Федеральное государственное автономное образовательное учреждение

высшего образования

«Национальный исследовательский университет «Высшая школа экономики»

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ, ВЫПОЛНЯЕМОЕ В ПЕРИОД ПРАКТИКИ**

студенту 4 курса очной формы обучения

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Степанову Никите Андреевичу | | | | | | | |
| *(фамилия, имя, отчество при наличии)* | | | | | | | |
| образовательной программы | | | | «Информатика и вычислительная техника» | | | |
|  |  |  |  | *(наименование образовательной программы)* | | | |
| уровня | | бакалавриат | | | | | |
| по направлению/ специальности | | | | | 09.03.01 «Информатика и вычислительная техника | | |
|  | | | | | *(код и название направления/ специальности)* | | |
| факультета | | | Московский институт электроники и математики  им. А.Н.Тихонова | | | | |
| Вид практики | | | Производственная | | | | |
| Тип практики | | | Преддипломная | | | | |
| Сроки прохождения практики | | | | | с | | 01.04.2020 |
|  |  |  |  | | по | | 14.04.2020 |
| Место прохождения практики: | | | | | | Московский институт электроники и математики им. А.Н.Тихонова | |

|  |  |
| --- | --- |
| Цель прохождения практики[[1]](#footnote-1): | |
| Разработка прототипа программной реализации модели процесса слияния макро- и | |
| микро- операций в ядре CPU создаваемой в рамках ВКР | |
|  | |
|  | |
| Задачи практики1: | |
| Закрепление теоретических знаний, полученных в рамках подготовки ВКР | |
| Систематизация и обобщение материалов для подготовки ВКР | |
| Получение и отработка навыков проектирования программных продуктов | |
| Обработка полученных материалов и оформление отчета о прохождении практики | |
|  | |
| Содержание практики (вопросы, подлежащие изучению): | |
| 1. | Обзор, анализ и обоснование выбора программных средств и языков программирования |
| 2. | Поиск источников информации для анализа |
| 3. | Анализ принципов выполнения процедуры микро-слияния в ядре CPU |
| 4. | Разработка программного алгоритма модели работы блоков CPU при выполнении процедуры микро-слияния |

|  |  |
| --- | --- |
| 5. | Разработка графического интерфейса для программного обеспечения |
| 6. | Разработка прототипа программного обеспечения |
| 7. | Тестирование разработанного программного обеспечения |
|  |  |
| Планируемые результаты: | |
| 1. | Название и обоснование выбора основных источников информации |
| 2. | Графический интерфейс |
| 3. | Перечень принципов, параметров и условий слияния микро-операций |
| 4. | Алгоритм процедуры слияния микро-операций |
| 5. | Перечень программных средств и языков программирования для разработки |
| 6. | Прототип программного обеспечения |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Руководитель практики от НИУ ВШЭ | | | | | | | | | | | | | | | |
| доцент | | | | | |  | |  | | |  | Варнавский А.Н. | | | |
| *(должность)* | | | | | |  | | *(подпись)* | | |  | *(фамилия, инициалы)* | | | |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |
| СОГЛАСОВАНО | | | | | | | | | | | | | | | |
| Руководитель практики от профильной организации | | | | | | | | | | | | | | | |
| доцент | | | | | |  | |  | | |  | Иванова Е.М. | | | |
| *(должность)* | | | | | |  | | *(подпись)* | | |  | *(фамилия, инициалы)* | | | |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |
| Задание принято к исполнению | | | | | |  | 01.04.2020 | | | | | | | | |
| Студент | | | | | |  | |  | | |  | Степанов Н.А. | | | |
|  | | | | | |  | | *(подпись)* | | |  | *(фамилия, инициалы)* | | | |

Инструктаж по ознакомлению с требованиями охраны труда, техники безопасности, пожарной безопасности, а также правилами внутреннего трудового распорядка проведен 01.04.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Инструктирующий[[2]](#footnote-2): |  |  |  | Иванова Е.М. |
|  |  | *(подпись)* |  | *(фамилия, инициалы)* |
| Инструктируемый[[3]](#footnote-3): |  |  |  | Степанов Н.А. |
|  |  | *(подпись)* |  | *(фамилия, инициалы)* |

**ОП: «Информатика и вычислительная техника»**

**Уровень образования: бакалавриат**

**О Т Ч Е Т**

|  |  |  |
| --- | --- | --- |
| **по** | **преддипломной** | **практике** |
| *(производственной / научно-исследовательской / преддипломной)* | | |

|  |  |
| --- | --- |
| Выполнил студент гр. | БИВ161 |
| Степанов Никита Андреевич | |
| *(ФИО)* | |
|  | |
| *(подпись)* | |

|  |  |
| --- | --- |
| **Проверили:** |  |
| Доцент ДКИ Иванова Е.М. | |
| *(должность, ФИО руководителя от организации)* | |
|  |  |
| *(оценка)* | *(подпись)* |
|  |  |
| МП | *(дата)* |
| Доцент ДКИ Варнавский А.Н. | |
| *(должность, ФИО руководителя от факультета)* | |
|  |  |
| *(оценка)* | *(подпись)* |
|  |  |
|  | *(дата)* |

