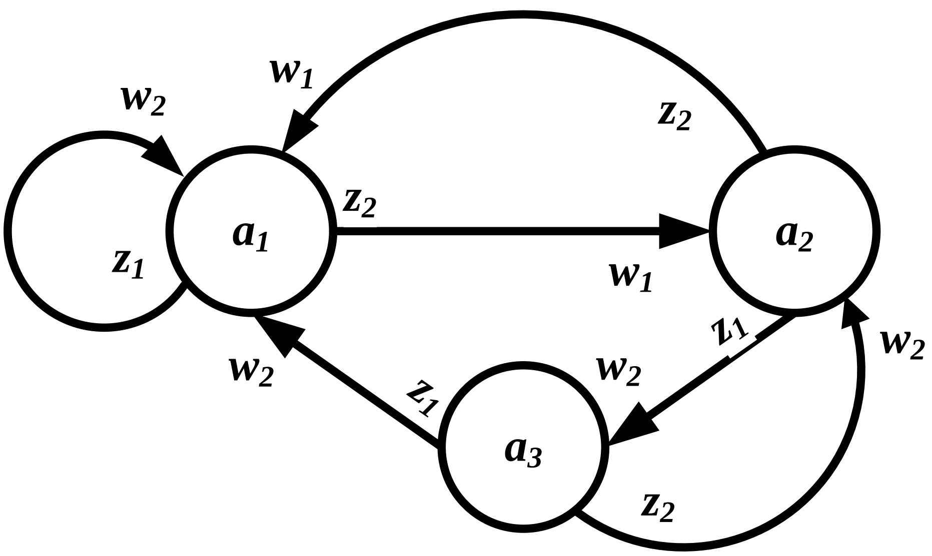
А.А. Ожиганов

ТЕОРИЯ АВТОМАТОВ

Учебное пособие



Санкт-Петербург

2013

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

А.А. Ожиганов

ТЕОРИЯ АВТОМАТОВ

Учебное пособие

Санкт-Петербург

2013

Ожиганов А.А. Теория автоматов. Учебное пособие. – СПб: НИУ ИТМО, 2013. – 84 с.

Целью данного учебного пособия является ознакомление студентов с методами синтеза цифровых автоматов. Приводятся сведения об абстрактных автоматах Мили и Мура. Рассматриваются табличный и графовый способы представления автоматов, вводится понятие реакции автомата на входное слово и определение эквивалентных автоматов. Представлены методы взаимного эквивалентного преобразования автоматов. Приводятся общие сведения о микропрограммном управлении, понятия микрокоманды, микрооперации, микропрограммы, способы представления микропрограмм в виде граф-схем алгоритмов (ГСА), формул переходов, матричных и логических схем алгоритмов. Приводятся методы разметки ГСА и правила построения по ним автоматов Мили и Мура. Рассматриваются методы канонического синтеза структурных автоматов. Приводятся примеры синтеза памяти структурного автомата на базе D –, T –, RS – и JK триггеров.

Пособие предназначено для студентов, специализирующихся в области информационных технологий и может быть использовано при подготовке бакалавров и магистров по направлениям 230100 «Информатика и вычислительная техника», 231000 «Программная инженерия» и инженеров по специальности 230101 «Вычислительные машины, комплексы, системы и сети».

Рекомендовано к печати советом факультета Компьютерных технологий и управления 19 февраля 2013 г., протокол № 2.

В 2009 году Университет стал победителем многоэтапного конкурса, в результате которого определены 12 ведущих университетов России, которым присвоена категория «Национальный исследовательский университет». Министерством образования и науки Российской Федерации была утверждена программа его развития на 2009–2018 годы. В 2011 году Университет получил наименование «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Содержание

1. АБСТРАКТНЫЕ АВТОМАТЫ

1.1. Определение абстрактного автомата

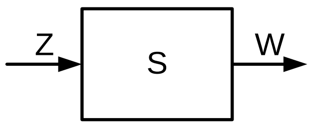
Абстрактный автомат (АА) является математической моделью дискретного устройства и описывается шестикомпонентным набором $S=(A,Z,W,\delta, \lambda,a_1)$,где

1. $A = \{a_1, \dots, a_m, \dots, a_M\}$ - множество состояний или алфавит состояний АА.
2. $Z = \{z_1, \dots, z_f, \dots, z_F\}$ - множество входных сигналов или входной алфавит АА.
3. $W = \{w_1, \dots, w_g, \dots, w_G\}$ - множество выходных сигналов или выходной алфавит АА.
4. δ - функция переходов АА, которая некоторым парам состояние – входной сигнал (a_m, z_f) ставит в соответствие состояние АА a_s, т.е. $a_s = \delta(a_m,z_f)$, $a_s \in A$.
5. λ - функция выходов АА, которая некоторым парам состояние – входной сигнал (a_m, z_f) ставит в соответствие выходной сигнал АА w_g, т.е. $w_g=\lambda(a_m,z_f)$, $w_g \in W$.
6. a_1 - начальное состояние АА.

Под алфавитом понимается непустое множество попарно различимых символов. Буквы - это элементы алфавита. Слово - конечная, упорядоченная последовательность букв.

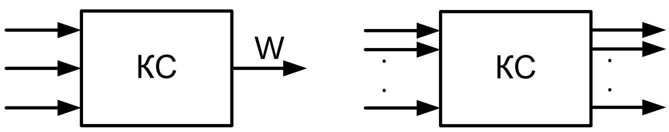
Примечание: бывают и пятикомпонентные наборы. Начальное состояние введено для удобства.

АА имеет один вход и один выход



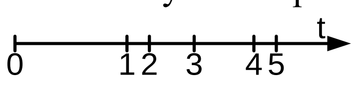
,

в противоположность комбинационным схемам (КС), которые могут иметь несколько входов и один или несколько выходов



.

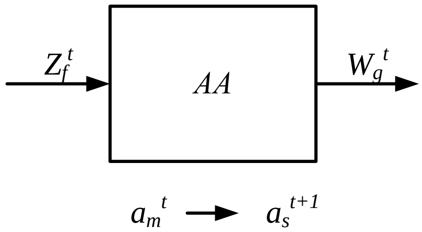
АА работает в дискретные моменты времени, причем промежутки между интервалами времени могут быть различными.



t=0, 1, 2, 3, … . В момент времени t=0 автомат всегда находится в состоянии a_1.

Основным отличием АА от КС является то, что выходные сигналы АА зависят от того, что поступало на его вход раньше.

В каждый момент времени АА, будучи в состоянии a_m^t, способен воспринимать одну из букв входного алфавита z_f^t. В соответствии с функцией переходов δ, АА перейдет в состояние a_1^{t+1} с выдачей выходного сигнала, который вырабатывается в соответствии с функцией выходов λ.



Под работой АА мы будем понимать не более, чем преобразование входных слов в выходные слова.

На уровне АА нас не интересует содержимое прямоугольника, который можно назвать «черным ящиком». Такой подход сродни системному подходу, когда рассматривается поведение системы. При структурном подходе рассматривается устройство на уровне структур.

1.2. Автоматы Мили и Мура

Ниже приведены законы функционирования для автоматов Мили и Мура.

Автомат Мили	Автомат Мура
$a(t+1) = \delta\ (a(t), z(t))$	$a(t+1) = \delta\ (a(t), z(t))$
$w\ (t) = \lambda\ (a(t), z(t))$	$w(t) = \lambda\ (a(t))$

Таким образом, в автоматах Мура выходной сигнал определяется только состоянием автомата в какой-то момент времени и не зависит от входного сигнала в этот же момент времени.

1.3. Способы задания автоматов

1.3.1. Табличный способ задания автомата Мили

Автомат Мили может быть задан таблицей переходов и таблицей выходов.

В таблице переходов АА Мили на пересечении столбца a_m и строки z_f записывается состояние a_s, которое есть функция δ от a_m и z_f.

	a_m	
z_f	a_s	$= \delta(a_m, z_f)$

В таблице выходов АА Мили на пересечении столбца a_m и строки z_f записывается выходной сигнал, который есть функция λ от a_m и z_f.

	a_m	
z_f	w_g	$= \lambda(a_m, z_f)$

Пример: Рассмотрим задание автомата Мили табличным способом (автомат имеет два входных сигнала, два выходных сигнала и три состояния).

Таблица переходов автомата Мили

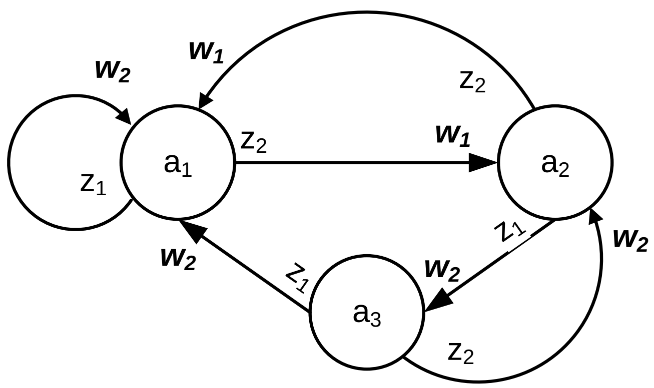
δ	a_1	a_2	a_3
z_1	a_1	a_3	a_1
z_2	a_2	a_1	a_2

Таблица выходов автомата Мили

λ	a_1	a_2	a_3
z_1	w_2	w_2	w_2
z_2	w_1	w_1	w_2

1.3.2. Графический способ задания автомата Мили

Под графом понимается множество вершин, соединенных линиями, причем расстояние между вершинами нас не интересует. Если линии со стрелками, то граф называется направленным (ориентированным), если без стрелок - неориентированным.



На рисунке приведен граф автомата Мили на 3 состояния, имеющий 2 входных сигнала и 2 выходных сигнала (см. предыдущий пример).

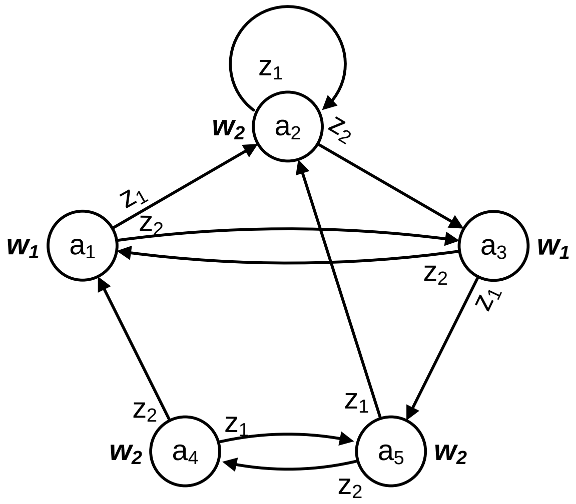
1.3.3. Табличный способ задания автомата Мура

В автомате Мура выходной сигнал зависит только от состояния автомата и не зависит от входного сигнала. Поэтому достаточно для задания автомата Мура в таблице переходов добавить одну строку. Получается отмеченная таблица переходов автомата Мура.

λ	w_1	w_2	w_1	w_2	w_2
δ	a_1	a_2	a_3	a_4	a_5
z_1	a_2	a_2	a_5	a_5	a_2
z_2	a_3	a_3	a_1	a_1	a_4

1.3.4. Графический способ задания автомата Мура

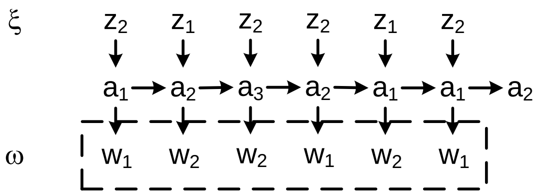
На рисунке приведен граф автомата Мура на 5 состояний, имеющий 2 входных сигнала и 2 выходных сигнала.



1.4. Реакция автоматов на входное слово

Автомат Мили

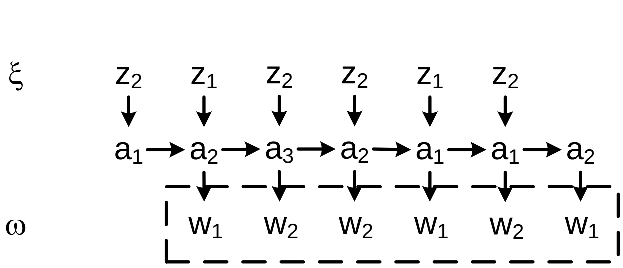
Допустим, входное слово ξ поступает на вход автомата буква за буквой.



Выходное слово ω называется реакцией автомата Мили на входное слово ξ в состоянии a_1 (строится по таблице переходов и выходов).

Реакцию автомата на входное слово ξ можно заменить обходом графа.

Автомат Мура



Выходное слово ω называется реакцией автомата Мура на входное слово ξ в состоянии a_1.

В рассматриваемом примере для автоматов Мили и Мура реакции автоматов на одинаковое входное слово совпадают, но они сдвинуты на один такт.

Автоматы Мили и Мура дающие одинаковые реакции на одинаковые входные слова называются эквивалентными.

Данное замечание приводит к задаче построения эквивалентных автоматов, дающих одинаковые реакции на одинаковые входные слова.

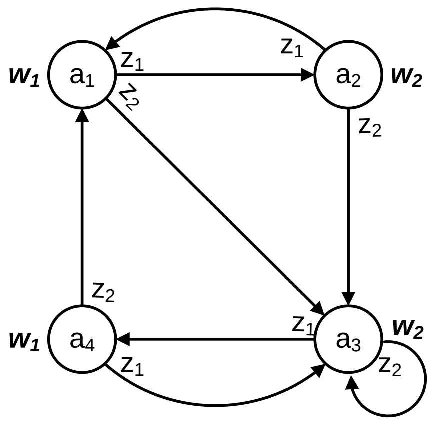
1.5. Взаимная транспозиция автоматов Мили и Мура

1.5.1. Переход от автомата Мура к автомату Мили

Шестикомпонентным набором с индексом *A* будем обозначать автомат Мура, а с индексом *B* - автомат Мили.

И так пусть задан автомат Мура

$$S_A = (A_A, Z_A, W_A, \delta_A, \lambda_A, a_{1A}).$$



Требуется перейти к автомату Мили

$$S_B = (A_B, Z_B, W_B, \delta_B, \lambda_B, a_{1B}),$$

у которого $Z_A = Z_B$, $W_A = W_B$, т.е. входные и выходные алфавиты совпадают.

Рассмотрим пример, в котором

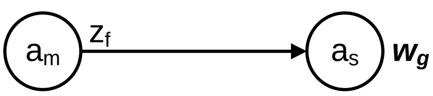
$$Z_A = \{z_1, z_2\} = Z_B,\ W_A = \{w_1, w_2\} = W_B,\ a_{1A} = a_{1B},$$

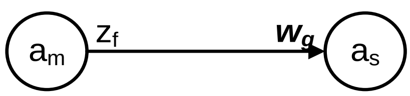
алфавит состояний автомата Мура содержит четыре элемента.

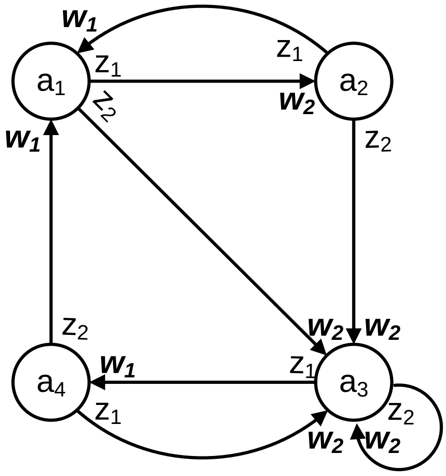
При переходе от автомата Мура к автомату Мили алфавиты состояний также совпадают, т.е. $A_A = A_B$.

Для определения соответствия между функциями переходов выходов автоматов Мура и Мили воспользуемся следующей вспомогательной таблицей

Мура	Мили
$\delta_A(a_m, z_f) = a_s$	$\delta_B(a_m, z_f) = a_s$
$\lambda_A(a_m) = w_g$	$\lambda_B(a_m, z_f) = w_g$







При переходе от автомата Мура к автомату Мили функции переходов также совпадают, а для определения функции выходов выходные сигналы с вершин опускается на входные дуги.

Проделав такие преобразования мы должны доказать, что получили автомат Мили, эквивалентный автомату Мура, т.е. что реакции автоматов на одинаковые входные воздействия совпадают.

Доказательство проводится по индукции, начиная со случая (смотри таблицу) и, далее, увеличивая слова на 1 получим доказательство.

При таком переходе (Мура к Мили) число состояний совпадает.

1.5.2. Переход от автомата Мили к автомату Мура

Пусть задан автомат Мили

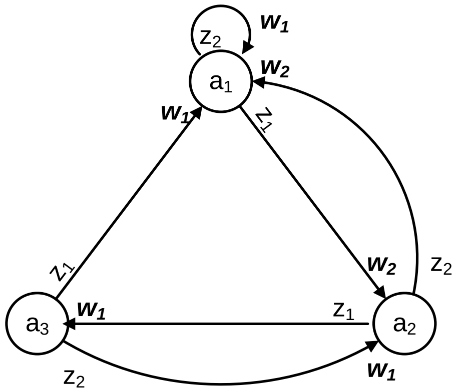
$$S_B = (A_B, Z_B, W_B, \delta_B, \lambda_B, a_{1B}).$$

Требуется перейти к автомату Мура

$$S_A = (A_A, Z_A, W_A, \delta_A, \lambda_A, a_{1A}),$$

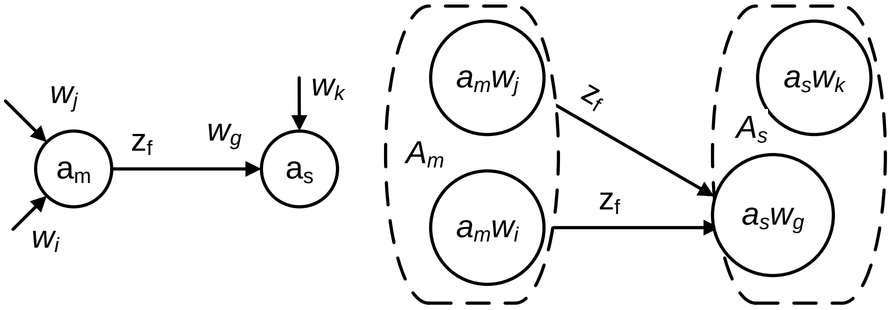
у которого $Z_B = Z_A$;$W_B = W_A$, т.е. входные и выходные алфавиты совпадают.

Рассмотрим пример, в котором $Z_B = \{z_1,z_2\}=Z_A$, $W_B =\{w_1,w_2\}=W_A$, алфавит состояний автомата Мили содержит три элемента.



Для определения алфавита состояний, функций переходов и выходов автомата Мура воспользуемся следующей вспомогательной таблицей.

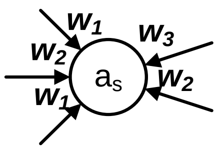
Мили Мура



В данном случае $A_B \neq A_A$. При таком переходе (Мили к Мура) каждому состоянию автомата Мили a_s ставится в соответствие множество всевозможных пар

$$a_s \rightarrow A_s = \{(a_s, w_g) \mid a_s = \delta(a_m, z_f), w_g = \lambda(a_m, z_f)\}$$

таких, что a_s есть функция δ от состояния и входного сигнала, w_g функция λ от состояния и входного сигнала.
Пример:



$$A_s = \{(a_s, w_1), (a_s, w_2), (a_s, w_3)\}.$$

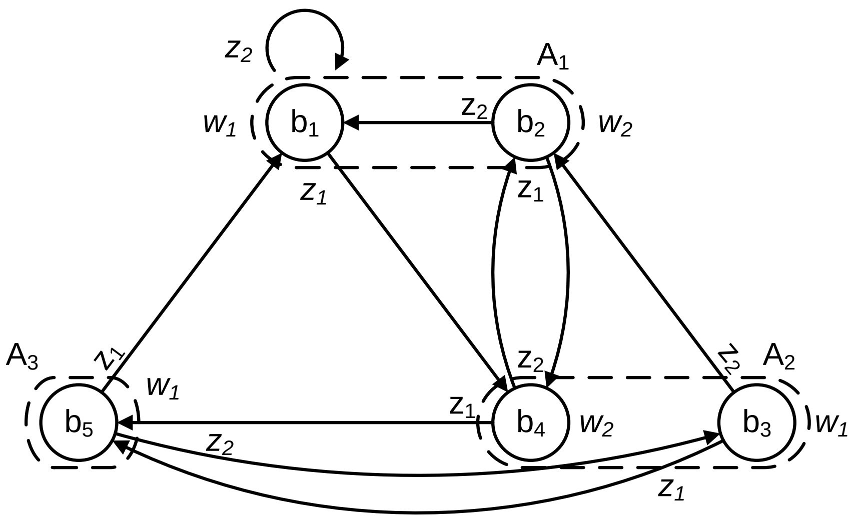
Для состояния

$$a_1 : A_1 = \begin{cases} (a_1, w_1) = b_1 \\ (a_1, w_2) = b_2 \end{cases}, a_2 : A_2 = \begin{cases} (a_2, w_1) = b_3 \\ (a_2, w_2) = b_4 \end{cases}, \qquad a_3 : A_3 = \{(a_3, w_1)\} = b_5.$$

В качестве начального состояния результирующего автомата может быть выбрано любое состояние Мура, порожденное начальным состоянием автомата Мили, т.е. состояния b_1 или b_2.

При определении функции переходов результирующего автомата Мура из всех состояний, порожденных одним состоянием автомата Мили, должны быть переходы под воздействием одинаковых входных сигналов.

Поскольку в автомате Мура выходной сигнал зависит только от состояния автомата, то в примере рядом с состояниями проставим соответствующие выходные сигналы (см. рисунок ниже).



И так если осуществить следующие преобразования, то получим.

Мили	Мура	Мили
$S_1 \rightarrow$	$S_2 \rightarrow$	S_3
3 состояния	5состояний	5состояний

Можно утверждать, что если S_1 эквивалентно S_2, а S_2 эквивалентно S_3, то S_1 эквивалентно S_3 (т.е. эквивалентность обладает свойством транзитивности).

Таким образом возникает задача минимизации автоматов, под которой понимается задача нахождения в классе всех эквивалентных автоматов автомата того же типа (Мили или Мура) с минимальным числом состояний.

1.6. Минимизация полностью определенных абстрактных автоматов

Эквивалентные автоматы могут иметь различное число состояний. В связи с этим возникает задача нахождения минимального (с минимальным числом состояний) автомата в классе эквивалентных между собой автоматов.

Рассмотрим алгоритм минимизации абстрактного автомата, предложенный Ауфенкампом и Хоном. Основная идея алгоритма состоит в разбиении всех состояний исходного абстрактного автомата на попарно не пересекаемые классы эквивалентных состояний. После разбиения происходит замена каждого класса эквивалентности одним состоянием. Получившийся в результате минимальный абстрактный автомат имеет столько же состояний, на сколько классов эквивалентности разбивается состояние исходного абстрактного автомата.

Два состояния a_m и a_s называются эквивалентными, (обозначаются $a_m \equiv a_s$), если $w(a_m, \xi) = w(a_s, \xi)$ для всевозможных входных слов ξ, т.е.

реакции автомата в этих состояниях на всевозможные входные слова совпадают. Если a_m и a_s не эквивалентны, то они различимы.

Более слабой эквивалентностью является k - эквивалентность. Состояния a_m и a_s k– эквивалентны ($a_m \overset{k}{\equiv} a_s$), если выходные функции состояний a_m и a_s равны для всевозможных входных слов ξ_k длины k. Если состояния не k - эквивалентны, то ни k - различимы.

Понятие эквивалентности и k - эквивалентности используется для разбиения множества на попарно различимые непересекающиеся классы эквивалентности.

Обозначение:

1. Эквивалентность–П;
2. k - эквивалентность - Π_k.

Разбиение состояний автомата на классы эквивалентности позволяет определить избыточные элементы в множестве состояний A.

Пусть состояния a_m и a_s эквивалентны. Это значит, что с точки зрения реакции автоматов на всевозможные входные слова не важно, в каком состоянии находится автомат a_m или a_s. Поэтому одно из этих состояний может быть удалено из алфавита состояний, например a_s.

Если каждый класс эквивалентности содержит только одно состояние, то множество A не сократимо.

Если один или несколько классов содержат более одного элемента, то все элементы в классах эквивалентности, кроме одного, могут быть исключены из A.

В результате получается автомат с минимальным числом состояний.

Алгоритм минимизации числа состояний автомата $S=(A, Z, W, \delta, \lambda, a_1)$ состоит из следующих шагов:

1. Находятся последовательные разбиения Π_1, Π_2, ... , Π_k, Π_{k+1} множества A на классы одно-, двух-, ... , k+1 - эквивалентных между собой состояний. Разбиение на классы производится до тех пор, пока на каком-то k+1 шаге не окажется, что $\Pi_{k+1} = \Pi_k$. Можно показать, что тогда разбиение $\Pi_k = \Pi$, то есть что k - эквивалентные состояния являются эквивалентными, и число шагов k, при котором $\Pi_k = \Pi$, не превышает M-1, где M - число состояний в множестве $A=\{a_1,a_2,\ldots,a_m\}$.
2. В каждом классе эквивалентности разбиения П выбираются по одному элементу, которые образуют множество A' состояний минимального автомата $S' = (A', Z, W, \delta', \lambda', a_1')$, эквивалентного исходному автомату S.
3. Функции переходов и выходов автомата S', определяются на множестве $A' *Z$, то есть δ': $A' *Z \rightarrow A'$, λ' : $A' *Z \rightarrow W$. Для этого в таблицах переходов и выходов вычеркиваются столбцы, соответствующие не вошедшим в множество A' состояниям, а в

оставшихся столбцах таблицы переходов все состояния заменяются на эквивалентные из множества A'.

4. В качестве a_1', выбирается одно из состояний, эквивалентных a_1

Пример. Задан автомат Мура имеющий три входных сигнала $Z=\{z_1,z_2,z_3\}$, два выходных сигнала $W = \{w_1, w_2\}$ и семь состояний автомата $A = \{a_1, a_2, \dots , a_6, a_7\}$. Используя алгоритм минимизировать автомат.

λ	w_1	w_1	w_2	w_2	w_1	w_2	w_1
δ	a_1	a_2	a_3	a_4	a_5	a_6	a_7
z_1	a_2	a_1	a_3	a_4	a_7	a_1	a_5
z_2	a_4	a_3	a_4	a_1	a_3	a_2	a_4
z_3	a_6	a_5	a_2	a_7	a_2	a_6	a_6

Находятся классы одноэквивалентных состояний (это делается по таблице выходов)

$B_1 = \{a_1, a_2, a_5, a_7\}$.

$B_2 = \{a_3, a_4, a_6\}$.

$\Pi_1 = \{B_1, B_2\}$.

Определяются классы k - эквивалентных состояний. Строим таблицу $\Pi_1(\Pi_k)$, заменяя состояния в таблице переходов соответствующими классами эквивалентности.

Проводится проверка. Если $\Pi_k = \Pi_{k+1}$, то перейти к пункту 4. Если не совпали, то вернуться к пункту 2 и находить соответствующие классы эквивалентности пока проверка не будет выполнена.

	B_1				B_2		
	a_1	a_2	a_5	a_7	a_3	a_4	a_6
z_1	B_1	B_1	B_1	B_1	B_2	B_2	B_2
z_2	B_2	B_2	B_2	B_2	B_2	B_1	B_1
z_3	B_2	B_1	B_1	B_2	B_1	B_1	B_2

$C_1 = \{ a_1, a_7 \};$

$C_2 = \{ a_2, a_5 \};$

$C_3 = \{ a_3 \};$

$C_4 = \{ a_4 \};$

$C_5 = \{ a_6 \};$

$\Pi_2 = \{C_1, C_2, C_3, C_4, C_5 \};\qquad \Pi_1 \neq \Pi_2;$

	C_1		C_2		C_3	C_4	C_5
	a_1	a_7	a_2	a_5	a_3	a_4	a_6
z_1	C_2	C_2	C_1	C_1	C_3	C_4	C_1
z_2	C_4	C_4	C_3	C_3	C_4	C_4	C_2
z_3	C_5	C_5	C_2	C_2	C_2	C_1	C_5

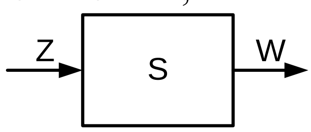
$Д_1 = \{ a_1, a_7 \};$
$Д_2 = \{ a_2, a_5 \};$
$Д_3 = \{ a_3 \};$
$Д_4 = \{ a_4 \};$
$Д_5 = \{ a_6 \};$
$П_3 = \{ Д_1, Д_2, Д_3, Д_4, Д_5 \};$
$П_2 = П_3;$
$a_1 \equiv a_7;$
$a_2 \equiv a_5.$

Автомат после минимизации:

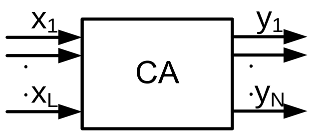
л	w_1	w_1	w_2	w_2	w_2
д	a_1	a_2	a_3	a_4	a_6
z_1	a_2	a_1	a_3	a_4	a_1
z_2	a_4	a_3	a_4	a_1	a_2
z_3	a_6	a_2	a_2	a_1	a_6

1. СТРУКТУРНЫЙ АВТОМАТ

В отличие от абстрактного автомата,



в структурном автомате (СА) имеется множество входных и выходных каналов или полюсов.



