1. **Цель создания и принципы построения САПР**

Целью создания САПР является сокращение материальных и временных ресурсов при проектировании и повышение качества проектирования при научно-обоснованном распределении функций между проектировщиком и компьютером. Основное отличие от ручного – замена физического эксперимента машинным.

Принципы организации:

1) САПР – человеко-машинная система

Основная функция проектировщика состоит в принятии проектных решений на этапах проектирования; подготовка исходных и промежуточных данных для проектирования.

2) САПР – иерархическая система

Иерархия проявляется в структуре технических и программных средств

3) САПР – открытая, мобильная и развивающаяся система

**2. Состав САПР**

Информ. обесп-е

Организ. обесп-е

Методич. обесп-е

Лингв. обесп-е

Матем. обесп-е

Прогр. обесп-е

Технич. обеспечение

*Техническое* представляет собой комплекс технических средств для организации и выполнения проектных процедур. В него могут быть включены специализированные периферийные устройства.

*Программное обеспечение* – комплекс программ, выполняющих проектные процедуры и организующих вычислительный процесс.

*Информационное обеспечение* – базы данных, содержащие информацию об исходных, справочных, промежуточных и результирующих данных, а также программы их обработки типа систем управления базами данных.

*Лингвистическое обеспечение* - совокупность языков САПР и средств их реализации (интерпретаторы, трансляторы, предтрансляторы, метатрансляторы).

*Математическое обеспечение* – модели объектов проектирования, алгоритмы и программы их обработки.

*Методическое обеспечение* – комплект документов для эксплуатации и использования САПР.

*Организационное обеспечение* – регламентирует взаимодействие проектировщиков в коллективе.

1. **Организация языков процессоров САПР**

Применение языков проектирования позволяет сформулировать описание объекта проектирования или задание на проектирование, программу управления процессом проектирования или выдачи документации. Такие описания являются представлением объектов, процессов или заданий на исходном языке.

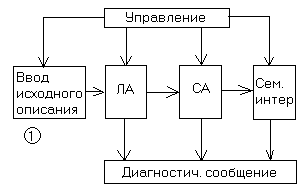
Для обработки формальных описаний требуется их преобразование во внутримашинные структуры данных в виде таблиц, массивов, деревьев, векторов и т. д.

Задание на проектирование преобразуется во внутреннее представление данных и вызовы необходимых для его исполнения проектных процедур, в результате выполнения которых генерируются промежуточные или окончательные решения в формате некоторых языков проектирования.

Такая схема преобразования информации позволяет сформулировать абстрактную модель САПР с точки зрения ЛИО в виде совокупности обрабатывающих подсистем: Pk=(Lk, Jk), Lk - язык, Jk – транслирующая программа (языковый процессор) для этого языка. Абстрактная модель представляет собой иерархию информационно-связанных надсистем, причём на k-ом уровне информация представляется на языке Lk, преобразуется языковым процессором Jk в представление на языке Lk+1.

Трансляторы и интерпретаторы

Языковый процессор состоит из следующих блоков:



1) для ввода и преобразования информации из алфавитно-цифровой или графической формы во внутримашинную двоичную. Для графических языков блок выполняется в виде программного процессора, который осуществляет преобразование данных от разнотипных графических устройств в единую промежуточную форму.

*Средства поддержки ЛИО*: в зависимости от типа исходных языков и методов их обработки выделяются следующие группы поддержки: проблемно-ориентированные системы или генераторы ППП, макрогенераторы, построенные по схеме предтрансляторов; метасистемы или системы построения трансляторов.

Проблемно-ориентированные системы предназначены для построения на их основе специальных программ. В качестве параметров настройки в системе этого класса может использоваться описание модели объекта или предметной области. Система обеспечивает автоматизированный подбор программных модулей и конструирует на их основе программы для исследования объектов заданного класса. Она обеспечивает также автоматический обмен данными между внутренней и внешней памятью ЭВМ, совмещение модулей, написанных на разных языках.

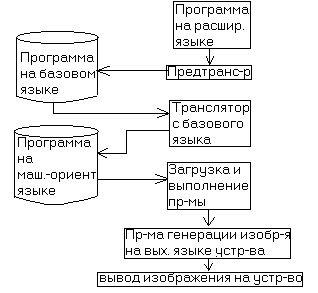
*Макрогенераторы и предтрансляторы* представляют средства для расширения языков программирования и занимают промежуточное положение между пакетами подпрограмм и специализированными языками проектирования. С помощью набора макроопределений создаётся надстройка над базовым языком, которая отражает специфику предметной области.

*Схема предтрансляции* Метасистемы подразделяются на многоязыковые системы программирования и метатрансляторы. Многоязыковые системы создаются на основе выделения общей части трансляторов с нескольких конкретных языков – ядра системы, которое реализует алгоритмы синтаксического анализа определённого класса языков.

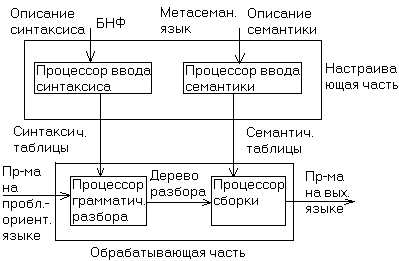
Если грамматика языка Lk относится к классу языков, распознаваемых ядром системы, то создание языкового процессора Jk сводится к описанию грамматики языка Lk на специальном метаязыке системы и генерации таблиц описания грамматики Gk, которые обеспечивают настройку синтаксического анализатора на язык Lk.

Метатрансляторы позволяют создавать языковые процессоры для языков с различными грамматиками. В качестве параметров в таких системах задаётся информация о синтаксисе и семантике языка . Для описания синтаксиса используется БНФ, семантики – специальный метасемантический язык.

*Структура метатранслятора*: Метатрансляторы позволяют автоматизировать создание яз пр-в для различных входных языков, обеспечивая его перевод на любой выбранный язык программирования.



*Схема предтрансляции*



*Системы метатранслятора* Генераторы синтаксических (YACC) и лексических генераторов UNIX. Исходные данные – спецификации лексики входного языка, результаты– LEX – подпрограммы на языке Си (или RATFOR).

Автоматизация разработки диалоговых языков в современных системах достигается за счёт применения унифицированного интерпретатора с класса диалоговых языков и двуязыковой концепции диалоговых программ. Основу двуязыковой концепции составляет использование простого специализированного языка для описания диаграмм состояний параллельно с применением языков программирования.

Диалоговая программа состоит из описания диаграмм состояний на специализированном языке и семантического описания реакции системы на языке программирования. Тексты этих частей могут быть совмещены. Текст диалоговой программы подвергается трансляции двумя трансляторами: с языка описания диаграмм состояний и с языка программирования. В результате первой трансляции создаётся файл описания алфавита и синтаксиса диалогового языка.

Применение двуязыковой концепции даёт возможность использовать единственный язык для описания диаграмм состояний вместе с различными языками программирования высокого уровня. Это исключает необходимость расширять каждый язык программирования диалоговыми компонентами, улучшает технологию и отладку диалоговых программ.

1. **Программное обеспечение САПР**

Специализированное

Общесистемное

Отрасл. применения

ПМК

ППП

программы

ППП – пакеты прикладных программ

ПМК – программно-методические комплексы

*Общесистемное* представляет собой операционные системы, системы автоматизации программирования, средства контроля и диагностики.

*Отраслевое применение* – стандартные программные пакеты, например, графический редактор, СУБД и т д.

*Специализированное обеспечение* – для реализации проектных процедур.

Программы, как правило, предназначены для реализации внутренних нужд предприятия и имеют малую тиражируемость. Пакеты прикладных программ бывают простой и сложной структуры. Пример простого ППП – САПР p-CAD. Конкретная конфигурация ППП определяется спецификой предметной области и формирующей их библиотеки программных модулей проектных процедур. ППП – сложные структуры имеют языковой монитор, предназначенный для описания задания на проектирование. Как правило, программные модули библиотек имеют параметры настройки.

ПМК состоят из процедурной, информационной и методической компонент.

ПО САПР – совокупность программ с необходимой программной документацией, предназначенных для автоматизации проектирования ПО, влияет на стоимость, функциональную полноту, эффективность, гибкость архитектуры САПР.

В структуре ПО выделяется: базовое ПО средств ВТ, базовое общесистемное ПО САПР, специализированное ПО САПР.

Базовое ПО СВТ включает ОС, системы программирования и сервисные программы. Состав БПО СВТ должен удовлетворять специфическим для САПР требованиям: обеспечение диалогового и мультипрограммного режимов работы, возможность динамической загрузки программ; язык управления должен быть доступен для пользователей - непрограммистов, сервисные программы должны обеспечить ведение библиотек программ и данных, копирование данных, динамическую настройку системы на конфигурацию ТС.

ОС современных ЭВМ рассчитаны на решение различных задач и являются универсальными. Это делает ОС далёкими от оптимальных с позиций конкретных задач, в том числе и САПР. Здесь используются 2 подхода: разработка проблемно-ориентированной ОС; использование иерархического построения БПО с универсальной ОС на верхнем уровне и подчинёнными проблемно-ориентированными мониторными системами на нижних уровнях.

Базовое общесистемное ПО САПР – включает программы межотраслевого применения, которые реализуют наиболее общие САПР – ориентированные функции обработки данных и содержат средства, обеспечивающие адаптацию и расширение БПО с учётом конкретных условий применения, в том числе настройку на конфигурацию ТС и технологию проектирования. БПО САПР составляет основу для разработки прикладного ПО и включает средства обработки графической информации, организации диалогового режима, выпуска чертёжной и текстовой документации, организации сетевой обработки данных.

Специализированное ППО подразделяется на проблемно- и объектно-ориентированное.

1) предназначено для автоматизации отдельных этапов (процедур) проектирования (расчёт электрических схем, конструкций). Проблемно-ориентированное ППО должно допускать настройку на проектирование объектов различного функционального назначения.