Оглавление

[Введение 3](#_Toc38037455)

[Содержательная часть 4](#_Toc38037456)

[1 Обзор, анализ и обоснование выбора программных средств и языков программирования 4](#_Toc38037457)

[2 Поиск источников информации для анализа 5](#_Toc38037458)

[3 Анализ принципов выполнения процедуры микро-слияния в ядре CPU 6](#_Toc38037459)

[4 Разработка программного алгоритма модели работы блоков CPU при выполнении процедуры микро-слияния 7](#_Toc38037460)

[5 Разработка графического интерфейса для программного обеспечения 9](#_Toc38037461)

[6 Разработка и отладка прототипа программного обеспечения 10](#_Toc38037462)

[7 Тестирование программного обеспечения 12](#_Toc38037463)

[Заключение 14](#_Toc38037464)

[Список использованных источников 15](#_Toc38037465)

[Приложения 16](#_Toc38037466)

[Приложение 1. Примеры готового программного кода на ассемблере. 16](#_Toc38037467)

Введение

Рассмотрим цель и задачи преддипломной практики.

Целью преддипломной практики является разработка программной реализации модели процедуры слияния макро- и микро- операций в ядре CPU.

Общими задачами преддипломной практики являются:

1. Закрепление теоретических знаний, полученных в рамках подготовки ВКР;
2. Систематизация и обобщение материалов для подготовки ВКР;
3. Получение и отработка навыков проектирования программных продуктов;
4. Обработка полученных материалов и оформление отчета о прохождении практики.

Задачами преддипломной практики, согласно индивидуальному заданию, являются:

1. Обзор, анализ и обоснование выбора программных средств и языков программирования;
2. Поиск источников информации для анализа;
3. Анализ принципов выполнения процедуры микро-слияния в ядре CPU;
4. Разработка программного алгоритма модели работы блоков CPU при выполнении процедуры микро-слияния;
5. Разработка графического интерфейса для программного обеспечения;
6. Разработка прототипа программного обеспечения
7. Тестирование разработанного программного обеспечения.

Содержательная часть

1. Обзор, анализ и обоснование выбора программных средств и языков программирования

Прежде всего для разработки программной модели нужно определить язык программирования для написания программной части. Среди современных и широко используемых языков можно выделить следующие языки программирования:

* С++
* Java
* Python
* C
* C#

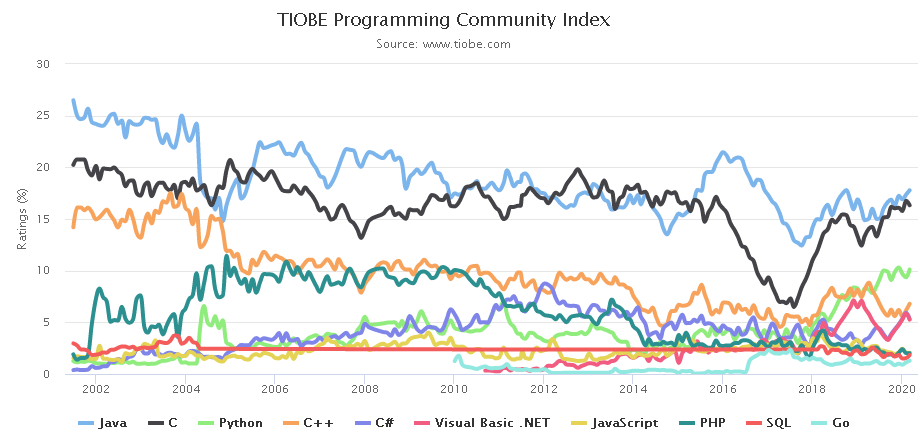


Рис.1. График популярности языков программирования.

Рассмотрим график популярности языков программирования на основе индекса TIOBE (рис. 2) [6]. Данный индекс оценивает популярность языка по частоте поисковых запросов, содержащих название языка. Согласно графику, можно наблюдать что на протяжении многих лет и на текущий момент самые популярные языки это C, Java и Python. Ввиду наличия более обширного опыта работы и удобства при работе с текстом наш выбор пал на Python. Стоит также отметить рост популярности языка за последние пару лет. Это связано с простотой и удобством языка что повлекло его широкое распространение в научной сфере. Однако у данного языка есть и недостатки в числе которых низкая скорость работы и интерпретируемость языка. Низкая скорость работы не повлияет на работу нашей модели ввиду её небольшого объёма. Однако интерпретируемость языка приводит к тому что нам необходимо найти способ создать исполняемый файл.

После поисков способа компиляции кода на Python в исполняемый файл формата .exe мы нашли библиотеку PyInstaller. Согласно документации данной библиотеки, она позволяет упаковать код на Python и все используемые зависимости в один исполняемый файл формата .exe.

Для реализации графического интерфейса нам необходимо использовать библиотеку, позволяющую реализовать графическую оболочку. Опираясь на наш опыт разработки интерфейсов под платформой Qt мы решили использовать библиотеку PyQt5. Данная библиотека позволяет реализовать интерфейс при помощи удобного редактора и связать его с кодом на Python.

