Thema, Ziele: Computerarithmetik, Monte Carlo Integration

Aufgabe 1: Inverse Square Root beim TMS320C28x

a) Der Befehl `EINVF32` gibt in 2 Clockzyklen einen ungefähr auf 8 Bit genauen Näherungswert von $\frac{1}{x}$. Mit diesem Näherungswert müssen noch zwei Newton-Raphson-Schritte durchgeführt werden, damit anschliessend das Resultat als single precision floating point (32 Bit) vorliegt. Der Pseudocode der nächsten Schritte lautet:

```
Ye = Estimate(1/X); // Resultat von EINVF32
Ye = Ye*(2.0 - Ye*X)
Ye = Ye*(2.0 - Ye*X)
```

b) Die Instruktion `EISQRTF32` erzeugt ebenfalls in 2 Clockzyklen eine Genauigkeit von ungefähr 8 Bits für die Funktion $\frac{1}{\sqrt{x}}$. Auch hier braucht es noch zwei Newton-Raphson-Schritte, um eine Genauigkeit von 32 Bits zu erreichen. Damit als Resultat $\sqrt{x}$ erscheint, muss diese Annäherung noch mit x multipliziert werden, da $\frac{1}{\sqrt{x}} \cdot x = \sqrt{x}$. Der Pseudocode für Y = sqrt(X) lautet:

```
Ye = Estimate(1/sqrt(X)); // Resultat von EISQRTF32
Ye = Ye*(1.5 - Ye*Ye*X*0.5)
Ye = Ye*(1.5 - Ye*Ye*X*0.5)
Y = X*Ye
```

c) Die verwendeten Instruktionen benötigen die folgenden Ausführungszeiten:

Instruktion	Clockzyklen
EISQRTF32	2
MPYF32	2
SUBF32	2
CMPF32	1
MOV32	1
LRETR	1

Das Programm wird die Pipeline wie folgt belegen:

EISQRTF32	MPYF32	MPYF32	NOP	MPYF32	NOP	SUBF32	NOP	MPYF32	NOP	MPYF32	NOP	MPYF32	NOP	SUBF32	CMPF32	MPYF32	NOP	MOV32	MPYF32	LRETR
	EISQRTF32	MPYF32	MPYF32		MPYF32		SUBF32		MPYF32		MPYF32		MPYF32		SUBF32		MPYF32			MPYF32
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

Insgesamt benötigt der obenstehende Assemblercode 21 Clockzyklen.

Aufgabe 2: Monte Carlo Integration

a) Befehl z.B. in Gnuplot:

```
plot [1:2] sqrt((x-1)*(2-x))*exp(-(x*x))
```

b) siehe Eclipseprojekt `./MonteCarlo`

c) Als Funktion kann ein Viertelkreis mit Radius 1 genommen werden, d.h. die Funktion $f(x) = \sqrt{1 - x^2}$. Diese muss dann von 0 bis 1 integriert werden. Die erhaltene Fläche entspricht dem Wert $\frac{\pi}{4}$.