

Chiviny Long-Marquardt

# **Clock Gating**

**STUDIENARBEIT**

an der

**UNIVERSITÄT TÜBINGEN**

---

**FAKULTÄT FÜR INFORMATIK**

Arbeitsbereich Technische Informatik

Tübingen, 30. April 2004

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
<b>2</b>	<b>Clock Gating</b>	<b>2</b>
2.1	Clock Gating Prinzip . . . . .	2
2.2	Energiekonsum . . . . .	5
2.3	Clock Distribution . . . . .	5
<b>3</b>	<b>Anwendungsbeispiele</b>	<b>9</b>
3.1	Deterministic Clock Gating . . . . .	9
3.2	Value-Based Clock Gating . . . . .	11
<b>4</b>	<b>Simulation</b>	<b>14</b>
4.1	Ziel der Simulation . . . . .	14
4.2	Überblick auf die Arbeitsweise der Implentierung . . . . .	15
4.3	Die Simulation im Detail . . . . .	16
4.3.1	Die Reservation Station im Detail . . . . .	16
4.3.2	Die Kommunikation zwischen allen Komponenten . . . . .	17
4.4	Konfiguration . . . . .	19
4.5	Functional Unit Belegung und Sparpotential ohne Berücksichtigung der Datenabhängigkeit . . . . .	21
4.6	Functional Unit Belegung und Sparpotential unter Berücksichtigung der Datenabhängigkeit . . . . .	22
4.7	Analyse der Simulationsergebnisse . . . . .	23
<b>5</b>	<b>Zusammenfassung und Ausblick</b>	<b>32</b>
<b>A</b>	<b>Instruction Set Listing</b>	<b>35</b>
<b>B</b>	<b>Testfall 3: Simulation mit gzip.source</b>	<b>38</b>

# 1 Einleitung

Eine Leistungssteigerung der Mikroprozessoren kann rein technologisch durch mehrere Maßnahmen erreicht werden. Eine davon besteht in der Erhöhung der Anzahl der Transistoren, die den Prozessor auf den Chip implementieren. Momentan ist es durchaus üblich, 64 Bit Mikroprozessoren herzustellen, wobei die Anzahl von Transistoren pro Chip zwischen 2 und 100 Millionen liegt.

Für die modernen CMOS Mikroprozessoren besteht leider der Energieverbrauch vorwiegend in Umschaltung der Transistoren, wobei der Energieverbrauch pro Transistor ist proportional zur Ladungskapazität  $c$ , Frequenz der Umschaltung  $f$  und zum Quadrat der anliegenden Spannung  $v$ .

$$P \sim c * v^2 * f$$

Somit steigt der Energiebedarf aktueller Mikroprozessoren unaufhaltsam mit der Anzahl von Transistoren. Während die ersten Mikroprozessoren ca. 10 Watt an elektrische Leistung verbrauchen, benötigen die aktuellen z.B 2 GHz P4 bis zu 100 Watt[1]. Langsam stellt der Energiebedarf moderner Rechner als ernstes Problem für den Rechnerarchitekt dar und wird zum Limit für die Leistungssteigerung. Mittlerweile ist nicht zu übersehen, dass die Reduktion an Energiebedarf zu den wichtigsten Aufgaben der Rechnerarchitektur gehört.

Verschiedene Studien beschäftigen sich mit dem Thema, Energiebedarf eines Rechners zu senken sowohl unter maximaler Rechenlast als auch bei der Leerlauf-Verluste durch Leckströme. Um ihren Leistungsbedarf zu senken, beherrschen viele Prozessoren eine Reihe von Techniken. Eine von diesen Techniken, über die wir hier diskutieren werden, ist Clock Gating.

Zu Beginn werden im Kapitel 2 das Prinzip und die typischen Grundschaltungen für Clock Gating Technik eingeführt. Danach folgt ein Überblick über die Hauptkomponenten des Energiekonsums. Im Kapitel 3 werden zwei Anwendungsbeispiele dargestellt, die veranschaulichen, wieviel Energie durch die Anwendung von Clock Gating gespart werden kann. Kapitel 4 stellt eine Simulation vor, die die Function Unit Belegung und das daraus folgende Sparpotential demonstriert. Zum Schluss werden Vorteile, Nachteile, Optimierungen und Ausblick im Kapitel 5 zusammengefasst.

## 2 Clock Gating

Clock Gating ist eine relativ neue Energie Reduktion Technik auf der Register Transfer Ebene. Sie nutzt die Tatsache aus, dass viele Bereiche auf dem Chip nicht in jedem Taktzyklus aktiv sind. Sie minimiert den Energiebedarf durch gezieltes Abschalten vom Clock Signals zu sequentiellen Elementen, wie FlipFlops(FFs) und Latches, und zu Dynamic Logic Gates, die in High Performance Execution Unit verwendet werden. Das Clock Signal wird während der Idle Cycles einfach abgeschaltet und es verhindert damit das unnötige Laden/Entladen der Schaltungskapazität.

### 2.1 Clock Gating Prinzip

Um Gating von Clock Signal zu realisieren, werden zusätzlich Control Signal und logische Gatter benötigt. Befindet sich das Control Signal in Low-Zustand, dann entsteht kein dynamischer Energiekonsum in einer Schaltung bzw. in dem entsprechenden Netzbereich auf dem Chip.

Es gibt zwei typische Grundschaltungen für Clock Gating[2], nämlich:

- Latch-Free Clock Gating Circuit und
- Latch-Based Clock Gating Circuit

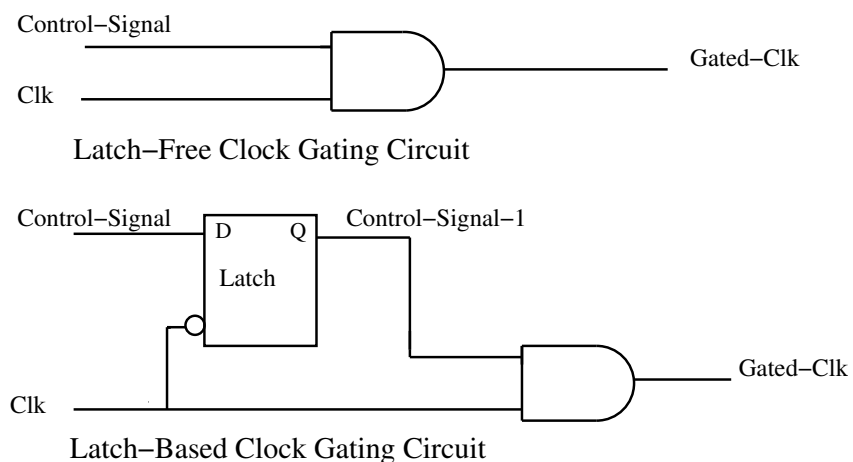


Abbildung 2.1: Clock Gating Circuit

Abbildung 2.1 veranschaulicht, wie Clock Gating prinzipiell aufgebaut ist.

Das obere Bild stellt einen Latch-Free Clock Gating Circuit dar. Anstatt das Clock Signal einfach mit dem Clock Pin einer Schaltung verbunden ist, wird es zuerst mit dem Control Signal per UND-Gatter, damit die betreffende Schaltung abhängig von Control Signal ein- oder ausgeschaltet werden kann. Darüber hinaus kann man je nach dem Register-Typ und Gating Style wie z.B Latch-Free oder Latch-Based Clock Gating, NAND-, NOR- oder OR-Gatter anstelle von UND-Gatter verwenden. Der Register-Typ unterscheidet sich z. B in Taktflankengesteuerte FlipFlops mit dem Übergang von 0 nach 1 oder Taktflankengesteuerte FlipFlops mit dem Übergang von 1 nach 0.

Das untere Bild zeigt einen Latch-Based Clock Gating Circuit, der zusätzlich noch ein Latch hat, das das möglicherweise instabile Control-Signal in das stabile Control-Signal-1 umzuwandeln.

Um die Funktionalität von Latch in Latch-Based Clock Gating besser zu verstehen, betrachten wir Abbildung 2.2[3] und Abbildung 2.3[3].

Abbildung 2.2 zeigt ein Beispiel für die Anwendung von Latch-Based Clock Gating. Das EN-Signal kommt aus einer Control Logic und wird bei dem Clockinvert-Übergang von 0 nach 1 als Input in das Latch übernommen. Dieser Eingangswert wird bis zur nächsten Clockinvert-Periode gespeichert. Damit bleibt auch das ENL-Signal während der Zeit stabil.

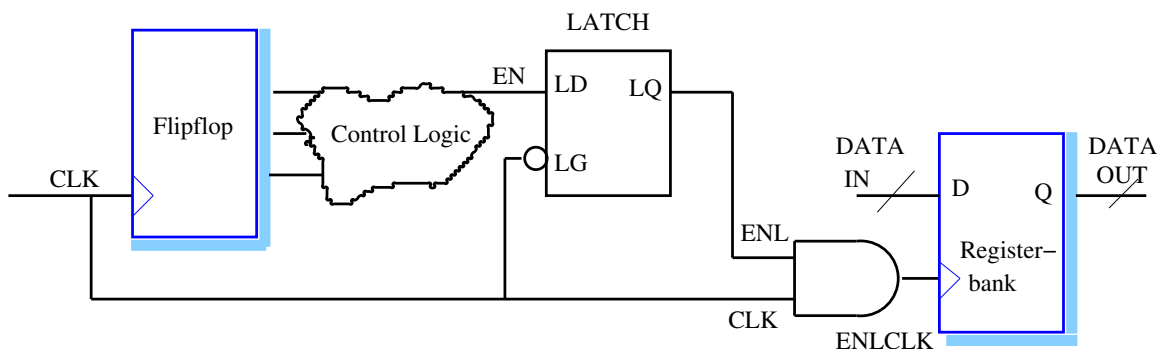


Abbildung 2.2: Latch-Based Clock Gating Circuit

Die Signalverläufe von Clock(CLK), Clock-Invert(CLK-INV), Enable-Signal(EN), Enable-Signal-Latch(ENL) und Clock Gating Logic Signal(ENLCLK) werden in der Abbildung 2.3 wie folgt beschrieben:

- 1. Periode:
  - CLK geht von 0 nach 1 über :
    - Damit geht CLK-INV von 1 nach 0 über - Keine Zustandsänderung im Latch.
    - Da ENL = 0, dann ist auch ENLCLK = 0.
    - Hiermit wird die Schaltung ausgeschaltet.

## 2 Clock Gating

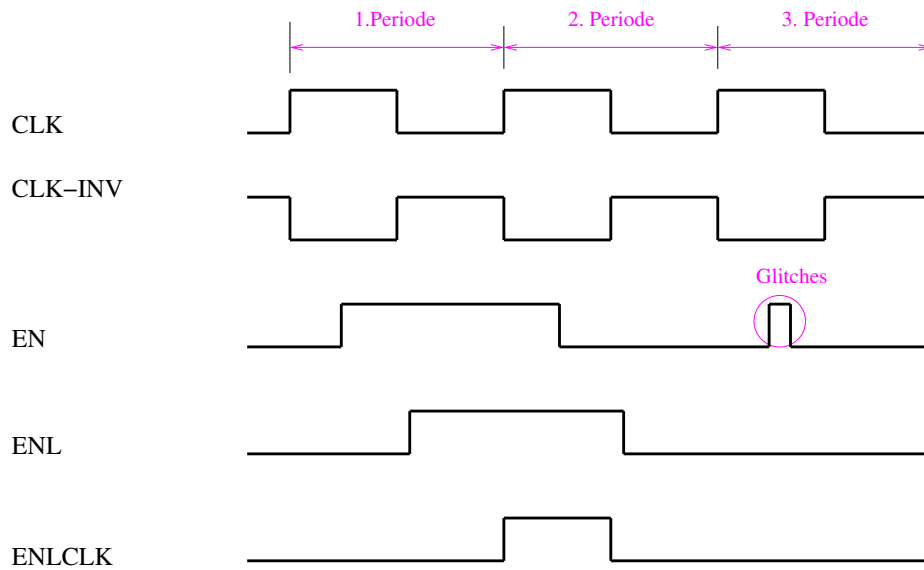


Abbildung 2.3: Latch-Based Clock Gating Signale

- CLK = 1 :  
Dann CLK-INV = 0.  
Damit erfolgt keine Zustandsänderung im Latch, obwohl EN von 0 nach 1 übergeht.  
ENLCLK = 0, da ENL = 0.  
Somit bleibt die Schaltung ausgeschaltet.
- CLK-Übergang von 1 nach 0 :  
CLK-INV geht von 0 nach 1 über - aktiver Zeitpunkt beim Latch  
Dann wird der Wert EN = 1 von Latch aufgenommen und bis zur nächsten CLK-INV-Periode gespeichert. Somit ist die Wertänderung von EN-Signal während der Zeit für Latch unsichtbar und ENL bleibt auch während der Zeit konstant.  
Da EN = 1, dann auch ENL = 1.  
Aber ENLCLK = 0, da CLK = 0. Hiermit bleibt die Schaltung ausgeschaltet.
- 2. Periode:
  - CLK geht von 0 nach 1 über:  
Dann geht CLK-INV von 1 nach 0 über.  
Da ENL := 1 und CLK = 1. Daraus folgt: ENLCLK := 1. Hiermit wird die Schaltung eingeschaltet.
  - CLK := 1:  
Damit CLK-INV := 0 und es erfolgt keine Zustandsänderung im Latch, obwohl EN geht von 1 nach 0 über.

ENL := 1. Daraus folgt: ENLCKL := 1, da CLK := 1. Hiermit bleibt die Schaltung eingeschaltet.

- CLK-Übergang von 1 nach 0:

CLK-INV geht von 0 nach 1 über - aktiver Zeitpunkt im Latch - und EN := 0.

EN := 0 wird von Latch aufgenommen und wieder bis zur nächsten CLK-INV-Periode gespeichert. Somit ist auch der Wert von ENL := 0 bis dahin unverändert. ENLCLK := 0, da ENL := 0 und CLK := 0. Hiermit wird die Schaltung ausgeschaltet.

- 3. Periode:

In dieser Periode ist ein Glitch von En-Signal während CLK-INV := 0 aufgetreten. Da bei CLK-INV-Übergang von 0 nach 1 in der zweiten Periode der stabile Wert von EN := 0 in Latch aufgenommen und gespeichert wurde, kann dieses Glitch das ENL-Signal und auch das ENLCLK-Signal nicht verfälschen, was auch der Vorteil von Latch-Based gegenüber Latch-Free Clock Gating ist.

Insgesamt zeigt es sich, dass das Latch die Signal-Änderung blockiert, wenn der Clock den Logic Wert 1 hat. Aber der Nachteil ist es, dass Latch auch Platz auf dem Chip in Anspruch nimmt.

## 2.2 Energiekonsum

Ein synchrones digitales System besteht typischerweise aus Serien von Registern und Verknüpfungselementen, die sich zwischen den Registern befinden. Die Register wiederum sind aus einer Kette von Flip Flops aufgebaut. Außerdem ist das System 'Clock' zum Clock Pin der Flip Flops und der Dynamic Logic Gates verbunden. Während sich die Register um die Systemsleistung kümmern, sind die Verknüpfungselemente für die funktionale Anforderung des Systems zuständig.

Es sind hiernach drei Hauptkomponenten von Energiekonsum zu berücksichtigen, nämlich:

- Energiekonsum durch Verknüpfungselemente
- Energiekonsum durch Flip Flops und
- Energiekonsum durch Clock System

Der Kapitel 2.3 veranschaulicht, warum ein Clock System viel Energie konsumiert.

## 2.3 Clock Distribution

Nach wie vor arbeiten die Mikroprozessoren vorwiegend mit dem Synchronen Clock System und es wird danach gestrebt, dass alle Clock Signale in dem System ihre Ziele in etwa dem gleichen

Zeitpunkt erreichen.

In solchem System besteht die Clock Periode T aus:

$$T = T_{prop} + T_{comb} + T_{setup} + T_{skew}, \text{ wobei}$$

$T_{prop}$  : die Zeit, die ein Signal braucht, um durch Flip Flops durchzugehen

$T_{comb}$ : die Zeit für ein Signal, um durch Verknüpfungselemente durchzugehen

$T_{setup}$ : die Zeit, die ein Signal braucht, um zu stabilisieren.

$T_{skew}$  : die Differenz in der Ankunftszeit des Clocksignals zwischen zwei benachbarter Orten in einem Clock System.

Abbildung 2.4[4] veranschaulicht die Komponenten der Clock Periode.

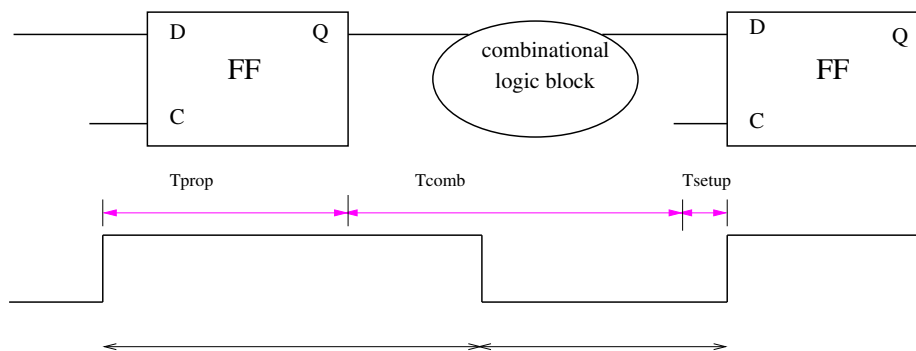


Abbildung 2.4: Clock Periode

Die Erhöhung der Clock Frequenz ( $1/T$ ) erfolgt durch die Reduzierung der Clock Periode. Aber um die Clock Periode zu reduzieren, muss Clock Skew verkleinert werden, was wiederum zu Erhöhung des Energiekonsums in Clock Netzwerk führt. Deshalb ist auch sehr wichtig, einen Kompromiß zwischen System Speed und Power Dissipation in Clock Netzwerk zu finden.

Ein Clock Netzwerk verteilt ein Clock Signal vom Clock Generator zu Clock Inputs des synchronen Komponenten. Es gibt viele Algorithmen für Clock Netzwerken, die untersucht werden, um möglichst klein Clock Skew zu erhalten.

In Abbildung 2.5 werden verschiedene Organisationsformen von Netzwerken dargestellt. Das Clock Netzwerk hat typischerweise einen Baumstruktur und besteht aus mehreren Clock Buffer Ebenen. Die Aufgabe der Clock Buffer ist es, die Clock Signale, die durch das Zusammenschalten von Impedances verschlechtert werden, zu verstärken.

Das linke Bild stellt einen Clock Tree mit Clock Buffer Ebenen dar, der geometrisch eine Maschen-Form aufweist. Die Lastwiderstände sind parallel geschaltet, was auch damit bezweckt, dass Clock

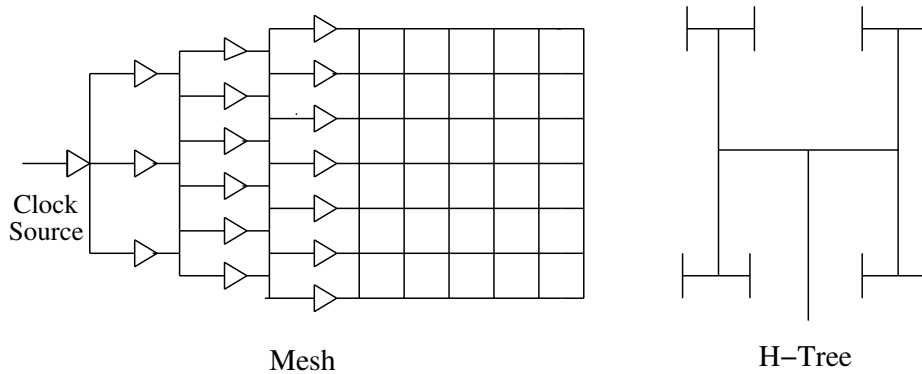


Abbildung 2.5: Clock Netzwerke

skew in dem gesamten System minimiert wird. Eine andere Struktur, nämlich der symmetrie H-Tree[5][6], hat im Vergleich zu der Maschenform zwar mehr Skew, aber verbraucht weniger Energie, da Clock Gating innerhalb der Baumstruktur leicht möglich ist.

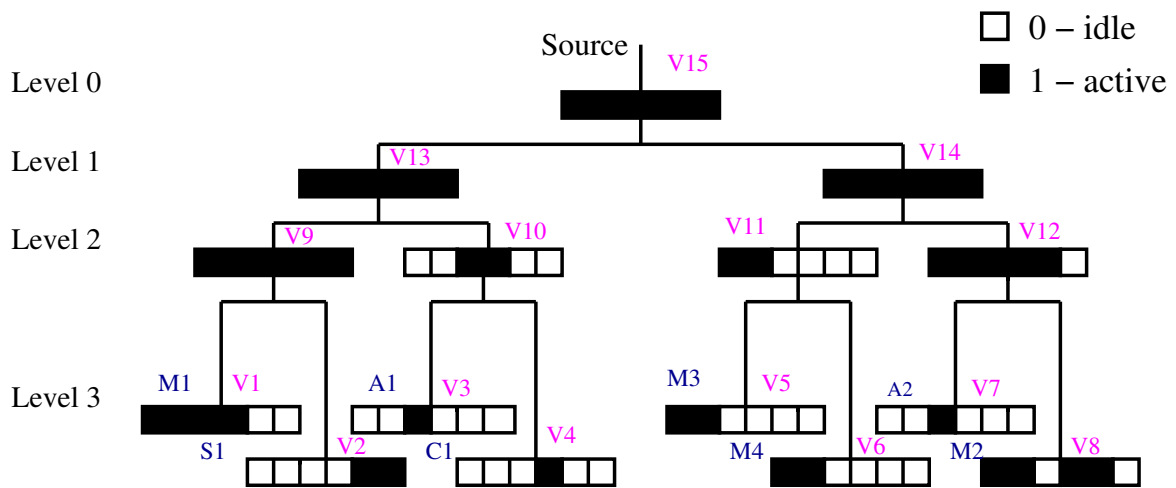


Abbildung 2.6: Clock Tree Circuit for Modules

Da für das Gating von Clock Signalen zusätzliche Control Signale und Gating Logic Gatter benötigt werden, ist es notwendig einen Kompromiß zwischen der Anzahl von Clock Tree Gatings und dem gesamten Energiekonsum des Clock Tree zu finden.

Abbildung 2.6[7] demonstriert ein Design, bei dem die Anzahl von Clock Tree Gating optimiert werden kann, und zwar durch die Ausnutzung der Ähnlichkeit in der Switching-Activity der synchronen Komponenten.

- Wir bezeichnen hier ein Set von Active/Idle Cycles eines einzelnen Moduls als Activity

Pattern, welche wir durch die Simulation[7] bekommen können.

- Auf Level 3 hängen Module an die Knoten v1 bis v8, nämlich:
  - M1, M2, M3 und M4 sind Multipliers
  - A1, A2 sind Adders
  - S1 ist Subtractor und
  - C1 ist Comparator.
- Auf Level 1 und 2 befinden sich die inneren Knoten, deren Activity Patterns durch bitweise OR-Operation der Activity Patterns des einzelnen Kind-Moduls resultieren. Das Beispiel unten veranschaulicht diese bitweise OR-Operation, wobei '0' und '1' idle bzw. active Zyklen darstellen:
  - A1 : Activity Patterns 001000 , und
  - C1 : Activity Patterns 000100 , damit
  - v10 : Activity Patterns 001100
- Auf Level 0 wird ein Clock Signal vom Clock Source(Clock Generator) erzeugt, das von der Knoten v15 aus an einzelnes Modul geschickt wird.

Somit ist es möglich mehrere Moduls in dem selben Clock Subtree durch ein Clock Gating zu steuern.

# 3 Anwendungsbeispiele

In diesem Kapitel werden zwei Beispiele für die Anwendung von Clock Gating vorgestellt, nämlich das Deterministic Clock Gating(DCG) und das Value-Based Clock Gating.

## 3.1 Deterministic Clock Gating

Deterministic Clock Gating(DCG) basiert auf die Beobachtung, dass für viele Stufen in der modernen Pipeline die Benutzung einer Schaltung in ein paar Zyklen voraus festgelegt werden kann. Somit wird eine Methodologie benötigt, die festlegt, wo wann und wie lange ein Clock Signal zu einer Schaltung ausgeschaltet werden kann. In unserem Beispiel wird DCG auf ein 8-issue, Out-Of-Order Superscalar Processor mit acht Stufen Pipeline untersucht.

Abbildung 3.1[8] stellt die Anwendung von DCG auf die Execution Units dar.

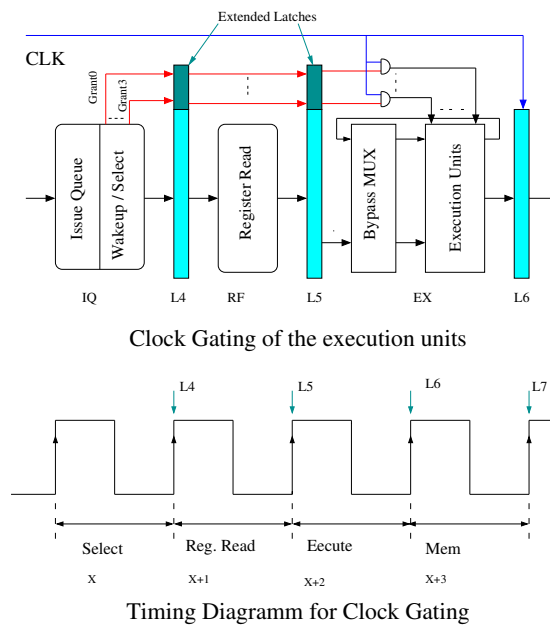


Abbildung 3.1: Deterministic Clock Gating bei ExecutionUnits

Die Pipeline hier umfasst acht Stufen, nämlich Instruction Fetch, Instruction Decode, Rename, Issue, Register Read, Execute, Memory und Writeback, wobei wir nur von Issue Queue bis Execute betrachten.

Die Issue Stufe verfügt über eine Issue Queue, die wiederum aus assoziativem Array besteht, und ein Wakeup/Select Combination Logic, um die verfügbaren Instruktionen zu selektieren und dann die entsprechenden GRANT-Signale zu erzeugen, die zum Clock Gate Control jedes Execution Unit gesendet werden. Dabei werden die Pipeline Latches für die Register Read und Execution Stufe um ein paar Bits erweitert, damit sie das GRANT-Signal aufnehmen, speichern und weitersenden können. Da erst am Ende der Issue Stufe festgelegt werden kann, welche Execution Units in der Ausführungsphase besetzt werden, hat das GRANT-Signal ungefähr einen Zyklus Zeit, um an Clock Gating Gatter zu gelangen. Das Ausgangssignal, nämlich das Clock Gating Signal wird zu dem entsprechenden Unit geleitet.

In dieser Anwendung ist es wichtig, dass eine sequentielle Priorität Strategie für Execution Units gleichen Typs verwendet wird, nämlich eine Strategie, bei der die Priorität einzelnes Units festgelegt wird. Die Unit, die höhere Priorität besitzen, werden zuerst besetzt, wenn es verfügbare Befehle gibt. Somit sind die Units mit hoher Priorität immer beschäftigt, während die Units mit niedriger Priorität für eine lange Zeit durch Clock Gating abgeschaltet werden.

Abbildung 3.2[8] zeigt die Ergebnisse, wieviel Prozent Energie vom Energieverbrauch durch DCG gespart werden.

