Technische Universität Clausthal
Institut für Informatik
Prof. G. Kemnitz

01/20/2014

Laborübung 3 Rechnerarchitektur

Hinweise: Schreiben Sie die Lösungen, so weit es möglich ist, auf die Aufgabenblätter. Tragen Sie Namen, Matrikelnummer und Studiengang in die nachfolgende Tabelle ein und schreiben Sie auf jedes zusätzlich abgegebene Blatt ihre Matrikelnummer. Lassen Sie vorgeführte Experimente vom Betreuer gegenzeichnen. Für eine Bescheinigung der erfolgreichen Teilnahme sind in jeder einzelnen Laborübung mindestens 40% und insgesamt mindestens 50% der Punkte erforderlich.

Name	Matrikelnummer	Studiengang	Punkte von 20	$\geq 40\%$

Aufgabe 1:

Das nachfolgende Assemblerprogramm initialisiert Port C als Ausgang und kopiert in einer Endlosschleife vier Bits von Port A nach Port C mit gespiegelter Zuordnung:

		T	r25	r24
`ser r24`	Port C	0	0x1A	
`out 7, r24`	als Ausg.			
`main:`				
`in r24, 0`	r24 ← PINA			0b0110 1101
`clr r25`	r25 ← 0			
`bst r24, 0`	r25.7 ← r24.0			
`bld r25, 7`				
`bst r24, 1`	r25.7 ← r24.0			
`bld r25, 6`				
`bst r24, 2`	r25.7 ← r24.0			
`bld r25, 5`				
`bst r24, 3`	r25.7 ← r24.0			
`bld r25, 4`				
`out 8, r25`	PORTC ← r25			
`jmp main`				

a) Tragen Sie in der Tabelle neben dem Programm die fehlenden Registerwerte nach Abarbeitung des Befehls links daneben ein. 2P

b) Schreiben Sie ein C-Programm, das dieselbe Funktion nachbildet[1]. 2P

[1] Funktionsgleich bedeutet, dass für alle Wertepaare von b und c derselben Werte in a geschrieben wird. Unterschiede in der Befehlsfolge, den Adress- und Registerzuordnungen sind zulässig.

Aufgabe 2:

Gegeben sind die Bezeichner, Adressen und Anfangswerte der globalen Variablen und das disassemblierte Programm:

Name	Value	Type
a		uint16_t{data}@0x0204
b	0x3842	uint16_t{data}@0x0200
c	0x5fa3	uint16_t{data}@0x0202

	r25	r24	r19	r18
0085 LDS R18,0x0200				
0087 LDS R19,0x0201				
0089 LDS R24,0x0202				
008B LDS R25,0x0203				
008D ADD R24,R18				
008E ADC R25,R19				
008F SUBI R24,0x0D				
0090 SBCI R25,0xFF				
0091 STS 0x0205,R25				
0093 STS 0x0204,R24				

a) Tragen Sie in der Tabelle neben dem disassemblierten Programm die Registerwerte nach Abarbeitung des Befehls links daneben ein. 3P

b) Tragen Sie in der Tabelle mit den Bezeichnern und Adressen den Wert, der nach Abarbeitung aller Befehle in die Variable a gespeichert wird, ein. 1P

c) Schreiben Sie ein C-Programm, das dieselbe Funktion nachbildet. 2P

Aufgabe 3:

Für das nachfolgende Multiplikationsprogramm sind zusätzlich die Adressen zu den Variablen und das disassemblierte Programm gegeben:

```
uint32_t a;
uint16_t b=0xd13F;
uint16_t c=0x76ab;
void main(){
    a = b*c;
}
```

Name	Value	Type
a		uint32_t{data}@0x0204
b		uint16_t{data}@0x0202
c		uint16_t{data}@0x0200

	r25	r24	r21	r20	r19	r18	r1	r0
00092 LDS R20,0x0200								
00094 LDS R21,0x0201								
00096 LDS R18,0x0202								
00098 LDS R19,0x0203								
0009A MUL R20,R18								
0009B MOVW R24,R0								
0009C MUL R20,R19								
0009D ADD R25,R0								
0009E MUL R21,R18								
0009F ADD R25,R0								
000A0 CLR R1								
000A1 LDI R26,0x00								
000A2 LDI R27,0x00								
000A3 STS 0x0204,R24								
000A5 STS 0x0205,R25							r27	r26
000A7 STS 0x0206,R26								
000A9 STS 0x0207,R27								
000AB RET								

a) Tragen Sie in der Tabelle neben dem disassemblierten Programm die Registerwerte nach Abarbeitung des Befehls rechts daneben ein. 4P

b) Tragen Sie in der Tabelle mit den Bezeichnern und Adressen deren Werte nach Abarbeitung des letzten Befehls ein. 2P

c) Schreiben Sie das Programm in Assembler neu so, dass auch die höherwertigen beiden Ergebnisbytes richtig berechnet werden. 4P