



Артамонов Евгений Иванович окончил Московский энергетический институт в 1965 г. по специальности «Математические счетно – решающие приборы и устройства». С 1955 г. работает в Институте проблем управления им. В.А. Трапезникова РАН в начале в должности техника электромеханика, затем

инженера, старшего научного сотрудника, с 1.10.1978 г. по настоящее время зав. лаб. «Компьютерной графики, специализированных технических и программных средств».

В 1972 г. Е.И. Артамонов защитил диссертацию к.т.н. на тему «Синтез структур специализированных технических средств» по специальности 05.13.05. С конца 60-х годов Е.И. Артамонов приступает к разработке принципов построения программного обеспечения специализированных программных средств, в частности, систем автоматизированного проектирования (САПР).

В 1999 г. Е.И. Артамонов защитил диссертацию д.т.н. на тему «Модели и методы проектирования интерактивных систем» по специальностям 05.13.12 и 05.13.16. В 2000 г. ему присвоено звание профессора.

С 2000 г. Е.И. Артамонов возглавляет кафедру «Инженерной и компьютерной графики» Московского технического университета связи и информатики (МТУСИ). Е.И. Артамоновым опубликовано более 100 статей, одна книга, три брошюры, пять методических пособий, получено более 12-ти авторских свидетельств на изобретения специализированных вычислительных устройств и одно свидетельство на программу универсальной автоматической трассировки соединений между элементами на плоскости.

Начиная с 2001-го года, Е.И. Артамонов является научным руководителем ежегодной Международной конференции и выставки "Системы проектирования, технологической подготовки производства и управления этапами жизненного цикла промышленного продукта (CAD/CAM/PDM)".

В 2009 г. Е.И. Артамонов возглавил со стороны ИПУ РАН Научно-образовательный центр «ИПУ-МГТУ СТАНКИН».

Е.И. Артамонов



ИНТЕРАКТИВНЫЕ СИСТЕМЫ. СИНТЕЗ СТРУКТУР

Москва
2010

**МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
СВЯЗИ И ИНФОРМАТИКИ**

Е.И. Артамонов

**ИНТЕРАКТИВНЫЕ СИСТЕМЫ.
СИНТЕЗ СТРУКТУР**

**Москва
2010**

УДК 744
ББК 85.15
А 86

Рецензенты:

доктор технических наук, профессор Хачумов В.М.
доктор технических наук, профессор Толок А.В.

Артамонов Е.И. Интерактивные системы. Синтез структур
– М.: Инсвязьиздат, 2010. – 185 с.: ил.

ISBN

В монографии рассмотрены принципы структурной организации интерактивных систем (ИС), реализованных в виде технических и программных средств. Приведена история развития графических средств взаимодействия пользователя с процессорной частью ИС, показана зависимость структурной организации процессорной части ИС от характеристик и способов согласования с внешними устройствами. Даны сравнительные оценки известных методов структурного анализа и синтеза систем, предложен метод проектирования структур ИС, отличающийся от известных проведением предварительных операций со структурами алгоритмов функционирования, учитывающих форматы и типы данных объектов информации.

Рассмотрены примеры последовательностей синтеза структур систем реализованных на технических средствах. Особое внимание уделено синтезу структур программно реализованных ИС со сложными и разнообразными структурами данных, к которым относятся системы 2D и 3D конструирования, а также системы автоматической трассировки соединений между элементами на плоскости.

Книга адресована студентам вузов, аспирантам и специалистам в области проектирования интерактивных систем.

© Артамонов Е.И., 2010

СОДЕРЖАНИЕ

Введение	6
Глава 1. Компьютерная графика и интерактивные системы.....	15
1.1 Структурная организация интерактивных систем.....	16
1.2 Классификация интерактивных систем.....	18
1.3 Внешние устройства в интерактивных системах.....	21
Г л а в а 2. Понятие алгоритма функционирования.	
Способы описания схем алгоритма.....	33
2.1 Алгоритмы функционирования. Операнды и операции.	33
2.2 Способы описания схем алгоритмов	35
2.2.1 Формульная запись алгоритмов.....	36
2.2.2 Запись алгоритмов на естественном языке	37
2.2.3 Запись алгоритмов на псевдокодах и алгоритмических языках	37
2.2.4 Графическая запись алгоритмов	38
Г л а в а 3. Этапы жизненного цикла промышленного продукта.....	45
3.1 Этапы жизненного цикла промышленного продукта.....	45
3.2 CALS-технологии	46
3.2 Состав электронной схемной документации на изделия РЭА	47
3.3.1 Типы схем	47
3.3.2 Обобщенная компьютерная модель изделий РЭА	48
3.3.3 Компьютерная модель схемы электрической структурной	51
3.3.4 Компьютерная модель схемы электрической функциональной.....	53
3.3.5 Компьютерная модель схемы электрической принципиальной.....	54
3.3.6 Компьютерная модель схемы электрической монтажной.....	55
3.3.7 Функциональное моделирование схем и устройств.....	58
Г л а в а 4. Структурное проектирование систем	60
4.1 Метод синтеза структур интерактивных систем.....	61
4.2 Операции со структурами алгоритмов	63
4.3 Формирование обобщенной структурной модели ИС	68
4.4 Примеры систематизации вариантов реализаций	

локальных алгоритмов	72
Г л а в а 5. Синтез структур интерактивных технических средств	77
5.1 Систематизация реализаций локальных алгоритмов для интерактивных технических средств с одноразрядными переменными	77
5.1.1 Анализ и систематизация способов преобразования информации	77
5.1.2 Систематизация способов преобразования информации	91
5.1.3 Анализ и систематизация способов хранения информации	92
5.1.4 Анализ и систематизация способов реализации операций сложения (вычитания) двух переменных	93
5.1.5 Способы выполнения операций умножения	94
5.2 Проектирование структур системы измерения параметров технологических процессов (СИП)	98
5.2.1 Операции со структурами алгоритмов СИП	99
5.2.2 Анализ вариантов структурной реализации СИП	101
5.3 Проектирование структуры системы смешения нефтепродуктов	102
Г л а в а 6. Синтез структур интерактивных программных средств проектирования машиностроительных конструкций	106
6.1 Классификация интерактивных программных средств	106
6.2 Формирование обобщенных моделей ИПС	109
6.3 Проектирование 2D систем конструирования	111
6.4 Примеры структур данных 2D систем конструирования	116
6.5 Проектирование 3D систем конструирования	119
Г л а в а 7. Синтез структур интерактивных программных средств проектирования радиоэлектронных устройств	127
7.1 Синтез вариантов построения СТП	127
7.1.1 Основные операции алгоритма функционирования СТП	127
7.1.2 Преобразование структуры алгоритма функционирования СТП	132
7.2 Алгоритм формирования растровой модели трассируемого поля	133

7.3 Алгоритм определения кратчайшего пути	139
7.4 Структурная организация комплекса программных средств проектирования РЭА из набора имеющихся на рынке систем	147
7.5 Система «Графика-01-Т». Интерактивные средства взаимодействия пользователя с системой	151
7.5.1 Структуры данных системы «Графика-ТР».....	153
7.5.2 Интерактивные средства взаимодействия пользователя с системой «Графика-ТР»	154
Г л а в а 8. Эффективность использования интерактивных программных систем.....	163
8.1 Моделирование внешнего облика орбитальной станции МИР	164
8.2 Моделирование процессов автоматической перестыковки модулей станции МИР	166
8.3 Моделирование крупногабаритной космической конструкции «Фермы-3»	167
8.4 Моделирование процесса установки и развертывания космического рефлектора	168
8.5 Моделирование процесса раскрытия Большого Космического Ретранслятора.....	168
8.5.1 Математические модели на этапах жизненного цикла крупногабаритных динамических конструкций	169
8.5.2 Особенности механизма развертывания Большого Космического Рефлектора.....	171
8.5.3 Структура обобщенной взаимосвязанной модели	172
8.6 Моделирование эргономических свойств пульта безопасности атомного реактора	173
8.6.1 Моделирование и эргономический анализ средств ОДУ	174
8.6.2 Структура системы.....	175
8.6.3 Управление процессом моделирования и эргономическим анализом	176
Заключение	178
Список литературы.....	179

ВВЕДЕНИЕ

Дальнейшее повышение производительности труда, снижение себестоимости продукции возможны лишь за счет внедрения развитых автоматизированных систем для процессов проектирования и производства конечного продукта, таких как системы автоматизированного проектирования (САПР), автоматизированные системы управления технологическими процессами (АСУ ТП), гибкие автоматизированные производства (ГАП) и т.п. Внедрение указанных систем позволяет ускорить получение конечного продукта, выполнить работы, которые невозможно сделать вручную, высвободить людские ресурсы, повысить качество конечного продукта, освободить человека от рутинной малопродуктивной работы.

В последнее время образовался новый класс автоматизированных систем – интерактивные системы (ИС), в которых за счет развитых средств взаимодействия, особенно графических, и повышенного быстродействия технических средств происходит общение пользователей с системой в реальном масштабе времени. В ИС достаточно часто аппаратная реализация специальных функций системы оказывается предпочтительней программной по стоимости, быстродействию, занимаемой памяти центрального процессора и целому ряду других критериев.

Стали актуальными проблемы исследования принципов структурной организации интерактивных систем, реализованных в виде технических и программных средств, разработки методов синтеза, анализа и моделирования на ЭВМ структур таких систем, используемых структур данных, способов согласования с внешними устройствами и средствами взаимодействия с пользователями.

При создании таких систем большое внимание уделяется разработке стандартов по методологии организации систем, стандартов на инвариантные части систем и на структуру данных для обмена с прикладными частями. Таким образом разработаны международные стандарты по CALS-технологиям, при проектировании программного обеспечения получила распространение CASE-технология (Computer Aided Software Engineering), техника и средства структурного анализа SADT (Structured Analyzes and Design Technique), включающего метод общего описания и спецификации алгоритмов функционирования систем IDEF (Integrated Computer

Aided DEFinition method), стандарт для описания данных об изделии (STEP), стандарты представления текстовой информации (SGML) и графики (CGM) и т.п.

Опыт, накопленный по проектированию и созданию интерактивных систем, позволяет перейти к решению проблемы теоретического обоснования принципов построения систем, выявлению общих закономерностей в структурной организации как технических, так и программных средств таких систем, обоснованию и выбору структур данных, способов согласования отдельных подсистем. Важным вопросом является автоматизированное проектирование интерактивных систем. Из-за отсутствия методов автоматизированного проектирования интерактивных систем увеличивается время разработки и, соответственно, стоимость, ухудшается их качество. Для создания таких систем привлекаются большие людские ресурсы. Практически отсутствуют работы по методам проектирования программно-аппаратных систем.

История развития интерактивных систем

Понятие «интерактивные системы (ИС)» возникло с начала 70-х годов применительно к системам компьютерной графики и автоматизированного проектирования [5, 35, 49, 58]. Под «интерактивными» понимались системы, в которых за счет развитых средств взаимодействия пользователя с системой и, в частности, графического интерфейса пользователя (Graphic User Interface – GUI), происходит общение с системой в реальном масштабе времени.

Графический интерфейс и компьютерная графика существенным образом повлияли на развитие программных и технических средств интерактивных систем. Прежде всего, это коснулось внешних устройств и процессорной части интерактивных систем, поскольку графическая информация, с одной стороны, обладает большим изобразительным разнообразием, чем текстовая, с другой стороны, она требует большой памяти и высокого быстродействия при вводе информации в ИС, обработке и выводе.

Так, например, в 70-х годах на системах без графического интерфейса, работающих не в интерактивном, а в пакетном режиме (вычислительные системы класса IBM-360 [40]), для моделирования газодинамических процессов (объем обрабатываемой инфор-

мации 10^7 чисел) и их визуализации требовалось два месяца работы. А в 80-х годах с развитием внешних устройств и средств интерактивного взаимодействия пользователей с ЭВМ для тех же газодинамических процессов на персональном компьютере результат можно было наблюдать в виде 5-ти минутного фильма на экране графического дисплея [35].

Сложность задач объемного (3D) геометрического моделирования и анимации в реальном времени предъявляют все более высокие требования к производительности и объемам памяти вычислительных систем. Под эти требования быстрыми темпами совершенствуется архитектура процессорной части вычислительных систем, средства 2D и 3D визуализации высокого и сверхвысокого разрешения, графические процессоры и средства интерактивного взаимодействия пользователя с ЭВМ. Так в 2006 году на супер-ЭВМ BLUE GENE/L(IBM) проведено полномасштабное моделирование авиадвигателя Pratt & Whitney, включая и объемное геометрическое моделирование, что позволило практически избежать натурных испытаний. Использование BLUE GENE сократило сроки разработки двигателя до 15 месяцев по сравнению с 2,5 годами моделирования на менее мощных супер-ЭВМ. Супер-ЭВМ BLUE GENE/L содержит 130 тыс. процессоров. Аналогичное моделирование на персональном компьютере заняло бы 700 лет [70].

Появление ИС с развитыми средствами ввода-вывода геометрических моделей привело к образованию большого количества разнообразных структур данных связанных с графическими языками пользователей, кодированием графической информации во внешних устройствах, с определенными способами внутрипроцессорного кодирования геометрических моделей. Впервые начали обсуждаться структуры данных, возникла проблема их унификации и стандартизации. В одной из первых редакциях международного стандарта Grafical Kernel Systems (GKS) в 1970г. впервые появилось понятие «метафайл», в котором хранилась бы определенным образом закодированная геометрическая модель изделия.

А первая публикация в России с обзором структур графических данных появилась только в 1995г. [28].¹

Интерактивные системы имеют, как правило, достаточно сложную структурную организацию. К настоящему времени не накоплено достаточного опыта по принципам построения таких систем, поскольку разработчики скрывают их внутреннюю организацию так же, как это было в свое время со структурами данных внешних устройств, когда некоторые фирмы отказывались раскрывать секреты кодирования данных в своих устройствах (см. [28]).

В таблице 1 схематично представлены основные этапы развития отечественных средств автоматизированного проектирования и этапы организационной работы ИПУ РАН по созданию этих средств.

Первая строка таблицы содержит этапы создания отечественных программных средств автоматизированного проектирования.

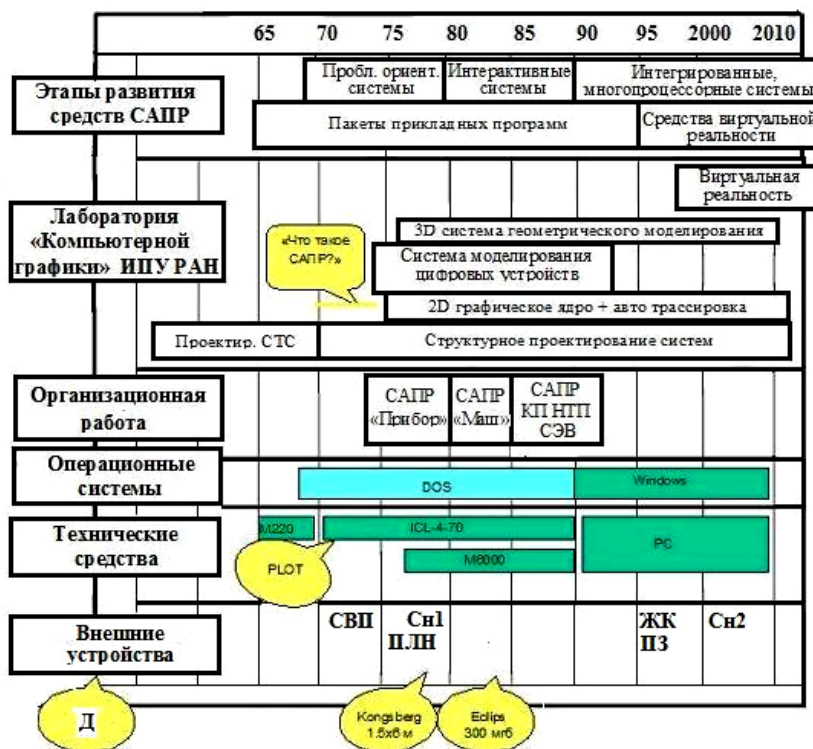
На второй строке представлены основные исследования, проводимые в разное время в области разработки средств САПР в ИПУ РАН.

На третьей строке приведена организационная работа, которая проводилась ИПУ РАН по САПР вначале в рамках Министерства приборостроения (САПР «Прибор»), затем среди предприятий министерств гражданского машиностроения (САПР «Маш») и далее совместно со странами-членами СЭВ и СФРЮ по Комплексной Программе Научно-технического Прогресса (КП НТП СЭВ, проблема 5.2.1).

¹ Впервые в России проблема «Как строить системы автоматизированного проектирования (САПР)?» обсуждалась на общероссийском семинаре "Математическое обеспечение систем с машинной графикой" в 1972 г. в г. Ижевске. Рассматривались структурные реализации в виде пакетов прикладных программ (обсуждались пакеты «Графор» и «ФАП-КФ») и в виде проблемно-ориентированной системы со своей специализированной структурной организацией и своими структурами данных (обсуждалась структура системы «Графика»).

Таблица 1

Этапы развития отечественных средств автоматизированного проектирования



На четвертой строке показан период смены операционных систем, который в начале 90-х годов в корне поменял принцип взаимодействия пользователя с ЭВМ.

На последних строках представлены технические средства, на которых в Институте проблем управления РАН (ИПУ РАН) проводились исследования принципов организации и разработка программных и технических средств САПР. Фигурные вставки снизу показывают отличительные признаки технических средств, оказавших влияние на развитие программного обеспечения САПР. В частности, наличие плоттера фирмы CallComp в комплектации

ЭВМ ICL-4-70 позволило начать работы по компьютерной графике в Институте проблем управления РАН (ИПУ РАН). Наличие секционного координатографа размерами 1,5 x 4,5 м² и графического дисплея на электроннолучевой трубке с послесвечением в комплексе фирмы Kongsberg позволило создать универсальные средства взаимодействия пользователя с ЭВМ и интерфейсы в проблемно ориентированных системах САПР. Комплекс Eclips для проектирования интегральных схем показал возможность работы в реальном времени при проектировании сложных объектов, отличием комплекса от такого же класса ЭВМ М 6000 являлись увеличенная емкость сменного магнитного диска более, чем в 100 раз, наличие цветного векторного дисплея с разрешением 1024 x 1024.

На последней строке таблицы, кроме того показано появление в разное время различных технических средств взаимодействия пользователей с ЭВМ: графического дисплея (Д), светового пера (СВП), сенсорных панелей первого и второго поколений (Сн1, Сн2), планшетов (ПЛН), жидкокристаллических (ЖК) и плазменных панелей (ПЗ).

CAD/CAM – системы зародились практически с момента выпуска первых ЭВМ (50-ые – 60-ые годы прошлого столетия), их развитие проходило по следующим основным этапам (первая строка табл.1): пакетная организация прикладных программ, создание проблемно ориентированных систем проектирования, совершенствование средств взаимодействия пользователей с системами, интеграция систем, использование средств виртуальной реальности.

Пакетная организация прикладных программ. На первом этапе из-за малого быстродействия ЭВМ, а также отсутствия технических средств компьютерной графики, средства автоматизации проектирования ограничивались пакетами прикладных программ. Решение любой задачи сводилось к написанию общей программы с обращением к конкретным процедурам пакетов прикладных программ, компиляции программы и после этого осуществлялся, ввод исходных данных и решение конкретной задачи. В это время, в первую очередь, решались задачи расчетного характера. Например, в области автоматизированного проектирования под руководством академика А.Я. Матюхина [47, 51] был разработан язык

функционального моделирования дискретных систем (МОДИС). МОДИС создавался на ЭВМ М-220 и длительное время эксплуатировался на единой серии ЭВМ, вплоть до появления персональных компьютеров.

С появлением графопостроителей в 60-х годах прошлого столетия возникла компьютерная графика и внесла революционные изменения в развитие всей вычислительной техники и, в частности, программных средств геометрического моделирования. В разное время были созданы пакеты графических программ такие, как PLOT 10, Графор [27], Графал, ФАП КФ [36, 37] и т.п. Идеология пакетов прикладных программ (ППП) сохранилась и до настоящего времени, несмотря на то, что в 70-х годах в промышленном применении наметился переход от ППП к проблемно ориентированным системам. ППП совершенствовались далее в направлении стандартизации, на их основе стали создаваться системы проектирования, отдельные алгоритмы стали реализовываться аппаратными средствами. Однако, на все это ушло около 40 лет.

К концу 70-х годов наступила эра **проблемно ориентированных систем проектирования**. Основными задачами, связанными с системной организацией средств проектирования, были разработка структур данных и их стандартизация, разработка принципов структурной организации систем и средств взаимодействия пользователей с системами. По структурам данных одним из первых обсуждался стандарт на графическое ядро GKS [73], затем IGES – 1982 г., ISO 10303 STEP - 1994 г. и т.п.

Средства взаимодействия пользователей с системами. Первые системы проектирования работали в пакетном режиме, т.е. в систему вводился пакет исходных данных, анализировался результат и, если результат оказывался неудовлетворительный, то вся процедура повторялась.

Каждая система имела собственный язык описания исходных данных [65], начали появляться специальные аппаратные средства, ускоряющие процесс ввода исходных данных, графопостроители постепенно заменялись графическими дисплеями. С начала 70-х годов с развитием средств взаимодействия пользователей с ЭВМ применительно к компьютерной графике и системам автоматизированного проектирования появилось понятие «интерактивные системы» [5, 35, 49, 58], которые в реальном времени обрабатыва-

ли каждый оператор входного языка. С совершенствованием устройств ввода отпала необходимость вводить операторы входного языка в текстовом виде, достаточно было указать на значки, их заменяющие, в перечне, представленном на экране дисплея. В программном обеспечении наметился переход в структурной организации интерактивных систем от пакетов графических программ к программным системам. Сформировалось понятие системной целостности программного обеспечения ИС, объединяющее процессорную часть, средства ввода-вывода информации и управления базами данных. Примерами таких систем в настоящее время являются: AutoCAD, SolidWorks, I-Deas и др.

В 80-х годах появились персональные компьютеры.

Интеграция систем. С начала 90-х годов появилось понятие CALS-технологии (Continuous Acquisition and Life-Cycle Support), технологии непрерывного компьютерного сопровождения изделия на всех этапах его жизненного цикла от маркетинга до утилизации [52]. CALS-технология отличается от традиционной технологии следующими особенностями: на всех этапах жизненного цикла изделия создается электронная документация, используется Единая обобщенная модель изделия и международные стандарты на форматы и структуры данных по обмену информацией об изделии. Кроме того, CALS-технология предусматривает параллельную и территориально распределенную работу над создаваемым изделием. В соответствии с этой технологией на рынке появились интегрированные системы, охватывающие несколько этапов жизненного цикла изделий, например, CAD/CAM, CAD/CAM/PDM и т.п.

Средства виртуальной реальности (СВР). Развитие СВР позволило по-новому взглянуть на решение некоторых задач автоматизированного проектирования. Существующие системы автоматизированного проектирования не всегда удовлетворяют пользователей по части визуализации сложных объемных геометрических моделей в реальном времени, затруднено создание инструкций и технических руководств по сборке и эксплуатации изделий с использованием 3D моделей. Поэтому некоторые задачи решаются с использованием языков виртуальной реальности VRML, Open GL [61], Direct X и др., или в комбинации языковых средств и стандартных структур данных, полученных из систем проектирования.

Хочу поблагодарить своих дорогих друзей, с которыми я имел счастье работать: Дмитрия Павловского, Михаила Щеголькова и Анатолия Шурупова за разработку принципов построения и создание интерактивных систем 2D и 3D систем конструирования; Олега Соловьёва, Владимира Паршукова и Людмилу Сизову за исследования структурной организации и создание системы автоматической трассировки соединений между элементами на плоскости; Алексея Разумовского за вклад в анализ структур данных систем проектирования, написания программ преобразования структур данных, а также за разработку объемных геометрических моделей всех модулей орбитальной космической станции МИР; Владимира Ромакина, разработавшего теоретические основы построения систем объемного геометрического моделирования на средствах виртуальной реальности и создавшего динамические модели систем раскрытия Большого Космического Рефлектора, эргономического анализа пультов безопасности АЭС, а также моделирования рельефов местности; Андрея Балабанова, систематизировавшего структуры данных систем объемного геометрического моделирования, разработавшего 3D модели объектов для решения задач виртуальной сборки в машиностроении, ситуационного моделирования, создания виртуальных тренажеров в медицине.

Проректор по научной работе МТУСИ кандидат технических наук В.С. Алешин и редактор В.М. Аладин оказали автору большую помощь при издании настоящей книги. В связи с этим автор выражает им глубокую благодарность.

Глава 1.

КОМПЬЮТЕРНАЯ ГРАФИКА И ИНТЕРАКТИВНЫЕ СИСТЕМЫ

Компьютерная графика возникла в 60-х годах прошлого столетия и внесла революционные изменения в развитие вычислительной техники.

Само понятие «компьютерная графика» В. Гилой [35] формулирует так:

компьютерная графика = структура данных + графические алгоритмы + языки взаимодействия.

Эта формулировка сделана по аналогии с определением понятия «программы» по Н. Вирту [76]

программа = алгоритм + структура данных.

На начальном этапе развития компьютерной графики большее внимание уделялось в основном графическим алгоритмам и геометрическим преобразованиям, чем структурам данных и языкам взаимодействия.

Только с конца 60-х годов с развитием средств взаимодействия пользователей с ЭВМ применительно к компьютерной графике и системам автоматизированного проектирования появилось понятие «интерактивные системы (ИС)» [49]. Под «интерактивными» понимались системы, в которых за счет развитых средств взаимодействия, особенно графических, и увеличенных быстродействия и объемов памяти процессорной части технических средств происходит общение пользователей с системой в реальном масштабе времени.

К началу 70-х годов начали серийно производиться средства интерактивного взаимодействия пользователя с ЭВМ такие, как графические дисплеи, устройства типа «джойстик», «мышь» и др. В программном обеспечении наметился переход в структурной организации интерактивных систем от пакетов графических программ к программным системам.

В настоящее время области использования ИС существенно расширились, размерность решаемых задач резко возросла, а сред-

ства графического взаимодействия пользователя с ЭВМ (как технические, так и программные) стали более совершенными.

Под **интерактивными системами** далее будем понимать системы, в которых в реальном времени происходит обработка информации, поступающей от датчиков технологических процессов и оператора, осуществляется управление процессом решения прикладной задачи по жестким, заранее запрограммированным алгоритмам, выполнение операций над графическими данными по созданию геометрических моделей, расчету процессов визуализации моделей объектов и их параметров, производится управление исполнительными механизмами, технологическими процессами и представление полученных результатов оператору.

Интерактивные системы реализуются в виде программных, аппаратных (технических) или программно-аппаратных средств. Иногда аппаратная реализация специальных функций ИС оказывается предпочтительней программной из-за большей скорости обработки информации.

Пакеты графических подпрограмм также продолжили своё развитие и к настоящему времени известность получили Open GL, Direct X и др.

1.1 Структурная организация интерактивных систем

Отличительной особенностью интерактивных систем является их непосредственная связь в реальном времени с технологическими процессами и человеком-оператором. Эта связь осуществляется через устройства ввода информации (средства интерактивного взаимодействия, датчики технологических процессов) и устройства вывода (дисплеи, показывающие приборы и исполнительные механизмы). Внешние устройства в интерактивных системах используют специфические структуры данных, а также различные точности представления информации. Это требует специальных преобразователей для согласования внешнего представления информации с внутрисистемным представлением.

Обобщенную структуру ИС можно представить так, как показано на рис. 1.1. Она содержит процессорную часть системы, языковые и интерактивные средства взаимодействия пользователей с

системой, средства визуализации, ввода и вывода информации, управления базами данных и управления всей системой.



Рисунок 1.1 Обобщенная структура ИС

Процессорная часть системы включает проблемно ориентированный и графический процессоры, выполняющие операции заданного алгоритма функционирования. Процессоры генерируют и обрабатывают математические модели технологических процессов и объектов, включая и геометрические модели.

Средства ввода и вывода содержат соответствующие драйверы и преобразователи, согласующие внутрисистемное представление информации с кодами внешних устройств. Эти средства ввода и вывода можно разделить на две основные части: традиционные, относящиеся к обычным, часто используемым интерактивным средствам взаимодействия пользователя с компьютером, и специализированные, определяющие взаимосвязь с другими компьютерами, технологическими процессами и системами.

Используя традиционные интерактивные средства взаимодействия, пользователь вводит исходные данные, управляет процессом функционирования системы и получает информацию о состоянии процесса.

В специализированных средствах взаимодействия информация поступает на вход от датчиков технологических процессов или от различных специализированных устройств ввода графической информации. По выходу информация представляется оператору, передается по каналам связи с другими системами или управляет исполнительными механизмами. Наличие специализированных средств ввода-вывода не только характеризует усложнение алгоритмов функционирования ИС, но и порождает новый взгляд, во-первых, на проблему структурной организации средств управления вводом-выводом информации, во вторых, на организацию всей системы в целом, поскольку почти каждое внешнее устройство использует свой специфический способ кодирования информации.

1.2 Классификация интерактивных систем

По способу связи процессорной части со специализированными внешними средствами интерактивные системы можно разделить на следующие четыре класса (см. рис. 1.2): со связями по входу и выходу, только по входу, только по выходу, без связи с внешними устройствами.

В первом классе ИС практически присутствуют все функции взаимодействия с внешней средой. К этому классу ИС относятся системы, обеспечивающие комплексное решение задач измерения, контроля, регулирования, управления и проектирования. Примерами таких систем являются регуляторы, оптимизаторы, корректирующие устройства, системы безопасности, тренажеры, а также интегрированные системы автоматизированного проектирования и управления технологическими процессами.

В ИС второго класса информация поступает только по входу и представляется оператору. К этому классу относятся системы измерения, контроля и последующего вычисления параметров различных объектов и технологических процессов. К ним относятся системы для комплексного решения задач измерения и вычисления параметров различных объектов и технологических процессов, интеллектуальные системы автоматизированного и автоматического ввода изображений, биоинформационные системы.

№ класса	Интерактивные системы				
	Устр. ввода	Драйверы Преобр. информации	Примеры систем	Драйверы Преобр. информации	Устр. вывода
1	+	+	Интегрированные системы проектирования и управления, безопасности. Тренажеры	+	+
2	+	+	Системы измерения и визуализации параметров технологических процессов		
3			Программное управление исполнительными механизмами, станками с ЧПУ	+	+
4			Системы проектирования, ситуационного моделирования, игровые системы		

Рисунок 1.2 Классификация интерактивных систем по связности с внешними устройствами

На ИС третьего класса информация поступает непосредственно от оператора, с различных носителей информации (магнитных и оптических дисков, магнитных лент перфокарт, перфолент и т.п.) или по каналам связи. К этому классу ИС относятся системы программного управления, осуществляющие перемещение исполнительных органов в соответствии с заданной программой, системы автоматизированного управления различными механизмами. Примерами таких систем являются системы управления графопостроителями, координатографами, фотоплоттерами, устройствами для монтажа радиокомпонентов, прошивочными автоматами, устройствами стереолитографии, технологическими установками и станками с ЧПУ.

В четвертом классе ИС информация по входу и выходу связана с оператором. В таких системах используются традиционные сред-

ства взаимодействия. Примерами этого класса систем являются специализированные вычислительные устройства, выполняющие элементарные арифметические операции, операции преобразования для долговременного хранения информации и последующего представления оператору, а также специализированные графические процессоры или процессоры для выполнения большого количества рутинных операций. К этому же классу относятся локальные автоматизированные системы проектирования, системы ситуационного моделирования, игровые системы и т.п.

В приведенной классификации на первый взгляд создается впечатление, что подключение тех или иных внешних устройств не может принципиально поменять принадлежность к конкретному классу ИС. Однако, подключение внешних устройств не только требует соответствующих драйверов и преобразователей, согласующих способы кодирования информации в процессорной части с кодами внешних устройств, но и изменяет алгоритм функционирования системы, а, следовательно, и принадлежность к другому классу. В качестве примера рассмотрим игровые системы с традиционными внешними устройствами, которые относятся к четвертому классу. В этих системах выходная информация представляется оператору на экране дисплея и воздействует на оператора только через зрительное восприятие информации. Подключение к таким системам, например, имитатора руля автомобиля, или более сложных устройств, типа кресла оператора с возможностью управляемого вращения его в пространстве, приводит к появлению обратной связи на действия оператора в виде изменения нагрузки на руль или управления перемещением кресла оператора. При этом существенно усложняется алгоритм функционирования системы и такую систему следует отнести к классу тренажеров.

В последнее время все больше стираются различия в структурной организации различных классов интерактивных систем. Общими признаками таких систем, являются одинаковые методы построения интерактивных средств взаимодействия с пользователями, средств управления внешними устройствами, средств измерения и визуализации, методы работы с различными структурами данных и геометрическими моделями, организации базовых инструментальных средств.

1.3 Внешние устройства в интерактивных системах

Как уже отмечалось, наличие определенного набора внешних устройств накладывает соответствующие требования на состав блоков ИС, используемые структуры данных и алгоритмы их преобразования. Перечислим внешние устройства, с которыми могут работать интерактивные системы, и особенности кодирования информации в них.

Классификация устройств ввода по функциональному назначению представлена на рис. 1.3. К ним относятся: устройства ввода с клавиатуры, средства управления маркером на экране дисплея, датчики технологических процессов, графические устройства, считыватели информации с запоминающих устройств, сетевые устройства.



Рисунок 1.3 Классификация устройств ввода

К классу **устройств ввода с клавиатуры** относятся алфавитно-цифровая и функциональная клавиатура. Для обнаружения нажатия клавиши используется несколько различных способов: механическое замыкание контактов, изменение емкости, изменение

магнитного поля, прерывание луча света и т.д. В последнее время появился набор электронных версий ввода текстовой информации, например, последовательно указывая средствами управления маркером на изображение клавиш, размещенных на экране дисплея, информация вводится в компьютер. Другой пример: лазерным лучом изображение клавиш высвечиваются на столе перед пользователем и, чтобы ввести текст, вы стучите пальцами по столу. Место прерывания лазерного луча фиксируется и введенный символ распознается компьютером. Информация с клавиатур различного типа поступает последовательно по одному символу в процессорную часть ИС в формате ASCII (American Standard Code for Information Interchange).

К классу **средств управления маркером** на экране дисплея относятся устройства типа "мышь", "рычаг", "шар", планшеты (ПЛН в табл. 1 Введения), сенсорные панели (Сн1, Сн2), световое перо (СВП). Наибольшее распространение в настоящее время получили «мышь» и Сн2. Принципы работы этих устройств примерно одинаковы: перемещение рабочих органов по осям X и Y вызывает генерацию импульсов на добавление или вычитание абсолютных значений координат местоположения маркера на экране дисплея. В результате появляется возможность выбирать необходимые данные из представленного на экране перечня (меню), задавать местоположение отдельных элементов и сами элементы, производить редактирование изображения.

Сенсорная панель позволяет непосредственно указывать объекты на экране дисплея либо специальной указкой, либо пальцем руки, а также управлять положением маркера. Простейшая сенсорная панель (Сн1 в табл. 1 Введения) представляет собой рамку, размещаемую по периметру экрана дисплея. На двух перпендикулярных сторонах рамки размещается некоторое количество светодиодов, а на противоположных к ним – фотоприемники. Координаты указания определяются по перекрытию лучей от светодиодов. Более сложные сенсорные панели используют прозрачную (стеклянную) поверхность, покрытую прозрачным проводящим слоем окиси олова. Факт указания определяется по изменению сопротивления. Известны сенсорные панели и с другими принципами определения координат места прикосновения пальца, например, с использованием поверхностных акустических волн. По-

следние (Сн2 в табл. 1 Введения) в настоящее время практически устанавливаются на всех переносных компьютерах.

Планшеты (ПЛН в табл. 1 Введения) генерируют координаты местоположения специальной указки на рабочем поле и позволяют не только управлять маркером, но и последовательно по точкам вводить в компьютер координаты векторных изображений с точностью порядка 0.5-1мм. Основное использование таких устройств приходилось на 70-80-е годы прошлого столетия, когда проходил переход к безбумажной технологии в проектировании, в компьютеры вводилась вручную информация с чертежей деталей машиностроения, принципиальных схем, печатных плат, интегральных схем и т.п., создавались электронные базы данных различного функционального назначения.

Планшеты по принципу функционирования делятся на потенциометрические, акустические, емкостные, магнитоэлектрические и магнитострикционные.

В *потенциометрических планшетах* рабочая поверхность представляет собой резистивное покрытие. По границам к этому покрытию подводится ток попеременно по X и Y направлениям. Указка планшета имеет гальванический контакт с резистивным покрытием. Значения координат определяется по падению напряжения в точке контакта.

В *акустических планшетах* указка излучает ультразвуковой сигнал, который принимается ленточными микрофонами, расположенными на двух смежных сторонах планшета. По времени прихода звука к микрофонам определяется точное положение указки. Акустические планшеты с тремя группами микрофонов могут выдавать трехмерную координатную информацию.

В *емкостных планшетах* под непроводящей рабочей поверхностью генерируется электромагнитное поле с помощью взаимно перпендикулярных групп проводников. Проводники в каждой группе должны быть точно параллельны и находиться на одинаковых расстояниях друг от друга. Эти проводники служат передающими антеннами. На передающие антенны поочередно подается высокочастотное напряжение. Сигнал принимается емкостным датчиком указки. По соотношению амплитуд сигналов можно узнать точное расположение между антеннами.

В *магнитоэлектрических планшетах* катушка в указке и проводники под рабочей поверхностью планшета могут рассматриваться как первичная и вторичная обмотки трансформатора. Если приемная катушка находится на указке, то конструктивно этот планшет подобен емкостному планшету. Существенно большее разрешение достигается при использовании обмотки зонда как передатчика, но в этом случае катушка зонда должна иметь много витков, чтобы сгенерировать достаточно мощное поле.

Магнитострикционные планшеты используют магнитострикционные проволоки как носители сигнала, которые под воздействием внешнего магнитного поля незначительно изменяют свою форму. Магнитное поле, вызываемое передающими катушками на краю планшета и перпендикулярное магнитострикционным проволокам, генерирует изменение их длин. Это изменение длины распространяется вдоль проволоки как волна механического напряжения со скоростью около 5000 м/с. Волна, попадая в приемную катушку, расположенную в указке планшета, из-за изменения потока формирует в катушке импульс напряжения. Время прихода волны пропорционально расстоянию от передающей катушки на краю планшета до зонда. Так как расстояние всегда измеряется вдоль проволоки, то не требуется, чтобы проволоки были абсолютно параллельны. Проволоки можно размещать на расстоянии в 2-3 мм., при этом на планшете гарантируется достаточное изменение потока. Этот принцип имеет относительно высокую точность (0.01 мм) и широко используется в робототехнике и в большинстве планшетов.

Точность представления информации на выходе средств перемещения маркера различна. Наименьшей точностью обладает световое перо и сенсорная панель. Как правило, с этих устройств последовательно посимвольно передается информация о координатах местоположения маркера.

Класс **датчиков технологических процессов** является наиболее представительным. К нему относятся, например, датчики температуры, давления, расходов жидкости, скорости вращения валов двигателей, угловых перемещений и т.п. Информация с датчиков может поступать в виде аналоговых сигналов тока или напряжения, частотных сигналов, число-импульсных (унитарных) кодов, время-импульсных сигналов, параллельных кодов.

К классу **графических устройств ввода изображений** относятся векторные и растровые устройства.

Векторные устройства (кодировщики) выполняются в виде планшетов (ПЛН) с элементами считывания координат. Они могут быть полуавтоматические и автоматические. По сути дела полуавтоматический кодировщик - просто большой планшет с достаточными размерами и разрешением, оснащенный так или иначе выполненной клавиатурой. В полуавтоматических кодировщиках оператор вручную перемещает рабочий орган по поверхности, распознает элементы документа (линии, отметки, знаки и т.п.) и идентифицирует их с помощью клавиатуры. Небольшая функциональная клавиатура обычно размещается непосредственно на рабочем органе, а дополнительные – на рабочей поверхности (в простейших случаях в качестве клавиатуры используется часть рабочей поверхности). Определение координат происходит автоматически по положению рабочего органа. Для повышения точности ввода координат рабочие органы (визеры) изготавливаются так, что допускают прецизионную установку в заданную точку. Если координаты расположены в узлах некоторой сетки, то точность обеспечивается округлением до ближайшего узла сетки. Обычно в полуавтоматических кодировщиках используется два основных режима работы: дискретный, когда оператор устанавливает указку в требуемую точку и выдает команду определения координаты, и непрерывный, когда оператор перемещает указку вдоль некоторой линии, а устройство автоматически генерирует последовательность координат. Информация с этих устройств представляется также, как и с планшетов.

Растровыми устройствами считывания являются сканеры и оптические устройства ввода. Имеется три основных варианта построения сканеров:

1. линейка фотоприемников перемещается относительно оригинала (планшетные сканеры);
2. оригинал перемещается относительно неподвижной линейки фотоприемников;
3. изображение неподвижного оригинала проецируется на матрицу фотоприемников, установленных в фокусе объектива (проекционные сканеры).

Примерами оптических устройств ввода являются цифровые фотоаппараты и видеокамеры, WEB-камеры и т.п. В этих устройствах также изображение сцены через объектив поступает на матрицу фотоприемников.

Информация с этих устройств поступает в растровой форме в форматах TIF, GIF, BMP, PCX, JPEG и т.д.

К классу **запоминающих устройств** относятся считыватели с перфокарт, перфолент, магнитных лент, магнитных, оптических и флэш-дисков. С использованием указанных средств может быть введена предварительная настройка систем на заданный алгоритм функционирования, загружены в базу данных описания типовых графических изображений, типовых моделей, результаты предварительного кодирования графических образов, предварительных расчетов. Информация в системы с указанных устройств поступает в параллельно-последовательном коде.

В настоящее время с учетом возможностей микроэлектроники функции запоминающих устройств существенно расширяются, они, например, одновременно могут запоминать не только цифровые сигналы, но и аналоговые. Так флэш-диск ET-L22 (MP3 плеер Explay) имеет память – один Гбайт, производит чтение и запись аудиоформатов – MP3, WMA, ACT, WAV, ASF и др.; записывает и воспроизводит речевую информацию в форматах WAV и ACT; имеет стереофоническое FM-радио.

Сетевые устройства включают модемы и различные сетевые карты.

Классификация устройств вывода представлена на рис. 1.4. По функциональному назначению устройства вывода могут быть разделены на следующие классы: 2D и 3D средства визуализации, принтеры, устройства с 2D координатным управлением, исполнительные механизмы, станки с числовым программным управлением (ЧПУ), запоминающие и сетевые устройства.

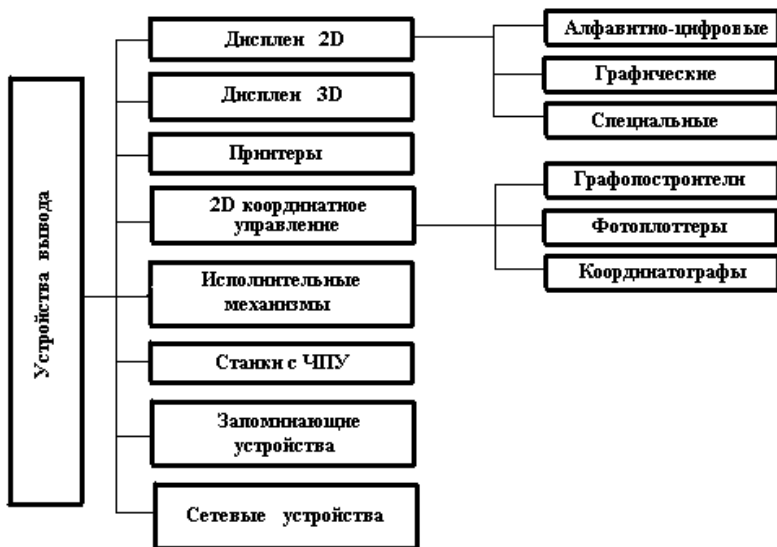


Рисунок 1.4 Классификация устройств вывода

К классу **2D средств визуализации** относятся дисплеи, различного рода индикаторы состояний приборов, узлов и технологических процессов.

Дисплеи по типу отображаемой информации можно разделить на алфавитно-цифровые, графические и специализированные, а по способу отображения – на векторные и растровые.

Первый векторный дисплей на электроннолучевой трубке появился в 1950г. для вывода графической информации из ЭВМ Whirlwind-I (Ураган- I) (Д в табл. 1 Введения). Это был дисплей с произвольным сканированием луча.

В последнее время большее распространение получили растровые дисплеи. Качество изображения в растровых дисплеях в существенной степени зависит от объема дисплейной памяти. По принципу функционирования растровые дисплеи могут основываться на использовании электроннолучевых трубок, жидкокристаллических или плазменных матриц.

Жидкокристаллические (ЖК) дисплеи основаны на поляризационных эффектах в жидких кристаллах, а плазменные (ПЗ)- используют явление свечения при разряде в газе. Оба типа дисплеев

выполнены в виде плоской панели, отличаются высокими разрешением и четкостью изображения, малым энергопотреблением и долговечностью.

В последнее время ЖК-технология по таким техническим характеристикам, как углы обзора, время отклика матрицы, размер видимой области и др., вплотную приблизились к плазменной, а в ряде случаев даже технически ее опередила.

Приведем примеры лучших технических характеристик, полученных в настоящее время на основе двух разных технологий. Жидкокристаллический дисплей 460 Pn фирмы Samsung, максимальный размер экрана 46 дюймов, разрешение 1366 x 768, потребляемая мощность 280 Вт, масса 28 кг. Плазменный дисплей TH-65PHD8 фирмы Panasonic, максимальный размер экрана 65 дюймов, разрешение 1366 x 768, потребляемая мощность 615 Вт, масса 78 кг [43].

К классу **3D средств визуализации** относятся устройства быстрого прототипирования и производства [74]. Такие устройства появились в начале 90-х годов прошлого столетия. Принцип их функционирования основан на расслаивании компьютерной 3D-модели объекта, послойной материализации ее в виде трехмерной физической модели. Лидирующими фирмами на рынке таких устройств являются: 3D Systems Corp., Helisys Inc., Stratasys Inc. Созданные этими фирмами устройства отличаются в основном способами формирования слоев. Таким образом, функционируют устройства стереолитографии, синтеза моделей из термопластов, ламинированных листов бумаги и полимерной² нити. Использование таких устройств позволило ведущим автомобильным компаниям сократить сроки разработки и выпуска новых моделей автомобилей с 3 – 5 лет до 1.5 – 2 лет [33].

Класс **печатающих устройств** (принтеров) содержит построчные и растровые ПУ. Растровые ПУ включают лазерные и струйные устройства.

Класс устройств с **2D координатным управлением** включает графопостроители, координатографы и фотоплоттеры. Обычно

² Эти устройства еще называют 3D-принтеры

точность графопостроителей составляет 0,1-0,05 мм, координатографов – 0,05-0,02 мм, фотоплоттеров – 0,05-0,01 мм.

Графопостроители используются для получения твердых копий графических изображений, в частности, чертежной документации. По своей конструкции делятся на планшетные и рулонные. В планшетных графопостроителях (см. рис. 1.5) носитель информации неподвижно закреплен на плоском столе.

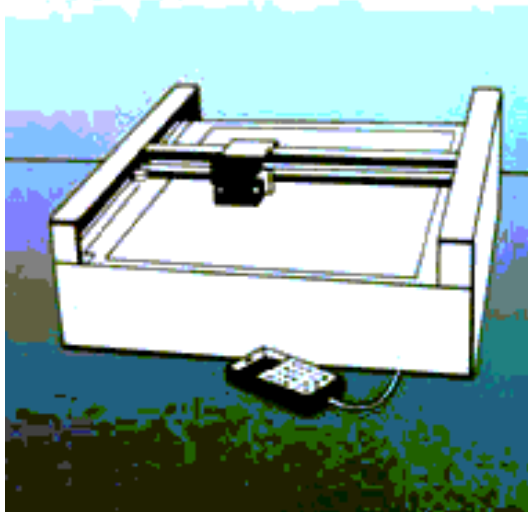


Рисунок 1.5 Общий вид графопостроителя-фотоплоттера

Закрепление может быть электростатическое, вакуумное, механическое за счет притягивания пластинок, прижимающих бумагу, к электромагнитам, вмонтированным в поверхность стола. Головка с пишущим инструментом перемещается по двум перпендикулярным направлениям.

Вне зависимости от способа перемещения носителя, система привода графопостроителей использует либо шаговые двигатели, поворачивающиеся на фиксированный угол при подаче одного импульса, либо исполнительную систему с обратной связью, содержащую двигатели привода и датчики положения. Перемещения инструмента графопостроителя под управлением шаговыми двигателями обычно выполняются на 1 шаг по одному из 8 направлений

с дискретностью по углу – 45° , поэтому изображение прямой линии в общем случае представляется в виде ломаной линии.

Координатографы [71] используются для изготовления различных шаблонов, фрезерования материалов из пластика, вырезания шаблонов для плазовых работ и т.п. По принципу работы аналогичны векторным графопостроителям, отличаются наличием сменного инструмента и, соответственно, увеличенной мощностью исполнительных механизмов, а также размерами рабочего поля. В качестве сменного инструмента могут использоваться специальные режущие инструменты, фрезы, сверла и т.п. Общий вид, например, координатографа DP 3X05 фирмы Glaser показан на рис. 1.6.