2) Предназначено для частичного или комплексного автоматизирования проектирования объектов определённого класса, начиная от ввода исходных данных и кончая проектной документацией, допускает настройку на серии типовых элементов внутри этого класса.

Специализированное ППО включает следующие компоненты: программы с необходимой документацией, пакеты программ, программно-методические комплексы.

ППП (пакеты прикладных программ) реализует определённую последовательность проектных операций и обеспечивает получение проектных решений (ex, ППП моделирования электронных схем). ППП простой структуры выполняются в виде библиотек программ, которые используются прикладными программистами для разработки отдельных подсистем. ППП сложной структуры имеет свой монитор и специализированный язык для описания объекта или процесса, который ориентирован на пользователя-непрограммиста.

ПМК (программно-методические комплексы) представляют совокупность компонентов программного, информационного и методического обеспечения. ПМК характерны для создания интегрированных САПР, охватывающих все или почти все этапы проектирования. Трудоёмкость создания ПО таких систем возрастает на 1-2 порядка по сравнению с автоматизацией отдельных проектных решений.

ПО САПР разрабатывается на основе утверждённых ТЗ на программы, ППП или ПМК. В ТЗ на ПО САПР устанавливается требование к его функциональным характеристикам, входным и выходным данным, информационной совместимости, надёжности.

ПО средств ВТ включает ОС, системы программирования, системные ППП, контрольно-диагностические системы.

ОС – управляющие программы (мониторные средства) и программы обслуживания УП планируют ресурсы ВС, обеспечивают взаимодействие с внешней средой, контролируют исправность ТС и организуют восстановление процессов и данных после неисправностей.

Программы обслуживания – сервисные функции: редактирование текстов, копирование файлов, обслуживание библиотек.

Системы программирования – системные ППП – для расширения функций

ОС РВ обеспечивает мультизадачный и мультипрограммный режимы работы, разделение ресурсов системы на базе приоритетов, временную выгрузку задач из ОП на МД. Важной особенностью является возможность многотерминальной работы.

РАФОС ориентирована на создание комплексов, в состав которых могут входить несколько спецпроцессоров.

ДИАМС дисковая диалоговая многопультовая система, предназначена для использования в сложных территориально распределённых автоматизированных системах.

ДОС КП (комплект пользования) – коллективное пользование, обеспечивает эффект использования ресурсов управляющего ВК в режиме разделения времени с квантованием и приоритетной диспетчеризацией.

РОС РВ (расширенная) – обеспечивает мультизадачное выполнение программ в реальном масштабе времени, содержит средства поддержки сетей ЭВМ.

Мобильные ОС. Основные преимущества – простота переноса ПО (особенно прикладного) с ЭВМ одного типа на ЭВМ другого типа, унмверсальность самой ОС. UNIX, DEMOS, ИНМОС, XENIX.

ОС ПЭВМ: СР/М И MS-DOS, M86 и ADOC – отечественные аналоги.

1. **Техническое обеспечение САПР**

На структуру и состав технического обеспечения оказывают влияние следующие факторы:

1. необходимая производительность САПР
2. степень интегрированности комплекса система-проектировщик
3. необходимый перечень подготавливаемой управляющей технологической информации и документации

Наиболее эффективной и гибкой архитектурой организации технических средств САПР СВТ является двухуровневая. Верхний уровень представляет собой мощный центральный процессор с развитой системой хранилищ данных, на котором выполняются наиболее трудоёмкие проектные процедуры и который хранит основные базы данных.

Нижний уровень представляет собой специализированное автоматизированное рабочее место, которое проектировщик использует для подготовки данных, анализа результатов проектирования, выполнения несложных процедур проектирования и подготовки проектной документации.

ОС ЦП

ПО ЦП

Банк данных

…..

I уровень

ЦП

ВУn

ВУ1

БДм

БД1

УСО

ОС АРМ

АРМ

ПО АРМ

II уровень

УСО

АЦД

ГП

…

CD-R

……

ПО ЦП – программное обеспечение центрального процессора

АРМ – автоматизированное рабочее место

УСО – устройство сопряжения

ГП – графопостроитель

АЦД – алфавитно-цифровой дисплей

1. **Задачи компоновки. Технические требования**

Компоновка конструктивных узлов:

2 критерия: функциональная законченность и электромагнитная совместимость элементов. Выполняется с помощью точных и приближённых алгоритмов: точные алгоритмы основаны на применении методов математического программирования и требуют значительных ресурсов. В основном на практике используются приближённые алгоритмы, которые при незначительном снижении качества существенно сокращают вычислительные ресурсы. Приближённые алгоритмы делятся на: последовательного заполнения конструктивных узлов и итерационные алгоритмы. Первая группа выполняется путём реализации последовательности шагов, связанных с добавлением в уже скомпонованный узел эле-мента из списка нескомпонованных и оценки целевой функции качества.

На каждом шаге оптимизирующее значение функции Sa=(Sa1,Sa2,..). В качестве аргументов параметров Sai используются факторы, влияющие на качество компоновки. Пусть некоторые элементы из общего числа N скомпонованы в конструктивные узлы и необходимо в один из них включить очередной элемент a. Для каждого узла q подсчитывается значение функции Sa.

Одним из вариантов использования функции Sa может быть использование следующих параметров:

Sa1 – число связей между размещаемым элементом a и другими элементами, рас-положенными в конструктивном узле q.

Sa2 – число новых связей, которые должны быть подведены к контактам узла q при размещении на нём элемента a.

Sa3 – число общих связей, соединяющих ранее скомпонованные узлы, исключая узел q.

Sa4 – отношение числа свободных мест в узле q к числу элементов, не включённых ни в какой узел и соединённых с элементом a.

Sa5 – число связей между элементом a и другими элементами в ранее скомпонованных узлах.

Функция Sa имеет вид:

Sa=(K1\*Sa1+K2\*Sa2+K3\*Sa3+K4\*Sa4+K5\*Sa5)/(K2\*Sa2+1)

K1-K5 – весовые коэффициенты, подобранные опытным путём.Sa принимает оптималь-ное решение в максимуме.

При выполнении алгоритма перед каждым подсчётом функции Sa необходимо вы-полнить проверку ограничений на максимально возможное число элементов и внешних контактов конструктивных узлов.

В итерационных алгоритмах на 1-м этапе задаётся некоторая произвольная компо-новка. Затем компонуемые элементы меняются местами и подсчитывается значение кри-терия качества компоновки M. Смысловое значение критерия аналогично функции Sa. Одну итерацию составляют L циклов проб с возможными заменами. L – число компонуе-мых элементов.

Во время каждого цикла элемент a, расположенный в узле q, меняется местами со всеми L-n\*q элементами, расположенными в других узлах. После каждой замены места-ми a и b подсчитывается изменение критерия качества. Среди всех возможных выбирает-ся максимальное изменение критерия.

Очевидно, что время работы алгоритма больше, чем у предыдущего.

В результате решения задач компоновки определяется множество элементов, принадлежащих каждому конструктивному узлу.

1. **Информационное обеспечение САПР СВТ. Состав.**

ИО САПР СВТ

Типовые сведения

Способы организации, размещения и обработки данных

Справочные данные об

элементах и комплектующих

СУБД

Типовые проектные решения

Способы организации

Способы размещения

Библиотеки стандартных

элементов

последовательное последовательная

…………………….

прямое иерархическая

библиотечное ассоциативная

индексно- реляционная

последовательное

программы

языки

СУБД

описание поиск

обработка введение деревьев

манипулирование восстановление

структурированная информация САПР обычно представляется в виде записей. Запись имеет семантический или смысловой ключ, позволяющий её идентифицировать; ключ порядка, указывающий местоположение записи.

Библиотечная организация предполагает наличие разделов. Индексно-последовательное размещение использует разбиение ключей порядка на группы (индексы).

Под способами организации понимается объединение записей или групп записей в единое целое в соответствии с определёнными принципами с целью поиска и модификации информации.

Ассоциативная организация использует объединение данных в соответствии с признаком, например, семантическим ключом.

признак ассоциативного

поиска

информационная часть

линейка индикаторов

ассоциативная часть

информационная составляющая

Если слово-обозначение найдено, то в линейке индикаторов считывается информационная составляющая.

В операции ассоциативного поиска активно используется операция маскирования.

Если сработали несколько индикаторов, то многозначный ответ, линейка индикаторов – анализатор многократного совпадения, которое, в соответствии с определённой стратегией, считывает информационные составляющие.

Реляционный способ используется во многих современных СУБД. Имеется реляционная алгебра, составляющие компоненты которой – объекты и операции. Объектами являются записи, объединённые в таблицы. Операциями могут быть равно, эквивалентно, больше, меньше, содержит и т.д. В результате применения операций над объектами формируются новые объекты (таблица).

1. **Этапы проектирования ЭВМ**

1) системное

2) логическое

3) алгоритмическое

4) техническое

На этапе 1 решаются задачи:

А) определяется состав и основные характеристики функциональных блоков вычисли-тельной системы

Б) разрабатываются технические требования на способ взаимодействия блоков и их сопряжение между собой

В) определяется структура вычислительных процессов - последовательная, параллельная, конвейерная, их комбинации и их количественные характеристики

Г) выбираются основные принципы организации операционной системы

Исходными данными этого этапа являются требования технического задания, характеристики по производительности, объёму памяти, время доступа к внешним устройствам, усреднённые параметры решаемых задач (память, время), габариты, вес и т.д.

На этапе логического проектирования разрабатывается логическая структура функциональных блоков, в зависимости от выбранного микросхемного базиса разрабатываются средства контроля и диагностики, строятся детальные временные диаграммы.

