Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №3

по дисциплине Проектирование вычислительных систем

Студент: Саржевский Иван

Группа: P3402

г. Санкт-Петербург

2020 г.

Задача архитектурного проектирования

При проектировании определенного класса систем важным критерием оптимизации является минимизация зависимости по данным одних элементов от других. В основном речь идет о системах на кристалле, однако это может касаться и других классов систем в некоторых частных случаях, например когда несколько сервисов используют одну базу данных. В таком случае, обычной практикой является попытка объединения нескольких блоков с самой большой зависимостью по данным в один блок. Однако бывает очень сложно количественно сравнить степень зависимости по данным одних блоков от других, особенно на этапе проектирования системы, а слияние блоков на более поздних этапах будет означать внесение крупных изменений в проект, что повлечет за собой множество временных и финансовых издержек, не говоря о возможных рисках.

Самым естественным инструментом для решения такого рода проблемы кажется диаграмма потоков данных. Однако её недостаток в том, что она призвана описать зависимость одних блоков от других качественно, а не количественно. Иными словами, по ней можно однозначно сказать, что блок A зависит от блока B и блока C, однако очень сложно определить *в какой степени* A зависит от каждого из них. Теоретически, это можно исправить дополнительной нотацией над потоками данных. Но тем не менее, даже в таком случае информация о зависимости по данным будет сильно распределена по всей диаграмме, и понадобится определенное количество времени чтобы точно определить степень взаимной зависимости нескольких блоков.

Другим инструментом для визуализации подобной метрики может послужить диаграмма деятельности. Описав алгоритм в этой нотации, можно отследить какие акторы в каких случаях получают данные от каких акторов. Однако эта диаграмма еще меньше подходит для данной задачи, как минимум потому, что связь между акторами носит более алгоритмический характер без привязки к конкретным данным. Опять же, её можно расширить дополнительной нотацией, но тогда мы столкнемся с теми же недостатками - глядя на нее будет сложно быстро и точно определить степень зависимости акторов друг от друга в полной мере.

Целью данной лабораторной работы стала разработка инструмента, позволяющего быстро и наглядно оценить степень зависимости по данным различных блоков для принятия решений об оптимизации системы.

Описание модифицированного инструмента проектирования архитектурного уровня

Основным недостатком аналогов является то, что они являются *модуле-центричными*, то есть описывают блоки и потоки данных между ними. Появилось предположение, что для решения данной задачи инструмент должен быть *дата-центричным*, то есть строиться вокруг непосредственно данных, и описывать как ими распоряжаются существующие блоки.

Исходя из этого предположения были выделены основные сущности для разрабатываемой диаграммы. Во-первых, центральное место должны занимать непосредственно данные. Необходимо определить пул данных, которые циркулируют между блоками в рассматриваемой системе. Тогда появляются вполне естественные отношения между блоками и данными - для каждой единицы данных блок может выступать в качестве *продьюсера*, то есть блока, генерирующего эту единицу данных, или *консьюмера*, то есть блока, использующего эту единицу данных.

Таким образом, было решено, что диаграмма будет иметь следующий вид:

- В центре находится непосредственно стек данных
- Вокруг него расположены блоки системы
- Справа от каждой единицы данных расположены стрелки к её консьюмерам
- Слева от каждой единицы данных расположены стрелки от её продьюсеров
- Для удобства данные, расположенные подряд и идущие к одному консьюмеру или приходящие от одного продьюсера объединяются в блоки
- Предлагается использовать цветовые разграничители для блоков, это не обязательно, однако увеличивает читаемость диаграммы в разы, как будет видно из примера

Пример использования модифицированного инструмента архитектурного проектирования

На рисунке 1 представлен пример использования такой диаграммы для системы на кристалле, которая выполняет два вида обработки входных сигналов с использованием блоков аппаратных ускорителей.