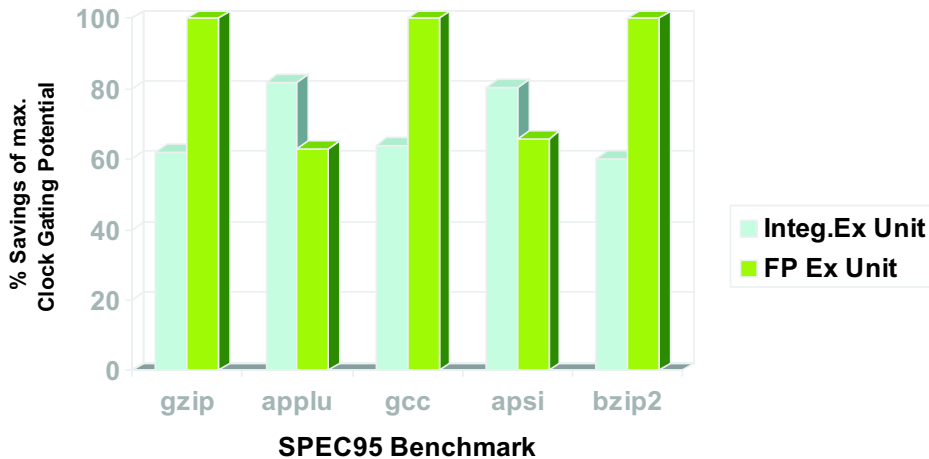


Abbildung 3.2: Execution Unit Power Savings

- Die ersten Säulen jeder Gruppe stellen Power Savings bei den Integer Execution Units dar.
- Die jeweils zweiten Säulen zeigen Power Savings bei den Floating point Execution Units.

Insgesamt zeigt sich, dass man bei der Verwendung von SPECint95 Benchmarks die Energie bei Integer Execution Unit ca. 65% Energie sparen kann, während bei Floating Point Execution Unit sogar bis 100%.

## 3.2 Value-Based Clock Gating

Ein anderes Beispiel für die Anwendung von Clock Gating ist das Value-Based Clock Gating.

Die Grundidee dieses Verfahrens resultiert aus den Untersuchungen, dass viele 64-Bit Architektur Prozessoren im allgemeinen nicht wirklich die 64-Bit Adresse und Operationen verwenden. Viel mehr benötigen die arithmetischen Operationen Operanden, die kleiner als 64-Bit Datenbreite haben. Noch dazu unterstützt die neue Tendenz der Instruction Set Architektur die Parallelität, indem verschiedene Teilwörter in der selben ALU gleichzeitig ausgeführt werden dürfen. Viele Prozessoren schliessen Instruction Set für Multimedia Operationen ein, die überwiegend mit 8-Bit oder 16-Bit Operanden arbeiten. Solche Operationen sind nicht nur in Multimedia Code zu finden, sondern auch in anderen Anwendungen. Zum Beispiel bei SPECint95 Benchmarks, operieren mehr als die Hälfte der Operationen mit 16-Bit oder weniger breiten Operanden.

Die Value-Based Clock Gating Technik nutzt diese Tatsache aus, um den Energiebedarf bei dem 64-Bit Integer Execution Unit zu reduzieren. Ohne Unterstützung von Compiler benötigt diese Technik einen Hardware-Mechanismus für die dynamische Erkennung und Ausnutzung der schmalen Datenbreite.

Es gibt verschiedene Hardware Implementierungen für diese Technik. Abbildung 3.3[9] stellt eine Varianz der Value-Based Clock Gating Architektur dar.

Während der Laufzeit kann diese Einheit erkennen, ob die erst höheren 48-Bit der Input Operands null sind oder nicht. Zum Beispiel in einer Addition, wenn diese 48-Bit Null sind, werden diese nicht in Latch gespeichert und auch nicht zu Execution Unit weitergeleitet.

Die übrige 16-Bits werden wie üblich in Pipeline Latch für diese Execution Stufe gespeichert und dann zu Integer Functional Unit geschickt. Die ersten 48-Bits der Operanden werden getrennt in einem extra Latch gespeichert, abhängig von einem Zero48-Signal, das den Input Operand seit von Reservation Stations oder seit von Bypass Network begleitet. Das Zero48-Signal wird von Zero Detection Logic generiert und setzt das entsprechende Flag, wenn die 48 höhere Bits null sind. Da manche Operanden direkt von Cache kommen, werden sie bereits beim Laden auf Zero Check geprüft.

Wenn wir wissen, dass die zwei Input Operanden 16-bit oder weniger als 16-bit Breit sind, dann

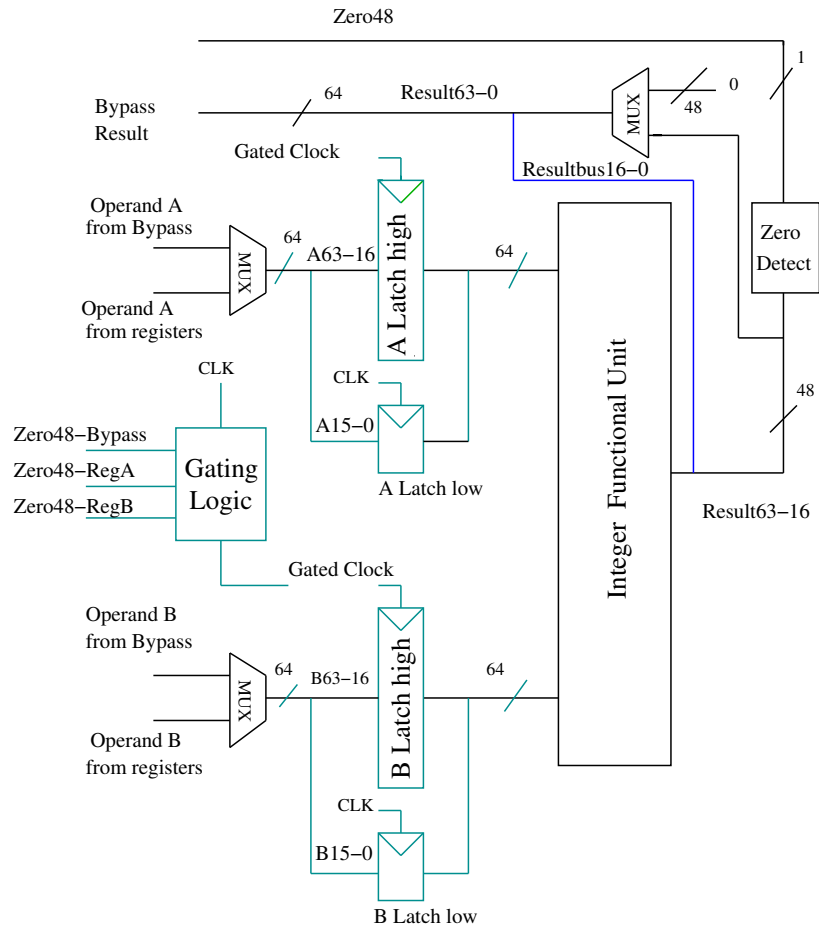


Abbildung 3.3: Operand-Based Clock Gating Architektur

ist es relativ einfach die Datenbreite des Ergebnisses festzulegen. Zum Beispiel in einer Addition, falls das Carry Out Signal der 16-bit Stelle null ist und die Operanden 16-bit Breit oder weniger als 16-bit Breite haben, dann wissen wir, dass das Ergebnis maximal nur 16-bit Breit hat. Es ist wichtig, dass das Zero48 Signal erst nach dem Carry Out Signal generiert wird, aber jedenfalls bevor das Addition-Ergebnis fertig gestellt wird.

Es ist möglich, dass solche Zero Detect Hardware und die dazugehörigen Flags in den Reservation Stations mancher Prozessoren schon vorhanden und zwar wegen der Untersuchung auf Division durch Null. Das Gated Clock Signal wird dazu verwendet, um den 48-bit Teil der 64-bit ALU abzuschalten. Es wird durch ANDing von den Zero48 Signals jeweiligen Operanden realisiert.

Abbildung 3.4[9] veranschaulicht den Energiekonsum einzelner Programme von SPECint95 Benchmarks unter Anwendung von Operand-Based Clock Gating auf die 64-bit ALU.

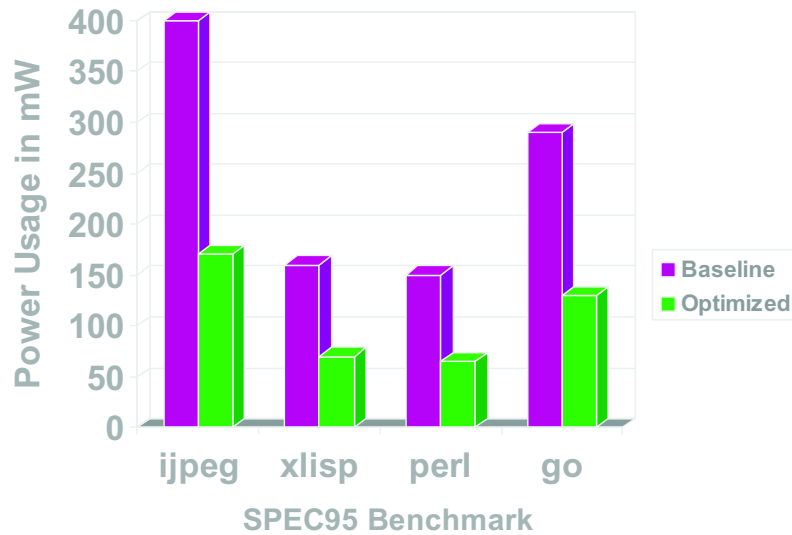


Abbildung 3.4: Power Usage bei 64-Bit Integer Execution Unit

Insgesamt zeigt sich, dass bei der mit Value-Based Clock Gating Optimierten Architektur der durchschnittliche Energieverbrauch auf ca. 54% des ursprünglichen Energieverbrauchs reduziert.

# 4 Simulation

## 4.1 Ziel der Simulation

In diesem Kapitel wird eine Simulation vorgestellt. Ziel dieser Arbeit ist es, die Belegung einzelner Functional Unit zu simulieren und das daraus folgende Sparpotential zu demonstrieren.

Abbildung 4.1 zeigt grob, wie die Simulation abläuft.

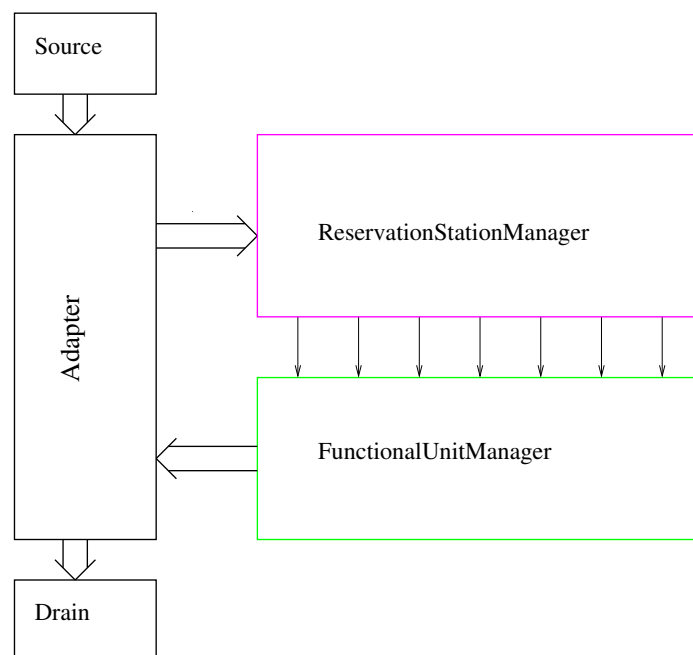


Abbildung 4.1: Simulation

Zu Beginn des Kapitels wird die Arbeitsweise der Implementierung grob beschrieben. Danach folgen die Details, wie die einzelne Komponenten z.B die Reservation Stations und Functional Units, arbeiten und miteinander kommunizieren. Nach der Beschreibung der Implementierung werden die Belegung von Functional Units und das Sparpotential ohne und unter Berücksichtigung der Datenabhängigkeit von Befehlen vorgestellt und diskutiert. Zuletzt werden die Simulationsergebnisse

in Bezug auf bestimmte Konfigurationsdaten und auf bestimmte Programme aufgelistet

## 4.2 Überblick auf die Arbeitsweise der Implentierung

Hier sind zwei wichtige Teile zu implementieren, nämlich die Reservation Stations und die Functional Units, die eigenständig von den ReservationStation- und FunctionalUnitManagern verwaltet werden. Zudem wird ein Adapter benötigt, der die Kommunikation zwischen Source, Drain, ReservationStation- und FunctionalUnitManager ermöglicht.

Die Befehle von Source werden in Form von Envelopes in einer Liste eingepackt, die dann mit Hilfe von dem Adapter an ReservationStationManager weitergegeben. Da für uns nur die Belegung von Functional Units und die Reihenfolge der Befehle von besonderer Wichtigkeit sind, werden die Ergebnisse von Befehlen nicht ermittelt. Das einzelne Functional Unit gibt nach der Verarbeitung die Envelopes zurück, die dann von dem FunctionalUnitManager in eine Liste eingepackt und an Drain weitergeliefert wird.

Bei den Reservation Stations wird angenommen, dass stets genügend Speicher vorhanden ist und wir betrachten deshalb nur die Datenabhängigkeit von Befehlen. Um diese Datenabhängigkeit ermitteln zu können, wird noch eine Komponente eingebaut, nämlich die Komponente ActiveInExecution. Diese Komponente speichert und verwaltet Befehle, die sich während der Execution in Functional Units befinden.

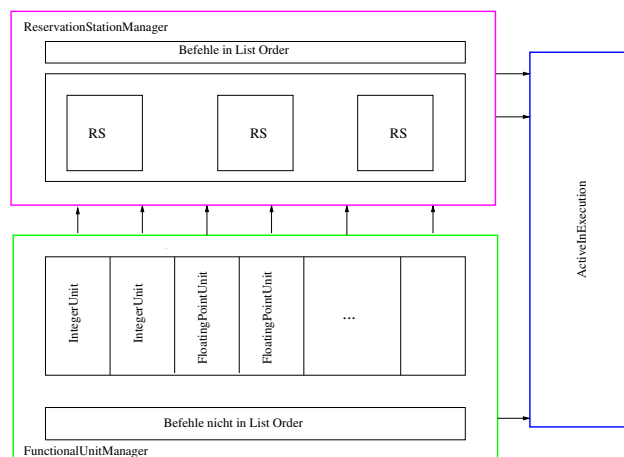


Abbildung 4.2: Erweiterung der Simulation

Abbildung 4.2 zeigt ActiveInExecution zusammen mit ReservationStationManager und FunctionalUnitManager.

## 4.3 Die Simulation im Detail

### 4.3.1 Die Reservation Station im Detail

Im folgenden veranschaulicht Abbildung 4.3 ein Reservation Station im Detail.

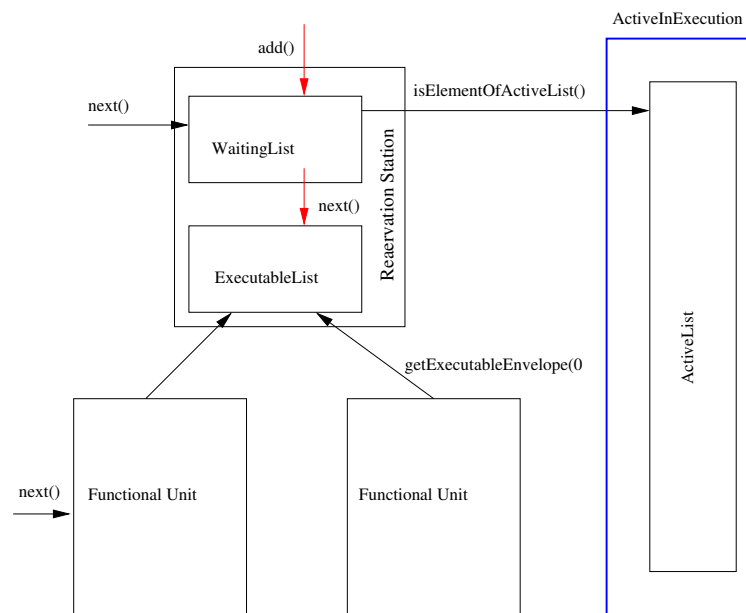


Abbildung 4.3: Reservation Station im Detail

- Reservation Station - Für diese Komponente wird die Klasse ReservationStation geschrieben, die die Methoden add(), next() und getExecutableEnvelope() enthält, um die Zuständigkeit von Reservation Station zu realisieren und die Schnittstelle zu Functional Unit zu bauen.
  - add() - fügt die in jedem Zyklus ankommende Envelope in WaitingList ein, dessen Element aus einem Paar Envelope und seine Abhängigkeitsliste besteht.
  - next() - ruft die Methode isElementOfActiveList() auf, um die Datenabhängigkeit von Befehlen in WaitingList zu prüfen. Besteht keine Datenabhängigkeit zwischen den Befehlen, dann wird das Envelope an die ExecutableList weitergegeben. Dieses Envelope ist bereit für die Operation in einem Functional Unit.
  - getExecutableEnvelope() ermöglicht einem Functional Unit, ein Envelope aus der ihm zugeordneten Reservation Station zur Verarbeitung zu holen.
- ReservationStationManager - implementiert durch die Klasse ReservationStationManager. Die wichtigste Methoden, die die Funktionalität von ReservationStationManager erfüllen

sollen, sind:

- next()- verwaltet die Reservation Stations
- add() - ermöglicht das Einfügen der Envelope-Liste in die ActiveList und veranlässt gleichzeitig das Einfügen des Envelope in die zugeordnete Reservation Station.

### 4.3.2 Die Kommunikation zwischen allen Komponenten

Die Zuständigkeit der restlichen Komponenten und die Kommunikation zwischen allen Komponenten untereinander werden in Abbildung 4.4 verdeutlicht.

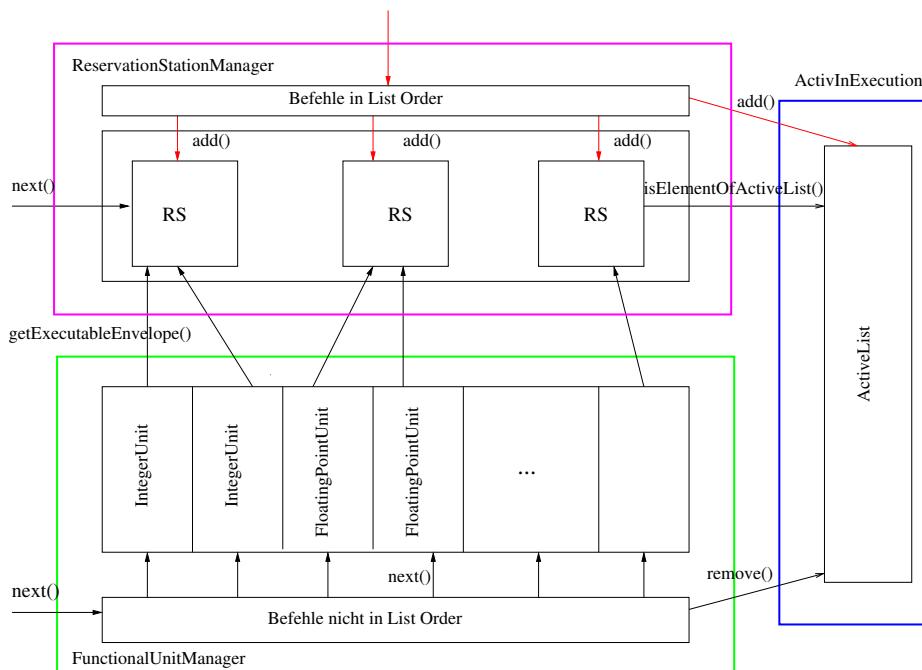


Abbildung 4.4: Simulation im Detail

- FunctionalUnitManager - verwaltet die Functional Units durch Methode next(). Hiermit wird beachtet, dass ein Functional Unit mit hoher Priorität zuerst belegt wird. Außerdem gibt es noch die Methode idleCycleStatus(), die für die Ausgabe der Elemente von maxIdleListing in aufsteigend sortierte Reihenfolge von maxIdle Cycles zuständig ist.
  - next() - ermöglicht das Einpacken der Envelope nach der Verarbeitung in eine Liste und löscht den Befehl, der gerade in Functional Unit fertig verarbeitet wurde, von ActiveList.
- Functional Unit - implementiert durch die Klasse FunctionalUnit, derer Funktionalität durch die Methode next() realisiert wird.

next() - ist der Mittelpunkt sowohl dieser Klasse als auch der gesamten Simulation. Durch den Aufruf von next() wird die Functional Unit von einem Zyklus weitersimuliert. Dabei werden zwei Fälle geprüft:

- Fall 1: Ist die Functional Unit im Moment bei der Verarbeitung eines Envelope, dessen Latency gerade den Wert Null hat, dann ist die Verarbeitung fertig. Die Unit gibt das Envelope zurück.  
Andernfalls wird die Latency um 1 dekrementiert.
- Fall 2: Ist die Functional Unit nicht besetzt, dann ruft das Functional Unit die Methode getExecutableEnvelope() auf, um ein Befehl-Envelope aus der ihm zugeordneten Reservation Station zu holen. Dabei wird noch geprüft, ob es in ExecutableList dieser Reservation Station ein Envelope zu holen oder zu verarbeiten gibt. Wenn ja, dann wird dieses Envelope in die Functional Unit geholt. Gleichzeitig wird die Latency dieses Envelope dekrementiert und Beschäftigungszyklen(Busy Cycles) dieser Unit inkrementiert. Ansonsten bleibt die Unit weiterhin unbesetzt und seine Nicht-Beschäftigungszeit(Idle Cycles) wird inkrementiert. Um Anzahl der Zyklen jeder Pausezeit(maxIdle) einzelnes Units auflisten zu können, benötigen wir die Komponente FuIdleCollector. Der FuIdleCollector speichert alle auftretenden Möglichkeiten von maxIdle Cycles und derer Häufigkeit.
- FuIdleCollector - realisiert durch die Klasse FuIdleCollector.  
Die wichtige Methoden sind idle(), busy() und getIdleCycleCount().
  - idle() - zählt Zyklen ohne Arbeit (Idle Cycles).
  - busy() - zählt Zyklen mit Arbeit (Busy Cycles).
  - getIdleCycleCount() - Durch diese Methode kann auf die Häufigkeit von längeren Idle Zeiten zugegriffen werden. Sie unterstützt nämlich die Ausgaben von Ergebnissen der Simulation.
- ActiveInExecution - verwaltet Befehle, die sich während der Execution in Functional Units und in Reservation Stations befinden. Diese Komponente wird durch die Klasse ActiveInExecution implementiert, die folgenden Methoden enthält, nämlich:
  - add() - fügt die Envelope-Liste , die in jedem Zyklus von Adapter rausfließt, in die ActiveList ein. Sie wird durch add() von ReservationStationManager aufgerufen.
  - isElementOfActiveList() - prüft für einen Befehl(Envelope) von WaitingList, ob der gleiche Befehl auch in ActiveList enthalten ist.
  - remove() - löscht Befehle, die gerade fertig verarbeitet wurden, aus der ActiveList.

## 4.4 Konfiguration

Zur Wahrung der Architektur-Unabhängigkeit werden Daten wie folgt konfiguriert:

- Das Instruction Set entspricht dem der 32-Bit PowerPC Microprocessor Family[10]
- Die Benennung von Functional Units, die Zuordnung von Befehlen zu Functional Units basieren auf die Unterlagen von PowerPC 604[11]
- Die Anzahl von Functional Units wird flexibel festgelegt und zwar unter Berücksichtigung der Implementierungsbedingungen.

In der Tabelle 4.1 werden Functional Unit Typen und die Anzahl des jeweiligen Typs aufgelistet.

```
#Functional Unit Listing
#Functional Units nach PowerPC 604 benannt
# Fu-Type          |          NumberOfFunctionalUnit
#-----|-----
SFX                |          2          # Simple Integer Unit
CFX                |          1          # Complex Integer Unit
FXU                |          1          # Fixed-Point Unit
FPU                |          1          # Floating Point Unit
LSU                |          4          # Load-Store Unit
BPU                |          1          # Branch Processing Unit
unbek              |          1          # Zuordnung unklar
```

Tabelle 4.1: Functional Units Listing

Diese Tabelle wird in einer Datei namens FuncReservaStationPowerPc604.txt festgehalten, die während der Simulation eingelesen wird.

Außer CFX, FXU, FPU und BPU, die nur eine Unit haben, werden LSU mit 4 units und SFX mit 2 Units festgelegt. Die Auswahl von 4 LSU liegt darin begründet, dass das DES-Trace Programm viel Load- und Store-Befehle enthält. Diese Menge könnte sich sehr negativ auf die Ermittlung von Datenabhängigkeiten auswirken, z. B die Abhängigkeitskette enthält mehr als 100 Befehle, wenn die Anzahl von Load/Store Units kleiner als 4 ausgewählt wird. Außer LSU hat auch SFX so viele Befehle zu verarbeiten, so dass 2 SFX Units benötigt werden. Hier ist es auch zu beachten, dass so viel Units gleichen Typs als nötig Energieverschwendung verursacht. Außerdem wird die Functional Unit namens 'unbek' festgelegt, um ein paar Befehle von Supervisor Level oder Optional Instructions aufzunehmen, da ihre Zuordnung unklar sind.

Tabelle 4.2 listet einige Befehle von PowerPC Familie auf, während die restlichen Befehle in Anhang A aufgelistet sind.

```
# Instruction Set Listing
# Zuordnung Instruction-Functional Unit anhand PowerPC 604
# Latency anhand PowerPC 604 festgelegt
```

# Mnemonic	Functional Unit	Latency
add	SFX	1
add.	SFX	1
addc	SFX	1
adde	SFX	1
adde.	SFX	1
addi	SFX	1
addic	SFX	1
addic.	SFX	1
addis	SFX	1
addme	SFX	1
addze	SFX	1
divw	CFX	21
divwu	CFX	21
mulhw	CFX	5
mulhwu	CFX	5
mulli	CFX	3
mullw	CFX	5
mullw.	CFX	5
neg	SFX	1
neg	SFX	1
subf	SFX	1
subfc	SFX	1
subfic	SFX	1

Tabelle 4.2: Integer Arithmetic Instructions

Zusätzlich werden ihre Latencies und ihre Zuordnung zu Functional Unit Typ angegeben, die für die Simulation von Functional Units sehr entscheidend sind. Analog wie die Functional Units List werden die gesamte Daten von Instruction Set Listing in eine Datei namens InstructionSetListing-PowerPC604.txt festgehalten, die während der Simulation eingelesen wird.

## 4.5 Functional Unit Belegung und Sparpotential ohne Berücksichtigung der Datenabhängigkeit

Tabelle 4.3 veranschaulicht die Belegung von Functional Units im Idealfall, d.h ohne Berücksichtigung von Structural Hazard und Datenabhängigkeiten.

FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0: andi.	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0: andi.	SFX0: rlwinm	SFX1: add
FPU0:	LSU0: lhzx	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: rlwinm	SFX1: add
FPU0:	LSU0: lhzx	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: rlwinm	SFX1: add
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: add	SFX1:
FPU0:	LSU0: stw	LSU1: lh	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1:
FPU0:	LSU0: stw	LSU1: lh	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0:	LSU1: lh	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1:
FPU0:	LSU0: lbz	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: rlwinm.	SFX1:
FPU0:	LSU0: lbz	LSU1: lwz	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lbz	LSU1: lwz	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2: lwz	LSU3:	BP00:	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1:	LSU2: lwz	LSU3:	BP00:	CFX0:	FXU0:	SFX0: or	SFX1: cmp
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: or	SFX1:
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: rlwinm	SFX1: xori
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: subfic	SFX1: xori
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: adde	SFX1: xori
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: subfic	SFX1: xori
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: adde	SFX1: or.
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: addi	SFX1: or.
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: cmpi	SFX1: rlwinm
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1:
FPU0:	LSU0: lwz	LSU1: stw	LSU2: lwz	LSU3: stw	BP00:	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: stw	LSU2: lwz	LSU3: stw	BP00:	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2: lwz	LSU3: stw	BP00: b	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: add	SFX1: add
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0: mfer	FXU0:	SFX0: rlwinm	SFX1:
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0: mfer	FXU0:	SFX0: xori	SFX1: subfic
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0: mfer	FXU0:	SFX0: xori	SFX1: adde
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0: mfer	FXU0:	SFX0: xori	SFX1: adde
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: or.	SFX1: cmpi
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: or.	SFX1:
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BP00: b	CFX0:	FXU0:	SFX0: addi	SFX1:
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: cml	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0: andi.	SFX0: rlwinm	SFX1:
FPU0:	LSU0: lhzx	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0: andi.	SFX0: add	SFX1:
FPU0:	LSU0: lhzx	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: rlwinm	SFX1:
FPU0:	LSU0:	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: add	SFX1: rlwinm
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: add	SFX1: add
FPU0:	LSU0: stw	LSU1: lh	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: cmpi	SFX1:
FPU0:	LSU0: lbz	LSU1: lh	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lbz	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: rlwinm.	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lwz	LSU1: lwz	LSU2: lwz	LSU3:	BP00:	CFX0:	FXU0:	SFX0: or	SFX1:
FPU0:	LSU0: lwz	LSU1:	LSU2: lwz	LSU3:	BP00:	CFX0:	FXU0:	SFX0: or	SFX1: cmp
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1:
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: xori	SFX1: subfic
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: xori	SFX1: adde
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: xori	SFX1: subfic
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: xori	SFX1: adde
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: or.	SFX1: cmpi
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00: bc	CFX0:	FXU0:	SFX0: or.	SFX1: xori
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: rlwinm	SFX1: xori
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: subfic	SFX1: xori
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: adde	SFX1: rlwinm
FPU0:	LSU0: stw	LSU1: lwz	LSU2:	LSU3:	BP00:	CFX0:	FXU0:	SFX0: or	SFX1:
FPU0:	LSU0: stw	LSU1: lwz	LSU2: lwz	LSU3:	BP00:	CFX0:	FXU0:	SFX0: or	SFX1: cmpi

Tabelle 4.3: Functional Unit Belegung ohne Berücksichtigung der Datenabhängigkeit

Die Spalten LSU0, LSU1, LSU2 und LSU3 vertreten die Functional Units, die gleichen Typ aufweisen. Sie werden mit Prioritätswerten eingestuft, nämlich eine Unit, die hohe Priorität hat, bekommt zuerst die Arbeit und nur wenn diese belegt ist, dann übernimmt eine andere Unit, die die nächsthöheren Priorität besitzt, diese Arbeit. Analog gilt bei den Units SFX0 und SFX1. Für die Simulation werden ca. 130 Befehle hier verfolgt, und zwar von der Nummer 17472 bis zu 17599.

Hier ist ersichtlich, dass das Unit SFX0 fast immer belegt ist, während die Units LSU0, LSU1, BPU0 und SFX1 öfter abgeschaltet werden können. Die anderen Units, nämlich LSU2, LSU3, CFX0 und FXU0 sind nur ab und zu beschäftigt. man könnte sogar denken, dass LSU2 und LSU3 überflüssig sind. Außerdem da das Testprogramm ein reines Integer-Programm ist, bekommt FPU0 keine Befehle zu verarbeiten. Insgesamt haben die Units LSU0, LSU1, LSU2, LSU3, BPU0, SFX1 und FPU0 das Sparpotential und ist dort das Einbauen von Clock Gating sehr sinnvoll, um Energie zu sparen.

## 4.6 Functional Unit Belegung und Sparpotential unter Berücksichtigung der Datenabhängigkeit

Um die Ergebnisse praxisnah zu erhalten, werden die ca 130 Befehle von Kapitel 4.3 mit Berücksichtigung von Datenabhängigkeiten verfolgt.

FPU0:	LSU0: lzw	LSU1: lbzu	LSU2: lzw	LSU3:	BPU0: b	CFX0:	FXU0:	SFX0: or.	SFX1: or
FPU0:	LSU0:	LSU1: lbzu	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: subfic	SFX1: or
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: addi	SFX1: cmpli
FPU0:	LSU0: stw	LSU1: lbzu	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: stw	LSU1: lbzu	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0:	LSU1:	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: lbzu	LSU1: lzw	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: rlwinm.
FPU0:	LSU0: lbzu	LSU1: lzw	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0: andi.	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: ande	SFX1: cmpl
FPU0:	LSU0: lbzu	LSU1: stw	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: lbzu	LSU1: stw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1: add
FPU0:	LSU0:	LSU1: stw	LSU2:	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: lbzu	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmpli
FPU0:	LSU0: lbzu	LSU1: lzw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: lzw	LSU1: lzw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: add	SFX1: add
FPU0:	LSU0: stw	LSU1: lbzu	LSU2: stw	LSU3: lhzx	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: or
FPU0:	LSU0: stw	LSU1:	LSU2: stw	LSU3: lhzx	BPU0: bc	CFX0:	FXU0:	SFX0: cmp	SFX1: add
FPU0:	LSU0: stw	LSU1: lbzu	LSU2: stw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: add	SFX1: rlwinm
FPU0:	LSU0: stw	LSU1: lbzu	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: stw	LSU1: lzw	LSU2: lzw	LSU3: stw	BPU0: bc	CFX0:	FXU0:	SFX0: addi	SFX1: rlwinm
FPU0:	LSU0: stw	LSU1: lzw	LSU2: lzw	LSU3: stw	BPU0: bc	CFX0:	FXU0:	SFX0: xori	SFX1: xori
FPU0:	LSU0: lhz	LSU1: lbz	LSU2: lbzu	LSU3: stw	BPU0: bc	CFX0:	FXU0:	SFX0: xori	SFX1: xori
FPU0:	LSU0: lhz	LSU1: lbz	LSU2: lbzu	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: rlwinm
FPU0:	LSU0: lzw	LSU1: lzw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0: lzw	LSU1: lzw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: add	SFX1: cmpli
FPU0:	LSU0: lzw	LSU1: lzw	LSU2:	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: subfic
FPU0:	LSU0: lzw	LSU1: lzw	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: subfic	SFX1: cmpi
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmp	SFX1: subf
FPU0:	LSU0: lzw	LSU1: lzw	LSU2: lzw	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: cmpi	SFX1: rlwinm.
FPU0:	LSU0: lzw	LSU1: lzw	LSU2: lzw	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: cmpi	SFX1: cmp
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: add	SFX1: add
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: addi	SFX1: cmp
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BPU0: b	CFX0: mfcr	FXU0:	SFX0: ande	SFX1: adde
FPU0:	LSU0: stw	LSU1: stw	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: cmpl	SFX1: xori
FPU0:	LSU0: stw	LSU1: stw	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: rlwinm	SFX1: xori
FPU0:	LSU0: lzw	LSU1: stw	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: add	SFX1: add
FPU0:	LSU0: lzw	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: cmpi	SFX1: or.
FPU0:	LSU0: stw	LSU1: stw	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1: or.
FPU0:	LSU0: stw	LSU1: stw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: addi	SFX1: or.
FPU0:	LSU0: stw	LSU1: stw	LSU2: lzw	LSU3: lzw	BPU0: bc	CFX0:	FXU0:	SFX0: subfic	SFX1: or.
FPU0:	LSU0: lzw	LSU1: lzw	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1: cmpi
FPU0:	LSU0: lzw	LSU1: lzw	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: or.	SFX1: cmpi
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0:	SFX0: or.	SFX1: add
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0: andi.	SFX0: ande	SFX1: or.
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0: mfcr	FXU0: andi.	SFX0: rlwinm	SFX1: or.
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: add	SFX1: add
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BPU0: b	CFX0:	FXU0:	SFX0: rlwinm	SFX1: add
FPU0:	LSU0: stw	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1: cmpi
FPU0:	LSU0: stw	LSU1: lhzx	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: or.	SFX1: cmpi
FPU0:	LSU0: stw	LSU1: lhzx	LSU2: lzw	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: or.	SFX1: add
FPU0:	LSU0: lzw	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0:	SFX1:
FPU0:	LSU0: lzw	LSU1: stw	LSU2: lhz	LSU3: lbz	BPU0: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1: or
FPU0:	LSU0: lzw	LSU1: stw	LSU2: lhz	LSU3: lbz	BPU0: bc	CFX0:	FXU0:	SFX0:	SFX1: or
FPU0:	LSU0: lzw	LSU1: stw	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: add	SFX1: rlwinm
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: rlwinm.

Tabelle 4.4: Functional Unit Belegung unter Berücksichtigung der Datenabhängigkeit

## 4 Simulation

FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: cmpi	SFX1: rlwinm.
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: xori	SFX1: xori
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: xori	SFX1: xori
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: addi	SFX1: cmpi
FPU0:	LSU0:	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: rlwinm	SFX1: cmp
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: subfic	SFX1: subfic
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BPU0: bc	CFX0:	FXU0:	SFX0: xori	SFX1:
FPU0:	LSU0: lwz	LSU1:	LSU2:	LSU3:	BPU0:	CFX0:	FXU0:	SFX0: xori	SFX1: adde
FPU0:	LSU0: lwz	LSU1: lwz	LSU2:	LSU3:	BPU0:	CFX0:	FXU0:	SFX0: adde	SFX1: cmpi

Tabelle 4.5: Functional Unit Belegung unter Berücksichtigung der Datenabhängigkeit(Fortsetzung)

Im Unterschied zur Tabelle 4.3 bekommen die Units LSU0, LSU1, LSU2 und LSU3 mehr Arbeit. Hier ist die Überlegung von 4 LSUs richtig, um den Kompromiß zwischen der Systemleistung und dem Energiebedarf zu erhalten. Auch die Auswahl von 2 FSX und 1 BPU scheint optimal. zu sein. Analog wie vorheriger Tabelle können CFX0, FXU0 und FPU0 für lange Zeit abgeschaltet werden, da sie nur selten oder nicht besetzt sind. Die anderen Units, nämlich LSU0, LSU1, LSU2, LSU43, CFX0 und FXU0, können während der Nicht-Beschäftigungszeit abgeschaltet werden.

### 4.7 Analyse der Simulationsergebnisse

Nach der Analyse des Sparpotentials werden hier verschiedene Ergebnisse der Simulation vorgestellt. Als Testfall 1 wird zuerst eine Kurze Version von DES-Trace PowerPC, die ca. 40000 Befehle umfasst, ausgeführt und zwar mit 2 neu eingeholten Befehlen in jedem Takt.

Die Ergebnisse werden in Tabellen 4.6 und 4.7 festgehalten.

In den Tabellen werden alle auftretenden maxIdle-Klassen und derer Häufigkeit in der jeweiligen Functional Unit aufgelistet und veranschaulicht.

Es ist auffällig, dass die Units von Typen 'FPU0' und 'unbek' keinerlei maxIdle haben, was auch bedeutet, dass sie während der Ausführung keine kurze Pause-Zeit aufweisen, sondern vom Anfang bis zum Ende des Programms unbesetzt bleiben. Diese Units können ganz abgeschaltet werden und der Leckstrom spielt hier eine entscheidende Rolle.

Bei den meisten Units, außer FXU0, treten maxIdle von einem Zyklus sehr häufig auf, d.h, das Energie-Sparen durch Clock Gating nicht realisierbar ist. Danach folgen die maximalen Idles zwischen 2 und 10 Zyklen. In diesem Fall ist die Realisierung von Clock Gating möglich. Bei manchen Units(außer LSU0), wie z.B LSU1, LSU2, LSU3, BPU0, CFX0, FXU0, SFX0 und SFX1 treten in bestimmte Stelle des Programms große maxIdle auf, z.B bei CFX0 2688. In solcher Situation ist das lange Abschalten der Units durch Clock Gating unvermeidbar.

## 4 Simulation

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
1	0	1873	846	254	160	398	5	0	0	3505	987
2	0	587	244	229	36	876	5	0	0	424	187
3	0	465	43	100	32	898	6	0	1	83	162
4	0	308	438	48	25	484	23	0	81	90	454
5	0	31	121	47	42	130	6	0	4	22	242
6	0	115	36	184	44	120	2	0	193	5	253
7	0	93	109	279	21	29	0	0	156	15	56
8	0	2	14	112	76	71	0	0	2	3	86
9	0	0	68	12	6	69	21	0	1	1	61
10	0	0	29	8	9	55	23	0	24	5	203
11	0	0	11	5	21	3	3	0	21	6	5
12	0	0	0	40	3	0	3	0	0	20	5
13	0	0	0	60	4	19	4	0	5	14	31
14	0	0	0	10	3	16	0	0	9	1	1
15	0	0	0	5	7	35	4	0	2	5	0
16	0	0	0	5	38	8	5	0	0	4	2
17	0	0	9	2	38	1	2	0	0	1	0
18	0	0	0	3	31	1	3	0	0	5	0
19	0	0	1	3	6	3	7	0	0	3	5
20	0	0	0	1	2	1	4	0	0	0	0
21	0	0	2	0	2	0	4	0	0	0	0
22	0	0	3	0	0	0	1	0	0	0	1
23	0	0	3	4	1	0	1	0	0	0	0
24	0	0	0	9	6	0	1	0	0	0	0
25	0	0	0	0	10	1	0	0	0	2	3
26	0	0	0	0	2	1	6	0	0	0	5
27	0	0	1	0	1	4	0	0	0	1	0
28	0	0	0	1	3	1	2	0	14	2	4
29	0	0	0	7	2	0	6	0	23	2	0
30	0	0	0	4	3	0	27	0	9	2	1
31	0	0	1	3	3	1	2	0	8	0	0
32	0	0	2	1	3	1	2	0	4	1	1
33	0	0	1	1	1	13	1	0	5	1	0
34	0	0	0	1	8	14	2	0	9	2	2
35	0	0	0	1	12	0	30	0	12	3	2
36	0	0	3	1	24	0	21	0	40	0	3
37	0	0	4	0	2	0	5	0	23	0	2
38	0	0	0	0	1	0	0	0	1	0	0
39	0	0	2	0	1	0	3	0	5	0	0
40	0	0	0	0	0	0	0	0	1	0	0
41	0	0	2	0	1	0	3	0	1	0	0
42	0	0	6	1	1	0	1	0	3	0	0
43	0	0	2	0	0	0	4	0	0	0	0
44	0	0	1	1	2	0	1	0	0	0	0
45	0	0	2	2	1	0	1	0	3	0	1
46	0	0	4	4	1	0	1	0	0	0	0
47	0	0	2	0	2	0	0	0	0	0	0
48	0	0	1	1	0	0	0	0	3	0	1
49	0	0	1	7	1	0	0	0	6	0	1
50	0	0	1	5	3	0	1	0	0	0	1
51	0	0	0	3	3	0	1	0	0	0	0
52	0	0	0	0	0	1	3	0	0	0	2
53	0	0	0	2	3	0	1	0	1	0	0
54	0	0	3	4	6	0	0	0	0	0	2
55	0	0	3	4	7	0	1	0	0	0	1
56	0	0	0	0	4	1	0	0	0	0	4
57	0	0	1	0	1	0	2	0	5	0	0
58	0	0	14	1	1	0	1	0	9	0	1
59	0	0	0	2	6	1	1	0	1	0	8
60	0	0	1	0	0	0	0	0	0	0	0
61	0	0	1	0	1	0	1	0	0	0	0
62	0	0	1	0	0	0	2	0	0	0	0
63	0	0	2	0	1	0	2	0	0	0	0
64	0	0	1	3	4	0	2	0	0	0	0
65	0	0	1	4	0	0	1	0	0	0	0
66	0	0	1	5	2	0	0	0	1	0	0
67	0	0	4	9	0	0	1	0	0	0	0
68	0	0	3	6	4	0	1	0	1	0	0
69	0	0	0	2	3	0	0	0	1	0	0
70	0	0	0	0	1	0	0	0	0	0	0
71	0	0	3	0	5	0	0	0	0	0	0
72	0	0	0	2	9	0	1	0	0	0	0
73	0	0	3	2	2	0	0	0	0	0	0
74	0	0	0	0	3	0	0	0	0	0	0
75	0	0	4	1	0	0	1	0	0	0	0
77	0	0	6	1	1	0	0	0	9	0	0
78	0	0	2	1	0	0	0	0	2	0	0
80	0	0	1	1	2	0	1	0	1	0	0

Tabelle 4.6: DES-Trace-40000-Instructions: maxIdleCycle, Häufigkeit

## 4 Simulation

---

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
81	0	0	0	4	2	0	0	0	2	0	0
82	0	0	5	0	2	0	0	0	0	0	0
83	0	0	1	0	0	0	0	0	0	0	0
84	0	0	0	0	1	0	0	0	0	0	0
86	0	0	1	9	0	0	0	0	0	0	0
87	0	0	0	1	2	0	0	0	0	0	0
89	0	0	0	1	2	0	1	0	2	0	0
90	0	0	2	2	0	0	0	0	0	0	0
91	0	0	0	2	7	0	0	0	0	0	0
92	0	0	1	0	1	0	0	0	1	0	0
93	0	0	0	0	0	0	0	0	1	0	0
94	0	0	1	1	0	0	0	0	0	0	0
95	0	0	1	1	3	0	0	0	0	0	0
96	0	0	0	1	1	0	0	0	1	0	0
97	0	0	0	1	0	0	0	0	1	0	0
98	0	0	0	0	1	0	0	0	0	0	0
99	0	0	0	1	1	0	0	0	0	0	0
101	0	0	0	1	1	0	1	0	1	0	0
102	0	0	0	0	0	0	0	1	0	0	0
104	0	0	0	0	0	0	1	0	0	0	0
105	0	0	0	0	0	0	0	0	2	0	0
106	0	0	0	0	0	0	1	0	1	0	0
107	0	0	0	0	0	0	0	0	1	0	0
109	0	0	0	0	0	0	0	0	3	0	0
110	0	0	0	0	0	0	0	0	3	0	0
111	0	0	0	0	0	0	0	0	2	0	0
112	0	0	0	0	0	0	0	0	1	0	0
114	0	0	0	0	0	0	0	0	3	0	0
115	0	0	0	0	0	0	0	0	3	0	0
116	0	0	0	1	1	0	0	0	0	0	0
117	0	0	0	0	0	0	0	0	1	0	0
118	0	0	0	1	0	0	1	0	1	0	0
119	0	0	0	0	1	0	0	0	1	0	0
122	0	0	0	0	0	0	0	0	1	0	0
124	0	0	0	0	0	0	2	0	4	0	0
125	0	0	0	0	0	0	2	0	0	0	0
128	0	0	0	0	0	0	0	0	8	0	0
129	0	0	0	0	0	0	0	0	1	0	0
131	0	0	0	0	0	0	0	0	1	0	0
132	0	0	0	0	0	0	0	0	3	0	0
133	0	0	0	0	0	0	0	0	2	0	0
134	0	0	0	0	0	0	1	0	0	0	0
135	0	0	0	0	0	0	1	0	0	0	0
136	0	0	0	0	0	0	1	0	0	0	0
137	0	0	0	0	0	0	1	0	1	0	0
138	0	0	0	0	0	0	0	0	1	0	0
142	0	0	0	0	0	0	0	0	1	0	0
143	0	0	0	0	0	0	1	0	0	0	0
144	0	0	0	0	0	0	0	0	1	0	0
145	0	0	0	0	0	0	1	0	0	0	0
146	0	0	0	0	0	0	0	0	1	0	0
147	0	0	0	0	0	0	1	0	0	0	0
148	0	0	0	0	0	0	3	0	1	0	0
150	0	0	0	0	0	0	3	0	1	0	0
160	0	0	0	0	0	0	1	0	0	0	0
161	0	0	0	0	0	0	2	0	0	0	0
162	0	0	0	0	0	0	1	0	0	0	0
164	0	0	0	0	0	0	1	0	1	0	0
171	0	0	0	0	0	0	1	0	0	0	0
173	0	0	0	0	0	0	2	0	0	0	0
174	0	0	0	0	0	0	1	0	0	0	0
185	0	0	0	0	0	0	7	0	0	0	0
192	0	0	0	0	0	0	0	0	1	0	0
196	0	0	0	0	0	0	2	0	0	0	0
197	0	0	0	0	0	0	2	0	0	0	0
205	0	0	0	0	0	0	1	0	0	0	0
209	0	0	0	0	0	0	1	0	0	0	0
231	0	0	0	0	0	0	1	0	0	0	0
232	0	0	0	0	0	0	1	0	0	0	0
234	0	0	0	0	0	0	1	0	0	0	0
2684	0	0	0	0	1	0	0	0	0	0	0
2688	0	0	0	0	0	0	1	0	0	0	0

Tabelle 4.7: DES-Trace-40000-Instructions: maxIdleCycle, Häufigkeit(Fortsetzung)

## 4 Simulation

Insgesamt kann aus den zwei Tabellen konkret entnommen werden, bei welchem maxIdle und bei welcher Functional Unit das Clock Gating nötig und effektiv ist, um Energie zu sparen.

Als Testfall 2 wird die komplette Version von DES-Trace genommen, die insgesamt mehr als 1,8 Millionen Anweisungen enthält. In diesem Testfall wird nur ein Befehl in einer Taktfrequenz ausgegeben. Dieser Testfall wird durchgeführt, um die korrekten und aussagekräftigen Ergebnisse zu erzielen. Die Ergebnisse sind in Tabellen 4.8 bis 4.12 festgehalten.

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
1	0	137620	8326	1966	36	6901	743	0	0	238002	78533
2	0	28832	8295	587	0	5502	4	0	0	13330	30188
3	0	26756	813	46	2	2465	1696	0	0	3405	4554
4	0	4217	893	101	4	2114	44	0	3	6440	7305
5	0	48351	1713	45	3	3810	14	0	2	240	3432
6	0	2410	3963	14	7	4288	15	0	2	3656	5373
7	0	4824	148	300	2	776	34	0	2	297	25406
8	0	183	781	64	4	2745	8	0	0	193	1776
9	0	649	434	35	2	1497	107	0	0	140	1721
10	0	122	1245	27	0	899	18	0	0	54	1271
11	0	132	282	7	1	1014	16	0	2	12	69
12	0	31	140	3	0	84	8	0	7	159	496
13	0	433	36	3	1	160	24	0	642	13	75
14	0	166	653	236	28	1165	63	0	1	43	92
15	0	33	129	60	0	254	335	0	322	75	120
16	0	1874	310	6	1	285	60	0	1	16	471
17	0	7	1798	5	1	37	248	0	321	14	1584
18	0	1546	289	7	0	47	21	0	0	16	280
19	0	51	67	3	1	1595	19	0	0	12	15
20	0	4	15	267	4	3	6	0	263	137	1586
21	0	0	73	1	1	4	332	0	0	4	21689
22	0	1	15	8	2	36	3	0	0	7	414
23	0	0	1734	7	263	345	21	0	0	5	53
24	0	2	131	1559	52	296	8	0	1	27	117
25	0	12	317	8	2	1	6	0	1	41	5
26	0	5	31	12	2	1	3	0	0	298	3
27	0	0	123	4	1	1544	2	0	0	12	36
28	0	0	2164	6	1	5	5	0	16	326	39
29	0	0	67	4	1542	0	9	0	57	3	12
30	0	2	1612	6	0	1	4	0	4	16	33
31	0	6	35	7	1	1	5	0	1	12	21
32	0	25	129	56	3	0	4	0	1	5	17
33	0	0	15	33	0	2	3	0	8	12	10
34	0	0	7	5	2	0	2	0	3	5	48
35	0	0	58	32	2	0	2	0	14	0	4
36	0	0	18533	4	3	0	0	0	0	12	35
37	0	0	1	1	1	0	4	0	0	274	16
38	0	1544	15	6	2	0	3	0	1	4	12
39	0	0	31	45	5	0	15	0	0	71	49
40	0	0	2	30	9	0	3	0	0	54	50
41	0	0	13	7	0	0	2	0	1	0	9
42	0	0	6	6	0	0	3	0	0	0	22
43	0	1544	1544	3	10	0	2	0	1	0	6
44	0	0	0	16	7	0	77	0	2	0	6
45	0	0	4	42	2	0	16	0	7	0	8
46	0	0	1	4	1	0	33	0	0	0	11
47	0	0	6	11	0	0	36	0	0	0	37
48	0	0	3	4	3	1544	1	0	1	0	25
49	0	0	1552	10	2	0	7	0	0	0	80
50	0	0	2	1	1	0	3	0	0	0	57
51	0	0	6	169	6	0	0	0	1	0	8
52	0	0	0	17	5	0	30	0	1	0	18
53	0	2	1544	10	7	0	5	0	1	0	22
54	0	0	1	27	2	24	3	0	2	0	78
55	0	0	2	8	29	2	16	0	0	0	22
56	0	0	2	1	0	6	2	0	0	0	2
57	0	0	15	5	2	0	9	0	50	0	20
58	0	0	0	19	0	0	15	0	47	0	7
59	0	0	1	18	1	0	1	0	162	0	34
60	0	0	0	3	2	0	1	0	0	0	1
61	0	0	0	12	1	0	0	0	46	0	1
62	0	0	0	64	0	0	5	0	3	0	15
63	0	0	0	10	7	0	3	0	2	0	4
64	0	0	0	11	1	0	0	0	0	0	11
65	0	0	0	5	0	0	7	0	3	0	37