1. Поиск источников информации для анализа

Изначально был определен вектор развития нашей программы модуляции слияния процессов – из-за отсутствия макро-слияния у процессоров архитектуры AMD как такового (и по нескольким другим причинам), мы решили моделировать слияние именно в ядре процессоров Intel. По этой причине поиск источников информации был всегда ориентирован на литературу по соответствующим процессорам.

В первую очередь, был рассмотрен документ, неизменно использующийся при работе с процессорами Intel – мануал «Intel 64 and IA-32 Architectures Optimization Reference» [1]. В нем содержится основная информация по работе процессоров архитектуры Intel разных поколений (например, Pentium M, Sandy Bridge и др.), а также данные по разработке, развитию и организации работы процессоров компанией Intel. К сожалению, в подобных мануалах содержится не полная информация, необходимая пользователям для изучения процессоров, т. к. не все данные возможно выносить на всеобщее обозрение из-за возможности кражи разработок конкурирующими компаниями. По этой причине слияние микро- и макро- операций в данном источнике описано вскользь, затрагивая основные темы, но не углубляясь в саму технологию. Но, с другой стороны, в данном источнике описано множество особенностей работы процессора, которые в дальнейшем помогли нам понять основную концепцию исполнения инструкций кода в блоках CPU.

Далее, было изучено несколько мануалов датского ученого Агнера Фога [2][3], которые в последствии стали основными источниками информации в нашей работе. В данных мануалах содержатся исследования слияния микро- и макро- операций в различных процессорах Intel, подкрепленные теорией и сводными таблицами по каждой архитектуре. Именно мануалы Агнера Фога в дальнейшем позволили точно смоделировать процессы слияния микро- и макро- операций и использовать множество различных настраиваемых параметров.

Дальнейший поиск источников заключался в нахождении информации для доработки некоторых аспектов процедур слияния и написания технической литературы. Влияние слияния на оптимизацию исполнения микро-операций было изучено в работе исследователей М. Тарама, А. Венката и Д. М. Тулсена из института Калифорнии «Mobilizing the Micro-Ops: Exploiting Context Sensitive Decoding for Security and Energy Efficiency» [5]. В другом исследовании C. Селио, Д. Даббелта, Д. А. Паттерсона и К. Асановича «The Renewed Case for the Reduced Instruction Set Computer: Avoiding ISA Bloat with Macro-Op Fusion for RISC-V» [4] выделена информация об улучшении работы процессора после слияния макро-операций в архитектурах типа RISC-V.

1. Анализ принципов выполнения процедуры микро-слияния в ядре CPU

В первую очередь, определим, что такое микро-слияние. Макро-операция попадает в процессор, декодируется и разбивается на несколько микро-операций. Среди них можно выделить основные:

* Чтение (Read/Load)
* Изменение (Modify)
* Вычисление и сохранение адреса (Adress)
* Запись (Write/Save)

Стоит отметить, что микро-операция Modify является основной и присутствует почти во всех декодируемых инструкциях. Для ее исполнения в процессоре отведены специальные блоки (порты) ALU. Остальные микро-операции обычно исполняются в своих собственных блоках, не связанных с ALU.

В разных архитектурах декодируемые макро-операции могут представлять собой набор разных микро-операций. Например, в архитектуре «Pentium M» в макро-операцию условного перехода LOOP входит 11 микро-операций типа изменения данных, а в архитектуре следующего поколения «Core 2» в ту же декодированную макро-операцию LOOP входит 7 микро-операции того же типа.

Согласно мануалам Агнера Фога, слияние микро-операций бывает трех типов:

1. Вычисление адреса + запись данных;

2. Чтение данных + изменение данных;

3. Комбинированный: чтение данных + изменение данных + запись данных;

Использование слияния микро-операций приводит к улучшению проводящей способности процессора, увеличивая его производительность и общую скорость исполнения инструкций.

1. Разработка программного алгоритма модели работы блоков CPU при выполнении процедуры микро-слияния

Для симуляции процедуры слияния микро-операций был выведен обобщенный алгоритм для проверки всех условий и выполнения слияния.

Рассмотрим алгоритм процедуры слияния микро-операций в ядре процессора. Процедура слияния позволяет двум и более микро-операциям, полученным в результате декодирования макро-операции, сливаться в одну целую операцию, что позволяет экономить место в буферах. Вследствие этого растет пропускная способность, а слитые операции в итоге выполняются на двух отдельных исполнительных блоках как не объединенные операции. Пример слияния можно наблюдать на рис. 2:

