Лекция 1

Проектирование – это разработка технической документации изделия, согласно функциональным требованиям..

В ТЗ описываются функции, которые должен выполнять объект. После этого описывается назначение, область применения и т.д.

Автоматизация проектирования – это использование ЭВМ при проектировании изделия. Такой способ называется человеко-механическим.

Автоматическое проектирования – это проектирование без участия (использования) людей.

Автоматическое проектирование не используется, потому что технология производства ЭВМ очень быстро развивается.

Первая автоматизированная система была создана в 1956 году (Крей). Фазы:

1) Анализ логич. уровней на предмет лог. ошибок, документированных ошибок, ошибок синхронизации и синтаксиса. Контролировала нагрузки на входах и выходах для каждой схемы. Сортировала булевые уравнения и выдавала результат. Проверяла правильность исходных данных.

2)На панели для моделирования можно было вводить значения для булевых уравнений и при выполнении наблюдать ход решения.

3)Выполняла компоновку элементов логической структуры по конструкторским узлам, рассчитывала связь шасси и элементом, выдавала списки расстояний между точками источника и приемниками, определяла цвета проводов, по которым шли сигналы и длинны проводников.

Задача проектирования достаточно сложная. Ограничений в этом процессе нет.

Подходы к проектированию:

1. Деление по времени выполнения работ.
2. Деление по учету (по характеру учета документации).
3. Блочно-иерархичный способ деления.

Деление по времени выполнения работ. Этапы проектирования:

- научно-исследовательская работа (Её результат- предложение для исп. полученных результатов);
- эскизное проектирование(определяет какими методами, элементами будет выполнено проэкт. , делается только эскизный проект без конкретной реализации);
- техническое проектирование (рабочее);
- испытания (по результатам вносятся изменения).

Деление по учету (вертикальное). Этапы проектирования:

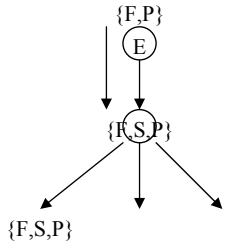
- функциональное проектирование;
- алгоритмическое проектирование;
- конструкторское проектирование;
- технологическое проектирование.

Функциональное	Алгоритмическое	Конструкторское	Технологическое
Разработка структурных и функциональных схем; определение особенностей структур, принципов функционирования, важнейших параметров.	Разработка алгоритмов; выбор системы команд, правил пересылки; обработка информации в системе.	Реализация разработанных схем физическими элементами	Определение технологии производства. Разработка технологических схем, карт и т.д.
Системный уровень; функционально-логический; схемотехнический; конструкторский; компонентный.	Программный: системы модулей проектирования микропрограмм	Технический элемент замены; шкаф-стойка; панель; модуль; кристалл; ячейка.	Принципиальная схема технологического процесса; маршрутная технология; технологические операции.

При проектировании задаются функции {F} и параметры {P}. Для выполнения функций используются элементы {E}. Функции {F} этих элементов и структура {S}, которая объединит элементы с некоторыми параметрами {P}.

На этапе системного уровня это процессы, каналы, переф. устройства…

F- ф-ное описание . S- структурное опис. P- параметрическое описание.



Такое разделение идет пока не встретится неделимый элемент.

На ф-льно логическом уровне это конкретные элементы.

На схемо-техническом уровне это конкретные элементы объединённые в некоторую структуру.

Структура –эл-нт и связи между ними. $S=\{E,\psi\}$

Лекция №2

Система – совокупность элементов объединенных в одно целое для достижения определенных целей.

Цель – множество результатов определяемых назначением системы.

Понятием элемента системы определяется уровнем представления системы. Самый верхний уровень представления системы – системный, следующий уровень – функционально-логический, последний – конструкторский. На каждом уровне представления система описывается своим элементом.

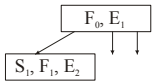
На уровни можно поделить с помощью уровней отрыва, методами описания (проектирования).

Методика проектирования основывается на цифровых автоматах, логических элементах.

Любая система описывается $C=\{F, S, P\}$

F – множество функций, на реализацию которых нацелена система. Может быть задано математически либо словесно.

S – описание структуры, которая реализуется элементами и связями.



$S = \{E, \psi\}$ множество элементарных объектов и связи между ними.

Различают равные (полностью одинаковые) и эквивалентные структуры.

Эквивалентные структуры реализуют одни и те же функции, но состоят из разных элементов или связей.

P – множество значений параметров системы в целом и по элементам, которые обеспечивают реализацию функционального описания и ограничений заданной циклической структуры.

Свойства сложной вычислительной системы:

- Многообразие функций, реализуемых системой
- Сильная взаимосвязь этих функций
- Иерархическое представление системы в целом

В результате того, что сложная система представляется иерархией, то существует два подхода к процессу представления:

⇒ проектирование «сверху вниз»

⇒ проектирование «снизу вверх»

В первом случае исходными являются функции системы в целом и множество элементов, из которых может быть составлена заданная система.

После синтеза структуры отделяются элементы, существующие и те которые необходимо проектировать, то есть выполнить переход к следующему уровню проектирования.

Проектирование «снизу вверх» – с элементов и функций нижнего уровня необходимо построить систему, чтобы выполнять функции следующего уровня.

При проектировании вычислительной системы метод «снизу вверх» нерационален.

Проектирование выполняется в два этапа:

1. задача синтеза

2. задача анализа

Под синтезом понимается построение системы по заданному закону функционирования.

Анализ – определение законов функционального описания по заданному структурному описанию системы и по параметрическому описанию её элементов.

Для решения задачи синтеза нужно получить $F,\ E \rightarrow S,\ P$

Задача анализа обратна задачи синтеза $S,\ P_э \rightarrow F,\ P$

Задача синтеза первична, потому что первичный анализ выполнен на предварительных этапах. На синтез любого объекта накладывается множество ограничений:

1) связанные с методом решения задачи и охватывающие такие вопросы как наличие знаний, сроки и имеющиеся в расположении технические средства проектирования;
2) связанные с требованиями ТЗ, назначением параметров объекта;
3) формируется физическими принципами реализации законов функционирования и получении его предельно желаемых характеристик.

Дополнительные ограничения накладываются способами и формами взаимодействия объектов с внешней средой и методами оптимизации взаимодействия человека с проектируемым объектом в процессе функционирования объекта и его эксплуатации.

Задача синтеза называется алгоритмически разрешимой, если существуют формальные методы (правила) получения структурного описания по заданному функциональному описанию, в противном случае задача не разрешима.

Алгоритмические методы решаются либо ручными методами проектирования, либо методами полного перебора вариантов.

Проектирование всех вариантов может быть сокращено при помощи методов последовательного приближения.

Метод последовательного приближения.

Синтезируется некоторый вариант системы и выполняется её анализ. Выбираются те параметры, которые имеют наихудшие значения. На следующем этапе улучшается предыдущее, чтобы худшие параметры улучшить.

Задачи синтеза

	Структурный	Параметрический
Цель	Определение структуры проектируемого элемента	Определение значений параметров полученной структуры

Задача улучшения значений – задача оптимизации. Задача структурной оптимизации – улучшение структуры, параметрической оптимизации – улучшение значений параметров.

Задача синтеза на этапе системного проектирования решается вручную, либо методом последовательных приближений. На этапе функционально-логическом – теория цифровых автоматов (булевая алгебра). На этапе конструкторского проектирования – теория множеств и теория графов.

Задача анализа связанна с оценкой полученных решений, состоит в определении функций и показателей качества для объекта с заданной структурой.

Методы решения задачи (моделирование):

– математический – связанно с решением системы уравнений
– алгоритмический – связанный с некоторой …

Математические модели:

° функциональные – информационные процессы, происходящие в моделируемом объекте;
° структурные – отображаются структурой объекта, представляется в виде графа, матриц элементов.

Различают полные и макромодели.

Полная математическая модель предполагает описание всех элементов на самом верхнем уровне.

Макромодель предполагает описание системы с помощью моделей элементов более высокого уровня.

Вначале используется наименее адекватная модель.

Имитационная модель:

– вероятностная модель, использующая СМО, сети Петри и т.д.
– любая модель

Лекция №3

Любая модель представляется $I=F(x,Q)$, где

x – множество внутренних параметров

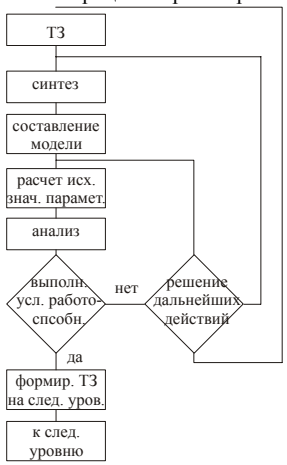
Q – множество внешних параметров

Если уравнение задано в явном виде, то модель является аналитической, если в виде алгоритма функционирования объекта то – алгоритмическая модель.

Существует два вида анализа: ●одновариантный; ●многовариантный.

