Курс «Основы кибернетики»

Основы проектирования цифровых интегральных схем

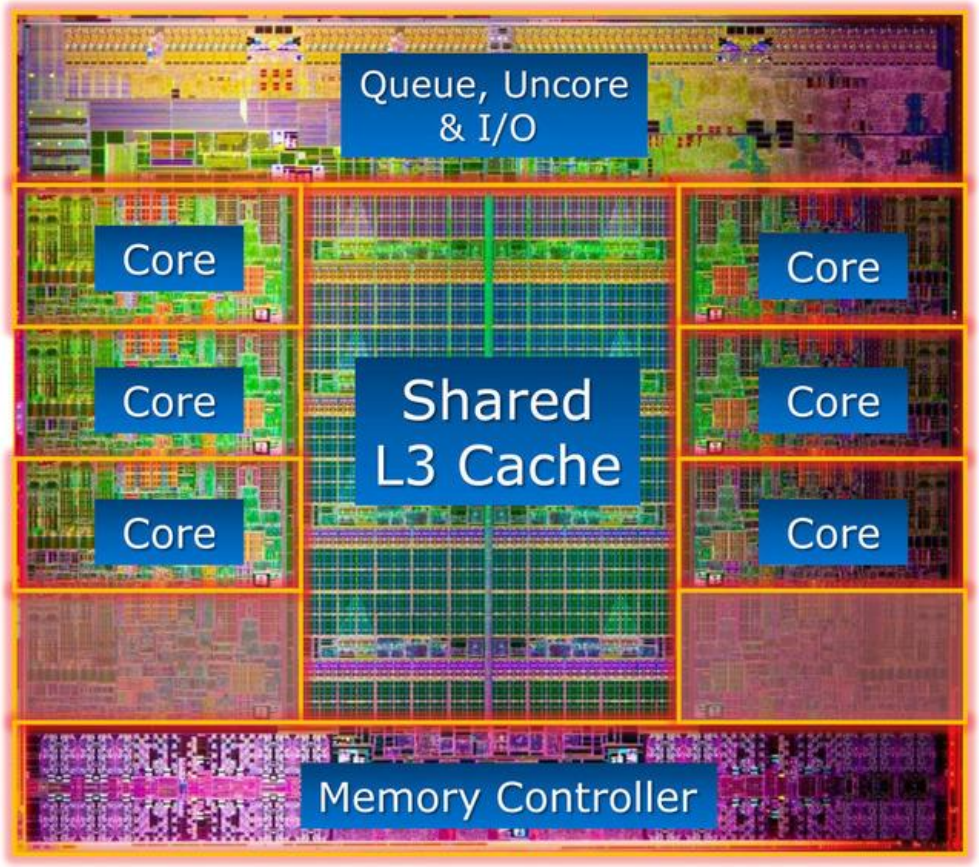
Весна 2017

План лекции

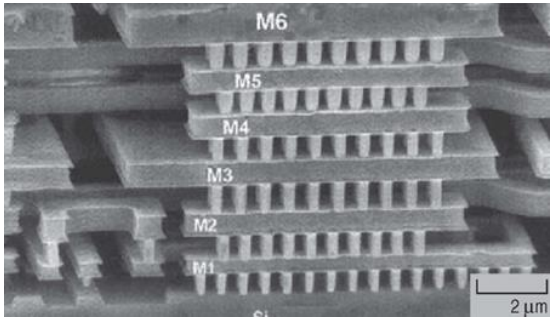
- Общие сведения о проектировании цифровых интегральных схем (ИС)
- Уровни абстракции при проектировании ИС
- Основные стратегии проектирования ИС
- Примеры:
  - Двухуровневый логический синтез
  - Каталоги оптимальных схем
  - Проверка эквивалентности и функциональная коррекция схем

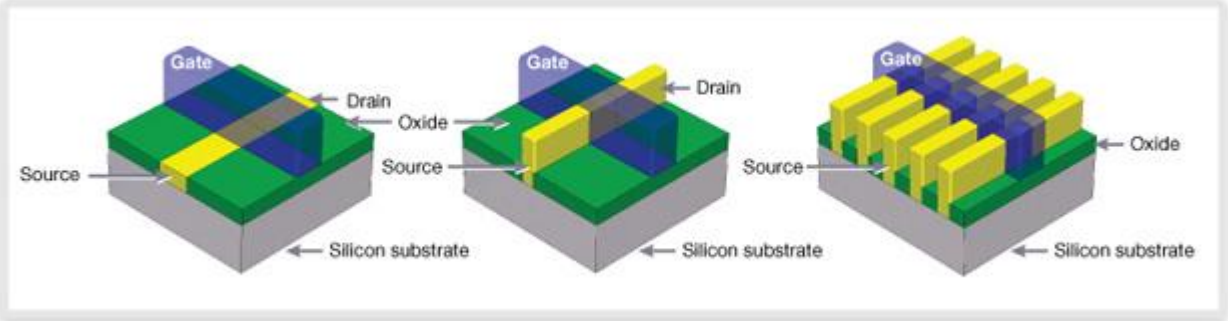
Общие сведения о проектировании цифровых ИС

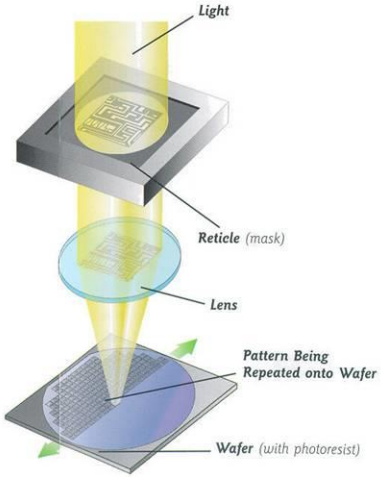
Сверхбольшая ИС

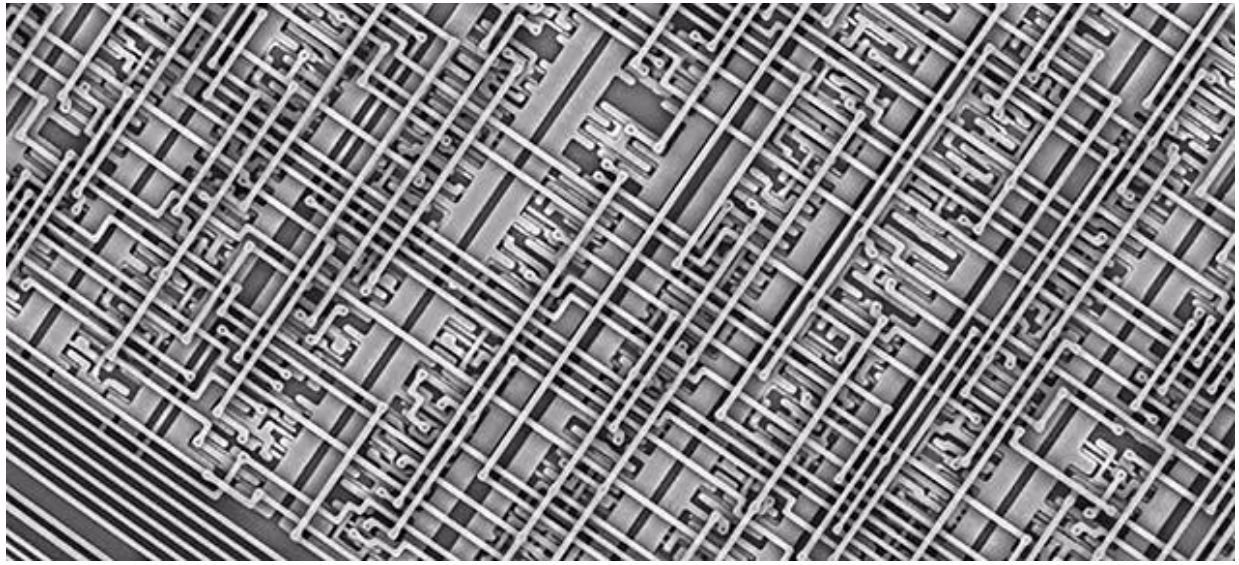


Элементная база цифровых ИС



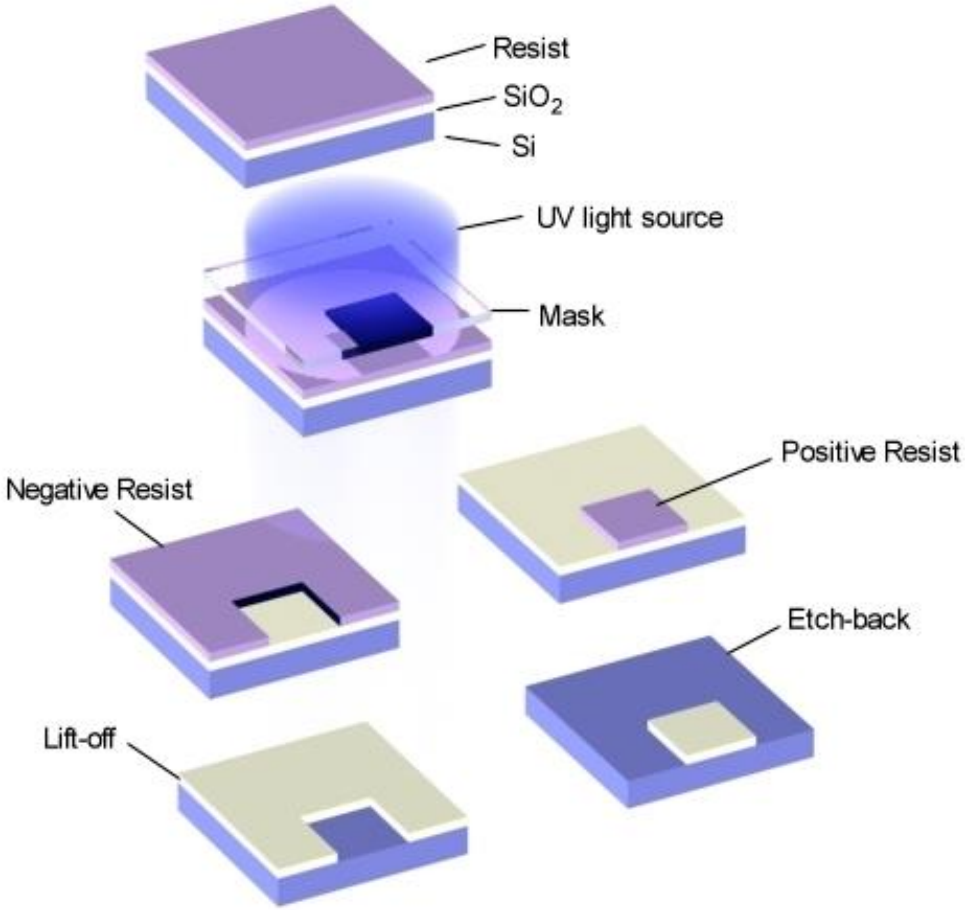




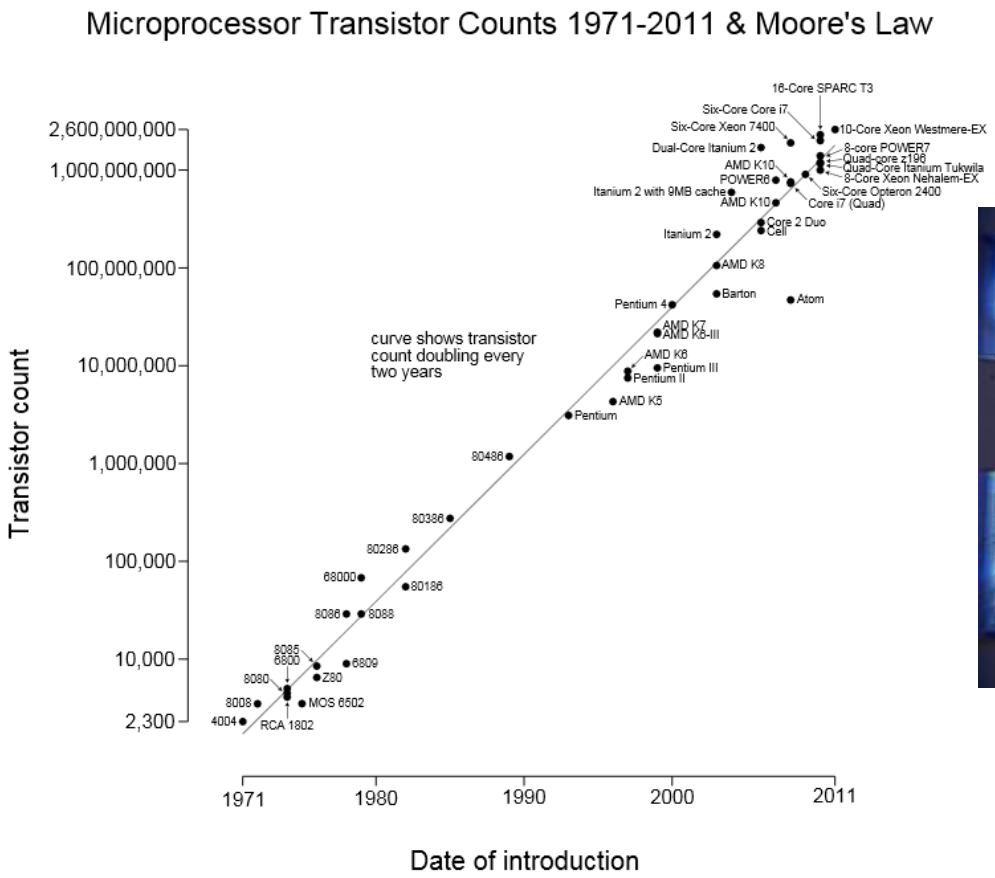


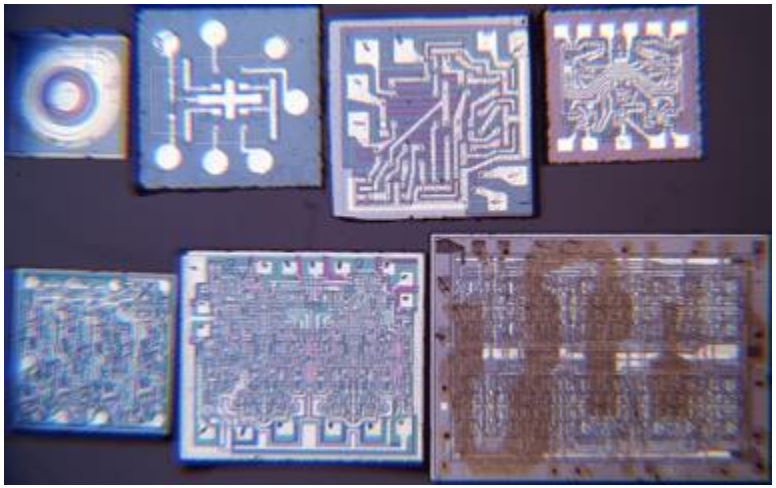
Производство ИС

Производство ИС



Закон Мура





Гордон Мур (1965 г.) – удвоение числа транзисторов на интегральной схеме будет происходить каждые два года.