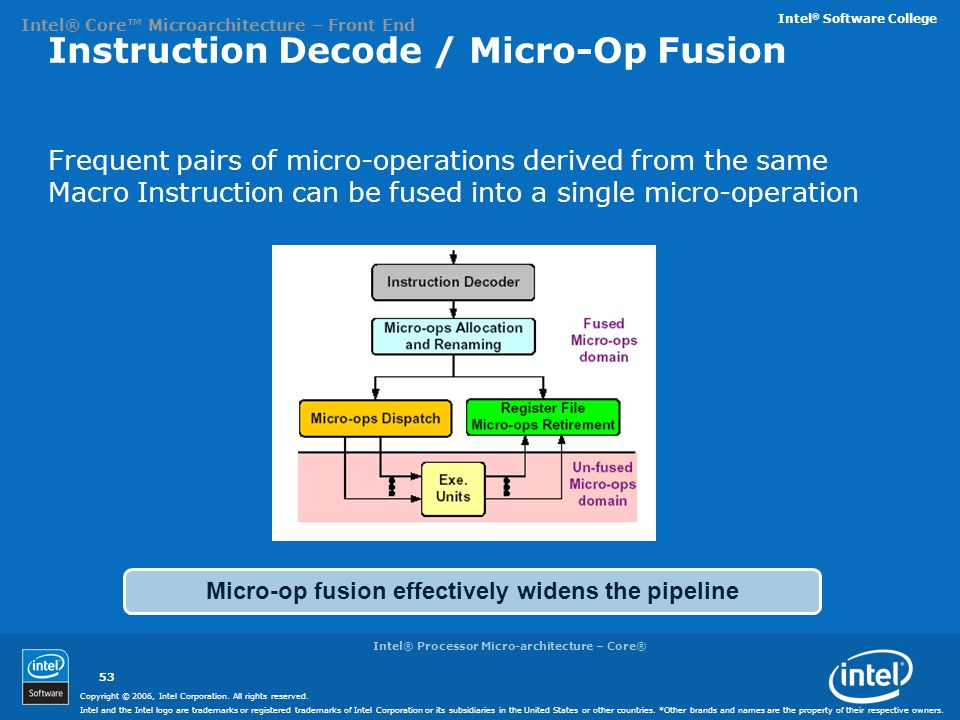


Рис. 2. Пример слияния микро-операций.

Микро-слиянию подвержены только два типа комбинаций: операции записи в память и операции чтения-изменения памяти. Таким образом, получаем три случая, в которых происходит слияние. В первом случае происходит объединение операции вычисления адреса в памяти и операции перемещения данных. Во втором происходит объединение операции чтения данных из памяти и операции преобразования данных. В случае ситуации, когда выполняются обе комбинации слияния - записи в память и вычисления адреса, чтение и изменение данных не происходит, однако в современных архитектурах есть возможность производить слияние третьего типа. В нашей программе мы создали возможность для пользователя попробовать все три типа, установив соответствующие настройки перед модуляцией слияния. Также, из-за разности исполняемых микро-операций в различных процессорах, пользователь может выбрать архитектуру, в соответствии с которой будет исполняться слияние микро-операций.

1. Разработка графического интерфейса для программного обеспечения

Графический интерфейс итоговой программы представляет собой стандартный набор элементов, созданных с помощью QT Designer. Слияние операций подсвечивается желтым цветом в таблицах (видно далее на рис. 4 и 5). В распоряжении пользователя находится набор различных настроек, устанавливаемых с помощью выпадающих списков, списков с множественным и одинарным выбором. Интерфейс прост и понятен даже пользователям, далёким от современных компьютерных тенденций.