Одновариантный анализ заключается в исследовании свойств объекта в заданной точке пространства параметра (анализ переходных процессов, устойчивости и т.д.). Многовариантный анализ – исследование свойства объекта в окрестностях заданной точки пространства (статистический анализ, анализ чувствительности).

Схема процесса проектирования на каждом этапе проектирования.



Исходными данными является ТЗ, которое формируется на предыдущем этапе (либо до начала проектирования).

Задача синтеза – задача оптимизации, если мы хотим улучшить полученный результат.

При постановке задачи оптимизации, необходимо преобразовать физическое представления о назначении и степени полезности объекта в математическую формулировку экстремальной задачи, то есть сформулировать цели оптимизации и формировать понятие оптимальности.

Цель оптимизации выражается в критериях оптимизации.

Критерий – правила предпочтения сравниваемых вариантов.

$Y=F(x)$ – целевая функция оптимизации

x – вектор всех параметров, значения которых могут изменятся

$X=(x_1 \dots x_n)$ – фиксация значений, определяет каждый вариант проектируемого объекта.

В вектор входит часть внутренних параметров (те параметры, которые имеют для нас значение), остальная часть берется либо произвольно, либо учитывается в самом описании функции.

Целевая функция определяется таким образом, чтобы по её значениям можно было определить лучший вариант из множества возможных вариантов.

Значения задаются либо в виде равенств $\varphi(x)=0$, либо неравенств $\psi(x)\geq 0,\ \beta_i \leq x_i \leq \alpha_i$

Если все значения заданы в идее равенств, то задача называется безусловной, иначе – условной.

$\underset{x\in xD}{extr} F(x),\ где\ xD=\{x|,\ \varphi(x)=0,\ \psi(x)\leq 0$ – задача мат. программирования

Если φ,ψ – линейные, то задача линейного программирования, если хотя бы одна нелинейная, то задача нелинейного программирования.

Если значения вектора дискретные, то задача дискретного программирования, и частично не дискретная если значения …

Дискретное программирование называется целочисленным, если x принимает целые значения.

Если $\boldsymbol{xD}$ область булевых переменных, то задача бивалентного программирования.

Задача структурной оптимизации $S=\{E,\ \psi\}$

Под оптимальным понимается такой вариант структуры параметров, который удовлетворяет всем системным, конструктивным, технологическим, электрическим, экономическим требованиям ТЗ, а критерии оптимальности принимают экстремальные значения.

При этом при решении задач структурной оптимизации обычно оперируют независимыми переменными, а функции $\varphi(x),\ \psi(x)$ как правило нелинейно зависят от самих элементов x.

$\underset{x\in xD}{extr} F(x)=F(x_1 \dots x_m),\ где\ xD=\{x|,\ \varphi(x)=0,\ \psi(x)\geq 0 \quad \varphi(x),\psi(x)\in G$

Выбирается один критерий, который полностью характеризует объект – такой критерий называется частным.

При проектировании сложных объектов такой критерий найти невозможно. Функция, которая позволяет объединить разные критерии, называется обобщенной.

Для решения однокритериальных задач разрабатывается множество математических аппаратов.

Многокритериальные задачи – задачи векторной оптимизации, все известные методы решения которой непосредственно или косвенно сводят к задачам скалярной оптимизации.
Задачи скалярной оптимизации – частные критерии объединены в некоторый составной критерий в котором целевая функция $F(x)=\Phi(F_1(x)\dots F_m(x))$
Если можно найти объективную взаимосвязь между F_1 и F_m, то решение задачи является объективным.
В качестве обобщенного параметра выбирают формально объединенное множество критериев позволяющих достаточно просто оценить исследуемый объект.
В зависимости от используемого правила различают аддитивные критерии, мультипликативные критерии и минимаксные или максиминые.
Если оптимизация выполняется из учета статистического разброса значений параметров, то соответствующий критерий называют детерминированным. Если разброс параметров учитывается, то такой критерий называется статистическим. Они наиболее полно характеризуют, но их использование ведет к увеличению затрат машинного времени. Поэтому, они используются при решении задач верхнего уровня иерархии.

Лекция №4
Частные критерии
Частный критерий – некоторый один параметр, который характеризует качество объекта при ограничениях на все остальные.
$\underset{x\in xD}{extr}\, F(x),\ где\ \text{xD}=\{\text{x}|,\ \varphi(\text{x})>0,\ \psi(\text{x})=0$
Если все ограничения заданы только равенством, то функция изменения параметра приводит к изменению всей функции и запасов вариаций задача не имеет.
Обобщенные критерии (интегральные).
Для их определения используются математические приёмы (аддитивный, мультипликативный, минимаксный критерии), которые позволяют определять значения общего параметра.
Необходимо иметь один критерий, по которому сравниваются критерии всех систем, и среди них выбирается наилучший. Такой критерий называется обобщенным (интегральным).
Аддитивный критерий.
Целевая функция – путем сложения нормированных значений частных критериев. Нормированные значения представляют собой отношение реального значения нормального критерия к некоторой нормирующей величине, измеряющейся в тех же единицах, что приводит к безразмерной величине данного критерия.
1) В качестве делителей выбираем директивные значения параметров, заданное заказчиком.
Недостаток: заданная в ТЗ величина рассматривается как образцовая, что не всегда полезно.
2) Выбираем экстремальные значения критериев, достигаемых в области существующих проектных решений.
3) Выбираем разность между минимальным и максимальным параметром в области существующих проектных решений.
Выбор варианта является субъективным решением
$$F(x)=\sum_{i=1}^{n}\frac{F_i(x)}{F_i^H(x)}=\sum_{i=1}^{n}f_i(x)$$
Необходимо учитывать важность параметров, добавим коэффициент $F(x)=\sum_{i=1}^{n}c_i f_i(x)$
$F_i^H(x)$ – нормирующий делитель от частного коэффициента.
Недостаток такого критерия:
– нет логической взаимосвязи между оцениваемыми параметрами
– при вычислении данного критерия может происходить взаимная компенсация значений параметров
Ограничения на экстремальные значения параметров и коэффициентов значимости – является машинным приемом для точного определения критерия.
Мультипликативный критерий
Используется принцип справедливой компенсации абсолютных значений нормированных частных критериев. В целом ряде задач используется не абсолютное значение, а принцип относительной компенсации, то есть справедливым следует считать такой компонент, когда суммарный уровень относительного снижения значений одного или нескольких критериев не превышает суммарного уровня относительного увеличения значений других критериев.
$\sum_{i=1}^{n}\frac{\Delta F_i(x)}{F_i(x)}=0$, где n – количество параметров, $F_i(x)$ – начальное значение частного коэффициента,
$\Delta F_i(x)$ –изменение значений в новом варианте.
С учетом коэффициента значимости $\sum_{i=1}^{n}C_i\frac{\Delta F_i(x)}{F_i(x)}=0$
С учетом $\Delta F_i(x)$ $f_i(x)$, то формулу можно преобразовать
$$\sum_{i=1}^{n}\frac{\Delta F_i(x)}{F_i(x)}=\sum_{i=1}^{n}d\left(\ln F_i(x)\right)=d\left(\ln\prod_{i=1}^{n}F_i(x)\right)=0$$
Это всё сложно реализовать, поэтому необходимо варьировать параметрами в произведении.
$F_i(x)=\prod_{i=1}^{n}F_i(x)$ – критерий является мультипликативным.
С учетом коэффициента значимости используют $F_i(x)=\prod_{i=1}^{n}C_iF_i(x)$
Недостаток тот же что и в аддитивном критерии.
Максиминный критерий
Критерий основан на принципе компромиссности, используется идея равномерности.
Необходимо найти такой вариант, у которого все относительные значения всех критериев будут одинаковые. $f_i(x)=k\ \ i=\overline{1,n}\ x=\{x_1\dots x_n\}\ \ k=const$ или с учетом значимости $c_if_i(x)=k$
Необходимо найти такую совокупность значений параметров, чтобы целевая функция принимала наихудшие значения параметров, чтобы в общем функция принимала максимальные значения.
$F(x)=\max_{F(x)}\min_{i}f_i(x)\ \ i=\overline{1,n}\ \ \ x=\{x_1\dots x_n\}$
Ещё называется принципом гарантированного результата
$F(x)=\min_{F(x)}\max_{i}f_i(x)$ – минимаксный выбор
При учете значимости $F(x)=\min_{F(x)}\max_{i}\left(f_i(x)c_i\right)$
Наиболее часто используемые методы нахождения экспертных оценок: ●метод ранжирования; ●метод присвоения баллов.
Метод ранжирования: собирается l экспертов, им предлагается расставить n критериев по рангу, причем самый важный принимает n-ый ранг, а наименее важный – 1-ый ранг.
Ранг каждого элемента определяется
$C_i=\frac{\sum_{k=1}^{l}r_i^k}{\sum_{k=1}^{l}\sum_{i=1}^{n}r_i}$, где r – ранг, а C_i – значимость параметра, r_i^k – ранг i–го критерия выставленный k–ым экспертом.
Метод приписывания баллов. Также эксперты проставляют баллы от 0 до 10. Несколько параметров могут иметь одинаковые баллы и могут использовать дробные числа.
H_i^k – балл i–го элемента выставленный k–ым экспертом.
$H_i^k=\frac{h_i^k}{\sum_{i=1}^{n}h_i^k}$, $\sum_{i=1}^{n}h_i^k$ – сумма балов выставленных k–ым экспертом всем элементам
$$C_i=\frac{\sum_{k=1}^{l}H_i^k}{\sum_{k=1}^{l}\sum_{i=1}^{n}H_i^k}$$