Фундаментальные ограничения

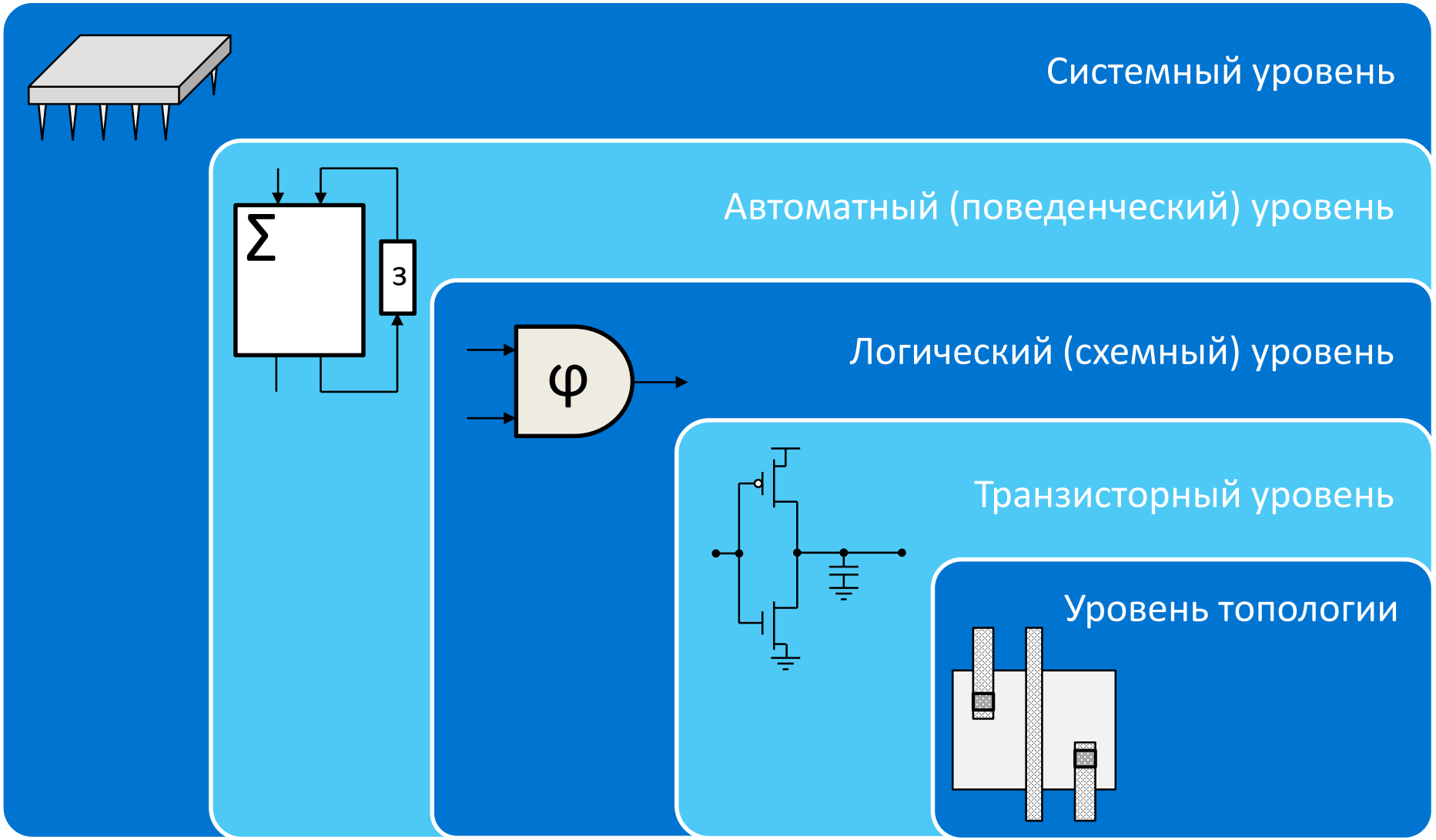
- Транзисторы и провода имеют конечные размеры
- Транзисторы можно расположить только в 2-х и 3-х мерном пространстве
- Скорость света ограничена

Средства автоматизации проектирования цифровых ИС

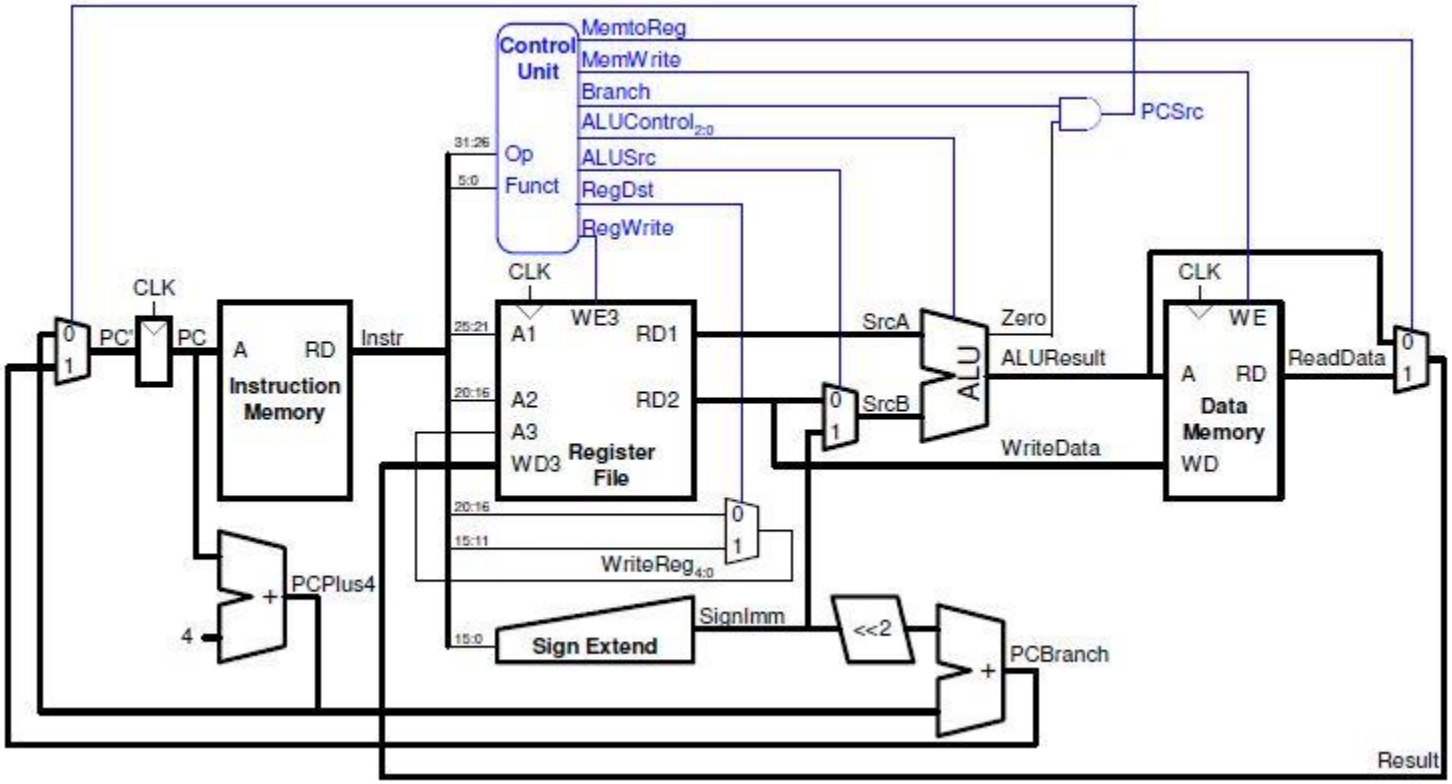
- Современные ИС невозможно спроектировать вручную
- Нужны специальные программы для автоматизации различных этапов проектирования ИС
- При этом требуется как понимание возникающих при этом математических задач, так и особенностей технологий производства ИС

Уровни абстракции при проектировании цифровых СБИС

Уровни абстракции при проектировании цифровых СБИС



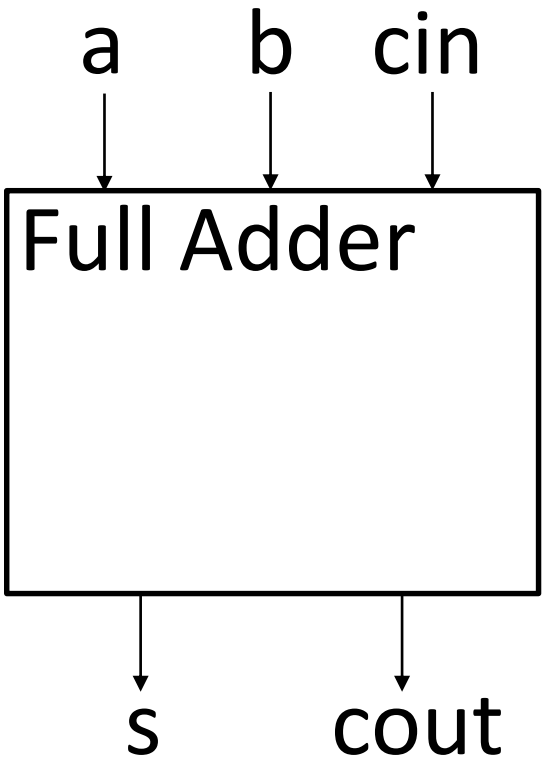
Системный уровень



Системный уровень

- Моделирование системы взаимодействующих процессов/сигналов
- Подходы к моделированию:
  - взаимодействие систем/компонент системы
  - система команд (instruction set simulation)
  - микроархитектура
  - использование языков описания аппаратуры (Verilog, SystemVerilog, SystemC)

Поведенческий уровень

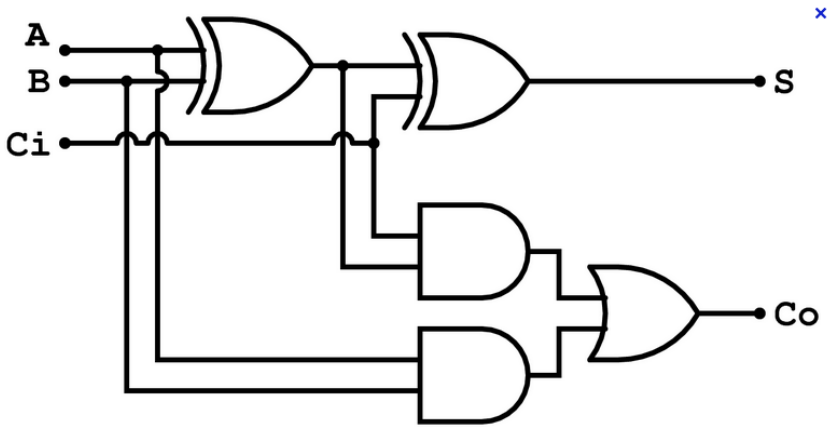



```
`timescale 1ns / 1ps
module FullAdder (
        input a,
        input b,
        input cin,
        output s,
        output cout );

       assign {cout,s} = a + b + cin;

endmodule
```

Поведенческий уровень

- Моделирование поведения/функционирования процесса/сигнала
- Подходы к моделированию:
  - использование языков описания аппаратуры (Verilog, VHDL)
  - register-transfer level (RTL)
  - использование автоматов и других математических моделей

