## Johnny\_1 Lineare Programme

Die Lösungen zeigt dir Konrad Zuse.

|  |  |  |  |
| --- | --- | --- | --- |
| Video | Vorkenntnisse | Hilfsmittel |  |
| johnny\_1.mp4 | Aufbau des von-Neumann-Rechners | Johnny Doku.pdf  Subtraktion | S. 7 - 10::johnny viedos:johnny sub.png |

1. Informiere dich über die Bedienung des von-Neumann-Simulators Johnny im Video johnny\_1.mp4. Du kannst parallel auch die Aufgabe 3 lösen.
2. Lies die angegebenen Seiten der Johnny-Dokumentation.  
   Notiere die Wirkung der folgenden Makro(Assembler)-Befehle:

|  |  |
| --- | --- |
| TAKE |  |
| SAVE |  |
| ADD / SUB |  |
| INC / DEC |  |

1. Implementiere das Programm aus dem Video und teste die Ausführung.  
     
   **000: TAKE 004  
   001: ADD 005  
   002: SAVE 006  
   003: HLT 000  
   004: 012  
   005: 008**

Notiere die Adresse der Speicherzelle des Ergebnisses: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Modifiziere die Anweisung in der Speicherzelle **001:** auf den Befehl  **SUB 005.** Teste das modifizierte Programm**.**  
   Notiere bei der gegebenen Speicherbelegung den Term, der nun berechnet wird. Notiere das mathematisch korrekte und das berechnete Ergebnis.

|  |  |
| --- | --- |
| Speicherbelegung |  |
| Term |  |
| mathematisch korrektes Ergebnis |  |
| berechnetes Ergebnis |  |

1. Modifiziere das Programm aus Aufgabe 4 so, dass die Operanden vertauscht werden.  
   Notiere bei der gegebenen Speicherbelegung den Term, der nun berechnet wird. Notiere das mathematisch korrekte und das berechnete Ergebnis.

|  |  |
| --- | --- |
| Speicherbelegung |  |
| Term |  |
| mathematisch korrektes Ergebnis |  |
| berechnetes Ergebnis |  |

1. Entwickle ein kurzes Programm, das die Summe aus dem in der **Zelle 10** gespeicherten Wert und seinen beiden Nachfolgern berechnet und in der **Zelle 11** abspeichert.  
   Beachte, dass du nur einen Wert (in **Zelle 10**) vorgibst. Das Programm soll die zuaddierenden Nachfolger eigenständig bestimmen.  
   Hinweis: Falls dein Programm länger wird, kannst du statt der Zellen 10 ,11 die Zellen 20 und 21 verwenden,  
     
   Beispiel : **10: 7 --> 11: 7 + 8 + 9 = 24** oder **10: 5 --> 11: 5 + 6 + 7 = 18**  
   Notiere den Quelltext.

|  |  |  |
| --- | --- | --- |
| Adr | Asm | Op |
| 0 |  |  |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |
| 12 |  |  |
| 13 |  |  |
| 14 |  |  |
| 15 |  |  |
| 16 |  |  |
| 17 |  |  |
| 18 |  |  |
| 19 |  |  |
| 20 |  |  |
| 21 |  |  |

## Johnny\_2 Sprunganweisungen

Die Lösungen zeigt dir die Z3.

|  |  |  |  |
| --- | --- | --- | --- |
| Video | Vorkenntnisse | Hilfsmittel |  |
| johnny\_2.mp4 | Johnny\_1 | Johnny Doku.pdf | S. 7 - 10 |

1. Informiere dich über die Wirkung von Sprunganweisungen des von-Neumann-Simulators Johnny im Video. johnny\_2.mp4   
   Du kannst parallel auch die Aufgabe 3 lösen.
2. Lies die angegebenen Seiten der Dokumentation.  
   Notiere die Wirkung der folgenden Makro(Assembler)-Befehle:

|  |  |
| --- | --- |
| TST |  |
| JMP |  |

1. Implementiere das Programm zur Bestimmung des Max zweier Speicherzellen aus dem Video.  
     
   **000: TAKE 017  
   001: SUB 018  
   002: SAVE 021  
   003: TST 021  
   004: JMP 008  
   005: TAKE 018  
   006: SAVE 020  
   007: HLT 000  
   008: TAKE 017  
   009: SAVE 020  
   010: HLT 000**
2. Notiere die Adresse der Speicherzelle des Ergebnisses: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Teste das Programm aus Aufgabe 3 unter folgenden Bedingungen:  
   a) In **017** steht ein Wert, der größer ist, als der Wert in **018**.  
   b) In **017** steht ein Wert, der kleiner ist, als der Wert in **018**.  
   c) In **017** und **018** steht jeweils der gleiche Wert.  
   Notiere die Speicherbelegungen und kennzeichne die Speicherzelle, deren Wert in die Zelle **020** kopiert wird:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **(017)** | **(018)** | **(020)** |
| a) |  |  |  |
| b) |  |  |  |
| c) |  |  |  |

1. Modifiziere das Programm aus Aufgabe 3 so, dass das Minimum der Speicherzellen **017** und **018** in der Zelle **020** gespeichert wird.
2. Modifiziere das Programm aus Aufgabe 3 so, dass das Maximum der aus den Zellen **017**, **018** und **019** in der Zelle **020** gespeichert wird.  
   Notiere den Quelltext.

|  |  |  |
| --- | --- | --- |
| Adr | Asm | Op |
| 0 |  |  |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |
| 12 |  |  |
| 13 |  |  |
| 14 |  |  |
| 15 |  |  |
| 16 |  |  |
| 17 |  |  |
| 18 |  |  |
| 19 |  |  |
| 20 |  |  |
| 21 |  |  |

## Johnny\_3 Zyklen in der Programmabarbeitung

|  |  |  |  |
| --- | --- | --- | --- |
| Video | Vorkenntnisse | Hilfsmittel |  |
| johnny\_3.mp4 | Johnny\_1  Johnny\_2 | division.ram test.ram |  |

1. Informiere dich über Programmierung von Zyklen im von-Neumann-Simulator Johnny im Video johnny\_3.mp4.
2. Protokolliere die Arbeitsweise des Programms zur Bestimmung des Produkts der Werte in den Speicherzellen 010 (=7) und 011 (=4).  
     