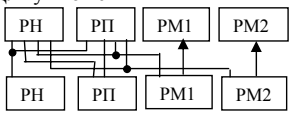
Для более точной характеристики необходимо учитывать значимость (компетентность) эксперта.
μ – коэффициент компетентности

$$C_i = \frac{\sum_{k=1}^{l} \mu_k H_i^k}{\sum_{k=1}^{l}\sum_{i=1}^{n} \mu_k H_i^k}$$

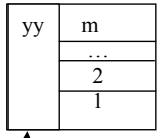
	P1	P2			Pk			Pn	Ад	М	min/max	если не устраивает
Вар. 1	25	23						25				вар. 1 создается
Вар. 2												вар. 2

М-мультипликативность.
Ад- адитивность.

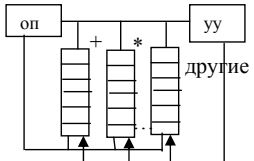
Создаются обобщенные характеристики для определения наилучшего варианта, если из множества параметров не выбирается оптимальный.
Мультипл. – параметрическая оптимизация , когда приращ. имеет большее значение чем абсолютное значение.
Аддитивный – при важности абсолютного параметра(большое значение имеет абсолютное значение).
Минимаксный – задает структурную и параметрическую оптимизацию при нахождении уравненных значений (гарантированный результат).
Для умножения



РМ2- регистр множимого 2



универсальный



способ попороще чем универс.

Особенность конвеерного процессора заключается в том, что распространение осуществляется как в пространстве так и во времени. Таким образом конвеер максимально исп. частотные особенности элементной базы.

Лекции 5-7
САПР
Виды обеспечения САПР
Все определения, приведены ниже, даны в соответствии с существующими ГОСТ и стандартами по АП.
САПР – организационно-техническая система, состоящая из комплекса средств автоматизации проектирования, взаимосвязанного с необходимыми подразделениями проектной организации или коллективом специалистов и выполняющая АП.
Требования к составу и структуре ТС формируются из:
1. Общих требований к структуре САПР.
2. Эффективного решения выделенного класса задач проектирования.
3. Активного включения пользователя в процесс проектирования.
4. Возможности работ с графическим материалом; включая процессы ввода и обработки и вывода информации.

ТО САПР называется совокупность взаимосвязанных и взаимодействующих технических средств, предназначенных для выполнения АП.
Техническое обеспечение САПР
Включает в себя технические средства, с помощью которых решаются задачи проектирования. К ТО относятся устройства вычислительной и организационной технике, средства передачи данных, измерительные и др. устройства.
К ТО САПР предъявляются следующие требования:
1. Удобство пользования, возможность оперативного взаимодействия инженеров с ЭВМ.
2. Достаточная производительность и объем оперативной памяти для решения задач всех этапов проектирования за приемлемое время.
3. Возможность одновременной работы с ТС необходимого количества пользователей для эффективной деятельности всего коллектива разработчиков.
4. Открытость комплекса ТС для решения и модернизации системы по мере прогресса техники.
5. Высокая надежность, приемлемая стоимость.

Удовлетворение перечисленных требований возможно только в условиях организации ТО в виде специализированной ВС, допускающей функционирование в нескольких режимах. Такое ТО называется комплексом ТС САПР.
Математическое обеспечение САПР
МО САПР состоит из математических моделей объектов проектирования, методов и алгоритмов выполнения проектных операций и процедур. (синтез структуры, получение математических моделей, одновариантный и многовариантный анализ, параметрическая оптимизация).
В МО САПР можно выделить специальную часть: в значительной мере отражающую специфику объекта проектирования, физические и информационные особенности его функционирования и тесно привязанную к конкретным иерархическим уровням; и инвариантную часть: включает в себя методы и алгоритмы, слабосвязанные с особенностями математических моделей и используемых на многих иерархических уровнях (методы и алгоритмы многовариантного анализа и параметрической оптимизации).
Требования к МО САПР:
Свойства МО оказывают существенное, а иногда и определяющее влияние на возможности и показателями САПР.
Универсальность. Под универсальности МО понимается его применимость к широкому классу проектируемых объектов. Одно из отличий расчетных методов САПР от ручных высокая степень универсальности.
Высокая степень универсальности нужна для того, чтобы САПР была применима к любым или большинству объектов проектируемых на предприятии.
Степень универсальности не имеет количественной оценки. Реализуя ту или иную модель или метод, разработчик МО должен указать четкие границы их применимости.
Алгоритмическая надежность. Свойство компонента МО давать при его применении верные результаты называется алгоритмической надежностью.
К сожалению не всегда условие применимости моделей и методов могут быть найдены, наследованы и сформулированы в виде конкретных инструкций пользователем. Часто применимость, компонента но зависит от конкретных условий, которые не всегда поддаются исчерпывающему учету и классификации. Методы и алгоритмы, не имеющее строгого обоснования, называются эвристическими.
Отсутствие четко сформулированных условий применимости приводит к тому, что эвристические методы могут использоваться некорректно, что приведет к далекому от истинного решению, либо решение вообще не будет получено (в следствии отсутствия сходимости). При этом может не оказаться данных, позволяющих обнаружить некорректность решения. Следовательно, возможна ситуация, когда неверное решение будет использоваться дальше.
Степень универсальности характеризуется заранее оговоренными ограничениями, а алгоритмическая надежность – ограничениями, заранее не оговоренными.
Количественной оценкой алгоритмической надежности служит вероятность получения правильных результатов при соблюдении оговоренных ограничений на применение метода. Если эта вероятность равна 1 или близка к ней – метод алгоритмически надежен.
Применение алгоритмически ненадежных методов нежелательно хотя и допустимо в случаях, когда неправильные результаты легко распознаются. (конструкторское проектирование)
Обусловленность математических моделей и задач. Это свойство не увеличивать погрешности. О плохой обусловленности говорят, когда малые погрешности походных данных приводят к большим погрешностям результатов.
На каждом этапе выполнений промежуточные исходные данные и результаты, свои источники погрешностей. При плохой обусловленности погрешности могут резко возрасти, что может привести как к снижению точности, так и к росту затрат машинного времени. Для анализа и оптимизации объектов с плохо обусловленными математическими моделями необходимо применять специальные методы с повышенной алгоритмической надежностью.
Точность. Степень совпадения расчетных и истинных результатов алгоритмически надежные методы могут давать различную точность решения.