Логический (схемный) уровень

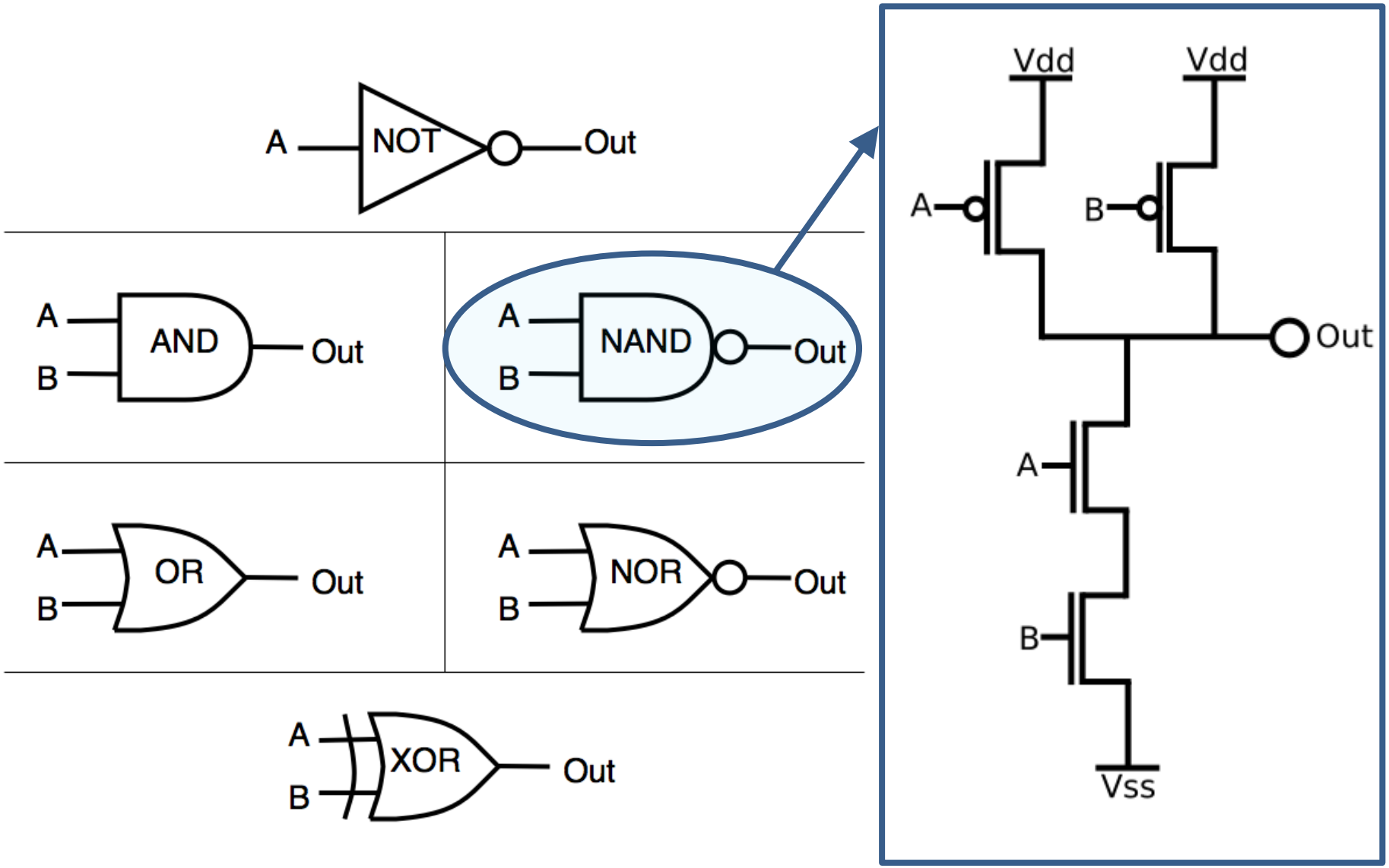



```
`timescale 1ns / 1ps
module FullAdder (
        input a,
        input b,
        input cin,
        output s,
        output cout );

        // wires (from ands to or)
        wire w1, w2, w3;

        // carry-out circuitry
        and( w1, a, b );
        and( w2, a, cin );
        and( w3, b, cin );
        or( cout, w1, w2, w3 );

        // sum
        xor( s, a, b, cin );

endmodule
```

Логический (схемный) уровень

- Моделирование структуры и основных элементов блока, реализующего заданный процесс/сигнал
- Подходы к моделированию:
  - использование языков описания аппаратуры (Verilog, VHDL)
  - netlist, gate-level design
  - математические модели схем
    - схемы из функциональных элементов (СФЭ) и их обобщения
    - And-Inverter Graphs (AIG)
    - Binary Decision Diagrams (BDD)
    - и др.

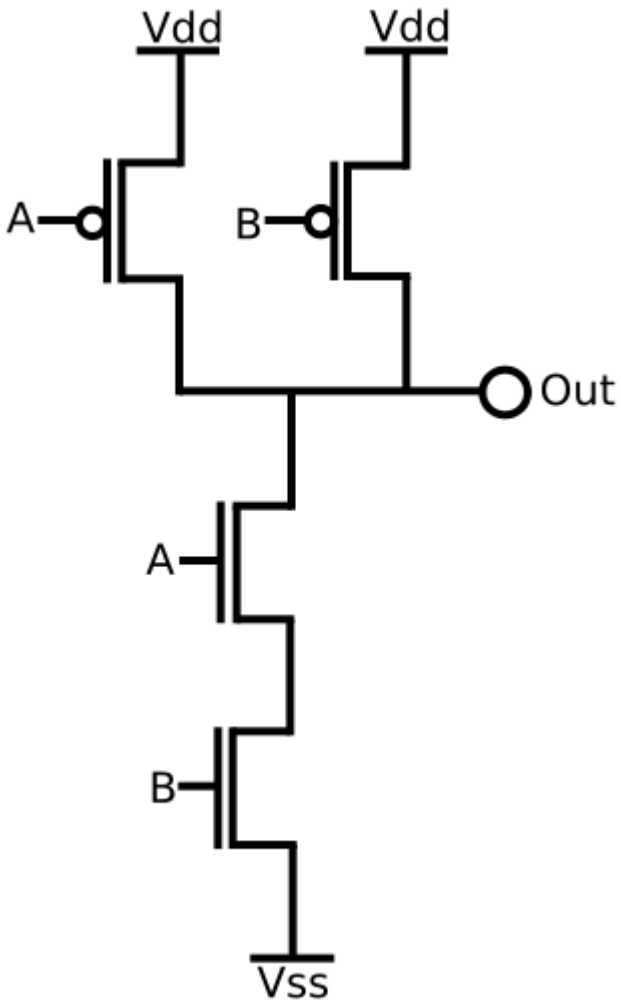
Транзисторный уровень

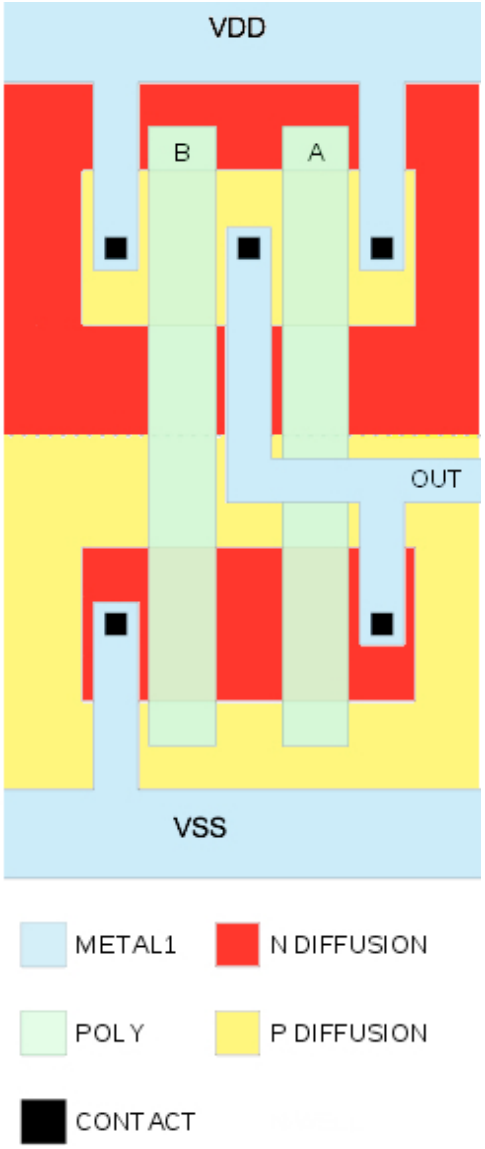


Транзисторный уровень

- Моделирование структуры основных логических элементов интегральной схемы
- Определение/оценка основных физических характеристик логических элементов (размер, задержка, энергопотребление и др.)
- Подходы к моделированию:
  - использование различных транзисторных моделей схем (контактные схемы и их различные модификации)
  - имитационной моделирование
  - SPICE моделирование

Уровень топологии





Уровень топологии

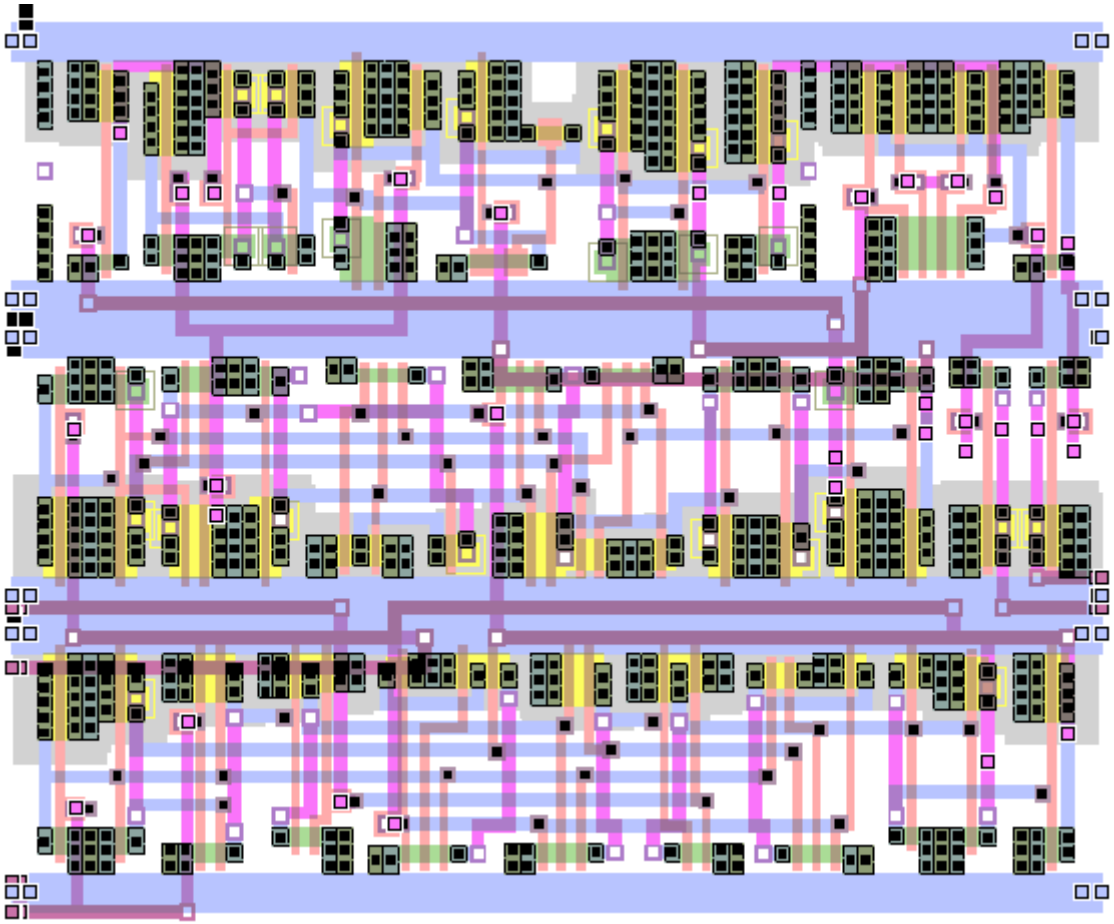
- Моделирование топологии (структуры и геометрии всех слоев) проектируемого устройства
- Основные задачи:
  - Design Rule Check (DRC)
  - Layout vs Schematics (LVS)
  - Оптимизация топологии и повышение выхода годных
    - Optical Proximity Correction(OPC)
    - Double/Triple patterning

Основные стратегии проектирования цифровых ИС

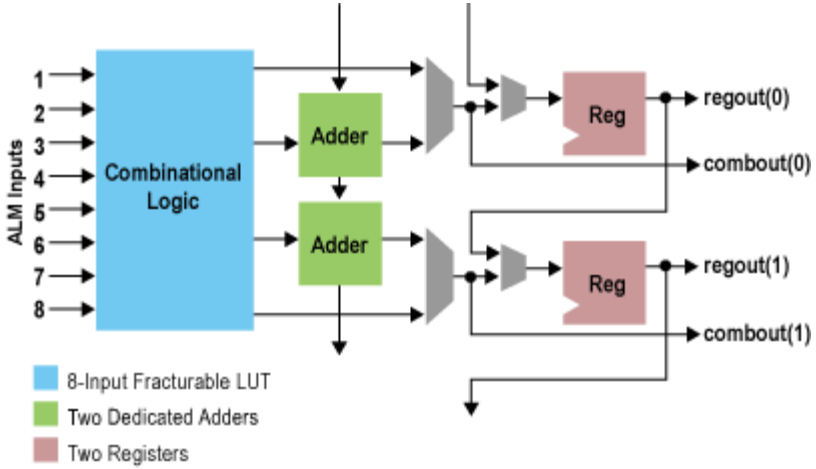
Основные стратегии проектирования цифровых СБИС

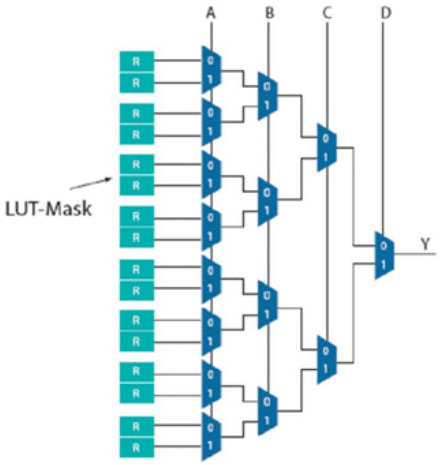
- Заказное проектирование
  - Все основные элементы ИС проектируются индивидуально и вручную
  - Высокая стоимость
- Полу-заказное проектирование
  - Использование заранее спроектированных элементов (IP-блоки, библиотечные элементы)
  - Использование средств автоматизации
  - Дополнительные ограничения на различных этапах проектирования
- Программируемые ИС
  - Все основные элементы ИС заранее спроектированы
  - Возможность настраивать устройство во время работы и/или на этапе проектирования