   **000:  NULL 012   
   001:  TAKE 012   
   002:  ADD 010   
   003:  SAVE 012   
   004:  DEC 011   
   005:  TST 011   
   006:  JMP 001   
   007:  HLT 000**

Programmcounter:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 000 --> |  |  |  |  |
| --> |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Speicherbelegung:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 010 |  |  |  |  |  |  |
| 011 |  |  |  |  |  |  |
| 012 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. Implementiere das Programm zur Multiplikation der Werte in den Speicherzellen 010 und 011.

## 000:  NULL 012 001:  TAKE 012 002:  ADD 010 003:  SAVE 012 004:  DEC 011 005:  TST 011 006:  JMP 001 007:  HLT 000

Teste das Programm mit verschiedenen Werten.   
Beachte beim Test auch die Multiplikation mit 0.

MERKE:  
**zzz:   DEC xxx   
zzz+1:  TST xxx   
zzz+2:  JMP yyy**Eine Befehlsfolge aus **DEC xxx, TST XXX** und **JMP yyy** erzeugt einen Zyklus in einem Johnny-Programm zwischen den Speicherzellen **yyy** und **zzz**.

1. Lade das Programm **div.ram** in den Speicher. Mache dich mit dem Programm vertraut. Welche Speicherzellen werden wofür angezielt?  
   Protokolliere den Programmablauf in der Tabelle auf der nächsten Seite.

## 000: NULL 020 001: TAKE 018 002: SAVE 021 003: SUB 019 004: SAVE 018 005: INC 020 006: TST 018 007: JMP 001

**008: TAKE 019  
 009: SUB 021  
 010: SAVE 019  
 011: TST 019  
 012: JMP 014  
 013: JMP 015  
 014: DEC 020  
 015: HLT 000  
 016:  
 017:   
 018: 006  
 019: 002  
 020:**

## Programmcounter:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 000 --> |  |  |  |  |
| --> |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Speicherbelegung:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. Lade das Programm **test.ram** in den Speicher.   
   In den Zellen 000 bis 017 steht das Programm selbst.   
   Ab der Zelle 021 können fortlaufend Daten größer 0 eingetragen werden. Eine 0 schließt die Datenfolge ab.  
   Das Ergebnis findest du nach der Abarbeitung des Programms im Akkumulator und in der Zelle 020.

Programm für Aufgabe 5

|  |  |
| --- | --- |
| **000:  TAKE 021 001:  SAVE 020  002:  TAKE 021  003:  SUB 020  004:  SAVE 019  005:  TST 019  006:  JMP 008  007:  JMP 010  008: TAKE 021 009: SAVE 020 010: INC 002 011: INC 008** | **012: INC 013**  **013: TST 021 014: JMP 002 015: HLT 000 016: TAKE 020 017: HLT 000 018:  019: 020: 021: 009 022: 007 023: 025** |

Protokolliere den Programmlauf für die gegebenen Werte. Achte **besonders** darauf, dass sich der Quelltext des Programms während der Laufzeit ändert!

1. Teste das Programm für verschiedene Datenreihen. Achte dabei darauf, dass du das Programm für jede Datenreihe neu laden und die Daten ab Zelle 21 anpassen musst.  
   Beschreibe die Wirkung des Programms.

Wirkung: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Johnny\_4 Erweiterungen**

|  |  |  |  |
| --- | --- | --- | --- |
| Video |  | Hilfsmittel |  |
|  |  |  |  |

1. Erweitere das Programm zur Division, so dass auch der ganzzahlige Rest bei der Division gespeichert wird.
2. Implementiere ein Programm zur Berechnung der Potenz a^2 (a^3, a^n).
3. Dokumentiere den Programmablauf für ein Programm aus 2.
4. Implementiere ein Programm zur Berechnung der Fakultät.
5. Dokumentiere den Programmablauf für das Programm aus 4.
6. Schreibe ein Programm, das sich selbst im Speicher vermehrt, also sich selbst an eine andere Stelle des Speichers, etwa ab Speicherstelle 50 kopiert.
7. Jetzt wird's echt schwierig:   
   Ändere das Programm so ab, das es nach dem "Vermehrungsvorgang" in die Kopie springt, und diese sich wiederum vermehrt.  
   Quelle Aufgabe 6 und 7:  
   https://www.inf-schule.de/rechner/johnny/zusatzmaterial/exkurs\_selbstmodifikation
8. Vergleiche den Johnny-Befehlssatz mit realen Prozessoren.
9. Vergleiche CISC und RISC-Prozessorarchitekturen.
10. Erläutere eine Idee, die Subtraktion auf den Bereich der ganzen Zahlen zu erweitern (Bsp. 8 - 12 = -4).