Tabelle 4.8: DES-Trace-Complete: maxIdleCycle, Häufigkeit

## 4 Simulation

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
66	0	0	15	10	0	0	2	0	5	0	6
67	0	0	11	3	0	0	0	0	14	0	12
68	0	0	1	32	2	0	0	0	0	0	73
69	0	0	0	45	1	0	0	0	0	0	29
70	0	0	0	10	0	0	5	0	1	0	194
71	0	0	0	4	0	0	1	0	16	0	301
72	0	0	0	6	5	0	17	0	0	0	3
73	0	0	0	6	1	0	0	0	11	0	1
74	0	0	0	7	1	0	69	0	5	0	16
75	0	0	19	77	0	0	31	0	101	0	5
76	0	0	0	31	0	0	40	0	0	0	2
77	0	0	0	37	0	0	197	0	1	0	7
78	0	0	0	57	1	0	46	0	0	0	4
79	0	0	0	4	3	1544	14	0	1	0	4
80	0	0	0	16	0	0	4	0	16	0	4
81	0	0	0	3	1	0	3	0	10	0	1
82	0	0	0	3	3	0	8	0	1	0	4
83	0	0	0	11	0	0	15	0	2	0	1
84	0	0	0	5	0	2	25	0	2	0	180
85	0	0	13	12	5	0	0	0	5	0	2
86	0	0	0	3	7	0	2	0	0	0	2
87	0	0	0	9	4	0	0	0	0	0	2
88	0	0	0	2	0	0	24	0	6	0	0
89	0	0	0	7	12	0	5	0	0	0	4
90	0	0	0	7	1	0	2	0	1	0	0
91	0	0	0	0	1	0	3	0	1	0	0
92	0	0	0	2	1	0	4	0	0	0	1
93	0	0	11	1	0	0	4	0	2	0	1
94	0	0	0	12	2	0	5	0	2	0	1
95	0	0	0	4	2	0	2	0	2	0	0
96	0	0	0	6	1	0	5	0	21	0	9
97	0	0	0	1	1	0	8	0	31	0	0
98	0	0	0	0	2	0	0	0	11	0	0
99	0	0	0	1	0	0	3	0	5	0	0
100	0	0	0	55	2	0	16	0	0	0	1
101	0	0	0	196	10	0	3	0	3	0	0
102	0	0	18	6	15	0	3	0	1	0	0
103	0	0	0	27	0	0	7	0	0	0	0
104	0	0	0	6	1	0	3	0	2	0	0
105	0	0	0	3	0	0	2	0	9	0	0
106	0	0	0	1	1	0	14	0	8	0	3
107	0	0	0	1	0	0	0	0	2	0	0
108	0	0	0	22	1	0	3	0	12	0	0
109	0	0	0	133	0	0	3	0	6	0	1
110	0	0	0	3	2	0	1	0	4	0	1
111	0	0	17	72	6	0	1	0	73	0	1
112	0	0	0	15	1	0	1	0	18	0	0
113	0	0	0	4	17	0	1	0	3	0	0
114	0	0	0	7	1	0	0	0	4	0	0
115	0	0	0	2	0	0	10	0	2	0	0
116	0	0	0	2	1	0	0	0	0	0	2
117	0	0	0	10	0	0	13	0	3	0	2
118	0	0	0	0	1	0	0	0	2	0	3
119	0	0	0	3	1	0	0	0	6	0	0
120	0	0	0	4	3	0	0	0	0	0	0
121	0	1	29	48	3	0	1	0	3	0	0
122	0	0	0	4	0	0	1	0	3	0	1
123	0	0	1	4	0	0	1	0	5	0	0
124	0	0	0	2	1	0	24	0	1	0	0
125	0	0	1	6	31	0	0	0	0	0	1
126	0	0	0	2	3	0	1	0	3	0	0
127	0	0	0	0	0	0	1	0	16	0	0
128	0	0	0	5	0	0	3	0	3	0	0
129	0	0	20	4	3	0	2	0	1	0	0
130	0	0	0	22	3	0	1	0	1	0	1
131	0	0	0	2	0	0	69	0	0	0	0
132	0	0	0	2	0	0	1	0	1	0	1
133	0	0	0	0	3	0	13	0	2	0	0
134	0	0	0	1	0	0	0	0	0	0	1
135	0	0	0	1	0	0	5	0	6	0	1
136	0	0	1	0	1	0	1	0	2	0	0
137	0	0	0	2	1	0	0	0	14	0	0
138	0	0	16	1	3	0	0	0	0	0	1
139	0	0	0	18	0	0	2	0	0	0	0
140	0	0	0	10	1	0	2	0	4	0	0
141	0	0	0	0	2	0	10	0	0	0	0
142	0	0	0	11	6	0	11	0	18	0	0
143	0	0	0	3	0	0	9	0	3	0	0
144	0	0	0	0	9	0	1	0	2	0	0
145	0	0	0	2	3	0	1	0	5	0	0
146	0	0	0	1	0	0	0	0	0	0	0
147	0	0	19	3	8	0	1	0	0	0	0
148	0	0	0	21	0	0	3	0	9	0	0
149	0	0	0	5	0	0	1	0	4	0	0
150	0	0	0	0	0	0	0	0	33	0	1
151	0	0	0	0	0	0	20	0	14	0	0
152	0	0	0	4	1	0	0	0	1	0	0
153	0	0	0	5	1	0	6	0	9	0	0
154	0	0	0	0	4	0	0	0	1	0	0
155	0	0	0	6	0	0	0	0	0	0	0
156	0	0	0	2	0	0	5	0	0	0	0
157	0	0	15	16	0	0	1	0	1	0	1
158	0	0	0	1	3	0	1	0	13	0	0
159	0	0	0	1	2	0	0	0	4	0	0

Tabelle 4.9: DES-Trace-Complete: maxIdleCycle, Häufigkeit (Fortsetzung)

## 4 Simulation

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
160	0	0	0	4	16	0	14	0	5	0	0
161	0	0	0	4	15	0	0	0	4	0	0
162	0	0	0	0	0	0	0	0	4	0	0
163	0	0	0	0	0	0	2	0	0	0	0
164	0	0	0	2	2	0	2	0	4	0	0
165	0	14	1	2	0	0	0	2	0	1	0
166	0	0	9	0	0	0	0	0	8	0	0
167	0	0	1	0	0	1	0	1	1	0	0
168	0	0	9	3	0	0	0	1	0	0	0
169	0	0	1	0	0	29	0	13	0	0	0
170	0	0	0	0	0	32	0	1	0	0	0
171	0	0	0	0	0	1	0	7	0	0	0
172	0	0	0	0	5	0	0	1	0	0	0
173	0	0	0	0	1	0	0	0	0	0	0
174	0	7	4	6	0	1	0	11	0	0	0
175	0	0	5	1	0	0	0	0	0	0	0
176	0	0	2	6	0	1	0	1	0	0	0
177	0	0	2	0	0	2	0	1	0	0	0
178	0	0	0	0	0	31	0	11	0	0	0
179	0	0	3	0	0	2	0	0	0	0	0
180	0	0	2	0	0	3	0	5	0	0	0
181	0	0	2	6	0	1	0	2	0	0	0
182	0	1	1	7	0	3	0	3	0	1	0
183	0	11	0	8	0	1	0	1	0	0	0
184	0	0	8	12	0	3	0	1	0	0	0
185	0	0	1	1	0	0	0	0	0	0	0
186	0	0	3	1	0	2	0	2	0	0	0
187	0	0	0	3	0	17	0	18	0	0	0
188	0	0	1	0	0	0	0	4	0	0	0
189	0	0	3	4	0	0	0	14	0	0	0
190	0	0	0	2	0	1	0	4	0	0	0
191	0	0	0	1	0	0	0	1	0	0	0
192	0	0	1	1	0	0	0	1	0	0	0
193	0	11	6	0	0	0	0	3	0	0	0
194	0	0	0	1	0	9	0	1	0	0	0
195	0	0	8	6	0	3	0	0	0	0	0
196	0	0	0	1	0	21	0	13	0	0	0
197	0	0	1	9	0	0	0	2	0	0	0
198	0	0	0	3	0	0	0	3	0	0	0
199	0	0	0	6	0	0	0	0	0	0	0
200	0	0	2	0	0	0	0	5	0	0	0
201	0	2	0	2	0	0	0	1	0	0	0
202	0	0	2	1	0	2	0	0	0	0	0
203	0	0	0	1	0	1	0	3	0	0	0
204	0	0	0	0	2	0	0	0	0	0	0
205	0	0	0	0	10	0	1	15	0	0	0
206	0	0	0	3	0	18	0	0	0	0	0
207	0	0	1	1	0	0	0	3	0	0	0
208	0	0	0	3	0	4	0	1	0	0	0
209	0	0	0	6	0	9	0	0	0	0	0
210	0	1	0	4	0	0	0	0	0	0	0
211	0	0	2	0	0	1	0	0	0	0	0
212	0	0	0	0	0	0	0	2	0	0	0
213	0	0	0	0	0	1	0	7	0	0	0
214	0	0	0	2	0	1	0	13	0	0	0
215	0	0	1	4	0	0	0	1	0	0	0
216	0	0	0	0	0	0	0	1	0	0	0
217	0	0	2	2	0	8	0	2	0	0	0
218	0	0	0	6	0	10	0	1	0	0	0
219	0	0	0	3	0	0	0	2	0	0	0
220	0	0	1	1	0	3	0	0	0	0	0
221	0	0	0	0	0	1	0	0	0	0	0
222	0	1	0	1	0	0	0	1	0	0	0
223	0	0	0	0	0	1	0	15	0	0	0
224	0	0	0	3	0	0	0	1	0	0	0
225	0	0	0	3	0	2	0	0	0	0	0
226	0	0	0	3	0	0	0	0	0	0	0
227	0	0	0	2	0	2	0	1	0	0	0
228	0	0	3	3	0	1	0	5	0	0	0
229	0	2	2	1	0	3	0	0	0	0	0
230	0	0	0	1	0	5	0	0	0	0	0
231	0	0	1	3	0	0	0	6	0	0	0
232	0	0	0	1	0	1	0	28	0	0	0
233	0	0	1	1	0	1	0	1	0	0	0
234	0	0	2	1	0	1	0	1	0	0	0
235	0	0	1	5	0	0	0	0	0	0	0
236	0	0	0	0	42	0	1	2	0	0	0
237	0	1	0	6	0	0	0	0	0	0	0
238	0	0	2	8	0	0	0	0	0	0	0
239	0	0	0	0	0	1	0	2	0	0	0
240	0	0	0	3	0	0	0	4	0	0	0
241	0	0	0	0	0	3	0	15	0	0	0
242	0	0	1	2	0	8	0	0	0	0	0
243	0	0	0	0	0	0	0	2	0	0	0
244	0	0	0	2	0	1	0	2	0	0	0
245	0	0	0	3	0	1	0	1	0	0	0
246	0	3	0	2	0	1	0	1	0	0	0
247	0	0	3	7	0	1	0	0	0	0	0
248	0	0	0	3	0	5	0	1	0	0	0
249	0	0	0	0	0	0	0	3	0	0	0
250	0	0	0	0	0	0	0	12	0	0	0
251	0	0	1	0	0	4	0	0	0	0	0

Tabelle 4.10: DES-Trace-Complete: maxIdleCycle, Häufigkeit(Fortsetzung)

## 4 Simulation

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
252	0	0	0	0	1	0	2	0	1	0	0
253	0	0	0	0	6	0	6	0	0	0	0
254	0	0	0	0	1	0	7	0	1	0	0
255	0	0	0	1	5	0	0	0	1	0	0
256	0	0	0	0	5	0	2	0	2	0	0
257	0	0	0	0	2	0	0	0	0	0	0
258	0	0	0	1	2	0	0	0	2	0	1
259	0	0	0	0	3	0	2	0	18	0	0
260	0	0	0	0	0	0	0	0	2	0	0
261	0	0	0	0	4	0	0	0	0	0	0
262	0	0	0	0	3	0	0	0	1	0	0
263	0	0	0	0	1	0	0	0	0	0	0
264	0	0	0	0	0	0	1	0	0	0	0
265	0	0	1	0	1	0	1	0	0	0	0
266	0	0	0	0	7	0	2	0	0	0	0
267	0	0	0	2	3	0	0	0	3	0	0
268	0	0	0	0	1	0	1	0	14	0	0
269	0	0	0	0	1	0	1	0	0	0	0
270	0	0	0	0	1	0	0	0	1	0	0
271	0	0	0	0	3	0	0	0	0	0	0
273	0	0	0	0	8	0	1	0	0	0	0
274	0	0	0	0	1	0	1	0	0	0	0
275	0	0	0	0	16	0	0	0	0	0	0
276	0	0	0	0	3	0	0	0	2	0	0
277	0	0	0	0	2	0	1	0	12	0	0
278	0	0	0	0	0	0	2	0	0	0	0
280	0	0	0	0	6	0	0	0	0	0	0
281	0	0	0	0	2	0	0	0	0	0	0
282	0	0	0	0	1	0	1	0	0	0	0
283	0	0	0	0	9	0	0	0	1	0	0
284	0	0	0	0	9	0	0	0	0	0	0
285	0	0	0	0	1	0	0	0	0	0	0
286	0	0	0	0	1	0	0	0	5	0	0
287	0	0	0	1	3	0	0	0	0	0	0
289	0	0	0	0	5	0	0	0	0	0	0
290	0	0	0	0	3	0	0	0	1	0	0
291	0	0	0	0	3	0	0	0	2	0	0
292	0	0	0	0	3	0	0	0	1	0	0
294	0	0	0	0	6	0	0	0	3	0	0
295	0	0	0	1	1	0	0	0	9	0	0
297	0	0	0	0	3	0	0	0	0	0	0
298	0	0	0	1	4	0	0	0	0	0	0
299	0	0	0	0	5	0	0	0	0	0	0
300	0	0	0	1	2	0	0	0	0	0	0
301	0	0	0	0	4	0	1	0	1	0	0
302	0	0	0	0	2	0	2	0	1	0	0
303	0	0	0	0	3	0	0	0	0	0	0
304	0	0	0	0	3	0	0	0	11	0	0
305	0	0	0	0	2	0	0	0	0	0	0
306	0	0	0	0	5	0	0	0	2	0	0
307	0	0	0	0	4	0	0	0	0	0	0
308	0	0	0	0	1	0	0	0	0	0	0
309	0	0	0	0	1	0	0	0	0	0	0
310	0	0	0	0	1	0	0	0	0	0	0
311	0	0	0	0	1	0	2	0	0	0	0
312	0	0	0	0	1	0	1	0	0	0	0
313	0	0	0	0	5	0	0	0	2	0	0
314	0	0	0	0	6	0	1	0	0	0	0
315	0	0	0	0	1	0	0	0	0	0	0
316	0	0	0	0	2	0	0	0	0	0	0
317	0	0	0	0	2	0	0	0	0	0	0
318	0	0	0	0	1	0	0	0	0	0	0
319	0	0	0	0	2	0	0	0	0	0	0
322	0	0	0	0	1	0	1	0	2	0	0
324	0	0	0	0	0	0	2	0	0	0	0
325	0	0	0	0	1	0	0	0	0	0	0
326	0	0	0	0	3	0	5	0	0	0	0
327	0	0	0	0	1	0	0	0	0	0	0
328	0	0	0	0	2	0	0	0	1	0	0
330	0	0	0	0	1	0	0	0	0	0	0
331	0	0	0	0	0	0	0	0	1	0	0
332	0	0	0	0	1	0	0	0	0	0	0
333	0	0	0	0	3	0	0	0	0	0	0
336	0	0	0	0	2	0	0	0	0	0	0
337	0	0	0	0	1	0	0	0	0	0	0
338	0	0	0	0	1	0	0	0	0	0	0
339	0	0	0	0	0	0	0	0	1	0	0
340	0	0	0	0	0	0	0	0	1	0	0
341	0	0	0	0	2	0	0	0	0	0	0
342	0	0	0	0	1	0	1	0	0	0	0
344	0	0	0	0	1	0	0	0	0	0	0
345	0	0	0	0	1	0	0	0	0	0	0
346	0	0	0	0	1	0	0	0	0	0	0
347	0	0	0	0	1	0	0	0	1	0	0
348	0	0	0	0	1	0	1	0	0	0	1
349	0	0	0	0	1	0	1	0	1	0	0
351	0	0	0	0	1	0	1	0	0	0	0
352	0	0	0	0	2	0	0	0	1	0	0
353	0	0	0	0	4	0	0	0	0	0	1
354	0	0	0	0	3	0	0	0	0	0	0
356	0	0	0	0	5	0	0	0	0	0	0
357	0	0	0	0	2	0	0	0	0	0	0
358	0	0	0	0	2	0	0	0	3	0	0
360	0	0	0	0	1	0	0	0	0	0	0

Tabelle 4.11: DES-Trace-Complete: maxIdleCycle, Häufigkeit(Fortsetzung)

## 4 Simulation

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
361	0	0	0	0	1	0	0	0	0	0	0
362	0	0	0	0	1	0	0	0	1	0	0
364	0	0	0	0	2	0	0	0	0	0	0
366	0	0	0	0	1	0	0	0	0	0	0
367	0	0	0	0	1	0	0	0	0	0	0
369	0	0	0	0	1	0	0	0	0	0	0
371	0	0	0	0	1	0	0	0	0	0	0
373	0	0	0	0	1	0	0	0	0	0	0
374	0	0	0	0	1	0	0	0	0	0	0
375	0	0	0	0	2	0	0	0	0	0	0
376	0	0	0	0	0	0	0	0	1	0	0
378	0	0	0	0	1	0	0	0	0	0	0
379	0	0	0	0	1	0	0	0	0	0	0
380	0	0	0	0	1	0	0	0	1	0	0
381	0	0	0	0	1	0	0	0	0	0	0
382	0	0	0	0	0	0	1	0	0	0	0
383	0	0	0	0	2	0	0	0	0	0	0
386	0	0	0	0	2	0	0	0	0	0	0
387	0	0	0	0	1	0	0	0	0	0	0
389	0	0	0	0	1	0	0	0	0	0	0
392	0	0	0	0	3	0	0	0	0	0	0
394	0	0	0	0	3	0	0	0	0	0	0
398	0	0	0	0	1	0	0	0	0	0	0
400	0	0	0	0	0	0	1	0	0	0	0
402	0	0	0	0	1	0	0	0	0	0	0
403	0	0	0	0	1	0	0	0	0	0	0
407	0	0	0	0	1	0	0	0	0	0	0
410	0	0	0	0	1	0	0	0	0	0	0
415	0	0	0	0	1	0	0	0	0	0	0
416	0	0	0	0	1	0	0	0	0	0	0
417	0	0	0	0	1	0	0	0	0	0	0
427	0	0	0	0	1	0	0	0	0	0	0
428	0	0	0	0	1	0	0	0	0	0	0
431	0	0	0	0	3	0	0	0	0	0	0
436	0	0	0	0	1	0	0	0	0	0	0
445	0	0	0	0	1	0	0	0	0	0	0
454	0	0	0	0	1	0	0	0	0	0	0
455	0	0	0	0	1	0	0	0	0	0	0
458	0	0	0	0	2	0	0	0	0	0	0
459	0	0	0	0	1	0	0	0	0	0	0
463	0	0	0	0	1	0	0	0	0	0	0
466	0	0	0	0	1	0	0	0	0	0	0
467	0	0	0	0	1	0	0	0	0	0	0
469	0	0	0	0	3	0	0	0	0	0	0
473	0	0	0	0	1	0	0	0	0	0	0
481	0	0	0	0	1	0	0	0	0	0	0
491	0	0	0	0	0	0	0	0	1	0	0
496	0	0	0	0	1	0	0	0	0	0	0
499	0	0	0	0	1	0	0	0	0	0	0
508	0	0	0	0	2	0	0	0	0	0	0
519	0	0	0	0	1	0	0	0	0	0	0
520	0	0	0	0	1	0	0	0	0	0	0
526	0	0	0	0	1	0	0	0	0	0	0
536	0	0	0	0	1	0	0	0	0	0	0
550	0	0	0	0	0	0	0	0	0	0	1
567	0	0	0	1	0	0	0	0	0	0	0
568	0	0	0	0	1	0	0	0	0	0	0
569	0	0	0	0	1	0	0	0	0	0	0
576	0	0	0	0	0	1544	0	0	0	0	0
584	0	0	0	0	1	0	0	0	0	0	0
589	0	0	0	0	1	0	0	0	0	0	0
592	0	0	0	0	1	0	0	0	0	0	0
605	0	0	0	0	1	0	0	0	0	0	0
606	0	0	0	0	0	0	1	0	0	0	0
632	0	0	0	0	1	0	0	0	0	0	0
650	0	0	0	0	1	0	0	0	0	0	0
651	0	0	0	0	1	0	0	0	0	0	0
662	0	0	0	0	2	0	0	0	0	0	0
700	0	0	0	0	1	0	0	0	0	0	0
709	0	0	0	0	1	0	0	0	0	0	0
728	0	0	0	1542	1542	0	0	0	0	0	0
730	0	0	0	1	0	0	0	0	0	0	0
731	0	0	0	1	1	0	1	0	0	0	0
732	0	0	0	0	1	0	1	0	0	0	0
746	0	0	0	0	0	0	0	1	0	0	0
756	0	0	0	0	0	0	1542	0	0	0	0
765	0	0	0	0	0	0	1	0	0	0	0
772	0	0	0	0	1	0	0	0	0	0	0
783	0	0	0	0	0	0	0	0	1	0	0
854	0	0	0	1	0	0	0	0	0	0	0
864	0	0	0	0	1	0	0	0	0	0	0
1004	0	0	0	0	1	0	0	0	0	0	0
1009	0	0	0	0	1	0	0	0	0	0	0
1034	0	0	0	0	0	0	2	0	0	0	0
1088	0	0	0	0	0	0	0	0	1	0	0
1124	0	0	0	0	1	0	0	0	0	0	0
1144	0	0	0	0	1	0	0	0	0	0	0
1156	0	0	0	0	1	0	0	0	0	0	0
1523	0	0	0	0	0	0	0	0	0	0	1
1528	1	0	0	0	0	0	0	0	0	0	0
1532	0	1	1	1	1	0	0	0	0	0	0
1672	0	0	0	0	0	0	0	0	1	0	0
1706	0	0	0	0	0	0	1	0	0	0	0
1919	0	0	0	0	0	0	0	0	1	0	0
2036	0	0	0	0	0	0	0	0	0	1	0

Tabelle 4.12: DES-Trace-Complete: maxIdleCycle, Häufigkeit(Fortsetzung)

Die Ergebnisse weisen folgende Eigenschaften auf:

- Die Units 'FPU0' und 'unbek' sind wie bei Test 1 unbesetzt und können während der ganzen Ausführung abgeschaltet werden.
- Analog zum Test 1 sind maxIdle von einem Zyklus bei den meisten Units außer FXU0 häufig vorgekommen. In diesem Fall ist das Abschalten der Units durch Clock Gating nicht realisierbar. Danach folgt die Häufigkeit von maxIdle ab 2 Zyklen. Hier ist möglich, Clock Gating einzusetzen.
- Bei den Units LSU1, LSU2, LSU3, BPU0, FXU0, SFX1 treten maximale Idles von mehr als 10 Zyklen häufiger als bei Test 1 auf. In diesem Fall ist das lange Abschalten der Units durch Clock Gating sehr nötig.

Insgesamt stellen diese Ergebnisse Eigenschaften dar, die zum größten Teil Gemeinsamkeit mit den Ergebnissen von Test 1 haben.

Um die Stabilität und Robustheit der Simulation-Programme zu testen, wird noch ein weiteres Testprogramm ausgeführt, nämlich `gzip.source`, die ca 30 Millionen Anweisungen enthält. Dabei wird in einer Taktfrequenz nur ein neuer Befehl geholt, um den Rückstau bei der Abhängigkeitskette zu vermeiden. Der Test verläuft erfolgreich und die Ergebnisse werden im Anhang B festgehalten.

## 5 Zusammenfassung und Ausblick

Clock Gating ist eine effektive Power Reduktion Technik auf der Register Transfer Ebene. Sie minimiert den Energiekonsum durch gezieltes Abschalten des Clock Signals einer Schaltung. Damit wird das Laden/Entladen der Schaltungskapazitaet während der Idle Cycles verhindert. Mit Hilfe des simulierten Activity Patterns einzelner Synchronen Komponente in einem Synchronem Clock System kann sogar ein Sektion dieses Systems, in dem die Komponenten ähnliche Activity Patterns haben, abgeschaltet werden.

Als Beispiele für die Anwendung von Clock Gating werden Deterministic Clock Gating und Value-Based Clock Gating diskutiert.

Das Deterministic Clock Gating ist eine Methode, bei der den Energiekonsum von Integer- und FP-Execution Unit während der Idle Cycles gespart werden kann. Hier ist auch sehr entscheidend, dass bei superskalaren Prozessoren die Anzahl von Integer Execution Units gleichen Typs unter Berücksichtigung von Energiekonsum optimal ausgewählt wird. Sind mehrere Execution Units gleichen Typs vorhanden und nicht immer genügend Befehle zur parallelen Ausführung vorhanden, ist das Prioritätsprinzip für die Einstufung von Execution Units notwendig. Durch dieses Prioritätsprinzip werden die Execution Units mit höherer Priorität immer besetzt, während die mit der niedrigeren für eine lange Zeit abgeschaltet oder sogar in Schlafmodus versetzt werden können. Damit wird nicht nur die dynamische Leistung der Execution Units, sondern auch die Leckleistung abgesenkt. Ein anderes Beispiel ist die Value-Based Clock Gating Methode. Im Unterschied zum Deterministic Clock Gating wird hier den Energiekonsum durch Abschalten der ersten 48-bit Teil von 64-bit Integer Executions verringert, wenn die ersten 48-bit der Operanden null sind, nämlich mit Hilfe eines Hardware-Mechanismus.

Um die Belegung von Functional Units und das daraus resultierende Sparpotential zu demonstrieren, wurde eine Simulation durchgeführt. Diese Simulation wurde so gestaltet, dass die Konfigurationsdaten architekturunabhängig ist. Als Testprogramm wurde zuerst ein Trace von DES, die knap mehr als 40000 Anweisungen umfasst, ausgewählt. Danach erfolgte die Ausführung der kompletten Version von DES-Trace, die mehr als 1,8 Millionen Anweisungen enthält, um die Korrektheit und Stabilität der Simulationsprogramme zu testen. Die beiden Testfälle verliefen erfolgreich und lieferten aussagekräftige Ergebnisse. Aus den Ergebnissen kann entnommen werden, dass bei DES-Trace der Energiekonsum durch Functional Units, besonders bei High Performance Execution Units, gewaltig gespart werden kann. Abgesehen davon, dass die Floating Point Unit die ganze

Zeit abgeschaltet oder in Schlafmodus versetzt werden kann, können auch die anderen Execution Units öfter abgeschaltet werden. Dabei muss beachtet werden, dass sie mindestens 2 oder 3 Zyklen nicht benötigt werden, damit das Ein- und Ausschalten nicht so oft vorkommt, was auch wieder viel Energie konsumiert. Leider sind auch maximale Idle von einem Zyklus bei den meisten häufig, was das Sparen von Energie durch Clock Gating nicht realisierbar ist.

Um Clock Gating Logic zu implementieren, werden Gatter und möglicherweise Latches, im Fall Latch-Based Clock Gating, zur Speicherung von Kontrollsignalen benötigt, die wiederum auch viel Energie konsumieren. Clock Gating Logic an vielen Stellen in einem Clock System ist zwar möglich, aber es bedarf einer Strategie und einer Optimierung für die Auswahl der Anzahl von Clock Gating und die zu besetzenden Stellen, damit die Stromvergeudung umgangen werden kann und auch funktionelle Fehler, z. B. Abschalten einer Schaltung während der Beschäftigungszeit, zu vermeiden.

### **Verbesserungsmöglichkeiten**

Da die Implementierung der Simulation sowohl komplizierter als auch aufwändiger als geplant war, und zwar in Bezug auf den Umfang und auf den Zeitraum der Studienarbeit, sind noch weitere Optimierungen möglich, um die Arbeitsbedingungen sehr praxisnahe zu simulieren z. B.:

- Structural Hazard soll noch berücksichtigt werden,
- jede Functional Unit soll auf einem eigenständigen Programmfragment, nämlich Thread, laufen können,
- Load/Store Units LSU0, LSU1, LSU2, LSU3 sollen in pipeline laufen.

### **Ausblick**

In Bezug auf die Reduzierung des Energiekonsums durch Clock Gating bei Functional Units sind noch folgende Erweiterungen denkbar:

- Minimieren der Häufigkeit des maxIdle von einem Zyklus
- Reduzierung der Leckleistung.