На этапе алгоритмического проектирования решаются следующие задачи:

А) компромиссное деление математического обеспечения на внутреннее, реализуемое схемно (или микропрограммно), и внешнее, реализуемое программно

Б) разработка системы команд

В) формирование набора микроопераций и логических условий, необходимых для реализации системы команд

Г) разработка временных диаграмм и алгоритмов функционирования блоков

На этапе технического(конструкторского, схемотехнического) проектирования решаются задачи:

А) компоновка логических элементов в подсхемы заданной сложности

Б) размещение структурных элементов в пространстве или на плоскости

В) соединение компонент друг с другом(трассировка)

Г) выпуск технической документации, необходимой для производства и эксплуатации

1. **Классификация языков САПР**

Включает в себя термины и определения, правила формализации языка и методов реализации.

Языки САПР

Языки управления

Языки проектирования

Языки программирования

внутренние

входные

выходные

сопровождения

ЯОО

ЯОЗ

моделирования

схемный

графический

Универсальные языки достаточно эффективно использовать для реализации САПР, так как они обладают развитыми возможностями для описания алгоритмов, характерных для ПО САПР. Однако при их использовании в качестве языков проектирования программы получаются очень громоздкими и неудобными. Это объясняется необходимостью специальной подготовки в области программирования и сложностью самих процессов трансляции, генерации и отладки программ для управления последовательностью проектных процедур и операций в соответствии с заданием на проектирование, которое составляется на этих языках.

Поэтому необходимо создание специализированных проблемно-ориентированных языков проектирования и языковых процессоров к ним.

*Внутренние* - их появление объясняется выделением в САПР некоторых подсистем, ex, графического ввода, графического документирования, архива чертежей, инвариантных к классам объектов проектирования и необходимостью унификации представления входных и выходных данных для этих подсистем.

*Промежуточные* языки позволяют легко включать инвариантные подсистемы в различные САПР путём разработки специальных программ-конверторов, которые выполняют преобразования данных из входных языков различных систем в некоторый промежуточный (внутренний) язык определённой инвариантной подсистемы. Примерами внутренних языков могут служить язык представления графической и текстовой информации (ЯГТИ) , языки графических метафайлов для хранения данных в архивах чертежей.

Эта классификация по признаку: *место языка в процессе автоматизации проектирования*.

*По признаку связи с универсальными языками*  различают автономные и расширенные языки. Автономные языки имеют собственные грамматики, трансляторы и могут применяться независимо. Расширяющиеся языки строятся на основе грамматики другого языка и являются его проблемно-ориентированными дополнениями. Такой подход позволяет использовать в языках проектирования все имеющиеся в базовом языке мощные средства обработки данных и упростить связь языков проектирования с другими программными средствами системы.

*По оперативности* языки разделяют на диалоговые и пассивные. Диалоговые обеспечивают взаимодействие проектировщика с ЭВМ на основе взаимного обмена сообщениями в реальном масштабе времени. Это позволяет оперативно получать все промежуточные результаты и управлять процессами проектирования. Пассивные языки позволяют задавать входные данные и последовательность проектных операций и процедур в виде некоторого символического описания с последующей трансляцией этих описаний и выполнением в пакетном режиме.

*По способу представления информации* выделяют алфавитно-цифровые, графические, голосовые и смешанные языки проектирования.

В качестве начального нетерминального символа во входных языках проектирования чаще всего выбирают понятие “директива”. Остальные нетерминальные символы определяют лексические классы входного языка: ДМ - директива, Д - действие, ОД – объект действия, СД – способ действия и т.д.

БНФ – грамматики (слева металингвистич. переменная, которая обозначает соответствующую конструкцию; наличие рекурсий)

# Входные языки

## Входные графические языки нашли широкое распространение в САПР, так как часто исходная информация об объекте имеет графическую форму в виде чертежей, эскизов, функциональных зависимостей. Эти языки составляют основу лингвистического обеспечения в подсистемах геометрического моделирования и машинной графики.

Для описания геометрии объектов проектирования используются следующие способы: координатный*,* структурно-символьный, аналитический и рецепторный.

1. задаются координаты всех характерных точек объекта и их связность;
2. предварительное выделение типовых базовых графических элементов, создание и дальнейшее использование библиотеки БГЭ для составления форм объекта;
3. описание графического изображения в виде математических отношений (уравнений поверхностей и линий);
4. использование мозаичного представления в виде матрицы.

В основу разработки программных средств машинной графики и геометрии положены процедурные графические расширения определённых языков программирования. Это объясняется прежде всего трудностью разработки универсального графического языка высокого уровня, который отвечал бы большому числу требований, выдвигаемых развитием технических средств машинной графики, сферами применения такого языка.

Расширение определённого языка графическими процедурами позволяет основать единый язык для разработки всех компонентов ПО САПР, в том числе трансляторов из входных графических языков, программ обработки и хранения изображений, формирование данных на выходных языках для вывода на графические устройства.

Графические расширения ФОРТРАНа: ГРАФОР, СМОГ, ГРАФИТ, ФАП-КФ и другие. Расширения производятся двумя методами: прямого расширения и расширения с использованием предтрансляторов. Первый метод состоит в создании пакета подпрограмм, каждая из которых реализует какую-либо одну геометрическую или графическую функцию. Обращение к подпрограмме производится из прикладной программы, для написания которой используется расширенный язык.

Графическая программа представляет собой совокупность операторов обращения к различным подпрограммам, то есть изображение описывается путём использования синтаксических конструкций и типов данных расширяемого алгоритмического языка. Примером процедурного языка этого типа является ГРАФОР, в котором выделяются следующие уровни: программы, выполняющие общие организационные функции (задание границ страницы, закрытие страницы, задание единиц измерения и т.д.); программы построения графических примитивов (прямая, окружность, символы программы построения графиков, программы построения изолиний), трёхмерных сцен и т.д.

Второй метод предполагает добавление новых синтаксических элементов (графических переменных, констант, типов данных и операций над ними) таким образом, чтобы процедуры построения изображения были написаны наглядно и в компактном виде. Такой подход требует разработки *предтрансляторов*, которые осуществляют перевод программы на расширенном языке в программу на базовом языке, которая затем транслируется с использованием имеющегося на ЭВМ транслятора.

*Входные языки логического моделирования* ПРОЛОГ, КОМОЛ, СТРУКТУРА. Задаются сведения об объекте в виде описания схем и протекающих в них процессах, эти языки используются для проектирования логических и функциональных схем ЭВМ. Языки моделирования GPSS, СИМСКРИПТ, НЕДИС используются для описания информационных процессов.

Реализация диалоговых языков САПР

ДЯ служат средствами оперативного взаимодействия проектировщика с САПР в реальном масштабе времени. В процессе диалога создаётся цифровая модель проектируемого объекта, выполняются расчёт и анализ характеристик объекта, формируется необходимая для его изготовления документация или программы для управления автоматами. Одним из основных требований к ДЯ является близость к естественным.

По способам ввода команд наибольшее распространение получили языки типа “запрос-ответ” на основе: директив пользователя; выбора альтернативных возможностей; заполнение пользователем форматов.

В директивных языках основным форматом представления является текстовая строка, а основным устройством ввода - алфавитно-цифровая клавиатура. Недостатком таких языков является то, что пользователь должен помнить все элементы словаря и правила синтаксиса.

В языках, основанных на процедуре выбора альтернативных возможностей (типа меню), конструирование предложения происходит путём указания предоставляемых на экране элементов словаря или выполнения действий с устройствами графического ввода. В этом случае основная информация, предоставляемая пользователю, имеет контрольно-справочный характер о возможностях системы, о состоянии цифровой модели объекта и базы данных. Альтернативные языки обладают следующими достоинствами:

более высоким б/д ввода, так как пользователь не должен полностью вводить слово или фразу;

от пользователя не требуется знания форматов и символики множества директив, а достаточно знать простые правила работы с устройствами интерактивного ввода для выбора нужной команды из набора предоставляемых.

упрощается этап лексического и синтаксического анализа.

Процедура трансляции предложений языка существенно упрощается, если синтаксический разбор начинается не по окончании ввода всего предложения, а по мере ввода его отдельных членов. В этом случае или предоставляется та часть словаря, из которой выбирается очередной член предложения. Языки с такой организацией ввода называются *диалоговыми языками со сменными наборами команд*.

Диалоговые языки, основанные на использовании графических изображений и устройств ввода графических данных, называются диалоговыми графическими языками. В них выделяют языки изображений и языки действий. Язык изображений предназначен для вывода из ЭВМ графического представления проектируемого объекта, сведений о состоянии обрабатываемых данных, вычислительного процесса. Язык действий – это язык, на котором пользователь вводит в ЭВМ свои ответы и задания.

Для улучшения взаимодействия человека и ЭВМ в диалоговых графических языках используется ряд приёмов в организации языка изображений и сервисных программных средств. Поле экрана графического дисплея разделяется на ряд областей по функциональному назначению: рабочую, в которой воспроизводится собственно графическое представление объекта проектирования; процессов – предназначенную для отображения ключевых слов команд пользователя; графических данных – для отображения графических объектов (стандартных или построенных ранее), используемых для построения изображений; сопровождения диалога; контроля данных.

Диалоговые графические системы должны иметь адаптивные сервисные средства, которые позволяют настраиваться на уровень подготовленности пользователя или предусматривают возможность отключения соответствующих блоков сервисных средств самим пользователем.

Функционирование диалоговых графических систем представляется в виде периодического процесса, управляемого с помощью команд пользователя, которые последовательно переводят систему из одного состояния в другое. Грамматика ДГС G=(Vt,Vн,So,P), Vt включает набор слов и действий пользователя с устройством ввода, Vн – составленный из номеров состояний системы с начальным состоянием So, P: β Ui C, интерпретируется как переход системы из состояния с номером β ∈ Vн в состояние C ∈ Vт, представляющих i-ю команду управления диалогом. Это подмножество КС - языков.

*Интерпретаторы* диалоговых графических языков предназначены для анализа правильности действий пользователя в процессе диалога и для организации выполнения в ЭВМ программ, соответствующих ожидаемой реакции на эти действия.