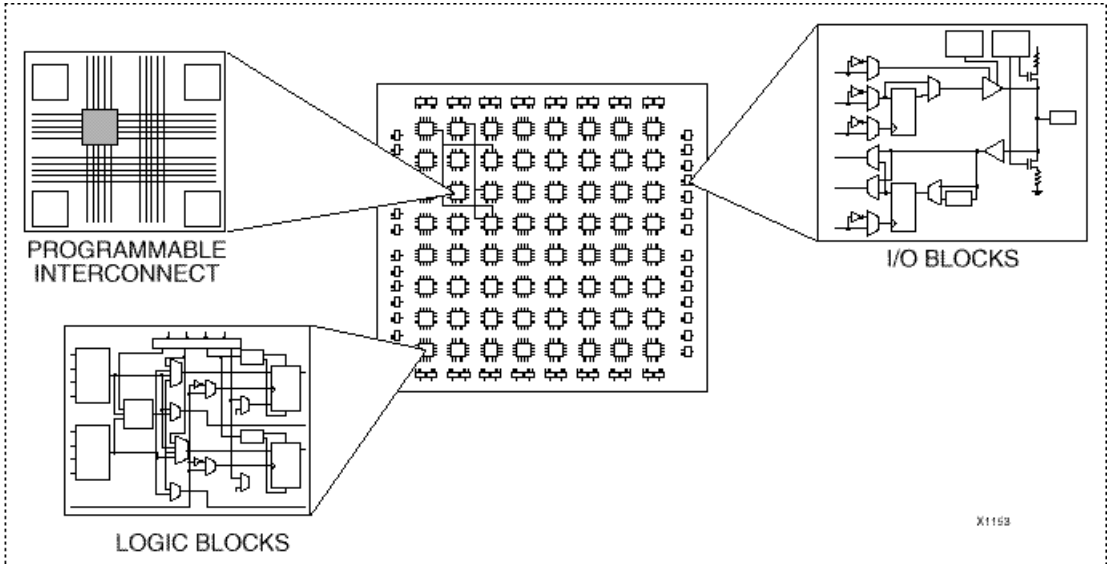
Методология проектирования на основе стандартных ячеек

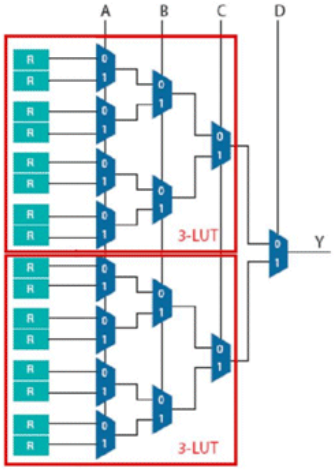


Программируемые логические интегральные схемы (ПЛИС)



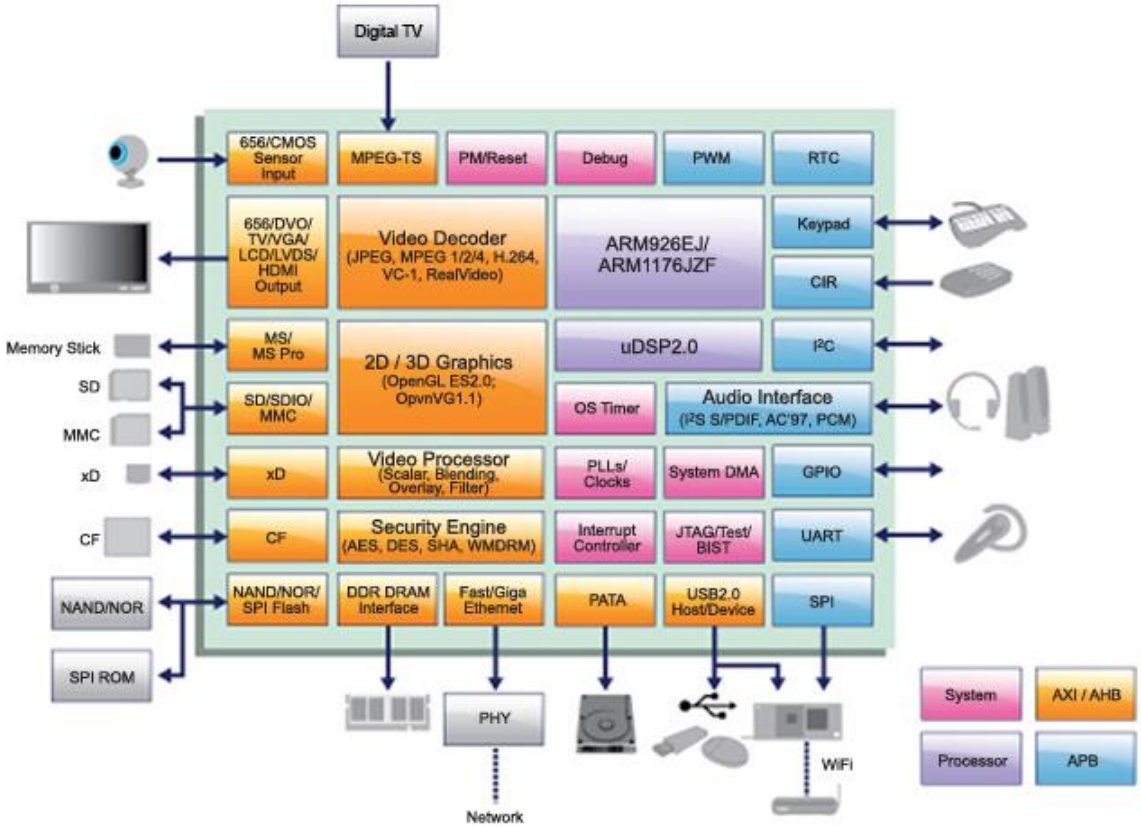


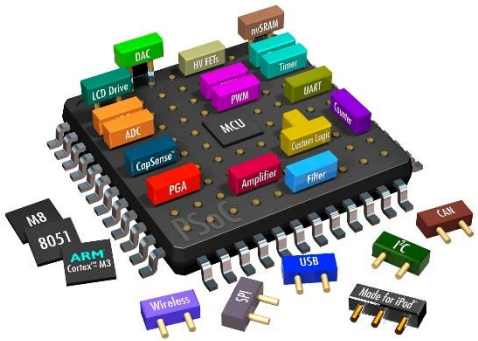




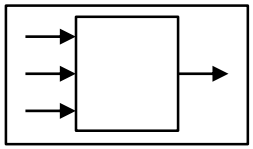
a'b'c'd' + abcd + abc'd' = 1000 0000 0000 1001 = 0x8009

Системы на кристалле

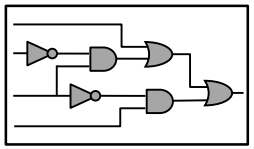


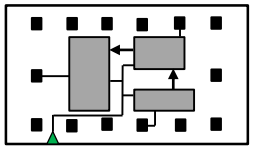


Упрощенный маршрут проектирования

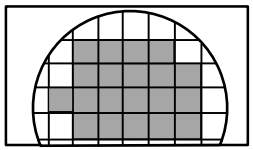


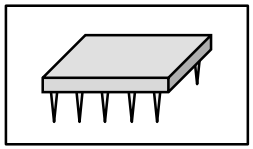
ENTITY test is
port a: in bit;
end ENTITY test;





DRC
LVS
ERC





Спецификация системы

↓

Проектирование архитектуры

↓

Функциональное проектирование

↓

Логическое проектирование

↓

Физическое проектирование

↓

Верификация топологии

↓

Изготовление

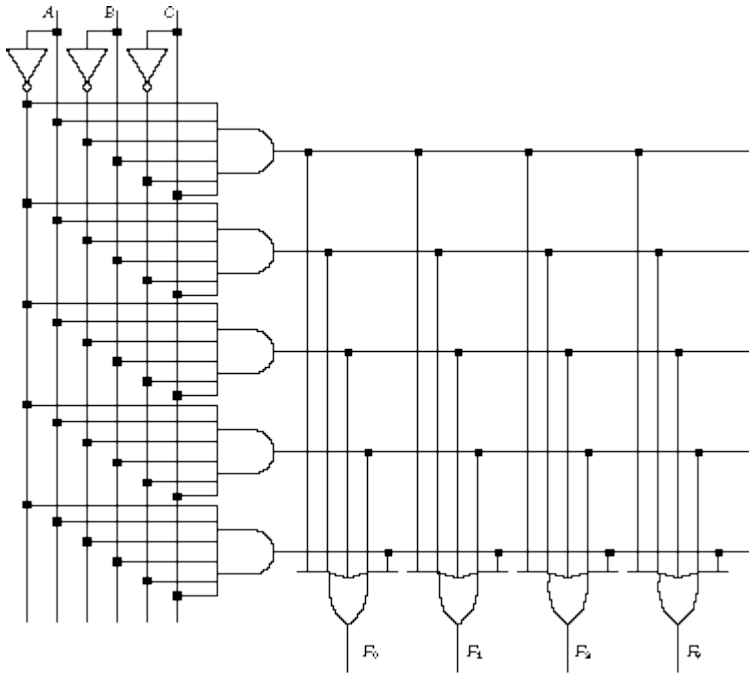
↓

Корпусирование и финальное тестирование

Пример: двухуровневый логический синтез

Программируемые логически матрицы (ПЛМ)

- Реализация системы булевых функций на основе ДНФ
- Первая интегральная схема TMS2000 (Texas Instruments, 1970)
- Оптимизация числа литералов приводит к уменьшению площади схемы.



PLA before programming

Минимизация ДНФ – основные подходы

- Известные методы минимизации ДНФ
  - Эквивалентные преобразования. Сложно применим при большом числе переменных и неясно как оценить качество полученного результата.
  - Метод карт Карно. Применим только при малом числе переменных.
  - Метод Квайна. Экспоненциальная сложность в худшем случае.
- Эвристические подходы - свойства.
  - Поиск ДНФ близких к минимальным.
  - Последовательное (итеративное) улучшение текущего решения.
- Эвристические подходы
  - Градиентный метод
  - Алгоритм ESPRESSO

Алгоритм ESPRESSO – общая идея

- Пример – функция 4-х переменных, заданная таблицей
- Указаны только единичные наборы
- Покрытие построено из некоторого набора импликант функции (не обязательно простых)

$x_3\ x_4$ \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	0*10
K_3	1001
K_4	1101
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Расширение» граней (EXPAND) – последовательное преобразование граней в максимальные грани
- Для каждой грани может быть несколько различных способов сделать её максимальной
- В нашем примере «расширяются» грани K_2, K_3, K_4
- Полученное покрытие состоит только из максимальных граней
- Полученное покрытие может не быть тупиковым или минимальным

$x_3\ x_4$ \ x_1	0	0	1	1
x_2	0	1	1	0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_3	1001
K_4	1101
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Расширение» граней (EXPAND) – последовательное преобразование граней в максимальные грани
- Для каждой грани может быть несколько различных способов сделать её максимальной
- В нашем примере «расширяются» грани K_2, K_3, K_4
- Полученное покрытие состоит только из максимальных граней
- Полученное покрытие может не быть тупиковым или минимальным

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_3	1**1
K_4	1101
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Расширение» граней (EXPAND) – последовательное преобразование граней в максимальные грани
- Для каждой грани может быть несколько различных способов сделать её максимальной
- В нашем примере «расширяются» грани K_2, K_3, K_4
- Полученное покрытие состоит только из максимальных граней
- Полученное покрытие может не быть тупиковым или минимальным

$x_3\ x_4$ \ x_1	0	0	1	1
x_2	0	1	1	0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_3	1**1
K_4	**01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Расширение» граней (EXPAND) – последовательное преобразование граней в максимальные грани
- Для каждой грани может быть несколько различных способов сделать её максимальной
- В нашем примере «расширяются» грани K_2, K_3, K_4
- Полученное покрытие состоит только из максимальных граней
- Полученное покрытие может не быть тупиковым или минимальным

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_3	1**1
K_4	**01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Тупиковое покрытие» – удаление граней, которые покрываются другими максимальными гранями (IRREDUNDANT)
- В нашем примере можно удалить K_3
- Полученное покрытие является тупиковым
- Полученное покрытие может не быть минимальным

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_3	1**1
K_4	**01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Тупиковое покрытие» – удаление граней, которые покрываются другими максимальными гранями (IRREDUNDANT)
- В нашем примере можно удалить K_3
- Полученное покрытие является тупиковым
- Полученное покрытие может не быть минимальным

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_4	**01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Сужение» граней (REDUCE) – последовательное уменьшение каждой из граней при сохранении покрытия
- Результирующее покрытие состоит не только из максимальных граней
- Позволяет изменить форму покрытия
- Порядок «сужения» граней имеет важное значение
- Является ключевым шагом, так как позволяет в дальнейшем произвести такое «расширение» граней, которое позволит покрыть другие грани
- В нашем примере можно «сузить», например, K_2 и K_4

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	0*10
K_4	**01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Сужение» граней (REDUCE) – последовательное уменьшение каждой из граней при сохранении покрытия
- Результирующее покрытие состоит не только из максимальных граней
- Позволяет изменить форму покрытия
- Порядок «сужения» граней имеет важное значение
- Является ключевым шагом, так как позволяет в дальнейшем произвести такое «расширение» граней, которое позволит покрыть другие грани
- В нашем примере можно «сузить», например, K_2 и K_4

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	0*10
K_4	1*01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

- «Сужение» граней (REDUCE) – последовательное уменьшение каждой из граней при сохранении покрытия
- Результирующее покрытие состоит не только из максимальных граней
- Позволяет изменить форму покрытия
- Порядок «сужения» граней имеет важное значение
- Является ключевым шагом, так как позволяет в дальнейшем произвести такое «расширение» граней, которое позволит покрыть другие грани
- В нашем примере можно «сузить», например, K_2 и K_4

x_3 x_4 \ x_1 x_2	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	0*10
K_4	1*01
K_5	1*1*