Множество сигналов, поступающих на каждый из входов СА, образуют структурный входной алфавит. Множество сигналов, вырабатывающихся на каждом из выходов СА, образуют структурный

2.1. Переход от абстрактного автомата к структурному автомату

При переходе от АА к СА необходимо закодировать входные сигналы, выходные сигналы и состояния автомата.

Каждому входному сигналу АА z_f ставится в соответствие двоичный вектор

$$z_f \rightarrow (e_{f1}, \ldots, e_{fl}, \ldots, e_{fL}), e_{fl} \in \{0, 1\}, L =]log_2 F[,$$

где F - число входных сигналов АА. $]b[$-запись означает ближайшее целое число большее b или равное ему, если b целое.

Каждому выходному сигналу АА w_g ставится в соответствие двоичный вектор

$$w_g \rightarrow (e_{g1}, \ldots, e_{gn}, \ldots, e_{gN}), e_{gn} \in \{0, 1\}, N =]\, log_2 G\, [,$$

где G - число выходных сигналов АА.

Каждому состоянию АА a_m ставится в соответствие двоичный вектор

$$a_m \rightarrow (e_{m1}, \ldots, e_{mr}, \ldots, e_{mR}), a_m \in \{0, 1\}, R =]\, log_2 M\, [,$$

где M - число состояний АА.

Пример: *рассмотрим абстрактный автомат Мили, заданный таблично, на три состояния, три входных сигнала и пять выходных сигналов. Необходимо перейти от АА к СА.*

$T_1(\delta)$	a_1	a_2	a_3
z_1	a_2	-	a_1
z_2	a_3	a_1	-
z_3	a_2	a_3	a_3

$T_2(\lambda)$	a_1	a_2	a_3
z_1	w_1	-	w_2
z_2	w_4	w_5	-
z_3	w_2	w_1	w_3

Закодируем входные сигналы.

T_3	x_1	x_2
z_1	0	0
z_2	0	1
z_3	1	0

Для нашего структурного автомата будет достаточно двух входных сигналов при условии кодирования двоичным кодом.

Закодируем выходные сигналы.

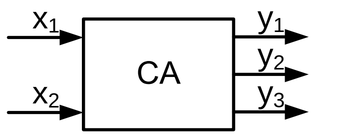
T_4	y_1	y_2	y_3
w_1	*0*	*0*	*1*
w_2	*0*	*1*	*0*
w_3	*0*	*1*	*1*
w_4	*1*	*0*	*0*
w_5	*1*	*0*	*1*

Для нашего структурного автомата будет достаточно трех выходных сигналов при условии кодирования двоичным кодом.

Закодируем состояния автомата.

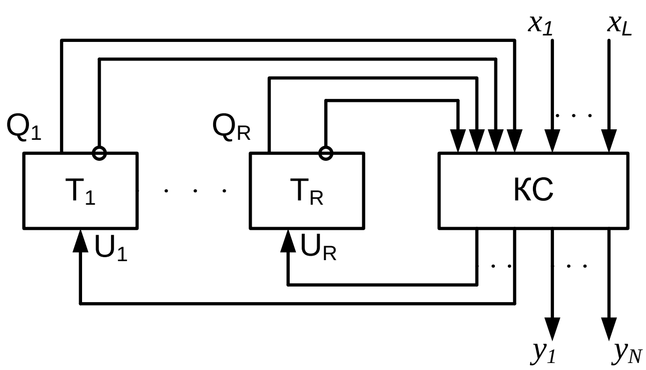
T_5	Q_1	Q_2
a_1	*0*	*0*
a_2	*0*	*1*
a_3	*1*	*1*

Для нашего примера СА будет выглядеть так



2.2. Канонический метод структурного синтеза

При каноническом (стандартном) методе структурного синтеза автомат представляют в виде двух частей: памяти (П) и комбинационной схемы (КС). КС состоит из логических элементов, а память из автоматов памяти. Элементарными автоматами памяти являются триггерные схемы.



На вход КС поступают входные сигналы $x_1,\ldots,x_l,\ldots, x_L$ и сигналы обратной связи с элементов памяти автомата $Q_1,\ldots,Q_r,\ldots,Q_R$. На основании закона функционирования автомата в заданном базисе строится КС, которая вырабатывает выходные сигналы функций возбуждения $U_1,\ldots,U_r,\ldots,U_R$ элементов памяти.

Теорема о структурной полноте. Любая система элементарных автоматов, содержащая автомат Мура (обладающий полнотой переходов и полнотой выходов) и какую-либо функционально полную систему логических элементов, является структурно полной.

Полнота переходов автомата Мура означает, что для любой пары состояний (b_i, b_j) найдется входной сигнал, который переведет автомат из состояния b_i в состояние b_j. Другими словами, это означает, что в каждом из столбцов отмеченной таблицы переходов АА Мура должны быть перечислены все состояния. Полнота выходов автомата Мура означает, что каждому состоянию автомата соответствует свой отличный от других выходной сигнал.

Если автомат Мура обладает полнотой выходов, то можно отказаться от алфавита выходных сигналов и для обозначения последних использовать алфавит состояний.

При наличии структурно полной системы задача синтеза любого автомата сводится к задаче синтеза его комбинационной части.

Таким образом, задача синтеза автомата сводится к нахождению функций

$$U_r = U_r(Q_1,\ldots, Q_R, x_1,\ldots, x_L),\ r = 1,\ldots, R;$$
$$y_n = y_n(Q_1,\ldots, Q_R, x_1,\ldots, x_L),\ n = 1,\ldots, N.$$

То есть применительно к нашему примеру

$$U_1 = U_1\,(Q_1,Q_2, x_1,x_2);$$
$$U_2 = U_2\,(Q_1,Q_2, x_1,x_2);$$
$$y_1 = y_1\,(Q_1,Q_2, x_1,x_2);$$
$$y_2 = y_2\,(Q_1,Q_2, x_1,x_2);$$
$$y_3 = y_3\,(Q_1,Q_2, x_1,x_2).$$

Будем считать, что имеем функционально-полную систему логических элементов (универсальный базис) И, ИЛИ, НЕ и автомат Мура, обладающий полнотой переходов и полнотой выходов.

Т$_6$	b_1	b_2
q_1	b_1	b_2
q_2	b_2	b_1

Необходимо перейти к СА. Для этого закодируем входные и состояния автомата

T_7	U
q_1	*0*
q_2	*1*

T_8	Q
b_1	*0*
b_2	*1*

Так как автомат Мура обладает полнотой переходов и полнотой выходов, то для обозначения выходных сигналов будем использовать алфавит состояний автомата.

На основании таблиц 6, 7 и 8 строим таблицу 9, которая является структурной таблицей, описывающей функционирование автомата памяти:

T_9 Q / U	*0*	*1*
0	*0*	*1*
1	*1*	*0*

По таблицам 1 и 2 с учетом таблиц 3, 4 и 5 строим таблицу переходов 10 и таблицу выходов 11 структурного автомата. В этих таблицах входные, выходные переменные и состояния автомата заменены их кодами. Это будут таблицы, описывающие функционирование структурного автомата.

T_{10} $Q_1\, Q_2 = \delta\,(Q_1, Q_2, x_1, x_2)$

Q_1Q_2 / x_1x_2	*00*	*01*	*11*
00	*01*	-	*00*
01	*11*	*00*	-
10	*01*	*11*	*11*

T_{11} $y_1\, y_2\, y_3 = \lambda\,(Q_1, Q_2, x_1, x_2)$

Q_1Q_2 / x_1x_2	*00*	*01*	*11*
00	*001*	-	*010*
01	*100*	*101*	-
10	*010*	*001*	*011*
	y_1 y_2 y_3	y_1 y_2 y_3	y_1 y_2 y_3

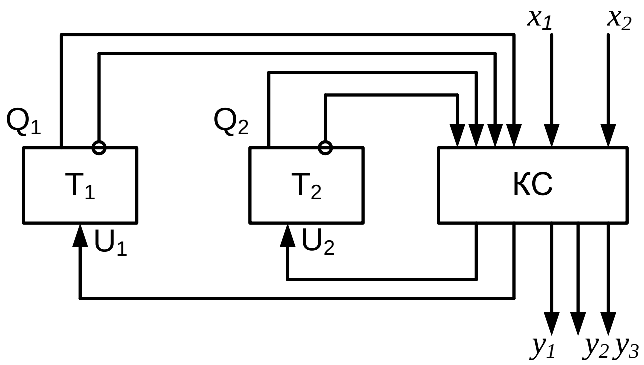
По таблице 11 строим дизъюнктивные нормальные формы для y_1, y_2 и y_3.

$$y_1 = \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1Q_2\bar{x}_1x_2 = 1 \vee 5;$$

$$y_2 = \bar{Q}_1\bar{Q}_2x_1\bar{x}_2 \vee Q_1Q_2\bar{x}_1\bar{x}_2 \vee Q_1Q_2x_1\bar{x}_2 = 2 \vee 12 \vee 14;$$

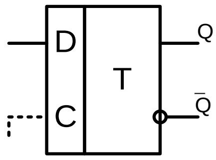
$$y_3 = \bar{Q}_1\bar{Q}_2\bar{x}_1\bar{x}_2 \vee \bar{Q}_1Q_2\bar{x}_1x_2 \vee \bar{Q}_1Q_2x_1\bar{x}_2 \vee Q_1Q_2x_1\bar{x}_2 = 0 \vee 2 \vee 6 \vee 14.$$

В общем виде структурная схема автомата для нашего примера будет выглядеть следующим образом.



2.2.1. Синтез автоматов на *D* -, *T* -, *RS* -, *JK* - триггерах

Напомним, что любой триггер - это автомат Мура, обладающий полнотой переходов и полнотой выходов.
D– триггер (задержка).



C - синхронизирующий вход *D* - триггера. Сигнал на выходе *Q* этого триггера повторяет сигнал на его входе *D*.

Закон функционирования *D*–триггера отражен в таблице ниже.

Q / *D*	*0*	*1*
0	*0*	*0*
1	*1*	*1*

По табл. 10 с учетом закона функционирования *D*–триггера строится таблица сигналов функций возбуждения

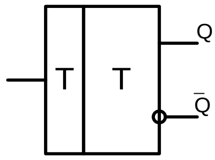
$D_1D_2 = \mu(Q_1, Q_2, x_1, x_2)$			
Q_1Q_2 / x_1x_2	*00*	*01*	*11*
00	*01*	-	*00*
01	*11*	*00*	-
10	*01*	*11*	*11*
	D_1 D_2	D_1 D_2	D_1 D_2

При синтезе автомата на D - триггерах таблица сигналов функций возбуждения совпадает с таблицей переходов СА (табл. 10), из которой сразу можно получить выражения в виде дизъюнктивных нормальных форм для сигналов функций возбуждения D_1 и D_2, т.е.

$$D_1 = \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1Q_2x_1\bar{x}_2 \vee Q_1Q_2x_1\bar{x}_2 = 1 \vee 6 \vee 14.$$

$$D_2 = \bar{Q}_1\bar{Q}_2\bar{x}_1\bar{x}_2 \vee \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1\bar{Q}_2x_1\bar{x}_2 \vee \bar{Q}_1Q_2x_1\bar{x}_2 \vee Q_1Q_2x_1\bar{x}_2 = 0 \vee 1 \vee 2 \vee 6 \vee 14.$$

T-триггер (триггер со счетным входом - переключатель).



Закон функционирования T–триггера отражен в таблице ниже ($Q^{t+1}=Q^t \oplus T$).

T \ *Q*	*0*	*1*
0	*0*	*1*
1	*1*	*0*

Если входной сигнал триггера равен 0,то состояние триггера не меняется.

По табл. 10 с учетом закона функционирования T – триггера строится таблица сигналов функций возбуждения

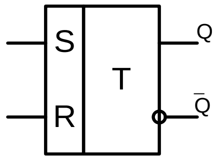
$T_1T_2 = \mu(Q_1, Q_2, x_1, x_2)$			
x_1x_2 \ Q_1Q_2	*0 0*	*0 1*	*1 1*
00	*0 1*	-	*11*
01	*1 1*	*0 1*	-
10	*0 1*	*1 0*	*00*
	T_1 T_2	T_1 T_2	T_1 T_2

По этой таблице можно получить выражения в виде дизъюнктивных нормальных форм для сигналов функций возбуждения T_1 и T_2, т.е.

$$T_1 = \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1Q_2x_1\bar{x}_2 \vee Q_1Q_2\bar{x}_1\bar{x}_2 = 1 \vee 6 \vee 12.$$

$$T_2 = \bar{Q}_1\bar{Q}_2\bar{x}_1\bar{x}_2 \vee \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1\bar{Q}_2x_1\bar{x}_2 \vee \bar{Q}_1Q_2\bar{x}_1x_2 \vee Q_1Q_2\bar{x}_1\bar{x}_2 = 0 \vee 1 \vee 2 \vee 5 \vee 12.$$

RS – триггер (установить – сбросить).



Закон функционирования *RS* – триггера отражен в таблице ниже

RS \ *Q*	*0*	*1*
00	*0*	*1*
01	*1*	*1*
10	*0*	*0*
11	-	-

Для удобства дальнейшего изложения воспользуемся системой подставок *RS* - триггера.

	R	*S*
"0" → "0"	-	0
"0" → "1"	0	1
"1" → "0"	1	0
"1" → "1"	0	-

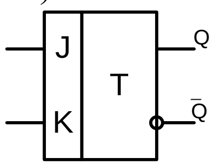
По табл. 10 с учетом системы подставок *RS* - триггера строится таблица сигналов функций возбуждения R_1, S_1, R_2, S_2

$R_1S_1R_2S_2 = \mu(Q_1, Q_2, x_1, x_2)$						
x_1x_2 \ Q_1Q_2	*00*		*01*		*11*	
00	*-0*	*01*	-	-	*10*	*10*
01	*01*	*01*	*-0*	*10*	-	-
10	*-0*	*01*	*01*	*0-*	*0-*	*0-*
	R_1S_1	R_2S_2	R_1S_1	R_2S_2	R_1S_1	R_2S_2

По этой таблице можно получить выражения в виде дизъюнктивных нормальных форм для сигналов функций возбуждения R_1, S_1, R_2, S_2, т.е.

$$R_1 = Q_1 Q_2 \bar{x}_1 \bar{x}_2 = 12.$$
$$S_1 = \bar{Q}_1 \bar{Q}_2 \bar{x}_1 x_2 \vee \bar{Q}_1 Q_2 x_1 \bar{x}_2 = 1 \vee 6.$$
$$R_2 = \bar{Q}_1 Q_2 \bar{x}_1 x_2 \vee Q_1 Q_2 \bar{x}_1 \bar{x}_2 = 5 \vee 12.$$
$$S_2 = \bar{Q}_1 \bar{Q}_2 \bar{x}_1 \bar{x}_2 \vee \bar{Q}_1 \bar{Q}_2 \bar{x}_1 x_2 \vee \bar{Q}_1 \bar{Q}_2 x_1 \bar{x}_2 = 0 \vee 1 \vee 2.$$

JK – триггер (прыгать-держать).



Закон функционирования *JK* – триггера отражен в таблице ниже

JK \ *Q*	*0*	*1*
00	*0*	*1*
01	*0*	*0*
10	*1*	*1*
11	*1*	*0*

Для удобства дальнейшего изложения воспользуемся системой подставок *JK* - триггера.

	J	*K*
"0" → "0"	0	-
"0" → "1"	1	-
"1" → "0"	-	1
"1" → "1"	-	0

По табл. 10 с учетом системы подставок *JK* - триггера строится таблица сигналов функций возбуждения J_1, K_1, J_2, K_2

$J_1K_1J_2K_2 = \mu(Q_1, Q_2, x_1, x_2)$						
x_1x_2 \ Q_1Q_2	*00*		*01*		*11*	
00	*0-*	*1-*	*-*	*-*	*-1*	*-1*
01	*1-*	*1-*	*0-*	*-1*	*-*	*-*
10	*0-*	*1-*	*1-*	*-0*	*-0*	*-0*
	J_1K_1	J_2K_2	J_1K_1	J_2K_2	J_1K_1	J_2K_2

По этой таблице можно получить выражения в виде дизъюнктивных нормальных форм для сигналов функций возбуждения J_1, K_1, J_2, K_2, т.е.

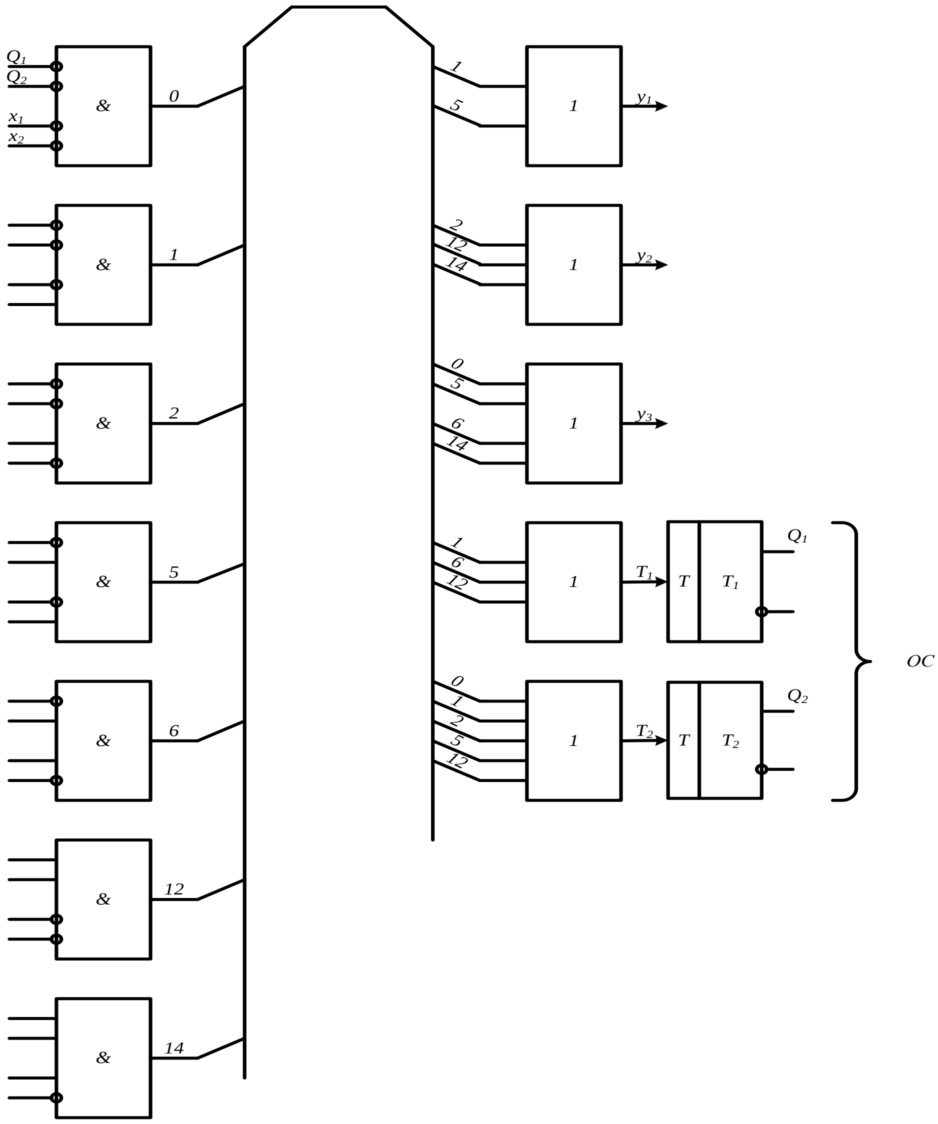
$$J_1 = \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1Q_2x_1\bar{x}_2 = 1\vee 6.$$

$$K_1 = Q_1Q_2\bar{x}_1\bar{x}_2 = 12.$$

$$J_2 = \bar{Q}_1\bar{Q}_2\bar{x}_1\bar{x}_2 \vee \bar{Q}_1\bar{Q}_2\bar{x}_1x_2 \vee \bar{Q}_1\bar{Q}_2x_1\bar{x}_2 = 0\vee 1\vee 2.$$

$$K_2 = \bar{Q}_1Q_2\bar{x}_1x_2 \vee Q_1Q_2\bar{x}_1\bar{x}_2 = 5\vee 12.$$

Для примера построим функциональную схему автомата на T - триггерах.



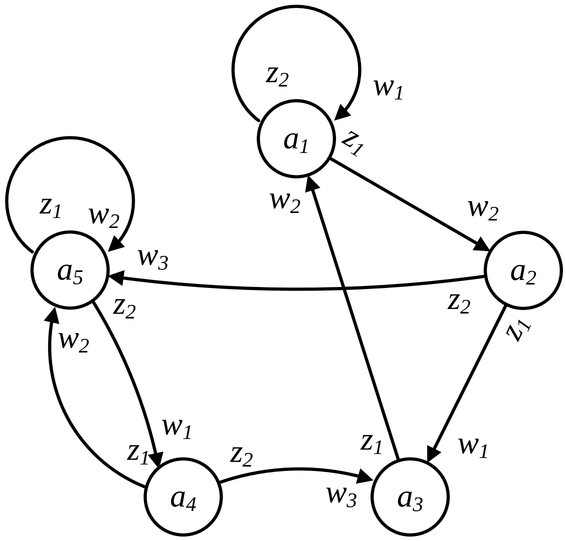
Для минимизации схем ИЛИ, реализующих выходные функции y_1, $y_2, \ldots, y_N$ используется следующий алгоритм:

1. Каждому w_g ставится в соответствие целое число N_g, равное числу появлений w_g в таблице выходов.
2. При кодировании w_g упорядочиваем их по мере убывания N_g.
3. Выходной сигнал, имеющий самое большое N_g, кодируется кодом из всех 0.

4. Последующие выходные сигналы кодируются кодами с одной единицей, затем с двумя и т.д.

2.3. Графический метод структурного синтеза

Рассмотрим синтез автомата на D - триггерах, который задан графом.



Как и в случае задания автомата табличным способом, при переходе АА к СА необходимо закодировать входные сигналы, выходные сигналы и состояния автомата.

Входные сигналы	
	x
z_1	*0*
z_2	*1*

Закодируем выходные сигналы в соответствии с алгоритмом, рассмотренным выше.

w_g	N_g
w_1	*3*
w_2	*4*
w_3	*2*

Выходные сигналы		
	y_1	y_2
w_2	*0*	*0*
w_1	*0*	*1*
w_3	*1*	*0*

При кодировании состояний автомата для минимизации числа конъюнкций в выражениях для $D_1,\ldots,D_R$ используется следующий алгоритм.

1. Каждому состоянию a_i ставится в соответствие целое число M_i, равное числу переходов в это состояние.
2. При кодировании a_i упорядочиваем их по убыванию M_i.
3. Состояние, имеющее самое большое M_i, кодируется всеми нулями.
4. Для кодирования следующих состояний используются коды с одной единицей, затем с двумя и т.д.

Закодируем состояние автомата в соответствии с рассмотренным алгоритмом.

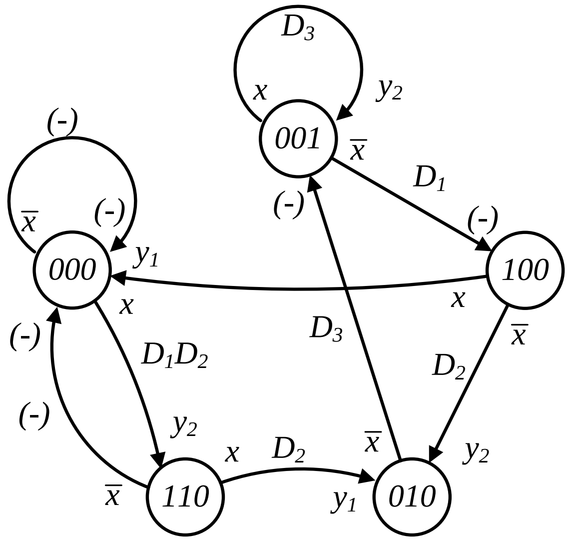
a_i	M_i
a_1	2
a_2	1
a_3	2
a_4	1
a_5	3

	Q_1	Q_2	Q_3
a_5	0	0	0
a_1	0	0	1
a_3	0	1	0
a_2	1	0	0
a_4	1	1	0

При переходе от АА, заданному в виде графа, к СА, условимся:
1. Вместо входных сигналов (в примере 0,1) записывать на дугах графа конъюнкции (в примере $x, \bar{x}$).
2. Вместо выходных сигналов (в примере 00, 01, 10) будем записывать на дугах графа автомата символы тех компонент, которые в этих сигналах принимают значение 1 (в примере y_1, y_2).

В соответствии с кодированием входных сигналов, выходных сигналов и состояний автомата построим граф СА.

Далее припишем дугам графа символы тех компонент функций возбуждения, которые принимают значения 1 на соответствующих переходах.



Затем, прямо по графу, построим дизъюнктивные нормальные формы для y_1, y_2 и D_1, D_2, D_3.

$$y_1 = Q_1\bar{Q}_2\bar{Q}_3 x \vee Q_1 Q_2 \bar{Q}_3 x = 9 \vee 13;$$

$$y_2 = \bar{Q}_1\bar{Q}_2 Q_3 x \vee Q_1\bar{Q}_2\bar{Q}_3\bar{x} \vee \bar{Q}_1\bar{Q}_2\bar{Q}_3 x = 3 \vee 8 \vee 1;$$

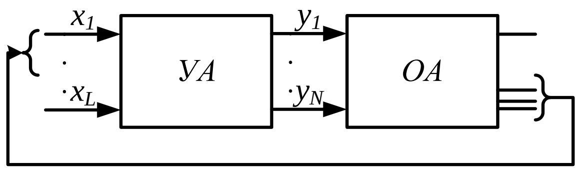
$$D_1 = \bar{Q}_1\bar{Q}_2 Q_3\bar{x} \vee \bar{Q}_1\bar{Q}_2\bar{Q}_3 x = 3 \vee 1;$$

$$D_2 = Q_1\bar{Q}_2\bar{Q}_3\bar{x} \vee Q_1 Q_2\bar{Q}_3 x \vee \bar{Q}_1\bar{Q}_2\bar{Q}_3 x = 8 \vee 13 \vee 1;$$

$$D_3 = \bar{Q}_1 Q_2\bar{Q}_3\bar{x} \vee \bar{Q}_1\bar{Q}_2 Q_3 x = 4 \vee 3.$$

3. **Операторные схемы алгоритмов. Микропрограмма**

Элементарный неделимый акт обработки информации в операционном автомате (ОА) называется микрооперацией (МО).



УА - управляющий автомат,

ОА - операционный автомат.

Множество МО, которые могут выполняться в ОА, обозначим через $Y=\{y_1,\dots,y_n\}$. Эти операции выполняются в ОА под действием сигналов, которые приходят из УА. Для того, чтобы в ОА выполнялась МО y_n, по каналу, соответствующему y_n должен прийти сигнал равный 1, т.е. $y_n = 1$.

Множество МО, которые могут выполняться в ОА одновременно, будем называть микрокомандой (МК). Будем считать, что у нас этих микрокоманд $Y_1,\dots,Y_T$. Например, пусть $Y_i = \{y_1, y_3, y_8\}$, т.е. $y_1 = y_3 = y_8 = 1$.

Для изменения порядка выполнения микрокоманд используются логические условия, причем множество логических условий $X = \{x_1,\dots, x_L\}$.

Каждая микрокоманда Y_i может содержать множество булевых функций $\alpha_{i1}, \dots, \alpha_{iT}$ от переменных $x_1,\dots, x_L$. При этом, если после выполнения команды $Y_j \alpha_{ij} = 1$, то следующей командой будет выполняться команда Y_j. Выражение α_{ij} называется функцией перехода от микрокоманды Y_i к микрокоманде Y_j.

Свойства функции перехода.

1. Ортогональность: $\alpha_{ij} \times \alpha_{it} = 0,\ j \neq t.$

   Две команды не могут выполняться одновременно.

2. Полнота: $\bigcap_{j=1}^{T} \alpha_{ij} = 1.$

   Хотя бы одна микрокоманда должна выполняться обязательно.