Моделью интерпретатора является конечный автомат(преобразователь):

A=(X , Y , S , So ,Z , Ф, ψ), Ф(перехода): X\*S S; ψ(выхода)

вход выход сост. нач. сост. конеч. X\*S Y;

При семантической интерпретации конечного автомата как модели диалоговой программы его элементам придаётся следующий смысл: X – множество допустимых воздействий пользователя (команды языка); Y – множество имён семантических подпрограмм, осуществляющих выполнение команд пользователя.

*Пример графа автомата*

Представление интерпретатора в виде иерархической системы автоматов допускает простое отображение на традиционные структуры программирования. Если каждому автомату такой системы поставить в соответствие диалоговую подпрограмму, интерпретирующую диалоговый подъязык, то диалоговую графическую программу можно представить в виде совокупности диалоговых подпрограмм (рис).

1. **Задачи размещения. Технические требования**

В соответствии с иерархическими принципами рассматриваются задачи размещения элементов в ТЭЗах печатных плат в блоках, блоков в шкафах.

ТЭЗ – типовой элемент замены

ТЭЗ элемент(модуль, СБИС, БИС, навесной элемент) ТЭЗ(печатная плата) блок шкаф

Цель размещения: максимальное снижение искажений сигналов и максимальное упрощение последовательности трассировки.

Выделяют следующие основные критерии размещения: минимизация суммарной длины соединения в устройстве, минимизация числа пересечений проводников, относящихся к различным сигналам, максимизация числа цепей с возможно более простой конфигурацией. Первый критерий является наиболее общим и чаще всего используется.

Задача размещения формулируется следующим образом: расположить элементы в пространстве так, чтобы оптимальным образом были бы выполнены системы ограничений. Решение производится точными и приближёнными методами.

Приближённые алгоритмы разделяются на три группы:

А) алгоритмы, основанные на использовании силовых функций

Б) перестановочные алгоритмы

В) последовательные алгоритмы

В первой группе элементы заменяются материальными точками, предварительно материальные точки произвольно размещаются в монтажном пространстве, между ними вводятся силы притяжения и отталкивания, пропорциональные числу связей между элементами. Такая динамическая система описывается в виде системы нелинейных дифференциальных уравнений, решения которой и являются координатами материальных точек на плоскости.

Недостатками метода являются:

А) замены элементов, имеющих определённые размеры, материальными точками

Б) переход от непрерывных координат к дискретным

В) приближённый способ подбора силовых коэффициентов

Г) приближённое решение системы нелинейных дифференциальных уравнений

В *перестановочных* алгоритмах задаётся любое начальное размещение элементов в дискретном монтажном пространстве, затем с помощью попарных перестановок минимизируется суммарная длина связей между ними. Эта группа является наиболее распространённой.

Оптимальное размещение находится методом полного перебора. Это недостаток. Используется модификация этого метода, связанная с выбором для перестановки 1-го элемента, который имеет наибольшее число связей со всеми остальными.

В *последовательных* алгоритмах элементы размещаются в монтажном пространстве последовательно, при этом не требуется задавать их начальное расположение. Предварительно размещаются один или несколько элементов. Такое размещение задаётся проектировщиком. При последовательном размещении каждого нового элемента в пространстве учитывается число связей с уже размещёнными и ещё не размещёнными элементами. При этом выбирается элемент, имеющий максимальное число связей с размещёнными элементами и по возможности небольшое число связей с ещё не размещёнными.

Выбранный элемент располагается в таком установочном месте монтажного пространства, которое обеспечивает минимизацию суммарной длины всех проводных связей.

При необходимости связи объединяются в группы. Для них вводятся весовые коэффициенты, учитывающие важность группы. Преимущество алгоритмов последовательного размещения – быстрота и простота реализации.

1. **Математическое обеспечение САПР СВТ.**

Q

Y

X

X – вектор внутренних параметров модели, характеризующий её состояние

Q – вектор внешних параметров (параметров воздействий)

Y – вектор выходных параметров

Как правило, в моделях отражаются свойства, связанные с решением задач определённого этапа проектирования.

Свойства математических моделей:

1. Точность

Пусть Yjm – значение выходного параметра Yj, рассчитанное с помощью модели; Yju – естественное значение реального объекта.

Легко вычислить абсолютную погрешность по формуле:

Ej=|(Yju-Yjm)/Yjm|

Вектор погрешностей:

E=(E1,E2,…,En)

на практике переходят к скалярной величине и рассматривают её как погрешность модели

1. Универсальность

На практике используются универсальные модели, предназначенные для длительного применения.

1. Адекватность

Характеризуется областью, определяемой вектором Q (внешних воздействий). В этой области модель позволяет решать возложенные на неё задачи.

Есть область адекватности – ОД. Она определяется параметрами вектора Q.

Для упрощения математической обработки области адекватности рассматривают ограниченную область адекватности.

1. Вычислительные затраты

2 основных критерия – память и время.

Требование точности, универсальности и адекватности с 1-й стороны, и уменьшение вычислительных затрат с другой являются противоречивыми.

1. **Характеристика логического этапа проектирования ЭВМ**.

На этапе логического проектирования разрабатывается логическая структура функцио-нальных блоков, в зависимости от выбранного микросхемного базиса разрабатываются средства контроля и диагностики, строятся детальные временные диаграммы.

1. **Формирование списка соединений**

Исходными данными является принципиальная схема в виде графического образа или описания на схемном языке. Результатом работы программ этого этапа является следующая типовая таблица:

|  |  |  |  |
| --- | --- | --- | --- |
| Тип элемента | Номер элемента | Номер контакта | Наименование цепи |

На этом этапе выявляются следующие семантические ошибки: висячие контакты, при использовании схемного языка возможны следующие типы семантических ошибок: не описан элемент, повторно описан элемент, повторно описана цепь.

Компоновка конструктивных узлов:

2 критерия: функциональная законченность и электромагнитная совместимость элемен-тов. Выполняется с помощью точных и приближённых алгоритмов: точные алгоритмы основаны на применении методов математического программирования и требуют значи-тельных ресурсов. В основном на практике используются приближённые алгоритмы, ко-торые при незначительном снижении качества существенно сокращают вычислительные ресурсы. Приближённые алгоритмы делятся на: последовательного заполнения конструк-тивных узлов и итерационные алгоритмы. Первая группа выполняется путём реализации последовательности шагов, связанных с добавлением в уже скомпонованный узел эле-мента из списка нескомпонованных и оценки целевой функции качества.

На каждом шаге оптимизирующее значение функции Sa=(Sa1,Sa2,..). В качестве аргументов параметров Sai используются факторы, влияющие на качество компоновки. Пусть некоторые элементы из общего числа N скомпонованы в конструктивные узлы и необходимо в один из них включить очередной элемент a. Для каждого узла q подсчиты-вается значение функции Sa.

Одним из вариантов использования функции Sa может быть использование следу-ющих параметров:

Sa1 – число связей между размещаемым элементом a и другими элементами, рас-положенными в конструктивном узле q.

Sa2 – число новых связей, которые должны быть подведены к контактам узла q при размещении на нём элемента a.

Sa3 – число общих связей, соединяющих ранее скомпонованные узлы, исключая узел q.

Sa4 – отношение числа свободных мест в узле q к числу элементов, не включённых ни в какой узел и соединённых с элементом a.

Sa5 – число связей между элементом a и другими элементами в ранее скомпоно-ванных узлах.

Функция Sa имеет вид:

Sa=(K1\*Sa1+K2\*Sa2+K3\*Sa3+K4\*Sa4+K5\*Sa5)/(K2\*Sa2+1)

K1-K5 – весовые коэффициенты, подобранные опытным путём.Sa принимает оптимальное решение в максимуме.

При выполнении алгоритма перед каждым подсчётом функции Sa необходимо вы-полнить проверку ограничений на максимально возможное число элементов и внешних контактов конструктивных узлов.

В итерационных алгоритмах на 1-м этапе задаётся некоторая произвольная компоновка. Затем компонуемые элементы меняются местами и подсчитывается значение кри-терия качества компоновки M. Смысловое значение критерия аналогично функции Sa. Одну итерацию составляют L циклов проб с возможными заменами. L – число компонуе-мых элементов.

Во время каждого цикла элемент a, расположенный в узле q, меняется местами со всеми L-n\*q элементами, расположенными в других узлах. После каждой замены местами a и b подсчитывается изменение критерия качества. Среди всех возможных выбирается максимальное изменение критерия.

Очевидно, что время работы алгоритма больше, чем у предыдущего.

В результате решения задач компоновки определяется множество элементов, принадлежащих каждому конструктивному узлу.

1. **Характеристика системного этапа проектирования ЭВМ.**

На этапе решаются задачи:

А) определяется состав и основные характеристики функциональных блоков вычисли-тельной системы

Б) разрабатываются технические требования на способ взаимодействия блоков и их сопряжение между собой

В) определяется структура вычислительных процессов - последовательная, параллель-ная, конвейерная, их комбинации и их количественные характеристики

Г) выбираются основные принципы организации операционной системы

Исходными данными этого этапа являются требования технического задания, характери-стики по производительности, объёму памяти, время доступа к внешним устройствам, усреднённые параметры решаемых задач (память, время), габариты, вес и т.д.