# Literaturverzeichnis

- [1] John L.Hennesy, David A. Patterson. Computer Architecture, A Quantitative Approach. Morgan Kaufmann Publishers, 3. edition 2003
- [2] Frank Emmett, Mark Biegel. Power Reduction Through RTL Clock Gating, Automotive Integrated Electronics Corporation, SNUG San Jose 2000
- [3] Power Compiler Reference Manual, Synopsys, Release 2002.05, May 2002
- [4] John L.Hennesy, David A. Patterson. Computer Organization and Design, The Hardware/Software Interface, Morgan Kaufmann Publishers, second edition 1996
- [5] Eby G. Friedman. Clock Distribution Networks in Synchronous Digital Integrated Circuits, Proceedings of the IEEE , vol. 89. No.5, May 2001
- [6] Amir H. Farrahi, Chunhong Chen, Ankur Srivastava, Gustavo Tellez, and Majid Sarrafzadn. Activity-Driven Clock Design,IEEE Transactions on Computer-Aided Design of Intergrated Circuits and Systems, vol. 20, No. 6 June 2001
- [7] Sean Stetson. Low Jitter Clock Distribution Networks, Dissertation Proposal, The University of Michigan, July 17, 1997
- [8] Hai Li, Swarup Bhunia, Yiran Chen, T. N. Vijaykumar, and Kaushik Roy. Deterministic Clock Gating for Microprocessor Power Reduction, 1285 EE Building, ECE Department, Perdue University <hl. bhunias, yc, viyay, kaushik>@ecn.purdue.edu
- [9] David Brooks and Margaret Martonosi. Value-Based Clock Gating and Operation Packing: Dynamic Strategies for Improving Processor Power and Performance, ACM Transactions on Computer Systems, Vol. 18, No. 2, May 2000, Pages 89-126
- [10] IBM, The PowerPC Compiler Writer's Guide, International Business Machines Corporation 1996. 1-96 Printed in the United State of America.
- [11] IBM, PowerPC Microprocessor FamilyThe Programming Environments for 32-Bit Microprocessors, 02/21/2000

# A Instruction Set Listing

```

# Zuordnung Instruction-Functional Unit anhand PowerPC 604
# Latency anhand PowerPC 604 festgelegt

# Integer Arithmetic Instructions
add      |      SFX      |      1
add.     |      SFX      |      1
addc     |      SFX      |      1
adde     |      SFX      |      1
adde.   |      SFX      |      1
addi     |      SFX      |      1
addic    |      SFX      |      1
addic.  |      SFX      |      1
addis   |      SFX      |      1
addme   |      SFX      |      1
addze   |      SFX      |      1
divw    |      CFX      |     21
divwu   |      CFX      |     21
mulhw   |      CFX      |      5
mulhwu  |      CFX      |      5
mulli   |      CFX      |      3
mullw   |      CFX      |      5
mullw.  |      CFX      |      5
neg     |      SFX      |      1
subf    |      SFX      |      1
subfc   |      SFX      |      1
subfic  |      SFX      |      1
subfe   |      SFX      |      1
subfme  |      SFX      |      1
subfze  |      SFX      |      1

# Integer Compare Instructions
cmp     |      SFX      |      1
cmpi   |      SFX      |      1
cmpl   |      SFX      |      1
cmpli  |      SFX      |      1

# Integer Logical Instructions
and     |      SFX      |      2
and.   |      SFX      |      2
andc   |      FXU      |      2
andc.  |      FXU      |      2
andi   |      FXU      |      2
andis. |      FXU      |      2
cntlzw|      SFX      |      2
eqv    |      SFX      |      2

# Integer Logical Instructions (Fortsetzung)
extsb  |      SFX      |      1
extsb. |      SFX      |      1
extsh  |      SFX      |      1
extsh. |      SFX      |      1
nand   |      SFX      |      2
nor    |      SFX      |      2
nor.   |      SFX      |      2
or     |      SFX      |      2
or.    |      SFX      |      2
orc    |      SFX      |      2
ori    |      SFX      |      2
oris   |      SFX      |      2
xor    |      SFX      |      2
xor.   |      SFX      |      2
xori   |      SFX      |      2
xoris  |      SFX      |      2

```

Tabelle A0: Instruction Set Listing

## A Instruction Set Listing

---

# Integer Rotate Instructions			
rlwmi		SFX	2
rlwinm		SFX	1
rlwinm.		SFX	1
rlwnm		SFX	1
# Integer Shift Instructions			
slw		SFX	2
sraw		SFX	1
srawl		SFX	1
srw		SFX	1
# Floating-Point Arithmetic Instructions			
fadd		FPU	4
fadds		FPU	4
fdiv		FPU	33
fdivs		FPU	19
fmul		FPU	6
fmuls		FPU	6
fres		FPU	19
frsqrte		FPU	4
fsub		FPU	3
fsubs		FPU	3
fsel		FPU	4
fsqrt		FPU	4
fsqrts		FPU	4
# Floating-Point Multiply-Add Instructions			
fmadd		FPU	4
fmadds		FPU	4
fmsub		FPU	4
fmsubs		FPU	4
fnmadd		FPU	4
fnmadds		FPU	4
# Floating-Point Rounding and Conversion Instructions			
fctiw		FPU	4
fctiwz		FPU	3
frsp		FPU	3
# Floating-Point Compare Instructions			
fcmpo		FPU	3
fcmpu		FPU	3
# Floating-Point Status and Control Register Instructions			
mcrfs		FPU	3
mffs		FPU	4
mtfsb0		FPU	3
mtfsb1		FPU	3
mtfsf		FPU	3
mtfsfi		FPU	3
# Integer Load Instructions			
lbz		LSU	2
lbzu		LSU	2
lbzux		LSU	2
lbzx		LSU	2
lha		LSU	2
lhau		LSU	2
lhaux		LSU	2
lhax		LSU	2
lhz		LSU	2
lhzu		LSU	2
lhzux		LSU	2
lhzx		LSU	2
lwz		LSU	2
lwzu		LSU	2
lwzux		LSU	2
lwzx		LSU	2
# Integer Store Instructions			
stb		LSU	3
stbu		LSU	3
stbux		LSU	3
stbx		LSU	3
sth		LSU	3
sthu		LSU	3
sthux		LSU	3
sthx		LSU	3
stw		LSU	3
stwu		LSU	3
stwux		LSU	3
stwx		LSU	3
# Integer Load and Store with Byte Reverse Instructions			
lhbr		LSU	2
lhbrx		LSU	2
lwbr		LSU	2
sthbr		LSU	3
stwbr		LSU	3
# Integer Load and Store Multiple Instructions			
lmw		LSU	2
stmw		LSU	2

Tabelle A.1: instruction Set Listing (Fortsetzung)

## A Instruction Set Listing

---

# Integer Load and Store String Instructions			
lswi		LSU	2
lswx		LSU	2
stswi		LSU	2
stswx		LSU	2
# Memory Synchronization Instructions			
eieic		LSU	1
isync		LSU	1
lwarx		LSU	3
stwcx.		LSU	4
sync		LSU	1
# Floating Point Load Instructions			
lfd		LSU	3
lfdub		LSU	3
lfdubx		LSU	3
lfdx		LSU	3
lfs		LSU	3
lfsu		LSU	3
lfsux		LSU	3
# Floating Point Store Instructions			
stfd		LSU	3
stfdub		LSU	3
stfdubx		LSU	3
stfdx		LSU	3
stfiwx		LSU	3
stfs		LSU	3
stfsu		LSU	3
stfsux		LSU	3
stfsx		LSU	3
# Floating-Point Move Instructions			
fabs		FPU	1
fmr		FPU	1
fnabs		FPU	1
fneg		FPU	1
# Branch Instructions			
b		BPU	1
b1		BPU	1
bc		BPU	1
bcctr		BPU	1
bcctrl		BPU	1
bcrl		BPU	1
bcrl1		BPU	1
# Condition Register Logical Instructions			
crand		BPU	1
crandc		BPU	1
creqv		BPU	1
crnand		BPU	1
crnor		BPU	1
cror		BPU	1
crorc		BPU	1
crxor		BPU	1
# System Linkage Instructions			
rfl		BPU	1 # selbst ergänzt
sc		BPU	1 # selbst ergänzt
# Trap Instructions			
tw		BPU	1 # selbst ergänzt
twi		BPU	1 # selbst ergänzt
# Processor Control Instructions			
mcrf		BPU	1
mcrxr		FXU	3
mfcrr		CFX	3
mfmsr		CFX	1 # selbst ergänzt
mfmspr		CFX	1 # selbst ergänzt
mftb		CFX	1 # selbst ergänzt
mtcrr		CFX	1
mtmsr		CFX	1 # selbst ergänzt
mtmspr		CFX	1 # selbst ergänzt
# Cache Management Instructions			
dcba		LSU	3
dcbf		LSU	3
dcbi		LSU	3
dcbst		LSU	3
dcbt		LSU	2
dcbstst		LSU	2
dcbz		LSU	3
icbi		LSU	1
# Segment Register Manipulation Instructions			
mfsr		CFX	1 # selbst ergänzt
mfsrin		CFX	1 # selbst ergänzt
mtsrr		CFX	1 # selbst ergänzt
mtsrin		CFX	1 # selbst ergänzt
# Lookaside Buffer Management Instructions and External Control Instructions			
tlbia		unbek	1 # Zuordnung unklar
tlbie		unbek	1 # Zuordnung unklar
tlbsync		unbek	1 # Zuordnung unklar
eciwx		unbek	1 # Zuordnung unklar
ecowx		unbek	1 # Zuordnung unklar

Tabelle A.2: Instruction Set Listing (Fortsetzung)

## B Testfall 3: Simulation mit gzip.source

maxIdle	Häufigkeit										
	FPU0	LSU0	LSU1	LSU2	LSU3	BPU0	CFX0	unbek0	FXU0	SFX0	SFX1
1	0	225176	52543	12506	0	170787	0	0	0	1028486	45074
2	0	295009	64647	546	0	111690	0	0	0	277161	16849
3	0	282785	52369	27	0	197791	9	0	0	359122	29009
4	0	272447	33666	50	0	33573	0	0	0	317937	12743
5	0	106905	62299	95	0	275767	0	0	0	985	9208
6	0	62833	78240	5455	31	56556	0	0	0	3	45240
7	0	19830	37017	547	0	130844	0	0	0	0	132824
8	0	46	17033	508	0	38441	0	0	0	6	119510
9	0	12792	6235	631	60	5232	0	0	0	0	66087
10	0	265	8242	6470	0	98	0	0	0	0	0
11	0	4	42557	5730	0	90	9	0	0	0	14725
12	0	0	27733	4	4	411	0	0	0	0	903
13	0	0	19207	2100	0	39110	0	0	0	0	1842
14	0	3	1595	2052	24	67648	0	0	0	6	0
15	0	0	4044	0	0	20428	0	0	0	0	3399
16	0	0	12127	134	0	918	3	0	0	6138	0
17	0	0	5914	0	0	30	3	0	0	0	12179
18	0	0	347	1332	15	3384	0	0	0	0	0
19	0	3	4528	16	8	52	3	0	0	0	1485
20	0	0	637	420	75	9088	0	0	0	0	0
21	0	0	18219	303	21	0	0	0	0	0	9393
22	0	0	595	6469	0	0	0	0	0	0	916
23	0	0	792	94	0	1594	9	0	0	0	2006
24	0	0	10766	43	0	43	0	0	0	0	13417
25	0	0	1035	1066	49	5499	0	0	0	0	549
26	0	0	1649	62	16	0	0	0	0	0	0
27	0	0	13	5353	0	0	3	0	0	0	4500
28	0	0	18	71	0	0	0	0	0	0	3
29	0	0	110	128	4	0	0	0	0	0	1608
30	0	0	9374	201	0	0	0	0	0	0	9032
31	0	0	8946	550	0	0	0	0	0	0	0
32	0	0	528	126	0	0	0	0	0	0	10313
33	0	0	362	0	0	0	0	0	0	0	0
34	0	0	3632	50	0	0	0	0	0	0	0
35	0	0	9114	57	26	0	0	0	0	0	3
36	0	0	8	5	0	0	0	0	0	0	0
37	0	0	11	34	0	0	0	0	0	0	0
38	0	0	89	77	0	0	0	0	0	0	0
39	0	0	4741	3	0	0	2	0	0	0	113
40	0	0	0	14	0	0	3	0	0	0	0
41	0	0	1	842	9	0	0	0	0	0	185
42	0	0	696	211	3	0	0	0	0	0	1065
43	0	0	59	91	0	0	0	0	0	0	94
44	0	0	123	7	0	0	0	0	0	0	0
45	0	0	148	6	0	0	0	0	0	0	0
46	0	0	1	2095	0	0	0	0	0	0	0
47	0	0	15	71	0	0	0	0	0	0	410
48	0	0	49	0	0	0	0	0	0	0	7311
49	0	0	26	2	0	0	0	0	0	0	0
50	0	0	0	4939	3	0	0	0	0	0	0
51	0	0	1	14077	54	0	0	0	0	0	947
52	0	0	8	103	0	0	0	0	1	0	6692
53	0	0	12	105	62	0	0	0	0	0	0
54	0	0	0	109	0	0	0	0	0	0	225
55	0	0	3	0	0	0	0	0	0	0	9481

Tabelle B.1: gzip.source: maxIdle, Häufigkeit

B Testfall 3: Simulation mit gzip.source

56	0	0	3	10	0	0	0	0	0	0	0
57	0	0	10	570	0	0	0	0	0	0	0
59	0	0	0	6	0	0	0	0	0	0	42
60	0	0	9	0	0	0	0	0	0	0	0
61	0	0	23	69	0	0	0	0	0	0	0
62	0	0	0	21	0	0	0	0	0	0	0
63	0	0	0	23	0	0	0	0	0	0	36
64	0	0	0	56	7	0	0	0	0	0	0
65	0	0	0	15	40	0	0	0	0	0	0
66	0	0	0	4254	0	0	0	0	0	0	0
67	0	0	0	11	0	0	2	0	0	0	0
72	0	0	0	31	0	0	0	0	0	0	0
73	0	0	0	122	0	0	0	0	0	0	17
74	0	0	0	32	52	0	0	0	0	0	0
75	0	0	0	126	0	0	0	0	66	0	0
76	0	0	0	7	0	0	0	0	5231	0	0
77	0	0	0	31	4	0	2920	0	25	0	17
79	0	0	0	1	0	0	0	0	525	0	0
81	0	0	0	4	0	0	0	0	0	0	14
82	0	0	0	9	0	0	0	0	0	0	0
83	0	0	0	4215	0	0	0	0	1	0	0
84	0	0	0	9	0	0	0	0	0	0	0
85	0	0	0	40	11	0	0	0	0	0	12
86	0	0	0	69	0	0	0	0	4	0	0
87	0	0	0	601	0	0	0	0	309	0	2905
88	0	0	0	2	0	0	0	0	751	0	25
89	0	0	0	3	0	0	0	0	4	0	15
90	0	0	0	1	0	0	0	0	129	0	0
91	0	0	0	210	0	0	0	0	68	0	0
92	0	0	0	1627	0	0	1936	0	0	0	0
93	0	0	0	4	0	0	0	0	0	0	15
94	0	0	0	14	0	0	0	0	0	0	0
95	0	0	0	17	1	0	0	0	0	0	0
96	0	0	0	428	1	0	0	0	0	0	0
97	0	0	0	38	0	0	0	0	0	0	9
98	0	0	0	20	0	0	0	0	0	0	0
99	0	0	0	24	0	0	0	0	39	0	0
100	0	0	0	172	0	0	0	0	0	0	0
101	0	0	0	521	0	0	0	0	1	0	0
102	0	0	0	2	0	0	0	0	21	0	1885
103	0	0	0	6	0	0	0	0	0	0	0
104	0	0	0	26	0	0	0	0	0	0	0
105	0	0	0	154	0	0	0	0	0	0	0
106	0	0	0	1	2	0	0	0	0	0	0
107	0	0	0	11	4	0	0	0	0	0	337
108	0	0	0	79	0	0	0	0	0	0	0
109	0	0	0	68	0	0	0	0	0	0	0
110	0	0	0	151	0	0	0	0	0	0	0
111	0	0	0	150	0	0	0	0	0	0	6
113	0	0	0	23	0	0	0	0	0	0	0
114	0	0	0	53	0	0	0	0	0	0	0
115	0	0	0	28	0	0	0	0	0	0	2
116	0	0	0	3	0	0	0	0	0	0	0
117	0	0	0	32	2	0	1	0	0	0	0
118	0	0	0	39	0	0	0	0	0	0	0
119	0	0	0	25	1	0	0	0	0	0	2
120	0	0	0	2	1	0	0	0	0	0	0
121	0	0	0	7	0	0	0	0	0	0	0
122	0	0	0	10	0	0	0	0	296	0	153
123	0	0	0	26	0	0	0	0	0	0	1
124	0	0	0	1	0	0	0	0	0	0	0
125	0	0	0	15	0	0	0	0	106	0	0
126	0	0	0	21	0	0	0	0	0	0	0
127	0	0	0	39	0	0	0	0	0	0	2
128	0	0	0	5	0	0	0	0	0	0	0
130	0	0	0	2	2	0	0	0	0	0	0
131	0	0	0	7	0	0	0	0	0	0	2
132	0	0	0	3	0	0	0	0	0	0	0
133	0	0	0	5	0	0	0	0	0	0	0
134	0	0	0	5	0	0	0	0	55	0	0
135	0	0	0	17	0	0	0	0	0	0	0
136	0	0	0	0	0	0	0	0	30	0	0
137	0	0	0	3	0	0	0	0	3708	0	0
138	0	0	0	0	0	0	2324	0	35	0	0
139	0	0	0	4	0	0	0	0	0	0	0
140	0	0	0	5	9	0	0	0	49	0	0
141	0	0	0	4	2	0	0	0	493	0	7
142	0	0	0	9	0	0	358	0	8	0	0
143	0	0	0	118	0	0	0	0	0	0	0
144	0	0	0	1	0	0	0	0	1	0	0
145	0	0	0	1	0	0	0	0	143	0	4
146	0	0	0	0	0	0	138	0	1334	0	0
147	0	0	0	0	0	0	893	0	26	0	0
148	0	0	0	1	0	0	0	0	121	0	0
149	0	0	0	0	0	0	0	0	516	0	1
150	0	0	0	82	0	0	380	0	347	0	0
151	0	0	0	0	4	0	245	0	1	0	0
152	0	0	0	2	0	0	0	0	29	0	0
153	0	0	0	6	0	0	0	0	70	0	3
154	0	0	0	34	0	0	51	0	117	0	0
155	0	0	0	0	0	0	277	0	422	0	0
156	0	0	0	368	0	0	317	0	31	0	2
157	0	0	0	0	0	0	0	0	70	0	0
158	0	0	0	18	0	0	82	0	238	0	0
159	0	0	0	50	0	0	132	0	122	0	0
160	0	0	0	120	0	0	87	0	16	0	0
161	0	0	0	0	1	0	0	0	11	0	0
162	0	0	0	62	1	0	0	0	71	0	0

Tabelle B.2: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

163		0	0	0	30	2	0	72	0	47	0	0
164		0	0	0	67	0	0	40	0	106	0	0
165		0	0	0	312	1	0	65	0	93	0	0
166		0	0	0	37	0	0	91	0	29	0	0
167		0	0	0	15	0	0	49	0	74	0	0
168		0	0	0	315	0	0	48	0	42	0	0
169		0	0	0	128	0	0	24	0	27	0	0
170		0	0	0	0	0	0	116	0	6	0	0
171		0	0	0	32	1	0	2	0	32	0	0
172		0	0	0	155	5	0	20	0	14	0	0
173		0	0	0	91	2	0	45	0	23	0	0
174		0	0	0	256	2	0	18	0	7	0	0
175		0	0	0	9	0	0	4	0	10	0	2
176		0	0	0	23	0	0	3	0	43	0	0
177		0	0	0	178	0	0	17	0	14	0	0
178		0	0	0	454	0	0	101	0	6	0	0
179		0	0	0	0	0	0	0	0	8	0	2
180		0	0	0	21	0	0	20	0	22	0	0
181		0	0	0	64	0	0	22	0	18	0	0
182		0	0	0	275	0	0	79	0	4	0	0
183		0	0	0	161	7	0	0	0	125	0	0
184		0	0	0	1396	15	0	328	0	5	0	0
185		0	0	0	25	10	0	9	0	6	0	0
186		0	0	0	241	10	0	4	0	1	0	0
187		0	0	0	79	0	0	47	0	31	0	2
188		0	0	0	642	1	0	111	0	11	0	0
189		0	0	0	12	0	0	5	0	14	0	0
190		0	0	0	132	0	0	20	0	3	0	0
191		0	0	0	72	0	0	29	0	26	0	1
192		0	0	0	605	0	0	0	0	53	0	0
193		0	0	0	6	0	0	281	0	3	0	0
194		0	0	0	161	1	0	11	0	1	0	0
195		0	0	0	28	4	0	2	0	22	0	0
196		0	0	0	247	3	0	78	0	21	0	0
197		0	0	0	0	2	0	119	0	52	0	0
198		0	0	0	22	0	0	102	0	64	0	0
199		0	0	0	27	0	0	8	0	6	0	0
200		0	0	0	248	0	0	45	0	17	0	0
201		0	0	0	0	0	0	14	0	18	0	0
202		0	0	0	26	0	0	232	0	7	0	0
203		0	0	0	6	0	0	3	0	1	0	0
204		0	0	0	28	0	0	1	0	7	0	0
205		0	0	0	3	2	0	69	0	3	0	0
206		0	0	0	14	6	0	416	0	9	0	0
207		0	0	0	6	4	0	0	0	21	0	0
208		0	0	0	87	2	0	6	0	23	0	0
209		0	0	0	43	0	0	28	0	63	0	1
210		0	0	0	6	0	0	163	0	15	0	0
211		0	0	0	16	0	0	142	0	28	0	0
212		0	0	0	59	3	0	1273	0	4	0	0
213		0	0	0	24	0	0	2	0	17	0	0
214		0	0	0	4	0	0	33	0	7	0	0
215		0	0	0	1	1	0	71	0	122	0	0
216		0	0	0	37	6	0	1299	0	10	0	0
217		0	0	0	68	10	0	2	0	14	0	0
218		0	0	0	48	2	0	66	0	30	0	0
219		0	0	0	526	6	0	73	0	44	0	0
220		0	0	0	38	5	0	214	0	22	0	0
221		0	0	0	59	0	0	3	0	28	0	0
222		0	0	0	30	0	0	28	0	14	0	0
223		0	0	0	183	0	0	44	0	26	0	0
224		0	0	0	33	0	0	222	0	107	0	0
225		0	0	0	47	0	0	272	0	18	0	0
226		0	0	0	61	0	0	0	0	133	0	0
227		0	0	0	309	9	0	23	0	105	0	0
228		0	0	0	10	4	0	400	0	34	0	0
229		0	0	0	29	7	0	64	0	15	0	0
230		0	0	0	56	2	0	0	0	50	0	0
231		0	0	0	147	0	0	23	0	46	0	0
232		0	0	0	11	1	0	67	0	38	0	0
233		0	0	0	35	1	0	120	0	79	0	0
234		0	0	0	46	0	0	76	0	23	0	0
235		0	0	0	224	0	0	59	0	90	0	0
236		0	0	0	9	4	0	47	0	63	0	0
237		0	0	0	3	0	0	176	0	66	0	0
238		0	0	0	20	1	0	38	0	131	0	0
239		0	0	0	67	11	0	254	0	55	0	0
240		0	0	0	8	6	0	12	0	22	0	0
241		0	0	0	1	1	0	29	0	70	0	0
242		0	0	0	27	1	0	27	0	107	0	0
243		0	0	0	46	0	0	125	0	269	0	0
244		0	0	0	5	0	0	78	0	79	0	0
245		0	0	0	1	0	0	28	0	58	0	0
246		0	0	0	14	0	0	69	0	24	0	0
247		0	0	0	11	0	0	528	0	143	0	0
248		0	0	0	6	0	0	162	0	192	0	0
249		0	0	0	1	2	0	10	0	38	0	0
250		0	0	0	5	7	0	13	0	40	0	0
251		0	0	0	14	7	0	87	0	118	0	0
252		0	0	0	11	5	0	42	0	121	0	0
253		0	0	0	1	6	0	91	0	75	0	0
254		0	0	0	7	16	0	8	0	580	0	0
255		0	0	0	5	0	0	50	0	82	0	0
256		0	0	0	9	0	0	44	0	105	0	0
257		0	0	0	0	0	0	200	0	41	0	0
258		0	0	0	28	0	0	6	0	283	0	0
259		0	0	0	4	0	0	92	0	67	0	0
260		0	0	0	12	104	0	29	0	79	0	0

