Краткие сообщения

УДК 621.399

А.В. Мухлаев, М.Д. Сеченов

Адаптивный интерфейс пользователя САПР

Диалог разработчика с САПР осуществляется, как правило, через пользовательское меню. Традиционное меню, наряду с очевидными достоинствами, имеет и ряд недостатков, таких, например, как большой размер окон, отводимых под команды меню, что "скрывает" от пользователя значительную часть рабочего чертежа; несколько уровней вложенности окон, что затрудняет перемещение и отслеживание текущего положения в меню и т.д.

Предлагается адаптивный интерфейс пользователя САПР, имеющий возможность автоматической настройки и оптимизации формы меню под особенности и стиль работы с САПР конкретного индивидуального разработчика.

Для достижения указанной цели предлагаются следующие приемы. Анализ показывает, что пользователь САПР в процессе работы использует в среднем около 30 процентов команд, предлагаемых меню. Этот факт обусловлен типом задач, решаемых конструктором, его стилем работы и наличием схожих по функциональному назначению команд (например: save - save as, line - polyline). Поэтому предлагается исключать из числа отображаемых на экране команд меню те опции, которые не были использованы пользователем более чем N сеансов подряд. При этом параметр N задается самим разработчиком. Доступ же к "невидимым" в текущий момент командам может быть осуществлен через одну дополнительную опцию, которая получила имя anover command. Такой прием позволяет уменьшить размер окон меню (до 70 процентов), что освобождает дополнительные видимые области рабочего чертежа.

Второй и третий приемы связаны с оптимизацией перемещения пользователя по окнам и уровням вложенности интерфейса. Предлагается численно оценивать количество переходов между опциями высшего уровня иерархии (например: file, editor, command и т.д.) и располагать по соседству те из них, между которыми чаще всего наблюдаются переходы. Это позволяет сократить время перемещения по меню, особенно если для этого используются клавиши управления курсором. Еще один прием связан с оценкой частот перехода между командами меню. Так, например, чаще всего после команды save используется команда quit, а после команды erase - команда redraw. Порядок использования команд в первую очередь зависит от стиля работы пользователя и, в значительной мере, индивидуален. Поэтому предлагается во время каждого сеанса работы оценивать частоту перехода из текущей активной команды к остальным и после окончания выполнения текущей команды автоматически активизировать (подсвечивать) ту команду меню, частота перехода к которой, полученная на основе предыдущих вычислений, является максимальной.

Частота перехода Pij из команды i к команде j оценивается по формуле,

$$Pij=Kij\backslash Ki,$$

где Kij - количество переходов из команды i к команде j,

Ki - общее количество переходов из команды i.

Матрица частот переходов хранится в специальном системном файле, является индивидуальной для каждого конкретного пользователя и обновляется во время каждого сеанса работы. Указанный подход позволяет значительно сократить время перемещения по окнам и командам меню.

Приведенная выше концепция построения пользовательского интерфейса САПР была реализована на языке AUTOLISP для пакета программ AUTOCAD 12.0. В качестве инициализирующего меню используется стандартный интерфейс acad.mnu. Проведенные сеансы работы с системой AUTOCAD 12.0 различными пользователями показали, что предлагаемый подход позволяет оптимизировать и настроить стандартный интерфейс acad.mnu под конкретного разработчика, повышает эффективность и удобство его работы.

УДК 658.512

Н.Н. Рябец, М.Н. Рябец

Исследование алгоритма упорядочения каналов трассировки СБИС

Проблема выявления последовательности проведения трасс в каналах особенно остро стоит при проектировании СБИС.

Предлагается новый подход к решению этой проблемы. Для удачного размещения элементов на кристалле необходимо, по крайней мере, выполнить следующие процедуры: квазиоптимальное размещение блоков (набор связных элементов или библиотечных наборов элементов); назначение области для дальнейшей трассировки соединений; формирование канального графа; проведение глобальной трассировки, используя известные алгоритмы, например, построения дерева Штейнера; упорядочение каналов (назначение приоритетов каналам); детальная трассировка. Многие этапы этой задачи решены [1,2]. Отмечается, что поскольку каналы взаимосвязаны, то после фиксации трассировки отдельного канала его структура приводит к существенным ограничениям для трассировки последующих соединений.

В предлагаемом алгоритме используются известные структурные графы. Считаем, что основной задачей при распределении соединений по каналам является минимизация общей площади кристалла. Также необходимо учесть временные затраты и общую длину соединений. Оценку кристалла можно выразить как H= lsd + C(Lsd - lsd), где Lsd, lsd, соответственно длина критического пути по высоте и ширине кристалла. Коэффициент C расположен между 0 и 1.Как правило, C=0.95. В случае, если C=1, все сегменты каналов будут реализованы, когда C=0, трассировка затруднительна и алгоритм сводится к алгоритму упаковки блоков. Задача распределения цепей по каналам в соответствии с приведенной оценкой решается следующим образом.

При определенном начальном размещении блоков (элементов) формируем канальный граф реализаций. Для примера используем двумерное представление блоков. Тогда алгоритм можно представить следующим образом.

Шаг 1. Проводим горизонтальные и вертикальные линии границ элементов. Если они соприкасаются с линиями других элементов, то они пересекают эти линии.

Шаг 2. Если, таким образом, выполненные действия приводят к перекрытию областей, незанятых физическими элементами, то они совмещаются.

Шаг 3. Определяем точки пересечения и запоминаем их.

Шаг 4. Определяем корень дерева разбиения со всеми элементами и линии сегментов.

Шаг 5. Проводим разбиение. (В начальном случае - корня дерева).

Шаг 6.Генерируем канальный граф.

Алгоритм был исследован на ряде реальных и случайно генерируемых СБИС. Исследовались два кристалла: первый содержит 10 блоков (элементов), количество цепей - 203, количество контактов - 351; второй содержит 33 блока (элемента), количество цепей - 123, количество контактов - 211. Результаты экспериментов