Рисунок 1.6 Координатограф DP 3X05 фирмы Glaser (Швейцария)

Его технические характеристики: активная рабочая поверхность (DP3705 – 1205 x 860 мм; DP3805 – 1605 x 1210 мм), разрешение 0.0025 мм, точность $\pm 0,02$ мм, повторяемость ± 0.01 мм, скорость 200 мм/с, ускорение 700 мм/с^2 .

Фотоплоттеры применяются для изготовления фотошаблонов в технологии получения высокоточных изображений, в частности, изображений конфигурации проводников, контактных

площадок, элементов на печатных платах и интегральных схемах. Фотоплоттеры делятся на векторные и растровые. Общий вид векторного фотоплоттера DP 1545 показан на рис. 1.5. Этот плоттер совмещает в себе функции графопостроителя и фотоплоттера, в первом случае используется головка с пишущим инструментом, во втором – фотоголовка. Его характеристики: активная рабочая поверхность 550 x 395 мм; разрешение 0.008 мм; точность $\pm 0,05$ мм; повторяемость $\pm 0,03$ мм; скорость 30 мм/с, ускорение 1500 мм/с².

Характеристики растрового фотоплоттера COMET 6060: лазерная головка 670 нм (красный), размер точки лазера 10,7 мкм, активная рабочая поверхность 720 x 610 мм, разрешение 0,02 мм (1270 DPI), 0,01 мм (2540 DPI), 0,005 мм (5080 DPI), точность $\pm 0,01$ мм.

На вход класса устройств с 2D координатным управлением информация поступает в форматах Gerber, HP-GL (Hewlett Packard Graphics Language), PostScript. Последний используется в устройствах растрового типа.

В рассматриваемом классе устройств в последнее время наблюдается тенденция в совмещении функций печатающих устройств и планшетных графопостроителей. Так швейцарская фирма ZUN для нанесения высококачественных цветных изображений на различные материалы выпустила серию планшетных принтеров со следующими характеристиками (см. рис. 1.7): ширина печати – 2150 мм, длина печати – не ограничена (зависит от длины материала), толщина материала – до 40 мм, разрешение печати 360/540/720 dpi.

К классу **исполнительных механизмов** относятся, например, шаговые двигатели, двигатели постоянного и переменного тока, клапаны, в том числе и цифровые клапаны, и т.п.

Большая часть систем управления перечисленных внешних устройств имеет нетрадиционные, разнообразные и сложные структуры построения собственной процессорной части, сложные структуры данных, требуют достаточно высокого быстродействия, непосредственно связаны с датчиками и исполнительными механизмами, то есть являются в свою очередь интерактивными системами.

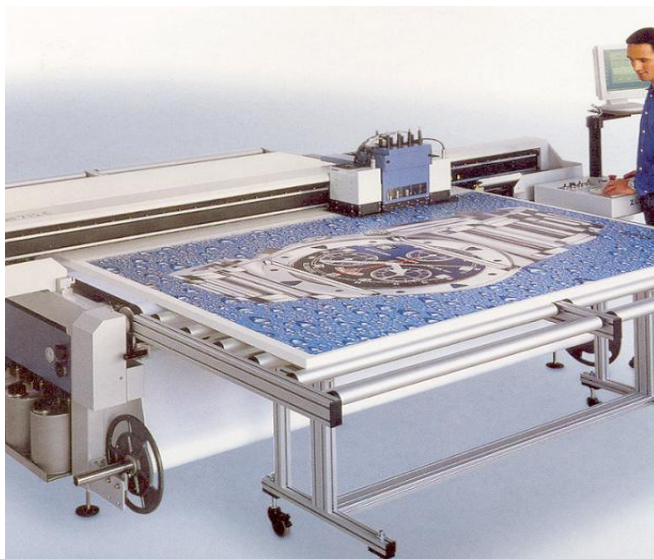


Рисунок 1.7 Планшетный принтер фирмы ZUN

Глава 2.

ПОНЯТИЕ АЛГОРИТМА ФУНКЦИОНИРОВАНИЯ. СПОСОБЫ ОПИСАНИЯ СХЕМ АЛГОРИТМОВ

2.1 Алгоритмы функционирования. Операнды и операции

Слово «алгоритм» произошло от имени среднеазиатского математика аль-Хорезми (IX в.) и использовалось в математике для обозначения правил выполнения четырех арифметических действий: сложения, вычитания, умножения и деления. В настоящее время понятие алгоритма используют не только в математике. Его применяют во многих областях человеческой деятельности, например, говорят об алгоритме управления производственным процессом, алгоритме игры в шахматы, алгоритме пользования бытовым прибором, алгоритме поиска пути в лабиринте, алгоритме управления полетом ракеты и т.п. [60].

Алгоритм – понятное и точное предписание (указание) исполнителю совершить определенную законченную последовательность действий для достижения указанной цели или решения поставленной задачи.

Формализация понятия алгоритма была предложена в 1936 году английским математиком А. Тьюрингом, который формально описал конструкцию некоторой абстрактной машины (машины Тьюринга), как исполнителя алгоритма, и высказал основной тезис о том, что всякий алгоритм может быть реализован соответствующей машиной Тьюринга. Примерно в то же время американским математиком Э.Постом предложена другая алгоритмическая схема – машина Поста. Эти алгоритмические схемы эквиваленты в том смысле, что алгоритмы, описываемые в одной из схем, могут быть также описаны и в другой.

Предполагается, что любая система работает по заданному алгоритму функционирования E , который характеризуется следующей тройкой параметров

$$E = \{P, O, \Gamma\}, \quad (2.1)$$

где P – множество объектов информации, поступающих в процессе функционирования систем по входам, выходам и связям между блоками;

O – множество операций над объектами информации;

G – множество связей между операциями;

Множество объектов информации в свою очередь характеризуется тройкой

$$P = \{A, \Phi, \Delta\}, \quad (2.2)$$

где A – множество способов внутренней организации объектов информации;

Φ – множество форм внешнего представления объектов информации;

Δ – множество степеней детализации внутреннего представления объектов информации.

В нотации Бэкуса характеристики алгоритма функционирования могут быть записаны:

Алгоритм \Rightarrow законченная последовательность действий | цель | Исполнитель

Последовательность действий \Rightarrow операции | операнды (объекты)

Операнд (объект) \Rightarrow способы внутренней организации (A) | формы представления (Φ) | степень детализации (Δ) | операнд (объект)

Формы представления \Rightarrow входная | выходная

В таблице 2.1 приведены примеры операций и объектов по некоторым областям использования.

Таблица 2.1

Примеры операций и объектов информации (операндов)

№ п/п	Области исполь- зования	Опера- ции	Объекты			
			A	Φ		Δ
				вход	выход	
1	Про- граммное обеспе- чение	преоб- разова- ния, редак- тирова-	.doc, .rtf, .dxf, .dwg, .pcb, .bmp, .jpg и др.	послед- оват. парал- лельн. по-	послед- оват. парал- лельн. послед-	символн. целые, действит., логи-

№ п/п	Области исполь- зования	Опера- ции	Объекты			
			А	Ф		Δ
				вход	выход	
		ние, вычис- литель- ные опера- ции		след- парал.	парал.	чекс.
2	Схемы РЭА (усили- тели, логиче- ские схемы, суммато- ры, про- цессоры)	преоб- разова- ния, редак- тирова- ние, вычис- литель- ные опе- рации	двоичный, двоичн- десят., десятич- ный, код Хемин- га MP3, WMA, ACT, WAV, ASF и др.	анало- говый, частот- ный, им- пульс- ный, циф- ровой	анало- говый, частот- ный, импульс ный цифро- вой	1%, 0.1%, максим. число двоичн. разрядов
3	Бытовая техника (Утюг)	Глаже- ние бе- лья Забива- ние гвоздей	Конструк- тивные эле- менты для использо- вания по прямому назначению То же, можно без нагрева- тельного элемента	Сеть 220в, ручка Ручка	Темпе- ратура плоско- сти на- грева Любая удобная для за- бивания гвоздей поверх- ность утюга	Точ- ность поддер- жан. заданной темпера- туры. Точ- ность обработ- ки рабо- чей по- верхно- сти

2.2 Способы описания схем алгоритмов

Алгоритмы могут быть записаны с разной степенью детализации. На ранних этапах проектирования, когда формируется струк-

тура алгоритма, записывается последовательность крупных операций, без подробного описания отдельных объектов и их параметров, далее, на следующих этапах, последовательно уточняются операции и параметры объектов [50, 60].

Существуют следующие способы записи алгоритмов:

- формульная;
- языковая, с использованием естественных, алгоритмических языков и псевдокодов;
- графическая.

2.2.1 Формульная запись алгоритма

Пример записи алгоритма цифрового вычисления пропорционально-интегрального закона регулирования процентного соотношения компонентов в жидких смесях (в частности при получении бензинов)

$$\mu_j[nT_c] = K_1 \left(x_j[nT_c] + \frac{1}{K_2} \sum_{i=1}^n x_j[iT_c] \right),$$

где

$\mu_j[nT_c]$ – регулирующее воздействие в цифровой форме, вычисленное по каждому j -ому каналу;

$$x_j[nT_c] = \varphi_{3j} - \varphi_j[nT_c] = \alpha_j \Pi \left(1 + \beta_j \Delta \Theta_j \right) - \int_0^{T_c/2} \varphi_j(t) dt - \text{раз-}$$

ность между заданным и текущим значением, вычисленная по j -ому каналу;

$\varphi_{3j}, \varphi_j(t)$ – заданное и текущее значение регулируемого параметра, текущее значение поступает на вход системы в виде частотного сигнала;

α_j – процентное содержание j -го компонента в смеси;

Π – производительность системы смещения;

β_j – коэффициент объемного расширения j -го компонента;

$\Delta \Theta_j$ – приращение температуры компонента относительно 20С;

T_c – время цикла работы системы;

k_1, k_2 – параметры настройки, $\delta_{k1} = 2\%$, k_2 принимает значения 2,4,8,16.

2.2.2 Запись алгоритма на естественном языке

Пример записи алгоритма на естественном языке рассмотрим на последовательности основных операций при автоматизированном проектировании устройств РЭА:

1. Разработка схемы электрической принципиальной (СхПР).
 - 1.1. Создание базы данных элементов СхПР.
 - 1.2. Размещение элементов СхПР.
 - 1.3. Трассировка соединений между элементами на СхПР.
 - 1.4. Выпуск документации на СхПР.
 - 1.4.1. Таблицы перечня элементов на СхПР.
 - 1.4.2. Чертеж СхПР.
2. Функциональное моделирование устройства.
3. Если результаты моделирования удовлетворяют требованиям технического задания на разработку устройства, то перейти к п. 4, иначе к п. 1.
4. Разработка схем электрических монтажных (СхМ).
 - 4.1. Создание базы данных элементов СхМ.
 - 4.2. Размещение элементов на печатных платах (ПП).
 - 4.3. Подготовка модели трассируемого поля ПП.
 - 4.4. Трассировка соединений между элементами на ПП.
 - 4.5. Выпуск фотошаблонов.
 - 4.6. Разработка документации на схемы монтажные.
5. Проектирование конструкции.

2.2.3 Запись алгоритмов на псевдокодах и алгоритмических языках

Приведем пример записи алгоритма по п. 2.2.1.:

Начало

Ввод исходных данных: N (общее количество циклов работы системы), M (общее количество компонентов в

смеси), $\Pi, \alpha_j, \beta_j, k_{1j}, k_{2j}, n=0$;

Начало цикла по n (время работы системы)

$M2: n=n+1$;

Если $n \geq N$ то $M1$;

$j=0$;

Начало цикла по j (компоненты смеси)

$j=j+1$;

Если $j \geq M$ то $M2$;

Ввод

$\Delta\Theta_j$

$$\varphi_{3j} = \alpha_j \Pi (1 + \beta_j \Delta\Theta_j)$$

Ввод (Измерение текущего значения j -го компонента)

$$\varphi_j[nT_c] = \int_0^{T_c/2} \varphi_j(t) dt$$

$$x_j[nT_c] = \phi_{3j} - \phi_j[nT_c]$$

$$\mu_j[nT_c] = K_1 \left(x_j[nT_c] + \frac{1}{K_2} \sum_{i=1}^n x_j[iT_c] \right)$$

Вывод $\mu_j[nT_c]$

Конец цикла по j

Конец цикла по n

$M1$: **Конец**

2.2.4 Графическая запись алгоритмов

Практически используются несколько способов графической записи алгоритмов, включая отечественные и международные стандарты. К ним относятся:

- отечественный стандарт ГОСТ 19.003-80 ЕСПД (аналог международного стандарта ИСО 1028) [50];
- международный стандарт IDEF0 [62, 63];
- универсальный язык моделирования UML [30, 31, 57] и др.

Графическая запись алгоритмов (схемы алгоритмов) представляется в виде последовательности соединенных блоков, обозначающих их функциональные особенности. Известные способы записи схем алгоритмов отличаются конфигурациями функциональных блоков, применяемыми средствами указания функций конкретных блоков, связями между блоками и программной поддержкой, автоматизирующей процесс работы со схемами алгоритмов.

2.2.4.1 ГОСТ 19.003-80 ЕСПД

Блоки схем алгоритмов показаны на рис. 2.1. Они включают изображения начала и конца алгоритма (а), ввода-вывода информации (б), выполнения операций (в), условного перехода (г), соединительных линий (д) между блоками и др.

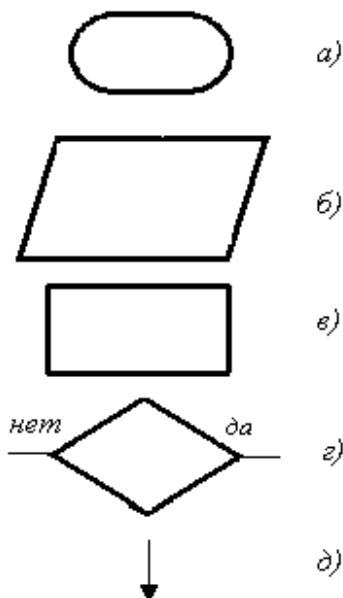


Рисунок 2.1 Пример блоков схем алгоритмов по ГОСТ 19.003-80 ЕСПД

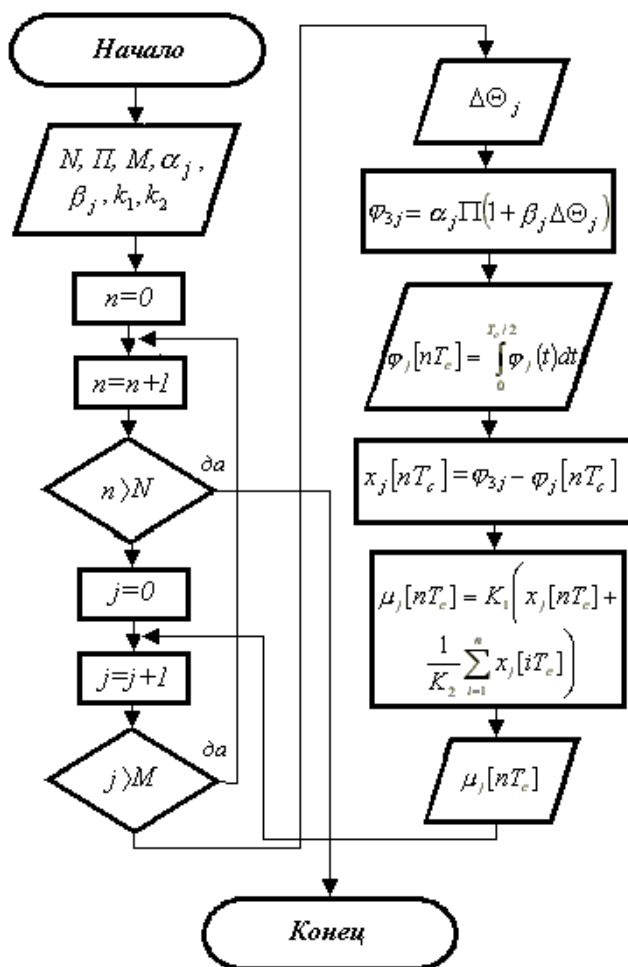


Рисунок 2.2 Схема алгоритма по ГОСТ 19.003-80

Установлены правила изображения блоков. Соотношения между шириной a и высотой b обычно составляют $b = 1.5a$, иногда $b = 2a$. Минимальное значение $a = 10$ мм. Ширину a можно увеличивать кратно 5 мм. На одной схеме рекомендовано блоки изображать одинаковых размеров.

Для алгоритма по п.2.2.1. схема по ГОСТ 19.003-80 показана на рис. 2.2.

2.2.4.2 Международный стандарт IDEF0

Технологию IDEF0 можно считать конечным этапом развития хорошо известного метода SADT (Structured Analysis and Design Technique) – диаграмм. Стандарт IDEF0 (Integrated DEFinition) был разработан в 1981 году в рамках программы автоматизации промышленных предприятий ICAM (Integrated Computer Aided Manufacturing), предложенной департаментом Военно-Воздушных Сил США. Последняя его редакция была выпущена в декабре 1993 года Национальным Институтом по Стандартам и Технологиям США (NIST) [34].

В основе IDEF0 лежат такие понятия, как функциональный блок (Activity Box), интерфейсная дуга (Argow), декомпозиция (Decomposition).

Функциональный блок графически изображается в виде прямоугольника, и, по требованиям стандарта, его название должно содержать глагольную форму. Каждая из четырех сторон функционального блока имеет своё значение: верхняя сторона – «Управление» (Control), левая сторона – «Вход» (Input), правая – «Выход» (Output), нижняя сторона – «Механизм» (Mechanism)³. Каждому функциональному блоку в рамках системы присваивается уникальный идентификационный номер.

Интерфейсная дуга (Argow), изображается в виде однонаправленной стрелки и имеет свое уникальное наименование (Argow Label), содержащее, по требованию стандарта, отглагольное существительное.

С помощью интерфейсных дуг отображаются различные объекты, в той или иной степени определяющие процессы, происходящие в системе. Такими объектами могут быть потоки данных и информации (документы, данные, инструкции и т.д.) или элементы реального мира (детали, аппаратура, сотрудники и т. д.). В зависимости от того, куда подходит интерфейсная дуга, она называется

³ Механизм реализации – уточнение автора

«входящей», «исходящей» или «управляющей». «Источником» (началом) каждой дуги может быть только выходная сторона функционального блока, а «приемником» (концом) – любая из трех других сторон. Любой функциональный блок должен иметь, по крайней мере, одну управляющую и одну исходящую интерфейсные дуги.

Третьим основным понятием стандарта IDEF0 является декомпозиция (Decomposition). Принцип декомпозиции применяется при разбиении сложного процесса на составляющие его функции. При этом уровень детализации процесса определяется непосредственно разработчиком модели. Декомпозиция позволяет представлять модель системы в виде иерархической структуры отдельных диаграмм, что делает ее менее перегруженной и легко усваиваемой.

Стандарт IDEF0 рекомендует ограничить сложность изображаемых на одном листе IDEF0-моделей шестью блоками и четырьмя подходящими или выходящими дугами.

Принцип работы с IDEF0 – диаграммами покажем на примере изображения этапов жизненного цикла проектируемых изделий РЭА.

В процессе декомпозиции функциональный блок, который в контекстной диаграмме отображает систему как единое целое, подвергается детализации на другой диаграмме (см. рис. 2.3 и 2.4).

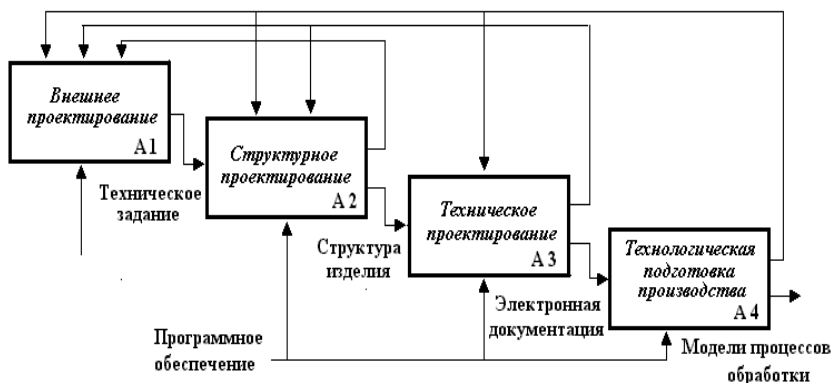


Рисунок 2.3 Этапы жизненного цикла проектируемых изделий

Функциональные блоки диаграммы второго уровня (дочерней – Child diagram) отображают главные подфункции функционального блока контекстной диаграммы и называются дочерними блоками (Child Box). В свою очередь, функциональный блок-предок называется родительским блоком (Parent Box), а диаграмма, к которой он принадлежит – родительской диаграммой (Parent Diagram).

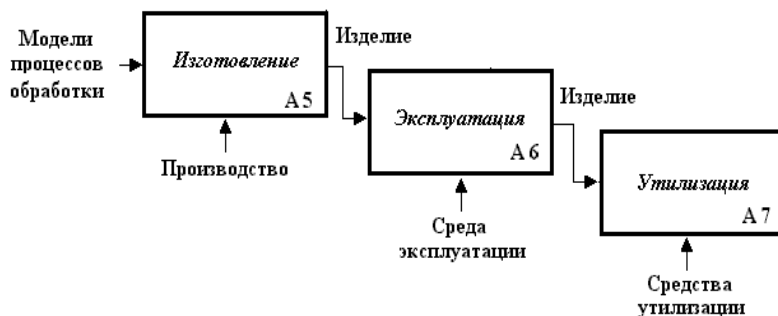


Рисунок 2.4 Продолжение этапов жизненного цикла проектируемых изделий

Каждая из подфункций дочерней диаграммы может быть далее детализирована путем аналогичной декомпозиции соответствующего ей функционального блока. При этом все интерфейсные дуги, входящие в функциональный блок или исходящие из него, фиксируются на дочерней диаграмме. Таким образом, достигается структурная целостность IDEF0-модели. Все функциональные блоки нумеруются. Номер блока обозначается за префиксом. В подавляющем большинстве используется префикс А. Контекстный блок всегда имеет блок А0 (на рисунках не показан). В рассматриваемом примере блок А0 разделяется на блоки А1, А2,... А7, которые образуют второй уровень иерархии. Для каждого уровня иерархии добавляется одна цифра. На третьем уровне (рис. 2.5.) детально рассматривается блок А3 (Техническое проектирование), содержащий блоки А31 (Проектирование принципиальной схемы), А32 (Функциональное моделирование принципиальной схемы),

A33 (Проектирование монтажной схемы), A34 (Разработка электронной документации).

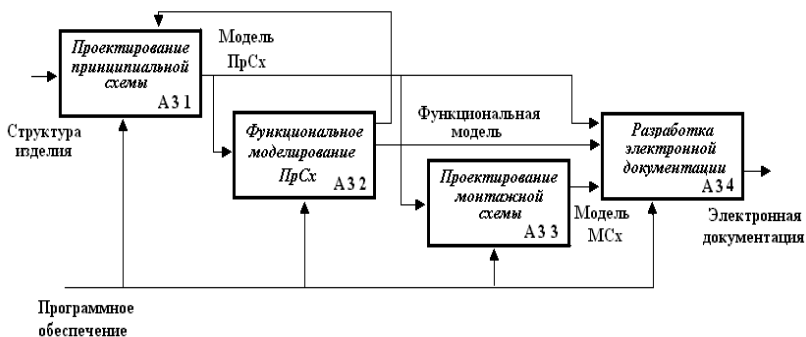


Рисунок 2.5 Блок А3 (Техническое проектирование)

Для практического использования стандарта разработано программное обеспечение IDEF-моделирования BPWin и Design/IDEF [62].

Глава 3.

ЭТАПЫ ЖИЗНЕННОГО ЦИКЛА ПРОМЫШЛЕННОГО ПРОДУКТА

3.1 Этапы жизненного цикла промышленного продукта

Основными этапами жизненного цикла (ЖЦ) промышленного продукта являются: внешнее проектирование или предпроектные исследования, структурное проектирование, техническое проектирование, технологическая подготовка производства, изготовление объекта, эксплуатация и его утилизация (рис. 3.1.).

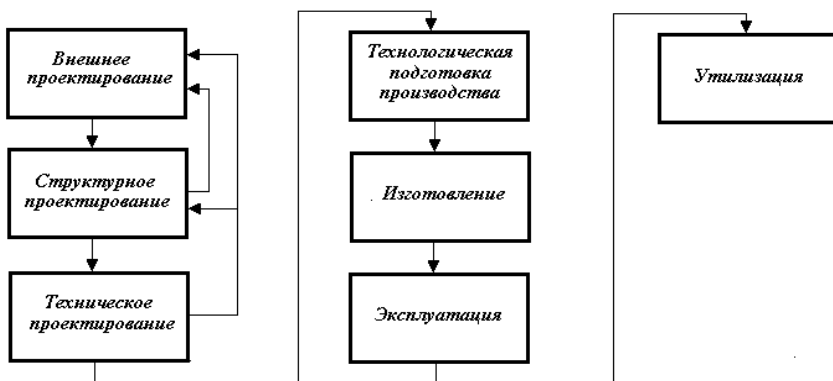


Рисунок 3.1 Основные этапы жизненного цикла промышленного продукта

На этапе внешнего проектирования проводятся маркетинговые исследования рынка, формулируется постановка задачи, составляются требования технического задания, разрабатываются алгоритмы функционирования, определяются общие характеристики входных и выходных параметров проектируемого объекта. Выбирается набор критериев, по которым будут сравниваться различные варианты построения объекта.

На этапе структурного проектирования производится синтез и анализ вариантов построения объекта и его отдельных узлов, составляются математические модели вариантов структурной организации и принципов функционирования отдельных узлов и всего

объекта, производится всесторонние исследования моделей с целью выбора наиболее предпочтительного варианта.

На этапе технического проектирования создается конструкторская документация, составляются спецификации, ведомости комплектующих изделий и т.п. В процессе разработки конструкторской документации возможно решение различного рода расчетных задач, задач компоновки и размещения отдельных узлов проектируемого объекта, трассировка соединений между узлами и элементами объекта.

На этапе технологической подготовки производства ведется разработка структуры технологического процесса, моделирование процесса обработки изделия, разработка пооперационной документации и получение управляющей информации для станков с ЧПУ.

На этапе изготовления создаются отдельные узлы объекта, производится их сборка и тестирование.

Процесс проектирования носит явно выраженный итерационный характер, то есть наблюдается обратная связь с последующих этапов проектирования на предыдущие. При этом производится коррекция и уточнение исходных данных, переход к рассмотрению других вариантов построения и т.п.

3.2 CALS-технологии

С начала 90-х годов появилось понятие CALS (Continuous Acquisition and Life-Cycle Support) – технологии непрерывного компьютерного сопровождения изделия на всех этапах его жизненного цикла от маркетинга до утилизации [48, 52].

CALS-технология отличается от традиционной технологии следующими особенностями:

1. На всех этапах жизненного цикла (ЖЦ) изделия создается электронная документация.
2. Для всех этапов ЖЦ создается и используется Единая обобщенная модель изделия.
3. Разработка и использование международных стандартов на форматы и структуры данных по обмену информацией об изделии [75].
4. Параллельная и территориально распределенная работа над

создаваемым изделием.

В настоящее время интерактивные системы используются на всех этапах жизненного цикла (ЖЦ) промышленного продукта. Таким образом, например, при решении расчетных задач используются CAE (Computer Aided Engineering) – системы, при проектировании – CAD (Computer Aided Design) – системы, при технологической подготовке производства и изготовлении – CAM (Computer Aided Manufacturing) – системы. Системы ERP (Enterprise Resources Planning) и PDM (Product Data Management) применяются при управлении ресурсами предприятия и данными на всех этапах жизненного цикла промышленного продукта.

3.3 Состав электронной схемной документации на изделия РЭА

Схемой называется конструкторский документ, на котором составные части изделия или установки изображены в виде условных графических обозначений (УГО) и показаны связи между ними [50, 39, 53].

3.3.1 Типы схем

Схемы по типам можно разделить на структурные, функциональные, принципиальные, соединений (монтажные).

Структурные – определяют основные функциональные части изделия, их назначение и взаимосвязи. Функциональные – разъясняют определенные процессы, протекающие в отдельных функциональных цепях изделия или установки, или в изделии в целом.

Принципиальные – определяют полный состав элементов и связей между ними и дают детальное представление о принципах работы изделия или установки (они служат основанием для разработки других конструкторских документов).

Схемы соединений (монтажные) – показывают соединения составных частей изделия и определяют провода, жгуты, кабели или трубопроводы, осуществляющие эти соединения, а также места их присоединений и ввода.

В чертежной документации используются сокращенные наименования вида и типа схем. Вид Э – электрическая, Г – гидрав-

лическая и т.п. Тип схемы обозначают цифрами: 1 – структурная, 2 – функциональная, 3 – принципиальная, 4 – соединений (монтажная). Например, схема электрическая принципиальная в чертежных документах обозначается Э3, схема электрическая монтажная – Э4 и т.п.

3.3.2 Обобщенная компьютерная модель изделий РЭА

Компьютерные модели изделий в информационных системах практически отображаются их структурами данных. Существенной составляющей в CALS-технологиях является унификация структур данных представления одинаковых моделей в различных отраслевых информационных системах. Унификация структур данных является достаточно сложной проблемой и для своего решения требует значительного времени и средств. Тем не менее, в настоящее время получены уже определенные результаты.

Так, корпорация Тошиба, которая осуществляет крупный проект по созданию библиотеки электронных изделий, разрабатывает интегрированную базу данных InterLIB, для которой обеспечивается совместимость как со структурой данных PLIB, так и со структурой данных EPISTLE (схема данных EPISTLE является основой стандарта ISO 15926 OIL&GAS). Использование базы данных InterLIB обеспечит взаимный обмен данными между библиотеками PLIB и Хранилищем Данных, соответствующим стандарту ISO 15926.

Кроме того, разрабатываются унифицированные протоколы не только на электронные изделия, но и для других применений. В нефтегазовой промышленности Англии, например, разработаны Протоколы STEP AP221 [75] (функциональная и организационная модель предприятия), AP227 (пространственная конфигурация перерабатывающего завода, включающая описание системы трубопроводов), AP231 (описание характеристик перерабатывающего оборудования) и специальный стандарт описания нефтегазового оборудования ISO 15926 OIL&GAS.

Таким образом, использование идеологии CALS-технологий позволяет производить обмен данными на всех этапах жизненного цикла создаваемого изделия от маркетинга, конструкторского проектирования, технологической подготовки производства и до ин-

формационного сопровождения эксплуатации изделия, что называется интегрированной логистической поддержкой. Совокупность структур данных, используемых на всех этапах жизненного цикла изделий, будем далее называть обобщенной компьютерной моделью изделия

Обобщенная компьютерная модель (ОМ) [17] содержит набор взаимосвязанных компьютерных моделей для всех этапов проектирования. На рис. 3.2 показана упрощенная структура процесса проектирования РЭА. Основанием для проектирования является техническое задание, а результатом – документация на технологическую подготовку производства (ТПП) и изготовление РЭА. Выделены две стадии разработки РЭА: структурное и техническое проектирование. На этих стадиях, соответственно, решаются задачи, связанные с этапами проектирования схем электрических структурных (СЭС) и функциональных (СЭФ), а также принципиальных (СЭП) и монтажных (СЭМ). Обобщенная компьютерная модель (ОМ) содержит две основные составляющие: схемную и функциональную.

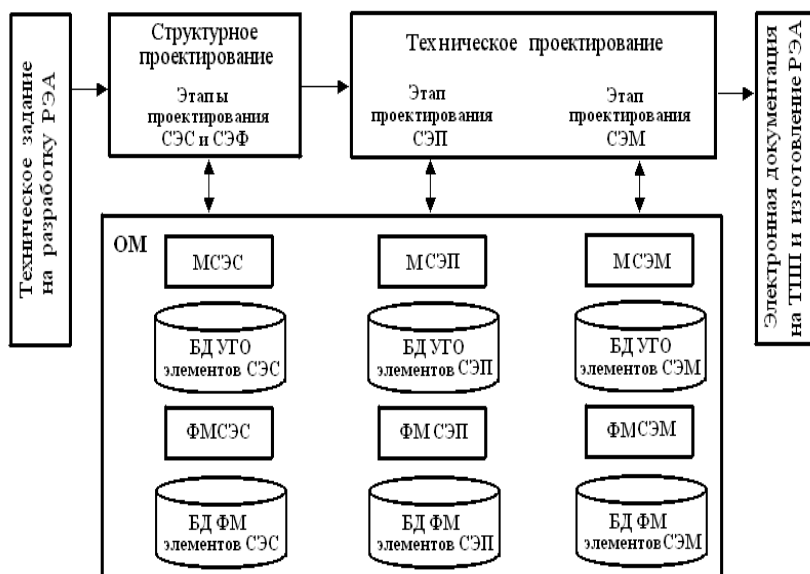


Рисунок 3.2 Структура процесса проектирования РЭА

Схемная составляющая является некоторой статической программой, как бы отображающей физический внешний вид отдельных элементов, узлов и блоков проектируемой аппаратуры. Эта составляющая включает взаимосвязанные компьютерные модели МСЭС, МСЭП и МСЭМ, создаваемые на разных этапах проектирования, соответственно для схем электрических структурных (СЭС), принципиальных (СЭП) и монтажных (СЭМ) Под моделью схем будем понимать не только их единственное изображение с конкретными текущими значениями координат элементов и связей, а еще и некоторое математическое описание ее граф - схемы, вершинами которой являются отдельные функциональные части аппаратуры, а дугами – связи между ними. Такие модели легко преобразуются в электронный документ, а также в модели, которые используются на последующих этапах. Связи между контактами отдельных элементов схемы (основных функциональных частей изделия) создаются за счет специальных программных средств, используемых в системах автоматизированного проектирования. Например, в системе Графика-01-Т, на практике создания и эксплуатации которой написан этот текст, в описаниях элементов схемы вводится специальное понятие «контакт», определяющее координаты подсоединения внешних связей. Это позволило в моделях схем все связи представить в виде «резиновых нитей», что дает возможность легко перемещать и модифицировать отдельные элементы при переходе с этапа на этап, сохраняя их связность.

На рис. 3.3 приведен пример модели принципиальной схемы (МСЭП). Такая модель, например МСЭП, модифицируется на этапе проектирования СЭМ в том смысле, что, во-первых, изображение каждого элемента заменяется на изображение его посадочного места на печатной плате, во-вторых, местоположение каждого элемента изменяется и подчиняется законам проектирования монтажных схем при неизменной связности между элементами на этих этапах.

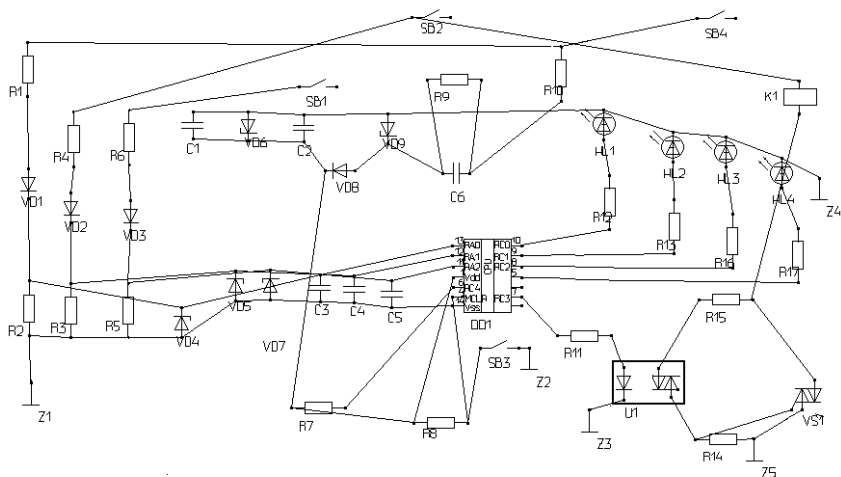


Рисунок 3.3 Пример модели принципиальной схемы

Функциональная составляющая (функциональная модель), как правило, должна присутствовать на каждом уровне разработки схемной документации. Она также представляет собой программу, описывающую с меньшей или с большей степенью детализации динамические процессы функционирования всех элементов, узлов и блоков проектируемого изделия. На рис. 3.2 в составе ОМ показаны функциональные модели ФМСЭС, ФМСЭП и ФМСЭМ, а также базы данных функциональных моделей (БД ФМ) отдельных элементов, которые создаются на соответствующих этапах проектирования.

Далее на примере разработки и эксплуатации системы автоматизированного проектирования схемной документации (Графика-01-Т) предлагается состав компьютерных моделей СЭС, СЭФ, СЭП, СЭМ и перечень электронных документов.

3.3.3 Компьютерная модель схемы электрической структурной

Компьютерной моделью схемы электрической структурной (СЭС) называется совокупность описания условных графических обозначений (УГО) основных функциональных частей изде-

лия с относительными координатами их контактов (входов, выходов) и связей между контактами, достаточных для редактирования и образования электронных документов в системах автоматизированного проектирования. Пример СЭС показан на рис. 3.4.

Связи между контактами отдельных элементов схемы (основных функциональных частей изделия) образуются за счет специальных программных средств, используемых в системах автоматизированного проектирования.

Компьютерная модель СЭС содержит:

- перечень УГО основных функциональных частей изделия СЭС;
- описания УГО основных функциональных частей изделия СЭС и описания относительных координат их контактов;
- описания связей между контактами частей СЭС;
- ссылки на компьютерную модель СЭФ.

Состав электронных документов СЭС:

- библиотека УГО основных функциональных частей изделия СЭС с описаниями относительных координат их контактов;
- компьютерная модель СЭС;
- электронный чертеж размещения УГО основных функциональных частей изделия СЭС и связей между ними;

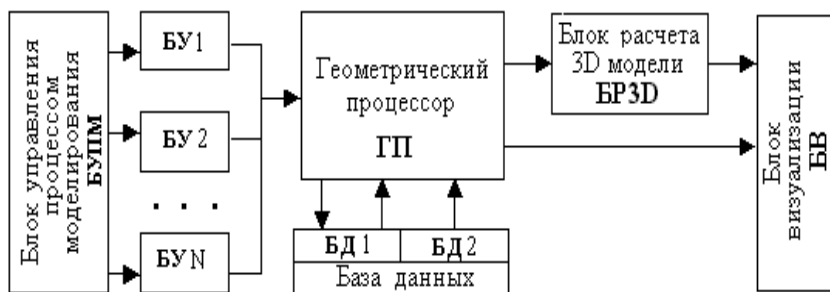


Рисунок 3.4 Схема электрическая структурная

- электронная таблица перечня основных функциональных частей изделия СЭС.

3.3.4 Компьютерная модель схемы электрической функциональной

Известно, что исполнение чертежной документации вручную регламентируется ГОСТ 2.701 - 84 [50], а компьютерная документация – ГОСТ 2.052 – 2006 [39, 53, 54]. Здесь можно отметить несколько обстоятельств, по которым указанные документы не способствуют развитию систем логистической поддержки РЭА. Например, в [50] записано что, схема электрическая функциональная разъясняет определенные процессы, протекающие в отдельных функциональных цепях изделия или установки, или в изделии в целом. Однако, вряд ли на схеме можно показать детально принципы функционирования, это можно выполнить только с использованием программных систем моделирования, предварительно описав архитектуру РЭА на специальных языковых средствах [56, 66]. Поэтому предлагается в компьютерную модель схемы электрической функциональной включить описание модели функционирования соответствующей схемы.

Что же касается ГОСТ 2.052 – 2006, то в нем регламентируется только электронная документация без ссылок на модели. Таким образом, в настоящей работе делается попытка восполнить существующий пробел в организации обобщенной компьютерной модели в логистической поддержке РЭА.

Компьютерной моделью схемы электрической функциональной (СЭФ) называется совокупность описания УГО **основных частей изделия** с относительными координатами контактов (входов, выходов) и связей между контактами, а также описания их процессов функционирования, достаточных для адекватного функционального моделирования изделия, редактирования графической и функциональной частей схемы, образования соответствующих электронных документов в системах автоматизированного проектирования.

Компьютерная модель СЭФ содержит:

- перечень УГО основных функциональных частей изделия СЭФ;
- описания УГО основных функциональных частей изделия СЭФ и описания относительных координат их контактов;

- описания связей между контактами основных функциональных частей изделия СЭФ;
- описания процессов функционирования основных частей изделия СЭФ;
- ссылки на компьютерную модель СЭС;
- ссылки на компьютерную модель СЭП.

Состав электронных документов СЭФ:

- библиотека УГО основных функциональных частей изделия СЭФ с описаниями относительных координат их контактов;
- библиотека описаний процессов функционирования основных функциональных частей изделия СЭФ;
- компьютерная модель СЭФ;
- чертеж размещения УГО основных функциональных частей изделия СЭФ и связей между ними;
- таблица перечня элементов СЭФ.

3.3.5 Компьютерная модель схемы электрической принципиальной

Компьютерной моделью схемы электрической принципиальной (СЭП) называется совокупность описаний УГО элементов схемы с относительными координатами контактов элементов и связей между контактами элементов, достаточных для редактирования и образования электронных документов в системах автоматизированного проектирования. Пример схематического изображения компьютерной модели СЭП показано на рис. 3.5.

Компьютерная модель СЭП содержит:

- перечень УГО элементов СЭП;
- описания УГО элементов СЭП и описания относительных координат контактов элементов;
- описания связей между контактами элементов схемы;
- ссылки на компьютерные модели СЭС, СЭФ, СЭМ.

Состав электронных документов на СЭП:

- библиотека УГО элементов СЭП с описаниями относительных координат контактов элементов;
- компьютерная модель СЭП;

- чертеж размещения УГО элементов СЭП со связями между ними;
- таблица перечня элементов СЭП.

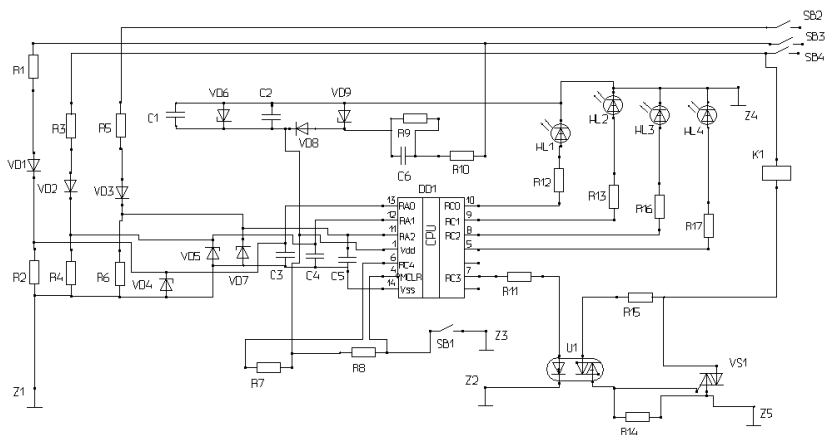


Рисунок 3.5 Компьютерная модель схемы электрической принципиальной

3.3.6 Компьютерная модель схемы электрической монтажной

Компьютерной моделью схемы электрической монтажной (СЭМ) называется совокупность установочных мест элементов СЭМ, располагаемых на соответствующих конструктивах (блоках, печатных платах, интегральных схемах и т.п.), с относительными координатами контактов на установочных местах элементов и связей между контактами, достаточных для редактирования и образования электронных монтажных документов в системах автоматизированного проектирования.

Компьютерная модель СЭМ содержит:

- перечень типовых графических изображений установочных мест элементов СЭМ;
- описания типовых графических изображений установочных мест элементов схемы с относительными координатами их контактов;

- описания типовых графических изображений внешнего вида элементов с относительными координатами контактов;
- описания связей между контактами установочных мест элементов СЭМ;
- ссылки на компьютерную модель СЭП;
- ссылки на 3D компьютерную модель используемых конструктивов, включая 3D модели элементов.

Визуальное отображение компьютерной модели СЭМ для СЭП (рис. 3.5) представлено на рис. 3.6.

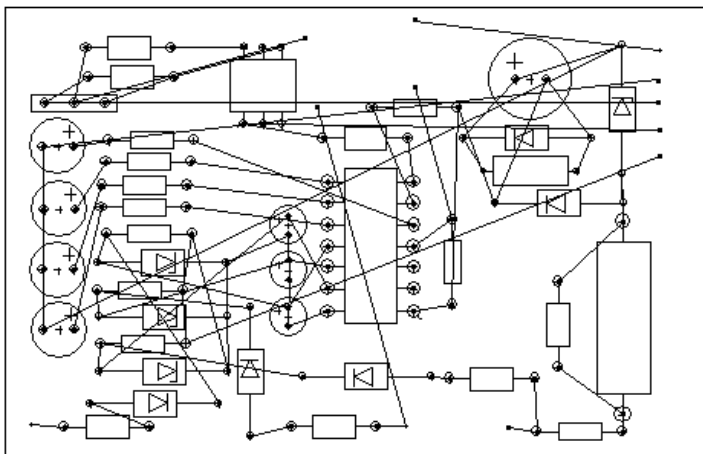


Рисунок 3.6 Компьютерная модель схемы электрической монтажной

Состав электронных документов СЭМ:

- библиотека типовых графических изображений установочных мест элементов схемы электрической принципиальной с описаниями относительных координат контактов установочных мест элементов;
- библиотека внешнего вида 3D моделей элементов и используемых конструктивов;
- компьютерная модель СЭМ;
- чертеж размещения типовых графических изображений установочных мест элементов схемы;
- чертеж размещения типовых графических изображений внешних видов элементов схемы монтажной;

- послойные чертежи размещения соединений между элементами СЭМ;
- сборочный чертеж СЭМ;
- таблица перечня соединений между элементами;
- таблица сверления отверстий на печатных платах (ПП);
- управляющая программа для изготовления фотошаблонов печатных плат (Gerber или HPGL - файл);
- управляющая программа для сверления отверстий на печатных платах (ПП) для сверлильного станка с ЧПУ (Gerber или HPGL - файл);
- управляющая программа для станков автоматической установки элементов на печатные платы.

На рис. 3.7 показан сборочный чертеж схемы монтажной для СЭП (рис. 3.5), совмещенный с МСЭМ и результатами автоматической трассировки.

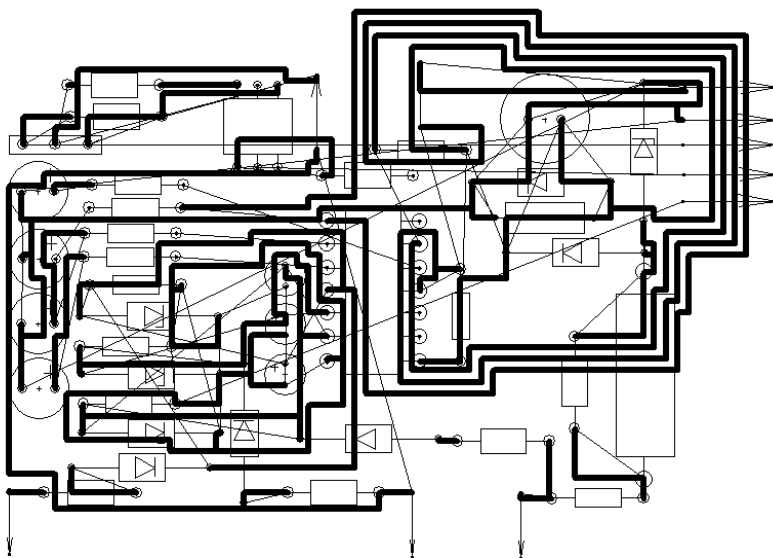


Рисунок 3.7 Сборочный чертеж схемы монтажной

На рис. 3.8 представлен пример объемной геометрической модели печатной платы в сборе.



Рисунок 3.8 Объемная геометрическая модель печатной платы в сборе

3.3.7 Функциональное моделирование схем и устройств

Функциональная модель схем и устройств создается с использованием специальных языков. Ниже приведем краткую информацию по языкам моделирования ABEL [56] и VHDL [66].

Язык описания схем ABEL (Advanced Boolean Equation Language) предназначен для описания моделей логических схем реализуемых на программируемых логических устройствах (ПЛУ) или других устройствах.

Структура программы на языке ABEL содержит заголовок, включающий имя программы и комментарии, описания входов и выходов реализуемых логических функций, описания собственно логических функций, объявления типа ПЛУ или другого устройства, в котором должны быть реализованы указанные логические функции. Кроме того, вводятся «векторы проверок», которыми задаются ожидаемые значения логических функций.

В середине 80-х годов Министерство обороны США и Институт инженеров по электротехнике и электронике поддерживали разработку довольно мощного языка описания схем VHDL. В 1987 году принят стандарт языка VHDL-87, затем этот стандарт был расширен (VHDL-93) [66].

Отличительными особенностями языка являются:

- возможность иерархического разбиения схемы на составные элементы и создания иерархической базы данных алгоритмических описаний элементов;
- каждый элемент устройства имеет ясно очерченный интерфейс для его соединения с другими элементами и точное функциональное описание для его моделирования;

- язык позволяет моделировать временные характеристики переходных процессов, как в последовательных, так и в параллельных структурах, синхронных и асинхронных схемах.