Tabelle B.3: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

### B Testfall 3: Simulation mit gzip.source

261		0	0	0	3	14	0	114	0	97	0	0
262		0	0	0	9	8	0	41	0	221	0	0
263		0	0	0	4	18	0	419	0	23	0	0
264		0	0	0	9	3	0	10	0	85	0	0
265		0	0	0	4	3	0	148	0	46	0	0
266		0	0	0	13	1	0	21	0	175	0	0
267		0	0	0	1	1	0	194	0	36	0	0
268		0	0	0	2	0	0	2	0	41	0	0
269		0	0	0	8	99	0	43	0	31	0	0
270		0	0	0	10	1	0	18	0	195	0	0
271		0	0	0	0	5	0	159	0	23	0	0
272		0	0	0	12	31	0	3	0	27	0	0
273		0	0	0	15	21	0	36	0	74	0	0
274		0	0	0	9	6	0	9	0	89	0	0
275		0	0	0	0	2	0	73	0	11	0	0
276		0	0	0	3	1	0	5	0	75	0	0
277		0	0	0	5	7	0	25	0	44	0	0
278		0	0	0	10	85	0	4	0	121	0	0
279		0	0	0	0	37	0	98	0	89	0	0
280		0	0	0	2	3	0	1	0	24	0	0
281		0	0	0	5	27	0	5	0	20	0	0
282		0	0	0	5	143	0	13	0	96	0	0
283		0	0	0	0	5	0	75	0	121	0	0
284		0	0	0	2	9	0	14	0	21	0	0
285		0	0	0	4	9	0	3	0	79	0	0
286		0	0	0	4	5	0	1	0	67	0	0
287		0	0	0	0	61	0	38	0	85	0	0
288		0	0	0	1	521	0	56	0	104	0	0
289		0	0	0	3	0	0	5	0	553	0	0
290		0	0	0	2	13	0	5	0	65	0	0
291		0	0	0	1	13	0	39	0	85	0	0
292		0	0	0	2	6	0	57	0	59	0	0
293		0	0	0	0	7	0	21	0	255	0	0
294		0	0	0	1	26	0	379	0	92	0	0
295		0	0	0	2	6	0	42	0	55	0	0
296		0	0	0	2	36	0	37	0	49	0	0
297		0	0	0	1	32	0	86	0	318	0	0
298		0	0	0	3	1	0	286	0	97	0	0
299		0	0	0	1	12	0	14	0	65	0	0
300		0	0	0	1	75	0	39	0	40	0	0
301		0	0	0	2	47	0	55	0	213	0	0
302		0	0	0	1	0	0	137	0	52	0	0
303		0	0	0	1	4	0	111	0	54	0	0
304		0	0	0	0	12	0	31	0	214	0	0
305		0	0	0	0	2	0	41	0	175	0	0
306		0	0	0	0	41	0	349	0	49	0	0
307		0	0	0	1	251	0	31	0	29	0	0
308		0	0	0	1	11	0	13	0	152	0	0
309		0	0	0	0	4	0	36	0	73	0	0
310		0	0	0	0	3	0	121	0	31	0	0
311		0	0	0	0	17	0	34	0	14	0	2
312		0	0	0	1	5	0	34	0	112	0	0
313		0	0	0	0	6	0	33	0	54	0	0
314		0	0	0	0	1	0	77	0	52	0	0
315		0	0	0	0	10	0	67	0	14	0	1
316		0	0	0	0	6	0	23	0	86	0	0
317		0	0	0	0	6	0	26	0	41	0	0
318		0	0	0	1	5	0	51	0	24	0	0
319		0	0	0	0	50	0	21	0	9	0	0
320		0	0	0	0	2	0	11	0	60	0	0
321		0	0	0	0	1	0	16	0	15	0	0
322		0	0	0	1	0	0	23	0	30	0	0
323		0	0	0	0	10	0	24	0	20	0	0
324		0	0	0	0	1	0	23	0	158	0	0
325		0	0	0	1	0	0	19	0	24	0	0
326		0	0	0	0	0	0	19	0	33	0	0
327		0	0	0	1	1	0	11	0	9	0	0
328		0	0	0	0	28	0	8	0	69	0	0
329		0	0	0	0	1	0	37	0	21	0	0
330		0	0	0	1	7	0	9	0	23	0	0
331		0	0	0	0	2	0	11	0	20	0	0
332		0	0	0	1	8	0	7	0	95	0	0
333		0	0	0	0	0	0	15	0	11	0	0
334		0	0	0	1	0	0	13	0	17	0	0
335		0	0	0	0	3	0	19	0	9	0	0
336		0	0	0	1	1	0	6	0	67	0	0
337		0	0	0	0	18	0	22	0	16	0	0
338		0	0	0	0	224	0	9	0	15	0	0
339		0	0	0	0	1	0	6	0	107	0	0
340		0	0	0	0	1	0	14	0	64	0	0
341		0	0	0	0	9	0	36	0	17	0	0
342		0	0	0	1	14	0	2	0	13	0	0
343		0	0	0	0	1	0	25	0	38	0	0
344		0	0	0	0	2	0	18	0	43	0	0
345		0	0	0	0	1	0	15	0	12	0	0
346		0	0	0	1	2	0	5	0	14	0	0
347		0	0	0	0	13	0	6	0	60	0	0
348		0	0	0	0	3	0	9	0	23	0	0
349		0	0	0	0	3	0	19	0	10	0	0
350		0	0	0	1	39	0	3	0	17	0	0
351		0	0	0	0	3	0	3	0	46	0	0
352		0	0	0	0	1	0	4	0	19	0	0
353		0	0	0	0	6	0	12	0	11	0	0
354		0	0	0	0	2	0	3	0	8	0	0
355		0	0	0	0	0	0	6	0	47	0	0
356		0	0	0	0	8	0	8	0	12	0	0
357		0	0	0	0	2	0	10	0	5	0	0
358		0	0	0	0	0	0	2	0	4	0	0

Tabelle B.4: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

359	1	0	0	0	5	0	4	0	26	0	0
360	1	0	0	0	7	0	5	0	6	0	0
361	1	0	0	0	3	0	5	0	4	0	0
362	1	0	0	0	6	0	0	0	5	0	0
363	1	0	0	0	6	0	6	0	17	0	0
364	1	0	0	0	0	0	1	0	4	0	0
365	1	0	0	0	5	0	4	0	9	0	0
366	1	0	0	0	23	0	4	0	9	0	0
367	1	0	0	0	0	0	1	0	11	0	0
368	1	0	0	0	5	0	3	0	4	0	0
369	1	0	0	0	13	0	6	0	5	0	0
370	1	0	0	1	1	0	5	0	5	0	0
371	1	0	0	0	2	0	2	0	7	0	0
372	1	0	0	0	5	0	77	0	2	0	0
373	1	0	0	0	1	0	0	0	2	0	0
374	1	0	0	0	0	0	2	0	2	0	0
375	1	0	0	1	6	0	15	0	3	0	0
376	1	0	0	0	0	0	23	0	1	0	0
377	1	0	0	0	1	0	0	0	4	0	0
378	1	0	0	0	5	0	8	0	14	0	0
379	1	0	0	1	2	0	13	0	0	0	0
380	1	0	0	1	1	0	6	0	2	0	0
381	1	0	0	0	5	0	30	0	1	0	0
382	1	0	0	0	0	0	1	0	5	0	0
383	1	0	0	1	0	0	7	0	3	0	0
384	1	0	0	0	2	0	95	0	1	0	0
385	1	0	0	0	5	0	11	0	0	0	0
386	1	0	0	0	1	0	1	0	6	0	0
387	1	0	0	0	1	0	16	0	0	0	0
388	1	0	0	0	2	0	15	0	0	0	0
389	1	0	0	0	0	0	10	0	1	0	0
390	1	0	0	0	1	0	19	0	2	0	0
391	1	0	0	0	1	0	4	0	0	0	0
392	1	0	0	0	0	0	9	0	3	0	0
393	1	0	0	0	1	0	37	0	4	0	0
394	1	0	0	0	2	0	4	0	3	0	0
395	1	0	0	0	5	0	2	0	2	0	0
396	1	0	0	0	1	0	6	0	1	0	0
397	1	0	0	0	6	0	9	0	4	0	0
398	1	0	0	0	0	0	3	0	4	0	0
399	1	0	0	0	3	0	9	0	3	0	0
400	1	0	0	0	1	0	7	0	1	0	0
401	1	0	0	0	0	0	3	0	0	0	0
402	1	0	0	0	1	0	6	0	1	0	0
403	1	0	0	0	0	0	7	0	1	0	0
404	1	0	0	0	0	0	6	0	1	0	0
405	1	0	0	0	3	0	6	0	0	0	0
406	1	0	0	0	2	0	5	0	3	0	0
407	1	0	0	0	1	0	11	0	0	0	0
408	1	0	0	0	6	0	1	0	0	0	0
409	1	0	0	0	1	0	3	0	2	0	0
410	1	0	0	0	2	0	6	0	0	0	0
411	1	0	0	0	4	0	1	0	0	0	0
412	1	0	0	0	1	0	2	0	2	0	0
413	1	0	0	0	1	0	3	0	2	0	0
414	1	0	0	0	1	0	3	0	0	0	0
415	1	0	0	0	3	0	3	0	0	0	0
416	1	0	0	0	11	0	6	0	0	0	0
417	1	0	0	0	4	0	4	0	0	0	0
418	1	0	0	0	5	0	3	0	0	0	0
419	1	0	0	0	4	0	8	0	0	0	0
420	1	0	0	0	4	0	3	0	0	0	0
421	1	0	0	0	5	0	4	0	0	0	0
422	1	0	0	0	1	0	5	0	0	0	0
423	1	0	0	0	0	0	3	0	0	0	0
425	1	0	0	0	4	0	3	0	0	0	0
426	1	0	0	0	5	0	1	0	0	0	0
427	1	0	0	0	7	0	4	0	0	0	0
428	1	0	0	0	9	0	0	0	0	0	0
429	1	0	0	0	7	0	0	0	0	0	0
430	1	0	0	0	1	0	1	0	0	0	0
431	1	0	0	0	3	0	2	0	0	0	0
432	1	0	0	0	0	0	0	0	1	0	0
433	1	0	0	0	2	0	1	0	0	0	0
434	1	0	0	0	2	0	3	0	1	0	0
435	1	0	0	0	5	0	1	0	0	0	0
436	1	0	0	0	8	0	0	0	0	0	0
437	1	0	0	0	2	0	0	0	0	0	0
438	1	0	0	0	5	0	1	0	0	0	0
439	1	0	0	0	6	0	3	0	1	0	0
440	1	0	0	0	1	0	0	0	1	0	0
441	1	0	0	0	3	0	2	0	0	0	0
442	1	0	0	0	1	0	0	0	0	0	0
443	1	0	0	0	2	0	0	0	0	0	0
444	1	0	0	0	6	0	2	0	0	0	0
445	1	0	0	0	2	0	2	0	0	0	0
446	1	0	0	0	1	0	0	0	0	0	0
447	1	0	0	0	7	0	0	0	0	0	0
448	1	0	0	2	4	0	1	0	0	0	0
449	1	0	0	0	4	0	0	0	0	0	0
450	1	0	0	0	2	0	18	0	0	0	0
451	1	0	0	0	2	0	1	0	1	0	0
452	1	0	0	0	3	0	1	0	1	0	0
453	1	0	0	0	0	0	4	0	1	0	0
454	1	0	0	0	2	0	4	0	0	0	0
455	1	0	0	0	2	0	0	0	0	0	0
456	1	0	0	0	4	0	0	0	0	0	0
457	1	0	0	0	0	0	1	0	0	0	0
458	1	0	0	1	1	0	1	0	0	0	0

Tabelle B.5: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

459	1	0	0	0	0	7	0	14	0	0	0	1
460	1	0	0	0	0	5	0	1	0	0	0	0
461	1	0	0	0	0	2	0	3	0	0	0	0
462	1	0	0	0	0	1	0	44	0	0	0	0
463	1	0	0	0	0	3	0	3	0	0	0	1
464	1	0	0	0	0	1	0	3	0	0	0	0
465	1	0	0	0	0	0	0	11	0	0	0	0
466	1	0	0	0	0	3	0	6	0	0	0	0
467	1	0	0	0	0	1	0	1	0	0	0	0
468	1	0	0	0	0	4	0	4	0	0	0	0
469	1	0	0	0	1	2	0	1	0	0	0	0
470	1	0	0	0	0	0	0	4	0	0	0	0
471	1	0	0	0	0	6	0	20	0	0	0	0
472	1	0	0	0	0	1	0	1	0	0	0	0
473	1	0	0	0	0	3	0	0	0	0	0	0
474	1	0	0	0	0	2	0	1	0	0	0	0
475	1	0	0	0	0	1	0	4	0	0	0	0
476	1	0	0	0	0	1	0	0	0	0	0	0
477	1	0	0	0	0	3	0	3	0	0	0	0
478	1	0	0	0	0	1	0	2	0	0	0	0
479	1	0	0	0	0	0	0	3	0	0	0	0
480	1	0	0	0	0	2	0	4	0	0	0	0
481	1	0	0	0	0	3	0	4	0	0	0	0
482	1	0	0	0	0	0	0	2	0	0	0	0
483	1	0	0	0	0	4	0	1	0	0	0	0
484	1	0	0	0	0	2	0	1	0	1	0	0
485	1	0	0	0	0	1	0	2	0	0	0	0
486	1	0	0	0	0	10	0	1	0	0	0	0
487	1	0	0	0	0	4	0	0	0	0	0	0
488	1	0	0	0	0	1	0	3	0	0	0	0
489	1	0	0	0	0	2	0	3	0	0	0	0
490	1	0	0	0	0	2	0	0	0	0	0	0
491	1	0	0	0	0	1	0	2	0	0	0	0
492	1	0	0	0	0	2	0	3	0	0	0	0
493	1	0	0	0	0	1	0	1	0	0	0	0
494	1	0	0	0	0	7	0	3	0	0	0	0
495	1	0	0	0	0	4	0	2	0	0	0	0
496	1	0	0	0	0	3	0	0	0	0	0	0
497	1	0	0	0	0	4	0	2	0	0	0	0
498	1	0	0	0	0	2	0	1	0	0	0	0
499	1	0	0	0	0	2	0	2	0	0	0	0
500	1	0	0	0	0	7	0	1	0	0	0	0
502	1	0	0	0	0	2	0	2	0	0	0	0
503	1	0	0	0	0	4	0	0	0	0	0	0
504	1	0	0	0	0	6	0	1	0	0	0	0
505	1	0	0	0	0	4	0	3	0	0	0	0
506	1	0	0	0	0	5	0	2	0	0	0	0
507	1	0	0	0	0	2	0	1	0	2	0	0
508	1	0	0	0	0	2	0	1	0	0	0	0
509	1	0	0	0	0	10	0	2	0	0	0	0
510	1	0	0	0	0	1	0	0	0	0	0	0
512	1	0	0	0	0	2	0	1	0	0	0	0
513	1	0	0	0	0	2	0	0	0	0	0	0
514	1	0	0	0	0	3	0	0	0	0	0	0
515	1	0	0	0	0	2	0	0	0	0	0	0
516	1	0	0	0	0	4	0	0	0	0	0	0
517	1	0	0	0	0	7	0	1	0	1	0	0
518	1	0	0	0	0	3	0	0	0	0	0	0
519	1	0	0	0	0	1	0	2	0	0	0	0
520	1	0	0	0	0	1	0	0	0	0	0	0
521	1	0	0	0	0	6	0	1	0	0	0	0
522	1	0	0	0	0	7	0	1	0	0	0	0
523	1	0	0	0	0	3	0	3	0	0	0	0
524	1	0	0	0	0	6	0	2	0	0	0	0
525	1	0	0	0	0	4	0	3	0	0	0	0
526	1	0	0	0	0	12	0	0	0	0	0	0
527	1	0	0	0	0	4	0	2	0	0	0	0
528	1	0	0	0	0	16	0	10	0	1	0	0
529	1	0	0	0	0	2	0	1	0	0	0	0
530	1	0	0	0	0	4	0	0	0	0	0	0
531	1	0	0	0	0	5	0	6	0	0	0	0
532	1	0	0	0	0	5	0	1	0	0	0	0
533	1	0	0	0	0	8	0	1	0	0	0	0
534	1	0	0	0	0	4	0	0	0	0	0	0
535	1	0	0	0	0	3	0	0	0	0	0	0
536	1	0	0	0	0	7	0	0	0	0	0	0
537	1	0	0	0	0	17	0	3	0	0	0	0
538	1	0	0	0	0	4	0	0	0	0	0	0
539	1	0	0	0	0	3	0	0	0	0	0	0
540	1	0	0	0	0	15	0	19	0	0	0	0
541	1	0	0	0	0	6	0	1	0	0	0	0
542	1	0	0	0	0	14	0	0	0	0	0	0
543	1	0	0	0	0	6	0	3	0	0	0	0
544	1	0	0	0	0	8	0	0	0	0	0	0
545	1	0	0	0	0	10	0	2	0	0	0	0
546	1	0	0	0	0	16	0	2	0	0	0	0
547	1	0	0	0	0	7	0	1	0	0	0	0
548	1	0	0	0	0	7	0	1	0	0	0	0
549	1	0	0	0	0	2	0	13	0	0	0	0
550	1	0	0	0	0	16	0	1	0	0	0	0
551	1	0	0	0	0	12	0	1	0	0	0	0
552	1	0	0	0	0	22	0	0	0	0	0	0
553	1	0	0	0	0	2	0	3	0	0	0	0
554	1	0	0	0	0	12	0	0	0	0	0	0
555	1	0	0	0	0	8	0	0	0	0	0	0

Tabelle B.6: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

556	1	0	0	0	15	0	1	0	0	0	0
557	1	0	0	0	5	0	2	0	0	0	0
558	1	0	0	0	10	0	4	0	0	0	0
559	1	0	0	0	8	0	0	0	0	0	0
560	1	0	0	0	16	0	1	0	0	0	0
561	1	0	0	0	2	0	1	0	0	0	0
562	1	0	0	0	5	0	2	0	0	0	0
563	1	0	0	0	7	0	2	0	0	0	0
564	1	0	0	0	16	0	0	0	0	0	0
565	1	0	0	0	5	0	0	0	0	0	0
566	1	0	0	0	4	0	0	0	0	0	0
567	1	0	0	0	6	0	0	0	0	0	0
568	1	0	0	0	17	0	0	0	0	0	0
569	1	0	0	0	7	0	0	0	0	0	0
570	1	0	0	0	6	0	0	0	0	0	0
571	1	0	0	0	14	0	1	0	0	0	0
572	1	0	0	0	15	0	2	0	0	0	0
573	1	0	0	0	14	0	0	0	0	0	0
574	1	0	0	0	7	0	1	0	0	0	0
575	1	0	0	0	13	0	0	0	0	0	0
576	1	0	0	0	7	0	0	0	0	0	0
577	1	0	0	0	10	0	0	0	0	0	0
578	1	0	0	0	6	0	2	0	0	0	0
579	1	0	0	0	11	0	0	0	0	0	0
580	1	0	0	0	4	0	2	0	0	0	0
581	1	0	0	0	8	0	2	0	0	0	0
582	1	0	0	0	12	0	0	0	0	0	0
583	1	0	0	0	16	0	1	0	0	0	0
584	1	0	0	0	14	0	0	0	0	0	0
585	1	0	0	0	14	0	0	0	0	0	0
586	1	0	0	0	14	0	1	0	0	0	0
587	1	0	0	0	24	0	1	0	0	0	0
588	1	0	0	0	3	0	1	0	0	0	0
589	1	0	0	0	9	0	0	0	0	0	0
590	1	0	0	0	11	0	1	0	0	0	0
591	1	0	0	0	18	0	1	0	0	0	0
592	1	0	0	0	10	0	0	0	0	0	0
593	1	0	0	0	12	0	0	0	0	0	0
594	1	0	0	0	3	0	0	0	0	0	0
595	1	0	0	0	13	0	1	0	0	0	0
596	1	0	0	1	6	0	0	0	0	0	0
597	1	0	0	0	8	0	0	0	0	0	0
598	1	0	0	0	6	0	1	0	0	0	0
599	1	0	0	0	10	0	0	0	0	0	0
600	1	0	0	0	12	0	1	0	0	0	0
601	1	0	0	0	8	0	0	0	0	0	0
602	1	0	0	0	21	0	0	0	0	0	0
603	1	0	0	0	10	0	0	0	0	0	0
604	1	0	0	0	8	0	0	0	0	0	0
605	1	0	0	0	4	0	0	0	0	0	0
606	1	0	0	1	18	0	0	0	0	0	0
607	1	0	0	0	7	0	0	0	0	0	0
608	1	0	0	0	6	0	0	0	0	0	0
609	1	0	0	0	5	0	0	0	0	0	0
610	1	0	0	0	13	0	0	0	0	0	0
611	1	0	0	0	5	0	0	0	0	0	0
612	1	0	0	0	5	0	1	0	0	0	0
613	1	0	0	0	4	0	0	0	0	0	0
614	1	0	0	0	11	0	1	0	0	0	0
615	1	0	0	0	3	0	1	0	0	0	0
616	1	0	0	0	7	0	1	0	0	0	0
617	1	0	0	0	3	0	0	0	0	0	0
618	1	0	0	0	16	0	10	0	0	0	0
619	1	0	0	0	3	0	0	0	0	0	0
620	1	0	0	0	2	0	1	0	0	0	0
621	1	0	0	0	4	0	3	0	0	0	0
622	1	0	0	0	6	0	2	0	0	0	0
623	1	0	0	0	2	0	0	0	0	0	0
624	1	0	0	0	4	0	1	0	0	0	0
625	1	0	0	0	3	0	0	0	0	0	0
626	1	0	0	0	7	0	0	0	0	0	0
627	1	0	0	0	4	0	7	0	0	0	0
629	1	0	0	0	2	0	0	0	0	0	0
630	1	0	0	0	6	0	0	0	0	0	0
631	1	0	0	0	3	0	1	0	0	0	0
632	1	0	0	0	7	0	0	0	0	0	0
633	1	0	0	0	5	0	0	0	0	0	0
634	1	0	0	0	6	0	1	0	0	0	0
635	1	0	0	0	4	0	0	0	0	0	0
636	1	0	0	0	0	0	4	0	0	0	0
637	1	0	0	0	10	0	0	0	0	0	0
638	1	0	0	0	6	0	0	0	0	0	0
639	1	0	0	0	1	0	0	0	0	0	0
640	1	0	0	0	4	0	0	0	0	0	0
641	1	0	0	0	5	0	1	0	0	0	0
642	1	0	0	0	8	0	0	0	0	0	0
643	1	0	0	0	3	0	0	0	0	0	0
644	1	0	0	0	5	0	2	0	0	0	0
645	1	0	0	0	3	0	0	0	0	0	0
646	1	0	0	0	5	0	0	0	0	0	0
647	1	0	0	0	4	0	0	0	0	0	0
648	1	0	0	0	6	0	0	0	0	0	0
649	1	0	0	0	2	0	1	0	0	0	0
650	1	0	0	0	6	0	1	0	0	0	0
651	1	0	0	0	2	0	0	0	0	0	0
652	1	0	0	0	3	0	0	0	0	0	0
653	1	0	0	0	6	0	0	0	0	0	0
654	1	0	0	0	3	0	1	0	0	0	0
655	1	0	0	0	2	0	0	0	0	0	0

Tabelle B.7: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

656	1	0	0	0	0	2	0	0	0	0	0	0
657	1	0	0	0	0	5	0	0	0	0	0	0
658	1	0	0	0	0	4	0	0	0	0	0	0
659	1	0	0	0	0	2	0	2	0	0	0	0
660	1	0	0	0	0	1	0	0	0	0	0	0
661	1	0	0	0	0	3	0	0	0	0	0	0
662	1	0	0	0	0	2	0	0	0	0	0	0
663	1	0	0	0	0	3	0	0	0	0	0	0
664	1	0	0	0	0	2	0	0	0	0	0	0
665	1	0	0	0	0	3	0	0	0	0	0	0
666	1	0	0	0	0	0	0	0	0	1	0	0
667	1	0	0	0	0	1	0	1	0	0	0	0
668	1	0	0	0	0	3	0	1	0	0	0	0
669	1	0	0	0	0	1	0	0	0	0	0	0
670	1	0	0	0	0	2	0	0	0	0	0	0
671	1	0	0	0	0	4	0	0	0	0	0	0
672	1	0	0	0	0	3	0	0	0	0	0	0
673	1	0	0	0	0	1	0	1	0	0	0	0
674	1	0	0	0	0	1	0	1	0	0	0	0
675	1	0	0	0	0	2	0	0	0	0	0	0
676	1	0	0	0	0	9	0	0	0	1	0	0
677	1	0	0	0	0	1	0	1	0	0	0	0
678	1	0	0	0	0	1	0	0	0	0	0	0
680	1	0	0	0	0	5	0	0	0	0	0	0
681	1	0	0	0	0	2	0	0	0	0	0	0
682	1	0	0	0	0	1	0	0	0	0	0	0
683	1	0	0	0	0	2	0	0	0	0	0	0
684	1	0	0	0	0	2	0	0	0	0	0	0
685	1	0	0	0	0	1	0	0	0	0	0	0
687	1	0	0	0	0	1	0	0	0	0	0	0
689	1	0	0	0	0	1	0	1	0	0	0	0
690	1	0	0	0	0	3	0	0	0	0	0	0
691	1	0	0	0	0	2	0	0	0	0	0	0
692	1	0	0	0	0	2	0	0	0	0	0	0
693	1	0	0	0	0	3	0	0	0	0	0	0
694	1	0	0	0	0	3	0	0	0	0	0	0
695	1	0	0	0	0	1	0	0	0	0	0	0
696	1	0	0	0	0	0	0	4	0	0	0	0
697	1	0	0	0	0	4	0	0	0	0	0	0
698	1	0	0	0	0	2	0	0	0	0	0	0
700	1	0	0	0	0	1	0	1	0	0	0	0
701	1	0	0	0	0	5	0	1	0	0	0	0
702	1	0	0	0	0	2	0	0	0	0	0	0
703	1	0	0	0	0	4	0	0	0	0	0	0
704	1	0	0	0	0	3	0	0	0	0	0	0
705	1	0	0	0	0	3	0	4	0	0	0	0
706	1	0	0	0	0	1	0	0	0	0	0	0
707	1	0	0	0	0	3	0	0	0	0	0	0
708	1	0	0	0	0	2	0	0	0	0	0	0
709	1	0	0	0	0	2	0	0	0	0	0	0
710	1	0	0	0	0	2	0	0	0	0	0	0
711	1	0	0	0	0	3	0	1	0	0	0	0
713	1	0	0	0	0	1	0	0	0	0	0	0
714	1	0	0	0	0	2	0	1	0	0	0	0
715	1	0	0	0	0	1	0	0	0	0	0	0
716	1	0	0	0	0	5	0	0	0	0	0	0
717	1	0	0	0	0	1	0	0	0	0	0	0
718	1	0	0	0	0	3	0	1	0	0	0	0
719	1	0	0	0	0	2	0	0	0	0	0	0
720	1	0	0	0	0	1	0	0	0	0	0	0
721	1	0	0	0	0	0	0	1	0	0	0	0
722	1	0	0	0	0	1	0	0	0	0	0	0
723	1	0	0	0	0	5	0	0	0	0	0	0
724	1	0	0	0	0	2	0	0	0	0	0	0
725	1	0	0	0	0	1	0	0	0	0	0	0
726	1	0	0	0	0	2	0	0	0	0	0	0
727	1	0	0	0	0	4	0	1	0	0	0	0
728	1	0	0	0	0	3	0	0	0	0	0	0
729	1	0	0	0	0	2	0	0	0	0	0	0
730	1	0	0	0	0	2	0	0	0	0	0	0
731	1	0	0	0	0	2	0	0	0	0	0	0
732	1	0	0	0	0	2	0	0	0	0	0	0
733	1	0	0	0	0	2	0	0	0	0	0	0
735	1	0	0	0	0	3	0	0	0	0	0	0
736	1	0	0	0	0	2	0	0	0	0	0	0
737	1	0	0	0	0	3	0	0	0	0	0	0
738	1	0	0	0	0	6	0	0	0	0	0	0
739	1	0	0	0	0	3	0	0	0	0	0	0
740	1	0	0	0	0	4	0	0	0	0	0	0
741	1	0	0	0	0	4	0	0	0	0	0	0
742	1	0	0	0	0	1	0	0	0	0	0	0
743	1	0	0	0	0	3	0	0	0	0	0	0
745	1	0	0	0	0	1	0	0	0	0	0	0
746	1	0	0	0	0	2	0	0	0	0	0	0
747	1	0	0	0	0	5	0	0	0	0	0	0
749	1	0	0	0	0	2	0	0	0	0	0	0
752	1	0	0	0	0	2	0	0	0	0	0	0
753	1	0	0	0	0	2	0	0	0	0	0	0
754	1	0	0	0	0	3	0	0	0	0	0	0
755	1	0	0	0	0	1	0	0	0	0	0	0
756	1	0	0	0	0	3	0	0	0	0	0	0
757	1	0	0	0	0	3	0	0	0	0	0	0
758	1	0	0	0	0	3	0	0	0	0	0	0
759	1	0	0	0	0	4	0	0	0	0	0	0
760	1	0	0	0	0	4	0	0	0	0	0	0
763	1	0	0	0	0	4	0	0	0	0	0	0