Алгоритм ESPRESSO – общая идея

$x_3\ x_4$ \ $x_1\ x_2$	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_4	1**1
K_5	1*1*

«Расширение» граней

$x_3\ x_4$ \ $x_1\ x_2$	0 0	0 1	1 1	1 0
0 0	1	1		
0 1	1	1	1	1
1 1			1	1
1 0	1	1	1	1

Грани

K_1	0*0*
K_2	**10
K_4	1**1
K_5	1*1*

«Тупиковое покрытие»

Алгоритм ESPRESSO

- Этапы алгоритма
  - «Сужение» граней (REDUCE)
  - «Расширение» граней (EXPAND)
  - «Тупиковое покрытие» (IRREDUNDANT)
- Разработка алгоритма началась в компании IBM и была завершена в университет Berkeley
- Литература
  - Brayton, Hachtel, McMullen, Sangiovanni-Vincentelli, Logic Minimization Algorithms for VLSI Synthesis, Kluwer Academic Press, 1984
  - Richard L. Rudell, (1986-06-05), “Multiple-Valued Logic Minimization for PLA Synthesis” Memo No. UCB/ERL M86-65 (U. California Berkeley M.S. Thesis)
  - Giovanni DeMicheli, Synthesis and Optimization of Digital Circuits, McGraw Hill, 1994

Пример: каталоги оптимальных схем

Задача синтеза контактных схем

- Пусть $X = \{x_1, \dots x_n, \dots\}$ - счетный алфавит входных переменных, а $P_2(n)$ - множество всех функций алгебры логики (ФАЛ) от переменных $x_1, \dots x_n$.
- Пусть $U^{\text{К}}$ - класс (1,1)-контактных схем(КС) от переменных из алфавита X.
- Сложностью $L(\Sigma)$ КС $\Sigma \in U^{\text{К}}$ называется общее число контактов в этой схеме.
- Сложностью $L(f)$ ФАЛ $f \in P_2(n)$ называется минимальная сложность КС $\Sigma \in U^{\text{К}}$, реализующей ФАЛ f.
- Функция Шеннона для сложности КС:

$$L^{\text{К}}(n) = \max_{f \in P_2(n)} L^{\text{К}}(f).$$

Известные результаты в области синтеза контактных схем от малого числа переменных

- 1954-1959гг. появились первые каталоги оптимальных и близких к ним КС для типовых функций от четырех переменных:
  - $L^K(4) \leq 14$ (Поварова Г.Н., 1954);
  - $L^K(4) \leq 13$ (Васильев Ю.Л., 1959).
- В 1981 г. Сусов В.Ю. при помощи алгоритмов переборного типа уточнил каталог Васильева Ю.Л.
- Для функции Шеннона для сложности КС от пяти переменных известны следующие оценки:
  - $L^K(5) \leq 30$ (Шеннон К. Э., 1949);
  - $L^K(5) \leq 28$ (Поваров Г.Н., 1959);
  - $L^K(5) \geq 19$ (Сусов В.Ю., 1981).

Инверсно-конгруэтные типы функций алгебры логики

- Инверсно-конгруэнтным типом ФАЛ будем называть множество ФАЛ, получаемых друг из друга при помощи применения операций перестановки и инвертирования переменных.
- Так как операции переименования и инвертирования переменных не меняют структуры КС, то можно считать, что КС Σ, реализующая ФАЛ f, соответствуют инверсно-конгруэнтному типу, которому принадлежит указанная ФАЛ.
- Множество $P_2^*(n)$ всех инверсно-конгруэнтных типов образует разбиение на классы эквивалентности всех ФАЛ из $P_2(n)$.
- $|P_2(5)| = 4\ 294\ 967\ 296.$
- $|P_2^*(5)| = 1\ 228\ 158.$

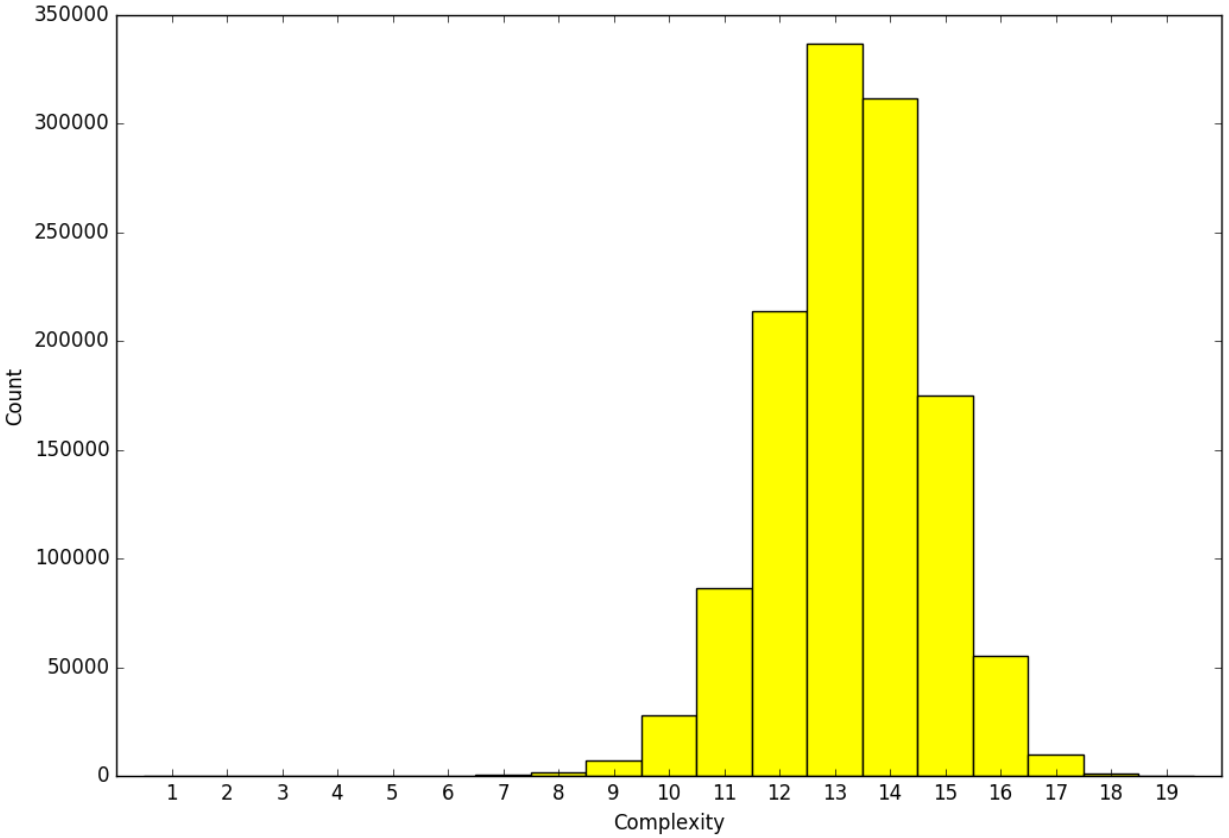
Каталог минимальных и близких к ним контактных схем

- Разработан ряд алгоритмов для оптимальных и близких к оптимальным КС из класса $U^{\text{К}}$.
- Используя разработанные алгоритмы, был составлен каталог минимальных и близких к ним КС для функций от не более, чем пяти переменных.
- На основе построенного каталога КС, была получена следующая оценка:

$$L^{\text{К}}(5) \leq 19.$$

Каталог минимальных и близких к ним контактных схем

http://mks2.cs.msu.ru



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	2	4	12	40	112	461	1797	7319	27677	86624	213992	336779	311798	175207	55363	10096	863	9

Области применения каталогов оптимальных схем

- Основа для построения библиотек логических элементов
- Системы логического синтеза на основе эквивалентных преобразований
- Исследование структуры и других свойств оптимальных схем

Пример: проверка эквивалентности и функциональная коррекция схем

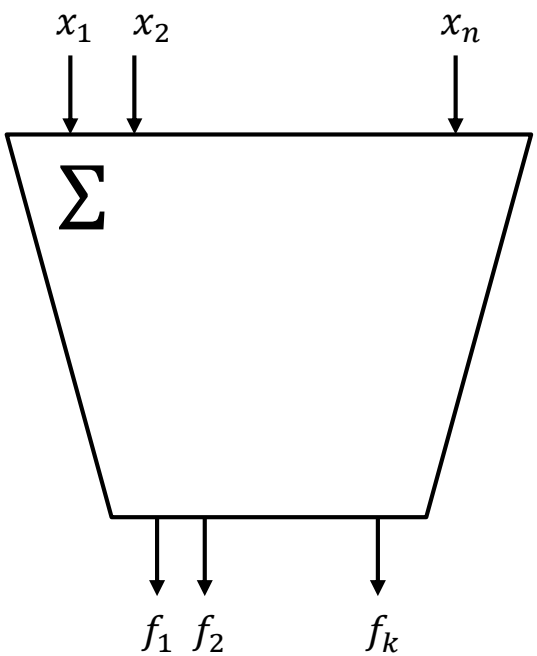
2015 CAD Contest at ICCAD

- Команда: 1 студент, 2 аспиранта, 1 научный руководитель
- Задача «Large-Scale Equivalence Checking and Function Correction»
- Команда заняла первое место
- Награждение: конференция ICCAD 2015 (Austin, Texas, USA)
- Две публикации по результатам исследования задачи

IC/CAD Contest 2015 CAD Contest

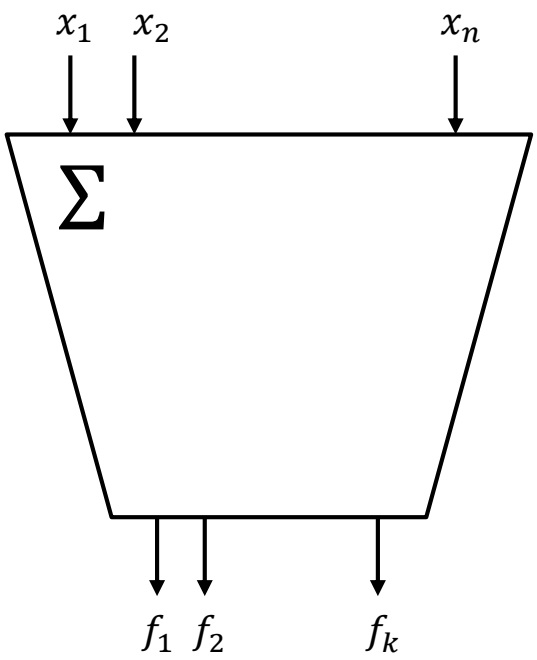
IEEE/ACM ICCAD 2015 INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN 34th Edition

Задача проверки эквивалентности СФЭ



Σ – СФЭ, реализующая систему функций алгебры логики $\{f_1, \dots, f_k\}$ от переменных $x_1, \dots, x_n$.

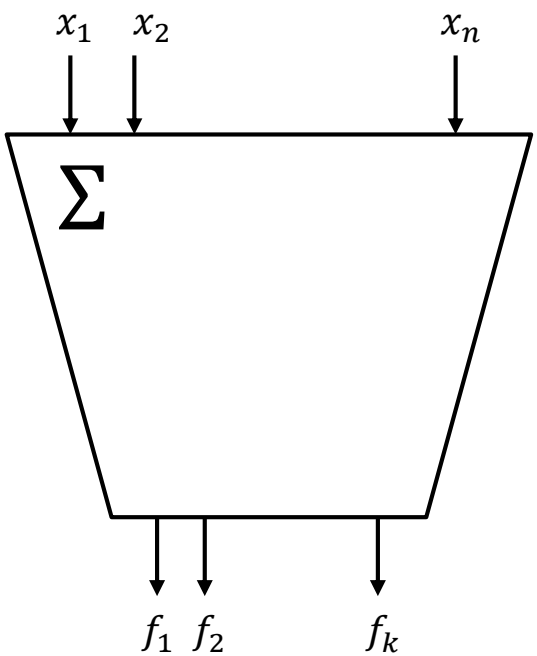
Задача проверки эквивалентности СФЭ



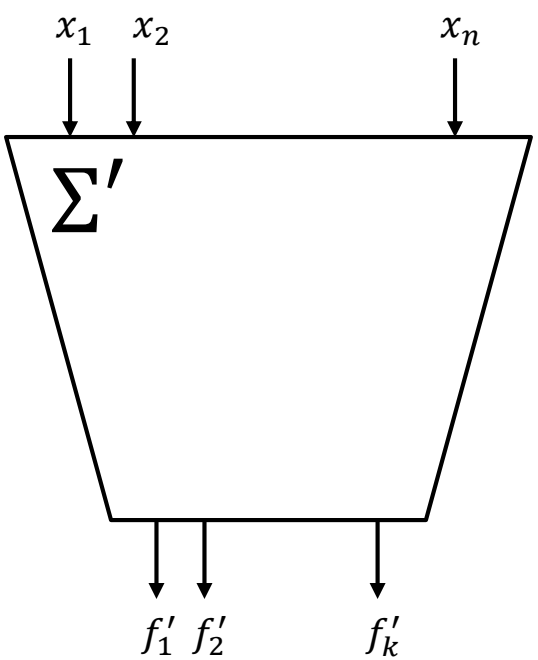
Логический синтез

Σ – СФЭ, реализующая систему функций алгебры логики $\{f_1, \dots, f_k\}$ от переменных $x_1, \dots, x_n$.