В большинстве случаев решения проектных задач характеризуются:
Совместимым использованием многих компонентов но, что затрудняет определение вклада в общую погрешность каждого из них;
Вектором характером результатов, т.е. результатом решения является значение не отдельного параметра, а многих параметров.
В связи с этим оценка точности производится с помощью специальных вычислительных экспериментов, в которых создаются условия для раздельной оценки погрешностей, вносимых математическими моделями экспериментов, алгоритмами анализа и оптимизации. В этих экспериментах используются специальные задачи, называемые тестовыми.
Количественная оценка решения тестовой задачи если одна из норм вектора относительных погрешностей.
Относительная погрешность компонента ε оценивается по совокупности учитываемых параметров одной из норм вектора $\varepsilon=(\varepsilon_1,\varepsilon_2,\ldots\varepsilon_m)$, где $\varepsilon_j = \frac{\left(y_j^p - y_j\right)}{y_j}$, m – норма. $\varepsilon_{\max} = \max_{j\in[1\ldots m]}\left|\varepsilon_j\right|$ или e – норма $\varepsilon_{общ} = -\sqrt{\sum_{j=1}^{m}\varepsilon_j^2}$, где ε_j – относительная погрешность j-го элемента вектора результатов, m – размерность вектора.
Затраты машинного времени. Универсальные модели и методы характеризуются сравнительно большим объемом вычислений, растущим с увеличением размерности задач. Поэтому при решении большинства задач в САПР затрата машинного времени Тм значительны. Обычно именно Тм является главным ограничивающим фактором при попытке повысить сложность проектируемых на ЭВМ объектов. Поэтому минимизация Тм – одно из основных требований к МО САПР.
При использовании в САПР многопроцессорных ВС уменьшить время счета можно с помощью параллельных вычислений. В связи с этим одним из показателей экономичности МО является его приспособленность к распараллеливанию вычислительного процесса.
Требование высокой степени универсальности, алгоритмической надежности, точности с одной стороны и малых затрат машинного времени, с другой стороны, противоречивы. Поэтому любой конкретный компонент МО, отражая определенный компромисс этих требований, может быть эффективным при решении одной задачи и малоэффективным при решении другой. Поэтому в САПР целесообразно иметь библиотеки с наборами моделей и методов. Каждый компонент МО определенного целевого назначения должен быть представлен несколькими разновидностями, обеспечивающими разную степень удовлетворения противоречивых требований.
Затраты памяти. Являются вторым после Тм показателем экономичности МО. Они определяются длиной программы и объемом используемых массивов данных. Несмотря на значительное увеличение емкости оперативной памяти в современных ЭВМ, требование экономичности по ее затратам остается актуальным.
Для разрешения возникающих трудностей можно использовать внешнюю память, но частые обмены данными между внешней и оперативной памятью приводят к значительному увеличению Тм. Поэтому при больших объемах программ и массивов обрабатываемой информации целесообразно использовать МО, допускающее построение оверлейных программных структур и реализующие принципы диакоптической обработки информации.
Значительное влияние на развитие МО САПР оказало стремление повысить экономичность используемых моделей и алгоритмов, имеющих частный характер, так и совершенствованием общих принципов создание МО, эффективного по затратам Тм и памяти. К таким принципам относятся учет разреженности матриц, исследование сложных систем по частям (диакоптические методы исследования), макромоделирование, событийность анализа и разноканальное использование способностей человека в интерактивных процедурах.
Учет разреженности матриц основан на хранении в ЭВМ ненулевых элементов матриц и выполнении арифметических операций только над ними.
Диакоптика основана на использовании структурных особенностей схем, а не на принятии каких-либо упрощающих допущений. При этом, производится расчленение математических моделей на части, исследуемые самостоятельно. Это позволяет упорядочить и минимизировать количество обменов информацией между оперативной и внешней памятью при анализе сложных систем, а также выбрать для исследования каждой части наиболее выгодные режимы.

Макромоделирование лежит в основе направления, связанного с рациональным выбором математических моделей элементов при построении математической модели системы. Оно реализует возможность использования при анализе одного и того же объекта нескольких моделей, различающихся сложностью, точностью и полнотой отображения свойств объекта, трудоемкостью требующихся вычислений и т.д. каждая из моделей соответствует определенному разбиению системы на элементы и выбору определенных моделей полученных элементов. Наиболее детальное разбиение в рамках данного иерархического уровня приводит к получению полной математической модели, характеризующейся высокой точностью и большим объемом требуемых вычислений. Повысить экономичность можно делением системы на более крупные блоки с использованием для них упрощенных математических моделей – макромоделей.
Событийность анализа заключается в том, что при имитации процессов, протекающих в исследуемом объекте во времени, вычисления проводятся только для тех элементов, состояния которых на очередном временном шаге может измениться.
Рациональное использование эвристических способностей человека в интерактивных процедурах позволяет человеку вмешиваться в ход вычислений и выбирать наиболее перспективные продолжения на основе эвристических оценок.
Инвариантные средства МО. Включают методы и алгоритмы, слабо связанные с особенностями математических моделей проектируемых объектов и используемые на многих иерархических уровнях. Это прежде всего методы и алгоритмы оптимизации, рассмотренные выше, многовариантный анализ и логикокомбинаторные методы решения задач.
Основными видами многовариантного анализа в задачах проектирования являются анализы чувствительности и статистический.
Анализ чувствительности. Цель анализа чувствительности – определение коэффициента чувствительности, называемый также коэффициентом влияния:

$$a_{ji} = \frac{\partial y_j}{\partial y_i}; i = \overline{1,n}, j = \overline{1,m} \qquad b_{ji} = \frac{a_{ji} x_{iном.}}{y_{jпот.}}$$

где a_{ji} и b_{ji} – абсолютный и относительный коэффициент чувствительности выходного параметра y_j к изменениям внутреннего параметра x_i. $x_{iном.}$ и $y_{jном}$ – номинальные значения y_j и x_i.
Результаты анализа чувствительности m выходных параметров к изменениям и внутренних параметров представляют собой mn коэффициент чувствительности, составляющие матрицу a_{ji} или b_{ji} . Если исследуется влияние временных параметров, то вместо x_i и $x_{iном}$ фигурирует временный параметр q_j и его номинальное значение.
Анализ чувствительности применяется, если X и Q можно считать непрерывными величинами, а y_j являются дифференциальными функциями своих аргументов x_i и q_i.
Результаты анализа чувствительности используются при решении таких задач как параметрическая оптимизация, расчет допусков, оценка точности выходных параметров. Именно по значениям a_{ji} и b_{ji} определяются параметры, существенно влияющее от несущественных, определяют направление изменений внутренних параметров для улучшения выходных параметров X и Q для выполнения точностных требований к параметрам У.
Основными методами анализа чувствительности являются методы приращений, прямой и вариационный методы.
Основным достоинством метода приращений являются его универсальности, а также возможность распараллеливания вычислительного процесса, простоту программной реализации.
Недостатками являются невысокая точность, сравнительно большая трудоемкость вычислений.
Метод приращений – метод численного дифференцирования зависимости У=F(X,Q). Алгоритм метода приращений включает в себя (n+1) – краткое обращение к модели для выполнения У, где n – количество варьируемых параметров. В этом варианте задаются номинальные значения аргументов и, следовательно, результатом обращения к модели будет $У_{ном.}=(У_{1ном.},\ldots У_{mном.})$, т.е. номинальное значение вектора У. В очередном (i+1) варианте среди оставшихся n вариантов задается отклонение ΔXi от $X_{iном.}$ Только одному из варьируемых параметров. В результате выполнения (i+1) варианта получают для вектора У значения $У_i=(y_{1i},y_{2i},\ldots y_{mi})$ по которому оценивается очередной столбец матрицы абсолютной чувствительности $A_i=(Y_i-Y_{ном.})\Delta X_i$. Любой из найденных коэффициентов A_i легко пересчитать в B_i.
Статистический анализ.

Целью статистического анализа является получение оценок рассеяния выходных параметров У и вероятностей выполнения заданных условий работоспособности для проектируемого объекта.
В случае объектов СМО сами выходные параметры имеют вероятностный смысл, когда цель статистического анализа – расчет таких параметров. Причинами рассевания выходных параметров У являются нестабильность внешних параметров Q и случайный характер внутренних параметров Х.
Результатами статистического анализа могут быть гистограммы выходных параметров, оценки математических ожиданий M_j и среднеквадратичных отклонений Бj каждого из y_j, максимально возможные отклонения y_j каждого отклонения y_j от $y_{jном.}$, оценки коэффициентов корреляции r_{ji} между параметрами y_i и x_i, а также выходных параметров СМО.
В качестве исходных данных фигурируют статистические сведения о рассеянии внутренних параметров и данные ТЗ о допустимых диапазонах изменения или законах распределения Q.
Наибольшее распространение при статистическом анализе получили методы наихудшего случая и статистических испытаний (метод Монте-Карло).
Метод наихудшего случая. Служит для определения диапазонов возможного рассеяния выходных параметров без оценки плотности распределения этих параметров.
Пусть на некоторый выходной параметр задано условие работоспособности в виде y‹TT. Тогда интерес представляет верхняя граница диапазона рассеяния, т.к. большие значения у наиболее опасные с точки зрения невыполнения условия работоспособности. Верхняя граница диапазона рассеяния достигается в наихудшем случае, когда все аргументы y=f(x) принимают самые неблагоприятные значения. Самым неблагоприятным значением X_i ,будет $X_{imax}=X_{iном.}+\Delta X_i$ при выполнении условий

$$(y < TT \wedge \frac{dy}{dx_i} > 0) \vee (y > TT \wedge \frac{dy}{dx_i} < 0) \text{ или}$$

$$x_{i\min} = x_{iнно.} - \Delta x, если (y < TT \wedge dy \Big/ dx_i < 0) \vee (y > TT \wedge dy \Big/ dx_i > 0)$$, где ΔX_i – допуск на внутренний параметр X_i.
Алгоритм метода наихудшего случая состоит из операторов:
1. Анализ чувствительности, в результате которого определяются коэффициенты чувствительности dy/dx.
2. Задание, параметрам X_i самых неблагоприятных значений.
3. Выполнение анализа объекта в наихудшем случае.

Каждому выходному параметру $У_i$ соответствует свой наихудший случай. Если объект характеризуется m выходными параметрами и n внутренними параметрами, то всего требуется n+m+1 вариантов обращения к модели объекта.
Преимущество данного метода в том, что для его применения не требуется знание законов распределения значений внутренних параметров. Достаточно знать лишь допуске ΔX_i. Недостаток в том, что вероятность сочетания в объекте самых неблагоприятных или близких к ним значений параметров очень мала. Поэтому оценка рассеяния параметра поэтому методу оказывается значительность больше чем по более точному методу статистических испытаний.
Метод Монте-Карло (метод статистических испытаний). Этот метод позволяет получить более полные статистические сведения о выходных параметрах исследуемого объекта.
Алгоритм метода:
1. Задание значений внутренних параметров (аргументов зависимости У от Х и Q) в очередном статистическом испытании.
2. Расчет У.
3. Накопление статистических сумм.
4. Обработка накопленных сумм для получения результатов статистического анализа.

