Karlsruher Institut für Technologie

Institut für Technische Informatik
Lehrstuhl für Rechnerarchitektur und Parallelverarbeitung
Prof. Dr. rer. nat. Wolfgang Karl

Zentralübung Rechnerstrukturen: Fehlertoleranz und Sprungvorhersage

3. Aufgabenblatt

Besprechung: 21. Mai 2015

1 Fehlertoleranz

1. Blockdiagramm und Systemfunktion

   Gegeben sei ein portables Rechnersystem bestehend aus einer Batterie B, einer Hauptplatine H, der eigentlichen Recheneinheit R, dem Speicher S und einer redundant ausgelegten Kommunikation über die Komponenten K_1 bis K_3. Zum fehlerfreien Betrieb des Systems sind die Batterie, die Hauptplatine, die Recheneneinheit, der Speicher und mindestens eine Kommunikationskomponente erforderlich.

   Erstellen Sie sowohl das Zuverlässigkeitsblockdiagramm als auch die Systemfunktion (Strukturformel), den Fehlerbaum und berechnen Sie die Funktionswahrscheinlichkeit.

2. Maßzahlen und Berechnung

   a) Ein RAID6-System besteht aus 10 Festplattenspeichern. Hiervon dürfen zwei ausfallen, ohne dass es zu einem Datenverlust kommt. Unter der Annahme, die Funktionswahrscheinlichkeit pro Festplatte betrage $\varphi(F) = 0,99$, wie hoch ist die Chance auf Datenverlust?

   b) Eine Festplatte habe eine MTTF von 20 Jahren im Dauerbetrieb. Die Reparaturzeit (MTTR) setze sich zusammen aus der Zeit für das Herunterfahren des Rechners (2 Minuten), Austausch der Festplatte (10 Minuten) und anschließendes Hochfahren des Rechners (2 Minuten).

      Berechnen Sie die Punktverfügbarkeit V.

   c) Welche Annahme steckt hinter der Punktverfügbarkeitsberechnung bezüglich der Ausfallrate λ? Wie ist dies in Bezug auf die sogenannte Badewannenkurve zu interpretieren?

d) Gegeben sei ein 2-von-3-System, dessen Komponenten zufallsverteilt mit gleicher Rate ausfallen. Die Überlebenswahrscheinlichkeit einer Komponente wird durch die Formel $R(t) = e^{-\lambda \cdot t}$, $t > 0$ beschrieben.

i. Wie groß ist die Ausfallrate für eine einzelne Komponente?

ii. Bestimmen Sie die Zeitintervalle, in denen das 2-von-3-System eine größere Überlebenswahrscheinlichkeit als eine einzelne Komponente aufweist.

iii. Bestimmen Sie λ derart, dass die mittlere Lebensdauer für das gegebene 2-von-3-System $\frac{5}{6}$ beträgt.

2 Sprungvorhersage I

a) Um das Leerlaufen der Pipeline bei Kontrollflussbefehlen zu vermeiden, existieren statische, sowie dynamische Techniken, die jeweils zu verschiedenen Teilen durch Hardware und Software unterstützt werden. Nennen Sie diese und stellen Sie die wesentlichen Unterschiede gegenüber.

b) Zeichnen Sie einen 2-Bit-Prädiktor einmal mit Sättigungszähler und einmal mit Hysteresezähler. Worin liegt die Motivation zur Verwendung eines Hysterese- anstelle eines Sättigungszählers?

c) Füllen Sie die Tabelle für die Vorhersagen und Zustände der obigen zwei Prädiktoren und das unten angegebene Programm aus, wobei alle Sprünge auf denselben Prädiktor zugreifen und die Prädiktoren mit Predict Weakly Not Taken initialisiert seien.

```
1  INIT:   ADD  R1,R0,#0  ; R1=0
2          ADD  R2,R0,#2  ; R2=2
3
4  START:  BNE  R1,R0,L1  ; if (R1!=0) goto L1
5          ADD  R1,R0,#1  ; R1=1
6
7  L1:     SUB  R3,R1,R2  ; R3=R1-R2
8          BNE  R3,R0,L2  ; if (R1!=R2) goto L2
9          ADD  R1,R0,#0  ; R1=0
10         J    START     ; goto START
11
12 L2:     ADD  R1,R0,#2  ; R1=2
13         J    START     ; goto START
```

Listing 1: Programmstück in MIPS-Assembler

Sättigungszähler:

Sprung 1			Sprung 2		
Prädiktion	Sprung	Neue Vorh.	Prädiktion	Sprung	Neue Vorh.
WNT	NT			T	
	T			NT	
	NT			T	
	T			NT	

Hysteresezähler:

Sprung 1			Sprung 2		
Prädiktion	Sprung	Neue Vorh.	Prädiktion	Sprung	Neue Vorh.
WNT	NT			T	
	T			NT	
	NT			T	
	T			NT	

d) Nehmen Sie nun an, dass jeder Sprung über einen eigenen Prädiktor verfüge (Lokale Prädiktion). Welchen Unterschied stellen Sie fest, worauf lässt sich dieser zurückführen?