В данной системе есть такие блоки:

- Контроллер пользовательского ввода: ялвяется продьюсером для сигналов X1-X7 и CMD и консьюмером сигнала STATE1
- Управляющее устройство: продьюсер сигналов STATE1 и STATE2, консьюмер сигнала CMD
- Обработчик сигнала 1: продьюсер сигналов Y1-Y5 и консьюмер сигналов X1-X7, C1 и STATE2
- Обработчик сигнала 2: продьюсер сигналов Y1-Y5 и консьюмер сигналов X1-X5, C1-C3 и STATE2
- Контроллер памяти: продьюсер сигналов C1-C3 и консьюмер сигнала STATE2
- Контроллер вывода: консьюмер сигналов Y1-Y5 и STATE1

Простыми словами - система получает извне массив из 7-ми чисел и управляющую команду. В зависимости от управляющей команды упровляющее устройство генерирует служебные сигналы, которые активируют контроллер памяти, контроллер вывода и один из двух обработчиков входного сигнала. Первая функция возвращает 5 выходных значений на основании семи входных и одной константы из памяти, а вторая возвращает 5 значений на основании пяти входных и трех констант из памяти.

Так как прямоугольник самого большого консьюмера группы данных идет к ней ближе всего, беглого взгяда на диаграмму достаточно, чтобы понять, что от "зеленой"группы данных (входных данных, полученных от контроллера пользовательского ввода) больше всего зависит "красный"блок (обработчик сигнала 1), а "желтый"блок (обработчик сигнала 2) зависит от нее в меньшей степени. Если же посмотреть на "фиолетовую"группу данных (данные полученные из памяти), то можно сразу увидеть обратную ситуацию - второй обработчик сигнала зависит от этой группы данных сильнее первого.