Совокупность микрокоманд и функций перехода образует микропрограмму.

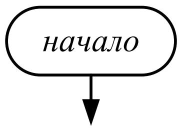
Таким образом, для описания микропрограммы необходимо задать последовательность микрокоманд и функций перехода, определяющих порядок выполнения МК.

Для описания микропрограмм будем использовать язык граф-схем алгоритмов.

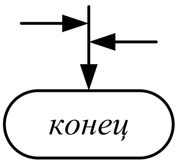
3.1. Граф-схемы алгоритмов

Граф-схема алгоритмов (ГСА) – это ориентированный связанный граф, содержащий вершины 4-х типов:

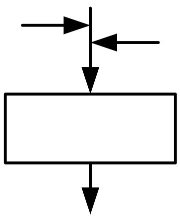
1. Одну начальную вершину с одним выходом и без входа



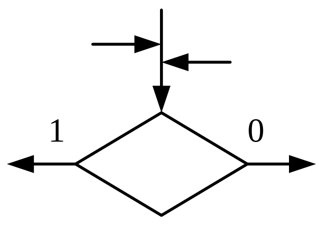
2. Одну конечную вершину



3. Конечное множество операторных вершин



4. Конечное множество условных вершин

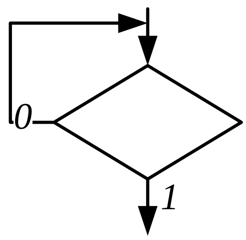


Конечная, операторная и условная вершины имеют по одному входу, начальная вершина входов не имеет. Начальная и операторные вершины имеют по одному выходу, а условные вершины по два выхода, помеченные как 0 и 1. Конечная вершина выходов не имеет.

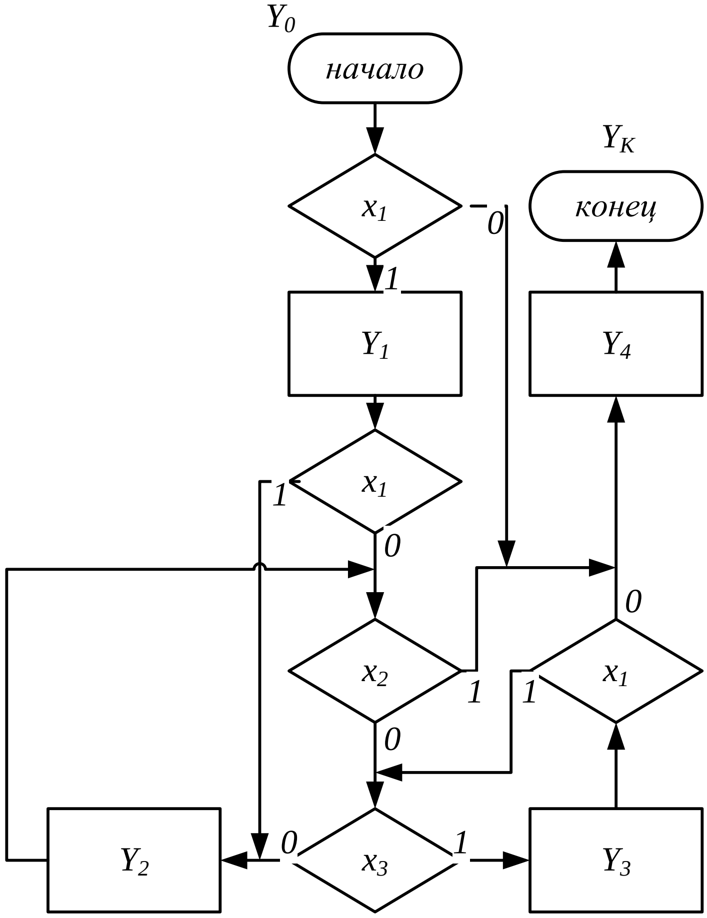
Правила составления ГСА.

1. Входы и выходы вершин соединяются друг с другом с помощью дуг, направленных всегда от выходов ко входу.
2. Каждый выход соединяется точно с одним входом.
3. Каждый вход соединяется, по крайней мере, с одним выходом.
4. Через каждую вершину проходит как минимум один путь из начальной вершины в конечную.

5. В каждой операторной вершине записывается микрокоманда Y_t, т.е. подмножество множества микроопераций: $Y = \{y_1, ..., y_n\}$. Допускается $Y_t=0$.
6. В каждой условной вершине записывается логическое условие - один из элементов множества $X = \{x_1, ..., x_L\}$.
7. Разрешается запись одинаковых микроопераций и одинаковых логических условий в различных операторных и условных вершинах соответственно.
8. Один из выходов условной вершины может быть соединен с ее входом. Такая вершина носит название ждущей вершины.



Пример ГСА



3.2. Выполнение ГСА на определенной последовательности наборов

Иногда возникает задача определения эквивалентности ГСА, описывающих одну и туже микропрограмму. Пусть задана некоторая последовательность переменных x_1, x_2, x_3 (наборы могут и совпадать) из ГСА, приведенной выше.

	x_1	x_2	x_3
w	*1*	*1*	*1*
w	*0*	*0*	*0*
w	*0*	*0*	*1*
	1	*1*	*1*
	1	*1*	*0*
	0	*1*	*0*
	1	*1*	*1*

Необходимо определить значение ГСА на заданной последовательности наборов. Это значение определяется с помощью «блуждания» по ГСА.

Определим этот процесс следующим образом.

Выбирается начальный оператор Y_0 и первый набор w. С учетом этого набора производится "блуждание по ГСА". Получаем значения Y_0 Y_1. Затем набор меняется на следующий и т.д. Y_0 Y_1 Y_2 Y_3 Y_2 $Y_4 Y_k$.

То, что мы получили, называется значением ГСА на заданной последовательности наборов.

Процедура "блуждания" заканчивается, когда:

1. исчерпаны все наборы;
2. дошли до оператора $Y_к$.

Если, дойдя до $Y_к$, мы не исчерпали наборов, то надо начинать с оператора Y_0 сначала.

ГСА$_1$ и ГСА$_2$ эквивалентны или равносильны, если их значения совпадают на всевозможных последовательностях наборов: ГСА$_1 \equiv$ ГСА$_2$.

3.3. Система формул перехода

Когда задают путь в ГСА, то перечисляют или последовательность вершин, или последовательность дуг. Напомним, что α_{ij} - функция перехода от оператора Y_i к Y_j. Тогда зададим этот переход с помощью последовательности вершин

$$Y_i x_{i1}^{e_{i1}} \ldots x_{iR}^{e_{iR}} Y_j, \text{ где } e_{ir} \in \{0,1\}. \qquad (*)$$

Условимся $e_{ir}^0 = \overline{e}_{ir}; e_{ir}^1 = e_{ir}$. На примере, переход из Y_1 в Y_4 будет выглядеть следующим образом: $Y_1\overline{x}_1 x_2 Y_4$. Каждому пути вида (*) поставим в соответствие конъюнкцию $\alpha_{ij} = \bigcap\limits_{r=1}^{R} x_{ir}^{e_{ir}}$.

Например, для рассматриваемой ГСА $\alpha_{14} = \overline{x_1} x_2$.

Если путей перехода несколько, то берется дизъюнкция соответствующих конъюнкций, например, $\alpha_{12} = \overline{x}_1\overline{x}_2\overline{x}_3 \vee x_1$.

Тот факт, что из оператора Y_1 имеются пути переходов в Y_2, Y_3 и Y_4, записывается следующим образом:

$$Y_1 \to x_1 Y_2 \vee \overline{x}_1 x_2 Y_4 \vee \overline{x}_1\overline{x}_2 x_3 Y_3 \vee \overline{x}_1\overline{x}_2\overline{x}_3 Y_2 .$$

Выражение вида $Y_i \to \bigcup\limits_{t=1}^{T+1} \alpha_{it} Y_t$ носит название формулы перехода от оператора Y_i или из оператора Y_i.

Здесь α_{ij} функция перехода от оператора Y_i к Y_j – булева функция двоичных переменных $x_1, \ldots, x_L$; $\alpha_{ij} Y_j$ называется отмеченной функцией. Она может принимать следующие значения $\alpha_{ij} Y_j = \begin{Bmatrix} Y_j, \textit{если}\ \alpha_{ij} = 1 \\ 0, \textit{если}\ \alpha_{ij} = 0 \end{Bmatrix}$.

Над булевыми функциями, входящими в отмеченные функции можно производить все преобразования булевой алгебры, а также пользоваться следующими соотношениями:

1. $\alpha\beta Y_j \vee \alpha\, \gamma\, Y_t = \alpha(\beta Y_j \vee \gamma Y_t)$.
2. $\alpha Y_j \vee \beta Y_j = (\alpha \vee \beta) Y_j$.
3. Если $\alpha \equiv \beta$, то $\alpha Y_j = \beta Y_j$.

Здесь α, β и γ- булевы функции переменных $x_1, \ldots, x_L$. Множество формул перехода для всех i=1,…,T образуют систему формул перехода (СФП). Для ГСА, которая приведена выше, эта система имеет вид:

$Y_0 \to x_1 Y_1 \vee \overline{x}_1 Y_4$.

$Y_1 \to x_1 Y_2 \vee \overline{x}_1\overline{x}_2\overline{x}_3 Y_2 \vee \overline{x}_1\overline{x}_2 x_3 Y_3 \vee \overline{x}_1 x_2 Y_4$.

$Y_2 \to \overline{x}_2\overline{x}_3 Y_2 \vee \overline{x}_2 x_3 Y_3 \vee x_2 Y_4$.

$Y_3 \to x_1 x_3 Y_3 \vee x_1\overline{x}_3 Y_2 \vee \overline{x}_1 Y_4$.

$Y_4 \to Y_\kappa$.

Представление формулы перехода в виде $Y_i \to x_l A_l \vee \overline{x}_l B$, $\textit{где}\ x_l \in \{x_1, \ldots, x_L\}$, а A и B – подформулы перехода не зависящие от x_l, носит название разложения формулы перехода по переменной x_l. Приведем пример разложения формулы перехода для оператора Y_1 по переменной x_1:

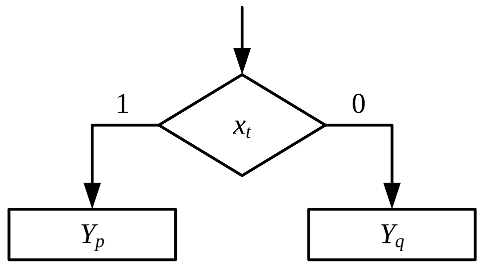
$$Y_1 \to x_1Y_2 \vee \bar{x}_1\bar{x}_2\bar{x}_3Y_2 \vee \bar{x}_1\bar{x}_2x_3Y_3 \vee \bar{x}_1x_2Y_4 =$$
$$= x_1Y_2 \vee \bar{x}_1(\bar{x}_2\bar{x}_3Y_2 \vee \bar{x}_2x_3Y_3 \vee x_2Y_4) =$$
$$= x_1Y_2 \vee \bar{x}_1(\bar{x}_2(\bar{x}_3Y_2 \vee x_3Y_3) \vee x_2Y_4),$$

т.е. произвели разложение по [x_1,x_2,x_3].

Разложение нужно производить до тех пор, пока внутри самых внутренних скобок не окажется выражение вида:

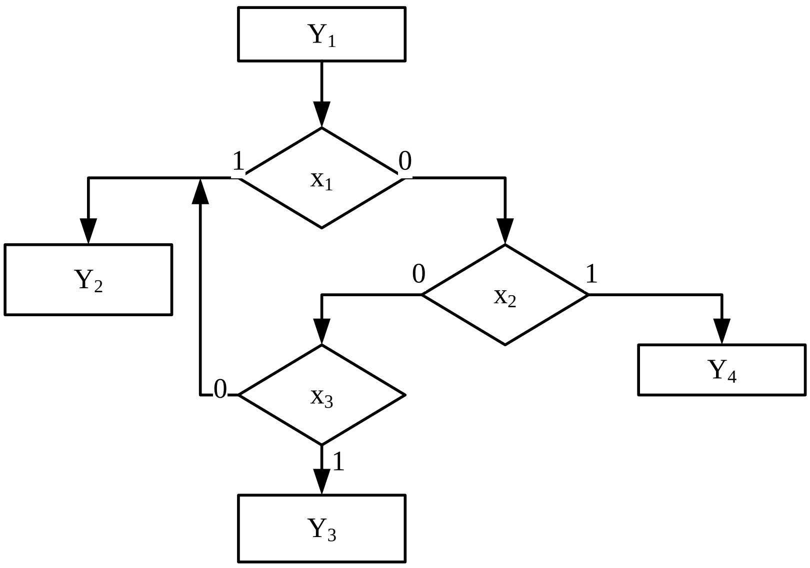
$$(x_tY_p \vee \bar{x}_tY_q)\,, \qquad (**)$$

которому однозначно соответствует следующий фрагмент ГСА



Полученная в результате формула перехода называется скобочной формулой перехода, а система таких формул - системой скобочных формулперехода (ССкФП).

По скобочному виду для оператора Y_1 строим фрагмент ГСА:



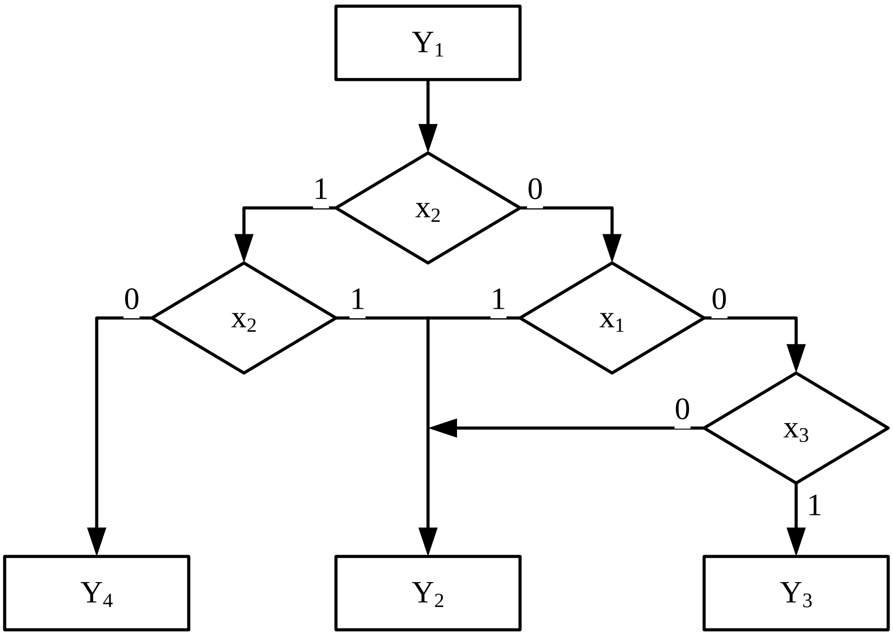
Мы получили подграф для случая разложения оператора Y_1 в последовательности переменных x_1, x_2, x_3.

Если какие-то члены формулы перехода не зависят от x_l, то при разложении по x_lнеобходимо сначала умножить их на выражение $(x_l \vee \bar{x}_l)$. Таким образом, любая формула перехода может быть разложена по любой входящей в нее переменной. Если разложить ту же формулу перехода в другой последовательности переменных x, то получим новую скобочную форму для формулы перехода Y_1.

Сделаем разложение формулы перехода для оператора Y_1 в последовательности x_2, x_1, x_3, принимая во внимание, что $(x_2 \vee \bar{x}_2) = 1$

$$
\begin{aligned}
&Y_1 \to x_1 x_2 Y_2 \vee x_1 \bar{x}_2 Y_2 \vee \bar{x}_1 \bar{x}_2 \bar{x}_3 Y_2 \vee \bar{x}_1 \bar{x}_2 x_3 Y_3 \vee \bar{x}_1 x_2 Y_4 = \\
&= x_2 (x_1 Y_2 \vee \vee \bar{x}_1 Y_4) \vee \bar{x}_2 (x_1 Y_2 \vee \bar{x}_1 \bar{x}_3 Y_2 \vee \bar{x}_1 x_3 Y_3) = \quad . \\
&= x_2 (x_1 Y_2 \vee \vee \bar{x}_1 Y_4) \vee \bar{x}_2 (x_1 Y_2 \vee \bar{x}_1 (\bar{x}_3 Y_2 \vee x_3 Y_3))
\end{aligned}
$$

По полученному скобочному виду для оператора Y_1 строим фрагмент ГСА:



Нетрудно видеть, что фрагменты ГСА, соответствующие различным разложениям формулы перехода Y_1 приводят к различным подграфам. Более того, оказывается, что минимальность ГСА зависит от «удачного» разложения всей системы формул перехода, поскольку желательно при этом разложении максимизировать число подобных членов.

3.4. Матричные схемы алгоритмов

Пусть ГСА Г имеет начальную вершину Y_0, конечную - $Y_к$ и Т операторных вершин с записанными в них различными операторами Y_1, …, $Y_Т$.

Матричная схема алгоритмов (МСА) М, соответствующая ГСА Г, есть квадратная матрица, строки которой отмечены символами Y_0, Y_1, …, $Y_Т$, а столбцы – символами Y_1, …, $Y_Т$,$Y_к$.

В этой матрице на пересечении строки Y_i и столбца Y_j стоит функция перехода α_{ij} из оператора Y_i к оператору Y_j, т.е.

	Y_j	
Y_i	α_{ij}	

Пример МСА, соответствующей ГСА, приведенной выше.

	Y_1	Y_2	Y_3	Y_4	Y_k
Y_0	x_1			$\bar{x}_1$	
Y_1		x_1 $\bar{x}_1\bar{x}_2\bar{x}_3$	$\bar{x}_1\bar{x}_2x_3$	$\bar{x}_1x_2$	
Y_2		$\bar{x}_2\bar{x}_3$	$\bar{x}_2x_3$	x_2	
Y_3		$x_1\bar{x}_3$	x_2x_3	$\bar{x}_1$	
Y_4					1

3.5. Учет распределения сдвигов

Пусть задана матричная схема алгоритма (МСА)

	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_k
Y_0	$\overline{x_2}x_1$			$x_2\overline{x_1}$ $\overline{x_2x_1}$	$x_2\,x_1$		
Y_1			$x_2\,x_1$				$\overline{x_2}$ $x_2\overline{x_1}$
Y_2	$\overline{x_2}x_1$				x_2 $\overline{x_2x_1}$		
Y_3				$\overline{x_1}$	x_1		
Y_4						$x_2x_1x_3$	$\overline{x_2}$ $x_2\overline{x_1}$ $x_2\,x_1\overline{x_3}$
Y_5		x_1					$\overline{x_1}$
Y_6							1

Если каждому оператору Y_t поставлено в соответствие множество B_t ($Y_t \rightarrow B_t$) двоичных переменных, которые могут изменяться в процессе выполнения оператора Y_t, то говорят, что задано распределение сдвигов.

Если каждая их двоичных переменных (логических условий) может изменяться любым оператором Y_t, то такое РС называется универсальным. Если B_t=0 для любого оператора Y_t, то такое распределение сдвигов называется пустым.

Пусть для рассматриваемой МСА задано следующее распределение сдвигов:

$Y_0 \rightarrow B_0 = \{x_1, x_2, x_3\}$; $Y_4 \rightarrow B_4 = \{x_2\}$;
$Y_1 \rightarrow B_1 = \{x_2, x_3\}$; $Y_5 \rightarrow B_5 = \{x_1, x_2, x_3\}$;
$Y_2 \rightarrow B_2 = \{x_2\}$; $Y_6 \rightarrow B_6 = \{x_2\}$;
$Y_3 \rightarrow B_3 = \{x_1, x_2, x_3\}$; $Y_к \rightarrow B_7 = \{x_1, x_2, x_3\}$.

Из анализа столбца Y_1 видно, что в неравных нулю функциях перехода переменная x_1 всегда встречается без инверсии. Это значит, что Y_1 может выполняться только при $x_1 = 1$. Обращаясь к РС, видим, что $x_1 \notin B_1$, т.е. x_1 не изменяется во время выполнения Y_1. Поэтому после выполнения $Y_1 x_1$ остается равным 1. И тогда в строке Y_1 переменную x_1можно заменить на 1, а $\overline{x_1}$ на 0. Аналогичной замены x_2 на 0 сделать нельзя, т.к. $x_2 \in B_1$ и может изменяться в процессе выполнения Y_1.

Продолжаем аналогичным образом анализ всех столбцов. В результате оказывается, что все функции перехода α_{i6}= 0, то есть оператор Y_6 никогда не выполняется. Следовательно, строку и столбец Y_6 из МСА можно удалить.

МСА после минимизации с учетом РС.

	Y_1	Y_2	Y_3	Y_4	Y_5	Y_k
Y_0	$\overline{x_2} x_1$			$\frac{x_2 \overline{x_1}}{x_2 x_1}$	$x_2 x_1$	
Y_1			x_2			$\overline{x_2}$
Y_2	$\overline{x_2}$				x_2	
Y_3				$\overline{x_1}$	x_1	
Y_4						1
Y_5		x_1				$\overline{x_1}$

3.6. Объединение граф-схем алгоритмов

При описании алгоритма работы сложной системы часто представляется целесообразным построение нескольких ГСА, каждая из которых описывает часть общего поведения системы (частные ГСА). Например, при проектировании центрального устройства ЭВМ проще построить граф-схемы выполнения отдельных операций. Очевидно, что в этих ГСА некоторые операторные и условные вершины будут одинаковыми, и если для каждой ГСА синтезировать отдельный автомат управления, то результат синтеза может быть далеко не оптимальным.

В связи с этим возникает задача объединения частных ГСА в единую граф-схему, решение которой позволит минимизировать суммарное число операторных и условных вершин.

Пусть имеются ГСА $\Gamma_1, \ldots, \Gamma_q, \ldots, \Gamma_Q$, в каждой из которых операторы не повторяются, но среди различных граф-схем могут встречаться одинаковые операторы. Требуется построить объединенную ГСА, которая равносильна каждой из ГСА Γ_q при тех условиях, когда должна выполняться эта ГСА.

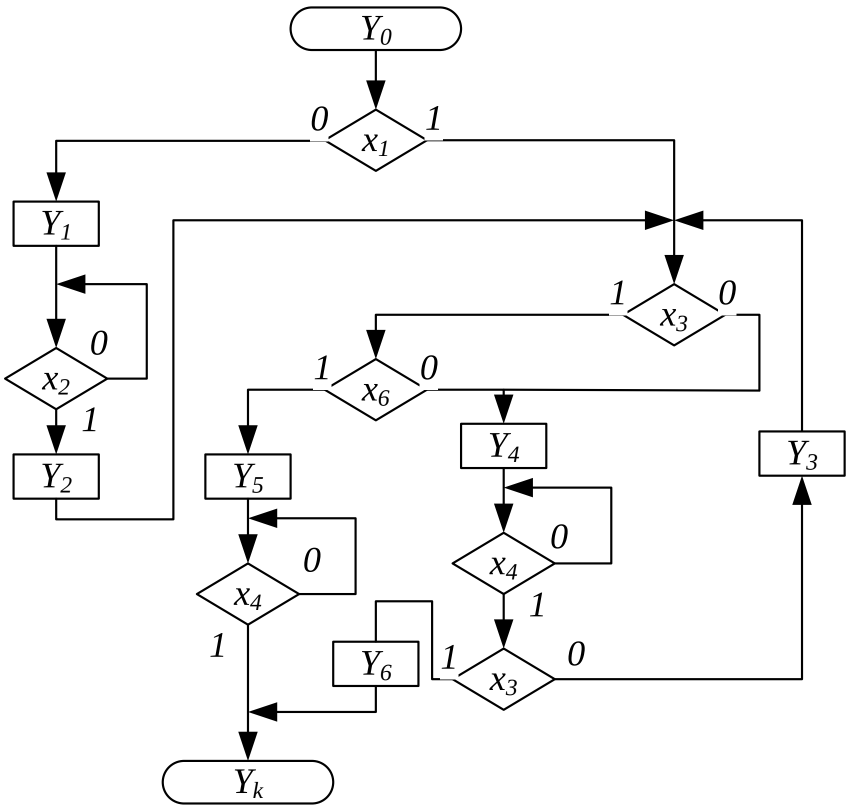
Обычно решение задачи объединения разбивается на ряд этапов:

1. Для каждой ГСА Γ_q строится соответствующая ей МСА M_q $(q = 1,\ldots,Q)$.
2. Каждая МСА M_q кодируется вектором $((e_{q_1},\ldots, e_{q_n},\ldots, e_{q_N})$, где $e_{q_n} \in \{0,1\}$, $N =]\log_2 Q[$;] a [- как и ранее, означает наименьшее целое число, больше чем a или равное ему, если a - целое.
3. Каждой МСА M_q ставится в соответствие конъюнкция $P_1^{e_{q_1}} \ldots P_n^{e_{q_n}} \ldots P_N^{e_{q_N}}$, где $(e_{q_1},\ldots, e_{q_n},\ldots, e_{q_N})$ код МСА M_q ($P_n^0 = \overline{P_n}$, $P_n^1 = P_n$).
4. Строится объединенная МСА M, строки и столбцы которой отмечены всеми операторами, входящими в объединение множеств операторов МСА $M_1,\ldots,M_Q$. Элементы α_{ij} МСА M равны $\bigcup_{q=1}^{Q} (\alpha_{ij})_q P_q$, где $(\alpha_{ij})_q$ - элемент МСА M_q, стоящий на пересечении строки Y_i и столбца Y_j.
5. Поскольку P_q ($q = 1, \ldots, Q$) равна единице все время пока МСА M "работает" как МСА M_q, ни один оператор не меняет значения переменных $P_1,\ldots,P_N$ (относительно этих переменных имеем пустое распределение сдвигов), в связи с чем МСА M в некоторых случаях можно упростить. В результате получается сминимизированная с учетом распределения сдвигов МСА M^*.

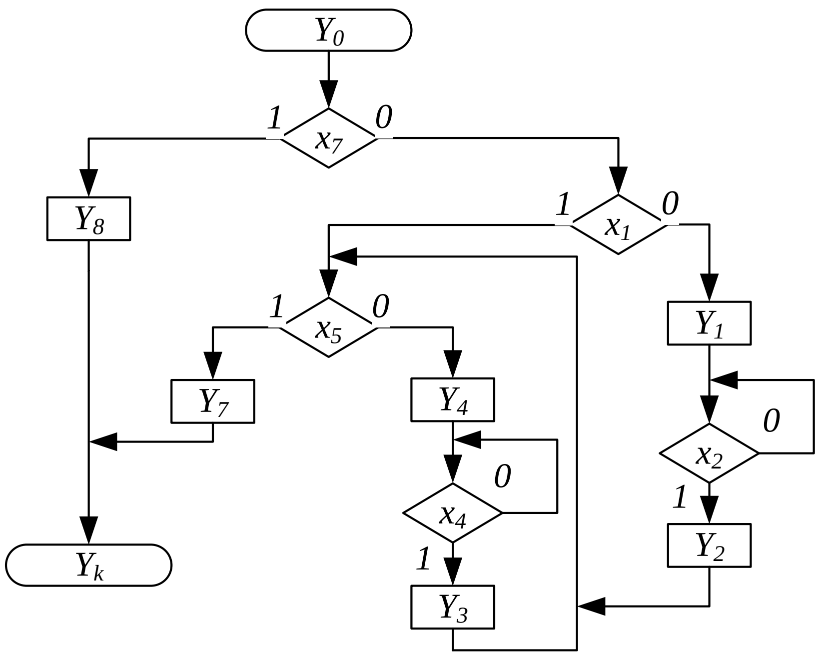
Рассмотрим объединение граф-схем алгоритмов на ***примере.***

Пусть заданы три ГСА Γ_1, ГСА Γ_2 и ГСА Γ_3, которые необходимо объединить.

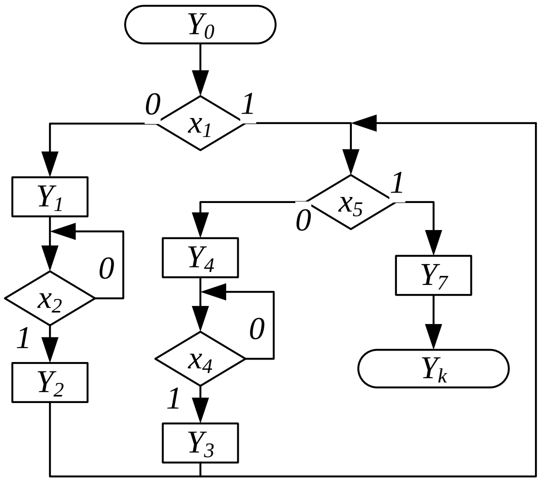
ГСА $Г_1$


Y_0
x_1
Y_1
x_2
Y_2
x_3
x_6
Y_5
x_4
Y_4
x_4
Y_3
Y_6
x_3
Y_k

ГСА $Г_2$


Y_0
x_7
Y_8
x_1
x_5
Y_7
Y_4
x_4
Y_3
Y_1
x_2
Y_2
Y_k

ГСА Г$_3$



Для каждой из них построим МСА, т.е. МСА M_1, M_2, M_3.