Sättigungszähler:

Sprung 1			Sprung 2		
Prädiktion	Sprung	Neue Vorh.	Prädiktion	Sprung	Neue Vorh.
WNT	NT		WNT	T	
	T			NT	
	NT			T	
	T			NT	

Hysteresezähler:

Sprung 1			Sprung 2		
Prädiktion	Sprung	Neue Vorh.	Prädiktion	Sprung	Neue Vorh.
WNT	NT		T		
	T		NT		
	NT		T		
	T		NT		

e) Gegeben Sei nun ein (1,2)-Korrelationsprädiktor, der global verwendet werde. Das Schieberegister BHR sei mit NT initialisiert, und die beiden 2-Bit-Hysteresepräditoren mit Predict Weakly Taken. Füllen Sie untenstehende Tabelle für die oben ermittelten Sprungausgänge aus.

Sprungzeile	Richtung	Aktuelle Vorhersage			Neue Vorhersage	
		Historie	Prädiktor	Vorh.	Akt. Historie	Akt. Prädiktoren
4			(,)			(,)
8			(,)			(,)
4			(,)			(,)
8			(,)			(,)
4			(,)			(,)
8			(,)			(,)
4			(,)			(,)
8			(,)			(,)

3 Sprungvorhersage II

Gegeben sei der folgende MIPS-Code. Beachten Sie, dass Register R0 in der MIPS-ISA immer den Wert 0 hat. Beachten Sie weiterhin, dass MIPS-Instruktionen immer an Wortgrenzen ausgerichtet sind, d.h. die niedrigsten zwei Bits der Instruktionsadresse sind immer 0. In diesem Beispiel werden trotzdem die niederwertigsten Bits der Sprungadresse zur Indizierung der Sprungvorhersagetabellen verwendet.

```
0x100     li   R2, 0             ; v = 0
0x104     li   R3, 100           ; Loop bound for LoopI
0x108     li   R4, 0             ; i = 0
LoopI:
0x10C     beq  R4, R3, EndLoopI  ; Exit LoopI if i == 100
0x110     li   R5, 0             ; j = 0
LoopJ:
0x114     beq  R5, R3, EndLoopJ  ; Exit LoopJ if J == 100
0x118     add  R6, R5, R4        ; j + i
0x11C     andi R6, R6, 1         ; (j+i)%2
0x120     bne  R6, R0, EndIf     ; Skip if (j+i)%2 != 0
0x124     add  R2, R2, R5        ; v +=j
EndIf:
0x128     addi R5, R5, 1         ; j++
0x12C     beq  R0, R0, LoopJ     ; Go back to LoopJ
EndLoopJ:
0x130     addi R4, R4, 1         ; i++
0x134     beq  R0, R0, LoopI     ; Go back to LoopI
```

Listing 2: Programmstück in MIPS-Assembler

Bestimmen Sie nun für diesen Assembler-Code die exakte Anzahl an Fehlvorhersagen der Sprungvorhersage, die während der Ausführung auftreten, wenn folgende Prädiktoren verwendet werden:

1. Ein Always-Taken Prädiktor
2. Ein globaler 1-Bit Prädiktor, initialisiert mit Taken.
3. Ein 1-Bit Prädiktor mit 32 Einträgen, die niedrigsten Bits der Instruktionsadresse werden zur Indizierung des Eintrags verwendet, initialisiert mit Taken.
4. Ein 2-Bit Prädiktor mit 16 Einträgen, die niedrigsten Bits der Instruktionsadresse werden zur Indizierung des Eintrags verwendet, initialisiert mit Strongly Taken.