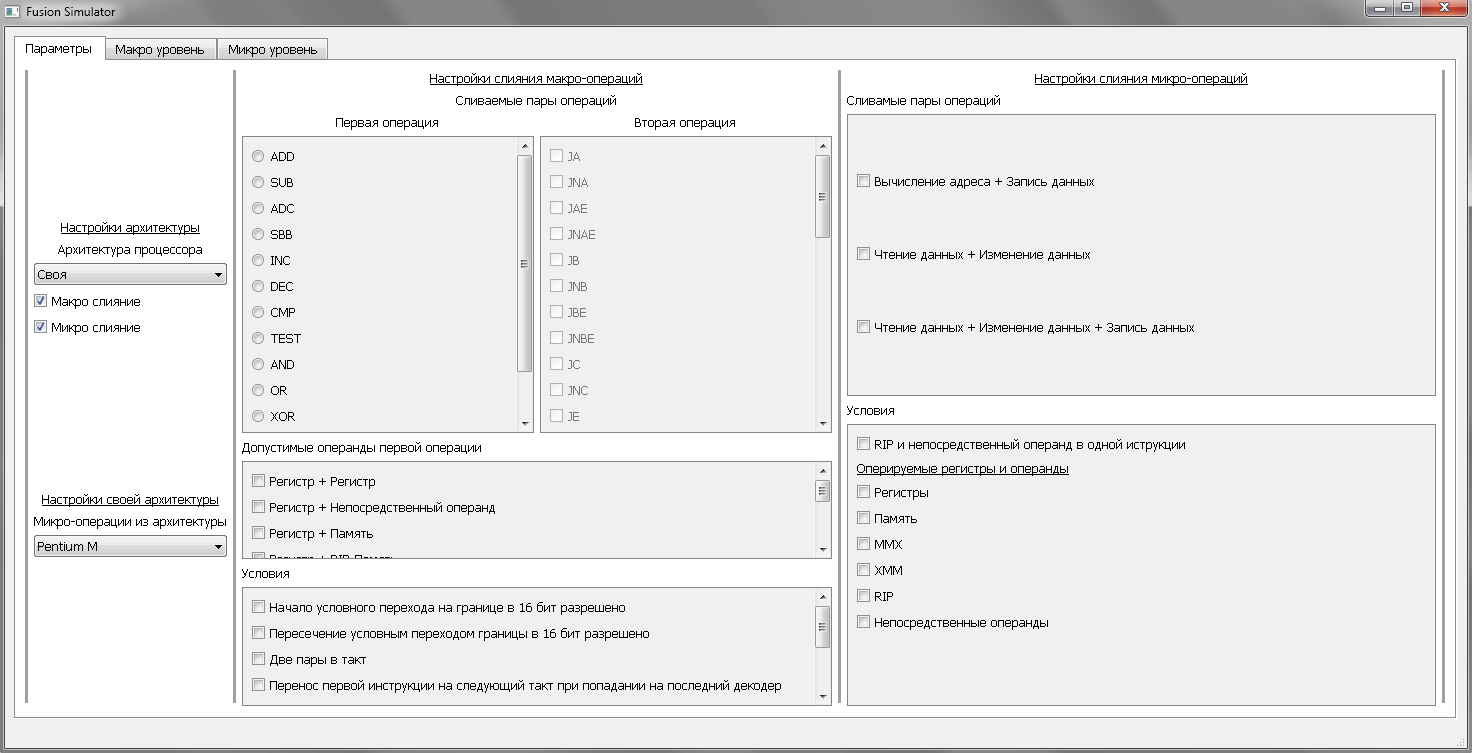


Рис. 3. Графический интерфейс программной модели.

Всё приложение поделено на три вкладки:

* Параметры, где пользователь может установить необходимые ему настройки микро- и макро- слияния;
* Макро уровень, где пользователь может записывать исполняемый на ассемблере код, производить модуляцию. Также в правой части отображается результат слияния;
* Микро уровень, где пользователь наблюдает полный спектр микро операций с их возможным слиянием;

Микро и макро операции отображаются в виде таблиц с номером такта, операндами и дополнительными параметрами, предоставляя пользователю полную информацию по их исполнению в блоках процессора.

1. Разработка и отладка прототипа программного обеспечения

В результате работ с использованием выбранных инструментов разработки был создан прототип программного обеспечения, позволяющий производить симуляцию процедуры слияния макро- и микро- операций в ядре CPU на архитектурах процессоров компании Intel, а также с применением собственных настроек симуляции.

Рассмотрим итог с точки зрения требований технического задания. Согласно заданию, программная модель должна позволять проследить выполнение процедуры слияния машинных инструкций на уровне макро- и микро- кода.