1. **Характеристика технического этапа проектирования ЭВМ.**

На этапе технического(конструкторского, схемотехнического) проектирования решаются задачи:

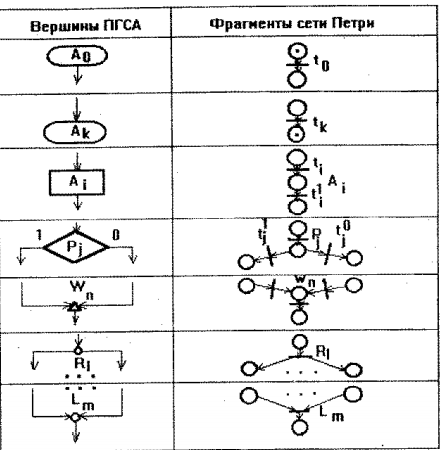
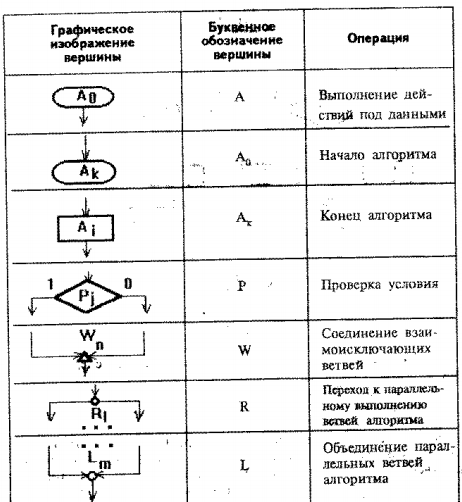
А) компоновка логических элементов в подсхемы заданной сложности

Б) размещение структурных элементов в пространстве или на плоскости

В) соединение компонент друг с другом(трассировка)

Г) выпуск технической документации, необходимой для производства и эксплуатации

1. **Моделирование сетями Петри программного обеспечения ЭВМ.**



параллельная граф схема алгоритма корректна если алгоритм: детерминирован, конечен

параллельные операторные схема алгоритма(ПОСА)

поса корректна, когда соотв. сеть петри жива и безопасна

методика моделирования

1. построение для поса соотв. сети петри
2. проверка работоспособности
3. определение свойств
4. определение временных хар. сети петри

моделирование позволяет получать качественные(корректность) и количественные(временные) оценки алгоритмов

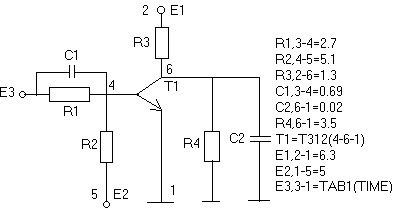
1. **Классификация языков САПР. Схемные языки**

Схемные языки предназначены для ввода данных об объекте проектирования, представленного в виде структурных, функциональных или принципиальных схем, которые отображают множество элементов схемы и связи между ними.

Описание схем с помощью схемных языков (СПАРС, АРОПС, КРОСС) состоит из предложений, каждое из которых содержит сведения об элементе схемы и его связях. Сведения об элементе включают его тип, имя (номер) и числовое значение параметров. Связи задаются номерами узлов, к которым подсоединяются внешние входы (выходы) элементов, а также типами связей.

Различают форматные и бесформатные схемные языки. На форматных языках описание представляется в виде таблиц и входных документов. Такая форма удобна для ввода схем с однородными элементами по числу связей и количеству параметров.

В бесформатных языках части предложений и сами предложения отделяются друг от друга специальными разделителями , ; / . Бесформатные языки более удобны для описания схем с разнохарактерными элементами.



Выводы Т1 (4-6-1) даны в последовательности база, коллектор, эмиттер. Е1, Е2, Е3 описывают источники напряжения. Тип функциональной зависимости источника напряжения Е3 от времени задаётся таблицей таб.1

1. **Задачи трассировки. Технические требования**

Является наиболее сложной из задач технического проектирования. Заключается в прокладке проводящих трасс между контактами размещённых элементов в соответствии со списком соединений. Алгоритмы трассировки учитывают следующие факторы:

Основная функция качества – минимальная длина трассы, минимальное число изгибов трассы, минимальное число переходных отверстий между слоями трассировки. Часто вводятся ограничения на направление прокладки.

Основная группа алгоритмов – волновые или алгоритмы Ли.

Исходными данными алгоритмов трассировки является список соединений. Задача трассировки имеет метрический и топологический аспекты. Метрический аспект связан с учётом геометрии элементов и геометрическим расположением на плоскости. Топологический аспект связан с учётом ограничений на число допустимых пересечений, числом слоёв печатной платы, то есть связан с пространственным расположением отдельных компонентов схемы и их соединений. Классификация алгоритмов методов трассировки:

Построение минимальных деревьев

Список соединений

Трассировка по магистралям

Комбинированные алгоритмы

Трассировка соединений

Динамические схемы упорядочения

Оценка длин соединений, числа пересечений, изгибов

Очерёдность прокладки соединений

Ортогональное расслоение с минимизацией переходов

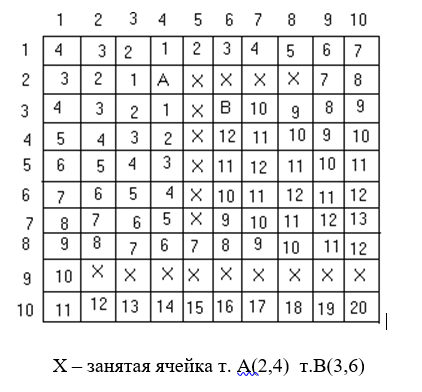
Раскраска графа пересечений

расслоение

Канальная трассировка

Волновой алгоритм и его модификации

В последовательных алгоритмах трассировки трассы цепей проводятся в определённом порядке одна за другой. Обычно начинают прокладку трасс либо с самых длинных соединений, так как в заполненном монтажном пространстве их затем труднее формировать, либо с самых коротких соединений, которые плотнее заполняют монтажное пространство. Проложенная трасса фиксируется и при дальнейшей трассировке рассматривается как препятствие, то есть занятые ячейки монтажного пространства. Таким образом, в последовательных алгоритмах выполняется локальная оптимизация при прокладке каждой трассы, но в результате после проведения ряда трасс некоторые участки монтажного пространства могут блокироваться, что не позволяет выполнить трассировку автоматически. В итерационных алгоритмах после прокладки всех трасс, которая производится без учёта взаимного влияния трасс, определяется функция качества трассировки как взвешенные суммы параметров трассы: длина, число пересечений, изгибов. Наихудшие трассы удаляются, процесс трассировки повторяется с учётом наиболее лучших соединений как препятствий. Разработаны алгоритмы, в которых объединены последовательные и итерационные процедуры, в которых мешающие трассы деформируются и удаляются, то есть изменяется конфигурация уже проложенных соединений или очерёдность прокладки трасс. Большинство известных универсальных алгоритмов трассировки основаны на волновом алгоритме определения пути, трассы соединений контактов элементов, минимизирующей некоторую многомерную функцию качества этого пути.



Фронт распространения волны имеет ортогональные направления, каждой клетке в процессе прохождения волны присваивается масса. Масса точки А равна 0 ,масса точки В=11.

Критерии:

1. минимальное число перегибов трассы
2. наиболее полное заполнение монтажного пространства

среди множества путей одинаковой длины выбирается тот, который имеет наименьшее число изгибов и наиболее полно заполняющий монтажное пространство.

Cледующая группа алгоритмов трассировки – лучевые. Между точками А и В проводится последовательность лучей, причём при каждом переходе к последующей точке определяется направление трассы, минимально отличающееся от предыдущего направления луча в соответствии с диаграммой приоритетов:



Рассмотренный критерий оптимальности (масса точки) вводится не для управления процессом трассировки, а для оценки качества полученного решения. В ряде случаев делается попытка учесть последующие шаги трассировки и организовать параллельную трассировку всех соединений. Примерами являются алгоритмы, использующие канальное представление магистралей соединений

В1 В2 В3

Г1

Г2

Г3

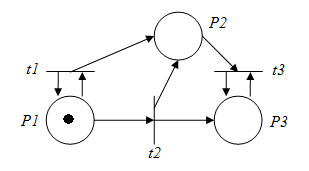
В подобных устройствах, реализуемых на двусторонних печатных платах, имеются два слоя с вертикальной и горизонтальной коммутацией; для сопряжения слоёв используются контактные переходы. Трассировка в два этапа: предварительная с целью распределения трасс по каналам при равномерной их загрузке и окончательная, в процессе которой уточняется расположение соединений по магистралям каналов.

1. **Построение дерева достижимых маркирований для сетей Петри**

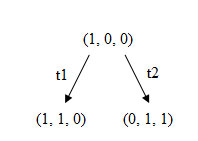
**Дерево достижимости**

Основная задача анализа сетей Петри – задача достижимости: достижима ли маркировка μ' из начальной маркировки μ0 данной сети Петри.

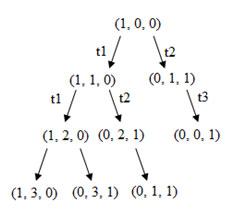
Для ее решения имеется два основных подхода. Первый основан на построении **дерева достижимости**. Дерево достижимости – это ориентированное корневое дерево, вершинам которого, соответствуют возможные маркировки, а дугам – переходы. Начальная маркировка соответствует корню дерева. Из него исходят дуги, соответствующие разрешенным переходам. На каждом шаге строится очередной ярус дерева.

Пусть имеется сеть Петри, представленная на рисунке 1. 

Ее начальная маркировка - (1, 0, 0). В этой начальной маркировке разрешены переходы **t1** и **t2**.



Теперь необходимо рассмотреть все маркировки, достижимые из новых маркировок.

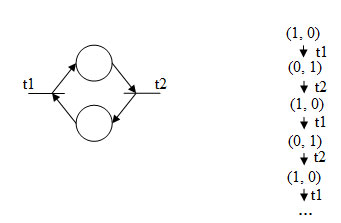


В нашем примере можно запустить переход t1 столько раз сколько необходимо для того, чтобы получить произвольное число меток в P2. Это бесконечное число маркировок обозначим символом **ω**.



Заметим, что маркировка (0, 0, 1) **пассивная**; никакой переход в ней не является разрешенным, поэтому никакие новые маркировки из этой пассивной маркировки в дереве порождаться не будут. Кроме того, необходимо отметить, что маркировка (0,1, 1), порождаемая запуском **t3** в (0, 2, 1) была уже порождена непосредственно из начальной маркировки запуском t2.