Пример описания схемы комбинационного умножителя 8х8 на языке VHDL:

<i>library IEEE;</i>	- объявление библиотеки Института стандартов IEEE
<i>use IEEE.std_logic_1164.all;</i>	- серия используемых интегральных схем
<i>use IEEE.std_logic_arith.all;</i>	- стандартные библиотеки программ функционального моделирования
<i>entity vmul8x8i is</i>	- описание объекта моделирования
<i>port (</i>	
<i> X: in UNSIGNED (7 downto 0);</i>	- описание параметров входов и выходов
<i> Y: in UNSIGNED (7 downto 0);</i>	
<i> P: out UNSIGNED (7 downto 0);</i>	
<i>);</i>	
<i>end vmul8x8i;</i>	
<i>architecture vmul8x8i_arch of</i>	- запуск процесса моделирования
<i> vmul8x8i is</i>	
<i>begin</i>	
<i> P <= X * Y;</i>	
<i>End vmul8x8i_arch;</i>	- конец процесса моделирования

Таким образом, основная идея по переходу на электронную документацию и создание математических моделей на различных этапах проектирования изделий, сформулированная в 60-х годах прошлого столетия (см., например, [8, 9]), в 90-х годах была закреплена в виде международных стандартов по CALS - технологии непрерывного компьютерного сопровождения изделия на всех этапах его жизненного цикла от маркетинга до утилизации. В работе показаны возможности создания компьютерных моделей на отдельных этапах ЖЦ РЭА, перечислен набор электронных документов для каждого из этапов, показана возможность формирования функциональной и обобщенной компьютерной модели (ОМ) для разных этапов ЖЦ.

Глава 4.

СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ СИСТЕМ

Варианты структурной организации систем рассматриваются на начальных этапах жизненного цикла создаваемых систем с использованием различных методов структурного проектирования. Особенности используемых методов в существенной степени проявляются на последующих этапах проектирования и формируют соответствующие эксплуатационные характеристики систем такие, как сложность разработки, наладки и использования, надежность, экономичность, безопасность и др.

Формализованные методы структурного проектирования позволяют ускорить процесс разработки систем, улучшить их эксплуатационные характеристики и обеспечить передачу знаний по принципам структурной организации. Эти методы включают процедуры формальной генерации возможных вариантов построения структур систем, их анализа, обоснования и выбора лучшего варианта, а также процедуры регенерации структур, перепроектирования, кодификации и создания руководств по эксплуатации [16].

Методы структурного проектирования базируются на понятиях анализа и синтеза, а также на предпочтениях, акцентах и последовательностях использования этих понятий разработчиками различных методов.

Примерами методов, в которых в большей степени используется анализ структур, генерируемых проектировщиками вариантов структур систем, являются технологии IDEF (Integrated DEFINition) [63, 72], ARIS [41], универсальный язык моделирования UML [30, 31, 57]. Эти методы определяют правила описания структур систем в виде набора взаимосвязанных блоков, позволяют анализировать структуры систем и информационные потоки в них, а также документировать все процессы, происходящие в системах.

К методам, основанным на формализованном синтезе, позволяющем генерировать возможные варианты структур систем по заданным алгоритмам функционирования, относятся методы синтеза автоматизированных систем управления (АСУ) [45, 46]. Эти методы характеризуются хорошими теоретическими обоснования-

ми, широтой постановки задач синтеза, охватывают различные этапы жизненного цикла проектируемых АСУ.

Однако перечисленные методы для современного этапа развития информационных технологий имеют ряд недостатков. К ним можно в первую очередь отнести практическое игнорирование структур данных и их возможных преобразований, особенно это касается сложных структур данных CAD/CAM/PDM – систем. Во вторую очередь – отсутствие каких-либо средств систематизации реализаций типовых операций в алгоритмах функционирования комплексов и систем, за некоторым исключением может быть универсального языка моделирования UML. В третью – отсутствие методов формализованной генерации вариантов построения комплексов и систем на уровне структур алгоритмов. В четвертую – отсутствие формального определения минимальной неделимой части проектируемой системы (модуля), на которые может производиться декомпозиция систем.

4.1 Метод синтеза структур интерактивных систем

Перечислим основные особенности разработанного автором метода синтеза структур интерактивных систем (ИС) [25, 69].

Исходной информацией при проектировании являются технические требования на разработку ИС, включающие характеристики алгоритма функционирования и рекомендации по используемым элементам и блокам систем. Проектирование начинается с выбора показателя оптимальности проектируемого ИС.

На первой стадии проектирования структур проводятся операции со структурой алгоритма функционирования ИС. Вводятся формальные определения наименьшей неделимой части алгоритма – локального алгоритма (ЛА) и соответствующего ему множества локальных структур (ЛС). Общий алгоритм функционирования вначале, по определенным правилам, разделяется на локальные алгоритмы. Образуется первый вариант структуры общего алгоритма из набора ЛА. Затем производится последовательное объединение ЛА и образование новых локальных алгоритмов для их последующей реализации в виде новых множеств ЛС. Операция объединения ЛА продолжается до тех пор, пока все ЛА не будут объединены в одном общем алгоритме функционирования систе-

мы. В результате проведения этих преобразований образуются различные варианты сочетаний локальных алгоритмов. Для каждого локального алгоритма определяются его основные характеристики: набор операций, точность представления, типы входов и выходов операндов. Полученная информация является исходной для второй и третьей стадий проектирования структур ИС.

На второй стадии, стадии формирования локальных структур (ЛС), проводится систематизация множества возможных вариантов реализации каждого ЛА. Для каждого ЛА строится модель ЛС, представленная в виде структурной сети, вершинами которой являются возможные формы представления информации, а дугами – значения качественных показателей соответствующих реализаций.

На третьей стадии для выбора лучшей реализации структуры ИС предусматривается построение обобщенной структуры модели (ОСМ) ИС, состоящей из комбинации моделей ЛС, образованных на второй стадии синтеза структур. Лучшая структура ИС определяется как кратчайший путь от входа к выходу в сети обобщенной модели ИС, в которой дуги сети соответствуют ЛС, а их вес определяется значением выбранного качественного показателя ЛС. Окончательный выбор варианта структуры производится разработчиком на основании сведений, полученных на третьей стадии с учетом дополнительных требований на разработку ИС и возможности его технической реализации.

Рассматриваемый метод синтеза структур ИС позволяет определить источники образования различных вариантов структурной организации систем, установить взаимосвязи между характеристиками алгоритмов функционирования систем и реализациями алгоритмов, формализовать процесс генерации множества вариантов структурной организации ИС.

В разработанном методе источниками многовариантности являются:

1. Операции со структурами алгоритмов: декомпозиция алгоритмов на части и их объединения.
2. Разнообразие реализаций частей алгоритмов.
3. Множество характеристик объектов информации (операндов).
4. Преобразование характеристик объектов информации и систематизация реализаций.
5. Критерии оценки вариантов построения систем.

4.2 Операции со структурами алгоритмов

В большей части известных методов проектирования структур систем предусматривается предварительное разделение (декомпозиция) алгоритмов функционирования систем на минимально неделимые части, позволяющее определить возможные варианты их структурной реализации, понять принципы функционирования систем и оптимизировать эксплуатационные характеристики систем. Такое разделение алгоритмов используется как при анализе, так и синтезе систем.

При выборе минимально неделимых частей существуют два крайних случая, первый – связан с очень детальным делением алгоритма, вплоть до отдельных операций, тогда анализ алгоритмов будет затруднен из-за обилия мелочей, не характеризующих общий принцип функционирования проектируемых систем, а стоимость реализации не окажется меньшей. Второй случай связан с тем, что желание слишком укрупнить части алгоритма приведет к потере важной информации, определяющей лучший вариант реализации системы.

В некоторых работах (например, в [32]) декомпозицию структур алгоритмов предлагается проводить по принципу связности операций в алгоритмах. Однако, если учесть, что реализации операций в алгоритмах функционирования систем в общем случае могут различаться не только количественными показателями операндов, но и внутренними и внешними их характеристиками, то оказывается, что помимо связности операций необходимо принимать во внимание следующие характеристики операндов (см. раздел 2.1):

- А - множество способов внутренней организации операндов или их способов кодирования;
- Ф - множество форм внешнего представления, используемых для согласования реализаций отдельных операций в общем алгоритме функционирования;
- Δ - множество степеней детализации количественных показателей операндов или множество точностей их представления.

Тогда, учитывая характеристики операндов, минимально неделимая часть алгоритма и ее реализация формально могут быть определены следующим образом.

Определение 1. Связанную часть алгоритма последовательного выполнения k операций $S = \overline{(s_1, s_2, \dots, s_k)}$ над операндами с $\alpha_i = \text{const}$ ($\alpha_i \in A$) и $\delta_i = \text{const}$ ($\delta_i \in \Delta$) $\forall i \in \overline{(s_1, s_2, \dots, s_k)}$ назовем «локальным алгоритмом» (ЛА).

Определение 2. Реализацию ЛА в виде технических или программных средств назовем локальной структурой (ЛС).

Понятие «локальная структура» аналогично известному из литературы понятию «модуль». Разница между этими понятиями заключается в том, что в ЛС заведомо определены степень детализации операндов (точность представления информации) и их способ внутренней организации (кодирования). Последнее дает возможность точно сформулировать правила первоначального разделения алгоритма функционирования на локальные, сократить и упорядочить перебор общего числа вариантов структурной организации систем.

Покажем далее, что минимальной неделимой частью алгоритма является ЛА, а его реализацией – множество ЛС. Доказательство проведем применительно к программно технической реализации алгоритмов.

Пусть в алгоритме содержится $S = \sum_{i=1}^k S_i$ операций, где S_i – число операций i -го типа, k – число типов операций.

Сравним стоимости двух вариантов реализации. В варианте а) каждая операция выполняется в отдельном операционном блоке (устройстве, подпрограмме, подсистеме, системе). В варианте б) операции одного типа выполняются последовательно в одном специализированном операционном блоке.

Будем полагать, что любые операции, в том числе ввод, вывод, запись и чтение из памяти, преобразование информации и т.п., реализуются в операционных блоках. При этом не учитывается стоимость реализации операций управления процессом выполнения алгоритма, поскольку она зависит только от общего числа

операций S , которое является постоянным. Тогда стоимость выполнения алгоритма в реализации типа а)

$$Q_1 = \sum_{i=1}^k \sum_{j=1}^{S_i} q_{oi} W_{ji}, \quad (4.1)$$

где q_{oi} – стоимость реализации одного разряда операции i -го типа;

W_{ji} – сложность кодирования операндов в j -ом операционном блоке i -го типа.

Определение 3. Сложностью кодирования $W_i^{\alpha_i}$ операнда p_i в коде $\alpha_i \in A$ с точностью $\delta_i \in \Delta$ назовем полное количество двоичных разрядов, требуемое для представления $\forall p_i \in N$ на технических средствах системы.

Для позиционной системы счисления с основанием α_i имеем

$$W_i^{\alpha_i} = \left\lceil \log_{\alpha_i} \frac{1}{\delta_i} \right\rceil \log_2 \alpha_i, \quad (4.2)$$

где $\lceil X \rceil$ – означает ближайшее целое число к X .

Стоимость реализации алгоритма в ПО типа б)

$$Q_2 = \sum_{i=1}^k q_{oi} W_{imax} + q_{mo} \sum_{i=1}^k (S_i - 1) W_{imax} \quad (4.3)$$

где q_{mo} – стоимость реализации одного элемента памяти,

$$W_{imax} = \max \{W_{ji}\} \text{ для } \forall i \in \{1, \dots, S_i\}.$$

Определим, при каких условиях лучшей реализацией будет ПО типа а), т.е. разделение на ЛА будет целесообразно, если $Q_1 < Q_2$

$$\sum_{i=1}^k \sum_{j=1}^{S_i} q_{oi} W_{ji} < \sum_{i=1}^k q_{oi} W_{imax} + q_{mo} \left(\sum_{i=1}^k (S_i - 1) \right) W_{imax} \quad (4.4)$$

Для выполнения условия достаточно, чтобы имело место

$$\sum_{j=1}^{S_i} q_{oi} W_{ji} < q_{oi} W_{imax} + q_{mo} W_{imax} + q_{mo} W_{imax} (S_i - 1), \text{ для } \forall i \in \{1, \dots, k\}$$

$$\sum_{j=1}^{S_i} W_j < W_{i\max} \left(1 + \frac{q_{mo}}{q_{oi}} (S_i - 1) \right) \quad (4.5)$$

Таким образом, действительно, соотношение (4.5), учитывая (4.2), выполняется только при $\delta_i \neq const$ и/или $\alpha_i \neq const \forall i \in \{1, \dots, S_i\}$. Правая часть соотношения (4.5) оказывается меньше только при постоянных значениях δ_i и α_i , поскольку стоимость разряда памяти q_{mo} меньше стоимости разряда операционного блока q_{oi} . Приведенное доказательство позволяет сформулировать первое правило в операциях со структурами алгоритмов – правило первоначального разделения структуры алгоритма на локальные (ЛА).

Правило 1. Первоначальное разделение алгоритма на локальные алгоритмы (ЛА) выполняется при наличии хотя бы одного из следующих условий: отсутствие связи с предыдущими операциями, $\delta_i \neq const$, $\alpha_i \neq const \forall i \in \{1, \dots, S_i\}$.

Операция разделения структуры алгоритма на ЛА иногда приводит к ситуации, когда возникает неопределенность по выбору способа кодирования α_i и форм $\varphi_i \in \Phi$ внешнего представления объектов информации между ЛА. Разработаны правила выбора параметров α_i и φ_i , в соответствии с которыми, например, для специализированных вычислительных устройств, при наличии неопределенности по выбору способа кодирования промежуточной информации предпочтение следует отдавать двоичному коду, а форму внешнего представления – унитарному (число – импульсному) коду.

После операций разделения структуры алгоритма на ЛА образуется первый вариант структуры алгоритмов системы, в котором находятся ЛА, связанные друг с другом по информации. Этот вариант может быть подвергнут дальнейшему преобразованию путем объединения ЛА в более крупные с целью образования других возможных вариантов реализации систем. Объединенные ЛА в свою очередь являются новыми локальными алгоритмами. Операции объединения ЛА производятся многократно. Каждому ЛА со-

ответствует множество локальных структур. По мере укрупнения ЛА уменьшается количество блоков структуры и увеличивается их сложность.

Операции объединения проводятся по следующим правилам.

Правило 2. При объединении двух или нескольких локальных алгоритмов способы кодирования информации должны быть приведены к единой форме. Из двух способов кодирования выбирается способ, определенный техническим заданием на проектирование системы. Невозможно объединение двух ЛА с заранее заданными в техническом задании кодами.

Правило 3. При объединении двух и более ЛА с различными точностями представления информации точность представления данных в обобщенном алгоритме выбирается как $\delta = \min\{\delta_i\}$, $\forall i \in \{1, \dots, n\}$, где n – число объединяемых ЛА.

При объединении возникает новый ЛА с последовательным выполнением операций. Новому ЛА соответствует новая локальная структура, имеющая в общем случае увеличенный объем памяти по отношению к первоначальным ЛС. Основные параметры операторов этого ЛА определяются из технического задания или по указанным правилам. Параллельные алгоритмы могут быть преобразованы в последовательные, если это допустимо с точки зрения времени вычислений.

Таким образом, за счет операций с алгоритмами генерируется множество вариантов структурной организации систем. Количество вариантов структур не превосходит общего числа сочетаний ЛА, полученных при начальном разделении.

Наличие множеств значений A , Φ и Δ характерно в большей степени для интерактивных систем [4], в которых информация по входу и выходу связана с датчиками технологических процессов, исполнительными механизмами, средствами интерактивного взаимодействия и другими системами. Для таких систем стоимость вариантов структурной организации может быть представлена в более детализированном виде

$$Q_1 = Q_o + Q_{mo} + Q_{tr} + Q_m + Q_{in} + Q_{ou} + Q_{go} + Q_{mg} + Q_{co} =$$

$$\sum_{i=1}^k \sum_{j=1}^{S_i} q_{oi} W_{oji} + \sum_{i=1}^k q_{moi} W_{moi} S_i + \sum_{i=1}^{k-1} q_{tri} W_{tri} +$$

$$+ q_m W_m + q_{in} W_{in} + q_{ou} W_{ou} + q_{go} W_{go} + q_{mg} W_{mg} + S q_{co} W_{co}, \quad (4.6)$$

где $Q_o, Q_{mo}, Q_{tr}, Q_m, Q_{ex}, Q_{ou}, Q_{go}, Q_{mg}, Q_{co}$ – стоимость операционных блоков, памяти в операционных блоках, блоков преобразования, общей памяти, блоков ввода и вывода информации, графического операционного блока и графической памяти, блока управления;

$q_{oi}, q_{moi}, q_{tri}, q_m, q_{in}, q_{ou}, q_{go}, q_{mg}, q_{co}$ – стоимость реализации одного разряда операции i -го типа, соответственно для тех же блоков;

$W_o, W_{mo}, W_{tr}, W_m, W_{in}, W_{ou}, W_{go}, W_{mg}, W_{co}$ – сложность кодирования операндов, соответственно для тех же блоков.

Из выражения (4.6) следует:

- при объединении двух операционных блоков (сокращении числа k) возрастает стоимость Q_{mo} за счет увеличения ячеек памяти в операционных блоках и необходимости выбора $W_{mo \max}$ из двух возможных;
- передача информации между операционными блоками производится через преобразователи способов кодирования (Q_{tr}), чем больше число операционных блоков с разными способами кодирования, тем больше преобразователей,
- стоимости $Q_m, Q_{in}, Q_{ou}, Q_{go}, Q_{mg}, Q_{co}$ не зависят от взаимосвязей операционных блоков.

4.3 Формирование обобщенной структурной модели ИС

Обобщенной структурной моделью (ОСМ) будем называть совокупность ЛС, упорядоченную структурой алгоритма функционирования и охватывающую на уровне ЛС возможные структуры данных и реализации отдельных операций. Задачу поиска

лучшей архитектуры можно свести к построению сети обобщенной структурной модели, определению вершин сети (структур данных) и ребер (качественных показателей реализации отдельных операций), нахождению кратчайшего пути в сети.

Фрагмент структуры обобщенной модели, например, для двух способов кодирования α_1 и α_2 показан на рис. 4.1. Левая сторона сети является входом, правая – выходом.

Множества локальных структур $ЛС_1^1-ЛС_3^1$ и $ЛС_1^2-ЛС_3^2$ реализуют локальные алгоритмы для способов кодирования α_1 и α_2 соответственно.

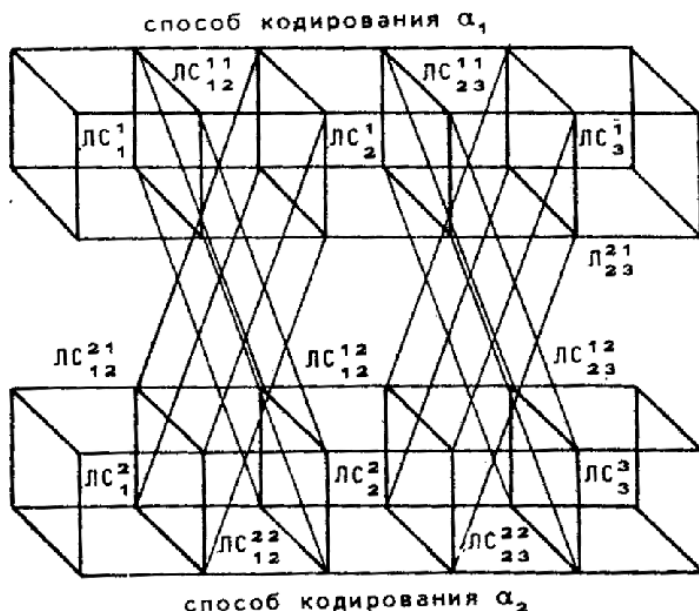


Рисунок 4.1 Фрагмент структуры обобщенной модели

Из рисунка видно, что преобразования данных в обобщенной модели занимают значительное место. Так $ЛС_{12}^{12}$, $ЛС_{12}^{21}$, $ЛС_{23}^{12}$, $ЛС_{23}^{21}$ реализуют ЛА преобразований способов кодирования, вертикальные плоскости в сети представляют преобразования форматов файлов и типов данных. $ЛС_{12}^{12}$ и $ЛС_{23}^{12}$, $ЛС_{12}^{21}$ и $ЛС_{23}^{21}$ являются эквивалентными; $ЛС_{12}^{11}$, $ЛС_{23}^{11}$, $ЛС_{12}^{22}$, $ЛС_{23}^{22}$ отображают

связи между вершинами предыдущих и последующих моделей ЛС (значения их дуг равны 0). Качество проектируемой системы в существенной мере зависит от количества преобразователей, используемых не только внутри комплекса, но и для согласования с другими, внешними по отношению к проектируемой, системами. Сокращение общего количества преобразователей возможно за счет использования стандартных структур данных.

Реализация каждой операции на множестве локальных структур заранее готовится в виде программы или аппаратуры, соответственно, подсчитываются и значения качественных показателей каждой реализации.

Характеристики сети преобразования форматов файлов данных могут быть заранее подготовлены, записаны в базу данных и использованы в процессе синтеза интерактивные программные системы (ИПС). Последовательность подготовки сети преобразования включает определение вершин и ребер сети, определение значения качественных показателей реализаций, соответствующих каждому ребру сети.

Для каждой проблемной ориентации структуры сетей могут быть уточнены, причем определение качественных показателей реализаций может производиться как теоретически, так и практически, путем оценки объемов памяти, быстродействия и других характеристик, представляющих интерес.

В табл. 4.1 приведен пример временных затрат при преобразованиях форматов и типов данных в вертикальных плоскостях сети обобщенной модели ИС (рис. 4.1). Информация, представленная в таблице, получена экспериментальным путем для одного типа компьютера. Время указано в условных единицах. В качестве форматов данных рассматривались одноразрядная переменная (O), список (C), таблица (T). Типы данных представлены в виде символов $\{S\}$, целых $\{C\}$ и действительных $\{D\}$ чисел. Параметры I и J указывают длину строки и количество строк соответственно. По вертикали расположены форматы и типы данных на входе, по горизонтали – на выходе блока преобразования. Из рисунка видно, что время преобразования убывает при переходе к более простым форматам данных, от таблиц к спискам и одноразрядным пере-

менным, а также при переходе от символьного к действительному и целочисленному типу данных.

Таблица 4.1

Временные затраты по преобразованию форматов и типов данных

	$T\{C\}$	$T\{D\}$	$T\{S\}$	$C\{C\}$	$C\{D\}$	$C\{S\}$	$O\{C\}$	$O\{D\}$	$O\{S\}$
$T\{C\}$	18IJ	40IJ	400IJ	14IJ	36IJ	380IJ	13IJ	34IJ	380IJ
$T\{D\}$	32IJ	20IJ	1050IJ	29IJ	16IJ	1020IJ	29IJ	16IJ	1020IJ
$T\{S\}$	500IJ	530IJ	500IJ	480IJ	500IJ	480IJ	480IJ	500IJ	530IJ
$C\{C\}$	13IJ	37IJ	27IJ	10I	32I	260I	8I	29I	260I
$C\{D\}$	30IJ	15IJ	900IJ	26I	11I	890I	24I	8I	890I
$C\{S\}$	360IJ	380IJ	360IJ	350I	380I	360I	310I	310I	350I
$O\{C\}$	13IJ	36IJ	250IJ	9I	31I	250I	6	32	25
$O\{D\}$	29IJ	16IJ	880IJ	25I	10I	880I	27	7	880
$O\{S\}$	340IJ	370IJ	350IJ	340I	370I	350I	34	36	36

Далее, для упрощения изображения, обобщенную структурную модель (ОСМ) будем представлять в виде многослойной сети так, как показано на рис. 4.2. Вершинами сети являются формы внешнего представления операндов $\varphi_i \in \Phi$ (на рис. 4.2 обозначены цифрами), дугами – возможные реализации отдельных операций. Дуги сети показывают последовательное выполнение операций и имеют веса, соответствующие выбранным критериям качества реализаций ($Q_o, Q_{tr}, Q_m, Q_{ex}, Q_{ou}$). Овалами обозначены полные

сети преобразований форм внешнего представления операндов. Каждому слою ОСМ соответствуют реализации в одном из кодов $a_i \in A$. Предполагается, что послойные сети связаны между собой через вершины «вход», «выход» и вершины преобразования (отмечены овалами). Построенная таким образом ОСМ, показывает все возможные варианты структурной организации проектируемой системы для одного из сочетаний локальных алгоритмов. Задача выбора лучшей реализации сводится к определению кратчайшего пути от входа к выходу сети ОСМ.

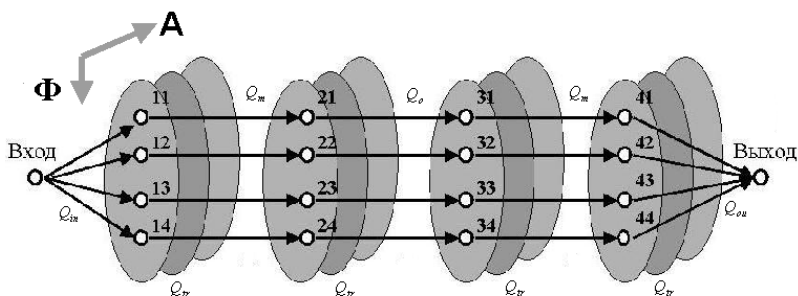


Рисунок 4.2 Обобщенная структурная модель

4.4 Примеры систематизации вариантов реализации локальных алгоритмов

Рассмотрим примеры систематизации вариантов реализации локальных алгоритмов [6, 11]. Допустим, что проектируемая система реализует алгоритм вычисления суммы $X(t)$ двух переменных $N(t)$ и $M(t)$ в моменты времени t

$$X(t) = N(t) + M(t) \quad (4.7)$$

Требуется найти возможные варианты реализации заданного алгоритма для двух классов систем: систем, построенных на элементах вычислительной техники, и систем, реализованных программно. В первом классе переменные $X(t)$, $N(t)$ и $M(t)$ принимают числовые значения, во втором – они представляют собой более сложные структуры данных, например, описывающие вектор-

ные графические изображения. На первый взгляд для такого простого алгоритма вариантов реализации должно быть мало. Однако, с учетом множеств A , Φ , Δ для трех переменных общее число вариантов может оказаться существенным. На рис. 4.3. для системы первого класса при постоянных значениях $\alpha_i \in A$ и $\delta_i \in \Delta$ представлены возможные варианты реализации операции сложения в виде сети, вершинами которой являются формы представления информации $\varphi_i \in \Phi$, дугами – стоимости реализации отдельных операций. Сеть включает части, представляющие возможные преобразования элементов множества Φ ($g_{\text{првх}}$, $g_{\text{првых}}$), и часть, отображающая реализации операций сложения ($g_{\text{сл}}$). В таблице 4.2 указаны реализации, соответствующие цифрам на дугах сети.

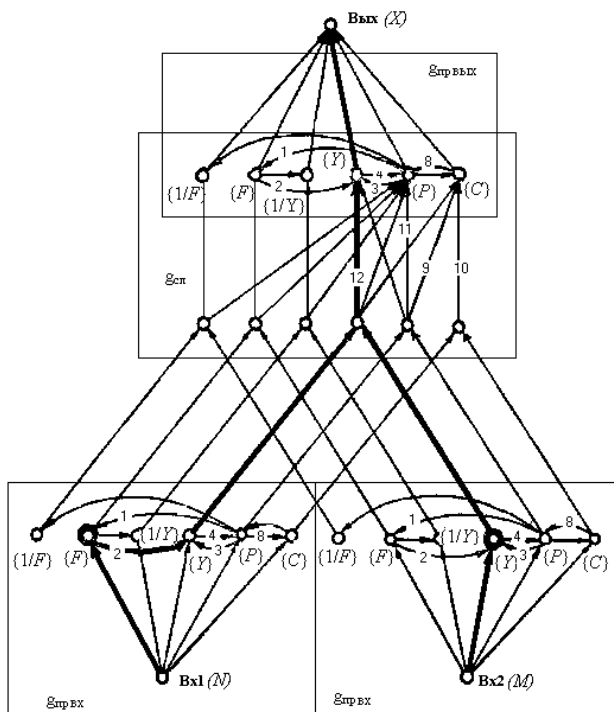


Рисунок 4.3 Систематизация реализаций операции сложения двух чисел

Допустим, что параметр $N(t)$ представлен в виде $\{F\}$ -сигнала, $M(t)$ – в форме $\{Y\}$ -кода, результат $X(t)$ – в форме $\{Y\}$ -кода. Тогда один из вариантов реализации (на рисунке обозначен толстыми стрелками) будет состоять из преобразователя $N(t)\{F\}$ в $N(t)\{Y\}$ и схемы ИЛИ (для последовательного объединения (сложения) $N(t)\{Y\}$ и $M(t)\{Y\}$), на выходе которой образуется результат $X(t)\{Y\}$.

Таблица 4.2

Реализации, соответствующие цифрам на дугах сети

Обо- значе- ние	Наименование реализации
1	Преобразователь $\{P\}$ -кода в $\{F\}$ -сигнал
2	Преобразователь $\{F\}$ -сигнала в $\{Y\}$ -код
3	Преобразователь $\{P\}$ -кода в $\{Y\}$ -код
4	Счетчик с входом или выходом в виде $\{P\}$ -кода
5	Реверсивный счетчик с выходом в виде $\{Y\}$ -кода
6	Регистр параллельного типа
7	Сдвиговый регистр
8	Сдвиговый регистр с входом или выходом в виде $\{P\}$ -кода
9	Сумматор параллельного типа со сдвигом в сторону младших разрядов
10	Сумматор одноразрядный
11	Сумматор параллельного типа
12	Схема ИЛИ

Для программной реализации сложения двух векторных изображений систематизация проведена по способам внутренней организации объектов информации – множеству A форматов данных (рис. 4.4). Выбраны следующие форматы данных: GERBER – (1), HPGL – (2), .dxf – (3), .plt – (4), STEP – (5). Сеть включает части, представляющие возможные преобразования элементов множества A – $g_{\text{првх}}$, $g_{\text{првых}}$ (на рисунке обозначены овалами), и часть,

отображающая реализации операций сложения – $g_{сл}$. Большую часть реализаций преобразований легко найти в сети Internet.

Из рисунка видно, что операция сложения двух векторных изображений реализуется одной из пяти программных систем последовательным вводом параметров $N(t)$ и $M(t)$. Допустим, что параметр $N(t)$ представлен в формате GERBER, $M(t)$ – в формате HPGL, результат $X(t)$ – в формате STEP. Тогда один из вариантов реализации (на рисунке обозначен толстыми стрелками) будет состоять из программ преобразования $N(t)\{GERBER\}$ в $N(t)\{.dxf\}$, $M(t)\{HPGL\}$ в $M(t)\{.dxf\}$, сложения с использованием системы AutoCAD, на выходе которой образуется результат $X(t)\{.dxf\}$, и программы преобразования $X(t)\{.dxf\}$ в $X(t)\{STEP\}$.

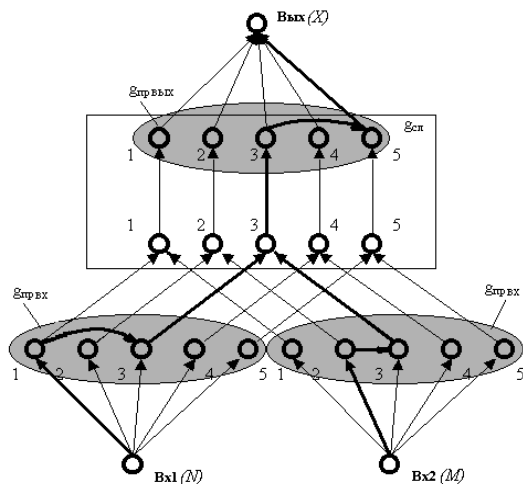


Рисунок 4.4 Систематизация реализаций операции сложения двух векторных изображений

Следует заметить, что современные системы автоматизированного проектирования имеют встроенные преобразователи собственных структур данных в широко используемые структуры и

обратно, особенно это касается структур, признанных стандартными.

Из рассмотрения рис. 4.3 и 4.4 следует, что общее число вариантов реализации систем первого и второго классов в большей степени определяется возможными преобразованиями множеств A и Φ , чем вариантами реализации собственно операции сложения.

Глава 5.

СИНТЕЗ СТРУКТУР ИНТЕРАКТИВНЫХ ТЕХНИЧЕСКИХ СРЕДСТВ

5.1 Систематизация реализаций локальных алгоритмов для интерактивных технических средств с одноразрядными переменными

Рассмотрим сравнительные характеристики реализаций различных локальных алгоритмов интерактивных технических средств (ИТС), в частности, преобразований форм представления информации, операций сложения, умножения и хранения [42, 55, 56]. В качестве способа кодирования ограничимся двоичным кодом, а формы представления информации выберем в виде частотного сигнала $\{F\}$, унитарного $\{Y\}$, параллельного $\{P\}$ и последовательного $\{C\}$ кода.

5.1.1 Анализ и систематизация способов преобразования информации

Среди реализаций, преобразующих частотный сигнал в унитарный код, можно выделить две основные группы, отличающиеся методами преобразования.

К первой группе относятся устройства, в которых подсчитывается число периодов измеряемой частоты $f(t)$ за фиксированный интервал времени измерения T из. Цифровое значение измеряемой частоты подсчитывается в соответствии с выражением

$$N\{F\} = \int_0^{T_{из}} f(t) dt. \quad (5.1)$$

Требования к точности преобразования обычно устанавливаются по отношению к определенному номинальному значению частоты или по отношению к диапазону изменения частот.

При заданной точности преобразования время $T_{из}$ может быть определено из выражений

$$T_{из} = \frac{100}{\delta \cdot f_n(t)} \quad \text{или} \quad T_{из} = \frac{100}{\delta(f_{max} - f_{min})} \quad (5.2)$$

где δ – точность преобразования (в процентах).

$f_n(t)$, f_{\max} , f_{\min} – соответственно номинальное, максимальное и минимальное значения частот.

Преобразование осуществляется при помощи времязадающего блока, состоящего из генератора Γ опорной частоты f_0 и счетчика $COUVP$ (рис. 5.1).

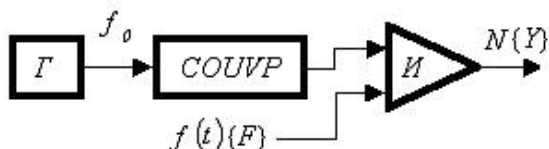


Рисунок 5.1 Преобразование частотного сигнала в унитарный код

На рисунке в фигурных скобках указана форма представления информации. Разрядность двоичного счетчика r_g можно рассчитать следующим образом

$$r_g = \frac{\lg T_{из} \cdot f_0}{\lg 2} \quad (5.3)$$

Ко второй группе относятся устройства, в которых осуществляется измерение одного или нескольких n периодов преобразуемого сигнала $f(t)$ путем заполнения этих периодов импульсами эталонной частоты f_0 . Цифровое значение преобразуемого сигнала вычисляется следующим образом

$$N \left\{ \frac{1}{y} \right\} = f_0 \frac{n}{f(t)} = f_0 T. \quad (5.4)$$

В дальнейшем форму представления информации в виде унитарного кода, значение которого обратно пропорционально измеряемому сигналу, будем обозначать через $\{1/Y\}$.

Этот метод преобразования реализуется при помощи делителя *COUVP* входного частотного сигнала $f(t)$ и генератора Γ эталонной частоты f_0 (рис. 5.2).

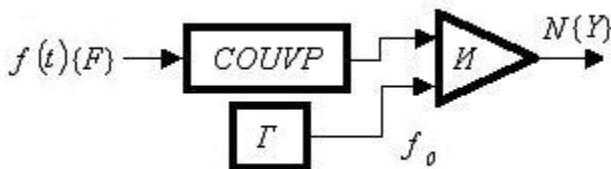


Рисунок 5.2 Преобразование периода частотного сигнала в унитарный код

Значение величин T , n , f_0 находятся из условий обеспечения заданной точности измерения

$$T \cdot n \cdot f_0 = \frac{100}{\delta}. \quad (5.5)$$

Разрядность счетчика – делителя соответственно определяется выражением

$$r_g = \frac{\lg T \cdot n \cdot f_0}{\lg 2}. \quad (5.6)$$

Практически разница в количестве оборудования преобразователей обоих типов определяется счетчиками-делителями.

На основании полученных выражений для времени преобразования и разрядности счетчиков можно определить качественные показатели оборудования узлов, обеспечивающих преобразование частоты в унитарный код.

Унитарный код непосредственно преобразуется только в **параллельный код**. Такое преобразование возможно при помощи счетчиков. Количество разрядов счетчиков зависит от максимального значения числа, представленного в унитарном коде N_{\max} и для двоичного кода определяется по формуле

$$r = \frac{\lg N_{\max}}{\lg 2}. \quad (5.7)$$

Время преобразования $T_{\text{пр}}$ зависит от частоты следования импульсов и способов построения счетчиков, пренебрегая последними, получим

$$T_{\text{пр}} = \frac{1}{f_0} N_{\max}. \quad (5.8)$$

Время-импульсные сигналы непосредственно преобразуются в **унитарный код**. Реализация этого преобразования представляет собой схему совпадения, на которую поступает преобразуемый сигнал $T(t)$ и импульсы частотой f_0 от генератора. Время преобразования фактически равно максимальной длительности T_{\max} преобразуемого сигнала. Частота f_0 генератора выбирается из условия обеспечения заданной точности преобразования δ

$$f_0 = \frac{100}{\delta \cdot T_{\max}}. \quad (5.9)$$

Последовательный код непосредственно преобразуется в **параллельный код**. Любые другие преобразования возможны только через промежуточные преобразования последовательного кода в параллельный. Реализация такого преобразования осуществляется при помощи сдвигового регистра (в общем случае реверсивного).

Количество разрядов $r_{\text{ср}}$ регистра зависит от точности представления информации в регистре, то есть от максимального значения числа. Для двоичного кода справедливо выражение (6.7). Время преобразования последовательного кода определяется длиной слова и способом построения сдвигового регистра и может изменяться в диапазоне

$$\text{от } T_{\text{пр}} = \frac{2r_{\text{ср}}}{f_{\text{си}}} \text{ до } T_{\text{пр}} = \frac{4r_{\text{ср}}}{f_{\text{си}}}, \quad (5.10)$$

где $f_{\text{си}}$ — частота следования импульсов синхронизации.

Числа, представленные **параллельным кодом**, преобразуются в последовательный, унитарный коды и в частотный сигнал.

Преобразование **параллельного кода в последовательный** производится двумя способами. В первом – используется сдвиговый регистр. Под действием синхронизирующих импульсов информация в регистре последовательно передвигается от разряда к разряду. Для сохранения исходной информации замыкается выход регистра с его входом. Во втором способе используется регистр без сдвига информации, а также логические схемы совпадения и одна схема ИЛИ, обеспечивающие последовательный выбор разрядов этого регистра. Для этого способа преобразования применяется блок, осуществляющий временное разделение синхронизирующих импульсов.

Число разрядов и время преобразования для этих способов преобразования определяются по формулам (5.7, 5.8).

Преобразование **параллельного кода в унитарный** производится при помощи схем, представленных на рис. 5.3, 5.4.

Первая схема состоит из счетчика *COUVP*, дифференцирующей цепочки *Дц*, триггера управления *Тр* и схемы совпадения *И*. Преобразуемое число вводится в счетчик параллельным кодом таким образом, чтобы в счетчике число представлялось дополнительным кодом.

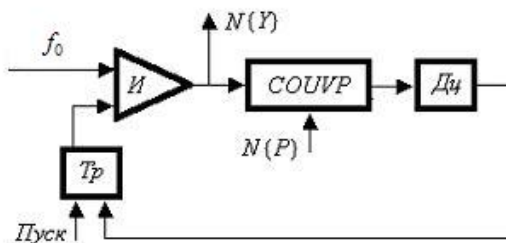


Рисунок 5.3 Первая схема преобразования параллельного кода в унитарный

При срабатывании триггера управления по команде "пуск" на вход первого разряда счетчика через схемы совпадения поступают импульсы с частотой f_0 от генератора, которые добавляются к

ранее записанному числу. Импульс переполнения счетчика устанавливает триггер управления в исходное положение. Количество импульсов, прошедшее через схему совпадения, равно числу, введенному в счетчик параллельным кодом.

Вторая схема преобразования (рис. 5.4) использует счетчик *COUVP*, схему совпадения *И*, триггер управления *Тр* и схему сравнения *СС*. Преобразуемое число хранится в регистре памяти *РР*. В исходном состоянии все разряды счетчика находятся в "0". При срабатывании триггера управления по команде "пуск" на вход первого разряда счетчика через схему совпадения поступают импульсы f_0 от генератора.

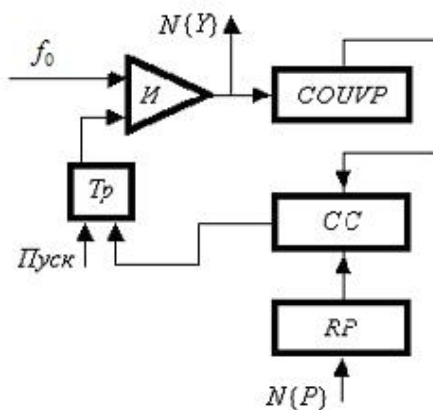


Рисунок 5.4 Вторая схема преобразования параллельного кода в унитарный

Схема сравнения фиксирует момент совпадения чисел, записанных в регистре и в счетчике. В этот момент триггер управления возвращается в исходное состояние. Количество импульсов, прошедших через схему совпадения, равно числу $N\{Y\}$, записанному в регистре.

Число разрядов счетчика и время преобразования в обоих случаях определяется по формулам (5.7, 5.8).

Преобразование параллельного кода в унитарный, как это следует из вышерассмотренных схем, производится с использованием

предварительного преобразования во время-импульсный сигнал Θ . При этом длительность импульса пропорциональна исходному числу, представленному параллельным кодом.

Преобразование **параллельного кода в частотный сигнал** в общем случае может осуществляться двумя способами, в которых выходным сигналом является величина прямо пропорциональная и обратно пропорциональная числу, представленному параллельным кодом.

Реализация, использующая первый способ преобразования, содержит счетчик с логическими схемами. На вход счетчика поступают импульсы частотой f_0 от генератора. К входам логических схем присоединяются выходы схем переноса разрядов счетчика и разряды преобразуемого кода (рис. 5.5).

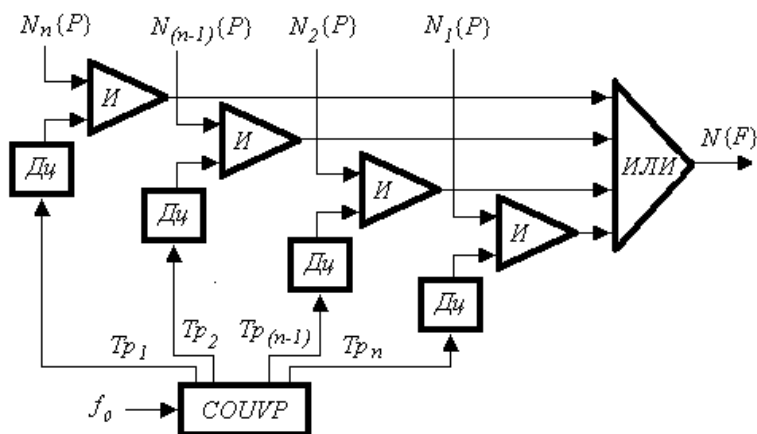


Рисунок 5.5 Первый вариант преобразования параллельного кода в частотный сигнал

Сигналы перехода триггеров счетчика из состояния "0" в состояние "1" поступают на схемы совпадения. На другой вход схем совпадения подаются потенциалы преобразуемого кода $N\{P\}$, причем сигнал от младшего разряда T_{p1} поступает на ту же схему совпадения что и сигнал старшего разряда кода $N\{P\}$. Выходной сигнал $N\{F\}$ снимается на выходе схемы ИЛИ. Особенностью такого способа преобразования является тот факт, что преобразование

можно считать выполненным с точностью до $1/2^n$, если на вход счётчика поступило 2^n импульсов частоты f_0 . В противном случае преобразование осуществляется с большей погрешностью.

Помимо преобразования чисел, представленных двоичным кодом, возможно также преобразование чисел в двоично-десятичных кодах.

Второй способ преобразования числа, представленного параллельным кодом, когда выходной частотный сигнал обратно пропорционален числу, производится при помощи счётчика и логических схем (рис. 5.6).

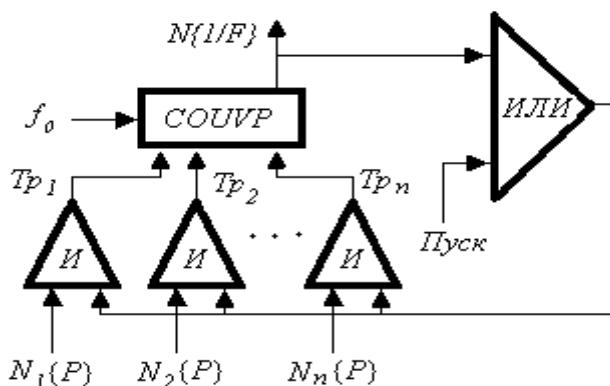


Рисунок 5.6 Второй вариант преобразования параллельного кода в частотный сигнал

Принцип работы преобразователя заключается в исходной установке счётчика на триггерах $T_{p1} \div T_{pn}$, записи числа $N\{P\}$ в дополнительном коде по сигналу "пуск" через схемы совпадения. Далее к числу $N\{P\}$ добавляются импульсы с частотой f_0 .

При заполнении счётчика фиксируется момент перехода T_p из "1" в "0" и снова записывается в счётчик число через схемы совпадения.

Значение частотного сигнала $N\{1/F\}$ на выходе преобразователя будет соответствовать выражению

$$N \left\{ \frac{1}{F} \right\} = \frac{f_0}{N \{P\}} \quad (5.11)$$

Такого типа преобразователи в литературе известны под названиями счетчики с обратными связями [7, 13]. Эти счетчики используются как делители частоты с постоянным коэффициентом деления.

Во втором способе преобразования независимо от типа используемого кода частотный сигнал равномерно распределяется во времени. Время преобразования числа определяется требуемой точностью преобразования или точностью представления чисел и может быть выражено следующим образом

$$T = \left(\frac{100}{\delta} \right)^2 \cdot \frac{1}{f_0}. \quad (5.12)$$

Предполагая, что при необходимости легко можно провести систематизацию преобразований входной и выходной информации представленной в виде других физических величин, в качестве примера рассмотрим возможные варианты преобразования аналоговых сигналов в виде тока или напряжения. Остановимся на некоторых характеристиках основных способов преобразования этих сигналов в перечисленные выше формы представления информации: частотный сигнал $\{F, 1/F\}$, унитарный $\{Y\}$, параллельный $\{P\}$ и последовательный $\{C\}$ коды.

Преобразование **напряжения (или тока) в частотный сигнал** возможно при помощи управляемых генераторов. Выходной сигнал в частотной форме в таком преобразователе прямо пропорционален входному напряжению $u_{\text{вх}}$

$$f = k \cdot u_{\text{вх}} + b. \quad (5.13)$$

Возможен второй способ преобразования в частотный сигнал $(1/F)$. В этом случае выходной сигнал обратно пропорционален приложенному на вход напряжению $u_{\text{вх}}$

$$f = \frac{b}{u_{\text{ex}}} . \quad (5.14)$$

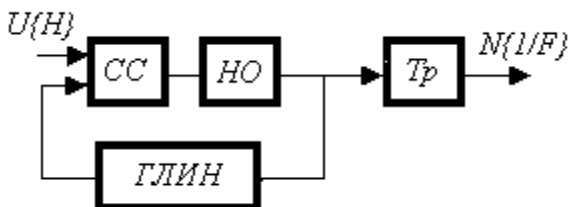


Рисунок 5.7 Преобразователь напряжения в частотный сигнал

Реализация такого преобразователя использует генератор линейно изменяющегося напряжения (ГЛИН), схему сравнения (СС), нуль-орган (НО) и триггер (Тр) со счетным входом (рис. 5.7). Принцип работы преобразователя заключается в многократном сравнении линейно изменяющегося напряжения с входным напряжением u_{ex} . При равенстве этих двух напряжений срабатывает нуль-орган, напряжение на выходе генератора уменьшается до начального, нуль-орган возвращается в исходное состояние. Далее снова запускается генератор. Выходной частотный сигнал снимается с триггера, который переходит в новое состояние всякий раз при срабатывании нуль-органа.

Этот преобразователь может быть использован при преобразовании напряжения в длительность импульса $U\{T\}$. В этом случае схема запускается от внешнего источника. С целью повышения точности преобразования иногда дополнительно в схему вводится нуль-орган, срабатывающий на начальном участке линейно изменяющегося напряжения. Выходной сигнал – длительность импульса $U\{T\}$ снимается с выхода триггера, срабатывающего последовательно при изменении состояния второго и первого нуль-органов.

Преобразователи напряжения в цифровую форму, у которых выходной сигнал может быть представлен в унитарном и параллельном кодах, могут быть выполнены по различным схемам. В последнее время, в связи с бурным развитием микроэлектроники, наибольшее распространение получил способ преобразования, ос-

нованный на двойном интегрировании. Схема такого преобразователя представлена на рис. 5.8. Он состоит из интегрирующего усилителя ИУ, нуля-органа НО, логической схемы И, счетчика импульсов *COUVP*, ключа *K* и источника эталонного напряжения $U_{\mathcal{Z}}$.