Задача проверки эквивалентности СФЭ



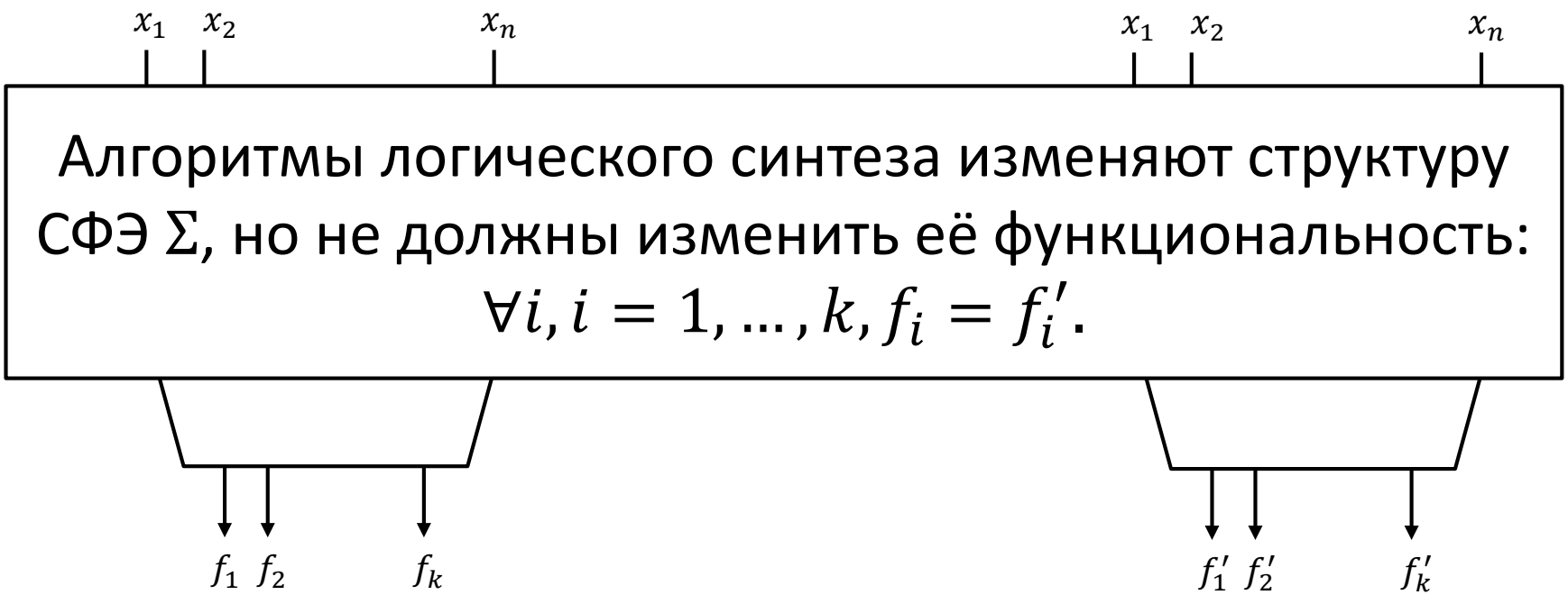
Логический синтез



Σ – СФЭ, реализующая систему функций алгебры логики $\{f_1, \dots, f_k\}$ от переменных $x_1, \dots, x_n$.

Σ' – СФЭ, получающаяся в результате применения алгоритмов логического синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

Задача проверки эквивалентности СФЭ



Σ – СФЭ, реализующая систему функций алгебры логики $\{f_1, \dots, f_k\}$ от переменных $x_1, \dots, x_n$.

Σ' – СФЭ, получающаяся в результате применения алгоритмов логического синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

Задача проверки эквивалентности СФЭ

x_1 x_2 x_n x_1 x_2 x_n

Алгоритмы логического синтеза изменяют структуру СФЭ Σ, но не должны изменить её функциональность:

$$\forall i, i = 1, \dots, k, f_i = f_i'.$$

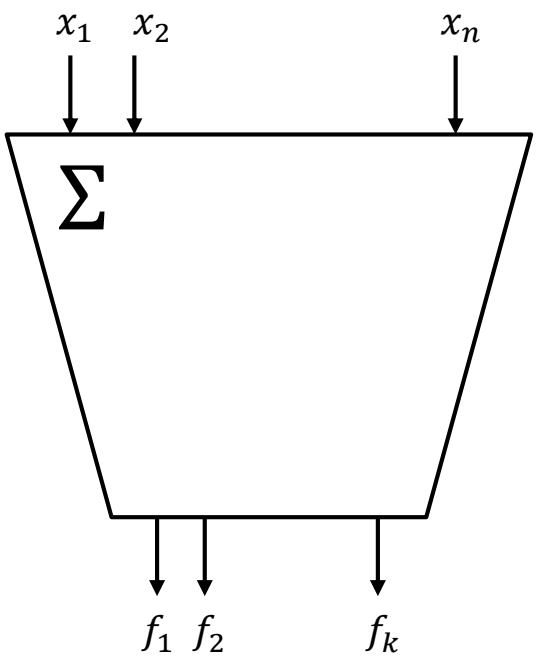
Задача проверки эквивалентности для СФЭ Σ и Σ' заключается в том, чтобы установить, что

$$\forall i, i = 1, \dots, k, f_i = f_i'.$$

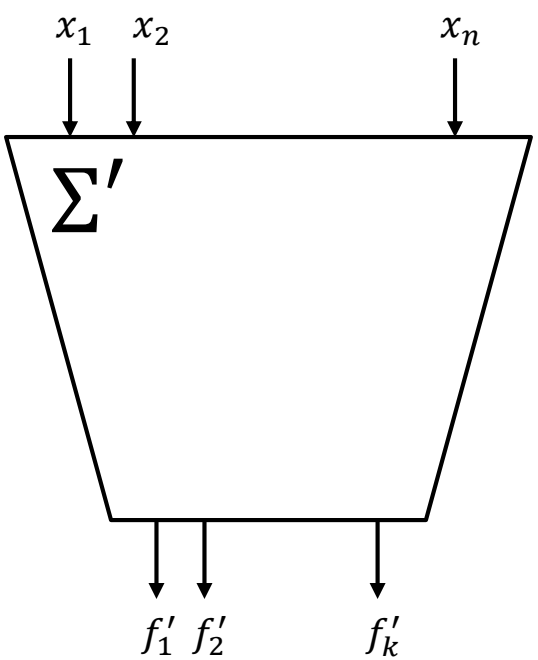
от переменных $x_1, \dots, x_n$.

синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

Задача функциональной коррекции СФЭ



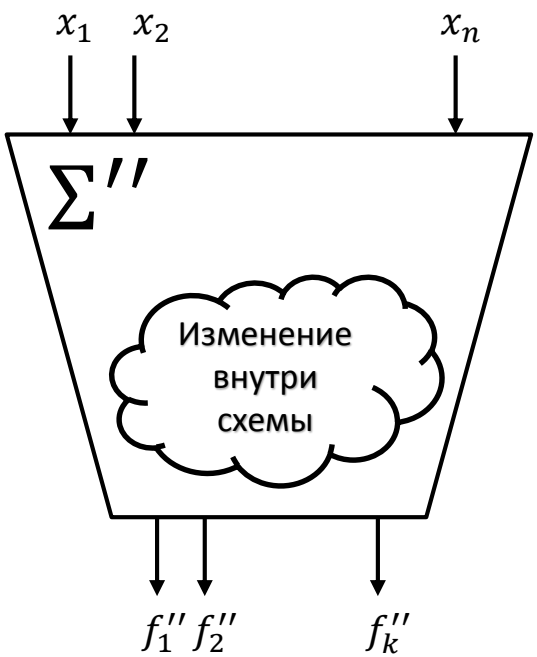
Логический синтез

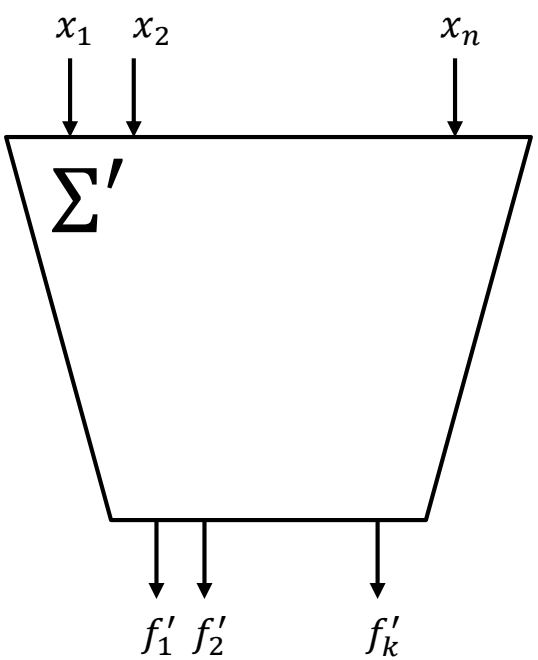


Σ – СФЭ, реализующая систему функций алгебры логики $\{f_1, \dots, f_k\}$ от переменных $x_1, \dots, x_n$.

Σ' – СФЭ, получающаяся в результате применения алгоритмов логического синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

Задача функциональной коррекции СФЭ

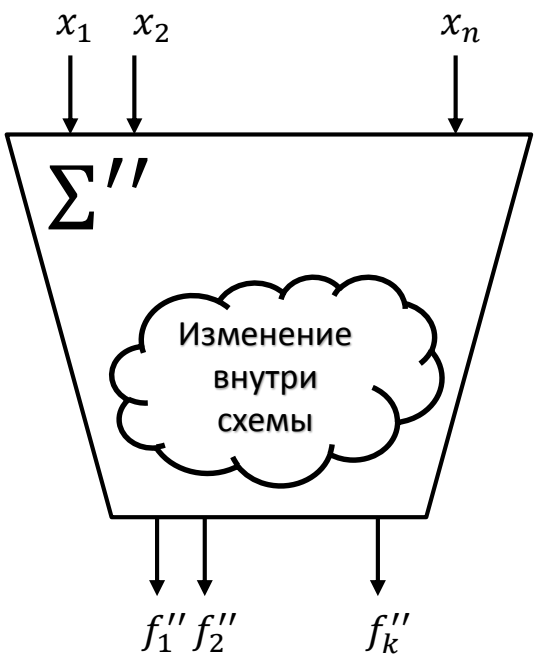




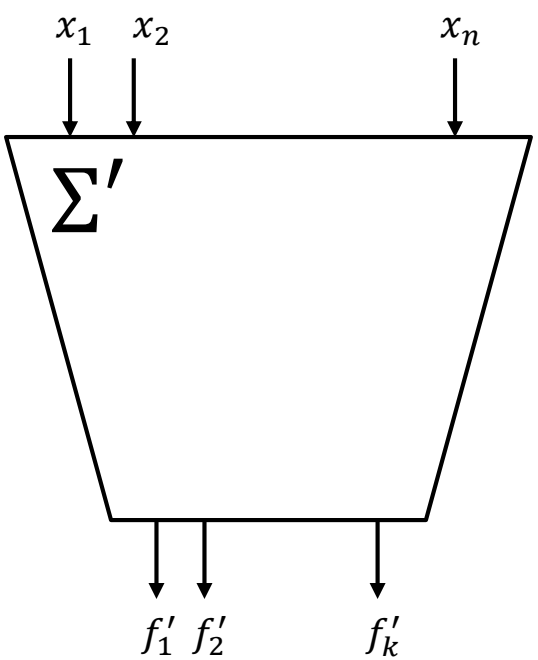
Σ'' – СФЭ, полученная в результате изменения спецификации в СФЭ Σ и реализующая систему $\{f_1'', \dots, f_k''\}$

Σ' – СФЭ, получающаяся в результате применения алгоритмов логического синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

Задача функциональной коррекции СФЭ



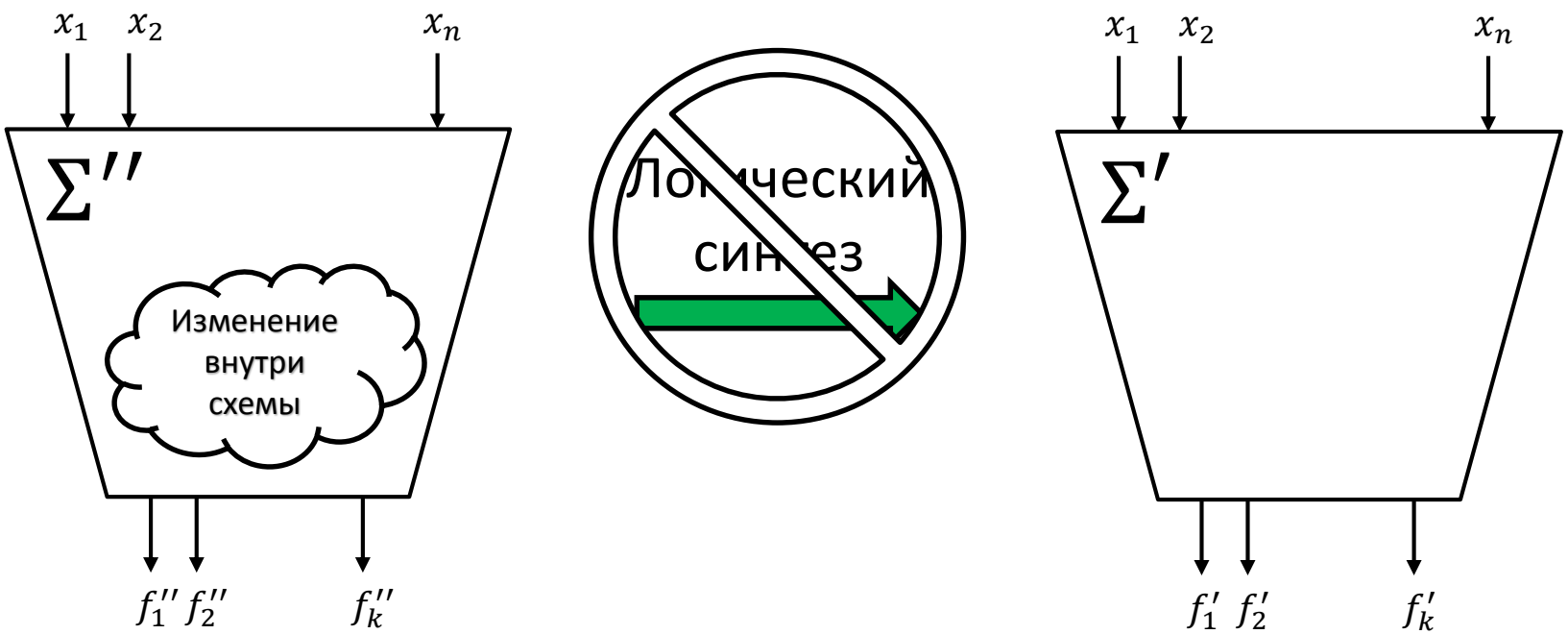
Логический синтез



Σ'' – СФЭ, полученная в результате изменения спецификации в СФЭ Σ и реализующая систему $\{f_1'', \dots, f_k''\}$