Операторы 1-3 выполняются в каждом испытании и могут быть распараллелены. Оператор 4 завершает статистический анализ.
Задание случайных значений параметров выполняется в соответствии с их законами распределения.
Накопление статистических сумм нужно для последующей оценки.

$$S_{j1} = \sum_{k=1}^{N} y_{jk};$$

$$S_{j2} = \sum_{k=1}^{N} y_{jk}^2;$$

$$S_{ji3} = \sum_{k=1}^{N} y_{jk} x_{ik};$$

$$S_4 = \sum_{k=1}^{N} q_k$$, где y_{jk} и x_{ik} – значения y_j и x_i в *к*-ом испытании;
q_k=1 если в *к*-ом испытании некоторые условия выполнено иначе q=0;
N – количество выполненных испытаний.
Обработка накопленных сумм производится после выполнения N испытаний.
При этом оценивается:
Математическое ожидание M_j и дисперсия σ_j^2 выходных параметров y_j:

$$M_j = \frac{S_{j1}}{N}; \quad \sigma_j^2 = \frac{(S_{j2} - NM_j^2)}{(N-1)}$$

Коэффициенты корреляции γ_{ij} между параметрами y_j и x_i :

$$r_{ij} = S_{ji3} - NM_j M_i \Big/ (N-1)\sigma_j\sigma_i$$

Вероятность выполнения условий работоспособности
$p = S_4 \Big/ N$, где M_i и σ_i - математические ожидания и среднеквадратичное отклонение параметра x_i.
Преимущество метода – в его универсальности.
Недостаток – большой объем требуемых вычислений.

Лекция№8
Логико-комбинаторные методы.
Относятся к задачам дискретного программирования. Обычно используются в процессе решения задачи структурного синтеза. Для их постановки требуется :
1) найти перечень некоторых параметров, объеденненных в вектор Х, значения которых характеризуют свойства вариантов и позволяют различить варианты между собой.
2) Сформулировать целевую функцию F(x) и ограничения в пространстве значений параметров, входящих в вектор Х.

Общая формулировка задачи дискретного программирования:
$extrF(x); XD = \{x | \varphi(x) > 0, \psi(x) = 0, x \in G\}$,
$X \in XD$
где G – счетное множество (в частном случае конечное).
Методы:
1. Отсечения - метод Гомори.
2. Комбинаторные – ветвей и границ.
3. Приближенные – локальной оптимизации.

Комбинаторные методы и **методы отсечения** являются итерационными. Их называемой точности, т.к. итерационный процесс сходится к экстремальной точке задачи.
В отличие от них применение приближенных методов не гарантирует получения экстремума, хотя в большинстве случаев найденное решение близко к оптимизации.
В методах отсечения каждая итерация представляет собой решение задачи непрерывного математического программирования. На 1-ой итерации решается исходная задача, по со снятым условиям дискретности $x \in \sigma$, т.е. допустимая область $XD = \{x | \varphi(x) \geq 0, \varphi(x) = 0\}$. По результатам решения задачи вводится некоторое ограничение $U_1(x)$, сужающее XD, но таким образом, чтобы экстремальная точка исходной задачи оставалась в этой области. Далее процедура повторяется, вводятся новые ограничения $U_k(x)$ (производятся отсечения области XD) до тех пор, пока результат очередной итерации не будет удовлетворять условию $x \in G$.
Узловой момент для реализации идей отсечения – наличие корректных правил для формулирования дополнительных ограничений $U_k(x)$. Такие правила удается найти в частных случаях.

Типичный пример методов отсечения – метод Гомори для решения задач целочисленного линейного программирования.

Метод Гомори

Пусть задано множество {A} – множество параметров объекта и {B} –множество вариантов, удовлетв. Некоторым данным условиям.

Строится матрица A*B

	A_1	A_2	…	A_n
	0.5	0.1	…	0.3
B_1				
B_2				
B_m				

Для каждого варианта подсчит. Коэф. Удовлетворения свойству A_1 , а потом суммируем коэф. Выбираем вариант, котор., имеет B_1 вводим в результирующий множитель. Затем эта строка вычеркивается и вычеркивается столбец, который отвечает за max A. Повтор до тех пор пока не покроет все свойства

Метод Гомори для булевой матрицы.

	A_1	A_2	A_3	A_4	A_5	A_6	A_7
$a_3\ a_5$	0	0	1	0	1	0	0
$a_1\ a_3\ a_5\ a_6$	0	1	1	0	1	1	0
$a_1\ a_4$	1	0	0	1	0	0	0
$a_4\ a_6$	1	0	0	0	0	1	0
$a_2\ a_3\ a_7$	0	1	1	0	0	0	1

Выбираем столбец с мин. Кол-вом «1» и строки, кот. Имеют 1 в данном столбце -> результат.. Потом вычеркиваем этот столбец, строки и столбцы, кот. Отвечают а в данной вычерк. Строке {($a_2\ a_3\ a_7$),($a_1\ a_4$),($a_4\ a_6$),($a_3\ a_5$),($a_1\ a_3\ a_5\ a_6$)}

Комбинаторные методы предназначаются для решения комбинаторных задач, т.е. для поиска наилучшего варианта в конечном множестве.

Могут быть решены методом полного перебора всех вариантов. Однако полный перебор практически неосуществим в большинстве случаев из-за примерно большого объема вычислений. Поэтому комбинаторные методы – это, как правило методы сокращенного перебора. Большую группу комбинаторных методов составляют методы сокращенного неявного перебора – методы ветвей и границ.

Так в задаче размещения р в микросхем на плате количество вариантов может доходить до р!, в задаче синтеза листов для проверки логических схем количество вариантов входных наборов при р входах – до 2^p.

Приближенные методы отличаются большим разнообразием, многие из них тесно связаны со спецификой конкретной задачи и относятся к специальному МО. При описании и алгоритмической реализации большой группы приближенных методов используются представления и терминология теории графов.

К приближенным методам, имеющим более общий характер, относятся методы **локальной оптимизации.**

Лекция 9

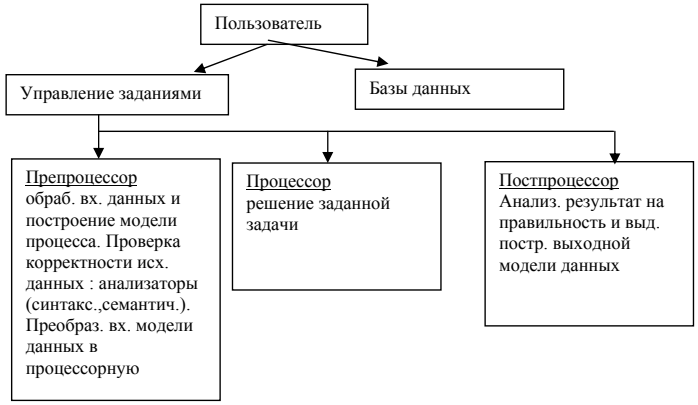
Лингвистическое обеспечение САПР (общая характеристика)

Лингвистическое обеспеч. – вкл. совокупность языковых средств, используемых в САПР(языки расписания моделей, алгоритмические языки, входные языки…).

Все языковые средства делят на 2 большие группы: 1) Языки программирования 2) Языки проектирования

Выбор языка прогр. осущ. из возможностей и необход. задачи.

Языке проектирования- структуры подсистемы САПР с точки зрения лингвистики.



Лингвист. обесп. включает группы языков

1) Инструментальные
2) управления заданиями – диалоговые языки, позволяющие определить последоват. выполнения задач в процессоре, препроцессоре и постпроцессоре(GUI).
3) Описания проектируемого объекта
   - функционального описания
   - структурного описания

Привязаны к уровню проектирования. Различают: табличные и текстовые (содержат целый ряд конструкция, позволяющих описывать последовательность операций проектируемого объекта, либо структуру этого объекта)

4) Документирования – это либо языки описания схемной документации, позволяющие идентифицировать условн. граф. обозначения и задавать таблицу связей. Либо языки аналогичные системе команд описывающие систему команд управл. ВУ

Лекция 10

Языки описания проэктных заданий.

