ФОРМАЛЬНАЯ ВЕРИФИКАЦИЯ РЕЗУЛЬТАТОВ ВЫПОЛНЕНИЯ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ ЛОГИЧЕСКОЙ ОПТИМИЗАЦИИ

Логинова И. П.
Объединённый институт проблем информатики Национальной академии наук Беларуси
Минск, Республика Беларусь
E-mail: irilog@mail.ru

Предложена методика проверки корректной работы параллельных программ логической оптимизации на базе программного фреймворка, предоставляющего удобный инструментарий для проведения экспериментальных исследований, сравнения эффективности алгоритмов и формальной верификации. Фреймворк интегрирован с системой логической оптимизации функционально–структурных описаний дискретных устройств FLC-2, позволяет запускать программы оптимизации как на Windows, так и и Linux ОС, а также использовать инструмент формальной верификации FormalPro (ф.Mentor Graphics) для верификации результатов параллельных программ оптимизации.

Введение

На современном этапе развития в САПР СБИС появляются задачи и направления, обусловленные возрастающей сложностью элементной базы, новыми технологиями и динамически меняющимися подходами к проектированию. Инструменты САПР делают возможным автоматизацию многих аспектов процесса проектирования благодаря использованию эффективных алгоритмов и их программных реализаций. Большая часть алгоритмов из области логического синтеза, используется в составе различных САПР, в том числе отечественных [1]. Исторически сложилось так, что такие алгоритмы, реализованы на языках высокого уровня в виде программ с последовательно выполняемым кодом, без привязки к архитектуре вычислительных средств. В тоже время развитие высокопроизводительных многоядерных и многопроцессорных компьютерных систем приводит к повсеместному применению параллельных и распределенных вычислений, в том числе и в сфере САПР. Однако при разработке программ с использованием параллельных и распределенных вычислений возникают трудности с их отладкой и верификацией.

Разработка параллельной программы проводится либо на основе нового параллельного алгоритма, либо путем распараллеливания некоторых шагов существующего алгоритма. При любом подходе возникают проблема поиска и выявления ошибок, появляющихся в процессе выполнения параллельного кода. Для поддержки процесса отладки параллельных программ имеются инструменты отладки, основанные на сборе и автоматическом анализе информации по результату выполнения программ и предназначенные для многопоточных и OpenMP программ, например, Intel Thread Checker [2]. В дополнение к традиционным методам отладки выявление дефектов в параллельных программах также можно вести при помощи тестирования и верификации.

I. Верификация как основной инструмент проверки

На первый взгляд для проверки правильности работы параллельной программы можно сравнить результаты выполнения последовательной и параллельной версий. Но часто результаты параллельной версии могут отличаться от результатов последовательной версии программы. Одна из причин различия результатов - изменение порядка выполнения процессов и операций при вычислениях. Результаты параллельных расчетов могут не совпадать при различных запусках даже при одних и тех же начальных данных, поскольку условия запуска (например, загрузка и состояние вычислительной системы) могут влиять на порядок вычислений. Различие в порядке выполняемых вычислений - является одной из принципиальных проблем при разработке параллельных программ.

Традиционный способ проверки работы программы путём тестирования - как и любое тестирование, не гарантирует правильности полученного решения для любой программы и параллельной в том числе. Т.е. тестирование лишь выявляет некоторые ошибки, но не даёт гарантии их полного отсутствия. Поэтому при тестировании говорить о полученном правильном решении можно только для одного конкретного запуска программы. Альтернативой тестированию являются моделирование и верификация. Доказать отсутствие ошибок в программе позволяет проведение верификации различными методами непосредственно в процессе разработки программы. Поскольку результатом работы программ оптимизации объектов проектирования является получение оптимизированных описаний тех же объектов, появляется необходимость в доказательстве эквивалентности описаний объекта до и после оптимизации.На этапе логического проектирования для проверки корректности описаний исходных и результирующих объектов программ оптимизации применяют системы фор-

мальной верификации, входящие в состав САПР СБИС. Поскольку одним из направлений применения систем формальной верификации - проверка на эквивалентность исходного объекта и объекта, получаемого в результате оптимизации, этот подход можно применить для проверки объекта оптимизации, полученного в результате работы параллельной программы.

