Übung: Apfelmännchen, Mandelbrot-Menge

Allgemein

Die Mandelbrot-Menge ist die Menge alle Komplexenzahlen $\underline{c}$ für welche die Folge

$$\underline{z}_{n+1} = \underline{z}_n^2 + \underline{c} \quad \text{mit } \underline{z}_0 = 0$$

konvergiert.
Für den Fall, dass der Betrag von $\underline{z}_n$ größer als 2 ($\underline{z}_n^2 > 4$) ist gilt die Folge als divergent, wird dieses nach einer Anzahl N_{max} Iterationen nicht erreicht, so gilt die Folge als konvergent.
Aus der Anzahl n der Iterationen, bei der die Entscheidung über das Verhalten der Folge getroffen wird, kann ein Farbwert für den entsprechenden Punkt $\underline{c}$ in der Komplexenebene berechnet werden.
Die farblich markieren Punkte ergeben eine Figur, ein Apfelmännchen.

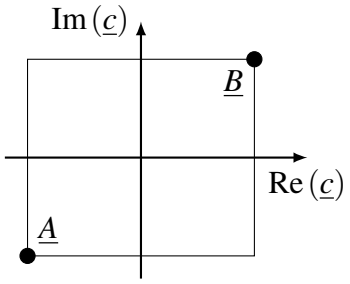
Aufgabenstellung

Programmieren Sie in Assembler-Sprache ein Programm, zur Berechnung von Apfelmännchen.

Vorgegeben

Ebene

Der Bereich für die Zahl $\underline{c}$ wird durch die Angabe der Linken-Unten-Ecke ($\underline{A}$) und der Rechten-Oberen-Ecke ($\underline{B}$) eingegrenzt.
(positiv reelle Achse nach rechts, positiv imaginäre Achse nach oben)
Weiterhin ist die Anzahl (P_x) der zu berechnenden Punkte ($\underline{c}$) entlang der reellen Achse gegeben.
Der Abstand zwischen den Punkten entlang der reellen und der imaginären Achse ist gleichzusetzen.



Definitionen

Die Werte vom $\underline{A}$, $\underline{B}$, P_x und N_{max} sind am Programm-Anfang als Konstanten (`EQU`) zu definieren.

Farbwerte

Die "Farbwerte" der einzelnen Punkte ($\underline{c}$) sind als ASCII-Zeichen über die serielle Schnittstelle ausgegeben[1].
Die Zuordnung zwischen der Anzahl (n) der Iteration bis zur Entscheidung und den ASCII-Zeichen ist wie folgt:

$n = N_{max}$	: ' ' ;	32_d
$n \mod 8 = 0$:	'¤' ;	164_d
$n \mod 8 = 1$:	'+' ;	43_d
$n \mod 8 = 2$:	'©' ;	169_d
$n \mod 8 = 3$:	'−' ;	45_d
$n \mod 8 = 4$:	'*' ;	42_d
$n \mod 8 = 5$:	'@' ;	64_d
$n \mod 8 = 6$:	'·' ;	183_d
$n \mod 8 = 7$:	'®' ;	174_d

Zahlenformat

Die Real- und Imaginärteile der Komplexenzahlen sind als 16 Bit Festkomma Zweier-Komplement Zahlen darzustellen.
Das Format der Zahlen hat 6 Vorkomma- und 10 Nachkomma-Stellen : `VVVVVV.NNNNNNNNNN`
z.B.
$1{,}5_d$: `0000 01.10 0000 0000` : (0600_h) ; $-1{,}5_d$: `1111 10.10 0000 0000` : ($fa00_h$)

Startwerte

a)	$\underline{A} = -2{,}25 - \imath\,1{,}5$;	$\underline{B} = 0{,}75 + \imath\,1{,}5$	mit jeweils	$P_x = 20_d$ und $N_{max} = 20_d$
				$P_x = 111_d$ und $N_{max} = 20_d$
				$P_x = 111_d$ und $N_{max} = 60_d$
b)	$\underline{A} = -0{,}8 - \imath\,0{,}4$;	$\underline{B} = 0{,}6 + \imath\,1{,}0$	mit	$P_x = 111_d$ und $N_{max} = 60_d$

Frage

a) Gibt es ein P_{max} ? Wenn ja: warum? Wenn nein: warum nicht?

[1] Konfiguration der Seriellen Schnittstelle 0 mit den unten angegebenen Parametern.

Randbedingungen

Prozessor:

Infineon C517a (ohne ROM), Prozessorfrequenz: 24 MHz
(alle On-Chip Peripheral Components des Prozessors dürfen verwendet werden, z.B. MDU)

Schnittstelle:

Serial Interface 0, 1 Startbit, 8 Datenbit, 1 Stoppbit, keine Parität, kein Handshaking,
Baudrate 28800 1/s

Programmierung:

Daten-Ausgabe / Dokumentation

Die Ausgaben erfolgen über die Serielle-Schnittstelle (ASCII- Zeichen), welche vom Simulator in ein Fenster und in eine Datei geschrieben werden können.
Die Speicherung der Werte in eine Datei erfolgt Mithilfe des Debuggers durch die Eingabe folgendes Befehls im COMMAND-Fenster: `SLOG > xx.txt`

Abzugeben

Auf Papier (?), nur gelocht und geheftet: | Deckblatt →

- Deckblatt mit Thema, Namen und Kurs
- Kurze Einleitung (Problembeschreibung)
- Antwort auf die Frage von oben
- Kontroll-Ausgaben:
  Die berechneten Apfelmännchen (den Schriftgrad soweit verringern, dass jedes Bild auf ein Blatt A4 passt.)
- Beschreibung der wichtigsten Programmteile, des Programmablaufs (Flußdiagramme oder Struktogramme)
- Kommentierter (!) Quell-Code

Und als e-mail an ralf.baehnisch@dlr.de : Quell-Code

Programmendwurf

Systemnaheprogrammierung

Mandelbrotmenge (Apfelmännchen)

Matrikelnummer 1

TINF 19 B
3. Semester
2018

Hinweise

- Die pseudoassembler Befehle `low()` und `high()` bestimmen das low- bzw. high- Byte des Arguments.
- Die Ausgabe kann etwas variieren.
- Beispiel Ausgabe für $\underline{A} = -2{,}25 - \imath 1{,}5$; $\underline{B} = 0{,}75 + \imath 1{,}5$ mit $P_x = 20_d$ und $N_{max} = 20_d$

```
+++++++©©©©©©©©©©©©©©
++++++©©©©©©©©©©©©©©
+++++©©©−−−−−−−−−©©©
++++©©−−−−−−−⋆@−@⋆−©©
++++©−−−−−⋆⋆@·®¤⋆−−
+++©−−−−−⋆⋆@·¤  ·@⋆−
+++−−−−−⋆@·@      ·−
+++−−−⋆·@·®       ·⋆
++−⋆⋆@·®@⋆©       +⋆
++⋆⋆@@+           ®⋆
++¤©⋆             ···⋆
++⋆⋆@·+           ®⋆
++−⋆⋆@·®@@−       ¤⋆
+++−−−⋆·@·®       ®⋆
+++−−−−−⋆@·©     −·−
+++©−−−−−⋆⋆@·+  ·@⋆−
++++©−−−−−⋆⋆@·©¤⋆−−
++++©©−−−−−⋆⋆@ ·⋆−©©
+++++©©©−−−−−−−−−©©©
++++++©©©©©©©©©©©©©©
```