M_1	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	$Y_к$
Y_0	$\overline{x_1}$			$x_1\overline{x_3}$ $x_1x_3\overline{x_6}$	$x_1x_3x_6$		
Y_1		x_2					
Y_2				$\overline{x_3}$ $x_3\overline{x_6}$	x_3x_6		
Y_3				$\overline{x_3}$ $x_3\overline{x_6}$	x_3x_6		
Y_4			$x_4\overline{x_3}$			x_4x_3	
Y_5							x_4
Y_6							1

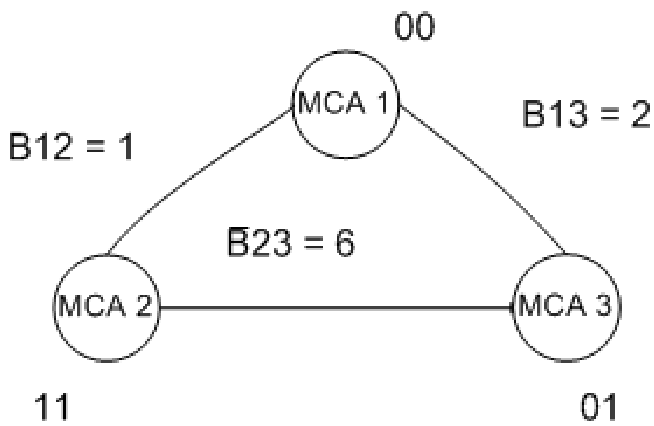
M_2	Y_1	Y_2	Y_3	Y_4	Y_7	Y_8	$Y_к$
Y_0	$\overline{x_7x_1}$			$\overline{x_7}x_1\overline{x_5}$	$\overline{x_7}x_1x_5$	x_7	
Y_1		x_2					
Y_2				$\overline{x_5}$	x_5		
Y_3				$\overline{x_5}$	x_5		
Y_4			x_4				
Y_7							1
Y_8							1

M_3	Y_1	Y_2	Y_3	Y_4	Y_7	$Y_к$
Y_0	$\overline{x_1}$			$x_1\overline{x_5}$	x_1x_5	
Y_1		x_2				
Y_2				$\overline{x_5}$	x_5	
Y_3				$\overline{x_5}$	x_5	
Y_4			x_4			
Y_7						1

При кодировании частных МСА M_q определим коэффициент схожести между матрицами M_λ и M_q, $(\lambda, q) \in \{1, 2, \dots, Q\}$, как $B_{\lambda q} = \Sigma\, \alpha_{ij}$.

Здесь α_{ij} - формула перехода от оператора Y_i к Y_j в матрицах M_λ и M_q. Для примера

$B_{12} = 1$;
$B_{13} = 1 + 1 = 2$;
$B_{23} = 1 + 1 + 1 + 1 + 1 + 1 = 6$.



Соседними кодами закодируем те МСА, которые имеют наибольшую связность, т.е. $МСА_1$– 00, $МСА_2$– 11, $МСА_3$ - 01.

Каждой МСА - M_1, M_2, M_3 поставим в соответствие конъюнкцию, т.е.

$$МСА_1 \rightarrow \overline{P_1}\,\overline{P_2};\ МСА_2 \rightarrow P_1P_2;\ МСА_3 \rightarrow \overline{P_1}P_2.$$

Построим объединенную МСА M, строки и столбцы которой отмечены всеми операторами, входящими в объединение множеств операторов МСА M_1, M_2 и M_3.

M	***Y₁***	***Y₂***	***Y₃***	***Y₄***	***Y₅***	***Y₆***	***Y₇***	***Y₈***	***Yк***
Y_0	$\overline{P_1 P_2}\,\overline{x_1}$ $P_1 P_2 \overline{x_7 x_1}$ $\overline{P_1} P_2 \overline{x_1}$			$\overline{P_1 P_2} x_1 \overline{x_3}$ $\overline{P_1 P_2} x_1 x_3 \overline{x_6}$ $P_1 P_2 \overline{x_7} x_1 \overline{x_5}$ $\overline{P_1} P_2 x_1 x_5$	$\overline{P_1 P_2} x_1 x_3 x_6$		$P_1 P_2 \overline{x_7} x_1 x_5$ $\overline{P_1} P_2 x_1 x_5$	$P_1 P_2 x_7$	
Y_1		$\overline{P_1 P_2} x_2$ $P_1 P_2 x_2$ $\overline{P_1} P_2 x_2$							
Y_2				$\overline{P_1 P_2}\,\overline{x_3}$ $\overline{P_1 P_2}\, x_3 \overline{x_6}$ $P_1 P_2 \overline{x_5}$ $\overline{P_1} P_2 \overline{x_5}$	$\overline{P_1 P_2} x_3 x_6$		$P_1 P_2 x_5$ $\overline{P_1} P_2 x_5$		
Y_3				$\overline{P_1 P_2} x_3$ $\overline{P_1 P_2}\, x_3 \overline{x_6}$ $P_1 P_2 \overline{x_5}$ $\overline{P_1} P_2 \overline{x_5}$	$\overline{P_1 P_2} x_3 x_6$		$P_1 P_2 x_5$ $\overline{P_1} P_2 x_5$		
Y_4			$\overline{P_1 P_2} x_4 \overline{x_3}$ $P_1 P_2 x_4$ $\overline{P_1} P_2 x_4$			$\overline{P_1 P_2} x_4 x_3$			
Y_5									$\overline{P_1 P_2} x_4$
Y_6									$\overline{P_1 P_2}$
Y_7									$P_1 P_2$ $\overline{P_1} P_2$
Y_8									$P_1 P_2$

Осуществим минимизацию объединенной МСА *M* с учетом распределения сдвигов. После этого запишем систему формул перехода и приведем их к скобочному виду.

$$Y_0 = \overline{P_1}\,\overline{P_2}\,\overline{x_1} Y_1 \vee P_1 P_2 \overline{x_7}\,\overline{x_1} Y_1 \vee \overline{P_1} P_2 \overline{x_1} Y_1 \vee \overline{P_1}\,\overline{P_2} x_1 \overline{x_3} Y_4 \vee \overline{P_1}\,\overline{P_2} x_1 x_3 \overline{x_6} Y_4 \vee$$

$$\vee P_1 P_2 \overline{x_7} x_1 \overline{x_5} Y_4 \vee \overline{P_1} P_2 x_1 \overline{x_5} Y_4 \vee \overline{P_1}\,\overline{P_2} x_1 x_3 x_6 Y_5 \vee P_1 P_2 \overline{x_7} x_1 x_5 Y_7 \vee \overline{P_1} P_2 x_1 x_5 Y_7 \vee$$

$$\vee P_1 P_2 x_7 Y_8 = \overline{P_1} (\overline{P_2}\,\overline{x_1} Y_1 \vee P_2 \overline{x_1} Y_1 \vee \overline{P_2} x_1 \overline{x_3} Y_4 \vee \overline{P_2} x_1 x_3 \overline{x_6} Y_4 \vee P_2 x_1 \overline{x_5} Y_4 \vee \overline{P_2} x_1 x_3 x_6 Y_5 \vee$$

$$\vee P_2 x_1 x_5 Y_7) \vee P_1 P_2 (\overline{x_7}\,\overline{x_1} Y_1 \vee \overline{x_7} x_1 \overline{x_5} Y_4 \vee \overline{x_7} x_1 x_5 Y_7 \vee x_7 Y_8) =$$

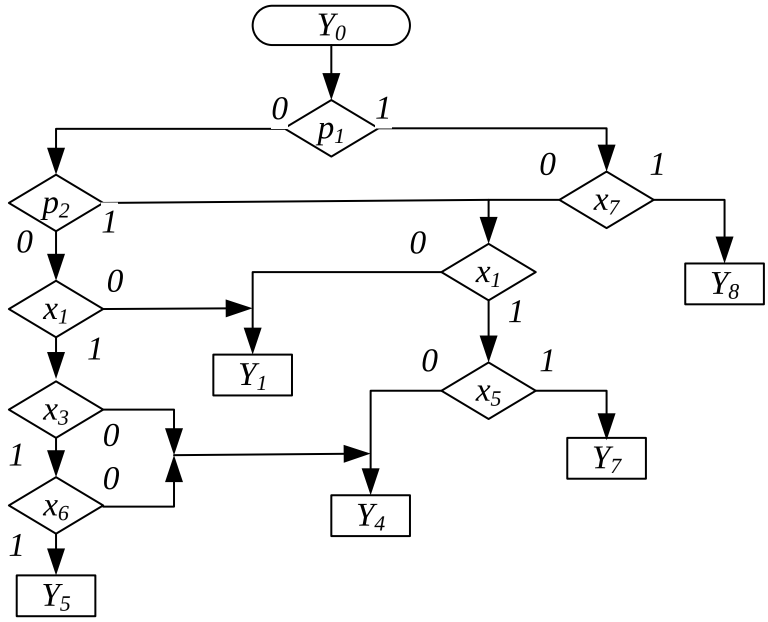
$$= \overline{P_1} (\overline{P_2} (\overline{x_1} Y_1 \vee x_1 \overline{x_3} Y_4 \vee x_1 x_3 \overline{x_6} Y_4 \vee x_1 x_3 x_6 Y_5) \vee P_2 (\overline{x_1} Y_1 \vee x_1 \overline{x_5} Y_4 \vee x_1 x_5 Y_7)) \vee$$

$$\vee P_1 \overline{P_1} (\overline{x_7} (\overline{x_1} Y_1 \vee x_1 \overline{x_5} Y_4 \vee x_1 x_5 Y_7) \vee x_7 Y_8) = \overline{P_1} (\overline{P_2} (\overline{x_1} Y_1 \vee x_1 (\overline{x_3} Y_4 \vee$$

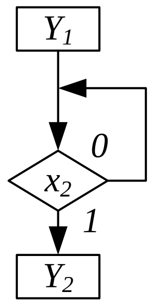
$$\vee x_3 \overline{x_6} Y_4 \vee x_3 x_6 Y_5) \vee P_2 (\overline{x_1} Y_1 \vee x_1 (\overline{x_5} Y_4 \vee x_5 Y_7)) \vee P_1 (\overline{x_7} (\overline{x_1} Y_1 \vee x_1 (\overline{x_5} Y_4 \vee x_5 Y_7)) \vee$$

$$\vee x_7 Y_8) = \overline{P_1} (\overline{P_2} (\overline{x_1} Y_1 \vee x_1 (\overline{x_3} Y_4 \vee x_3 (\overline{x_6} Y_4 \vee x_6 Y_5))) \vee P_2 (\overline{x_1} Y_1 \vee$$

$$\vee x_1 (\overline{x_5} Y_4 \vee x_5 Y_7))) \vee P_1 (\overline{x_7} (\overline{x_1} Y_1 \vee x_1 (\overline{x_5} Y_4 \vee x_5 Y_7)) \vee x_7 Y_8)$$



$$Y_1 = \overline{P_1}\,\overline{P_2}\,x_2 Y_2 \vee P_1 P_2 x_2 Y_2 \vee \overline{P_1} P_2 x_2 Y_2 \vee (P_1 \overline{P_2}\,x_2 Y_2) = x_2 Y_2$$

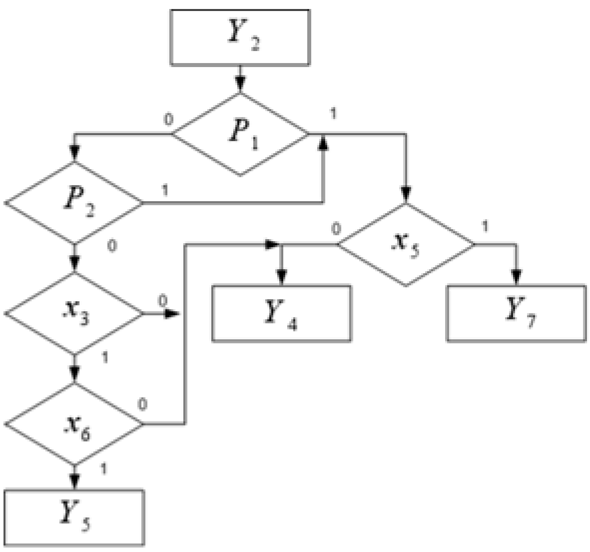


$$Y_2 = \overline{P_1}\,\overline{P_2}\,\overline{x_3} Y_4 \vee \overline{P_1}\,\overline{P_2}\,x_3 \overline{x_6} Y_4 \vee P_1 P_2 \overline{x_5} Y_4 \vee \overline{P_1} P_2 \overline{x_5} Y_4 \vee \overline{P_1}\,\overline{P_2}\,x_3 x_6 Y_5 \vee P_1 P_2 x_5 Y_7 \vee \overline{P_1} P_2 x_5 Y_7 =$$

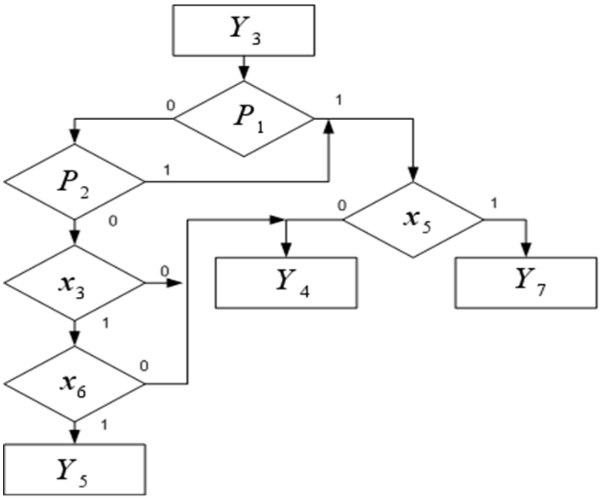
$$= \overline{P_1}(\overline{P_2}\,\overline{x_3} Y_4 \vee \overline{P_2}\,x_3 \overline{x_6} Y_4 \vee P_2 \overline{x_5} Y_4 \vee P_2 x_5 Y_7 \vee \overline{P_2}\,x_3 x_6 Y_5) \vee$$

$$\vee P_1 P_2 (\overline{x_5} Y_4 \vee x_5 Y_7) = \overline{P_1}(\overline{P_2}(\overline{x_3} Y_4 \vee x_3 \overline{x_6} Y_4 \vee x_3 x_6 Y_5) \vee P_2(\overline{x_5} Y_4 \vee x_5 Y_7)) \vee P_1(\overline{x_5} Y_4 \vee x_5 Y_7) =$$

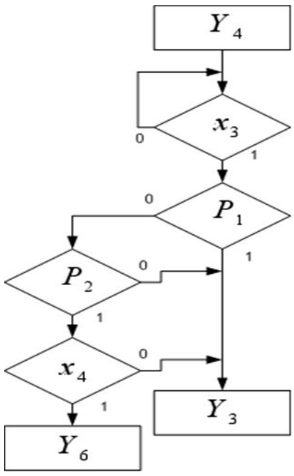
$$= \overline{P_1}(\overline{P_2}(\overline{x_3} Y_4 \vee x_3(\overline{x_6} Y_4 \vee x_6 Y_5) \vee P_2(\overline{x_5} Y_4 \vee x_5 Y_7)) \vee P_1(\overline{x_5} Y_4 \vee x_5 Y_7))$$



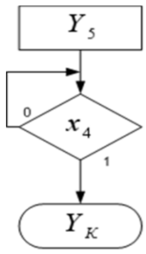
$$Y_3 = \overline{P_1}\,\overline{P_2}\,\overline{x_3}Y_4 \vee \overline{P_1}\,\overline{P_2}x_3\overline{x_6}Y_4 \vee \overline{P_1}\,\overline{P_2}x_3x_6Y_5 \vee P_1P_2\overline{x_5}Y_4 \vee \overline{P_1}P_2\overline{x_5}Y_4 \vee P_1P_2x_5Y_7 \vee \overline{P_1}P_2x_5Y_7 = \overline{P_1}(\overline{P_2}\,\overline{x_3}Y_4 \vee \overline{P_2}x_3\overline{x_6}Y_4 \vee \overline{P_2}x_3x_6Y_5 \vee P_2x_5Y_4 \vee P_2x_5Y_7) \vee$$
$$\vee P_1P_2(\overline{x_5}Y_4 \vee x_5Y_7) = \overline{P_1}(\overline{P_2}(\overline{x_3}Y_4 \vee x_3(\overline{x_6}Y_4 \vee x_6Y_5)) \vee P_2(\overline{x_5}Y_4 \vee x_5Y_7)) \vee P_1(\overline{x_5}Y_4 \vee x_5Y_7)$$



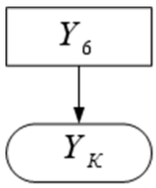
$$Y_4 = \overline{P_1}\,\overline{P_2}x_4\overline{x_3}Y_3 \vee P_1P_2x_4Y_3 \vee \overline{P_1}P_2x_4Y_3 \vee \overline{P_1}\,\overline{P_2}x_4x_3Y_6 = x_4(\overline{P_1}(\overline{P_2}(\overline{x_3}Y_3 \vee x_3Y_6) \vee P_2Y_3) \vee P_1P_2Y_3) = x_4(\overline{P_1}(\overline{P_2}(\overline{x_3}Y_3 \vee x_3Y_6) \vee P_2Y_3) \vee P_1Y_3)$$



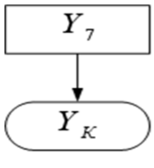
$Y_5 = x_4Y_K$



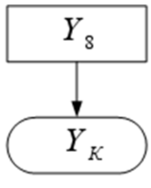
$Y_6 = Y_K$



$Y_7 = P_1Y_K \vee \overline{P_1}Y_K = Y_K$



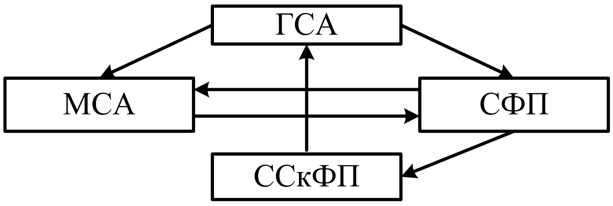
$Y_8 = Y_K$



3.7. Взаимосвязь алгоритмов

Таким образом, микропрограмма может быть представлена в виде ГСА, СФП, ССкФП или МСА.

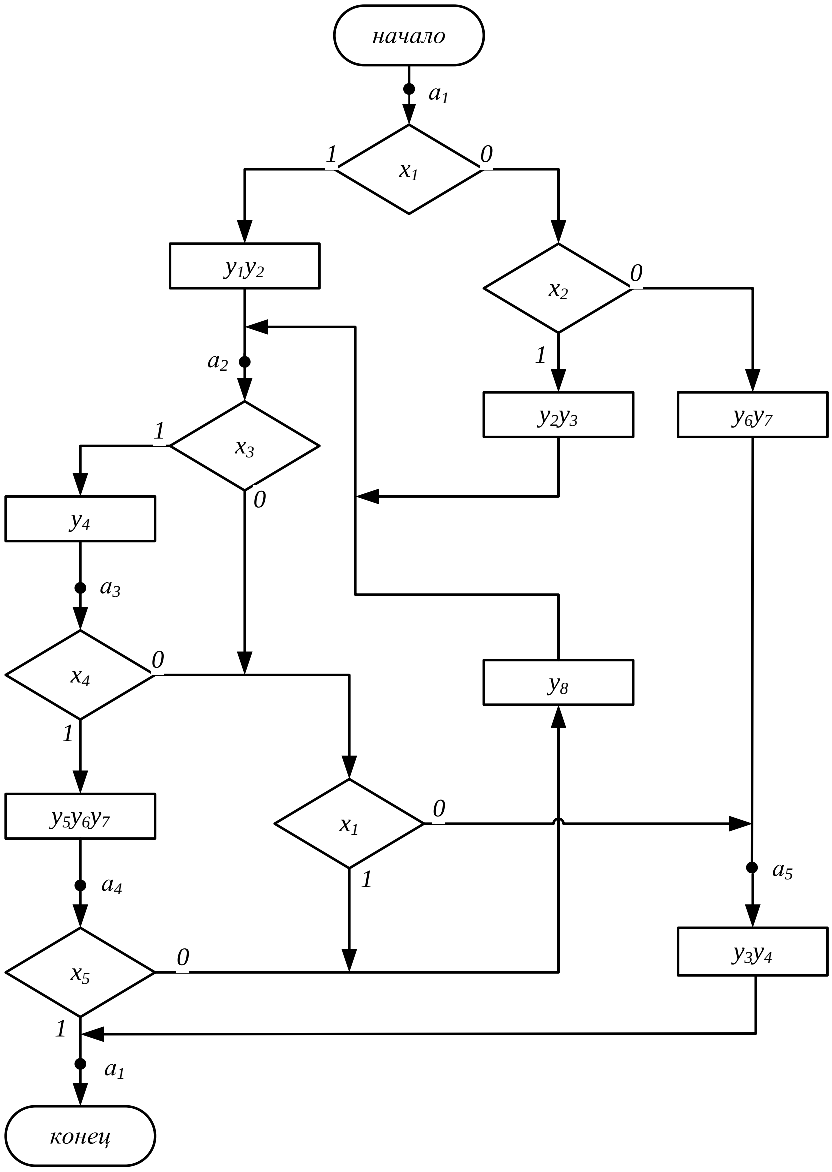
Их взаимосвязи и возможные преобразования, отражены на рисунке ниже.



4. Синтез микропрограммного автомата по граф-схеме алгоритма (на примере автомата Мили)

Конечный автомат, реализующий микропрограмму работы дискретного устройства, называется микропрограммным автоматом (МПА).

Изобразим следующую ГСА.



Синтез МПА по ГСА осуществляется в два этапа.

А. Получение отмеченной ГСА.

1. Вход вершины, следующей за начальной, отмечаем символом a_1.
2. Вход конечной вершины также отмечается символом a_1.
3. Входы вершин (условных и операторных), следующих за операторными, отмечаются символами a_2, a_3 и т.д.
4. Если вход вершины отмечается, то не более одного раза.

В. Составление путей перехода.

Рассмотрим пути следующих видов:

$$a_m x_{m1}^{e_{m1}} ... x_{mR}^{e_{mR}} Y_t a_s, \tag{1}$$

где $e_{mr} \in \{0,1\}; x_{mr}^0 = \overline{x}_{mr}, x_{mr}^1 = x_{mr}$.

Это путь из отметки a_m в отметку a_s, проходящий через конечное множество условных вершин и операторную вершину Y_t. В путях перехода вида (1) не исключен случай R=0, т.е. $a_m Y_t a_s$.

$$a_m x_{m1}^{e_{m1}} ... x_{mR}^{e_{mR}} a_1. \tag{2}$$

Это путь из отметки a_m в отметку a_1, проходящий только через конечное множество условных вершин.

Пути вида (1) и (2) называются путями переходов ГСА.

Для приведенной ГСА пути переходов будут выглядеть следующим образом.

$a_1 x_1 (y_1 y_2) a_2$	$a_3 x_4 (y_5 y_6 y_7) a_4$
$a_1 \overline{x_1}\, x_2 (y_1 y_3) a_2$	$a_3 \overline{x_4} x_1 (y_8) a_2$
$a_1 \overline{x_1} x_2 (y_1 y_3) a_5$	$a_3 \overline{x_4 x_1} (y_3 y_4) a_1$
$a_2 x_3 (y_4) a_3$	$a_4 x_5 a_1$
$a_2 \overline{x_3}\, x_1 (y_3) a_2$	$a_4 \overline{x_5} (y_8) a_2$
$a_2 \overline{x_3}\, x_1 (y_3 y_4) a_1$	$a_5 (y_3 y_4) a_1.$

4.1. Синтез графа микропрограммного автомата

Отметкам a_1, a_2,…. ГСА, полученным на этапе А, поставим в соответствие состояния МПА (на графе отметки соответствуют вершинам графа).

Каждому пути вида (1) поставим в соответствие переход МПА из состояния a_m в состояние a_s под воздействием входного набора

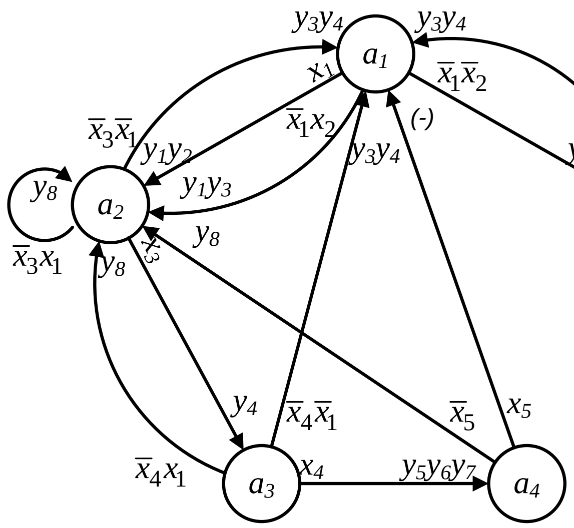
$$X(a_m, a_s) = \bigcap_{r=1}^{R} x_{mr}^{e_{mr}}$$

поступающего на вход автомата, с выдачей микрокоманды Y_t.

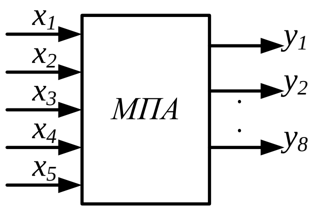
Каждому пути перехода вида (2) поставим в соответствие переход автомата из состояния a_m в состояние a_1 под действием входного набора

$$X(a_m, a_1) = \bigcap_{r=1}^{R} x_{mr}^{e_{mr}}$$

с выдачей пустой микрокоманды Y_0. Пустая микрокоманда - это микрокоманда, у которой число микроопераций равно нулю.



Для примера, входными сигналами автомата являются пятикомпонентные наборы, состоящие из 0 и 1. Выходными сигналами являются восьмикомпонентные наборы, состоящие из 0 и 1.



Переход вида $a_2\overline{x_3}\overline{x_1}(y_3y_4)a_1$ означает, что автомат должен перейти из состояния a_2 в состояние a_1 под воздействием всех наборов входных сигналов, которые покрываются кубом

0	*d*	*0*	*d*	*d*
x_1	x_2	x_3	x_4	x_5

с выдает на этом переходе выходной набор

0	0	1	1	0	0	0	0
y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8

Переход из a_5 в a_1 осуществляется под воздействием всех наборов входных сигналов, которые покрываются кубом *ddddd*, т.е. под воздействием любого входного набора. Пусть имеется следующая последовательность наборов

	x_1	x_2	x_3	x_4	x_5	
a_1	0	1	0	1	0	$y_1\, y_3$
a_2	1	0	1	0	1	y_4
a_3	1	1	0	0	1	y_8
a_2	0	1	0	1	0	$y_3\, y_4$
a_1	1	1	0	1	0	
	1	0	1	1	1	

Произведем «блуждание» по ГСА для данного набора. Получаем значение ГСА на этой последовательности наборов Y_0 ($y_1\, y_3$) (y_4) (y_8) ($y_3\, y_4$) $Y_к$.

Будем говорить, что МСА *S*реализует некоторую ГСА Г, если его реакции в состоянии a_1 на всевозможных последовательностях наборов совпадают со значениями ГСА Г на тех же последовательностях (без учета начального Y_0 и конечного $Y_к$ операторов). Из рассмотренного способа построения МПА ясно, что он реализует исходную ГСА.

При большом числе состояний и большом числе переходов графический способ представления автомата теряет свою наглядность. При этом оказывается удобнее переходить к таблицам.

4.2. Таблицы переходов микропрограммного автомата

Различают следующие таблицы переходов МПА:

1. Структурные и не структурные.
2. Прямые и обратные.
3. С узлами и без узлов.

Для пояснения сути вопроса воспользуемся предыдущим примером. В таблице:

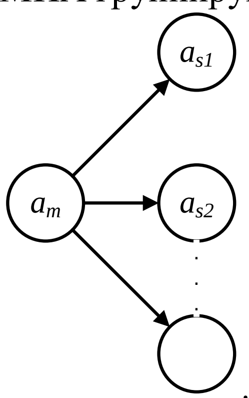
a_m	- исходное состояние МПА;
$K(a_m)$	- код исходного состояния;
a_s	- состояние перехода;
$K(a_s)$	- код состояния перехода;
$X(a_m, a_s)$	- входной набор, под действием какого осуществляется переход из из состояния a_m в состояние a_s;
$Y(a_m, a_s)$	- выходной набор, который должен выработаться на переходе МПА из состояния a_m в состояние a_s;
$F(a_m, a_s)$	- те сигналы функции возбуждения, которые необходимо выработать (с учетом типа триггера, на котором реализуется память автомата), чтобы перевести элементы памяти автомата в состояние, соответствующее $K(a_s)$;
h	- номер строки.