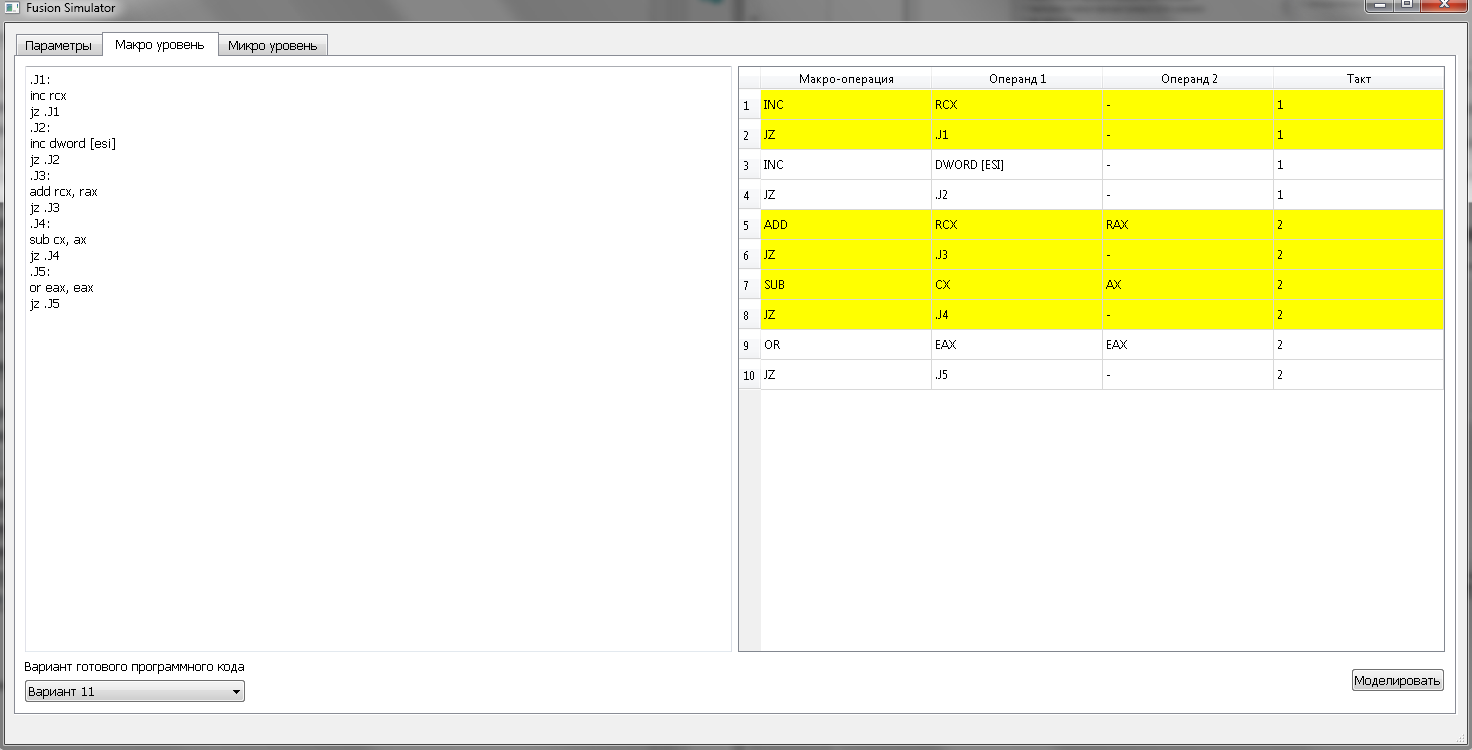


Рис. 4. Макро уровень симуляции.

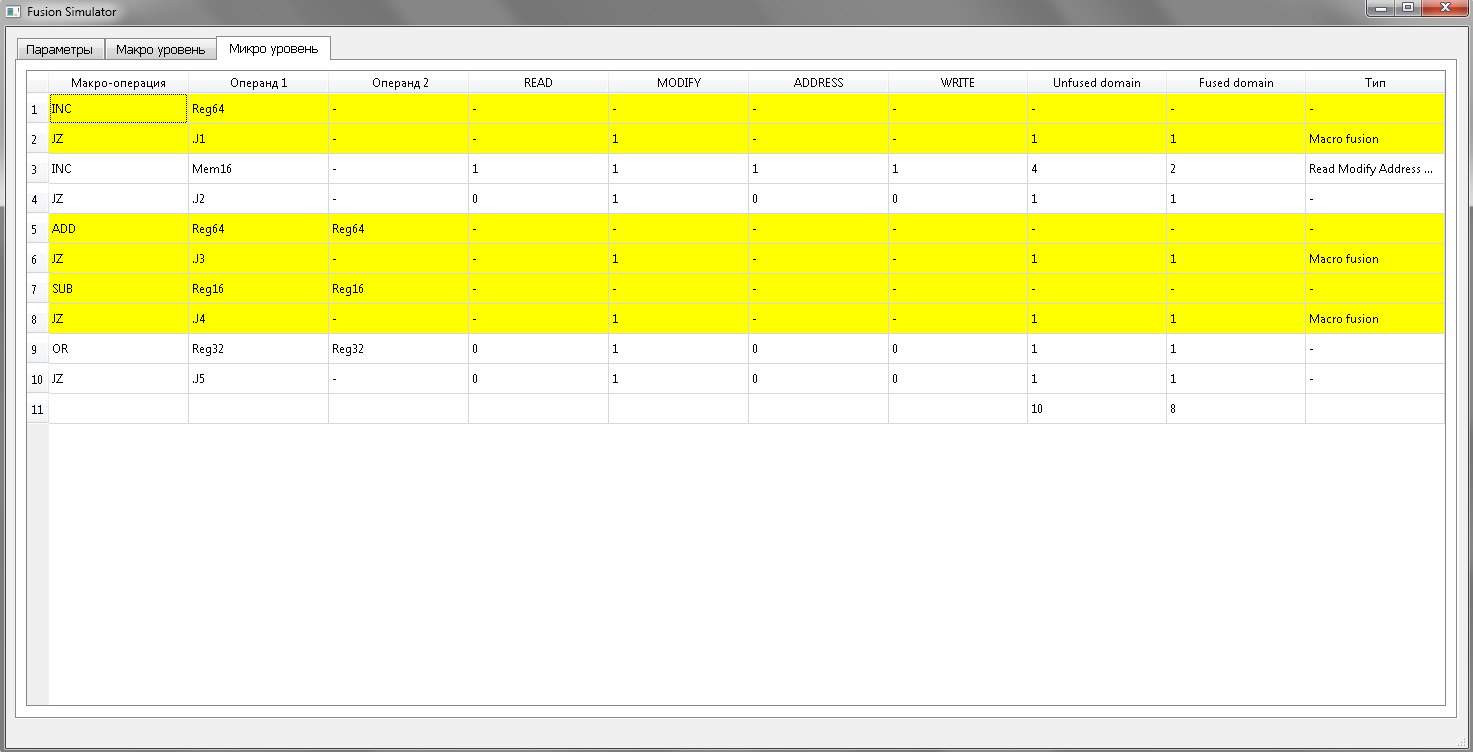


Рис. 5. Микро уровень симуляции.