Если эту процедуру повторять снова и снова, каждая достижимая маркировка окажется порожденной. Однако получившееся дерево достижимости может оказаться бесконечным, так как будет порождена каждая маркировка из множества достижимости. Даже сеть Петри с конечным множеством достижимости может иметь бесконечное дерево



Для превращения дерева в полезный инструмент анализа необходимо найти средства ограничения его до конечного размера. Это может осуществляться несколькими способами. Прежде всего, необходимо использовать те средства, которые ограничивают введение новых маркировок (называемых граничными вершинами) на каждом шаге. Для этого могут вводиться **пассивные маркировки**, то есть маркировки, в которых нет разрешенных переходов. Такие пассивные маркировки называются **терминальными вершинами**. Другой класс маркировок - это маркировки, ранее встречавшиеся в дереве. Такие маркировки называются **дублирующими вершинами**: никакие последующие маркировки можно не рассматривать - все они будут порождены из места первого появления дублирующей маркировки в дереве. Таким образом, в дереве на рисунке 5 маркировка (0, 1, 1), получившаяся в результате выполнения последовательности **t1, t2, t3**, не будет порождать какие-либо новые вершины в дереве, поскольку она ранее встречалась в дереве в результате выполнения последовательности t2 из начальной маркировки.

Каждая вершина может классифицироваться как граничная вершина, терминальная вершина, дублирующая вершина или как внутренняя вершина. **Граничными** являются вершины, еще не обработанные алгоритмом, а после обработки они могут стать терминальными, дублирующими или внутренними вершинами.

Если все вершины дерева - терминальные, дублирующие или внутренние, то обработка данным алгоритмом останавливается.

1. **Моделирование сетями Петри вычислительных процессов и автоматов**
2. **Расширенные сети Петри**



**Временная сеть Петри**

Такая сеть позволяет более реалистично отражать процессы в ВС. Во временных сетях каждому переходу **tj** сопоставляется время **τj**. Если переход возбуждается, то метки, вызвавшие запуск перехода, покидают входные позиции **Pre(tj).** Порождение меток в выходных позициях **Post(tj)** происходит через время **τj** .

*Формальное определение временной сети:*

TN = {N, τ},

где N - сеть Петри; τ: T → R0 -функция времён срабатывания, сопоставляющая каждому переходу постоянное время срабатывания; R0 - множество неотрицательных рациональных чисел.

**Сеть с приоритетами переходов**

*Формальное определение:*

PRN = {N, PR},

где N - сеть Петри; PR - отношение приоритетности (порядка), задаваемое на множестве переходов Т и определяющее порядок потребления меток возбуждёнными переходами в условиях конфликта за метку.

**Временная сеть с приоритетами переходов**

Такая сеть объединяет элементы, описанные в рассмотренных выше классах сетей.

*Формальное определение:*

PRTN = {N, τ, PR}.

**Раскрашенные сети**

В цветных сетях вводится понятие цвета для фишек. В общем случае может быть **n** цветов. В вычислительной технике используются трехцветные сети (**n = 3**). Такие сети используются для моделирования аппаратных средств. Правила возбуждения переходов дополняются условиями, предполагающими выбор меток определённых цветов из позиций Pre(tj). Срабатывание переходов сопровождается посылкой в позиции Post(tj) меток, с задаваемыми значениями цвета.

**Ингибиторная сеть**

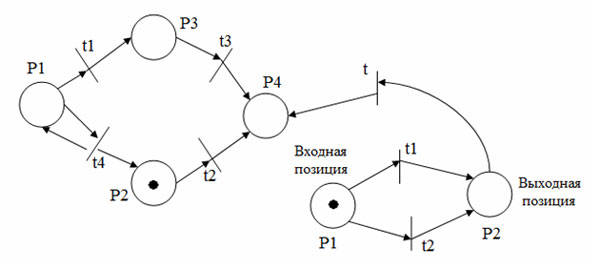
Это сеть Петри, дополненная специальной функцией инцидентности **IIN: Р х Т -> {0, 1}**, которая вводит ингибиторные (запрещающие) дуги для тех пар **(p, t)**, для которых **IIN(Р, Т) = 1**. Ингибиторные дуги связывают только позиции с переходами, на рисунках их изображают заканчивающимися не стрелками, а маленькими кружочками.

Переход **t** в ингибиторных сетях может сработать, если каждая его входная позиция, соединенная с переходом обычной дугой с кратностью **w(p, t)** содержит не менее **w(p, t)** фишек, а каждая входная позиция, соединенная с переходом **t** ингибиторной дугой (ее кратность всегда равна 1), имеет нулевую разметку.

**Предикатные сети** - это сети с логическим описанием состояния системы.

**Струтурированные сети**

В структурированных сетях некоторые из переходов являются сложными. При их срабатывании запускается сеть другого уровня иерархии



Срабатывание **t2** приводит к запуску сети другого уровня. Выполнение сложного перехода заключается в помещении во входную позицию по сети фишки. После выполнения сети фишка появляется в ее выходной позиции, затем формируются фишки в выходных позициях сложного перехода. Преобразование сети к виду, имеющему один вход и один выход, всегда возможно. Такие сети используются для моделирования модульных вычислительных систем.

1. **Свойства сетей Петри. Их определение по дереву достижимостей.**

Основные свойства сетей Петри:

1. Ограниченность или K-ограниченность
2. Безопасность
3. Сохраняемость
4. Достижимость
5. Живость

**Ограниченность**. Это свойство связано с введением ограничений на число меток в позициях.

Позиция pi называетсяk-ограниченной,если количество фишек в ней не может превышать целого числа k. Сеть Петри называется ограниченной, если все ее позиции ограничены. Ограниченную сеть Петри можно реализовать аппаратно, а неограниченную нельзя.

**Безопасность**. Позиция сети Петри называется безопасной, если число фишек в ней никогда не превышает единицы. Сеть Петри безопасна, если безопасны все ее позиции.

**Сохраняемость**. Сеть Петри называется строго сохраняющей, если сумма фишек по всем позициям остается строго постоянной в процессе выполнения сети, т. е. для всех возможных маркировок μ'.

Это свойство особенно присуще сетям, в которых фишки интерпретируются как ресурсы; фишки не должны ни создаваться, ни уничтожаться, следовательно, число входов в каждый переход должно равняться числу выходов.

**Живость**. Это свойство системы, означающее, что из любого состояния, достижимого из начального, возможен переход в любое другое достижимое состояние. Свойство позволяет исследовать систему на отсутствие тупиков, зацикливаний или каких-либо блокировок в процессе моделирования.

**Достижимость**. Свойство достижимости используется при установлении возможности возникновения некоторой ситуации в системе. Пусть проверяемая ситуация описывается разметкой 'μ. Возникает задача: достижима ли маркировка μ' из начальной маркировки μ0 данной сети Петри. Задача достижимости является одной из наиболее важных задач анализа сетей Петри.

**Анализ сетей Петри на основе дерева достижимости.**

Сеть Петри ограничена тогда и только тогда, когда символ **ω** отсутствует в ее дереве достижимости. Если сеть ограничена и символ ω отсутствует в дереве достижимости, то сеть представляет систему конечных состояний. Это позволяет решить вопросы анализа простым перебором и проверкой конечного множества всех достижимых маркировок.

Сеть Петри является **сохраняющей**, если она не теряет и не порождает метки, а просто передвигает их. Свойство сохранения проверяется по дереву достижимости вычислением для каждой маркировки суммы меток. Если сумма одинакова для каждой достижимой маркировки, сеть - сохраняющая.

Задача **покрываемости** маркировки М маркировкой М’ сводится к поиску на дереве такой вершины х, состояние которой покрывает состояние М. Если такой вершины М(х) не существует, маркировка М не покрывается никакой достижимой маркировкой.

Таким образом, дерево достижимости можно использовать для решения задач безопасности, ограниченности, сохранения и покрываемости. К сожалению, в общем случае его нельзя использовать для решения задач достижимости и активности, а также для определения возможной последовательности запусков. Решение этих задач ограничено существованием символа ω. Символ ω означает потерю информации, конкретные количества меток отбрасываются, учитывается только существование их большого числа. Вместе с тем, в отдельных конкретных случаях дерево достижимости позволяет судить о свойствах достижимости и активности. Например, сеть, дерево достижимости которой содержит терминальную вершину, не активна. Аналогично искомая маркировка M’ в задаче достижимости может встретиться в дереве достижимости, что означает ее достижимость. Кроме того, если маркировка не покрывается некоторой вершиной дерева достижимости, то она недостижима.

1. **Матричный анализ сетей Петри**

Матричный подход основывается на представлении сети двумя матрицами Д- и Д+, представляющими входную и выходную функции сети. Каждая матрица имеет **m** строк (по одной на переход) и **n** столбцов (по одному на позицию). Определим матрицы Д-(j,i)=K(Pi,I(tj)) и Д+(j,i)=K(Pi,O(tj)). Д- описывает входы в переходы, Д+ - выходы из переходов, K – кратность позиции по входам и выходам.

Введём m - вектор e(j), содержащий нули везде, за исключением j-й компоненты. Переход tj представляется m - вектором e(j).

Тогда переход tj в маркировке М разрешён, если М≥e(j)Д-, а результат запуска перехода tj в маркировке М обозначим функцией δ(M,tj) определяющей новую маркировку М’:

δ(M,tj)=M - e(j) Д- + e(j) Д+ = M+ e(j)(- Д- + Д+) = M + e(j)Д,

где *Д = Д+ - Д-* - составная матрица изменений состояний сети.

Тогда для последовательности запусков переходов G={tj1,tj2,…,tjk}имеем:

δ(M,G) =δ(M, tj1, tj2,…, tjk) =

= M + e(j1)Д + e(j2)Д + … + e(jk)Д =

= M +[ e(j1) + e(j2) + … + e(jk)]Д = M + f(G)Д.