Σ' – СФЭ, получающаяся в результате применения алгоритмов логического синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

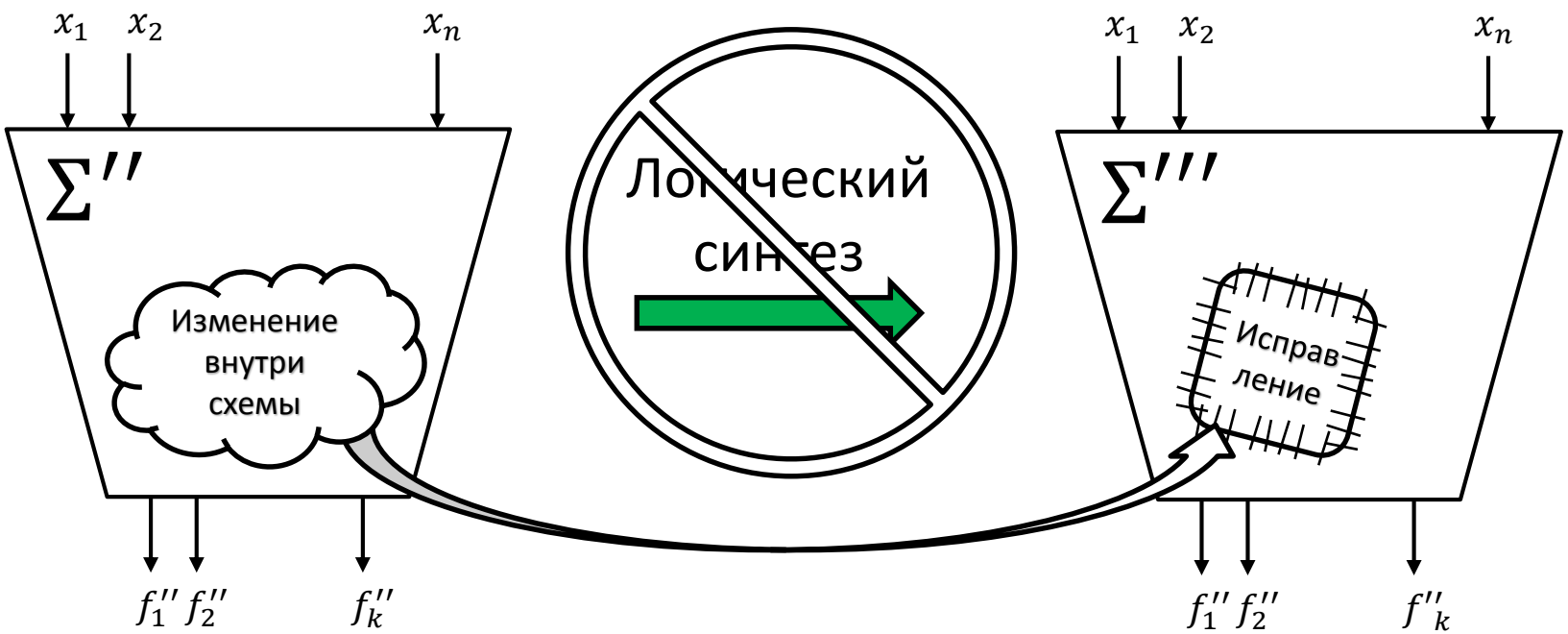
Задача функциональной коррекции СФЭ



Σ'' – СФЭ, полученная в результате изменения спецификации в СФЭ Σ и реализующая систему $\{f_1'', \dots, f_k''\}$

Σ' – СФЭ, получающаяся в результате применения алгоритмов логического синтеза и реализующая систему $\{f_1', \dots, f_k'\}$.

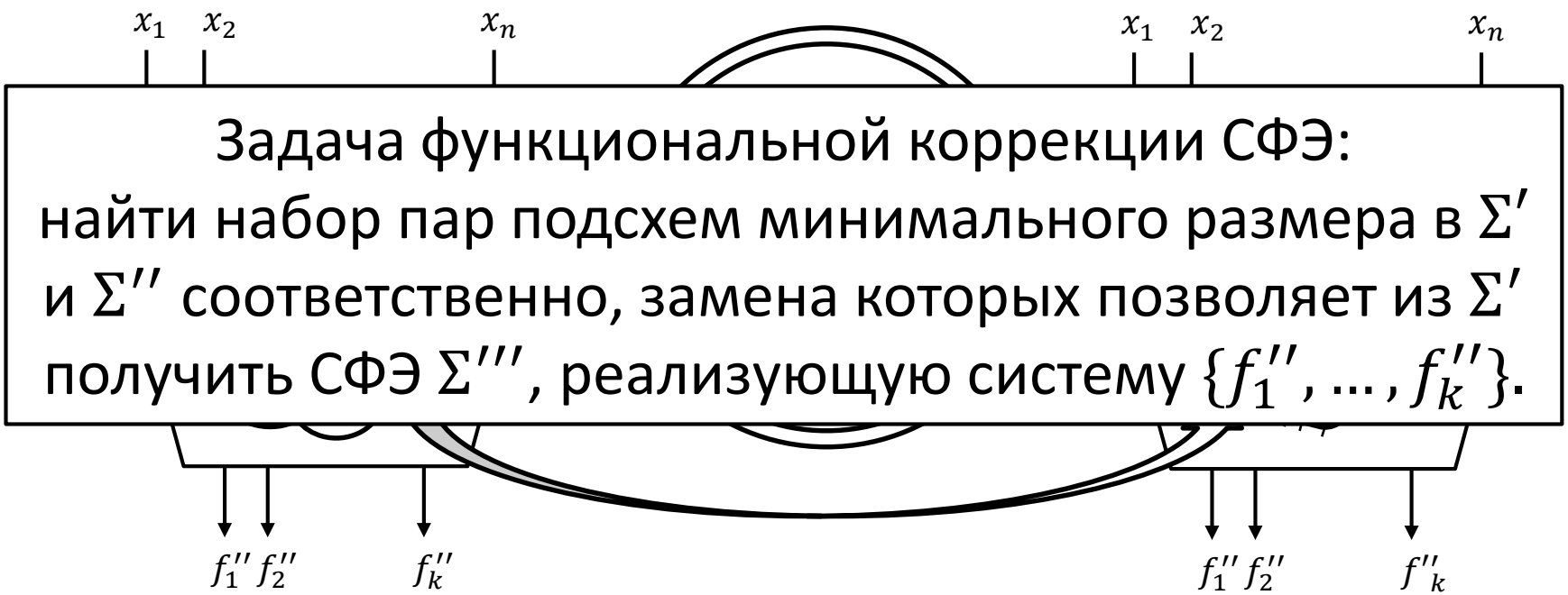
Задача функциональной коррекции СФЭ



Σ'' – СФЭ, полученная в результате изменения спецификации в СФЭ Σ и реализующая систему $\{f_1'', \dots, f_k''\}$

Σ''' – СФЭ, получающаяся из СФЭ Σ' в результате локальных замен подсхем и реализующая систему $\{f_1'', \dots, f_k'''\}$.

Задача функциональной коррекции СФЭ



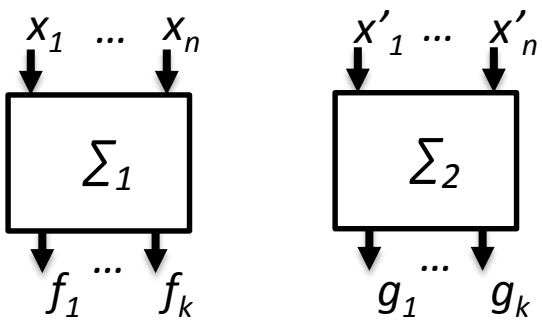
Задача функциональной коррекции СФЭ:
найти набор пар подсхем минимального размера в Σ' и Σ'' соответственно, замена которых позволяет из Σ' получить СФЭ Σ''', реализующую систему $\{f_1'', \dots, f_k''\}$.

Σ'' – СФЭ, полученная в результате изменения спецификации в СФЭ Σ и реализующая систему $\{f_1'', \dots, f_k''\}$

Σ''' – СФЭ, получающаяся из СФЭ Σ' в результате локальных замен подсхем и реализующая систему $\{f_1'', \dots, f_k''\}$.

Множества сравнения и отождествления

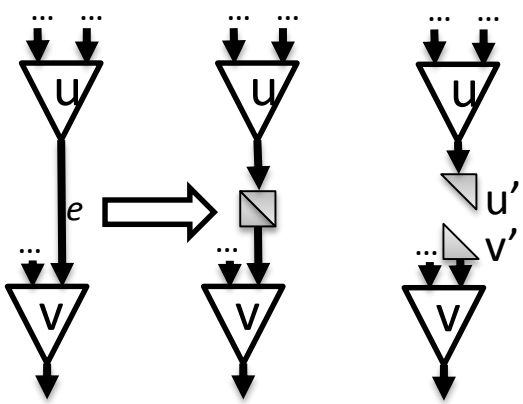
- ***Множество отождествления (МО)*** – произвольный набор входов СФЭ Σ_1 и Σ_2
- ***Множество сравнения (МС)*** – произвольное множество вершин СФЭ Σ_1 и Σ_2, отличных от входов
- МС считается ***эквивалентным***, если оно содержит две и более вершин и указанные вершины реализуют попарно равные булевы функции
- Иначе, МС считается ***неэквивалентным***
- Пусть сначала заданы k МС и МО на основе взаимно-однозначного соответствия входов и выходов СФЭ Σ_1 и Σ_2
- Множество всех МС СФЭ Σ_1 и Σ_2 назовем ***разрезающим множеством (РМ)***
- РМ, состоящее только из эквивалентных МС называется ***эквивалентным***



МО: $(\boldsymbol{x_1}, \boldsymbol{x'_1}) \dots (\boldsymbol{x_n}, \boldsymbol{x'_n})$
МС: $(\boldsymbol{f_1}, \boldsymbol{g_1}) \dots (\boldsymbol{f_k}, \boldsymbol{g_k})$

Формальное понятие разреза

- ***Разрезом*** ориентированного ребра $e = (u, v)$ будем называть пару вершин (u', v'), получающуюся в результате удаления ребра e и добавления ребер (u, u') и (v', v)
- Выходная вершина u' реализует ту же булеву функцию, что и вершина u, или ее отрицание (***инверсный*** разрез) и включается в существующее МС или формирует новое
- Входная вершина v' помечена символом новой входной переменной и включается в существующее или новое МО
- Новые ребра также могут быть разрезаны



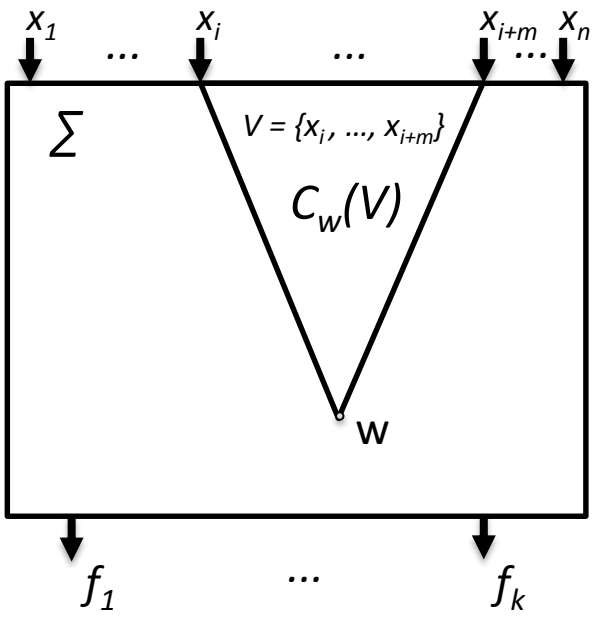
$$\mathrm{PM} = \mathrm{MC}_{\mathbf{1}} \cup \cdots \cup \mathrm{MC}_{\boldsymbol{k}}$$

$$\mathrm{MC}' = \mathrm{MC}_{\boldsymbol{i}} \cup \boldsymbol{u}'$$
$$\mathrm{PM}' = (\mathrm{PM} \backslash \mathrm{MC}_{\boldsymbol{i}}) \cup \mathrm{MC}'$$

$$\mathrm{PM}' = \mathrm{PM} \cup \{\boldsymbol{u}'\}$$

Конусы СФЭ

- ***Вершина w достижима из вершины u,*** если в графе СФЭ существует ориентированный (u, w)-путь.
- Для вершины w через $\boldsymbol{V}$ обозначим множество всех входов СФЭ, из которых w достижима.
- ***Максимальный конус $C_w(V)$ вершины w –*** множество всех вершин из которых вершина w достижима, отличных от элементов множества V
- $|C_w(V)|$ ***- вес*** конуса
- ***Максимальное дерево вершины w –*** максимальный конус этой вершины, состоящий только из вершин, полустепень исхода которых равна 1