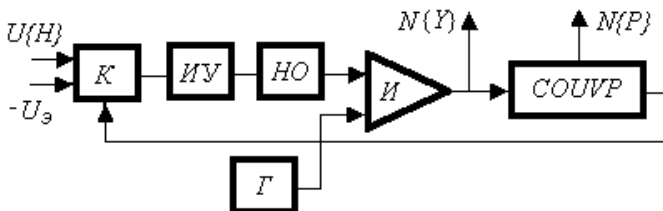


Рисунок 5.8 Первый вариант преобразования напряжения в цифровую форму

Принцип действия преобразователя заключается в следующем. В начале преобразования на вход счетчика за время T_3 поступают импульсы частоты f_0 от генератора до полного заполнения счетчика. За это же время через ключ *K* к интегрирующему усилителю подключается входное напряжение $u_{\text{вх}}$. Выходное напряжение усилителя при этом пропорционально $u_{\text{вх}} \cdot T_3$. В момент заполнения счетчика схема управления подключает через ключ **K** к интегрирующему усилителю эталонное напряжение $U_{\mathcal{Z}}$. Напряжение на выходе усилителя начинает уменьшаться. Ноль-орган фиксирует момент времени, при котором $U_{\mathcal{Z}} \cdot T_{np} = u_{\text{вх}} \cdot T_3$. Этот момент времени соответствует концу преобразования. За время T_{np} на вход счетчика также поступали импульсы частоты f_0 . Число в счетчике в конце преобразования соответствует

$$N\{P\} = T_{np} \cdot f_0 = \frac{u_{\text{вх}}}{u_{\mathcal{Z}}} \cdot T_3 \cdot f_0 \quad (5.15)$$

Время преобразования для такой схемы целесообразно оценивать величиной $2T_3$. Точность преобразования составляет 0,1 %.

Другим примером преобразователей напряжения в сигнал, представленный унитарным и параллельным кодом, может служить преобразователь с обратной связью (рис. 5.9).

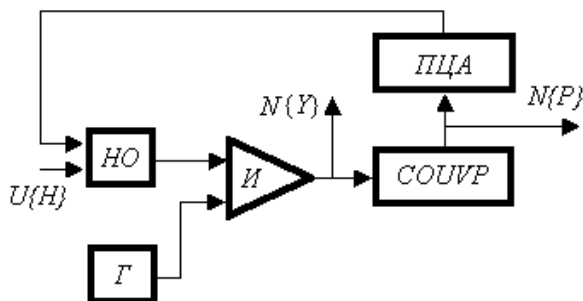


Рисунок 5.9 Второй вариант преобразования напряжения в цифровую форму

Преобразователь состоит из нуля-органа *HO*, генератора импульсов *Г*, счетчика *COUVP* и преобразователя цифро-аналогового ЦЦА. На счетчик в заданные моменты времени подаются импульсы с генератора. Выходное напряжение цифро-аналогового преобразователя ступенчато возрастает. Ноль-орган отключает схему в момент совпадения входного напряжения с напряжением обратной связи. В процессе преобразования на счетчик в унитарном коде поступает число, соответствующее входному напряжению. В конце преобразования со счетчика может быть снята информация в параллельном коде.

Более быстродействующие преобразователи аналогового сигнала в цифровой код могут быть построены по принципу слежения. В этом преобразователе в отличие от предыдущего используется реверсивный счетчик. На шины сложения или вычитания счетчика в зависимости от состояния нуля-органа непрерывно поступают импульсы. В результате выходное напряжение с цифро-аналогового преобразователя непрерывно "следит" за входным напряжением, а состояние счетчика соответствует входному напряжению. Выходная информация представляется в параллельном коде.

К группе преобразователей напряжения в параллельный код могут быть отнесены также преобразователи поразрядного коди-

рования. Примером таких преобразователей может служить схема, представленная на рис. 5.10.

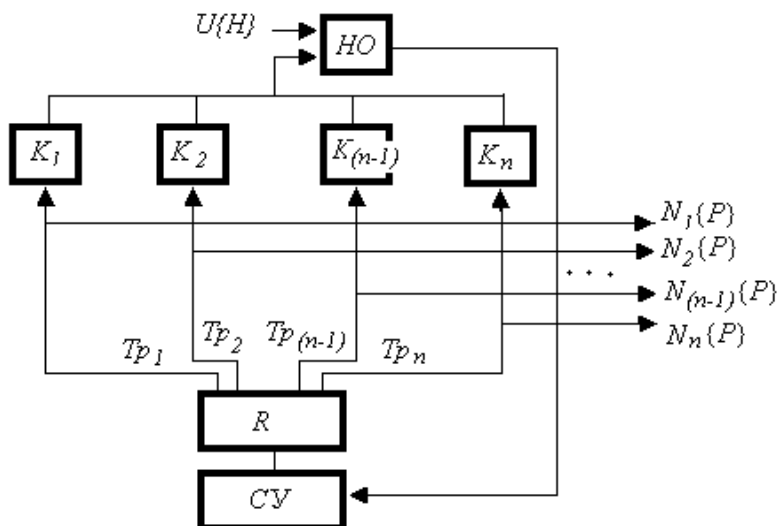


Рисунок 5.10 Преобразователь напряжения в параллельный код на основе метода поразрядного кодирования

Преобразователь состоит из схемы управления $СУ$, регистра R , нуля-органа HO и ключей K_1, K_2, \dots, K_n , обеспечивающих включение i -го эталонного напряжения с весом 2^i в зависимости от состояния i -го разряда регистра. Принцип работы преобразователя заключается в уравнивании входного напряжения U суммой эталонных напряжений.

Процесс преобразования начинается со сравнения входного напряжения с эталоном максимального веса. В зависимости от срабатывания нуля-органа в соответствующих разрядах регистра записывается 0 или 1. Сравнение напряжений и фиксация состояний регистра далее производятся последовательно по разрядам. Число, пропорциональное входному напряжению, считывается с выхода регистра R в конце преобразования. Очевидно, что на основе такой схемы преобразования может быть осуществлено также преобразование напряжения в последовательный код.

Наиболее быстродействующими устройствами являются преобразователи, использующие метод считывания. При этом производится непосредственное преобразование входной величины в параллельный код. Схема такого преобразователя показана на рис.5.11.

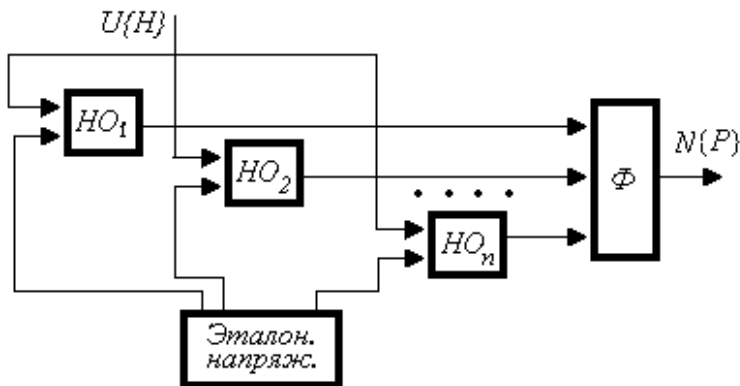


Рисунок 5.11 Преобразователь напряжения в параллельный код на основе метода непосредственного считывания

Преобразователь состоит из нуль-органов $HO_1 \div HO_n$, количество которых равно необходимому числу разрядов выходного кода, датчиков эталонных напряжений и формирователя кода Φ . Принцип работы преобразователя заключается в одновременном сравнении входного напряжения с эталонными напряжениями.

Рассмотренные типы аналого-цифровых преобразователей относятся примерно к одному классу по точности преобразования и перекрывают диапазон до 0,1 %.

Сравнение отдельных типов преобразователей может быть произведено по различным качественным показателям: быстродействию, количеству оборудования, сложности наладки и т.п.

Далее остановимся на некоторых методах преобразования сигналов, представленных в частотной форме $\{F, 1/F\}$ или в виде параллельного кода в аналоговые сигналы (ток или напряжение). Непосредственные преобразования других форм представления ин-

формации (в виде унитарного или последовательного кодов) в настоящее время затруднены.

Преобразование частотных сигналов $\{F, 1/F\}$ и сигналов в виде длительности импульсов $\{T\}$ в аналоговую форму осуществляется за счет сглаживающих фильтров (дуга 13 на рис. 5.12). При этом сигналы в виде длительности импульсов подаются на сглаживающий фильтр периодически с постоянной частотой следования. Расход оборудования и динамические характеристики для этих видов преобразования определяются характеристиками сглаживающих фильтров.

Преобразование сигналов, представленных в параллельном коде, осуществляется при помощи регистра памяти и ключей, которые подключают эталонные токи или напряжения в соответствии с весами включенных разрядов регистра (дуга 14 на рис. 5.12). Эталонные аналоговые сигналы суммируются на общей нагрузке.

5.1.2 Систематизация способов преобразования информации

На примере рассмотренных выше реализаций покажем возможность систематизации способов преобразования информации. На рис. 5.12 представлена обобщенная модель преобразования форм представления информации в виде ориентированного графа, отображающего взаимосвязь различных реализаций преобразования (ребра графа) через множество форм представления информации (вершины графа). Ребра графа ориентированы от входа к выходу и отображают выполнение части какого-либо алгоритма, связанной с операциями преобразования форм представления информации.

Цифры на ребрах, в основном, указывают на номера рисунков гл.5, где изображены соответствующие реализации, или, в некоторых случаях, эти реализации описаны в тексте со ссылкой на номер ребра графа рис. 5.12.

Центральная часть графа, в общем случае, должна представлять собой полный граф, в котором каждая вершина связана со всеми другими вершинами. Однако, не всегда может быть найдена та или иная реализация на определенном этапе развития технических средств.

В этом разделе мы ограничились только систематизацией реализаций по множеству Φ (форм представления информации). Такую же систематизацию необходимо провести по множествам Λ (способам кодирования) и Δ (точностям представления информации).

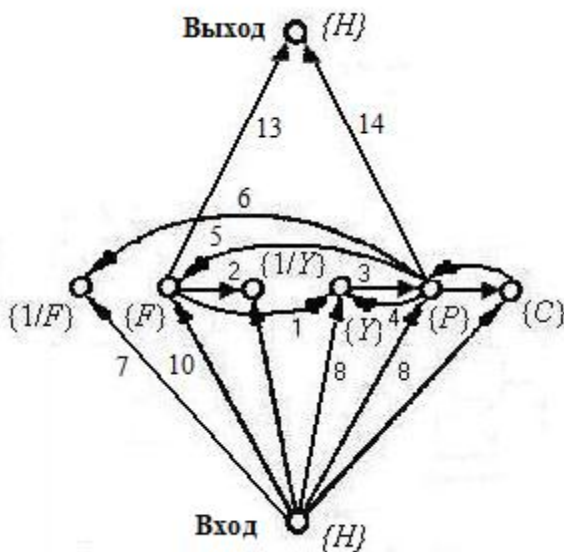


Рисунок 5.12 Обобщенная модель преобразования форм представления информации

5.1.3 Анализ и систематизация способов хранения информации

Хранение информации осуществляется на различное время, на различных носителях информации и с различным временем считывания и записи информации.

Остановимся на основных способах хранения информации с использованием электронных блоков памяти.

На рис. 5.13 представлена обобщенная модель способов хранения информации также в виде ориентированного графа, отобра-

жающего взаимосвязь различных реализаций блоков памяти (ребра графа) через множество форм представления информации (вершины графа).

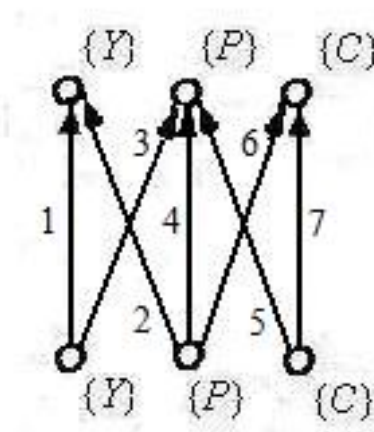


Рисунок 5.13 Обобщенная модель способов хранения информации

Ребра графа пронумерованы, их номера соответствуют следующим реализациям:

- 1 – схема рис.5.4 с дополнительными связями выхода счетчика со входом регистра;
- 2 – схема рис. 5.4;
- 3 – счетчик;
- 4 – регистр;
- 5 – сдвиговый регистр с выводом информации в $\{P\}$ -коде;
- 6 – сдвиговый регистр с вводом информации в $\{P\}$ -коде;
- 7 – сдвиговый регистр.

5.1.4 Анализ и систематизация способов реализации операций сложения (вычитания).

На рис. 5.14 представлена обобщенная модель способов сложения двух чисел. Основной особенностью выполнения операции сложения (вычитания) является необходимость представления слагаемых в одинаковой форме. При этом результат сложения может

быть получен в форме отличной от формы представления слагаемых.

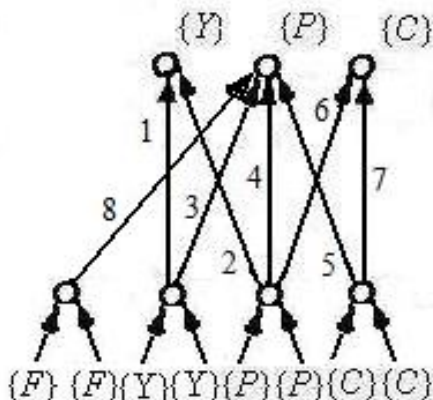


Рисунок 5.14 Обобщенная модель способов сложения двух чисел

На графе рис. 5.14 номера ребер соответствуют следующим реализациям:

- 1 – схема ИЛИ;
- 2 – в схеме рис.5.4 счетчик заменяется на накапливающий сумматор с дополнительными связями выхода сумматора со входом регистра;
- 3 – реверсивный счетчик;
- 4 – накапливающий сумматор;
- 5 – одноразрядный сумматор со сдвиговым регистром для фиксации полученного результата в $\{P\}$ -коде;
- 6 – накапливающий сумматор с преобразованием $\{P\}$ -кода в последовательный;
- 7 – одноразрядный сумматор;
- 8 – реверсивный счетчик с блоком временного разделения частотных импульсов двух слагаемых.

5.1.5 Анализ и систематизация способов выполнения операций умножения

Можно выделить пять основных способов выполнения операций умножения.

Первый способ основан на представлении сомножителей в виде частотного сигнала $N_1\{F\}$ и длительности импульса $N_2\{T\}$.

Результат умножения $N_3\{Y\}$, в этом случае может быть получен в унитарном коде за счет прохождения импульсов частотой $f(N_1)$ через схему совпадения за время $T(N_2)$.

Реализация этого способа для форм представления сомножителей в параллельном коде включает преобразователь числа $N_1\{P\}$ в частотный сигнал и преобразователь числа $N_2\{P\}$ в длительность импульса (см. рис. 5.15). Первый преобразователь состоит из счетчика COUVP1, регистра R2 и группы схем И между ними, второй – из счетчика COUVP2. Делитель Д с коэффициентом деления k_d определяет точность выполнения операции умножения.

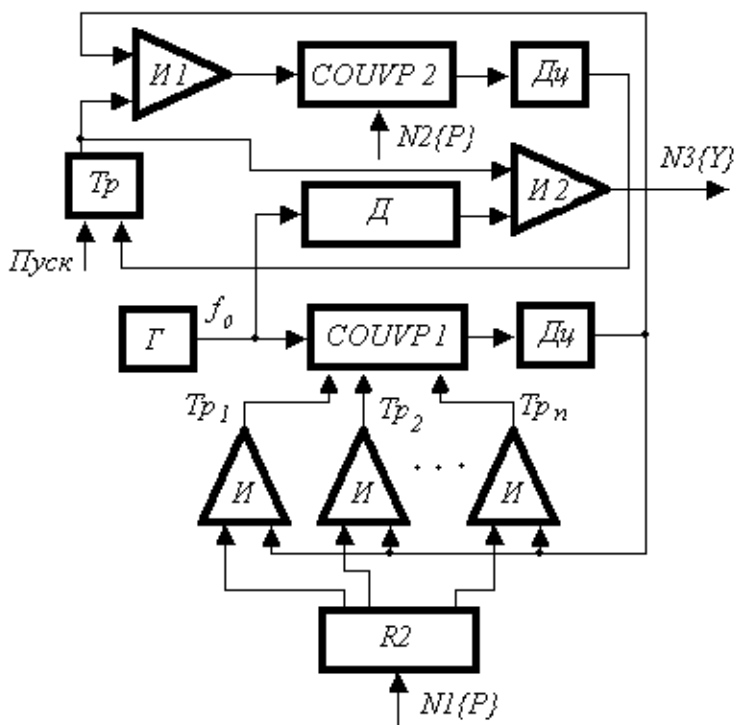


Рисунок 5.15 Первый способ выполнения операций умножения

В исходном состоянии в счетчиках записаны соответствующие сомножители в обратных кодах. По импульсу *Пуск*, синхронизированному с частотой f_0 генератора Γ (на рис. не показано), триггер Tr переключается в состояние «1», импульсы переполнения счетчика $COUVP1$ с частотой $f(N1)$, с одной стороны, каждый раз записывают в него значение $N1$, с другой - поступают через схему $И1$ на счетчик $COUVP2$. Импульс переполнения $COUVP2$ переводит Tr в состояние «0» и прохождение импульсов через схему $И1$ заканчивается. За это время через схему $И2$ проходит $N3\{Y\}$ импульсов, соответствующих результату умножения

$$N3\{Y\} = N2 \cdot \frac{N1 \cdot f_0}{f_0 \cdot k_d} = \frac{N2 \cdot N1}{k_d} \quad (5.16)$$

Коэффициент k_d через точности представления переменных определяется

$$k_d = \frac{100 \cdot \delta_3}{\delta_1 \cdot \delta_2} \quad (5.17)$$

Описанный способ умножения отличается сравнительно длительным временем выполнения операций. Этот недостаток может быть несущественным при повышении быстродействия логических элементов и при выполнении операций с ограниченной точностью.

Существенными преимуществами рассматриваемого способа являются наличие простых методов передачи информации и использование малой номенклатуры типовых узлов. Способ позволяет, кроме того, оперировать одновременно с тремя величинами за счет изменения в некоторых пределах коэффициента k_d , что в значительной мере сокращает количество оборудования блока управления.

Второй способ умножения характеризуется одновременным представлением сомножителей и произведения параллельным кодом. Процесс умножения осуществляется за счет логических схем [42, 55, 56]. Это наиболее быстродействующий способ умножения, но требующий значительной затраты оборудования.

В третьем способе умножения множитель представляется в унитарном коде, множимое – в параллельном коде. Реализация такого способа основывается на использовании сумматора накапливающего типа. Принцип работы схемы умножения заключается в дополнении значения множимого по каждому импульсу множителя.

Характерной особенностью такого способа является необходимость построения сумматора, разрядность которого определяется суммарной точностью представления сомножителей.

В четвёртом способе, в отличие от предыдущего, множитель представляется в последовательном коде. Реализация этого спосо-

ба умножения содержит накапливающий сумматор $SMNC$ со сдвигом в сторону старших разрядов, сдвиговый регистр CP , регистр $R2$, ключи K . Принцип работы схемы умножения заключается в добавлении каждый раз в сумматор $SMNC$ множимого в параллельном коде по значению «1» в младшем разряде CP , сдвига информации в $SMNC$ в сторону старших разрядов и сдвига в CP в сторону младших разрядов. Эта операция продолжается до последнего значимого разряда в CP .

В пятом способе применяется последовательная схема умножения. Для хранения и сдвигов чисел в устройствах, использующих этот способ, применяются три регистра со сдвигом. Первые два регистра имеют разрядность, определяемую точностью представления сомножителей. Третий регистр использует $2R$ разрядов. Суммирование частных произведений производится при помощи одноразрядного сумматора.

Время перемножения двух R -разрядных чисел в таком устройстве равно

$$T = [(R + 1) \cdot R + 1] \cdot t_1 \quad (5.18)$$

где t_1 – время одного такта.

На рис. 5.16 представлена обобщенная модель основных способов умножения двух чисел в виде ориентированного графа. Цифры на ребрах графа соответствуют последовательности рассмотрения способов умножения.

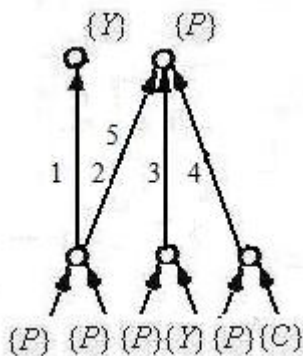


Рисунок 5.16 Обобщенная модель основных способов умножения двух чисел

Таким образом, на основе созданных обобщенных моделей реализации отдельных операций алгоритмов можно построить обобщенную модель реализации проектируемого устройства или системы. В качестве примера на рис. 4.3 представлена обобщенная модель реализации операций сложения двух чисел с преобразованиями форм представления информации на входе и выходе.

Далее рассмотрим примеры синтеза структур интерактивных технических средств (ИТС).

5.2 Проектирование структур системы измерения параметров технологических процессов (СИП)

Система предназначена для высокоточного измерения, нормализации и визуализации с точностью 0,1% в десятичном коде параметров технологических процессов, представленных на входе в виде $\{F\}$ -сигнала

$$N_i[nT_c] = \int_0^{T_c/2} N_i(t) dt$$

Рассмотрим наиболее характерные варианты построения структуры СИП.

5.2.1 Операции со структурами алгоритмов СИП

На рис. 5.17 показан первый вариант разбиения структуры общего алгоритма функционирования системы на локальные. В соответствие с правилом 1 (см. раздел 4.2) выделены четыре независимых канала измерения и визуализации частотных сигналов. Каждый канал разделен на три части: A_1^i , A_2^i , A_3^i , поскольку они отличаются способами кодирования. В A_1^i используется аналоговый частотный сигнал, в A_2^i - двоичный код, в A_3^i - десятичный код. Форма передачи промежуточной информации между локальными алгоритмами (также по правилу 1) выбрана в виде $\{Y\}$ -кода.

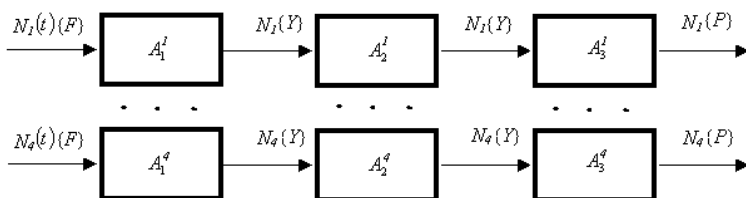


Рисунок 5.17 Первый вариант структуры СИП при начальном разделении алгоритма функционирования на локальные алгоритмы

Второй вариант структуры алгоритма системы показан на рис. 5.18. В этом варианте объединены ЛА A_2^i всех каналов. Способ кодирования в объединенном ЛА не изменяется, также остается в виде двоичного кода.

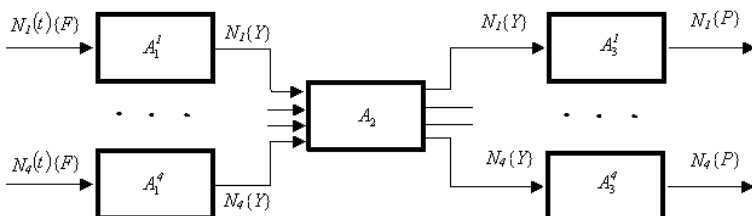


Рисунок 5.18 Второй вариант структуры СИП при последовательном объединении локальных алгоритмов

В третьем варианте объединены ЛА A_2^i , A_3^i по каждому каналу (см. рис. 5.19). По правилу 2 (раздел 4.1) в объединенных ЛА выбирается десятичный способ кодирования.

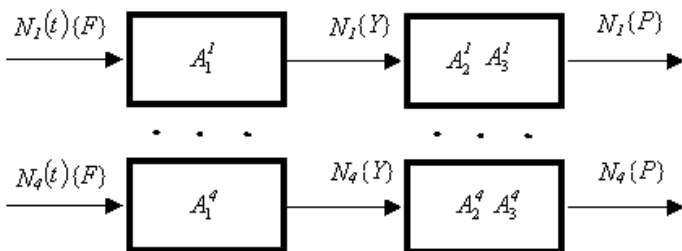


Рисунок 5.19 Третий вариант структуры СИП

На рис. 5.20 представлен четвертый вариант структуры алгоритма, в котором в одном ЛА объединены алгоритмы A_2^i , A_3^i всех каналов.

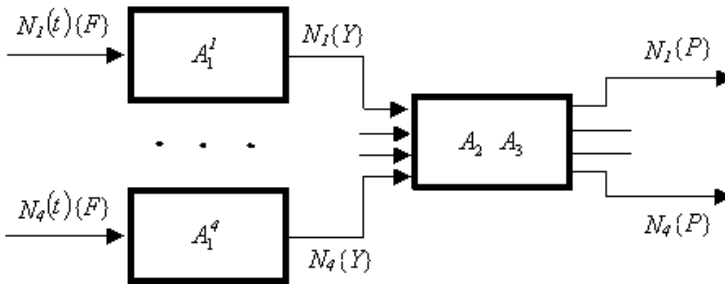


Рисунок 5.20 Четвертый вариант структуры СИП

Последний, пятый вариант (рис. 5.21) соответствует объединению всех ЛА A_1^i , A_2^i , A_3^i , по всем каналам. Следует заметить, что здесь рассмотрены только наиболее характерные варианты структур алгоритма СИП. Формально необходимо рассмотреть все сочетания ЛА, образованные при первоначальной операции декомпозиции.

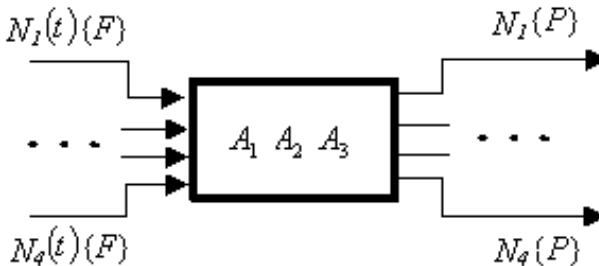


Рисунок 5.21 Пятый вариант структуры СИП

5.2.2 Анализ вариантов структурной реализации СИП

Первый вариант структурной реализации (рис. 5.17) по каждому каналу может содержать следующие блоки:

- преобразователь частотного сигнала $\{F\}$ в унитарный $\{Y\}$ код для реализации ЛА A_1 (рис.5.1, заметим, что в этой схеме генератор и счетчик-делитель, задающие цикл измерения частотного сигнала, могут использоваться одновременно для всех каналов);
- реверсивный счетчик (РС) импульсов $\{Y\}$ кода с преобразованием параллельного $\{P\}$ кода в унитарный $\{Y\}$ код, за счет вычитания накопленного числа в РС до «0», используется для реализации ЛА A_2 ;
- двоично-десятичный счетчик с выводом числа в $\{P\}$ коде для последующего преобразования числа в десятичный код с целью визуализации измеряемого параметра, используется для реализации ЛА A_3 ;

Второй вариант реализации отличается от первого последовательным измерением каждого из параметров в одном реверсивном счетчике и последовательной передачи измеряемых параметров по каналам в двоично-десятичные счетчики.

В третьем варианте объединяются последовательно локальные алгоритмы A_2 и A_3 и реализуются в одном двоично-десятичном счетчике.

В четвертом варианте происходит последовательное измерение частотных сигналов в одном двоично-десятичном счетчике с передачей информации в регистры $\{P\}$ кодом по каждому из каналов для хранения результатов измерения, преобразования в десятичный код и визуализации.

Пятый вариант отличается от предыдущего параллельным измерением частотного сигнала по всем каналам за короткие промежутки времени, для чего используется блок временного разделения импульсов от каждого датчика [44], один двоично-десятичный счетчик и четыре регистра. Каждый регистр связан со счетчиком $\{P\}$ кодом по входу и выходу через логические схемы таким образом, чтобы за короткие промежутки времени в каждом регистре накапливалась своя информация об измеряемом параметре. Накопленная информация в регистрах за общее время измерения преобразуется в десятичный код и визуализируется.

Последний принцип измерения частотных сигналов использовался в различных специализированных устройствах [15, 18].

5.3 Проектирование структуры системы смешения нефтепродуктов

Система предназначена для высокоточного регулирования заданных соотношений жидких компонентов в смеси. Количество компонентов – 14. Входная информация о каждом компоненте поступает в систему управления от турбинных расходомеров и датчиков температуры, начальные параметры процесса смешения устанавливаются оператором. Выходная информация управляет исполнительными механизмами и представляется оператору. Для каждого j -го компонента вычисляется регулирующее воздействие в соответствии с пропорционально – интегральным законом регулирования

$$\mu_j[nT_c] = K_1 \left(x[nT_c] + \frac{1}{K_2} \sum_{i=1}^n x[iT_c] \right),$$

где $x_j[nT_c] = \varphi_{3j} - \varphi_j[nT_c] = \alpha_j \Pi (1 + \beta_j \Delta \tau_j) - \int_0^{T_c/2} \varphi_j(t) dt$ – отклонение

текущего значения компонента $\varphi_j[nT_c]$ от заданного

φ_{3j} ;

α_j – процентное содержание компонента в смеси;

Π – производительность системы смешения;

β_j – коэффициент объемного расширения компонента;

$\Delta \tau_j$ – приращение температуры компонента относительно 20С;

T_c – время цикла работы системы;

n – количество циклов работы системы;

K_1, K_2 – параметры настройки, $\delta_{k1} = 2\%$.

На рис. 5.22 показаны основные варианты структур алгоритмов, образованные после проведения операций с алгоритмами. На рисунке в фигурных скобках указаны следующие формы внешнего представления различных параметров: $\{F\}$ – частотный сигнал, $\{P\}$ – параллельный код, $\{Y\}$ – число импульсный или унитарный код.

Первоначальное разделение алгоритма функционирования на локальные показано на рис. 5.22а. В этом варианте деление общего алгоритма произведено на ЛА по отдельным каналам (параллельным и несвязанным). Далее образованы локальные алгоритмы для каждого канала следующим образом:

$A_1 = \beta \Delta \tau$ – меньшая точность выполнения операции умножения по сравнению с точностью вычисления регулирующего воздействия;

$$A_2 = \varphi_{3j} = \alpha_j \Pi(1 + \beta_j \Delta \tau_j), \quad A_3 = \varphi_j[nT_c] = \int_0^{T_c/2} \varphi_j(t) dt \quad \text{– алгоритмы параллельные;}$$

$$A_4 = A_2 - A_3;$$

$A_5 = A_4 / K_2$ – возможно использование упрощенного способа выполнения операции деления, поскольку величина K_2 принимает только фиксированные значения 2, 4, 8, 16;

$$A_6 = A_4 + A_5;$$

$A_7 = K_1 A_6$ – меньшая точность выполнения операции умножения, поскольку коэффициент K_1 задается только фиксированными значениями от 1 до 5.

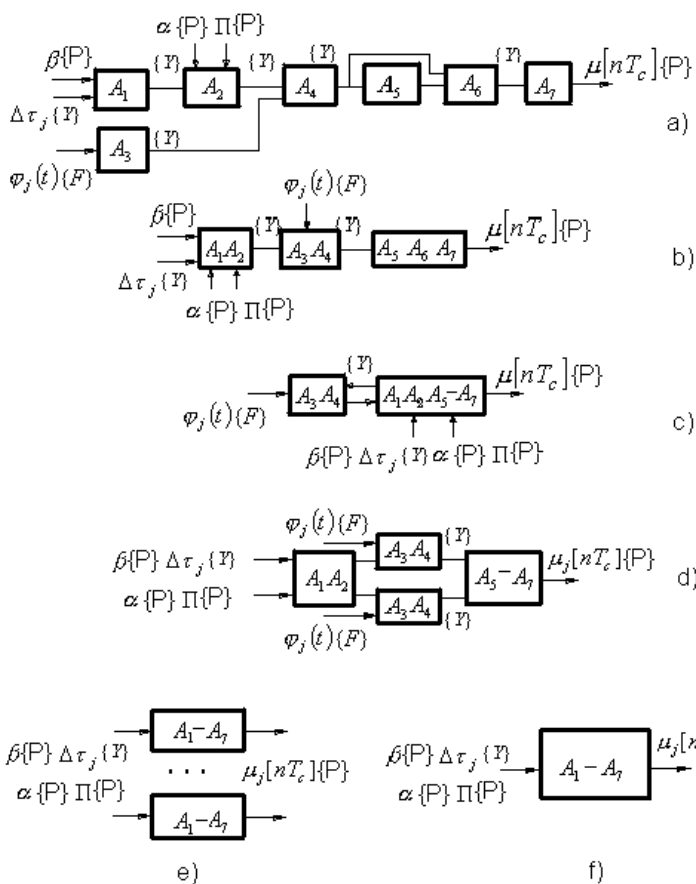


Рисунок 5.22 Пример операций со структурами алгоритмов

Далее проведем последовательное объединение локальных алгоритмов. Некоторые варианты соединения локальных алгоритмов, полученные после операций объединения, показаны на рис. 5.22(b, c, d). Эти варианты предполагают независимые вычисления регулирующих воздействий по каждому каналу. Дополнительные варианты могут быть получены при объединении локальных алгоритмов для всех каналов (см. рис. 5.22(e, f)).

Таким образом, на основе операций со структурами алгоритмов формируется множество сочетаний локальных алгоритмов, в

каждом из которых определены не только выполняемые операции, но и характеристики операндов, влияющие на качество реализаций алгоритмов. Каждый локальный алгоритм, в свою очередь, имеет множество реализаций, определяемое характеристиками операндов. На примере измерения параметров технологических процессов показана процедура перехода от структур алгоритмов к структурам реализаций.

Глава 6.

СИНТЕЗ СТРУКТУР ИНТЕРАКТИВНЫХ ПРОГРАММНЫХ СРЕДСТВ ПРОЕКТИРОВАНИЯ МАШИНОСТРОИТЕЛЬНЫХ КОНСТРУКЦИЙ

6.1 Классификация интерактивных программных средств

Ранее в разделе 1.2 была проведена классификация интерактивных систем по связности их процессорной части со специализированными внешними устройствами, которые за счет специфических структур данных существенным образом влияют на общую структуру ИС. Теперь проведем классификацию интерактивных программных средств (ИПС) с учетом их основных функциональных особенностей. ИПС условно можно разделить на следующие три класса: отдельные специализированные программы (СП), пакеты прикладных программ (ППП), универсальные системы программ (УСП).

В табл. 6.1 приведены примеры ИПС такого рода, указан класс ИПС, назначение, способы взаимодействия пользователя с ИПС, структуры данных генерируемых и обменных файлов.

СП по способу связи с оператором и внешними устройствами относятся к третьему классу ИС. В качестве примера в первой строке табл. 6.1 приведен интерпретатор языка GERBER для управления 2D устройствами (графопостроителями, фотоплоттерами, координатографами). В связи с широким распространением таких устройств, выпускаемых фирмой GERBER, язык фактически приобрел значение международного стандарта. Программы, написанные на этом языке, формируют исполняемый файл *.gbr.

ППП (вторая строка табл. 6.1) представляют собой набор программ, часто используемых в отдельных областях применения. Так, например, в области компьютерной графики известны пакеты графических программ такие, как PLOT-10, “Графор” [27], “ФАП-КФ” [37], последнее время VRML, Open GL [61], Direct X и др. ППП объединяет некоторая выделенная общая область памяти для передачи данных между программами, однако структуры данных не регламентируются, их определяет пользователь.

Таблица 6.1

Классификация интерактивных программных средств по функциональным свойствам

	Класс ИПС	Назначение	Способ взаимодействия	Структуры данных	Обменные файлы
GERBER	Программа (СП)	Управление 2D устройствами	Текстов. язык описания 2D графики	*.gbr	*.gbr
Open GL	Пакет графических подпрограмм (ППП)	Расширение графическими функциями универсальных языков программирования	Текстовое описание функций языка	Определяет пользователь	-
Графика 01-T	Универсальн. система программ (УСП)	2D графика, авто-трассировка, проект. РЭА	Языковые и интеракт. средства	*.bmp, *.b, *.i	*.dxf, *.bmp
AutoCAD	— ” —	2D и 3D графика, машиностр и др.	— ” —	*.dwg, *.dxf, *.bmp	*.dxf, *.bmp,
P-CAD	— ” —	2D графика, автотрассировка, проект. РЭА	— ” —	*.pcb, *.csh, *.plt, *.dxf, *.pdf, *.lib	*.dxf, *.pcb, *.csh, *.dxf, *.idf
Inventor	— ” —	2D и 3D графика,	— ” —	*.ipt, *.prt,	*.dxf, *.ipt,

	Класс ИПС	Назначение	Способ взаимодействия	Структуры данных	Обменные файлы
		проект. РЭА и др.		*.iam, *.idw	
Solid Works	— ” —	2D и 3D графика, проект. машиностр и др.	— ” —	*.sldprt, *.sldasm , *.slddrw *.iges, *.stp, *.wrl, *.stl, *.ipt,	*.dxf, *.dwg, *.iges, *.stp, *.wrl, *.stl, *.ipt

В УСП выделяется инвариантная по отношению к решаемым задачам часть, включающая в свой состав средства взаимодействия пользователя с системой и внешними устройствами, систему управления базами данных и проблемно-ориентированную часть. СП широко используют унифицированные и стандартные структуры данных типа *.gbr (GERBER), *.dxf (AutoCAD), *.iges (IDGES), *.stp (STEP), *.wrl (VRML), что позволяет производить обмен данными между системами. В последующих строках таблицы 6.1 перечислены некоторые примеры СП, указаны области применения и используемые структуры данных.

Из таблицы следует, что для решения конкретной задачи есть принципиально разные способы ее реализации. Формализация процесса выбора лучшей программной реализации может быть сведена к систематизации множеств вариантов реализации заданных алгоритмов по множеству возможных структур данных. Для этого множество реализаций представим в виде сети [8], в которой дуги соответствуют качественным показателям конкретной реализации, а вершины определяют входы и выходы (структуры данных) этих реализаций. Структуры данных будут являться связующим звеном множества реализаций в сети. Выбор лучшей реализации может быть сведен к решению задачи определения кратчайшего пути в сети такого рода.

6.2 Формирование обобщенных моделей ИПС

Покажем возможность систематизации множества реализаций на примере выбора лучшего варианта построения программного комплекса по виртуальной сборке автомобильного генератора. На рис. 6.1 представлена сеть, отображающая возможные варианты структурной организации тренажера по сборке генератора. Дуги сети соответствуют программным реализациям, а вершины – структурам данных.

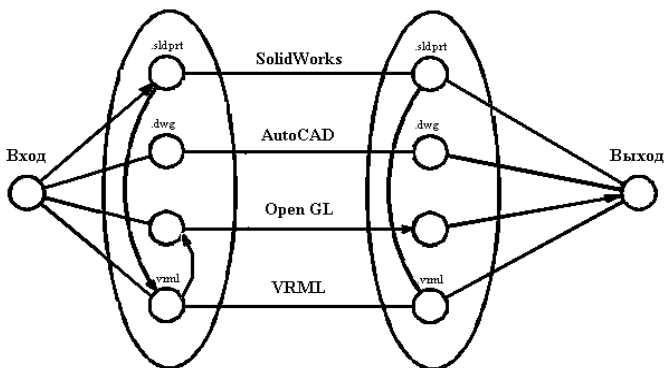


Рисунок 6.1 Обобщенная модель реализации тренажера по сборке генератора

Для упрощения задачи выбраны только три последовательные множества дуг сети от входа к выходу, соответствующие реализациям операций алгоритма функционирования тренажера.

Первое множество связано с преобразованиями структур данных на входе в структуры, указанные на вершинах первого овала сети, а также с операциями по созданию 3D моделей отдельных деталей генератора и формированию базы данных (БД) деталей. Овалами обозначена полная сеть преобразований структур данных, то есть каждая вершина внутри овала соединена с другими.

Второе – содержит произвольную последовательность операций по выбору одной модели из БД, пространственные ее перемещения, фиксацию, выбор модели второй детали, ее перемещения для соединения с первой и т.д. до полной сборки модели всего генератора. Для примера показаны только 4 типа ПО (4 типа дуг, со-

единающие вершины «Вход», «Выход»): VRML, Open GL, AutoCAD, SolidWorks.

Третье множество включает операции преобразования структур данных на выходе (второй овал сети).

Особенностями алгоритма сборки виртуального объекта являются необходимость задания геометрического описания его составляющих, их положение в пространстве и параметры сопряжения друг с другом. В геометрическое описание объекта входят множество вершин, граней и векторов нормали. Координаты вершин для каждого объекта описываются в однородных координатах в системе координат этого объекта.

Почти все операции могут быть реализованы на каждом из четырех типов ПО, за исключением операций преобразования в реальном времени координат моделей деталей. Создание базы данных моделей деталей лучше выполнять, например, на системах AutoCAD или SolidWorks, поскольку в такого рода системах предусмотрены специальные пользовательские средства. Однако разработчику это обойдется от \$2000 до \$20000 за одно рабочее место, при этом большая часть программных средств, в такого рода тренажерах, не будет использована. Базу данных деталей можно создавать непосредственно на VRML или на Open GL, в этом случае потребуются высоко квалифицированные программисты и большее время для создания моделей каждой детали.

Были разработаны программные средства [8], структура которых соответствует последовательности дуг, отмеченной на рис. 6.1 стрелками.

На рис. 6.2 (смотри цветную вкладку) показан процесс сборки генератора с использованием этих средств: рис. 6.2 а) – частично собранный генератор, 6.2 б) и в) – отдельные детали из базы данных. На рис. 6.2 б) – отмечается требуемая деталь перед сборкой и средствами перемещения курсора она устанавливается на нужное место.

Представленная на рис. 6.1 сеть фактически является обобщенной моделью [9] реализации тренажера, то есть для решения поставленной задачи потенциально можно использовать все ресурсы, указанные в сети.

Далее приведем примеры выбора структур 2D (СК) и 3D систем объемного конструирования (СОК), а также структур системы

автоматической трассировки соединений между элементами на плоскости.

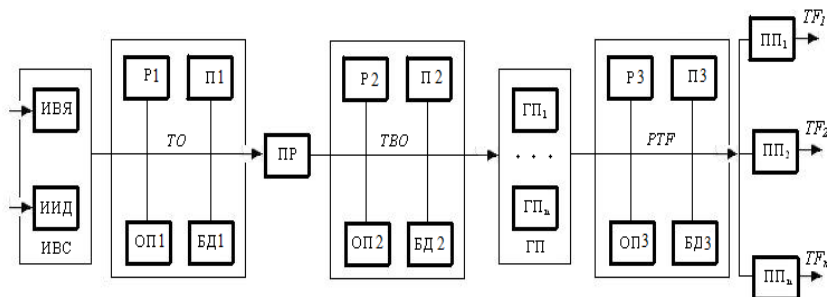


Рисунок 6.3 Первый вариант структуры СК

6.3 Проектирование структур 2D систем конструирования

Структура алгоритма функционирования систем конструирования (СК) показана на рис. 6.3. На рисунке выделены в виде отдельных блоков локальные алгоритмы, т.е. алгоритмы с сильно связными операциями, использующие один способ кодирования и единую точность представления информации. Выделены следующие локальные алгоритмы:

- ИВС – алгоритмы интерпретации входных сообщений, включающие интерпретацию входного графического языка (ИВЯ) и интерпретацию интерактивных действий пользователя (ИИД) [26].
- ПР – алгоритм преобразования таблицы описаний моделей и процессов проектирования в терминах входного языка (ТО) в таблицу входных описаний (ТВО).
- ГП – алгоритмы геометрических построений, осуществляющие вычисления координат характерных точек простых геометрических фигур, текста, алгоритмы штриховок, преобразования координат и т.п.
- ПП_i – алгоритмы преобразований информации для вывода на *i*-ое внешнее устройство.

Кроме того, по j -ым способам представления информации (структурам данных: TO , TBO , PTF) распределены алгоритмы виртуальной записи и чтения из оперативной памяти ($ОП_j$), управления базами данных ($БД_j$), редактирования (P_j) и преобразования ($Π_j$) для вывода отредактированной информации на внешние устройства, в частности, на дисплей. На рисунке не показаны алгоритмические средства управления последовательностью вычислений.

При первоначальном разделении на локальные алгоритмы выбраны следующие способы представления информации в системе:

1. Параметрическая таблица описания входной информации (TO). Информация поступает и хранится в символьном виде. В общем случае в TO представлены геометрические модели, действия по их преобразованию, а также описания процесса проектирования. В TO хранится протокол действий пользователя при работе с системой. Этот способ представления практически соответствует международным стандартам $NAPLS$ и $IGES$. Характеристики структуры данных TO представлены следующим образом

$$\begin{aligned} TO\{S, Tb\}[a_{TO}, \delta_{TO}](r) \Rightarrow Op^1; Op^2; \dots Op^n \Rightarrow \\ Id^1, Pa_1^1, Pa_2^1 \dots Pa_k^1; Id^2, Pa_1^2, Pa_2^2, \dots Pa_l^2; \\ Id^n, Pa_1^n, Pa_2^n \dots Pa_m^n, \end{aligned} \quad (6.1)$$

где Op^i – i -ый оператор в таблице описаний из общего числа n операторов;

Id^i – идентификатор i -го оператора;

Pa_j^i – j -ая группа параметров i -го оператора.

В записях структур данных здесь и далее по тексту в фигурных скобках будет указываться форма представления информации (например, $\{S, Tb\}$ – означает символьная таблица), в квадратных скобках – способ кодирования и точность представления информации $[a_{TO}, \delta_{TO}]$, знак (?) означает неопределенность в некоторых элементах структур данных.

2. Таблица входных описаний (*TBO*) служит для внутрисистемного представления информации, в которой идентификатор и параметры являются фиксированными действительными значениями, в остальном структура *TBO* аналогична структуре *TO*

$$TBO\{D, Tb\}[a_{TBO}, \delta_{TBO}](r).$$

3. Предтерминальный файл (*PTF*). В *PTF* представлены результаты работы системы в аппаратно-независимом виде. В *PTF* используются фиксированные значения параметров, подготовленные для вывода на внешние устройства.

$$\begin{aligned} PTF \{D, ?\}[a_{PTF}, \delta_{PTF}](?) \Rightarrow x_1^1, y_1^1, x_2^1, y_2^1, \dots, x_n^1, y_n^1, Sl^1, \\ x_1^2, y_1^2, x_2^2, y_2^2, \dots, x_m^2, y_m^2, Sl^2, \\ \dots \\ x_1^k, y_1^k, x_2^k, y_2^k, \dots, x_l^k, y_l^k, Sl^k, \end{aligned} \quad (6.2)$$

где x_i^j, y_i^j – координаты i -ой точки ломаной линии, порожденной j -ым оператором, Sl^j -служебная информация.

4. Терминальные файлы (*TF i*). В *TF i* представлены результаты в кодах i -го устройства, на которое производится вывод информации. Структура файлов в общем виде выглядит следующим образом

$$\begin{aligned} TF \{S, ?\}[a_{TF}, \delta_{TF}](?) \Rightarrow Op^1; Op^2; \dots Op^n \Rightarrow \\ Id^1, Pa_1^1, Pa_2^1 \dots Pa_k^1; Id^2, Pa_1^2, Pa_2^2, \dots Pa_l^2; \\ Id^n, Pa_1^n, Pa_2^n \dots Pa_m^n; \end{aligned} \quad (6.3)$$

Информация в *TF*, как правило, представляется в символьном виде. Достаточно часто используются стандартные способы кодирования в виде форматов GERBER или HPGL.

Способы представления, кодирования и точность информации на входах и выходах системы определяются внешними устройствами или системами. Представление, кодирование и точность промежуточной информации задается разработчиком и, затем, уточняются на последней стадии синтеза структуры системы.

Перечислим основные выводы по анализу первого варианта структуры системы.

1. Система имеет существенную избыточность, поскольку повторяются блоки редактирования, записи и чтения из оперативной памяти и базы данных. В системе может быть получено высокое быстродействие на отдельных операциях при работе с базами данных. Последнее оказывается возможным из-за наличия четырех способов представления информации. В зависимости от требуемой операции всегда можно подобрать способ представления данных, при котором не потребуются дополнительные преобразования.

2. Информация в ОП1 и БД1 хранится в символьном виде и поэтому она представлена менее компактно, чем в ОП2 и БД2. Блоки Р2 и П2 сложнее и операции в них выполняются медленнее, чем в Р1 и П1 соответственно, поскольку требуются дополнительные преобразования числовой информации в символьную и наоборот.

3. Блоки ОП3 и БД3 занимают больше памяти, чем ОП2 и БД2, потому что в *PTF* геометрические объекты развернуты до координат характерных точек. Вывод информации на внешние устройства из ОП3 и БД3 производится быстрее, чем из ОП2 и БД2, поскольку не расходуется время на работу геометрического процессора ГП.

4. Хранить информацию в одном *PTF* выгоднее, чем в нескольких *TF i*, с точки зрения минимизации занимаемой памяти. При этом существенно сокращается количество блоков для работы непосредственно с терминальными файлами. Однако время, расходуемое на работу преобразователей ПП_і увеличивается. В ряде случаев этим временем можно пренебречь, учитывая, что внешние устройства работают медленнее, чем процессор. В то же время, например, при создании компьютерных фильмов кадры фильма приходится хранить в виде последовательности дисплейных файлов *TF i*.