Tabelle B.8: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

764	1	0	0	0	0	1	0	0	0	0	0	0
765	1	0	0	0	0	2	0	0	0	0	0	0
767	1	0	0	0	0	6	0	0	0	0	0	0
768	1	0	0	0	0	1	0	0	0	0	0	0
769	1	0	0	0	0	1	0	0	0	0	0	0
770	1	0	0	0	0	2	0	0	0	0	0	0
771	1	0	0	0	0	3	0	0	0	0	0	0
772	1	0	0	0	0	2	0	0	0	0	0	0
773	1	0	0	0	0	1	0	0	0	0	0	0
774	1	0	0	0	0	3	0	3	0	0	0	0
775	1	0	0	0	0	2	0	0	0	0	0	0
776	1	0	0	0	0	4	0	0	0	0	0	0
777	1	0	0	0	0	1	0	0	0	0	0	0
778	1	0	0	0	0	1	0	1	0	0	0	0
780	1	0	0	0	0	3	0	0	0	0	0	0
781	1	0	0	0	0	2	0	0	0	0	0	0
782	1	0	0	0	0	2	0	0	0	0	0	0
783	1	0	0	0	0	1	0	0	0	0	0	0
784	1	0	0	0	0	6	0	0	0	0	0	0
785	1	0	0	0	0	2	0	0	0	0	0	0
786	1	0	0	0	0	2	0	0	0	0	0	0
788	1	0	0	0	0	3	0	0	0	0	0	0
789	1	0	0	0	0	6	0	0	0	0	0	0
790	1	0	0	0	0	0	0	1	0	0	0	0
791	1	0	0	0	0	3	0	0	0	0	0	0
792	1	0	0	0	0	6	0	0	0	0	0	0
793	1	0	0	0	0	0	0	1	0	0	0	0
794	1	0	0	0	0	5	0	0	0	0	0	0
795	1	0	0	0	0	1	0	1	0	0	0	0
796	1	0	0	0	0	2	0	0	0	0	0	0
797	1	0	0	0	0	5	0	0	0	0	0	0
798	1	0	0	0	0	4	0	0	0	0	0	0
799	1	0	0	0	0	5	0	0	0	0	0	0
800	1	0	0	0	0	3	0	0	0	0	0	0
801	1	0	0	0	0	2	0	0	0	0	0	0
802	1	0	0	0	0	2	0	0	0	0	0	0
803	1	0	0	0	0	5	0	0	0	0	0	0
804	1	0	0	0	0	4	0	0	0	0	0	0
805	1	0	0	0	0	1	0	0	0	0	0	0
806	1	0	0	0	0	3	0	0	0	0	0	0
807	1	0	0	0	0	3	0	0	0	0	0	0
808	1	0	0	0	0	4	0	1	0	0	0	0
809	1	0	0	0	0	1	0	0	0	0	0	0
810	1	0	0	0	0	4	0	0	0	0	0	0
811	1	0	0	0	0	3	0	0	0	0	0	0
812	1	0	0	0	0	6	0	0	0	0	0	0
813	1	0	0	0	0	8	0	0	0	0	0	0
814	1	0	0	0	0	5	0	0	0	0	0	0
815	1	0	0	0	0	5	0	0	0	0	0	0
816	1	0	0	0	0	8	0	0	0	0	0	0
817	1	0	0	0	0	2	0	0	0	0	0	0
818	1	0	0	0	0	3	0	0	0	0	0	0
819	1	0	0	0	0	2	0	0	0	0	0	0
820	1	0	0	0	0	3	0	0	0	0	0	0
821	1	0	0	0	0	2	0	0	0	0	0	0
822	1	0	0	0	0	8	0	0	0	0	0	0
823	1	0	0	0	0	3	0	0	0	0	0	0
824	1	0	0	0	0	6	0	0	0	0	0	0
825	1	0	0	0	0	2	0	0	0	0	0	0
826	1	0	0	0	0	4	0	0	0	0	0	0
827	1	0	0	0	0	3	0	0	0	0	0	0
828	1	0	0	0	0	2	0	0	0	0	0	0
829	1	0	0	0	0	4	0	0	0	0	0	0
830	1	0	0	0	0	4	0	0	0	0	0	0
831	1	0	0	0	0	4	0	0	0	0	0	0
832	1	0	0	0	0	4	0	0	0	0	0	0
833	1	0	0	0	0	4	0	0	0	0	0	0
834	1	0	0	0	0	7	0	0	0	0	0	0
835	1	0	0	0	0	2	0	0	0	0	0	0
836	1	0	0	0	0	2	0	0	0	0	0	0
837	1	0	0	0	0	4	0	0	0	0	0	0
838	1	0	0	0	0	4	0	0	0	0	0	0
839	1	0	0	0	0	4	0	0	0	0	0	0
840	1	0	0	0	0	4	0	0	0	0	0	0
841	1	0	0	0	0	6	0	0	0	0	0	0
842	1	0	0	0	0	2	0	0	0	0	0	0
843	1	0	0	0	0	4	0	0	0	0	0	0
844	1	0	0	0	0	4	0	0	0	0	0	0
845	1	0	0	0	0	3	0	0	0	0	0	0
846	1	0	0	0	0	4	0	0	0	0	0	0
847	1	0	0	0	0	9	0	0	0	0	0	0
848	1	0	0	0	0	7	0	0	0	0	0	0
849	1	0	0	0	0	5	0	0	0	0	0	0
850	1	0	0	0	0	2	0	0	0	0	0	0
851	1	0	0	0	0	1	0	0	0	0	0	0
852	1	0	0	0	0	1	0	0	0	0	0	0
853	1	0	0	0	0	5	0	0	0	0	0	0
854	1	0	0	0	0	3	0	0	0	0	0	0
855	1	0	0	0	0	8	0	0	0	0	0	0
856	1	0	0	0	0	2	0	0	0	0	0	0
857	1	0	0	0	0	3	0	0	0	0	0	0
858	1	0	0	0	0	7	0	1	0	0	0	0
859	1	0	0	0	0	6	0	0	0	0	0	0
860	1	0	0	0	0	3	0	0	0	0	0	0
861	1	0	0	0	0	6	0	0	0	0	0	0
862	1	0	0	0	0	2	0	0	0	0	0	0

Tabelle B.9: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

863	1	0	0	0	0	4	0	0	0	0	0	0
864	1	0	0	0	0	4	0	1	0	0	0	0
865	1	0	0	0	0	2	0	0	0	0	0	0
866	1	0	0	0	0	1	0	0	0	0	0	0
867	1	0	0	0	0	2	0	0	0	0	0	0
868	1	0	0	0	0	4	0	0	0	0	0	0
869	1	0	0	0	0	4	0	0	0	0	0	0
870	1	0	0	0	0	1	0	0	0	0	0	0
871	1	0	0	0	0	4	0	0	0	0	0	0
872	1	0	0	0	0	5	0	0	0	0	0	0
873	1	0	0	0	0	3	0	0	0	0	0	0
874	1	0	0	0	0	6	0	0	0	0	0	0
875	1	0	0	0	0	2	0	0	0	0	0	0
876	1	0	0	0	0	3	0	1	0	0	0	0
878	1	0	0	0	0	4	0	0	0	0	0	0
879	1	0	0	0	0	6	0	0	0	0	0	0
880	1	0	0	0	0	7	0	0	0	0	0	0
881	1	0	0	0	0	8	0	0	0	0	0	0
882	1	0	0	0	0	9	0	0	0	0	0	0
885	1	0	0	0	0	3	0	0	0	0	0	0
886	1	0	0	0	0	1	0	0	0	0	0	0
887	1	0	0	0	0	1	0	0	0	0	0	0
888	1	0	0	0	0	2	0	0	0	0	0	0
889	1	0	0	0	0	4	0	0	0	0	0	0
890	1	0	0	0	0	3	0	0	0	0	0	0
891	1	0	0	0	0	2	0	0	0	0	0	0
892	1	0	0	0	0	2	0	0	0	0	0	0
893	1	0	0	0	0	2	0	0	0	0	0	0
894	1	0	0	0	0	4	0	0	0	0	0	0
895	1	0	0	0	0	3	0	0	0	0	0	0
896	1	0	0	0	0	2	0	0	0	0	0	0
897	1	0	0	0	0	4	0	0	0	0	0	0
898	1	0	0	0	0	3	0	0	0	0	0	0
899	1	0	0	0	0	1	0	0	0	0	0	0
900	1	0	0	0	0	3	0	0	0	0	0	0
901	1	0	0	0	0	1	0	0	0	0	0	0
902	1	0	0	0	0	3	0	1	0	0	0	0
903	1	0	0	0	0	2	0	0	0	0	0	0
904	1	0	0	0	0	4	0	0	0	0	0	0
905	1	0	0	0	0	2	0	0	0	0	0	0
906	1	0	0	0	0	5	0	0	0	0	0	0
907	1	0	0	0	0	2	0	0	0	0	0	0
908	1	0	0	0	0	2	0	0	0	0	0	0
909	1	0	0	0	0	8	0	0	0	0	0	0
910	1	0	0	0	0	2	0	0	0	0	0	0
911	1	0	0	0	0	2	0	0	0	0	0	0
912	1	0	0	0	0	4	0	0	0	0	0	0
913	1	0	0	0	0	1	0	0	0	0	0	0
914	1	0	0	0	0	2	0	0	0	0	0	0
915	1	0	0	0	0	2	0	0	0	0	0	0
916	1	0	0	0	0	2	0	0	0	0	0	0
917	1	0	0	0	0	3	0	0	0	0	0	0
918	1	0	0	0	0	1	0	0	0	0	0	0
919	1	0	0	0	0	3	0	0	0	0	0	0
920	1	0	0	0	0	4	0	0	0	0	0	0
921	1	0	0	0	0	1	0	0	0	0	0	0
922	1	0	0	0	0	4	0	0	0	0	0	0
923	1	0	0	0	0	3	0	0	0	0	0	0
924	1	0	0	0	0	2	0	0	0	0	0	0
925	1	0	0	0	0	4	0	0	0	0	0	0
926	1	0	0	0	0	1	0	0	0	0	0	0
927	1	0	0	0	0	2	0	0	0	0	0	0
929	1	0	0	0	0	1	0	0	0	0	0	0
930	1	0	0	0	0	2	0	0	0	0	0	0
931	1	0	0	0	0	2	0	0	0	0	0	0
932	1	0	0	0	0	1	0	0	0	0	0	0
933	1	0	0	0	0	3	0	0	0	0	0	0
934	1	0	0	0	0	1	0	0	0	0	0	0
935	1	0	0	0	0	3	0	0	0	0	0	0
936	1	0	0	0	0	1	0	0	0	0	0	0
938	1	0	0	0	0	2	0	0	0	0	0	0
939	1	0	0	0	0	5	0	0	0	0	0	0
940	1	0	0	0	0	4	0	0	0	0	0	0
941	1	0	0	0	0	2	0	0	0	0	0	0
942	1	0	0	0	0	1	0	0	0	0	0	0
943	1	0	0	0	0	1	0	0	0	0	0	0
944	1	0	0	0	0	2	0	0	0	0	0	0
945	1	0	0	0	0	4	0	0	0	0	0	0
946	1	0	0	0	0	1	0	0	0	0	0	0
948	1	0	0	0	0	3	0	0	0	0	0	0
950	1	0	0	0	0	3	0	0	0	0	0	0
952	1	0	0	0	0	4	0	0	0	0	0	0
953	1	0	0	0	0	2	0	0	0	0	0	0
954	1	0	0	0	0	2	0	0	0	0	0	0
955	1	0	0	0	0	2	0	0	0	0	0	0
956	1	0	0	0	0	1	0	0	0	0	0	0
957	1	0	0	0	0	2	0	0	0	0	0	0
958	1	0	0	0	0	1	0	0	0	0	0	0
959	1	0	0	0	0	2	0	0	0	0	0	0
962	1	0	0	0	0	1	0	0	0	0	0	0

Tabelle B.10: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

964		0	0	0	0	3	0	0	0	0	0	0
966		0	0	0	0	2	0	0	0	0	0	0
968		0	0	0	0	1	0	0	0	0	0	0
970		0	0	0	0	2	0	0	0	0	0	0
971		0	0	0	0	1	0	0	0	0	0	0
972		0	0	0	0	4	0	0	0	0	0	0
973		0	0	0	0	3	0	0	0	0	0	0
974		0	0	0	0	2	0	0	0	0	0	0
975		0	0	0	0	2	0	0	0	0	0	0
976		0	0	0	0	2	0	0	0	0	0	0
979		0	0	0	0	1	0	0	0	0	0	0
980		0	0	0	0	4	0	0	0	0	0	0
981		0	0	0	0	1	0	0	0	0	0	0
982		0	0	0	0	1	0	0	0	0	0	0
983		0	0	0	0	1	0	0	0	0	0	0
984		0	0	0	0	1	0	0	0	0	0	0
985		0	0	0	0	1	0	0	0	0	0	0
986		0	0	0	0	0	0	1	0	0	0	0
990		0	0	0	0	2	0	0	0	0	0	0
991		0	0	0	0	2	0	0	0	0	0	0
992		0	0	0	0	3	0	0	0	0	0	0
993		0	0	0	0	3	0	0	0	0	0	0
994		0	0	0	0	1	0	0	0	0	0	0
995		0	0	0	0	3	0	0	0	0	0	0
996		0	0	0	0	1	0	0	0	0	0	0
997		0	0	0	0	1	0	0	0	0	0	0
998		0	0	0	0	2	0	0	0	0	0	0
1000		0	0	0	0	1	0	0	0	0	0	0
1001		0	0	0	0	1	0	0	0	0	0	0
1007		0	0	0	0	1	0	0	0	0	0	0
1008		0	0	0	0	1	0	1	0	0	0	0
1010		0	0	0	0	2	0	0	0	0	0	0
1011		0	0	0	0	2	0	0	0	0	0	0
1012		0	0	0	0	3	0	0	0	0	0	0
1013		0	0	0	0	1	0	0	0	0	0	0
1014		0	0	0	0	1	0	0	0	0	0	0
1015		0	0	0	0	7	0	0	0	0	0	0
1016		0	0	0	0	1	0	0	0	0	0	0
1017		0	0	0	0	3	0	0	0	0	0	0
1018		0	0	0	0	4	0	0	0	0	0	0
1019		0	0	0	0	1	0	0	0	0	0	0
1020		0	0	0	0	1	0	0	0	0	0	0
1021		0	0	0	0	1	0	0	0	0	0	0
1022		0	0	0	0	1	0	0	0	0	0	0
1023		0	0	0	0	2	0	0	0	0	0	0
1024		0	0	0	0	1	0	0	0	0	0	0
1025		0	0	0	0	1	0	0	0	0	0	0
1026		0	0	0	0	3	0	0	0	0	0	0
1027		0	0	0	0	1	0	0	0	0	0	0
1028		0	0	0	0	3	0	0	0	0	0	0
1029		0	0	0	0	1	0	0	0	0	0	0
1031		0	0	0	0	1	0	0	0	0	0	0
1032		0	0	0	0	1	0	0	0	0	0	0
1033		0	0	0	0	1	0	0	0	0	0	0
1034		0	0	0	0	1	0	0	0	0	0	0
1035		0	0	0	0	2	0	0	0	0	0	0
1037		0	0	0	0	1	0	0	0	0	0	0
1038		0	0	0	0	3	0	0	0	0	0	0
1039		0	0	0	0	1	0	0	0	0	0	0
1040		0	0	0	0	3	0	0	0	0	0	0
1041		0	0	0	0	1	0	0	0	0	0	0
1042		0	0	0	0	1	0	0	0	0	0	0
1045		0	0	0	0	1	0	0	0	0	0	0
1046		0	0	0	0	3	0	0	0	0	0	0
1047		0	0	0	0	1	0	0	0	0	0	0
1048		0	0	0	0	1	0	0	0	0	0	0
1049		0	0	0	0	3	0	0	0	0	0	0
1052		0	0	0	0	1	0	0	0	0	0	0
1054		0	0	0	0	1	0	0	0	0	0	0
1055		0	0	0	0	1	0	0	0	0	0	0
1056		0	0	0	0	3	0	0	0	0	0	0
1057		0	0	0	0	2	0	0	0	0	0	0
1058		0	0	0	0	1	0	0	0	0	0	0
1061		0	0	0	0	2	0	0	0	0	0	0
1062		0	0	0	0	3	0	0	0	0	0	0
1063		0	0	0	0	1	0	0	0	0	0	0
1064		0	0	0	0	1	0	0	0	0	0	0
1066		0	0	0	0	1	0	0	0	0	0	0
1068		0	0	0	0	3	0	0	0	0	0	0
1069		0	0	0	0	2	0	0	0	0	0	0
1070		0	0	0	0	1	0	0	0	0	0	0
1071		0	0	0	0	1	0	0	0	0	0	0
1072		0	0	0	0	2	0	0	0	0	0	0
1073		0	0	0	0	4	0	0	0	0	0	0
1075		0	0	0	0	2	0	0	0	0	0	0
1078		0	0	0	0	5	0	0	0	0	0	0
1082		0	0	0	0	1	0	0	0	0	0	0
1084		0	0	0	0	3	0	0	0	0	0	0
1085		0	0	0	0	2	0	0	0	0	0	0

Tabelle B.11: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

1084		0	0	0	0	3	0	0	0	0	0	0
1085		0	0	0	0	2	0	0	0	0	0	0
1088		0	0	0	0	2	0	0	0	0	0	0
1090		0	0	0	0	2	0	0	0	0	0	0
1092		0	0	0	0	3	0	0	0	0	0	0
1093		0	0	0	0	4	0	0	0	0	0	0
1094		0	0	0	0	1	0	0	0	0	0	0
1095		0	0	0	0	3	0	0	0	0	0	0
1096		0	0	0	0	2	0	0	0	0	0	0
1097		0	0	0	0	2	0	0	0	0	0	0
1098		0	0	0	0	1	0	0	0	0	0	0
1099		0	0	0	0	2	0	0	0	0	0	0
1100		0	0	0	0	2	0	0	0	0	0	0
1101		0	0	0	0	2	0	0	0	0	0	0
1102		0	0	0	0	5	0	0	0	0	0	0
1103		0	0	0	0	2	0	0	0	0	0	0
1104		0	0	0	0	1	0	0	0	0	0	0
1105		0	0	0	0	1	0	0	0	0	0	0
1106		0	0	0	0	1	0	0	0	0	0	0
1107		0	0	0	0	4	0	0	0	0	0	0
1108		0	0	0	0	1	0	0	0	0	0	0
1109		0	0	0	0	3	0	0	0	0	0	1
1110		0	0	0	0	3	0	0	0	0	0	0
1111		0	0	0	0	2	0	0	0	0	0	0
1112		0	0	0	0	1	0	0	0	0	0	0
1113		0	0	0	0	3	0	0	0	0	0	0
1114		0	0	0	0	3	0	0	0	0	0	0
1115		0	0	0	0	3	0	0	0	0	0	0
1116		0	0	0	0	1	0	0	0	0	0	0
1117		0	0	0	0	5	0	0	0	0	0	0
1118		0	0	0	0	1	0	0	0	0	0	0
1119		0	0	0	0	1	0	0	0	0	0	0
1120		0	0	0	0	1	0	0	0	0	0	0
1121		0	0	0	0	1	0	0	0	0	0	0
1122		0	0	0	0	3	0	0	0	0	0	0
1123		0	0	0	0	4	0	0	0	0	0	0
1124		0	0	0	0	1	0	0	0	0	0	0
1125		0	0	0	0	3	0	0	0	0	0	0
1126		0	0	0	0	2	0	0	0	0	0	0
1127		0	0	0	0	3	0	0	0	0	0	0
1128		0	0	0	0	1	0	0	0	0	0	0
1130		0	0	0	0	1	0	0	0	0	0	0
1132		0	0	0	0	1	0	0	0	0	0	0
1133		0	0	0	0	2	0	0	0	0	0	0
1134		0	0	0	0	2	0	0	0	0	0	0
1135		0	0	0	0	3	0	0	0	0	0	0
1136		0	0	0	0	1	0	0	0	0	0	0
1137		0	0	0	0	1	0	0	0	0	0	0
1138		0	0	0	0	1	0	0	0	0	0	0
1140		0	0	0	0	4	0	0	0	0	0	0
1141		0	0	0	0	2	0	0	0	0	0	0
1143		0	0	0	0	1	0	0	0	0	0	0
1144		0	0	0	0	1	0	0	0	0	0	0
1145		0	0	0	0	1	0	0	0	0	0	0
1146		0	0	0	0	2	0	0	0	0	0	0
1147		0	0	0	0	4	0	0	0	0	0	0
1149		0	0	0	0	1	0	0	0	0	0	0
1150		0	0	0	0	1	0	0	0	0	0	0
1151		0	0	0	0	3	0	0	0	0	0	0
1153		0	0	0	0	2	0	0	0	0	0	0
1154		0	0	0	0	1	0	0	0	0	0	0
1155		0	0	0	0	2	0	0	0	0	0	0
1156		0	0	0	0	5	0	0	0	0	0	0
1157		0	0	0	0	3	0	0	0	0	0	0
1158		0	0	0	0	4	0	0	0	0	0	0
1159		0	0	0	0	2	0	0	0	0	0	0
1161		0	0	0	0	1	0	0	0	0	0	0
1162		0	0	0	0	2	0	0	0	0	0	0
1163		0	0	0	0	3	0	0	0	0	0	0
1164		0	0	0	0	1	0	0	0	0	0	0
1165		0	0	0	0	1	0	0	0	0	0	0
1166		0	0	0	0	2	0	0	0	0	0	0
1168		0	0	0	0	2	0	0	0	0	0	0
1169		0	0	0	0	3	0	0	0	0	0	0
1171		0	0	0	0	1	0	0	0	0	0	0
1172		0	0	0	0	2	0	0	0	0	0	0
1173		0	0	0	0	1	0	0	0	0	0	0
1174		0	0	0	0	2	0	0	0	0	0	0
1175		0	0	0	0	2	0	0	0	0	0	0
1176		0	0	0	0	3	0	0	0	0	0	0
1177		0	0	0	0	3	0	0	0	0	0	0
1178		0	0	0	0	1	0	0	0	0	0	0
1179		0	0	0	0	1	0	0	0	0	0	0
1180		0	0	0	0	1	0	0	0	0	0	0
1181		0	0	0	0	2	0	0	0	0	0	0
1182		0	0	0	0	2	0	0	0	0	0	0
1183		0	0	0	0	1	0	0	0	0	0	0
1184		0	0	0	0	1	0	0	0	0	0	0
1185		0	0	0	0	1	0	0	0	0	0	0
1186		0	0	0	0	2	0	0	0	0	0	0
1189		0	0	0	0	4	0	0	0	0	0	0
1191		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.12: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

1177		0	0	0	0	3	0	0	0	0	0	0
1178		0	0	0	0	1	0	0	0	0	0	0
1179		0	0	0	0	1	0	0	0	0	0	0
1180		0	0	0	0	1	0	0	0	0	0	0
1181		0	0	0	0	2	0	0	0	0	0	0
1182		0	0	0	0	2	0	0	0	0	0	0
1183		0	0	0	0	1	0	0	0	0	0	0
1184		0	0	0	0	1	0	0	0	0	0	0
1185		0	0	0	0	1	0	0	0	0	0	0
1186		0	0	0	0	2	0	0	0	0	0	0
1189		0	0	0	0	4	0	0	0	0	0	0
1191		0	0	0	0	1	0	0	0	0	0	0
1193		0	0	0	0	1	0	0	0	0	0	0
1195		0	0	0	0	1	0	0	0	0	0	0
1196		0	0	0	0	4	0	0	0	0	0	0
1198		0	0	0	0	1	0	0	0	0	0	0
1199		0	0	0	0	4	0	0	0	0	0	0
1202		0	0	0	0	1	0	0	0	0	0	0
1204		0	0	0	0	1	0	0	0	0	0	0
1205		0	0	0	0	2	0	0	0	0	0	0
1208		0	0	0	0	1	0	0	0	0	0	0
1209		0	0	0	0	3	0	0	0	0	0	0
1210		0	0	0	0	3	0	0	0	0	0	0
1211		0	0	0	0	3	0	0	0	0	0	0
1212		0	0	0	0	1	0	0	0	0	0	0
1214		0	0	0	0	1	0	0	0	0	0	0
1216		0	0	0	0	1	0	0	0	0	0	0
1217		0	0	0	0	2	0	0	0	0	0	0
1218		0	0	0	0	2	0	0	0	0	0	0
1219		0	0	0	0	3	0	0	0	0	0	0
1220		0	0	0	0	2	0	0	0	0	0	0
1223		0	0	0	0	1	0	0	0	0	0	0
1224		0	0	0	0	2	0	0	0	0	0	0
1225		0	0	0	0	1	0	0	0	0	0	0
1226		0	0	0	1	0	0	0	0	0	0	0
1228		0	0	0	0	1	0	0	0	0	0	0
1229		0	0	0	0	2	0	0	0	0	0	0
1231		0	0	0	0	2	0	0	0	0	0	0
1234		0	0	0	0	1	0	0	0	0	0	0
1235		0	0	0	0	1	0	0	0	0	0	0
1236		0	0	0	0	3	0	0	0	0	0	0
1237		0	0	0	0	1	0	0	0	0	0	0
1239		0	0	0	0	3	0	0	0	0	0	0
1241		0	0	0	0	2	0	0	0	0	0	0
1242		0	0	0	0	1	0	0	0	0	0	0
1244		0	0	0	0	3	0	0	0	0	0	0
1245		0	0	0	0	1	0	0	0	0	0	0
1246		0	0	0	0	2	0	0	0	0	0	0
1247		0	0	0	0	3	0	0	0	0	0	0
1248		0	0	0	0	1	0	0	0	0	0	0
1249		0	0	0	0	2	0	0	0	0	0	0
1250		0	0	0	0	1	0	0	0	0	0	0
1251		0	0	0	0	1	0	0	0	0	0	0
1252		0	0	0	0	2	0	0	0	0	0	0
1253		0	0	0	0	2	0	0	0	0	0	0
1254		0	0	0	0	2	0	0	0	0	0	0
1259		0	0	0	0	2	0	0	0	0	0	0
1265		0	0	0	0	2	0	0	0	0	0	0
1266		0	0	0	0	1	0	0	0	0	0	0
1267		0	0	0	0	1	0	0	0	0	0	0
1268		0	0	0	0	1	0	0	0	0	0	0
1269		0	0	0	0	1	0	0	0	0	0	0
1270		0	0	0	0	1	0	0	0	0	0	0
1271		0	0	0	0	2	0	0	0	0	0	0
1274		0	0	0	0	2	0	0	0	0	0	0
1275		0	0	0	0	1	0	0	0	0	0	0
1276		0	0	0	0	1	0	0	0	0	0	0
1277		0	0	0	0	4	0	0	0	0	0	0
1278		0	0	0	0	3	0	0	0	0	0	0
1279		0	0	0	0	1	0	0	0	0	0	0
1282		0	0	0	0	3	0	0	0	0	0	0
1285		0	0	0	0	2	0	0	0	0	0	0
1286		0	0	0	0	1	0	0	0	0	0	0
1288		0	0	0	0	1	0	0	0	0	0	0
1289		0	0	0	0	2	0	0	0	0	0	0
1290		0	0	0	0	1	0	0	0	0	0	0
1291		0	0	0	0	2	0	0	0	0	0	0
1294		0	0	0	0	1	0	0	0	0	0	0
1296		0	0	0	0	1	0	0	0	1	0	0
1297		0	0	0	0	0	0	1	0	0	0	0
1298		0	0	0	0	1	0	0	0	0	0	0
1300		0	0	0	0	3	0	0	0	0	0	0
1302		0	0	0	0	4	0	0	0	0	0	0
1306		0	0	0	0	1	0	0	0	0	0	0
1307		0	0	0	0	1	0	0	0	0	0	0
1309		0	0	0	0	1	0	0	0	0	0	0
1311		0	0	0	0	2	0	0	0	0	0	0
1313		0	0	0	0	1	0	0	0	0	0	0
1323		0	0	0	0	1	0	0	0	0	0	0
1328		0	0	0	0	1	0	0	0	0	0	0
1330		0	0	0	0	2	0	0	0	0	0	0
1334		0	0	0	0	1	0	0	0	0	0	0
1337		0	0	0	0	1	0	0	0	0	0	0
1338		0	0	0	0	1	0	0	0	0	0	0
1341		0	0	0	0	1	0	0	0	0	0	0
1342		0	0	0	0	1	0	0	0	0	0	0
1343		0	0	0	0	2	0	0	0	0	0	0
1344		0	0	0	0	2	0	0	0	0	0	0
1348		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.13: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

