Arne Bahlo Johannes Wahl

Praktikumsprotokoll Termin 2

Aufgabe 1

Datentyp	Bit	kleinste Zahl Hex / Dez	größte Zahl Hex / Dez	Anmerkungen
char	8	0x80	0x7F / 127	
short int	16	0x8000 / -32768	0x7FFF / 32767	
int	(16) 32 (64)	0x80000000/ -2.147.483.648	0x7FFFFFFF/ 2.147.483.647	Hängt vom verwendeten Compiler und Betriebssystem ab
unsigned int	16	0x0 / 0	FFFF / 65.535	
long int	32 (64)	0x8000000000000000/ -–9.223.372.036.854.775.808	0x7FFFFFFFFFFFFFFF / –9.223.372.036.854.775.807	Hängt vom verwendeten Compiler und Betriebssystem ab

Gleitpunktzahldatentypen

Datentyp	Bit	kleinste Zahl Hex / Dez	größte Zahl Hex/ Dez
float	32	ca. 3,4-E38	ca. 3,4E38
double	64	ca. 1,7E-308	ca. 1,7E308
long double	80	ca. 1,2E-4932	ca. 1,2E4932

Aufgabe 2

Dez	int	float
1	00 00 00 01	3F800000
-1	FF FF FF FF	BF800000
65536	0x0000FFFF	477FFF00

Dez	int	float
1,024 * 10^3	0x00000400	0x44FFFFFF

Aufgabe 3
Siehe Anhang

Aufgabe 4
Es werden nun nicht die Werte direkt in die Scratch-Register geschrieben, sondern es werden globale words angelegt.

Aufgabe 5
Erst werden die Werte der lokalen Variablen in ein Scratch-Register geschrieben und dann von diesen auf den Framepointer.

Aufgabe 6
Die globale Variable wird nicht auf den Framepointer geschrieben und es wird nicht die Adresse vom Framepointer an die Funktion übergeben, sondern die der globalen Variable.

Aufgabe 7
Die Adressen der Variablen werden beim Anlegen auf den Framepointer geschrieben und vor dem Funktionsaufruf werden die Werte an den Adressen aus dem Framepointer in die Scratchregister geschrieben.

```
        .file   "practice1.c"
        .text
        .align  2
        .global main
        .type   main, %function
main:
        @ args = 0, pretend = 0, frame = 8
        @ frame_needed = 1, uses_anonymous_args = 0
        @ link register save eliminated.
        @ Prolog
        str  fp, [sp, #-4]! @ fp = sp = sp - 4
        add  fp, sp, #0 @ fp = sp
        sub  sp, sp, #8 @ sp = sp -8, Platz schaffen auf dem sp

        mov  r3, #1 @ r3 = 1
        str  r3, [fp, #-8] @ fp-8 = [r3], r3 auf den framepointer schreiben
        mov  r3, #2 @ r3 = 2

        str  r3, [fp, #-4] @ fp-4 = [r3], r3 auf den fp schreiben

        mov  r3, #0 @ r3 = 0
        mov  r0, r3 @ r0 = r3

        @ Epilog
        add  sp, fp, #0 @ sp = fp + 0
        ldmfd   sp!, {fp} @ load sp to fp and increment
        bx   lr
        .size   main, .-main
        .ident  "GCC: (crosstool-NG 1.14.1) 4.4.6"
```