Последующие варианты построения системы образуются за счет проведения операций объединения ЛА в смысле их реализации в виде одного программного блока. Для простоты изложения не будем далее рассматривать весь процесс образования вариантов структур системы. Ограничимся рассмотрением только наиболее характерных структур СК.

На рис. 6.4 показан один из следующих вариантов построения системы. В этом варианте отсутствует промежуточное представление информации в виде *TBO* и, соответственно, блоки Р2, П2, ОП2, БД2. Появляется новый преобразователь ПР1. Кодирование информации на выходе этого преобразователя отличается от кодирования в *TBO*, оно может быть более простым, поскольку каждая команда должна сразу исполняться геометрическим процессором ГП.

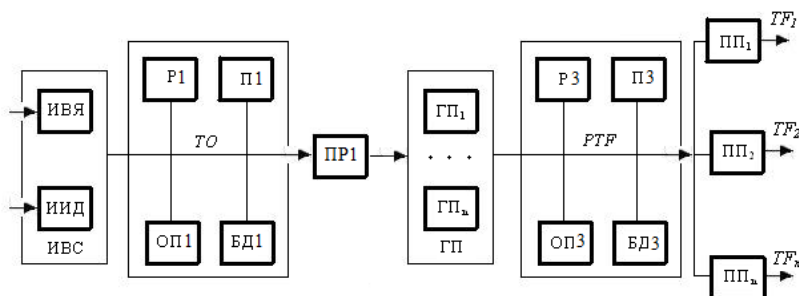


Рисунок 6.4 Второй вариант структуры СК

Второй вариант построения системы занимает меньше памяти, чем первый, однако имеет меньшее быстродействие на операциях с объектами, хранящимися в базе данных на входе геометрического процессора.

Третий вариант построения структуры системы показан на рис. 6.5.

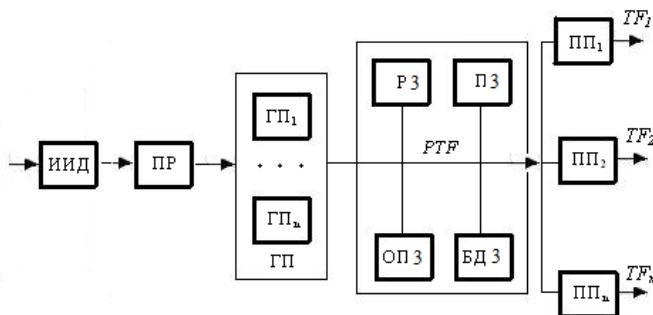


Рисунок 6.5 Третий вариант структуры СК

В нем на входе системы имеется только интерпретатор интерактивных действий пользователя ИД и преобразователь ПР, от которого информация по каждой команде непосредственно передается на геометрический процессор ГП.

На входе ГП отсутствуют встроенные в систему средства редактирования и работы с базами данных, а также средства языкового взаимодействия пользователя с системой. Их функции могут выполнять средства операционной системы или другие известные системы. Для этого, например, может быть использован текстовый редактор, компилятор с языка СИ, одна из известных систем управления базами данных и т.п. При этом снижаются расходы на разработку системы, сокращается время разработки. Однако, естественно, требуется больше памяти, теряется время на вызов другой системы, требуются специальные средства для вызова этих систем из основной системы.

Выше рассмотрены только три наиболее характерные варианты построения системы. При синтезе структуры системы образуется существенно большее количество вариантов, то есть все возможные сочетания локальных алгоритмов.

6.4 Примеры структур данных 2D систем конструирования

Структура языка GERBER

К классу терминальных файлов $TF\{S, Tb\}$ относятся файлы, созданные на языке GERBER для вывода информации на векторные графопостроители, координатографы и фотоплоттеры. Синтаксис языка рассмотрим на примере упрощенной версии языка GERBER 6000.

В соответствии со стандартом RS-274-C ассоциации EIA команды языка делятся на четыре группы:

- выбор версии языка GERBER G_n ($n=2$);
- задание координат X_n, Y_n ($n=6$);
- выбор номера пера для графопостроителя, типа инструмента для координатографа, диафрагм для фотоплоттера (далее по

тексту – инструмента) Dn (n=2); работа инструментом D01, D02, D03;

➤ управление Mn (n=2).

Каждая последовательность команд должна завершаться знаком “*”. Команды задания координат X и Y могут посылаются парами или отдельно. Начальные нули допускаются, но не являются обязательными. Заданные параметры действительны вплоть до ввода новых значений. Назначение стандартных команд показано в табл. 6.2. Другие команды игнорируются. Индикации ошибок нет.

Таблица 6.2

Перечень команд языка GERBER 6000

Команда	Назначение
Gn	n= 54 – версия GERBER 6000
R00	Относительное перемещение инструмента. По умолчанию – абсолютное
Xn	Относительное или абсолютное перемещение вдоль оси X n - целое число в диапазоне –999999 и +999999. Одна цифра равняется относительному перемещению на .001 мм. или .0001 дюйм, .01 мм. или .001 дюйм. При включении питания, по умолчанию - .001 мм
Yn	Относительное или абсолютное перемещение вдоль оси Y
D01	Работа инструментом
D02	Подъем инструмента
D03	Работа инструментом с ранее заданными параметрами
Dn	Целые числа от 10 до 29 и от 70 до 73 задают один тип инструмента. Эта команда не опускает инструмент. Работа указанным типом инструментов определяется добавлением команды D03
M00	Перемещение инструмента в положение 0,0 (начало координат)
M02	Конец программы
M64	Установка начала координат в настоящем положении

Команда	Назначение
	нии инструмента и продолжение работы
M65	Перемещение инструмента на +8 дюймов за наибольшие значения X и Y. Это положение считается новым началом координат и работа продолжается

Приведем пример описания (см. табл. 6.3) на языке GERBER 6000 элемента векторного изображения (рис. 6.6).

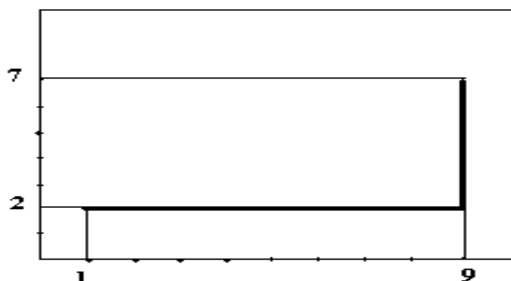


Рисунок 6.6 Пример векторного изображения

Таблица 6.3

Описание на языке GERBER 6000 элемента векторного изображения

Программа	Комментарий
G54*	Выбор версии языка GERBER 6000.
R00*	Выбор относительных координат.
D14*	Выбор номера инструмента (перо №2 или диафрагма №5).
X1000Y2000 D02*	Перемещение в относительных координатах на 8мм вдоль оси X с поднятым инструментом.
X8000D01*	Перемещение инструмента на 8мм вдоль оси X с опущенным инструментом. Рисуются линия пером №2 или экспонируется диафрагмой №5.
Y5000D01*	Перемещение на 5мм вдоль оси Y с опущен-

Программа	Комментарий
	ным инструментом.
D20*	Выбор другого номера инструмента (типа пера, номера диафрагмы).
D03*	Работа ранее выбранным инструментом.
M00*	Поднятие инструмента и перемещение в положение 0,0.

Все записи в терминах языка представляют собой символы, включая и числовые значения координат. Обычно в системах автоматизированного проектирования на выходе используются специальные постпроцессоры, автоматически преобразующие символьную информацию языка в управляющие сигналы внешнего устройства. В ряде случаев пользователь может самостоятельно описать в терминах языка элементы изображения. Файлы могут создаваться в любом текстовом редакторе. После написания программы на языке GERBER 6000 файлу присваивается расширение (.gbr). Постпроцессор читает такой файл и выводит соответствующую информацию на внешнее устройство.

6.5 Проектирование структур 3D систем конструирования

Разработка 3D-систем конструирования (СОК) является актуальной задачей. С решением этой задачи тесно связано создание и развитие средств автоматизации проектирования в таких важных отраслях, как машиностроение и приборостроение. Системы объемного конструирования позволяют решать по-новому задачи компоновки пространственных объектов, моделирования процессов сборки, моделирования процессов обработки на станках с ЧПУ, подготовки информации для расчетов методом конечных элементов. Решение этих задач существенно сокращает время проектирования (иногда в десятки раз), повышает качество конечного продукта, исключает необходимость физического моделирования.

Алгоритм функционирования СОК включает следующие основные операции (см. рис. 6.7).

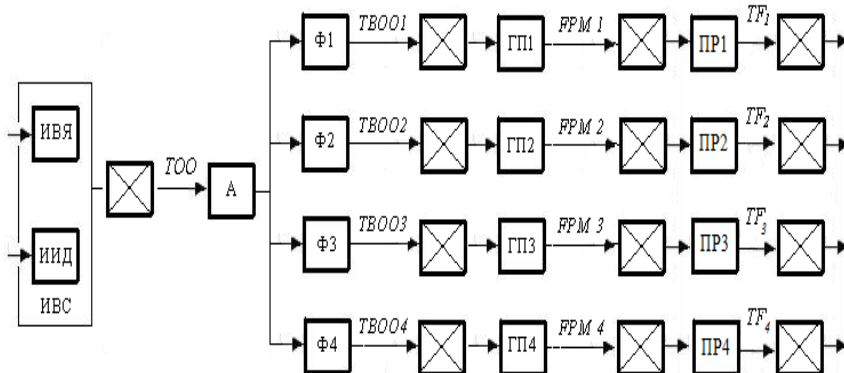


Рисунок 6.7 Первый вариант структуры СОК

1. Ввод описаний графических образов и их преобразований (блок ИВС). Информация вводится с клавиатуры алфавитно-цифрового дисплея или указыванием на элементы меню, размещенные на экране графического дисплея или мини-планшете. Выходная информация представляется в виде таблицы описаний объемных геометрических моделей $TOO\{S, Tb\}[a_{TOO}, \delta_{TOO}](?)$, структура таблицы описаний аналогична (6.1) [26].

2. Редактирование символьного описания входной информации (Р). Запись и чтение символьного описания из базы данных (БД). Информация на входе и выходе этих блоков представляется также аналогично (6.1). В дальнейшем перечисленные блоки будут указываться на рисунке всякий раз, когда образуется новая структура данных. Для простоты изображения подключение этих блоков будем обозначать в виде квадрата с диагоналями.

3. Анализ (А) и преобразование таблицы TOO в таблицу входных описаний $TBOO$ (блоки $\Phi 1$ - $\Phi 4$). Выделены следующие типы геометрических моделей: поверхностная (сеточная), проволочная (каркасная) многогранная, 2,5 D модель, 3 D твердотельная модель.

Возможны два типа формирования **поверхностных моделей**: ортогональное разбиение поверхности и разбиение поверхности на конечные элементы (КЭ) или заплаты.

Первый тип опишем следующим образом

$$\begin{aligned}
 TBOO\{D, Tb\}[a_{TBOO}, \delta_{TBOO}](R) \Rightarrow \\
 (x_1^1, y_1^1, z_1^1, x_1^1, y_2^1, z_2^1 \dots x_1^1, y_n^1, z_n^1, Sl^1, \\
 x_2^2, y_1^2, z_1^2, x_2^2, y_2^2, z_2^2 \dots x_2^2, y_m^2, z_m^2, Sl^2, \\
 \dots \dots \dots \\
 x_p^k, y_1^k, z_1^k, x_p^k, y_2^k, z_2^k \dots x_p^k, y_l^k, z_l^k, Sl^k, \\
 x_1^1, y_1^1, z_1^1, x_2^1, y_1^1, z_2^1 \dots x_n^1, y_1^1, z_n^1, Sl^{k+1}, \\
 x_1^2, y_2^2, z_1^2, x_2^2, y_2^2, z_2^2 \dots x_m^2, y_2^2, z_m^2, Sl^{k+2}, \\
 \dots \dots \dots \\
 x_1^k, y_p^k, z_1^k, x_2^k, y_p^k, z_2^k \dots x_l^k, y_p^k, z_l^k, Sl^{2k},
 \end{aligned} \tag{6.4}$$

где x_i^j, y_i^j, z_i^j – координаты точек пространственной ломаной линии, образованной пересечением поверхности ортогональными плоскостями по осям x и y ;

Sl^j – записи служебной информации; (например, цвет линий);

p – число точек в каждой ломаной линии;

k – число ломаных по каждой из осей координат.

Пример первого типа формирования поверхности показан на рис. 6.8.

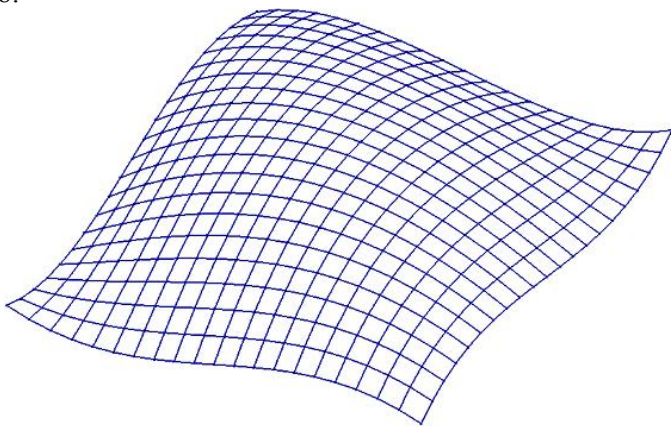


Рисунок 6.8 Первый тип формирования поверхностной модели

Второй тип формирования поверхности можно представить как

$$TBOO\{D, Tb\}[a_{TBOO}, \delta_{TBOO}] \Rightarrow x_1^1, y_1^1, z_1^1, x_2^1, y_2^1, z_2^1 \dots x_r^1, y_r^1, z_r^1, \\ x_1^2, y_1^2, z_1^2, x_2^2, y_2^2, z_2^2 \dots x_r^2, y_r^2, z_r^2, \\ \dots \dots \dots x_1^k, y_1^k, z_1^k, x_2^k, y_2^k, z_2^k \dots x_r^k, y_r^k, z_r^k, \quad (6.5)$$

где x_i^j, y_i^j, z_i^j – координаты точек пространственной замкнутой ломаной линии;

k – число ломаных в объемной модели;

r – общее количество элементарных плоских участков (заплат) модели поверхности (например, треугольных);

Sl^j – записи служебной информации (например, цвет заплата, нормаль к заплате).

Пример второго типа формирования поверхности показан на рис. 6.9.

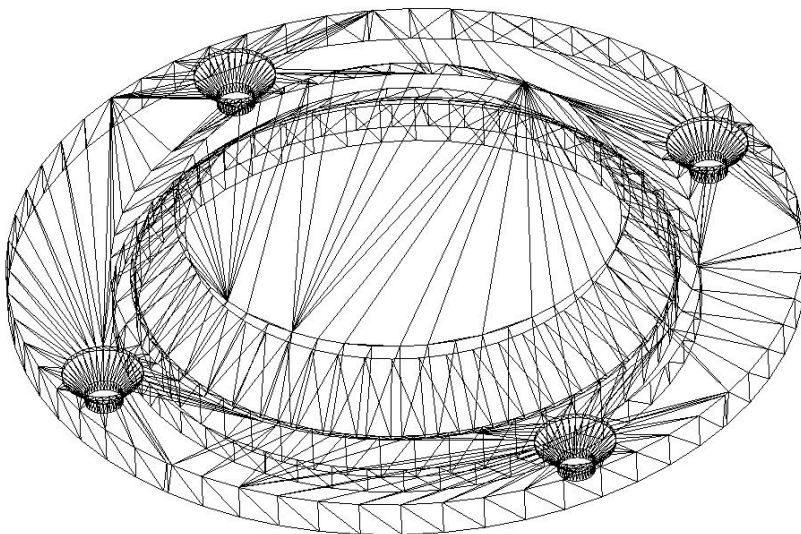


Рисунок 6.9 Второй тип формирования поверхностной модели

Проволочная (каркасная) многогранная модель может быть описана следующим образом

$$\begin{aligned}
 TBOO\{D, Tb\}[a_{TBOO}, \delta_{TBOO}] \Rightarrow \\
 x_1^1, y_1^1, z_1^1, x_2^1, y_2^1, z_2^1 \dots x_n^1, y_n^1, z_n^1, Sl^1, \\
 x_1^2, y_1^2, z_1^2, x_2^2, y_2^2, z_2^2 \dots x_m^2, y_m^2, z_m^2, Sl^2, \\
 \dots\dots\dots \\
 x_1^k, y_1^k, z_1^k, x_2^k, y_2^k, z_2^k \dots x_l^k, y_l^k, z_l^k, Sl^k,
 \end{aligned} \tag{6.6}$$

где x_i^j, y_i^j, z_i^j – координаты точек пространственной ломаной линии, описывающей контур грани модели;
 n, m, l – число вершин каждой грани;
 k – число граней в объемной модели;
 Sl^j – записи служебной информации (нормаль к каждой грани).

Примеры проволочных моделей показаны на рис. 6.10.

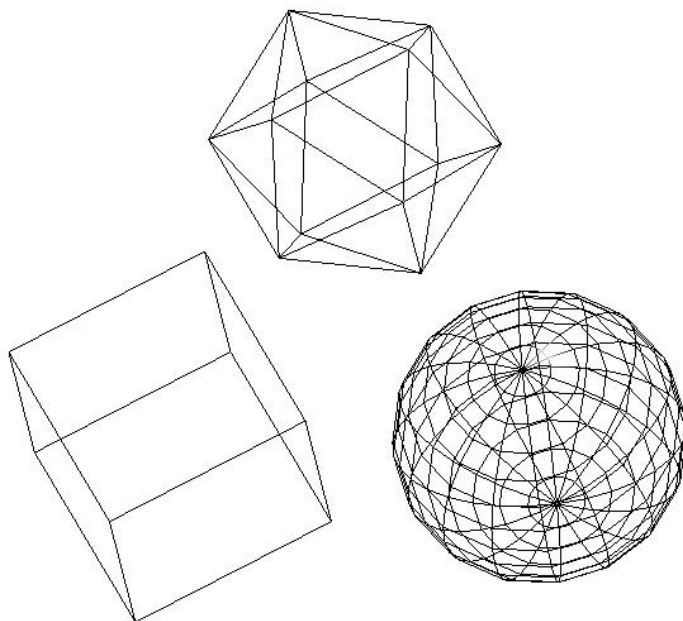


Рисунок 6.10 Примеры проволочных моделей

2,5 D модель имеет следующий вид (рис. 6.11 смотри цветную вкладку)

$$TBOO\{D, Tb\}[a_{TBOO}, \delta_{TBOO}] \Rightarrow x_1, y_1, x_2, y_2, \dots, x_l, y_l, M_z, \quad (6.7)$$

где x_i, y_i – координаты точек ломаной линии, описывающей плоское сечение тела;

l – число точек в ломаной линии;

M_z – матрица трансформации плоского сечения по оси z .

3D твердотельная модель может быть представлена следующим образом

$$TBOO\{D, Tb\}[a_{TBOO}, \delta_{TBOO}] \Rightarrow Tl^1; Tl^2; \dots Tl^n, \quad (6.8)$$

где Tl^i – описания n простых объемных геометрических тел.

На рис. 6.12 (смотри цветную вкладку) показана 3D твердотельная модель, состоящая из нескольких цилиндров. Процесс создания этой модели состоит из логических операций сложения, вычитания, отсечения и т.п. (рис. 6.13 смотри цветную вкладку).

4. Геометрические преобразования моделей (ГП1-ГП4). Информация на входе представлена в виде (6.3 – 6.7), на выходе образуется файл проволоочной модели ($FPM\{D, ?\}[a_{FPM}, \delta_{FPM}]$).

5. Преобразование FPM в 2D PTF (6.2) - блоки Пр1-Пр4. Редактирование PTF , запись и чтение из базы данных.

После проведения операций с алгоритмом функционирования СОК, выделено несколько наиболее характерных вариантов построения.

Структура первого варианта СОК будет соответствовать совокупности блоков, реализующих алгоритмы, представленные на рис. 6.6. Последующие варианты образуются за счет проведения операций последовательного объединения этих локальных алгоритмов. Смысл объединения ЛА такой же, как и в случае с СК, заключается в сокращении общего числа блоков, реализующих одинаковые алгоритмы, определении лучших структур данных.

Анализируя структуру, представленную на рис.6.6, можно заметить, что такая структура избыточна, поскольку имеется целый ряд блоков, реализующих одинаковые алгоритмы. К таким блокам

можно отнести блоки ГП1-ГП4, ПР1-ПР4, а также блоки, реализующие алгоритмы редактирования, записи и чтения из базы данных. Кроме того, структуры данных представляют информацию об объекте конструирования с большой избыточностью. Отсутствие связей между различными типами моделей приводит к усложнению целого ряда операций в СОК.

Совершенствование структуры данных может быть проведено за счет объединения локальных алгоритмов с разными способами представления моделей. Лучший вариант показан на рис. 6.14.

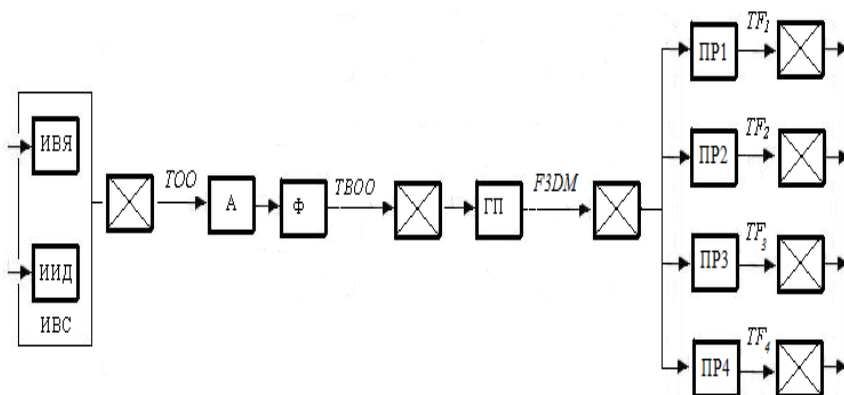


Рисунок 6.14 Второй вариант структуры СОК

Он содержит 3D геометрический процессор ГП, объединяющий функции блоков Ф1-Ф4, ГП1-ГП4. Появляется новая структура данных объединенной 3D-модели

$$\begin{aligned}
 & F3DM \{D, ?\} [a_{F3DM}, \delta_{F3DM}] \Rightarrow \\
 & x_1^1, y_1^1, z_1^1, x_2^1, y_2^1, z_2^1, \dots, x_k^1, y_k^1, z_k^1, Sl^1; \\
 & Gr_1^1, Gr_2^1, \dots, Gr_k^1; Tl^1; \\
 & x_1^2, y_1^2, z_1^2, x_2^2, y_2^2, z_2^2, \dots, x_l^2, y_l^2, z_l^2, Sl^2; Gr_1^2, Gr_2^2, \dots, Gr_l^2; Tl^2; \\
 & \dots \\
 & x_1^n, y_1^n, z_1^n, x_2^n, y_2^n, z_2^n, \dots, x_m^n, y_m^n, z_m^n, Sl^n, \dots, Gr_1^n, Gr_2^n, \dots, Gr_m^n; Tl^n; \quad (6.8)
 \end{aligned}$$

где x_i^j, y_i^j, z_i^j – координаты вершин граней каждого тела;

Gr_i^j – список номеров вершин, входящих в i -ую грань j -го тела;

Tl^j – список номеров граней j -ого тела объемной модели;

Sl^j – записи служебной информации.

Таким образом, были проведены исследования по систематизации различных алгоритмов 2D и 3D систем объемного конструирования, исследованы структуры данных, разработаны преобразователи отдельных структур данных, созданы программные реализации алгоритмов: системы «Графика-81-2D» и «Графика-81-3D».

На основе системы «Графика-81-2D» [1] в последующем разработана система автоматической трассировки соединений между элементами на плоскости «Графика-01-T» (см. далее гл.7).

Система «Графика-81-3D» использовалась для создания объемных геометрических моделей внешнего облика всех модулей орбитальной космической станции «МИР» [14], для моделирования конструкции фермы к станции «МИР» (проект «Ферма-3») [12], для моделирования внешних обликов космических аппаратов «Вега-1» и «Ураган», а также и в других приложениях [67, 68] (см. далее гл.8).

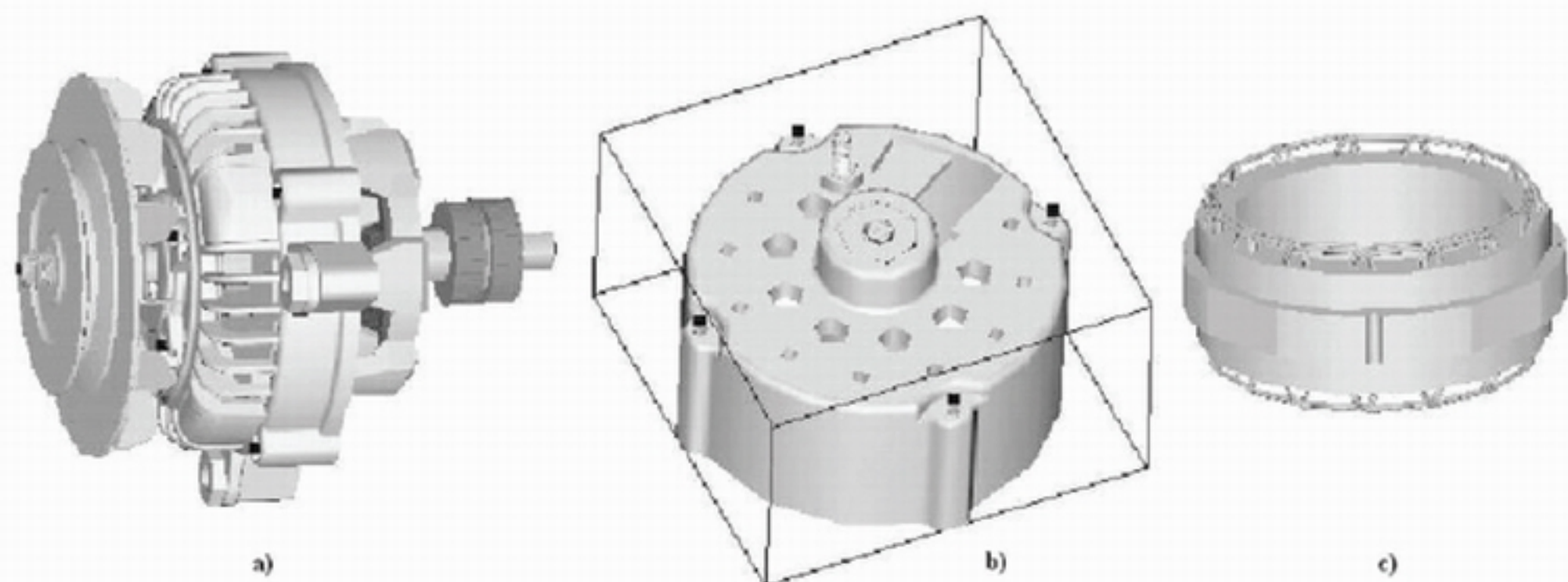


Рисунок 6.2 Процесс виртуальной сборки генератора

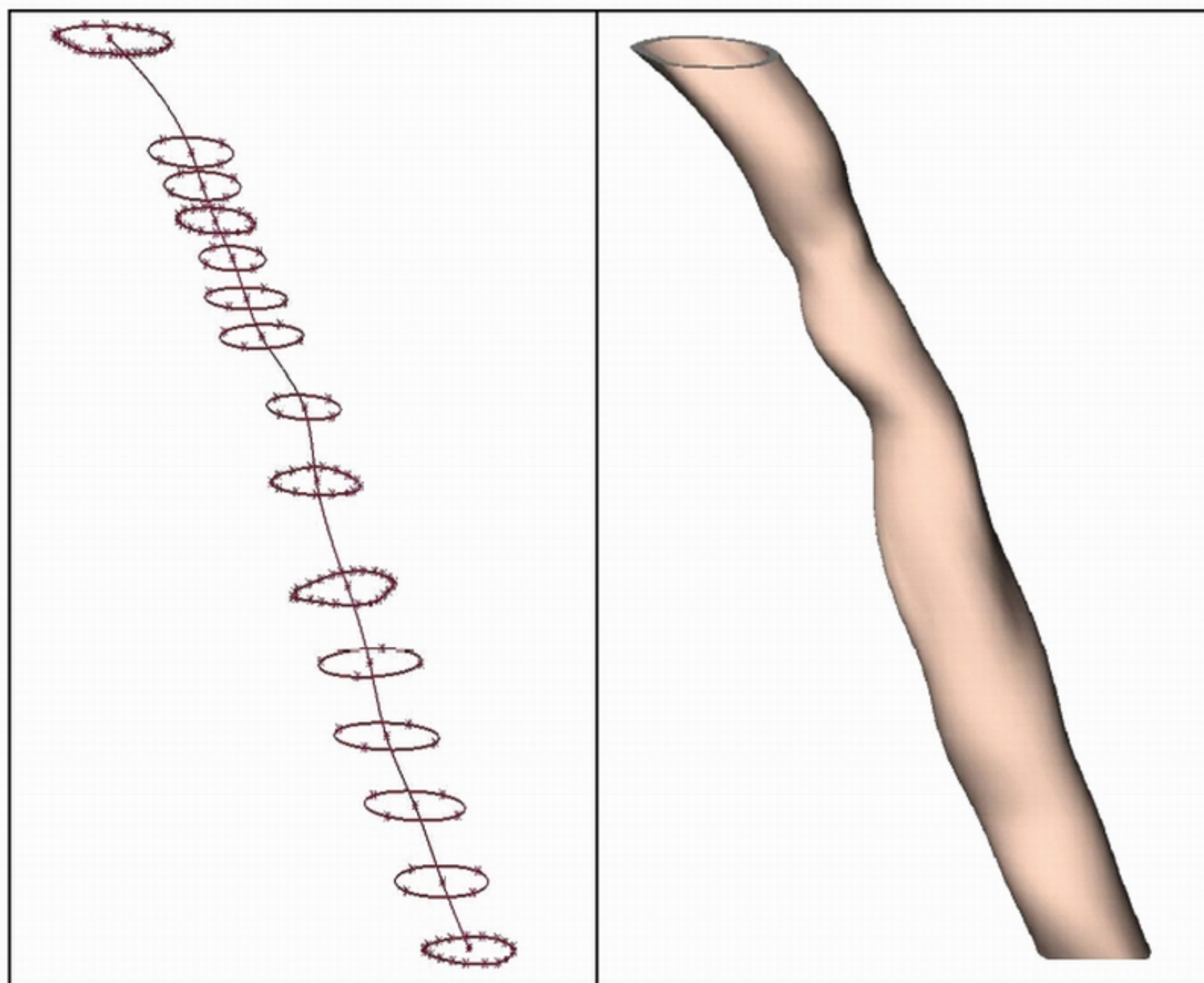


Рисунок 6.11 Пример 2,5 D модели

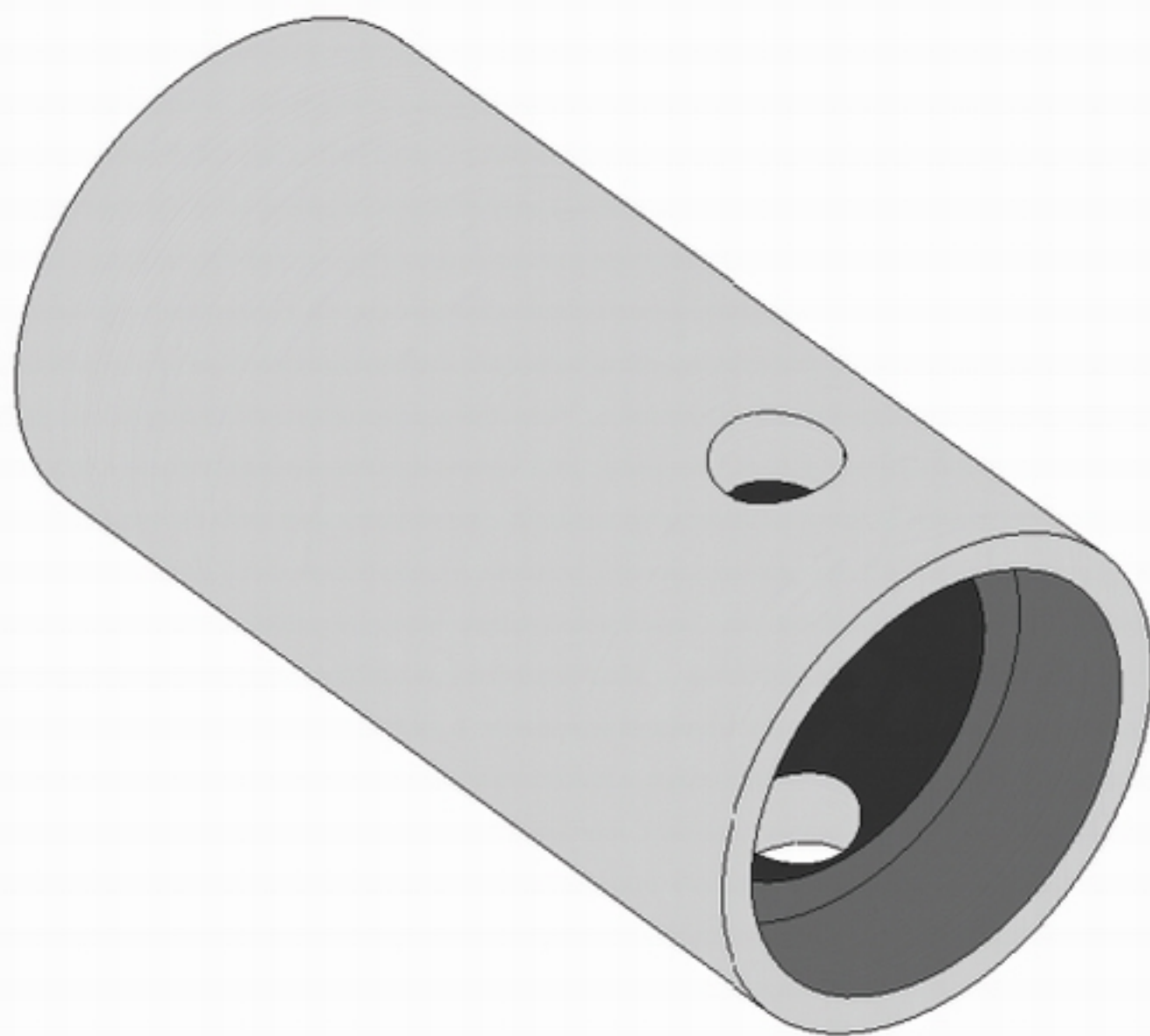


Рисунок 6.12 Пример 3D твердотельной модели



Рисунок 6.13 Процесс формирования 3D твердотельной модели

Глава 7.

СИНТЕЗ СТРУКТУР ИНТЕРАКТИВНЫХ ПРОГРАММНЫХ СРЕДСТВ ПРОЕКТИРОВАНИЯ РАДИОЭЛЕКТРОННЫХ УСТРОЙСТВ

В этом разделе ограничимся рассмотрением принципов построения ИПС для проектирования схемной документации на устройства РЭА, в частности, на системах 2D проектирования с автоматической трассировкой соединений между элементами на плоскости (СТП). СТП находят широкое применение при разработке принципиальных схем, топологии печатных плат, топологии интегральных схем. СТП могут быть использованы при прокладывании трасс соединительных кабелей, проводки и т.п. В настоящее время известно большое число алгоритмов автоматической трассировки соединений, известны их различные реализации и эксплуатационные характеристики. Однако отсутствуют теоретические работы по организации интерактивного режима трассировки, выбору и обоснованию структуры программного обеспечения, отсутствуют примеры аппаратной реализации, что приводит к наличию обилия реализаций трудно стыкуемых между собой, невзаимозаменяемых. Из-за отсутствия теоретических работ по выбору структур ПО сложилась ситуация, когда разработчики СТП по-разному понимают состав системы, ее место в интегрированной системе проектирования, например, в системе проектирования РЭА.

В соответствии с этим в настоящей работе при проектировании СТП основное внимание уделяется исследованию возможных вариантов построения, синтезу вариантов, оптимальных по заранее выбранным критериям, разработке и исследованию универсальных алгоритмов трассировки.

7.1 Синтез вариантов построения СТП

7.1.1 Основные операции алгоритма функционирования СТП

Перечислим основные операции алгоритма функционирования СТП.

1. Ввод описания геометрической модели трассируемого поля (ОМ) (рис. 7.1).

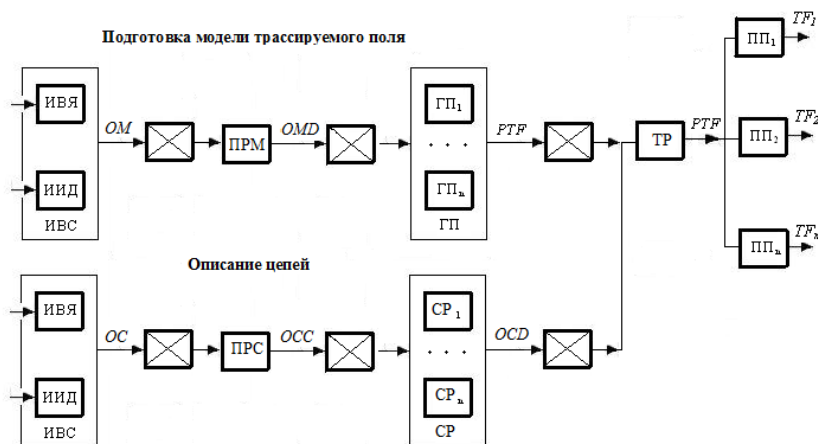


Рисунок 7.1 Первый вариант построения СТП

В СТП используются такие же средства ввода (блок ИВС), как и в СК. Передача информации в систему трассировки о трассируемом поле может представляться несколькими способами.

При **первом** способе информация поступает и хранится в виде символьной таблицы. В общем случае в ОМ может быть представлена геометрическая модель и действия по ее преобразованию. В частном случае, в ОМ хранится протокол действий пользователя при работе с системой. Характеристики структуры данных $OM\{S, Tb\}[a_{OM}, \delta_{OM}](r_{OM})$ записываются аналогично (рис. 7.1).

$$\begin{aligned}
 OMD\{D, Tb\}[a_{OM}, \delta_{OM}](r_{OM}) &\Rightarrow Op^1; Op^2; \dots Op^n \Rightarrow \\
 &\Rightarrow Id^1, Pa_1^1, Pa_2^1 \dots Pa_k^1; Id^2, Pa_1^2, Pa_2^2, \dots Pa_l^2; \\
 &\dots \dots \dots \\
 &Id^n, Pa_1^n, Pa_2^n \dots Pa_m^n;
 \end{aligned} \tag{7.1}$$

где Id^i – идентификатор i -го оператора;

Pa_j^i – j -ая группа параметров i -го оператора.

Параметры представлены в виде действительных (D) чисел, чтобы обеспечить требуемую точность трассировки соединений. На рис. 7.1 показан преобразователь (блок ПРМ) структуры данных описания модели OM в OMD .

В **третьем** случае информация представляется в аппаратно-независимом виде с фиксированными значениями параметров, подготовленными для вывода на внешние устройства

$$\begin{aligned}
 OM\{D, Tb\}[a_{OM}, \delta_{OM}](r_{OM}) \Rightarrow \\
 x_1^1, y_1^1, x_2^1, y_2^1, \dots, x_n^1, y_n^1, 4 * Sl^1, \\
 x_1^2, y_1^2, x_2^2, y_2^2, \dots, x_m^2, y_m^2, 4 * Sl^2, \\
 \dots \\
 x_1^k, y_1^k, x_2^k, y_2^k, \dots, x_l^k, y_l^k, 4 * Sl^k,
 \end{aligned} \tag{7.2}$$

где x_i^j, y_i^j – координаты i -ой вершины j -ой ломаной линии;

Sl^j – записи служебной информации.

При **четвертом** способе ввода используется символьное аппаратно-независимое представление информации с меньшей разрядностью параметра (r_{OM}). Примером могут служить языки типа *GERBER*, *HPGL* и т.п.

2. Редактирование описания модели трассируемого поля, запись и чтение из базы данных. На рис. 7.1 эти блоки обозначены квадратами с двумя диагоналями.

3. Преобразование описания модели в модель трассируемого поля. Эта операция выполняется геометрическим процессором ГП, реализующим переход от структуры данных OMD (7.1) к структуре PTF (6.2).

4. Ввод описания цепей (OC), соединяющих между собой контакты элементов схемы (второй блок ИВС). Информация на входе представляется в виде символьной таблицы

$$OC\{S, Tb\}[a_{OC}, \delta_{OC}](r_{OC}) \Rightarrow$$

$$Nm_i, Nk_j, Nm_k, Nk_l \dots Nm_m, Nk_n, P_c, \\ \dots \\ Nm_{ii}, Nk_{jj}, Nm_{kk}, Nk_{ll} \dots Nm_{mm}, Nk_{nn}, P_c, P_o, \quad (7.3)$$

где Nm_i, Nk_j – номера модулей и контактов цепи;

P_c – признак конца описания текущей цепи;

P_o – признак конца описания всех цепей.

5. Редактирование описания цепей в формате OC , запись и чтение из базы данных.

6. Преобразование списка цепей (блок ПРС) в целочисленную (C) таблицу. Вход – (7.3), выход

$$OCC\{C, Tb\}[a_{OC}, \delta_{OC}](r_{OC}) \Rightarrow \\ Nm_i, Nk_j, Nm_k, Nk_l \dots Nm_m, Nk_n, P_c, \\ \dots \\ Nm_{ii}, Nk_{jj}, Nm_{kk}, Nk_{ll} \dots Nm_{mm}, Nk_{nn}, P_c, P_o, \quad (7.4)$$

7. Размещение элементов на плоскости.

Предполагается, что размещение элементов может быть ручное и автоматическое (рис. 7.1 – блок средств размещения СР). При ручном размещении элементы на плоскости устанавливаются в интерактивном режиме, возможно с представлением пользователю функции качества размещения. При автоматическом размещении реализуется алгоритм минимизации общей длины связей между элементами. Вход – структура данных OC (7.3), выход – таблица действительных чисел координат последовательно соединяемых контактов

$$OCD\{D, Tb\}[a_{OC}, \delta_{OC}](r_{OC}) \Rightarrow \\ x_{ij}, y_{ij}, x_{kl}, y_{kl} \dots x_{mn}, y_{mn}, P_c, \\ x_{iijj}, y_{iijj}, x_{kkll}, y_{kkll} \dots x_{mmnn}, y_{mmnn}, P_c, P_o, \quad (7.5)$$

где x_η, y_η – координаты соединяемых контактов.

8. Сортировка списка цепей по возрастанию или убыванию длины отрезков цепи. Операция реализуется блоком СР, Входной и выходной информацией является файл *OCD* (7.5).

9. Запись и чтение списка цепей из базы данных.

10. Автоматическая трассировка соединений на плоскости. Общая задача трассировки формулируется следующим образом: по списку цепей *OCD*, содержащему n координат точек начала и конца трасс, определить дополнительные m координат промежуточных точек перегиба трасс таким образом, чтобы длина каждой трассы была минимальна и ни одна точка трассы не пересекала ранее проведенные трассы или элементы модели трассируемого поля.

Известны два принципиально различных варианта реализации алгоритма автоматической трассировки: геометрический и растровый (матричный).

В основу геометрического алгоритма положено аналитическое описание конфигурации модели трассируемого поля. При этом, как показано на рис. 7.1, входом являются структура модели в виде файла *PTF* (6.2) и список описания цепей в форме *OCD* (7.5). Результат работы геометрического алгоритма представляется в форме *PTF* (6.2).

В основе растрового алгоритма лежит матричная (растровая) модель трассируемого поля. Результат работы этого алгоритма предполагается получать в виде целочисленной матрицы M (вариант структуры СТП с матричной моделью показан на рис. 7.3)

$$M\{C, Tb\}[\alpha_M, \delta_M](r_M) \Rightarrow \begin{bmatrix} m_{11} & \dots & m_{1n} \\ \dots & \dots & \dots \\ m_{n1} & \dots & m_{nm} \end{bmatrix}, \quad (7.6)$$

где m_{ij} – ячейки матрицы.

Используя такую структуру данных можно получить больше функциональных возможностей, в том числе и по реализации классических алгоритмов.

11. Редактирование, запись и чтение из базы данных *PTF* .

12. Преобразование структуры данных PTF в TF (блоки ПП) и вывод результатов трассировки на графический дисплей, графопостроитель, координатограф, станки с ЧПУ.

7.1.2 Преобразование структуры алгоритма функционирования СТП

Анализируя операции алгоритма СТП, можно заметить, что достаточная часть операций по подготовке исходных данных, редактированию, по работе с базами данных и выводу информации на внешние устройства имеет дело с геометрическими структурами данных. Поэтому при рассмотрении парных сочетаний локальных алгоритмов СК и СТП естественным образом напрашивается реализация алгоритма трассировки (блок ТР) в структурах данных системы конструирования, то есть целесообразно использовать геометрические алгоритмы трассировки. При этом отпадает необходимость в дополнительных преобразованиях структур данных специально под алгоритм трассировки. Это и отображено на рис. 7.1, как один из лучших вариантов построения структуры СТП. Вторым вариантом является вариант, объединяющий части подготовки модели трассируемого поля и описания цепей (см. рис. 7.2), поскольку ряд операций по вводу информации в СТП и СК являются одинаковыми.

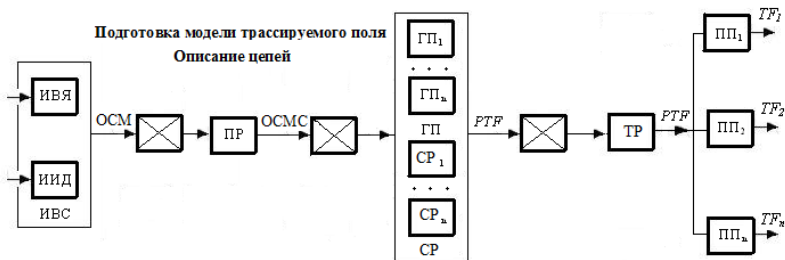


Рисунок 7.2 Второй вариант построения СТП

В качестве третьего варианта рассмотрим вариант с растровым алгоритмом трассировки (см. рис. 7.3). В структуре СТП появляются два дополнительных преобразователя структур данных. Пре-

образователь ПР1 формирует матрицу M модели трассируемого поля на основе PTF файла, а преобразователь ПР2 делает обратные преобразования.

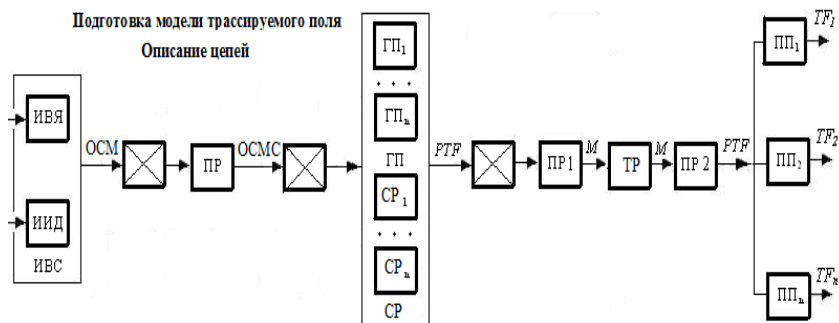


Рисунок 7.3 Третий вариант построения СТП

7.2 Алгоритм формирования растровой модели трассируемого поля

Рассмотрим более подробно алгоритм работы блока ПР1 (рис. 7.3) – преобразования аналоговой модели трассируемого поля (файл PTF) в растровую модель (матрицу M). Напомним, что файл PTF представляет собой набор отрезков прямой линии, поэтому, используя алгоритм Брезенхейма [61], легко построить проекцию каждого отрезка на матрицу M . Рассмотрим этот алгоритм более подробно.

Уравнение прямой линии имеет вид

$$y = kx + b, \quad (7.7)$$

где k – тангенс угла наклона прямой;

b – точка ее пересечения с осью y .

Если известно, что два конца отрезка заданы как точки с координатами (x_0, y_0) (x_{end}, y_{end}) , как показано на рис. 7.4, то можно найти значения тангенса угла наклона k и точки пересечения прямой с осью y по следующим формулам

$$k = \frac{y_{end} - y_0}{x_{end} - x_0}, \quad (7.8)$$

$$b = y_0 - kx_0. \quad (7.9)$$

Алгоритмы изображения прямых линий построены на уравнении прямой (7.7) и формулах (7.8) и (7.9).

Для любого заданного интервала координат x (Δx) вдоль прямой линии из уравнения (7.8) можно найти соответствующий интервал координат y (Δy) как

$$\Delta y = k\Delta x \quad (7.10)$$

Аналогично можно найти интервал координат x (Δx), соответствующий заданному Δy

$$\Delta x = \frac{\Delta y}{k}. \quad (7.11)$$

Для иллюстрации алгоритма Брезенхейма рассмотрим процесс преобразования векторной линии в растровую для прямой с положительным тангенсом угла наклона, меньше 1.0. В этом случае положения пикселей на прямой определяется разбиением на единичные интервалы по координате x . Начиная с левого конца (x_0, y_0) прямой линии и при переходе к следующему соседнему столбцу (координата x) наносится на график пиксель, который по своему номеру строки y является самым близким к направлению прямой. На рис. 7.3 показан i -ый шаг этого процесса. Предположим, что мы определили, что следует отобразить пиксель с координатами (x_i, y_i) . Далее необходимо решить, какой пиксель должен изображаться в столбце $x_{i+1} = x_i + 1$. Выбор необходимо сделать из пикселей с координатами $(x_i + 1, y_i)$ и $(x_i + 1, y_i + 1)$.