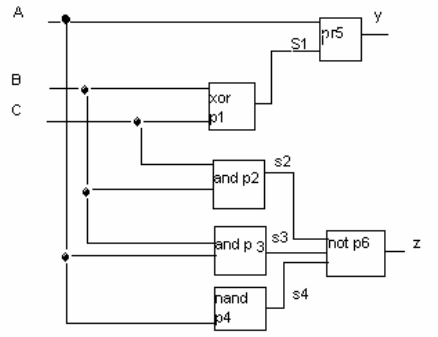
Языки позволяющие описывать проектируемые объекты и соотв. Вх. данные.
Языки делятся на функциональное, структ., параметр. Описание
O={F,S,P} P может быть в явном и неявном виде.

Языки задания F описания.
Функц. Описание. Опред. Динамич. Объекта может быть представлен в текстовом, граф. и табл. Виде.
Для текстового(аналит.) вида представления дотаточно предусмотреть средства аналогичные языкам программиров., кот. Должны быть дополнены средствами явного задания времени и возможности распараллеливания. Такие средства моут быть представлениы как:
Reserve(F) резервировать устройства Release(F) освободить Wait(t) ждать Hold(T) Halt(S)
Allocate(St) занять объем памяти, deallocate(st), initiate(pr) инициировать процесс.;

```
Sim proc;
Facility cpu(n) disk(m);
Storage (cm);
While(T>0) do
  Begin
  Initiate job;
   Hold(expute(2));
End sim;
```

```
Process job;
Allocate (cm) space
 While (Tcp>0) do
 Begin
   Reserve(cpu);
   Hold (dt);
   Initiate io;
   Release(cpu);
   Wait(io);
  End job;
```

Процесс sim выполн. Все время моделирования job не законч. До тех пор пока не законч. Io. Языки такого типа использ. На этапе сист. Проектирования. На более низких уровнях использ. Более близкие языки регистровой передачи HDL, SHDL, VHDL, SHL

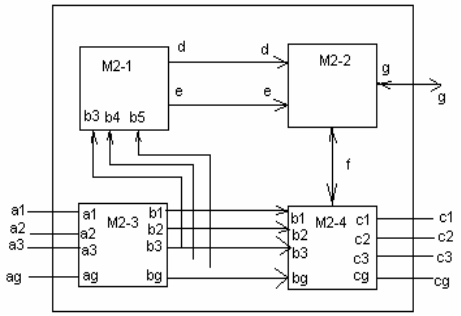


S1<= B xor C

В языке типа рег. Пер. использ. Описание последов. Во времени.

```
Entity fead is
Port ( A: in bit;
       B: in bit;
       C: in bit;
       Y : out bit;
       Z: out bit;  )
End entity fead;

Architecture fead arch if fead is;
Signal s1,s2,s3,s4:bit;
 Begin
 P1 process(B,C);
 Begin

 Wait  for 6 ns
End process p;
```

```
P2 process (B,C)
  Begin
  S2<=B and C;
 Wait for 4 ns;
End process p2;
P3   P4   P5  p6
End
Architecture fead. Arch;
```



Использ. Архитектура представл. В неявном виде. А в языках сист. Времени вообще не описываются действ., а только время выполнения. Функцион. Опис. Можно задать таблицей. Для этого языки функц. Описания представляют собой текст. Языки.

Языки структ. описания

Т.к. любая структура может быть представл. В виде графа, то используются графовые модели. S={E,ψ}E-элементы, ψ – связи между ними.

Графы задаются в идее таблиц или в виде списков связности. Так же с помощью матриц связности и матриц инцидентности(связь вершин и ребер) Такое задание усложн. Устройство из-за большого кол-ва нулей. Поэтому от матриц уходят к табличному и списковому описанию.

В яз. Типа регистровых передач текстовые описания в заголовках внешних связей непосредственно описания контактов Эл-ов между собой.

```
Name M1-1
In out M1-1: A[1:8], C[1:8], h,g;
In A[1:8], h;
Out C[1:8];
In out g
Value H[1:8],:[1,8]
        1,   0
g    h
H-монтажная логика запрешена
:[1:8] монт. Или
1-выс. Уров., 0 –низ. Уров.
Type ПК1, ПК2,RG1 – типы

COMP
 ПК 1 – М 2-1
ПК2-М2-2
REG-M2-3—M2-4
connect
 A[1:8] ->M2-3.A[1:8];
 :M2-3.B[1:8] -> M2-4.B[1:8];
 :M2-3.B[3:5] -> M2-1.B[3:5];
 :M2-4.B[1:8] -> .C[1:8];
 .h->M2-3.h;
 M2-1.d->M2-2.d;
 M2-1.e->M2-2.e;
 M2-2.f->M2-4.f;
 M2-2.g<->g;
end M1-1;
```

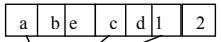
Списки задаются последовательностью соединений (в табл. так же)

Табличный язык

Представляет отдельно связи по входам выходам всех элементов.
тут достаточно таблиц чтоб описать все связи.

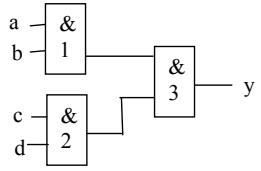
1) Таблица ф-ций TF

2) Табл. входов

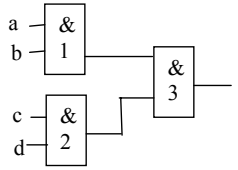


3) Табл. катологов

4) Табл. выходов
5) Каталог выходов



Лекция 11
Чтобы определить язык надо задать 4 множества {Vт, Vн,P,S}
Vт – терминальный словарь
Vн- нгетерминальный словарь
P- правила синтаксиса и семантики
S – начальные установки
Семантика- значение слов (правила построения конструкций).
Языки спискового типа
Список эл-тов с описанием всех связей этих эл-тов . Сдесь сокращена избыточная информация присутствующая в языках табличного типа. но все равно этот язык избыточен, так как это необходимо для контроля.



Сх 4(кол. входов), a,b,c,d(входы), 3(кол. выходов), y(вых)

(идентиф.),&(ф-ция),2(кол. вх),1(кол. связей по первому вх), a(связь), 1(кол. связ. по второму вх.), b, 1(кол. вых), 1(связь по вых), 3, 2, &, 2, 1, c, 1, d, 1, 1, 3

Требования к языкам : удобен, понятен, имеет возможность для расширения, **однозначен.**

Инф. обесп- совокупность сведений, необхожимых для выполн. проэктирования. Основную часть инф. обеспеч. составляют БД и характеризуютсчя особенностями этих БД
вся инф., которая используется должна быть структурирована, поэтому исп. БД.

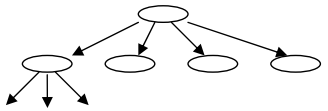
39. Базы данных – структурированное хранилище информации с СУБД(система управления БД)
Функции СУБД: 1)поиск
2)изменение
3)удаление

База данных – совокупность следующих компонент
1) Концептуальные модели данных(как формируется информация в БД)
2) Инструментальные средства, обеспечивающие погружение данных в некоторое хранилище построенное на основе концептлитическихуальной модели данных
3) Совокупность инструментальных средств и методов, обеспечивающих физическое представление данных на определенных носителях.

Требования к БД
1) Инф-ия должна быть центральным понятием
2) Приложения должны строится вокруг базы данных
3) По возможности необходимо исключить избыточность
4) Инф-ия должна хранится в одном месте(диске). Таким образом изменения в информацию вносятся лишь один раз.
5) Противоречивые копии исключаются, что обеспечивает целостность данных
6) Необх. Обесп. Целостность данных
7) Способ внесения изменений и обработки данных должны учитывать относительность времени (Для концептуальных и ана БД это время различно)
Концептуальные(транзакционные) БД – базы, обеспечивающие процесс проэктирования, приминительно к данной области.
Аналитические – базы на основании которых формируются отчеты или составляется окончательный вариант проектного решения (это совокупность уже полученных решений, к ней доступ нужен реже)
Концептуальная БД должна быть близко, доступ к ней должен быть простым

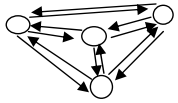
Иерархические БД(древовидные)

Определяются связями “один ко многим”



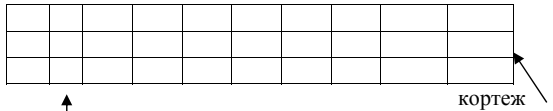
Существует корень
Описание связи на физическом уровне требует описание связей эл-та со след уровнем эл-тов
Недостаток: 1) Необходимость прохода всей цепочки при поиске инф-ии или внесения изменений
2) Чтобы добавить новые связи между эл-тами данной структуры необходимо проделать очень большую работу

Семантическая(Сетевая) модель
Строится на связях «многие ко многим»
Не свойственно хранение данных на одном месте
Связь легко восстановить
Проблема реплекации – если изменение внесено, то необходимо внести изменения во все связи этого



элемента

Реляционные БД
Нету явного описания связей. Основана на основе мат апарата- реляционная алгебра которая позволяет моделировать модели данных и на основе нее создавать запросы.
Понятие связи интерпретируется в запрос на спец языке SQL
Основой является запись (кортеж)



каждый столбец домен
каждый домен может быть ключем

Можно сформировать любую связь между различными элементами разных таблиц
В основе лежит электронная таблица

Обьектные модели
БД OLE
object linking and embedding — связывание и внедрение обьектов
com component
object model — -модель компонент обьект

OLE – общее предписание для перестройки самих БД во множество компонентов.
Это множество интерфейсов , которое описывают как могут взаимодействовать два любых компонента, каждый из которых работает с таблицами или наборами записей. Эти компоненты наз. поставщиками таблчных данных.
TDP(tabulate data provider)
Интерфейс играет ключевую роль

В основе процессор запросов
Основные функции:
-расшифровка запросов
-планирование действий по его выполнению
Вся информация в центральном сервере. На этом сервере лежат таблицы компонент всей общей БД
OLE(то есть какая информация и где содержится)
Пользователь обращается к процессору запросов на SQL и интерфейс находит необходимую инф-ию и соотв связи.