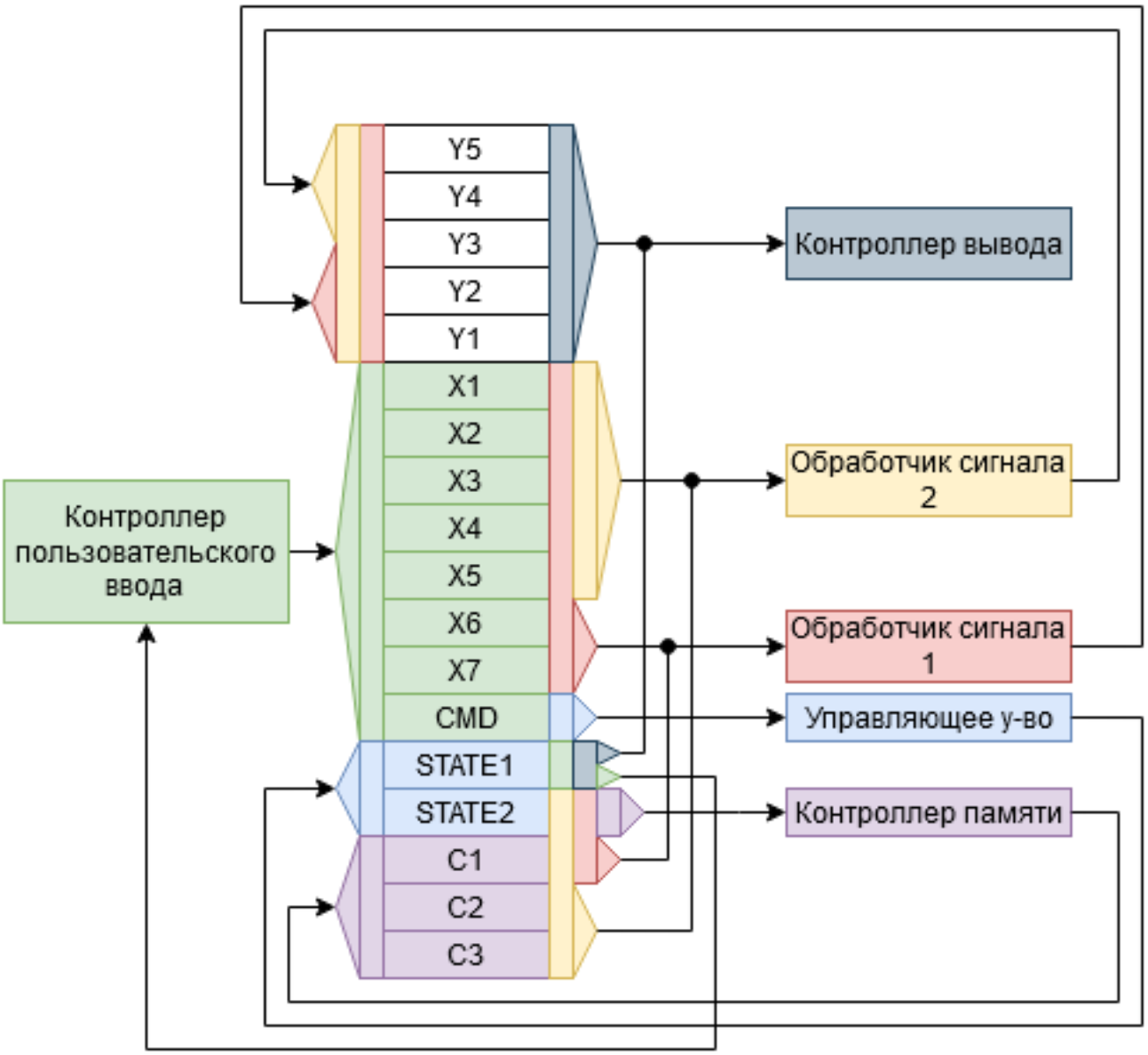

Рис. 1: Пример использования диаграммы

Критерии оценки инструмента архитектурного проектирования

- Наглядность (отношения степени зависимости по данным одного блока от другого)
- Сложность реализации
- Объем информации (чем меньше не относящейся к решаемой задаче информации представлено, тем лучше)
- Обобщенность (количество проблем, которые можно решить помимо поставленной)
- Масштабируемость (насколько падает читаемость при увеличении сложности проекта)
- Абстракция (дает ли информацию в отрыве от знания функций блоков)

Сравнительный анализ с другими инструментами в соответствии со сформулированным критериями

Для сравнения выбраны: разработанная диаграмма зависимости по данным, диаграмма потоков данных и диаграмма активности. Оценка производилась по десятибальной шкале.

Критерий	Зависимости по данным	Потоков данных	Активности
Наглядность	9	5	3
Сложность реализации	4	9	6
Объем информации	10	8	4
Обобщенность	2	9	10
Масштабируемость	1	9	5
Абстракция	10	4	1

Вывод

В ходе выполнения данной лабораторной работы был разработан новый инструмент архитектурного проектирования - диаграмма зависимости по данным.

В ходе сравнительного анализа, оказалось что этот инструмент может быть полезен, однако для относительно узкого класса задач и не для любой системы. Он действительно позволяет быстро и интуитивно дать количественную оценку степени зависимости по данным одних блоков от других, однако имеет ряд серьезных недостатков.

В первую очередь можно назвать недостатком то, что он подходит для решения одной, в общем случае узкой, задачи оптимизации системы по пересылкам данных. Однако, учитывая что хороших аналогов для решения такой задачи нет, можно позиционировать это как узконаправленный инструмент.

Гораздо более серьезный недостаток - работа только с равноценными по размеру данными. Если одна ячейка в стеке данных будет представлять разное количество фактических данных, то наглядность метода потеряется. Теоретически можно представлять данные, занимающие больший объем большим количеством ячеек, однако при большом разбросе в размере данных диаграмма рискует стать слишком громоздкой и нечитаемой.

Что является третьим существенным недостатком данной диаграммы - при усложнении системы, в контексте рассматриваемой проблемы понимаем под этим увеличение потоков

данных между узлами, даже при однородных данных диаграмма может стать слишком сложной и потерять свою читаемость.

Вышеперечисленные недостатки сильно ограничивают область применения данного подхода, однако в системах, подобных описанной в примере, такой инструмент может быть успешно использован для анализа архитектуры с точки зрения завимостей по данным между блоками для последующих оптимизаций.

Рис. 2: Лепестковая диаграмма сравнения инструментов архитектурного проектирования