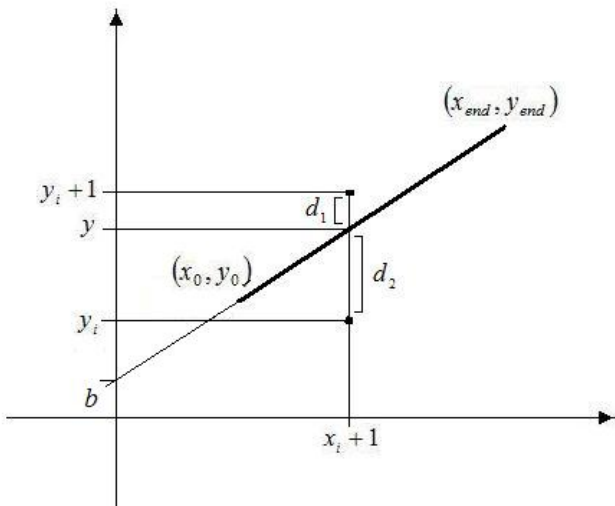


Рисунок 7.4 График векторного отрезка прямой

В точке с координатами $x_i + 1$ обозначим расстояния по вертикали от пикселей до математической прямой d_1 и d_2 . Координата y математической прямой в столбце пикселей $x_i + 1$ находится как

$$y = k(x_i + 1) + b \quad (7.12)$$

Тогда

$$d_2 = y - y_i = k(x_i + 1) + b - y_i \quad (7.13)$$

и

$$d_1 = (y_i + 1) - y = y_i + 1 - k(x_i + 1) - b. \quad (7.14)$$

Чтобы определить, какой из этих двух пикселей ближе к заданной прямой, можно провести проверку, основанную на разности двух расстояний до пикселей

$$d_2 - d_1 = 2k(x_i + 1) - 2y_i + 2b - 1. \quad (7.15)$$

Основным принципом, заложенным в этом алгоритме, является сокращение циклических вычислительных операций над вещественными переменными за счет разделения структуры алгоритма на две части: единовременные операции с вещественными переменными

ными и циклические операции с целыми переменными. Тогда для определения того какой из пикселей изображается выше или ниже математической прямой вводится понятие параметр принятия решения (p_i). Этот параметр для i -го шага алгоритма построения прямой линии находится путем такого преобразования уравнения (7.15), чтобы в него входили только целые числа, что существенно сокращает время вычисления. Это можно сделать, проведя замену $k = \Delta y / \Delta x$, где Δy и Δx – вертикальные и горизонтальные расстояния между концами отрезка, и вычислить параметр принятия решения

$$p_i = \Delta x(d_2 - d_1) = 2\Delta y \cdot x_i - 2\Delta x \cdot y_i + c. \quad (7.16)$$

Знак параметра p_i будет таким же, как и знак $d_2 - d_1$, поскольку в примере $\Delta x > 0$. Параметр c – это постоянная со значением $2\Delta y + \Delta x(2b - 1)$, которая не зависит от координат пикселя и находится в ходе рекурсивных вычислений, необходимых для расчета p_i . Если пиксель с координатой $y_i + 1$ (т.е. $d_2 < d_1$), то параметр принятия решения p_i будет отрицательным. В этом случае на график наносится нижний пиксель. В противном случае наносится верхний пиксель.

Изменение значения координаты вдоль направления прямой происходит при единичном шаге, как в направлении x , так и в направлении y . Следовательно, с помощью схемы целочисленного прироста можно найти значения последующих параметров принятия решения. На шаге $i + 1$ параметр принятия решения находится из уравнения (7.16)

$$p_{i+1} = 2\Delta y \cdot x_{i+1} - 2\Delta x \cdot y_{i+1} + c. \quad (7.17)$$

Вычитая уравнение (7.16) из предыдущего уравнения, получим

$$p_{i+1} - p_i = 2\Delta y(x_{i+1} - x_i) - 2\Delta x(y_{i+1} - y_i) \quad (7.18)$$

однако $x_{i+1} = x_i + 1$, поэтому

$$p_{i+1} = p_i + 2\Delta y - 2\Delta x(y_{i+1} - y_i) \quad (7.19)$$

где член $y_{i+1} - y_i$ равен либо 0, либо 1, в зависимости от знака параметра p_i .

Такой рекурсивный расчет параметра принятия решения выполняется в каждой точке с целым значением координаты x , начиная с координаты левого конца отрезка. Первый параметр p_0 находится из уравнения (7.16) в начальной точке с координатами (x_0, y_0)

$$p_0 = 2\Delta y - \Delta x. \quad (7.20)$$

При выполнении алгоритма постоянные $2\Delta y$ и $2\Delta y - 2\Delta x$ рассчитываются один раз для каждой прямой, поэтому арифметика алгоритма ограничивается только целочисленным сложением и вычитанием этих двух констант.

Алгоритм Брезенхейма построения отрезка прямой линии для $|k| < 1$ (см. рис. 7.4):

1. Ввести два конца отрезка и поместить левый конец в точку с координатами (x_0, y_0) .

2. Построить первую точку.

3. Вычислить постоянные Δx , Δy , $2\Delta y$, $2\Delta y - 2\Delta x$, $end = \Delta x - 1$ и найти начальное значение параметра принятия решения: $p_0 = 2\Delta y - \Delta x$

4. Для каждого x_i вдоль прямой, начиная с $i = 0$, провести следующую проверку. Если $p_i < 0$, то следующую точку следует изображать на месте пикселя с координатами $(x_i + 1, y_i)$ и $p_{i+1} = p_i + 2\Delta y$.

В противном случае следующую точку следует изображать на месте пикселя с координатами $(x_i + 1, y_i + 1)$ и $p_{i+1} = p_i + 2\Delta y - 2\Delta x$.

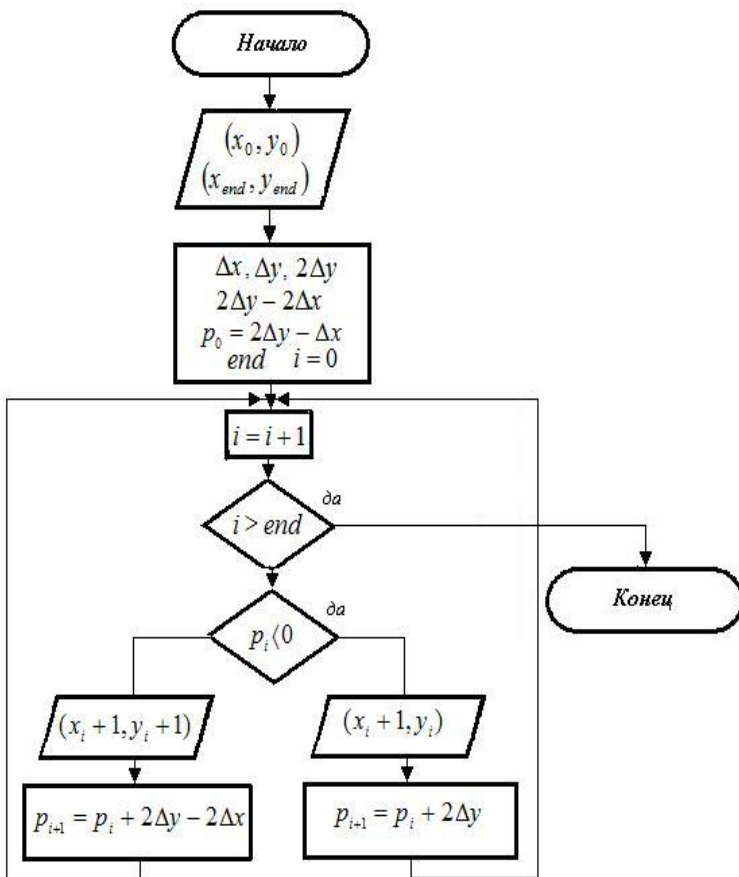


Рисунок 7.5 Алгоритм Брезенхейма построения растрового отрезка

5. Выполнить этап 4 $\Delta x - 1$ раз.

Пример:

Чтобы проиллюстрировать этот алгоритм, оцифруем отрезок прямой с концами (20,10) и (30, 18). Тангенс угла наклона этого отрезка равен 0,8 при

$$\Delta x = 10, \Delta y = 8.$$

Исходное значение параметра принятия решения

$$p_0 = 2\Delta y - \Delta x = 6,$$

а приросты для вычисления последующих параметров принятия решения равны

$$2\Delta y = 16, \quad 2\Delta y - 2\Delta x = -4.$$

Откладываем начальную точку $(x_0, y_0) = (20, 10)$ и с помощью параметра принятия решения определяем последующие положения пикселей по направлению прямой. В таблице 7.1 представлены расчетные значения параметра принятия решения по шагам алгоритма и соответствующие координаты расположения пикселей. Общая картина построения пикселей по направлению заданного отрезка прямой показана на рис. 7.6.

Таблица 7.1

Расчетные значения параметра принятия решения по шагам алгоритма и соответствующие координаты расположения

i	0	1	2	3	4	5	6	7	8	9
p_i	6	2	-2	14	10	6	2	-2	14	10
(x_{i+1}, y_{i+1})	21, 11	22, 12	23, 12	24, 13	25, 14	26, 15	27, 16	28, 16	29, 14	30, 18

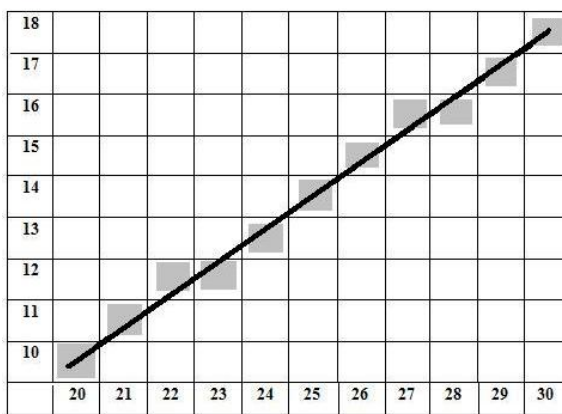


Рисунок 7.6 Общая картина построения пикселей по направлению заданного отрезка прямой

7.3 Алгоритм определения кратчайшего пути

В основу алгоритма автоматической трассировки соединений между элементами на плоскости положен широко применяемый в

различных приложениях алгоритм определения кратчайшего пути в ориентированных сетях. Рассмотрим его более подробно.

Пусть имеется сеть $G(V, A)$, где V – множество вершин сети, A – множество ориентированных дуг сети, соединяющих в любом порядке i -ую вершину с j -ой. На рис. 7.7 приведен пример сети. Вершины на рисунке обозначены кругами и пронумерованы цифрами на темном фоне, на дугах цифрами в разрыве указаны веса дуг.

Все вершины пронумерованы от 1 до N (общее число вершин сети). Всем дугам присваиваются значения $a(i, j) \geq 0$. Требуется найти кратчайший путь от вершины 1 до вершины N .

Алгоритм состоит из двух частей: вычисление минимального веса последней вершины сети и фиксация кратчайшего пути.

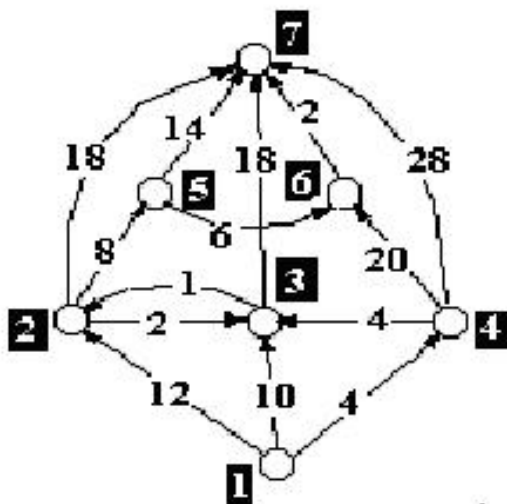


Рисунок 7.7 Пример сети $G(V, A)$

Вычисление минимального веса последней вершины сети

Последовательность определения веса последней вершины продемонстрирована в табл. 7.2 для примера рис. 7.7. В таблице в первом столбце показаны шаги алгоритма, остальные столбцы показывают значения весов вершин на каждом шаге алгоритма.

Таблица 7.2

Последовательность определения веса последней вершины

Шаг алгоритма	Вершины						
	1	2	3	4	5	6	7
Начуст.	0	∞	∞	∞	∞	∞	∞
1	-	12	10	4	∞	∞	∞
2	-	12	8	-	∞	24	32
3	-	9	-	-	∞	24	26
4	-	-	-	-	17	24	26
5	-	-	-	-	-	23	26
6	-	-	-	-	-	-	25

Алгоритм:

1. $i = 1, j = 1$,
2. Вершине b_1 присваивается вес 0. Остальным вершинам b_2, \dots, b_N присваивается ∞ .
3. $j = j + 1$.
4. Вычисляется вес вершины $b_j = b_i + a(i, j)$. Сравнивается предыдущее значение b_j с вычисленным на этом шаге алгоритма. Определяется меньшее значение ($b_{j \min}$). Вершине b_j присваивается меньшее значение: $b_j = b_{j \min}$ (В табл. 7.2 минимальные веса вершин выделены жирным шрифтом). Если $j = N$, то $j = 1$ и переход на п.5, иначе переход на п.3.

5. Вершина $b_{j_{min}}$ помечается специальным маркером и в дальнейших расчетах в качестве конца дуги не используется, если $j_{min} = N$, то переход на п.6, иначе $i = j_{min}$ и переход на п.3.
6. Конец. Таким образом, вершина N имеет минимальный вес $b_{N_{min}}$, соответствующий кратчайшему пути.

Таким образом, как следует из табл. 7.2 вычислительная сложность P операций алгоритма определяется следующим образом

$$P \leq m_0(N-1)^2 \quad (7.21)$$

где m_0 – количество операций по вычислению веса одной вершины;

N – общее число вершин.

Фиксация кратчайшего пути

На последнем шаге алгоритма получено конкретное значение длины пути, что вполне достаточно для некоторых приложений (см. например [59]). Что же касается систем трассировки, то для них необходимо дополнительно зафиксировать конфигурацию кратчайшего пути. При этом можно использовать несколько вариантов реализации алгоритма фиксации.

Принцип работы первого варианта алгоритма [29] продемонстрируем на примере табл. 7.3. В таблице линиями, соединяющими минимальные значения вершин на каждом шаге алгоритма, отмечен полученный кратчайший путь. Курсивом на свободных строках рядом с каждой линией указаны значения дуг, соединяющих соответствующие вершины в сети.

Первый вариант алгоритма фиксации:

1. $i = N-1, j = N, b_j = b_{N_{min}}$. Запись в файле «Конец дуги №» $(N)(i,j)$
2. $i = i - 1$, если $i = 0$ и $j=1$ переход на п.5.
3. $\kappa = b_j - a(i, j)$. Если $b_j = b_{N_{min}}$ или $b_j = \kappa$, то записать в файлах «Начало дуги №» $(i)(i,j)$
4. $j = j - 1$, если $j=1$, то $j = N$ переход на п.2, иначе переход на п.3.
5. Конец.

Таблица 7.3

Принцип работы первого варианта алгоритма

Шаг алгоритма	Вершины						
	1	2	3	4	5	6	7
Нач.уст.	0	∞	∞	∞	∞	∞	∞
			4				
1	-	12	10	4	∞	∞	∞
				4			
2	-	12	8	-	∞	24	32
		1					
3	-	9	-	-	∞	24	26
			8				
4	-	-	-	-	17	24	26
					6		
5	-	-	-	-	-	23	26
						2	
6	-	-	-	-	-	-	25

Следует заметить, что в алгоритме определения кратчайшего пути не обязательно все вершины с минимальными весами, полученными на каждом шаге алгоритма, могут быть включены в состав кратчайшего пути, и, что окончательный результат формируется только на последнем шаге. Покажем это на следующем примере.

Пусть на примере (рис. 7.7) вес дуги 4-7 будет равен 20-ти. Тогда процесс фиксации будет выглядеть так, как показано в табл. 7.4. Старый путь обозначен тонкими линиями, а новый – толстыми.

Второй вариант алгоритма фиксации использует список номеров вершин, от которых в последующей вершине появился минимальный вес. Таким образом, запоминается только «направ-

ление», откуда получился минимальный вес в каждой вершине сети. Минимальный путь в сети фиксируется последовательно от последней вершины к первой, по мере продвижения по списку «направлений».

Таблица 7.4

Процесс фиксации

Шаг алгоритма	Вершины						
	1	2	3	4	5	6	7
Нач. уст.	0	∞	∞	∞	∞	∞	∞
1	-	12	10	4	∞	∞	∞
2	-	12	8	-	∞	24	24
3	-	9	-	-	∞	24	24
4	-	-	-	-	17	24	24
5	-	-	-	-	-	23	24
6	-	-	-	-	-	-	24

Моделирование различных вариантов построения системы автоматической трассировки проводилось на программном обеспечении. Была создана программная система «Графика – 01-Т» [24], позволяющая описывать схемную документацию, формировать электронные модели устройств на разных этапах проектирования, решать задачи автоматической трассировки соединений между элементами на плоскости с использованием алгоритма определения кратчайшего пути.

В системе модель автоматической трассировки соединений для любой конфигурации проектируемой схемы представлена в виде матрицы M (рис. 7.8), на которую проецируется соответствующая схема. Размер матрицы выбирается исходя из требуемых минимальных расстояний между элементами схемы и максимальных ее размеров. Множество вершин V (на рисунке обозначено квадратами, пронумерованными с внешней стороны) состоит из точек подсоединения связей между элементами, множество дуг A (на рисунке обозначено стрелками) показывают направления расчета кратчайшего пути. Ортогональные дуги имеют вес 1, диагональные дуги – 1.42. Общее число элементов матрицы равно N . В процессе расчета пути каждая из вершин b_i принимает следующие значения:

0 – начало трассы;

p – препятствие;

∞ – первоначальные значения не подсчитанных вершин.

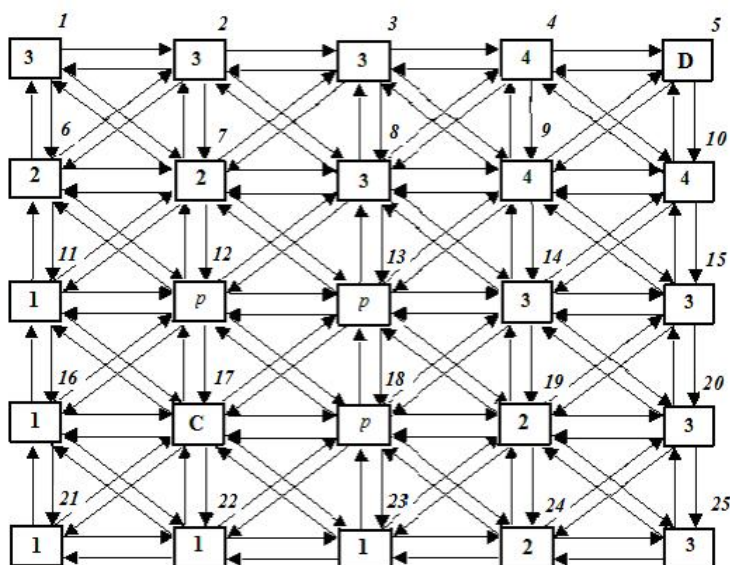


Рисунок 7.8 Вычислительная модель автоматической трассировки соединений для любой конфигурации проектируемой схемы

Остальные вершины в процессе расчета принимают значения между 0 и ∞ .

Началом трассы является вершина С, концом трассы – вершина D. Общий принцип расчета весов вершин напоминает процесс распространения фронта волны от точки С до точки D. На рис. 7.8 фронт каждой волны соответствует одной строке «шага алгоритма» таблицы 7.2 и помечается цифрами внутри квадратов. Расчеты заканчиваются на $(n - 1)$ -ом шаге, когда фронт волны достигнет вершину D.

Вычислительная сложность P операций алгоритма на однопроцессорном компьютере определяется соотношением

$$P \leq m_0(8N - N_p - 1) \quad (7.22)$$

где m_0 – количество операций по вычислению веса одной вершины,

N – общее число ячеек матрицы;

N_p – количество ячеек занятое препятствиями p .

Ускорить процесс определения кратчайшего пути возможно за счет замены программной реализации вычислительной модели автоматической трассировки аппаратной. В этом случае функции каждой вершины реализуются специализированным процессором (СП). Все СП функционируют параллельно. Вычислительная сложность P операций алгоритма на параллельно работающих СП определяется следующим образом

$$P = m_0 \left(\left\lceil \frac{\sqrt{N - N_p}}{2} \right\rceil + N_p \right), \quad (7.23)$$

где $\lceil X \rceil$ – означает целое число от X .

Последовательность работы СП показана на рис. 7.8. Цифрами в квадратах обозначена последовательность срабатывания СП. Наличие препятствий нарушает параллельность работы СП и тем самым увеличивает время его работы.

7.4 Структурная организация комплекса программных средств проектирования РЭА из набора имеющихся на рынке систем

Допустим, что реализуется алгоритм, представленный на рис. 7.9 [11], включающий этапы проектирования принципиальных схем (ПрСх) РЭА, печатных плат (ПП), выпуска конструкторско-технологической документации на 2D и 3D монтажные схемы (МСх).

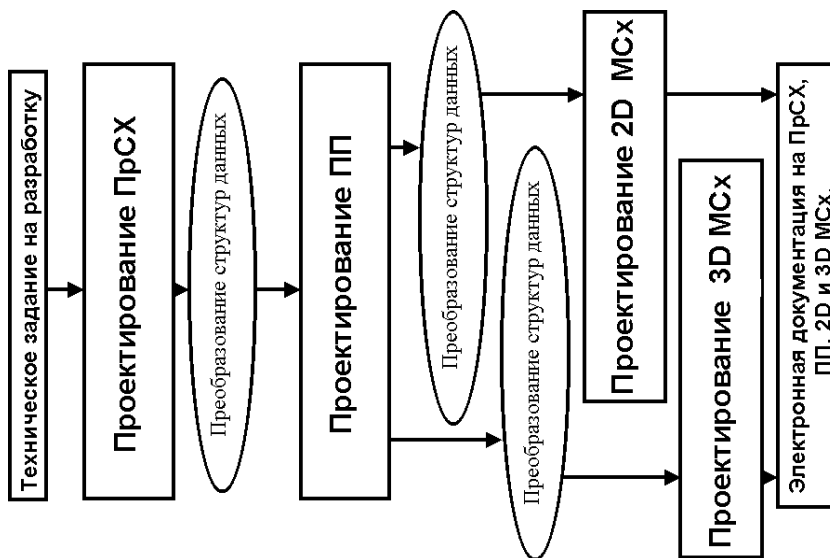


Рисунок 7.9 Обобщенная структурная модель реализаций нескольких этапов проектирования РЭА

На рынке в настоящее время существует достаточное количество отечественных и зарубежных систем, определенное сочетание которых может решить поставленную задачу. Например, могут быть использованы такие системы, как AutoCAD, P-CAD, OrCAD, Protel, Inventor, Solid Works, I-Deas и т.п. Однако, следует заметить, что, с одной стороны, каждая из систем имеет собственные структуры данных, с другой стороны, имеются международные стандарты по обмену данными между различными системами, поэтому на каждом из этапов должна решаться задача преобразова-

ния структур данных. Таким образом, на основе рис. 7.11 должна быть построена обобщенная структурная модель (ОСМ) из набора систем и программ преобразования структур данных с указанием значений их качественных показателей (критериев). Задача выбора лучшей реализации может быть сведена к алгоритму определения кратчайшего пути в сети ОСМ.

На рис. 7.10 – 7.14 показаны соответственно примеры из всех возможных вариантов структурных реализаций.

Основными операциями алгоритма функционирования САПР РЭА являются:

1. Разработка схемы электрической принципиальной (СхПР).
2. Создание базы данных элементов СхПР.
3. Размещение элементов СхПР.
4. Трассировка соединений между элементами на СхПР.
5. Выпуск таблицы перечня элементов на СхПР.
6. Разработка электронной документации и вывод чертежей на СхПР.
7. Автоматическая трассировка соединений между элементами на печатной плате (ПП).
8. Создание базы данных элементов ПП.
9. Размещение элементов ПП. Подготовка модели трассируемого поля ПП.
10. Трассировка соединений между элементами на ПП.
11. Выпуск фотошаблонов.
12. Разработка электронной документации на монтажные схемы (СхМ).
13. Проектирование конструктивов.

На рис. 7.9 показаны основные этапы проектирования, соответствующие указанному алгоритму. Реализация каждого этапа покрывается множеством имеющихся на рынке систем. Взаимодействие между системами, в том числе между отдельными этапами, производится через преобразования структур данных. На рисунке показаны возможные структуры данных. Лучшая реализация выбирается по заданным критериям. На рис. 7.10 – 7.14 приведены некоторые из возможных вариантов реализаций.

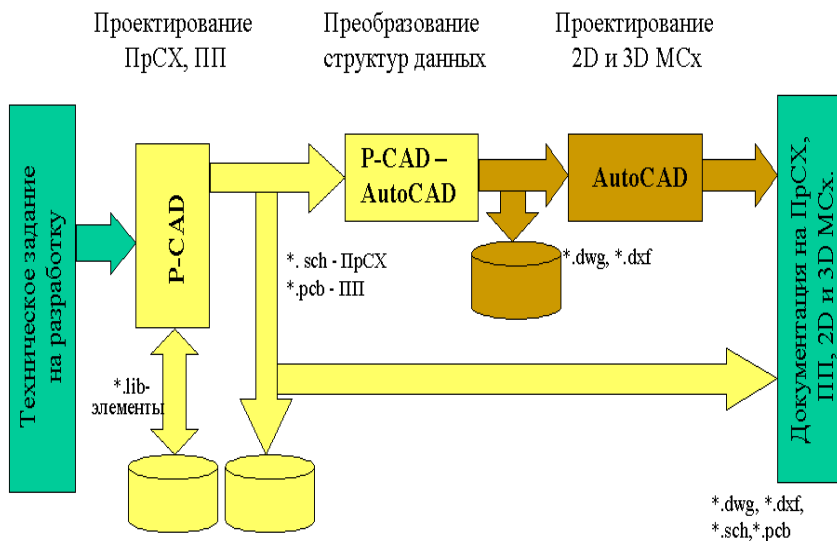


Рисунок 7.10 Первая реализация этапов проектирования РЭА

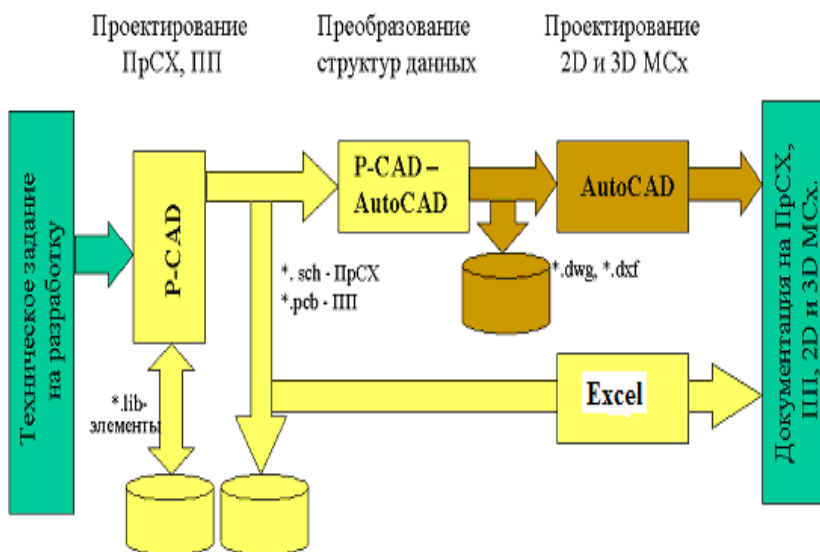


Рисунок 7.11 Вторая реализация этапов проектирования РЭА

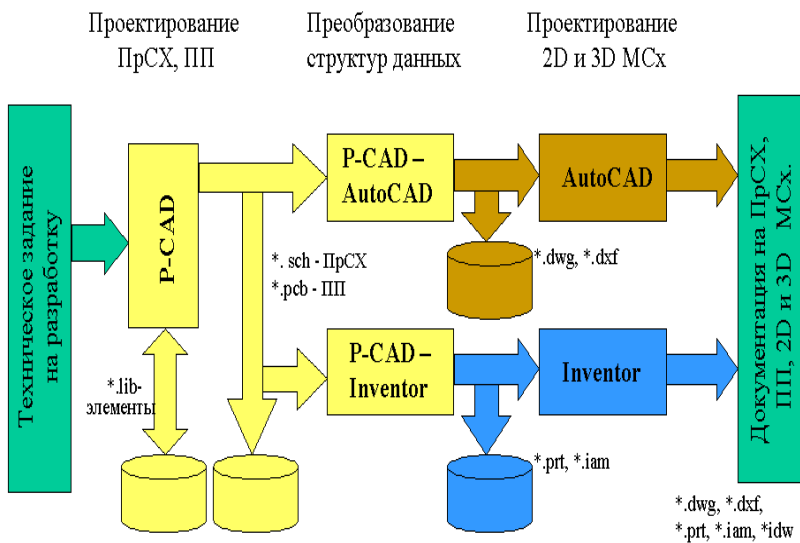


Рисунок 7.12 Третья реализация этапов проектирования РЭА

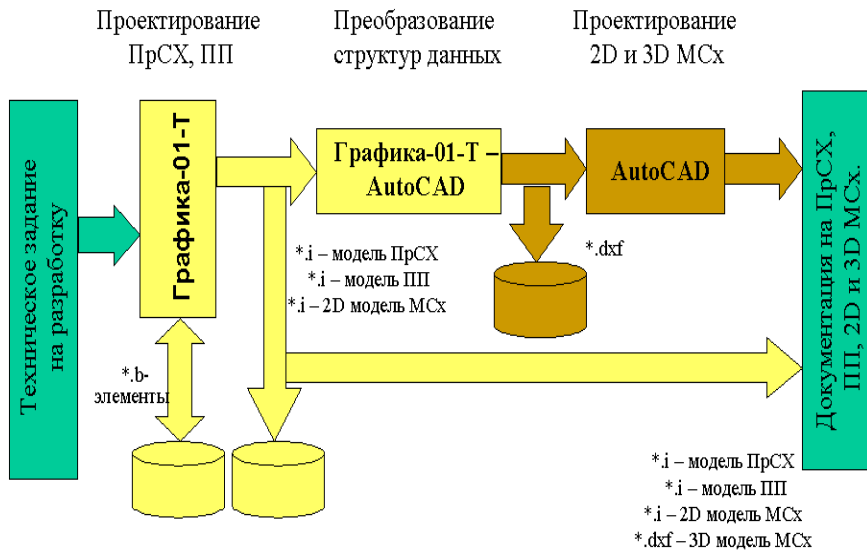


Рисунок 7.13 Четвертая реализация этапов проектирования РЭА

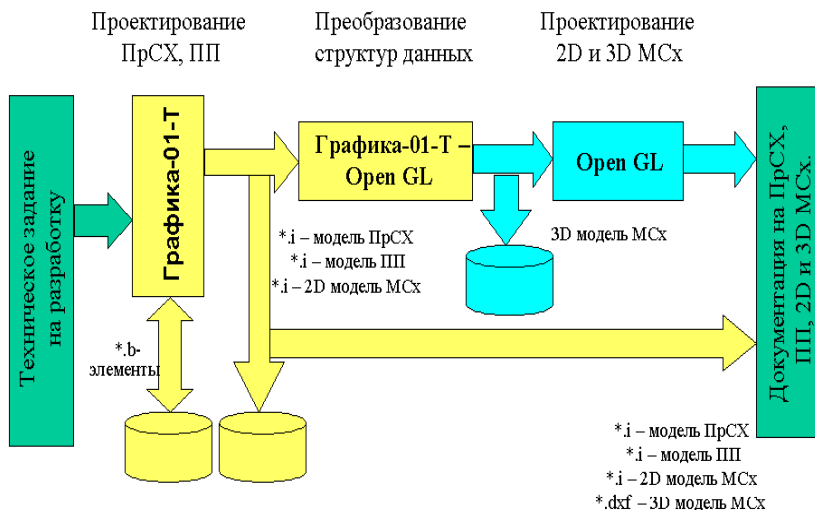


Рисунок 7.14 Вариант реализации на системе собственной разработки (Графика-01-Т) с добавлением элементов 3D монтажных схем, созданных на основе библиотеки Open GL

Используемые структуры данных на рисунках отмечены знаком (*.). На рис.7.10 – 7.13 показана реализация из набора существующих на рынке систем, на рис. 7.14 представлен вариант реализации на системе собственной разработки (Графика-01-Т) с добавлением элементов 3D монтажных схем, созданных на основе библиотеки Open GL. Последний вариант используется в процессе обучения студентов.

Следует заметить, что аналогичным образом формируются комплексы не только для проектирования в области РЭА, но и в машиностроении. Существенным моментом при выборе лучшей структурной реализации является определение типа критерия, на основании чего реализация считается лучшей.

7.5 Система «Графика-01-Т». Интерактивные средства взаимодействия пользователя с системой

Система предназначена для студентов, изучающих специализированные курсы по компьютерной графике и схмотехническо-

му проектированию.

Для проектирования схемной документации в настоящее время используется большое количество систем. К ним относятся, например, системы AutoCAD, P-CAD, Caddy и др. Эти системы различаются объемами занимаемой памяти, функциональными возможностями, алгоритмами формирования баз данных типовых элементов, размещения элементов на плоскости схемы и трассировки соединений между элементами. В таких системах используются различные принципы формирования таблиц спецификаций и соединений.

Под схемной документацией понимаются схемы алгоритмов, структурные, функциональные, принципиальные и монтажные схемы. В работе на основе существующей нормативной документации, сравнительного анализа функциональных возможностей и тенденций развития систем проектирования акцентируется внимание на следующих особенностях автоматизированного проектирования схемной документации:

1. Принципы организации интерфейсов пользователей на основе средств операционной системы Windows.
2. Организация файловой системы и управление вводом и выводом информации.
3. Структура функций черчения, редактирования, определения свойств и режимов.
4. Создание библиотек типовых элементов схемной документации.
5. Описание соединений между элементами схемы.
6. Формирование математических моделей на всех этапах проектирования схемной документации.
7. Размещение элементов в плоскости схемы и трассировка соединений между элементами.
8. Анализ и редактирование математической модели схем.
9. Создание спецификаций и таблиц соединений.

Таким образом, предлагается при обучении акцентировать внимание на изучении общих принципов автоматизированного проектирования схемной документации, особенно на формировании электронных моделей на каждом из этапов проектирования в соответствии с идеологией CALS-технологий. Такой подход имеет следующие преимущества по сравнению с изучением конкретных коммерческих систем проектирования:

1. Минимальные затраты на организацию учебного процесса и минимальные требования к вычислительным ресурсам используемых компьютеров.
2. Основные алгоритмы автоматизированного проектирования схемной документации сконцентрированы в одной системе.
3. Для изучения процессов проектирования всего спектра схемной документации не требуется использования нескольких систем.
4. Изучаются основные методы автоматизированного проектирования схемной документации, а не конкретная Система предназначена для подготовки исходных данных, проектирования структурных, функциональных, принципиальных и монтажных схем, а также печатных плат, выпуска чертежной документации в электронном виде или на твердых носителях. Далее будут приведены только отдельные выдержки из полной инструкции по работе с системой «Графика-ТР» касающиеся структур данных, файловой системы, средств для создания электронных моделей, состоящих из элементов различной конфигурации и связей между ними. Кроме того, выбор типов и параметров автоматической трассировки на основе алгоритма определения кратчайшего пути и собственно трассировка.

7.5.1 Структуры данных системы «Графика-ТР»

Система работает со следующими структурами данных:

- *.b – файл с именем (*) и расширением (.b) является внутренним двоичным файлом комплекса, в котором размещается описание 2D модели объекта;
- *.i – двоичный файл комплекса, в котором размещается последовательная запись координат геометрических примитивов в ви-

де ломаных линий и текста, описывающих 2D модель объекта, с разделителями между ними;

- *.ppb – файл с именем создаваемого двоичного файла с расширением .i формируется автоматически при использовании шрифтов WINDOWS; эти файлы необходимо копировать совместно при сохранении и повторном использовании созданной 2D модели схемы;
- *.bmp – стандартный растровый файл;
- *.gbr – символьный GERBER файл, предназначенный для вывода информации на внешние устройства (графопостроители, координатографы, принтеры и фото-плоттеры);
- *.hpg – символьный HPGL файл, предназначенный для вывода информации на внешние устройства (графопостроители, координатографы, принтеры и фото-плоттеры);
- *.plt – символьный HPGL файл системы PCAD, предназначенный для вывода информации на графопостроители.
- *.dxf – символьный файл системы AutoCAD, предназначенный для обмена с внешними системами.

7.5.2 Интерактивные средства взаимодействия пользователя с системой «Графика-ТР»

Работа с системой производится через пиктографическое меню, размещенное в верхней строке экрана дисплея (рис. 7.15). Меню содержит следующий набор функций: File, View, Options, Draw, Show, Edit, Trassir, Connection, ViewScheme.

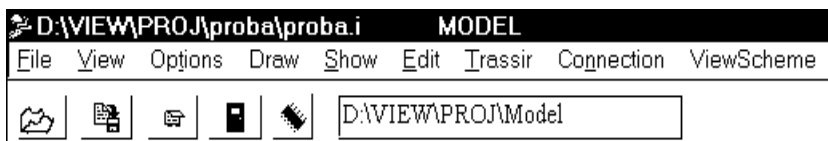


Рисунок 7.15 Пиктографическое меню

Функция **File** включает команды (рис. 7.16): NewGrafica, NewModel, Open, Select a component, Add, Copy, Preview, ReadTest, Trafaret, Renew Fcoord, Save bloc, Save as, Print, Close, Specification, Exit.

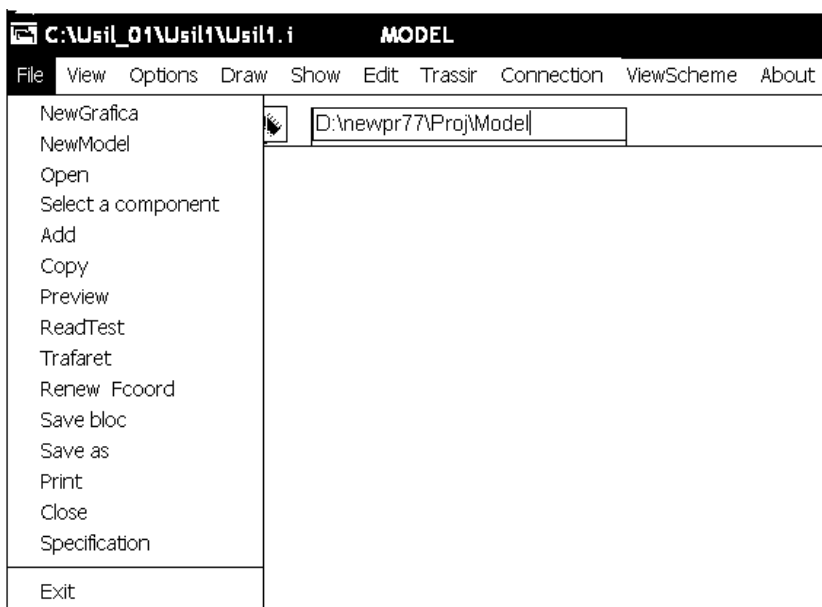


Рисунок 7.16 Меню функции File

Команда **NewModel** предназначена для создания новой принципиальной схемы. В рабочей (текущей) директории (в ней записан файл Hrgar7.exe) автоматически будет создана директория Model (если она отсутствовала) с подразделом Serv. В этих директориях будет храниться вся информация о схеме во время работы в программе, по завершению работы со схемой вся информация из этих директорий исчезнет.

Команда **Save bloc** используется для создания библиотеки типовых элементов схем или фрагментов чертежей (блоков).

Команда **Save as** сохраняет изображение, активное в текущий момент на экране дисплея, как файл в одном из следующих форматов: *.i, *.gbr, *.hpg, *.plt, *.bmp, *.dxf.

Команда **Print** выводит информацию на внешние устройства.

По команде **Exit** производится закрытие системы.

Функция **View** включает команды: **Zoom in**, **Zoom +**, **Zoom –**, **Original**, **All** (рис. 7.17).

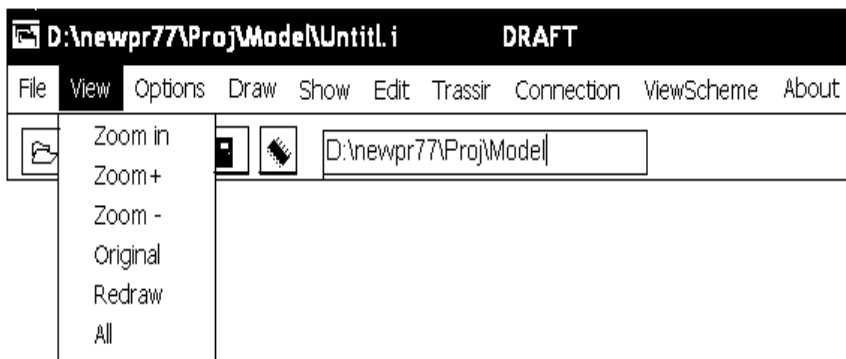


Рисунок 7.17 Меню функции View

Zoom in увеличивает выделенный фрагмент изображения до размеров поля экрана дисплея. Фрагмент изображения выделяется прямоугольником аналогично п.4.1.9.

Zoom+ последовательно увеличивает все изображение в 1.2 раза.

Zoom– последовательно уменьшает все изображение в 1.2 раза.

Original возвращает исходный масштаб изображения.

Redraw перерисовывает изображение на экране дисплея.

All все изображение размещает на экране дисплея.

Функция **Draw** включает команды построения геометрических и функциональных примитивов (рис. 7.18): **Polyline**, **Polyline_filled**, **Rectangle**, **Rect_filed**, **Circle**, **Arc**, **Ellips**, **Ell_arc**, **Text**, **Fasten cont**, **Free cont**. Для активизации последней команды рисования, кроме **Fasten cont** и **Free cont**, достаточно нажать правую кнопку «мыши».

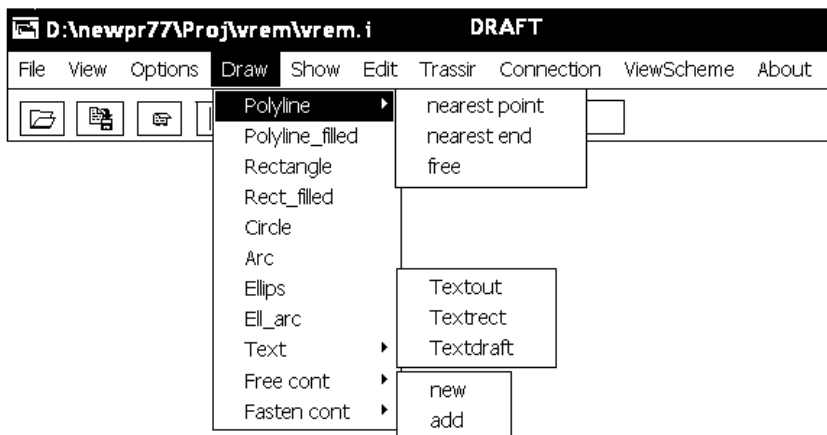


Рисунок 7.18 Меню функции Draw

Polyline – построение ломаных линий. Включает следующие команды: **nearest point** – привязка к ближайшей точке ближайшего примитива, **nearest end** – привязка к конечной точке ближайшего примитива, **free** – рисование ломаной без привязки (команда **Option->The current point** должна быть отключена off, в противном случае контакты будут притягиваться в соответствии выбранной опции).

Polyline_filled – построение заполненного многоугольника. Для получения замкнутого контура первая и последняя вершины многоугольника соединятся автоматически.

Rectangle – построение прямоугольников.

Rect_filled – построение заполненного прямоугольника.

Circle – построение окружностей по центру и радиусу.

Arc – построение дуги окружности по центру, радиусу и двум углам.

Ellips – построение эллипса по центру и двум радиусам.

Ell_arc – построение дуги эллипса по центру, двум радиусам и двум углам.

Text – написание символов.

Fasten cont – создание контактов на элементах (**new** – начать создание меток-контактов, **add** – добавление меток-контактов к ранее созданным). Понятие контакта специально введено для автоматической трассировки, контакт является активным элементом и используется для создания связей в модели проектируемой схемы.

Free cont – свободное размещение контактов на чертеже (**new** – начать создание меток-контактов, **add** – добавление меток-контактов к ранее созданным).

Функция **Edit** включает команды (рис. 7.19): **Undo**, **Move**, **Rotate**, **Mirror**, **Scale**, **Copy_pr**, **Copy_mirror**, **Delete**, **PenPrimitive**, **Otrezok-polyline**, **Move-uzel**. Команды действуют на предварительно выделенный фрагмент изображения. Для активизации последней команды рисования достаточно нажать правую кнопку «мыши».

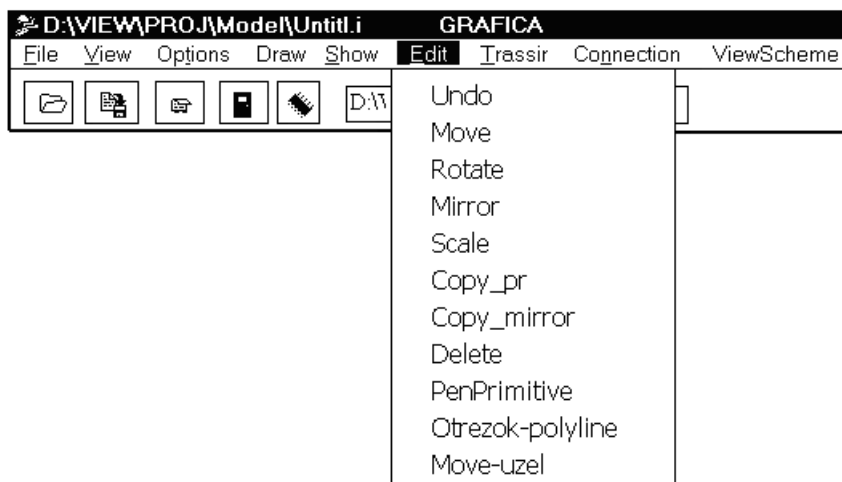


Рисунок 7.19 Меню функции Edit

Undo – возврат на шаг.

Move – перемещение примитива, элемента (блока) со связями (если они есть) на схеме.

Rotate – поворот примитива, элемента (блока) со связями на схеме.

Mirror – зеркальное отображение примитива, блока.

Scale – изменение масштаба примитива, блока.

Copy_pr – копирование примитивов.

Copy_mirror – копирование примитивов с зеркальным отображением.

Delete – удаление примитива, элемента (блока) и перестройка связей на принципиальной схеме.

PenPrimitive – изменение свойств примитива (цвет, толщина, тип линии).

Otrezok-polyline – преобразование отдельно взятого отрезка или отрезка, являющегося частью ломаной линии, в ломаную линию.

Move-uzel – смещение одного из концов отдельно взятого отрезка или отрезка, являющегося частью ломаной линии.

Функция **Connection** включает команды по созданию и удалению связей между элементами на схемах (рис. 7.20): **Creating, Del contact, Del connection.**

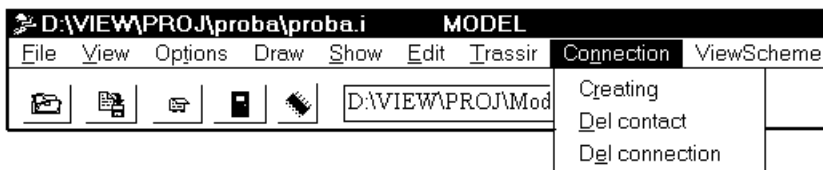


Рисунок 7.20 Меню функции Connection

Creating – создание связей (цепей) между элементами на схемах.

Del contact – удаление контакта из связи.

Del connection – удаление связи.

При вводе связей между элементами на схемах используется команда **Connection\Creating**. Курсор мыши последовательно подводится к каждому из контактов элементов, входящих в текущее соединение. При захвате курсором каждого контакта соединения нажимается левая клавиша мыши. На экране дисплея фиксируются связи между контактами в виде отрезка прямой линии голубого цвета. После указания всех контактов цепи нажимается правая клавиша мыши. Далее таким же образом вводятся следующие соединения схемы. В памяти компьютера формируется 2D модель схемы, в которой элементы связаны между собой таким образом, что при любом изменении размещения элементов в плоскости схемы, созданные соединения между элементами не изменяются и не разрываются. При перемещении элемента линии связи становятся «резиновыми» и вытягиваются за элементом. 2D модель схемы оказывается удобна при итерационном процессе решения задач размещения и трассировки с целью минимизации общей длины трасс, количества пересечений трасс, лучшей читаемости схемы и т.п.