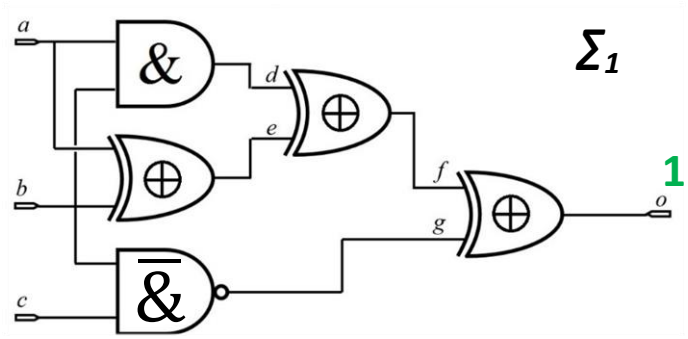
Задача разбиения СФЭ

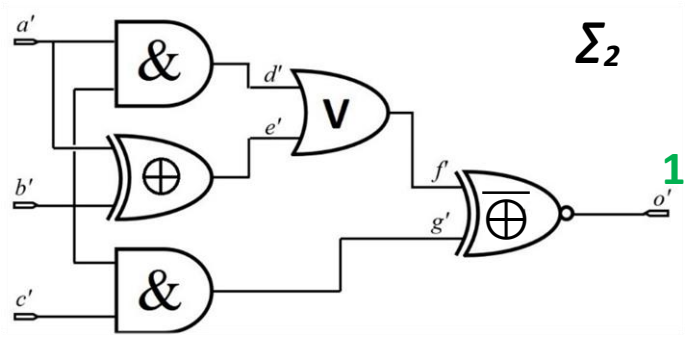
- ***Задача:*** вставить разрезы в исходные СФЭ Σ_1 и Σ_2, которые позволяют получить РМ с наилучшим рангом
- Ранг разрезающего множество определяется следующим образом:
  - Эквивалентные РМ всегда имеют ранг лучше, чем ранг РМ с неэквивалентными МС.
  - Ранг эквивалентного РМ определяется на основе сравнения весов максимальных конусов (в порядке убывания), порожденных вершинами МС. Например, следующие три решения указаны от лучшего к худшему:

    A: (10,8,3,3) B: (10,8,5) C: (12,1).
  - Ранг РМ с неэквивалентными МО определяется в результате сложения весов всех максимальных конусов всех вершин, входящих в неэквивалентные МО.

Примеры – эквивалентное РМ

- Пусть заданы СФЭ Σ_1 и Σ_2, реализующие булевы функции f и g:
$$f = (11010001), g = (11010001).$$
- Начальное РМ является эквивалентным
- Начальный ранг (без разрезов) – (5,5).

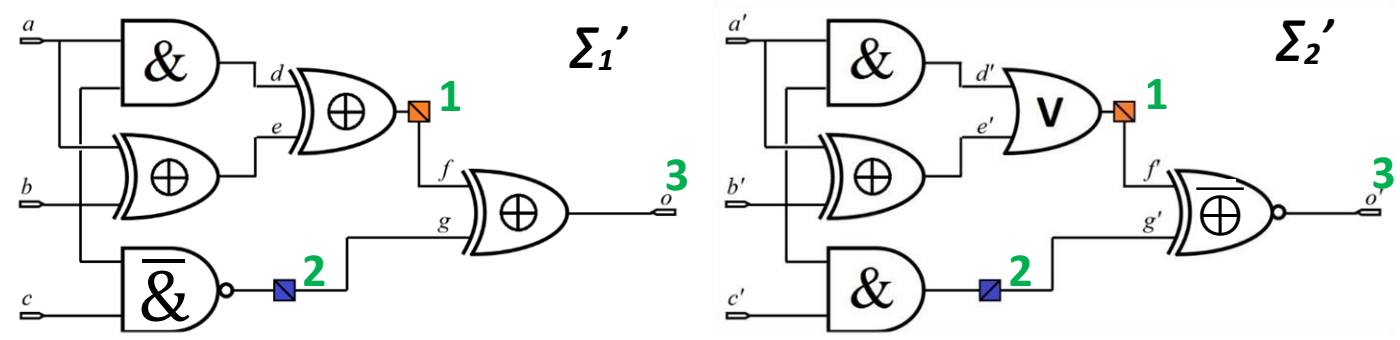




РМ: 1 **(o, o')**

Примеры – эквивалентное РМ

- Решение с введенными разрезами, которое порождает эквивалентное РМ
- Ранг решения – (4,4,2,2,1,1).

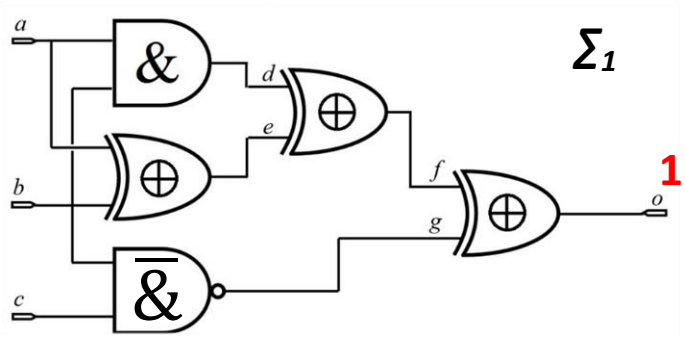


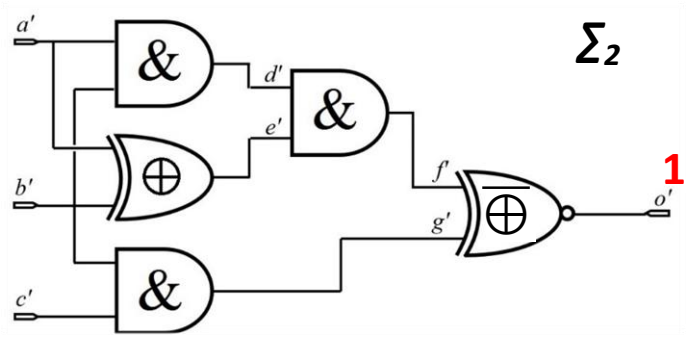
PM: **3 (o, o')** **1 (f, f')**

2 (g, g')

Примеры – неэквивалентное РМ

- Пусть заданы СФЭ Σ_1 и Σ_2, реализующие булевы функции f и g:
$$f = (11010001), g = (11101110).$$
- Исходное РМ не является эквивалентным
- Начальный ранг (без разрезов) : 5 + 5 = 10.

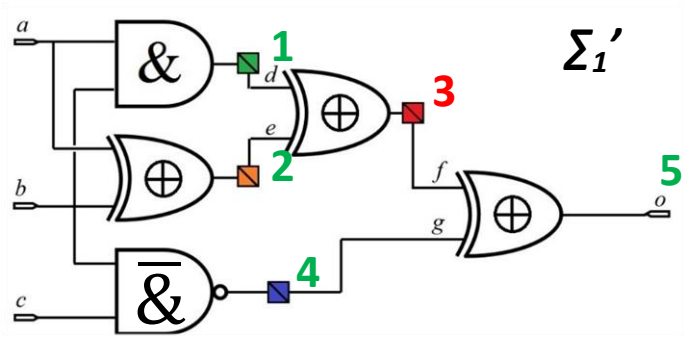


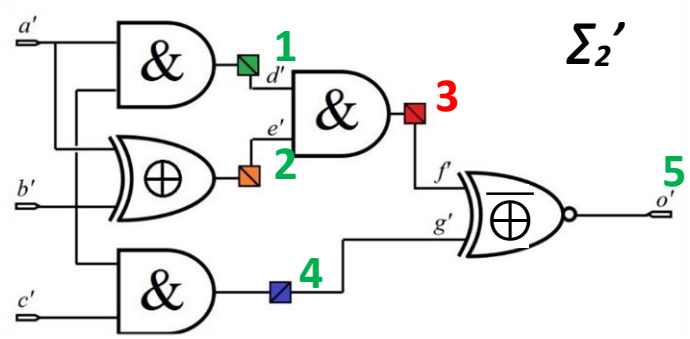


РМ: 1 **(o, o')**

Примеры – неэквивалентное РМ

- Решение с введенными разрезами, которое порождает РМ не являющееся эквивалентным
- Ранг решения: 2 + 2 = 4.





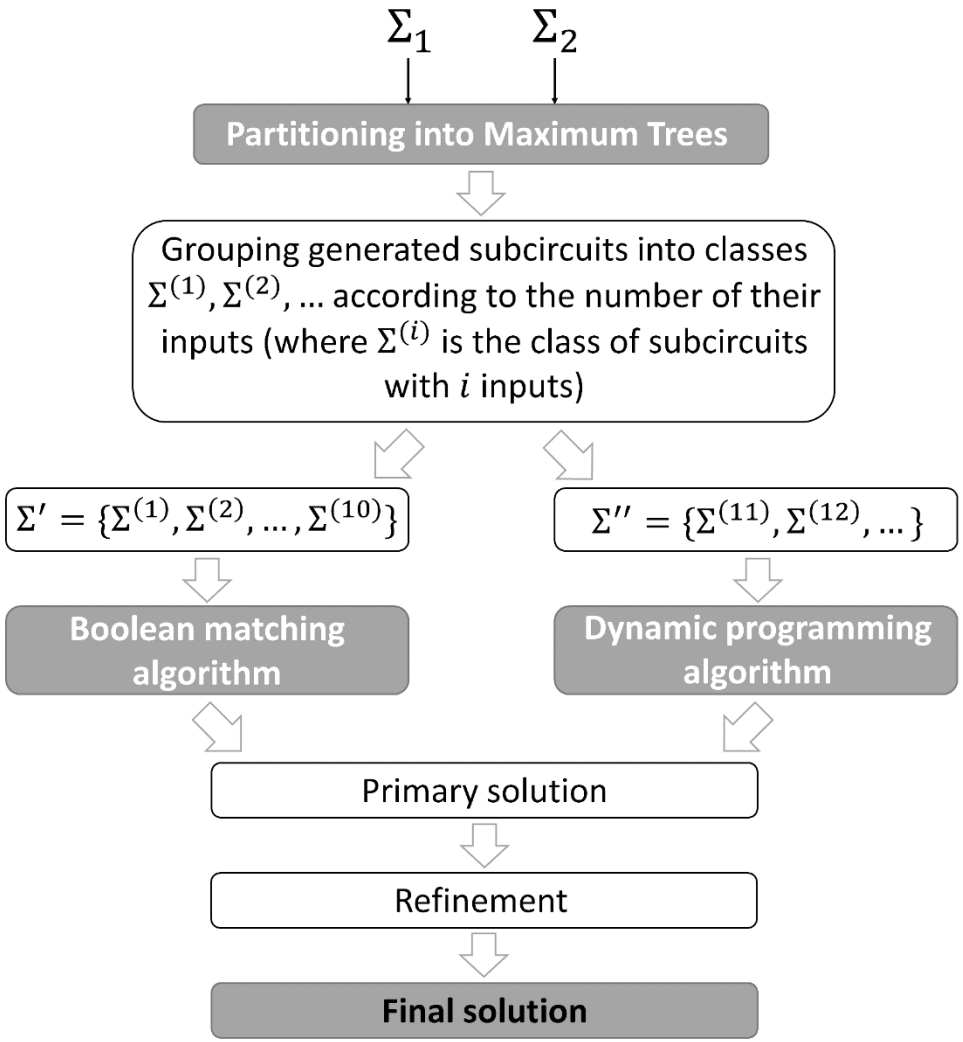
PM: **5 (o, o')** **1 (d, d')**

2 (e, e')

3 (f, f')

4 (g, g')

Общая схема решения



Экспериментальные результаты

Banchmark name	Number of inputs	Number of outputs	Number of nodes in Σ_1	Number of nodes in Σ_2	Total weight of non-equivalent compared sets			
					Initial	ICCAD'15	Antiufeev et al., 2016	Avtaikina et al., 2017
ut2	249	914	13876	10064	503826	41550	33819	19138
ut4	1364	5397	78169	52219	1064521	230280	167985	110228
ut8	2282	2726	51439	91236	14582834	267054	125682	96588
ut9	650	2474	59886	48061	1363736	188662	111966	72697
ut11	56	129	14409	14600	1524769	57660	20581	17212
ut13	99	128	28993	28052	2590770	113618	57010	50522
ut15	99	128	7323	17572	1516255	49356	26566	17367
ut17	128	256	82408	46395	9029081	256658	91325	78795
ut19	160	385	147784	74618	NA	24821951	199749	173663
ut21	80	288	213224	200217	48463859	3051084	194477	185050
ut23	80	192	94707	101754	NA	476021	100934	93448

Экспериментальные результаты

- Относительное сокращение суммарного веса не эквивалентных МС:

$$\left(1 - \frac{Solution\ cost}{Initial\ cost}\right) \times 100\%$$

Benchmark name	Relative reduction of non-equivalent comparison sets' weight, %		
	ICCAD'15	Antiufeev et al., 2016	Avtaikina et al., 2017
ut2	91,8	93,3	96,2
ut4	78,4	84,2	89,65
ut8	98,2	99,1	99.34
ut9	86,4	91,8	94,67
ut11	96,2	98,7	99,89
ut13	95,6	97,8	98,05
ut15	96,7	98,3	98,85
ut17	97,2	99,0	99,13
ut19	NA	NA	NA
ut21	93,7	99,6	99,62
ut23	NA	NA	NA

Студенческие соревнования

- CAD Contest at ICCAD
  - http://cad-contest-2017.el.cycu.edu.tw/CAD-contest-at-ICCAD2017/
- CADathlon at ICCAD
  - http://www.sigda.org/programs/cadathlon
- ISPD CAD Contest
  - http://www.ispd.cc/contests/