von a\n");

scanf("%i", &a);

printf("a: %i\tf(a): %i\n", a, sum2a(a));

return 0;

}

int sum2a(int a)

{

int b = 0;

int c = 0;

int d = 2;

c += d;

while(a != 0){

--a;

b += c;

c += d;

}

return b;

}