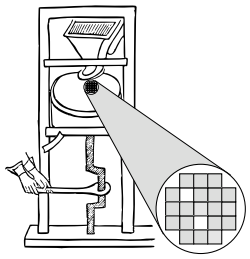
Technische Universität Darmstadt
FG Eingebettete Systeme und
ihre Anwendungen (ESA)

Prof. Dr. Andreas Koch
Holger Lange
Mathias Halbach (FG Rechnerarchitektur)



08.06.2006

Technische Grundlagen der Informatik II
5. Übung – Dinatos
Sommersemester 2006

Aufgabe 1: Dinatos-Befehle

Betrachtet wird das Dinatos-Zustandsdiagramm (Vorlesung Kapitel 5, Folie 14).

Geben Sie für die Befehle LDA#, LDA*, STA, STA*, GO und MUL an, welche Zustände bei der Befehlsinterpretation durchlaufen werden. Ordnen Sie dabei zusätzlich den Zuständen die jeweils auszuführenden Operationen zu.

Lösung: Notation: (Zustand) Operationen

LDA#	(1) BR <= m(P), P <= P + 1	(2) AC <= N		
LDA*		(2) AR <= N	(4) AR <= m(AR)	(3) AC <= m(AR)
STA			(5) m(AR) := AC	
STA*			(6) AR <= m(AR)	(5) m(AR) := AC
GO			(14) P <= m(AR)	
MUL			(7) BC <= m(AR)	(10) AC <= AC * BC

<= synchrone Zuweisung an Register, := asynchrone Zuweisung an Speicher

Aufgabe 2: Binäres Maschinenprogramm

Füllen Sie die leeren Felder aus (wie in Kap. 5, Folie 17 der Vorlesung). Korrigieren Sie dabei einen wahrscheinlichen Programmierfehler im gegebenen Dinatos-Programm. Hinweis: Eine Übersicht der Dinatos-Befehle finden Sie auf Folie 5-13.

Adresse	Inhalt hex	OPC	Adr./ Konst. (dez)	Kommentar	Operation	Ergebnis (hex)
						initial AC=0
0	03 00000F	lda#	15	Lade Konstante	AC <= 15	AC = 0000000F
1	02 000002	shr	2	Schiebe nach rechts	AC <= (15 >> 2)	AC = 00000003
2	06 000009	sta	9	Speichern	m[9] <= 3	m[9] = 3
3	03 000000	lda#	0	Lade Konstante	AC <= 0	AC = 0
4	05 000007	lda*	7	Lade indirekt	AC <= m[m[7]]	AC = 3
5	0E 000008	go	8	Gehe nach (indirekt)	P <= m[8]	P = 5
6	10 000000	stop		anhalten		
7	00000009			Adresse		
8	00000005			Fehler, ersetzen durch Adresse 6 (00000006) verhindert Endlosschleife		
9				Datum		

Aufgabe 3: Rechnen mit Dinatos

Zwei 2K-Zahlen $-2^{30} \leq X, Y \leq 2^{30}-1$ sind zu addieren. Dazu soll ein Dinatos-Assembler-Programm geschrieben werden, welches außer der Summe S der beiden Zahlen X und Y auch die Überlaufbedingung OV (overflow) erzeugt.

a) In welcher Beziehung stehen die beiden höchstwertigen Stellen (MSB) mit den Indizes 32 und 31 (bei Zählweise X_{32} bis X_1 bzw. Y_{32} bis Y_1) derartig normierter 2K-Zahlen?

Lösung:

Der Zahlenbereich wird nur zur Hälfte ausgeschöpft. Die 2K-Zahlen besitzen zwei identische Vorzeichenbits (Stellen 32 und 31). Man spricht auch von einer so genannten Schutzstelle. Wenn sich die beiden Stellen unterscheiden, liegt ein ungültiger Zahlenwert vor.

b) Wie kann ein auftretender Overflow bei der Addition solcher Zahlen erkannt werden? Fertigen Sie sich ein Tabelle für $n = 3$ mit 16 Einträgen an, in der Sie für alle Kombinationen von X, Y und S die Überlaufbedingung OV bestimmen. Vergleichen Sie anschließend das Ergebnis mit den drei Möglichkeiten von Folie 3-25, um OV zu bestimmen.