Интерфейс вып. Ф-ии:
1) Запрашивает табл., в которой содерж. хар-ка БД, входящих в общую БД
2) Для каждой таблицы запросить какие есть записи, форматы записей, типы значений, запросить необходимую инф-ию, выбрать ее, сформировать новые записи либо внести изм в старые записи.

Програмное обеспечение – множество приложений, которые используются для проэктирования
Различают ПО
1) общее – обеспечивает ОС инструментарием ЭВМ
2) Специальные – разраб для решения конкретных вопросов

Лекция 12
ПО САПР четко привязано к физической реализации. Ориентировано на двухуровневую систему:
1слой Рабочие приложения, основанные на GUI
2слой Некоторая БД, которая обеспечивает процесс проэктирования

В основе лежит Алгоритм+ БД=программа
Приложение является не гибким , плохо изменяемым и достаточно неудобным

Современные приложения расчитаны на 3х уровневую систему
1) Приложения рабочего стола (GUI)
2) Правила проэктирования (приложения обеспечивающие проэктирование)
3) БД(достаточно мощная)
Это уже логическая, а не физич структура приложений
При условии стандартизации лог алгоритм можна рассматривать независимо и формировать отдельно.
Каждый из этих логических слоев может быть реализован как некоторая компонента и распределен во времени и пространстве
Таким образом логическая 3х уровневая структура состоит из 3х серверов
- сервера данных
- сервера приложений
- сервера рабочего стола

Этапы системного проэктирования
На этом этапе выч система рассматривается с инф точки зрения. Исходными данными является тех задание на проэкт. Необходимо уточнить ТЗ, определить состав будущего устройства, определить что можно использовать уже готовое, а что необходимо изготовить
Результатом является структурная схема разрабатываемой системы
Задача синтеза не автоматизирована (нет формальных эл-тов). Основой является теория выч систем.
Синтез выполняется вручную с использованием ЭВМ, знаний, навыков, умений разработчика.

Основными задачами САПР являются задачи анализа спроэктированной системы.
Основным назначением модэлирования является:
1) Проверка логической правильности структурной организации проэктируемой системы
2) Оценка эксплуатационных качеств проэкта

Основнными модэлями используемыми на этом этапе являются СМО и СетиПетри

СМО(система массового обслуживания)
Состоят из 2х типов обьектов
1)стационарных (1)устр-во обслуживания 2)хранилище)-внутр.
2)временных –внешнее

1.Стационарные обьекты устройства обслуживания характеризуются занятостью(занято или свободно), возможностью организовать очереди
Хранилище(память)- основн характеристика. это занятый и свободный обьем.

У-во обслуживания в единицу времени может обслуживать только одну заявку
Заявка поступает на обслуживание только если есть долстаточный обьем памяти, если его нет, то заявка поступает в очередь
Очередь характ-ся:
дисциплиной обслуживания FIFO, LIFO
может иметь приоритет

2. Временные обьекты: заявки, транзакты
Их характеристики: время поступления, состояние(ждет или обслуживается), время пребывания в системе, приоритет

Для задания модели необходимо задать структуру устройства, параметры стационарных обьектов(время обслуживания, обьем памяти, хар-ки очередей), характеристики временных обьектов(интенсивность поступления, приоритеты)

Y=F(X,G)
X-внутр. параметры
G-внешн. параметры
Модель может быть аналитической и алгоритмической
Аналитические модели исп-ся для простейших СМО

Хар-ки простейших СМО:
1) входные потоки заявок обладают свойствами стационарности, ординарности и отсутствием последействия
2) стацион.-вероятность поступления определенного кол-ва заявок не зависит от времени (в фиксированный промежуток времени обрабатывается определенное количество заявок)
3) Ординарность-в любой момент времени обрабатывается только одна заявка
4) Отсутствие последствий – вероятность разных непересекающихся элементов времени не зависит от времени
5) Время обслуживания заявок должно быть распределено по экспоненциальному закону
6) Приоритеты не учитываются
7) Дисциплина обслуживания – FIFO
Аналит модели используются для ориентировочных моделей проэктируемой системы , либо в качестве макромоделей (при имитационном моделировании) отдельных элем-ов Вычисл системы
Результаты моделирования: производительность системы, вероятность обслуживания заявки, коэф загрузки оборудования, средняя, мин, макс длина очередей и т д.
Имитационная модель должна содержать в себе модели каждого из элементов СМО.(модель источника заявок, модели устройства обслуживания, модели памяти)
Модель источника заявок

Зависимая заявка – это заявка, которая генерируется другой заявкой.
Мод. источника заявок – алгоритм, по которому вычисляются моменты появления заявок. То есть модель чаще всего реализует алгоритм вычисления значения случайной величины, распределенной по заданному закону, эта величина характ-ет промежуток времени между двумя соседними заявками.
Может содержать алгоритм генерирования зависимых заявок.
Заявка начинает свое движение при поступлении на вход некоторой синхронизирующей заявки.
Каждый источник вырабатывает заявки одного типа с определенным приоритетом. Приоритет может задаватся, может вычислятся в модели самого источника. Приоритет может быть изменен при прохождении заявки.

Модель устройства обслуживания

Алгоритм выработки значений интервала обслуживания.
Для каждого типа заявки может устанавливатся свой закон распределения

Модель памяти

Алгоритм для определения обьема памяти требуемой заявки – случ величина. Параметрами является общая память и кол-во своб-ой и занятой.
Также содержит алгоритм управления очередями.
Связи определяются програмными средствами, которые называют узлами. Узлы определяют маршруты движения заявок между устройств обслуживания и памятью.
Типы узлов:

1) Определяют маршрут заявок в зависимости от их типа, либо по выполнению некоторого условия.
2) Разделение заявок на части или их обьединение
3) Для изменения параметров заявок

Имитац. модель – алгоритм, состоящий из упорядоченных обращений к моделям элементов.
Последовательность обращений определяется свойствами анализируемой вычислительной системы.
В такой модели необходим механизм отслеживания времени.

Лекция 13

Для представл. объектов исп. динамическую структуру объекта.
Динам. структура объекта представлена понятиями: 1) события 2)активности 3) процессы
Здесь необходимо время Т:
1) Время реальное
2) Время машинное – реальное время вып. программ.
3) Системное или модельное время – модель системного времени в машине.

Активность – любое действ. в вашей системе. Активности представлены промежутком времени, необходимым для вып. данного действия. { t1 hold (t); wait(t)}. Активности обязательно заданы промежутком времени.
Всегда необходимо указывать время выполнения данной активности, вычисляем время завершения данной активности.

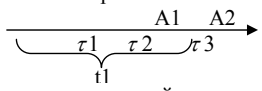
Событие- любое изм. состояния системы. События не требует времени, они происходят мгновенно.
События бывают:

1) Следования – предполагают последовательность выполнения некот. активностей.
2) Изменения состояния – предполагает изменения состояния системы.

Процесс – логически связанный набор активностей. Любой процесс может быть представлен активностью и наоборот. Процессы объединяют множество промежутков времени.
Процесс является динамическим объектом, представляющим единый акт выполнения совокупности логически связанных процессов. Выполнение процесса происходит во времени. В каждый момент времени в системе может выполнятся несколько процессов одного класса, находящихся в разных стадиях выполнения.
Любая система моделир.- последовательность событий. Между этими событиями происходят действия – активности. Эти активности тоже можно расписать как последовательность событий. Одни и те же действия можно представ. либо в виде активностей, либо в виде процессов.
Языки систем моделирования ориентированы на описания :

- событий
- процессов
- активностей

Основой любой системы моделир. является внутренняя система (планировщик), кот. организует последов. действий.
Ф-ции планировщ. : 1. Отслеживание времени работы
2. Отслеживание последовательности действий
В любой системе в любую единицу врем. может вып. одно или несколько действий. Необходимо обесп. моделир. паралел. во времени действий.
Во время T=t1 во времени у нас выполняется три активности. Мы будем в это время в реал. времени вып. эти активности.