Вектор f(G) = e(j1)+e(j2) …..+e(jk) называется вектором запуска последовательности tj1,tj2,…,tjk. i-й элемент вектора f(G), f(G)i - это число запусков перехода tj в последовательности tj1,tj2,…,tjk. Вектор запусков, следовательно, является вектором с неотрицательными целыми компонентами.

Рассмотрим задачу сохранения при известном векторе взвешивания меток W. Если M0 - начальная маркировка, а M’ - произвольная достижимая маркировка, необходимо, чтобы M0W = M’W. Тогда существует последовательность запусков переходов G, которая переводит сеть из M0 в M’. Поэтому,

M’ = δ(M0,G) = M0 + f(G)Д.

Следовательно, M0W = M’W = (M0 + f(G) Д) W = M0W + f(G) ДW. Исходя из условия сохранения, имеем: f(G)ДW = 0.

Поскольку это должно быть верно для всех f(G), имеем ДW = 0.

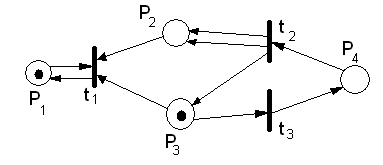
Таким образом, сеть Петри является сохраняющей тогда и только тогда, когда существует такой положительный вектор W, что ДW = 0. Это обеспечивает простой алгоритм проверки сохранения, а также позволяет получать вектор взвешивания W для сохраняющей сети. Для этого нужно решить систему Д W=0 относительно W.

Матричная теория является инструментом для решения проблемы достижимости. Пусть маркировка M’ достижима из M0, тогда существует последовательность (возможно пустая) запусков переходов G, которая приводит из M0 к M’. Это означает, что f(G) является неотрицательным целым решением следующего матричного уравнения для X:

M’ = M0 + X Д. (\*)

Следовательно, если M’ достижима из M0, тогда уравнение (\*) имеет решение в неотрицательных целых. Если уравнение не имеет решения, тогда M’ недостижима из M0.

Покажем возможности применения уравнения (\*) для анализа сети, приведённой на рис. 5.16.



|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |

Д- =

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 0 | 0 |
| 0 | 2 | 1 | 0 |
| 0 | 0 | 0 | 1 |

Д+=

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

Д = Д+ - Д- =

*Рис. 5.16. Пример сети и построения матрицы Д*

В начальной маркировке M0=(1,0,1,0) переход t3 разрешён и приводит к маркировке M’, где:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

M’=(1,0,1,0)+(0,0,1) =(1,0,1,0)+(0,0, 1,1)=(1,0,0,1)

Последовательность G={t­­3, t­­2, t­­3, t­­2, t­­1}представляется вектором запусков f(G)=(1,2,2) и получает маркировку M’:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

M’ = (1,0,1,0) + (1,2,2) =(1,0,1,0)+(0,3,-1,0)=(1,3,0,0)

Для определения того, является ли маркировка (1, 8, 0, 1) достижимой из маркировки (1, 0, 1, 0), имеем уравнение:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

(1, 8, 0, 1) = (1, 0, 1, 0) + X или

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

(0,8,-1,1)=X ,

которое имеет решение X=(0, 4, 5). Это соответствует последовательности G= {t3, t2, t3, t2, t3, t2, t3, t2, t3}.

Можно показать, что маркировка (1, 7, 0, 1) недостижима из маркировки (1, 0, 1, 0) поскольку матричное уравнение :

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

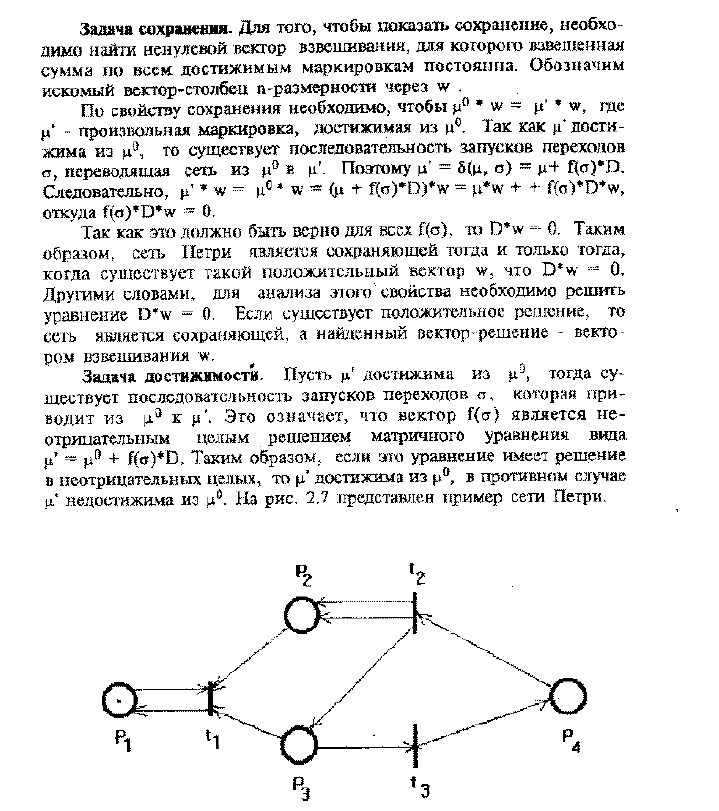
(1,7,0,1)=(1,0,1,0) + X или

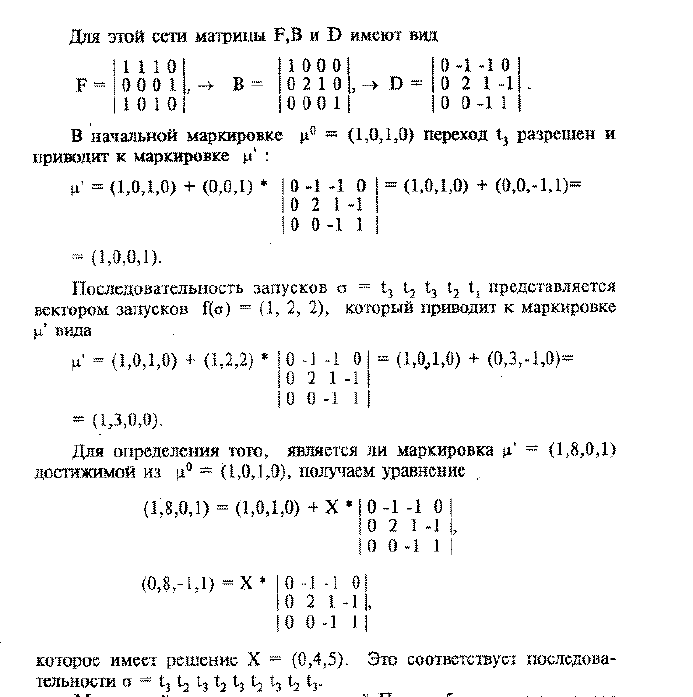
|  |  |  |  |
| --- | --- | --- | --- |
| 0 | -1 | -1 | 0 |
| 0 | +2 | +1 | -1 |
| 0 | 0 | -1 | +1 |

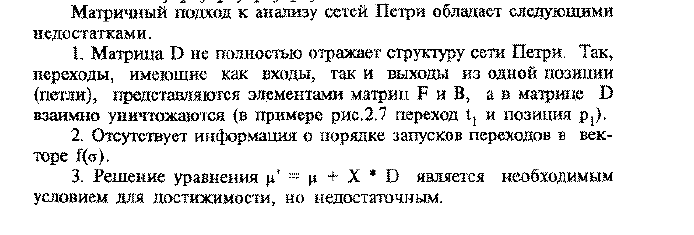
(0,7,-1,1)= X ,

не имеет решения.

1. **Матричный анализ сетей Петри. Задача достижимости Свойство сохранения**







1. **Кремниевые компиляторы**

Относятся к так называемым сквозным системам проектирования, реализующим тезис “от исходного описания к полному проектному решению”. Такие системы называ-ются системами автоматического проектирования. Доля невыполненных проектных опе-раций в автоматических системах проектирования не превышает 3-5%. Кремниевые ком-пиляторы применяются для проектирования топологии СБИС.

“Кремниевый” (название компилятора) обусловлено предметной областью.

“Компилятор” обусловлен наличием некоторых сходств между такими система-ми и программами-компиляторами.

Сходства:

1) Схожесть языков высокого уровня типа Module, Ado и языков кремние-вой компиляции.

2) Принципиальная возможность вмешательства программиста с целью ручной оптимизации объектного кода. Возможность проектировщика СБИС интерактивно корректировать с помощью графических средств план топологии слоёв кристалла с целью наиболее оптимального исполь-зования его площади.

Основу кремниевого компилятора составляет язык описания, на котором описыва-ется структура (архитектура) схемы и алгоритм её функционирования (поведения). Транслятор, переводящий исходное описание в проектное решение. Основными характе-ристиками объекта проектирования кремниевого компилятора являются: площадь кри-сталла и потребляемая мощность.

Отличие кремниевых компиляторов от систем подобного типа в том, что кремние-вый компилятор работает по принципу “сделать сразу хорошо”. Другие системы работа-ют по принципу “сделай и проверяй”. Проектные решения, полученные на различных этапах работы кремниевого компилятора, подвергаются верификации. Восстановление схемы по топологии не требуется. Для готовой микросхемы кремниевый компилятор ге-нерирует тесты для определения её статических и динамических параметров. В состав кремниевого компилятора входят следующие блоки:

Состав КК

Оптимизатор

Генератор управляющей части

Генератор операционной части

Экстрактор

Исходное описание языка компиляции

Различают архитектурные и поведенческие языки кремниевой компиляции. В ар-хитектурных языках описывается структура и функционирование схемы. Поведенческие языки, как правило, описывают функционирование схемы на довольно высоком языке спецификации, например, на уровне входных и выходных сигналов.