Как можно наблюдать из рис.4 и 5 данное условие выполняется.

Также среди условий технического задания указано что модель должна быть спроектирована в виде функционально независимого модуля, запускаемого под ОС Windows без необходимости установки дополнительного ПО. Данное условие выполнятся за счет использования библиотеки PyInstall описанной ранее что позволяет создать единый исполняемый exe файл без зависимостей.

Следующее требование гласит что исходными данными является фрагмент программного кода на Ассемблере, включающем несколько инструкций подлежащих слиянию. Как можно видеть на рис. 4 данное условие выполняется. Программа может оперировать только с ограниченным числом команд ввиду сложной реализации симуляции отдельных инструкций и результатов их выполнения. Также предусмотрена проверка кода перед запуском симуляции с выделением проблемных фрагментов что позволяет легко исправлять ошибки в программном коде и не допускать недопустимые для симуляции варианты программного кода, а также код содержащий ошибки в написании.

Также программа должна позволять выбирать один из вариантов готового исходного кода на Ассемблере. На рис 4 представлен один из 15 вариантов исходного кода. Варианты готового программного кода позволяют продемонстрировать все возможности программы и различные варианты ситуаций, в которых те или иные настройки слияния оказывают наибольшее влияние (Приложение 1).

И последнее требование указывает на то что программа должна позволять выбирать архитектуру процессора для моделирования и выбирать какие инструкции подлежат слиянию. На рис. 3 можно наблюдать обширный список настроек и условий моделирования.

1. Тестирование программного обеспечения

Для тестирования приложения использовался код на ассемблере, включающий в себя всевозможные вариации микро- и макро- слияния и позволяющий в полной мере проявить функционал программы.

В ходе тестирования программного продукта было выявлено несколько проблем:

1. Невозможность реализации модуляции нескольких параметров слияния из-за сложности модуляции работы процессорных блоков. Подобные параметры в итоговом приложении были либо отброшены, либо их выполнение приведено в упрощенной форме.
2. Неточности в работе программного алгоритма. Так, например, было обнаружено, что не все случаи слияния микро-операций в программе обрабатываются верно. Из-за недостаточности информации по теме слияния операций, не удалось изначально определить ситуации, при которых микро-операции будут сливаться после уже произведённого слияния макро-операций. Для программной реализации данных случаев был улучшен алгоритм работы приложения и принято решение расширить таблицу микро-слияния на поле «Тип», что позволяет пользователю понять, в ходе какого исполнительного процесса произошло слияние операций.
3. Недостаточный функционал программы. Из-за влияния архитектуры на исполнение микро-операций и их слияние, ранее не было возможности точно производить модуляцию работы блоков процессора. Поэтому, после тестирования, был расширен функционал программы, добавлены необходимые данные для модуляции и добавлена возможность выбирать архитектуру, ориентируясь на которую программный алгоритм определяет места слияния микро-операций.

По результатам тестирования были устранены многие ошибки в программном коде, переделан интерфейс и добавлен функционал, изначально не предусмотренный техническим заданием для удобства моделирования. Примером такого функционала является частичная проверка исходного кода на ошибки.

Заключение

В ходе работ был определен и обоснован список инструментов разработки для решения поставленных задач.

Был определён и обоснован список источников, использованных для анализа алгоритмов слияния микро- и макро- операций.

Были выделены основополагающие принципы, условия и параметры слияния микро-операций в ядре CPU на основе анализа технической литературы.

Так же был разработан алгоритм модели работы блоков CPU при исполнении процедуры слияния микро операций в программной реализации.

Был разработан графический интерфейс для программного обеспечения доступный каждому пользователю.

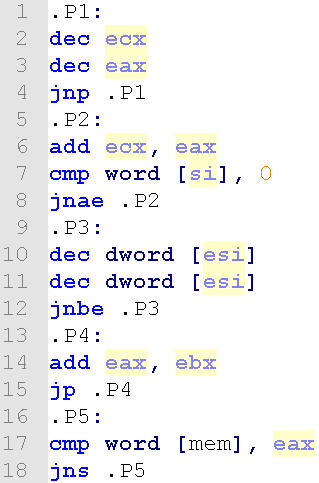
И в итоге был разработан прототип программной реализации модели процедуры слияния макро- и микро- операций в ядре CPU. Также, произведено тестирование данного прототипа, выявлены основные проблемы в работе приложения и алгоритма.