Функция **Trassir** производит автоматическую трассировку соединений между элементами на схемах. Предварительно векторная 2D модель схемы проецируется на плоскую матрицу, в которой формируется растровая модель схемы. Трассировка использует универсальный многокритериальный алгоритм определения кратчайшего пути на растровой модели и может легко модифицироваться. Критериями трассировки являются длина пути (цена элементарного шага на растровой модели), цена стремления к цели, цены пересечений и изгибов трасс.

Функция **Trassir** включает команды (рис. 7.21): **Trass.data**, **Direct**, **Inverse**.

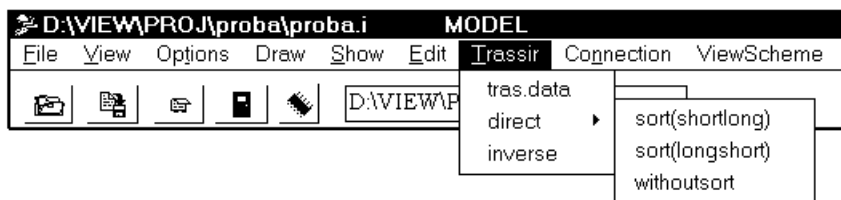
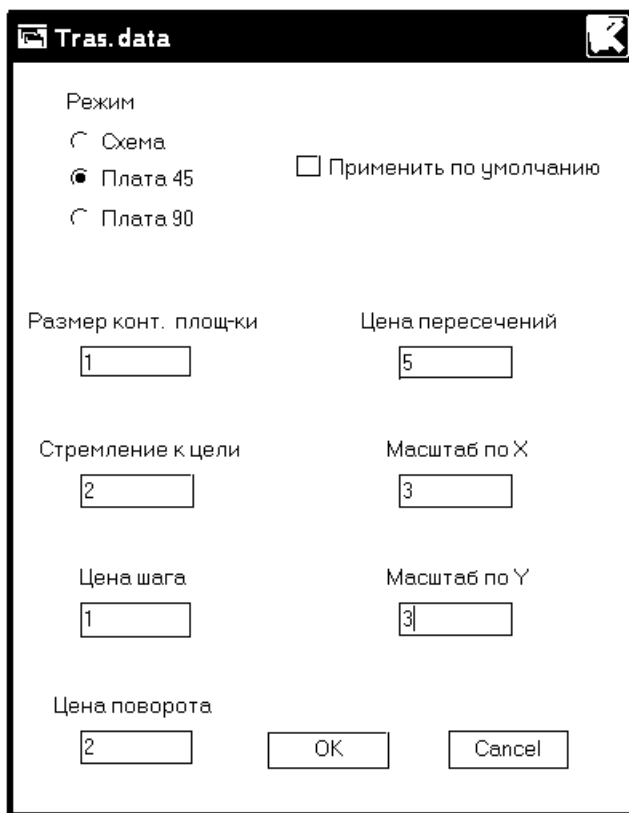


Рисунок 7.21 Меню функции Trassir

Trass.data – установка параметров автоматической трассировки соединений между элементами (рис. 7.22).



The image shows a dialog box titled "Tras.data" with a standard Windows icon on the left and a help icon on the right. The dialog contains several settings for automatic routing:

- Режим (Mode):** Three radio buttons are present: "Схема" (unselected), "Плата 45" (selected), and "Плата 90" (unselected). To the right of these is a checkbox labeled "Применить по умолчанию" (Apply by default), which is currently unchecked.
- Размер конт. площадки (Board size):** A text input field containing the value "1".
- Цена пересечений (Intersection cost):** A text input field containing the value "5".
- Стремление к цели (Goal orientation):** A text input field containing the value "2".
- Масштаб по X (Scale X):** A text input field containing the value "3".
- Цена шага (Step cost):** A text input field containing the value "1".
- Масштаб по Y (Scale Y):** A text input field containing the value "3".
- Цена поворота (Turn cost):** A text input field containing the value "2".

At the bottom right of the dialog are two buttons: "OK" and "Cancel".

Рисунок 7.22 Установка параметров автоматической трассировки соединений между элементами

Direct, Inverse – режимы трассировки. Первый используется для проведения трасс в местах свободных от любых элементов изображений. Второй – проводит трассы по векторным линиям изображений. Для проектирования схемной документации используется только режим **Direct**. Режим **Inverse** используется для определения кратчайшего пути, например, между объектами на географических картах. Режим **Direct**, в свою очередь, включает в

себя три функции: sort(shortlong), sort(longshort) и withoutsort – способы сортировки трасс.

На рис. 7.23 показан фрагмент автоматической трассировки соединений на печатной плате.

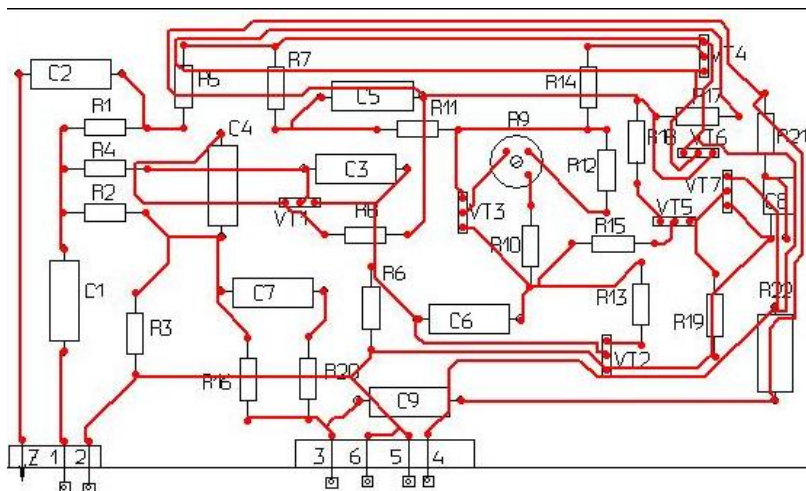


Рисунок 7.23 Фрагмент автоматической трассировки соединений на печатной плате

Глава 8

ЭФФЕКТИВНОСТЬ ИСПОЛЬЗОВАНИЯ ИНТЕРАКТИВНЫХ ПРОГРАММНЫХ СИСТЕМ

С середины 70-х годов начаты работы по исследованию принципов построения и созданию программно реализованных интерактивных систем, в частности, систем автоматизированного проектирования (САПР). Был предложен новый подход по структурной организации систем проектирования, похожий на аппаратную реализацию спецпроцессоров, в которых выделялась инвариантная по отношению к решаемым задачам часть систем, включающая в свой состав средства взаимодействия пользователя с системой и внешними устройствами, и проблемно-ориентированная часть.

К 1981 г. была разработана идеология построения и создан интерактивный интегрированный программный комплекс “ГРАФИКА-81”, включающий подсистемы выпуска конструкторско-технологической документации (“ГРАФИКА-81-2D”), моделирования пространственных конструкций (“ГРАФИКА-81-3D”), автоматического размещения элементов и трассировки соединений на принципиальных схемах и печатных платах (“ГРАФИКА-81-ТР”), подготовки управляющей информации для станков с ЧПУ.

Следует отметить, что разработка систем автоматизированного проектирования является достаточно сложной задачей, требует много времени и участия большого количества высококвалифицированных разработчиков. Стоимость программной реализации для некоторых зарубежных САПР составляет 20-50 тысяч долларов США за одно рабочее место, а затраты только на их продвижение на рынке измеряются миллионами долларов в квартал.

Сложность разработки таких систем в свое время усугублялась обилием разных технических средств и операционных систем, на которых последовательно создавался комплекс “ГРАФИКА-81” – от ICL-4-70, ЕС-ЭВМ, М-6000, СМ-1420 до персональных компьютеров.

К концу 90-х годов была сформулирована общая теория формального синтеза структур интерактивных систем, реализованных в виде технических и программных средств. В дальнейшем велась проработка методов объёмного геометрического моделирования,

структур и программного обеспечения интерактивных систем на основе средств виртуальной реальности.

8.1 Моделирование внешнего облика орбитальной станции МИР

Институтом проблем управления РАН совместно с Центром подготовки космонавтов им. Ю.А.Гагарина разработаны объемные геометрические модели внешнего облика всех модулей орбитальной космической станции «МИР». Модели предназначались для обучения космонавтов составу орбитального комплекса. Работа была закончена в 1995 г. Для создания объемных геометрических моделей использовался программный комплекс «Графика-81-3D» [1, 14, 26].

Модель орбитального комплекса МИР (рис. 8.1 смотри цветную вставку) содержит геометрические модели следующих модулей: «Базовый», «Квант-1», «Квант-2», «Кристалл», «Природа», «Спектр», грузовой транспортный корабль «Прогресс» и транспортный корабль «Союз». Модель каждого модуля содержит сведения о конструкции и компоновке в следующем объеме: габаритные размеры и взаимное расположение отсеков, иллюминаторов и стыковочных агрегатов; размещение, угловые и линейные размеры внешних элементов конструкций (антенн, солнечных батарей, датчиков, баков, гиродинов, двигателей, агрегатов и манипуляторов системы перестыковки, трасс выхода и фар).

Для модели всего комплекса характерно наличие большого числа мелких деталей (поручней, антенн, датчиков и т.п.) и крупногабаритных конструкций (корпусов, солнечных батарей и ферм). Соотношения размеров мелких деталей и крупногабаритных конструкций требуют от системы геометрического моделирования достаточно точного представления моделей в памяти компьютера.

Впервые в России были разработаны объемные геометрические модели станции МИР. Появилась возможность на этих моделях, во-первых, изучать месторасположение внешних элементов конструкции всех модулей станции, наблюдать, прокладывать и изучать предполагаемые маршруты перемещения космонавтов. Во-вторых, появилась возможность предварительного моделирования процессов развития общей конструкции станции МИР и ее

отдельных узлов, а также контроля внештатных ситуаций без использования физических моделей.

Запуск «Базового» модуля станции «МИР» (рис. 8.2 смотри цветную вставку) был осуществлен 20 февраля 1986 года. Станция «МИР» имела два причала с агрегатами, к которым могли пристыковываться космический аппарат "Союз" для доставки и смены экипажей, а также беспилотные модули различного назначения и грузовые корабли "Прогресс". Один из причалов станции оборудован устройством для автоматической перестыковки модулей на четыре боковых узла.

Модуль «Квант-1» (рис. 8.3 смотри цветную вставку) был запущен на орбиту 31.03.87 г. В нем размещались астрофизические приборы, установка для получения сверхчистых биологически активных веществ в невесомости, оборудование для визуальных наблюдений земной поверхности. На модуле располагались две ферменные конструкции «Рапана» (меньшая по размеру) и «Софора» (большая). Фермы предназначались для размещения отдельных приборов на удалении от общего корпуса станции и проведения различных экспериментов.

Модуль «Квант-2» (рис. 8.4 смотри цветную вставку) был включен в состав станции в декабре 1989 года. На модуле были установлены системы управления движением с использованием силовых гироскопов, системы электропитания, новые установки для получения кислорода и регенерации воды, приборы бытового назначения.

Модуль «Кристалл» (рис. 8.5 смотри цветную вставку) был пристыкован к станции в июле 1990 года. Стыковочный узел на модуле "Кристалл" предназначался для стыковки с многоразовыми кораблями типа "Буран" и "Шаттл". В июне 1995 года он был использован для стыковки с американским кораблем "Атлантис".

Модуль «Спектр» (рис. 8.6 смотри цветную вставку) дополнил станцию МИР 1 июня 1995 года. Модуль предназначался для проведения исследований природных ресурсов Земли, верхних слоев земной атмосферы, собственной внешней атмосферы орбитального комплекса, геофизических процессов естественного и искусственного происхождения в околоземном космическом пространстве и в верхних слоях земной атмосферы, для проведения медико-биологических исследований по совместным российско-

американским программам «Мир-Шаттл» и «Мир-НАСА», для оснащения станции дополнительными источниками электроэнергии.

Модуль «Природа» (рис. 8.7 смотри цветную вставку) был пристыкован к станции в 1996 году. Он предназначался для проведения научных и экологических исследований поверхности и атмосферы Земли, атмосферы вблизи станции, для проведения биологических экспериментов, исследования влияния космического излучения на организм человека, получения особо чистых лекарственных препаратов, исследования поведения различных материалов в условиях открытого пространства и др.

3D модель грузового транспортного корабля «Прогресс» показана на рис. 8.8 (смотри цветную вставку).

3D модель транспортного корабля «Союз» показана на рис. 8.9 (смотри цветную вставку).

8.2 Моделирование процессов автоматической перестыковки модулей станции «МИР»

Как уже отмечалось, один из причалов станции «МИР» оснащен специальным устройством автоматической перестыковки модулей. С целью обучения принципу работы этого устройства был разработан компьютерный фильм с использованием созданных 3D моделей.

Перестыковка модулей необходима для освобождения стыковочного узла станции к приему транспортного корабля. В процессе перестыковки задействованы три основных объекта: стыковочный узел, манипулятор стыковочного узла, перестыкуемый модуль. Эти объекты при перестыковке перемещаются относительно друг друга. Взаимодействие объектов наглядно показано в разработанном учебном компьютерном фильме.

Описание процесса перестыковки и создание кадров компьютерного фильма проводилось на языке комплекса «Графика-81-3D». На рис. 8.10 (смотри цветную вставку) показаны основные этапы перестыковки, соответствующие начальному состоянию, промежуточному, когда перестыкуемый модуль захвачен манипулятором (элементы манипулятора на рисунке обозначены красным цветом) и конечному состоянию.

8.3 Моделирование крупногабаритной космической конструкции «Фермы-3»

В космической технике геометрическое моделирование крупногабаритных конструкций часто связано с проблемой их доставки на орбиту, с перемещением внутри орбитальной станции, с работами космонавтов в скафандрах в условиях космического пространства и т.п.

Моделирование крупногабаритной космической конструкции (КГКК) «Ферма-3» проводилось по заказу НПО «Энергия». 3D модель «Фермы-3» показана на рис. 8.11 (смотри цветную вставку). Конструкция фермы состоит из четырех секций. Каждая секция имеет две поперечные диафрагмы (1), четыре И-образные панели (2), четыре диагонали (3) на рисунке показаны красным цветом.

На борт транспортного корабля «Ферма-3» поступала в сложенном положении, как показано в нижней части рис. 8.11. Развертывание каждой секции производилось специальными инструментами (4). Инструментами 4 каждая секция поднималась в вертикальное положение, диагонали 3 фиксировались винтами 5.

Процесс моделирования на этапах проектирования конструкции, проверки возможности ее установки на модуле «Квант-1» и развертывания включал следующие операции:

1. Описание геометрической модели конструкции «Фермы-3».

2. Проверка правильности ее развертывания до момента физической реализации.

3. Описание геометрических моделей платформы на модуле «Квант-1», а также моделей ферм «Рапана» и «Софора» для создания интерактивных электронных технических руководств (ИЭТР) [10, 22] космонавтам по подготовке места для монтажа «Фермы-3» и ее раскрытия.

Первые два пункта подтверждают сложившуюся за рубежом практику математического моделирования конструкции до момента ее физической реализации, поскольку предварительное моделирование экономит время и деньги, в чем неоднократно можно было убедиться в процессе работы.

Третий пункт детально показывал место и способ закрепления «Фермы-3», для чего необходимо было демонтировать ферму «Рапана», закрепить ее между модулями «Базовым» и «Квант-1», установить «Ферму-3» на место фермы «Рапана» и приступить к раскрытию «Фермы-3».

8.4 Моделирование процесса установки и развертывания космического рефлектора

Работа проводилась в 1999 г. по заказу НПО ЭГС для космического эксперимента по развертыванию рефлектора крупногабаритной космической антенны с апертурой в 6 м. Был создан компьютерный фильм (рис. 8.12 смотри цветную вставку), показывающий последовательность операций по выносу антенны в собранном состоянии в космическое пространство (а), закрепление ее на ферме «Софора» (б), развертывание рефлектора антенны и отталкивание ее от фермы в космическое пространство (в). В фильме специально акцентировалось внимание на процессе развертывания рефлектора. Развертывание проводилось электрическими двигателями, вмонтированными в конструкцию антенны. Операция развертывания в космосе крупногабаритной антенны на орбитальной станции МИР проходила в два этапа 23 и 28 июля 1999 года из-за ошибки при подключении напряжения питания двигателей.

8.5 Моделирование процесса раскрытия Большого Космического Ретранслятора

Работа выполнялась по заказу ООО ЭГС. Рассмотрен метод организации математических компьютерных моделей крупногабаритных динамических конструкций. Сформулированы проблемы и показана возможность создания обобщенной взаимосвязанной модели (ОВМ), включающей для начальных этапов их жизненного цикла (ЖЦ) структурную объемную геометрическую, функциональную, конструкторскую и расчетные модели. На примере проектирования Большого космического рефлектора (БКР) показан принцип формирования ОВМ [19, 23, 64].

8.5.1 Математические модели на этапах жизненного цикла крупногабаритных динамических конструкций

В CALS-технологиях [34, 48] математические модели конструкций используются на всех этапах их жизненного цикла (ЖЦ) и являются основой для создания обобщенной взаимосвязанной модели (OBM), отражающей все аспекты конструкции, которые могут потребоваться на каждом из этапов.

На примере моделирования крупногабаритных динамических конструкций рассмотрим основные проблемы, возникающие при создании OBM, и, в частности, структуры данных OBM. Заметим, что CALS-стандарты не регламентируют структуры данных OBM. Для динамических конструкций, в отличие от статических, характерно изменение их конфигурации в процессе эксплуатации, а это на этапе структурного проектирования приводит к необходимости создания параметрических геометрических моделей, что существенно усложняет процесс формирования OBM.

На рис. 8.14 приведен перечень некоторых начальных этапов ЖЦ крупногабаритных динамических конструкций (КДК) и указан набор моделей, используемых на этих этапах. Среди этапов выделены: разработка технического задания (ТЗ) на КДК, структурное и конструкторское проектирование, функциональное моделирование, проведение прочностных, тепловых и динамических расчетов, разработка документации для технологической подготовки производства и изготовления КДК.

На этапе структурного проектирования формируется структурная объемная геометрическая модель (СОГМ), содержащая параметрическое описание геометрических размеров отдельных элементов и всей конструкции в целом, а также описание взаимосвязей между элементами, достаточных для моделирования особенностей функционирования КДК.

На этапе конструкторского проектирования создается объемная геометрическая конструкторская модель (ОГМ) на основе предварительно разработанных 3D моделей отдельных элементов конструкции. ОГМ отличается от СОГМ фиксированными значениями всех параметров и большей степенью детализации описаний отдельных элементов, достаточных для изготовления конструкции с заданной точностью. На этом этапе формируется конструктор-

ская документация в виде 2D и 3D моделей, необходимых на последующих этапах ЖЦ.

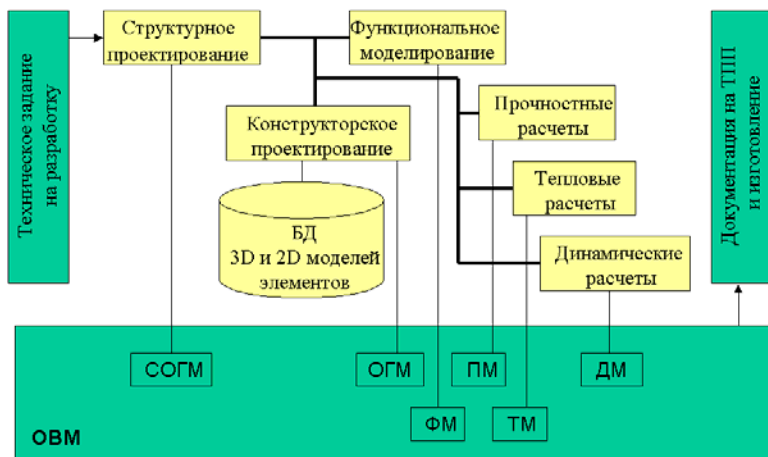


Рисунок 8.14 Основные типы математических моделей на отдельных этапах жизненного цикла изделий

На этапах функционального моделирования, проведения прочностных, тепловых и динамических расчетов создаются соответственно функциональная (ФМ), прочностная (ПМ), тепловая (ТМ), динамическая (ДМ) модели.

ФМ описывает алгоритмы функционирования динамических конструкций, а также при необходимости и алгоритмы функционирования датчиков, исполнительных механизмов, электрических приборов, расположенных на элементах конструкции. Для создания ФМ используются универсальные либо специализированные языки описания моделей и протекающих в них процессов.

Связь между этапами по информации на основе обобщенной взаимосвязанной модели (ОВМ) на рис. 8.13 показана толстыми линиями. Отсутствие направления передачи информации на линиях показывает итерационность процесса проектирования, т.е. возможность возврата информации с последующих этапов на предыдущие. Следует заметить, что описания ОГМ, ПМ, ТМ, ДМ и их форматы данных полностью зависят от используемых на этих этапах систем проектирования и систем проведения соответствующих расчетов.

Таким образом, обобщенная взаимосвязанная модель (ОВМ) конструкции для всех этапов ее жизненного цикла представляет собой набор математических моделей, непосредственно связанных унифицированными структурами данных для передачи информации между моделями. Далее на примере проектирования Большого космического рефлектора (БКР) покажем принцип формирования структурной объемной геометрической (СОГМ), функциональной (ФМ) и обобщенной (ОВМ) моделей, а также способы их программной реализации.

8.5.2 Особенности механизма развертывания Большого Космического Рефлектора

Моделируемая часть Большого Космического Рефлектора (БКР) представляет собой достаточно сложный механизм автоматического развертывания на орбите специального сетеполотна, отражающего направленные на него радиосигналы к приемнику, расположенному в заданных координатах на местности. Основными узлами этого механизма являются (см. рис. 8.15 смотри цветную вставку): центральный узел (CEI), система радиальных лепестков (RAR), силовое кольцо (RIA), отражающая поверхность (RSM) и система жесткости (STS). Система радиальных лепестков, удерживающая отражающую поверхность, с одной стороны закрепляется на центральном узле, с другой (внешние концы лепестков) закрепляются за силовое кольцо, выполненное в виде пантографа. В сложенном состоянии лепестки закручены вокруг центрального узла, внутренний диаметр силового кольца принимает минимальный размер (диагонали пантографа вытянуты вдоль центрального узла). Отражающая поверхность удерживается в сложенном состоянии специальными консолями, закрепленными на силовом кольце. При этом весь механизм развертывания принимает цилиндрическую форму.

Процесс развертывания включает следующие основные этапы: самопроизвольное развертывание за счет упругих сил деформированных лепестков; срабатывание электрических двигателей, поворачивающих диагонали пантографа и, тем самым, увеличивающих диаметр силового кольца; срабатывание системы жесткости и отключение двигателей.

8.5.3 Структура обобщенной взаимосвязанной модели

На рис. 8.16 показана структура обобщенной взаимосвязанной модели (ОВМ), содержащая различные модели основных узлов механизма разворачивания БКР. Модели ФМ, СОГМ, ОГМ, ПМ, ТМ, ДМ представлены отдельными слоями для каждого узла БКР. Основные параметры на входах и выходах функциональных моделей узлов указаны на связях между узлами. При этом приняты следующие обозначения:

r_0 – радиус центрального узла CEI,

h_0 – высота CEI,

s_0 – длина стойки CEI,

T_0 – точка крепления сетеполотна на CEI,

L_1 – длина лепестка,

Φ – угол поворота силового кольца,

$\{s_{ij}\}$ – множество длин стоек на лепестках,

R – радиус силового кольца,

$\{TR\}$ – множество точек крепления сетеполотна на стойках лепестков,

L_2 – длина рычага силового кольца,

H – высота силового кольца,

$\{LC_{ij}\}$ – множество длин консолей,

$\{TC\}$ – множество точек крепления сетеполотна на консолях,

$\{T\}$ – множество точек провисания сетеполотна.

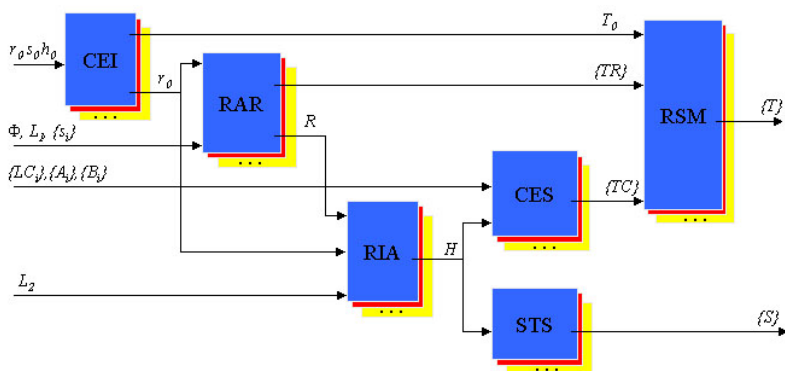


Рисунок 8.16 Структура обобщенной взаимосвязанной модели БКР

Визуализация общего вида СОГМ БКР представлена на рис.8.17 (смотри цветную вставку).

По результатам проведенных исследований механизма развертывания БКР разработан опытный образец (рис. 8.18 смотри цветную вставку).

8.6 Моделирование эргономических свойств пульта безопасности атомного реактора

Работа выполнена по заказу ФГУП «АтомЭнергоПроект» (2000 г.). Использование средств виртуальной реальности (ВР) открывает новые возможности по сокращению времени создания конечного продукта и трудозатрат на его производство, повышению конкурентоспособности, надежности и т.п. К таким возможностям, например, относятся:

1. Создание на основе функциональных и 3D моделей обобщенной модели разрабатываемого продукта для всех этапов его жизненного цикла.

2. Визуализация в реальном времени состояний объектов и процессов их функционирования на виртуальных моделях.

3. Компьютерная визуализация всех технологических процессов, включая предпроектные исследования, поиск заказчиков, представление проекта на тендер, сборка, продажа готовой продукции, эксплуатация, утилизация. Использование виртуальных моделей позволяет наблюдать технологические процессы не только последовательно по времени, как это делается в компьютерных фильмах, но и в пространстве.

4. Новые методы документирования, в которых первичным является электронная 3D модель конструкций, вторичным – чертежная документация на конструкцию, построенная в виде проекций.

Показательным примером использования новых методов описания технологических процессов сборки являются инструкции по сборке товаров, продаваемых сетью магазинов фирмы ИКЕА, основанных только на последовательности изобра-

жений 3D моделей конструкций, без каких-либо текстовых описаний.

Далее покажем возможности использования средств виртуальной реальности на примере моделирования и эргономического анализа комплекса технических средств оперативно диспетчерского управления (ОДУ) [20, 21].

8.6.1 Моделирование и эргономический анализ средств ОДУ

Комплекс технических средств ОДУ состоит из автоматизированных рабочих станций с одним или двумя дисплеями и панелей со средствами контроля и управления. Для создания 3D модели комплекса можно использовать имеющиеся на российском рынке CAD-системы: AutoCAD, 3ds max, Inventor, SolidWorks и др. Однако для решения узкоспециализированной задачи эргономического анализа комплекса использование таких систем представляется нецелесообразным ввиду их громоздкости и проблематичности в достижении реального времени. Поэтому была разработана программная система с поддержкой стандартных форматов данных по входу (.x, .dxf, .bmp) и возможностью эргономического анализа 3D моделей в реальном времени.

Система создана с использованием базовых средств программирования: языка C++ и графической библиотеки OpenGL (Open Graphics Library). Для эргономического анализа используется описание 3D модели в виде треугольных конечных элементов, записанное в формате .x. Этот формат содержит описания вершин (их координаты и нормали), порядок их объединения в треугольники, описание материалов и ссылки на текстуры. Для каждого треугольника указывается материал и, при необходимости, идентификатор текстуры и координаты для наложения текстуры. Информация о содержимом панелей комплекса хранится в растровом формате .bmp и представлена в 3D модели в виде текстур. Кроме того, ту же информацию можно передать в систему в векторном формате .dxf, который обеспечивает более точные (но и более медленные) визуализацию и расчеты.

Задача эргономического анализа комплекса заключается в определении зоны видимости оператором содержимого панелей, вычислении необходимых эргономических характеристик и сравнении полученных результатов с эргономическими требованиями ГОСТа) [38]. Зона видимости L определяется как пересечение 3D модели и конуса C , вершина которого лежит у глаз оператора, вектор высоты коллинеарен нормальной линии взора оператора, а угол между образующими совпадает с оптимальным углом обзора при повороте глаз оператора и равен 30° .

8.6.2 Структура системы

Структура системы включает блок хранения модели (БХМ), блок управления (БУ), блок визуализации (БВ) и блок эргономики (БЭ) (рис. 8.19).

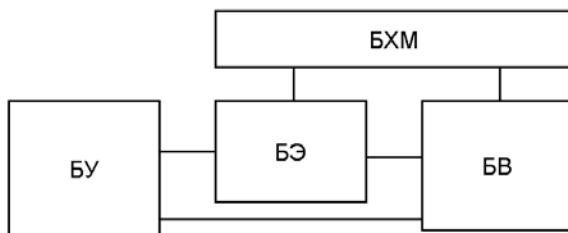


Рисунок 8.19 Структура системы

БХМ загружает 3D модель комплекса из внешнего файла формата .x, преобразует ее во внутреннее представление системы и хранит в оперативной памяти. Внутреннее представление организовано программно с помощью классов на языке C++. Вместе с 3D моделью БХМ загружает необходимые текстуры и файлы .dxf с векторным описанием содержимого панелей.

БУ позволяет пользователю выбирать различные режимы работы системы и перемещать оператора и камеру наблюдения в нужную точку пространства.

БВ отображает на дисплее проекции 3D моделей комплекса и зоны видимости на картинную плоскость с учетом установленных в БУ параметров. Для этого используется графическая библиотека

OpenGL, которая содержит необходимые функции отображения графических объектов с учетом материалов и текстур. Исходный текст программы на OpenGL представляет собой последовательную запись этих команд и практически не требует преобразования внутреннего представления 3D модели.

БЭ вычисляет зону видимости и другие необходимые эргономические характеристики с учетом установленных в БУ параметров, сравнивает их с эргономическими требованиями ГОСТа и передает результаты в БВ для вывода на экран.

8.6.3 Управление процессом моделирования и эргономическим анализом

Работа с системой состоит из нескольких этапов:

- выбор режима просмотра 3D модели, вид на комплекс со стороны оператора (рис. 8.20 и 8.22) или вид на комплекс и оператора со стороны наблюдателя (рис. 8.21 и 8.22);

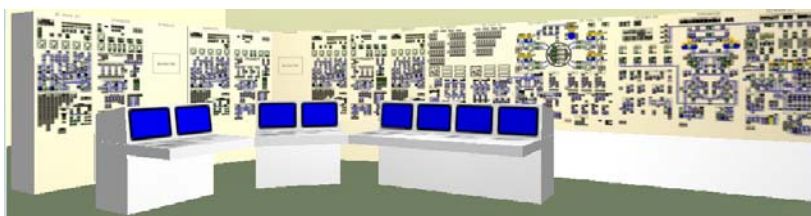


Рисунок 8.20 Общий вид комплекса технических средств ОДУ



Рисунок 8.21 Вид на комплекс и оператора со стороны наблюдателя

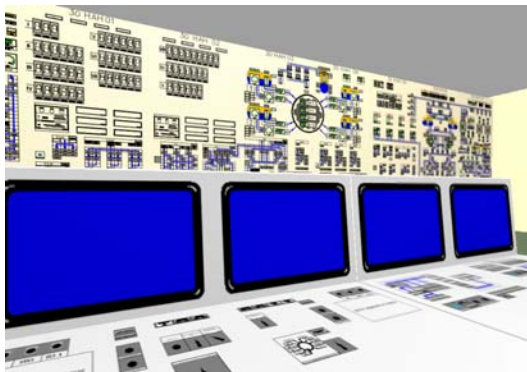


Рисунок 8.22 Вид на комплекс со стороны оператора

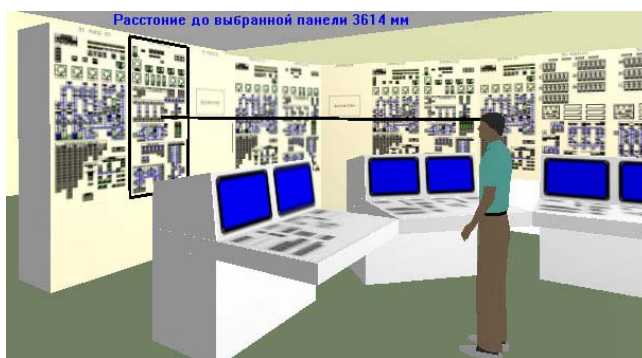


Рисунок 8.23 Пример выбора панели и определения фактических эргономических характеристик ее расположения по отношению к оператору

- размещение оператора в требуемой точке пространства (например, в предполагаемом рабочем месте) и выбор направления его взгляда;
- выбор положения оператора сидя или стоя;
- размещение камеры наблюдения в требуемой точке пространства;
- определение и подсветка зоны видимости оператором содержимого панелей;
- выбор панели или отдельных ее элементов (приборов, надписей на панелях, органов управления и т.п.) и определение фактических эргономических характеристик их расположения по от-

ношению к оператору. Информация об эргономических характеристиках выделенного элемента (размеры, расстояние до оператора, угол обзора и т.п.) выводится в верхней части экрана (рис. 8.22);

- сравнение полученных результатов с предъявляемыми требованиями;
- сохранение и/или распечатка результатов.

Система позволяет создавать, редактировать и проигрывать произвольный сценарий из перечисленных выше действий, который можно использовать, например, для демонстрации технических средств заказчику или обучения операторов.

Заключение

За рубежом средства виртуальной реальности начали использоваться в промышленности с 80-х годов прошлого столетия применительно к тренажерам, рекламной деятельности, анализу чрезвычайных ситуаций и т.п. Использование этих средств позволяет сократить время создания конечного продукта и трудозатраты на его производство, повысить конкурентоспособность и надежность. Российская промышленность еще ожидает своего пика использования средств виртуальной реальности, пока наибольшую активность в этом направлении проявляют только телевизионные компании в передачах о погоде и анализе чрезвычайных происшествий.



Рисунок 8.1 Объемная геометрическая модель орбитальной станции «Мир»

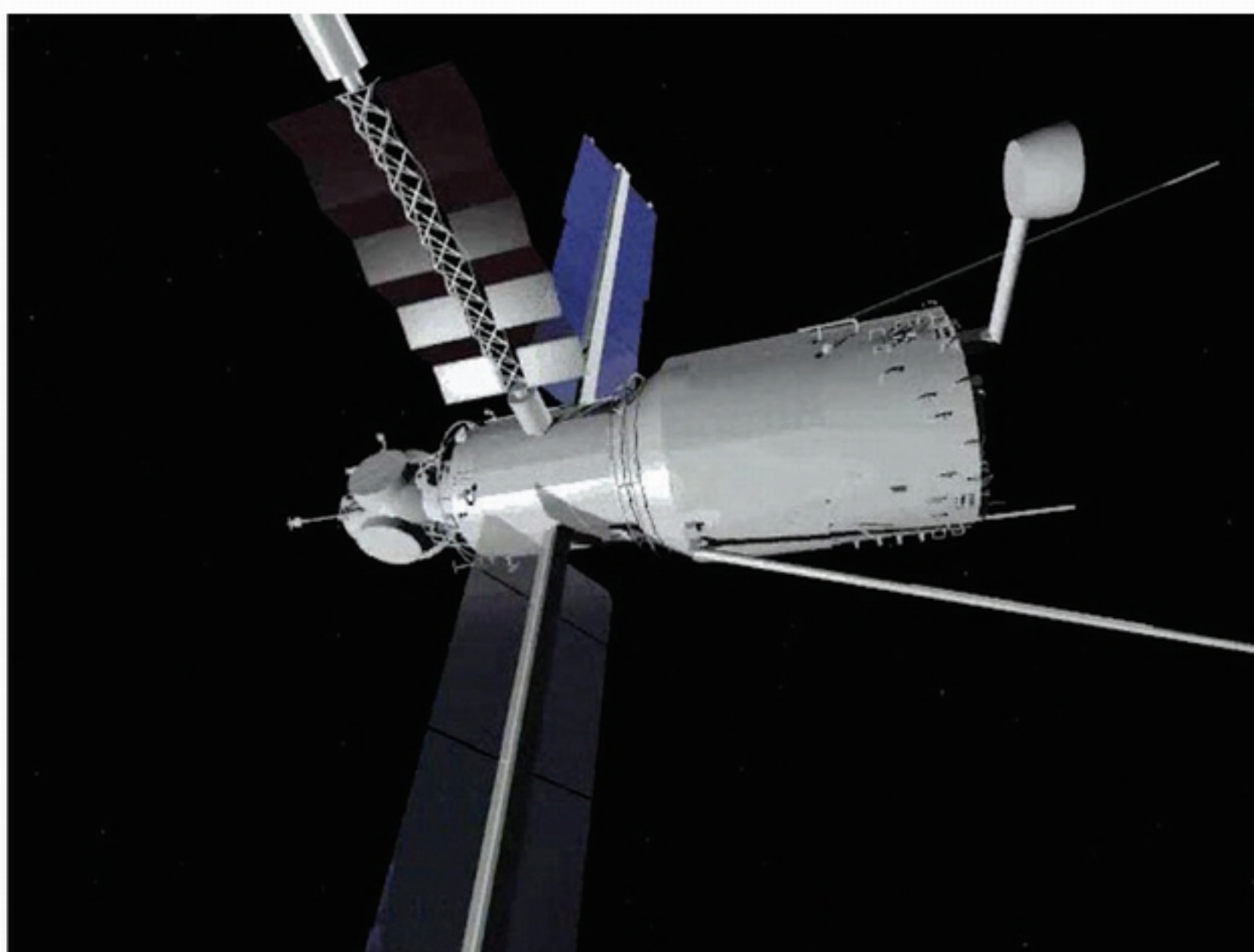


Рисунок 8.2 3D модель «Базового» модуля

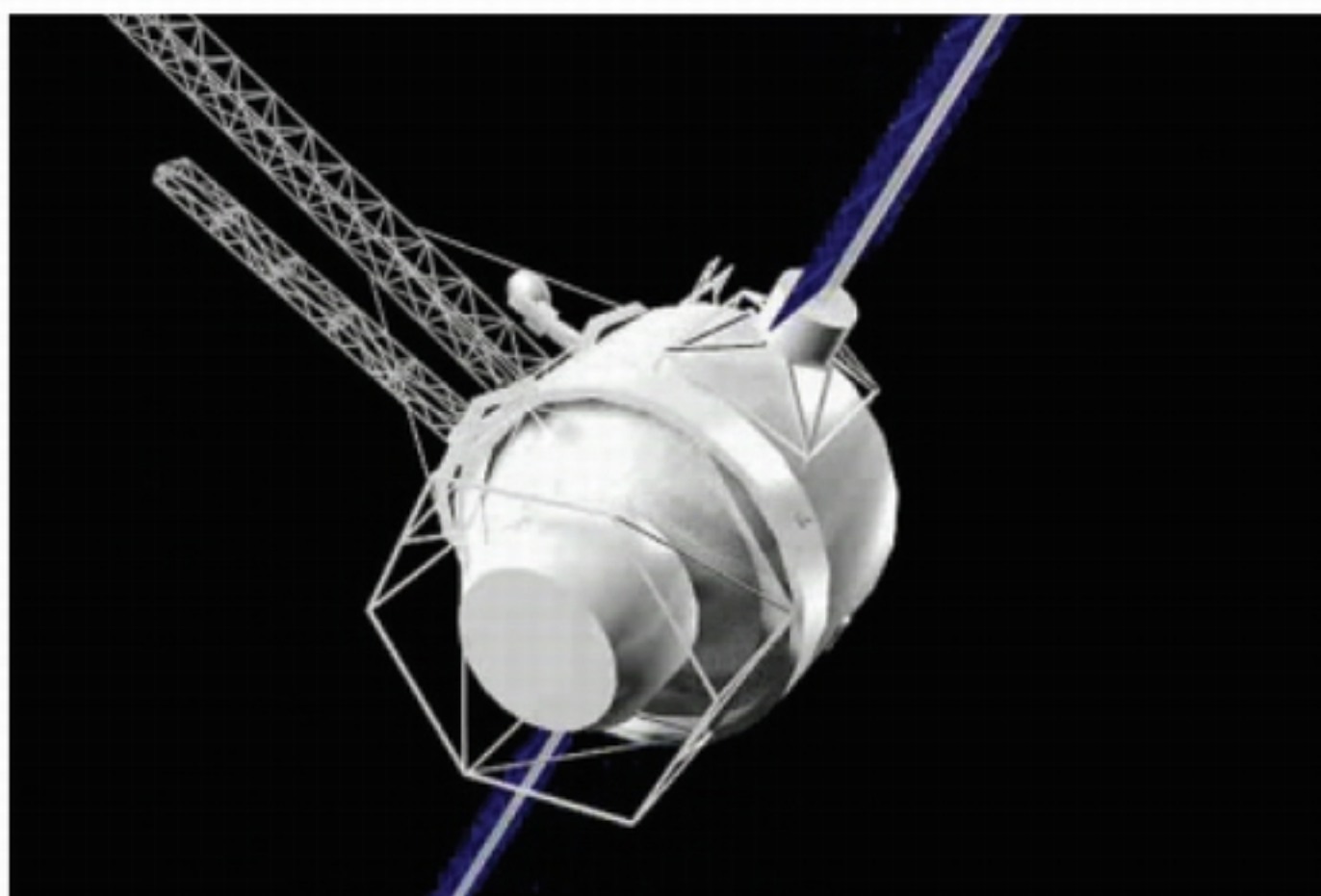


Рисунок 8.3 3D модель модуля «Квант-1»

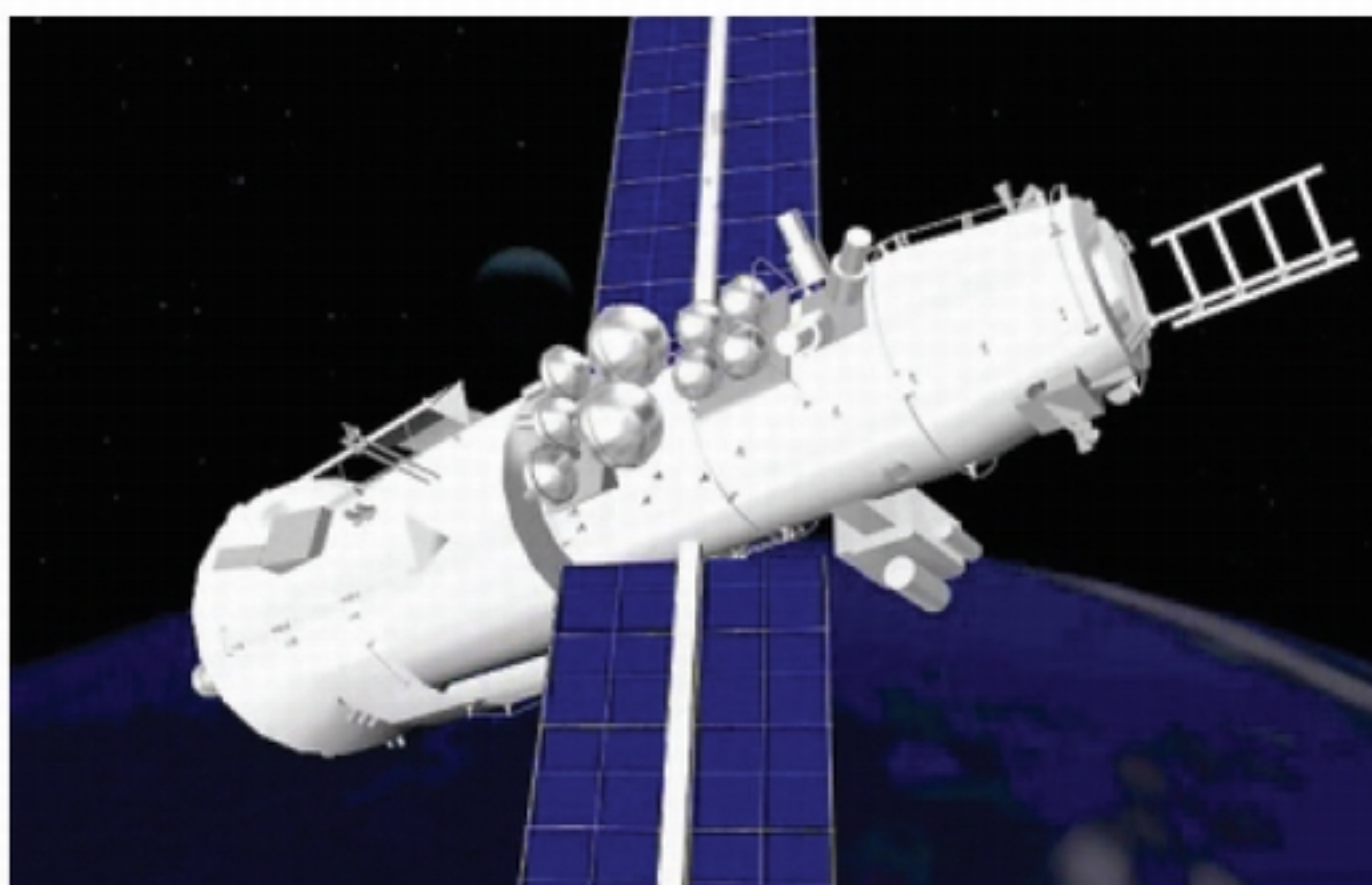


Рисунок 8.4 3D модель модуля «Квант-2»

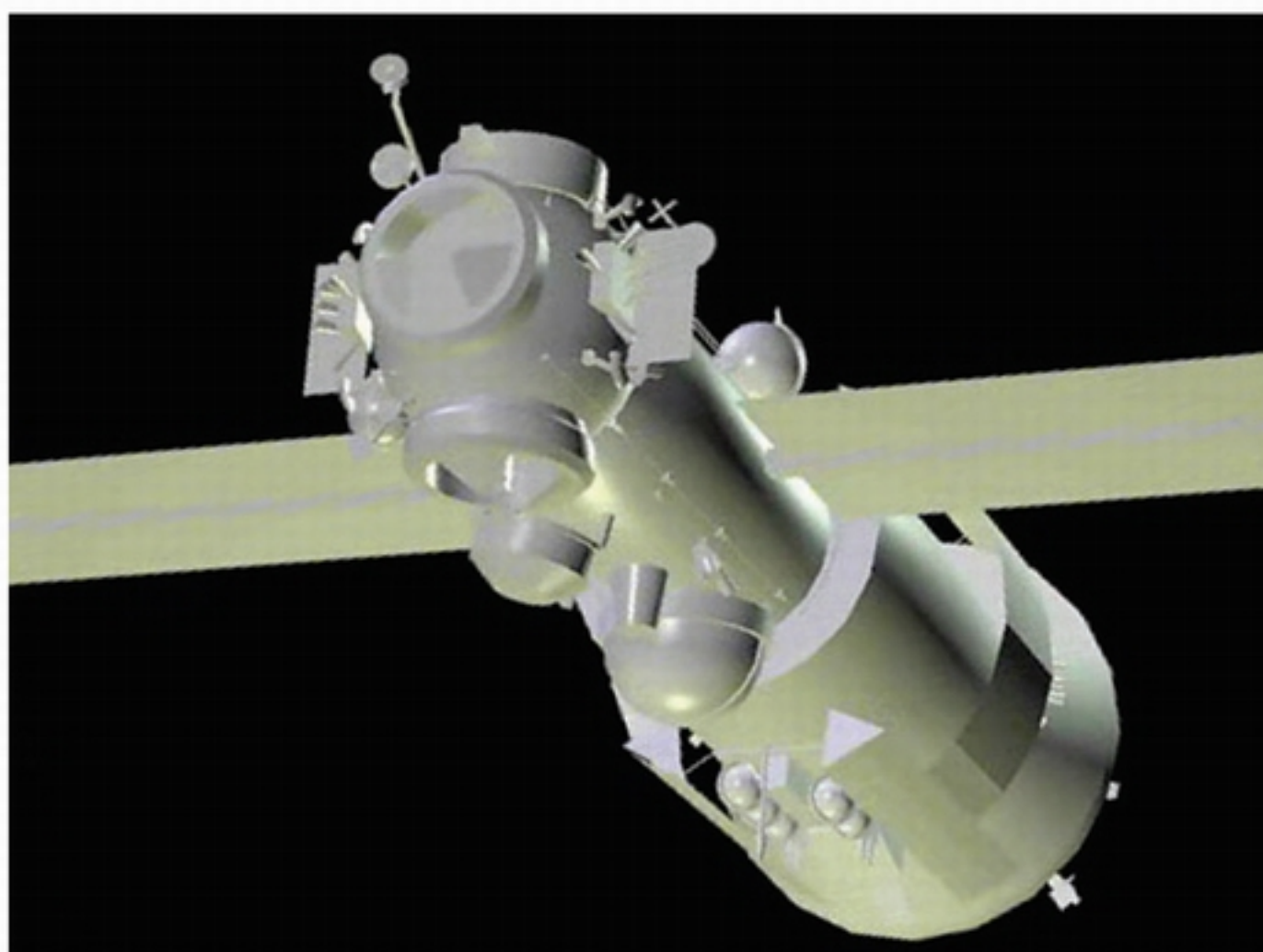


Рисунок 8.4 3D модель модуля «Квант-2»