Таблица, содержащая столбцы 1, 3, 5 и 6, носит название таблицы переходов МПА.

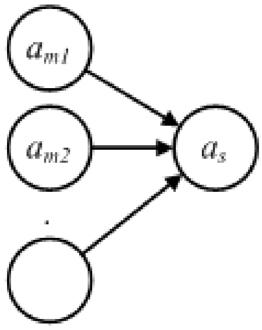
Таблица, содержащая все 7 столбцов, называется структурной таблицей переходов МПА.

a_m	$K(a_m)$	a_s	$K(a_s)$	x_h	Y_h	$F_h(D)$	$F_h(T)$	$F_h(JK)$	$F_h(RS)$	h
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
a_1	*001*	a_2 a_2 a_5	*010* *010* *101*	x_1 $\overline{x_1}x_2$ $\overline{x_1x_2}$	y_1y_2 y_1y_3 y_6y_7	D_2 D_2 D_1D_3	T_2T_3 T_2T_3 T_1	J_2K_3 $J_2K_3J_1$	S_2R_3 S_2R_3 S_1	*1* *2* *3*
a_2	*010*	a_3 a_2 a_1	*011* *010* *001*	x_3 $\overline{x_3}x_1$ $\overline{x_3x_1}$	y_4 y_8 y_3y_4	D_2D_3 D_2 D_3	T_3 - T_2T_3	J_3 - K_2J_3	S_3 - R_2S_3	*4* *5* *6*
a_3	*011*	a_4 a_2 a_1	*100* *010* *001*	x_4 $\overline{x_4}x_1$ $\overline{x_4x_1}$	$y_6y_5y_7$ y_8 y_3y_4	D_1 D_2 D_3	$T_2T_2T_3$ T_3 T_2	$J_1K_2K_3$ K_3 K_2	$S_1R_2R_3$ R_3 R_2	*7* *8* *9*
a_4	*100*	a_1 a_2	*000* *010*	x_5 $\overline{x_5}$	- y_8	D_3 D_2	T_1T_3 T_1T_2	K_1J_3 K_1J_2	R_1S_3 R_1S_2	*10* *11*
a_5	*101*	a_1	*001*	*1*	y_3y_4	D_3	T_1	K_1	R_1	*12*

В прямой таблице переходов МПА группируются переходы вида



В обратной таблице переходов МПА выбирается состояние a_s и группируются переходы вида



4.3. Модель структурного автомата при построении его в виде таблицы переходов

В этом случае автомат также будет описываться шестеркой, но в отличие от абстрактного автомата, элементами множеств будут не абстрактные сигналы, а переменные:

$$S = (A, X, Y, \delta, \lambda, a_1)$$

$A = \{a_1,...,a_m\}$ - множество состояний,

$X = \{a_1,...,a_L\}$ - множество входных переменных,

$Y = \{y_1,...,y_N\}$ - множество выходных переменных,

δ - функция переходов,

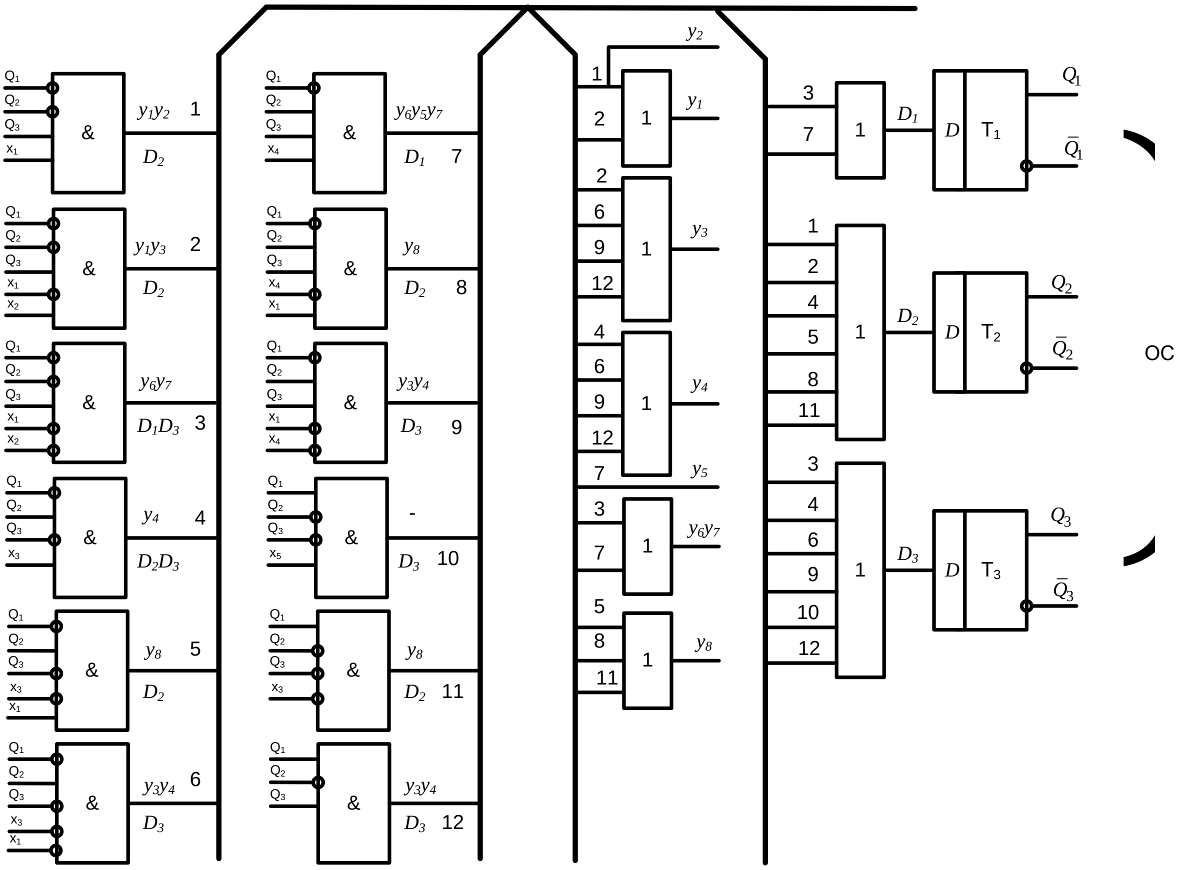
λ - функция выходов,

a_1 - начальное состояние.

Для описания функций δ и λ мы используем таблицу переходов МПА (столбцы 1, 3, 5, 6). Каждая строка этой таблицы имеет вид: $| a_m| a_s| X_h| Y_h|$.

Она содержит информацию о том, что автомат из состояния a_m переходит в состояние a_s под действием всех тех входных наборов, на которых конъюнкция X_h принимает значение, равное, т.е. X_h= 1. При этом на выходе автомата выдается один и тот же выходной набор, у которого компоненты, входящие вY_h, равны 1, а остальные равны 0. Каждой строке структурной таблицы переходов МПА соответствует конъюнкция A_mX_h, где A_m – конъюнкция, соответствующая коду состояния a_m. Пример: $A_1 X_1 = \bar{Q}_1\bar{Q}_2Q_3x_1$.

Ниже приведен пример построения функциональной схемы МПА сразу по прямой структурной таблице переходов (см. канонический метод структурного синтеза).



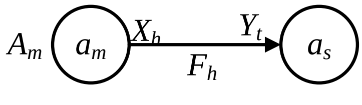
4.4. Синтез автоматов с жесткой логикой

Предполагается, что задана функционально полная система логических элементов, в базисе которых необходимо реализовать автомат. Будем считать, что цена логического элемента равна числу входов в этот элемент. Под ценой схемы будем понимать сумму цен логических элементов.

Минимизацию схем будем проводить по критерию цены. В отличие от матричной реализации МПА, где минимизируется площадь матрицы.

4.4.1. Замечание о тактировании (синхронизации) автомата

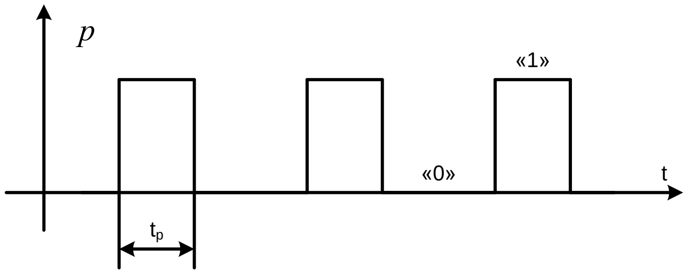
Переход автомата из состояния a_m в состояние a_s можно изобразить следующим подграфом



Такая запись является эквивалентной каждой строчке структурной таблицы переходов МПА.

В выражения для всех сигналов микроопераций и всех сигналов функций возбуждения входит конъюнкция A_mX_h и работа автомата происходит только тогда, когда эта конъюнкция A_mX_h=1.

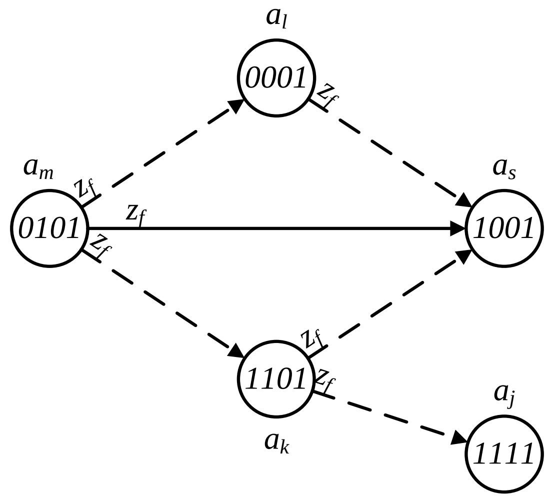
В эту конъюнкцию вводится тактирующая переменная p, таким образом, что pA_mX_h=1



Если говорить об информационном значении сигналов, то автомат будет тактироваться серией сигналов. Тогда работа комбинационной схемы будет тактироваться следующей конъюнкцией pA_mX_h=1. Тактирующий сигнал p в структурную таблицу не записывается, но всегда подразумевается.

4.4.2. Гонки в автомате

Для пояснения сути вопроса изобразим следующий рисунок.



Если при переходе автомата из состояния a_m в состояние a_s должны изменить свои состояния несколько (больше 2-х) элементов памяти, то между ними начинаются <u>состязания.</u> Это объясняется тем, что элементы памяти имеют различные времена переключения, а сигналы функций возбуждения имеют различную скорость распространения в комбинационной схеме. В результате этого при переходе автомата из состояния a_m в состояние a_s под действием входного сигнала z_f автомат может оказаться в одном из промежуточных состояний a_k или a_l (см. рисунок) в зависимости от того, какой из элементов памяти выиграет состязание.

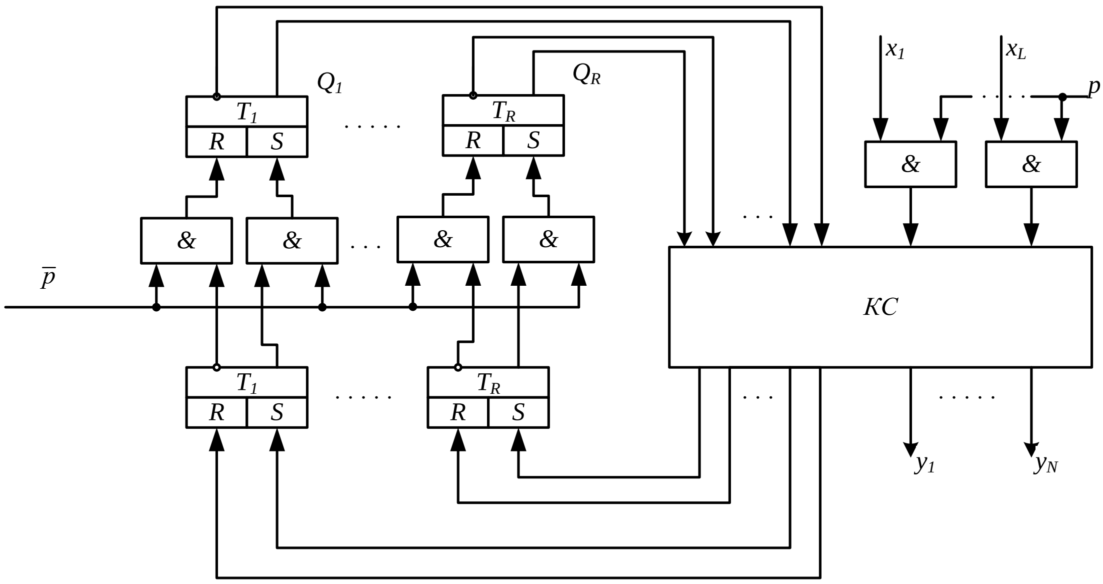
Если при этом же входном сигнале z_f автомат из состояния a_k или состояния a_l перейдет в состояние a_s, то такие состязания называются допустимыми или некритичными.

Если же, например, из состояния a_k, под действием входного сигнала z_f автомат перейдет в состояние a_j (см.пример), то такие состязания называются критическими или гонками.

4.4.3. Методы устранения гонок

1. Применение алгоритмов противогоночного кодирования.
   Недостатки:
   - кодирование заведомо избыточно,
   - создание таких автоматов трудоемко.
2. Тактирование входных сигналов.
   Недостаток:
   - длительность сигнала p ограничена как снизу, так и сверху.

   Снизу - так как все триггеры должны переключаться, сверху - так как тактирующий сигнал должен быть короче самого быстрого сигнала обратной связи.
3. Применение двойной памяти.



4.4.4. Преддешифратор обратной связи

Допустим, автомат имеет 4 элемента памяти. Тогда коды состояний могут быть отображены картой Карно на 4 переменные:

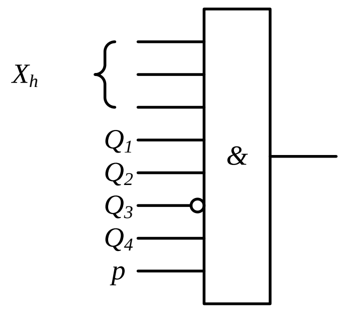
Q_3Q_4 \ Q_1Q_2	α_0 00	α_1 01	α_3 11	α_2 10
β_0' 00				
β_1' 01			A_M	
β_3' 11				
β_2' 10				

Здесь

$$\alpha_0 = \overline{Q_1Q_2} \qquad \beta_0' = \overline{Q_3Q_4}$$
$$\alpha_1 = \overline{Q_1}Q_2 \qquad \beta_1' = \overline{Q_3}Q_4$$
$$\alpha_2 = Q_1\overline{Q_2} \qquad \beta_2' = Q_3\overline{Q_4}$$
$$\alpha_3 = Q_1Q_2 \qquad \beta_3' = Q_3Q_4$$

Тогда $A_m = Q_1Q_2\overline{Q_3}Q_4 = a_3\,\beta_1'$. Будем считать, что $\beta = \beta'\,p$, тогда $pA_m = a_m\beta_m'p = a_m\beta_m$.

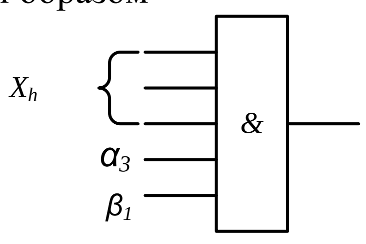
До применения преддешифратора конъюнкция $pA_mX_h = pQ_1Q_2\overline{Q_3}Q_4X_h$ реализуется следующим образом



После применения преддешифратора эта же конъюнкция

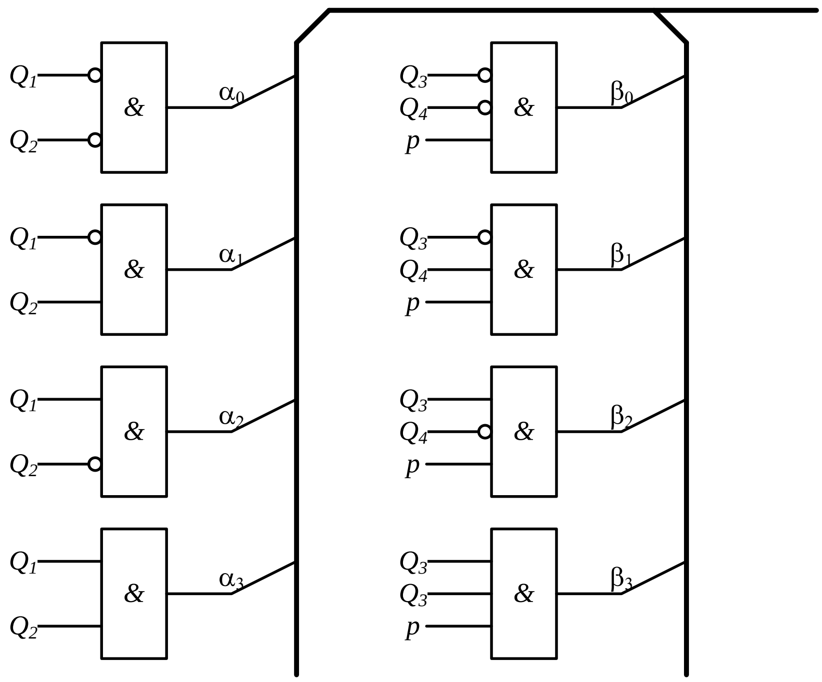
$$pA_mX_h = pQ_1Q_2\overline{Q_3}Q_4X_h = \alpha_3\beta_1X_h$$

реализуется следующим образом



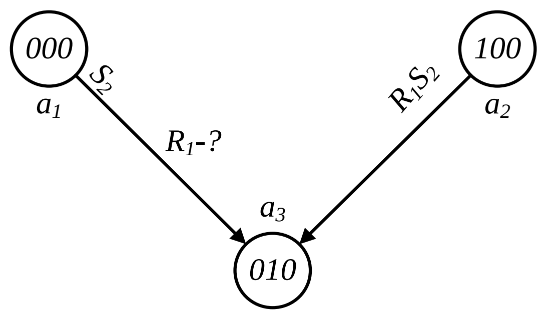
Полный преддешифратор для рассматриваемого примера будет выглядеть так, как показано на рисунке ниже.

Отсюда видно, что преддешифратор обратной связи позволяет сократить число сигналов обратной связи, а как следствие уменьшить цену схемы. Договоримся, что если число элементов памяти нечетное, то код делится таким образом, чтобы справа оставалась меньшая часть на 1.



4.4.5. Доопределение сигналов функций возбуждения

Рассмотрим подграф графа автомата



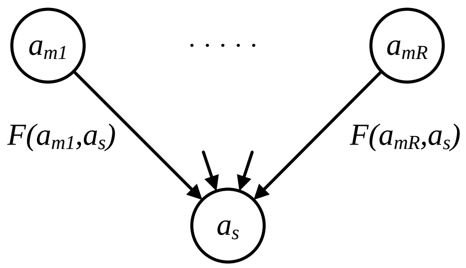
Пусть имеется триггер RS – типа. Из рисунка видно, что выработка сигнала функции возбуждения R_1 на переходе $a_1 \rightarrow a_3$ не нарушит нормальной работы автомата.

Обозначим через $G(a_3)$ множество состояний, из которых есть переходы в состояние a_3: $G(a_3) = \{a_1, a_2\}$.

Под доопределением сигнала функции возбуждения на переходах (a_1, a_3) и (a_2, a_3) будем понимать выдачу на обоих переходах одного и того же множества сигналов функции возбуждения $F(G(a_3), a_3)$,т.е.

$$F(G(a_3), a_3) = F(a_1, a_3) \cup F(a_2, a_3) = \{S_2\} \cup \{R_1, S_2\} = \{R_1, S_2\}.$$

В общем случае, если есть состояния $a_{m_1},...,a_{m_R}$ и состояние a_s, то



$G(a_s)=\{a_{m_1},...,a_{m_R}\}$. Таким образом, для перехода в состояние a_s нужно выдавать $F(G(a_s),a_s)=\bigcup\limits_{r=1}^{R}F(a_r,a_s)$.

Замечание: иногда в общих рассуждениях нас не будет интересовать конкретный выход преддешифратора, соответствующий a_m, а нужно лишь подчеркнуть, что речь идет о выходах преддешифратора при состоянии a_m. В таком случае индекс m будем указывать не внизу, а наверху:

$K(a_m)=$	111	01	$K(a_4)=$	101	10
	α_7	β_1		α_7	β_2
	α^m	β^m		α^4	β^4

4.4.6. Узел в ГСА

Запишем пути перехода из a_m в a_s и из a_n в a_s по рисункам представленным ниже.

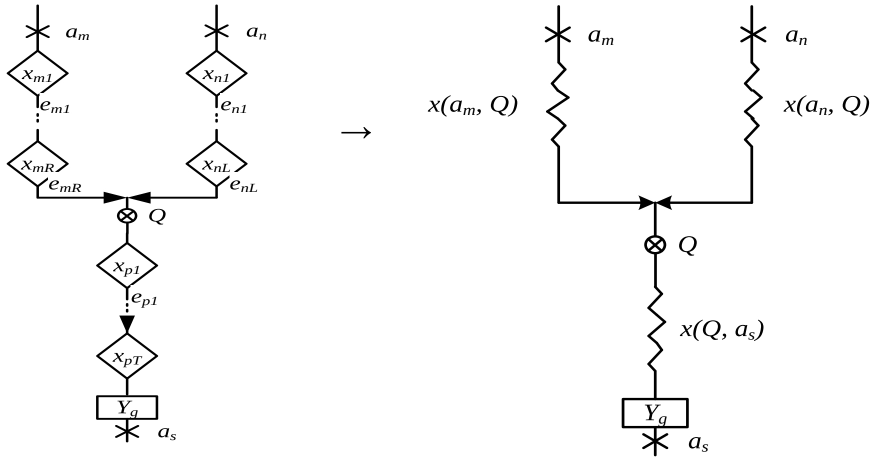
$$a_m x_{m_1}^{e_{m_1}}....x_{m_R}^{e_{m_R}}x_{p_1}^{e_{p_1}}....x_{p_T}^{e_{p_T}}Y_g a_s,$$

$$a_n x_{n_1}^{e_{n_1}}....x_{m_L}^{e_{m_L}}x_{p_1}^{e_{p_1}}....x_{p_T}^{e_{p_T}}Y_g a_s.$$

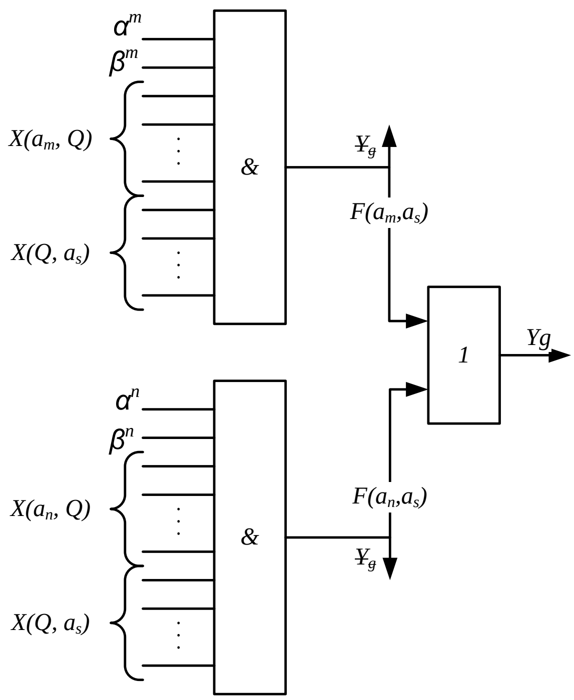
Выходной сигнал $X(a_m,a_s)$ представим в виде $X(a_m,Q)\ X(Q,a_s)$, а $X(a_n,a_s)$ в виде $X(a_n,Q)\ X(Q,a_s)$.

$$A_m=\alpha^m\beta^m.\ A_n=\alpha^n\beta^n.$$

Условные обозначения:



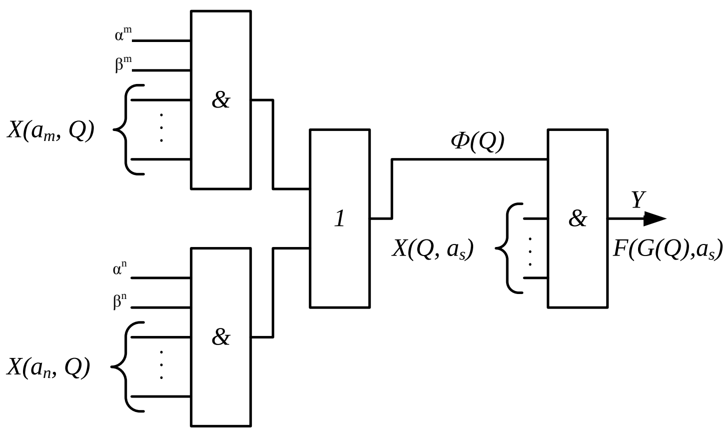
Если бы мы строили эту схему, то получили бы



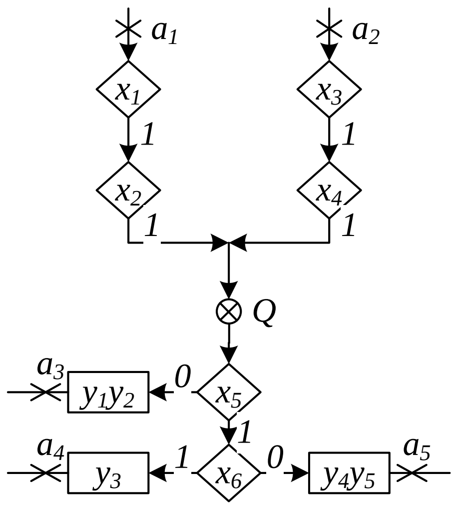
Вынести за скобки $X(Q,a_s)$ в данном случае нельзя, т.к. тогда сигналы функции возбуждения $F(a_m,a_s)$ и $F(a_n,a_s)$ оказываются независящими от $X(Q,a_s)$, что в действительности неверно. Из этого положения можно выйти следующим образом: доопределим сигналы функции возбуждения на переходах (a_m,a_s) и (a_n,a_s).

Множество состояний, из которых есть переход в точку Q, обозначим через $G(Q)=\{a_m,a_n\}$. Будем вырабатывать на обоих переходах один и тот же набор сигналов функции возбуждения: $F(G(Q), a_s) = F(Q_m, a_s) \cup F(a_n, a_s)$.

После до определения получим



Пример:



Коды состояний

$K(a_1) = 0000$

$K(a_2) = 0101$

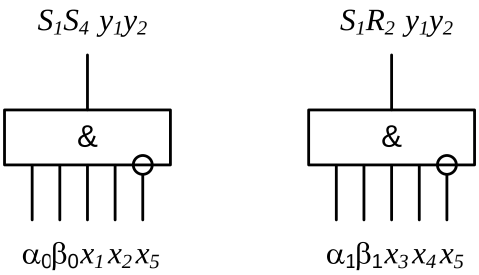
$K(a_3) = 1001$

$K(a_4) = 0011$

$K(a_5) = 0010$

$\alpha\beta$

Строим фрагменты схем для переходов (a_1, a_3) и (a_2, a_3):



Доопределим сигналы функций возбуждения на этих переходах:

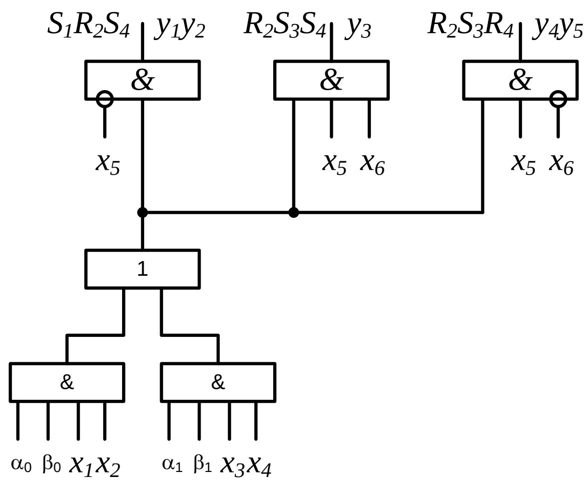
$G(Q)=\{a_1,a_2\}$.

$F(G(Q),a_3)=F(a_1,a_3) \cup F(a_2,a_3) = \{S_1,S_4\} \cup \{S_1,R_1\} = \{S_1R_2S_4\}$.

$F(G(Q),a_4)=F(a_1,a_4) \cup F(a_2,a_4) = \{S_3,S_4\} \cup \{R_2,S_3\} = \{R_2S_3S_4\}$.