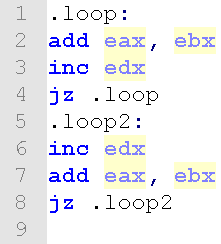
Список использованных источников

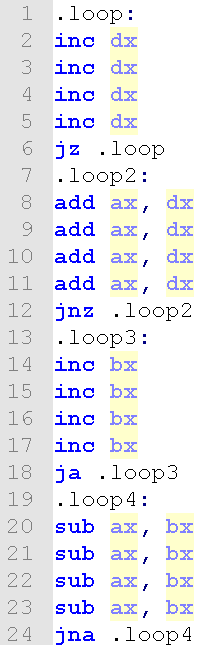
1. Intel 64 and IA-32 architectures optimization reference manual. Intel  Corporation, September 2019.
2. Agner Fog. The microarchitecture of Intel, AMD and VIA CPUs. Technical University of Denmark. September, 2018.
3. Agner Fog. Optimizing subroutines in assembly language. Technical University of Denmark. August, 2019.
4. C. Celio, D. Dabbelt, D. A. Patterson, K. Asanovic. The Renewed Case for the Reduced Instruction Set Computer: Avoiding ISA Bloat with Macro-Op Fusion for RISC-V. Electrical Engineering and Computer Sciences University of California. Berkeley, July 2016.
5. M. Taram, A. Venkat, D. M. Tullsen. Mobilizing the Micro-Ops: Exploiting Context Sensitive Decoding for Security and Energy Efficiency. University of California. San Diego, June 2018.
6. TIOBE - The Software Quality Company [Electronic resource]. URL: https://tiobe.com/tiobe-index/ (accessed: 09.04.2020).

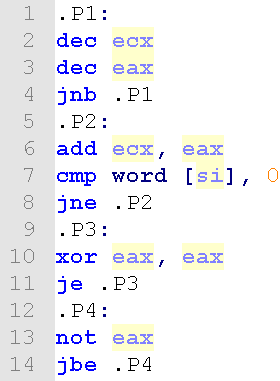
Приложения

Приложение 1. Примеры готового программного кода на ассемблере.









**ОП: «Информатика и вычислительная техника»**

**Уровень образования: бакалавриат**

**Д Н Е В Н И К**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **преддипломной** | | | **практики студента** | |
| *(производственной / научно-исследовательской / преддипломной)* | | |  | |
| группы | | | БИВ161 | |
| Степанова Никиты Андреевича | | | | |
| *(фамилия, имя, отчество)* | | | | |
| **Начат** | 01.04.2020 | **Окончен** | | 14.04.2020 |
|  | *(дата)* |  | | *(дата)* |

|  |  |  |
| --- | --- | --- |
| Оценка |  |  |
| Руководитель практики |  | Иванова Е.М. |
|  | *(подпись)* | *(должность, ФИО)* |

|  |  |
| --- | --- |
| Место прохождения практики | Московский институт электроники и математики им. А.Н. Тихонова |
| Должность, ФИО руководителя практики от организации | **Доцент ДКИ, Иванова Е.М.** |

**УЧЕТ ВЫПОЛНЕННОЙ РАБОТЫ**

|  |  |  |
| --- | --- | --- |
| Срок выполнения | Краткое содержание работы  *(заполняется практикантом)* | Отметка о выполнении работы  (комментарии, *подпись руководителя практики)* |
| 03.04.20 | Обзор, анализ и обоснование выбора программных средств и языков программирования |  |
| 04.04.20 | Поиск источников информации для анализа |  |
| 04.04.20 | Анализ принципов выполнения процедуры микро-слияния в ядре CPU |  |
| 05.04.20 | Разработка программного алгоритма модели работы блоков CPU при выполнении процедуры микро-слияния |  |
| 08.04.20 | Разработка графического интерфейса для программного обеспечения |  |
| 10.04.20 | Разработка и отладка прототипа программного обеспечения |  |
| 13.04.20 | Тестирование разработанного программного обеспечения |  |

|  |  |  |
| --- | --- | --- |
| Студент–практикант |  | Степанов Н. А. |
|  | *(подпись)* | *( ФИО)* |

**ОТЗЫВ**

**о работе студента с места прохождения практики**

студента МИЭМ НИУ ВШЭ группы БИВ161

Степанова Никиты Андреевича

За время прохождения преддипломной практики в период с 01.04.2020 по 14.04.2020 студент выполнил следующие задачи практики:

* Закрепил теоретические знания, полученные в рамках подготовки ВКР;
* Систематизировал и обобщил материалы для подготовки ВКР;
* Получил и отработал навыки проектирования программных продуктов.

А также выполнил следующий объем работ:

* Нашел и проанализировал источники информации, составил список соответствующей литературы
* Изучил принципы выполнения процедур микро-слияния в ядре CPU
* Разработал программный алгоритм процедуры слияния микро-операций
* Разработал графический интерфейс для программного обеспечения
* Разработал и произвел отладку прототипа программного обеспечения
* Произвел тестирование разработанного программного обеспечения, выявил основные ошибки и проблемные места в работе приложения

В ходе прохождения производственной практики студент показал свои знания и умения в области информационных технологий и проявил себя как ответственный, исполнительный и дисциплинированный специалист.

По результатам практики студент выполнил все поставленные задачи и заслуживает оценки \_\_ по 10-балльной системе.

|  |  |  |
| --- | --- | --- |
|  |  | Иванова Е.М. |
|  | *(подпись)* | *( ФИО)* |

1. В соответствии с программой практики. [↑](#footnote-ref-1)
2. Руководитель практики от профильной организации [↑](#footnote-ref-2)
3. Студент [↑](#footnote-ref-3)