Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра ВС

РАСЧЕТНО-ГРАФИЧЕСКОЕ ЗАДАНИЕ

по дисциплине «Функциональное и логическое программирование»

Вариант 19

Выполнил: студент гр. ИП-612

Самусев М.Р.

Проверил: Ефимов А.В.

Новосибирск 2018 г.

**ОГЛАВЛЕНИЕ**

[1. ЗАДАНИЕ 3](#_Toc533270392)

[2. АНАЛИЗ ВОЗМОЖНОСТЕЙ ПРОЦЕССОРОВ С МИКРОАРХИТЕКТУРОЙ ЭЛЬБРУС……………………………………………………………………………..3](#_Toc533270393)

[3. ПРИМЕР ФУНКЦИОНАЛЬНОЙ СТРУКТУРЫ ПРОЦЕССОРА 5](#_Toc533270394)

[4. ЧИСЛЕННЫЙ РАСЧЕТ И ПОСТРОИТЬ ГРАФИКИ ДЛЯ ФУНКЦИИ](#_Toc533270395) *[r](#_Toc533270395)*[(t) НАДЕЖНОСТИ И ФУНКЦИИ](#_Toc533270395)

[S(i, t) ГОТОВНОСТИ ЭВМ 9](#_Toc533270395)

[5. СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 12](#_Toc533270396)

# 1. ЗАДАНИЕ

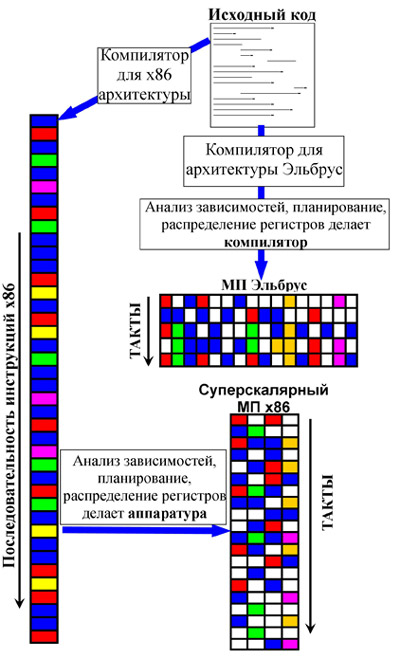
1. Произвести анализ возможностей процессоров с микроархитектурой Эльбрус. Привести пример функциональной структуры современного процессора.

2. Выполнить численный расчет и построить графики для функции *r*(*t*) надежности и функции S(i, t) готовности ЭВМ для следующих количественных характеристик:

– интенсивности отказов 10-2 1/*ч* ,

– интенсивности восстановления 1 1/*ч* .

1. Процессоры данной архитектуры сравниваются по производительности с современными процессорами фирмы интел в подсчете ГигаФлопс, при том, что имеют более скромные характеристики по частоте. Например при тактовой частоте всего в 1,3 ГГц Эльбрус 8С имеет производительность 250 гигафлопс (то есть 250 млрд операций в секунду) на операциях с одинарной точностью, что сравнимо с последними поколениями современного семейства процессоров Intel с архитектурой X86-64: 6-ядерный Intel Core i7-4930K (поколение Ivy Bridge) при тактовой частоте 3,7-4,2 ГГц имеет теоретический пик производительности в районе 177 гигафлопс.. Это достигается благодаря более продуманному распределению инструкций для процессора.



В традиционных архитектурах типа RISC или CISC (х86, PowerPC, SPARC, MIPS, ARM), на вход процессора поступает поток инструкций, которые рассчитаны на последовательное исполнение. Процессор может детектировать независимые операции и запускать их параллельно (суперскалярность) и даже менять их порядок (внеочередное исполнение). Однако динамический анализ зависимостей и поддержка внеочередного исполнения имеет свои ограничения: лучшие современные процессоры способны анализировать и запускать до 4-х команд за такт. Кроме того, соответствующие блоки внутри процессора потребляют заметное количество энергии.

В архитектуре «Эльбрус» основную работу по анализу зависимостей и оптимизации порядка операций берет на себя компилятор. Процессору на вход поступают т.н. «широкие команды», в каждой из которых закодированы инструкции для всех исполнительных устройств процессора, которые должны быть запущены на данном такте. От процессора не требуется анализировать зависимости между операндами или переставлять операции между широкими командами: все это делает компилятор, исходя из анализа исходного кода и планирования ресурсов процессора. В результате аппаратура процессора может быть проще и экономичнее.

Компилятор способен анализировать исходный код гораздо тщательнее, чем аппаратура RISC/CISC процессора, и находить больше независимых операций. Поэтому в архитектуре Эльбрус больше параллельно работающих исполнительных устройств, чем в традиционных архитектурах, и на многих алгоритмах она демонстрирует непревзойденную архитектурную скорость.