Экстрактор предназначен для извлечения из исходного описания информации, предназначенной для генерации операционной и управляющей частей.

Генератор операционной части, использующий библиотеку архитектурных реше-ний, генерирует структуру, обрабатывающую один бит. Затем как бы объединяются по-лученные схемные решения для обработки заданного слова информации. Активно ис-пользуется параметрический синтез.

Генератор управляющей части использует подходы, связанные с автоматическим проектированием управления на базе автоматов Мили и Мура. В основе работы многих кремниевых компиляторов положены функциональные заготовки в виде ПЛМ.

Достоинствами архитектурных компиляторов являются:

1) более широкий класс проектируемых устройств

2) возможность проектировщика вмешиваться в процесс проектирования, коррек-тировать проектные решения (в основном план топологии).

Недостатками являются:

1) детальное описание схемы и её функционирования

2) необходимость ориентации на библиотеку архитектурных решений

3) высокая профессиональная подготовка проектировщика

4) более длительный процесс получения проектного решения

Недостатками поведенческих компиляторов являются:

1) узкий класс проектируемых устройств

Достоинствами:

1) высокий уровень исходного описания без привязки к каким-либо определён-ным архитектурным решениям

2) быстрое получение проектного решения

Недостатки кремниевого компилятора:

1) довольно узкий класс законченных проектируемых устройств (контроллеры, память, магистральные чипы) , однако кремниевые компиляторы применяются для проектирования фрагментов сложных, в том числе и микропроцессорных СБИС

2) Высокая стоимость как программных, так и аппаратных средств

1. **PDM-, PLM-,ERP-системы**

PDM — «классика» систем управления продуктами

PDM (Product Data Management) — модуль, обеспечивающий управление комплексной информацией об изделии. Под последним термином могут пониматься разные объекты, в том числе, технически трудоемкие (суда, ракеты, сложные компьютерные сети). Среди ключевых функций системы стоит упомянуть управление:

* документацией об изделии — ее хранением, обработкой;
* инженерными и техническими данными, визуально-графическими и любыми иными сведениями, определяющими суть и особенности конкретных изделий;
* структурой продуктов, рабочими процессами;
* механизмом авторизации, автоматизации отчетности и так далее.

PDM-система дает возможность наладить взаимодействие между пользователями, контролировать большие потоки инженерно-технической информации, получать разграниченный доступ к данным на любой стадии разработки/изготовления изделий. Во многом, поэтому ее считают основной при выборе управленческого модуля.

PLM-системы — больший масштаб и широкая функциональность

PLM-системы, управляющие жизненными циклами продуктов в целом, предоставляют более «широкий» функционал и, собственно, включают в себя PDM. Управление изделиями — ключевой, но не единственный блок Product Lifecycle Management, и в разнице возможностей и состоит их принципиальное различие. PLM предоставляет много дополнительных «опций» — например, создание схем утилизации отходов производства — и несколько иной взгляд на бизнес. Как и чуть иные, но неизменно популярные ERM с их модулями управления финансами/персоналом/другими участками, они более комплексны, чем PDM.

В систему управления жизненными циклами продуктов включают модули:

* исследования рынка;
* проектирования, планирования, создания продуктов и рабочих процессов;
* закупки сырья, производства, проверки изделий;
* упаковки, хранения, продаж;
* технической и эксплуатационной поддержки;
* обеспечения взаимодействия между различными системами, интеграции их в общее информационное поле;
* утилизации и так далее.

Учет этапов цикла дает возможность предприятию комплексно уменьшать издержки производства, объединить все сложные процессы. Поэтому использование PLM-систем актуально для многооперационных предприятий в отрасли машиностроения, информационной сфере и так далее. Они помогут отслеживать каждый экземпляр или выпущенный продукт, учесть разнообразные требования. Если же нужно внедрить механизм управления изделиями в существующую среду или нет необходимости в масштабных комплексных решениях, можно ограничиться PDM-решением.

ERP-система (Enterprise Resource Planning) – это система управления ресурсами компании. Обычно их внедряют на крупных предприятиях со сложным производством, разветвленной филиальной сетью, большим ассортиментом выпускаемой продукции, повышенным объемом складских операций. Их главное достоинство заключается в том, что они позволяют объединить несколько задач: можно одновременно учитывать и планировать денежные средства, а также отслеживать их движение; формировать себестоимость и оценивать производительность труда на предприятии. Кроме того, все процессы становятся прозрачными. Она обеспечивает: 1. Объединение всех бизнес-процессов по единым правилам в рамках одной системы; 2. Оперативное получение руководством информации о всех сторонах деятельности предприятия; 3. Планирование и контроль деятельности организации (краткосрочные и долгосрочные планы различных подразделений увязываются между собой). В результате повышаются эффективность управления бизнесом и его конкурентоспособность. ERP-система удобна еще и тем, что ее можно внедрять частями (модулями), автоматизировав, например, сначала производство, а затем работу с персоналом. Набор модулей охватывает все направления деятельности, что позволяет автоматизировать практически все бизнес-процессы.

1. **Интеллектуальные САПР**

Подходы на этапах структурно-логического проектирования:

Верификация проекта

Синтез объекта

Проектирование на основе неформальных правил

Интуитивное проектирование

Объект проектирования

Выбор из библиотеки и формальный синтез

Выбор из библиотеки

Формальный синтез

*Формальный синтез* выполняется для конкретных задач. Сфера применения расширяется. *Выбор из библиотеки* и формальный синтез используется, когда не удаётся найти полный аналог из библиотеки. Пример: необходимо спроектировать суммирующий счётчик с коэффициентом пересчёта m. N-разрядный счётчик берётся из библиотеки, комбинационная схема, сбрасывающая его через m тактов, синтезируется формально. *Интуитивное проектирование* основано на опыте проектировщика в качестве программных средств, помогающих ему вести проектирование, используя системное моделирование. *Проектирование на основе неформальных правил*: в инструментальном компьютере собираются и накапливаются знания о возможных направлениях решения задач проектирования, затем эти знания используются для решения конкретной задачи, при этом используется система планирования, позволяющая находить требуемые решения на семантической сети определяющих соотношений. Реализуется технология экспертных систем.

Методы и средства искусственного интеллекта используются для задач синтеза и для верификации. Функцией системы является обоснование выбора правильности полученного технического решения. Интеллектуализация САПР связана с активным включением проектировщика в процесс проектирования на трудно формализуемых этапах проектирования. Это требует разработки адаптируемого к использованию человеко-машинного интерфейса; невозможность построения строгой математической модели предметной области обусловлена наличием условий неопределённости, появление которых связано со следующими причинами:

* + неопределённость некоторых проектных переменных и условий функционирования объекта на начальных этапах проектирования
  + отсутствие возможности количественного описания некоторых параметров объекта, ограничений и требований на проектирование, наличие субъективной оценки лиц, проводящих словесное описание значений параметров проектируемого объекта
  + невозможность учёта влияния или определения априорных оценок некоторых переменных или последствий выполняемых действий, возникающих по различным причинам
  + многокритериальность и большая размерность задач проектирования СВТ, невозможность задать целевую функцию проектирования в явном виде. Указанные причины затрудняют разработку алгоритмов проектирования на основе традиционных методов (например, теории графов, множеств). Переход к многокритериальным задачам в силу больших затрат распространения не получил. Таким образом, повышение эффективности алгоритмической базы САПР достигается благодаря её специализации. Это требует времени и материальных затрат на разработку и доработку алгоритмов САПР при смене класса объекта проектирования, повышения требований к квалификации проектировщика, способного учитывать особенности и правильно выбрать алгоритм

Характер взаимодействия проектировщика в интеллектуальных САПР должен соответствовать формам его традиционной деятельности при ручном проектировании. Таким образом, человеко-машинный интерфейс в таких системах основан на естественных языковых и графических средствах взаимодействия. Этот интерактивно-алгоритмический метод проектирования обеспечивает:

* + постоянное отображение хода процесса проектирования в форме динамически изменяющихся графических моделей разрабатываемого объекта или текстовых комментариев в системе
  + возможность перехода от ручного проектирования к автоматизированному и обратно в моменты определения пользователем
  + использование многоуровневых структурных моделей, возможность оперативного перехода от одного уровня к другому. Интеллектуальный интерфейс в САПР строится с использованием формализованных профессиональных знаний специалистов и обеспечивает возможность автоматизации творческой работы проектировщика и имитации процесса его логических рассуждений при решении слабоструктурированных проектных задач

Методы искусственного интеллекта позволяют организовать естественно-языковые взаимодействия пользователя и САПР, вести обучение на различных этапах проектирования.

Интеллектуализация САПР решает задачи адаптации компонентов системы к проектной ситуации и пользователю.

Одним из перспективных направлений интеллектуализации является интеграция инвариантной (проблемно-ориентированной) экспертной системы и проектной САПР.

Типовая структура интегрированной интеллектуальной САПР:

Интерфейс

естественно- графи-

языковой ческий

графическая интерпретация

проектного решения

Система представления знаний

Средства логического вывода

Лингвистический процессор

Базы данных

Средства визуализации и редактирования

Инструментальные средства проектирования

Система представления знаний содержит исходную обобщённую структуру знаний, то есть модель предметной области, включающую классификационную схему семантических единиц предметной области(понятий и отношений) , множество фактов, множество правил проектирования и совокупность метаправил(управляющих структур) , которые определяют порядок и способ обработки знаний в конкретных проектных ситуациях. Средство логического вывода на основе хранимых инженерных знаний и поступающих данных о проектном решении формирует множество рекомендаций пользователю, производит оценку свойств проекта и позволяет прогнозировать процесс проектирования. Указанные средства работают не с полной моделью предметной области, а с отдельными фрагментами в зависимости от характера решаемой задачи. Таким образом, важной функцией средств логического вывода является компиляция знаний, преобразование из одного формата внутреннего представления в другой, с помощью которого возможно более эффективно решать проектную задачу.