1350		0	0	0	0	2	0	0	0	0	0	0
1352		0	0	0	0	1	0	0	0	0	0	0
1359		0	0	0	0	1	0	0	0	0	0	0
1360		0	0	0	0	1	0	0	0	0	0	0
1362		0	0	0	0	1	0	0	0	0	0	0
1364		0	0	0	0	2	0	0	0	0	0	0
1365		0	0	0	0	1	0	0	0	0	0	0
1367		0	0	0	0	3	0	0	0	0	0	0
1368		0	0	0	0	2	0	0	0	0	0	0
1369		0	0	0	0	1	0	0	0	0	0	0
1370		0	0	0	0	1	0	0	0	0	0	0
1372		0	0	0	0	1	0	0	0	0	0	0
1376		0	0	0	0	1	0	0	0	0	0	0
1377		0	0	0	0	1	0	0	0	0	0	0
1378		0	0	0	0	2	0	0	0	0	0	0
1380		0	0	0	0	2	0	0	0	0	0	0
1382		0	0	0	0	1	0	0	0	0	0	0
1383		0	0	0	0	2	0	0	0	0	0	0
1384		0	0	0	0	1	0	0	0	0	0	0
1385		0	0	0	0	1	0	0	0	0	0	0
1387		0	0	0	0	1	0	0	0	0	0	0
1388		0	0	0	0	1	0	0	0	0	0	0
1389		0	0	0	0	1	0	0	0	0	0	0
1392		0	0	0	0	2	0	0	0	0	0	0
1394		0	0	0	0	1	0	0	0	0	0	0
1396		0	0	0	0	3	0	0	0	0	0	0
1398		0	0	0	0	2	0	0	0	0	0	0
1399		0	0	0	0	1	0	0	0	0	0	0
1400		0	0	0	0	1	0	0	0	0	0	0
1402		0	0	0	0	2	0	0	0	0	0	0
1404		0	0	0	0	1	0	0	0	0	0	0
1405		0	0	0	0	2	0	0	0	0	0	0
1406		0	0	0	0	1	0	0	0	0	0	0
1407		0	0	0	0	1	0	0	0	0	0	0
1410		0	0	0	0	1	0	0	0	0	0	0
1411		0	0	0	0	1	0	0	0	0	0	0
1412		0	0	0	0	2	0	0	0	0	0	0
1414		0	0	0	0	1	0	0	0	0	0	0
1415		0	0	0	0	3	0	0	0	0	0	0
1418		0	0	0	0	1	0	0	0	0	0	0
1419		0	0	0	0	2	0	0	0	0	0	0
1420		0	0	0	0	3	0	0	0	0	0	0
1421		0	0	0	0	3	0	0	0	0	0	0
1423		0	0	0	0	1	0	0	0	0	0	0
1424		0	0	0	0	1	0	0	0	0	0	0
1425		0	0	0	0	2	0	0	0	0	0	0
1427		0	0	0	0	2	0	0	0	0	0	0
1428		0	0	0	0	1	0	0	0	0	0	0
1430		0	0	0	0	1	0	0	0	0	0	0
1431		0	0	0	0	1	0	0	0	0	0	0
1432		0	0	0	0	1	0	0	0	0	0	0
1433		0	0	0	0	1	0	0	0	0	0	0
1436		0	0	0	0	2	0	0	0	0	0	0
1439		0	0	0	0	1	0	0	0	0	0	0
1442		0	0	0	0	1	0	0	0	0	0	0
1443		0	0	0	0	4	0	0	0	0	0	0
1444		0	0	0	0	1	0	0	0	0	0	0
1445		0	0	0	0	2	0	0	0	0	0	0
1446		0	0	0	0	1	0	0	0	0	0	0
1447		0	0	0	0	1	0	0	0	0	0	0
1448		0	0	0	0	1	0	0	0	0	0	0
1449		0	0	0	0	1	0	0	0	0	0	0
1450		0	0	0	0	2	0	0	0	0	0	0
1451		0	0	0	0	2	0	0	0	0	0	0
1455		0	0	0	0	2	0	0	0	0	0	0
1457		0	0	0	0	2	0	0	0	0	0	0
1458		0	0	0	0	1	0	0	0	0	0	0
1460		0	0	0	0	4	0	0	0	0	0	0
1464		0	0	0	0	1	0	0	0	0	0	0
1465		0	0	0	0	1	0	0	0	0	0	0
1467		0	0	0	0	1	0	0	0	0	0	0
1469		0	0	0	0	2	0	0	0	0	0	0
1470		0	0	0	0	1	0	0	0	0	0	0
1471		0	0	0	0	2	0	0	0	0	0	0
1472		0	0	0	0	1	0	0	0	0	0	0
1481		0	0	0	0	1	0	0	0	0	0	0
1483		0	0	0	0	2	0	0	0	0	0	0
1484		0	0	0	0	1	0	0	0	0	0	0
1486		0	0	0	0	1	0	0	0	0	0	0
1487		0	0	0	0	1	0	0	0	0	0	0
1489		0	0	0	0	1	0	0	0	0	0	0
1491		0	0	0	0	1	0	0	0	0	0	0
1494		0	0	0	0	1	0	0	0	0	0	0
1495		0	0	0	0	1	0	0	0	0	0	0
1498		0	0	0	0	1	0	0	0	0	0	0
1499		0	0	0	0	1	0	0	0	0	0	0
1500		0	0	0	0	1	0	0	0	0	0	0
1501		0	0	0	0	2	0	0	0	0	0	0
1504		0	0	0	0	1	0	0	0	0	0	0
1507		0	0	0	0	2	0	0	0	0	0	0
1508		0	0	0	0	1	0	0	0	0	0	0
1509		0	0	0	0	2	0	0	0	0	0	0
1510		0	0	0	0	1	0	0	0	0	0	0
1511		0	0	0	0	1	0	0	0	0	0	0
1512		0	0	0	0	1	0	0	0	0	0	0
1513		0	0	0	0	1	0	0	0	0	0	0
1515		0	0	0	0	1	0	0	0	0	0	0
1518		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.14: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

1520		0	0	0	0	2	0	0	0	0	0	0
1522		0	0	0	0	1	0	0	0	0	0	0
1523		0	0	0	0	1	0	0	0	0	0	0
1526		0	0	0	0	2	0	0	0	0	0	0
1529		0	0	0	0	2	0	0	0	0	0	0
1534		0	0	0	0	2	0	0	0	0	0	0
1536		0	0	0	0	1	0	0	0	0	0	0
1537		0	0	0	0	1	0	0	0	0	0	0
1539		0	0	0	0	3	0	0	0	0	0	0
1540		0	0	0	0	1	0	0	0	0	0	0
1545		0	0	0	0	1	0	0	0	0	0	0
1546		0	0	0	0	2	0	0	0	0	0	0
1548		0	0	0	0	1	0	0	0	0	0	0
1549		0	0	0	0	2	0	0	0	0	0	0
1552		0	0	0	0	1	0	0	0	0	0	0
1553		0	0	0	0	2	0	0	0	0	0	0
1556		0	0	0	0	1	0	0	0	0	0	0
1557		0	0	0	0	1	0	0	0	0	0	0
1559		0	0	0	0	1	0	0	0	0	0	0
1561		0	0	0	0	1	0	0	0	0	0	0
1564		0	0	0	0	2	0	0	0	0	0	0
1567		0	0	0	0	1	0	0	0	0	0	0
1569		0	0	0	0	1	0	0	0	0	0	0
1570		0	0	0	0	1	0	0	0	0	0	0
1571		0	0	0	0	1	0	0	0	0	0	0
1572		0	0	0	0	1	0	0	0	0	0	0
1575		0	0	0	0	1	0	0	0	0	0	0
1576		0	0	0	0	1	0	0	0	0	0	0
1581		0	0	0	0	2	0	0	0	0	0	0
1584		0	0	0	0	1	0	0	0	0	0	0
1587		0	0	0	0	1	0	0	0	0	0	0
1590		0	0	0	0	1	0	0	0	0	0	0
1592		0	0	0	0	1	0	0	0	0	0	0
1594		0	0	0	0	2	0	0	0	0	0	0
1598		0	0	0	0	1	0	0	0	0	0	0
1600		0	0	0	0	1	0	0	0	0	0	0
1601		0	0	0	0	1	0	0	0	0	0	0
1608		0	0	0	0	1	0	0	0	0	0	0
1610		0	0	0	0	1	0	0	0	0	0	0
1611		0	0	0	0	1	0	0	0	0	0	0
1613		0	0	0	0	1	0	0	0	0	0	0
1616		0	0	0	0	1	0	0	0	0	0	0
1617		0	0	0	0	1	0	0	0	0	0	0
1618		0	0	0	0	1	0	0	0	0	0	0
1621		0	0	0	0	1	0	0	0	0	0	0
1622		0	0	0	0	1	0	0	0	0	0	0
1624		0	0	0	0	1	0	0	0	0	0	0
1625		0	0	0	0	1	0	0	0	0	0	0
1626		0	0	0	0	1	0	0	0	0	0	0
1629		0	0	0	0	2	0	0	0	0	0	0
1634		0	0	0	0	1	0	0	0	0	0	0
1635		0	0	0	0	1	0	0	0	0	0	0
1636		0	0	0	0	1	0	0	0	0	0	0
1641		0	0	0	0	2	0	0	0	0	0	0
1642		0	0	0	0	1	0	0	0	0	0	0
1643		0	0	0	0	2	0	0	0	0	0	0
1645		0	0	0	0	1	0	0	0	0	0	0
1646		0	0	0	0	2	0	0	0	0	0	0
1647		0	0	0	0	1	0	0	0	0	0	0
1649		0	0	0	0	2	0	0	0	0	0	0
1650		0	0	0	0	2	0	0	0	0	0	0
1654		0	0	0	0	1	0	0	0	0	0	0
1655		0	0	0	0	1	0	0	0	0	0	0
1656		0	0	0	0	1	0	0	0	0	0	0
1657		0	0	0	0	1	0	0	0	0	0	0
1658		0	0	0	0	1	0	0	0	0	0	0
1670		0	0	0	0	1	0	0	0	0	0	0
1673		0	0	0	0	1	0	0	0	0	0	0
1676		0	0	0	0	1	0	0	0	0	0	0
1677		0	0	0	0	1	0	0	0	0	0	0
1679		0	0	0	0	2	0	0	0	0	0	0
1682		0	0	0	0	1	0	0	0	0	0	0
1683		0	0	0	0	2	0	0	0	0	0	0
1684		0	0	0	0	1	0	0	0	0	0	0
1685		0	0	0	0	1	0	0	0	0	0	0
1689		0	0	0	0	1	0	0	0	0	0	0
1691		0	0	0	0	1	0	0	0	0	0	0
1693		0	0	0	0	1	0	0	0	0	0	0
1695		0	0	0	0	3	0	0	0	0	0	0
1696		0	0	0	0	1	0	0	0	0	0	0
1697		0	0	0	0	1	0	0	0	0	0	0
1698		0	0	0	0	1	0	0	0	0	0	0
1699		0	0	0	0	1	0	0	0	0	0	0
1701		0	0	0	0	1	0	0	0	0	0	0
1705		0	0	0	0	2	0	0	0	0	0	0
1707		0	0	0	0	2	0	0	0	0	0	0
1708		0	0	0	0	2	0	0	0	0	0	0
1709		0	0	0	0	1	0	0	0	0	0	0
1712		0	0	0	0	1	0	0	0	0	0	0
1715		0	0	0	0	1	0	0	0	0	0	0
1719		0	0	0	0	2	0	0	0	0	0	0
1725		0	0	0	0	1	0	0	0	0	0	0
1726		0	0	0	0	3	0	0	0	0	0	0
1729		0	0	0	0	1	0	0	0	0	0	0
1731		0	0	0	0	2	0	0	0	0	0	0
1732		0	0	0	0	1	0	0	0	0	0	0
1733		0	0	0	0	1	0	0	0	0	0	0
1735		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.15: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

1736		0	0	0	0	1	0	0	0	0	0	0
1739		0	0	0	0	1	0	0	0	0	0	0
1740		0	0	0	0	1	0	0	0	0	0	0
1743		0	0	0	0	1	0	0	0	0	0	0
1756		0	0	0	0	2	0	0	0	0	0	0
1760		0	0	0	0	1	0	0	0	0	0	0
1761		0	0	0	0	1	0	0	0	0	0	0
1762		0	0	0	0	1	0	0	0	0	0	0
1770		0	0	0	0	1	0	0	0	0	0	0
1771		0	0	0	0	1	0	0	0	0	0	0
1772		0	0	0	0	1	0	0	0	0	0	0
1773		0	0	0	0	1	0	0	0	0	0	0
1778		0	0	0	0	1	0	0	0	0	0	0
1780		0	0	0	0	1	0	0	0	0	0	0
1781		0	0	0	0	2	0	0	0	0	0	0
1782		0	0	0	0	1	0	0	0	0	0	0
1783		0	0	0	0	1	0	0	0	0	0	0
1785		0	0	0	0	1	0	0	0	0	0	0
1788		0	0	0	0	1	0	0	0	0	0	0
1789		0	0	0	0	2	0	0	0	0	0	0
1792		0	0	0	0	1	0	0	0	0	0	0
1793		0	0	0	0	1	0	0	0	0	0	0
1794		0	0	0	0	3	0	0	0	0	0	0
1796		0	0	0	0	1	0	0	0	0	0	0
1797		0	0	0	0	1	0	0	0	0	0	0
1800		0	0	0	0	1	0	0	0	0	0	0
1802		0	0	0	0	2	0	0	0	0	0	0
1803		0	0	0	0	1	0	0	0	0	0	0
1806		0	0	0	0	1	0	0	0	0	0	0
1813		0	0	0	0	1	0	0	0	0	0	0
1822		0	0	0	0	1	0	0	0	0	0	0
1827		0	0	0	0	1	0	0	0	0	0	0
1832		0	0	0	0	1	0	0	0	0	0	0
1840		0	0	0	0	1	0	0	0	0	0	0
1845		0	0	0	0	1	0	0	0	0	0	0
1849		0	0	0	0	1	0	0	0	0	0	0
1854		0	0	0	0	1	0	0	0	0	0	0
1858		0	0	0	0	1	0	0	0	0	0	0
1861		0	0	0	0	1	0	0	0	0	0	0
1864		0	0	0	0	1	0	0	0	0	0	0
1877		0	0	0	0	1	0	0	0	0	0	0
1879		0	0	0	0	1	0	0	0	0	0	0
1882		0	0	0	0	2	0	0	0	0	0	0
1884		0	0	0	0	1	0	0	0	0	0	0
1891		0	0	0	0	1	0	0	0	0	0	0
1892		0	0	0	0	1	0	0	0	0	0	0
1894		0	0	0	0	1	0	0	0	0	0	0
1896		0	0	0	0	1	0	0	0	0	0	0
1899		0	0	0	0	1	0	0	0	0	0	0
1907		0	0	0	0	1	0	0	0	0	0	0
1914		0	0	0	0	1	0	0	0	0	0	0
1915		0	0	0	0	1	0	0	0	0	0	0
1917		0	0	0	0	1	0	0	0	0	0	0
1926		0	0	0	0	1	0	0	0	0	0	0
1927		0	0	0	0	2	0	0	0	0	0	0
1931		0	0	0	0	1	0	0	0	0	0	0
1935		0	0	0	0	1	0	0	0	0	0	0
1938		0	0	0	0	1	0	0	0	0	0	0
1939		0	0	0	0	2	0	0	0	0	0	0
1943		0	0	0	0	1	0	0	0	0	0	0
1944		0	0	0	0	2	0	0	0	0	0	0
1953		0	0	0	0	1	0	0	0	0	0	0
1954		0	0	0	0	1	0	0	0	0	0	0
1963		0	0	0	0	3	0	0	0	0	0	0
1975		0	0	0	0	1	0	0	0	0	0	0
1977		0	0	0	0	1	0	0	0	0	0	0
1981		0	0	0	0	1	0	0	0	0	0	0
1984		0	0	0	0	1	0	0	0	0	0	0
1991		0	0	0	0	1	0	0	0	0	0	0
1995		0	0	0	0	1	0	0	0	0	0	0
1996		0	0	0	0	1	0	0	0	0	0	0
1999		0	0	0	0	1	0	0	0	0	0	0
2003		0	0	0	0	1	0	0	0	0	0	0
2004		0	0	0	0	1	0	0	0	0	0	0
2006		0	0	0	0	1	0	0	0	0	0	0
2016		0	0	0	0	1	0	0	0	0	0	0
2017		0	0	0	0	1	0	0	0	0	0	0
2019		0	0	0	0	1	0	0	0	0	0	0
2027		0	0	0	0	2	0	0	0	0	0	0
2029		0	0	0	0	1	0	0	0	0	0	0
2030		0	0	0	0	1	0	0	0	0	0	0
2039		0	0	0	0	1	0	0	0	0	0	0
2041		0	0	0	0	1	0	0	0	0	0	0
2053		0	0	0	0	1	0	0	0	0	0	0
2057		0	0	0	0	1	0	0	0	0	0	0
2058		0	0	0	0	2	0	0	0	0	0	0
2063		0	0	0	0	1	0	0	0	0	0	0
2072		0	0	0	0	1	0	0	0	0	0	0
2077		0	0	0	0	1	0	0	0	0	0	0
2079		0	0	0	0	1	0	0	0	0	0	0
2084		0	0	0	0	2	0	0	0	0	0	0
2091		0	0	0	0	1	0	0	0	0	0	0
2104		0	0	0	0	1	0	0	0	0	0	0
2108		0	0	0	0	1	0	0	0	0	0	0
2109		0	0	0	0	1	0	0	0	0	0	0
2111		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.16: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

2114		0	0	0	0	2	0	0	0	0	0	0
2121		0	0	0	0	2	0	0	0	0	0	0
2123		0	0	0	0	2	0	0	0	0	0	0
2125		0	0	0	0	1	0	0	0	0	0	0
2126		0	0	0	0	1	0	0	0	0	0	0
2127		0	0	0	0	1	0	0	0	0	0	0
2128		0	0	0	0	1	0	0	0	0	0	0
2139		0	0	0	0	1	0	0	0	0	0	0
2145		0	0	0	0	1	0	0	0	0	0	0
2150		0	0	0	0	1	0	0	0	0	0	0
2157		0	0	0	0	1	0	0	0	0	0	0
2159		0	0	0	0	1	0	0	0	0	0	0
2164		0	0	0	0	1	0	0	0	0	0	0
2169		0	0	0	0	1	0	0	0	0	0	0
2186		0	0	0	0	1	0	0	0	0	0	0
2189		0	0	0	0	1	0	0	0	0	0	0
2209		0	0	0	0	1	0	0	0	0	0	0
2214		0	0	0	0	1	0	0	0	0	0	0
2215		0	0	0	0	1	0	0	0	0	0	0
2219		0	0	0	0	1	0	0	0	0	0	0
2220		0	0	0	0	2	0	0	0	0	0	0
2228		0	0	0	0	1	0	0	0	0	0	0
2234		0	0	0	0	1	0	0	0	0	0	0
2238		0	0	0	0	1	0	0	0	0	0	0
2239		0	0	0	0	1	0	0	0	0	0	0
2240		0	0	0	0	1	0	0	0	0	0	0
2246		0	0	0	0	1	0	0	0	0	0	0
2248		0	0	0	0	1	0	0	0	0	0	0
2251		0	0	0	0	1	0	0	0	0	0	0
2252		0	0	0	0	1	0	0	0	0	0	0
2260		0	0	0	0	1	0	0	0	0	0	0
2262		0	0	0	0	1	0	0	0	0	0	0
2263		0	0	0	0	1	0	0	0	0	0	0
2268		0	0	0	0	1	0	0	0	0	0	0
2274		0	0	0	0	1	0	0	0	0	0	0
2299		0	0	0	0	1	0	0	0	0	0	0
2302		0	0	0	0	1	0	0	0	0	0	0
2317		0	0	0	0	1	0	0	0	0	0	0
2323		0	0	0	0	1	0	0	0	0	0	0
2330		0	0	0	0	1	0	0	0	0	0	0
2337		0	0	0	0	1	0	0	0	0	0	0
2341		0	0	0	0	1	0	0	0	0	0	0
2342		0	0	0	0	2	0	0	0	0	0	0
2345		0	0	0	0	1	0	0	0	0	0	0
2348		0	0	0	0	1	0	0	0	0	0	0
2355		0	0	0	0	1	0	0	0	0	0	0
2358		0	0	0	0	1	0	0	0	0	0	0
2361		0	0	0	0	2	0	0	0	0	0	0
2362		0	0	0	0	1	0	0	0	0	0	0
2381		0	0	0	0	1	0	0	0	0	0	0
2388		0	0	0	0	1	0	0	0	0	0	0
2390		0	0	0	0	1	0	0	0	0	0	0
2396		0	0	0	0	1	0	0	0	0	0	0
2424		0	0	0	0	1	0	0	0	0	0	0
2425		0	0	0	0	1	0	0	0	0	0	0
2436		0	0	0	0	1	0	0	0	0	0	0
2440		0	0	0	0	1	0	0	0	0	0	0
2442		0	0	0	0	1	0	0	0	0	0	0
2443		0	0	0	0	1	0	0	0	0	0	0
2445		0	0	0	0	1	0	0	0	0	0	0
2446		0	0	0	0	1	0	0	0	0	0	0
2448		0	0	0	0	1	0	0	0	0	0	0
2457		0	0	0	0	1	0	0	0	0	0	0
2490		0	0	0	0	1	0	0	0	0	0	0
2492		0	0	0	0	1	0	0	0	0	0	0
2507		0	0	0	0	1	0	0	0	0	0	0
2521		0	0	0	0	1	0	0	0	0	0	0
2533		0	0	0	0	1	0	0	0	0	0	0
2557		0	0	0	0	1	0	0	0	0	0	0
2561		0	0	0	0	1	0	0	0	0	0	0
2602		0	0	0	0	1	0	0	0	0	0	0
2611		0	0	0	0	1	0	0	0	0	0	0
2627		0	0	0	0	1	0	0	0	0	0	0
2628		0	0	0	0	1	0	0	0	0	0	0
2633		0	0	0	0	1	0	0	0	0	0	0
2645		0	0	0	0	1	0	0	0	0	0	0
2648		0	0	0	0	1	0	0	0	0	0	0
2682		0	0	0	0	1	0	0	0	0	0	0
2722		0	0	0	0	1	0	0	0	0	0	0
2744		0	0	0	0	1	0	0	0	0	0	0
2760		0	0	0	0	1	0	0	0	0	0	0
2777		0	0	0	0	1	0	0	0	0	0	0
2809		0	0	0	0	1	0	0	0	0	0	0
2834		0	0	0	0	1	0	0	0	0	0	0
2836		0	0	0	0	1	0	0	0	0	0	0
2878		0	0	0	0	1	0	0	0	0	0	0
2891		0	0	0	0	1	0	0	0	0	0	0
2921		0	0	0	0	1	0	0	0	0	0	0
2964		0	0	0	0	1	0	0	0	0	0	0
2987		0	0	0	0	1	0	0	0	0	0	0
2995		0	0	0	0	1	0	0	0	0	0	0
3009		0	0	0	0	1	0	0	0	0	0	0
3047		0	0	0	0	1	0	0	0	0	0	0
3058		0	0	0	0	1	0	0	0	0	0	0
3117		0	0	0	0	1	0	0	0	0	0	0
3139		0	0	0	0	1	0	0	0	0	0	0
3142		0	0	0	0	1	0	0	0	0	0	0
3155		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.17: gzip.source: maxIdle, Häufigkeit (Fortsetzung)

B Testfall 3: Simulation mit gzip.source

---

3179		0	0	0	0	1	0	0	0	0	0	0
3210		0	0	0	0	2	0	0	0	0	0	0
3229		0	0	0	0	1	0	0	0	0	0	0
3232		0	0	0	0	1	0	0	0	0	0	0
3250		0	0	0	0	1	0	0	0	0	0	0
3297		0	0	0	0	1	0	0	0	0	0	0
3306		0	0	0	0	1	0	0	0	0	0	0
3352		0	0	0	0	1	0	0	0	0	0	0
3380		0	0	0	0	1	0	0	0	0	0	0
3500		0	0	0	0	1	0	0	0	0	0	0
3532		0	0	0	0	1	0	0	0	0	0	0
3557		0	0	0	0	1	0	0	0	0	0	0
3568		0	0	0	0	1	0	0	0	0	0	0
3603		0	0	0	0	1	0	0	0	0	0	0
3624		0	0	0	0	1	0	0	0	0	0	0
3688		0	0	0	0	1	0	0	0	0	0	0
3695		0	0	0	0	1	0	0	0	0	0	0
3795		0	0	0	0	1	0	0	0	0	0	0
4160		0	0	0	0	1	0	0	0	0	0	0
4173		0	0	0	0	1	0	0	0	0	0	0
4180		0	0	0	0	1	0	0	0	0	0	0
4200		0	0	0	0	1	0	0	0	0	0	0
4287		0	0	0	0	1	0	0	0	0	0	0
4346		0	0	0	0	1	0	0	0	0	0	0
4692		0	0	0	0	1	0	0	0	0	0	0
4736		0	0	0	0	1	0	0	0	0	0	0
5122		0	0	0	0	1	0	0	0	0	0	0
5364		0	0	0	0	1	0	0	0	0	0	0
5670		0	0	0	0	1	0	0	0	0	0	0
6174		0	0	0	0	1	0	0	0	0	0	0
6394		0	0	0	0	1	0	0	0	0	0	0
8019		0	0	0	0	1	0	0	0	0	0	0
8745		0	0	0	0	1	0	0	0	0	0	0
10118		0	0	0	0	1	0	0	0	0	0	0
13704		0	0	0	0	1	0	0	0	0	0	0
15454		0	0	0	0	1	0	0	0	0	0	0
16223		0	0	0	0	1	0	0	0	0	0	0
16277		0	0	0	0	1	0	0	0	0	0	0
21467		0	0	0	0	1	0	0	0	0	0	0
21516		0	0	0	0	0	0	6	0	0	0	0
21553		0	0	0	0	0	0	0	0	0	0	3
21559		0	0	0	0	0	0	0	0	0	0	3
23358		0	0	0	0	1	0	0	0	0	0	0
23952		0	0	0	0	1	0	0	0	0	0	0
24382		0	0	0	0	1	0	0	0	0	0	0
33790		0	0	0	0	1	0	0	0	0	0	0
36302		0	0	0	0	1	0	0	0	0	0	0
41951		0	0	0	0	1	0	0	0	0	0	0
51516		0	0	0	0	1	0	0	0	0	0	0
52815		0	0	0	0	1	0	0	0	0	0	0
61760		0	0	0	0	1	0	0	0	0	0	0
78141		0	0	0	0	1	0	0	0	0	0	0

Tabelle B.18: gzip.source: maxIdleCycle, Häufigkeit(Fortsetzung)