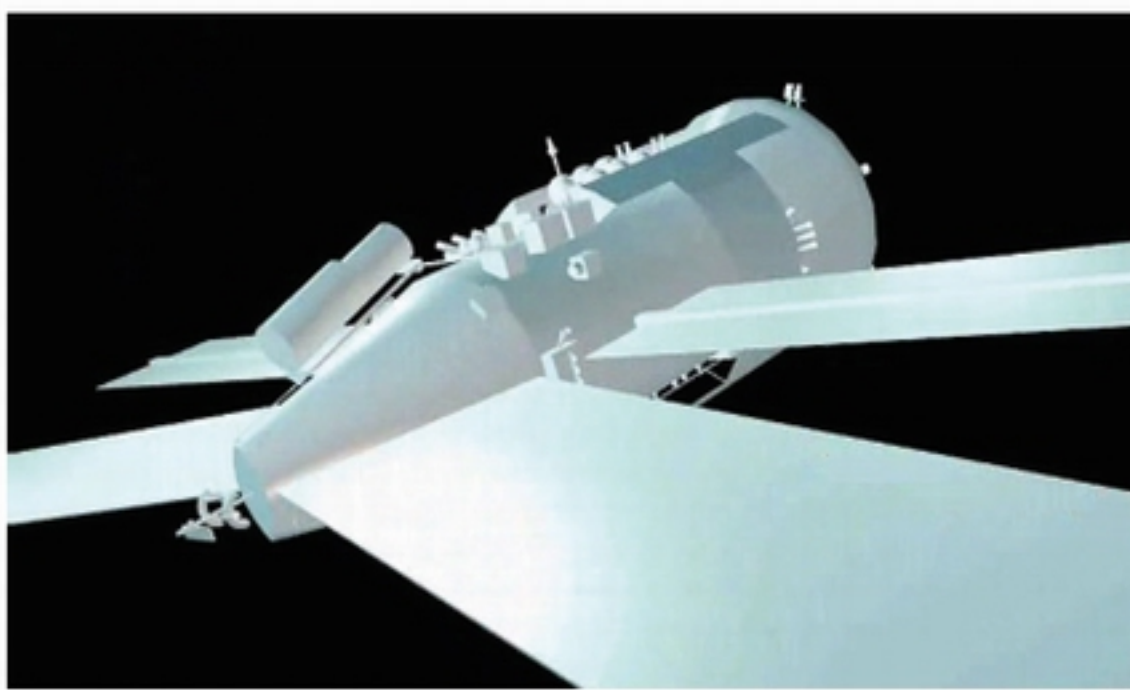


Рисунок 8.6 3D модель модуля «Спектр»

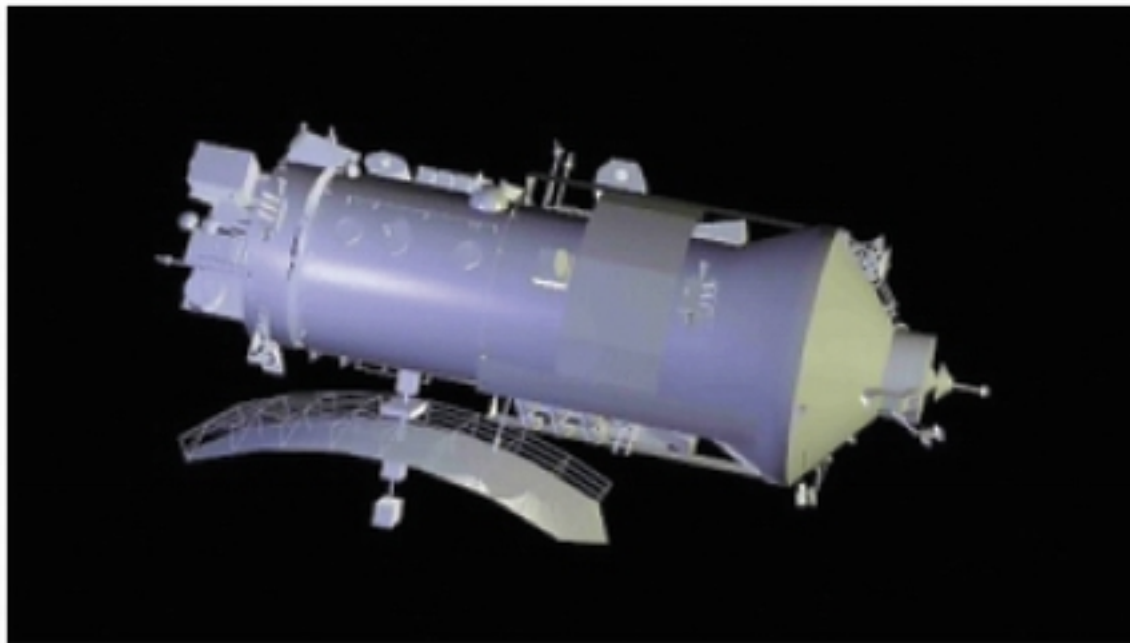


Рисунок 8.7 3D модель модуля «Природа»



Рисунок 8.8. 3D модель модуля «Прогресс»

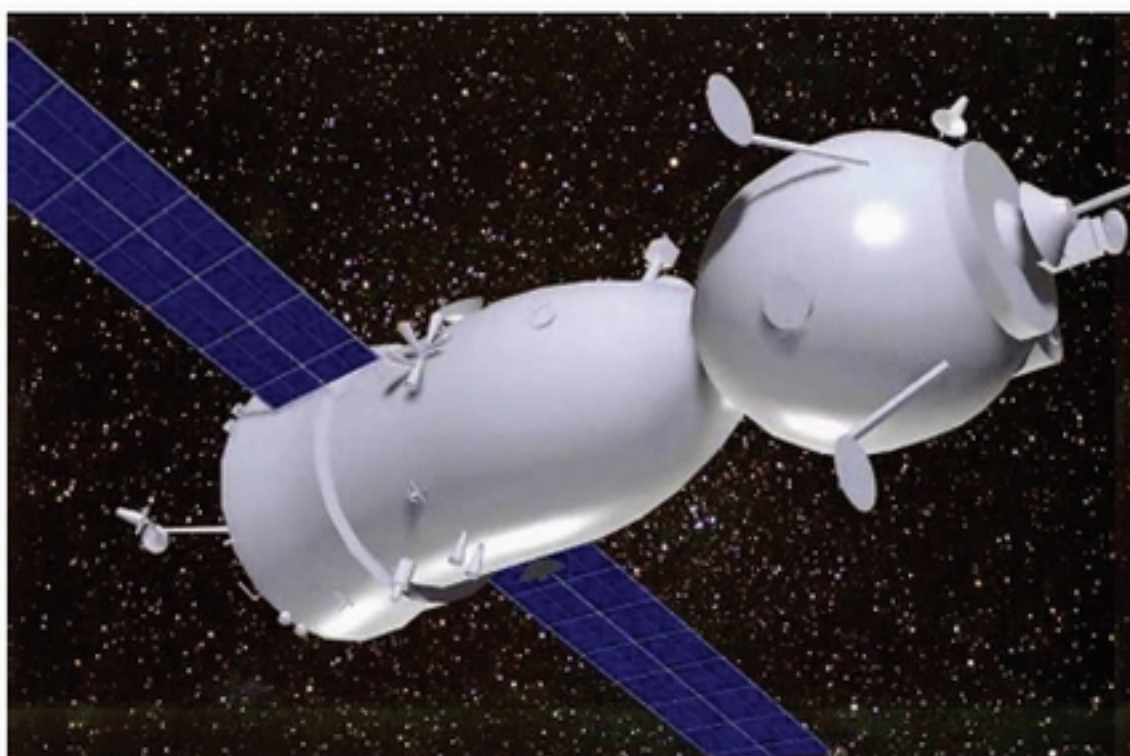


Рисунок 8.9 3D модель модуля «Союз»

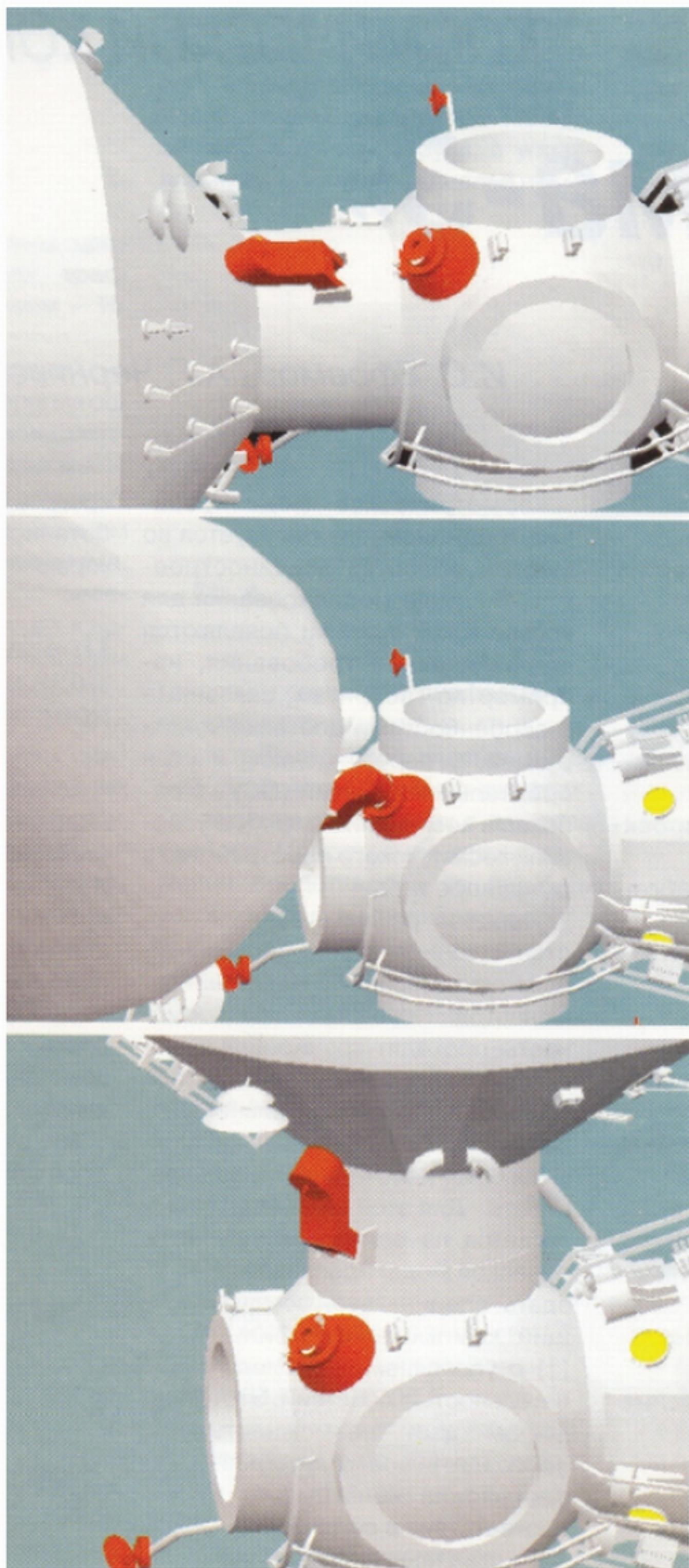


Рисунок 8.10 Основные этапы перестыковки

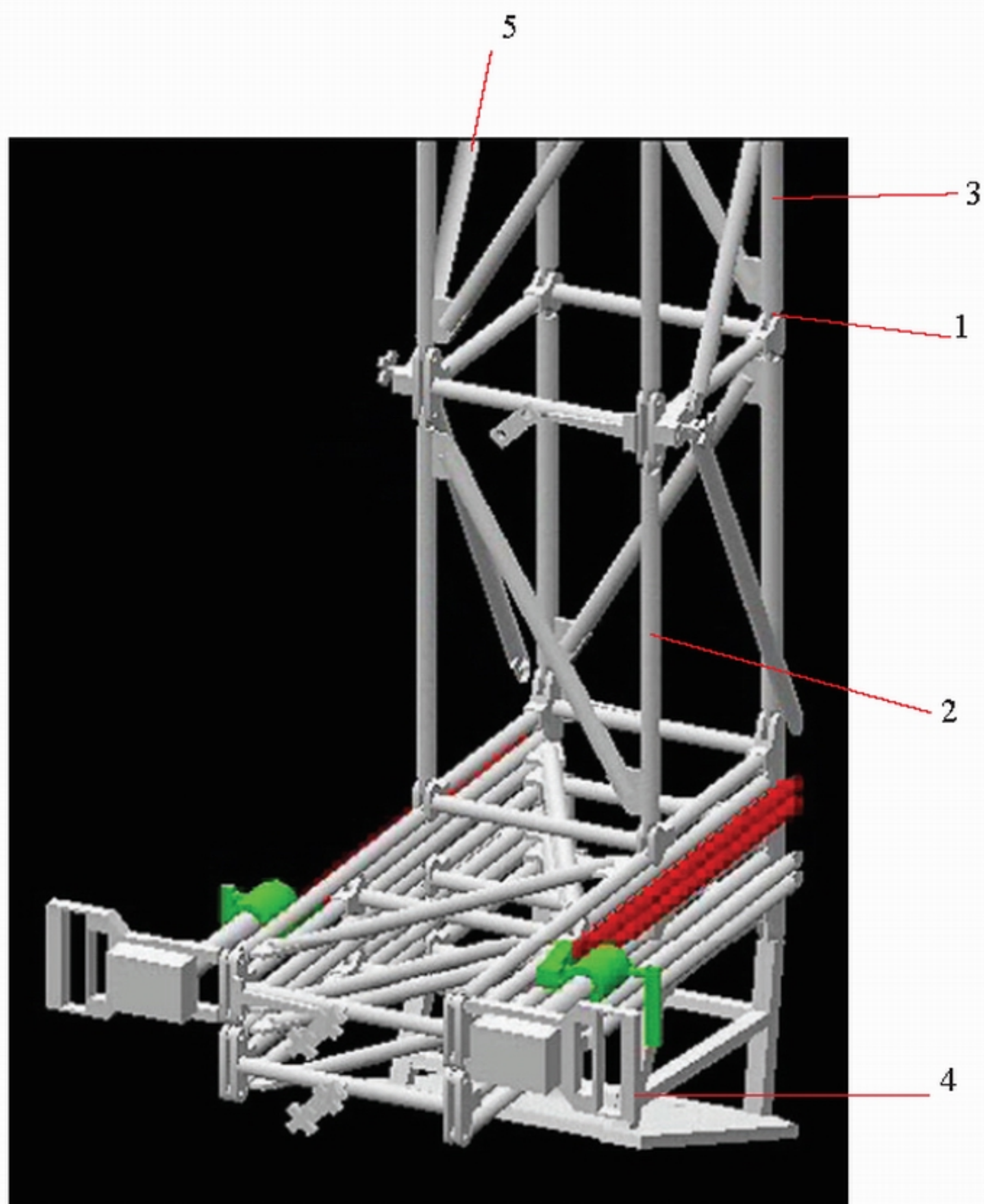


Рисунок 8.11 3D модель «Фермы-3»

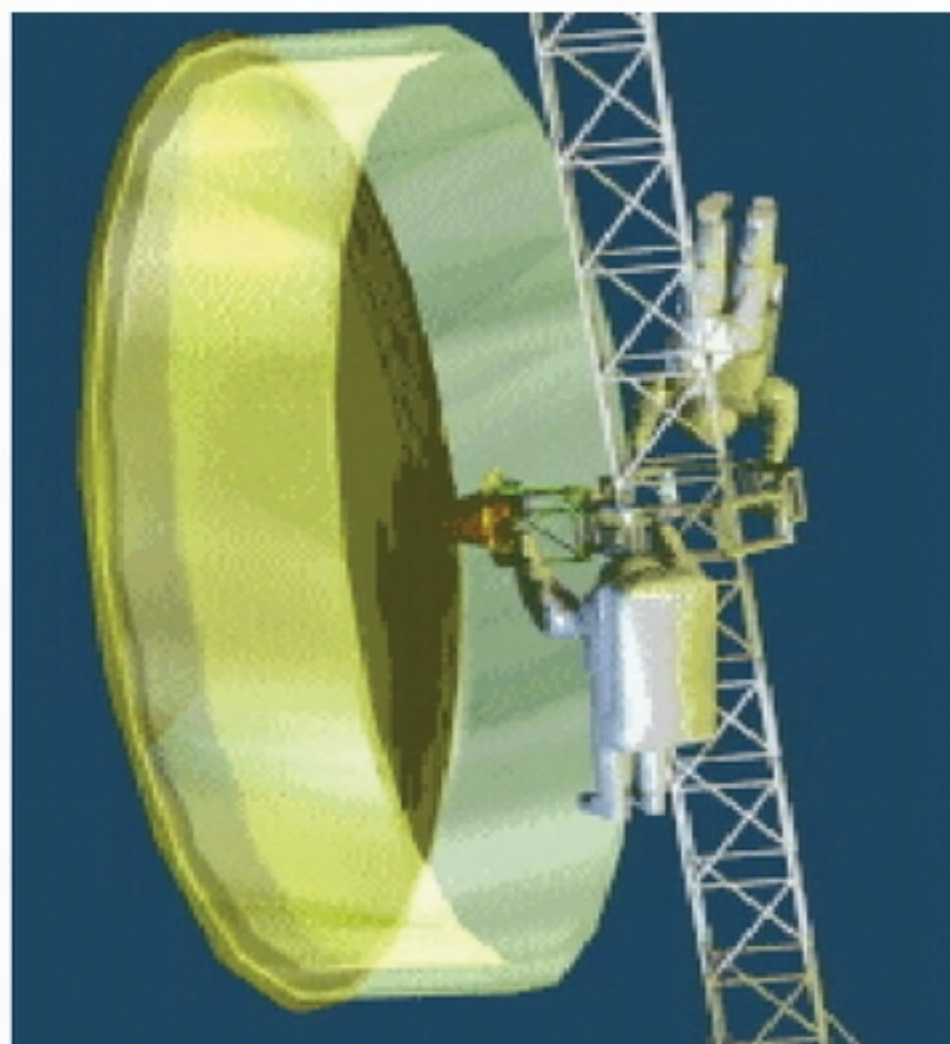
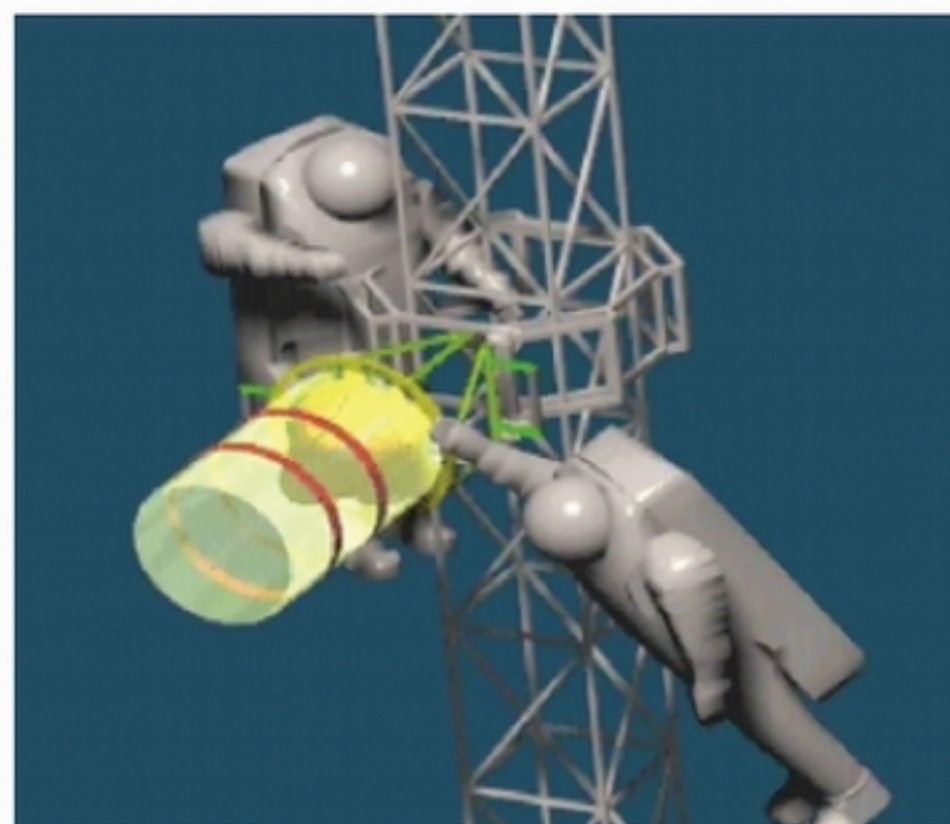


Рисунок 8.12 Кадры компьютерного фильма по установке крупногабаритной антенны.

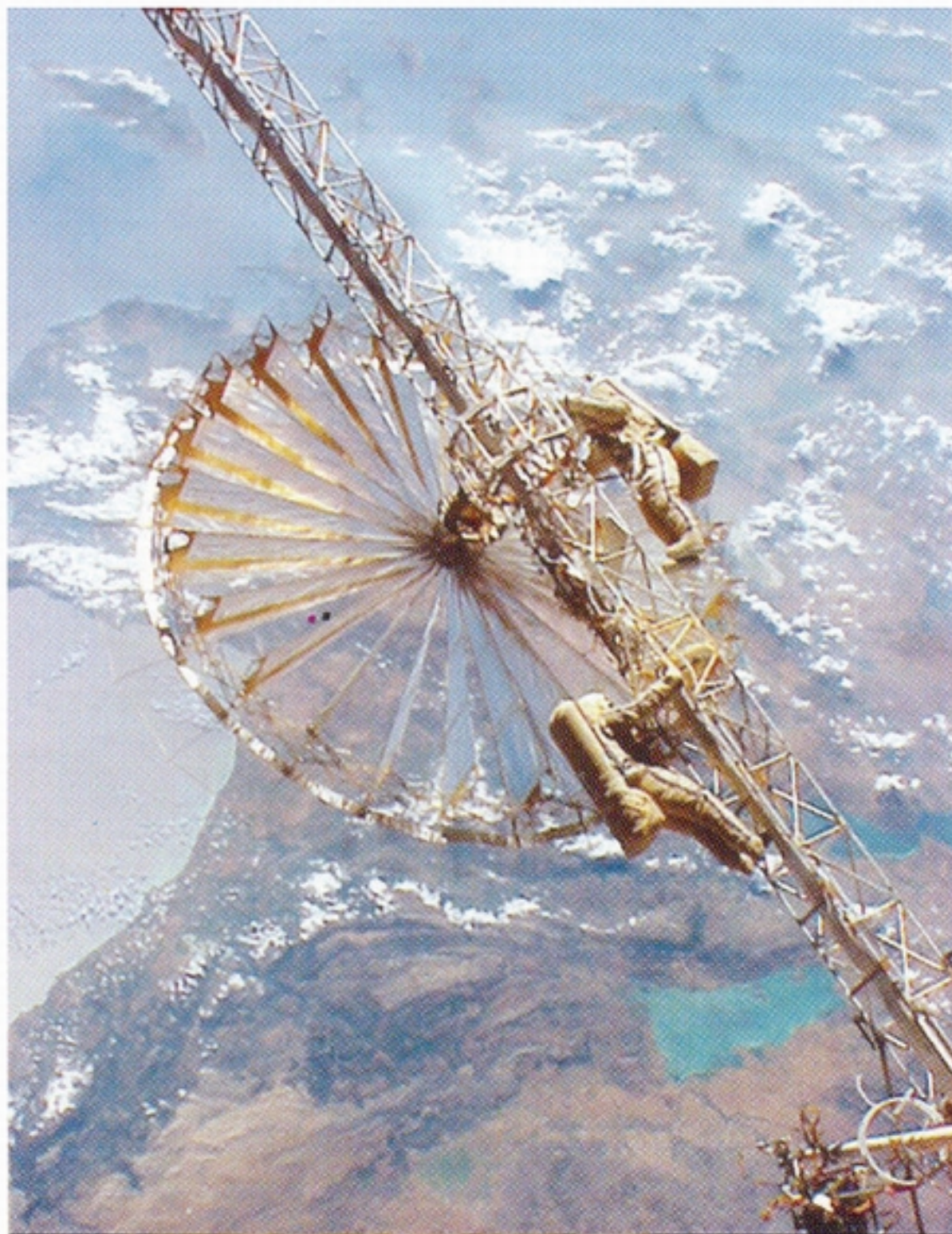


Рисунок 8.13 Развертывание рефлектора крупногабаритной антенны

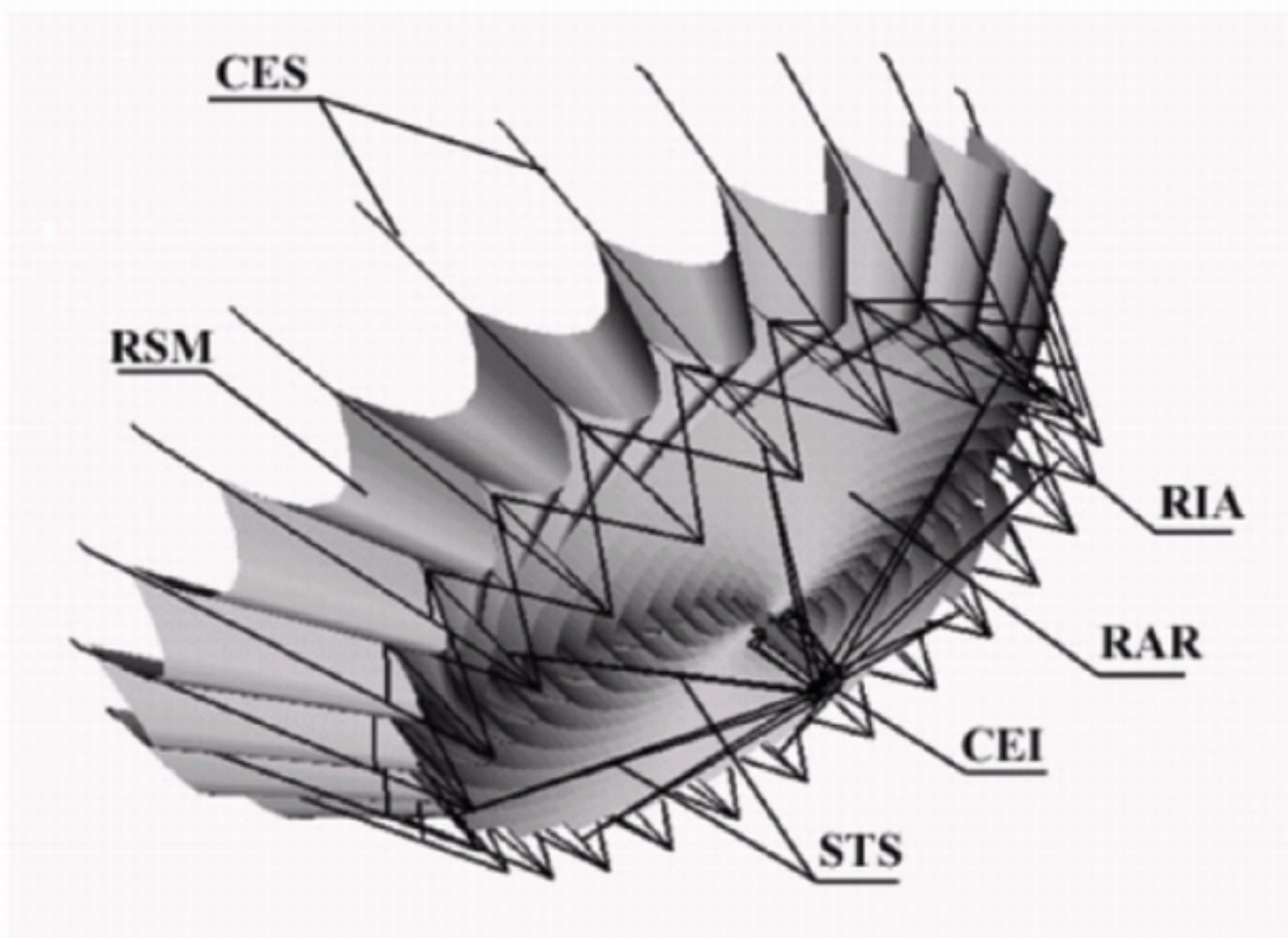


Рисунок 8.15 Основные узлы БКР

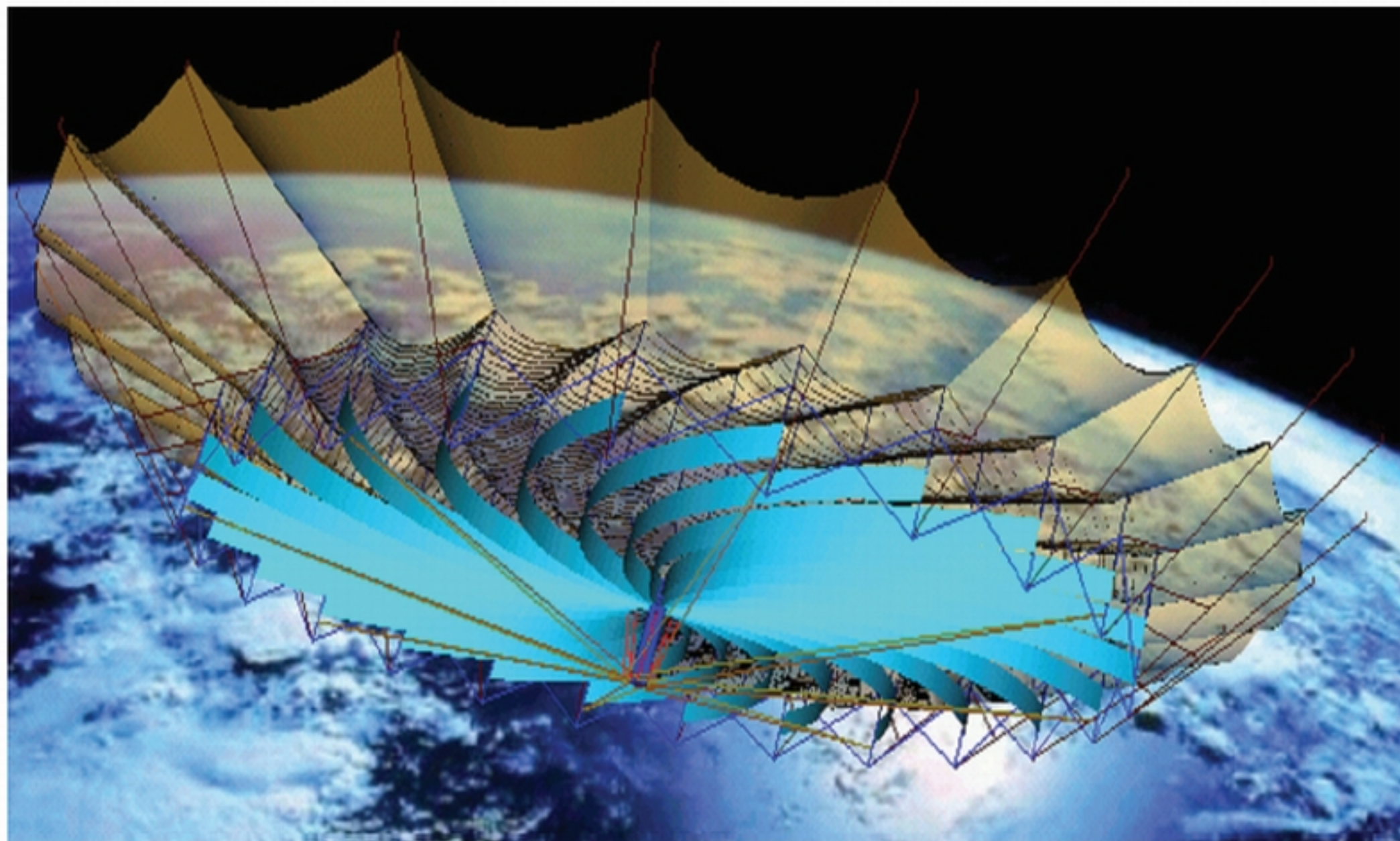


Рисунок 8.17 Моделирование процесса раскрытия БКР

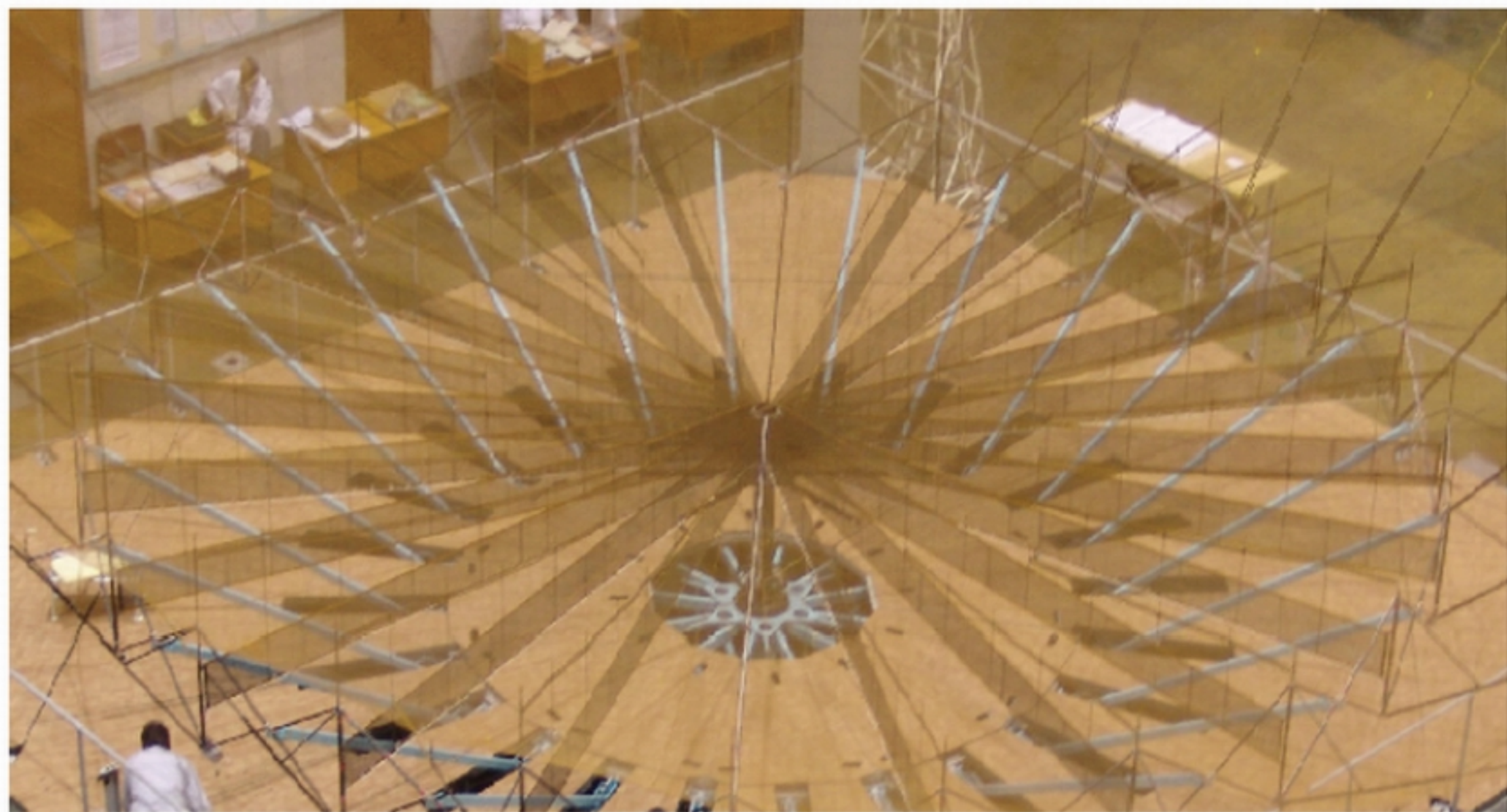


Рисунок 8.18 Раскрытие физической модели БКР

ЛИТЕРАТУРА

1. Арсланова В.М., Курицын В.В., Шурупов А.А., Томаш И., Фанта М. Комплекс ПО моделирования 3-х мерных объектов. Моделирование, идентификация и автоматизация проектирования производственных систем. Сб. трудов ИПУ, М., 1990. С. 32–40.
2. Артамонов Е.И. Варианты структурной организации программно-технического комплекса для решения задачи поиска кратчайшего пути. Труды 9-й Международной конференции CAD/CAM/PDM – 2009. – М.: Институт проблем управления РАН им. В.А. Трапезникова, 2009. Стр. 48–53.
3. Артамонов Е.И. Компьютерные модели изделий РЭА на этапах их жизненного цикла. Материалы 8-й международной конференции CAD/CAM/PDM – 2008. М.: Институт проблем управления РАН. – 2008.
4. Артамонов Е.И. Особенности синтеза архитектуры и классификация интерактивных систем. II Международная конференция «Идентификация систем и задачи управления» (SICPO 03) Москва, 29–31 января 2003 г., 23 с.
5. Артамонов Е.И. Принципы построения интерактивных систем проектирования // М.: Вопросы кибернетики, 1980.
6. Артамонов Е.И. Структурное проектирование систем // Информационные технологии в проектировании и производстве. 2008. №2. С. 3–11.
7. Артамонов Е.И. Цифровое времязадающее устройство. Приборы и техника эксперимента, № 3, Москва, 1968 г.
8. Артамонов Е.И., Балабанов А.В., Ромакин В.А. Операции на виртуальных моделях объектов машиностроения // Материалы 36-й междунар. конф. и дискуссионного научного клуба «Информационные технологии в науке, социологии, экономике и бизнесе (IT+SE'09)». – Открытое образование. – 2009. С. 70–72.
9. Артамонов Е.И., Балабанов А.В., Ромакин В.А. Технология создания специализированных систем на основе средств виртуальной реальности // 9-я международная конференция "Системы проектирования, технологической подготовки производства и управления этапами жизненного цикла про-

мышленного продукта" (CAD/CAM/PDM – 2009), М.:ИПУ РАН, 2009, С. 19-24.

10. Артамонов Е.И., Балабанов А.В., Ромакин В.А., Щегольков М.Ю. Принципы создания интерактивных технических руководств. // Материалы 35-й междунар. конф. «Информационные технологии в науке, образовании, телекоммуникации и бизнесе (IT+S&E*08)». – Открытое образование. – 2008. С. 97–98.
11. Артамонов Е.И., Высоких В.Ю. Варианты структурной организации систем проектирования РЭА. Материалы 6-й международной конференции CAD/CAM/PDM – 2006. М.: Институт проблем управления РАН. – 2006 г.
12. Артамонов Е.И., Высотин О.В., Разумовский А.И., Макаров А.М., Шурупов А.А. Объемное геометрическое моделирование орбитального комплекса «МИР». Автоматизация проектирования, №4, 1998, С. 3 – 7.
13. Артамонов Е.И., Дилигенский С.Н. Электронное времязадающее устройство. Приборы и техника эксперимента, № 6, Москва, 1963 г.
14. Артамонов Е.И., Загвоздкин И.А., Шурупов А.А., Щегольков М.Ю. Языки взаимодействия пользователя с ЭВМ в системе ГРАФИКА – 81. Институт проблем управления. Российская академия наук. М., 1993.
15. Артамонов Е.И., Круг Е.К., Кусовский Б.И., Белкин В.О. и др. Устройство для управления процессом смешения при получении многокомпонентной смеси. Авторское свидетельство № 224639.
16. Артамонов Е.И., Макаров В. Анализ и синтез архитектуры сложных программных систем. Приборы и системы. 7–2000 г. С. 22-29.
17. Артамонов Е.И., Ничипорович Т.А., Тенякшев А.М. Метод формирования обобщенных моделей построения систем. Материалы XXXIII международной конференции Информационные технологии в науке, образовании, телекоммуникации и бизнесе (IT+S&E*06), журнал «Открытое образование». – 2006г.

18. Артамонов Е.И., Преображенский Н.И. Автоматическое вычислительное устройство для тяговых испытаний тракторов. Доклады МИИСП т.5, вып. 3, 1969.
19. Артамонов Е.И., Разумовский А.И., Ромакин В.А., Чернявский А.Г., Чернявский А.А. Моделирование крупногабаритных динамических конструкций // Информационные технологии и вычислительные системы. – 2007. №2. – С. 25–30.
20. Артамонов Е.И., Ромакин В.А. Использование средств виртуальной реальности при проектировании и эксплуатации промышленных производств // Автоматизация в промышленности. – 2007. № 4. – С. 14–16.
21. Артамонов Е.И., Ромакин В.А. Моделирование и эргономический анализ пультов управления // Перспективы использования новых технологий и научно-технических решений в ракетно-космической и авиационной промышленности. Материалы международной конференции «AEROSPACE-2008». Под. ред. Артамонова Е.И. – М.: ИПУ РАН. – 2008 – С. 38–39.
22. Артамонов Е.И., Ромакин В.А., Балабанов А.В. Программные средства для создания электронных технических руководств // Материалы международной конференции «Перспективы использования новых технологий и научно-технических решений в ракетно-космической и авиационной промышленности». – М.: ИПУ РАН, 2008. – С. 40–41.
23. Артамонов Е.И., Ромакин В.А., Чернявский А.Г. Обобщенная модель большого космического рефлектора // Тез. VI междунар. конф. "CAD/CAM/PDM-2006". – М.: ИПУ РАН, 2006. – С. 114.
24. Артамонов Е.И., Сизова Л.Н. Автоматическая трассировка соединений (АТС). Свидетельство о государственной регистрации N2008613903 от 15 августа 2008г.
25. Артамонов Е.И., Хачумов В.М. Синтез структур специализированных средств машинной графики. МЦНТИ. Москва, 1991г. 145 с.
26. Артамонов Е.И., Шурупов А.А. Особенности разработки ПО подсистем 3-х мерной машинной графики. – Приборы и системы управления. М. 1986, №3, С. 21–23.

27. Баяковский Ю.М., Галактионов В.А. Графические протоколы (обзор). – Автометрия. – 1978. – № 5. – С. 3–11.
28. Борн Г. Форматы данных: Пер. с нем. – К.: Торгово-издательское бюро BHV, 1995 – 472 с.: ил. – ISBN 5-87685-023-3, АО «Санкт-Петербург оркестр», 1995.
29. Бурков В.Н., Новиков Д.А. Как управлять организациями. М.: Синтег, 2004. – 400 с., ил. (Серия «Управление организационными системами»).
30. Буч Г. и др. Язык UML. Руководство пользователя. Пер. с англ. – М.: ДМК, 2000. – 432 с.
31. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. 2-е изд.: Пер. с англ. – М.: Издательство Бином, СПб.: Невский диалект, 1999.
32. Бьерн Страуструп. Язык программирования C++. Третье издание. AT&T Labs. Мюррей-Хилл, Нью-Джерси. Перевод с английского. М.-СПб. 2000 г.
33. Васенов А.В., Скородумов С.В. Модель компактного интеллектуального производства на основе RPM-технологий. Материалы 1-ой Международной конференции и выставки CAD/CAM/PDM – 2001. М.: Институт проблем управления РАН. -2001 г. 9 стр.
34. Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. – М.: Финансы и статистика, 1998.
35. Гилой В. Интерактивная машинная графика. Структуры данных, алгоритмы, языки // М.: Мир, 1981.
36. Горелик А.Г. Пакет программ машинной графики для ЕС ЭВМ // М.: Машиностроение. – 1986.
37. Горелик А.Г., Буракова В.Я. Автоматизация геометрического моделирования с помощью языка “ФАПКФ” // Вестник машиностроения. – 1979. – № 10. – С. 66–69.
38. ГОСТ 23000–78 «Пульты управления. Общие эргономические требования».
39. Грозовский Г.И., Таллер С.П. Стандарты ЕСКД – нормативно-информационная основа построения автоматизированных систем проектирования и производства (САПР, CAD/CAM и др.). Материалы 8-й международной конференции

- CAD/CAM/PDM – 2008. М.: Институт проблем управления РАН. – 2008
40. Джермейн К. Программирование на IBM/360 // Пер. с англ. под ред. В.С.Штаркмана. – М.: Мир, 1973.
 41. Инструментарий ARIS: методы. М.: Весть-метатехнология. 2000.
 42. Карцев М. А. Арифметика цифровых машин. М: Наука, 1969. – 576 с.
 43. Комаров А. Инструмент для бизнеса. Формирование рынка информационных панелей. ITResearch. Аналитика российского рынка ИТ. – №4 – 2006.
 44. Круг Е.К., Артамонов Е.И., Дилигенский С.Н. Структурные схемы цифровых систем управления станцией смещения. Транспорт и хранение нефтепродуктов и углеводородного сырья. – № 4, – Москва – 1969.
 45. Кузнецов Н. А., Кульба В.В., Ковалевский С.С., Косяченко С.А. Методы анализа и синтеза модульных информационно-управляющих систем. М.: Физматлит, – 2002.
 46. Мамиконов А.Г., Цвиркун А.Д., Кульба В.В. Автоматизация проектирования АСУ. М.: Энергоиздат, 1981.
 47. Матюхин Н. Я. Некоторые вопросы применения электронных вычислительных машин при проектировании цифровой аппаратуры //В сб.: Вычислительная техника. Изд. МДНТП, 1965.
 48. Норенков И. П. CALS-стандарты. // Информационные технологии. Науч.-техн. Журн. – 2002. № 2 – С. 47-51.
 49. Ньюмен и Спрулл. Основы интерактивной машинной графики. Из-во «МИР». Москва 1975 г.
 50. Попова Г. Н., Алексеев С. Ю. Машиностроительное черчение: Справочник. – Л.: Машиностроение, Ленингр., 1986. – 447 с.
 51. Применение вычислительных машин для проектирования цифровых устройств // Под ред. Матюхина Н.Я. М.: Советское радио. – 1968.
 52. Соломенцев Ю.М. Информационно-вычислительные системы в машиностроении. CALS-технологии. – М.: Наука, 2003. – 292 с.

53. Таллер С.П. Новое в стандартах ЕСКД // Стандарты и качество. – 2007 г. – № 7.
54. Таллер С.П. Пичев С. Электронные стандарты ЕСКД//Стандарты и качество. – 2005 г. – № 9.
55. Уитсон Дж. 500 практических схем на ИС: Пер. с англ. – М.: Мир, 1992, 376 с.
56. Уэйкерли Дж. Ф. Проектирование цифровых устройств, том 1. Москва: Постмаркет, 2002. – 544 с.
57. Фаулер М., Скотт К. UML в кратком изложении. Применение стандартного языка объектного моделирования: Пер. с англ. – М.: Мир, 1999.
58. Фоли Дж., вэн Дэм А. Основы интерактивной машинной графики // М.: Мир. – 1985. Т. 1.
59. Форд Л.Р., Фалкерсон Д.Р. Поток в сетях. Перевод с англ. И.А. Вайнштейна. Издательство «МИР», Москва, 1966.
60. Фролов Г.Д., Кузнецов Э.И. Элементы информатики: Учеб. Пособие для пед. Ин-тов. – М.: Высш. Шк., 1989. – 304 с.
61. Херн Дональд, Бейкер М. Паулин. Компьютерная графика и стандарт OpenGL, 3-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс». 2005.
62. Черемных С.В., Семенов И.О., Ручкин В.С. Моделирование и анализ систем. IDEF-технологии: практикум. – М.: Финасы и статистика, 2006. – 192 с.
63. Черемных С.В., Семенов И.О., Ручкин В.С. Структурный анализ систем: IDEF-технологии. – М.: Финансы и статистика, 2003 – 208 с.
64. Чернявский А.Г., Чернявский А.А., Артамонов Е.И., Разумовский А.И., Ромакин В.А. Моделирование процесса развертывания Большого Космического Рефлектора (БКР) // Перспективы использования новых технологий и научно-технических решений в ракетно-космической и авиационной промышленности. Материалы международной конференции «AEROSPACE-2008». Под. ред. Артамонова Е.И. – М.: ИПУ РАН. – 2008 – С. 36–37.
65. Язык графического взаимодействия. Методические указания. САПР. 2-я ред. // М.: ВНИИНМАШ, 1983.
66. Analog and Mixed-Signal Modeling Using the VHDL-AMS Language. – New Orleans, 1999.

67. Artamonov E., Shurupov A. Efremov I., Petuchov V., Cherniavsky A. Modeling of the Large – Space Structures Deployment Process. EAST-WEST International Conference INFORMATION TECHNOLOGY IN DESIGN. EWITD, 96. M., 1996, p. 174-176.
68. Artamonov E., Shurupov A. Grafika-81-3D Programm Package of 3D Modeling. EAST-WEST International Conference INFORMATION TECHNOLOGY IN DESIGN. EWITD, 96. M., 1996, p. 323-332.
69. Artamonov E.I. Automation of digital device structure design. B-215, ACTA IMEKO, 1973, p. 561-570.
70. Gara A., Blumrich M.A., Chen D., Chin G.L.-T. and all. Overview of the Blue Gene/L system architecture// IBM Journal of Research and Development, 2005, Vol.49, № 2/3, P.7-19.
71. Geometric Product Models – An internordic joint project. First edition, KTH, Stockholm, October 1980.
72. Gibson M.L. The CASE Philosophy. BYTE, 1989.
73. ISD/DIN 7942, GKS, VERSION 7.2, ISO TC 97/SC 5/WG2, 1982.
74. Kochan D. Solid Freeform Manufacturing – Advanced Rappid Prototyping. Elsevier Science Publishers B.M., Amsterdam, London, New York, Tokyo 1993.
75. STEP Product Data Representation and Exchange/ ISO CD 10303-42 1994.
76. Wirth. N., Algorithms + Data structures = Programs, Programs, Prentice – Hall, Englewood Cliffs. N. J., 1975.