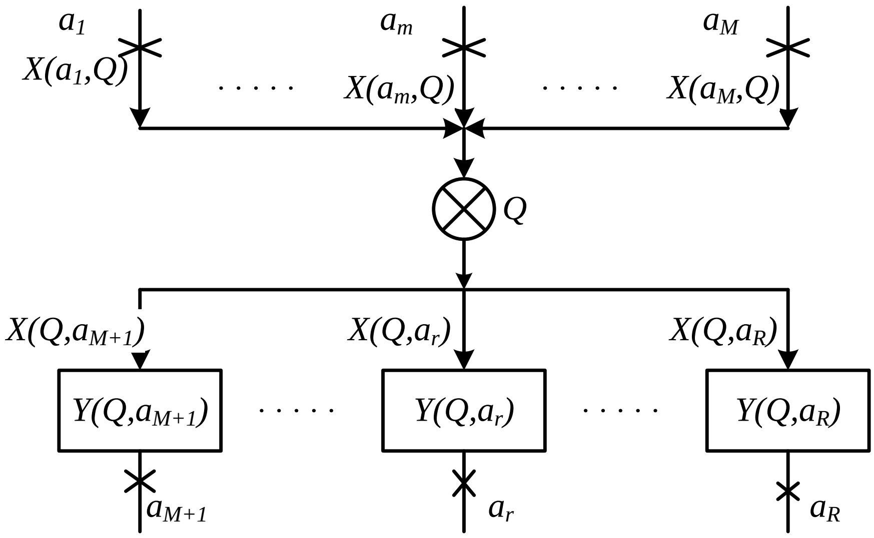
$F(G(Q),a_4)=F(a_1,a_5) \cup F(a_2,a_5) = \{S_3\} \cup \{R_2,S_3R_4\} = \{R_2S_3R_4\}$.

После доопределения схема примет вид:

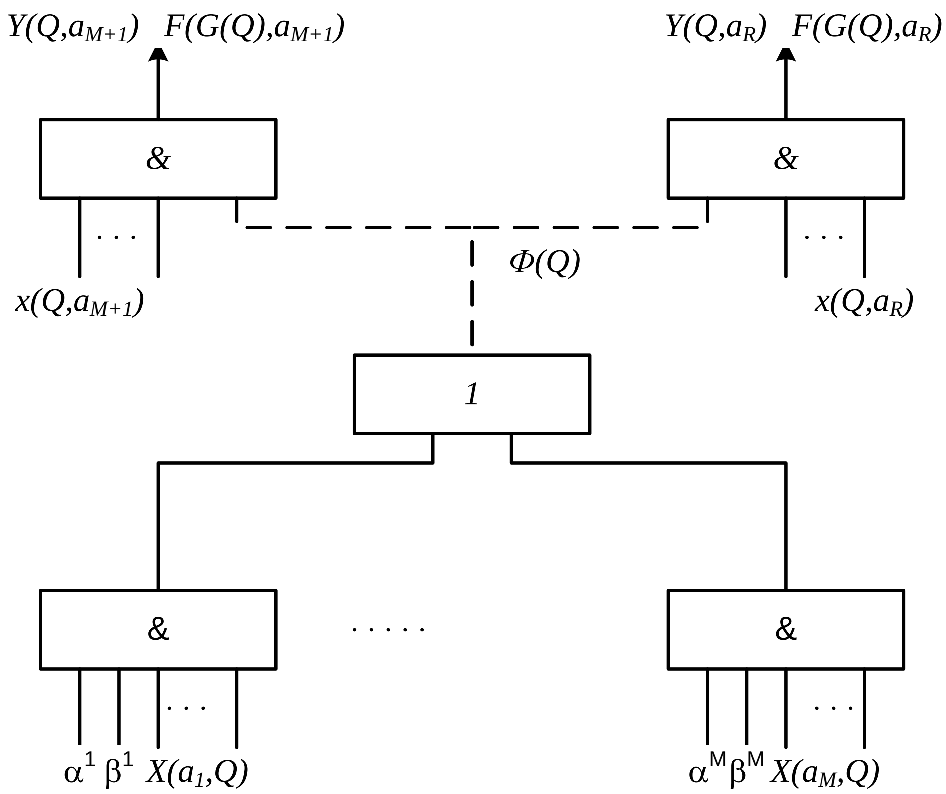


Общий случай

В общем случае могут быть пути, проходящие через точку Q из большого числа состояний в большое число состояний.



После до определения сигналов функций возбуждения из состояний $G(Q)=\{a_1,...,a_M\}$ при переходе в состояние a_r ($r = M+1, ..., R$), как и в предыдущем примере можно построить следующую схему:



Вход условной вершины ГСА будем называть узлом и отмечать символом Q_k, если эта вершина соединена с выходами не менее 2-х других вершин, хотя бы одна из которых - условная.

На отмеченной схеме алгоритма с узлами можно выделить пути следующих видов.

Из состояния в состояние:

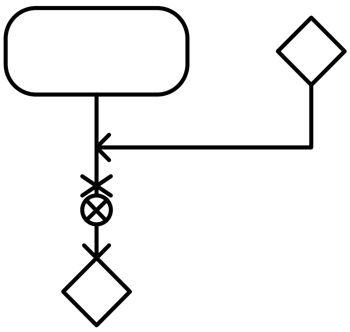
$$a_m x_{m_1}^{e_{m_1}} \ldots . x_{m_R}^{e_{m_R}} Y_t a_s;$$

$$a_m x_{m_1}^{e_{m_1}} \ldots . x_{m_R}^{e_{m_R}} a_1.$$

1. Из состояния в узел:

$$a_m x_{m_1}^{e_{m_1}} \ldots . x_{m_R}^{e_{m_R}} Q_k.$$

Здесь не исключен случай R=0, а именно $a_s Q_k$, т.е. когда вход одной и той же вершины отмечен и состоянием и узлом.



Замечание: если вход условной вершины отмечен и состоянием и узлом, то отметку узла необходимо располагать ближе к входу вершины, чтобы не пропустить путь из состояния в узел.

2. Из узла в состояние:

$$Q_p x_{p_1}^{e_{p_1}} \ldots . x_{p_T}^{e_{p_T}} Y_g a_s.$$

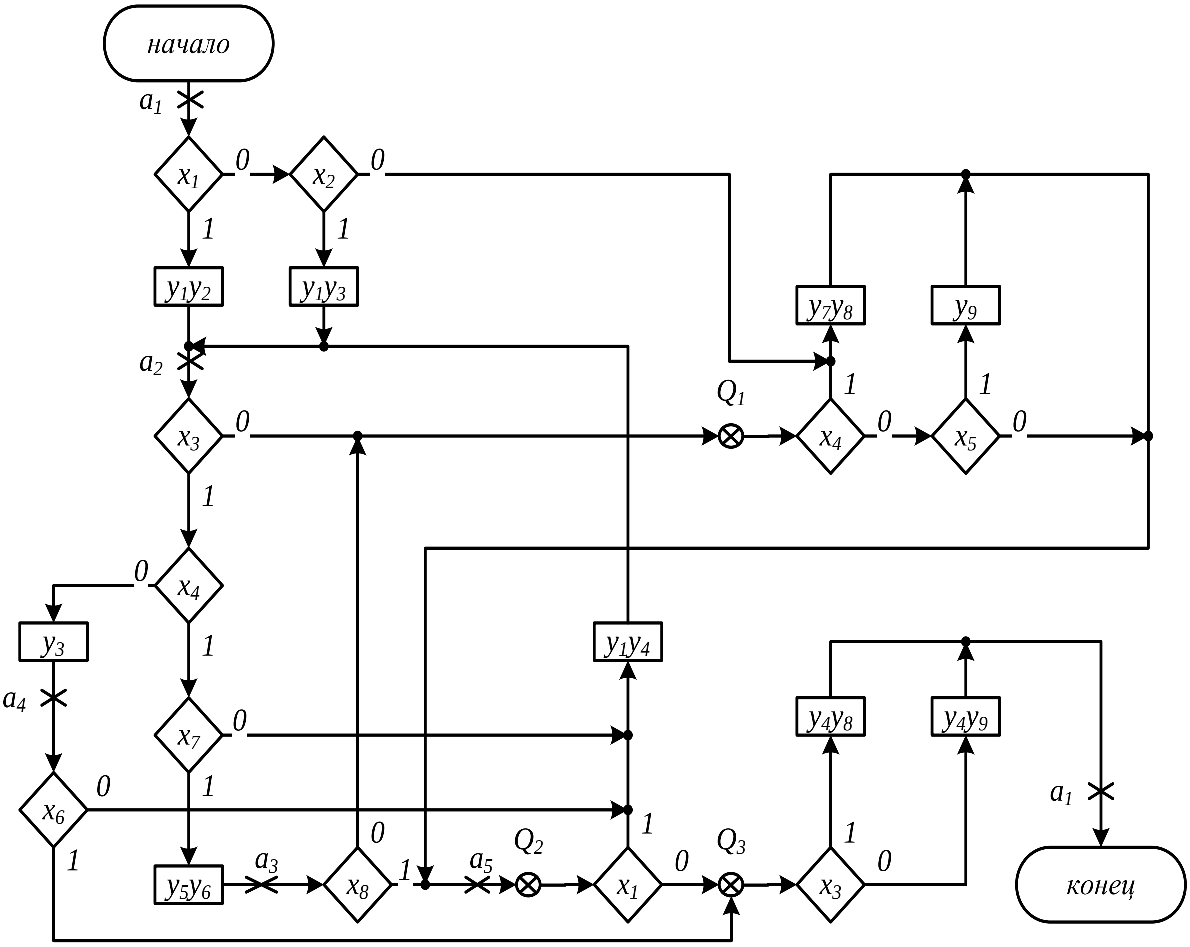
3. Из узла в узел:

$$Q_p x_{p_1}^{e_{p_1}} \ldots . x_{p_R}^{e_{p_R}} Q_k.$$

Сведем все вышеприведенное в таблицу:

$a_m\,Q_p$	$K(a_m)$	$a_s\,Q_k$	$K(a_1)$	X_h	Y_t	F_h
a_m	$K(a_m)$	a_s	$K(a_s)$	$X(a_m,a_s)$	$Y(a_m,a_s)$	$F(a_m,a_s)$
a_m	$K(a_m)$	Q_k	-	$X(a_m,Q_k)$	-	-
Q_p	-	a_1	$K(a_1)$	$X(Q_p, a_s)$	$Y(Q_p,a_s)$	$F(G(Q_p),a_s)$
Q_p	-	Q_k	-	$X(Q_p,Q_k)$	-	-

Пример построения по ГСА обратной структурной таблицы МПА с узлами.



Построение прямой таблицы переходов с узлами:

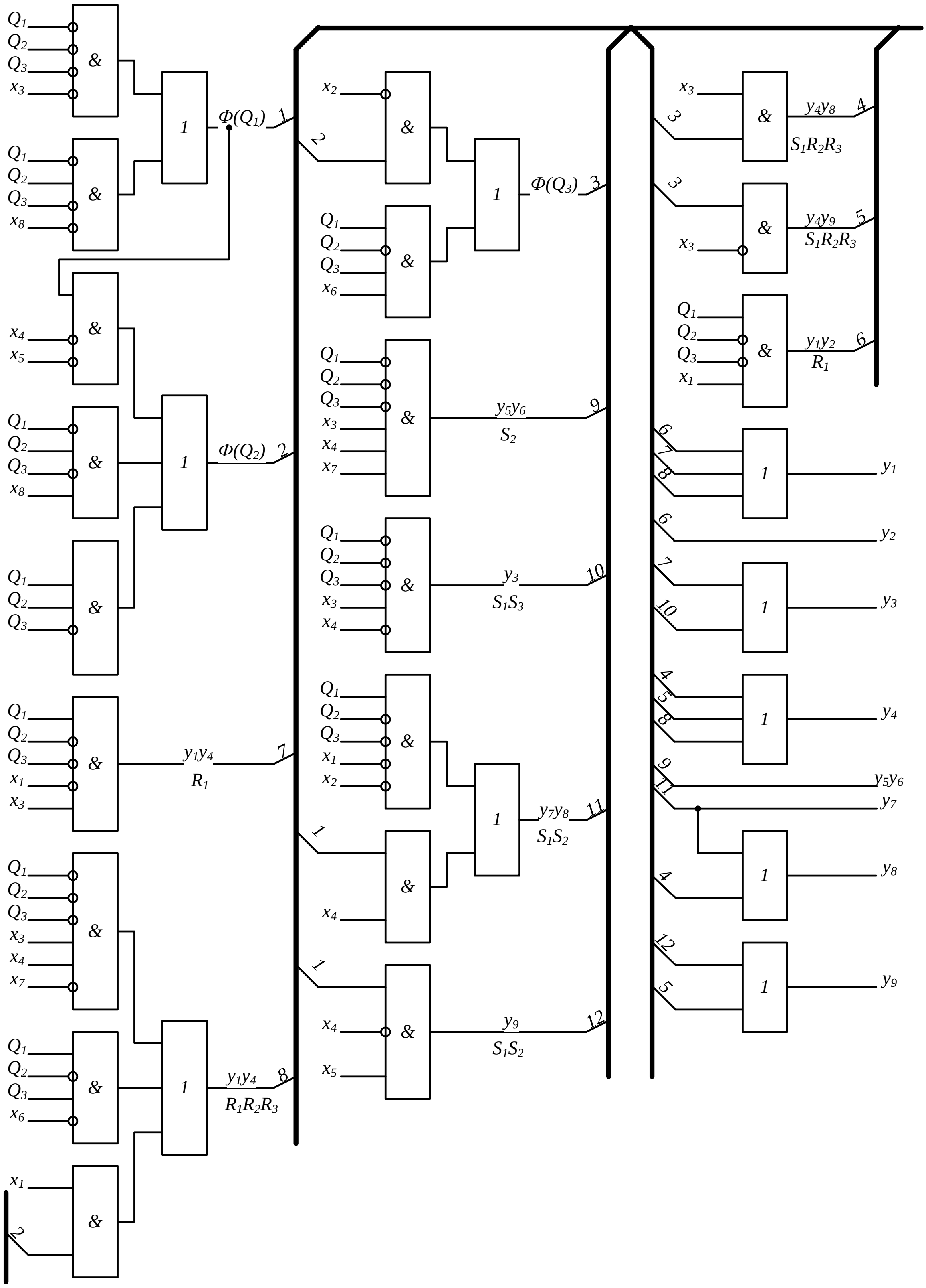
a_m, Q_p	a_s, Q_K	X_h	Y_t
a_1	a_2 a_2 a_5	x_1 $\overline{x_1}x_2$ $\overline{x_1}\overline{x_2}$	y_1y_2 y_1y_3 y_7y_8
a_2	a_3 a_2 a_4 Q_1	$x_3x_4x_7$ $x_3x_4\overline{x_7}$ $x_3\overline{x_4}$ $\overline{x_3}$	y_5y_6 y_1y_4 y_3 -
a_3	Q_2 Q_1	x_8 $\overline{x_8}$	- -
a_4	Q_3 Q_1	x_6 $\overline{x_6}$	- y_1y_4
a_5	Q_2	1	-
Q_1	a_5	x_4	y_7y_8
Q_1	a_5 Q_2	$\overline{x_4}\ x_5$ $\overline{x_4}\overline{x_5}$	y_9 -
Q_2	a_2 Q_3	x_1 $\overline{x_1}$	y_1y_4 -
Q_3	a_1 a_1	x_3 $\overline{x_3}$	y_4y_8 y_4y_9

$G(Q_3) = \{a_4, Q_2\} = \{a_2, a_3, a_4, a_5\};$
$G(Q_2) = \{a_3, a_5, Q_1\};$
$G(Q_1) = \{a_2, a_3\};$
$G(Q_3) = \{a_2, a_3, a_4, a_5\};$
$F(G(Q_3), a_1) = F(a_2, a_1) \cup F(a_3, a_1) \cup F(a_4, a_1) \cup F(a_5, a_1) =$
$= \{S_1\} \cup \{S_1R_2\} \cup \{R_3\} \cup \{R_2\} = \{S_1R_2R_3\}.$

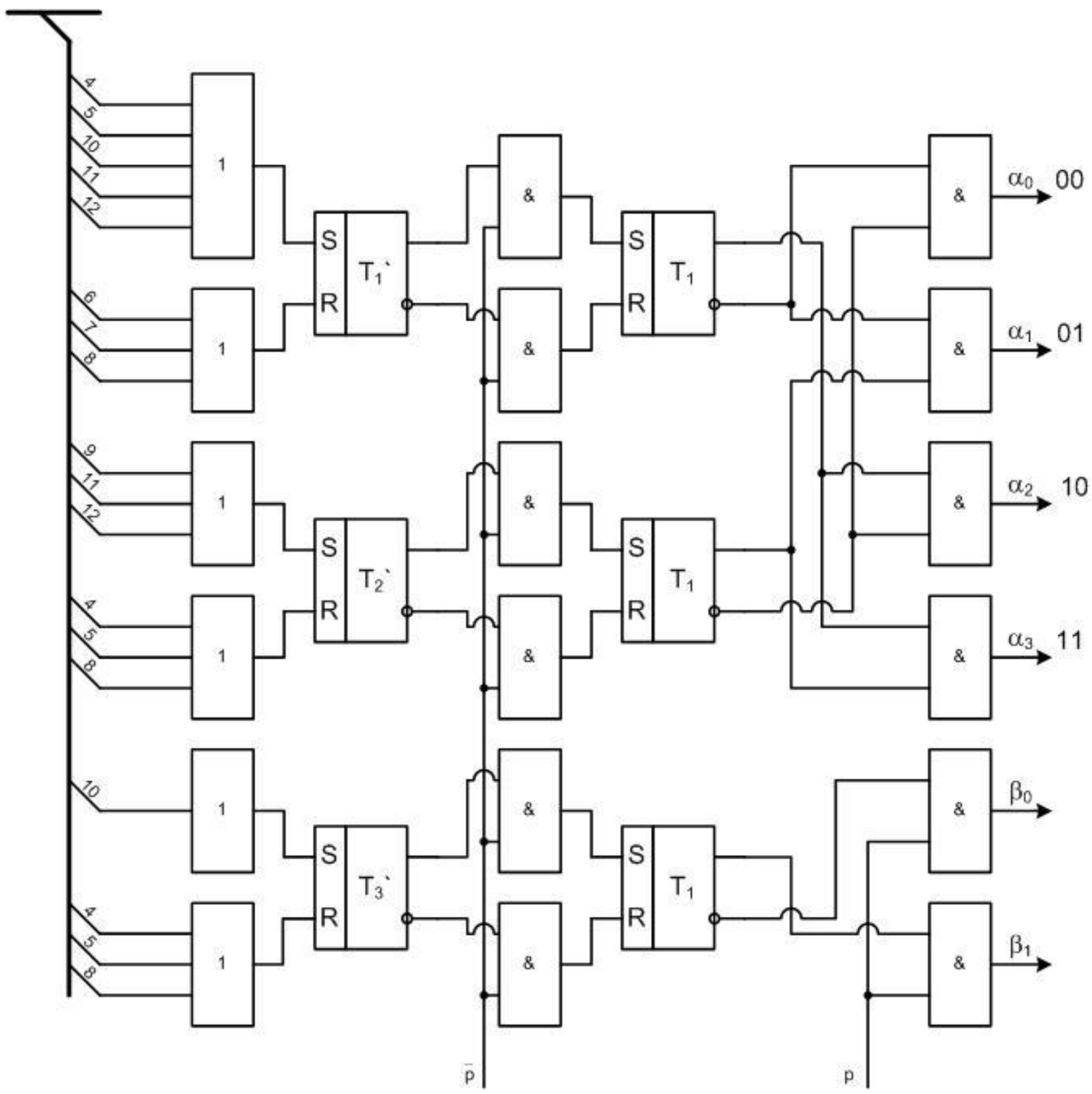
Построение обратной таблицы переходов с узлами:

a_mQ_p	$K(a_m)$	a_s, Q_K	$K(a_s)$	X_h	Y_t	F_h
a_2 a_3	000 010	Q_1	-	$\overline{x_3}$ $\overline{x_8}$	- -	- -
a_3 a_5 Q_1	010 110 -	Q_2	-	x_8 1 $\overline{x_4 x_5}$	- - -	- - -
a_4 Q_2	101 -	Q_3	-	x_6 $\overline{x_1}$	- -	- -
Q_3 Q_3	- -	a_1	100	x_3 $\overline{x_3}$	y_4y_8 y_4y_9	$S_1R_2R_3$ $S_1R_2R_3$
a_1 a_1 a_2 a_4 Q_2	100 100 000 101 -	a_2	000	x_1 $\overline{x_1}\, x_2$ $\overline{x_3}\, x_4\overline{x_7}$ $\overline{x_6}$ x_1	y_1y_2 y_1y_3 y_1y_4 y_1y_4 y_1y_4	R_1 R_1 - R_1R_3 R_1R_2
a_2	000	a_3	010	$x_3x_4x_7$	y_5y_6	S_2
a_2	000	a_4	101	$x_3\overline{x_4}$	y_3	S_1S_3
a_1 Q_1 Q_1	100 - -	a_5	110	$\overline{x_1}\, x_2$ x_4 $\overline{x_4}\, x_5$	y_7y_8 y_7y_8 y_9	S_2 S_1S_2 S_1S_2

4.4.7. Построение функциональной схемы МПА по обратной структурной таблице с узлами

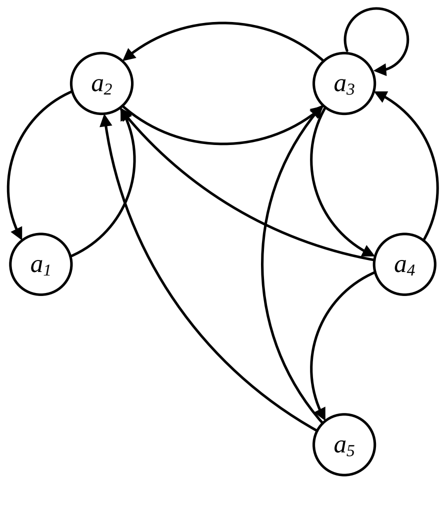


Q1
Q2
Q3
x3
&
1
Φ(Q1)
Φ(Q2)
Φ(Q3)
x2
x4
x5
x6
x7
x8
x1
y5y6
S2
y3
S1S3
y1y4
R1
y7y8
S1S2
y9
S1S2
y1y4
R1R2R3
y4y8
S1R2R3
y4y9
S1R2R3
y1y2
R1
y1
y2
y3
y4
y5y6
y7
y8
y9



4.4.8. Кодирование состояний

Рассмотрим алгоритм, позволяющий минимизировать суммарное число переключений элементов памяти автомата. Пусть имеем следующий граф автомата:



Для кодирования пяти состояний необходимо три переменных $Q_1Q_2Q_3$

$Q_2\ Q_3$ / Q_1	00	01	11	10
0	a_2	a_3	a_4	a_5
1	a_1			

Алгоритм состоит из следующих шагов:

1. Составление матрицы переходов ***T***.

$$T = \begin{vmatrix} a_1a_2 \\ a_2a_1 \\ a_2a_3 \\ a_3a_2 \\ a_3a_3 \\ a_3a_4 \\ a_4a_3 \\ a_4a_2 \\ a_4a_5 \\ a_5a_2 \\ a_5a_3 \end{vmatrix}$$

При переходе a_3a_3 ни один элемент памяти переключаться не будет, поэтому этот переход из матрицы ***T*** может быть исключен.

2. Подсчет весов состояний.

Под весом состояния понимается суммарное число переходов в это состояние и из этого состояния $a_1 – 2; a_2 – 6; a_3 – 5; a_4 – 4; a_5 – 3.$

3. Подсчет весов пар переходов.

Под весом пары понимается суммарный вес компонент (вес пары проставлен у матрицы ***T*** справа).

$$T = \begin{vmatrix} a_1a_2 \\ a_2a_1 \\ a_2a_3 \\ a_3a_2 \\ a_3a_3 \\ a_3a_4 \\ a_4a_3 \\ a_4a_2 \\ a_4a_5 \\ a_5a_2 \\ a_5a_3 \end{vmatrix} \begin{vmatrix} 8 \\ 8 \\ 11 \\ 11 \\ - \\ 9 \\ 9 \\ 10 \\ 7 \\ 9 \\ 8 \end{vmatrix}$$

4. Составление упорядоченной матрицы переходов ***M.***

Матрица ***M*** упорядочивается по мере убывания весов пар переходов.

$$T = \begin{vmatrix} a_2a_3 \\ a_3a_2 \\ a_4a_2 \\ a_3a_4 \\ a_4a_3 \\ a_5a_2 \\ a_1a_2 \\ a_2a_1 \\ a_5a_3 \\ a_4a_5 \end{vmatrix} \begin{vmatrix} 11 \\ 11 \\ 10 \\ 9 \\ 9 \\ 9 \\ 8 \\ 8 \\ 8 \\ 7 \end{vmatrix}$$

5. Составление упорядоченной сокращенной матрицы переходов ***M*.***

$$M^* = \begin{vmatrix} a_2a_3 \\ a_4a_2 \\ a_3a_4 \\ a_5a_2 \\ a_1a_2 \\ a_5a_3 \\ a_4a_5 \end{vmatrix}$$

6. Кодирование состояний

Поскольку состояние a_2 и a_1 оказались в нашем примере в первой строчке сокращенной упорядоченной матрицы ***M****, то их необходимо кодировать соседними кодами.

К(a_2) = 000; К(a_3) = 001.

$Q_2 Q_3$ / Q_1	*00*	*01*	*11*	*10*
0	a_2	a_3		
1				

Вычеркнем эту строку (a_2 a_3) из матрицы ***M*.***

Введем переменную j (это индекс состояния, которое кодируется на данном этапе) и построим матрицу $\boldsymbol{M_j}$, выбрав из ***M**** пары состояний с индексом j. В нашем примере j = 4.

$$M_4 = \begin{vmatrix} a_4a_2 \\ a_3a_4 \\ a_4a_5 \end{vmatrix}$$

Далее формируется множество B_j, состоящее из состояний, которые уже закодированы и присутствуют в матрице $\boldsymbol{M_j}$.

$$B_4 = \{a_3, a_2\}.$$

Формируются множества

$$C_3^1 = \{011, 101\}; C_2^1 = \{100, 011\}.$$

Это множества свободных кодов длины 1, относительно кодов состояний a_3 и a_2.

Далее формируется множество $D^1{}_j$. Единица говорит о том, что это коды длинны 1 относительно уже закодированных состояний (один из кодов этого множества должен быть присвоен состоянию a_4).

$$D_4^1 = C_3^1 \cup C_2^1 = \{011,101,010,100\}.$$

Затем строятся кодовые расстояния по Хеммингу:

$W_{011} = |011 - 000|^2 + |011 - 001|^2 \times 2 = 2 + 1 \times 2 = 4;$
$W_{101} = |101 - 000|^2 + |101 - 001|^2 \times 2 = 2 + 1 \times 2 = 4;$
$W_{010} = |010 - 000|^2 + |010 - 001|^2 \times 2 = 1 + 2 \times 2 = 5;$
$W_{100} = |100 - 000|^2 + |100 - 001|^2 \times 2 = 1 + 2 \times 2 = 5.$

Состоянию a_4 присваиваем код с минимальным кодовым расстоянием по Хеммингу.

В примере это коды 011 и 101;выберем K(a_4) = 011.

Q_2Q_3 / Q_1	*00*	*01*	*11*	*10*
0	a_2	a_3	a_4	
1				

$j = 5;$ $M = \begin{vmatrix} a_5 a_2 \\ a_5 a_3 \\ a_4 a_5 \end{vmatrix}. B_5 = \{a_2, a_3, a_4\}.$

$$C_2^1 = \{010,100\}; C_3^1 = \{101\}; C_4^1 = \{010,111\}; D_5^1 = \{010,100,101,111\}.$$

$W_{010} = |010 - 000|^2 + |010 - 001|^2 + |010 - 011|2 = 1 + 2 + 1 = 4;$
$W_{100} = |100 - 000|^2 + |100 - 001|^2 + |100 - 011|2 = 1 + 2 + 3 = 6;$
$W_{101} = |101 - 000|^2 + |101 - 001|^2 + |101 - 011|2 = 2 + 1 + 2 = 5;$
$W_{111} = |111 - 000|^2 + |111 - 001|^2 + |111 - 011|2 = 3 + 2 + 1 = 6.$

Состоянию a_5 присваиваем код с минимальным кодовым расстоянием по Хеммингу.

В примере это только код 010; выберем К(a_5) = 010.