Когда выполняются активности выбирается следующий момент времени T= t2 t2=min ($\tau 1$ $\tau 2$ $\tau 3$).
Должно выполнятся условие- свободно ли устройство, на котором вып. данная активность.

Языки ориентированные на описание активностей.

Планировщик, или сист. моделирования , содержит списки активностей и время (CSL).
В яз. ориент. на опис. событий. планировщик содержит переменную время и списки (очереди) событий. все события заданы явно в виде процедур.
Языки ориент. на опись процессов. События следования задаются неявно, а события изменения зад. явно. Здесь задаются активности. Процесс – совокупность активностей.
В зависимости от ориент. языка планировщ. содержат списки активностей, событий, процессов и еще очереди.

CSL

Моделирующ.программа состоит из :
1)заголовка (где опред. структура данных: устройства, очереди, задаётся нач. момент времени и планируется первая активность a0)
2)Описание активности a0 (условие начала выполнения, выполнение, действие по окончанию выполнения).
Программа состоит из множества описаний активностей. В реал. момент времени может вып. несколько активностей.
Планировщик содержит :
1) Текущее время
2)Список устройств(занято/свободно)
3) Список активностей:
- время начала активности
- идентификатор активности
- длительность выполнения
- точка реактивации (возврата)

4) Количество свободного и занятого пространства
Если ус-во занято, то активность становится в очередь.
Моделирующая программа аналогична предидущей, только вместо активности a0 событие L0: 1) Заголовок -//-
Описание события L0 :

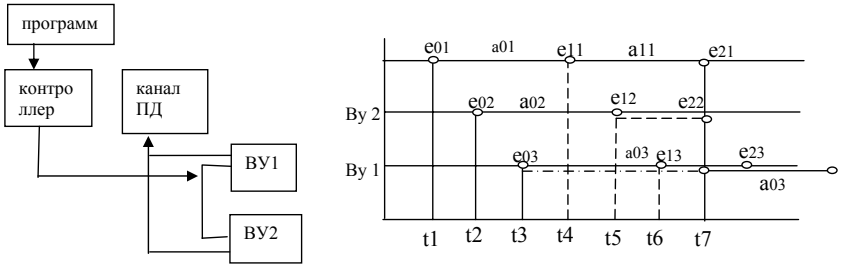
- условие вып. события L0 (сюда входит завершение предидущей активности)
- выполнение a0
- планирование L1

Планировщик содержит:
1) Текущее время
2)Список устройств (занято/свободно)
3) Список активностей
4) Список событий:
- время вып. событий (Т)
- идентификатор события
- точка реактивации (возврата) L

События объединяют в классы событий. Каждый класс описывается набором конструкций (опер.), характерных для вып. данного события.

Наличие такого опис. события , которое примен. в процессе моделирования определенное количество раз, говорит о последовательности событий одного класса. Каждое событие одного класса наз. динамическим экземпляром описания. События принадлежащие одному классу хар-ся одним и тем же набором операций, выполняемых над различными объектами одного класса, при чем принадлежность объекта данному классу каждый раз определяется (напр. значением ссылочных переменных либо динамических , по контексту). В этих языках все события задаются и описываются явно. Класс события задает набор действий, вызв. заявками.

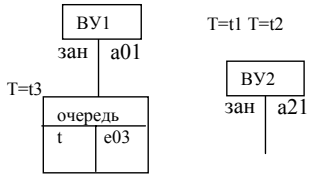
Пр.



e11,e12,e13- окончание установки
e01,e02,e03- начало установки
e21,e22,e23-окончание передачи данных

Списки событий

T	L
t1	L01
t2	L02
t3	L03
t4	L11
t5	L12
t7	L21
~~t7~~	~~L22~~



- нельзя потому что канал занят

Операторы описывающие процесс занятия ВУ1 одинаковые – классы событий. Для каждой заявки свой экземпляр.

Лекция 14

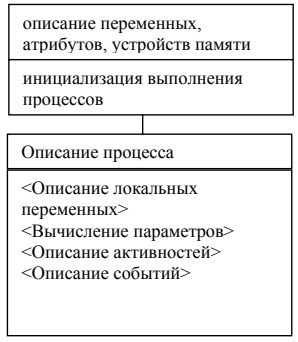
Процесс является динамическим объектом, представляющим единый акт выполнения совокупности логически связанных процессов. Выполнение процесса происходит во времени. В каждый момент времени в системе может выполнятся несколько процессов одного класса, находящихся в разных стадиях выполнения. Каждый класс процесса описывается совокупностью конструкций языка, представляющих выполнение данного процесса. Программа ориентир. на описание событий длинная, её можно сократить заменив на описание процессов (будут описываться только события следования). Каждый процесс инициируется другим процессом, который может находится как внутри системы, так и нет.

Конструкция описывающая поведение класса процессов, состоящая из указания активностей, входящих в процесс, в определенном соотношении следования условий управляющих их выполнение и воздействий, оказываемых процессом на атрибуты и состояния активных и пассивных объектов системы.

Существует система моделирования (ОС) и операционная программа, описывающая моделирование системы

Система моделирования состоит из :

Исходная (операционная программа) описывающая мод. системы состоит из:



(Программа моделирования выполняет имитацию работы модели.)

Посмотрим на **примере языка ASPOL**. (Symula, SOL, GPSS – все ориентированы на опис. процессов)

Программа в ASPOL состоит из основной программы, которая наз. SIM<имя>, это прогр. опис. глобальные переменные, устр-ва памяти и инициирует выполнение первых активностей и определяет время моделирования.

Для определения этого времени определяют кол-во заявок а заранее задается время моделирования.

Основной процесс выполняет управления всеми остальными процессами.

Любой процесс начинается: process< имя>
(лок. перменные)
заканчивается end process<имя>

Для инициирования выполнения процесса надо в другом процессе инициализ. этот процесс и фактические переменные: initiate <имя> (переменные)

Каждый процесс (и экземпляр класса) может иметь свой приоритет (неявно он равен приоритету процесса инициатора). Для изм. приоритета исп. конструкцию priority =<выражение>

Любой процесс может находится в одном из 4х состояний:

1)Готовности – процесс уже инициирован, но ещё не выполняется.
2)Выполнения
3)Задержки (hold (t)
4)Ожидания (ожидание освобождения устройства)

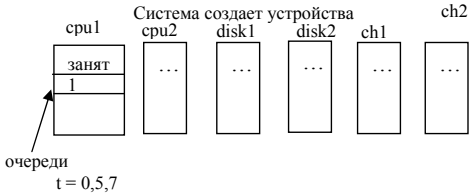
программа

```
sim mpos;
        def (n=2),(m=4),(k=2);
        facility cpu(n),disk(m),channel(k);
        storage cm; size (cm) 128
        while (time.le.20.*60.) do
        begin
                initiate job; hold(expnt1(12.));
        end
end sim;

process job;
        integer space, unit; real tcp,tio,records,dt;
        event ioend;
                tcp=erlang(10.,8.); records=tcp*random(2.,10.);
                tio=tcp/records; space=irandom(20,60);
                priority=60-space;
                allocate (cm) space;
```

```
        while (tcp.gt.0.) do
1 ->    begin
                reserve (cpu);
                        dt=epnt1(tio); tcp=tcp-dt;hold(dt);
2 ->                    unit=irandom(1,m); initiate io(unit,ioend);
                release (cpu); wait (ioend);
        end;
        deallocate (cm) space;
end process;

process io (unit, flag);  {процесс функционирования памяти}
        integer unit; real ts,tc; event flag;
                reserve (disk(unit));
                        ts=random(0.,75.); hold(ts);
                                reserve (channel);
                                        tc=.001*(2.5+random(0.,25.)); hold(tc);
                                release (channel);
                        release(disk(unit)); set(flag);
end process;
```

dt - время обработки одной записи
records - кол-во записей
События : event(<имена событий>). Событие принимает значение произошло, когда встречается оператор set (<имя события>).
Средства facility(<имя>), сдесь оно не занято, чтоб его занять исп. reserve(<имя>), освободить release(<имя>). Память описывается storeage(<имена памяти>), объём памяти size(<имя>)<выражение> Занять память allocate(имя)<выражение >, освободить память deallocate(имя)<выражение>.
переменные в программе: n-кол-во проц. m – кол-во внеш. памятей. k – кол-во каналов. disk – внешние устр-ва. cpu – процессор.

Система моделирования (по программе):



выполняемые пр-сы

t начала	имя пр-а	Имя транзакта	Врем. Заверш.	Тчк. возврата
0	Sim		1200	
0	Job	1		
5	sim	2		

t начала	имя пр-а	Имя транзакта	Врем. оконч задержки.	Тчк. возврата
0	Sim	2	5	*1
0	Job	1	7	*2
5	job	3	15	

про-сы в сост. ожидания

t начала	имя пр-а	Имя транзакта
0	Job	1
5	Job	2

Готовые процессы

Пр-сы в сост задержки.

t начала	имя пр-а	Имя транзакта	Событие
…	…	…	...