II. Предлагаемый подход

Таким образом, при работе с проектом в системе логического проектирования возникает задача контроля эквивалентности двух объектов проектирования, решением которой занимается система верификации FormalPro [3]. Система FormalPro может осуществлять проверку эквивалентности описаний схемы на уровне вентилей и RTL-модели, обеспечивает высокое качество верификации, поддерживает проекты практически неограниченного объема, включена в состав САПР Mentor Graphics и работает на ОС Sun Solaris и Linux. Для проведения экспериментальных исследований по сравнению эффективности различных программ логической оптимизации на различных операционных платформах, в том числе параллельных программ, а также проведения формальной верификации разработан фреймворк Plus. Он позволяет проводить формальную верификацию исходного и результирующего объекта оптимизации на уровне SF-описаний программой [4], в основе которой лежит реализация проверки выполнимости конъюнктивной нормальной формы. В фреймворке Plus реализована возможность проведения формальной верификации на уровне VHDL-описаний с использованием системы FormalPro [5]. Взаимодействие с FormalPro в фреймворке Plus реализовано посредством так называемой бесшовной интеграции функционала системы FormalPro с пользовательским интерфейсом Plus, что обеспечивает возможность работы с системой верификации на платформе Linux. Также в Plus реализована возможность проведения вычислительных экспериментов с программами логической оптимизации в Windows, Linux и на GRID кластере. Коротко, бесшовную интеграцию можно определить как обеспечение взаимодействия двух и более программных систем с «упрощением» пользовательского влияния на миграцию данных между системами. Основу такой интеграции в Plus составляет программная надстройка, которая позволяет организовать совместную работу двух ОС - Windows и Linux на основе программы для создания виртуальных машин VirtualBox (ф.Oracle)[5]. Реализовать интеграцию с FormalPro, запуск программ оптимизации в Linux, можно различными способами. Эти способы разделены на три класса (перечислены в порядке возрастания сложности): файлы импорта/экспорта, связь через API-интерфейс, полная (бесшовная) интеграция. Самым простым с точки зрения реализации является механизм обмена через структурированные файлы импорта/экспорта. В таком случае передача выполняется в соответствии с заранее согласованными форматами. Например, источник данных (фреймворк Plus или система FormalPro) формирует их, а затем через механизмы экспорта VirtualBox передает в файлы, которые читаются системой-приемником, т.е. FormalPro. Использование файлов экспорта/импорта в ряде случаев обеспечивает большую гибкость. поэтому большинство проектов по бесшовной интеграции выполняется именно таким способом - посредством включения элементов виртуализации.

Итак, инструментальными средствами фреймворка проводится: 1) проверка на эквивалентность исходного и оптимизированного описания, полученного в результате работы последовательной программы; 2) проверка на эквивалентность исходного и оптимизированного описания, полученного в результате работы параллельной программы; 3) проверка на эквивалентность оптимизированных описаний объектов, получаемых последовательной и параллельными программами оптимизации. Если на всех трех этапах проверки оказывается, что описания объектов эквивалентны, то можно утверждать, что параллельная программа логической оптимизации работает корректно и выдает правильное решение.

III. Список литературы

1. Бибило, П. Н. Система логической оптимизации функционально-структурных описаний цифровых устройств на основе продукционно-фреймовой модели представления знаний / П.Н. Бибило, В.И. Романов // Проблемы разработки перспективных микро- и наноэлектронных систем: сб. трудов IX Всероссийской научн.-техн. конф., Москва, 5 – 8 октября 2020 г. / ИППМ РАН; под общ. ред. акад. РАН А.Л. Стемпковского. – М.:, 2020. – Часть 4. - С. 9–16.
2. Reinders, J. Intel Threading Building Blocks: Outfitting C++ for Multi-Core Processor Parallelism / J. Reinders // Sebastopol: O'Reilly, 2007. 336 p.
3. Лохов, А. Функциональная верификация СБИС в свете решений Mentor Graphics/ А. Лохов // Электроника: наука, технология, бизнес. – 2004. – № 1. – С. 58 – 62
4. Черемисинова, Л. Д. Задачи верификации логических описаний комбинационных устройств / Л. Д. Черемисинова, Д. Я. Новиков // Информационные технологии и системы 2013 (ИТС 2013) : материалы международной научной конференции, БГУИР, Минск, Беларусь, 23 октября 2013 г.
5. Логинова, И. П. Верификация с использованием средств formalpro в системе логического проектирования заказных КМОП СБИС /И. П. Логинова // Новые информационные технологии в исследовании сложных структур : материалы Двенадцатой конференции с международным участием. 4-8 июня 2018 г. – Томск : Издательский Дом Томского государственного университета, 2018. – С. 72-73.