Q_1 \ Q_2Q_3	*00*	*01*	*11*	*10*
0	a_2	a_3	a_4	a_5
1				

j = 1, M_1 = $|a_1a_2|$. Очевидно, что для a_1 подойдет К(a_1) = 100.

Q_1 \ Q_2Q_3	*00*	*01*	*11*	*10*
0	a_2	a_3	a_4	a_5
1	a_1			

Кодирование завершено.

7. Расчет коэффициента кодирования

$$k = \frac{\sum t_{ms}}{P},$$

где P- общее число переходов; t_{ms} - число переключений элементов памяти при переходе из состояния a_m в состояние a_s.

$$M^* = \begin{array}{|c|}a_2a_3\\ a_4a_2\\ a_3a_4\\ a_5a_2\\ a_1a_2\\ a_5a_3\\ a_4a_5\end{array}\begin{array}{|c|}\multicolumn{1}{c}{t_{ms}}\\ 2\times1=2\\ 1\times2=2\\ 2\times1=2\\ 1\times1=1\\ 2\times1=2\\ 1\times2=2\\ 1\times1=1\end{array}$$

$$\sum t_{ms} = 12$$

Кодирование считается удовлетворительным, если $k \leq 1{,}5$.

Для нашего примера

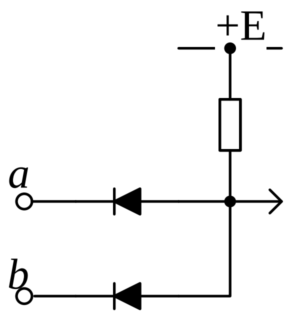
$$k = \frac{\sum t_{ms}}{P} = \frac{12}{10} = 1{,}2.$$

5. Матричная реализация МПА

При синтезе МПА с жесткой логикой в качестве критерия минимизации выбирается цена схемы. Под ценой схемы понимается сумма цен логических элементов. Под ценой логического элемента понимается число входов в этот логический элемент.

5.1. Способы кодирования потенциальных двоичных сигналов

При позитивном способе кодирования за единицу информации всегда принимается высокий уровень напряжения. Для негативной – наоборот. Рассмотрим следующую схему.



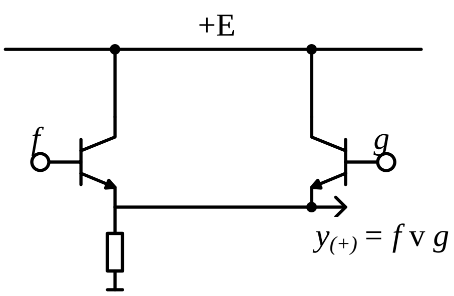
$$y = a \circ b$$

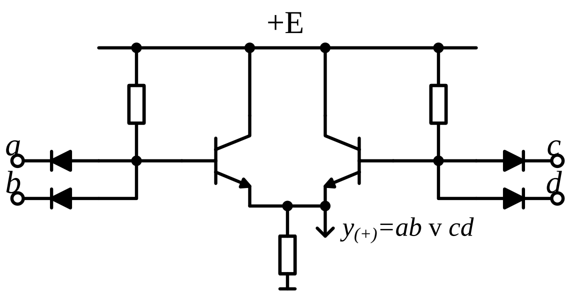
$$y_{(+)} = a \& b$$

$$y_{(-)} = a \vee b$$

a	*b*	*y*
(1) +E(0)	(1) +E (0)	(1) +E (0)
(0)⊥(1)	(1) +E (0)	(0)⊥(1)
(1) +E (0)	(0)⊥(1)	(0)⊥(1)
(0)⊥(1)	(0)⊥(1)	(0)⊥(1)

Тип проводимости транзистора	(+) Позитивная система кодирования	(-) Негативная система кодирования
n - p - n	“1” “0”	“0” “1”
p - n - p	“1” “0”	“0” “1”



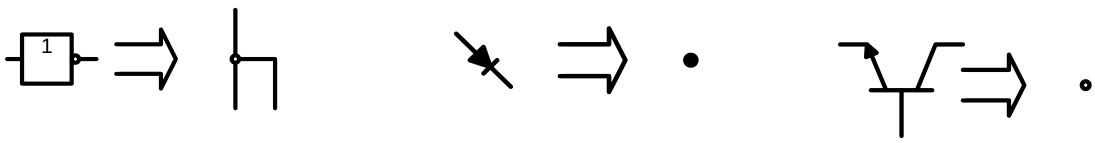


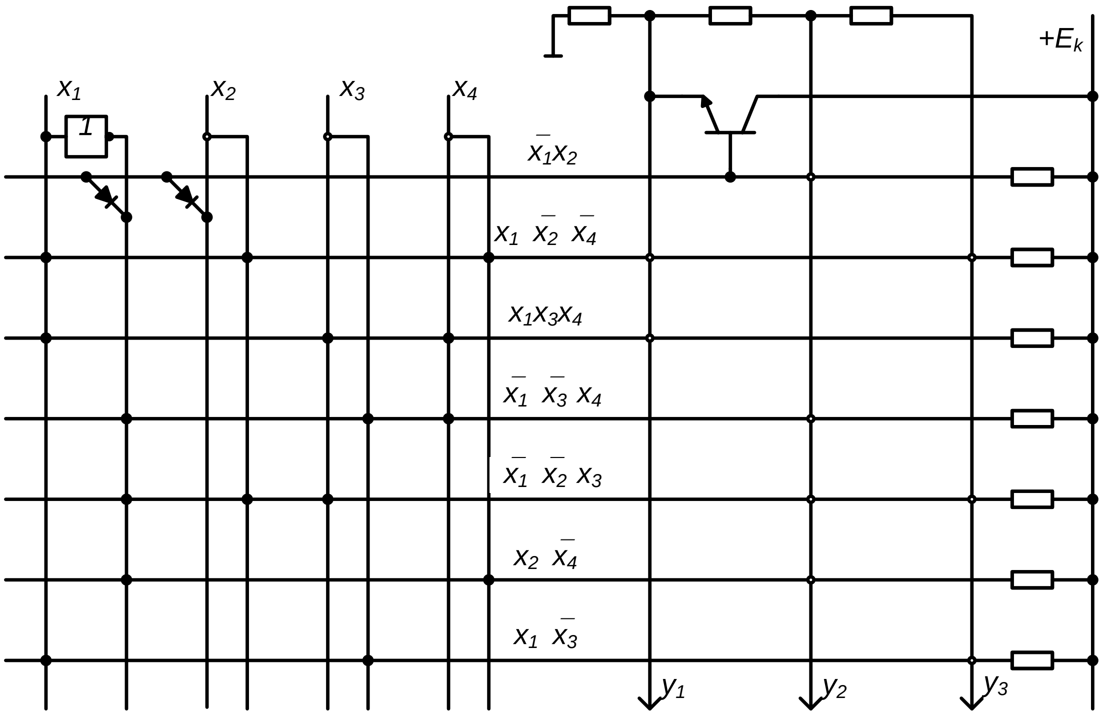
5.2. Матричная реализация комбинационных схем

Допустим, задана некоторая система булевых функций

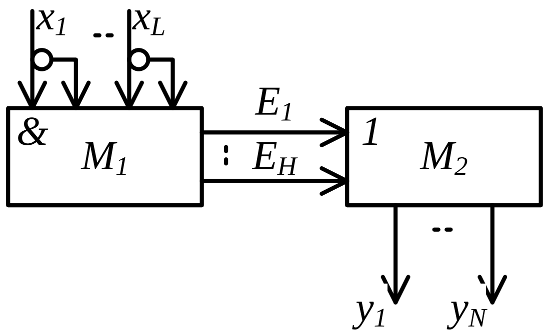
$$y_1 = \bar{x}_1 x_2 \vee x_1 \bar{x}_2 \bar{x}_4 \vee x_1 x_3 x_4;$$
$$y_2 = \bar{x}_1 x_2 \vee \bar{x}_1 \bar{x}_3 x_4 \vee \bar{x}_1 \bar{x}_2 x_3 \vee x_2 \bar{x}_4;$$
$$y_3 = x_1 \bar{x}_2 \bar{x}_4 \vee x_1 \bar{x}_3 \vee \bar{x}_1 \bar{x}_2 x_3.$$

Реализуем данную систему булевых функций на двухматричной структуре. При этом будем использовать следующие обозначения.





Общий вид двухматричной структуры



Особенности:

Нет смысла минимизировать суммарное число входов.

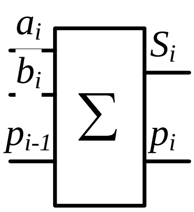
При матричной реализации комбинаторных схем минимизируется суммарная площадь матриц $\boldsymbol{M_1}$ и $\boldsymbol{M_2}$,т.е. **S(*M₁*)=2LH** и **S(*M₂*)=HN.**

При этом систему булевых функций необходимо привести к минимальной форме. Минимальной формой называется форма, у которой минимальное число букв.

При приведении системы булевых функций к минимальной форме используются методы минимизации булевых функций. Минимизация площади матриц $\boldsymbol{M_1}$ и $\boldsymbol{M_2}$ может быть осуществлена также с помощью замены переменных.

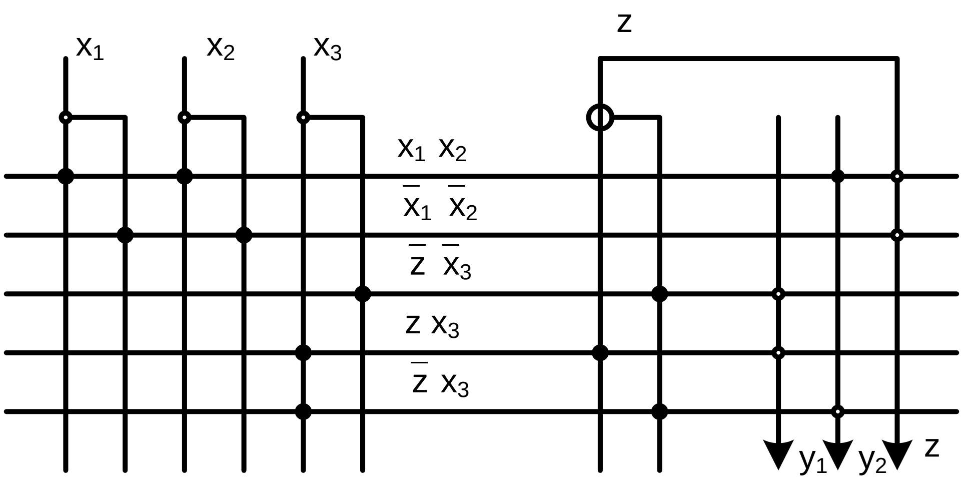
Пример. Матричная реализация комбинационного сумматора.

Пусть задана система, описывающая работу сумматора



$$y_1 = \overline{(x_1x_2 \vee \bar{x}_1\bar{x}_2)}\,\bar{x}_3 \vee (x_1x_2 \vee \bar{x}_1\bar{x}_2)\,x_3 - \textit{сумма};$$
$$y_2 = x_1x_2 \vee \overline{(x_1x_2 \vee \bar{x}_1\bar{x}_2)}\,x_3 - \textit{перенос}.$$

Введем следующую замену переменной $z = x_1x_2 \vee \bar{x}_1\bar{x}_2$, тогда
$y_1 = \bar{z}\,\bar{x}_3 \vee z\,x_3;\ y_2 = x_1x_2 \vee \bar{z}\,x_3.$



5.3. Тривиальная матричная реализация МПА

Пусть имеем прямую структурную таблиц МПА.

a_m	$K(a_m)$	a_s	$K(a_s)$	X_h	Y_t		D_h	h
a_1	000	a_2	010	x_1	$y_1y_2y_3$	Y_1	D_2	1
		a_2	010	$\overline{x_1}\, x_2$	$y_{10}y_{11}y_{12}$	Y_5	D_2	2
		a_5	100	$\overline{x_1 x_2}$	y_{13}	Y_7	D_1	3
a_2	010	a_1	000	$\overline{x_1 x_3}$	$y_2y_{10}y_{12}$	Y_6	-	4
		a_2	010	$\overline{x_1}\, x_3$	y_7y_8y	Y_4	D_2	5
		a_4	011	x_1	y_4	Y_2	$D_2\, D_3$	6
a_3	101	a_1	000	x_5	-	Y_0	-	7
		a_2	010	$\overline{x_5}$	$y_7y_8y_9$	Y_4	D_2	8
a_4	011	a_1	000	$\overline{x_2 x_3}$	$y_2y_{10}y_{11}$	Y_6	-	9
		a_2	010	$\overline{x_2}\, x_3$	$y_7y_8y_9$	Y_4	D_2	10
		a_3	101	x_2x_3	y_5y_6	Y_3	$D_1\, D_3$	11
		a_3	101	$x_2\overline{x_3}$	$y_7y_9y_{14}y_{15}$	Y_8	$D_1\, D_3$	12
a_5	100	a_1	000	x_4	$y_2y_{10}y_{12}$	Y_6	-	13
		a_5	100	$\overline{x_4}$	-	Y_0	D_1	14

Как и ранее, каждой строке соответствует конъюнкция A_mX_h, если она равна 1, то надо вырабатывать сигнал, чтобы перевести автомат из одного состояния в другое.

Для минимизации суммарной площади матриц при матричной реализации используются триггеры D и T типов.

Из множества Y всех микроопераций выделим подмножество Y^* микроопераций, которые имеют только по одной конъюнкции. К ним относятся:

$$Y^* = \{y_1, y_3, y_4, y_5, y_6, y_{11}, y_{13}, y_{14}, y_{15}\}.$$

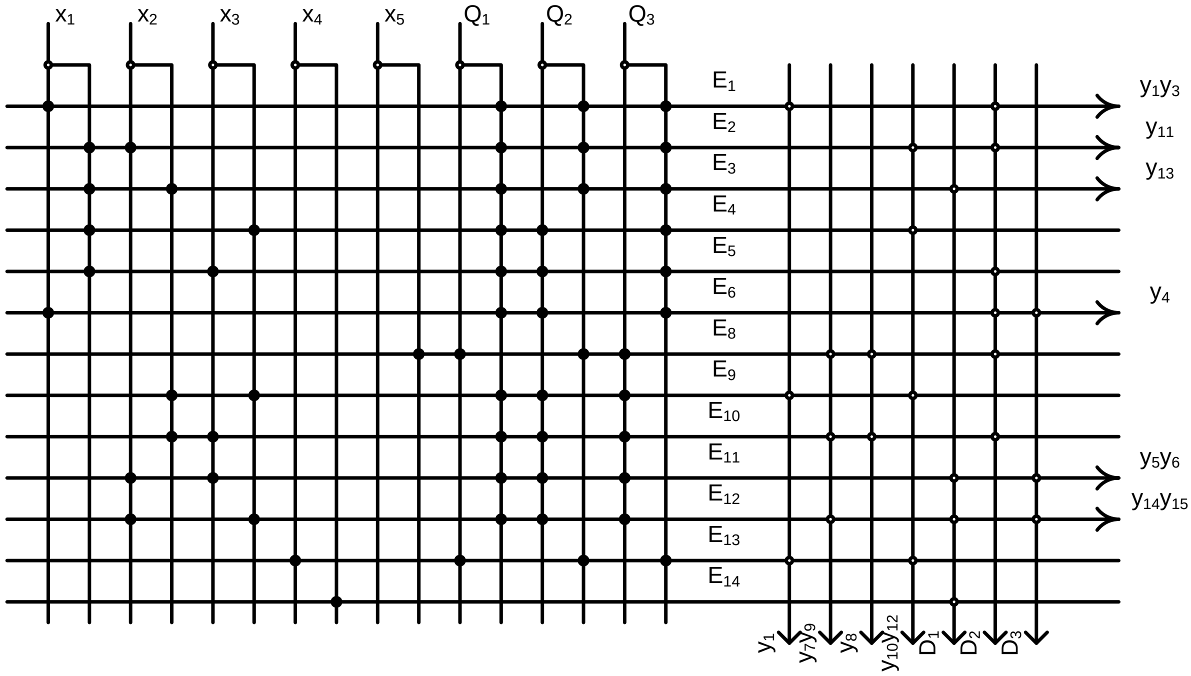
Далее находим

$$Y^{**}=Y \setminus Y^* = \{y_2, y_7, y_8, y_9y_{10}, y_{12}\}.$$

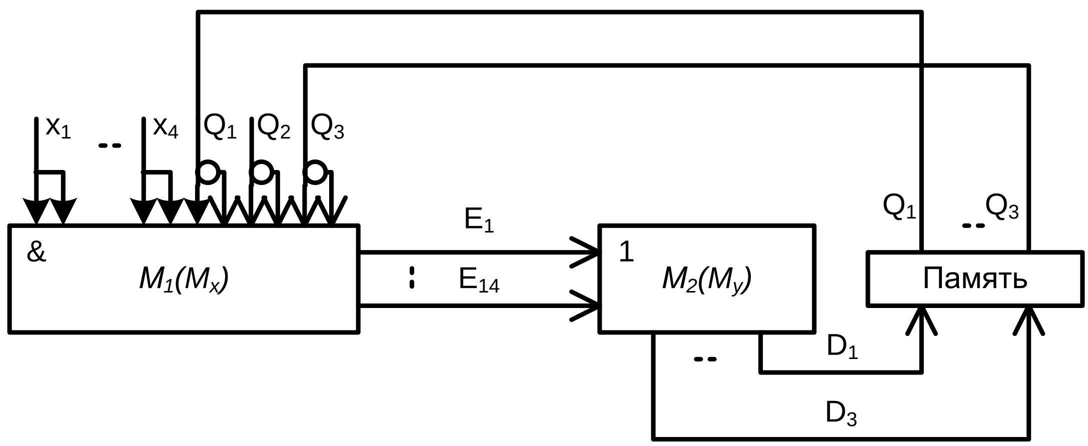
Затем, из множеств Y^* и Y^{**} выделяем те микрооперации, которые встречаются всегда вместе

$$\left.\begin{matrix}\{y_1, y_3\}\\ \{y_5, y_6\}\\ \{y_{14}, y_{15}\}\end{matrix}\right\} = Y'; \quad \left.\begin{matrix}\{y_7, y_8\}\\ \{y_{10}, y_{12}\}\end{matrix}\right\} = Y''$$

их будем вырабатывать на одной шине. Соответствующая матричная реализация МПА приведена ниже.



В общем виде тривиальная матричная реализация МПА будет выглядеть следующим образом.



5.4. Кодирование логических условий (замена входных переменных)

При матричной реализация МПА суммарная площадь матриц M_1 и M_2 определяется выражением $S=2H(L+R) + H(N+R)$. Двойка в сумме для M_1 объясняется тем, что нам необходимо иметь как прямые, так и инверсные значения.

В автоматах средней сложности обычно $L \approx 30$, $H \approx 200$, $R \approx 6$, $N \approx 50$. $S = 25{,}6 \times 10^3$ бит.

Обозначим через $X(a_m)$ множество входных переменных, встречающихся на всех переходах из состояния a_m. Для примера

$x(a_1) = \{x_1, x_2\}$; $x(a_2) = \{x_1, x_3\}$; $x(a_3) = \{x_5\}$; $x(a_4) = \{x_2, x_3\}$; $x(a_5) = \{x_4\}$.

Введем следующее обозначение: $G_m = |X(a_m)|$. Вертикальные черточки обозначают *число* элементов множества

$$G = \underbrace{max}_{m} G_m = \underbrace{max}_{m} |X(a_m)|.$$

Для примера $G_1=2$; $G_2=2$; $G_3=1$; $G_4=2$; $G_1=1$ и тогда $G=2$.
Введем вместо переменных $X=\{x_1, x_2, \ldots, x_L\}$ множество колирующих переменных $P=\{p_1, p_2, \ldots, p_G\}$. Для примера $X=\{x_1, x_2, x_3, x_4, x_5\}$, $P=\{p_1, x_2\}$.

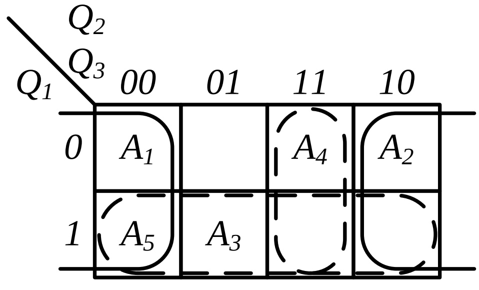
Допустим, что замена переменных уже произведена, и мы имеем следующую таблицу, построенную по столбцам a_m и X_h

P / A	p_1	p_2
a_1	x_1	x_2
a_2	x_1	x_3
a_3	x_5	-
a_4	x_2	x_3
a_5	-	x_4

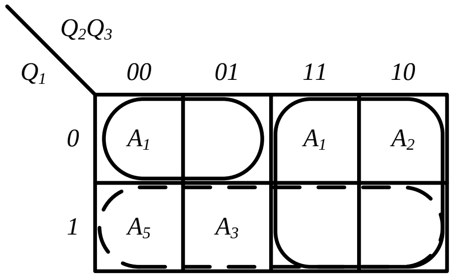
Если в состоянии a_m переменная p_g заменяет x_l, то, когда автомат находится в состоянии a_m должно выполнятся условие $p_g = x_l$; $A_m=1$.

$$p_1 = A_1x_1 \vee A_2x_1 \vee A_3x_5 \vee A_4x_2 \vee [A_5x_1 \vee A_5x_2 \vee A_5x_5] =$$
$$= (A_1 \vee A_2 \vee [A_5])x_1 \vee (A_4 \vee [A_5])x_2 \vee (A_3 \vee [A_5])x_5 = \overline{Q}_3x_1 \vee Q_2Q_3x_2 \vee Q_1x_5$$

Когда автомат находится в состоянии a_1, то конъюнкция $A_1= 1$ и соответственно $p_1 = x_1$. Если автомат находится в состоянии a_2, то конъюнкция $A_2 = 1$ и $p_1 = x_1$.$a_3 \rightarrow A_3 = 1$; $p_1 = x_5$; $a_4 \rightarrow A_4 = 1$; $p_1 = x_2$.



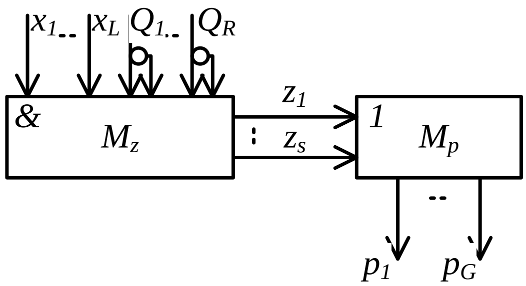
$$p_2 = A_1x_2 \vee A_2x_3 \vee A_4x_3 \vee A_5x_4 \vee [A_3x_2 \vee A_3x_3 \vee A_3x_4] =$$
$$= (A_1 \vee [A_3])x_2 \vee (A_2 \vee A_4 \vee [A_3])x_3 \vee (A_5 \vee [A_3])x_4 = \bar{Q}_1\bar{Q}_2x_2 \vee Q_2x_3 \vee Q_1x_4$$



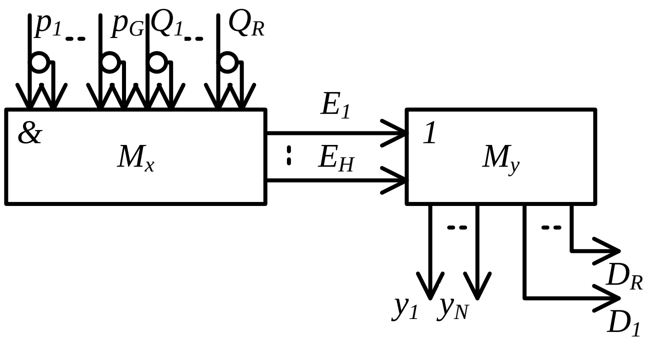
При замене входных переменных в структурной таблице можно вписать столбец P_h.

a_m	…	P_h
a_1		p_1
		$\bar{p}_1 p_2$
		$\bar{p}_1 \bar{p}_2$

Для выработки сигналов p_1, $p_2, \ldots,$ p_G строится двухматричная структура



Далее сигналы $p_1, p_2, \ldots, p_G$ подаются на матрицу M_x.



Таким образом, при замене входных переменных сокращается площадь матрицы M_x (M_y остаетсябез изменений).

В среднем G ≈ 5.

Без замены переменных $S(M_x) = 2(L + R)H = (60 + 12)\ 200 = 14{,}4 \times 10^3$.

С заменой переменных:

$S(M_x) = 2(G + R)H = 4{,}4 \times 10^3$;

$S(M_z) = (L + 2R)S = 1{,}5 \times 10^3$;
$S(M_p) = S * G = 0{,}2 \times 10^3$;
$S_{суммарная} = 6 * 10^3$.

5.5. Кодирование микроопераций

Напомним, что площадь матрицы M_y - $S(M_y) = H(N + R)$.

В общем случае МПА может реализовать $Y_0, Y_1, \dots, Y_T$ микрокоманд. Закодируем каждую микрокоманду Y_t следующим вектором:

$$K(Y_t) = (e_{t1}, \dots, e_{tD}), \text{ где } e_{td} \in \{0, 1\};\ D =]\log_2 (T + 1)[.$$

Для примера закодируем микрокоманды тривиальным образом, т.е. в соответствие с двоичным эквивалентом номера микрокоманды (можно закодировать и как - то по-другому).

$$K(Y_0) = 0000 \qquad B_0 = \overline{g_1}\,\overline{g_2}\,\overline{g_3}\,\overline{g_4}$$
$$K(Y_1) = 0001 \qquad B_1 = \overline{g_1}\,\overline{g_2}\,\overline{g_3}\,g_4$$
$$\dots\dots\dots\dots \qquad \dots$$
$$K(Y_8) = 1000 \qquad B_8 = g_1\overline{g_2}\,\overline{g_3}\,\overline{g_4}$$

Коду каждой микрокоманды Y_t ставим в соответствие конъюнкцию:

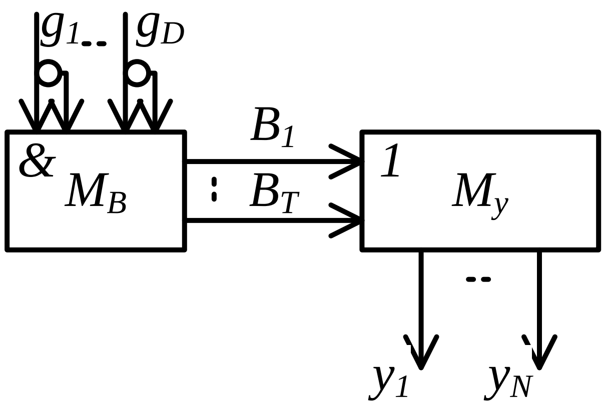
$$K(Y_t) \rightarrow B_t = g_1^{e_{t1}} \dots g_D^{e_{tD}}.$$

Из таблицы переходов МПА видно, что

$Y_0 = 0$;
$Y_1 = \{y_1, y_2, y_3\}$;
$Y_2 = \{y_4\}$;
$Y_3 = \{y_5, y_6\}$;
…
$Y_8 = \{y_7, y_1, y_{14}, y_{15}\}$.

После анализа микрокоманд можно увидеть, что, например, микрооперация $y_2 \in Y_1$ и $y_2 \in Y_6$. Т.е. $y_2 = 1$, когда выполняется конъюнкция $B_1 = 1$ и $B_6 = 1$. Другими словами, можно записать, что: $y_2 = B_1 \vee B_6$.

Для микрооперации y_7:
$y_7 \in Y_4$ и $y_7 \in Y_8$. $y_7 = B_4 \vee B_8$.
$y_1 = B_1 = \overline{g_1}\,\overline{g_2}\,\overline{g_3}\,g_4$;
$y_2 = B_1 \vee B_6 = \overline{g_1}\,\overline{g_2}\,\overline{g_3}\,g_4 \vee \overline{g_1}\,g_2 g_3 \overline{g_4}$;
$y_3 = B_1 = \overline{g_1}\,\overline{g_2}\,\overline{g_3}\,g_4$;
…
$y_{15} = B_8 = g_1\overline{g_2}\,\overline{g_3}\,\overline{g_4}$.



Кроме того, встречаются только один раз следующие микрооперации, причем некоторые из них образуют целые микрокоманды:

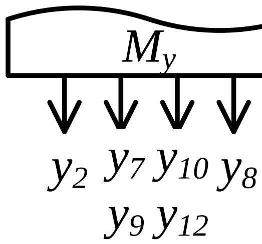
$$\left.\begin{array}{c}\{y_1, y_3\}\\ \{y_4\} = Y_2\\ \{y_5, y_6\} = Y_3\\ \{y_{11}\}\\ \{y_{13}\} = Y_7\\ \{y_{14}, y_{15}\}\end{array}\right\} = Y'.$$

Y^*- множество микроопераций, которые можно снять прямо с матрицы M_x. Тогда с матрицы M_y надо будет снимать оставшиеся микрооперации $Y^{**}=Y \backslash Y^*$ и кроме того, Y_2,Y_3 и Y_7 не нужно кодировать. Т.о., остается 6 микрокоманд, которые нужно закодировать. Для кодирования 6 микрокоманд нам потребуется 3 двоичные переменные.

