являющиеся вершинами полученных выпуклых многогранников. Применяя к множествам вершин утверждение 1 и учитывая, что для любой тройки вершин, по крайней мере, две принадлежат одному многограннику, получаем, что существует разделяющая плоскость, проходящая через грань либо через вершину одного многогранника и ребро другого. Отсюда следует, что сложность алгоритма построения разделяющей плоскости для выпуклых многогранников не превосходит порядок $(n+m)^3$.

Утверждение 2. Если два выпуклых многогранника отделимы, то найдется два непараллельных ребра этих многогранников такие, что плоскость, проходящая через одно из ребер параллельно другому, будет разделяющей. (Заметим, что выбор двух ребер, принадлежащих одной из граней выпуклой оболочки, определяет плоскость, проходящую через грань.)

Учитывая, что два множества отделимы тогда и только тогда, когда отделимы их выпуклые оболочки, а выпуклая оболочка конечного множества точек является выпуклым многогранником, утверждение 2 будет обоснованием для алгоритма, базирующегося на построении выпуклых оболочек. Перебор производится по всем граням обеих выпуклых оболочек, а затем по парам ребер, принадлежащих разным выпуклым оболочкам.

Укажем еще ряд приемов, которые при реализации алгоритмов в некоторых специальных случаях позволяют либо сократить объем вычислений, либо обнаружить разделяющую плоскость на более ранних этапах. Если в реализации алгоритма требуется ввод информации об исходных множествах, то при вводе легко провести тест на минимум и максимум по отдельным координатам всех вершин, т. е. заключить множества в параллелепипеды с гранями, параллельными координатным плоскостям. В случае отделимости параллелепипедов исходные множества также будут отделимы. Такой случай достаточно часто встречается в реальной практике.

Если одно из исходных множеств представляет собой совокупность вершин многогранника, являющегося трехмерным телом, то легко показать, что выпуклую оболочку достаточно строить на тех гранях множеств, которые «освещаются» источником, расположенным в одной из вершин второго множества. При определении «освещенности» проверяются только источник и сама грань без учета закрытия другими гранями.

Описанный здесь подход опробирован в задачах предобработки данных для системы синтеза визуальной обстановки.

ЛИТЕРАТУРА

1. **Sutherland I. E., Sproul R. F., Schumacker R. A.** A characterization of ten hidden surface algorithms.— Comp. Surv., 1974, v. 6, N 1.
2. **Fuchs H., Kedem Z. M., Naylor B.** Predetermining visibility priority in 3-*D* scenes.— Computer Graphics, 1979, v. 13, N 2.
3. **Fuchs H., Adram G. D., Grant E. D.** Near real-time shaded display of rigid objects.— Computer Graphics, 1983, v. 17, N 3.
4. **Дейкстра Э.** Дисциплина программирования.— М.: Мир, 1978.

Поступило в редакцию 4 марта 1986 г.

УДК 681.3.06

А. В. ИОФФЕ

(Новосибирск)

ТЕСТИРОВАНИЕ, ДИАГНОСТИКА И НАЛАДКА ЦИФРОВЫХ УСТРОЙСТВ С ИСПОЛЬЗОВАНИЕМ ИЕРАРХИЧЕСКОЙ СХЕМЫ ПРОГРАММНЫХ МОДЕЛЕЙ

В данной заметке рассмотрен подход к разработке автоматизированных средств тестирования аппаратуры, использованный при создании синтезирующих систем визуализации, описанных в [1—3].

Для устройства, подлежащего тестированию, строится иерархическая система программных моделей. Модели самого верхнего уровня описывают поведение устройства в целом или какой-то функционально-независимой его части в терминах сигналов в основных входных и выходных контрольных точках. Модели более низких уровней относятся к подсхемам, представленным меньшим объемом оборудования, и описывают сигналы на внутренних контрольных точках. Следует подчеркнуть, что программные модели не обязательно должны непосредственно имитировать работу оборудования: от них требуется лишь точность расчета состояний в контрольных точках в соответствии с функционированием устройства. За счет этого модели могут

быть реализованы с максимальной эффективностью, что весьма важно, так как в процессе глобальной диагностики модель подсхемы может участвовать в многократных циклических операциях, например при генерации случайных или целенаправленных входных тестовых последовательностей.

Иерархическая система программных моделей строится на «глубину», обеспечиваемую имеющимися в аппаратуре контрольными точками. Поскольку встраивание аппаратуры контроля ведет к дополнительному расходу оборудования, не всегда возможно или целесообразно контролировать таким образом всю схему. Некоторые узлы остаются с точки зрения автоматического контроля «черными ящиками». Тем не менее программные модели строятся и с большей степенью детализации, но сравнивать получаемые сигналы в контрольных точках с расчетными можно только вручную. Тестирующая система здесь может подсказывать последовательность анализа контрольных точек, не доступных из ЭВМ. При проведении операций контроля и диагностики в режиме диалога результаты расчета моделей удобно представлять не в форме таблиц с перечнем сигналов и их значений, а в виде графических изображений функциональных и принципиальных схем с указанием текущего состояния сигналов. Объем такого рода информации может оказаться слишком большим для одновременного отображения на экране дисплея; это требует введения дополнительных уровней детализации.

Методика тестирования может быть представлена следующим образом. Прежде всего неисправность устанавливается на самом верхнем уровне, охватывающем максимальный объем оборудования. Для этой цели применяется метод сигнатурного анализа [4]. Предварительную подготовку сигнатур в контрольных точках можно производить с помощью тех же программных моделей. Используя модели нижних уровней и промежуточные контрольные точки, тестирующая система пытается по возможности минимизировать объем оборудования, содержащего неисправность. Далее включаются более детальные модели, работа которых изображается в виде схемы на экране дисплея. Проверка недоступных для ЭВМ контрольных точек в аппаратуре проводится с помощью дополнительных диагностических средств (логического анализатора, пробника, осциллографа).

Наиболее полно предложенная методика была использована при разработке тестового и диагностического обеспечения конвейерных вычислителей видеопроцессора канала видеопреобразования, описанного в [3]. Для обеспечения доступа к контрольным точкам здесь использовалась специальная тестовая магистраль. Поскольку реально устройство работает на видеочастоте, в схему его управления от тестовой магистрали введен режим, в котором можно задавать на входах любые тестовые данные независимо от тактирующего конвейера синхрогенератора. Такой режим обеспечивает возможность проверки правильности работы всех ступеней конвейера. Для каждого из узлов видеопроцессора были разработаны две программные модели: одна быстрая для выявления факта неисправности, а другая детальная, позволяющая провести подробную диагностику. Опыт эксплуатации подтвердил эффективность предложенного подхода.

ЛИТЕРАТУРА

1. **Ковалев А. М., Талныкин Э. А.** Машинный синтез визуальной обстановки.— Автометрия, 1984, № 4.
2. **Буровцев В. А. и др.** Геометрический процессор синтезирующей системы визуализации.— Автометрия, 1986, № 4.
3. **Богданов В. В. и др.** Канал видеопреобразования синтезирующей системы визуализации.— Автометрия, 1986, № 4.
4. **Гордон Г., Надиг Н.** Локализация неисправностей в микропроцессорных системах при помощи шестнадцатиричных ключевых кодов.— Электроника, 1977, № 5.

Поступило в редакцию 16 февраля 1986 г.