Lösung:

X	Y	S	OV
$000(\pm 0)$	$000(\pm 0)$	$000(\pm 0)$	nein
$000(\pm 0)$	$001(+1)$	$001(+1)$	nein
$000(\pm 0)$	$110(-2)$	$110(-2)$	nein
$000(\pm 0)$	$111(-1)$	$111(-1)$	nein
$001(+1)$	$000(\pm 0)$	$001(+1)$	nein
$001(+1)$	$001(+1)$	$010(+2)$	ja
$001(+1)$	$110(-2)$	$111(-1)$	nein
$001(+1)$	$111(-1)$	$000(\pm 0)$	nein
$110(-2)$	$000(\pm 0)$	$110(-2)$	nein
$110(-2)$	$001(+1)$	$111(-1)$	nein
$110(-2)$	$110(-2)$	$100(-4)$	ja
$110(-2)$	$111(-1)$	$101(-3)$	ja
$111(-1)$	$000(\pm 0)$	$111(-1)$	nein
$111(-1)$	$001(+1)$	$000(\pm 0)$	nein
$111(-1)$	$110(-2)$	$101(-3)$	ja
$111(-1)$	$111(-1)$	$110(-2)$	nein

Als logische Formel ergibt sich $OV = S_3$ xor S_2. Dies entspricht der dritten Variante der Vorlesungsfolie, wenn C_{n+1} durch S_n und C_n durch S_{n-1} bei $OV = (C_{n+1} \oplus C_n)$ ersetzt wird: $OV = (S_n$ xor $S_{n-1})$.

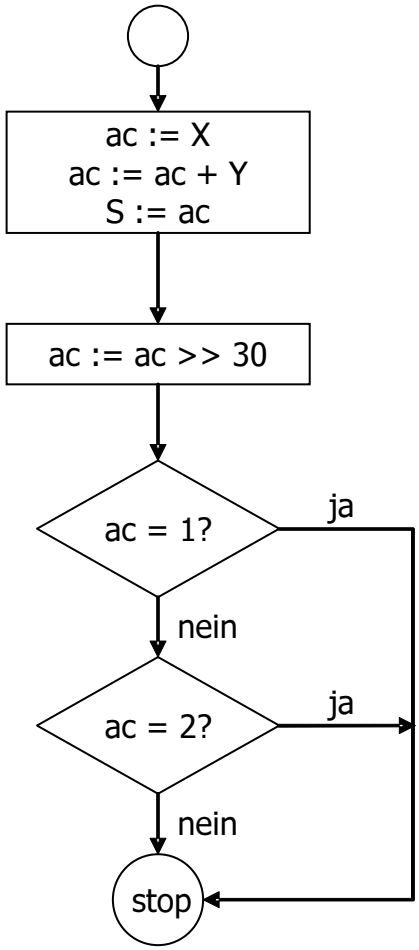
c) Beschreiben Sie Ihre Lösungsidee und fertigen Sie einen Ablaufplan (Flussdiagramm) an, aus dem der Ablauf Ihrer Idee hervorgeht.

Lösung:

Idee:

1. $X + Y$ bilden und in S speichern
2. $AC = S >> 30$
3. $AC = 1$ oder 2? (C setzen)

Flussdiagramm:



d) Schreiben Sie nun das Assembler-Programm. Die Operanden stehen in den Speicherzellen 18 und 19. Die Summe *S* soll in der Speicherzelle 20 abgelegt werden. Die Überlaufbedingung soll in dem C-Bit gespeichert werden, danach soll das Programm stoppen.

Lösung:

Dinatos-Programm:

```
0  LDA     18
1  ADD     19
2  STA     20
3  SHR     30
4  IF=#    1
5  BRNC#   1
6  GO#     8
7  IF=#    2
8  STOP
```