4 Sprungvorhersage III – SimpleScalar

Zur Evaluation von Mikroarchitekturen werden in der Forschung und in der Industrie unterschiedliche Simulatoren eingesetzt. Einer der bekanntesten ist SimpleScalar. Das SimpleScalar-Toolset ist eine Ansammlung von mehreren Mikroarchitektursimulatoren, angefangen von `sim- fast`, einem schnellen funktionalen Simulator, bis hin zu `sim-outorder`, einem Simulator mit einem komplexen, superskalaren Prozessormodel.

a) Installation
Laden Sie das SimpleScalar-Toolset von der SimpleScalar-Homepage[1] herunter und installieren Sie es. Alternativ können Sie von der Rechnerstrukturen-Homepage ein Virtualbox-Image eines Kubuntu-Systems herunterladen, in dem das SimpleScalar-Toolset inkl. Cross-Compiler bereits vollständig eingerichtet ist.

Weitere Hinweise:

- SimpleScalar kann für akademische Zwecke kostenfrei verwendet werden.
- Die Installation von zusätzlich benötigten Anwendungen, z.B. die des Cross-Compilers, erfordert eine gewisse Anpassungen. Ein Anleitung zur Installation finden Sie z.B. hier: `http://www.ann.ece.ufl.edu/courses/eel5764_10fal/project/1.36445-Simplescalar-installation-instructions.pdf`
- Unter Windows kann SimpleScalar mittels Cygwin verwendet werden. Eine Installation des Cross-Compilers ist in Cygwin aber nicht möglich.
`http://www.cygwin.com/`
- Informationen zu den einzelnen Simulatoren, sowie deren mögliche Parameter finden Sie im SimpleScalar User Guide
`http://www.simplescalar.com/docs/users_guide_v2.pdf`

b) Matrix-Matrix-Multiplikation
Der Simulator `sim-bpred` ist speziell für die Simulation von aktuellen Sprungvorhersageneinheiten entwickelt worden. Der Aufruf erfolgt durch:
`sim-bpred {-options} executable {arguments}`

Gegeben sei der Quellcode `mm-std.c` für eine Standard Matrix-Matrix-Multiplikation. Übersetzen Sie diesen Quellcode zunächst mit dem Cross-Compiler und simulieren Sie anschließend diese Anwendung mit dem `sim-bpred` Simulator. Verwenden Sie folgende Sprungvorhersageeinheiten und verwenden Sie die dazugehörenden Parameter beim Aufruf von `sim-bpred`:

- Always taken:
`-bpred taken`
- Always nottaken:
`-bpred nottaken`

[1] `www.simplescalar.com`

- Globaler 2 bit-Prädiktor:
  `-bpred:bimod 1`
- Prädiktortabelle mit 2 bit-Prädiktoren und 16 Einträgen:
  `-bpred:bimod 16`
- Globaler Korrelationsprediktor mit 1 bit-History und einem 2 bit-Prädiktor:
  `-bpred:2lev 1 1 1 0`

Welche Sprungvorhersageeinheit würden Sie für diese Anwendung auswählen? Begründen Sie Ihre Antwort.

Hinweis: Der Aufruf des Simulators erfolgt beispielsweise durch:
`sim-bpred -bpred:2lev 1 1 1 0 ./mm-std`

c) MiBench Benchmark Suite

Die MiBench Benchmark Suite[2] ist eine Sammlung von unterschiedlichen Anwendungen zur Evaluation von eingebetteten Systemen.

Vergleichen Sie nun die Benchmarks `basicmath`, `qsort` und `susan` aus dem Automotive-Teil der Benchmark-Suite hinsichtlich ihrer Leistung mit verschiedenen Sprungvorhersageeinheiten. Verwenden Sie hier dieselben Prädiktoren wie in Aufgabenteil b)

Welche Vorhersageeinheit würden Sie für jede Anwendung wählen? Welche Sprungvorhersageeinheit liefert insgesamt die beste Leistung?

Hinweis:

- Um die Benchmarks mit dem Cross-Compiler übersetzen zu können, müssen die zugehörigen Makefiles angepasst werden. Hierzu genügt es den `gcc` durch den Cross-Compiler `sslittlena-sstrix-gcc` zu ersetzen, sowie das Flag `-lm` zu entfernen.
- Verwenden Sie die vorhandenen, kleineren Eingabedatensätze.
- Im jeweiligen Verzeichnis des Benchmarks existiert die Datei `runme_small.sh`, welche Hinweise zum Aufruf des Benchmarks (z.B. `basicmath_small`) mit kleinen Eingabedatensätzen gibt.

[2] `http://www.eecs.umich.edu/mibench/`