Допустим, микрокоманды закодированы следующим образом:

$Y_0 \rightarrow 000$
$Y_1 \rightarrow 100$
$Y_4 \rightarrow 010$
$Y_5 \rightarrow 001$
$Y_6 \rightarrow 001$
$Y_8 \rightarrow 110$

В оставшемся множестве микроопераций Y^{**} всегда встречаются микрооперации $\{y_7, y_9\}$;$\{y_{10}, y_{12}\}$. В результате получаем, что с выхода матрицы M_y надо снимать следующие микрооперации:



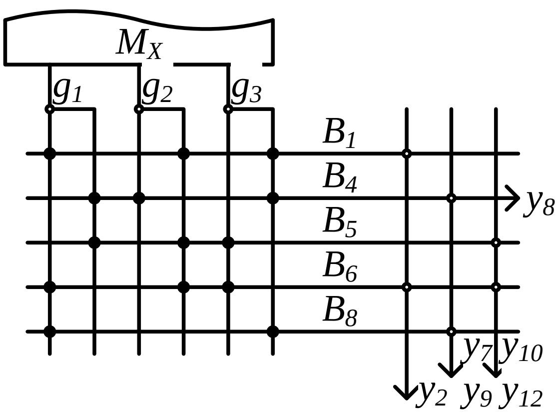
Тот факт, что $y_2 \in Y_1$, $y_2 \in Y_6$ запишем следующим образом:

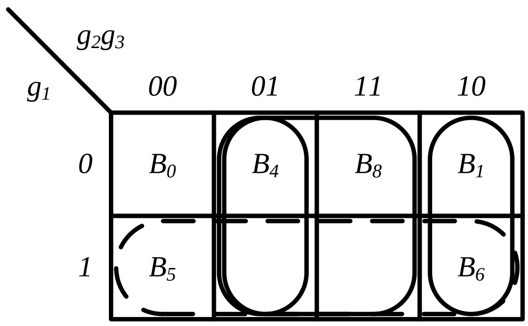
$y_2 \in Y_1,\ Y_6; y_2 \in B_1 \vee B_6 = \overline{g_1} g_2;$

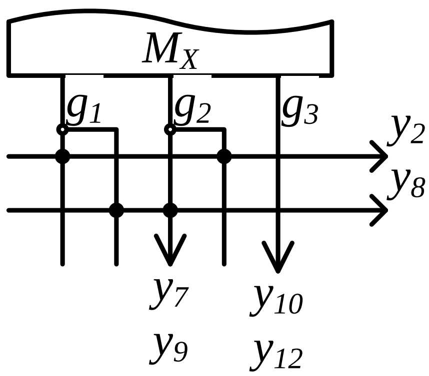
$\{y_7, y_9\} \in Y_4,\ Y_8;\ \{y_7, y_9\} \in B_4 \vee B_8 = g_2;$ (*)

$y_8 \in Y_4;\ y_8 \in B_4 = g_1 g_2;$

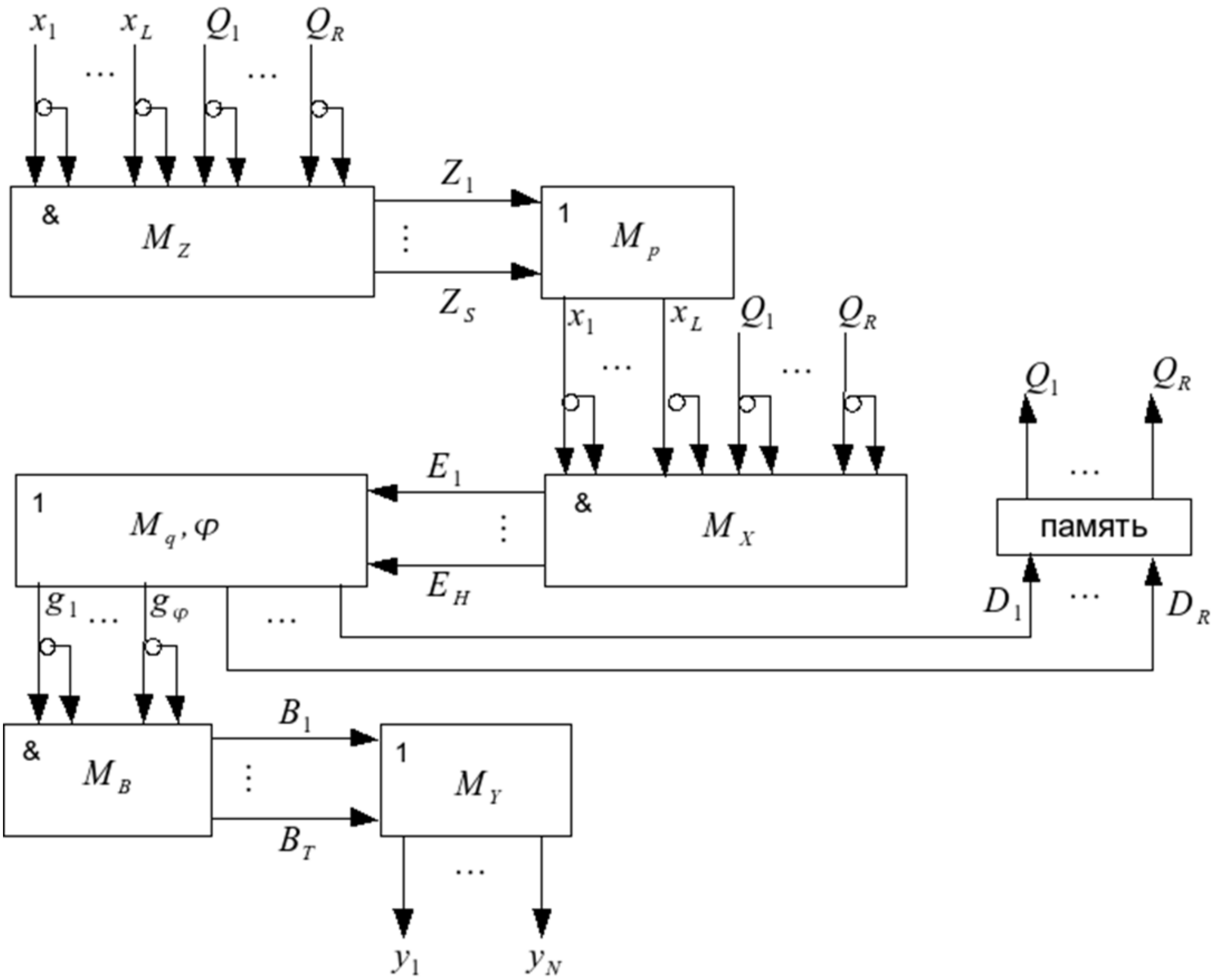
$\{y_{10}, y_{12}\} \in Y_5,\ Y_6;\ \{y_{10}, y_{12}\} \in B_5 \vee B_6 = g_3.$



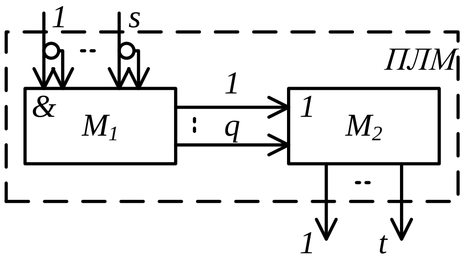




Если выражение типа (*) склеиваются в одну конъюнкцию (см. пример), то матрица M_y исчезнет, а матрица M_B, как правило, упрощается. Склеивание зависит, в первую очередь, от кодирования. Следовательно, микрокоманды надо кодировать так, чтобы максимальное число выражений в системе (*) склеивалось в одну конъюнкцию.

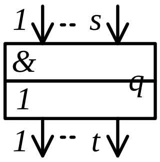


6. Программируемые логические матрицы (ПЛМ)



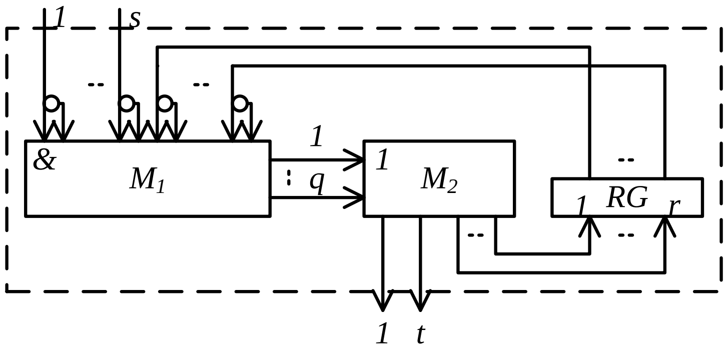
ПЛМ (*s*, *t*, *q*).
Различают ПЛМ 2-х типов:
ПЛМ, программируемые изготовителем;
ПЛМ, программируемые пользователем.

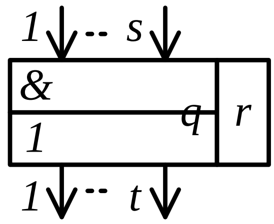
Условное обозначение



Здесь *s*– число входов, *t*– число выходов, а *q*– число термов.
Обычно
s - около 12,
t - около 24,
q - от 48 до 96.

Существуют ПЛМ с памятью   Условное обозначение



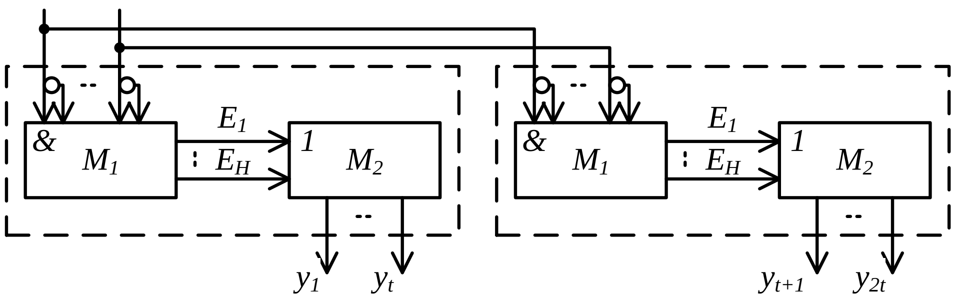


Допустим, задана система булевых функций, содержащая $x_1, \ldots, x_L$. $y_1, \ldots, y_N$ - выходные переменные, термы этой системы: $E_1, \ldots E_H$.

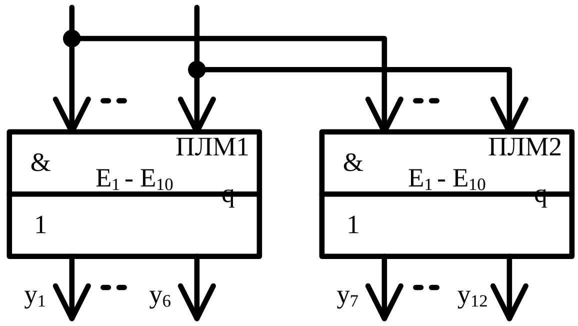
Такая система может быть реализована на одной ПЛМ с параметрами s, t, q, если $L \le s, N \le t, H \le q$.

6.1. Расширение ПЛМ по выходам

Такое расширение требуется, когда $L \le s$, $H \le q$, $N > t$. Выполняется оно так, как показано на рисунке ниже.

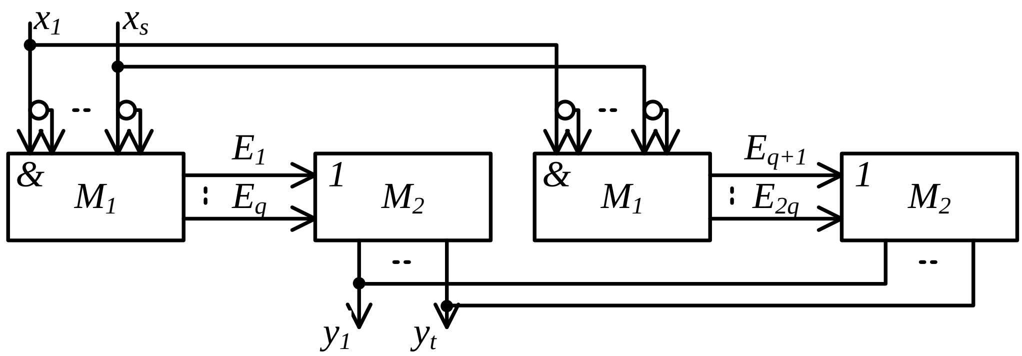


Пример использования ПЛМ (s = 6, t = 6, q = 10) для расширения по выходам.

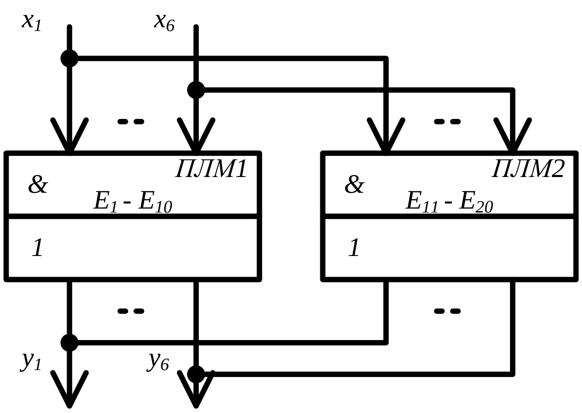


6.2. Расширение ПЛМ по термам

Такое расширение требуется, когда $L \le s,\ N \le t,\ \ H > q$. Выполняется оно так, как показано на рисунке ниже.

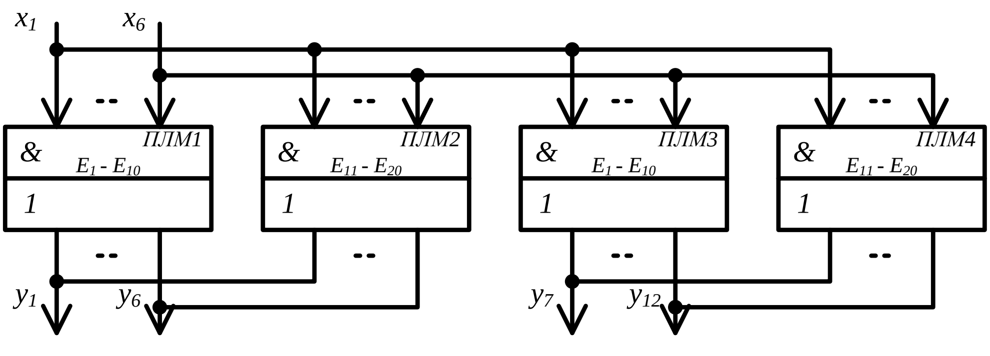


Пример использования ПЛМ (s = 6, t = 6, q = 10) для расширения по термам.



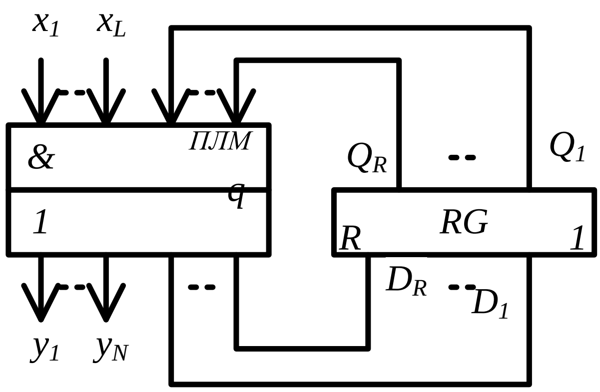
6.3. Совместное расширение ПЛМ по выходам и по термам

Если у ПЛМ не хватает ни выходов, ни термов, т.е. $L \le s,\ N > t,\ H > q$, то сначала необходимо выполнить расширение по термам, а затем расширение по выходам.



6.4. Синтез МПА на ПЛМ

Синтез МПА будем производить на основе прямой структурной таблицы переходов. Укрупненная структура МПА в этом случае будет выглядеть следующим образом.



a_m	$K(a_m)$	a_s	$K(a_s)$	X_h	Y_t		D_h $D_1D_2D_3$	h
a_1	001	a_2	011	$x_1 x_2$	y_3y_9	Y_2	011	1
		a_3	000	$x_1\overline{x_2}$	y_1y_4	Y_3	000	2
		a_5	010	$\overline{x_1}$	y_2	Y_6	010	3
a_2	011	a_2	011	x_1	y_8y_9	Y_7	011	4
		a_5	010	$\overline{x_1}$	$y_3y_5y_6$	Y_1	010	5
a_3	000	a_6	100	x_2	-	Y_0	100	6
		a_3	000	$\overline{x_2}\, x_3$	y_2	Y_6		7
		a_5	010	$\overline{x_2 x_3}$	y_3y_9	Y_2	010	8
a_4	110	a_6	100	1	y_4y_7	Y_5	100	9
a_5	010	a_1	001	$x_3\, x_1$	-	Y_0	001	10
		a_3	000	$x_3\overline{x_1}$	y_5y_6	Y_4		11
		a_4	110	$\overline{x_3}$	y_2	Y_6	110	12
a_6	100	a_6	100	x_2	y_4y_7	Y_5	100	13
		a_1	001	$\overline{x_2}$	$y_3y_5y_6$	Y_1		14

Для нашего примера входы ПЛМ будут загружаться входными сигналами: x_1, x_2, x_3, Q_1, Q_2, Q_3. То, что мы создадим должно вырабатывать входные сигналы: y_1, ... , y_9, D_1, D_2, D_3 - ограничение по входам не выполнятся. Термы: E_1, ... , E_{14} - ограничение по термам не выполнятся.

Разобьем выходные переменные и сигналы функции возбуждения на 2 подмножества:

$$B_1 = \{y_1, y_2, y_3, y_4, y_5, y_6\};\ B_2 = \{y_7, y_8, y_9, D_1, D_2, D_3\}.$$

Будем реализовывать эти две подсистемы отдельно следующим образом:

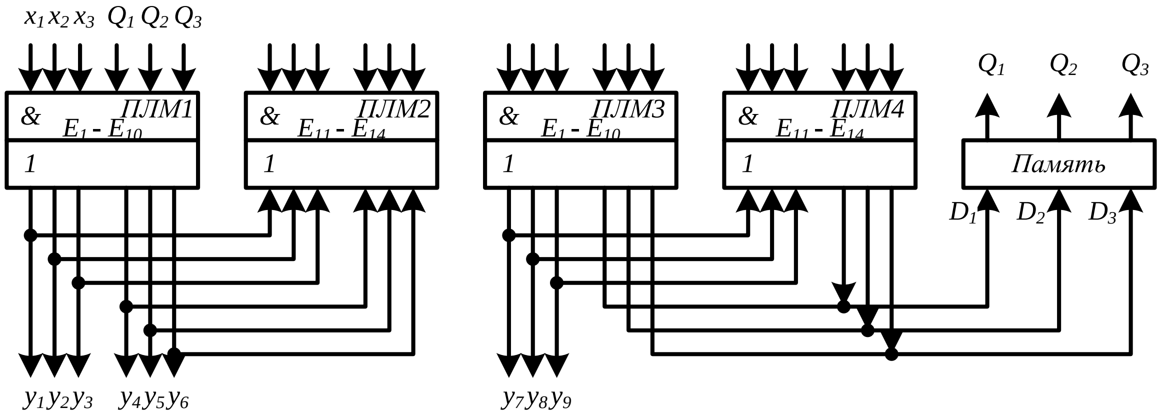


Таблица программирования ПЛМ 1; 2:

	$x_1\ x_2\ x_3\ Q_1\ Q_2\ Q_3$	$y_1\ y_2\ y_3\ y_4\ y_5\ y_6$
1	1 1 d 0 0 1	. . 1 . . .
2	1 0 d 0 0 1	1 . . 1 . .
3	0 d d 0 0 1	. 1 . . . .
4	1 d d 0 1 1	. . . . . .
5	0 d d 0 1 1	. . 1 . 1 1
.		
.	...	
.		
9	d dd 1 1 0	. . . 1 . .
10	1 d 1 0 1 0	. . . . . .
1	0 d 1 0 1 0	. . . . 1 1
2	d d 0 0 1 0	. 1 . . . .
3	d dd 1 0 0	. . . 1 . .
4	d 0 d 1 0 0	. . 1 . 1 1
.		
.	...	
.		
10	d ddddd	. . . . . .

Аналогичным образом строятся ПЛМ 3 и ПЛМ 4.

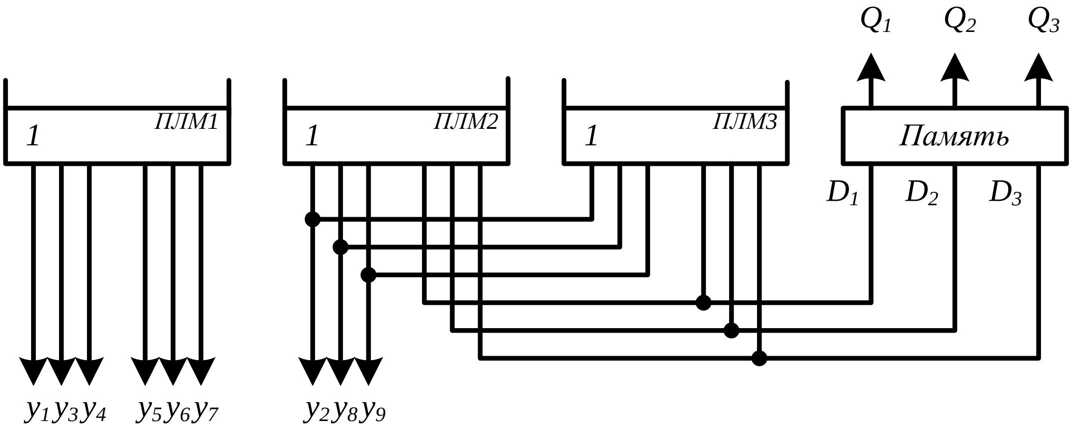
Разобьем выходные сигналы и сигналы функции возбуждения следующим образом:

$$B_1 = \{y_1, y_3, y_4, y_5, y_6, y_7\};\ B_2 = \{y_2, y_8, y_9, D_1, D_2, D_3\}.$$

Такое разбиение привело к тому, что для реализации подмножества B_1 потребовались следующие термы:

$$E_1, E_2, E_5, E_8, E_9, E_{11}, E_{13}, E_{14}.$$

Следовательно, здесь расширение по термам вводить не потребуется и множество B_1 может быть реализовано на одной ПЛМ.



Рекомендованная литература

1. С. И. Баранов. Синтез микропрограммных автоматов. Л., Энергия, 1979, 232 с.
2. Джон Хопкрофт, Раджив Мотвани, Джеффри Ульман. Введение в теорию автоматов, языков, вычислений, 2-е изд.: Пер. с англ. М.: Издательский дом 'Вильямс', 2008. 528 с.
3. Ю. Г. Карпов. Теория автоматов. – СПб.: Питер, 2003. – 208.: ил.
4. Коган Д.И., Бабкина Т.С. Основы теории конечных автоматов и регулярных языков. Учебное пособие. Издательство ННГУ. 2002. 238 с.
5. В. Г. Лазарев, Е. И. Пийль. Синтез управляющих автоматов. М., Энергатомиздат, 1989, 327 с.
6. В. Дж.Рейуод-Смит. Теория формальных языков. М., Радио и связь,1988, 127 с.
7. Самофалов К. Г. И др. Прикладная теория цифровых автоматов. — К.: Вища школа. Головное издательство 1987 — 375 стр.
8. А. Я. Савельев. Прикладная теория цифровых автоматов. М., В. шк., 1987, 272с.
9. А. Ш. Блох. Граф-схемы и их применение. Изд. «Вышэйшая школа», 1975, 302 с.
10. З. В. Алферова. Теория алгоритмов. М., Статистика, 1973, 164 с.
11. З.О. Джалиашвили, Л.В. Крылова, А.А. Ожиганов. Проектирование управляющих автоматов на программируемых логических матрицах. Методические указания к курсовому проектированию по дисциплине "Прикладная теория цифровых автоматов", Л.: ЛИТМО, 1991, 42 с.

В 2009 году Университет стал победителем многоэтапного конкурса, в результате которого определены 12 ведущих университетов России, которым присвоена категория «Национальный исследовательский университет». Министерством образования и науки Российской Федерации была утверждена программа его развития на 2009–2018 годы. В 2011 году Университет получил наименование «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

О кафедре

Кафедра ВТ СПбГУ ИТМО создана в 1937 году и является одной из старейших и авторитетнейших научно-педагогических школ России.

Первоначально кафедра называлась кафедрой математических и счетно-решающих приборов и устройств и занималась разработкой электромеханических вычислительных устройств и приборов управления. Свое нынешнее название кафедра получила в 1963 году.

Кафедра вычислительной техники является одной из крупнейших в университете, на которой работают высококвалифицированные специалисты, в том числе 8 профессоров и 15 доцентов, обучающие около 500 студентов и 30 аспирантов.

Кафедра имеет 4 компьютерных класса, объединяющих более 70 компьютеров в локальную вычислительную сеть кафедры и обеспечивающих доступ студентов ко всем информационным ресурсам кафедры и выход в Интернет. Кроме того, на кафедре имеются учебные и научно-исследовательские лаборатории по вычислительной технике, в которых работают студенты кафедры.

Чему мы учим

Традиционно на кафедре ВТ основной упор в подготовке специалистов делается на фундаментальную базовую подготовку в рамках общепрофессиональных и специальных дисциплин, охватывающих наиболее важные разделы вычислительной техники: функциональная схемотехника и микропроцессорная техника, алгоритмизация и программирование, информационные системы и базы данных, мультимедиатехнологии, вычислительные сети и средства телекоммуникации, защита информации и информационная безопасность. В то же время, кафедра предоставляет студентам старших курсов возможность специализироваться в более узких профессиональных областях в соответствии с их интересами.

Специализации на выбор

Кафедра ВТ ИТМО предлагает в рамках инженерной и магистерской подготовки студентам на выбор по 3 специализации.

1. Специализация в области информационно-управляющих систем направлена на подготовку специалистов, умеющих проектировать и разрабатывать управляющие системы реального времени на основе средств микропроцессорной техники. При этом студентам, обучающимся по этой специализации, предоставляется уникальная возможность участвовать в конкретных разработках реального оборудования, изучая все этапы проектирования и производства, вплоть до получения конечного продукта. Для этого на кафедре организована специальная учебно-производственная лаборатория, оснащенная самым современным оборудованием. Следует отметить, что в последнее время, в связи с подъемом отечественной промышленности, специалисты в области разработки и проектирования информационно-управляющих систем становятся все более востребованными, причем не только в России, но и за рубежом.

2. Кафедра вычислительной техники - одна из первых, начавшая в свое время подготовку специалистов в области открытых информационно-вычислительных систем. Сегодня студентам, специализирующимся в этой области, предоставляется уникальная возможность изучать и осваивать одно из самых мощных средств создания больших информационных систем - систему управления базами данных Oracle. При этом повышенные требования, предъявляемые к вычислительным ресурсам, с помощью которых реализуются базы данных в среде Oracle, удовлетворяются за счет организации на кафедре специализированного компьютерного класса, оснащенного мощными компьютерами фирмы SUN, связанными в локальную сеть кафедры. В то же время, студенты, специализирующиеся в данной области, получают хорошую базовую подготовку в области информационных систем, что позволяет им по завершению обучения успешно разрабатывать базы данных и знаний не только в среде Oracle, но и на основе любых других систем управления базами данных.

3. И, конечно же, кафедра не могла остаться в стороне от бурного натиска вычислительных сетей и средств телекоммуникаций в сфере компьютерных технологий. Наличие высокопрофессиональных кадров в данной области и соответствующей технической базы на кафедре (две локальные вычислительные сети, объединяющие около 80 компьютеров и предоставляющие возможность работы в разных операционных средах - Windows, Unix, Solaris), позволило организовать подготовку специалистов по данному направлению, включая изучение вопросов компьютерной безопасности, администрирования, оптимизации и проектирования вычислительных сетей.

Александр Аркадьевич Ожиганов

Теория автоматов

Учебное пособие

В авторской редакции А.А. Ожиганов
Редакционно-издательский отдел НИУ ИТМО
Зав. РИО Н.Ф. Гусарова
Лицензия ИД № 00408 от 05.11.99
Подписано к печати
Заказ №
Тираж 150 экз.
Отпечатано на ризографе

Редакционно-издательский отдел
Санкт-Петербургского национального
исследовательского университета
информационных технологий, механики
и оптики
197101, Санкт-Петербург, Кронверкский пр., 49