### 

### **Возможности архитектуры Эльбрус:**

* 6 каналов арифметико-логических устройств (АЛУ), работающих параллельно.
* Регистровый файл из 256 84-разрядных регистров.
* Аппаратная поддержка циклов, в том числе с конвейеризацией. Повышает эффективность использования ресурсов процессора.
* Программируемое асинхронное устройство предварительной подкачки данных с отдельными каналами считывания. Позволяет скрыть задержки от доступа к памяти и полнее использовать АЛУ.
* Поддержка спекулятивных вычислений и однобитовых предикатов. Позволяет уменьшить число переходов и параллельно исполнять несколько ветвей программы.
* Широкая команда, способная при максимальном заполнении задать в одном такте до 23 операций (более 33 операций при упаковке операндов в векторные команды).

1. Пример:

# 3. РАСЧЕТ ФУНКЦИЙ НАДЕЖНОСТИ И ГОТОВНОСТИ

Функция (или вероятность безотказной работы) относится к основным показателям надежности ЭВМ. Характеризует производительность ЭВМ на промежутке времени, то есть эта функция обеспечивает потенциально возможную производительность. Функцией надежности ЭВМ называется



где запись  означает вероятность того, что для всякого , принадлежащего промежутку времени  производительность  ЭВМ равна единице, т.е. равна потенциально возможной.

Функция  обладает следующими свойствами:

1.  Т.е. машина в момент начала функционирования находится в работоспособном состоянии.
2.  Событие, заключающееся в том, что ЭВМ работоспособна на конечном промежутке времени, является достоверным.
3.  для ;

Функцией ненадежности (или вероятностью отказа) ЭВМ называется



Функция  позволяет определить среднее время безотказной работы (средняя наработка до отказа). По определению, среднее время  безотказной работы ЭВМ и оценка  соответственно равны:



где – время безотказной работы -й машины, 

Интенсивностью отказов (лямбда-характеристикой) ЭВМ называется функция



Практически установлено, что зависимость интенсивности отказов от времени имеет место на периоде приработки ЭВМ. После приработки ЭВМ интенсивность отказов остается постоянной (до вхождения в предельное состояние или, по крайней мере, в течение промежутка времени, перекрывающего время морального старения). Следовательно, в нормальных условиях эксплуатации ЭВМ  а функция надежности и математическое ожидание времени *безотказной работы* соответственно равны:

.

– среднее число отказов, появляющихся в машине в единицу времени.

Подставляя известные нам данные получим следующую функцию для расчета надежности:

*r*(*t*)=exp(-1/\**t*)

*r*(*t*)=exp(-*t*/105);

Рассчитаем значения функции и построим график:

|  |  |
| --- | --- |
| *t, ч.* | *r(t)* |
| 0 | 1,000000 |
| 1 | 0,999990 |
| 5 | 0,999950 |
| 10 | 0,999900 |
| 100 | 0,999000 |
| 1000 | 0,990050 |
| 10000 | 0,904837 |
| 20000 | 0,818731 |
| 30000 | 0,740818 |
| 40000 | 0,670320 |
| 50000 | 0,606531 |
| 60000 | 0,548812 |
| 70000 | 0,496585 |
| 80000 | 0,449329 |
| 90000 | 0,406570 |
| 100000 | 0,367879 |
| 150000 | 0,223130 |
| 200000 | 0,135335 |
| 250000 | 0,082085 |
| 300000 | 0,049787 |
| 350000 | 0,030197 |
| 400000 | 0,018316 |
| 500000 | 0,006738 |
| 1000000 | 0,000045 |

Теперь рассчитаем значения функции готовности. Функция готовности ЭВМ

есть вероятность того, что (в условиях потока отказов и восстановлений) машина будет иметь в момент времени  производительность, равную единице, т.е. равную потенциально возможной.

Функция готовности ЭВМ обладает следующими свойствами:

1. 
2. 
3.   для 

Расчет будем производить по следующим формулам:

;

.

для начальных состояний ЭВМ , причем  соответствует состоянию отказа, а  – работоспособному состоянию машины, где *λ*=1/.

s (0, t)=10 / (10 + 1/105) – 10 / (10 + 1/105)\*exp((-t) \* (10 + 1/105))=

=0,999999 – 0,999999 \* exp((-t) \* 10,00001);

*s* (1, *t*)=10 / (10 + 1/105) + 0,000001 / (10 + 1/105)\*exp((-*t)* \* (10 + 1/105))=

=0,999999 + 0,00000099 \* exp((-t) \* 10,00001).

Рассчитаем значения функции и построим график:

|  |  |  |
| --- | --- | --- |
| *t*, ч. | *s*(0, *t*) | *s*(1, *t*) |
| 0 | 0,00000 | 0,99999999 |
| 0,001 | 0,00995 | 0,99999998 |
| 0,01 | 0,09516 | 0,99999990 |
| 0,05 | 0,39347 | 0,99999960 |
| 0,08 | 0,55067 | 0,99999944 |
| 0,1 | 0,63212 | 0,99999936 |
| 0,2 | 0,86466 | 0,99999913 |
| 0,3 | 0,95021 | 0,99999905 |

# 5. СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Хорошевский В.Г. Архитектура вычислительных систем. – М.: МГТУ им. Н.Э. Баумана, 2008. – 520 с.
2. Конспект лекций по курсу “Архитектура вычислительных систем”
3. Ким А. К., Перекатов В. И., Ермаков С. Г. Микропроцессоры и вычислительные комплексы семейства «Эльбрус». — СПб.: Питер, 2013. — 272 с